### Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Day</th>
<th>Class No.</th>
<th>Title</th>
<th>Chapters</th>
<th>HW Due date</th>
<th>Lab Due date</th>
<th>Exam</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 Nov</td>
<td>Wed</td>
<td>21</td>
<td>Boolean Algebra</td>
<td>13.2 – 13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13 Nov</td>
<td>Thu</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EXAM 2</td>
</tr>
<tr>
<td>14 Nov</td>
<td>Fri</td>
<td></td>
<td>Recitation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 Nov</td>
<td>Sat</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 Nov</td>
<td>Sun</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17 Nov</td>
<td>Mon</td>
<td>22</td>
<td>Combinational Logic</td>
<td>13.3 – 13.5</td>
<td></td>
<td></td>
<td>LAB 10</td>
</tr>
<tr>
<td>18 Nov</td>
<td>Tue</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19 Nov</td>
<td>Wed</td>
<td>23</td>
<td>Sequential Logic</td>
<td>14.1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Hardened or Softened by Afflictions

**Alma 62:41**

41 But behold, because of the exceedingly great length of the war between the Nephites and the Lamanites, many had become hardened, because of the exceedingly great length of the war; and many were softened because of their afflictions, insomuch that they did humble themselves before God, even in the depth of humility.
Lecture 21 – Binary Numbers & Boolean Algebra
Signed Binary Integers

3 common representations for **signed** integers:

1. Sign magnitude
2. 1’s compliment
3. 2’s compliment

For all 3 the **MSB** encodes the sign:

\[
\begin{align*}
0 &= + \\
1 &= -
\end{align*}
\]

Most common for computers
Sign-Magnitude

Range: \(-n^{-1} - 1 \rightarrow n^{-1} - 1\)

Representations
- \(01111_{\text{binary}} \Rightarrow 15_{\text{decimal}}\)
- \(11111 \Rightarrow -15\)
- \(00000 \Rightarrow 0\)
- \(10000 \Rightarrow -0\)

The MSB encodes the sign:
- \(0 = +\)
- \(1 = -\)

Problem
- Difficult addition/subtraction
  - check signs
  - convert to positive
  - use adder or subtractor as required
- How to add two sign-magnitude numbers?
  - Ex: \(1 + (-4)\)
1’s Complement

Range: $-n+1 \rightarrow n-1$

Representations
- $00110_{\text{binary}} \Rightarrow 6_{\text{decimal}}$
- $11001 \Rightarrow -6$
- $00000 \Rightarrow 0$
- $11111 \Rightarrow -0$

Problem
- Difficult addition/subtraction
  - no need to check signs as before
  - cumbersome logic circuits
    - end-around-carry
- How to add to one’s complement numbers?
  - Ex: $4 + (-3)$

To negate a number, Invert it, bit-by-bit.

MSB still encodes the sign:

\[
\begin{align*}
0 & = + \\
1 & = -
\end{align*}
\]
Two’s Complement

◆ Problems with sign-magnitude and 1’s complement
  ▲ two representations of zero (+0 and –0)
  ▲ arithmetic circuits are complex

◆ *Two’s complement* representation developed to make circuits easy for arithmetic.
  ▲ only one representation for zero
  ▲ just **ADD** the two numbers to get the right answer (regardless of sign)
Two’s Complement

**Range:** \[ -\left(2^{n-1}\right) \rightarrow \left(2^{n-1}\right) - 1 \]

**Representation:**
- If number is **positive** or zero,
  - normal binary representation, zeroes in upper bit(s)
- If number is **negative**,
  - start with positive number
  - flip every bit (i.e., take the one’s complement)
  - then add one

<table>
<thead>
<tr>
<th>Normal</th>
<th>One’s Comp</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>00101</td>
<td>11010</td>
<td>11011</td>
</tr>
<tr>
<td>01001</td>
<td>10110</td>
<td>10111</td>
</tr>
</tbody>
</table>

MSB still encodes the sign:
- 0 = +
- 1 = −
2’s Complement

Positional number representation with a twist

- **MSB** has a *negative* weight

\[
\begin{align*}
0110 & = 2^2 + 2^1 = 6 \\
1110 & = -2^3 + 2^2 + 2^1 = -2
\end{align*}
\]
2’s Complement

◆ Positional number representation with a twist

▲ MSB has a *negative* weight

\[
0110 = 2^2 + 2^1 = 6 \\
1110 = -2^3 + 2^2 + 2^1 = -2
\]
2’s Complement

- Positional number representation with a twist
  - MSB has a *negative* weight

\[
\begin{align*}
0110 &= 2^2 + 2^1 = 6 \\
1110 &= -2^3 + 2^2 + 2^1 = -2
\end{align*}
\]
2’s Complement

