

Introduction

This application note discusses the states of the Platform Manager™ 2 and L-ASC10 (Analog Sense and Control - ASC) device output pins – Open drain logic outputs (GPIO1-10), HVOUT, Trim DAC and Programmable FPGA IO (Platform Manager 2 only) – during power-up, reset, and programming, as well as the states of these pins on a device that has not yet been programmed by the user. An understanding of these aspects of the device operation is the key to designing systems that sequence supplies in a dependable fashion. Suggested methods for interfacing to the enable pin of the DC/DC converters and for performing in-system programming are also covered in this application note. This document also describes the procedure for Platform Manager 2 and L-ASC10 with Diamond® Programmer.

Power-on Reset State

The Platform Manager 2 and ASC devices operate at a VCC of 3.3 V. The 3.3 V can be supplied directly to the Platform Manager 2 and ASC from a dedicated voltage regulator or supplied from the Platform Manager 2 DC-DC converter. This section covers the power-on reset behavior based on the 3.3 V supply. The Platform Manager 2 DC-DC behavior is outside the scope of this document. See [Related Literature](#) for more details.

The Platform Manager 2 and ASC contain on-chip power-on reset circuitry to ensure that all parts of the device start up reliably. The Platform Manager 2 device contains both an ASC section and an FPGA section which execute their power-on reset behavior slightly differently.

ASC Behavior

The ASC device and ASC section of Platform Manager 2 enter power-on reset when VCCA is approximately 0.8 V. VCCA must reach a voltage greater than 2.5 V for the device to exit power-on reset. During power-on reset, the output pins will go to the states shown in Table 1.

Table 1. ASC Output States During Power-On Reset (Safe States)

Output Type	Power-on Reset State (Safe State)
ASC GPIO1-6, GPIO10	Low / Pull-down
ASC GPIO7-9	Hi-Z
HVOUT, Charge Pump Mode (as-shipped)	Low / Pull-down
HVOUT, open Drain Logic Mode	Low / Pull-down
Trim DAC	Hi-Z

The safe state output behavior described in Table 1 applies to the ASC GPIO even when the GPIO are programmed as inputs or to be driven by the on-chip output control block (OCB).

The ASC also has a bi-directional reset pin (RESETb) which is used to manage and communicate the ASC reset status with the Platform Manager 2 FPGA section or MachXO2™ FPGA in the system. Figure 1 shows an equivalent circuit for this pin; it is an open-drain output with a built-in pull-up resistance to VCCA. This pin should only be connected between mandatory ASCs and the Platform Manager 2 or MachXO2 FPGA. For more details see the data sheet section on system connections.

Figure 2 through Figure 5 illustrate the power-on reset behavior associated with the GPIO and HVOUT pins. 2.2 kOhm pull-up resistors to the VCCA pin were used on the output pins, with a 10K pull-up resistor used on the RESETb pin in all the plots.

Figure 1. RESETb Pin Equivalent Circuit

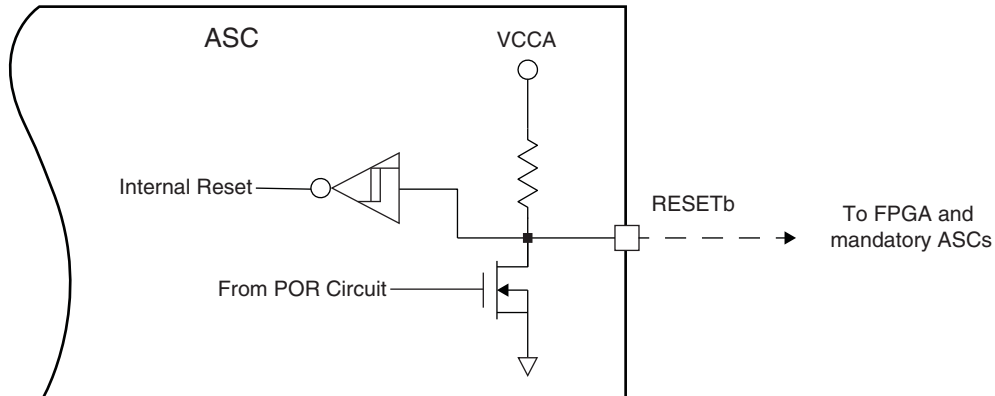


Figure 2 and Figure 3 show the output pin GPIO2 programmed to behave as a registered output. The difference between these plots is that in Figure 2 the output has been programmed to reset to a low level, whereas in Figure 3, it has been programmed to reset high. Note that GPIO2 is part of the GPIO group with a POR state (Safe State) of Low. The power supply ramps from zero to 3.3 V in 6 milliseconds.

Figure 2. Startup with Slow Supply; GPIO2 (Safe State Low); Output Resets Low

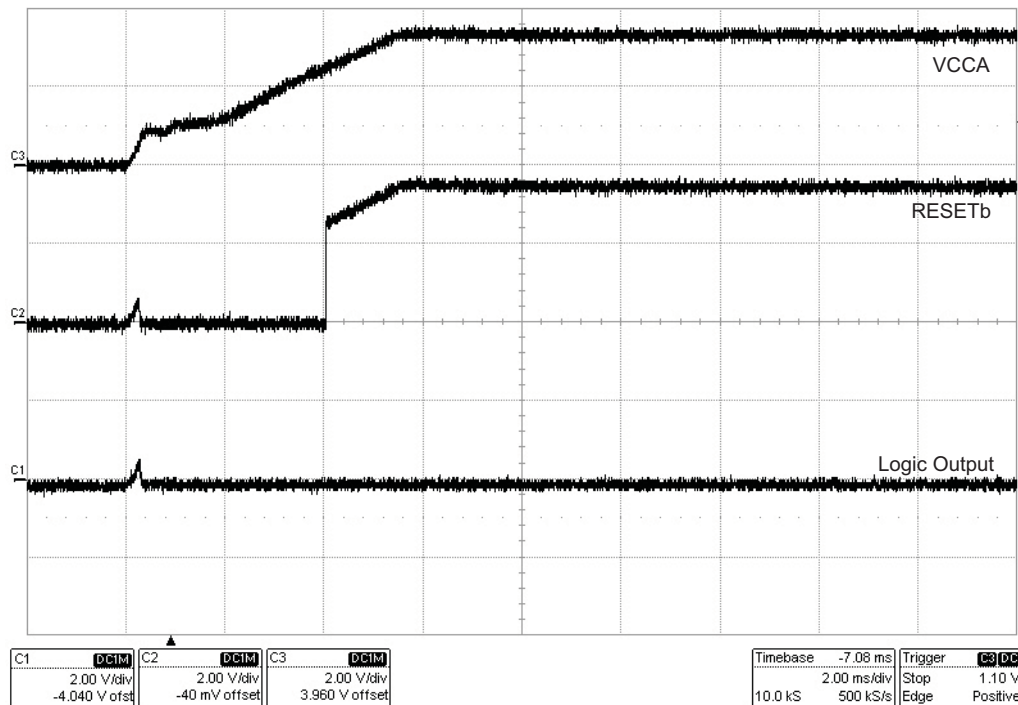
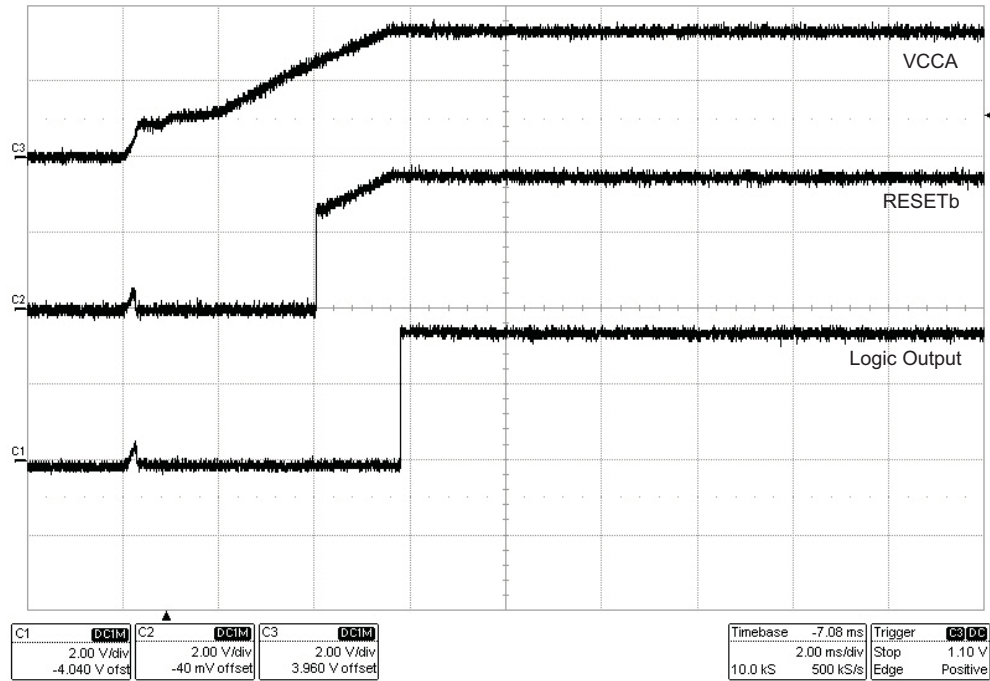


