Power Manager II Hercules Development Kit – Standard and Advanced Versions

User's Guide

December 2010
Revision: EB57_01.0

Introduction

Thank you for choosing the Lattice Semiconductor Power Manager II Hercules Development Kit. The Hercules Development Kit is an easy-to-use platform for evaluating and designing with the Power Manager II ispPAC®-POWR1220AT8 and MachXO™2280. The Hercules Development kit comes preprogrammed with several demonstration designs that highlight the features and programmability of the POWR1220AT8 and MachXO devices. These demo designs can be easily modified to implement your own designs.

Mechanical Specifications

Dimensions: 160mm [L] x 100mm [W] x 38mm [H]

Environmental Specifications

The evaluation board must be stored between -40°C and 100°C. The recommended operating temperature is between 0°C and 55°C.

Electrical Specifications

- 12V Input +/- 15%
- 5V Input +/- 10%
- 3.3V Input +/- 5%
- 12V Hot Swap Maximum Current 10A
- 5V Hot Swap Maximum Current 5A
- 3.3V Hot Swap Maximum Current 8A

Note 1: One of the benefits of using the POWR1220AT8 is that it can control large amounts of power using very small and inexpensive MOSFETs, as demonstrated on the Hercules Evaluation Board. The pre-programmed demonstration designs are carefully designed to operate the MOSFETs inside their safe operating area (SOA). When reprogramming the POWR1220AT8 on the Hercules Evaluation Board the designer must pay attention to the MOSFET's operation limits and SOA. Operation outside specified parameters can damage the MOSFET device and render the Hercules Evaluation Board inoperable.

Note 2: The devices on your evaluation board can be damaged without proper anti-static handling.

How to Use This Guide

This user's guide provides a complete overview of the Hercules Evaluation Board (both Standard and Advanced versions) and details the demonstration designs that are pre-programmed into the devices on the board. The Hercules demonstration designs illustrate many important design techniques and use existing Lattice reference designs that can be adapted to a custom design. All the elements of the Hercules demonstration designs are documented in this user's guide in sufficient detail so that the user can easily use, adapt and modify them to implement a target design.

Use the Quick Start Guide First

To get the most out of this user's guide, first use the Power Manager II Hercules Development Kit Quick Start Guide that comes with the Kit to experiment with the various demonstration designs that are pre-programmed into the POWR1220AT8 and MachXO devices.

Organization of this Guide

This guide begins with a general overview of the Development Kit which covers the contents, key features, software requirements and an overview of the architecture of the Hercules Evaluation Board. Following this overview are a series of Operational Descriptions for each of the Demonstration Designs. Included are descriptions of the functions performed by the MachXO, POWR1220AT8 and external support circuitry, as well as an understanding of the algorithms and/or techniques used in the design. You should consider each of these Operational Description sec-

tions to be a reference design that can be used and/or modified to perform similar functions in your custom application. Because there are some elements that are common to multiple designs there is also a Common Design section in which these 'sub-systems' are grouped. An Operational Description section may refer to the Common Design section where necessary instead of repeating the same material multiple times in different sections.

The Operational portion of this guide is composed of the following sections:

- **Initial Start-up** – The Initial Start-up design is responsible for managing the power-up sequence for the board, initializing the key components and checking that power rails are operating within proper ranges.
- **Hot Swap** – The Hot Swap design implements a controlled current and voltage ramp-up of the power supply in order to comply with the Hot Swap standard.
- **Redundant Supply** – The Redundant Supply (Power-OR'ing) design allows a redundant supply voltage to be used as a power source if the primary supply fails or is removed.
- **Fault Monitoring and Logging** – The Fault Monitoring and Logging design monitors power faults and logs a fault event to an external SPI Flash memory. The Fault Logger design supports HyperTerminal command operation as well as operation via board push-buttons.
- **Trim and Margin Up/Down** – The power supply Trim design controls the 1.2V power supply to better than 1% of its factory set value using an external closed loop trim technique. The power supply Margin Up/Down is used for corner testing and sets the 1.2V power supply to +7.5% (Up) or -7.5% (Down) of its factory value using a closed loop trim function. The closed loop approach continually adjusts the trim DAC value for high accuracy. An open loop trim technique can be selected instead, by the user, via the mode switch. In the open loop technique the trim DAC is set to a fixed (stored) value and is thus less accurate than the closed loop approach.
- **VID Trimming** – In the VID Trimming design the 1.2V DC-DC supply output is fine-tuned based on a VID value provided by the user. A closed loop trimming algorithm is used to keep the output voltage as constant as possible under load variations.
- **Common Elements** – This section contains Operational Descriptions of common functions and support circuits (like the LCD display and closed loop trim functions) used in multiple demonstration designs. These common routines are given here to avoid redundant descriptions in the Operational Description sections.

Included in the Power Manager II Hercules Development Kit

The Hercules Development Kit includes the Power Manager II Hercules Evaluation Board pre-loaded with the Hercules Demo Designs, a USB connector cable for board-to-computer connectivity, a wall adaptor to power the evaluation board, a Quick Start Guide and support documentation on the Lattice web site including this user's guide.

The Hercules Evaluation Board includes the MachXO (LCMX02280C) PLD and the Power Manager II (ispPAC-POWR1220AT8) device. The LCMX02280C is a non-volatile infinitely reconfigurable programmable logic device with up to 2280 Look-Up Tables (LUTs) of logic, up to 271 user I/Os and a variety of advanced features (block memory, PLLs, serial I/Os and a sleep-mode for low power). The POWR1220AT8 is a programmable, highly-integrated board power management device that monitors up to 12 circuit board power supplies, provides up to 20 open-drain digital outputs, and eight DC-DC trim/margin outputs. The device can manage a variety of system power functions (like CPU reset or fault interrupt), using the on-chip 48-macrocell CPLD and four programmable timers.

Standard and Advanced Versions

This user's guide covers both the Standard and Advanced Versions of the Hercules Development Kit. Most features are common to both kits and include the following:

- AC power adapter with 12V output
- USB cable to connect the board to a PC
- The Hercules Evaluation Board with the following circuits:
  - ispPAC-POWR1220AT8 Power Manager II device
  - MachXO 2280 Programmable Logic Device
  - ispMACH® 4000 Programmable Logic Device
  - USB interface for JTAG, I^2C, and SPI
  - Main and external 12V supply connections
  - 12V Hot Swap for Hot Swap Demo
  - 12V OR'ing for Redundant Power Supply Demo
  - 1.2V DC-DC supply for Margin, Trim, and VID Demos
  - SPI memory for Fault Logging Demo
  - Three-digit LCD display
  - Various LEDs and switches for status and control

The only difference between the Standard and Advanced versions is that the Advanced Evaluation Board is populated with the following circuits:

- 5V Hot Swap Circuit
- 3.3V Hot Swap Circuit
- 2.5V DC-DC supply with Enable and Trim control
- 3.3V DC-DC supply with Enable and Trim control
- cPCI connector to demonstrate Hot Swapping in a backplane
- cPCIe power connector to demonstrate Hot Swapping in a backplane
- MachXO Mini Evaluation Board Connector for board-to-board connections
- Phoenix style screw connectors for off-board loading
- Additional LEDs for status indication

Both the Standard and Advanced boards come pre-programmed with exactly the same set of Demonstration Designs. Advanced Demonstration Designs with documentation will be updated on the Lattice web site as they become available.

Note: Static electricity can severely shorten the lifespan of electronic components. Please handle the kit components carefully and follow the environmental guidelines shown later in this guide.

Features

The Power Manager II Hercules Development Kit includes:

- **Power Manager II Hercules Evaluation Board** – The Hercules Evaluation Board is a 3U form factor (100mm x 160mm) that features the following on-board components and circuits:

  Standard and Advanced Version Evaluation Board Features:

  – **Integrated Circuits**
  - POWR1220AT8 – U3
  - MachXO2280 with embedded LatticeMico8™ (LCMX02280C) – U4

- ispMACH 4000 CPLD for JTAG Management (LC4064, not user programmable) – U2
- USB Interface Future (FT2232D) and STMicro Serial Memory (M93C46) – U13
- 2MBit SPI Serial Flash Memory- STMicro (M25PE20) – U5

– **Power Supplies, Connections and Switches**
- External 12V Supply Connector – J2
- Main 12V DC Supply Connector – J3
- Main Supply Toggle Switch – SW1
- External 3.3V DC Supply Connector – J9
- 1.2V DC-DC Supply – DCM1
- 12V Hot Swap Support Circuit
- 12V OR'ing Support Circuit

– **LEDs and LCD Display**
- Supply Status LEDs – D1-D7
- 3-Digit LCD Display – U6
- Interface LEDs – D12-D14
- CPLD Output LEDs – D8-D11

– **Switches, Sliders, Buttons and Jumpers**
- VID 8-position DIP Switch – SW5
- USB - JTAG/I2C Interface Select 4-position DIP Switch – SW2
- Slide Potentiometer – R22
- Mode Button – SW3
- Reset Button – SW4
- VMON7, VMON8 Jumpers – J5 and J6
- JTAG Jumpers – J10, J11, J12, J15

– **Connectors and Interfaces**
- USB B-mini Connector – J14
- JTAG Interface – J18
- 10x10 Through-Hole User Prototype Area
- Probe and Test Points

Advanced Version Only Evaluation Board Features:

– **Power Supplies, Connections and Switches**
- 2.5V DC-DC Supply – DCM2
- 3.3V DC-DC Supply – DCM3
- 5V Hot Swap Circuit – Q5, DCM3 and associated circuitry
- 3.3V Hot Swap Circuit – Q4, DCM2 and associated circuitry

– **Connectors and Interfaces**
- MachXO Mini Evaluation Board Interface Header/Connector – J16
- cPCI P1/J1 Connector (Power and Interface Signals)
- cPCIe XP1/XJ1 Connector (Power only)
- Phoenix Style Screw Connector for Power Loading – J4
- MachXO Mini Evaluation Board Connector

- **Pre-Programmed Demos** – The Hercules Evaluation Kit comes pre-loaded with several demo designs that implement a wide variety of Power Supply Margin, Trimming, Monitoring, Fault Logging and Voltage Control functions.
- **Advanced Version Demonstrations (Advanced Version Only)** – Additional advanced demonstration designs that demonstrate 5.5V and 3.3V Power Hot Swap can be found on the Hercules support web pages.
- **USB Connector Cable** – The Hercules Evaluation Board includes a mini-B USB socket that can be used to connect the Hercules Evaluation Board to a host PC using the USB Connector Cable. The USB channel provides a programming interface to the Power Manager II and MachXO devices. This interface can also be used as a con-

trol and status port for the MachXO device when running the advanced demonstration and reference designs. It can also be used as a general purpose interface for user designs.

- **Quick Start Guide** – Provides information on connecting the Hercules Evaluation Board and running the pre-loaded support demo.
- **Hercules Development Kit Web Page** – The Hercules Development Kit web page provides access to the latest documentation, demo designs, reference designs, support utilities, and drivers for the kit.
- **Hercules Development Kit User's Guide** – The contents of this user's guide include demo operation, programming instructions, top-level functional descriptions of the evaluation board, descriptions of the on-board connectors, switches and a complete set of schematics of the Hercules Evaluation Board.

Software Requirements

The pre-loaded demonstrations on the Hercules Evaluation Board will run without the need to install any special software. All control and status information is available on the board using switches, push-buttons, LEDs and the LED display.

In order to modify the pre-loaded demonstrations, load reference designs or advanced designs or to create custom designs the following Lattice software tools are needed:

- **PAC-Designer®** – To modify pre-loaded demonstrations, advanced demonstrations or reference designs for the Power Manager II device.
- **ispLEVER®** – To modify pre-loaded demonstrations, advanced demonstrations or reference designs for the MachXO device.
- **ispLEVER Classic** – To modify the ispMACH 4000 JTAG MUX. Only recommended for extremely advanced users. *Note: Incorrect modification of the ispMACH 4000 JTAG MUX can make it impossible to program the Power Manager II and MachXO devices without significant modifications to the board.*
- **ispVM™ (Windows)** – To download programming files for the Power Manager II, USB drives or MachXO device.

Hercules Evaluation Board

The Hercules Evaluation Board has been designed to allow the user to quickly learn, evaluate, develop and test power supply control designs using the POWR1220AT8 in stand-alone applications or with support from a MachXO CPLD (the MachXO provides higher-level control functions for more complex power supply management applications). These two devices form the "core" of the Hercules Evaluation Board with the rest of the components providing support functions.

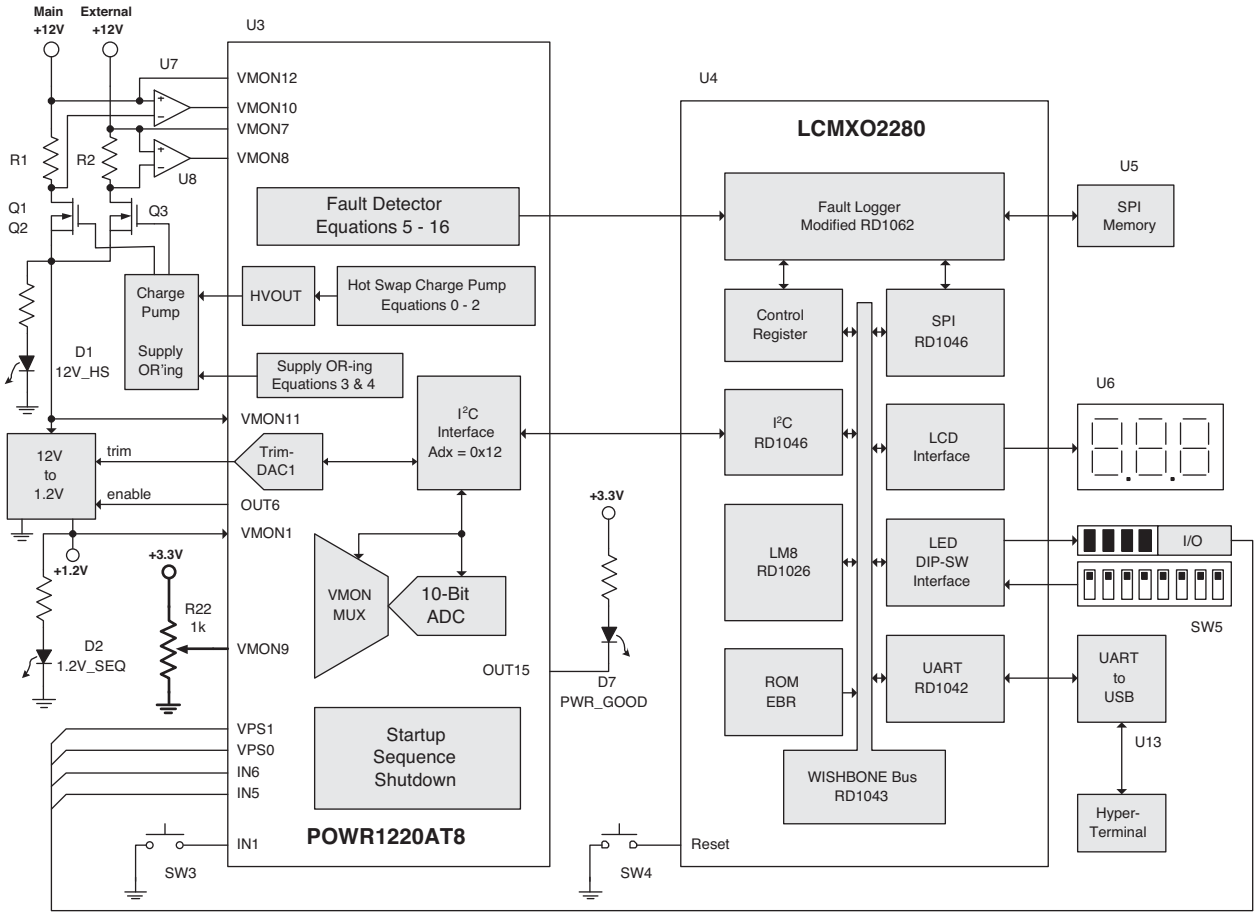
Hercules Evaluation Board Architecture Overview

In this section, the Hercules Evaluation Board is described at the top level. It identifies each of the key hardware components on the board and briefly describes their main functions. Because many of these elements is described in much more detail in the various Demonstration Design Operational Description sections, this section is just a short overview. Refer to the appropriate Operational Description Section for more details on each block.

The Hercules Evaluation Board is provided in one of two editions, Standard or Advanced. The bulk of the components and capabilities are common to each board so the description of the board will mostly be the same. Features of the Advanced board will be noted separately. All other features can be assumed to be common to each edition.

A block diagram of the overall system architecture of the Hercules Evaluation Board is shown in Figure 1. This illustrates how the various support circuits are connected to the power management ICs of the Hercules Evaluation Board. The POWR1220AT8 (U3) and the MachXO2280 (U4) are programmed to contain the main system functions on the board. The POWR1220AT8 is configured with sequence, exceptions, logic equations, and more to support all of the Demo Designs. The MachXO device is configured as a system-on-chip (SOC) based on a powerful embedded microcontroller, the LatticeMico8. This system is based on several existing reference designs for the MachXO. Each of the key hardware blocks is described in more detail below.

Figure 1. Hercules Evaluation Board Block Diagram



Power Manger II ispPAC-POWR1220AT8

VMON12 is used to monitor the Main 12V (J3) input supply voltage level. VMON10 monitors the Main 12V input supply high-side current. This current sensing circuit is used for Hot Swap, Power-OR'ing, and Over-current. VMON7 is used to monitor the external EXT 12V (J2) supply voltage level. VMON8 is connected to the EXT 12V I high-side current sensing circuit. VMON11 is used to sense the 12V supply at the hold-off capacitor and load side of the Hot Swap MOSFETs. VMON9 is used to sense the voltage on the slide potentiometer. OUT15 is used to control the power good LED. VPS1, VPS0, IN6 and IN5 are additional signals between the MachXO2280 and POWR1220AT8 available for custom designs Push-button mode switch SW3 connects to IN1 and the test results can be read by the MachXO2280 via the I^2C interface.

