LSN 2
Number Systems

ECT 224 Digital Computer Fundamentals
LSN 2 – Decimal Number System

- Decimal number system has 10 digits (0-9)
- Base 10 weighting system

\[
\ldots 10^5 \ 10^4 \ 10^3 \ 10^2 \ 10^1 \ 10^0 \cdot 10^{-1} \ 10^{-2} \ 10^{-3} \ 10^{-4} \ 10^{-5} \ldots
\]

- Decimal numbers are sums of the weighted digits

\[
25 = (2 \times 10^1) + (5 \times 10^0) \\
= 20 + 5 \\
= 25
\]

833 =
LSN 2 – Binary Number System

- Binary number system has 2 digits (0-1)
- Base 2 weighting system

\[ \ldots 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0 \ 2^{-1} \ 2^{-2} \ 2^{-3} \ 2^{-4} \ 2^{-5} \ldots \]
LSN 2 – Converting Binary to Decimal

• Binary numbers are sums of the weighted binary digits (bits)

\[ 11001 = (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) \]
\[ = 16 + 8 + 0 + 0 + 1 \]
\[ = 25 \]

\[ 10101 = \]

• Largest decimal number represented by \( n \) bits is \( 2^n - 1 \)

\[ 1111 = 15? \]
\[ 11111 = 31? \]
### LSN 2 – Converting Decimal to Binary

- **Sum of weights**
  - Find closest binary weight to desired decimal number and then work backwards

- **Division by 2 method**
  - Repeated division by 2 until whole number quotient is 0
  - Remainder is binary equivalent
  - Convert decimal 25 to binary:

<table>
<thead>
<tr>
<th>Division</th>
<th>Remainder</th>
</tr>
</thead>
<tbody>
<tr>
<td>25/2 = 12</td>
<td>1 (LSB)</td>
</tr>
<tr>
<td>12/2 = 6</td>
<td>0</td>
</tr>
<tr>
<td>6/2 = 3</td>
<td>0</td>
</tr>
<tr>
<td>3/2 = 1</td>
<td>1</td>
</tr>
<tr>
<td>1/2 = 0</td>
<td>1 (MSB)</td>
</tr>
</tbody>
</table>

Quotient = 0

25 = 11001
LBN 2 – Converting Decimal to Binary

• Fractional decimal conversion to binary
  – Sum of weights
    • Find closest binary weight to desired decimal number and then work backwards
  – Multiplication by 2 method
    • Repeated multiplication by 2 until desired resolution or until fractional places equal 0
    • Carry is binary equivalent
    • Convert decimal 0.4375 to binary:

\[
\begin{align*}
0.4375 \times 2 &= 0.875 & \text{Cary} = 0 & \text{MSB} \\
0.875 \times 2 &= 1.75 & 1 \\
0.75 \times 2 &= 1.5 & 1 \\
0.5 \times 2 &= 1.0 & 1 \text{ LSB}
\end{align*}
\]

Fractional places = 0

\[0.4375 = .0111\]
LSN 2 – Binary Arithmetic

• Addition

<table>
<thead>
<tr>
<th></th>
<th>Sum</th>
<th>Carry</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 + 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0 + 1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1 + 0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1 + 1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

– Example:

\[
\begin{array}{c}
1010 \\
+0111 \\
\hline
1110 \\
+1111 \\
\end{array}
\]
LSN 2 – Binary Arithmetic

• Subtraction

<table>
<thead>
<tr>
<th>Difference</th>
<th>Borrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 0</td>
<td>0</td>
</tr>
<tr>
<td>1 - 1</td>
<td>0</td>
</tr>
<tr>
<td>1 - 0</td>
<td>1</td>
</tr>
<tr>
<td>10 - 1</td>
<td>1</td>
</tr>
</tbody>
</table>

– Example:

```
  0101
- 0100
  0010
```
LSN 2 – Binary Arithmetic

• **Examples** (2.4 review from book)

\[
\begin{align*}
1101 & \quad + \quad 10111 \\
+ \quad 1010 & \quad + \quad 01101 \\
\hline
1101 & \quad + \quad 0100 \\
- \quad 0100 & \quad - \quad 0111
\end{align*}
\]
Chapter 2 – Signed Binary Numbers

