Ee6201 - Digital Logic Circuits

Unit 1

Number System & Digital Logic Families

* Decimal Number System - 0 to 9
* Binary Number System - 0 & 1
* Octal Number System - 0 to 7
* Hexadecimal N.S. - 0 to F
* Binary Coded Decimal

Decimal Number System

* Represented by using base 10

Ex: 6789 = 5*1000 + 6*100 + 7*10 + 8*1 + 9*1^{-1}

i.e. power of 10 goes on dec.

Binary Numbers System:

* Only 0 & 1 & represented with base 2

1010.01 \Rightarrow 1*2^3 + 0*2^2 + 1*2^1 + 0*2^0 + 0*2^{-1}

Octal Numbers System:

* Uses 0 to 7 with base 8

2850.176 \Rightarrow 2*8^3 + 8*8^2 + 5*8^1 + 0*8^0 + 1*8^{-1} + 7*8^{-2}

Hexadecimal Number System:

* Uses 0 to 9, A, B, C, D, E, F [upto 15] with base 16

3AF.95 \Rightarrow 3*16^2 + 10*16^1 + 15*16^0 + 9*16^{-1} + 5*16^{-2}

i.e. After 9 \rightarrow it is represented by alphabets A to F
Format of a Binary Number:

* Single digit in a binary no. is called "Bit"
* 4 binary digits → Nibble - \( 2^4 = 16 \) distinct values
* 8 " " → Byte - \( 2^8 = 256 \) " "
* 16 " " → Word - \( 2^{16} = 65536 \) " "
* 32 " " → Double-word - \( 2^{32} = 4294967296 \)

Procedure:

1. Multiply each digit: [Bit - Byte] by its radix corresponding power
2. Then add all the products.

1) Convert \((78)_8\) to decimal 2) \((AF)_{16}\) to decimal

\[
\begin{align*}
7 & \quad 8 \\
L & \times 8^0 = 2 \\
7 & \quad 8^1 = 56 \\
\Rightarrow & \quad 58 \\
\therefore & \quad (78)_8 = (58)_{10}
\end{align*}
\]

\[
\begin{align*}
A & \quad F \\
L & \times 16^0 = 15 \\
4 & \quad 16^1 = 64 \\
\Rightarrow & \quad 79 \\
\therefore & \quad (AF)_{16} = (79)_{10}
\end{align*}
\]

2) Convert \((101011)_{2}\) to decimal

\[
\begin{align*}
& \times 2^0 = 1 \\
& \times 2^1 = 2 \\
& \times 2^2 = 0 \\
& \times 2^3 = 8 \\
& \times 2^4 = 0 \\
& \times 2^5 = 32 \\
\Rightarrow & \quad 43 \\
\therefore & \quad (101011)_{2} = (43)_{10}
\end{align*}
\]

4) Convert \((7F7)_{16}\) to binary

\[
\begin{align*}
7 & \quad F \\
L & \times 16^0 = 7 \\
15 & \quad 16^1 = 240 \\
\Rightarrow & \quad 247 \\
\therefore & \quad (7F7)_{16} = (11111110111)_{2}
\end{align*}
\]

1) Convert to decimal

\[
\begin{align*}
& \times 2^0 = 1 \\
& \times 2^1 = 2 \\
& \times 2^2 = 0 \\
& \times 2^3 = 8 \\
& \times 2^4 = 0 \\
& \times 2^5 = 32 \\
\Rightarrow & \quad 43 \\
\therefore & \quad (101011)_{2} = (43)_{10}
\end{align*}
\]
From Table Group the No. from last for before decimal & from 1st next to point for Oechal

\[(4B \cdot 8891E)_{16}\]

\[
\begin{align*}
(0100\ 1011 \cdot 0010\ 1011\ 1000\ 0101)_{2} \\
(113.121024)_{8}
\end{align*}
\]

\[\text{6) Convert } (77.78)_{8} \text{ to other Number System.}\]

\[\text{Ans.: } (111111.111010)_{2}, \ (3F.E2)_{16}\]

\[\text{Addition:}\]

1) Add 53 + 47

\[\begin{align*}
\text{binary conversion:} & \\
53 & = 110101 \\
47 & = 101111 \\
\underline{+} & \\
1100100
\end{align*}\]

