CHAPTER 6  
Sequential Logic Design with PLDS

The first commercially available programmable logic devices were PLAs. PLAs are combinational logic devices containing a programmable AND-OR array. Some early PLAs included a flip-flop on each output of the AND-OR array and were thus the first sequential PLDs, sometimes called micro-sequencers. Most sequential PLDs apply a single, common clock signal to all of their flip-flops.

Second-generation PLDs, such as the GAL16V8 allows the designer to program each output individually to have a flip-flop or not. Because of their tremendous flexibility. Such devices have almost completely fixed-configuration parts. Till-newer devices called Field programmable gate arrays (FPGAs), provide functionality equivalent to several individual PAL or GAL devices coupled with a programmable set of on-chip interconnections. However this chapter deals with PLD-based design.

6.1 PLD Timing Specifications

Several timing parameters are specified for combinational and sequential PLDs. The most important ones are illustrated in Fig 6.1 and explained below.
Figure 6.1c

$t_{CF}$ This parameter also applies to registered outputs. It is the propagation delay from the rising edge of CLK to an internal registered output that goes back to feedback input. If specified, $t_{CF}$ is normally less than $t_{CO}$. However some manufacturers do not specify $t_{CF}$, in which case a logic designer must assume $t_{CF} = t_{CO}$.

$t_{SU}$ This parameter applies to primary, bidirectional, and feedback inputs that affects the D inputs of flip-flops. It is the set-up time requirement for the input signal to be stable before the rising edge of CLK.

$t_{H}$ This parameter applies to signals that affect the D inputs of flip-flops. It is the hold time requirement for the input-signal to be stable after the rising edge of the CLK.

$F_{max}$ This parameter applies to clocked operation. It is the maximum frequency at which the PLD can operate reliably, and is the reciprocal of the minimum clock period. Two versions of this parameter can be derived from the previous specifications, depending on whether the device is operating with external or internal feedback.

External feedback refers to a circuit in which a registered PLD output is connected to the input of another registered PLD with similar timing; for proper operation, the sum of $t_{CO}$ for the first PLD and $t_{SU}$ for the second must not exceed the clock period.

Internal feedback refers to a circuit in which a registered PLD output is fed back to the register in the same PLD; in this case the sum of the $t_{CF}$ and $t_{SU}$ must not exceed the clock period. Connected to the input of another registered PLD with similar timing; for proper operation, the sum of $t_{CO}$ for the first PLD and $t_{SU}$ for the second must not exceed the clock period.

The PLDs are available in various speed grades which are usually indicated by a suffix on the part number, such as “16V8-10”; the suffix usually refers to the $t_{PD}$ specification in nanoseconds. Table 6.1 shows the timing of several popular PLDs. Only $t_{PD}$ applied to combinational outputs of a device, while the last four columns apply to registered outputs. All of the timing specifications are worst-case numbers over the commercial operating range.

When sequential PLDs are used in applications with critical timing it’s important to remember that they have longer setup times than discrete edge triggered registers in the same technology, owing to the delay of the AND-OR array on each D input. Conversely,
under typical conditions, PLD actually has a negative hold-time requirement because of the delay through AND-OR array.

