Classification of Digital Circuits

• Combinational logic circuits.
  – Output depends only on present input.

• Sequential circuits.
  – Output depends on present input and present state of the circuit.
Combinational Logic Design Procedure

• Start with the problem statement.
• Determine the number of inputs variables and the required number of output variables.
• Derive a truth table that defines the required relationship between input and output.
• Simplify each output function (Karnaugh maps).
• Draw the logic diagram.
Half Adder Design Example

• A half adder computes the sum of two one bit Boolean inputs, which can be at most 10₂. This requires two outputs.

• Inputs: X, and Y.

Outputs: S and C.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>C</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Half Adder Design
Example

• $C = XY$
• $S = X \oplus Y$
Decoders

- A decoder is a multiple-input multiple-output logic circuit that converts coded inputs into coded outputs, where the inputs and outputs codes are different.
Function Realization With Decoders

- The outputs of a binary decoder provides $2^n$ minterms for its $n$ inputs, and a Boolean functions can be represented as a sum of minterms, therefore a decoder and one or more OR gates can be used to realize sums of minterms of $n$ variables.
Multiplexers

• Combinational logic circuit that selects binary information from one of many input lines and directs it to a single output line. The input is selected by the binary value on the select lines.
Demultiplexers

- Combinational logic circuit that receives binary information on a single input and sends this information to one of many possible output lines. The output is selected by the binary value on the select lines.
Function Realization With Multiplexers

- A multiplexer is basically a binary decoder whose outputs were ORed together and some extra input lines were added to each product term.
Function Realization With Multiplexers

• A sum of minterms can be realized by setting the corresponding input line of the chosen minterms to 1 and setting the input lines of the remaining minterms to 0.
Transition Time

- Time interval between two reference points on a waveform. These reference points are usually 10% and 90% of the voltage change.
  - *Rise time* ($t_r$) – Time interval when waveform is changing from a logic low to a logic high level.
  - *Fall time* ($t_f$) – Time interval when waveform is changing from a logic high to a logic low level.
Propagation Delay

- Time it takes for a change at the input of a device to produce a change at the output of the same.
  - \( t_{pLH} \) is the propagation delay when the output changes from LOW to HIGH.
  - \( t_{pHL} \) is the propagation delay when the output changes from HIGH to LOW.
  - \( t_{pLH} \) and \( t_{pHL} \) are not necessarily equal, and their values depends on the logic family.
Propagation Delay and Transition Time
Fanout

• The number of gate inputs that a single output can drive or operate without exceeding its worst case loading specifications.

  – $I_{\text{ILMax}}$ is the maximum current supplied by an input when a LOW logic level voltage is applied to that input.
  
  – $I_{\text{IHMax}}$ is the maximum current required by an input when a HIGH logic level voltage is applied to that input.
  
  – $I_{\text{OLMax}}$ is the maximum current into an output when this output is in the LOW state.
  
  – $I_{\text{OHMax}}$ is the maximum current provided by an output when this output is in the HIGH state.
Fanout

- GATE ACTING AS A CURRENT SINK
- GATE ACTING AS A CURRENT SOURCE
Fanout

\[
\text{LS fanout} = \frac{|I_{OL}|}{|I_{IL}|}
\]

\[
\text{HS fanout} = \frac{|I_{OH}|}{|I_{IH}|}
\]

\[
\text{fanout} = \min \{ \text{LS fanout}, \text{HS fanout} \}
\]
Power Dissipation

• The power consumed by the gate that must be available from the power supply. This does not include the power delivered from another gate.
  
  – $V_{CC}$: supply voltage.
  
  – $I_{CCH}$: current drawn by the circuit when the output of the gate is HIGH.
  
  – $I_{CCL}$: current drawn by the circuit when the output of the gate is LOW.
  
  – $I_{CC}$: average current drawn by the circuit.
  
  – $P_D$: average power dissipation.
Power Dissipation

\[ I_{CC} = \frac{I_{CCH} + I_{CCL}}{2} \]

\[ P_D = V_{CC} \times I_{CC} \]
DC Noise Margins

- The maximum amount of voltage variation (noise) that may be permitted from LOW or HIGH voltage levels.
  - $V_{OH_{Min}}$: the minimum output voltage in the HIGH state.
  - $V_{IH_{Min}}$: the minimum input voltage guaranteed to be recognized as a HIGH.
  - $V_{IL_{Max}}$: the maximum input voltage guaranteed to be recognized as a LOW.
  - $V_{OL_{Max}}$: the maximum output voltage in the LOW state.
DC Noise Margins

- Low-State = $V_{ILMax} - V_{OLMax}$
- High-State = $V_{OHMin} - V_{IHMin}$
Unused Inputs

• **Handle them as follows:**
  
  – Tie them to a used input in the same gate.
  