HVOUT is used to drive the external Charge-Pump circuit. The Charge-Pump circuit raises the gate voltage on FETs Q1-Q3 to eliminate any FET source to drain voltage drop from the 12V supply inputs. Trim-DAC1 is used for fine adjustments to the 12V to 1.2V DC-DC converter via its trim input. An enable signal is used to enable the DC-DC converter during the start-up sequence. The I^2C interface allows the MachXO2280 to read and write the status signals and control registers within the POWR1220AT8. This is useful during voltage monitoring, margining and closed-loop trim functions. The 10-bit ADC and VMON MUX blocks select and sense the various voltage monitor inputs, the result of which can be read out via the I^2C interface.

Logic equations within the POWR1220AT8 are used to control various outputs from the device. The equations are shown as listings in this document. Listing 1 shows the POWR1220AT8 start-up sequence. Listings 2 and 3 show the external power MOSFET gate charge pump equations. Listing 4 shows the control of the Hot Swap features via HVOUT. Listing 5 shows the control of the Power Supply OR'ing support circuitry switching the voltage input when

one of the 12V supply inputs is disconnected. Listing Equations 6 through 9 show the control the Fault Logging sequence which provides the fault indication and logged status signals to the MachXO2280.

The Start-up Sequence and Shutdown block contain the sequencing logic used to control board power start-up and shutdown timing. The specifics of these sequences will be explained later in the document during the various Demo Description sections.

A complete description of the device can be found in the ispPAC-POWR1220AT8 Data Sheet.

MachXO – LCMO2280C

The LatticeMico8 is an 8-bit microcontroller used for board-level control and sequencing (see RD1026 LatticeMico8 Microcontroller User's Guide). The Hercules Demo Design uses it to read the various user inputs (switches, buttons and UART commands) and provides status information on the status outputs (LEDs, LCD display and UART status). The LatticeMico8 is also used during demos to read and write to the SPI memory, access the POWR1220AT8 via the I^2C interface or the additional interface signals (VP1-0 and IN5-6).

The WISHBONE interface is used to connect the LatticeMico8 to the various internal peripherals (UART, SPI, I^2C, etc.) and memory (ROM EBR). See RD1043, LatticeMico8 to WISHBONE Interface Adapter Reference Design. WISHBONE is an open-source industry standard interface used on many Lattice reference designs to simplify processor-based design.

The I^2C interface is an implementation of the industry standard serial interface used to communicate between low bandwidth peripherals and memory. In the Hercules demo designs the LatticeMico8 uses this block to read and write to the POWR1220AT8 status and command registers. See RD1046, I^2C Master with WISHBONE Bus Interface Reference Design.

The UART block is an implementation of the industry standard serial interface. The Hercules demo design use the UART to read user commands and write status results via a hyper-terminal emulator on a PC. See RD1042, WISHBONE UART Reference Design.

The Fault Logger reference design (see RD1062, Three-Wire Power Supply Fault Logging Using Lattice Programmable Logic) is used to capture and store fault event information into the SPI memory when a fault is detected by the POWR1220AT8. The reference design is implemented in both the POWR1220AT8 and a Lattice PLD. In the Hercules demonstration designs the MachXO2280 implements the PLD portion of the reference design. The reference design uses the POWR1220AT8 to detect fault events which are sent to the MachXO2280. The MachXO2280 then logs the fault event in the SPI memory.

Other logic blocks within the MachXO2280 do not use reference designs. These blocks implement simple interface functions to control buttons and switches. The demo design code can be used as a starting point for implementing similar functions in a custom design.

A complete description of the device can be found in the MachXO Family Data Sheet.

MachXO Peripherals

Many of the board's external support components are peripherals of the MachXO and are used to provide system inputs and status outputs.

- **LCD Display (U6)** – The LCD display provides numeric outputs showing information such as voltage levels and fault counts.
- **VID-DIP Switches (SW5)** – The VID-DIP switches provide inputs to activate a particular Demo Design and provide control to that demo such as voltage levels for the trimming and margining.
- **Push-button Switch (SW4)** – This push-button switch provides an asynchronous reset to the SOC.
- **SPI Memory (U5)** – The SPI memory is used by the MachXO to log power supply fault information during operation of the pre-loaded demonstration program.

- **USB Interface (U13)** – The USB interface is used to reconfigure the MachXO and POWR1220AT8 device or to act as a user interface in custom designs.
- **LEDs (D8-D11)** – These four LEDs can be used by the MachXO to indicate demo status.

Note: Several reference designs are used in constructing the MachXO demonstration design. These are indicated in the associated block by listing the reference design designation (for example, RD1044 is the reference design designator for the SPI reference design). These designs are available free of charge on the Lattice web site and can be easily downloaded and modified for testing or to be included in a custom design. Refer to the detailed descriptions in the Reference Design section of this user's guide for more details.

Power Manager II Support Circuits

The Power Manager II device has several external support circuits used in the demonstration design. These circuits serve as useful illustrations of typical design techniques used in power supply management applications.

- **1.2V Trim Circuit** – The 1.2V Trim Circuit is a closed loop circuit used by the POWR1220AT8 to control the voltage trim feature on the DC-DC converter.
- **Charge Pump Circuit** – The Charge Pump Circuit is used by the POWR1220AT8 to boost the gate voltage on the power FETs (Q1, Q2 and Q3).
- **12V Power Supply OR'ing** – The 12V Power Supply OR'ing circuit is used to monitor input voltages and select the 12V supply from either the EXT 12V (J2) or the Main 12V (J3) by controlling the gate voltage on power FETs Q2 and Q3.
- **Current Sensing Circuit** – The Current Sensing Circuit (implemented with resistors R1 and R5 and current sense amplifiers U7 and U8) are used by the 12V Hot Swap control function to sense current during board insertion so inrush current can be controlled.
- **Slider Potentiometer** – The Slider Potentiometer Circuit attached to Voltage Monitor Input 9 (VMON9) is used to create an 0 to 3.3V adjustable voltage level to demonstrate an over/under voltage during the fault monitoring command.
- **Push-button Switch (SW3)** – This push-button switch is read via I^2C by the MachXO during various demos such as Fault Logger, Margin, Trim, and VID.

LED Power Indicators

The Hercules Evaluation Board includes several LEDs to monitor power levels.

- **D1** – Power indicator for 12V hot swap or an enabled 12V input from J2, J3 or ePCIx_12V
- **D2** – Power indicator for 1.2V on board DC-DC supply
- **D3** – 5V_HS power indicator for external 5V hot swap voltages
- **D4** – 3.3V_HS power indicator for external 3.3V hot swap voltages
- **D5** – Power indicator for external 12V from ePCI_+12V
- **D6** – Power indicator for external -12V from ePCI_-12V
- **D7** – Power good indicator from the POWR1220AT8
- **D13** – Power indicator from USB port J14
- **D14** – Power indicator for enabled on-board 3.3V

Test Points

There are several test points available on the top of the Hercules Evaluation Board. These test points are indicated with a white circle around the thru hole via and have a white test point label adjacent to them. Many of these signal points will be discussed later in this document when specific operations and sections of the schematic are described. Refer to the following list of test points when you wish to probe specific signals:

- **3V_A** – Current monitor for 3.3V supply; scale = 4A/V (VMON4)
- **5V_A** – Current monitor for 5V supply; scale = 20A/V (VMON6)
- **12V_A** – Current monitor for primary +12V supply; scale = 1A/V (Low) or 10A/V (High) (VMON10)
- **CHG_P** – Charge pump output (HVOUT1)
- **12V_OR** – Primary +12V supply enable (OUT12)
- **EXT_OR** – External +12V supply enable (OUT13)
- **12V_SDN** – +12V Shut down control (OUT11)
- **3V_IN** – Voltage monitor +3.3V input supply (VMON3)
- **5V_IN** – Voltage monitor +5V input supply (VMON5)
- **12V_IN** – Voltage monitor +12V primary input supply (VMON12 via resistor divider)
- **3V_HS** – +3.3V Hot Swapped supply (VMON7)
- **5V_HS** – +5V Hot Swapped supply (VMON8)
- **12V_HS** – +12V Hot Swapped supply (VMON11 via resistor divider)

Note: Make sure you use a high impedance probe to reduce signal loading and be careful not to short any pins together as this could damage the board's circuits. Check the schematic prior to probing and avoid loading the base of any discrete transistor as even a very small additional load can change circuit response.

Hercules Evaluation Board Photos

Photographs of the top and bottom of the standard and advanced Hercules Evaluation Boards are shown below. Component location references are relative to the top of the board with the silk screen text in the readable orientation (as shown in the photo). The major differences between the board can be seen to be the inclusion of: the cPCI and cPCIe connectors; DCM2 and DCM3, and hot swap circuits; the external voltage connector (J4); and the MachXO expansion header (J16).

Figure 2. Hercules Standard Version Evaluation Board Photo, Top Side

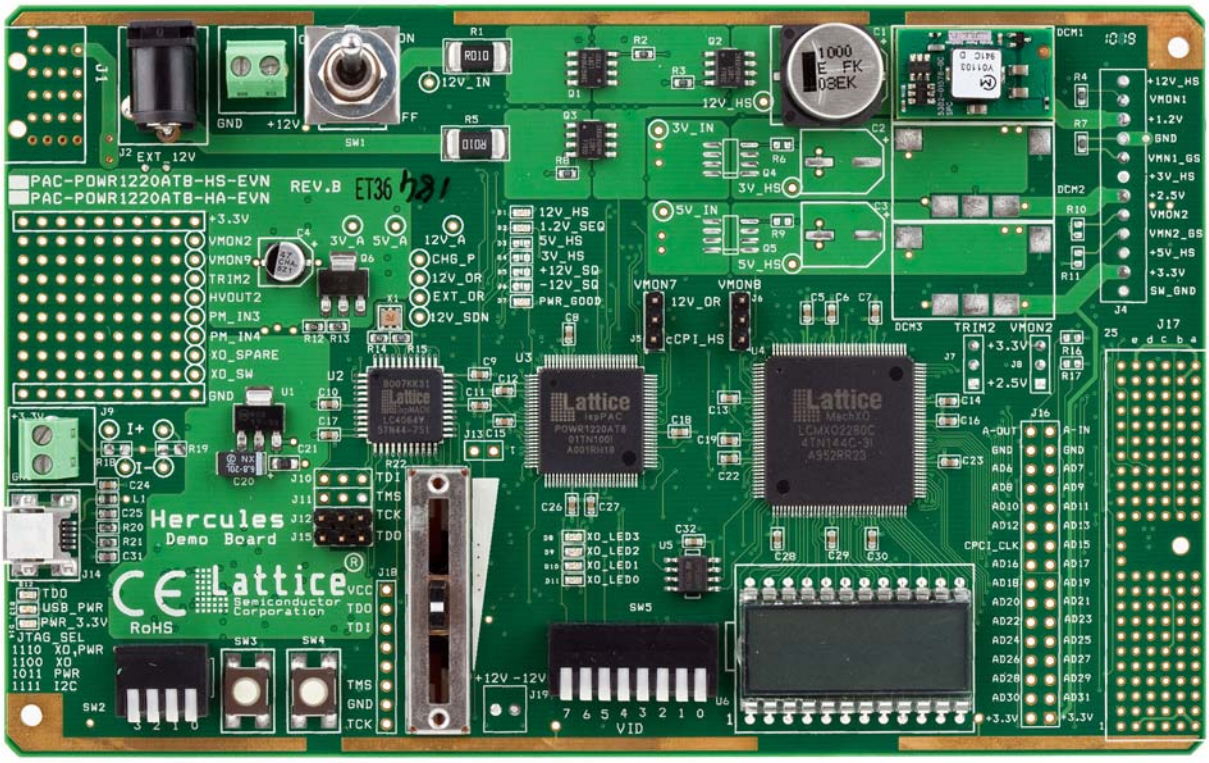


Figure 3. Hercules Standard Version Evaluation Board Photo, Bottom Side

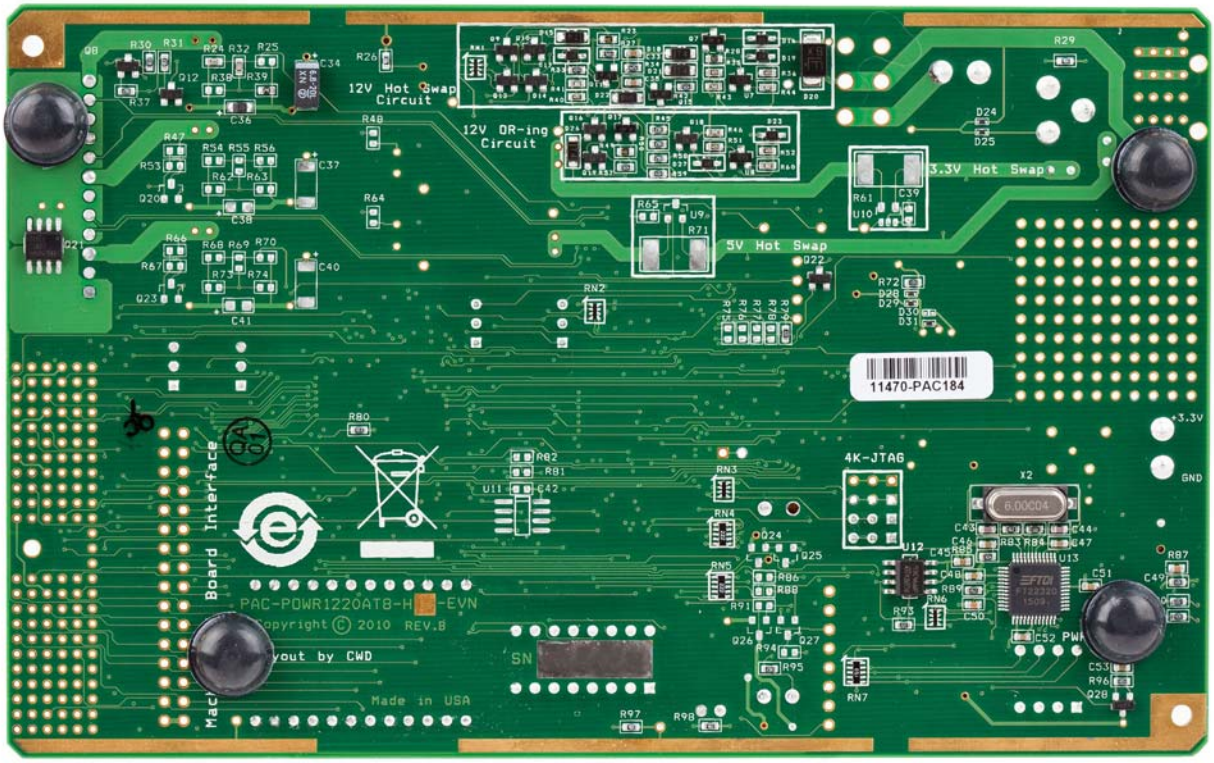


Figure 4. Hercules Advanced Version Evaluation Board Photo, Top Side

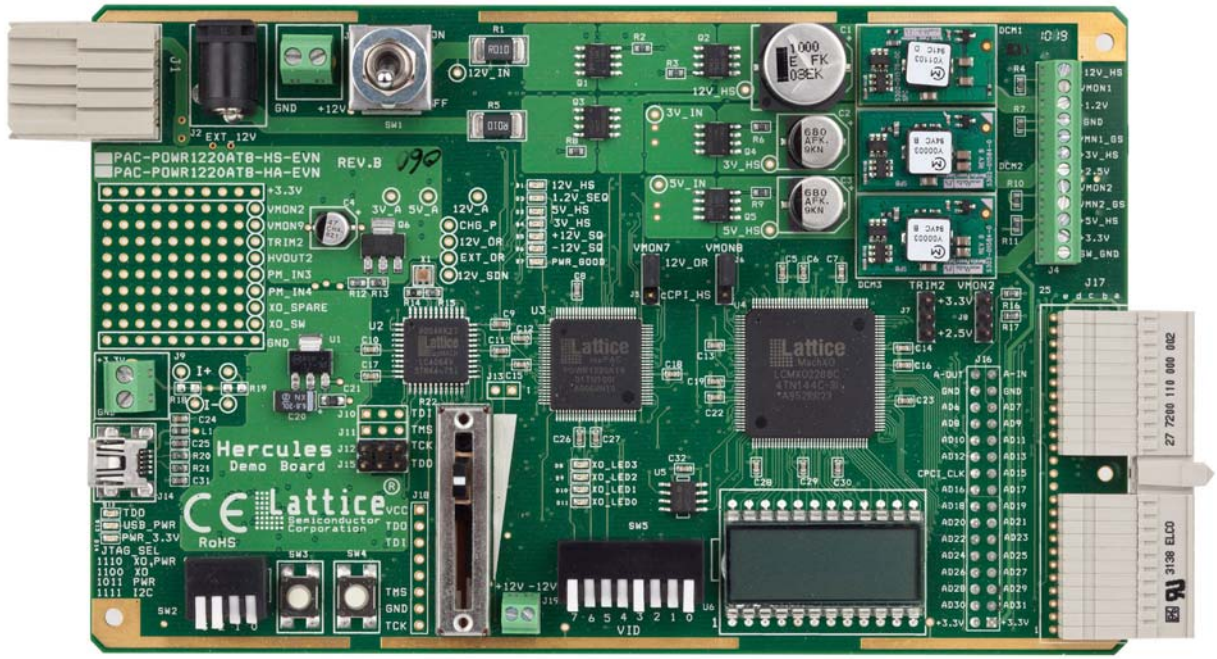
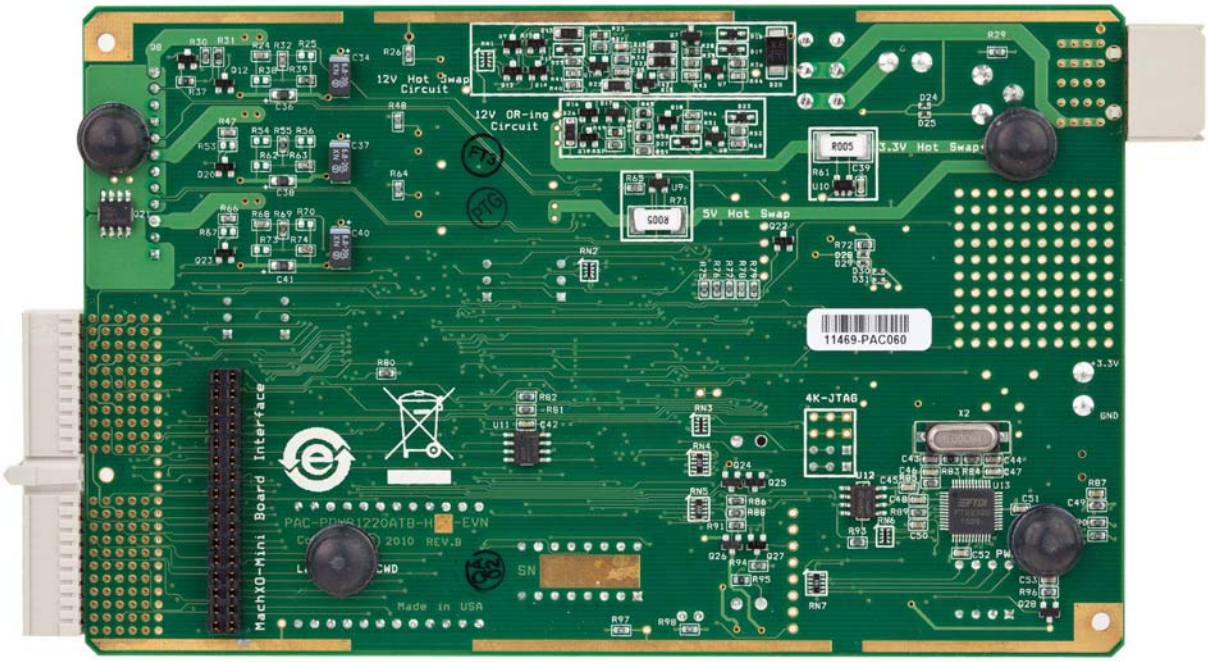