<table>
<thead>
<tr>
<th>Part Numbers</th>
<th>Suffix</th>
<th>tPD</th>
<th>tCO</th>
<th>tCF</th>
<th>tSU</th>
<th>tH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAL16L8, PAL16Rx, PAL20L8, PAL20Rx</td>
<td>-5</td>
<td>5</td>
<td>4</td>
<td>-</td>
<td>4.5</td>
<td>0</td>
</tr>
<tr>
<td>PAL16L8, PAL16Rx, PAL20L8, PAL20Rx</td>
<td>-7</td>
<td>7.5</td>
<td>6.5</td>
<td>-</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>PAL16L8, PAL16Rx, PAL20L8, PAL20Rx</td>
<td>-10</td>
<td>10</td>
<td>8</td>
<td>-</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>PAL16L8, PAL16Rx, PAL20L8, PAL20Rx</td>
<td>B</td>
<td>15</td>
<td>12</td>
<td>-</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>PAL16L8, PAL16Rx, PAL20L8, PAL20Rx</td>
<td>B-2</td>
<td>25</td>
<td>15</td>
<td>-</td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td>PAL16L8, PAL16Rx, PAL20L8, PAL20Rx</td>
<td>A</td>
<td>25</td>
<td>15</td>
<td>-</td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td>PALCE16V8, PALC20V8</td>
<td>-5</td>
<td>5</td>
<td>4</td>
<td>-</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>GAL16V8, GAL20V8</td>
<td>-7</td>
<td>7.5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>GAL16V8, GAL20V8</td>
<td>-10</td>
<td>10</td>
<td>7.5</td>
<td>6</td>
<td>7.5</td>
<td>0</td>
</tr>
<tr>
<td>GAL16V8, GAL20V8</td>
<td>-15</td>
<td>15</td>
<td>10</td>
<td>8</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>GAL16V8, GAL20V8</td>
<td>-25</td>
<td>25</td>
<td>12</td>
<td>10</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>PALCE22V10</td>
<td>-5</td>
<td>5</td>
<td>4</td>
<td>-</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>PALCE22V10</td>
<td>-7</td>
<td>7.5</td>
<td>4.5</td>
<td>-</td>
<td>4.5</td>
<td>0</td>
</tr>
<tr>
<td>GAL22V10</td>
<td>-10</td>
<td>10</td>
<td>7</td>
<td>2.5</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>GAL22V10</td>
<td>-15</td>
<td>15</td>
<td>8</td>
<td>2.5</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>GAL22V10</td>
<td>-25</td>
<td>25</td>
<td>15</td>
<td>13</td>
<td>15</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.1 Timing specifications in nanoseconds of popular bipolar and CMOS PLDs.

6.2 PLD Realizations of Sequential MSI Functions

Here are the several examples for PLD realization of MSI devices that are presented. The hardware descriptive language used for the examples presented here is ABEL. ABEL equations for registered PLD outputs use the clocked assignment operator; :=.

6.2.1 Edge-Triggered Registers

A GAL16V8 can be used to perform the function of MSI 74X374 8-bit register.

The ABEL program for this realization is given below in Table 6.2

Module Eight_bit_reg
Title ‘8-bit-Edge-Triggered Register’
Z74x374 device ‘P16V8R’;
@ALTERNATE
“Input pins
CLK, /OE
D1, D2, D3, D4, D5, D6, D7, D8 pin1, 11;

“Output pins
Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8 pin 19, 18, 17, 16, 15, 14, 13, 12;

“Set definitions
D = [D1, D2, D3, D4, D5, D6, D7, D8];
Q = [Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8];

Equations
Q := D;

end Eight_Bit_Reg

Table 6.2 ABEL program for an 8-bit register

<table>
<thead>
<tr>
<th>GAL16V8R</th>
<th>Z74X374</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>2</td>
</tr>
<tr>
<td>D2</td>
<td>3</td>
</tr>
<tr>
<td>D3</td>
<td>4</td>
</tr>
<tr>
<td>D4</td>
<td>5</td>
</tr>
<tr>
<td>D5</td>
<td>6</td>
</tr>
<tr>
<td>D6</td>
<td>7</td>
</tr>
<tr>
<td>D7</td>
<td>8</td>
</tr>
<tr>
<td>D8</td>
<td>9</td>
</tr>
<tr>
<td>/OE</td>
<td>11</td>
</tr>
<tr>
<td>Q1</td>
<td>10</td>
</tr>
<tr>
<td>Q2</td>
<td>11</td>
</tr>
<tr>
<td>Q3</td>
<td>12</td>
</tr>
<tr>
<td>Q4</td>
<td>13</td>
</tr>
<tr>
<td>Q5</td>
<td>14</td>
</tr>
<tr>
<td>Q6</td>
<td>15</td>
</tr>
<tr>
<td>Q7</td>
<td>16</td>
</tr>
<tr>
<td>Q8</td>
<td>17</td>
</tr>
</tbody>
</table>

Figure 6.2 PLD Realization of 74X374 MSI Register

The same function can be provided by a PAL16R8 if “P16V8R” is changed to “P16R8”.

Figure 6.2 shows the correspondence between 74x374 inputs and the PLD realization.

The ABEL program given above in Table 6.2 uses sets to describe the 8-bit inputs and outputs, which are treated identically by the almost-trivial equation, Q := D. This equation just generates one trivial product term per output.