Figure 3. Startup with Slow Supply; GPIO2 (Safe State Low); Output Resets High



The behavior in both cases shows that once the VCCA crosses 0.8 V, the output enters its safe state and the device is held in reset. When VCCA crosses 2.5 V, the RESETb signal is released. The output remains in the safe state for about 1.8 ms after RESETb is released (T_{SAFE} in the ASC data sheet) – this behavior is best seen in Figure 3. During this time the ASC is completing its internal calibration and the communication between the FPGA and the ASC is being initialized. The difference between the two cases is that in Figure 2 GPIO2 is programmed to Reset Low and in Figure 3 GPIO2 is programmed to Reset High.

Figure 4 and Figure 5 show the output pin GPIO8 programmed to behave as a registered output. The difference between these plots is that in Figure 4 the output has been programmed to reset to a low level, whereas in Figure 5, it has been programmed to reset high. Note that GPIO8 is part of the GPIO group with a POR state (Safe State) of Hi-Z. The output remains in the Hi-Z safe state from the time is power is applied, through the power-on reset process. The output will assert its reset level about 1.8ms after RESETb has been released. The power supply ramps from zero to 3.3 V in 6 milliseconds.

Figure 4. Startup with Slow Supply; GPIO8 (Safe State Hi-Z); Output Resets Low

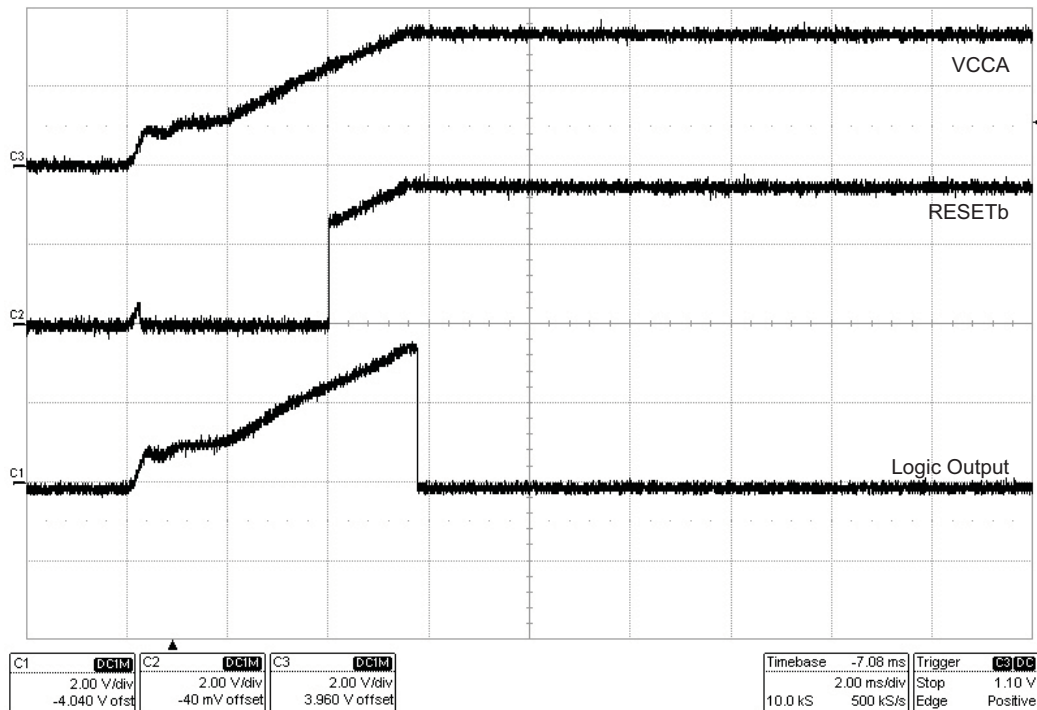


