Registers & Counters

• Registers.

• Shift Registers:
  – Serial in, serial out shift register
  – Serial in, parallel out shift register
  – Parallel in, serial out shift register
  – Parallel in, parallel out shift register
  – Shift Register Applications

• Counters:
  – Ripple Counters
  – Synchronous Counters
  – Counter Applications
Registers

- An n-bit register is a collection of n D flip-flops with a common clock used to store n related bits.

Example: 74LS175 4-bit register
Shift Registers

• Multi-bit register that moves stored data bits left/right (1 bit position per clock cycle)
  – Shift Left is towards MSB

  0 1 1 1 ← LSI

  1 1 1 1

  – Shift Right (or Shift Up) is towards MSB

  0 1 1 1

  RSI →

  1 0 1 1

  RSI
Serial In, Serial Out Shift Register

For a n-bit SRG:
Serial Out = Serial In delayed by n clock period

4-bit shift register example:
serin: 1 0 1 1 0 0 1 1 1 0
serout: - - - - 1 0 1 1 0 0
clock: ↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑
Serial In, Parallel Out Shift register

Serial to Parallel Converter

4-bit shift register example:
serin:   1 0 1 1 0 0 1 1 1 0
1Q:      -  1 0 1 1 0 0 1 1 1
2Q:      -  -  1 0 1 1 0 0 1 1
3Q:      -  -  -  1 0 1 1 0 0 1
4Q:      -  -  -  -  1 0 1 1 0 0

clock:    >>>>>>>>>>>>
Parallel In, Serial Out Shift Register

Parallel to Serial Converter

Load/Shift=1
$D_i \leftrightarrow Q_i$

Load/Shift=0
$Q_i \leftrightarrow Q_{i+1}$

SERIN
CLOCK
LOAD/SHIFT
1D
2D
ND

SEROUT

$1Q$
$2Q$
$\ldots$
$NQ$
Parallel In, Parallel Out Shift Register

General Purpose:
Makes any kind of (left) shift register
Bi-directional Universal Shift Registers

Modes:
- Hold
- Load
- Shift Right
- Shift Left

4-bit Bi-directional Universal (4-bit) PIPO

<table>
<thead>
<tr>
<th>Function</th>
<th>Mode S1 S0</th>
<th>Next state QA* QB* QC* QD*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hold</td>
<td>0 0</td>
<td>QA  QB  QC  QD</td>
</tr>
<tr>
<td>Shift right/up</td>
<td>0 1</td>
<td>RIN QA  QB  QC</td>
</tr>
<tr>
<td>Shift left/down</td>
<td>1 0</td>
<td>QB  QC  QD  LIN</td>
</tr>
<tr>
<td>Load</td>
<td>1 1</td>
<td>A  B  C  D</td>
</tr>
</tbody>
</table>
Universal SR Circuit

D      Q
CLK
CLR

S1 S0
SL
HO
LD
SR

D      Q
CLK
CLR

S1
S0
A
RIN

74x194
Shift Register Applications

• **State Registers**
  – Shift registers are often used as the state register in a sequential device. Usually, the next state is determined by shifting right and inserting a primary input or output into the next position (i.e. a finite memory machine)
  – Very effective for sequence detectors

• **Serial Interconnection of Systems**
  – keep interconnection cost low with serial interconnect

• **Bit Serial Operations**
  – Bit serial operations can be performed quickly through device iteration
  – Iteration (a purely combinational approach) is expensive (in terms of # of transistors, chip area, power, etc).
  – A sequential approach allows the reuse of combinational functional units throughout the multi-cycle operation
Shift Register Applications: Serial Interconnection of Systems