Figure 5. Hercules Advanced Version Evaluation Board Photo, Bottom Side



Hercules Evaluation Board Advanced Version Features

The advanced version of the board includes additional components and connectors that are not included on the standard version. These features are described in this section.

ePCIx 3V and 5V Power Supply and Hot Swap Circuitry
The advanced version supports two extra power supply sections. Hot swappable external 5V and 3.3V input supplies are supported and converted to 3.3V and 2.5V by two DC-DC converters (DCM2 and DCM3). Support circuitry is included to monitor current and voltage of the hot swappable 5V and 3.3V inputs along with voltage trimming of the associated 3.3V and 2.5V supplies. Extra LEDs (D3 and D4) are used to indicate the presence of the 3.3V and 5V at the hot swap MOSFET's power output. Two extra jumpers (J7 and J8) are included to select between 3.3V or 2.5V trimming since only a single trim output and VMON input are available.

RD1057, 5V and 3.3V Hot Swap Controller Reference Design describes the operation of the hot swap functions for these two supplies in detail. The reference design includes schematics, Power Manager II design files as well as simulation and verification results.

cPCI Interface
Support for cPCI is included on the advanced version of the evaluation board. Connector J17 is mounted on the right side of the board and includes both power and signal interfaces.

cPCIe Power Supply Interface
Support for cPCIe supply voltages is included on the advanced version of the evaluation board. Connector J1 is mounted on the left side of the board. It does not include the signal interface.

CAUTION: J1 is not keyed for a single orientation on the motherboard and can be inserted in either direction. The daughter card must be inserted so that the Hercules Evaluation Board does not overhang the side of the mother board.

Power Supply and Voltage Monitor Header
The advanced version of the board includes a 12 terminal twist header (J4) to make it easier to connect probes and loads to several of the signals on the board. The 1.2V, 2.5V and 3.3V supply outputs are available as well as VMON1 and VMON2.

MachXO Mini Evaluation Board Interface
The MachXO Mini Evaluation Board compatible expansion header provides 32 signals that can be used for custom designs with the MachXO device on the advanced version of the evaluation board. The header (J16) can be used to connect to other Lattice boards that conform to the MachXO Mini format. This allows more complex designs to be created by connecting multiple boards together. Visit the Lattice web site for a complete list of MachXO Mini compatible evaluation boards.

Operational Description – Introduction

The Operational Description section contains the detailed 'how the design works' portion of this user's guide. Each sub-section covers a key design element (or sub-system) that can be considered a stand-alone piece of the Hercules design. These sub-systems work together to implement the complete Hercules design, but the overall design is more easily understood by describing each individual piece.

Some functions used in a sub-system are common to other sub-systems. These functions will not be described multiple times, but are collected in a separate "Common Elements" section and are referenced by the appropriate section where required.

The sub-systems described in this section are:

- **Initial Start-up** – Powers up and initializes the board.
- **Hot Swap** – Implements a controlled current and voltage ramp-up of the power supply in order to comply with the Hot Swap standard.

- **Redundant Supply** – Allows a redundant supply voltage to be used as a power source if the primary supply fails or is removed.
- **Fault Monitoring and Logging** – Monitors power faults and logs a fault event to the SPI memory.
- **Trim and Margin Up/Down** – Controls the 1.2V power supply using either an open loop or external closed loop trim technique to trim (control voltage to better than 1%) or margin (adjust voltage up or down by 7.5%) the supply.
- **VID Trimming** – Fine tunes the 1.2V DC-DC supply output based on a digital VID value.
- **Shutdown** – A board-level shutdown sequence is initiated when both supplies are off or an over-current event occurs.
- **Common Elements** – This section contains Operational Descriptions of common functions and support circuits (such as the LCD display and closed loop trim functions) used in multiple demonstration designs. These common routines are given here to avoid redundant descriptions in the Operational Description sections.

Operational Description – Initial Start-Up

This section describes the Initial start-up process of the demonstration design. It shows how this portion of the design works in sufficient detail so that it can be used as a reference design and easily modified, if required, for a target application. The section is organized around the following key topics:

- Overview of the Start-up process
- Key Components, Functions and Algorithm of the Start-up Design
- Implementation of the Start-up Function

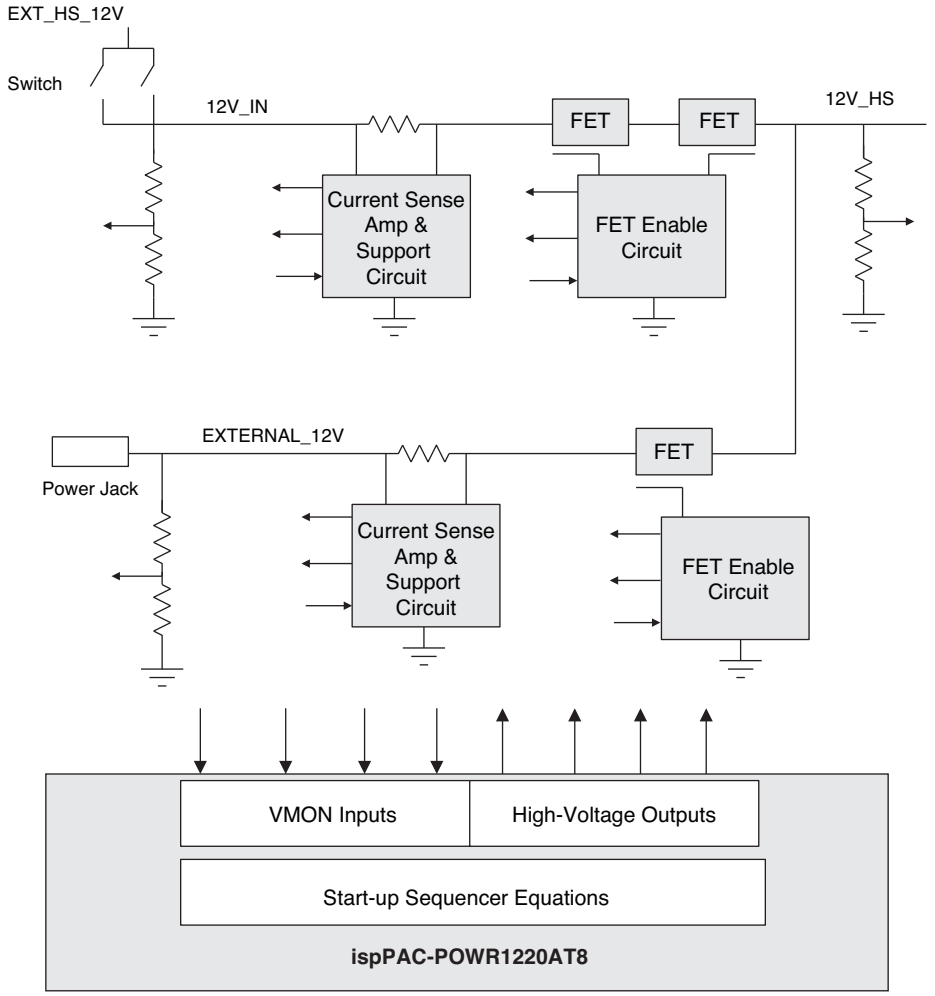
Overview of the Start-up Process

The process for starting up the on-board power supplies on the Hercules Evaluation Board is primarily controlled by the POWR1220AT8, but first the power to the POWR1220AT8 and other critical circuits (like the MachXO and SPI memory) needs to be available. Since the power to these circuits is shared among several functions, the process and implementation of applying power to them will be described in detail in the Common Elements Operational Description section later in this guide. For the purposes of this section we will assume power is already up and the critical circuits are operational.

Key Components, Functions and Algorithm of the Start-up Design

The main component responsible for the board start-up process is the POWR1220AT8. It uses several external support circuits (shown in Figure 6) to monitor and select one of the 12V power supply inputs, either EXT_HS_12V (from a user-selectable switch) or EXTERNAL_12V (from the wall adaptor jack) to develop the main on-board supply voltage 12V_HS. The POWR1220AT8 Voltage Monitor (VMON) inputs monitor the 12V inputs, the on-board 12V output and current sense voltages via its VMON inputs. Its high voltage outputs enable the MOSFET circuits that switch the off-board voltages onto the board in a controlled manner.

Figure 6. Main 12V Power Control Subsystem



Implementation of the Start-up Function

The first three steps of the Sequence State Machine within the POWR1220AT8, as shown in Listing 1, control the board start-up process. In Step 0, the POWR1220AT8 resets the main power supply control outputs OUT9_Ext_LowCurrent_ENA, OUT10_LowCurrent_ENA, OUT11_Shut_12V_Down, and OUT15_PWR_GOOD. The function of these outputs will be explained in detail in the appropriate Operation Description sections later in this document. Step 0 then waits for the AGOOD signal, an internal signal to the POWR1220AT8 that indicates that it is powered up successfully and that its analog calibration is complete. Before the sequence begins, the INPUT_12V_MIN or V7_EXT_IN12V_MIN signal must also be high enough to indicate that board power is available. These inputs are monitored by the POWR1220AT8 on its VMON pins.

Listing 1. POWR1220AT8 Start-up Sequence for the Hercules Demonstration Design

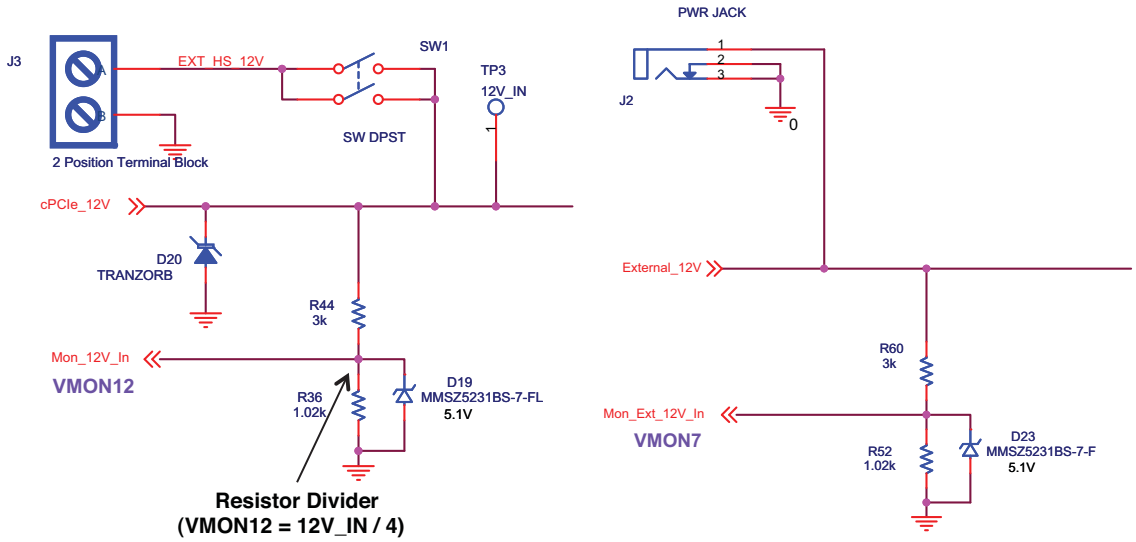
Step 0 Wait for AGOOD AND (INPUT_12V_MIN OR V7_EXT_IN_12V_MIN)
OUT9_Ext_LowCurrent_ENA = 1, OUT10_LowCurrent_ENA = 1,
OUT11_Shut_12V_Down = 1, OUT15_PWR_GOOD = 1, DUMP_STATUSn = 1

Step 1 Wait for 15.36ms using timer 1

Step 2 OUT11_Shut_12V_Down = 0

The voltage monitor setting for both these inputs is set in the Hercules Demonstration Design file U3_POWR1220_Demo.PAC (Demo_Hercules/Project/POWR1220) and they are shown, along with the relevant portion of the Hercules schematic, in Figure 7.

Figure 7. INPUT_12V and V7_EXT_IN_12V_MIN Voltage Monitor Inputs



V7_EXT_IN_12V_MIN = 2.407V (VMON7_B)

INPUT_12V_MIN = 2.407V (VMON 12)

The voltage at the VMON input will be approximately 3V or one-fourth of the 12V supply (due to the 1K Ohm and 3K Ohm values) and will stay below the 5.9V data sheet maximum. The minimum voltage for both VMONs is set at 2.407V. Both VMON inputs are also protected from a voltage that might exceed the operational specifications of the POWR1220AT8 by using diodes (D19 and D23) with a Zener voltage of 5.1V. This protection is required if the 12V supply is unregulated or contains noise spikes in excess of 23.6V. This protection is recommended to guard the POWR1220AT8 VMON inputs against board manufacturing errors such as incorrect divider values or missing divider components.

Step 0 will thus exit only when one of the +12V power sources is at 9.628V (2.407V * 4), which is high enough to begin the rest of the sequence. In Step 1 the state machine waits for 15.36ms before turning on the primary 12V supply. This gives the supply some time to become stable prior to enabling it. It also allows for contact bounce if SW1 is used to connect the external 12V supply from the terminal block. SW1 is provided to simulate the act of hot-swapping so the delay in this step covers any source of unstable 12V supply. After the delay, Step 2 sets the output OUT11_Shut_12V_Down to '0' which turns off Q14. This allows the hot-swap charge pump circuit to operate. The charge pump circuit drives the gate of the MOSFET (Q1) that connects the 12V input supply to the rest of the circuitry. The charge pump circuit is described in detail in the Hot-Swap section of this document.

At this point the main elements of the board have been enabled. The POWR1220AT8, the MachXO, SPI memory and USB converter are powered up. (They may not be initialized yet and this aspect will be covered later). The next steps are related to the hot swap implementation, so these steps will be described in more detail in the Operational Description for Hot Swap.

Operational Description – Hot Swap

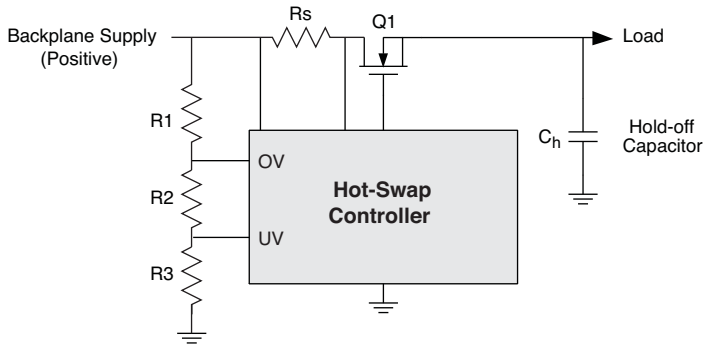
This section describes the hot swap function of the demonstration design. It shows how this portion of the design works in sufficient detail so that it can be used as a reference design and easily modified, if required, for a target application. The section is organized around the following key topics:

- Overview of the Hot Swap Function
- Key Components, Functions and Algorithm of the Hot Swap Design
- Implementation of the Hot Swap Subsystem in the Hercules Design
- Voltage Pump with Gate Driver Circuit
- Current Sense Amp Circuit
- POWR1220AT8 Hot Swap Controller
- Hot Swap Demonstration Results

Overview of the Hot Swap Function

Hot swap controllers limit the inrush current when a circuit board is plugged into a live backplane. In addition, these devices offer over-current, over-voltage and under-voltage protection to the circuit board. Figure 8 shows the block diagram of a typical hot-swap controller. The resistor Rs is the current sense resistor. The MOSFET Q1 is used to control the current through the circuit. The resistors R1, R2 and R3 are used to monitor the backplane voltage. The hold-off capacitor, Ch, is used to provide power to the board when the backplane voltage briefly drops below the low operating voltage (under voltage) threshold (say, for less than 10 ms).

