Creating Platform Manager Designs with PAC-Designer and Lattice Diamond Software

March 2013 Technical Note TN1259

Introduction

This technical note leads you through the basic steps of designing and implementing a Platform Manager™ design in both PAC-Designer® and Lattice Diamond® design software.

This document covers the following areas related to creating Platform Manager designs:

- Using PAC-Designer to create a Platform Manager project
- Configuring the basic analog functions of Platform Manager in PAC-Designer
- Defining CPLD and FPGA logic in PAC-Designer LogiBuilder
- Completing a basic Platform Manager design in PAC-Designer
- Completing a basic Platform Manager design in a mixed PAC-Designer / Diamond flow
- Finding major functions within the PAC-Designer Platform Manager interface

The design implemented in this technical note requires the following design tools:

- PAC-Designer (Version 6.23 or later)
- Lattice Diamond Software (Version 2.0 or later)
- Platform Manager Development Kit (Optional)

The design in this technical note consists of a complete design for the Platform Manager device. The design is started using only PAC-Designer. After that, the FPGA design portion is implemented in Lattice Diamond. Both parts are linked together using PAC-Designer. The completed design can then be demonstrated on the Platform Manager Development Kit.

You will use both the PAC-Designer and Lattice Diamond design software to create a Platform Manager programming file to implement the sample application. The steps for the design implementation are shown below:

Part 1: PAC-Designer Basics

1. Start a New LPTM10-12107 Design
2. Configure the Analog Inputs for Voltage Monitoring
3. Configure the CPLD Inputs for Control Inputs
4. Configure the CPLD Open Drain Outputs
5. Implement CPLD Sequence Logic
6. Implement CPLD Exception Logic
7. Implement CPLD Supervisory Logic
8. Additional Resources – Power Management Section

Part 2: Platform Manager FPGA Design in PAC-Designer

1. Configure the FPGA I/O Assignments
2. Configure an FPGA Timer
3. Implement the FPGA Sequence and Supervisory Logic

4. Additional Resources for FPGA Design in PAC-Designer:
    a. Utility IP
    b. Changing the Preferred Design Tool
    c. Working with the Text Editor Flow
    d. Simulating Platform Manager from PAC-Designer

Part 3: Platform Manager FPGA Design with Lattice Diamond

1. Setting up a Platform Manager Project in Lattice Diamond
2. Build an FPGA .JED File in Lattice Diamond
3. Link the PAC-Designer Project to the FPGA .JED File

Part 4: Optional – Review the Design on the Platform Manager Evaluation Board

Additional documents:

- DS1036, Platform Manager Data Sheet
- EB58, Platform Manager Development Kit User's Guide
- TN1223, Using the Platform Manager Successfully

Part 1: PAC-Designer Basics for Platform Manager

In this design you will be using the CPLD portion of the Platform Manager device to monitor and sequence supplies and then pass a signal to the FPGA so that it can sequence reset signals. On the evaluation board the supplies are simulated by the slider pots R19 and R20, while the reset signals are simulated using the LEDs.

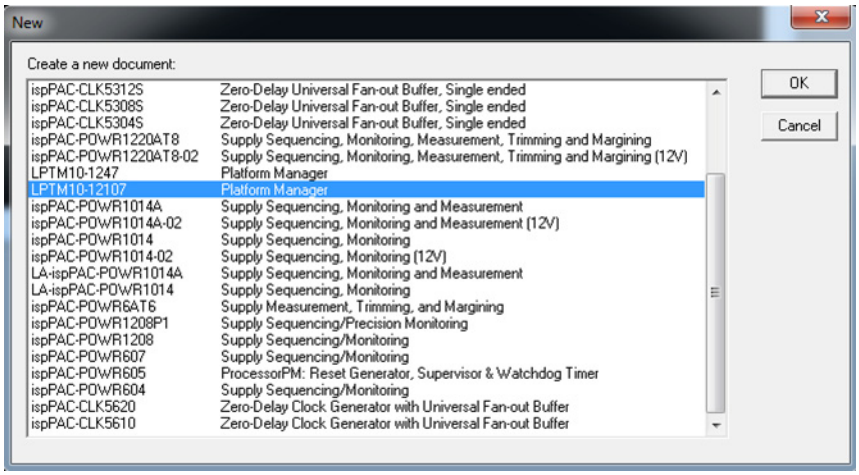
Task 1: Start a new LPTM10-12107 Design

PAC-Designer design files (.PAC files) include analog setting information, digital logic configuration, and project settings. These files are used to generate the .JED file used to program the configuration memory of the Platform Manager device.

To create a new design:

1. Start the PAC-Designer software by either selecting the PAC-Designer icon or **Start > Programs > Lattice PAC-Designer 6.23**. PAC-Designer launches in an empty view.
2. Choose **File > New**. The new project Dialog appears as shown in Figure 1.

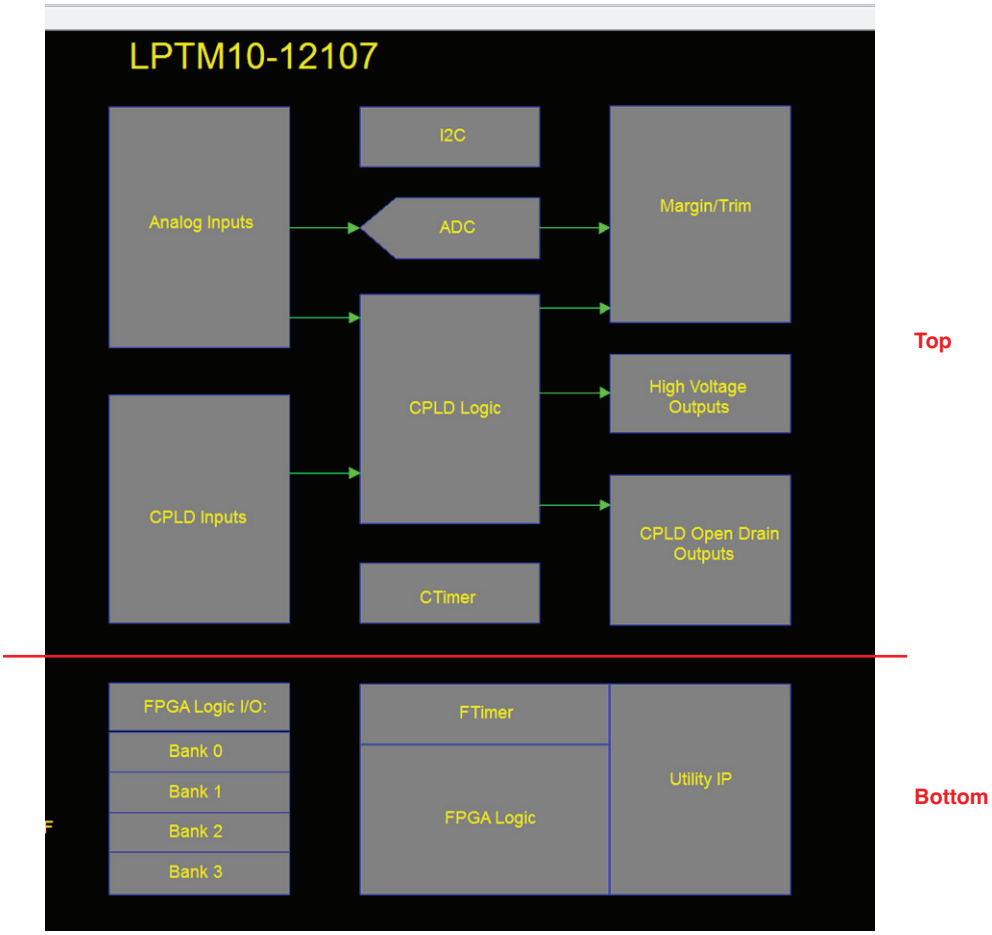
Figure 1. New Project Dialog



3. From the document list, choose **LPTM10-12107 Platform Manager** and click **OK**. The Save As Dialog will appear.
4. Browse to the **<install_path>\Examples** directory, specify the File name as **My_LPTM10_TN1259.pac**, and click **Save**. The block diagram schematic view will appear.

The schematic view of Platform Manager, shown in Figure 2, is divided into a top section and a bottom section. The top section is used to configure the Power Management section of the design. The bottom side is used to configure the Digital Management section of the design.

Figure 2. Schematic View of Platform Manager



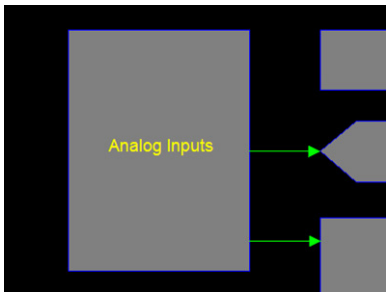
Task 2: Configure the Analog Inputs Block for Voltage Monitoring

The analog inputs of the Platform Manager are configured for voltage monitoring using the Analog Input Settings Dialog, shown in Figure 5

To open the Analog Input Settings Dialog:

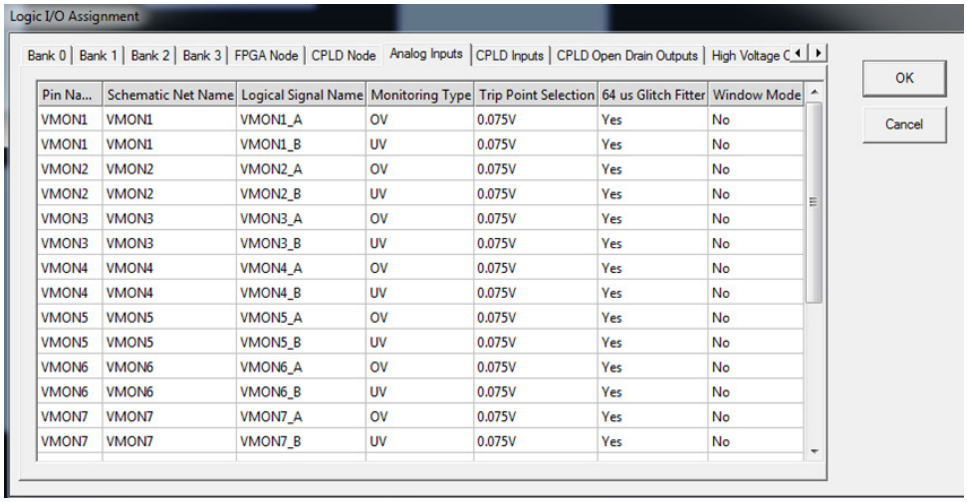
1. Double-click the **Analog Inputs** block in the schematic view, shown in Figure 3.

Figure 3. Analog Inputs Block



2. The Logic I/O Assignment Dialog will appear with the Analog Inputs tab active, shown in Figure 4. Double-click any value in the table to open the Analog Input Settings Dialog, shown in Figure 5.