Figure 5. Startup with Slow Supply; GPIO8 (Safe State Hi-Z); Output Resets High

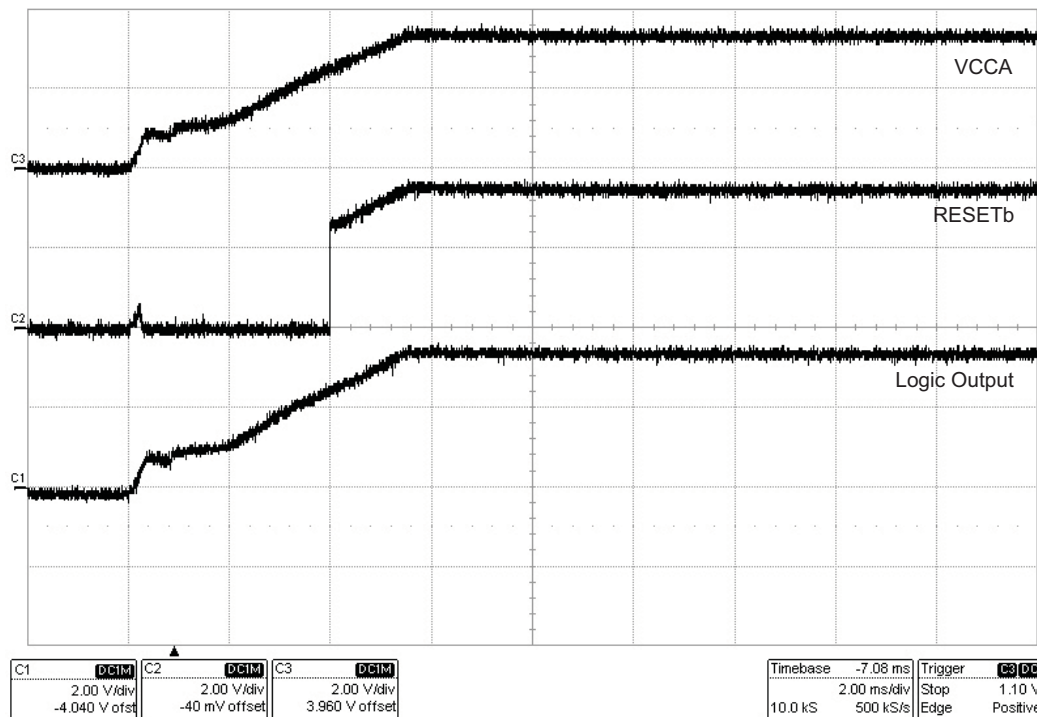


Figure 6 and Figure 7 show the GPIO waveforms observed when the same device and setup that were examined in Figure 2 and Figure 5 are powered from a supply that ramps from 0 to 3.3 V in 150 microseconds. Despite the faster power-up rate, RESETb still indicates that power-on reset starts when VCCA reaches 0.8 V. The GPIO enter

their safe state (Low for GPIO2 in Figure 6, Hi-Z for GPIO8 in Figure 7) at 0.8 V, similar to the slow rising VCCA. The outputs in this case will also remain in safe state for about 1.8ms before switching to their programmed reset levels. The switch to the programmed reset levels is not shown in the figures as the time scale is shorter to account for the faster supply ramp rate.

Figure 6. Startup with Fast Supply; GPIO2 (Safe State Low)

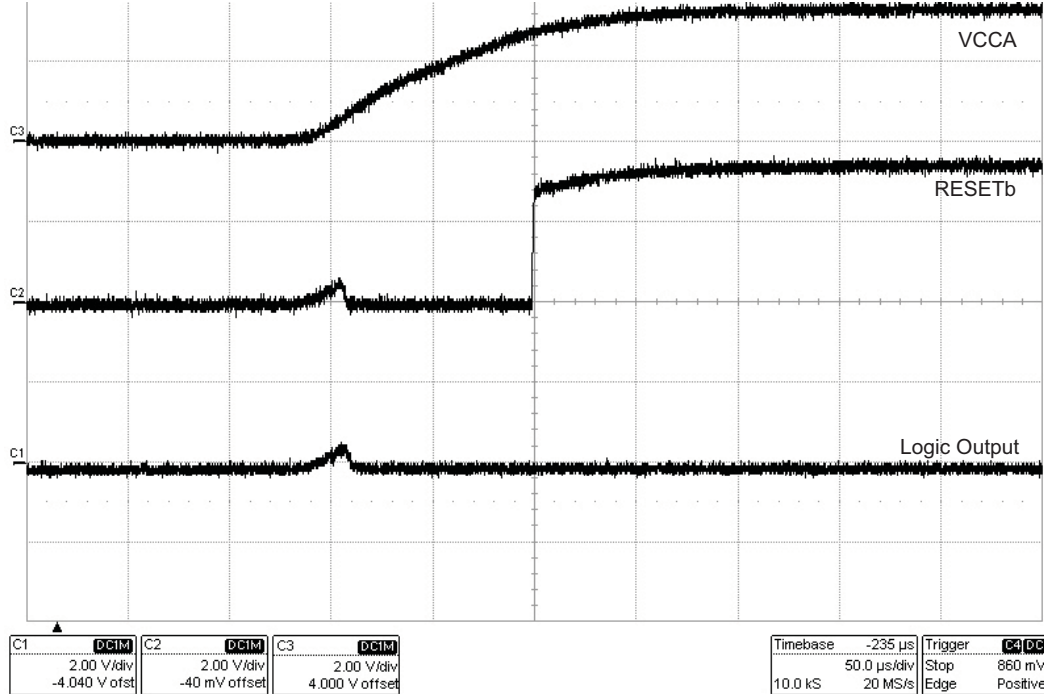
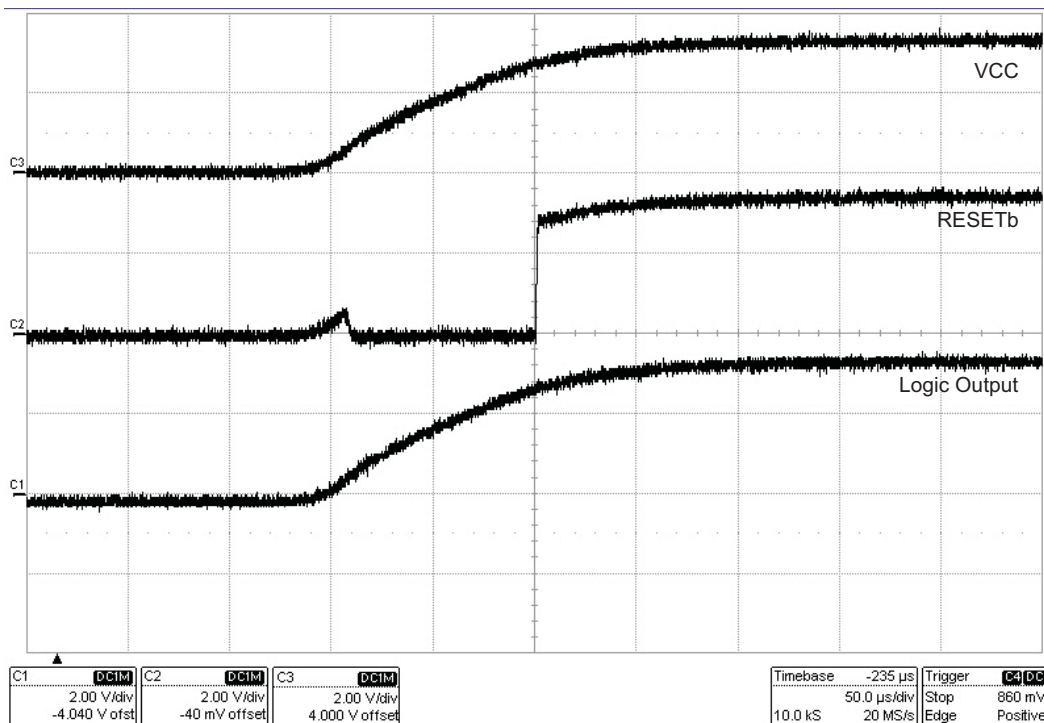


Figure 7. Startup with Fast Supply; GPIO8 (Safe State Hi-Z)



HVOUT pins will remain in their low safe state during power-on reset and for the time T_{SAFE} after the ASC exits power-on reset. HVOUT pins remain in the low state regardless of their programmed output mode (charge pump or open drain). This ensures that external MOSFETs controlled by the HVOUT pins will remain off when the ASC begins power supply sequencing operations.