• Signed binary number format

• Sign-Magnitude representation
  – Only sign bit changes between positive and negative numbers
  – Example:

    \[0 \ 0011010 \rightarrow 1 \ 0011010\]
Chapter 2 – Signed Binary Numbers

• 1’s Compliment representation
  – Change all 1’s to 0’s and all 0’s to 1’s
  – Example:
    
    10110010

  – Negative numbers are the 1’s Compliment of the corresponding positive number
  – Example:
    
    0 0011010 $\Rightarrow$ 1 1100101
Chapter 2 – Signed Binary Numbers

• 2’s Compliment representation
  – Method 1 → perform the 1’s Compliment and add ‘1’
  – Method 2 → starting with the LSB, retain all bit values up to and including the least significant 1
    • Take the 1’s compliment of remaining bits
  – Example:
    \[ \begin{align*} 10110010 & \quad \text{1's Compliment:} \quad 11001111 \\ & \quad \text{2's Compliment:} \quad 11001111 \end{align*} \]
– Negative numbers are the 2’s Compliment of the corresponding positive number
– Example:
  \[ \begin{align*} 0011010 & \quad \text{Positive:} \quad 1100110 \\ & \quad \text{2's Compliment:} \quad 11001110 \end{align*} \]
Chapter 2 – Signed Binary Numbers

• Range
  – For n bits there are $2^n$ combinations
  – With signed binary numbers there is a range from $-(2^{(n-1)})$ to $(2^{(n-1)} - 1)$
  – Example:
    
    \[ n = 8 \quad \text{range is from } -128 \text{ to } 127 \]
LSN 2 – Signed Binary Numbers

• Example
  – Express the number -39 as an 8-bit binary number using
    • Sign-Magnitude
    • 1’s Compliment
    • 2’s Compliment
Signed Binary Numbers

- **Decimal equivalents**
  - **Sign-Magnitude**
    - Sum of weighted binary digits excluding sign bit
    - Adjust sign based upon sign bit
  - **1’s Compliment**
    - Positive numbers:
      - Sum of weighted binary digits
    - Negative numbers:
      - Sum of weighted binary digits with a negative weight being assigned to the sign bit and adding ‘1’
      - Example:
        10010110
LSN 2 – Signed Binary Numbers

– 2’s Compliment
  • All numbers:
    – Sum of weighted binary digits with the negative weight being assigned to the sign bit
    – Example:
      10010110

• Preferred method
  – Single method for both positive and negative numbers
  – Standard binary arithmetic applies
LSN 2 – Floating point numbers

- Contain both integer and fractional parts along with a sign
- Represented by:
  - Mantissa = magnitude
  - Exponent = number of places the “decimal” point is to be moved

![Floating point number representation diagram]

- 8 bits for Exponent
- 23 bits for Mantissa
LSN 2 – Floating point numbers

• Example:
  Represent $4.533 \times 10^3$ as a single precision floating point number
LSN 2 – Hexadecimal Numbers

• Base 16 number system
  – 16 digits: 0 1 2 3 4 5 6 7 8 9 a b c d e f
    • Convenient representation since digital systems process binary data in $4n$ groupings
      – Each hexadecimal digit represents a 4-bit binary number
      – $4n$ binary digits (bits) → $n$ hexadecimal numbers
    • Counting:
      0 → f
      10 → 1f
      20 → 2f
      .
      .
      .
      n0 → nf
LSN 2 – Hexadecimal Numbers

• Binary / Hexadecimal conversion
  – Binary → Hexadecimal
    • Break binary number into 4-bit groupings starting at the right and use the associated hexadecimal value
    • Example:
      
      \[
      \begin{align*}
      0100100111001010 & \quad 1001110010101101
      \end{align*}
      \]
  – Hexadecimal → Binary
    • Replace each hexadecimal number with its associated 4-bit binary number
    • Example:
      
      \[
      \begin{align*}
      2 \text{ f d } 3_{16} & \quad 3 \text{ e a } 1_{16}
      \end{align*}
      \]
LSN 2 – Hexadecimal Numbers

