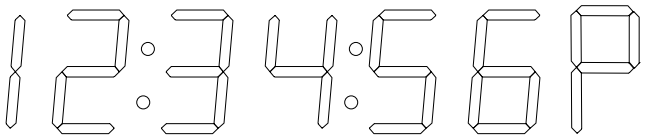
A Digital Clock Design Example

Introduction

The intent of this application note is to show how easy it is to design with an ispLSI® 1032 device by implementing a simple design using many of the features of the device and design software. The digital clock was chosen because its operation is understood by most designers. This example concentrates on the design process rather than the design itself. The design also fits easily into the ispLSI 1032 demonstration box which makes it easy to debug and demonstrate. Figure 1 shows an example of a digital clock design.

Figure 1. Digital Clock Design Example



In implementing this design, advanced features are used to demonstrate the flexibility of the design environment. With the pLSI® and ispLSI Development System (pDS®) software, it is simple to do a complete design using macro library elements which are similar to parts from a typical 7400 TTL Data Book. The logic in a macro can also be modified to meet exact design requirements. At the other extreme, for complete control over the logic within the device, the circuit may be implemented with Boolean equations. Once a custom circuit is created, it can be saved as a macro in a personal macro library for future use.

It is assumed that the reader has read the data sheet and understands the architecture of the ispLSI device. Reading the pDS software manuals makes it easier to understand what is being presented, but is not necessary.

The tools used in implementing this design are:

- pDS software running under Windows™ 3.1 on an IBM™-compatible PC
- An ispDOWNLOAD™ Cable

Entering and Compiling the Design

Before discussing details of the clock design, the following is a quick review of the design flow. In pDS software, designs are created using either Boolean equations or macros taken from the Lattice macro library.

Boolean logic is utilized because it is easy to use. The syntax used is similar to that used in Data I/O's ABEL™ software to design GAL® devices. With Boolean equations, designers have total control over the logic within the pLSI or ispLSI devices. Also, complete access to the advanced architectural features such as the product term Clocks and Reset, the Output Enable control and the hardware XOR is provided.

As powerful as Boolean equations are, it is time consuming to enter a large design using them. For that reason pDS software comes complete with a macro library of standard logic functions which designers can draw from. The macro library consists of several hundred logic elements ranging from simple gates (AND, OR, XOR) to complex functions like counters, multiplexers and adders. If a standard Lattice Semiconductor library macro is close to design requirements, it can be copied to a personal library and modified. This new macro is then saved and used in other designs as needed.

For a non-standard logic function used repeatedly in a design, a macro can be created using a combination of Boolean equations and other macros as described above.

The Design Process

The design process in Figure 2 includes the following simple steps:

1. Enter the design
2. Verify the logic
3. Route the cells
4. Generate the fusemap
5. Program the part

A Digital Clock Design Example

Enter the Design

Entering the design is done using the graphical interface. The Lattice pDS software displays a block diagram of the part similar to the one shown in the data sheet. The design equations or macros are entered by clicking on one of the Logic or I/O Cells using the mouse, and writing the equations into the cell using a simple text editor similar to the Windows Notepad. The graphical interface also allows advanced functions such as clearing a cell, naming a cell, copying the contents of one cell to another or saving the data in a cell to be recalled later.

Verify the Logic

Verifying the logic is done in two places. Each GLB and I/O cell is verified individually. A Cell Verify is a local verify of that single cell only. It checks for problems such as syntax errors, exceeding the number of allowable cell inputs, outputs or product terms and logic errors. Once the design is completely entered, the next step is to perform a Design Verify. The Design Verify performs a Cell Verify on cells which were not previously checked, and then checks all the interconnections within the device for dangling inputs and unconnected outputs. The design must pass a Design Verify before the following steps are performed.

Route the Cells

Routing the cells is the next step. The Router module moves the GLBs and I/O cells so that all specified networks can be interconnected. If you have connected certain signals to specific pins, this information is entered into the design using a menu option in the Router module. Aside from fixing the I/O pins, this is an entirely automatic process and requires no intervention. Due to the optimized design of the Global Routing Pool, route times can be very fast (averaging a few minutes), depending on the size of the design and type of PC.