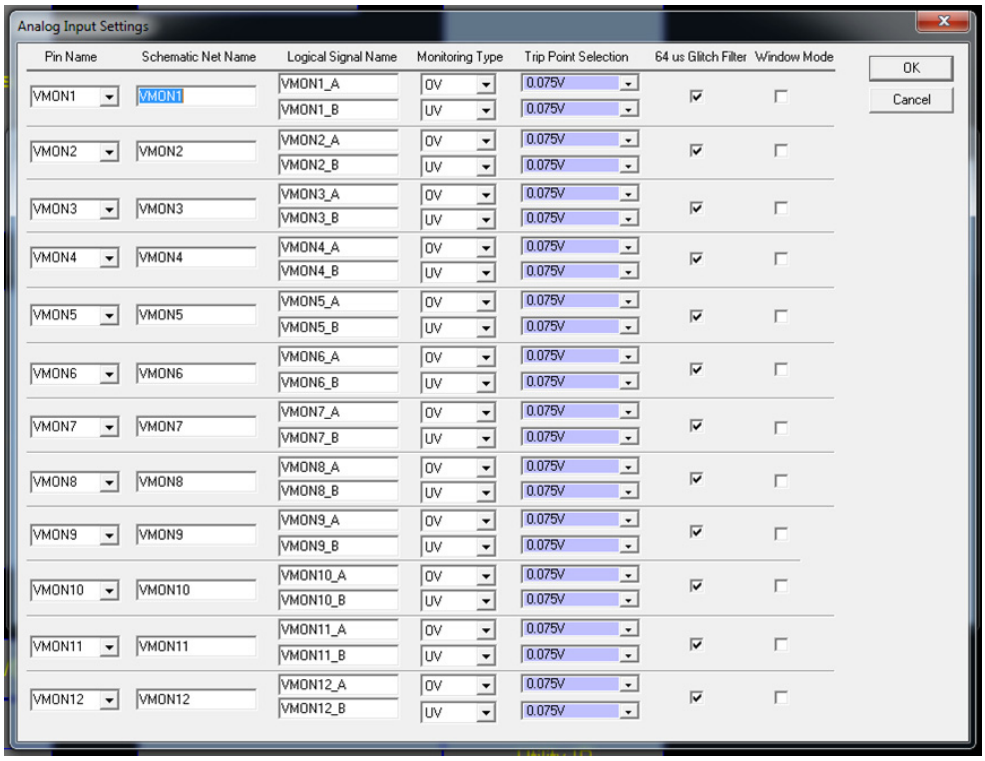
Figure 4. Logic I/O Assignment Dialog – Analog Inputs Tab



Pin Na...	Schematic Net Name	Logical Signal Name	Monitoring Type	Trip Point Selection	64 us Glitch Filter	Window Mode
VMON1	VMON1	VMON1_A	OV	0.075V	Yes	No
VMON1	VMON1	VMON1_B	UV	0.075V	Yes	No
VMON2	VMON2	VMON2_A	OV	0.075V	Yes	No
VMON2	VMON2	VMON2_B	UV	0.075V	Yes	No
VMON3	VMON3	VMON3_A	OV	0.075V	Yes	No
VMON3	VMON3	VMON3_B	UV	0.075V	Yes	No
VMON4	VMON4	VMON4_A	OV	0.075V	Yes	No
VMON4	VMON4	VMON4_B	UV	0.075V	Yes	No
VMON5	VMON5	VMON5_A	OV	0.075V	Yes	No
VMON5	VMON5	VMON5_B	UV	0.075V	Yes	No
VMON6	VMON6	VMON6_A	OV	0.075V	Yes	No
VMON6	VMON6	VMON6_B	UV	0.075V	Yes	No
VMON7	VMON7	VMON7_A	OV	0.075V	Yes	No
VMON7	VMON7	VMON7_B	UV	0.075V	Yes	No

3. The Analog Input Settings will appear as shown in Figure 5.

Figure 5. Analog Input Settings Dialog



These settings configure the following voltage monitor parameters:

- **Schematic Net Name** – User-defined name for the voltage rail at the pin.
- **Logical Signal Name** – User-defined name for the comparator outputs. This signal is input to the CPLD logic.
- **Monitoring Type** – Choose from OV or UV. Defines hysteresis behavior and available trip points of the comparator.
- **Trip Point Selection** – List of 368 pre-defined trip point values related to each logical signal.
- **64 us Glitch Filter** – Optional flag which causes events shorter than 64 us to be ignored.
- **Window Mode** – Combines the output of two comparators to provide window logic.

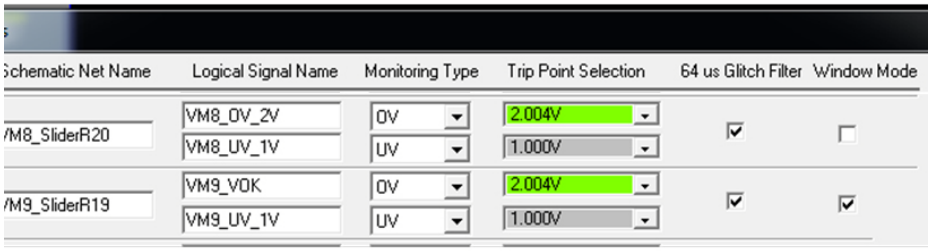
4. VMON8 and VMON9 are connected to the slider potentiometers R20 and R19 on the Platform Manager Evaluation Board. This design uses these two VMONs. Configure the settings as shown in Table 1.

Table 1. Analog Input Settings for Example Design

Pin Name	Schematic Net Name	Logical Signal Name	Monitoring Type	Trip Point	64 us Glitch Filter	Window Mode
VMON8	VM8_SliderR20	VM8_OV_2V	OV	2.004V	Yes	—
		VM8_UV_1V	UV	1.000V	—	—
VMON9	VM9_SliderR19	VM9_VOK	OV	2.004V	Yes	Yes
		VM9_UV_1V	UV	1.000V	—	—

VMON8 and VMON9 should now be configured as shown in Figure 6.

Figure 6. Analog Input Setting Configuration



5. Choose **OK** twice to return to the schematic view. Ignore the warning about CLOCK or RESET for now.

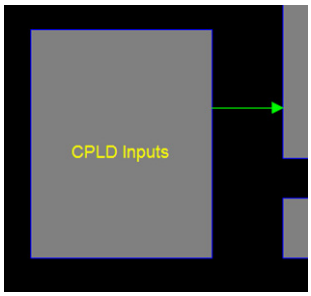
Note: See the Analog Monitor Inputs section of the Architecture Details in the Platform Manager Data Sheet for more details on working with the voltage monitors.

Task 3: Configure CPLD Input Settings

In this task, the CPLD inputs are configured. These are digital input pins passed to the CPLD logic.

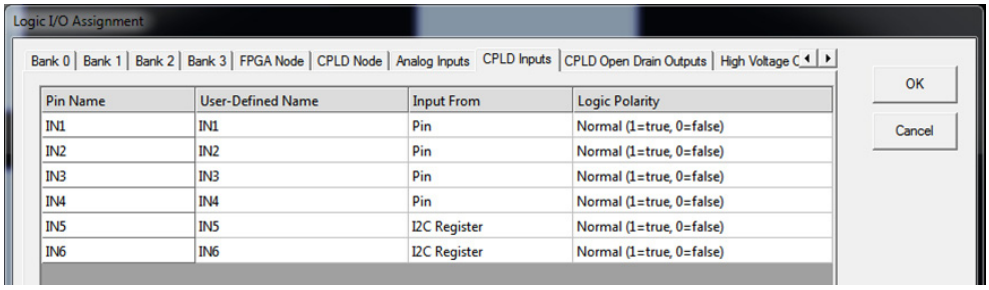
1. Double-click the **CPLD Inputs** block in the schematic view, shown in Figure 7.

Figure 7. CPLD Input Settings Block



2. The Logic I/O Assignment Dialog will open again, this time with the CPLD Inputs tab active. The CPLD input settings are edited directly in this view, shown in Figure 8.

Figure 8. Logic I/O Assignment Dialog – CPLD Inputs Tab



These settings configure the CPLD inputs:

- **User-Defined Name** – User-defined name for the digital inputs. This signal is input to the CPLD logic.
- **Input From** – Sets the signal source as an input pin or JTAG/I^2C.
- **Logic Polarity** – Sets the polarity to normal or inverted.

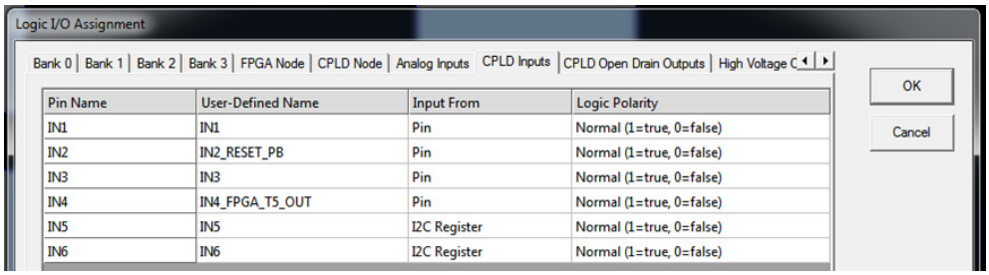
3. IN2 and IN4 are used in this design. IN2 is connected to a push-button reset switch and IN4 is connected to an FPGA output. Update the settings according to Table 2.

Table 2. CPLD Input Settings for the Example Design

Pin Name	User-Defined Name	Input From	Logic Polarity
IN2	IN2_RESET_PB	Pin	Normal
IN4	IN4_FPGA_T5_OUT	Pin	Normal

The input settings should match the screen shot in Figure 9.

Figure 9. CPLD Input Setting Configuration



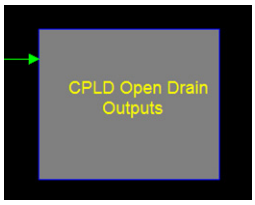
4. Choose **OK** to return to the schematic view (continue to ignore the CLOCK or RESET message for now).

Task 4: Configure the CPLD Open Drain Outputs

In this task the CPLD open drain outputs are configured. These are outputs driven by the CPLD logic.

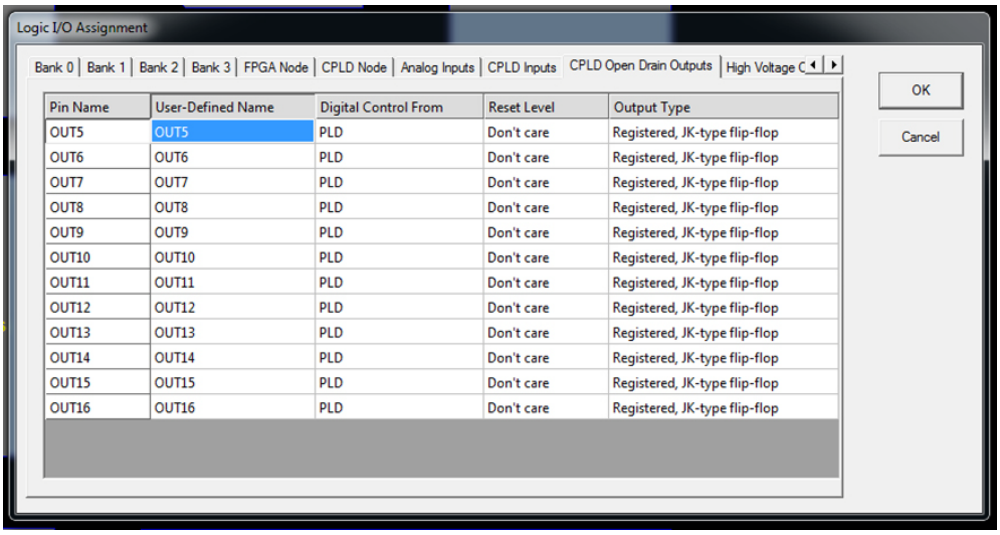
1. Double-click the **CPLD Open Drain Outputs** block in the Schematic View, shown in Figure 10.

