Registers and Counters

Jee-Hwan Ryu

School of Mechanical Engineering
Korea University of Technology and Education

Circuits that include flip-flops are usually classified by the function they perform:
- Registers
- Counters

Register is a group of flip-flops.

Each flip-flop is capable of storing one bit of information.

An n-bit register consists of a group of n flip-flops.

Register is a group of binary cells suitable for holding binary information.

A counter is essentially a register that goes through a predetermined sequence of states.
Registers

- Clock = 0 to 1; Input information is transferred to output: I -> A
- Clock = 0 and 1; Output unchanged
- Clear = 0; Clearing the register to all 0’s prior to its clocked operation.
- Clear: asynchronous input.

Register with Parallel Load

- **Synchronous digital systems** have a master clock generator that supplies a continuous train of clock pulses.
- The transfer of new information into a register is referred to as **loading** the register.
- If all the bits of the register are loaded simultaneously with a common clock pulse, we say that the loading is done in parallel.
- The load input determines whether the next pulse will accept new information or leave the information in the register intact.
Register with Parallel Load

- Load = 1; the inputs are transferred into the register
- Load = 0; maintain the content of the register
- Because the D flip-flop does not have a "no change"

Shift Registers

- Capable of shifting its binary information in one or both directions
Shift Registers: Serial Transfer

To prevent the loss of information stored in the source register.

Serial-Transfer Example

Table 6-1: Serial-Transfer Example

<table>
<thead>
<tr>
<th>Timing pulse</th>
<th>Shift register A</th>
<th>Shift register B</th>
<th>Serial output of B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial value</td>
<td>1 0 1 1</td>
<td>0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>After $T_1$</td>
<td>1 1 0 1</td>
<td>1 0 0 1</td>
<td>0</td>
</tr>
<tr>
<td>After $T_2$</td>
<td>1 1 1 0</td>
<td>1 1 0 0</td>
<td>1</td>
</tr>
<tr>
<td>After $T_3$</td>
<td>0 1 1 1</td>
<td>0 1 1 0</td>
<td>0</td>
</tr>
<tr>
<td>After $T_4$</td>
<td>1 0 1 1</td>
<td>1 0 1 1</td>
<td>0</td>
</tr>
</tbody>
</table>

A is transferred into B, while the content of A remains unchanged.

Korea University of Technology and Education
Serial Addition

- **Operation**
  - The A register -> augend
  - The B register -> addend
  - Carry -> 0
- The SO of A and B provide a pair of significant bits for the FA
- Output Q gives the input carry at z
- The shift-right control enables both registers and the carry flip-flop.
- The sum bit from S enters the leftmost flip-flop of A
- Parallel adder needs more circuits than serial adder

**Korea University of Technology and Education**

State Table for Serial Adder

<table>
<thead>
<tr>
<th>Present State</th>
<th>Inputs</th>
<th>Next State</th>
<th>Output</th>
<th>Flip-Flop Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q</td>
<td>X</td>
<td>y</td>
<td>Q</td>
<td>S</td>
</tr>
<tr>
<td>0</td>
<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>
<td>1</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>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</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>1</td>
</tr>
</tbody>
</table>

**Present value of carry**

- \( J_Q = x \cdot y \)
- \( K_Q = x' \cdot y' = (x + y) \)
- \( S = x \oplus y \oplus Q \)

**By k-map**
Universal Shift Register

If the register has both shifts and parallel load capabilities, it is referred to as a universal shift register.