Generate the Fusemap

The Fusemap generation module uses the routed design to generate the JEDEC file. The JEDEC file provides the data used to program the part. This file has a suffix of .JED. Like the Router program, this is an automatic process.

Programming the Device

The part can be programmed in one of two ways. When using an external device programmer, the user can invoke a communication program to transmit the JEDEC file to the programmer. When using in-system programming (ISP™) in a design, the Lattice design system invokes its own ISP control program (ISP Download). This program uses a cable connected to the PC's Parallel Port to program the part or multiple parts on the board itself.

Clock Design Description

The clock design includes the following modules:

- Control Logic
- Prescaler
- Counters
  - – Seconds
  - – Minutes
  - – Hours

Figure 3 shows a block diagram of the clock modules.

Control Logic

The Control Logic reads the input switches and controls the speed at which the seconds, minutes and hours are incremented. This allows a user to set the clock.

Figure 2. Design Process

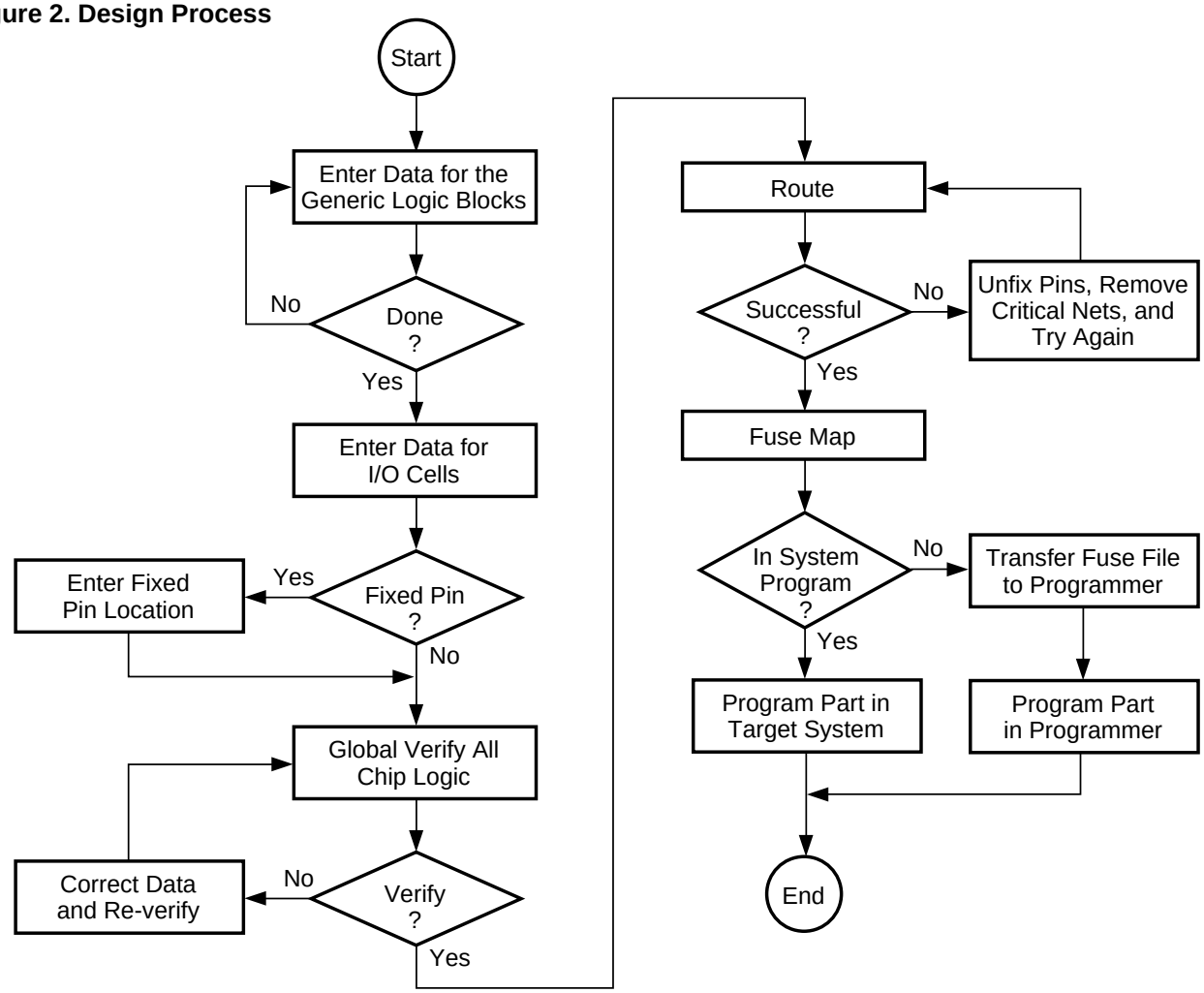
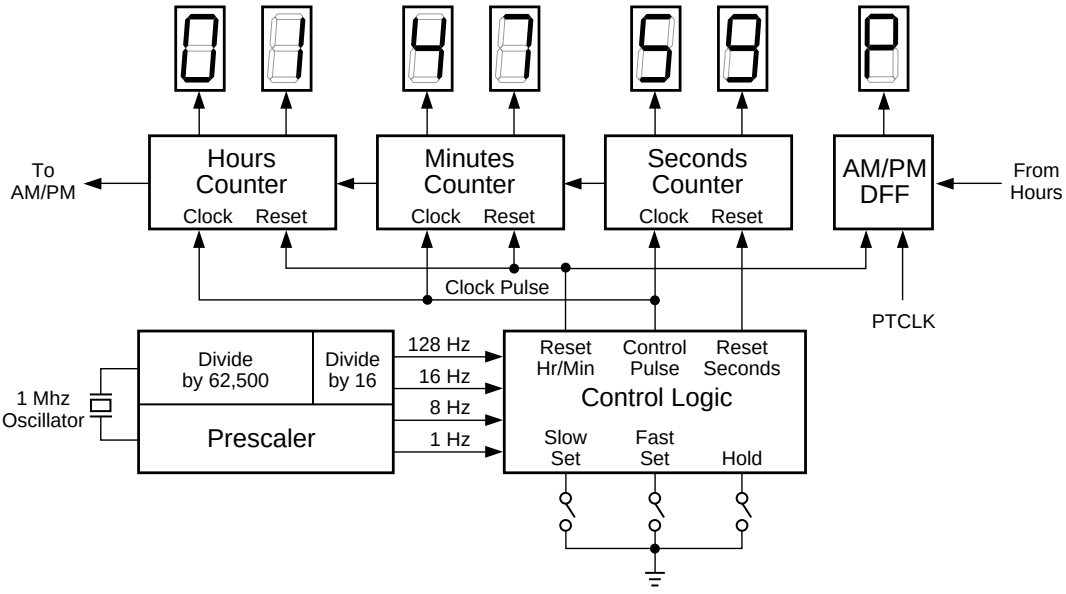


