Implementing Power Supply Sequencers with Power/Platform Management Devices and PAC-Designer LogiBuilder

October 2011 Application Note AN6042

Introduction

Power/Platform Management devices are a single-chip, fully-integrated solution to supervisory and control problems encountered when implementing on-board power conversion and distribution systems. They provide several types of on-chip resources which can be used to meet the requirements of these applications. Power/Platform Management devices have analog comparators with independently adjustable references, providing the ability to detect when a voltage is greater than or less than a given threshold. They have timers to implement programmable delays and embedded logic to support the implementation of state machines and sequencers, providing embeddable intelligence in a power supply design.

See Tables 1 and 2 for the Power/Platform Management devices applicable to this application note.

Table 1. Applicable Power Manager Devices Summary

Device	VMONs	Digital Inputs	HVOUTs	Open Drain Outputs	Trim DAC Outputs	Macrocells
ProcessorPM™-POWR605	6	2	None	5	None	16
ispPAC®-POWR607	6	2	2	5	None	16
ispPAC-POWR1014/A	10	4	2	12	None	24
ispPAC-POWR1220AT8	12	6	4	16	8	48

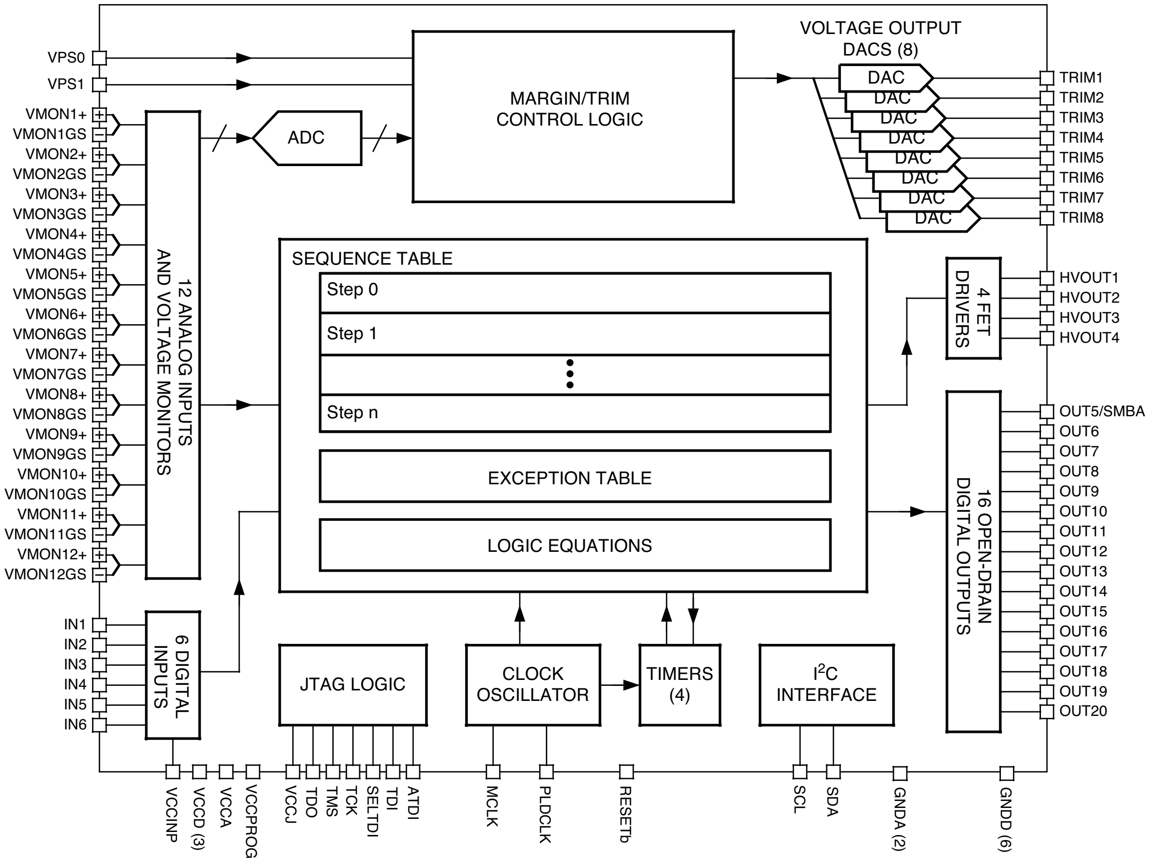
Table 2. Applicable Platform Manager™ Devices Summary

Device	VMONs	Digital Inputs	HVOUTs	Open Drain Outputs	Trim DAC Outputs	Digital I/O	CPLD Macrocells	FPGA LUTs
LPTM10-1247	12	4	4	12	6	31	48	640
LPTM10-12107	12	4	4	12	8	91	48	640

PAC-Designer LogiBuilder

LogiBuilder is a design environment that exists within PAC-Designer® and provides a simple, easy-to-learn method for configuring programmable logic, such as that incorporated in a Power/Platform Management device. LogiBuilder shields the user from the complexity and details of the underlying programmable logic by presenting a higher-level architecture (Figure 1) which the designer manipulates. For building sequence controllers, LogiBuilder provides an instruction set called the LogiBuilder Sequence Controller (LBSC). Instead of worrying about product terms and macrocell configurations, the designer develops configurations as a program containing a short sequence of LBSC instructions, which are sequentially executed. In addition to straight sequences, LBSC supports both looping and conditional execution through GOTO and IF-THEN-ELSE instructions. LBSC program sequences may contain up to 128 individual instructions. WAIT instructions are provided to support both waiting for specified events and user-defined delays implemented using timers.