A delayed calibration complete signal, called AGOOD in Platform Designer, becomes true in the FPGA logic after time T_{SAFE} has elapsed (about 1.8ms). This is dependent on the proper initialization and startup behavior of the ASC Interface (ASC-I/F) signals between the ASC and the FPGA. If the ASC-I/F does not startup properly, the AGOOD signal will not be asserted and the pins will remain in their safe states.

The Trim DAC outputs will maintain a high impedance state during the entire power up cycle. The Trim DACs will remain in the high impedance state until their associated TRIMHiZ signal is set to one over the ASC-I/F. This will not occur any sooner than the time it takes T_{SAFE} to elapse (about 1.8 ms).

FPGA Behavior

The PIO in the FPGA section of Platform Manager 2 will startup and configure in an independent process to the ASC section. The FPGA section enters power-on-reset when it crosses the VCC_POR level specified in the data sheet. Once the power-on-reset begins, the FPGA initializes its configuration from the internal flash memory. This is the $T_{refresh}$ time specified in the Platform Manager 2 data sheet.

During the power-on reset and configuration time, all the FPGA I/O will be held in tri-state. The I/Os are released to user functionality once the device has finished configuration. At that time, the FPGA I/Os controlled by the Platform Designer Logic will enter their reset state as defined by the Platform Designer software. FPGA I/Os which are controlled by user-defined HDL logic will follow the reset behavior defined in the user logic and ignore the Platform Designer reset settings.

Resets During Runtime

During runtime, there are two conditions which will cause the ASC to be reset: pulling the RESETb pin low, or an unexpected loss of WRCLK (ASC-I/F clock). The reset behavior of ASC in both cases involves the analog circuitry, all output pin drivers, and the I²C registers (both configuration and measurement). The output pins will go to their safe states, defined in Table 1, during a reset event.

At the conclusion of the reset event, the RESETb pin goes high. The output pins will remain in their safe state until time T_{safe} (app. 1.8 ms) has elapsed. Exiting the safe state depends on the ASC-I/F communication being properly established (including valid WRCLK detection). At that time the output pins will return to their defined reset state from the Platform Designer software.

Only Platform Manager 2 family devices should be connected to the RESETb pin of an ASC. This includes Platform Manager 2, MachXO2, or additional mandatory ASCs. See the data sheet section on system connections for more details.

The RESETb pin in Platform Manager 2 does not put the FPGA section into a power on reset state. It is used to signal the ASC-I/F and Logic Sequencing that the ASC section has gone into reset. The FPGA PIO pins controlled by the Platform Designer Logic will return to their reset as defined by the Platform Designer software. The FPGA behavior in response to an external ASC device being reset depends on the ASC designation of mandatory or optional. See the data sheet section on system connections for more details.

Behavior During Programming

The ASC is programmed over I²C. It can be directly programmed over I²C, or programmed via the JTAG-to-I²C interface in the Platform Manager 2 or MachXO2 FPGA. The ASC supports both background programming and offline (direct) programming. During a background programming operation, the device outputs are not disturbed as the device continues to execute in user mode. The configuration loaded in the background programming will be loaded as the active configuration after the next power-on reset. During offline programming using the JTAG to I²C mode, the device output pins are held in safe state. Refer to Table 1 for the output pin safe states.

The FPGA section of Platform Manager 2 is programmed by JTAG or I²C and can be programmed in background mode or offline mode. During offline programming, the I/O pins will return to tri-state. During a background programming mode the device remains in user mode and normal operation is not disturbed. The configuration loaded in the background programming will be loaded as the active configuration after the next power-on reset.

Shipped State of New Parts

The ASC device will hold its output pins in the safe state until communication is successfully initialized over the ASC-I/F. Provided that no ASC-I/F logic is programmed into the Platform Manager 2 or MachXO2 FPGA, the ASC outputs will remain in the safe state as shown in Table 1. The base I²C address of devices in the as-shipped state is 0x60. The ASC will update the three least significant bits of its I²C address dependent upon the value of the resistor connected at the I²C_Addr pin during Power-On Reset. (See the data sheet for more details). Note that ASC devices and the FPGA on evaluation boards may come with the ASC-I/F logic and a test sequence programmed by the board assembler and this sequence may take the ASC out of safe state.

Ensuring Reliable Operation at Power-Up

Power supply systems using the ASC should be designed with the safe state of each open drain output (either GPIO or HVOUT) kept in mind. GPIO1-6, GPIO10 and HVOUT1-4 (in open drain mode) should be interfaced to positive enable logic supplies (i.e. supplies which are turned on when the enabled pin is pulled to the DC/DC converter input supply) when possible. The low/pull-down safe state of the outputs will disable the supply during power-up, programming and reset. These open drain outputs can often be connected directly to the DC-DC enable pin using just a pull-up resistor. The DC-DC converter data sheet should be checked to see whether its enabled input has a built-in pull-up resistor or whether an external resistor must be used. Table 2 lists the values that should be used in Platform Designer Logic Output instructions and the reset levels that should be selected in the Ports and Nodes window for positive enable logic supplies.

GPIO7-9 should be interfaced to negative enable logic supplies (i.e. supplies which are turned on when the enable pin is pulled to ground). The high impedance safe state of these outputs will disable the supply during power-up, programming and reset. These open drain outputs can often be connected directly to the DC-DC enable pins. The DC-DC converter data sheet should be checked to see whether its enable input has a built-in pull-up resistor or whether an external resistor must be used. Table 3 lists the values that should be used in Platform Designer Logic Output instructions and the reset levels that should be selected in the Ports and Nodes window for negative enable logic supplies.

In situations where there are not enough GPIO pins with the required reset logic, the circuit in Figure 8 can be used to invert the output safe state for reliable control of the DC-DC supplies. Table 2 and Table 3 also cover the output and reset levels which should be selected with outputs using this inverter circuit.

Figure 8. Interfacing Output Pins to Supplies with Low Safe State and Negative Enable Logic (Or Hi-Z safe state and Positive Enable Logic)