  – Tie them to logic 1 through a pull-up resistor for AND & NAND gates.
  
  – Tie them to logic 0 through a pull-down resistor for OR & NOR gates.

![Diagrams](image)
Logic Families

• Transistor Transistor Logic (TTL) is one of the most popular and widespread of all logic families.
  – Very high number of SSI and MSI devices available in the market.
  – Several number of sub-families that provide a wide range of speed and power consumption.

• Sub families:
  – 74xx: The original TTL family.
    • These devices had a propagation delay of 10ns and a power consumption of 10mW, and they were introduced in the early 60’s.
Logic Families

• Sub families:
  
  – 74Hxx : High speed.
    
    • Speed was improved by reducing the internal resistors. Note that this improvement caused an increase in the power consumption.

  – 74Lxx : Low power.
    
    • Power consumption was improved by increasing the internal resistances, and the speed decreased.
Logic Families

• Sub families:
  – 74Sxx : Schottky.
    • The use of Schottky transistors improved the speed. The power dissipation is less than the 74Hxx sub-family.
  – 74LSxx : Low power Schottky.
    • Uses Schottky transistors to improve speed. High internal resistances improves power consumption.
Logic Families

- **Sub families:**
  - **74ASxx : Advanced Schottky.**
    - Twice as fast as 74Sxx with approximately the same power dissipation.
  - **74ALSxx : Advanced Low power Schottky.**
    - Lower power consumption and higher speed than 74LSxx.
  - **74Fxx : Fast.**
    - Performance is between 74ASxx and 74ALSxx.
Logic Families

- Note that parameters like $V_{OHMin}$, $V_{IHMin}$, $V_{ILMax}$, and $V_{OLMax}$ are all the same for the different sub-families, but parameters like $I_{ILMax}$, $I_{IHMax}$, $I_{OLMax}$, and $I_{OHMax}$ may differ.

- Most TTL sub-families have a corresponding 54-series (military) version, and these series operate in a wider temperature and voltage ranges.
Logic Families

• Complementary metal oxide semiconductor (CMOS) replaced TTL devices in the 90’s due to advances in the design of MOS circuits made in mid 80’s.

• Advantages:
  – Operate with a wider range of voltages that any other logic family.
  – Has high noise immunity.
  – Dissipates very low power at low frequencies.
  – It requires an extremely low driving current.
  – High fanout.
Logic Families

• Disadvantages:
  – Power consumption increases with frequency.
  – Susceptible to ESD - electrostatic discharges.

• Sub-families:
  – 40xx : Original CMOS family.
    • Fairly slow, but it has a low power dissipation.
  – 74HCxx : High speed CMOS.
    • Better current sinking and sourcing than 40xx. It uses voltage supply between 2 and 6 volts.
    • Higher voltage → higher speed.
    • Lower voltage → lower power consumption.
Logic Families

• Sub-families:
  – 74HCTxx : High speed CMOS, TTL compatible.
    • Better current sinking and sourcing than 40xx. It uses voltage supply of 5V. Compatible with TTL family.
  – 74ACxx : Advanced CMOS.
    • Very fast. It can source and sink high currents. Not TTL compatible.
  – 74ACTxx : Advanced CMOS, TTL compatible.
    • Same as 74ACxx, but it is compatible with TTL family.
Logic Families

• Sub-families:
  – 74FCTxx: Fast CMOS, TTL compatible.
    • It is faster and has lower power dissipation than the 74ACxx and 74ACTxx sub-families. Compatible with TTL family.