Figure 3. Block Diagram of the Clock



Slow Set, Fast Set, and Hold Signals

The Control Logic, shown in Figure 4, generates the signals necessary to set and run the clock. The inputs to the Control Logic are the three switches: Slow Set, Fast Set and Hold. These inputs are clocked using the Input Register in the I/O Cells. This eliminates switch bounce which affects how the logic operates.

Timing Signals

The other signals coming into the control logic are the timing signals 128Hz, 8Hz and 1PPM. These come from the prescaler and are used for the Fast Set function, Slow Set function and normal operating function respectively. The hours and minutes counters are normally clocked at a rate of one pulse per minute (1PPM). When setting the clock, this frequency is increased to 8 Hz for Slow Set and 128 Hz for Fast Set.

Figure 4 shows the schematic for this circuit and Table 1 shows the Truth Table.

The Prescaler

The Prescaler divides the 1MHz clock into the frequencies needed by the clock. The Prescaler is designed entirely using macros from the Lattice library. The prescaler circuit has two purposes. It divides the 1MHz XTAL Oscillator signal down to 1Hz for the seconds counter clock and also provides the frequencies necessary for the Slow Set and Fast Set functions.

Figure 4. Timing Signal Schematic

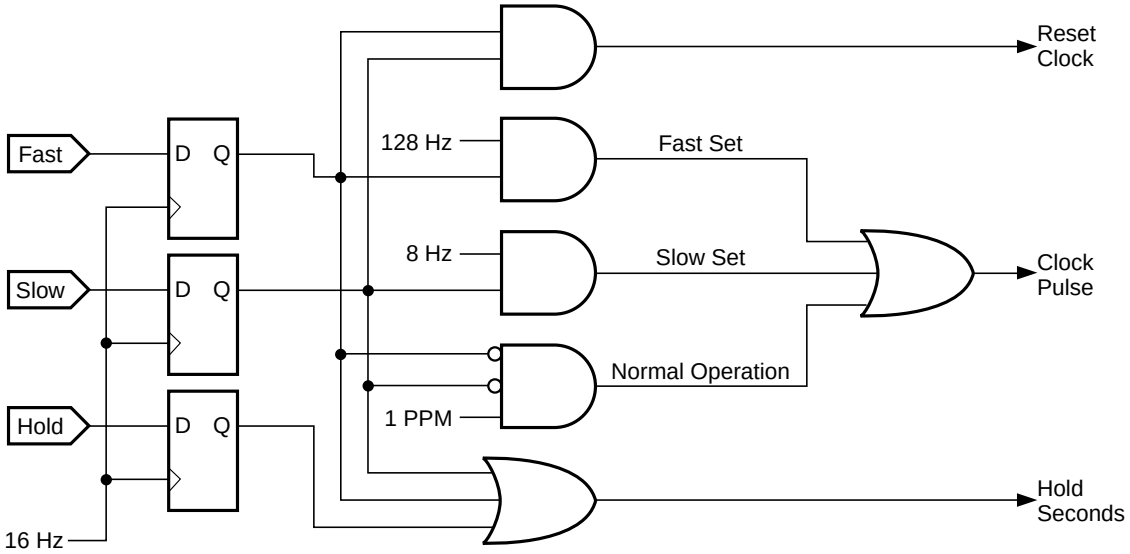


Table 1. Control Logic Truth Table

Slow	Fast	Hold	Clock Pulse	RST SEC	RST HR_MIN	Operation
0	0	0	1 Pulse Per Minute	0	0	Normal Operation
1	0	0	8 Hz	1	0	Slow Set
0	1	0	128 Hz	1	0	Fast Set
1	1	0	None	1	1	Reset to 12:00 AM
X	X	1	None	1	0	Hold Time

The circuit is implemented using two standard macros from the Lattice library (see Figure 5).

A 20-bit divider is necessary to divide the 1MHz clock signal down to 1Hz, but the largest counter in the library is eight bits. Therefore, three counter stages are needed to complete the division.

The approach chosen was to use two eight-bit preloadable counters and a four-bit binary counter cascaded together. The two eight-bit counters are configured as a single 16-bit divider in this circuit. Because a binary counter was chosen for the four-bit function, the mathematics are as follows:

1,000,000 Hz divided by 16 = 62500Hz.

Therefore, the output required of the 16-bit divider is 62500Hz.

65535	Minus	62500	=	3035
Maximum count of the 16 bit counter	Minus	The desired Division	=	Preload Value

A 16-bit divider preloaded to 3035 (0BDB in Hexadecimal) at each terminal count has an output frequency of 16Hz.

The frequencies necessary for the clock set functions are then chosen from the counter outputs. The 8Hz signal (CBU14, Output Q0) advances the minutes counter at the rate of one minute every 7.5 seconds. This is acceptable for the Slow Set function. The Fast Set function uses a 128Hz signal (C16Up, Output Q12) to advance the clock at a rate of one hour every two seconds. The 16 Hz signal is used in the I/O cell input registers as a debounce clock for the switches.

In the final design, the 16-bit counter is placed in GLBs A0 through A7, and the 4-bit counter in GLBs B0 and B1.

Counters

The Seconds and Minutes Counters are Modulo 60 counters which display a decimal count ranging from 00 to 59.