Figure 8. Example Hot Swap Controller Subsystem

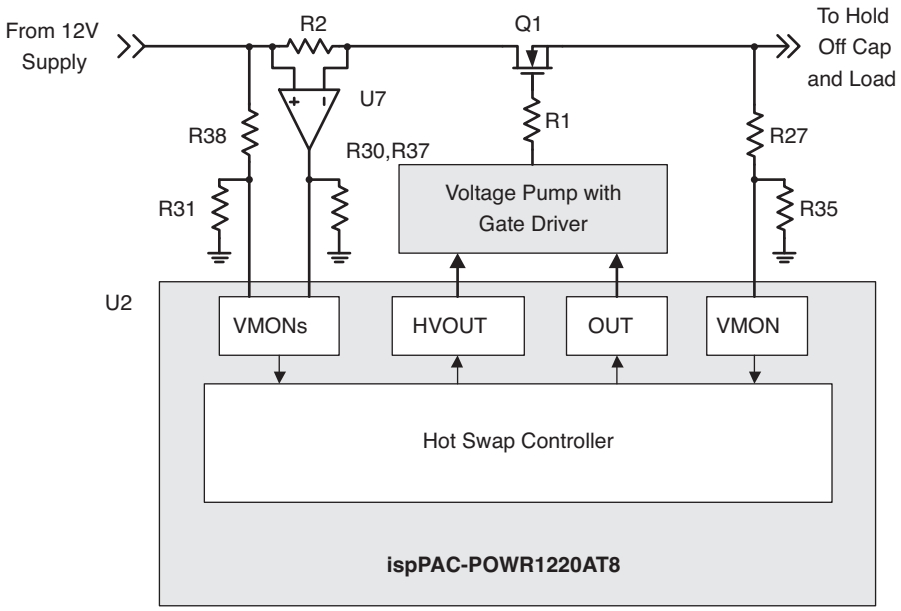


When the card is plugged into the live backplane, the hold-off capacitor, Ch, begins to charge, drawing a large amount of current from the backplane. The hot swap controller limits the in-rush current by controlling the voltage applied to the MOSFET gate. It monitors the current by using the voltage across current sense resistor Rs as feedback. The MOSFET will operate in this current limit mode until the capacitor Ch is fully charged. During the brief capacitor charging period, the in-rush current drawn from the backplane often can be significantly higher than the normal board operating current. As a result, the backplane voltage can dip below the under voltage threshold momentarily for the other cards attached to the backplane. The charge stored in the capacitor, Ch, keeps the card operating during this brief voltage dip period. Hot swap controllers are required to isolate the board from the backplane in case it develops a short circuit fault during operation. The hot swap controller monitors the current through the sense resistor Rs and when the current (voltage across the resistor Rs) increases beyond its threshold value, the hot swap controller turns the MOSFET off. Another monitored fault is if the backplane voltage drops below the under-voltage threshold or goes above the over-voltage threshold, the power to the load is shut off by turning the MOSFET off.

Key Components, Functions and Algorithm of the Hot Swap Design

Figure 9 shows the top-level view of a typical 12V Hot-Swap design using the POWR1220AT8 along with the key support circuits. Q1 is a power MOSFET that functions as the 12V power switch; when it is on, 12V is supplied to the rest of the board and when it is off the rest of the board is disconnected from the 12V source. Q1 must be fully turned-on in order to minimize the voltage drop from drain to source. This means the gate voltage needs to be driven to a level of about 18V. This will assure there is enough gate-to-source bias on top of the 12V level of the source and load. As the HVOUT pins can only drive to 12V (10V for non -02 devices) a voltage-pump circuit is used to generate the 18V. This circuit essentially adds the HVOUT voltage to the incoming 12V, minus a few diode drops, to provide the roughly 18V required to bias on Q1. Following Figure 9 is a description of each of the three key blocks (Voltage Pump with Gate Driver Circuit, Current Sense Amp Circuit and the POWR1220AT8 Hot Swap Controller) in the Hercules Hot Swap implementation.

Figure 9. 12V Hot Swap Design High-Level Block Diagram



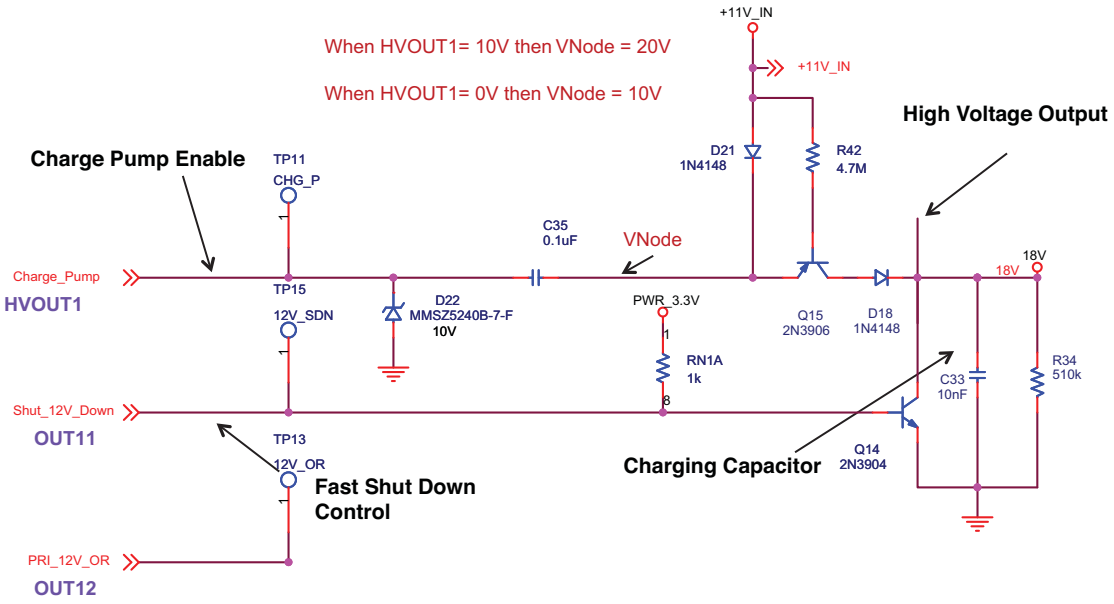
Implementation of the Hot Swap Subsystem in the Hercules Design

The implementation of the Hot-Swap subsystem in the Hercules design is very similar to the overview provided in the previous section. The key elements of the Hot-Swap design on the Hercules Evaluation Board are the POWR1220AT8 functioning as the Hot Swap controller, the charge pump with gate driver circuit, the current sense amp circuit and the voltage monitor support circuits.

Voltage Pump with Gate Driver Circuit on the Hercules Evaluation Board

Figure 10 shows the circuit used in the Hercules design to drive the power MOSFET (Q1 in Figure 9) that connects the external 12V power supply to the rest of the system. The HVOUT1 signal of the POWR1220AT8 is programmed to oscillate at about 20kHz swinging from zero to 10V with about an 80% duty cycle. When HVOUT1 is low, D21 charges C35 to 10V from the 11V supply and VNode will be at 10V. When HVOUT goes high, the voltage at the near side of C35 is elevated to 10V and the resulting voltage at the other side of C35 (VNode) is pumped up to 20V. The emitter of Q15 is also raised to 20V. The 20V can now bias on Q15 and D18 to charge up C33. Q15 is a bipolar PNP that functions as a voltage controlled rectifier. It only passes current when the emitter is higher than the 11V supply. After several cycles from HVOUT1 the voltage on C33 will be enough to bias the gate of the MOSFET (Q1 in Figure 9) and turn it on. The power MOSFET gate resistor R1 in Figure 9 is located close to the gate of Q1 to prevent parasitic oscillations on Q1.

Figure 10. Hercules Charge Pump Circuit Implementation



The fast shut down circuitry controlled by the POWR1220AT8 open drain output OUT11 enables or disables the supply. When OUT11 is low, Q14 will turn off allowing C33 to charge. When C33 is charged it will turn on the MOSFET connecting the 12V supply to the rest of the system. When OUT11 is high, Q14 will turn on keeping C33 from charging. This will keep the MOSFET off, disconnecting the 12V supply from the rest of the system.

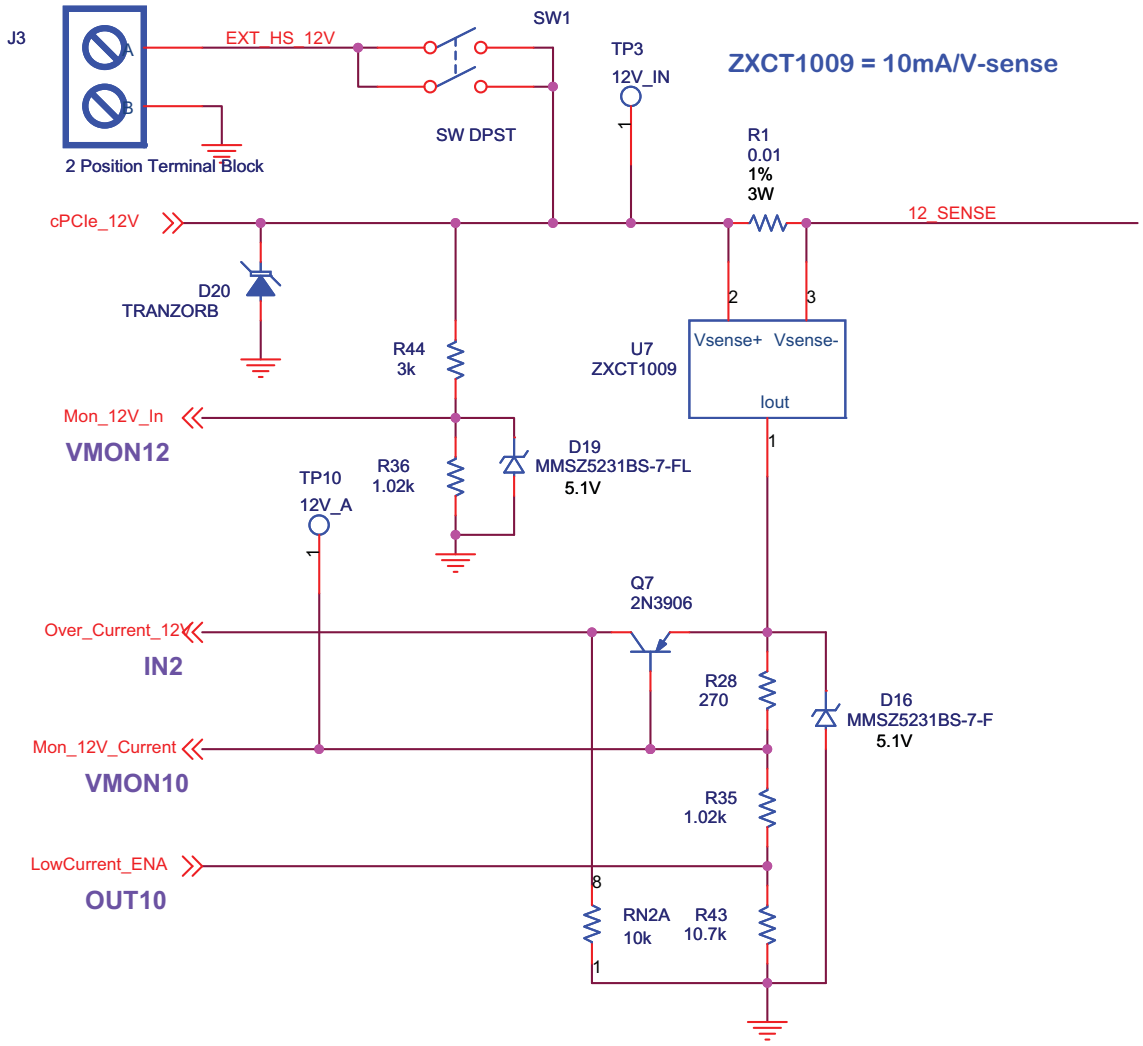
Current Sense Amp Circuit

The current sense circuit in the Hercules Hot-Swap design serves three main functions:

1. Limit the initial inrush current
2. Operate the MOSFET within its Safe Operation Area (SOA)
3. Provide short circuit and overload protection

Figure 11 shows an extract from the Hercules Evaluation Board schematic that illustrates how current sensing is implemented. Current sensing is done with the ZXCT1009 amplifier (U7) which produces 10mA output per volt sensed on the shunt resistor (R1). This output current passes through three resistors (R28, R35, and R43) in series to produce voltage levels that the digital input (IN2) and the analog VMON Input (VMON10) can sense. The resistor divider network of R35 and R43 (switchable) is used to create two different trip points depending upon the where the system is operating. Initially, when the card is first powering up the current is limited to approximately 1.5 amps by the VMON10 trip point and the hysteretic and SOA control of the MOSFET driver circuit. During this phase the current is passing through both resistors R35 and R43. After the 12V_HS supply reaches approximately 9V (determined by the V10_PRI_CUR_HIGH trip point setting of 1.503V) and the SOA of the MOSFET is achieved, the start-up phase is considered complete and the current limit can be raised to 15 amps by driving low the digital output OUT10, which will ground that point and effectively remove R43 from the circuit. At this time, the circuit is considered to be in an operational state and the remaining functions become active as determined by the sequence control.

Figure 11. Hercules Current Sensing Circuits



The IN2 digital input is used to sense an over-current condition and provides a fast shut down of the MOSFET circuit (Q1 in Figure 9). One of the 10k resistors in the array RN2A provides a pull-down on the over-current input. When the current through R28 is about 2.6mA there is enough voltage to bias the base-emitter junction of Q7 to turn it on and present the voltage at the emitter to the Over_Current_12V input. In all cases, this voltage will be enough to register as logic high at the digital input of the POWR1220AT8.

POWR1220AT8 Hot Swap Controller

The POWR1220AT8 uses the support circuits described above to control the operation of the power MOSFET. The POWR1220AT8 uses Supervisory Logic Equations to create the 20KHz timing signal used to modulate the charge pump enable and control the current into the MOSFET to keep it in the SOA. It uses the sequencer steps to create the logic required by the hot swap function.

Creating the 20KHz Charge Pump Timing Signal

In the Hercules implementation, the POWR1220AT8's 250KHz CPLD clock is used to create the signal to ramp up the charge pump using Supervisory Logic Equations, as shown in Listing 2. In Equations 0 and 1, the terminal count from 32us Timer 2 (TIMER2_TC) is registered by Extend_T2_TC (using the 4us clock) to add 8us to Timer2 (by delaying the reload of the timer value by two 4us clocks) giving a total of 40us. In practice an additional 4us will be measured (22.73KHz, 34us on and 10us off) because of an additional timer delay added from the ramping of

the voltage required to charge the capacitance connected to the HVOUT_1. Extend_T2_TC will have an 80% duty cycle with 32us on and 8us off). Extend_T2_TC will have an 80% duty cycle with 32us on and 8us off).

Listing 2. Hercules Charge Pump Timer Supervisory Logic Equations

```
// Create a fixed 80% duty cycle drive signal for external charge pump at
// 20kz by adjusting timer 2

//Extend Timer 2 TC width by 4 microseconds

EQ 0     Extend_T2_TC.D = TIMER2_TC

//Output of extend_T2_TC will be 32us on, 8us off (80%)
EQ 1     TIMER2_GATE.D = NOT Extend_T2_TC
```

Controlling the MOSFET Current

In order to control the charge pump so that the MOSFET does not operate outside the SOA (current does not exceed 1.5A during start-up) some additional checks need to be made. As shown in Listing 3, equation 2 creates the high voltage charge pump controlling output (HV1_ChargePump.D). This output is enabled (set to a '1') based on the duty cycle of Extend_T2_TC (since Extend_T2_TC is negated the charge pump will be on 80% of the time if enabled) required by the charge pump circuit. The charge pump is also enabled based on the value of the V10 and V8 current sensing measurements (V10_PRI_CUR_HIGH and V8_EXT_CUR_HIGH). If the current has not reached the 1.5A limit (NOT V10_PRI_CUR_HIGH AND NOT V8_EXT_CUR_HIGH) the enable can stay asserted. The enable is also asserted if the high side of the 12V supply has reached the voltage required for safe operation (HS_12V_HIGH is a '1').

Listing 3. Hercules Charge Pump Supervisory Equations

```
//Enable Charge Pump modulated by Extend_T2_TC
//Limit in-rush current to 1.5A initially
//Enable when 12V output is high enough

EQ 2     HV1_Charge_Pump.D = NOT Extend_T2_TC AND ((NOT
V10_PRI_CUR_HIGH AND NOT V8_EXT_CUR_HIGH) OR HS_12V_HIGH)
```

The POWR1220AT8 continues after the last initialization step (Listing 2, EQ 1) to bring the board to full power operation. Listing 4 shows the remaining steps. In Step 3 the POWR1220AT8 waits for 15.36ms for the HS_12V_HIGH monitor to show the 12V power has reached its target voltage. If the target voltage is not reached within the allocated time the sequence goes to the shutdown step. In Step 4 the low current measurement mode can be used (since the HS_12V_HIGH limit has been reached). The OUT10 (Figure 11) signal will shut off the lower resistor in the resistor network so the current sensing value can go to 15A (OUT9 for external 12V). A wait of 15.36ms is executed in Step 5 while the 12V supply reaches the full value. The Power_Good output is asserted in Step 6. This enables the loads and 1.2V supply as well as lighting the Power_Good LED. Step 7 is one more delay prior to starting the fault logging loop used by the Fault Logging demonstration. The Hot Swap function is complete.