Figure 1. LogiBuilder Architectural View of a Power/Platform Management Device



In addition to being able to specify a control sequence and delays using the timers, exceptions allow for fast, out-of sequence response to time-critical events. Exceptions will be described in detail later in this document.

LogiBuilder Sequence Controller Instruction Set

LogiBuilder provides the following instructions for designing control sequences:

- **OUTPUT** – The OUTPUT instruction is used to turn on or off the Power/Platform Management device's output signals (HVOUTs and open drain outputs). A single OUTPUT instruction can be used to simultaneously change the status of any number of output signals.
- **WAIT FOR** <Boolean Expression> – The WAIT FOR <Boolean expression> instruction suspends execution of the sequence until the specified expression becomes TRUE.
- **WAIT FOR** <time> **USING TIMER**<1..4> – The WAIT FOR <time> instruction is used to specify a fixed delay in the execution sequence. The value of <time> is determined by which timer (TIMER1...TIMER4) is specified.
- **IF** <Boolean Expression> **THEN GOTO** <step x> **ELSE GOTO** <step y> – The IF-THEN-GOTO instruction provides the ability to modify the sequence flow depending on the state of the inputs. If <Boolean expression> is TRUE, the next step in the sequence will be <step x>, otherwise the next step will be <step y>.
- **GOTO** <step x> – The GOTO instruction forces the sequence to jump to <step x>.
- **NOP** – The NOP instruction does not affect any of the outputs or the sequence of execution. It is effectively a single-cycle delay.
- **HALT** – The HALT instruction stops execution of the sequence. Outputs are held in their current state.

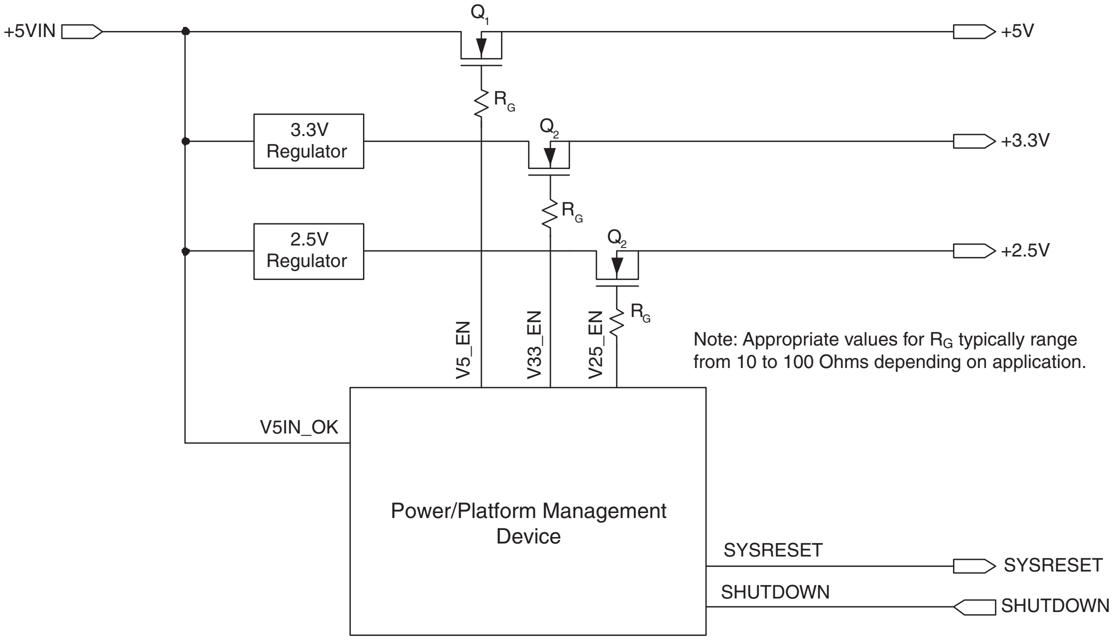
In addition to the above instructions, LogiBuilder also provides two optional 'pseudo-instructions':

- **Begin Startup Sequence** – This instruction marks the beginning of a LogiBuilder sequence. It is optional and may be deleted. It only needs to be present if the first instruction of a sequence would otherwise be a WAIT FOR <time> instruction.
- **Begin Shutdown Sequence** – This instruction marks the beginning of the shutdown section of the sequence, and disables the 'jump' component of exceptions (discussed in more detail later in this document) in the instructions following. If exception jumps are not used, this instruction may also be deleted.

