Counters

- The simplest circuit for binary counting is a multibit divider.

- Each bit toggles on the downward edge of the preceding bit.
- The timing is asynchronous.
- This particular circuit is called a ripple counter.
Transients

- The timing diagram for the ripple counter shows a delay for each transition.

- In a ripple counter at a clock edge each data bit must change before the next higher bit can change.
- The apparent counts that exist during the clock transition are called *transients*. 
Output Latches

- Transients can be eliminated by using D-type latches on the outputs.

- The ripple counter is updated on the falling edge of the clock.
- The D flip-flops are clocked on the rising edge of the clock, long after the values of all bits are set.
- Transients are suppressed.
Preset Counter

- Logic can preselect the early termination of the count to some value \( n \).

- This circuit divides by 10. When the clock causes both \( D_3 \) and \( D_1 \) to be high, a clear is sent to all flip-flops. The first case of this count is at the clock from 9 to 10.
- Transient problems will affect the operation of this circuit.
- Data latches will help transients here as well.

- The set and clear of the flip-flops can be used to preload a starting count.
Pulse Generator

- This circuit uses a counter to generate a periodic narrow pulse.

Each 74LS163 chip has four internal flip-flops.
- ENT and ENP are the JK inputs for the flip-flops.
- LD is the SET input for the flip-flops.

- The counter is loaded with a value D from 0-255 (0-FF\textsubscript{H}). With each clock rising edge the count increases by one. When FF\textsubscript{H} is reached, RC goes high. This reloads the counter to repeat the cycle. The output is high for one clock cycle and low for 256-D cycles.
Synchronous Counters

- A true synchronous counter requires that all flip-flops be clocked at the same time.
  1. Minimize noise since all inputs are well defined
  2. Reduce propagation time
  3. Eliminate transient counts

- The inputs must have additional logic to control each bit as in the JK divide by $2^n$.

\[ D_0 \text{ is dividing the input clock by 2.} \]
\[ D_1 \text{ is dividing the input clock by 4. It toggles when } D_0 = 1. \]
\[ D_2 \text{ is dividing the input clock by 8. It toggles when } D_1 \& D_0 = 1. \]
Divide by 3

- Other latches can be used to make counters such as this D-type divider.

- The truth table shows that the sequence repeats every 3 clock cycles.

<table>
<thead>
<tr>
<th>CLK</th>
<th>$D_0$</th>
<th>$Q_0$</th>
<th>$D_1$</th>
<th>$Q_1$</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
State Diagram

- A state diagram shows the sequence between possible outputs.

   ![State Diagram Example](image)

   - Forbidden states occur when a combination cannot be reached in the sequence.
   - In the Divide by 3 circuit, \( Q_0 = 1 \) and \( Q_1 = 1 \) cannot be reached. If it occurs, \( D_0 = 1 \) and \( D_1 = 0 \) so the next count is 1.

   - Compare to a state diagram for traffic signals.

   ![Traffic Signal Diagram](image)
Registers

- Registers are like latches and have multiple flip-flops on one IC with one clock and clear.
- Typically there is one output (or output pair $Q/\overline{Q}$) per input.
- All are designed to “hold” a set of bits.
- A transparent latch is based on RS flip-flops, and passes the input to the output when enabled and hold the output constant when disabled.

- A type-D register is based on D-type flip-flops, and transfers the input to the output only on a specified clock edge.

- Many registers have an enable feature to control whether or not the clock has an effect.
  - If not enabled, the register outputs are held constant.
Shift Registers

- A shift register moves a pattern of bits in an array of flip-flops without altering the pattern.
- This version is a Serial In/Parallel Out (SIPO) register.

- The truth table show the movement of the bits in the register.

<table>
<thead>
<tr>
<th>CLK</th>
<th>IN</th>
<th>Q₀</th>
<th>Q₁</th>
<th>Q₂</th>
<th>Q₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Parallel In/Single Out (PISO)

- A PISO register loads a set of bits then shifts them serially. The LD is a logic level that initiates a parallel load of input data. The CLK handles the shifting.

- In this example truth table the input data is 0110.
Psuedorandom Noise Generator

- A shift register can be used to generate a seemingly random stream of bits.

- If the register begins at 0, the input continues to be 0 and there is no change of state.