• Prefixes, usually added to device designation to identify the manufacturer.
  – MN: Motorola.
  – DM: National
  – N: Signetics
  – P: Intel
  – H: Harris
  – AMD: Advanced Micro Devices
Logic Families

- Prefixes, usually added to device designation to identify the manufacturer.
  - SN : Texas Instrument.
  - MN : Motorola.
  - DM : National
  - N : Signetics
  - P : Intel
  - H : Harris
  - AMD : Advanced Micro Devices

- Suffixes, identifies the packaging.
  - N : Plastic DIP (dual in-line package)
  - P : Plastic DIP
  - J : Ceramic DIP
  - W : Ceramic flat package.
  - D : Plastic ‘small outline’ package
Latches and Flip-Flops

• These sequential devices differ in the way their outputs are changed:
  – The output of a latch changes independent of a clocking signal.
  – The output of a flip–flop changes at specific times determined by a clocking signal.
S-R Latch

- SR latch based on NOR gates.
- The S input sets the Q output to 1 while R reset it to 0.
- When R=S=0 then the output keeps the previous value.
- When R=S=1 then Q=Q’=0, and the latch may go to an unpredictable next state.

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q₀</th>
<th>Q⁻₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Q₀</td>
<td>Q⁻₀</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</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>
</tbody>
</table>
S-R Latch

- S'R' latch based on NAND gates.
- The S' input sets the Q output to 1 while R' reset it to 0.
- When R'=S'=1 then the output keeps the previous value.
- When R'=S'=1 then Q=Q'=1, and the latch may go to an unpredictable next state.

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q</th>
<th>( \overline{Q} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</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>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Q_{-1}</td>
<td>Q_{-1}</td>
</tr>
</tbody>
</table>

Figure 2 /S-/R latch.
D Latch

• This latch eliminates the problem that occurs in the S’R’ latch when R=S=0.

• C is an enable input:
  – When C=1 then the output follows the input D and the latch is said to be *open*.
  – When C=0 then the output retains its last value and the latch is said to be *closed*.

![D latch with enable](image-url)

**Figure 3** D latch with enable.
D Latch

- For proper operation the D input must not change during a time interval around the falling edge of C.
- This time interval is defined by the setup time – $t_{\text{setup}}$ and the hold time – $t_{\text{hold}}$.

![Diagram showing stable and unstable states with setup and hold times](image)

**Figure 4** Setup and hold times for a D latch.
Edge Triggered D Flip-Flop

• This flip-flop is made out of two D latches. The first latch is the master, and the second the slave.

• When $C_k = 0$ the master is open and the slave is closed. $Q_m$ and $D_s$ follow $D_m$.

• When $C_k = 1$ the master is closed, the slave is open and $Q_m$ is transferred to $Q_s$. Note that $Q_s$ does not change because the master latch is closed leaving $Q_m$ fixed.
Edge Triggered D Flip-Flop

- The same constraints regarding *setup* and *hold time* discussed previously, also apply to the edge triggered D flip-flop.
Edge Triggered J-K Flip-Flop

- The operation of inputs J and K in the J-K flip-flop is similar to the operation of inputs S and R in the S-R flip-flop. The difference arises when J and K are asserted simultaneously. In this situation the output of the J-K flip-flop inverts its current state.

Figure 6 Edge-Triggered J-K Flip-Flop.
T Flip-Flop

• Also known as the *toggle* flip-flop.
• When input $T = 0$ the output $Q$ retain its previous value.
• When input $T = 1$ the output $Q$ inverts on every tick of the clock.
• When inputs $J$ and $K$ of a $J$-$K$ flip-flop are connected together, the $J$-$K$ flip-flop will behave like a $T$ flip-flop.
Sequential Logic Design Procedure

• Derive a state/output table from the problem specification.
• Minimize the number of states in the state/output table by eliminating equivalent states.
• Choose a set of state variables. Assign to each state a unique combination from the set derived above.
• Create a transition/output table.
Sequential Logic Design Procedure

• Choose a flip-flop type and construct its excitation table.
• Using the excitation table fill the values for the input excitation function columns on the transition/output table.
• Derive the excitation and output equations.
• Draw logic diagram.
Sequence Detector Design Example

• Design a sequential circuit with one input (I) and one output (Z) The output is asserted when the input sequence 0-1-1 is received.

• See state/output table below.

<table>
<thead>
<tr>
<th>Present State</th>
<th>Input</th>
<th>Next state</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I</td>
<td></td>
<td>Z</td>
</tr>
<tr>
<td>Init</td>
<td>0</td>
<td>S₀</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Init</td>
<td>0</td>
</tr>
<tr>
<td>S₀</td>
<td>0</td>
<td>S₀</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>S₀₁</td>
<td>0</td>
</tr>
<tr>
<td>S₀₁</td>
<td>0</td>
<td>S₀</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>S₀₁₁</td>
<td>0</td>
</tr>
<tr>
<td>S₀₁₁</td>
<td>0</td>
<td>S₀₁₁</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>S₀₁₁</td>
<td>1</td>
</tr>
</tbody>
</table>
Sequence Detector
Design Example

- Set of state variables and their unique assignment to the different states.