Listing 4. Hot Swap Control Sequence for Hercules Design

```
//If the 12V power does not reach 9.6V within 15ms,
//shut off MOSFET and Goto Shutdown
Step 3   Wait for HS_12V_HIGH or 15.36ms using Timer 1
         If Timeout Then Goto 12

//Use low current mode for 2A or less, high current for 2A or more
Step 4   OUT9_Ext_LowCurrent_ENA = 0,
         OUT10_LowCurrent_ENA = 0

//Delay 15.36ms while 12V power supply reaches PWR_GOOD level
Step 5   Wait for 15.36ms using timer 1
```

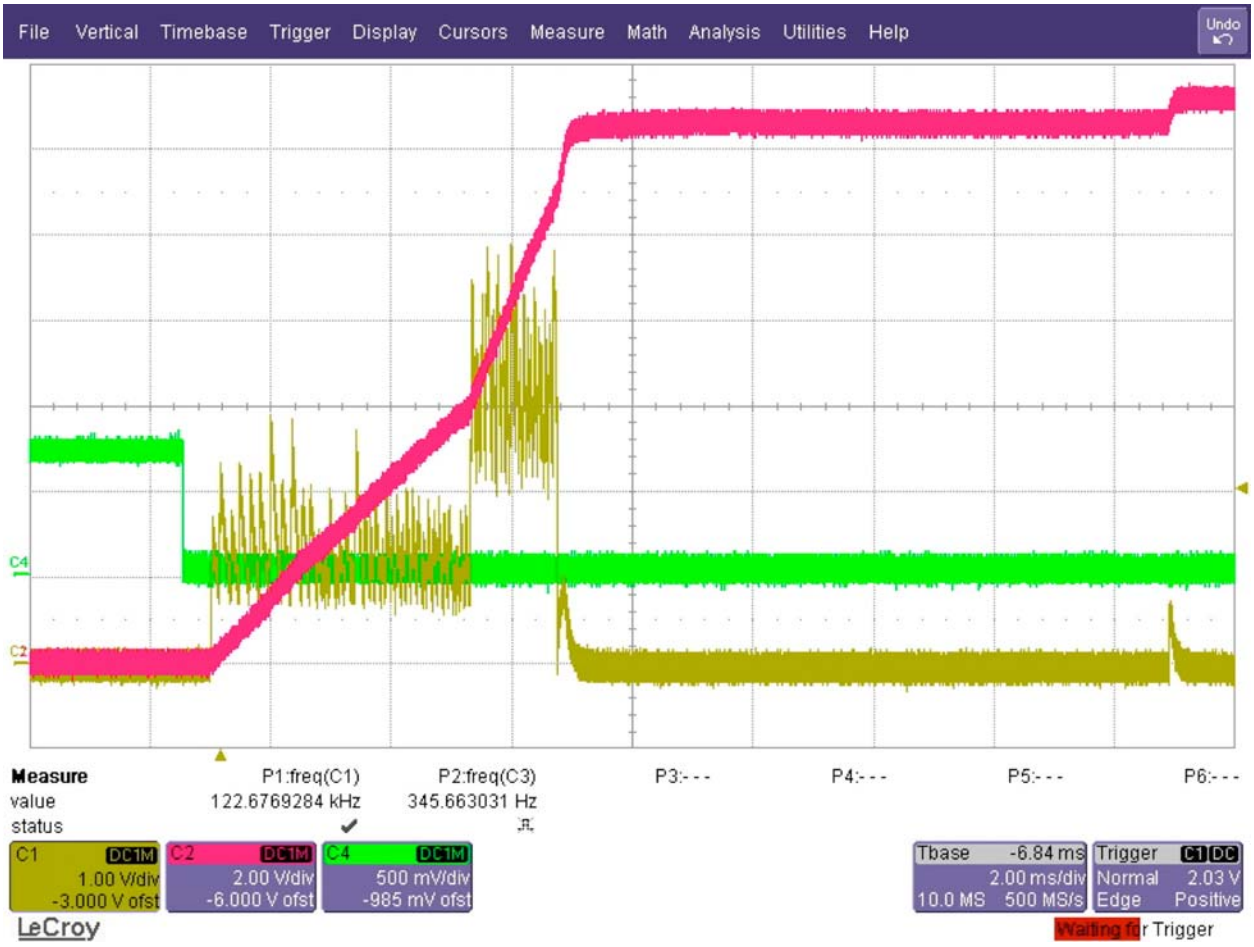
```
//Turn on Power Good LED, enable loads and 1.2V supply
Step 6   OUT15_PWR_GOOD = 0

//Delay 15.36ms prior to entering fault check loop
Step 7   Wait for 15.36ms using timer 1
```

Hot Swap Demonstration Results

A scope screen shot of the Hercules 12V hot swap feature charging the input capacitance is shown in Figure 12. Channel 1 (in yellow) is the 12V_A signal (12V current sense VMON10), Channel 2 (in pink) is the 12V_HS signal (12V hot swap output going into the rest of the system) and Channel 4 (in green) is the 12V_SDN signal (the shut-down control signal from the POWR1220AT8 OUT11).

The 12V_SDN signal starts out high keeping Q1, the power MOSFET, off. As it goes low, Q1 turns on and the 12V_HS output voltage begins to rise. The Mon_12V_Current signal becomes active and shows current flowing through the shunt resistor. Initially, current is limited to 1.5A to keep the MOSFET in the Safe Operating Area (SOA). After the voltage reaches the high voltage trip point, the current limit is relaxed and the output voltage ramp rate increases until the output capacitance is completely charged. Since there is no output load the current goes to zero.

Figure 12. Scope Screen Shot of 12V Hot Swap Test



Operational Description – Redundant Supply

This section describes the Redundant Supply function of the demonstration design. It shows how this portion of the design works in sufficient detail so that it can be used as a reference design and easily modified, if required, for a target application. The section is organized around the following key topics:

- Overview of the Redundant Supply Function
- Key Components, Functions and Algorithm of the Redundant Supply Design
- Implementation of the Redundant Supply Subsystem in the Hercules Design
  - 12V sense, Current Sense and MOSFET Gate Driver Circuit (described previously)
  - MOSFET Power-OR Circuit
  - DC-DC Converter Enable Circuit
  - POWR1220AT8 Redundant Supply Controller

Overview of the Redundant Supply Function

One method used to increase the reliability of high-reliability systems is through the use of systems that are powered by two or more (redundant) power supplies. These supplies are generated either by multiple sources or the system is connected to the main supply by the use of multiple paths. Loads connected to these redundant supplies can derive a single high-availability rail through the use of diodes. This arrangement is called a power rail OR'ing.

Using diodes is the simplest arrangement. Only the supply that has the highest voltage drives the main board voltage. If the supply voltages are roughly equal, the load power is shared between each of the sources. If a supply fails, the load is transferred to other supplies automatically without any interruption. Although this is the simplest and most reliable way of OR'ing supplies, this circuit has a disadvantage: it wastes power. Diodes usually drop about 700mV. If the load current is, say, 2A, the power dissipated by one diode is 1.4W. If there are ten boards in a shelf, the power dissipated in the 10 diodes is 14W, which stresses the cooling system. In addition, diodes that can dissipate more than 2W must be used. These diodes are not only expensive but also large, consuming more circuit board area.

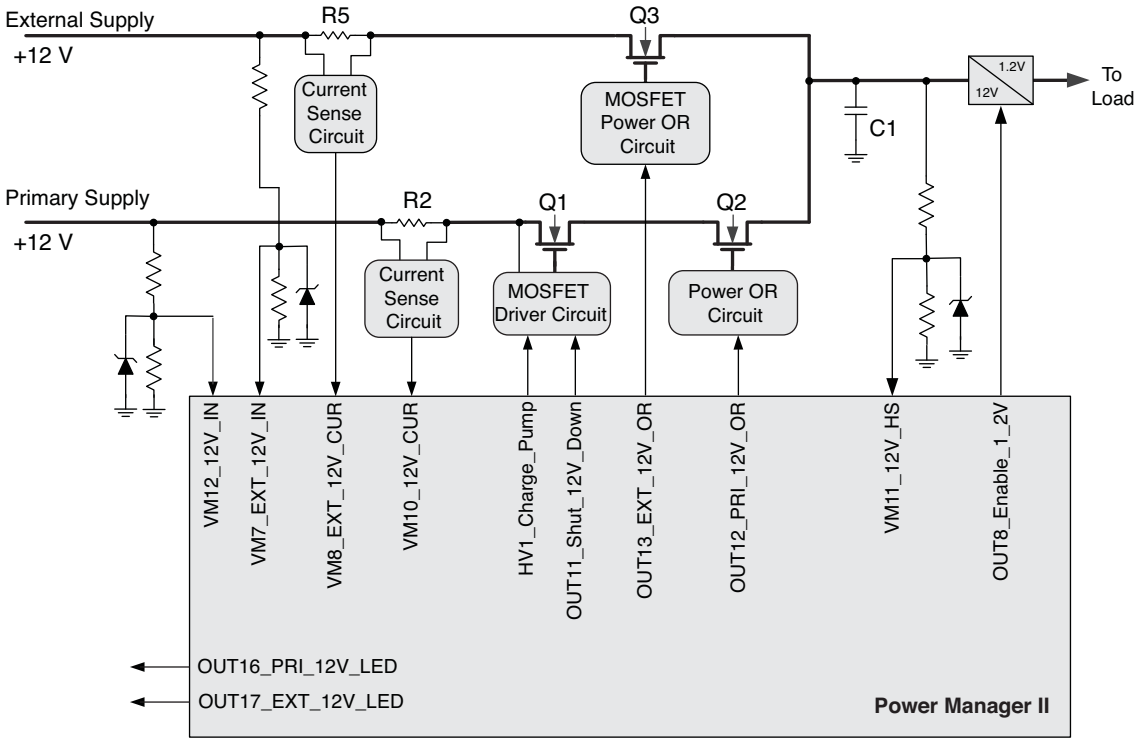
Modern-day Power-OR'ing circuits use MOSFETS to reduce the power dissipation significantly. Typical turn-on resistance of an N-channel MOSFET is about 25 mΩ, so the power dissipated by this MOSFET at 2A is 100mW. In other words, the power dissipation is reduced by 92%.

Key Components, Functions and Algorithm of the Redundant Supply Design

The Hercules Evaluation Board includes a Redundant Supply function. This function uses the POWR1220AT8 for control and some external support circuits to interface to the high voltage supply signals. We will first review the high-level architecture for a typical POWR1220AT8 controlled Redundant Supply design and then look at the Hercules implementation in more detail.

A typical MOSFET-based power supply OR'ing design using the POWR1220AT8 is shown in block diagram form in Figure 13. MOSFETs Q2 and Q3 and their body diodes perform the power supply OR'ing function. The 12V primary supply rail has a small series resistor (R1) to monitor the current for MOSFET SOA and over-current protection. The Q1 MOSFET is added to block the current through the Q2 body diode (turn on and off the 12V primary power supply). OR'ing the Q2 provides a low-impedance path around the Q2 body diode when the External Supply is equal to or lower than the Primary Supply. Q2 is turned off when the external supply is higher than the primary supply to prevent reverse current flow into the primary supply through the Q1 body diode. The external supply rail also has a small series resistor (R5) to monitor the current and a single N-channel MOSFET (Q3) which is used to lower the resistance in power in the OR'ing circuit. Q3 will be turned off if the primary supply is higher than the external supply to prevent reverse current into the external supply. Two voltage sensing resistor networks (on the left side of the figure) are used by the POWR1220AT8 to monitor the 12V input supply voltage supply levels and a similar circuit is used to monitor the output voltage level at the input to the DC-DC converter at the upper right in the figure. Following Figure 13 is a description of each of the key blocks (12V Voltage Sense Circuit, Current Sense Circuit, MOSFET Driver Circuit, MOSFET Power-OR Circuit, Power-OR Circuit, DC-DC Converter enable circuit and POWR1220AT8 Redundant Supply Controller) in the Hercules Redundant Supply implementation.

Figure 13. Typical Redundant 12V Power Management Block Diagram



Implementation of the Redundant Supply Subsystem in the Hercules Design

The implementation of the Redundant Supply subsystem in the Hercules design is very similar to the overview provided in the previous section. The key elements of the Redundant Supply design on the Hercules Evaluation Board are the POWR1220AT8 Redundant Supply controller, the MOSFET Power-OR circuit, the MOSFET driver circuit, the current sense circuit and the 12V voltage sense support circuits.

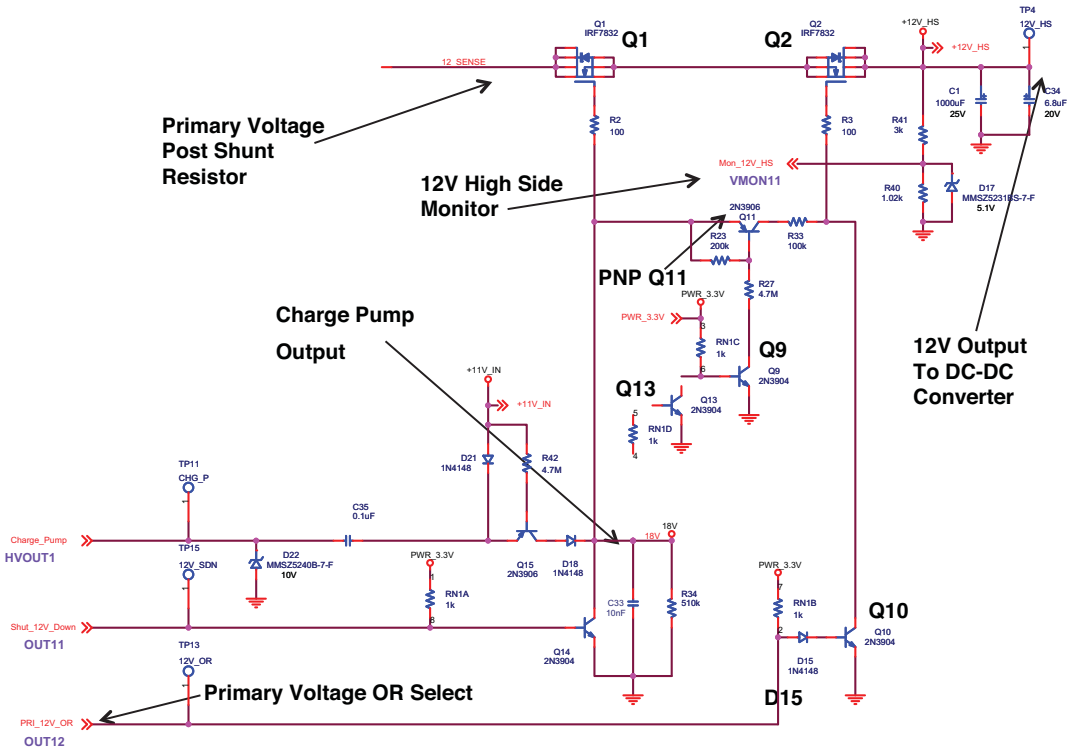
12V Voltage Sense, Current Sense and MOSFET Driver Circuits

The 12V Voltage Sense, Current Sense and MOSFET Driver Circuits used for the Redundant Supply function are also the ones used by the Hot Swap controller. The Hercules implementation of these circuits is described in detail in the Hot Swap Operational Description section and thus will not be repeated here.

MOSFET Power-OR Circuit

A portion of the schematic for the Hercules implementation of the 12V OR'ing circuit is shown in detail in Figure 14. This is the primary voltage leg of the Redundant Power-OR circuit (a single MOSFET circuit exists for the External 12V supply, the other leg of the circuit). The 12V Sense signal at the top left of the figure is the primary supply voltage after the R1 shunt resistor and before the Q1 MOSFET (as illustrated in Figure 13). The Charge Pump output controls the base on Q1 as described in the Hot Swap Operational description section. The PRI_12V_OR signal (OUT12) from the POWR1220AT8 is the key signal in Figure 14 for controlling the Redundant Supply function.

Figure 14. MOSFET Power-OR Circuit – Primary Voltage Leg



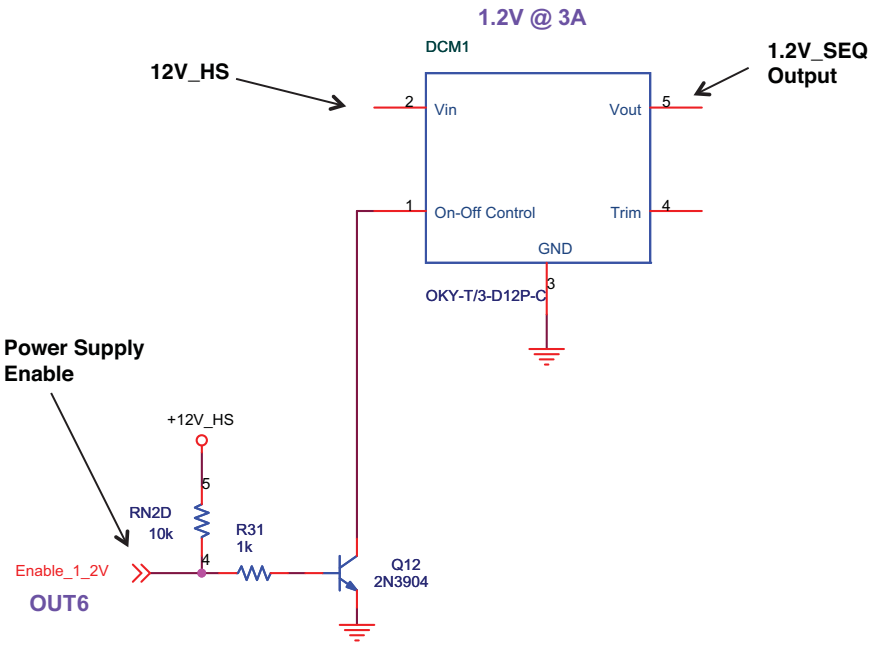
The POWR1220AT8 open collector output OUT12 is used to create the PRI_12V_OR signal. In order to put the primary voltage onto the DC-DC converter input both MOSFETs Q1 and Q2 must be turned on. MOSFET Q1 is independently turned on and the voltage on its input and current into the input are monitored by the POWR1220AT8. Q2 is controlled via the OUT12 signal to implement the Redundant Power feature.