- If the register begins at 1, that one bit will shift through the register at each clock cycle.
  - When it reaches Q_6 and Q_7 then those two clocks will input a 1 instead of a 0 to the input.
  - Those two consecutive bits clock through and at the end generate a 101 pattern to the input.
  - Only after 255 clock cycles does the number 1 reemerge.

- The register generates all values from 1-255 in an arbitrary order that is set by the specific feedback through the XOR gate.
### Truth Table with Feedback $Q_2@Q_3$

<table>
<thead>
<tr>
<th>CLK</th>
<th>$Q_0=Q_2@Q_3$</th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$Q_3$</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>0-&gt;1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>
Truth Table with Feedback $Q_1@Q_3$

<table>
<thead>
<tr>
<th>CLK</th>
<th>$Q_0=Q_1@Q_3$</th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$Q_3$</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0-1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>0-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- This feedback combination does not go through all 15 possible combinations, but only 7, effectively a 3-bit pseudorandom generator.

There are 6 possible feedback choices for 4 bits:

$Q_0 = Q_2@Q_3$ gives 15 numbers

$Q_0 = Q_1@Q_3$ gives 7 numbers

$Q_0 = Q_0@Q_3$ gives 15 numbers

$Q_0 = Q_1@Q_2$ gives 7 numbers after 2 is reached

$Q_0 = Q_0@Q_2$ gives 7 numbers after 3 is reached

$Q_0 = Q_0@Q_1$ gives 3 numbers after 6 is reached
The pseudorandom number generator state diagram shows the forbidden and isolated states.

- The pattern sequence here is 6-13-11, a cycle of 3. Most starting points end in this cycle.
- Starting points at 4, 8, 12 or 0 end up stuck at 0.
Memory

- Random Access Memory (RAM) is a selectable register.

The basic components of a RAM are

- Input address bits ($A_i$)
- Chip select bit ($CS$)
- Output enable bit ($OE$)
- Write enable bit ($WE$)
- Input/Output data bits ($D_i$)

Chip select, output enable, and write enable will sometimes come under other names with slightly different function. Some of these include memory enable, read/write, address strobe and data strobe. Strobes mean that the memory is controlled by a clock edge rather than a level.

- RAMs are usually specified by the number of possible addresses ($2^n$ where $n$ is the number is address bits) by the number of data bits.
- For example a chip with 18 address bits and 8 data bits would be a 256K x 8 RAM.
- Note that $K=2^{10}=1024$, which is not really 1000, but it is counted that way. $M=2^{20}$ and is treated as if it were $10^6$. 
**Static RAM (SRAM)**

- Static RAM uses flip-flops as the basic storage element. The “memory” position of the flip-flop holds the data and new data is inserted by asserting a 1 or 0 at the flip-flop input while it is enabled.
- The entire memory chip is nothing more than a huge array of flip-flops.
- Like any gate circuit, when the power is off, the signals go away, so any data stored would be lost.
- The biggest advantages of SRAMs are speed and simplicity.

**SRAM Timing**

![SRAM Timing Diagram]

**Battery-Backup SRAM**

This is typically designed as an printed circuit card that includes low-power CMOS SRAM and a long-life battery. When the power is off, a special ultra-low power circuit kicks in and preserves the data on the flip-flops.
**Dynamic RAM (DRAM)**

- Dynamic RAM uses charged capacitors as the basic storage element.
- A capacitor can hold a charge for a time based on the leakage resistance in parallel with the capacitor.
- On a chip this is about $10^9 \, \Omega$. With a 10 pf capacitance the leakage time constant is 10 ms.

![Diagram of capacitor and FET](image)

- DRAMs have the advantage of permitting greater memory density since there is only one FET per bit as opposed to 4 FETs in a gated flip-flop.
- The disadvantage is primarily the added circuitry needed to make sure that the leaking capacitors are repeatedly recharged.
- This requires regular reading and rewriting of all the memory bits on the chip.
**Read-Only Memory (ROM)**

- ROMs are nonvolatile memory chips.

The transistor in the shaded box either exists or is “burned” leaving an open connection. If the transistor is present a select gives a “0”, otherwise it gives a “1”.

- They are best used for applications where one wants a hardwired pattern to always be present (eg. startup program sequences, character generators, basic system instructions).

- PROM stands for programmable read-only memory.
- An eraseable PROM (EPROM) has circuitry to undo the burned connection.