- Positional number representation with a twist
  - MSB has a *negative* weight

\[
0110 = 2^2 + 2^1 = 6
\]
\[
1110 = -2^3 + 2^2 + 2^1 = -2
\]

11111111
2’s Complement

- Positional number representation with a twist
  - MSB has a *negative* weight

\[
\begin{align*}
0110 &= 2^2 + 2^1 = 6 \\
1110 &= -2^3 + 2^2 + 2^1 = -2
\end{align*}
\]
2’s Complement

- Positional number representation with a twist
  - **MSB** has a *negative* weight

\[
\begin{align*}
0110 & = 2^2 + 2^1 = 6 \\
1110 & = -2^3 + 2^2 + 2^1 = -2
\end{align*}
\]
Two’s Complement Shortcut

To take the two’s complement of a number:

1. copy bits from right to left until (and including) the first 1
2. flip remaining bits to the left

\[
\begin{array}{c}
011010000 \\
100101111 \\
\hline
\end{array}
\quad \text{(1’s comp)}
\quad 011010000
\begin{array}{c}
\downarrow \\
\text{(flip)}
\quad \downarrow \\
\text{(copy)}
\end{array}
\begin{array}{c}
100110000
\end{array}
\begin{array}{c}
100101111 \\
\downarrow \\
\text{(copy)}
\end{array}
\begin{array}{c}
100110000
\end{array}
\]
Two’s Complement

Example 3: What is $0110101_2$ in decimal?
What is its 2’s complement?
Example 3: What is $0110101_2$ in decimal? What is its 2’s complement?

$$0110101_2 = 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$= 53_{10}$$
Two’s Complement

**Example 3**: What is $0110101_2$ in decimal?
What is it’s 2’s complement?

$$0110101_2 = 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$= 53_{10}$$

$$0110101\quad (53)$$

$$1001010\quad (1’s\ comp)$$

$$+ 1$$

$$1001011\quad (-53)$$
Two’s Complement Negation

◆ To negate a number, invert all the bits and add 1 (or use shortcut)

<table>
<thead>
<tr>
<th>Number</th>
<th>Decimal Value</th>
<th>Negated Binary Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0110</td>
<td>6</td>
<td>1010</td>
</tr>
<tr>
<td>0111</td>
<td>7</td>
<td>1001</td>
</tr>
<tr>
<td>0000</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1111</td>
<td>-1</td>
<td>0001</td>
</tr>
<tr>
<td>0100</td>
<td>4</td>
<td>1100</td>
</tr>
<tr>
<td>1000</td>
<td>-8</td>
<td>1000</td>
</tr>
</tbody>
</table>

(??)
## Signed Binary Numbers

<table>
<thead>
<tr>
<th>Binary</th>
<th>Sign-magnitude</th>
<th>1’s compliment</th>
<th>2’s complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0 0 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0 1 0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0 1 1</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>1 0 0</td>
<td>-0</td>
<td>-3</td>
<td>-4</td>
</tr>
<tr>
<td>1 0 1</td>
<td>-1</td>
<td>-2</td>
<td>-3</td>
</tr>
<tr>
<td>1 1 0</td>
<td>-2</td>
<td>-1</td>
<td>-2</td>
</tr>
<tr>
<td>1 1 1</td>
<td>-3</td>
<td>-0</td>
<td>-1</td>
</tr>
</tbody>
</table>
Decimal to Binary Conversion

Positive numbers

1. start with empty result
2. if decimal number is odd, prepend ‘1’ to result
   else prepend ‘0’
3. divide number by 2, throw away fractional part
   (INTEGER divide)
4. if number is non-zero, go back to 2 else you are done

Negative numbers

do above for positive version of number and negate result.
## Decimal to Binary Conversion

<table>
<thead>
<tr>
<th>Number</th>
<th>Binary Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0101</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
</tr>
<tr>
<td>123</td>
<td>01111011</td>
</tr>
<tr>
<td>35</td>
<td>00100011</td>
</tr>
<tr>
<td>-35</td>
<td>1011101</td>
</tr>
<tr>
<td>1007</td>
<td>011111011111</td>
</tr>
</tbody>
</table>
Hexadecimal Notation

- Binary is hard to read and write by hand
- Hexadecimal is a common alternative
- 16 digits are 0123456789ABCDEF

0100 0111 1000 1111 = 0x478F
1101 1110 1010 1101 = 0xDEAD
1011 1110 1110 1111 = 0xBEEF
1010 0101 1010 0101 = 0xA5A5

1. Separate binary code into groups of 4 bits (starting from the right)
2. Translate each group into a single hex digit

0x is a common prefix for writing numbers which means hexadecimal
Binary to Hex Conversion