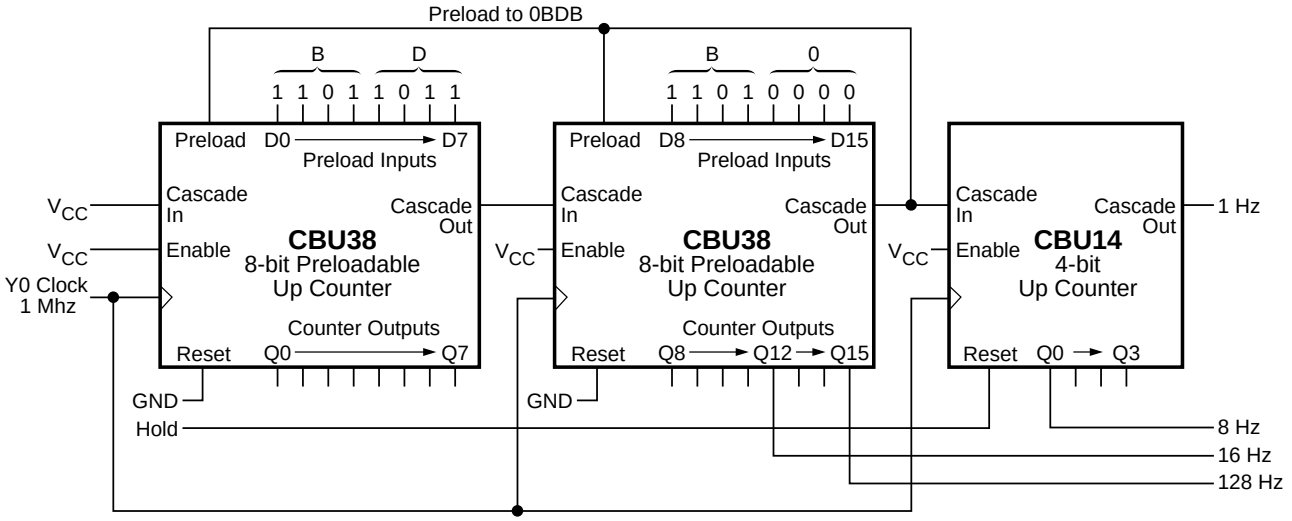
The Hours Counter is a special Modulo 24 counter which counts from one to 12, and has a separate output bit for AM-PM indication. This counter resets to 12 AM and never displays a count of 00.

The Seconds Counter is designed by using a standard macro from the Lattice library and modifying it to suit the needs of the design. This combines the use of macros and Boolean equations.

The Minutes and Hours Counters are designed using state machines optimized for the pLSI 1032 and ispLSI 1032 architectures. The counter which is created is then saved as a custom macro for later use. This optimization saves time and effort on future designs.

There are three controls for setting the clock. These are Slow Set, Fast Set, and Hold. The Slow Set button advances the clock at a rate suitable for selecting the correct minute. The Fast Set button advances the clock at a rate suitable for selecting the correct hour. When either of these buttons are pressed, the Seconds Counters are reset to 00.

Figure 5. Prescaler Sample with Standard Macros



When Slow Set and Fast Set are pressed at the same time, the clock resets to 12:00 AM. The Hold button disables the minutes and hours counters from counting, and resets the seconds to 00 and holds that count until the button is released. This allows the clock to be set to the exact second.

The outputs from the circuit are the seven segment outputs from the Hours, Minutes and Seconds Counters, and the AM/PM Indicator.

Seconds Counter

The Seconds Counter is implemented using both a standard macro from the library for the ones-of-seconds, and a modified counter Macro for the tens-of-seconds. The outputs of these counters is sent to two BCD and then to Seven Segment Display Macros to drive the LEDs.

The Seconds Counter counts from 0 to 59, and then resets to 0. An unmodified CDU24 decimal up counter is used for the Least Significant Digit, but the Most Significant Digit is a modulo 6 counter. This is not a standard function in the library. The easiest way to implement this function is to select a standard four-bit binary counter (CDU24) and modify as shown in Listing 3.

This counter can then be saved in a personal library for future use.