Example Problem

Perhaps the simplest way to show how LogiBuilder is used is to describe an example problem. Figure 2 shows a hypothetical power control system, such as that which might be used on a board with CPLDs and FPGAs requiring multiple supply voltages. In this circuit, the Power/Platform Management device monitors the input supply (+5VIN), and controls the local supplies, as well as interfaces to on-board logic through SYSRESET and SHUTDOWN signals.

Figure 2. Power/Platform Management Device Controlling Three Power Supplies



Note: Appropriate values for R_G typically range from 10 to 100 Ohms depending on application.

In this system, +5V is supplied to the board, and two separate regulators are also powered by this voltage to provide independent 3.3V and 2.5V supplies. To power this system up, the 3.3V supply must be brought up first, the 2.5V supply next, and finally the 5V supply. The system should pause ~10msec between bringing each supply up to allow for stabilization of circuitry on the board. Additionally, SYSRESET must be held LOW until all supplies have been activated. When all the supplies are activated, SYSRESET is brought high. When the SHUTDOWN line is brought HIGH, the power supplies must be turned off in the reverse order: 5V, 2.5V, and finally 3.3V.

This procedure can be expressed as a sequence of steps:

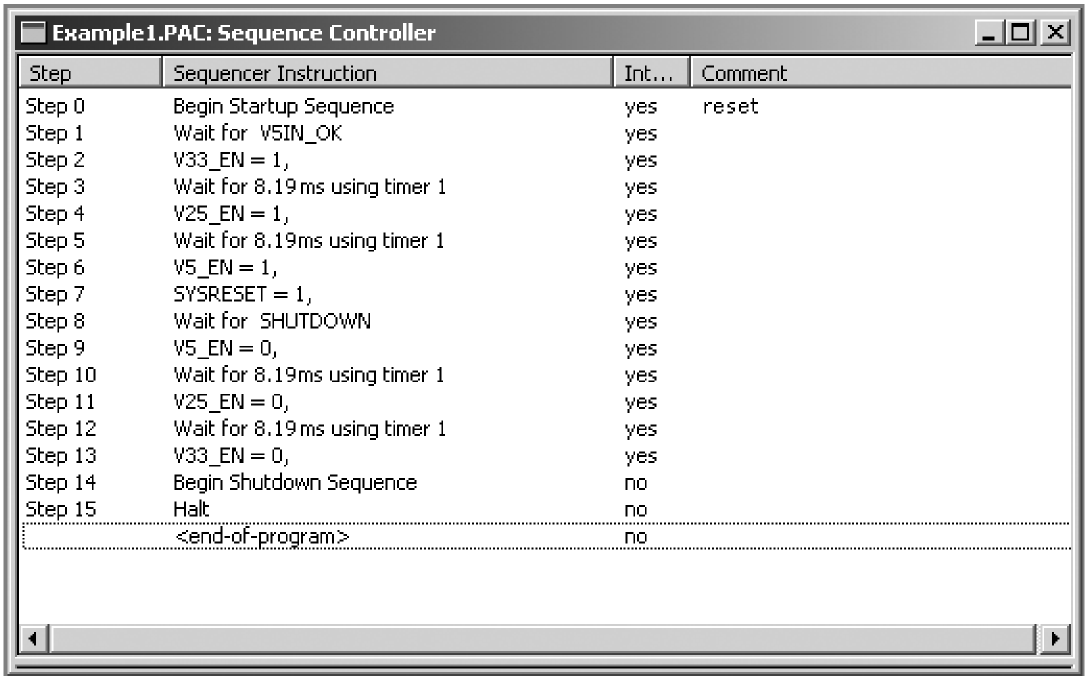
1. Wait for +5V to rise.
2. Activate the 3.3V supply.
3. Wait ~10msec.
4. Activate the 2.5V supply.

5. Wait ~10msec.
6. Activate the local +5V supply.
7. Bring SYSRESET high.
8. Wait for SHUTDOWN.
9. Turn off the +5V supply.
10. Wait ~10 msec.
11. Turn off the +2.5V supply.
12. Wait ~10 msec.
13. Turn off the 3.3V supply.
14. HALT.

A First Sequence Controller Program

The sequence developed in the last section can be directly translated into the LBSC sequence shown in Figure 3. Note that user-defined names (e.g. V33_EN) are used to describe pins instead of the default data sheet pin names. Pin names can be renamed in a design's top-level schematic view. Renaming pins with easily-recognizable mnemonics can both simplify the process of developing a LogiBuilder program, as well as provide concordance between documentation generated by LogiBuilder and signal names in higher-level schematics in which the Power/Platform Management device appears.