2) \(TF + FF\)

\[\begin{align*}
TF & = \ 01111111 \\
FF & = \ 11111111 \\
\underline{+} & \\
101111110 & \leftarrow 1 \uparrow \ E
\end{align*}\]

\[\text{\therefore TF + FF = 17E } \quad \text{[: Sum Format = Ans. format]}\]

3) \((1011)_{2} + (TF)_{16} + 52 + (37)_{8} \text{ using Binary Addition.}\]

\[\begin{align*}
TF_{16} & \Rightarrow 01111111 \\
35_{8} & \Rightarrow 45 \Rightarrow 0100\ 0101 \\
52 & \Rightarrow 0101\ 0010
\end{align*}\]

www.studentsfocus.com
UNIT 1

A) Add: \((57)_{16} + (3E)_{16} + (FF)_{16}\) using Binary & Hexa.

Hexa \(\rightarrow\) \(57\) \(\rightarrow\) \(34\)

\(3E\)\(\rightarrow\) \(15\) \(\rightarrow\) \(15\)

\(FF\) \(\rightarrow\) \(15\) \(\rightarrow\) \(15\)

\(36 \rightarrow \text{16}\) \(36\) \(\text{16}\) \(\text{15}\) \(\text{19}\) \(\text{4}\)

\((57)_{16} + (3E)_{16} + (FF)_{16} = (194)_{16}\)

Binary = (110010100)_2

Subtraction

Algorithm:

1) Take 2's Complement For Binary
2) Add the complement with 1st No.
3) Check for carry
   i) if present result is +ive & discard carry
   ii) Else - result is -ive & Find Complement value of that No.

Eq.: Subtract 88 from 72

\(\text{1st No} - 72\)

\(\text{88} - 10011001\)

\(\text{0110011} - \text{0110011}\)

www.studentsfocus.com
SRI VIDYA COLLEGE OF ENGINEERING & TECHNOLOGY COURSE MATERIAL UNIT 1

Binary Multiplication:

\[
\begin{align*}
10 \times 0 & = 0 \\
10 \times 1 & = 0 \\
3 \times 0 & = 0 \\
1 \times 1 & = 1
\end{align*}
\]

Division:

\[
\begin{align*}
100 \div 10 & = 10 \\
100 \div 1 & = 100
\end{align*}
\]

Binary Codes:

- **Weighted Code**
  - Each digit of the position represents a specific weight.
  
  \[581 \rightarrow 5 \times 10^2 + 8 \times 10^1 + 1 \times 10^0\]

- **Non-weighted Code**
  - Not assigned with any weight to each digit position.

  Eq - Excess-3 & Gray codes

Binary Coded Decimal:

- Each digit of a decimal number is represented by a separate group of 4-bits.

  Most common BCD code is 8-4-2-1 BCD

www.studentsfocus.com
BCD Addition:

1) $6 + 3 = 9$

\[
\begin{array}{c}
6 \\
+ 3 \\
\hline
9
\end{array}
\]

When the ans. > 9 add 6 - 0110 in the invalid BCD no. i.e. 1101.

\[
\begin{array}{c}
1101 \\
+ 0110 \\
\hline
1001 > 9 - invalid
\end{array}
\]

Gray Code: carry = 1

\[
\begin{array}{c}
0011 \\
+ 0110 \\
\hline
1001
\end{array}
\]

Ans. = 1

2) $6 + 7 = 13$

\[
\begin{array}{c}
6 \\
+ 7 \\
\hline
13
\end{array}
\]

\[
\begin{array}{c}
11001 \\
+ 0011 \\
\hline
11101
\end{array}
\]

Decoded: 0011

\[
\begin{array}{c}
0010 \\
+ 1001 \\
\hline
1000 \rightarrow invalid
\end{array}
\]

Gray Code: carry = 0

\[
\begin{array}{c}
0010 \\
+ 1001 \\
\hline
1011
\end{array}
\]

Ans. = 1

3) sum equals 9 (or) Less with carry 1

\[
\begin{array}{c}
8 + 9 = 17 \\
1000 \\
\downarrow 1001 \\
\hline
1001 \rightarrow invalid
\end{array}
\]

Gray Code: carry = 1

\[
\begin{array}{c}
0010 \\
+ 0001 \\
\hline
0011
\end{array}
\]

\[
\begin{array}{c}
0001 \\
+ 0011 \\
\hline
1000
\end{array}
\]