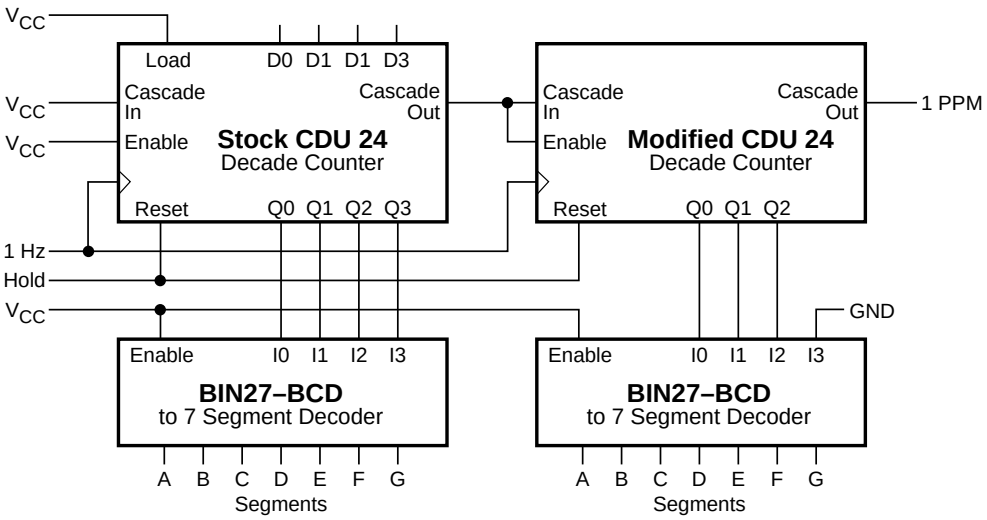
The synchronous reset inputs to the seconds counters are driven by the Hold signal from the control logic. The clock is set to the exact second by setting the Hours and Minutes Counters to a point several minutes ahead, and then pressing the Hold button until the correct second arrives (see Figure 6).

The counters and the seven segment decoders were placed in GLBs, B2 through B7.

Listing 3.

```
MACRO MODULO6 ([Q0..Q2],CLK,EN,CS);
MACROTYPE RX;
     MACROCOMMENT Custom 3 bit Modulo 6
     counter with Sync clear and enable
          for clock design;
     SIGTYPE [Q0..Q2] REG OUT;
     EQUATIONS
          Q0.CLK = CLK;
          Q0 = (Q0&!EN&!CS)
                  $$ (!Q0&EN&!CS);
// Output Q0 toggles after counts
// 0,2,and 4.
          Q1 = (Q1&!EN&!CS)
          $$ ((!Q2&!Q1&Q0&EN&!CS)
          # (!Q2&Q1&!Q0&EN&!CS));
// Output Q1 toggles after counts 1
// and 2.
          Q2 = (Q2&!EN&!CS)
          $$ ((!Q2& Q1& Q0&EN&!CS)
          # (Q2&!Q1&!Q0&EN&!CS));
// Output Q2 toggles after counts 3
// and 4.
     END
END
```

Figure 6. Sample Seconds Counter



A modulo 6 counter is needed for the tens-of-seconds, and it is easily created by modifying a standard Modulo 10 Counter Macro. Once the new macro is created, it is named and saved in the personal library.

The Minutes Counter

The architecture of the Lattice ispLSI and pLSI devices has been optimized for state machine use. The registers in the GLBs are synchronous and several product terms per register are added. Each product term has 18 inputs.

In the seconds counter, since the counters and the decoders are separate, seven GLBs are used. Taking advantage of the wide input gating available to create a state machine counter which directly drives the seven segment outputs, then the number of GLBs is reduced to four. Figure 7 shows a sample minutes counter.

The truth table for a seven segment display is shown in Figure 8.

The state machine is simple. The outputs are the segment drivers, and each output decodes the current state to determine what the next state is. The simplified equation for segment A is shown in Listing 4.

Listing 4. Segment A Equations

```
seg_A =  seg_A & seg_B & seg_C & seg_D
         & seg_E & seg_F & !seg_G
         // Decode state Zero
         # seg_A & seg_B & seg_C & seg_D
         & !seg_E & !seg_F & seg_G
         // Decode state Three
         # seg_A & !seg_B & seg_C
         & seg_D & !seg_E & seg_F
         & seg_G
         // Decode state Five
```

The output for segment A goes to ‘0’ on the following clock whenever states zero, three or five occur. For each of the segments there are fewer ‘0’ transitions than ‘1’. The ‘0’ transitions are decoded to save product terms, and then inverted in the output buffers. This is true on all of the segments except Segment G, which is left in its logic true form. This allows the counter to reset to a zero when a hardware reset is applied. All segments are on except Segment G.