The Q11 PNP transistor is turned on to enable the voltage and current from the charge pump output to the gate of the Q2 MOSFET. Q11 is turned on when Q9 turns on, which is when the OUT12 pin of the POWR1220AT8 is at Logic 0 (Q13 is off). When OUT12 is at logic 0 Q10 is also off. To turn Q2 (OR MOSFET) off, OUT12 is set to Logic 1. At that time Q9 turns off and Q10 turns on, draining the charge stored in the MOSFET gate, which turns Q2 off immediately. The diode D15 is introduced in the base circuit of Q10 to delay turning on Q10 compared to Q11, and to turn the Q10 off before Q11 to avoid the condition where Q11 and Q10 are both on at the same time, preventing independent operation of the Q1.

DC-DC Converted Enable Circuit

Figure 15 shows a portion of the Hercules schematic that prevents the DC-DC converter output (1.2V_SEQ) from being turned on before the POWR1220AT8 device commands it to. The 1.2V DC-DC converter has an On-Off Control pin which a transistor is used to short the On-Off Control pin to ground to keep the power supply turned off unless commanded to turn on by the POWR1220AT8 device via the Enable_1_2V signal (OUT6) logic 1. The base of the NPN transistor is pulled up to the 12V with a 10k resistor. The POWR1220AT8 digital outputs are open drain outputs which require an external pull-up resistor so this arrangement is compatible with the outputs.The +12V_HS level does not exceed the POWR1220AT8 operating since Q12 provides a current path for the resistor divider network. In order to insure that this circuit works correctly, this output (OUT6) of the POWR1220AT8 is configured to a reset state of high and is driven low when the power supply is scheduled to be turned on.

Figure 15. On-Off Control of 1.2V DC-DC Converter From POWR1220AT8



When the POWR1220AT8 commands a power supply to turn on it will drive the enable output (OUT6) low and the base of the transistor will also be pulled down. This will decrease the base-to-emitter voltage to a zero thus turn off the transistor. This creates an open on the On-Off Control pin to turn on the power supply. For more information about this type of circuit see application note AN6046, Interfacing Power Manager Devices with Modular DC-to-DC Converters.

POWR1220AT8 Redundant Supply Controller

The POWR1220AT8 implements the Redundant Power functions using Supervisory Logic equations. These equations create the OR selection signals for both the primary and external 12V supplies as well as the enable signal for the 1.2V DC-DC converter. Listing 5 shows the equations used for these functions. EQ3 controls the enable for the Primary 12V supply. The MOSFET should be turned on when the current through the sense resistor exceeds the threshold. (If the current is above the threshold the supply is working and can be connected to the rest of the system.) If the DC-DC converter is not enabled the supply need not be connected. The enable is active low so the equation is written to set OUT12 low only if OUT6_Enable_1_2V is low (this is active since it is active low) and V10_PRI_CUR_THRESHOLD is High (this is active since it is active High). EQ4 has the same structure and it is enabling the External Supply. (Note that both supplies can be enabled at the same time or both disabled). EQ 17 is used to create the 1.2V DC-DC converter enable signal. The 1.2V supply is to be enabled when power is good and the MachXO has enabled the supply. Since OUT6_Enable_1_2V is active low, it is set low if the MachXO enable (IN5_XO_PWR0) is High and the power good signal is Low (OUT15_PWR_GOOD is active Low). OUT15_PWR_GOOD is the indication that the power-up process has completed successfully. It is asserted in Step 6 of Listing 4.

Listing 5. Supervisory Logic Equations for Redundant Supply Selection and Enabling

```
//Turn on Primary supply if current exceeds threshold and
//load is turned on.
EQ 3OUT12_PRI_12V_OR.D =  NOT V10_PRI_CUR_THRESHOLD OR
OUT6_Enable_1_2V

//Turn on External supply if current exceeds threshold and
//load is turned on.
EQ 4OUT13_EXT_12V_OR.D =  NOT V8_EXT_CUR_THRESHOLD OR
OUT6_Enable_1_2V

//Enable 1.2V Supply and Loads when Power is Good and XO enables
EQ 17 OUT6_Enable_1_2V.D =  NOT ( IN5_XO_PWR0 AND NOT OUT15_PWR_GOOD )
```

Because the Redundant Supply control equations are implemented with Supervisory equations and are not tied to any sequencing logic the correct 12V source will be selected at any time during the demonstration program after power is good (and except during shutdown).

Operational Description – Fault Monitoring and Logging

This section describes, in detail, the operation of the Fault Monitoring and Logging demonstration design. It shows how the design works in sufficient detail so that it can be used as a reference design and easily modified, if required, for a target application. The section is organized around the following key topics:

- Review of the Fault Monitoring and Logging Demonstration Operation
- Key Components, Functions and Algorithm of the Fault Monitoring and Logging Design
- Voltage Fault Monitoring Implementation
  - Measuring Voltage Faults
  - Capture and Transmission of Fault Data from POWR1220AT8 to MachXO
- Fault Logging Function Implementation
  - Fault Logging Implementation
  - Writing the Fault Error Log to SPI Memory
  - Keeping Key Circuits Operational During a Power Fault
  - Reporting Error Status on the LCD Display
  - Interface to Buttons and Switches
- Implementation of the MCU-based Controller for the Fault Logging Function
  - Control flow diagram
  - Hyper-Terminal access to Fault Logging commands

Review of the Fault Monitoring and Logging Demonstration Operation

In Quick Start Guide Demo 4, the fault monitoring function is illustrated with a slide potentiometer that simulates a power supply and is monitored by VMON9. The LCD displays the voltage set by the slider. The full range of voltage is 0V to 3.25V. When the voltage is placed outside the voltage good "window" a fault is detected. A voltage fault is also detected if power is removed from the board. Detected faults are logged and stored to the serial memory. These faults can be displayed in Demo 4a mode.

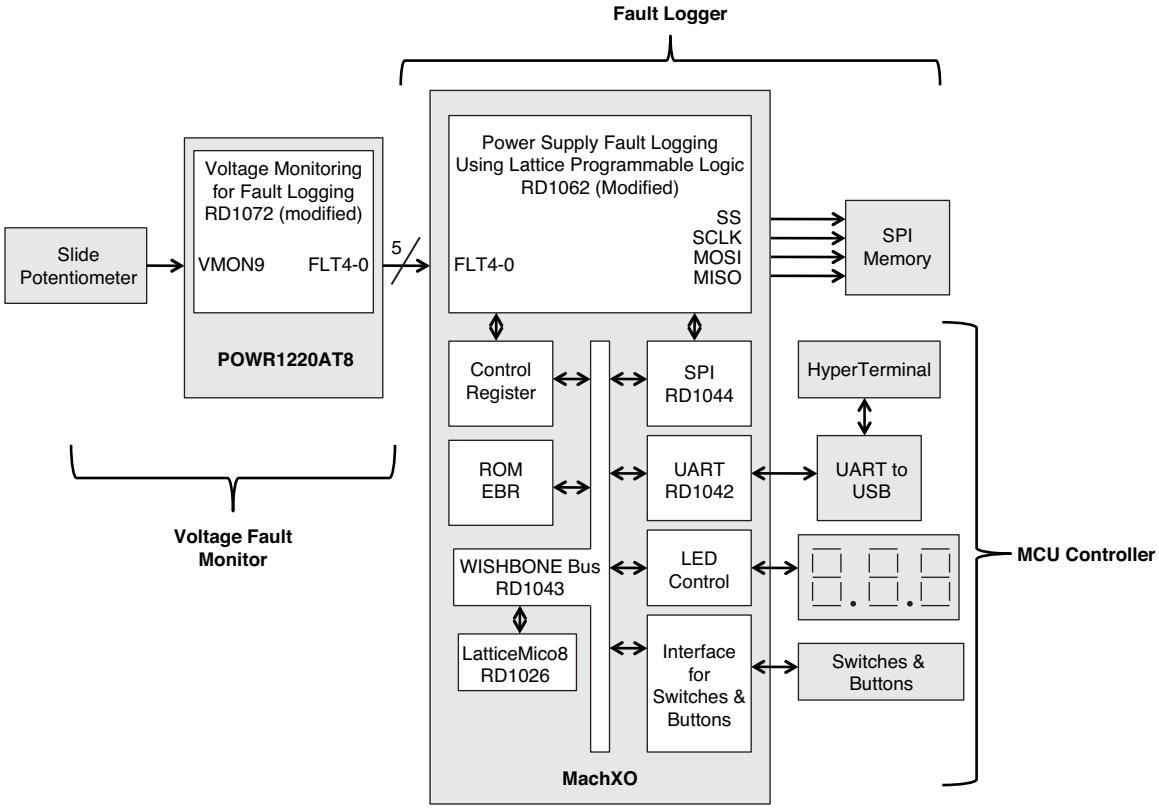
In Demo 4a mode the LCD shows the total count of faults recorded by the POWR1220AT8 and MachXO. In this mode pressing the mode switch (SW3) clears all the recorded faults.

Key Components, Functions and Algorithms – Fault Monitoring and Logging Subsystem

Several parts of the Hercules Evaluation board work together in the Demonstration Design to implement the Voltage Fault Monitoring and Logging subsystem (Quick Start Guide Demo Commands 4 and 4a). The key subsystem functions, shown in Figure 16, can be categorized as the Voltage Fault Monitor, the Fault Logger and the MCU con-

troller. Lattice has published several reference designs for the Power Manager II and MachXO devices which are used to implement this subsystem. RD1072, Voltage Monitoring for Fault Logging with ispPAC-POWR1220AT8 is used, with some modifications, in the POWR1220AT8 to implement the Voltage Fault Monitor function. RD1062, Three-Wire Power Supply Fault Logging Using Lattice Programmable Logic is used, with some modifications, in the MachXO to implement the Fault Logging function.

Figure 16. Fault Logging and Monitoring Subsystem Key Elements



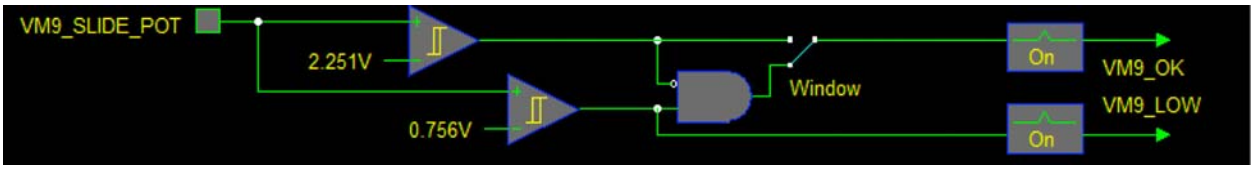
Voltage Fault Monitor Implementation

The Voltage Fault Monitor function is similar to the design used in RD1072. RD1072 is simpler than the Hercules design since it is more generic, allowing it to be more easily adapted to a range of designs. The additional functionality in the Hercules design results in some extra complexity in the Sequencing equations and the Supervisory logic equations as compared to RD1072. For example, the Hercules signal names are more specific to the Hercules design while the signals in RD1072 tend to be more generic and use the POWR1220AT8 default signal names. The following description borrows from that used in RD1072 but is specific to the Hercules design by using the Hercules signal names and logic equations. Users familiar with RD1072 will see the similarities.

Measuring Voltage Faults

In the Hercules design the slide potentiometer is used to generate a voltage fault event. The Voltage Fault Monitor function detects the output voltage from the slide potentiometer. The voltage is measured on the VMON9 analog input (called out as VM9_SLIDE_POT on the Hercules Evaluation Board schematic) on the POWR1220AT8. This input is configured in "window" mode within the PAC-Designer software schematic as shown in Figure 17.

Figure 17. POWR1220AT8 PAC-Designer Schematic, VMON9 Selected in Window Mode



The two comparators are set to 2.251V and 0.756V. When the VM9_SLIDE_POT input is below 2.251V and above 0.756V the VM9_OK signal is a logic '1', indicting the voltage is within the desired range. If the VM9_SLIDE_POT input is below 0.756V, the VM9_LOW signal is a logic '1'. The VM9_OK and VM9_LOW signals are available to the sequence control logic of the POWR1220AT8 to modify power supply operation.

Listing 6 shows the Fault Monitor loop in the POWR1220AT8 Hercules demonstration design (from the U3_POWR1220_demo.PAC file in the Project/Demo_Hercules/POWR1220 folder). The loop begins at Step 8, which is just after the start-up and initialization routine described in a previous section. In this step the VM9_OK signal is used to check if the slider voltage is out of range (NOT VM9_OK). If the command is enabled by the XO_PWR1 control signal (AND IN6_XO_PWR1) from the MachXO, the sequence goes to Step 9 where the DUMP_STATUSn signal is asserted (DUMP_STATUSn = 0). This assertion kicks off the status transmission to the MachXO. The sequence waits in Step 10 until the fault is recovered (VM9_OK), the command is finished (NOT IN6_XO_PWR1) or the timer expires (1966.08ms using Timer3 If Timeout Then Goto 8) and then transitions back to Step 8 to wait for another fault (either directly from Step 10 if a Timeout or from Step 11 if the logic condition passes).

Listing 6. PAC-Designer Sequence Logic for the Hercules Fault Logging Loop

```
//Wait for Fault if logging enabled

Step 8    Wait for NOT VM9_OK AND IN6_XO_PWR1(interruptible)

//Log Fault- Trigger the VMON Status Dump

Step 9    DUMP_STATUSn = 0

//Wait for fault recovery or timeout

Step 10   Wait for VM9_OK OR NOT IN6_XO_PWR1 or 1966.08ms using Timer 3, If timeout Then
Goto 8    (interruptible)

Step 11   Goto step 8
```

Steps 8 and 10 are enabled for interrupts. This allows the Exception condition that generates a transition to the Shutdown state to be taken in these steps. The other steps are executed very quickly, so a transition to the Shutdown state would be a waste of PLD resources. This approach is specific to the Hercules design and may not be appropriate for another design. A detailed description of the use of the Exception condition is given in the Shutdown Operational Description section later in this document.

It is important to note that another type of fault is logged by the Hercules Demonstration Design. When a catastrophic fault occurs a fault is also logged. This is done by using an Exception Condition in the Sequence Logic of the POWR1220AT8. This function is implemented in the Shutdown Sequence portion of the Sequencer Logic and will be explained in more detail as part of the Shutdown Operational Description section later in this document.

The fault logging function stays powered-up long enough to successfully save the fault status data in the SPI memory. This key design element is explained in the Keeping Key Circuits Operational During a Power Fault portion of the Fault Logger Implementation description later in this section.

Capture and Transmission of Fault Data from the POWR1220AT8 to the MachXO

When a fault is detected, the POWR1220AT8 indicates the fault to the MachXO. The voltage monitor status signals within the POWR1220AT8 are transmitted to the MachXO using a series of serial signals that sequentially shift out

the captured voltage monitor signals. The Hercules demonstration design implements the transmission function in a very similar fashion as RD1072.

When a fault event occurs in the Hercules demonstration design the POWR1220AT8 will send the status of the VMON signals (VMON1A/B through VMON10A/B); and the HS_12V_LOW, HS_12V_HIGH, INPUT_12V_OK and INPUT_12V_MIN signals. The status signals are one-bit digital values that show the results of the comparison on the inputs of the associated comparators (an example of one was shown in Figure 7). Since there are a total of 24 fault status signals there are 24 status bits.

In order to transmit the fault status bits to the MachXO using a small number of pins, the POWR1220AT8 device supplies the fault status signals serially over the four fault output pins. A three-bit counter is used to step through the various signals. In the Hercules design the counter is implemented with equations 5 through 11 as shown in Listing 7. This counter is reset to a value of 3 (via equations 8-10 where the asynchronous preset and reset are used) and then counts up using the sequence; 4, 5, 6, 7, 0, and 1 at the rate of the PLD clock. The counter is enabled when DUMP_STATUSn) is set low in the main loop shown previously. When the counter reaches a value of '010', the DUMP_STATUSn register is set high (via its asynchronous preset in equation 11) which then resets the counter and prevents multiple status dumps.

Listing 7. Counter Implementation in the Hercules Fault Logging Function

```
// Equations 5 - 7 provide a three-bit binary up-counter that
// is enabled by DUMP_STATUSn.

EQ  5 Cntr0.D = NOT Cntr0
EQ  6 Cntr1.D = ( Cntr0 AND NOT Cntr1 ) OR ( NOT Cntr0 AND Cntr1 )
EQ  7 Cntr2.D = ( NOT Cntr2 AND Cntr0 AND Cntr1 )
OR ( Cntr2 AND NOT ( Cntr0 AND Cntr1 ) )

// The binary counter is reset to a count of 3 when the
// DUMP_STATUSn flag is high (de-asserted) and counts 4,5,6,7,0,1,2 when
// the flag is low (asserted).

EQ  8 Cntr0.ap = DUMP_STATUSn
EQ  9 Cntr1.ap = DUMP_STATUSn
EQ 10 Cntr2.ar = DUMP_STATUSn

// Equation 11 automatically sets DUMP_STATUSn high to reset
// the binary counter, after the VMON status bits have been
// transmitted (at the count of 2).

EQ 11 DUMP_STATUSn.ap = NOT Cntr0 AND Cntr1 AND NOT Cntr2
```

The selector logic is implemented using Supervisory Logic Equations 8-11. The equations combine the counter and status bits to select a single status bit based on the counter value. Listing 8 shows these equations for the Hercules Design. The counter outputs (Cntr2-0) are decoded to select specific signals. For example, the previously described VM9_OK and VM9_LOW signals are output on counter vales of 000 and 001 respectively (shown in the last two lines of the OUT18_PM_FLT2 equation). Note that not all counter values are used and that the selector starts with Cntr2, Cntr1 and Cntr0 at '100'. The reason for this is explained in the next section. All these signals are combinatorial.