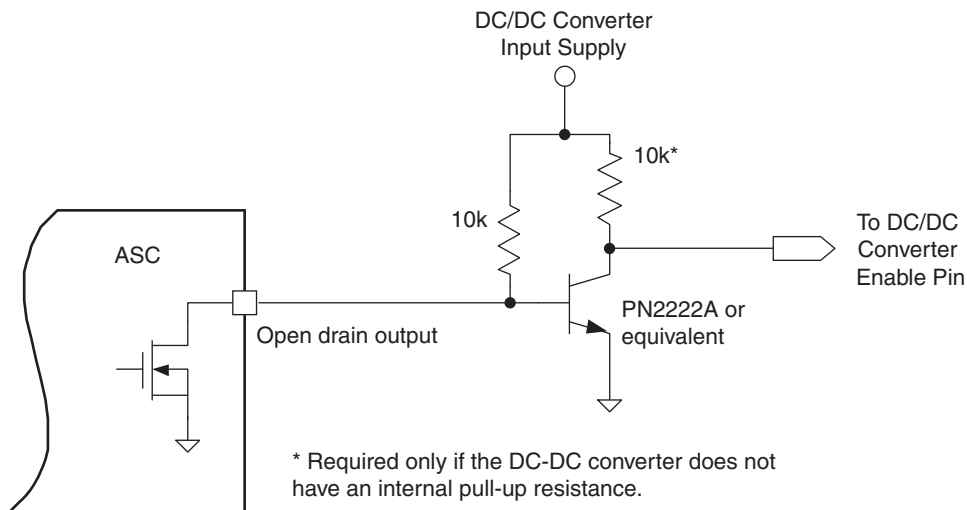


Table 2. Controlling Positive Logic Supplies

Output Pin Group	Safe State	Interface Method ¹	Reset Level	Logic Sequence OUTPUT Value to Turn Supply ON	Logic Sequence OUTPUT Value to Turn Supply OFF
GPIO1-6,10, HVOUT1-4	Low (Pull-down)	Connect directly to Open Drain Output	LOW	1 (Assert)	0 (Deassert)
GPIO7-9	Hi-Z	Use Inverter Circuit (Figure 8)	HIGH	0 (Deassert)	1 (Assert)

1. Interface circuit used should also conform to recommended operating conditions detailed in device data sheets. See [Related Literature](#) section for more information.

Table 3. Controlling Negative Logic Supplies

Output Pin Group	Safe State	Interface Method ¹	Reset Level	Logic Sequence OUTPUT Value to Turn Supply ON	Logic Sequence OUTPUT Value to Turn Supply OFF
GPIO1-6,10, HVOUT1-4	Low (Pull-down)	Use Inverter Circuit (Figure 8)	LOW	1 (Assert)	0 (Deassert)
GPIO7-9	Hi-Z	Connect directly to Open Drain Output	HIGH	0 (Deassert)	1 (Assert)

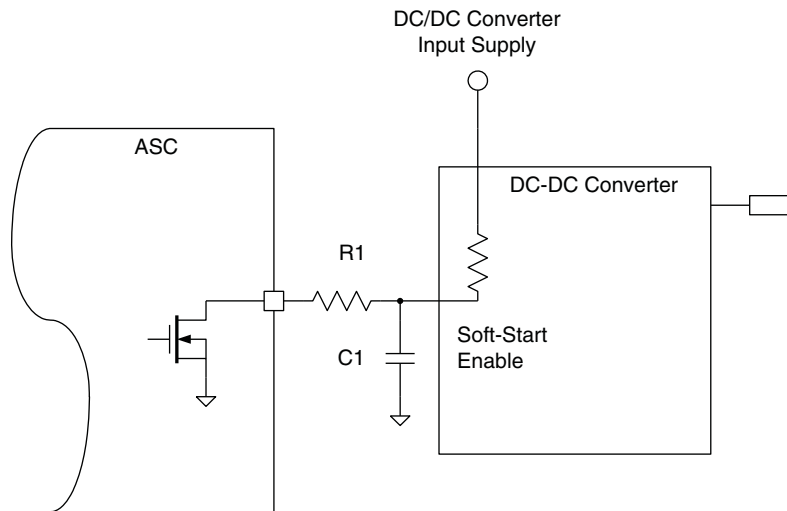
1. Interface circuit used should also conform to recommended operating conditions detailed in device data sheets. See [Related Literature](#) section for more information.

Capacitive Loading on Outputs

In certain applications, the outputs of the ASC are used to enable the soft-start feature of a DC-DC converter. This presents a capacitive load to the outputs of the ASC. During the power-up cycle of the ASC, the output pins will transition to their safe states. If the soft-start capacitors have any charge stored during this time, excessive currents will flow through the open-drain outputs to ground. These currents can result in an unreliable start-up of the ASC. Furthermore, during normal operation setting an output low can also result in a discharge current that exceeds the maximum limits in the data sheet.

To ensure reliable ASC start-up and to limit the current surge it is required to add a series resistor when a capacitive load is connected to an output as shown in Figure 9. The value of R1 should be such that when C1 is discharged by the open-drain output the current is less than the $I_{SINKMAX}$ limit in the data sheet. The $I_{SINKMAX}$ is for any single output and is found in the Absolute Maximum Ratings table of the data sheet. If multiple outputs have capacitive loads the $I_{SINKTOTAL}$ Maximum should be considered in setting a value for the series resistors. The $I_{SINKTOTAL}$ is found in the Digital Specifications table of the data sheet and applies to all outputs combined.

Figure 9. Limiting Capacitive Load Currents to Less than $I_{SINKMAX}$



Programming Procedure

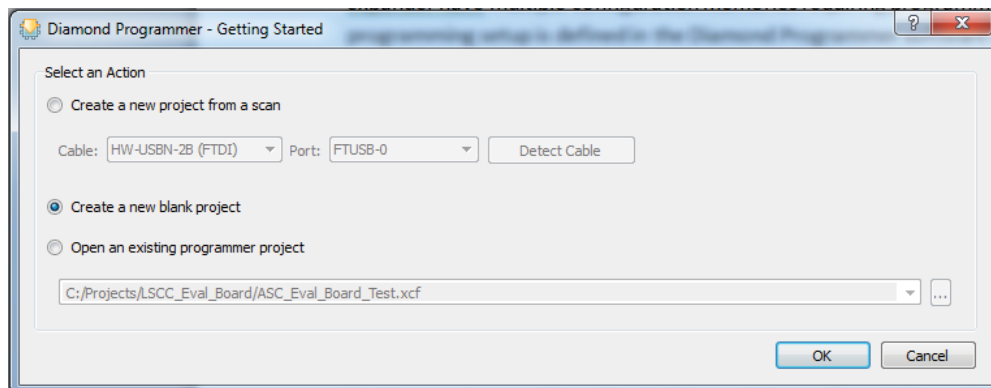
Hardware management controllers based on Platform Manager 2 and the ASC hardware management expander have multiple configuration memories requiring programming. The configuration memory programming setup is defined in the Diamond Programmer software tool. Depending on the system configuration, different steps are required to define the programming setup. The main system configurations are:

- Platform Manager 2
- Platform Manager 2 and Additional L-ASC10 Device(s)
- MachXO2 and L-ASC10 Devices(s)

The general procedure for programming each of these systems is described below. A basic understanding of the Diamond Programmer software tool is assumed by these descriptions. For more details on Diamond Programmer, see the Programming Tools User Guide in the Related Literature section of this document. This document covers the procedure for defining the programming operation for these system configurations in Diamond Programmer. Information on advanced operations like generating embedded programming algorithms, SVF files, or other topics can be found in the Programming Tools User Guide.

The step-by-step programming procedures shown in the following sections follow the *Create a new blank project* action in Diamond Programmer, as shown in Figure 10 below. Several of these steps can be bypassed by launching the Diamond Programmer directly from the Lattice Diamond software project used to generate the programming files.