Figure 10. CPLD Open Drain Outputs Block



2. The Logic I/O Assignments Dialog will appear with the CPLD Open Drain Outputs tab active. The CPLD Open Drain Outputs settings are edited directly in this view, as shown in Figure 11.

Figure 11. Logic I/O Assignment Dialog – CPLD Open Drain Outputs Tab



These settings configure the CPLD open drain outputs:

- **User-Defined Name** – Signal name used in logic
- **Digital Control From** – Sets output control source as PLD or I^2C register
- **Reset Level** – Status at reset for registered outputs
- **Output Type** – Defines CPLD output type as combinatorial or registered

Note: Outputs used in the CPLD logic sequence must be configured as registered JK-type flip-flops. Outputs controlled by supervisory equations in the CPLD logic can be configured as combinatorial, D-type flip-flop, or T-type flip-flop.

This design uses OUT6, OUT7, OUT8, OUT9 (mapped to LEDs on the Platform Manager Evaluation Board) as well as OUT13 and OUT16 (connected between the FPGA and CPLD sections of the Platform Manager on the evaluation board). Update the settings for these pins according to Table 3.

Table 3. CPLD Open Drain Output Settings

Pin Name	User-Defined Name	Digital Control From	Reset Level	Output Type
OUT6	OUT6_LED21	PLD	Set High	Registered, JK-type flip-flop
OUT7	OUT7_LED22	PLD	Set High	Registered, JK-type flip-flop
OUT8	OUT8_LED23	PLD	Set High	Registered, JK-type flip-flop
OUT9	OUT9_LED24	PLD	Set High	Combinatorial
OUT13	OUT13_toFPGA	PLD	Set High	Registered, JK-type flip-flop
OUT16	OUT16_Timer_toFPGA	PLD	Set High	Registered, D-type flip-flop

The CPLD Open Drain Output Settings should match Figure 12.

Figure 12. CPLD Open Drain Output Settings Configuration

Pin Name	User-Defined Name	Digital Control From	Reset Level	Output Type
OUT5	OUT5	PLD	Don't care	Registered, JK-type flip-flop
OUT6	OUT6_LED21	PLD	Set high	Registered, JK-type flip-flop
OUT7	OUT7_LED22	PLD	Set high	Registered, JK-type flip-flop
OUT8	OUT8_LED23	PLD	Set high	Registered, JK-type flip-flop
OUT9	OUT9_LED24	PLD	Set high	Combinatorial (non-registered)
OUT10	OUT10	PLD	Don't care	Registered, JK-type flip-flop
OUT11	OUT11	PLD	Don't care	Registered, JK-type flip-flop
OUT12	OUT12	PLD	Don't care	Registered, JK-type flip-flop
OUT13	OUT13_toFPGA	PLD	Set high	Registered, JK-type flip-flop
OUT14	OUT14	PLD	Don't care	Registered, JK-type flip-flop
OUT15	OUT15	PLD	Don't care	Registered, JK-type flip-flop
OUT16	OUT16_Timer_toFPGA	PLD	Set high	Registered, D-type flip-flop

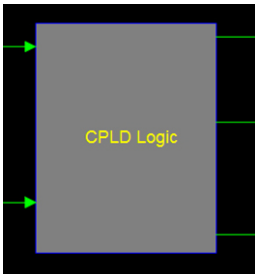
3. Choose **OK** to return to the schematic view (continue to ignore the CLOCK or RESET message for now).

Task 5: Implement CPLD Logic Sequence

In this task you will build a basic sequence and monitor using the CPLD logic of the Platform Manager device. The Logic section is used to control the outputs of the CPLD section based on the analog and CPLD inputs.

1. Double-click the **CPLD Logic** block in the Schematic View, shown in Figure 13.

Figure 13. CPLD Logic Block



The Sequence and Supervisory Logic (CPLD) window will open. The Sequencer will be populated with the building block steps of SM0 (State Machine 0). The Exception section of SM0 and the Supervisory Equation section at the bottom of the window are both empty by default.

Note: This section includes a logic implementation of several steps in PAC-Designer. Users experienced with Power Manager II logic implementation in PAC-Designer may want to skip this section. You can skip this step by importing the logic directly from the completed LPTM10_TN1259.PAC file. This file is available in the design files package that comes with this technical note on www.latticesemi.com. Go to ***File > Import LogiBuilder*** *and browse to the* ***LPTM10_TN1259.PAC*** *file. You can skip to "Part 2: Platform Manager FPGA Design in PAC-Designer" on page 12.*

2. In this example, you will implement a basic sequence design. The design is shown in text form below. For background information on each sequence instruction and the basics of the logic design, see AN6042, Implementing Power Supply Sequencers with Power/Platform Management Devices and PAC-Designer LogiBuilder.
3. To get started inserting and editing steps into the sequence, highlight **Step 0** and press the **Delete** key. A warning Dialog will appear; click **OK** so that Step 0 becomes the "Wait for AGOOD" step. Then double-click **Step 1 Begin Shutdown Sequence**. The Insert Step Dialog with appear. Select and insert the instructions to build the design in Listing 1.

*Note: You can insert steps in the sequence at any step by pressing the **Insert** key or choosing **Edit > Insert Instruction** from the menu. Likewise, you can remove steps in a sequence by highlighting the step and pressing the **Delete** key or choosing **Edit > Delete Instruction**.*

Listing 1. CPLD Logic Sequence

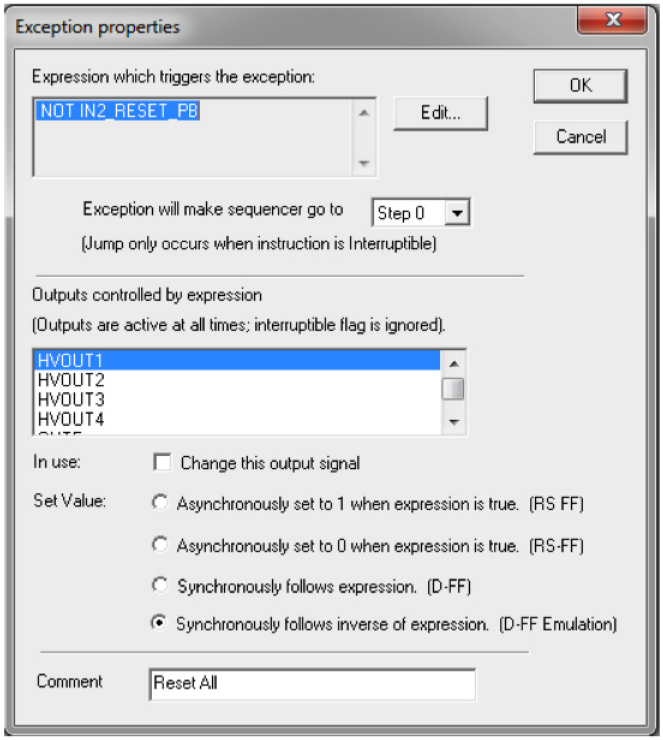
```
// Reset all Outputs. Wait for Analog Calibration
Step 0     Wait for AGOOD : output control OUT6_LED21=1, OUT7_LED22=1, OUT8_LED23=0,
OUT13_toFPGA=1

// If R19 slider pot is in middle position, branch to step 6. Else branch to Step 2.
Step 1   If VM9__VOK Then Goto 6 Else Goto 2

// NOP step needed for branch target before Wait for Timer instruction
Step 2    NOP

// Blink LED21 and loop back to check for R19 position
Step 3    Wait for 245.76ms using timer 1 : output control OUT6_LED21=1

// Blink LED21
Step 4    Wait for 245.76ms using timer 1 : output control OUT6_LED21=0

// Branch back to Step 1 to recheck R19 position
Step 5   Go to step 1

// Turn LED22 On. Wait for R20 slider pot to be in position
Step 6 Wait for VM8_UV_1V : output control OUT7_LED22=0

// Turn LED23 On. Wait here for either slide pot to be Under or Over Voltage
Step 7    Wait for NOT VM9_VOK OR NOT VM8_UV OR VM8_OV_2V : output control OUT8_LED23=0,
OUT13_toFPGA=0

// Wait for both slider pots to be set to zero
Step 8     Wait for NOT VM8_UV_1V AND NOT VM9UV_1V : output control OUT6_LED21=1,
OUT7_LED22=1, OUT8_LED23=1, OUT13_toFPGA=1

// Restart the Sequence
Step 9   Go to step 0

Step 10   Begin Shutdown Sequence

Step 11    Halt (end-of-program)
```

Task 6: Implement CPLD Exception Logic

The logic window also provides the interface for implementing sequence exceptions, multiple state machines, and supervisory logic.

1. For the design, you will add a simple exception equation. Double-click the **<end-of-exception-table>** placeholder in the middle section of the window.
2. E 0 will appear. The default equation is **If <booleanExpr>**. Double-click **E 0** to set up the equation.
3. Click on the **Edit** button to enter the following logic: **NOT IN2_RESET_PB**.
4. Enter the following comment: Reset All
5. The default jump is to Step 0 which is what is needed for a Reset. Click **OK** to complete the creation of exception E 0.

Figure 14. CPLD Exception Properties Dialog



Task 7: Implement CPLD Supervisory Logic

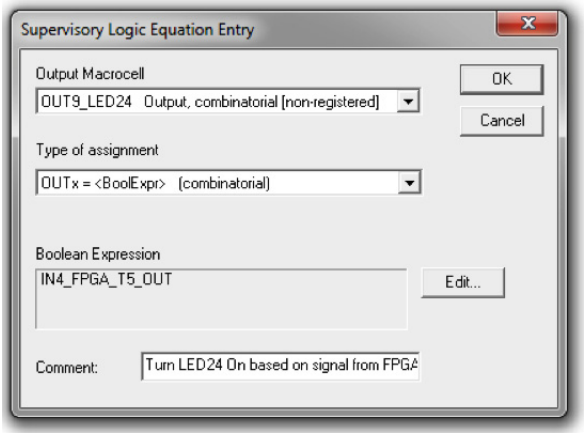
The logic window also provides the interface for implementing sequence exceptions, multiple state machines, and supervisory logic.

1. For the design, you will add a simple supervisory equation. Double-click the **<end-of-supervisory-logic-table>** placeholder in the bottom section of the window.
2. EQ 0 will appear. The default equation is **HVOUT1 = <booleanExpr>**. Double-click **EQ 0** to set up the equation.
3. Choose the output macrocell. This can be an output port, a user node, or a function node (timer or trim control). Choose **OUT9_LED24** as the output macrocell.
4. Choose the type of assignment. Since OUT9_LED24 was assigned as combinatorial in the CPLD Digital Outputs window, choose this type of assignment.

   Note: Updating the type of assignment in the equations window will automatically update the CPLD Digital Outputs selection.