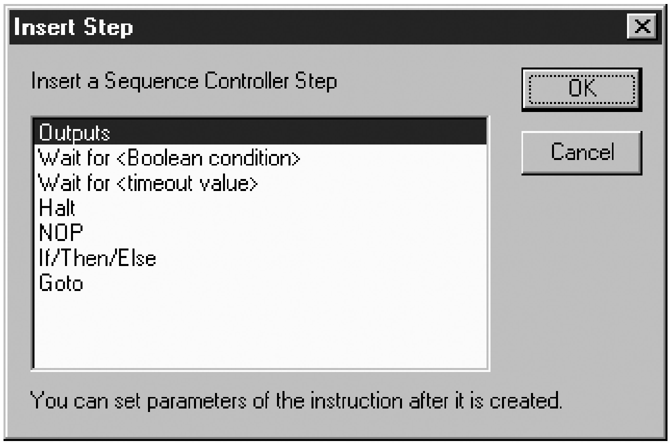
Figure 3. LogiBuilder Sequence Table Showing Implementation of Example



Step	Sequencer Instruction	Int...	Comment
Step 0	Begin Startup Sequence	yes	reset
Step 1	Wait for V5IN_OK	yes	
Step 2	V33_EN = 1,	yes	
Step 3	Wait for 8.19 ms using timer 1	yes	
Step 4	V25_EN = 1,	yes	
Step 5	Wait for 8.19ms using timer 1	yes	
Step 6	V5_EN = 1,	yes	
Step 7	SYSRESET = 1,	yes	
Step 8	Wait for SHUTDOWN	yes	
Step 9	V5_EN = 0,	yes	
Step 10	Wait for 8.19ms using timer 1	yes	
Step 11	V25_EN = 0,	yes	
Step 12	Wait for 8.19 ms using timer 1	yes	
Step 13	V33_EN = 0,	yes	
Step 14	Begin Shutdown Sequence	no	
Step 15	Halt	no	
	<end-of-program>	no	

Because LogiBuilder is dialog-driven, as opposed to a free-form text language, it is easy to use, and does not require learning a large number of syntactic details, as the various dialogs guide you in supplying necessary information. Individual steps in a LogiBuilder sequence can be added by selecting the insertion point with the mouse cursor and pressing the **INSERT** key. An Insert Step dialog (Figure 4) will appear, prompting you for the particular instruction you wish to add.

Figure 4. LogiBuilder 'Insert Step' Dialog



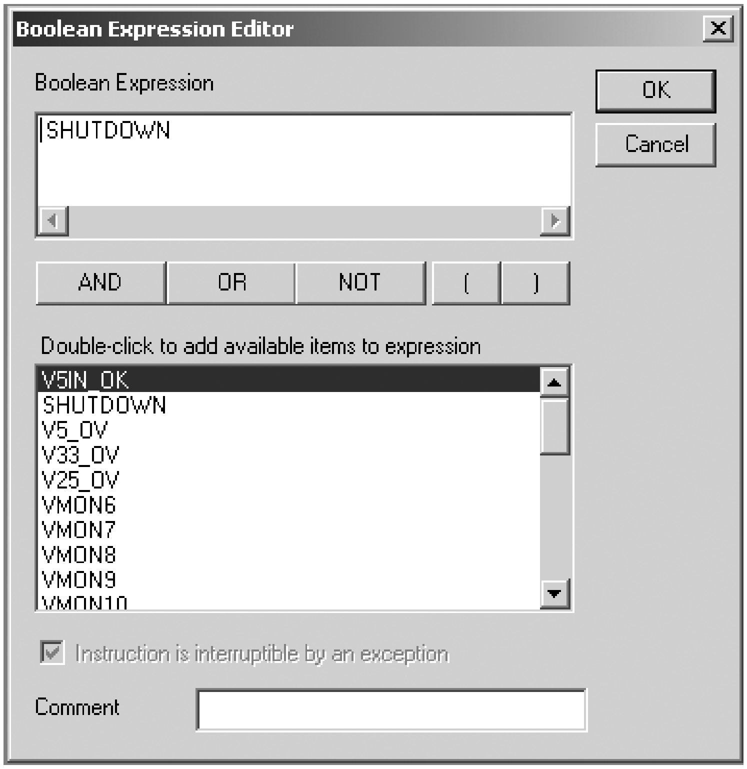
Insert a Sequence Controller Step

You can set parameters of the instruction after it is created.

Selecting the desired instruction type and clicking **OK** will insert a blank template for the selected instruction into the sequence.

To specify the details for a given instruction, simply double-click on that instruction in the Sequence Table. This will open the an appropriate dialog for editing for that particular instruction. As an example, clicking on step 7 (WAIT FOR SHUTDOWN) will invoke the Boolean Expression Editor dialog box (Figure 5).