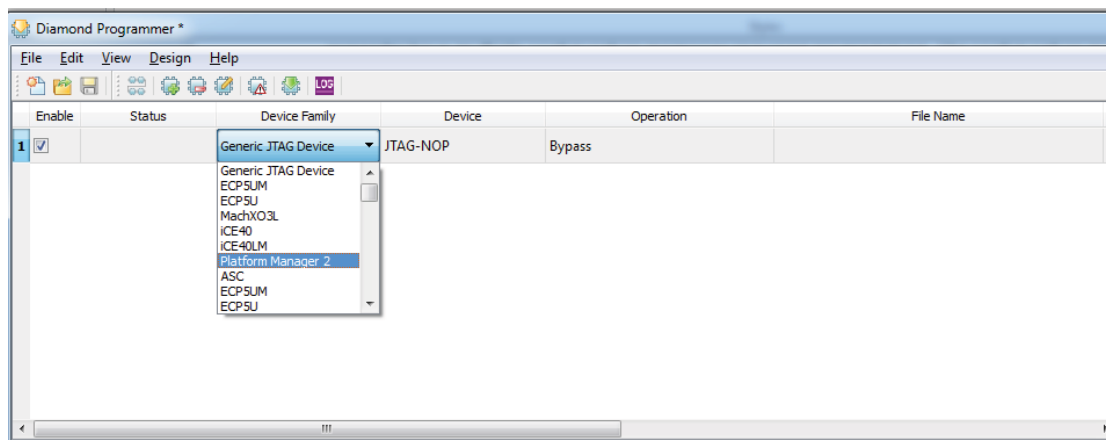
Figure 10. Diamond Programmer - Getting Started Screen



Platform Manager 2

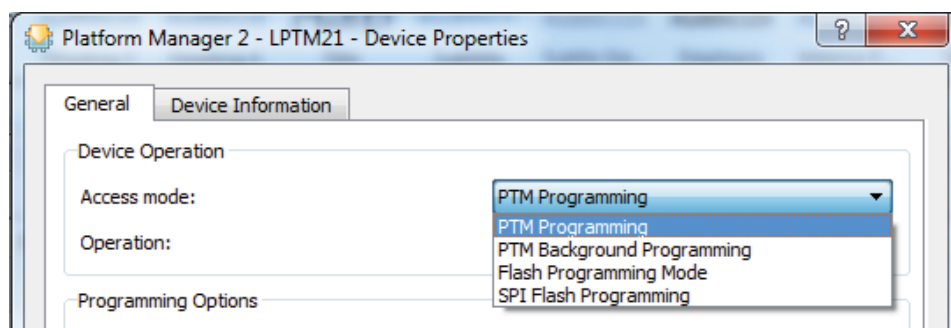
1. The first step in programming the Platform Manager 2 is to select **Platform Manager 2** from the Device Family list, as shown in Figure 11. The LPTM21 device is automatically selected.

Figure 11. Selecting Platform Manager 2 Device Family



2. Double-click the **PTM Erase, Program, Verify** box in the Operation column. The Device Properties dialog box opens.
3. Select the Access mode. Four access modes are available, as shown in Figure 12. These modes are:
 - **PTM Programming** – Offline JTAG programming of ASC and FPGA sections
 - **PTM Background Programming** – Background JTAG Programming of ASC and FPGA sections
 - **Flash Programming Mode** – Offline JTAG programming of FPGA section only
 - **SPI Flash Programming** – Programming external SPI flash via JTAG-SPI bridge

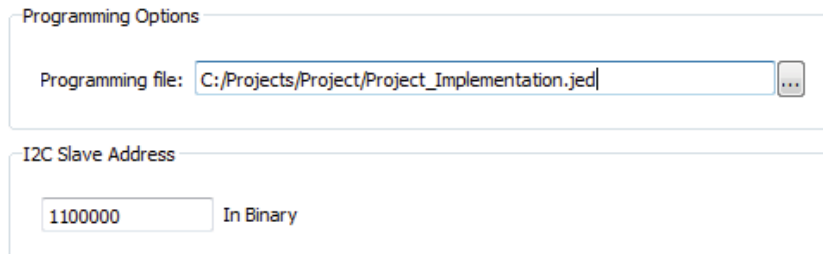
Figure 12. Choosing the Access Mode for Platform Manager 2 Programming



This example procedure works with the PTM Programming mode, although the steps apply to PTM Background Programming as well.

4. Select the operation. The **PTM Erase, Program, Verify** operation is the standard operation for programming the device.
5. Select the programming file. Browse to the .jed file generated from Platform Designer (the software tool for implementing designs in Platform Manager 2 systems.) Platform Designer generates a <Project>_<Implementation>.jed file (with Project being the project name, and Implementation the implementation name). Platform Designer places this in the top level Project folder, as shown in the Programming File dialog box in Figure 13.

Figure 13. Programming File Location for .jed File



Programming Options

Programming file: C:/Projects/Project/Project_Implementation.jed

I2C Slave Address

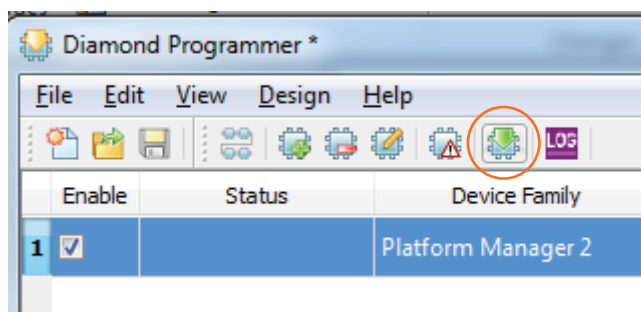
1100000 In Binary

Figure 13 also shows the I²C Slave Address. This is the address of the ASC section of the Platform Manager 2. The default value for any Platform Designer project is 1100000, the same as the default in Diamond Programmer. This must be the current address of the device you are trying to program.

6. In systems with no additional ASCs, you can click **OK** in the Device Properties dialog after completing steps 1-6. This completes the operation definition for Platform Manager 2.

You can proceed with programming the device directly in Diamond Programmer using the  button shown in Figure 14. You can also save your programmer project (.xcf file) for use with Diamond Deployment Tool if you plan to generate files for embedded programming or tester implementations.

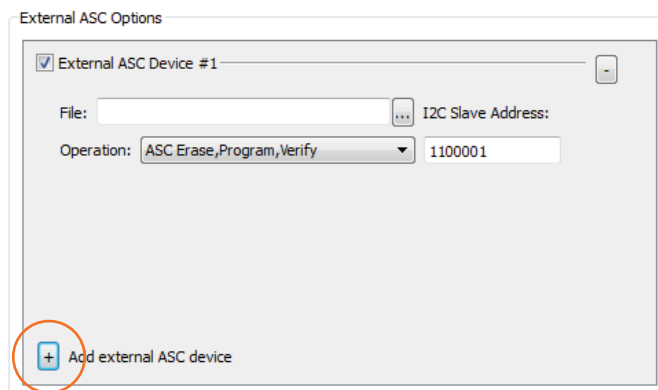
Figure 14. Programming Button in Diamond Programmer



Platform Manager 2 and Additional L-ASC10 Device(s)

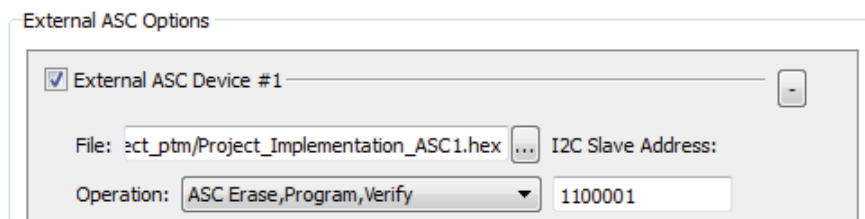
1. When programming systems based on Platform Manager 2 and additional L-ASC10 devices, repeat the Platform Manager 2 Programming Procedure steps 1-6 in the [Platform Manager 2](#) section.
2. In systems with additional ASCs, the External ASC Options need to be defined. Clicking the + - **Add external ASC device** brings additional ASCs into the programming project. Click + for each additional ASC device in the system, as shown in Figure 15.