5. Add the Boolean expression. For this simple design, create the expression **IN4_FPGA_T5_OUT**.
6. Add the comment **Turn LED24 on based on signal from FPGA**. The Supervisory Logic Equation Entry should now match Figure 15.

Figure 15. Supervisory Logic Equation Entry



7. Click **OK**. EQ 0 will now display the OUT9_LED24 assignment.

The completed design should look similar to the screen capture in Figure 16. If you chose to import the Logibuilder from the LPTM10_TN1259.PAC file, your design will include additional supervisory equations. These are added automatically during the FPGA Logibuilder entry.

Figure 16. CPLD LogiBuilder Screen Capture

Step	Sequencer Instruction	Outputs	Interruptible	Comment
SM0 Step 0	Wait for AGOOD	OUT6_LED21 = 1, OUT7_LED22 = 1, OUT8_LED23 = 1, OUT13_toFGPA = 1,	yes	Reset all Outputs.
SM0 Step 1	If VM9_VOK Then Goto 6 Else Goto 2		no	If R19 slider pot is middle position, step 6. Else branc
SM0 Step 2	NOP		no	NOP step needed
SM0 Step 3	Wait for 245.76ms using timer 1	OUT6_LED21 = 1,	yes	Blink LED21 and I
SM0 Step 4	Wait for 245.76ms using timer 1	OUT6_LED21 = 0,	yes	Blink LED21
SM0 Step 5	Go to step 1		no	Branch back to St
SM0 Step 6	Wait for VM8_UV_1V	OUT7_LED22 = 0,	yes	Turn LED22 On. V
SM0 Step 7	Wait for NOT VM9_VOK OR NOT VM8_UV_1V OR VM8_OV_2V	OUT8_LED23 = 0, OUT13_toFGPA = 0,	yes	Turn LED23 On. V
SM0 Step 8	Wait for NOT VM8_UV_1V AND NOT VM9UV_1V	OUT6_LED21 = 1, OUT7_LED22 = 1, OUT8_LED23 = 1, OUT13_toFGPA = 1,	no	Wait for both slid
SM0 Step 9	Go to step 0		no	Restart the Seque
SM0 Step 10	Begin Shutdown Sequence		no	
SM0 Step 11	Halt (end-of-program)		no	

Exception ID	Boolean Expression	Outputs	Exception Handler	Comment
E 0	If NOT IN2_RESET_PB <end-of-exception-tabl...	<no outputs s...	Starts at step 0	Reset All

SM0

Equation	Supervisory Logic Equation	Macrocell Configuration	Comment
EQ 0	OUT9_LED24 = IN4_FPGA_T5_OUT <end-of-supervisory-logic-table>	Output, combinatorial [non-regis...	Turn LED24 On based on signal fr...

Task 8: Additional Resources – Power Management Section

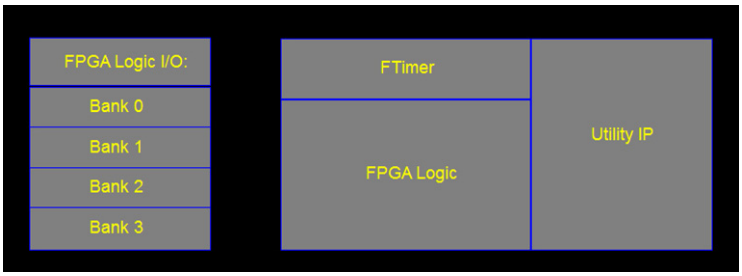
This concludes the section on the Power Management section of Platform Manager. The Power Management section also supports features like margin/trim and high voltage FET drivers. Please refer to the following documents for additional information regarding these features.

- AN6074, Interfacing the Trim Output of Power Manager II Devices to DC-DC Converters
- AN6048, Using Power MOSFETs with Power Manager Devices

Part 2: Platform Manager FPGA Design in PAC-Designer

This section of the technical note will take you through the design flow for implementing the FPGA configuration inside the PAC-Designer tool set. This is the default setting for a new Platform Manager design inside PAC-Designer. To get back to the schematic view use the menu item **Window** and select the schematic. The bottom portion of the PAC-Designer schematic view is used to configure the FPGA. This is shown in Figure 17.

Figure 17. FPGA Portion of Schematic View



There are four blocks in the schematic for the FPGA portion of the Platform Manager: FPGA Logic I/O block, FTimer block, FPGA Logic block, and Utility IP block. We will address each one separately.

Task 1: Configure the FPGA Logic I/O

1. Double click **Bank 0** in the schematic view The Logic I/O Assignment Dialog will appear, with the Bank 0 tab active. The Bank 0-3 tabs configure the following settings:
   - **User-Defined Name** – Signal name passed from I/O to FPGA logic
   - **Direction** – Defined as INOUT, IN, OUT, CLOCK (IN), or RESET (IN)
   - **DRIVE** – Sets the output current to 4, 8, 12, or 14 mA per output (N/A for IN, CLOCK, RESET directions)
   - **Pull Mode** – Configures Pull-Up, Pull-Down or None
   - **Register Type** – Choose between Registered or Combinatorial outputs
   - **Reset Level** – Choose between Don't care (default), Set low, and Set high

   Note: If you imported the .PAC file during the CPLD logic design step earlier, then the I/O and logic in the FPGA side will already be configured according to the steps below.

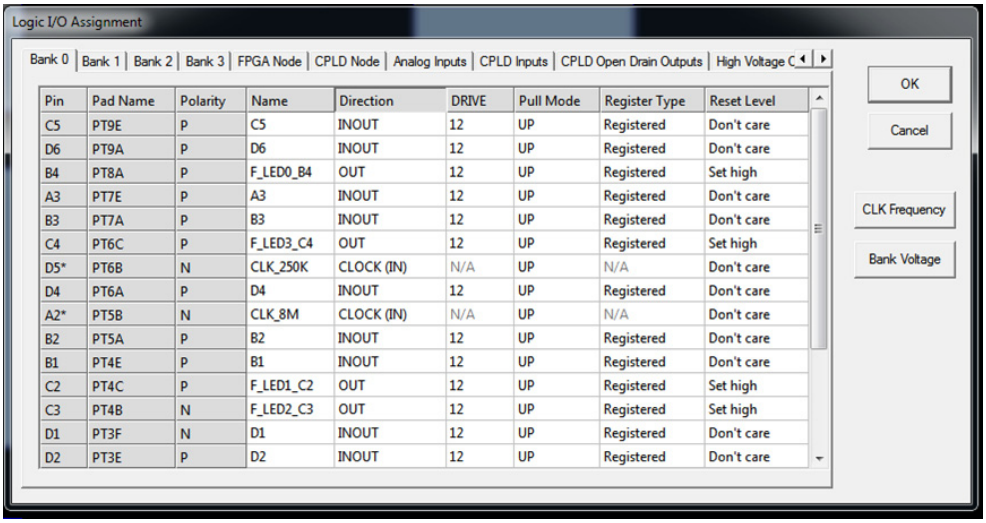
2. The design uses the DIP switches, push buttons, and LEDs on the Platform Manager Evaluation Board to demonstrate the FPGA capability. Configure the I/O in Bank 0 for the design according to the settings in Table 4.

Table 4. FPGA Bank 0 Assignments

Pin	Name	Direction	DRIVE	Pull Mode	Register Type	Reset Level
B4	F_LED0_B4	OUT	12	UP	Registered	Set High
C4	F_LED3_C4	OUT	12	UP	Registered	Set High
D5	CLK_250K	CLOCK (IN)	N/A	UP	N/A	Don't Care
A2	CLK_8M	CLOCK (IN)	N/A	UP	N/A	Don't Care
C2	F_LED1_C2	OUT	12	UP	Registered	Set High
C3	F_LED2_C3	OUT	12	UP	Registered	Set High
F4*	RESET_PB_F4	RESET (IN)	N/A	UP	N/A	Don't Care

The Bank 0 assignments should match Figure 18.

Figure 18. Bank 0 Setting Configuration



Pin	Pad Name	Polarity	Name	Direction	DRIVE	Pull Mode	Register Type	Reset Level
C5	PT9E	P	C5	INOUT	12	UP	Registered	Don't care
D6	PT9A	P	D6	INOUT	12	UP	Registered	Don't care
B4	PT8A	P	F_LED0_B4	OUT	12	UP	Registered	Set high
A3	PT7E	P	A3	INOUT	12	UP	Registered	Don't care
B3	PT7A	P	B3	INOUT	12	UP	Registered	Don't care
C4	PT6C	P	F_LED3_C4	OUT	12	UP	Registered	Set high
D5*	PT6B	N	CLK_250K	CLOCK (IN)	N/A	UP	N/A	Don't care
D4	PT6A	P	D4	INOUT	12	UP	Registered	Don't care
A2*	PT5B	N	CLK_8M	CLOCK (IN)	N/A	UP	N/A	Don't care
B2	PT5A	P	B2	INOUT	12	UP	Registered	Don't care
B1	PT4E	P	B1	INOUT	12	UP	Registered	Don't care
C2	PT4C	P	F_LED1_C2	OUT	12	UP	Registered	Set high
C3	PT4B	N	F_LED2_C3	OUT	12	UP	Registered	Set high
D1	PT3F	N	D1	INOUT	12	UP	Registered	Don't care
D2	PT3E	P	D2	INOUT	12	UP	Registered	Don't care

3. Select the **Bank 1** tab to configure the push-buttons and DIP switches according to the settings in Table 5.

Table 5. FPGA Bank 1 Assignments

Pin	Name	Direction	Drive	Pull Mode	Register Type	Reset Level
B9	A_SW3_B9	IN	N/A	UP	N/A	Don't Care
B8	A_SW2_B8	IN	N/A	UP	N/A	Don't Care
B7	A_SW1_B7	IN	N/A	UP	N/A	Don't Care
C7	PB_A_SW_C7	IN	N/A	UP	N/A	Don't Care
C6	A_SW4_C6	IN	N/A	UP	N/A	Don't Care
B6	PB_B_SW_B6	IN	N/A	UP	N/A	Don't Care

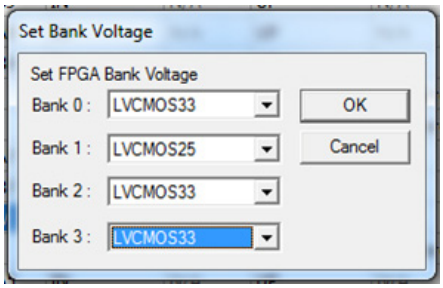
4. The design uses Bank 2 to communicate between the FPGA and CPLD. Set up the pins in Bank 2 according to the settings in Table 6.

Table 6. FPGA Bank 2 Assignments

Pin	Name	Direction	Drive	Pull Mode	Register Type	Reset Level
T6	CPLD_IN4_T6	OUT	12	UP	Registered	Set Low
T9	CPLD_OUT13_T9	IN	N/A	UP	N/A	Don't Care
R9	CPLD_OUT16_TIMER_R9	IN	N/A	UP	N/A	Don't Care

5. After setting up the Bank 0, 1, and 2 properties according to the tables above, click the **Bank Voltage** button found on the right side of the window. The Set Bank Voltage Dialog will open. Set Bank 0, 2, and 3 to **LVCMOS33**, while leaving Bank 1 at **LVCMOS25** (this matches the setup on the Platform Manager Development Kit). The setting should match Figure 19.