Figure 5. Boolean Expression Editor Dialog for Editing WAIT FOR Instruction



Boolean Expression: SHUTDOWN

Double-click to add available items to expression

☑ Instruction is interruptible by an exception

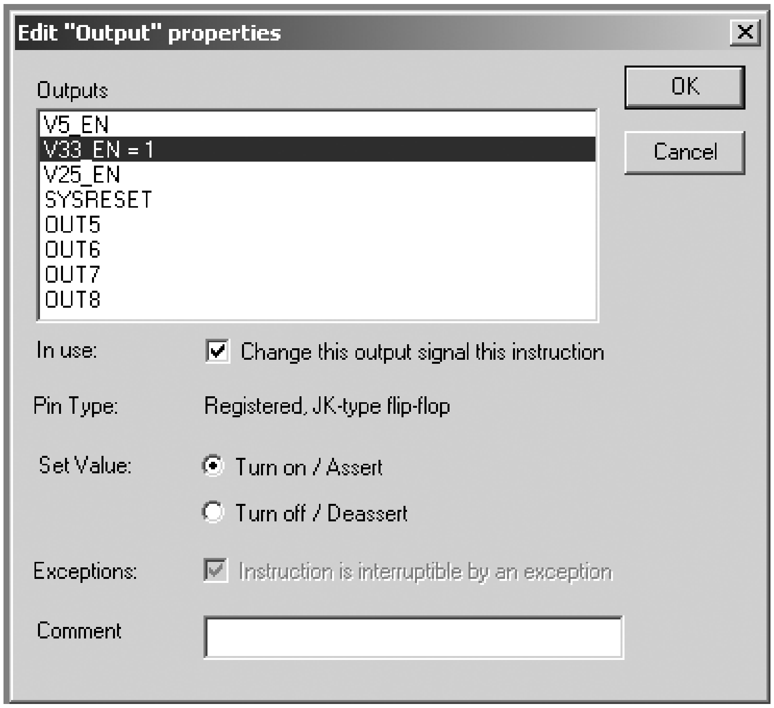
Comment:

This dialog box allows the user to specify a Boolean expression for which the sequence must wait to become true before it continues. An expression may be constructed either by typing it in free-form in the top editing field, or may be built by selecting listed variables (by double-clicking on the desired variable) and operators (AND, OR, NOT, '(', ')'). This dialog also provides an editable field for adding comments to the LogiBuilder sequence.

In addition to being used for defining parameters for the ‘WAIT FOR <Boolean>’ instruction, the Boolean expression editor dialog is also used for specifying Boolean conditions in other parts of LogiBuilder. In these cases the comment field and exception enable checkbox are disabled.

Double-clicking on an OUTPUT instruction will bring up the Edit Output Properties dialog (Figure 6). This dialog allows you to turn individual outputs on or off. In this dialog, you can select which outputs are to be changed in any given step (by checking the check-box), and if a particular output is to be modified, whether it should be turned on or off. Note that any outputs not specifically selected to be modified in a given OUTPUT instruction will remain unaffected by the execution of that instruction. Note that this dialog also provides a field for entering a comment.

Figure 6. Edit Outputs Dialog



Outputs

V5_EN
V33_EN = 1
V25_EN
SYSRESET
OUT5
OUT6
OUT7
OUT8

In use: ☑ Change this output signal this instruction

Pin Type: Registered, JK-type flip-flop

Set Value: ● Turn on / Assert
○ Turn off / Deassert

Exceptions: ☑ Instruction is interruptible by an exception

Comment

Using Exceptions to Interrupt a Sequence

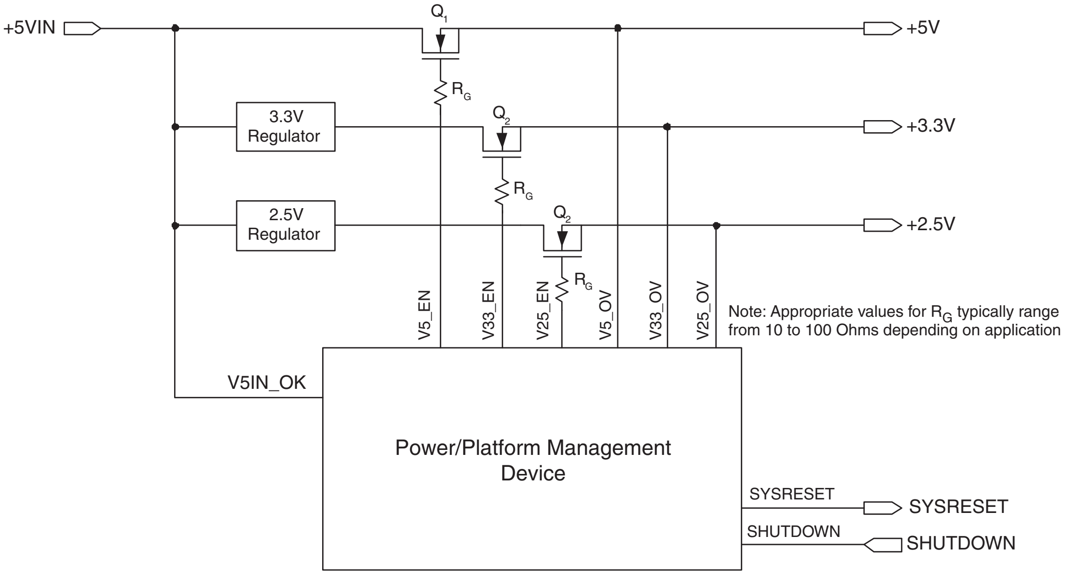
In the example above, the sequence was strictly linear, proceeding from start to finish, only stopping to wait for specified intervals or until specified events occurred. The IF-THEN-ELSE instruction can be used to test for a specific condition and redirect the execution of a sequence based on the results of that test.