Figure 15. Adding External ASC Devices to Platform Manager 2



3. Select the programming file for each ASC device added in the system. The ASC devices are programmed with .hex files which are generated by Platform Designer. Browse to the .hex files, which are found in the Project/Project_ptm/ folder. Each ASC hex file is named <Project>_<Implementation>_ASC#.hex as shown in Figure 16.

Figure 16. File Location for ASC1.hex File

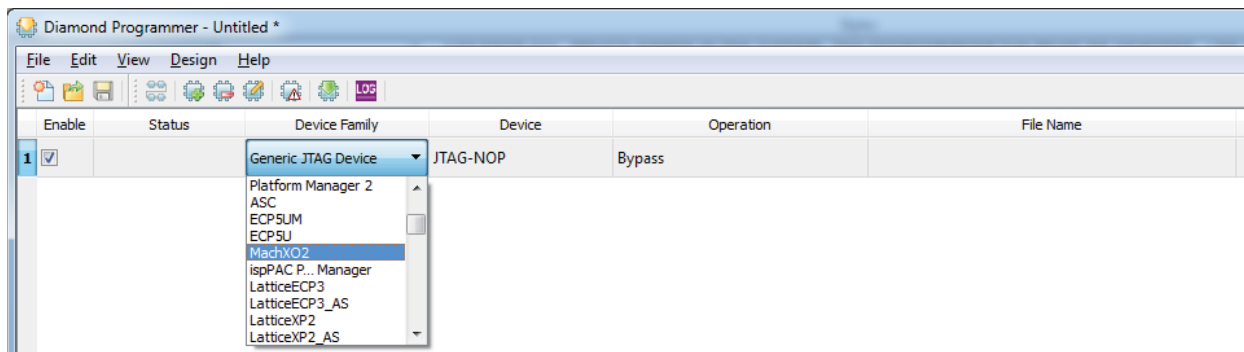


4. Select the operation. The **ASC Erase, Program, Verify** operation is the standard operation for programming the ASC devices.
5. The I²C Slave Address is also defined for each ASC. The default values populated in the dialog box match the default values generated in Platform Designer.
6. Once each external ASC in the system is added and the programming operation is defined, you can proceed with programming the device directly in Diamond Programmer using the  button shown in Figure 14. You can also save your programmer project (.xcf file) for use with Diamond Deployment Tool if you plan to generate files for embedded programming or tester implementations.

MachXO2 and L-ASC10 Devices(s)

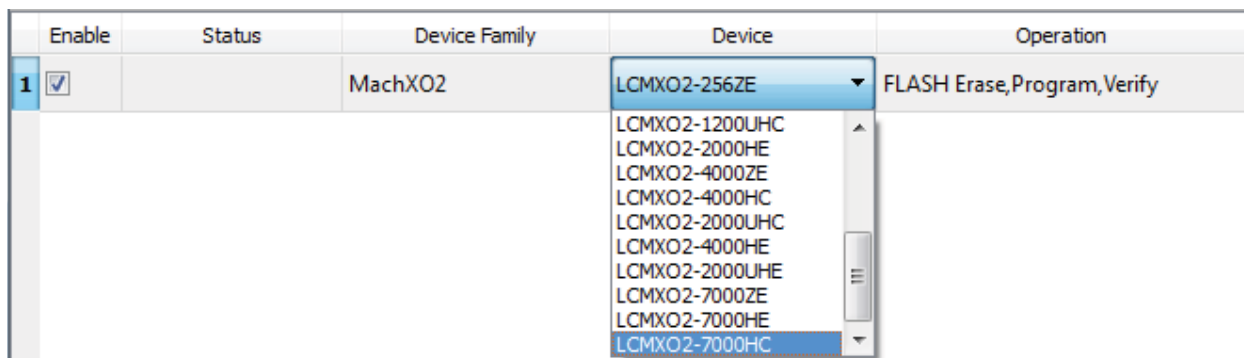
1. The first step in programming the MachXO2 and L-ASC10 device(s) is to select the MachXO2 from the device family, as shown in Figure 17.

Figure 17. Selecting MachXO2 Device Family



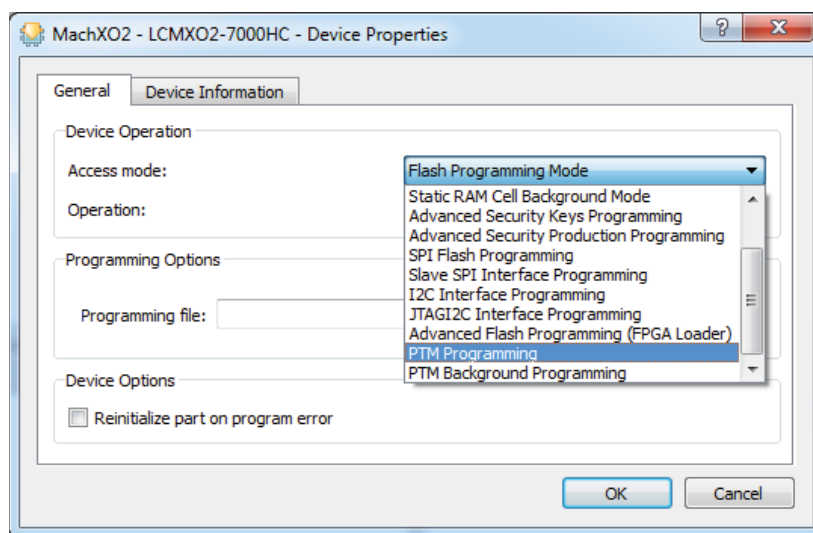
2. Select the MachXO2 used in the system from the Device menu, as shown in Figure 18.

Figure 18. Selecting Device from MachXO2 Device Family



3. Double-click the **FLASH Erase, Program, Verify** box in the Operation menu. The Device Properties dialog box opens.
4. Select the Access mode. MachXO2 supports numerous access modes, as described in TN1204, [MachXO2 Programming and Configuration Usage Guide](#). MachXO2 + ASC systems use the PTM Programming or PTM Background Programming modes, as shown in Figure 19. This example uses PTM Programming.