Listing 8. Fault Logging Status Signal Selector Implementation in the Hercules Design

```
//Cntr2-0 is used to select and shift out fault status bits

//VMON1 - VMON3 status bits are selected on Fault signal FLT0

OUT16_PM_FLT0 =
( VMON1_A AND NOT Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( VMON1_B AND Cntr0 AND NOT Cntr1 AND Cntr2 )  OR
( VMON2_A AND NOT Cntr0 AND Cntr1 AND Cntr2 ) OR
( VMON2_B AND Cntr0 AND Cntr1 AND NOT Cntr2 ) OR
( VMON3_A AND NOT Cntr0 AND NOT Cntr1 AND NOT Cntr2 ) OR
( VMON3_B AND Cntr0 AND NOT Cntr1 AND NOT Cntr2 )

//VMON4 - VMON6 status bits are selected on Fault signal FLT1

OUT17_PM_FLT1 =
( VMON4_A AND NOT Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( VMON4_B AND Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( VMON5_A AND NOT Cntr0 AND Cntr1 AND Cntr2 ) OR
( VMON5_B AND Cntr0 AND Cntr1 AND NOT Cntr2 ) OR
( VMON6_A AND NOT Cntr0 AND NOT Cntr1 AND NOT Cntr2 ) OR
( VMON6_B AND Cntr0 AND NOT Cntr1 AND NOT Cntr2 )

//VMON7 - VMON9 status bits are selected on Fault signal FLT2
//Generic signal names have been replaced with Hercules names

OUT18_PM_FLT2 =
( V7_EXT_IN_12V_OK AND NOT Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( V7_EXT_IN_12V_MIN AND Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( V8_EXT_CUR_HIGH AND NOT Cntr0 AND Cntr1 AND Cntr2 ) OR
( V8_EXT_CUR_THRESHOLD AND Cntr0 AND Cntr1 AND NOT Cntr2 ) OR
( VM9_OK AND NOT Cntr0 AND NOT Cntr1 AND NOT Cntr2 ) OR
( VM9_LOW AND Cntr0 AND NOT Cntr1 AND NOT Cntr2 )

//VMON10 - VMON12 status bits are selected on Fault signal FLT3
//Generic signal names have been replaced with Hercules names

OUT19_PM_FLT3 =
( V10_PRI_CUR_HIGH AND NOT Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( V10_PRI_CUR_THRESHOLD AND Cntr0 AND NOT Cntr1 AND Cntr2 ) OR
( HS_12V_HIGH AND NOT Cntr0 AND Cntr1 AND Cntr2 ) OR
( HS_12V_LOW AND Cntr0 AND Cntr1 AND NOT Cntr2 ) OR
( INPUT_12V_OK AND NOT Cntr0 AND NOT Cntr1 AND NOT Cntr2 ) OR
( INPUT_12V_MIN AND Cntr0 AND NOT Cntr1 AND NOT Cntr2 )
```

The last output signal needed to complete the data transmission between the POWR1220AT8 and the MachXO is a start signal. In the Hercules design this is done with the FLT4 signal. The FLT4 signal goes high to indicate to the MachXO that data is available on FLT3 through FLT0. Equation 16, shown in Listing 9, is a simple combinatorial output that routes Cntr2 to the OUT_20PM_FLT4 signal (FLT4). The counter is continually reset to the '011' state while DUMP_STATUSn is high (de-asserted). Once DUMP_STATUSn goes low (asserted) the counter begins counting and transitions to '100'. This puts the first set of status bits onto the status bus (the first entry in the selection logic described previously is with Cntr2, Cntr1 and Cntr0 at '100') and brings OUT20_PM_FLT4 high. Resetting

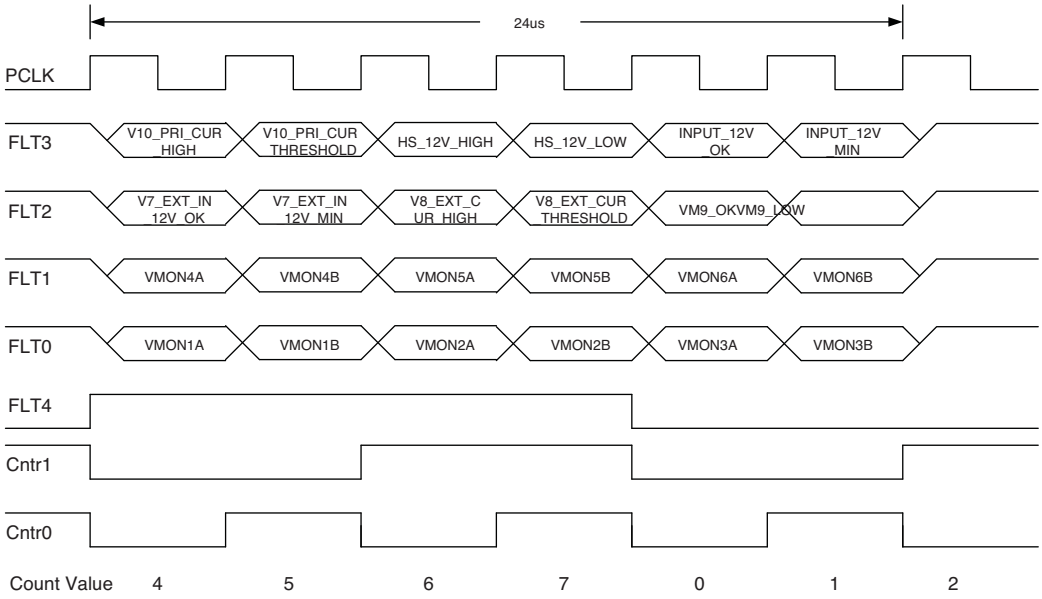
the counter to '001' thus allows the logic for OUT_20_PM_FLT4 to be simplified. (In fact, the Lattice fitter optimizes Cntr2 to be OUT20_PM_FLT4 and thus the Cntr2 signal is missing from the simulation and fitter report).

Listing 9. Fault Logging Trigger to MachXO

```
EQ16 OUT20_PM_FLT4 = Cntr2
```

The POWR1220AT8 sends the status data synchronous with respect to PCLK as shown in Figure 18. The values for the internal counter (Cnt2/FLT4, Cnt1, Cnt0) are shown at the bottom of the diagram to further illustrate that the counting sequence begins at 4 and uses the Cntr2 as the start signal to the MachXO. The entire transfer is accomplished in six clock cycles and requires only 24us.

Figure 18. Fault Logging Status Dump From the POWR1220AT8 Device



Fault Logger Implementation

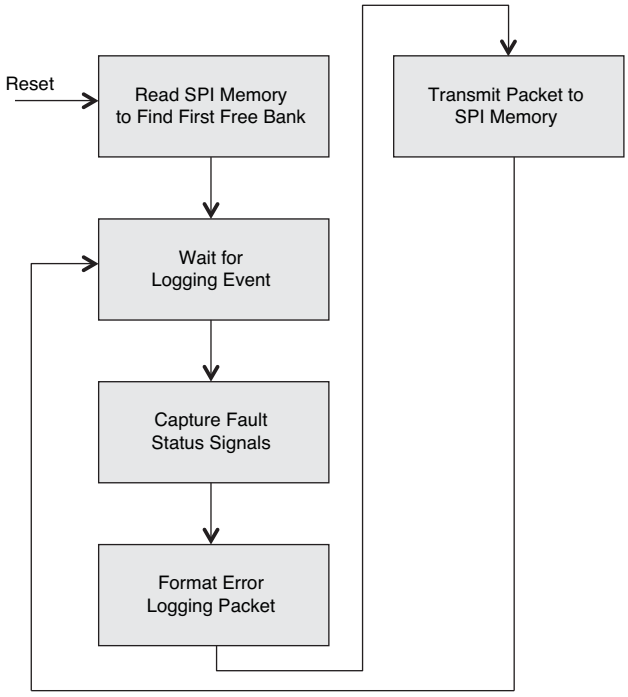
When a fault is detected the POWR1220AT8 indicates the fault to the MachXO. The desired status signals are captured and transmitted from the POWR1220AT8 to the MachXO using a series of serial signals that sequentially shift out the captured voltage monitor signals. The MachXO receives these signals, creates an error log data packet and then writes them to memory as an error logging event. The circuits implementing this function must stay active long enough during a voltage outage so that the fault event is logged correctly by the design.

RD1062, Three-Wire Power Supply Fault Logging Using Lattice Programmable Logic includes code for the MachXO. It uses an external SPI memory to store the logged fault event data. The Hercules demonstration design uses a slightly modified version of RD1062. The modifications are described in this section so that a user familiar with RD1062 will be able to easily see the changes and can use the reference design without confusion.

Figure 19 shows a flow diagram for the Hercules Fault Logging design within the MachXO. Fault logging is done on a block basis to simplify the design. This means that each block will hold a single fault event. Before a write is started the address for the first available block must be determined. This is done on system power-up or after a reset as shown in the first block in Figure 19. The fault logger reads the first byte of each SPI memory block sequentially to find the first open block (a value of OxC3 is written to the first byte to indicate it contains logging data) and stores the open block address. Thus the address for the next available block has already been determined by the MachXO design. After the free block is determined the program waits for a logging event to be asserted via FLT4. When an event is asserted the serial fault status is captured by the MachXO using a series of serial shift registers. The captured status bits are then included in the error logging packet along with timer informa-

tion. The fault event packet is then sent to the SPI memory. After the bank has been written to memory and the fault event is safely logged the program returns to wait for the next fault event.

Figure 19. Fault Logging Flow Diagram for MachXO Implementation



The balance of this section will describe the details behind the design of the fault logging function. The description is broken into several steps, which follow the sequence of operation of the fault logging design just described. These steps include:

- Capture of Fault Status Signal by the MachXO
- Formatting of the Error Logging Packet
- Writing the Fault Error Log to SPI Memory
- Keeping Key Circuits Operational During a Power Fault

Capture of Fault Status Signals by MachXO

As described in the previous section, the POWR1220AT8 sends 24 bits of fault data to the MachXO for logging in the SPI memory. Prior to sending the data to the SPI memory the status bits must be received and the full error log packet created. (Some of this process overlaps, but for now we will consider these processes to be sequential to facilitate understanding of the reception and generation functions).

The MachXO receives the serial data in four shift registers that capture the first six data bits after the FLT4 signal indicates capture should begin. A section of the MachXO code from the Hercules demonstration design (found in the fault_logging_five.v file) is shown in Listing 10. The status_line registers hold the data as it is being captured. Each register is shifted on position while data is available (during the first six cycles as determined by the capture counter 'capture_count' with the final capture indicated by the 'capture_end' signal).

Listing 10. Code for Shifting Status Bits into MachXO

```
// Capture Status Data

//shift data into status line registers
//The data is shifted into registers [7:1] of the status_lines
always @ (negedge pld_clk, posedge f_log_reset, posedge capture_end)
begin
      if (f_log_reset)
        begin
         status_line_0 = 0;
         status_line_1 = 0;
         status_line_2 = 0;
         status_line_3 = 0;
        end

      else if (wakeup & ~capture_end)
        begin
         status_line_0    = status_line_0 << 1;
         status_line_1 = status_line_1 << 1;
         status_line_2    = status_line_2 << 1;
         status_line_3    = status_line_3 << 1;
         status_line_0[0] = stat[0];
         status_line_1[0] = stat[1];
         status_line_2[0] = stat[2];
         status_line_3[0] = stat[3];
        end

      else
        begin
         status_line_0 = status_line_0;
         status_line_1 = status_line_1;
         status_line_2 = status_line_2;
         status_line_3 = status_line_3;
      end
end



//counter keeping track of status data shift
always @ (posedge pld_clk, posedge f_log_reset)
begin
      if (f_log_reset)
         capture_count = 0;
      else if (wakeup & ~capture_end)
         capture_count = capture_count +1;
      else
         capture_count = capture_count;
end



//end shift data into status registers
always @ (negedge pld_clk, posedge f_log_reset)
```

```
begin
      if (f_log_reset)
         capture_end = 0;

      // stop fault capture after 6th falling edge
      else if (capture_count == 6)
         capture_end = 1;
      else
         capture_end = capture_end;
end
```

Formatting of Error Logging Packet

After the data is received by the MachXO it must be formatted for writing to the SPI memory. The Hercules MachXO code in Listing 11 (found in the fault_logging_five.v file) formats the data with a series of assign statements to pack the status_line register contents into the fault_log_bytes (VMON bits are assigned to fault_log_bytes 1-3) used to build up the full 8-byte fault_log packet. Note that the constant value 8'hC3 is used in fault_log_byte0 to indicate the bank has been written to and timer values fill fault_log_bytes 4-7. The fault_log data is embedded in the full pp_stream data packet for transmission to the SPI memory. The pp_stream includes the write enable (WREN) and page program (PP) commands followed by a pad byte, the address, a pad byte and the fault_log data.

Listing 11. Formatting the Fault Log Data for Transmission to the SPI Memory

```
//formatting data for spi storage

assign fault_log_byte_0 = 8'hC3;

assign fault_log_byte_1 = {
{status_line_1[5]}, // VMON4_B
{status_line_1[6]},// VMON4_A
{status_line_0[1]},// VMON3_B
{status_line_0[2]},// VMON3_A
{status_line_0[3]},// VMON2_B
{status_line_0[4]},// VMON2_A
{status_line_0[5]},// VMON1_B
{status_line_0[6]}};// VMON1_A

assign fault_log_byte_2 = {
{status_line_2[3]}, // VMON8_B
{status_line_2[4]},// VMON8_A
{status_line_2[5]},// VMON7_B
{status_line_2[6]},// VMON7_A
{status_line_1[1]},// VMON6_B
{status_line_1[2]},// VMON6_A
{status_line_1[3]},// VMON5_B
{status_line_1[4]}};// VMON5_A


assign fault_log_byte_3 = {
{status_line_3[1]},// VMON12_B
{status_line_3[2]},// VMON12_B
{status_line_3[3]},// VMON11_B
{status_line_3[4]},// VMON11_B
{status_line_3[5]},// VMON10_B
{status_line_3[6]},// VMON10_A
{status_line_2[1]},// VMON9_B
```

```
{status_line_2[2]}}; // VMON9_A


assign fault_log_byte_4 = {timer[25:18]};
assign fault_log_byte_5 = {timer[33:26]};
assign fault_log_byte_6 = {timer[41:34]};
assign fault_log_byte_7 = {timer[49:42]};

assign fault_log = {fault_log_byte_0, fault_log_byte_1,
                    fault_log_byte_2, fault_log_byte_3,
                   fault_log_byte_4, fault_log_byte_5,
                    fault_log_byte_6, fault_log_byte_7};

assign pp_stream = {WREN,PP,pad,page_addr,pad,fault_log};
```

Writing the Fault Error Log to SPI Memory

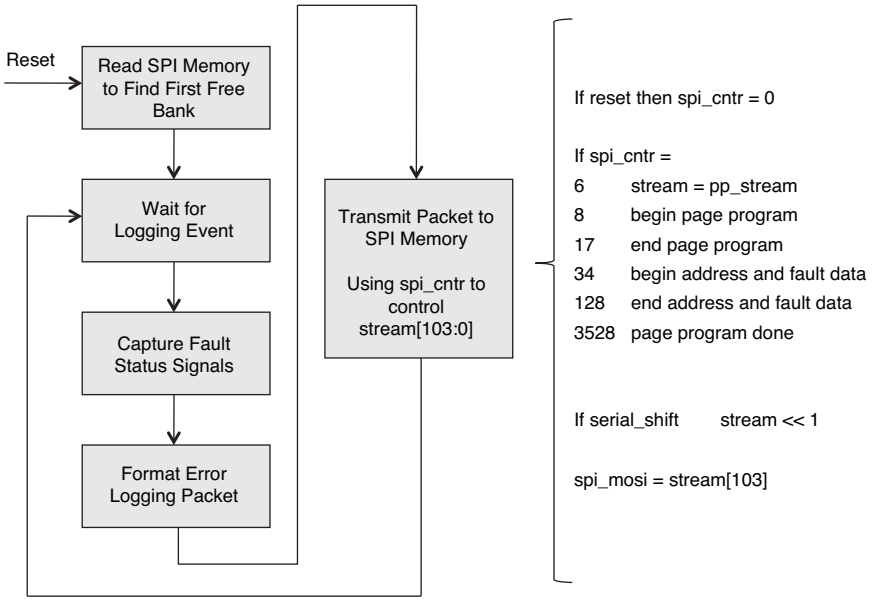
Once the Fault Error Log has been created the data packet is ready to be written to the SPI memory. Fault logging is done on a block basis to simplify the design. This means that each block will hold a single fault event. Before a write is started the address for the first available block must be determined. This is done on system power-up or after a reset. Thus, the address for the next available block has already been determined by the MachXO design. Figure 20 shows a high-level description of the program flow for the process of writing the fault logging data to SPI memory, beginning with the free bank search upon reset.

Once the next block is determined the program waits for a logging event to be detected (via the FLT4 input).

The pp_stream data packet is transferred to the SPI memory under control of the SPI Fault Logging Controller within the MachXO. This controller uses a counter (spi_cntr in the Hercules Verilog Code fault_logging_five.v file) to keep track of the process of writing the data to SPI memory. This counter increments as long as the fault logger is not reset, logging is active and the memory is ready. Various values of the counter are decoded (6, 8, 16, 32, etc.) to control specific functions.

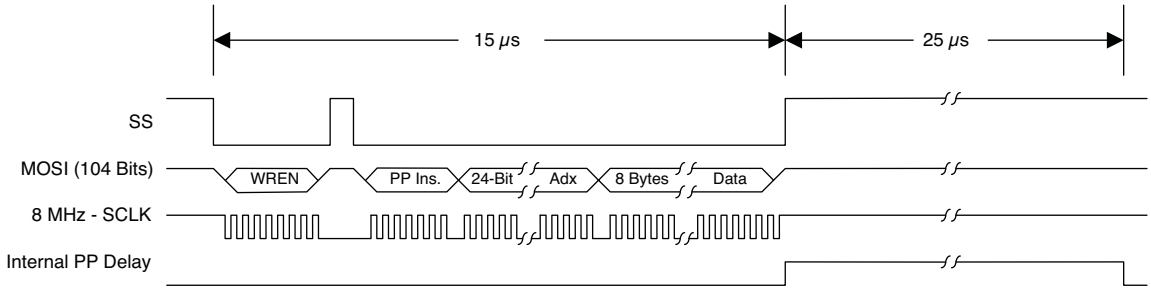
As shown in detail on the right side of Figure 20, the spi_cntr is reset to zero and then increments. After six clock cycles the serial shift register connected to the SPI memory input (see the assignment of spi_mosi = stream[103] at the bottom right of the figure) is loaded with the pp_stream data formatted in the previous step. When spi_cntr is 8, the Page Program instruction is transmitted and at 16 it is finished. At 32 the address and fault data transmission begins and at 128 it is complete. The Page Program function on eight bytes of data requires 0.45ms to complete so the spi_cntr value of 3528 is used to make sure the function completes prior to returning to wait for the next fault. The use of the spi_cntr to control operation is a simple and low macrocell count implementation.