There are frequently cases, however, when you want to be able to test for a given set of conditions on an essentially continuous basis, and take some action immediately when the conditions are satisfied. One example of such a situation would be an emergency shutdown in the case of over-voltage. If one of the power supply voltages exceeds a certain threshold, quickly shutting down all the derived supplies may protect the board from damage.

While one could perform this function by inserting frequent tests at critical points in the sequence, this approach has two drawbacks. First, because the instructions to perform the necessary tests need to be interspersed with those that perform the main task at hand, response time may be slow, particularly in the case in which the over-voltage condition occurs during a WAIT instruction. The second drawback is that adding a lot of extra instructions can increase the complexity of a sequence, and may make it more difficult to understand and debug.

For these reasons, LogiBuilder supports a capability called an exception. Exceptions provide the ability for a LogiBuilder program to quickly respond to conditions that are not specified in the main sequence. In the over-voltage detection example cited above, the circuit must be modified to permit sensing of the output voltages, as shown in Figure 7. Over-voltage conditions are sensed through the V5_OV, V33_OV, and V25_OV lines.

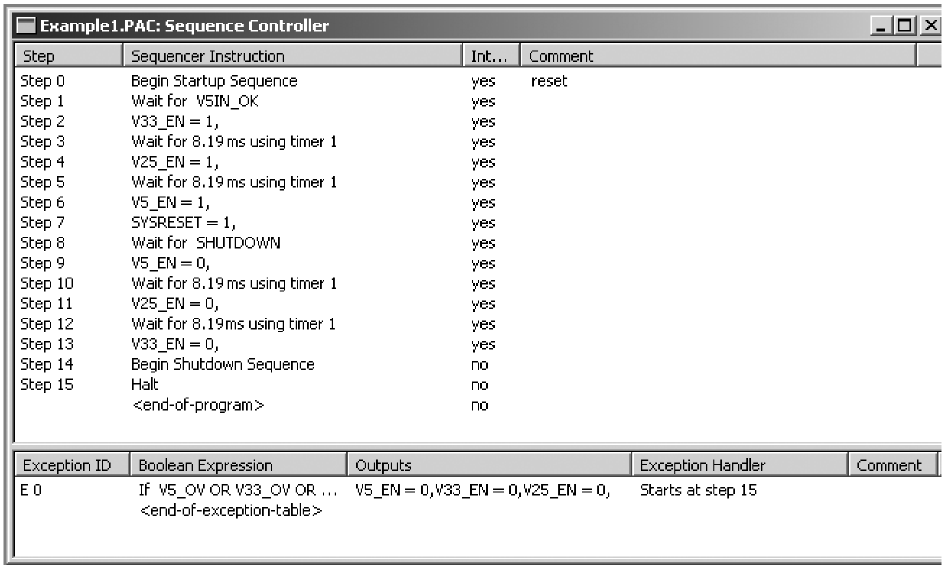
Figure 7. Modifications to Provide Over-voltage Detection



Note: Appropriate values for R_G typically range from 10 to 100 Ohms depending on application

If an over-voltage condition is seen, the specified action will be to turn everything off fast and HALT. This can be accomplished with minimal changes to the sequence by specifying an exception. Exceptions are listed in LogiBuilder in the exception table (Figure 8), which is displayed immediately beneath the sequence table. Each exception occupies one row in the exception table.

Figure 8. Sequence and Exception Tables for Over-voltage Shutdown Example



Step	Sequencer Instruction	Int...	Comment
Step 0	Begin Startup Sequence	yes	reset
Step 1	Wait for V5IN_OK	yes	
Step 2	V33_EN = 1,	yes	
Step 3	Wait for 8.19 ms using timer 1	yes	
Step 4	V25_EN = 1,	yes	
Step 5	Wait for 8.19 ms using timer 1	yes	
Step 6	V5_EN = 1,	yes	
Step 7	SYSRESET = 1,	yes	
Step 8	Wait for SHUTDOWN	yes	
Step 9	V5_EN = 0,	yes	
Step 10	Wait for 8.19 ms using timer 1	yes	
Step 11	V25_EN = 0,	yes	
Step 12	Wait for 8.19ms using timer 1	yes	
Step 13	V33_EN = 0,	yes	
Step 14	Begin Shutdown Sequence	no	
Step 15	Halt	no	
	<end-of-program>	no	

Exception ID	Boolean Expression	Outputs	Exception Handler	Comment
E 0	If V5_OV OR V33_OV OR ...	V5_EN = 0,V33_EN = 0,V25_EN = 0,	Starts at step 15	
	<end-of-exception-table>			

Monitoring the three outputs can be accomplished with the single exception shown in the exception table. There are three significant parts to the exception, each described by an entry in the table:

1. Boolean Expression
2. Outputs
3. Exception Handler

The Boolean Expression entry specifies the conditions under which the exception will be activated; in this case, when one or more of the monitored variables becomes true.