Figure 19. Bank Voltage Setting



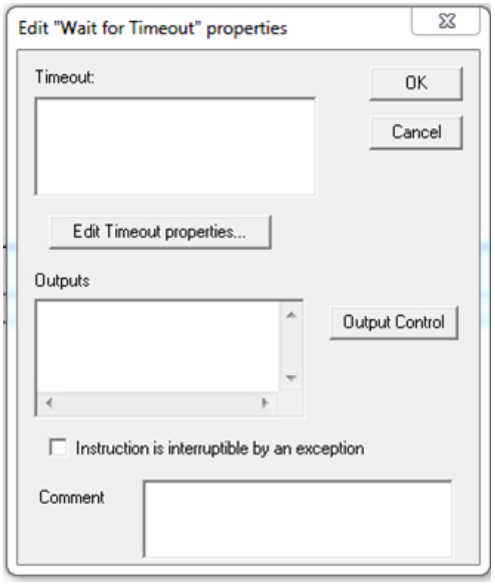
Click **OK** in the Bank Dialog, and then click **OK** again to complete the Logic I/O assignment phase. (Notice the RESET warning message you received earlier in the design is now gone, the FPGA reset has been assigned in this step).

Task 2: Add a Timer to the FPGA Section

You can add a timer to the FPGA section from either from the top level schematic view, by double-clicking **FTimer**, or inside the FPGA Logic view. The FPGA Logic view is used in the steps that follow.

1. Double-click the **FPGA Logic Block** (shown in Figure 17) to enter the Sequence and Supervisory Logic (FPGA) View. This view is used to implement sequences and supervisory logic inside the FPGA section of the Platform Manager device. This view can directly access signals for the FPGA I/O section.
2. Add a new step in the sequence by double-clicking **Step 1**. Choose **Wait for <timeout value>**. The “Wait for Timeout” step will be inserted into the sequence. Double-click on **Step 1** and the Edit “Wait for Timeout” Properties Dialog will be displayed, with the timeout choices empty, as shown in Figure 20.

Figure 20. Wait for Timeout Dialog



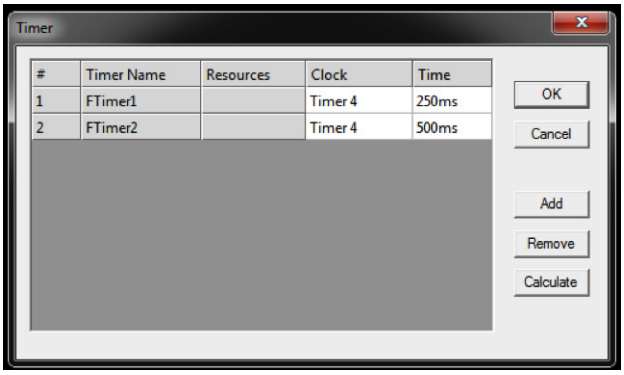
3. Click the **Edit Timeout Properties** button to add a new timer. The Timer Dialog will open. Choose the **Add** button on the right side to add a new timer.
4. FTimer1 will appear in the Timer list, with Clock source Timer4. This is the CPLD Timer4 output. You reserved OUT16 from the CPLD to make this connection to the FPGA for a timer source. It is also possible to source the timer clock from MCLK (8 MHz clock output from the CPLD) or PCLK (250 kHz clock out-

put from the CPLD). You can choose a time for the timeout in this Dialog as well. For the example, enter **250ms** into the box.

5. Add a second timer and enter **500ms** for the timeout.
6. Click **Calculate** for feedback on the number of FPGA LUTs required to generate these timers.

   The Timer Dialog should now look like Figure 21. Click **OK** to complete adding this timer and **OK** again to assign FTimer1 to Step 1. Adding these timers based on the Timer 4 clock will automatically insert two supervisory equations in the CPLD Logic. These equations handle the mapping of the timer to the OUT16 pin.

Figure 21. Timer Dialog



Task 3: Configure the FPGA Logic Inside PAC-Designer

Setting up the logic inside the FPGA in this view is identical to the steps covered earlier in the technical note in the CPLD Logic Design section (aside from the timer usage, which was covered in the previous task).

Note: If you imported the LogiBuilder code in the CPLD Logic design, the FPGA sequence and supervisory equations will already be implemented according to the listing below.

For this design, implement the sequence and supervisory logic shown in the listing below. Alternatively, you can import the code by selecting **File > Import Logibuilder** and browsing to the **LPTM10_TN1259.PAC** file.

Listing 2. FPGA Logic

```
//FPGA Logic reset
Step 0    Begin Startup Sequence

// Blink LED0 Off, waiting for CPLD to finish sequence
Step 1    Wait for 250ms using FTimer1 : output control F_LED0_B4=1

// Blink LED0 On
Step 2    Wait for 250ms using FTimer1 : output control F_LED0_B4=0

// If CPLD has finished sequence, jump to Step 4. Else branch back to step 0
Step 3    If NOT CPLD_OUT13_T9 Then Goto 4 Else Goto 0

// Delay 500ms before turn On LED1
Step 4    Wait for 500ms using FTimer2

// Turn On LED1 and delay 500ms
Step 5    Wait for 500ms using FTimer2 : output control F_LED1_C2=0

// Turn On LED2 and delay 500ms
Step 6    Wait for 500ms using FTimer2 : output control F_LED2_C3=0
```

// Turn On LED3, and Wait for PB_A_SW to be pushed or CPLD Reset
Step 7 Wait for NOT PB_A_SW_C7 OR CPLD_OUT13_T9 : output control F_LED3_C4=0

// Return to start and shut off all LEDs
Step 8 Go to step 0 : output control F_LED0_B4=1, F_LED3_C4=1, F_LED1_C2=1, F_LED3_C3=1

Step 9 Begin Shutdown Sequence

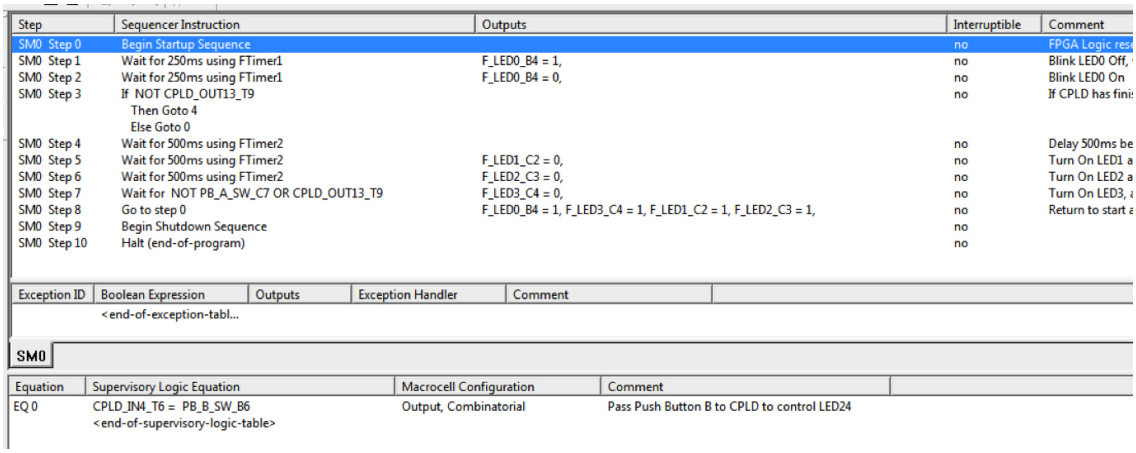
Step 10 Halt (end-of-program)

Supervisory Logic
// Pass Push Button B to CPLD to control LED24

Equation 0: CPLD_IN4_T6 = PB_B_SW_B6

When the design is entered correctly, it should look similar to the screen capture in Figure 22.

Figure 22. FPGA LogiBuilder Screen Capture



Step	Sequencer Instruction	Outputs	Interruptible	Comment
SM0 Step 0	Begin Startup Sequence		no	FPGA Logic res
SM0 Step 1	Wait for 250ms using FTimer1	F_LED0_B4 = 1,	no	Blink LED0 Off,
SM0 Step 2	Wait for 250ms using FTimer1	F_LED0_B4 = 0,	no	Blink LED0 On
SM0 Step 3	If NOT CPLD_OUT13_T9 Then Goto 4 Else Goto 0		no	If CPLD has finis
SM0 Step 4	Wait for 500ms using FTimer2		no	Delay 500ms be
SM0 Step 5	Wait for 500ms using FTimer2	F_LED1_C2 = 0,	no	Turn On LED1 a
SM0 Step 6	Wait for 500ms using FTimer2	F_LED2_C3 = 0,	no	Turn On LED2 a
SM0 Step 7	Wait for NOT PB_A_SW_C7 OR CPLD_OUT13_T9	F_LED3_C4 = 0,	no	Turn On LED3, a
SM0 Step 8	Go to step 0	F_LED0_B4 = 1, F_LED3_C4 = 1, F_LED1_C2 = 1, F_LED2_C3 = 1,	no	Return to start a
SM0 Step 9	Begin Shutdown Sequence		no	
SM0 Step 10	Halt (end-of-program)		no	

Exception ID	Boolean Expression	Outputs	Exception Handler	Comment
	<end-of-exception-tabl...			

SM0

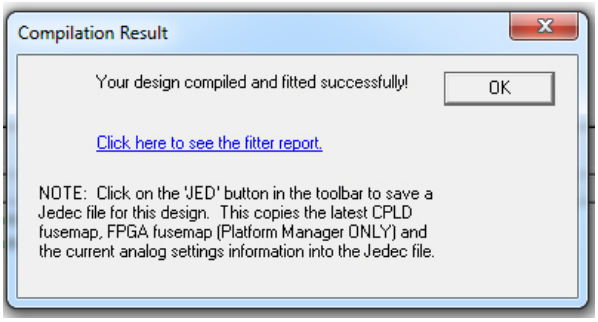
Equation	Supervisory Logic Equation	Macrocell Configuration	Comment
EQ 0	CPLD_IN4_T6 = PB_B_SW_B6 <end-of-supervisory-logic-table>	Output, Combinatorial	Pass Push Button B to CPLD to control LED24

Task 4: Compile the Combined CPLD/FPGA Design Inside PAC-Designer

Now that you have configured the analog, CPLD, and FPGA sections of Platform Manager, you can compile your design. This step can be accomplished from inside either the FPGA or CPLD Logic Design windows.

1. Select the **Tools > Compile LogiBuilder Design** menu option, or the  Compile Design tool.
2. The Compilation will launch a command prompt window and automatically execute a series of processes for both the CPLD and FPGA designs.
3. A properly entered design will result in a successful compilation and Figure 23 will display.

Figure 23. Compilation Result



You can find more information about your design, including the number of macrocells utilized in the CPLD, number of LUTs utilized in the FPGA, and more, by clicking the **Click here to see the fitter report** link.