<table>
<thead>
<tr>
<th>State</th>
<th>Q1</th>
<th>Q0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Init</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S₀</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>S₀₁</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>S₀₁₁</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Sequence Detector
Design Example

- Transition/output table

<table>
<thead>
<tr>
<th>Present State</th>
<th>Input</th>
<th>Next state</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>$Q_0$</td>
<td>$I$</td>
<td>$Q_1^*$</td>
</tr>
<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>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>
**Sequence Detector Design Example**

- See excitation table below.

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Required inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>D</strong></td>
<td><strong>J</strong></td>
<td><strong>K</strong></td>
</tr>
<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>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Sequence Detector Design Example

<table>
<thead>
<tr>
<th>Present State</th>
<th>Input</th>
<th>Next state</th>
<th>Output</th>
<th>Input Excitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Q_1 )</td>
<td>( Q_0 )</td>
<td>( I )</td>
<td>( Q_1^* )</td>
<td>( Q_0^* )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- Equations derived from the table above:
  - \( J_1 = IQ_0 \)
  - \( K_1 = I'Q_0 \)
  - \( J_0 = I'Q_1' \)
  - \( K_0 = IQ_1 \)
  - \( Z = Q_1Q_0' \)
Sequence Detector Design Example

• See the logic diagram for the circuit below.
Open Collector Devices

- Entire upper half of output circuit is omitted.
- External pull-up resistor is needed.
- If any input is low, then Q₂ and Q₃ are cutoff and the voltage Z is high.
Open Collector Devices

- Can be used to drive a load, such as LEDs, relays or other device.
- It is important to calculate a suitable resistor $R$.
- The current through the load must not exceed $I_{OLMax}$.

\[
R = \frac{V_{CC} - V_{LED} - V_{OL \, TYP}}{I_{LED}}
\]
Open Collector Devices

- Wired AND Logic – When two or more open-collector outputs are tied together with an external pull-up resistor, the circuit behaves as if the gates were connected to an AND gate.
Open Collector Devices

• Common Bus – Several open collector outputs may be connected together to create a common bus.

• The decoder, in the circuit shown below, selects which device outputs to the common bus by sending a high to the open collector output NAND gate connected to the chosen device.
Open Collector Devices

• To calculate the pull-up resistor of the figure in the next two pages, one must consider two cases:
  – Only one gate is active and forces the bus voltage to be LOW, $V_{OLTyp}$.
  – All gates are disabled forcing the bus voltage to be HIGH, $V_{OHHMin}$. 
Open Collector Devices

- Case when output voltage is $V_{OLTyp}$.
  - $I_R = I_{OLMax} - 3I_{ILMax}$
  - $R_{Min} = \frac{V_{CC}}{I_R}$
Open Collector Devices

- Case when output voltage is $V_{OHMin}$.
  - $I_R = 4I_{OHMax} + 3I_{IHMax}$
  - $R_{Max} = \frac{(V_{CC} - V_{OHMin})}{I_R}$
- Select: $R_{Min} \leq R \leq R_{Max}$
Tri-State Devices

• This kind of device include a third electrical state called *high impedance* or *Hi-Z*.

• This new state is controlled by an input control line called *output enable*. When this input is asserted the device behaves like a normal gate, otherwise, the output behaves like an open circuit.

<table>
<thead>
<tr>
<th>C'</th>
<th>A</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>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Hi-Z</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Hi-Z</td>
</tr>
</tbody>
</table>
Tri-State Devices
Tri-State Devices

• One application of tri-state devices is to be used to connect several devices to a single bus. When changing which output is connected to bus one must ensure that all outputs must first go into the hi-Z state thus avoiding the possibility that two outputs would be connected to the bus simultaneously.
Schmitt Trigger Gates

- Used for wave-shaping purposes.
- Level sensitive with output switching state at two distinct trigger levels, called *lower trigger level* $- \text{V}_{T-}$ and *upper trigger level* $- \text{V}_{T+}$. The difference between the two trigger levels is called *hysteresis* ($\text{V}_{T+} - \text{V}_{T-}$).
- Shown below is the input-output transfer characteristic.
Schmitt Trigger Gates

- Transforming an analog input signal into a clean square form signal.