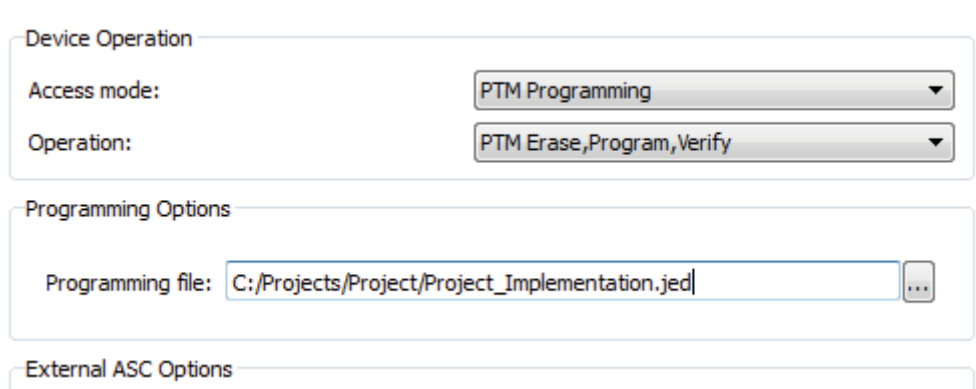
Figure 19. Selecting PTM Programming Access Mode for MachXO2 Devices



This example procedure works with the PTM Programming mode, although the steps apply to PTM Background Programming as well.

5. Select the operation. The **PTM Erase, Program, Verify** operation is the standard operation for programming the device.
6. Select the programming file. Browse to the .jed file generated from Platform Designer (the software tool for implementing designs in Platform Manager 2 systems.) Platform Designer generates a <Project>_<Implementation>.jed file (with Project being the project name, and Implementation the implementation name). Platform Designer places this in the top level Project folder, as shown in the Programming File dialog box in Figure 20.

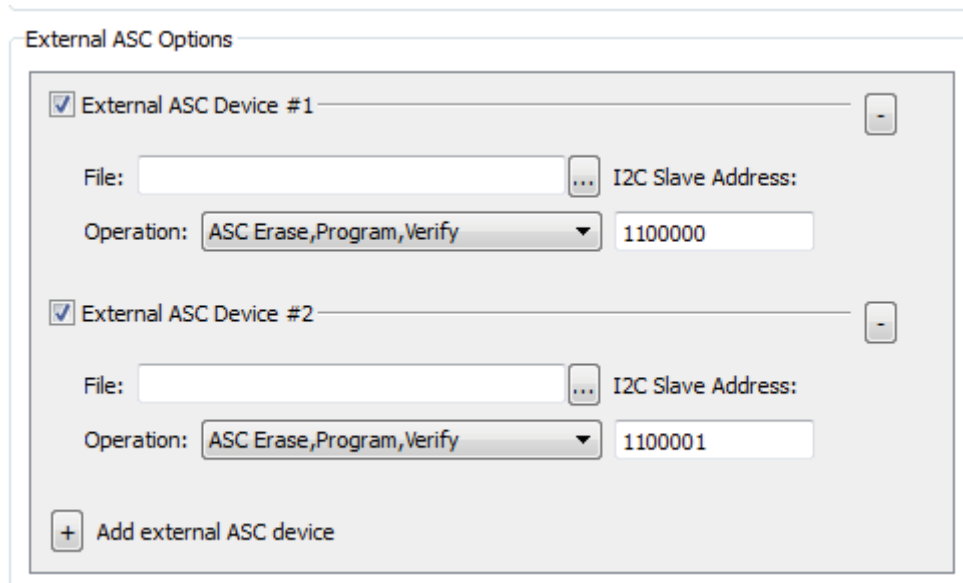
Figure 20. Programming File Location for .jed File



This sets the operation to program the MachXO2 only. Programming the ASC devices requires the following additional steps.

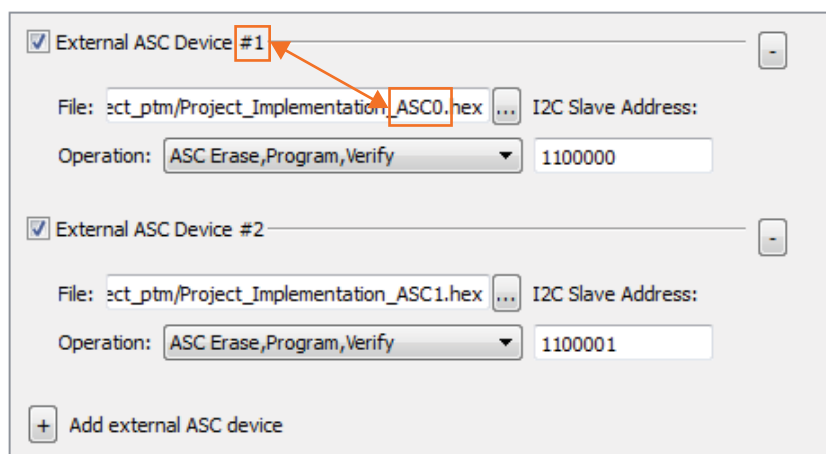
7. Click the **+ Add external ASC device** in the External ASC Options section to bring additional ASCs into the programming project. Click + for each additional ASC device in the system, as shown in Figure 21.

Figure 21. Adding External ASC Devices to the MachXO2 Programming Project



8. Select the programming file for each ASC device added in the system. The ASC devices are programmed with .hex files which are generated by Platform Designer. Browse to the .hex files, which are found in the Project/Project_ptm/ folder. Each ASC hex file is named <Project>_<Implementation>_ASC(#-1).hex as shown in Figure 22. Note that External ASC Device #1 is programmed with _ASC0.hex, External ASC Device #2 is programmed with _ASC1.hex, and so on.

Figure 22. Programming File Location and Numbering for ASC.hex Files



9. Select the operation. The **ASC Erase, Program, Verify** operation is the standard operation for programming the ASC devices.
10. The I²C Slave Address is also defined for each ASC. The default values populated in the dialog match the default values generated in Platform Designer. If a design uses an address other than the default, it must be entered correctly here for the programming to be successful.
11. Once each external ASC in the system has been added and the programming operation is defined, you can proceed with programming the device directly in Diamond Programmer using the  button shown in Figure 14. You can also save your programmer project (.xcf file) for use with Diamond Deployment Tool if you plan to generate files for embedded programming or tester implementations.

Summary

The startup characteristics of the open drain logic outputs (GPIO1-10), the HVOUT pins, and the Trim DACs of the ASC have been described as a function of power-on reset. The FPGA PIO of the Platform Manager 2 have also been described. The behavior of these outputs during runtime resets and during programming has also been described. The mix of high and low safe states of the ASC outputs have been designed to work directly with a variety of DC-DC converters with limited external components. A step-by-step procedure for programming the devices with Diamond Programmer has been provided. All of this information have been provided so that systems can be designed for predictable and reliable power sequencing.

Related Literature

- [Programming Tools User Guide](#)
- DS1042, [L-ASC10 Data Sheet](#)
- DS1043, [Platform Manager 2 Data Sheet](#)

Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
March 2015	2.1	Updated ASC Behavior section. Added description of GPIO as input and OCB.
		Updated Platform Manager 2 section. Removed LPTM20 reference.
August 2014	2.0	Added the Programming Procedure section.
December 2013	01.0	Initial release.