4. Finally, click the **Export JEDEC File** tool at the top of the PAC-Designer window, indicated by the 

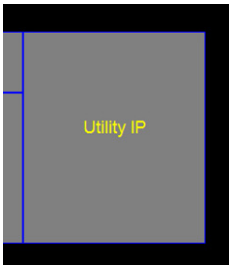
icon. This saves a .JED file in the project folder for programming later.
5. Optional: If you have a Platform Manager Evaluation Board, you can jump forward to "Part 4: Operation on the Platform Manager Evaluation Board (Optional)" on page 27 to verify the design. After the design is verified, jump back to the next section to learn more about Platform Manager.

Additional Resources for Platform Manager FPGA Design with PAC-Designer

Utility IP

The FPGA schematic section of Platform Manager in PAC-Designer includes a block called Utility IP, shown in Figure 24. The block provides an interface to specially developed Platform Manager IP for functions like Fault Logging, VID, and Closed Loop Trimming via I^2C. These are not used in this design, but the short description below will help in using them in other designs.

Figure 24. Utility IP Block



Double-clicking this block brings up the Utility IP configuration interface for these functions if the IP has been installed. For instructions on installing the Utility IP, see "Appendix A. Installing the Platform Manager Utility IP with IPexpress" on page 28. You can enable the IP for use in your design and launch the Lattice IP Core utility to configure the IP. For more details on working with these functions, see IPUG94, Platform Management Utility Functions IP Core User's Guide.

After configuring the functions using the Utility IP interface, you will need to assign the module ports. You will need to double-click the FPGA logic block and select **File > Import Sub-Module > Configure Sub-Module**. From this menu you can assign the signals from the IP to external ports of the Platform Manager FPGA or to internal nodes.

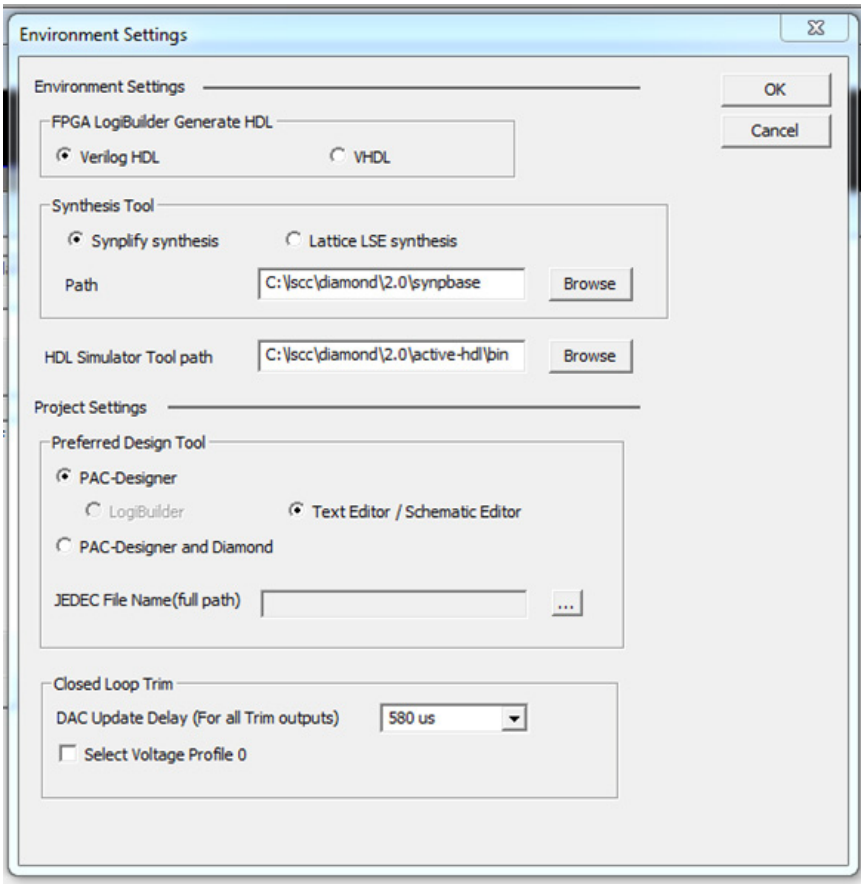
Changing the Preferred Design Tools

PAC-Designer supports working with two additional design flows for the FPGA portion of Platform Manager. To switch to these flows, select **Options > FPGA Environment** from the menu in the top-level schematic view in PAC-Designer.

The Environment Settings Dialog will open, shown in Figure 25. The middle section of the Dialog, titled Project Settings, allows you to choose your preferred design tool flow. Choices include:

- **PAC-Designer: LogiBuilder** – This is the default flow, already covered in detail in this technical note
- **PAC-Designer: Text Editor / Schematic Editor** – This flow is for users who want to edit their HDL design in Verilog or VHDL, but still use PAC-Designer to manage their HDL project. This flow is not recommended due to the restrictions of HDL design inside PAC-Designer. Designers who want to use HDL should consider the PAC-Designer and Diamond flow (next).
- **PAC-Designer and Diamond** – In this flow, the CPLD and analog design is completed inside PAC-Designer, but the full FPGA design and project management is completed inside Lattice Diamond software. PAC-Designer simply points to the FPGA JEDEC file and merges the CPLD/Analog JEDEC information with the FPGA JEDEC to complete the design.

Figure 25. FPGA Environment Settings

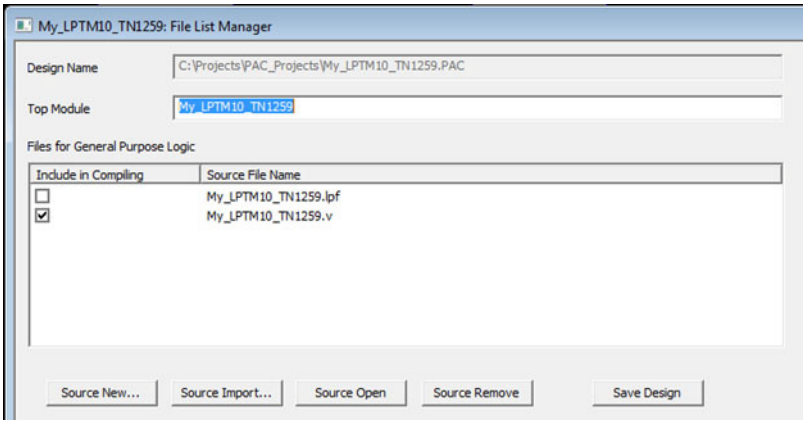


Note: Once you switch to the PAC-Designer: Text Editor/Schematic Editor you CANNOT switch back to the PAC-Designer: LogiBuilder preferred design tool.

Working with the Text Editor Flow

If you choose the text editor /schematic editor flow and choose OK, you will no longer be able to access the FPGA Logic design window, or FPGA I/O configuration window. Instead, when you double-click the FPGA Logic block in the schematic top level, the File List Manager Dialog will open as shown in Figure 26.

Figure 26. File List Manager



The file list allows you to manage your HDL project inside PAC-Designer. When you switch your preferred design tool from LogiBuilder to the Schematic/Text Editor, the list will automatically populate with a .lpf and .v file that correspond to the previously configured I/O and logic design.

Updates to your current project now must be made inside the .lpf file (I/O definitions) and the .v file. The .v (or vhdl, depending on your PAC-Designer settings) includes the generated LogiBuilder sequence and supervisory logic, as well as any configured FTimers. The PAC-Designer generated .v (My_LPTM10_TN1259.v) will include the I/O location information in synthesis directives found in the module definition.

The text editor flow enables the possibility to include additional HDL files, optimize the HDL implementation of the FPGA logic sequence and supervisor, and assign logical functions to the programmable I/O of the FPGA.

Inside the File List Manager view, which now launches when you double-click the FPGA Logic block in the top level of PAC-Designer, you will see the following controls:

- **Design Name**: This is a read-only display of the .PAC project name and path.
- **Top Module**: This is the top module in the HDL project. Normally the top module directly interacts with the external I/O of the FPGA and calls any sub-modules.
- **Files for General Purpose Logic**: This is the file list for the project. It is automatically populated when switching to the text editor flow with a .lpf and .v file that correspond to the previously configured I/O and logic design. Additionally, any imported file or enabled utility IP will also appear in the list. You can add, remove, or edit files in this list. You can also check and uncheck which files are included in the compilation.
- **Source New**: This button allows you to add a new Verilog, VHDL, or a schematic file to the logic project. After choosing the file name and type, the file list will automatically be populated with this new file.
- **Source Import**: This button is used to add an existing Verilog, VHDL, or schematic file to the logic project. After choosing the file name, the file list will automatically be updated to include the new file.
- **Source Open**: This button opens the highlighted source file for editing. Click on the file you would like to edit and choose **Source Open**.
- **Source Remove**: This button removes the highlighted source from the project. Click on the file you want to remove and choose **Source Remove**.
- **Save Design**: This button saves the updated file list (including compilation choices). Compiling the project will also automatically save the design.

Note: The Utility IP block will still be accessible from the PAC-Designer top level in the Schematic/Text Editor flow. However the generated utility IP is not directly compatible with the text editor flow. Thus, working in the Schematic/Text Editor Flow and using the utility IP is not recommended.

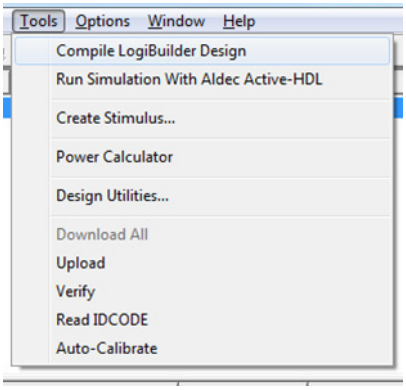
From this view, you can compile your HDL design. Choose **Tools > Compile HDL Design** to build your project.

Simulating Platform Manager from PAC-Designer

For both the PAC-Designer LogiBuilder and PAC-Designer Text Editor / Schematic design tool flows, simulation can be launched out of the PAC-Designer environment directly. In order to simulate the CPLD and FPGA sections of Platform Manager together, you will need to use the Aldec Active-HDL simulator. This simulator is included with the installation of Lattice Diamond software, which is required for designing with Platform Manager.

1. For this design, you can launch the simulator out of the FPGA Logic Design window. The first step is to select the menu **Tools > Create Stimulus**, shown in Figure 27.