- Decimal / Hexadecimal conversion
  - Hexadecimal → Decimal
    - Convert hexadecimal to binary then to decimal
    - Example:
      \[ a \ e \ 3 \ 4_{16} \]

- Convert hexadecimal directly to decimal by summing the base-16 weighted decimal representation of each digit
  - Example:
    \[ a \ e \ 3 \ 4_{16} \]
LSN 2 – Hexadecimal Numbers

• Decimal / Hexadecimal conversion
  – Decimal → Hexadecimal
    • Repeated division by 16
      – Remainder expressed in base 16 is the hexadecimal number from LSD to MSD
    • Example:
      25634
LSN 2 – Digital Codes and Parity

• Binary Coded Decimal (BCD)
  – Expresses each decimal digit with a binary number
    • Used for keypads and x-segment displays
  – 8421 code
    • Only uses the 4-bits representing the decimal numbers 8, 4, 2, and 1 \((2^3, 2^2, 2^1, 2^0)\)
      – 0 1 2 3 4 5 6 7 8 9
      
      0000 0001 0010 0011 0100 0101 0110 0111 1000 1001
    • There are 6 4-bit binary combinations not used (invalid)
      – 10 11 12 13 14 15
      
      1010 1011 1100 1101 1110 1111
LSN 2 – Digital Codes and Parity

– Decimal → BCD conversion
  • Replace each digit with appropriate BCD code
  • Example:
    8 9 6 1 2

– BCD → Decimal conversion
  • Find decimal conversion for each 4-bit sequence
  • Example:
    10011000000010100
LSN 2 – Digital Codes and Parity

– BCD Addition
  • Perform standard binary addition
    – If sum is greater than 9, add a binary 6 to sum to ensure a valid BCD code is generated
      » Add any carry to next BCD number
  • Example:
    
    \[
    \begin{array}{c}
    \text{100101110100} \\
    + \text{000100100000} \\
    \hline
    \text{100110001100}
    \end{array}
    \]
LSN 2 – Digital Codes and Parity

• Gray Code
  – Not a signed or arithmetic code
  – Only 1 bit position changes between consecutive numbers
  – Useful in applications where the number of bit changes needs to be limited to help reduce errors

(a) Binary

(b) Gray code
LSN 2 – Digital Codes and Parity

• ASCII Code
  – American Standard Code for Information Interchange
  – Represents 128 characters with an 8-bit binary code
    • MSB is always 0 (only uses 7 LSBs)
LSN 2 – Digital Codes and Parity

• Parity
  – Used for error detection
    • Odd: parity bit is used to create an odd number of 1’s
    • Even: parity bit is used to create an even number of 1’s

<table>
<thead>
<tr>
<th>EVEN PARITY</th>
<th>ODD PARITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P  BCD</td>
<td>P  BCD</td>
</tr>
<tr>
<td>0  0000</td>
<td>1  0000</td>
</tr>
<tr>
<td>1  0001</td>
<td>0  0001</td>
</tr>
<tr>
<td>1  0010</td>
<td>0  0010</td>
</tr>
<tr>
<td>0  0011</td>
<td>1  0011</td>
</tr>
<tr>
<td>1  0100</td>
<td>0  0100</td>
</tr>
<tr>
<td>0  0101</td>
<td>1  0101</td>
</tr>
<tr>
<td>0  0110</td>
<td>1  0110</td>
</tr>
<tr>
<td>1  0111</td>
<td>0  0111</td>
</tr>
<tr>
<td>1  1000</td>
<td>0  1000</td>
</tr>
<tr>
<td>0  1001</td>
<td>1  1001</td>
</tr>
</tbody>
</table>
LSN 2 – Homework

• Reading:
  – Chapter 2.1 – 2.6
  – Chapter 2.6 – 2.11

• Problems – HW2
  – Chapter 2, problems 6, 9, 11(a, g), 13(c, f), 15, 16, 21(d,e), 22(g,h), 23, 24, 25, 26, 27, 28
    • Show all work for credit

• Problems – HW3
  – 30, 37(e, f, g), 38(d, e, f), 39(c, d, h), 40, 47(j, k, l), 51(a, d, g, j), 53(c, d, f), 59, 64
    • Show all work for credit

sschneider@udayton.edu