- A clear control to clear the register to 0.
- A clock input to synchronize the operations.
- A shift-right control to enable the shift right operation and the serial input and output lines associated with the shift right.
- A shift-left control to enable the shift left operation and the serial input and output lines associated with the shift left.
- A parallel-load control to enable a parallel transfer and the n input lines associated with the parallel transfer.
- n parallel output lines
- A control state that leaves the information in the register unchanged in the presence of the clock.
Universal Shift Register

- Shift register are often used to interface digital systems.
- Suppose it is necessary to transmit an n-bit quantity between two points.
  - It will be expensive to use n lines to transmit the n bits in parallel.
  - It is more economical to use a single line and transmit the information serially, one bit at a time.
  - The transmitter accepts the n-bit data in parallel into a shift register and then transmits the data serially along the common line.
  - The receiver accepts the data serially into a shift register.
  - When all n-bits are received, they can be taken from the outputs of the register in parallel.
- The transmitter: a parallel-to-serial conversion of data, the receiver: a serial-to-parallel conversion.
Counters

- A register that goes through a prescribed sequence of states upon the application of input pulses is called a counter.
- The input pulses may be clock pulses or they may originated from some external source and may occur at a fixed interval of time or at random.
- A counter that follows the binary number sequence is called a binary counter.
- An n-bit binary counter consists of n flip-flops and can count in binary from 0 through \(2^n - 1\).

Counters in two categories

- Ripple counters
- Synchronous counters

Ripple counters

- The flip-flop output transition serves as a source for triggering other flip-flops.
- The C input some or all flip-flops are triggered not by the common clock pulses, but rather by the transition that occurs in other flip-flop outputs.

Synchronous counters

- The C inputs of all flip-flops receive the common clock.
Binary Ripple Counters

Series connection of Complementing flip-flops

How about for Cont-down?
- Positive edge triggered
- Or connected to comp. out

Fig. 6-8 4-Bit Binary Ripple Counter

Korea University of Technology and Education

Count Sequence for a Binary Ripple Counter

<table>
<thead>
<tr>
<th>Count sequence $A_3$ $A_2$ $A_1$ $A_0$</th>
<th>Conditions for Complementing</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0 \ 0 \ 0 \ 0$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 0 \ 0 \ 1$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 0 \ 1 \ 0$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 0 \ 1 \ 1$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 1 \ 0 \ 0$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 1 \ 0 \ 1$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 1 \ 1 \ 0$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$0 \ 1 \ 1 \ 1$</td>
<td>Complement $A_0$</td>
</tr>
<tr>
<td>$\ldots \ldots$</td>
<td>$A_0$ will go from 1 to 0 and complement $A_1$; $A_1$ will go from 1 to 0 and complement $A_2$; $A_0$ will go from 1 to 0 and complement $A_1$; and so on...</td>
</tr>
<tr>
<td>$1 \ 0 \ 0 \ 0$</td>
<td></td>
</tr>
</tbody>
</table>

Korea University of Technology and Education
1. Q1 is complemented on the negative edge of every count pulse.
2. Q2 is complemented if Q8=0 and Q1 goes from 1 to 0. Q2 is cleared if Q8=1 and Q1 goes from 1 to 0.
3. Q4 is complemented when Q2 goes from 1 to 0.
4. Q8 is complemented when Q4Q2=11 and Q1 goes from 1 to 0. Q8 is cleared if either Q4 or Q2 is 0 and Q1 goes from 1 to 0.
To count from 0 to 999, we need a three-decade counter. When Q8 in one decade goes from 1 to 0, it triggers the count.

Clock pulses are applied to the inputs of all flip-flops. A common clock triggers all flip-flops simultaneously rather than one at a time as in a ripple counter.
Synchronous Binary Counter

- The first stage A0 has its J and K equal to 1 if the counter is enabled.
- The other J and K inputs are equal to 1 if all previous low-order bits are equal to 1 and the count is enabled.
- A flip-flop in any other position is complemented when all the bits in the lower significant positions are equal to 1.
  - A3A2A1A0=0011 → 0100

Up-Down Binary Counter

- Up input control=1 : count up (the T inputs receive their signals from the values of the previous normal outputs of the flip-flops.)
- Down input control=1, up input control=0 : count down
- Up=down=0 : unchanged state
- Up=down=1 : count up
Example

- Design a synchronous BCD counter with J–K flip-flops.

\[
\begin{align*}
T_{Q1} &= 1 \\
T_{Q2} &= Q'Q_1 \\
T_{Q4} &= Q_2Q_1 \\
T_{Q8} &= Q_8Q_1 + Q_4Q_2Q_1 \\
y &= Q_8Q_1
\end{align*}
\]
Binary Counter with Parallel Load