Transmitter

Parallel Data from A-to-D converter

Control Circuits

Parallel-to-serial converter

CLOCK

/SYNC

Serial DATA

One bit

Serial-to-parallel converter

Control Circuits

Parallel Data to D-to-A converter

Receiver
Shift Register Applications Example:
8-Bit Serial Adder

Sequential Implementation of:
\[ Z[7..0] = X[7..0] + Y[7..0] \]
Counters

- Clocked sequential circuit with single-cycle state diagram
  - Modulo-m counter = divide-by-m counter
  - Most Common: n-bit binary counter, where $m = 2^n \rightarrow n$ flip-flops, counts $0 \ldots 2^n-1$
4-bit Ripple Counter

- 1 bit divide-by-2
- 2 bit divide-by-4
- 3 bit divide-by-8
- 4 bit divide-by-16

Uses Minimal Logic

Diagrams of 4-bit Ripple Counter:

1. CLK → Q0
2. Q0 → Q1
3. Q1 → Q2
4. Q2 → Q3
5. Q3
Ripple Counter Problem

\[ n \cdot T_{CQ} \text{ for MSB change for n-bit ripple counter} \Rightarrow \text{minimum clk period} \]
Synchronous Counters

• All clock inputs connected to common CLK signal
  – All flip-flop outputs change simultaneously after CLK
  – Faster than ripple counters
  – More complex logic
  – Most frequently used type of counter
Synchronous Serial Counter

- Flip-flops enabled when all lower flip-flops = 1.
- Enable propagates serially — limits speed
- Requires $(n-1) \Delta t < T_{CLK}$
- All outputs change simultaneously $t_{cQ}$ after $CLK \uparrow$
Synchronous Parallel Counter

- Single-level enable logic per flip-flop
- Fastest and most complex type of counter
- Requires $\Delta t < T_{CLK}$
- All outputs change simultaneously $t_{CQ}$ after $CLK\uparrow$
74X163 4-bit Synchronous Parallel Counter

Common Clock
Synchronous Clear
Synchronous Load
Count Enable = ENP \cdot ENT

Load Data Inputs

\begin{itemize}
\item >CLK
\item CLR
\item LD
\item ENP
\item ENT
\item A
\item B
\item C
\item D
\item QA
\item QB
\item QC
\item QD
\item RCO
\end{itemize}

LSB
MSB

RCO = Ripple Carry Out, when Count = 1111 and ENT = 1
## 74X163 State Table

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Current State</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>/CLR /LD ENT ENP</td>
<td>QD  QC  QB  QA</td>
<td>QD* QC* QB* QA*</td>
</tr>
<tr>
<td>0 X X X</td>
<td>X X X X</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>1 0 X X</td>
<td>X X X X</td>
<td>D C B A</td>
</tr>
<tr>
<td>1 1 0 X</td>
<td>X X X X</td>
<td>QD QC QB QA</td>
</tr>
<tr>
<td>1 1 1 0</td>
<td>X X X X</td>
<td>QD QC QB QA</td>
</tr>
<tr>
<td>Count</td>
<td>0 0 0 1</td>
<td>0 0 1 0</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>0 0 1 1</td>
<td>0 1 0 0</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>0 1 0 0</td>
<td>0 1 1 0</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>1 0 0 0</td>
<td>1 0 0 1</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>1 0 1 1</td>
<td>1 0 1 0</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>1 1 0 0</td>
<td>1 1 0 1</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>1 1 1 0</td>
<td>1 1 1 0</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>1 1 1 1</td>
<td>0 0 0 0</td>
</tr>
</tbody>
</table>

Legend:
- **Clear**
- **Load**
- **Hold**
74X169 Up/Down Counter

74X169

>CLK
UP/DN
LD
ENP
ENT
A
B
C
D
QA
QB
QC
QD
RCO

UP/DN = 1 = up  →  RCO = 15
UP/DN = 0 = down →  RCO = 0

up  down  up
Ex: 0,1,2, 1,0,15,14, 15,0,1,2
     RCO  RCO
Counter Applications

- Count the number of times an event takes place
- Control the number of steps in a sequence of fixed actions (a sequencer)
- Generate timing signals (frequency divider, etc.)