Figure 27. Create Stimulus Selection

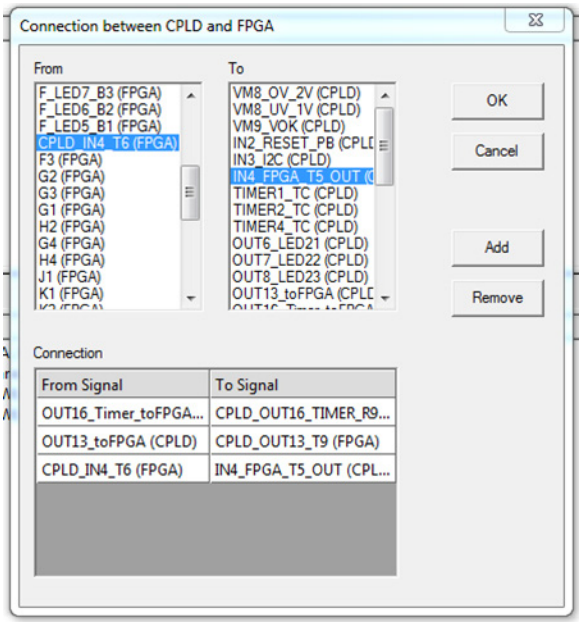


2. Creating the stimulus is accomplished in two steps. First, the connections between the CPLD and FPGA portions of the Platform Manager have to be defined. Then a stimulus template is generated by the software that you can modify. The OUT16 of the CPLD, reserved for the Timer output to the FPGA, is by default connected to the R9 pin of the FPGA. *Note: If using the PAC-Designer – Text Editor / Schematic Editor then this connection is not automatically made.*

   The Connection Between CPLD and FPGA Dialog is used to inform the simulator about connections you will physically place on your PCB between the CPLD and FPGA. Since this design uses the Platform Manager Evaluation Board, you defined a couple of signals which are used to pass information between the CPLD and FPGA logic sequences. Select **OUT13_toFPGA** in the (From) column and **CPLD_OUT13_T9** in the (To) column. Then click **Add** to move this signal down to the connections area.

3. Repeat this process using the **CPLD_IN4_T6** (From) and **IN4_FPGA_T5_OUT** (To). Your connection scheme should match Figure 28. Click **OK** to proceed.

Figure 28. Connection Between CPLD and FPGA Dialog



Click **OK** to proceed. The Simulation Setting Dialog will open populated with an automatically generated test module and test file. From this Dialog, an advanced user could add or remove additional test files. Highlight the **tf_MyPACProject.v** file and choose the **Edit Stimulus** button.

The text editor will open, displaying the test bench and stimulus code (return to PAC-Designer an click **OK** to close the Simulation Settings Dialog). Most of the test file defines the connections, I/O, and default signals like AGOOD and RESET. You can update your own stimulus starting at the **///Add Your Stimulus Code Here** (see Figure 29). This section defines the different input signals to the simulation. These are a mixture of voltage monitor outputs, CPLD digital inputs, and FPGA digital inputs.

Figure 29. Stimulus File - User Test-Bench Location

```
//-----------------------------------------------------
// Add Your Stimulus code here
//-----------------------------------------------------

initial begin
   // By default, Initialize all input signals to '0'
   A_SW3_B9 = 1'b0;
   B_SW3_A9 = 1'b0;
   A_SW2_B8 = 1'b0;
   B_SW2_A7 = 1'b0;
   A_SW1_B7 = 1'b0;
   PB_A_SW_C7 = 1'b0;
   B_SW1_A5 = 1'b0;
   B_SW4_B5 = 1'b0;
   PB_B_SW_B6 = 1'b0;
   A_SW4_C6 = 1'b0;
   CPLD_OUT13_T9 = 1'b0;
   VM8_OV_2V = 1'b0;
   VM8_UV_1V = 1'b0;
   VM9_VOK = 1'b0;
   IN2_RESET_PB = 1'b0;
   IN3_I2C = 1'b0;
   IN4_FPGA_T5_OUT = 1'b0;
   // user test-bench code ...
end
```

Editing and updating the simulation stimulus is outside the scope of this technical note. Experienced HDL designers should be familiar with this process. For information specific to the Lattice CPLD architecture, see AN6044, Simulating Power Supply Sequences for Power Manager Devices Using PAC-Designer LogiBuilder.

Save the stimulus test file and return to PAC-Designer. From the FPGA Logic view, select the menu item **Tools > Run Simulation with Aldec Active-HDL**. The Active-HDL simulator will launch (you may be prompted to overwrite the .work file, choose **OK**). The simulator will run for a default time of 100 ms. At the completion of the simulation, the waveform viewer will display the external I/O signals of the design (including the Voltage Monitor outputs).

The Active-HDL simulator is designed for high-speed logic simulation. Typical Platform Manager designs use long timeout values such as 250ms and 500ms. Simulating designs with such long timer values may take a long time to complete and take up a lot of disk space. Thus, it is often helpful to reduce the timer values in the design only for simulation, just remember to restore the timer values to the real world values before testing on an evaluation board or programming the final board.

Learn More About Simulation

- See the Help File in Aldec Active-HDL, found at **Help > Product Help** or the Verilog tutorial at **Help > Interactive Verilog Tutorial**
- For Lattice Power Manager simulation details, see AN6044, Simulating Power Supply Sequences for Power Manager Devices Using PAC-Designer LogiBuilder

Part 3: Platform Manager FPGA Design with Lattice Diamond

This section of the technical note covers the third design flow option for the Platform Manager – PAC-Designer and Diamond.

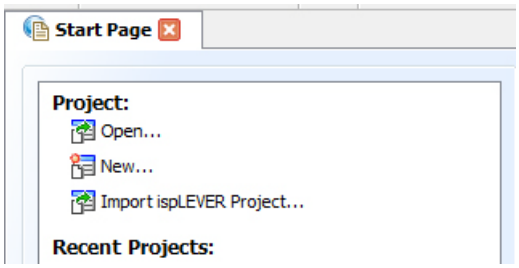
This design flow offloads the entire FPGA design to the Lattice Diamond software. There are many advantages to this flow for an experienced FPGA designer, as Diamond offers many HDL design features not supported inside PAC-Designer. This is the recommended design flow for any complex Platform Manager design.

Task 1: Setting up a Platform Manager Project in Lattice Diamond

Before changing the tool settings in PAC-Designer, you need to create and compile your Lattice Diamond project.

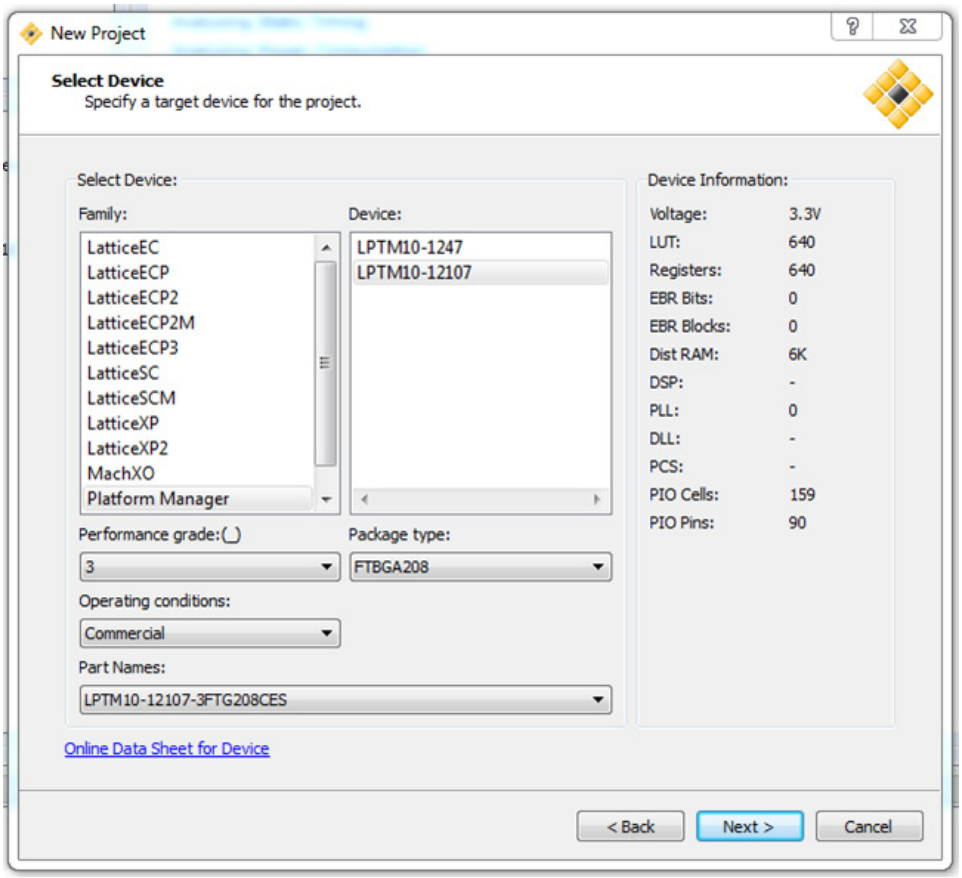
1. Launch Diamond (**Start > Lattice Diamond 2.0 > Lattice Diamond**). You will now be at the start page. Choose **Project: New**, as shown in Figure 30.

Figure 30. Lattice Diamond Start Page



2. Click **Next** to start the new project process. You will be prompted to select a Project Name and Location, along with an implementation.
3. Choose **MyPtMProject** as the Name. Browse to the design folder location and create a new folder called Diamond_Project. Name the implementation **Design1**. Click **Next**.
4. You will be prompted to add source code. For this example, you will use the HDL code generated by the PAC-Designer flow. Choose **Add Source…** and browse to **My_LPTM10_12107_TN1259_FPGA_files** subdirectory in your PAC-Designer project directory. Select **My_LPT10_12107_TN1259.v** and click **OK**. Then click **Next**.
5. Select the Platform Manager device to complete the project creation. Choose **Platform Manager** from the Family list, and **LPTM10-12107** from the Device list, as shown in Figure 31. Leave all the other options as defaults and click **Next**.
6. Select the Synthesis tool for this project. Use the default and click **Next**.

Figure 31. Device Selector – Platform Manager



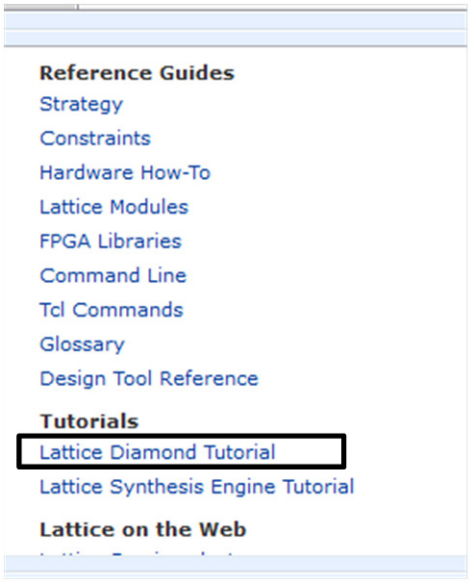
7. The New Project – Project Information screen will be displayed. Click **Finish** to generate your project.

Task 2: Build an FPGA .jed File in Lattice Diamond

The Diamond Design screen will launch showing the default Reports view. There are numerous different views and tools included in Diamond.

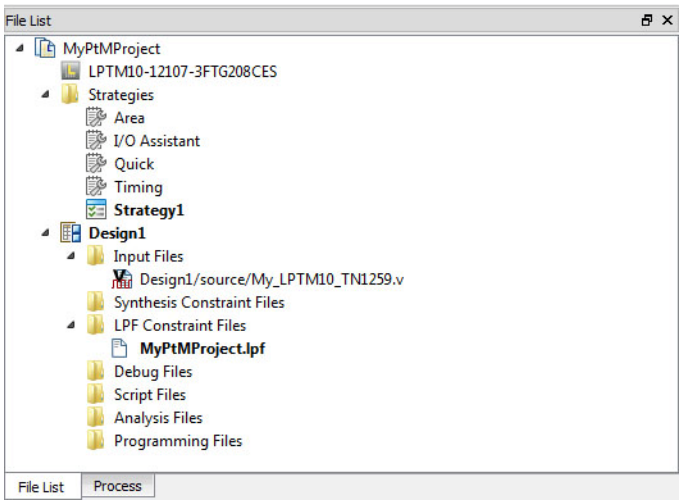
Note: For users new to Lattice Diamond software, the Diamond Tutorial is a good place to start getting to know the basic Diamond flow. The launch shortcut (on the Start Page tab) is shown in Figure 32.