MSI 8 bit register 74x377 cannot be realized using GAL16V8, as this requires 9 inputs and 8 outputs. Therefore to realize this GAL20V8 need to be used.
6.2.2 Shift Registers

Shift registers use substantially more of PLD’s capability than do the 8bit registers discussed in the previous section. Table 6.3 gives an ABEL program for realizing 74x194 universal shift register using a 16V8

module Four_Bit_Shift_Reg
    title ‘4-bit Universal Shift Register’
    Z74X194 device ‘P16V8R’
    @Alternate

    “Input pins

    CLK, /OE          pin1, pin11;
    RIN, A, B, C, D, LIN pin 2, 3, 4, 5, 6, 7;
    S1, S0, /CLR      pin 8, 9, 12;

    “Output pins

    QA, QB, QC, QD     pin 19, 18, 17, 16;

    “Set definitions

    INPUT = [A, B, C, D ];
    LEFTIN = [B, C, D, LIN ];
    RIGHTIN = [RIN, A, B, C ];
    OUTPUT = [QA, QB, QC, QD ];
    CTRL = [S1, S0];
    HOLD = (CTRL == [0,0] );
    RIGHT = (CTRL == [0,1] );
    LEFT = (CTRL == [1,0] );
    LOAD = (CTRL == [1,1] );

    Equations

    OUT : = /CLR * (HOLD * OUT + RIGHT* RIGHTIN +LEFT *LEFTIN +LOAD * INPUT);
    end Four_Bit_Shift_Reg
6.2.3 Counters

The most popular MSI counter is 74x163 4-bit binary counter. Using ABEL hardware descriptive language the same can be defined in a simple manner as shown below.

```ABEL
module Four-Bit_Counter
  title '4-bit Binary Counter'
  Z74X163 device 'P16V8R'

  "Input pins
  CLK, !OE       pin 1, 11;
  A, B, C, D     pin 2, 3, 4, 5;
  !LD, !CLR, ENP, ENT pin 6, 7, 8, 9;

  "Output pins
  QA, QB, QC, QD, RCO pin 19, 18, 17, 16, 15;

  "Set definitions
  INPUT = [D, C, B, A ];
  COUNT = [QD, QC, QB, QA ];
```

Figure 6.3 PLD realization of 74x194 universal shift register with synchronous clear.
Equations

```
COUNT := !CLR & (LD & INPUT # !LD & (ENT & ENP) & (COUNT + 1)
      # !LD & (ENT & ENP) & (COUNT);
RCO = (COUNT == [1, 1, 1, 1]) & ENT;
```

end Four_Bit_Counter

Table 6.4 ABEL program for a 4-bit binary counter.

```
QA := (/LD * /CLR * ENP * ENT * /QA + /LD * /CLR * /ENT * QA + /LD * /CLR *
      /ENP * QA
      + LD * /CLR * A);
QB := (/LD + /CLR * ENP * ENT*QA * /QB + /LD * /CLR * /ENT *QB + /LD * /CLR *
      /ENP *QB
      + /LD * /CLR * /QA * QB + LD * /CLR * B);
QC := (/LD * /CLR * ENP * ENT * QB * QA * /QC + /LD * /CLR * /ENT * QC + /LD
      * /CLR *
      /ENP * QC + /LD * /CLR * /QB * QC + /LD * /CLR * /QA *QC + LD * /CLR * C);
QD := (/LD * /CLR * ENP * ENT * QC * QB * QA * /QD + /LD * /CLR * /ENT * QD
      + /LD * /CLR * /ENP * QD + /LD * /CLR * /QC * QD + /LD * /CLR * /QB * QD + /LD
      * /CLR * /QA * QD + LD * /CLR * D);
RCO = ( ENT * QA * QB * QC * QD);
```

Table 6.5 Minimized equations for a 4-bit binary counter.

6.2.4 Counter Design with PLDS

Binary counters are good candidates for PLD-based design, for the following reasons

- A large state machine can often be decomposed into two or more smaller state machines where one of the smaller machines is a binary counter that keeps track of how long the other machine should stay in a particular state. This may simplify both the conceptual design and the circuit design of the machine.

- Many applications require almost-binary-modulus counters with special requirements for initialization, state detection, or state skipping. For example a counter in
an elevator controller may skip state 13. Instead of using an off-the-shelf binary counter and extra logic for the special requirements, a designer can put precisely the required functions in a PLD.

- Most standard MSI counters have only 4 bits, while a single 24-pin PLD can be used to create a binary counter with up to 10 bits.