Exercise: 84 + 18, 46 + 58, 15 + 18

BCD Subtraction:

1) Find 9's complement of the number

2) Add two numbers using BCD addition,

If carry is not generated result is 9's complement of the number

*To find 9's complement of the result, add carry to the result.*
1) \((46)_{10} - (32)_{10}\)

1) a's complement of 32 = \(99 - 32 = 67\)

2) Add 46 & a's complement.

\[
\begin{align*}
46 + 67 & = 0100 + 0110 \\
\underline{0111} & = 1010 \\
\end{align*}
\]

3) Add the carry.

\[
1010 + 1010 + 1100 = 0000 + 1100
\]

2) \((46)_{10} - (56)_{10}\)

1) a's comp. of 56 \(= 99 - 56 = 43\)

ii) Add 46 + 43 \(= 0100 + 0110 = 1010\)

iii) Each digit of 1010 is not assigned to each coded number.

iv) 89 + 9's complement = \(99 - 89 = 10\)

v) Result = \((-10)_{10}\)

**Excess 3 code**

- derived from the natural BCD code

by adding '3' to each coded number.

Example: decimal 12 \(\Rightarrow 0001 + 0010 = 0011\)

www.studentsfocus.com
Gray Code:

- Non-weighted Code
- 0 to 15 decimal
- Special Case of Unit-distance Code

Decimal  Gray Code

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Gray Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>0011</td>
</tr>
<tr>
<td>3</td>
<td>0010</td>
</tr>
<tr>
<td>4</td>
<td>0110</td>
</tr>
<tr>
<td>5</td>
<td>0111</td>
</tr>
<tr>
<td>6</td>
<td>0101</td>
</tr>
<tr>
<td>7</td>
<td>0100</td>
</tr>
<tr>
<td>8</td>
<td>1100</td>
</tr>
<tr>
<td>9</td>
<td>1101</td>
</tr>
<tr>
<td>10</td>
<td>1111</td>
</tr>
<tr>
<td>11</td>
<td>1110</td>
</tr>
<tr>
<td>12</td>
<td>1010</td>
</tr>
<tr>
<td>13</td>
<td>1011</td>
</tr>
<tr>
<td>14</td>
<td>1001</td>
</tr>
</tbody>
</table>
Classification of Logic Families

- Bipolar
  - Saturated
    - RTL
    - DTL
    - DCTL
    - 1$^2$L
    - HTL
    - TTL
  - Unsaturated
    - Schottky
    - TTL Emitter
    - ECL Coupled Logic

Characteristics of DCL:

1. Propagation delay:—
   - The time interval between the application of an input pulse and the occurrence of the result output pulse.

   - Shorter the propagation delay, Higher the speed of the circuit.

2. \( t_{PLH} \) - measured when output is changing from Logic 0 to 1 [Low to High]

3. \( t_{PHL} \) - measured when output is changing from Logic 1 to 0 [High to Low]

Fan-in — refers to the no. of inputs
Fan-out — The maximum no. of inputs of several gates that can be driven by the output of logic gate

Current Sinking - when current flows from the power supply through load & through a device such as LED.
Current sourcing: when current flows from power supply, out of the device OLP & through the load to Gnd.

**RTL - Circuit**

*When both the ILP’s are low, the transistors Q1, Q2 are off.*  
*The OLP is HIGH*  
*VCE of transistor is 0.2V.*  
*As the no. of gates connected to OLP n, OLP voltage V*  

**Diode - Transistor Logic:**

*when both the ILP’s are low, diode DA, DB conducts & results 0.7V at point P. ∴ G1, G2 OFF*  
*Hence low level on any ILP cause 0 conduct resulting***
This causes the base current of transistor \( Q_1 \) to flow through \( R_D, D_1, D_2 \) & the base of the transistor \( Q_1 \) & \( V_{CE(Cut)} = 0.2V \) = Logic 0.

* For \( Q_1 \), saturation we require more than \( 2.1V \) \[ V_{D1 (0.1)} + V_{D2 (0.1)} + V_{BE (0.4)} \]

at Point P. This improves the Noise Margin.