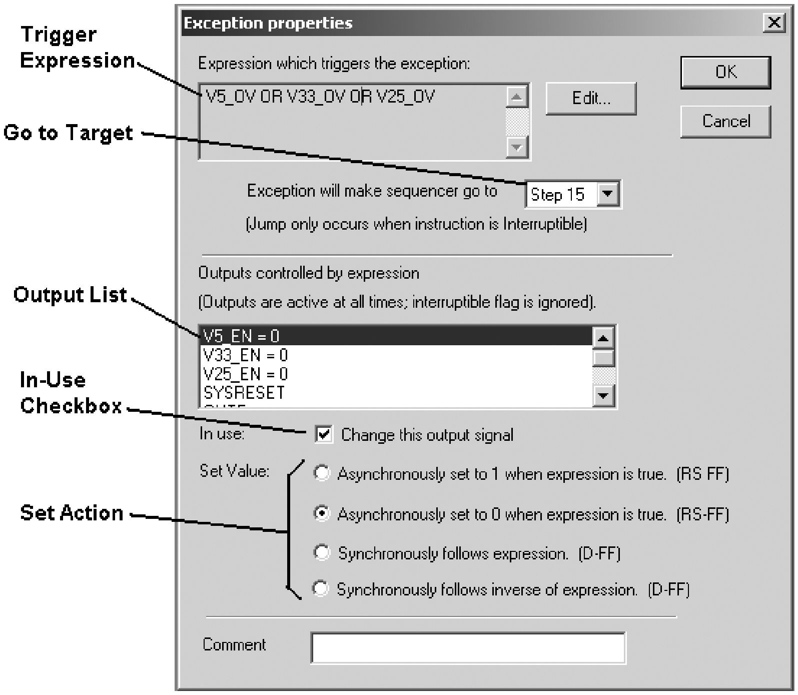
The Outputs entry specified any actions which should be performed on the outputs. In this example, we want to turn off all of the power supplies immediately.

The Exception Handler entry specifies a point at which the sequence should begin execution when the exception occurs. For this exception, we want to HALT when this exception has occurred (after turning off all the outputs). Specifying that the exception begin execution at sequencer step 15 (which contains a HALT instruction) will fulfill this requirement.

Note the column labeled 'Interruptible' in the main LogiBuilder sequence (Figure 8). A 'Yes' or 'No' in this column indicates whether this instruction may be interrupted by the 'Jump' part of the exception, which is specified in the exception window under the 'Exception Handler' column. All instructions after the BEGIN SHUTDOWN SEQUENCE instruction will have a 'No' entry in this column and therefore will not be interruptible by any exceptions. This feature is provided so that sub-sequences which are jumped to in the event of an exception cannot be interrupted by other exceptions while executing.

To create an exception, select a line in the exception table and press the **INSERT** key; this will create a blank exception template. Double-clicking on this template will bring up the Exception Properties dialog box (Figure 9).

Figure 9. Exception Properties Dialog



Exception properties

Expression which triggers the exception:

V5_OV OR V33_OV OR V25_OV

Exception will make sequencer go to Step 15

(Jump only occurs when instruction is Interruptible)

Outputs controlled by expression

(Outputs are active at all times; interruptible flag is ignored).

V5_EN = 0
V33_EN = 0
V25_EN = 0
SYSRESET

In use: ☑ Change this output signal

Set Value:
○ Asynchronously set to 1 when expression is true. (RS FF)
● Asynchronously set to 0 when expression is true. (RS-FF)
○ Synchronously follows expression. (D-FF)
○ Synchronously follows inverse of expression. (D-FF)

Comment

The Exception Properties dialog allows the user to specify all of the relevant information required to implement exceptions. In addition to being able to specify exceptions which jump to a particular sequence step, this dialog also allows the user to specify immediate actions which can be conditionally performed. The exception dialog has several parts.

Trigger Expression defines when the exception becomes active. The trigger expression is displayed in the indicated text box. Clicking on the 'Edit' button to the right of this field opens up the Boolean Expression Editor, allowing the user to enter and edit trigger expressions.

Go to Target specifies a sequence step to which control will transfer to when the trigger expression becomes true. The target step is selected from the specified drop-down list. It is also possible to create exceptions which do not cause a sequence jump, but only perform actions on the outputs. To handle these cases, the 'not used' entry can be selected from this list.

The lower half of the exception dialog is used to specify actions which can be taken when the exception occurs, in addition to a sequence jump. The **Output List** box allows the user to specify an output which should be modified as a result of the exception. To select an output for modification, select it, and check the **In-Use Checkbox** immediately below the output list.