◆ Every four bits is a hex digit.
    ✤ start grouping from right-hand side

```
0111 0101 1000 1110 1001 1101 0111
3     A     8     F     4     D     7

This is not a new machine representation, just a convenient way to write the number.
```
Boolean Algebra
Boolean Algebra: the mathematics associated with binary numbers

Developed by George Boole in 1854

Variables in boolean algebra can take only one of two possible values:

0 → FALSE
1 → TRUE
Logic Functions

3 different ways to represent logic functions:

1. **Equation**: a mathematical representation of a logic function

\[ out = \overline{sab} + \overline{sab} + \overline{sab} + sab \]

- **Final logic output**
- **Each letter variable represents a top-level input to the logic function**
- **Mathematical operations (i.e. addition and multiplication) are boolean algebra operations**

A bar over a variable represents an inverting or a NOT operation.
Logic Functions

3 different ways to represent logic functions:

2. **Gates**: a visual block representation of the function

Four 3-input AND gates feeding into one 4-input OR gate
3 different ways to represent logic functions:

3. **Truth Table**: indicates what the output will be for every possible input combination.

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

If there are $n$ inputs (left-hand columns) there will be $2^n$ entries (rows) in the table. **EX**: 3 inputs require $2^3 = 8$ rows.

There will always be at least one output (right-hand columns).

For each input combination (row) outputs will be either 0 or 1.
The Inverter

Truth-table

<table>
<thead>
<tr>
<th>IN</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Equation

\[ \text{OUT} = \overline{\text{IN}} \]

Gate

Inverter Symbols

Usually abbreviated with just a ‘bubble’ next to another gate input/output
The AND Gate

### Gate

![AND Symbol](image)

### Truth Table

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

### Equation

\[ OUT = A \cdot B \]

**NB:** multiplication operation $\rightarrow$ AND
The OR Gate

**Gate**

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

**Truth Table**

**Equation**

\[ \text{OUT} = A + B \]

**NB:** addition operation → OR
The NAND Gate (NOT-AND)

NAND Symbols

Truth Table

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

Equation

OUT = A \cdot B
The NOR Gate (NOT-OR)

Gate

Truth Table

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

NOR Symbols

Equation

\[ \text{OUT} = A + B \]
You should know how to Translate

These are three different ways of representing logical information

You can convert any one of them to any other

Logic Equations

Logic Gates

Truth Tables

Truth Table:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Equations to Gates

\[ y = \overline{\text{NOT}(s)} \quad \text{AND} \quad a \quad \text{AND} \quad \overline{\text{NOT}(b)} \]

OR

\[ y = \overline{s} \cdot \overline{a} \cdot \overline{b} \]
Equations to Gates

\[ y = \text{NOT}(s) \ \text{AND} \ a \ \text{AND} \ \text{NOT}(b) \]

\[ y = \overline{s} \cdot a \cdot \overline{b} \]

\[ \overline{s} \quad \overline{a} \quad \overline{b} \]

[Diagrams of gates representing the equations]
Gates to Equations

\[ \overline{s} \quad a \quad \overline{b} \quad \overline{s} \quad a \quad b \quad s \quad \overline{a} \quad b \quad s \quad a \quad b \]

\[ \text{out} \]
Gates to Equations

\[ \text{out} = (\bar{s} \cdot a \cdot \bar{b}) + (\bar{s} \cdot a \cdot b) + (s \cdot \bar{a} \cdot b) + (s \cdot a \cdot b) \]

OR

\[ \text{out} = \bar{s}ab + \bar{s}ab + s\bar{a}b + sab \]
Truth Tables to Gates

- Each row of truth table is an AND gate
- Each output column is an OR gate

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>OUT</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>1</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>0</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>
Truth Tables to Gates

- Each row of truth table is an **AND** gate
- Each output column is an **OR** gate

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>OUT</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>1</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>0</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>
Truth Table to Equations

Write out truth table a combination of AND’s and OR’s

equivalent to gates

easily converted to gates

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>OUT</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>1</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>0</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>
Truth Table to Equations

- Write out truth table a combination of AND’s and OR’s
- Equivalent to gates
- Easily converted to gates

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>OUT</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>1</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>0</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>

\[
\text{out} = \overline{s}ab + \overline{s}ab + \overline{s}ab + s ab
\]
Equations to Truth Tables

For each AND term

fill in the proper row on the truth table

\[ out = \overline{s}a\overline{b} + \overline{s}ab + s\overline{a}b + sab \]
Equations to Truth Tables

For each **AND** term,

fill in the proper row on the truth table

\[
\text{out} = \overline{s}ab + \overline{s}ab + s\overline{a}b + sab
\]

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>OUT</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>1</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>0</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>