- Parallel load for initial number
- Input load control=1; disables the count sequence, data transfer
- Load = 0 and count = 1; count
- Load = 0 and count = 0; unchanged
- Carry out = 1 (all flip-flop = 1 and Count = 1)

Korea University of Technology and Education

BCD Counter using Binary Counter with Parallel Load

- The AND gate detects the occurrence of state 1001(9) in the output. In this state, the load input is enabled and all-0's input is loaded into register.
- The NAND gate detects the count of 1010(10), as soon as this count occurs the register is cleared.
- A momentary spike occurs in output A2 as the count goes from 1001 to 1010 and immediately to 0000

Korea University of Technology and Education
Other Counters

(a) Logic diagram

(b) State diagram

Example

• Design a counter with the following repeated binary sequence: 0, 1, 2, 4, 6. Use D flip-flops
**Ring Counter**

- For generating **timing signal** that control the sequence of operations
- A circular shift register with only one flip-flop being set at any **particular time**; all others are cleared.
- The single bit is shifted from one flip-flop to the other.
- \(2^n\) timing signals need \(2^n\) flip-flops

![Ring Counter Diagram](image)

---

**Johnson Counter**

- A circular shift register with the complement output of the last flip-flop connected to the input of the first flip-flop.
- A \(k\)-bit switch-tail ring counter will go through a sequence of \(2^k\) states.
- A **Johnson counter** is a \(k\)-bit switch-tail ring counter with **2k decoding gates** to provide outputs for \(2^k\) timing signals.

![Johnson Counter Diagram](image)

---

*Korea University of Technology and Education*
Memory Decoding

- The equivalent logic of a binary cell that stores one bit of information
- The binary cell stores one bit in its internal flip-flop
- It has three inputs and one output. The read/write input determines the cell operation when it is selected.

![Logic diagram](a) Logic diagram

![Block diagram](b) Block diagram

**Fig. 7-5 Memory Cell**

---

**Internal Construction**

4 words of 4 bits

=16 binary cells

![Diagram](Fig. 7-6 Diagram of a 4×4 RAM)
Read-Only Memory

$\text{ROM} = \text{AND gates connected as a decoder} + \text{a number of OR gates}$

![Diagram of a 5 x 32 decoder and AND gates connected as a decoder](image)

Fig. 7-10 Internal Logic of a $32 \times 8$ ROM

**ROM Truth Table (Partial)**

<table>
<thead>
<tr>
<th>Inputs I4 I3 I2 I1 I0</th>
<th>Outputs A7 A6 A5 A4 A3 A2 A1 A0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0</td>
<td>1 0 1 1 0 1 1 0</td>
</tr>
<tr>
<td>0 0 0 0 1 1</td>
<td>0 0 0 1 1 1 0 0</td>
</tr>
<tr>
<td>0 0 0 1 0 1</td>
<td>1 1 0 0 0 1 0 1</td>
</tr>
<tr>
<td>0 0 0 1 1 1</td>
<td>1 0 1 1 0 0 1 0</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>1 1 1 0 0 1</td>
<td>0 0 0 0 1 0 0 1</td>
</tr>
<tr>
<td>1 1 1 0 1 1</td>
<td>1 1 0 0 0 0 1 0</td>
</tr>
<tr>
<td>1 1 1 1 0 0</td>
<td>0 1 0 0 0 0 0 0</td>
</tr>
<tr>
<td>1 1 1 1 1 1</td>
<td>0 0 1 1 0 0 1 1</td>
</tr>
</tbody>
</table>

Korea University of Technology and Education
Combinational Circuit Implementation

\[ A_7(I_4, I_3, I_2, I_0) = \text{Sum of minterms}(0, 2, 3, \ldots, 29) \]

Input → 00011(3)
Others → all '0'
Output → 10110010

Programmable LOGIC ARRAY

Programmable AND array
Programmable OR array