Transistor Transistor Logic

* When the input voltage is LOW, the output voltage is HIGH
  * When \( Q_1 \) is closed, the output is \( V_{out} = 0V \)
  * When \( Q_1 \) is open, the output is \( V_{out} = Vcc \)

2 - 1P TTL NAND Gate

www.studentsfocus.com
Q₁ - multiple emitter transistor, one for each I/P to the gate

- The I/P voltages A, B are either LOW, HIGH, 15 V, or 0 V.
- If either A or B or Both are LOW, the corresponding diode conducts & the base of Q₁ is pulled down to 0.7 V, which reduces base voltage of Q₂ to “0” i.e., Q₂ is OFF.
- When Q₂ is open, Q₄ is OFF & Q₃ base is pulled HIGH, & I/P is pulled up to HIGH.
- When A & B are HIGH, the emitter diode of Q₁ are reverse biased & OFF. D₄ forward conduction which forces Q₂ to go HIGH. In turn Q₄ goes into saturation, producing a “Low” I/P.

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

Truth Table
<table>
<thead>
<tr>
<th>Parameter</th>
<th>RTL</th>
<th>DTL</th>
<th>TTL</th>
<th>ECL</th>
<th>CMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Components</td>
<td>R &amp; T</td>
<td>R, D, T</td>
<td>R, D, T</td>
<td>R, T</td>
<td>N, P, MOSFET</td>
</tr>
<tr>
<td>Circuits</td>
<td>Simple</td>
<td>Moderate</td>
<td>Complex</td>
<td>Complex</td>
<td>Moderate</td>
</tr>
<tr>
<td>Noise Margin</td>
<td>Nominal</td>
<td>Good</td>
<td>V. Good</td>
<td>Good</td>
<td>V. Good</td>
</tr>
<tr>
<td>Fan-Out</td>
<td>Low (4)</td>
<td>Medium (8)</td>
<td>None (10)</td>
<td>High (25)</td>
<td>50</td>
</tr>
<tr>
<td>Power Dissipation</td>
<td>12</td>
<td>8-12</td>
<td>10</td>
<td>40-55</td>
<td>0.1</td>
</tr>
<tr>
<td>in mW/gate</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Error Detection & Correction Codes:**

* When digital data/information is transmitted from one Ckt. (or) system to another error occurs due to the presence of noise.
* To maintain the data integrity by transmitter & receiver, an extra bit is added in the data.
* The data along with an extra bit forms the code.
* Codes which allow only error detection are called as error detection code.
* Codes allow both detection and correction are called as error detecting & correcting code.

**Parity Bit**

- Used for the purpose of detecting errors during transmission.
- It is the extra bit added, to make the number of 1's odd or even.
The circuit that generates the parity bit in the transmitter is called a parity generator, and the circuit that checks the parity in the receiver is called a parity checker.

In even parity, the added parity bit will make the total no. of 1's as even & odd makes total no. of 1's as odd.

Hamming code:

- not only provides the detection of a bit error, but also identification which bit is in error.
- It is a detecting & correcting code.

No. of Parity bits:

\[ 2^p \geq x + p + 1 \]

i.e. \( n = 4 \), \( P \) is trail of error.

\[ 2^p \geq 4 + p + 1 \]
\[ 2^p \geq 5 + p \]
when \( P = 2 \) \( 2^2 = 4, 4 \neq 5+2 \)
\[ P = 3 \] \( 2^3 = 8, 8 \geq 5+3 \)

Equ.: \( P = 3 \)

Assigning values to Parity bit

\[ D_7 D_6 D_5 D_4 D_3 D_2 D_1 \]
\[ \rightarrow D_7 D_6 D_5 D_4 D_3 P_2 P_1 \]

Assigning \( P_1 = 1 \) since \( P_1 = 00 \) the bits 1011, 1101, 1111 are checked for odd/even parity.
Encode the binary word 1011 into 7 bit even parity Hamming Code.

2^2 = 4 + p + 1

when p = 3,

5  8  8

Total bits = 4 + 3 = 7

D7 D6 D5 P4 P3 P2 P1

Info bit 1

1 0 1 0 0

P1 = 1 0 1 1 0 1 0 0 1 0 1 0 0

Have 2 ones

P3 must be added to make even parity

P3 = 1 0 1 1 0 1 0 0 1 0 1 0 0

P2 = 0

P4 = 1 0 1 1 0 1 0

2 ones - 10 last bit - Even Parity

To maintain the data bit forms

P3 = 0

Parity bits

1 0 1 1 0 1 0 0

check completed