Figure 7. Sample Minutes Counter

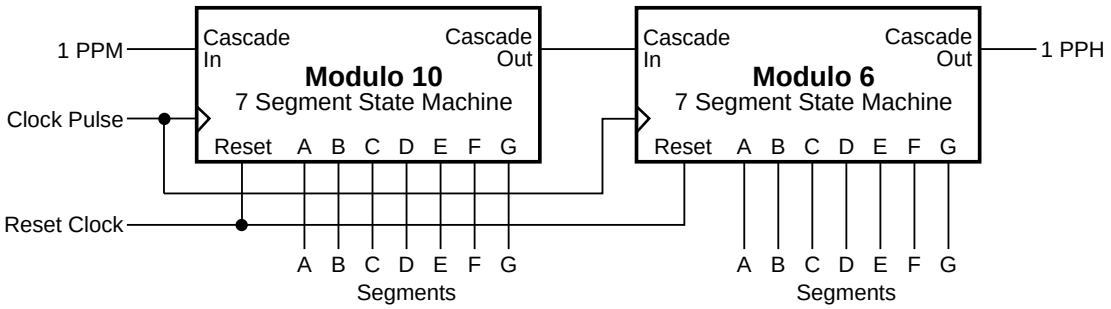
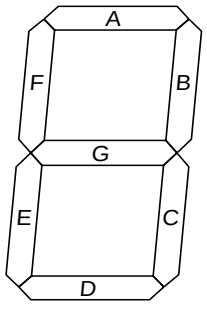


Figure 8. Seven Segment Truth Table



STATE	SEGMENT A	B	C	D	E	F	G	TC
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	0	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	0	0	1	1	1

The Terminal Count (TC) output enables the next stage. The Tens-of-Minutes Counter is similar in construction except that only the states from zero to five are decoded, and the terminal count occurs at state five instead of nine (see Figure 9).

By designing the counters to make best use of the features of the pLSI device family, logic for this counter function is reduced by 40 percent. The Minutes Counters are placed in GLBs C0 through C7 in the final design.

The Hours Counter

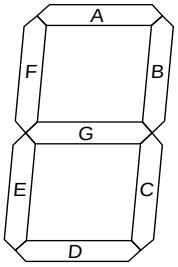
The Hours Counter is constructed using a state machine similar to the one used in the minutes counter. The count sequence for hours is unique compared to most counters.

In the hours stage, both digits are designed as a single counter stage. The reset signal for the hours stage resets the counter to 12 rather than zero (see Figure 10).

A carry out signal is still generated from this counter because an AM/PM indicator is desired, but the carry out is generated when the counter reaches 12 instead of when it rolls over to one. This is consistent with the way clocks operate. Morning starts at 12:00 AM and afternoon starts at 12:00 PM. The AM/PM stage is a D-type flip-flop which uses the carry out signal as an asynchronous product term clock. This register also uses an asynchronous reset to force it to start at 12:00 AM when the clock is reset (see Figure 11).

The Hours Counter and the AM/PM logic are placed in GLBs D0 through D4 in the example file.

Figure 9. Terminal Count at State 5



	SEGMENT							
STATE	A	B	C	D	E	F	G	TC
0	1	1	1	1	1	0	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	0	1	1	0
3	1	1	1	1	0	1	1	0
4	0	1	1	0	1	1	1	0
5	1	0	1	1	1	1	1	1

