Register

- Registers = sets of D FF's used to store data
- Counters = registers that go through predetermined sequences of states

Register

- Sequential logic (can be defined by state table)
- Storage for vector of binary variables
  
  store data, store state, perform data movement/processing, ...

Figure 5-1 4-bit register

- 4 D FF's
- All FF's triggered at same time $\Rightarrow$ parallel load
- $0$ applied to CLEAR resets all FF's asynchronously
  
  (used before, not during, normal clocked operation)
Shift Register

- Shifts stored bits laterally
- One serial input, one serial output
- Parallel output available

<table>
<thead>
<tr>
<th>X</th>
<th>D₁</th>
<th>D₂</th>
<th>D₃</th>
<th>D₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t+1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t+2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t+3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t+4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t+5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t+6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Controlling shift (clock gating)

Shift clock

Perhaps no skew inside SREG, but skew with other blocks still an issue
Shift Register With Parallel Load

- Data can be loaded, accessed in parallel mode
- Useful for parallel to serial translations (conversions)

More Functions: Use Multiplexers

E.g., shift left, shift right, no change, parallel load $\Rightarrow$ 4-to-1 MUX
Controlling Input to Registers

**Goal:** Selectively store data in some registers, leave others' contents alone

- Clock gating: Stop the clock when want no change
- Use FF type that can "hold" (e.g., JK, T w/o inputs)
- Add logic, load control signal

![Gated Clock Diagram]

**Issue:** Different clock arrival times for FFs with, without gating on clock = "skew"

- Better approach: gate the inputs (cf. Figure 5-2)

![Gating Inputs Diagram]

Equivalent: LOAD = \( S_0 \)

- Run clock continuously, use load control to change register contents
Serial Addition

- Add 0011, 1011 (and store result in A)

Exercise:
Given a shift register, compute the parity of its contents