Figure 32. Lattice Diamond Tutorial Shortcut



To complete the Platform Manager design you will generate the FPGA JEDEC file and complete the PAC-Designer / Diamond Design flow. When Diamond opens, the left side window displays the File List as shown in Figure 33.

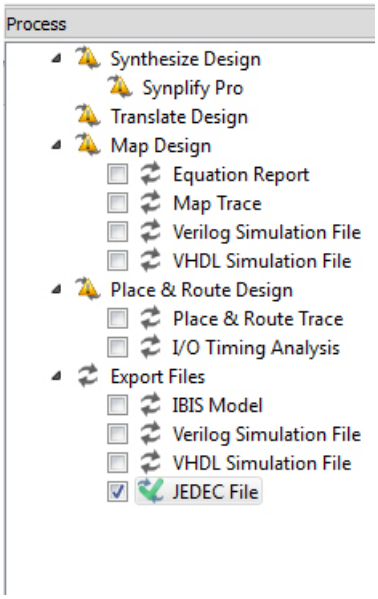
Figure 33. Project File List



The File List includes the .v files added in the Create New Project wizard. A default constraint file (.lpf) is also added to the project.

1. Click the process tab next to the file list tab to display the list of processes available in Lattice Diamond software. These are explained in more detail in the Lattice Diamond Tutorial.
2. In order to complete this design for the FPGA portion of platform manager, check the **JEDEC File** box (see Figure 34). Right-click the JEDEC file and choose **Rerun All** in order to generate the FPGA JEDEC file.

Figure 34. Process Window



The process should execute successfully for this design and generate a MyPtMProject_Design1.jed inside the ../Diamond_Project/Design1 folder.

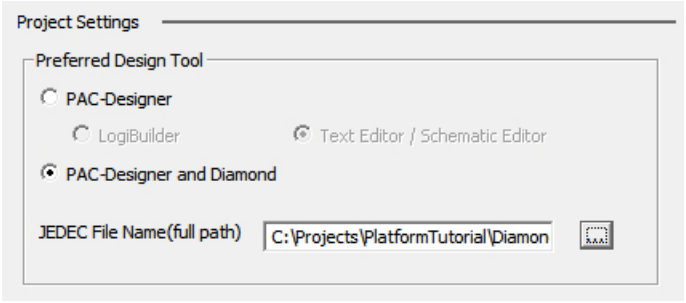
Note: Using the .v file from the PAC-Designer LogiBuilder automatically assigns the I/O pins in the FPGA to the physical pins chosen in PAC-Designer in this Lattice Diamond Project. In a normal project flow, you will need to use either the Spreadsheet View or the .lpf constraint file to locate the physical I/O after the Map Design step in the process. See Lattice Diamond help for more details.

Task 3: Link the PAC-Designer Project to the FPGA .jed

Once the FPGA .JED file is created in Lattice Diamond, return to PAC-Designer. In order to work in the PAC-Designer and Diamond flow, you need to select the menu item **Options > FPGA Environment** to open the Environment Settings Dialog.

1. Choose **PAC-Designer and Diamond** as your preferred tool flow. In the JEDEC File Name (full path) Dialog (see Figure 35), browse to the generated **MyPtMProject_Design1.jed** file (located in the Diamond_Project folder). Then click **OK** to accept the new settings.

Figure 35. Project Settings - Preferred Design Tool



2. In this flow, you can only access the CPLD resources in PAC-Designer. All the FPGA resources are reserved for Diamond. If you have successfully compiled your CPLD logic project, you can click the 

button in PAC-Designer to merge the CPLD and FPGA JEDEC files for programming.

Part 4: Operation on the Platform Manager Evaluation Board (Optional)

If you have access to a Platform Manager Evaluation Board you can use Lattice ispVM™ System software or Diamond Programmer software to program the exported .JED file into the evaluation board and double check the operation.

More details on working with the programming software can be found in AN6062, Using ispVM System to Program ispPAC Devices.

To complete the design, you can download the design onto the Platform Manager Evaluation Board. The board should operate as follows:

0. Before power up: Set both slider pots all the way down (zero volts).
1. Power up
2. LED0 and LED21 are blinking at a 250ms rate.
   (CPLD steps 1-5; FPGA steps 1-3)
3. Move R19 (left) to the middle position. LED22 turns on.
   (CPLD step 6)
4. Move R20 (right) to the middle position. LED23 turns on.
   (CPLD step 7)
5. LED0, LED1, LED2, and LED3 sequence on with a delay of 500ms.
   (CPLD step 7, FPGA steps 4-7)
6. Press PB_A_SW. LED0, LED1, LED2, and LED3. Turn off and re-sequence on with a delay of 500ms.
   (FPGA steps 7, 8, 0-7)
7. Press PB_B_SW. LED24 turns on while switch is held on.
   (CPLD equation 2 and FPGA equation 0)
8. Slide either pot away from the middle. All LEDs turn off and LED0 is blinking.
   (CPLD steps 7 and 8, FPGA steps 7, 8, 0-3)
9. Slide both pots all the way down and repeat steps 2-5.
10. Press S1 (RESET_ALL). All LEDs are turned off while the switch is down.
    (CPLD equation 0 and step 0, FPGA step 0)

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
March 2013	01.0	Initial release.

Appendix A. Installing the Platform Manager Utility IP with IPexpress

1. After IPexpress™ opens, click on the  tool to update the list of IPs from the Lattice server.
2. After the update, scroll down to the Connectivity folder and look for the Platform Manager Utility.
3. Select the latest version and right-click to select **Download Platform Manager Utility**.
4. You may get a warning that the IP is not compatible with the current version of the Diamond software. Click **OK** to continue. Save the installer to a temporary or download folder.
5. Browse to the installer and run it. It should create a LatticeCore™ folder and place the Platform Manager Utility IP inside it when finished.
6. Return to PAC-Designer and save any work and then close it down.
7. Restart PAC-Designer and reopen your design. Click on the IP Utility block and configure as needed.

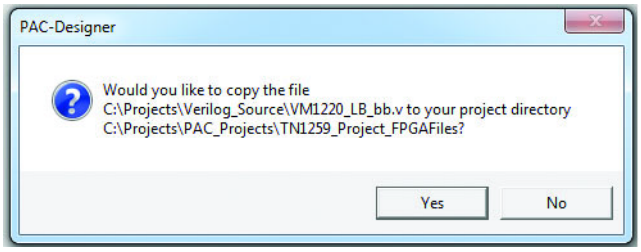
Appendix B: Transferring Platform Manager Designs

Platform Manager designs require special handling if they are to be transferred between users or directories. Most PAC-Designer projects can be transferred simply by sharing the .PAC file, however Platform Manager designs rely on several additional files related to the FPGA design which may not be transferred properly simply by sharing the .PAC file. Users who plan to transfer their designs should follow these guidelines.

Copy Files to Project Directory During Source Import (Text Editor Flow Only)

When performing **Source Import** operation, click **Yes** when asked to copy the files to the project directory, as shown in Figure 36.

Figure 36. Source Import – File Copy Dialog

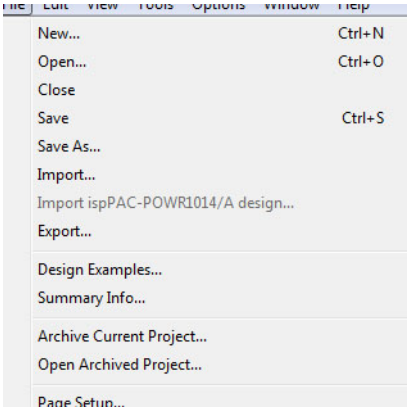


Copying these files to the source directory enables the archive function described below to execute properly.

Use Archive Operations Rather Than Save As or Directory Zip (All Flows)

The **Archive Current Project** and **Open Archived Project** operations in the **File** menu should be used to move and share Platform Manager projects. From the block diagram view, open the **File** menu to access the archive functions, as shown in Figure 37.

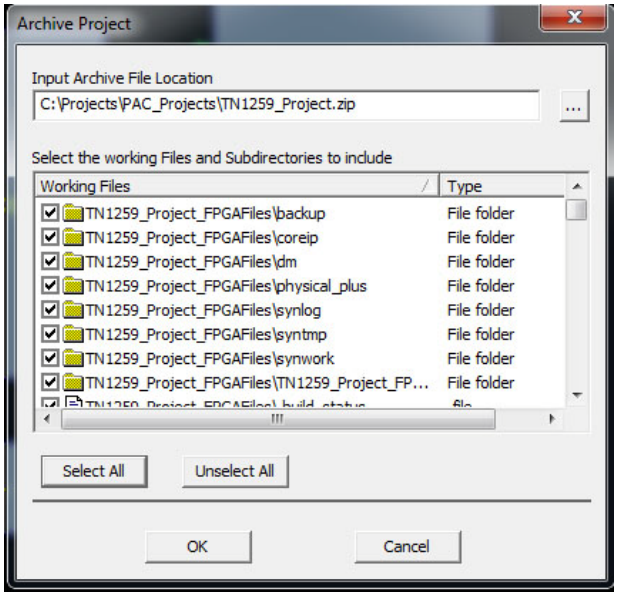
Figure 37. Archive Operations – Menu Location



Archiving the Current Project:
Using the Archive Current Project operation is the simplest way to archive a project.

1. From the **File** menu, select **Archive Current Project**.
2. The **Archive Project** dialog opens, where you can select the folders and files that should be included in the .zip file, as shown in Figure 38.
3. Choose **Select All** to include all the files from the project for the archive.

Figure 38. Archive Project – File Selection Dialog



Once the archive operation is completed, you can share the zip file with another developer. You can also move the zip file to other locations on your PC.

Opening the Archived Project:
Use the **Open Archived Project** operation to work with an archived project from another developer or file location. This is a better practice than unzipping the project in the file explorer, since opening the archived project inside PAC-Designer will re-target the active directories inside the project.

1. From the **File** menu, select **Open Archived Project**
2. The **Open Archived Project** operation will prompt you for a folder location. Indicate the folder where the project will be unzipped.

Once the operation is completed you can continue working on the design.

Transferring PAC-Designer and Diamond Flow Designs:

Working in the PAC-Designer and Diamond flow is a special case. You may still wish to archive the project when transferring, but keep in mind that the entire FPGA project is maintained separately by Diamond and may not be archived.

Anytime you share a PAC-Designer file from this flow with another user, the Diamond Project will need to be archived and shared separately. Additionally, the PAC-Designer file includes a reference to the Diamond generated JEDEC file, set in the **Options > FPGA Environment** menu (see Figure 39). This reference is a full path and will not be updated by using the archive operations. You will need to update it manually before generating the combined JEDEC file in PAC-Designer.

Figure 39. FPGA JEDEC File Name – Reference Entry