Figure 10. Sample Hours Counter

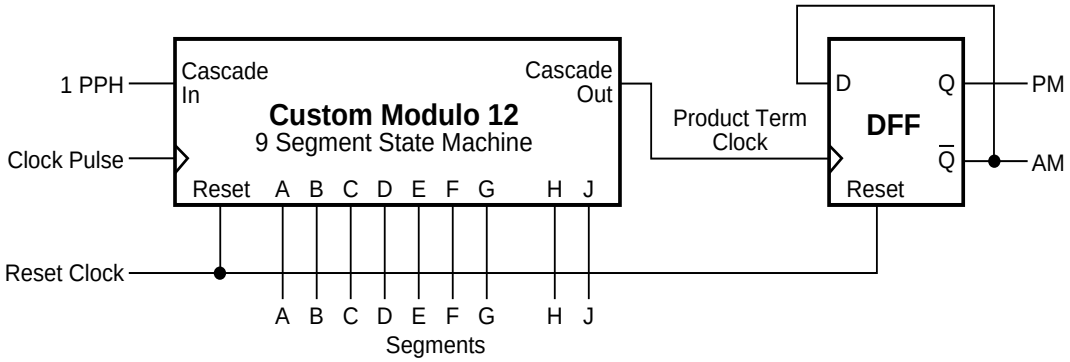
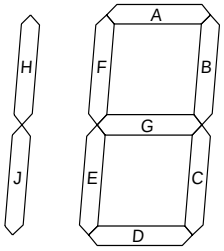


Figure 11. Sample 9 Segment Digit



	SEGMENT									
STATE	A	B	C	D	E	F	G	H	J	TC
1	0	1	1	0	0	0	0	0	0	1
2	1	1	0	1	1	0	1	0	0	1
3	1	1	1	1	0	0	1	0	0	1
4	0	1	1	0	0	1	1	0	0	1
5	1	0	1	1	0	1	1	0	0	1
6	0	0	1	1	1	1	1	0	0	1
7	1	1	1	0	0	0	0	0	0	1
8	1	1	1	1	1	1	1	0	0	1
9	1	1	1	0	0	1	1	0	0	1
10	1	1	1	1	1	1	0	1	1	1
11	0	1	1	0	0	0	0	1	1	0
12	1	1	0	1	1	0	1	1	1	1

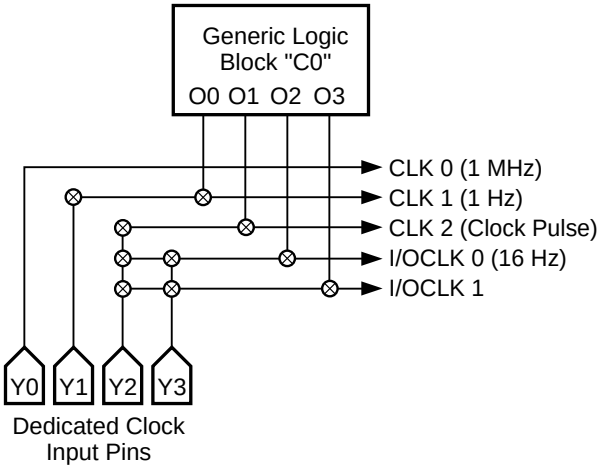
Clock Management

This design makes maximum use of the various Clock modes of the ispLSI 1032. In each GLB, there are four possible clock sources, CLK 0, CLK 1, CLK 2, and a PTCLK. The first source, CLK 0, is a synchronous clock, and is permanently connected to the Y0 Clock Input pin on the device. CLK 1 and CLK 2 are also synchronous and can come from either the external clock pins (Y1 or Y2) or can be generated within the device using the internal clock GLB, 'C0.' The fourth, PTCLK, comes from a Product Term within the GLB. This clock is asynchronous (see Figure 12).

In this clock design, the 1MHz reference clock from the Demo Board is brought in using the Y0 Clock pin and is the clock source used to drive the Prescaler. The 1Hz output of the Prescaler is then routed through the 'C0' GLB to become the CLK 1 Source. This clock is used to increment the seconds counters. The minutes and hours counters are clocked by the signal Clock Pulse on the CLK 2 distribution line. This signal is one pulse per minute during normal operation, 8Hz during Slow Set and 128Hz during Fast Set operations.

The AM/PM Indicator is a D-type flip-flop which is clocked asynchronously using a product term clock (see Figure 10).

Figure 12. Clock Management Modes



LATTICE SEMICONDUCTOR CORPORATION
5555 Northeast Moore Court
Hillsboro, Oregon 97124 U.S.A.
Tel.: (503) 681-0118
FAX: (503) 681-3037
http://www.latticesemi.com

November 1996