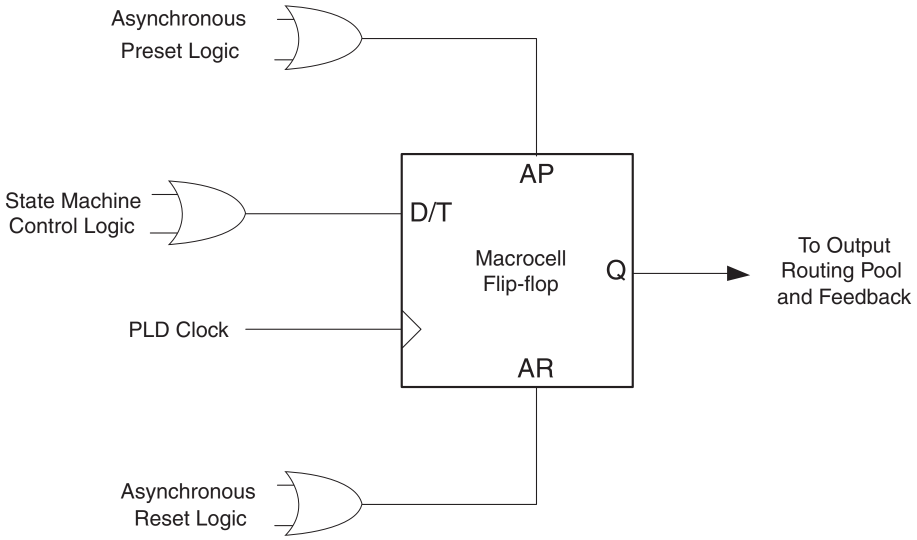
Once an output is selected for modification, there are several ways in which it can be modified (**Set Action**). The first two are to asynchronously set or reset it. These options are selected by checking the 'Asynchronously set to 1…' or 'Asynchronously set to 0…' options. When the exception condition occurs, the selected output will be immediately set or reset, and will stay in that condition until the exception trigger expression becomes false and it is set to another state elsewhere in the sequence. Note that in order to modify an output asynchronously, that output must be defined as an output in the sequence window. The asynchronous set actions are useful when you want to turn something off very quickly.

Synchronously set actions can be used to implement random logic functions. When an output is selected to 'Synchronously follows expression', that output is defined as a 'D' flip-flop, and the value of the trigger expression is synchronously clocked into that flip-flop's 'D' input. When an output is selected to 'Synchronously follows inverse of expression', the output is again defined as a 'D' flip-flop, but it is the logical inverse (NOT) of the trigger expression's value that is fed into that flip-flop's 'D' input. When an output is specified to be modified by synchronous set actions, that output **should not be used anywhere in the main sequence**.

Note that whenever either a synchronous or asynchronous action is specified as part of an exception, that output action **will be active in ALL steps of a sequence, regardless of the state of the 'interruptible' column**. The status of the 'interruptible' column only enables and disables the 'go to' part of an exception. The synchronous and asynchronous output actions defined in exceptions are active at all times.

The different types of set actions map directly into the Power/Platform Management device's logic macrocell structure. Figure 10 shows a simplified schematic of a Power/Platform Management device's macrocell, as used by LogiBuilder.

Figure 10. Simplified Schematic of Logic Macrocell as Seen by LogiBuilder



Each macrocell contains a flip-flop that is synchronously clocked from the PLD clock. The logic specified by LogiBuilder to specify sequences and control outputs is fed into the D/T input of the flip-flop (the decision to use a given flip-flop in 'D' or 'T' modes is made transparently by the logic optimization and fitting tools). When an exception specifies that an output be synchronously set or reset, the logic required to implement these functions is combined with that used to implement the main sequence, and fed into the D/T input.

When an exception specifies an asynchronous set or reset, however, the required logic is fed into the flip-flop's AP (asynchronous preset) or AR (asynchronous reset) inputs. While this results in instantaneous response, it also usually requires that one or more additional macrocells to be recruited to implement the necessary functions. Because of these additional hardware resource requirements, one should avoid using the asynchronous set and reset modes unless their instantaneous characteristics are absolutely needed.

Summary

This application note has described PAC-Designer's LogiBuilder system, and shown how it can be used to develop power-supply control logic to be implemented in a Power/Platform Management device's power-supply sequencer IC. A small design was presented which was used to introduce the various capabilities of LogiBuilder and its Sequence Control instruction set. By providing an intuitive, easy-to-learn, and integrated design environment, LogiBuilder reduces development time required for designing a set of control logic from hours to minutes. Because LogiBuilder is part of the PAC-Designer software environment, implementing a design in the physical hardware is a single-click download operation.

Related Literature

- DS1034, ProcessorPM-POWR605 Data Sheet
- DS1011, ispPAC-POWR607 Data Sheet
- DS1014, ispPAC-POWR1014/A Data Sheet
- DS1015, ispPAC-POWR1220AT8 Data Sheet
- DS1035, Platform Manager Data Sheet

Technical Support Assistance

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

Revision History

Date	Version	Change Summary
January 2003	01.0	Initial release.
April 2008	01.1	Title changed from “Implementing Power Supply Sequencers with the ispPAC-POWR1208 and PAC-Designer LogiBuilder” to “Implementing Power Supply Sequencers with Power Manager Devices and PAC-Designer LogiBuilder.”
October 2011	01.2	Updated to cover a broader range of Power/Platform Management devices.