Figure 20. Flow Diagram for Writing Fault Logging Data to SPI Memory



The Fault Logging Controller serializes the pp_stream data on the SPI MOSI pin as illustrated by the timing diagram in Figure 21. The transfer takes less that 50us but, as mentioned earlier, the complete program cycle takes closer to 0.5ms. This requires some planning to make sure the fault logging data can be successfully written to SPI memory prior to the voltage getting too low for correct operation. This design consideration is covered in the next portion of this section.

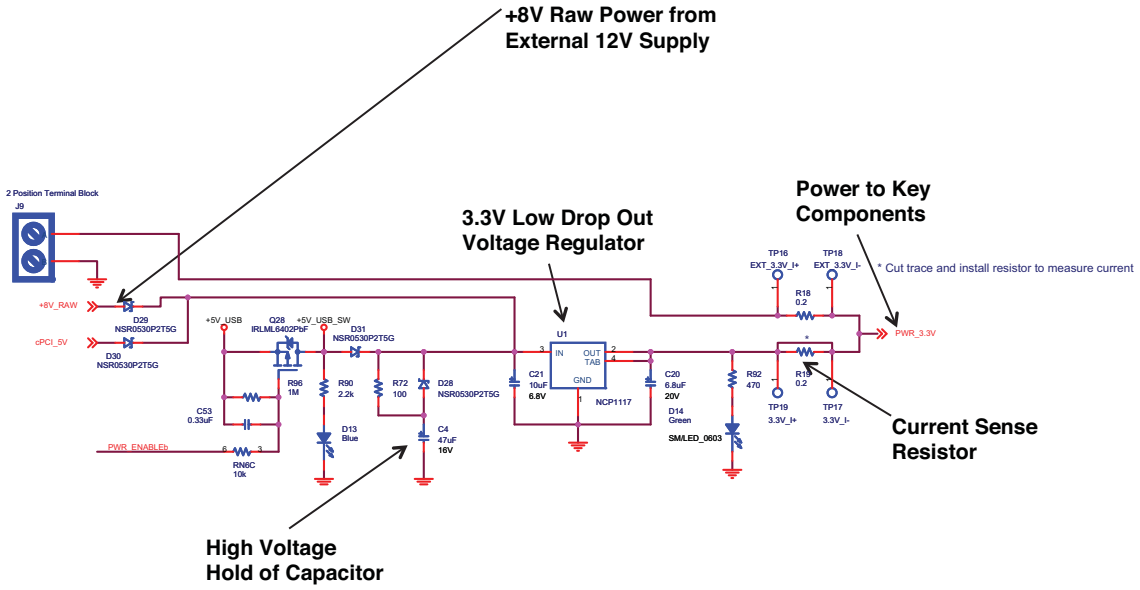
Figure 21. Fault Logging to SPI Waveforms



Keeping Key Circuits Operational During a Power Fault

In order for fault logging to be successful if the main power source begins to fail, the critical fault logging circuits must stay active long enough to successfully complete the logging of the fault event. In the Hercules demonstration design the circuit shown in Figure 22 is used to supply the 3.3V power (via the PWR_3.3V) to all the critical devices (POWR1220AT8, MachXO, SPI memory, etc) long enough after a power failure for a logging event to be written to the SPI memory successfully.

Figure 22. 3.3V Power Stays Active During a Power Failure Long Enough to Finish Fault Logging



The external 12V power signal shown in Figure 22 is used to generate +8V raw power (+8V_Raw). The +8V raw signal goes through diode D29 to the input on the 3.3V Low Drop Out (LDO) voltage regulator (U1). This regulator generates the 3.3V power for the critical components on the board. In order to give the key components sufficient time to log a fault event after the 12V external power is removed, the voltage on the input to the 3.3V regulator must be keep high enough (and sufficient current supplied) for the 0.5ms it will take to write the fault data to the SPI memory.

A 47uF capacitor (C4) is connected to the LDO input through a diode (D28) and a parallel resistor (R72). When the 12V external power is applied, C4 is charged through R72. When the 12V external power is removed, C4 will discharge through the D28, the LDO and the load resistance. The voltage on the LDO input will stay high enough to keep the 3.3V output active until the capacitor is sufficiently discharged to 3.5V (or so) as required for the LDO to operate correctly. The time required for C4 to discharge depends on the voltage on C4 when power is removed (around 7V since there is a diode drop from the +8V_raw supply), the voltage required on the LDO input (around 3.5V) and the current required by the critical circuits during fault logging. In order to determine the current requirements a sense resistor (R19) can be used to measure the current required during a logging event. Knowing the current requirement, the time required to successfully log the fault, and the voltage difference, the capacitance required for C4 can be determined by using the following equations.

Voltage = Charge/Capacitance

Voltage = Current * Time/ Capacitance

Capacitance = Time * Current / (VH - VL)

If we assume a target of 0.5ms as a minimum time for fault logging to occur.

Then Capacitance = 0.5ms * 25mA/(7V-3.5V) or 3.57uF. In the Hercules design a capacitance of 47uF was determined to be more than sufficient for a successful logging event to be written to SPI memory.

Note: A more detailed description of the Hercules board-level power rail structure is given in the Common Elements Operational Description section. This includes a more detailed description of how 3.3V power is generated for the POWR1220AT8 and other key circuits.

MCU Command Controller

The LatticeMico8, as shown in Listing 12, is used to control some of the operations of the Hercules demonstration design. It uses the WISHBONE bus () to communicate with the UART reference design block (RD1042), the SPI reference design block (RD1044), the control register block and the LCD, switch and button interface. The full assembly code used to control board operation is included in the Hercules demonstration design files (project/XO/Hercules_demo.asm). There are many useful and well commented routines included in the program that can be used "as-is" or customized for use in a new design. The reader can review these routines by reading that file.

The control register block and the SPI reference design block are used by the LatticeMico8 to connect to the MachXO portion of the Fault Logger reference design (RD1062) via its processor interface. The Fault Logger reference design processor interface consists of two main sections – one that duplicates the SPI Memory interface signals (SS, SCLK, MOSI, and MISO) and one that provides control and status signals (busy, mem_full, ready and mem_clear). The SPI reference design block provides an interface between the Fault Logger and the LatticeMico8 for the SPI memory signals while the control register block provides the interface between the status and control signals and the LatticeMico8. *Note: The processor interface portion of the Fault Logger reference design is used unmodified in the Hercules demonstration design, unlike the previously described interface to the POWR1220AT8.*

The LatticeMico8 can access the SPI memory via the SPI reference design block and the Fault Logger reference design. Handshaking and arbitration logic within the Fault Logging reference design controls access to the SPI memory via a busy and ready handshake with the processor. (The Fault Logging block has priority access to the SPI memory so fault information is not lost.) The LatticeMico8 can read and write to the SPI memory to access fault logging information using a set of user commands via the USB-to-UART Converter and a HyperTerminal program on the PC. See the HyperTerminal User Interface section below for a more detailed description.

Reporting Error Status on the LCD Display

The LCD is used to report the error status during the Hercules Fault Logging demonstration. Because this function is shared between several subsystems it is explained in the Common Elements portion of this section.

Interface to Buttons and Switches

The LatticeMico8 reads the values for the buttons and switches to control the execution of the Hercules Fault Logging demonstration. Because this function is shared between several subsystems it is explained in the Common Elements portion of this section.

HyperTerminal Access to Fault Logger Commands

In addition to the user I/O via the switches, push-buttons, LEDs and LCD display the Hercules Fault Logging demonstration design can use a HyperTerminal link to control aspects of the demonstration. The Hercules Fault Logger command window via HyperTerminal is shown in Figure 23. Possible commands include:

0: Displays the command menu

1: Displays the number of faults since the last reset

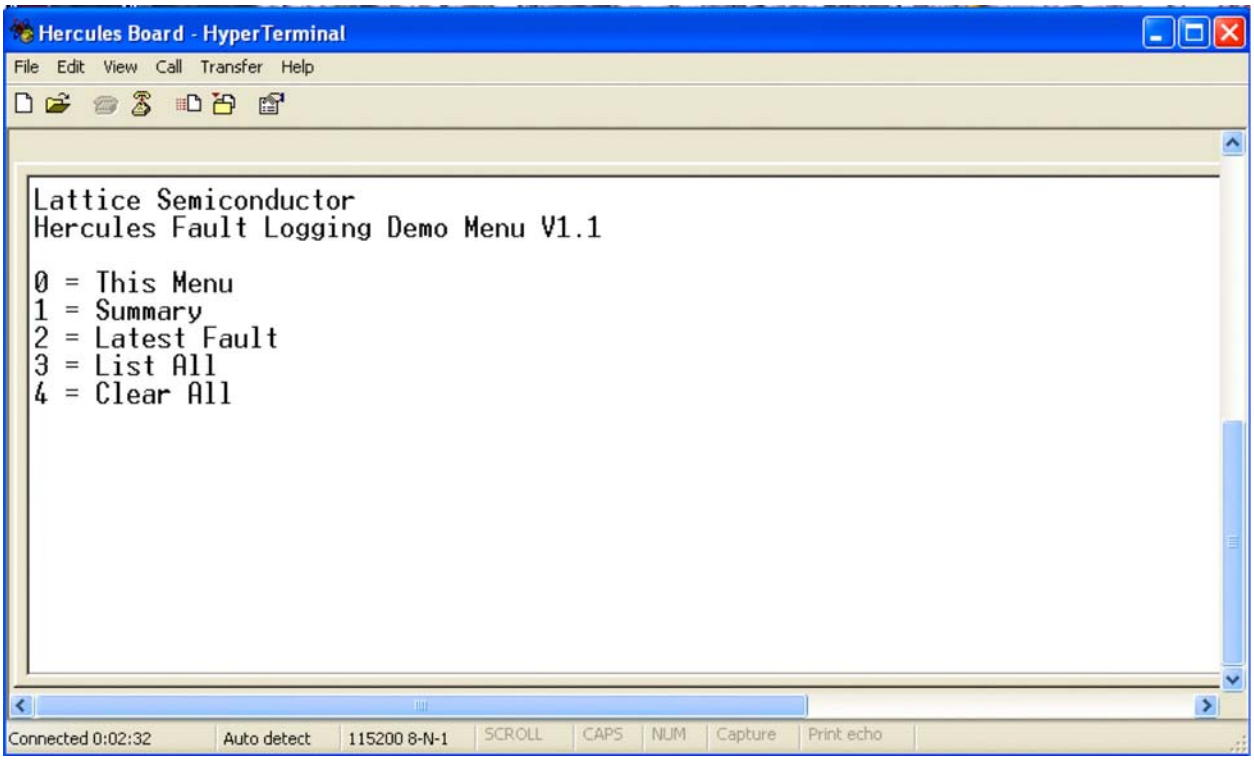
2: Displays the details of the last fault event

3: Cycles through all logged fault events

4: Clears all fault data

Note: A detailed description of the method for configuring the PC for HyperTerminal access of the Hercules Evaluation Board is given in Appendix C.

Figure 23. HyperTerminal Command Window for Hercules Fault Logger Demonstration Design



The HyperTerminal program on the PC uses the USB interface on the Hercules Evaluation Board to communicate with the LatticeMico8. The USB interface on the Hercules Evaluation Board is converted to UART signals (using U13, a USB-to-UART converter) and the LatticeMico8 uses the UART reference design (RD1042) inside the MachXO to receive and transmit data. Single character commands are decoded by the LatticeMico8 and then the command is executed. The source for the command decode and execute function is found in the hercules_demo.asm file in the project/demo_hercules/project/XO folder of the Hercules demonstration design. The commands are all created as stand-alone subroutines that can be useful in creating a custom design. For example, Listing 12 shows the subroutine responsible for displaying the information for the last logged fault. The function of this routine will not be described here. The user can study the full program listing and via in-line comments determine the operation of the code fairly easily. Listing 12 is provided as an example of the type of routines to look for when creating a custom design.

Listing 12. Example Hercules Demonstration Design LatticeMico8 Subroutine

```
;=============================================================================#
; Subroutine: Sub_Fault_Last                                                  #
; Input : none                                                                #
; Output: none                                                                #
; Used  : r0, r2, r4, r5, r9, r14, r15 by                                     #
;         various UART and SPI subroutines.                                   #
; Description: Calls SPI routines to find last fault recorded.                #
;              Reads last fault page, formats and sends the data by UART   # ;
               routines to HyperTerminal display.                            #
;              Called from Sub_uart_parse_input.                              #
;=============================================================================#
Sub_Fault_Last:
  call   Sub_logger_busy
  call   Sub_Fault_Total
```

```
    mov    r2, r4
    or     r2, r2
    bz     fault_last_none
    ; adjust count to page index
    subi   r2, 0x01
    call   Sub_VMON
    call   Sub_Time
    ; check to see if SPI flash is Full
    movi   r15, CONTROL_STATUS
    import r0,  REG_LOG_STATUS
    andi   r0,  LOG_FULL
    bz     Fault_Last_Done
    movi   r9,  MSG_NEW_LINE
    call   Sub_uart_send_msg
    movi   r14, MSG_PAGE_2
    movi   r9,  MSG_SPI_FULL
    call   Sub_uart_send_msg

Fault_Last_Done:
    call   Sub_logger_release
    b      Fault_done

fault_last_none:
    call   Sub_logger_release
    movi   r9, MSG_NEW_LINE
    call   Sub_uart_send_msg
    movi   r9,  MSG_NO_FAULTS_D
    movi   r14, MSG_PAGE_2
    call   Sub_uart_send_msg
    b      Fault_done
```

MCU Command Controller Reference Designs

The following reference designs used in the MCU Controller are used unmodified in the Hercules demonstration design. Users can consult documentation on the Lattice web site for more information.

- RD1042, WISHBONE UART
- RD1044, SPI WISHBONE Controller
- RD1043, LatticeMico8 to WISHBONE Interface Adapter
- RD1026, LatticeMico8 Microcontroller User's Guide

Operational Description – Trim and Margin Up/Down

This section describes, in detail, the operation of the Trim and Margin Up/Down functions of the demonstration design. It shows how the design works in sufficient detail so that it can be used as a reference design and easily modified, if required, for a target application. The section is organized around the following key topics:

- Overview of Trimming and Margining
- Review of the Trim and Margin Up/Down Demonstration Operation
- Overview of Key Components, Functions and Algorithm of the Trim and Margin Up/Down Design
  - Margin and Trimming Circuits
    - Overview of the Trimming and Margining Block of the POWR1220AT8
    - POWR1220AT8 Trim/Margin Operation Modes
    - Trim and Margin Resistor Network

  – MCU Controller

- Implementation of the Trim Function in the Hercules Design
- Implementation of the Margin Up/Down Function in the Hercules Design

Overview of Trimming and Margining

Trimming and margining are very similar functions that use many of the same elements in the POWR1220AT8 device. Margining is typically used to adjust a power supply to an extreme of the normal operating range – an upper or lower operating margin, usually a few percent of the factory value of the supply. This can be used during testing to set supplies at the upper or lower range of operation as a 'stress test' for weak electronics components. Thus, a typical margining function might run a supply at plus 5% or minus 5%.

Note: Margining can be done either in an open loop mode, where the voltage is set by the trim output (DAC) but not adjusted, or in a closed loop mode where a voltage monitor signal (ADC) is used to measure the output voltage and the trim output (DAC) is automatically adjusted to keep the supply output voltage at the target voltage level. The POWR1220AT8 device supports both modes of operation.

The trimming function provides a much more precise and flexible setting for a voltage output and can involve a more complex feedback or control function. For example, trimming could be used to adjust a supply to a precise value, perhaps to reduce the voltage on a high-performance processor when it is running at a slower operating frequency, and thus reducing operating power. The supply level can be increased when faster processor operation is required. On the POWR1220AT8, trim outputs can be adjusted over a full range of 320mV with a step size of 2.5mV.

Review of the Trim and Margin Up/Down Demonstration Operation

Trim Demonstration Operation

During Quick Start Guide Demo 1 the POWR1220AT8 trims the 1.2V switch mode supply and the output voltage level is measured and displayed on the LCD panel. When the mode switch (SW3) is not pressed the external closed loop trimming function is used and the value of the Trim-DAC is automatically adjusted to keep the output voltage at the target voltage level. This results in a more accurate voltage level. When the mode switch (SW3) is pressed the open loop mode is selected and the Trim-DAC output is set to a fixed (stored) value. This results in a less accurate voltage level.

Note: The difference in output voltage, from closed loop to open loop modes, is the result of combined errors in the DAC output, resistor tolerances, initial accuracy of the DC-DC converter, and voltage drops across traces and connections.

Margin Up/Down Demonstration Operation

During the Margin Up/Down demonstration the 1.2V supply is trimmed to plus (Demo 2a) or minus (Demo 2) 7.5% of its factory value (1.2V - 0.09V = 1.11V, or 1.2V + 0.09V = 1.29V). This is sometimes also referred to as margining the supply by 7.5%.

In the closed loop mode (the mode switch, SW3, is not pressed), VMON 1 is used to measure the 1.2V DC-DC power supply. The value is read by the POWR1220AT8 ADC and is displayed on the LCD. Pressing the mode switch changes the trim mode switching from closed loop to open loop trim.

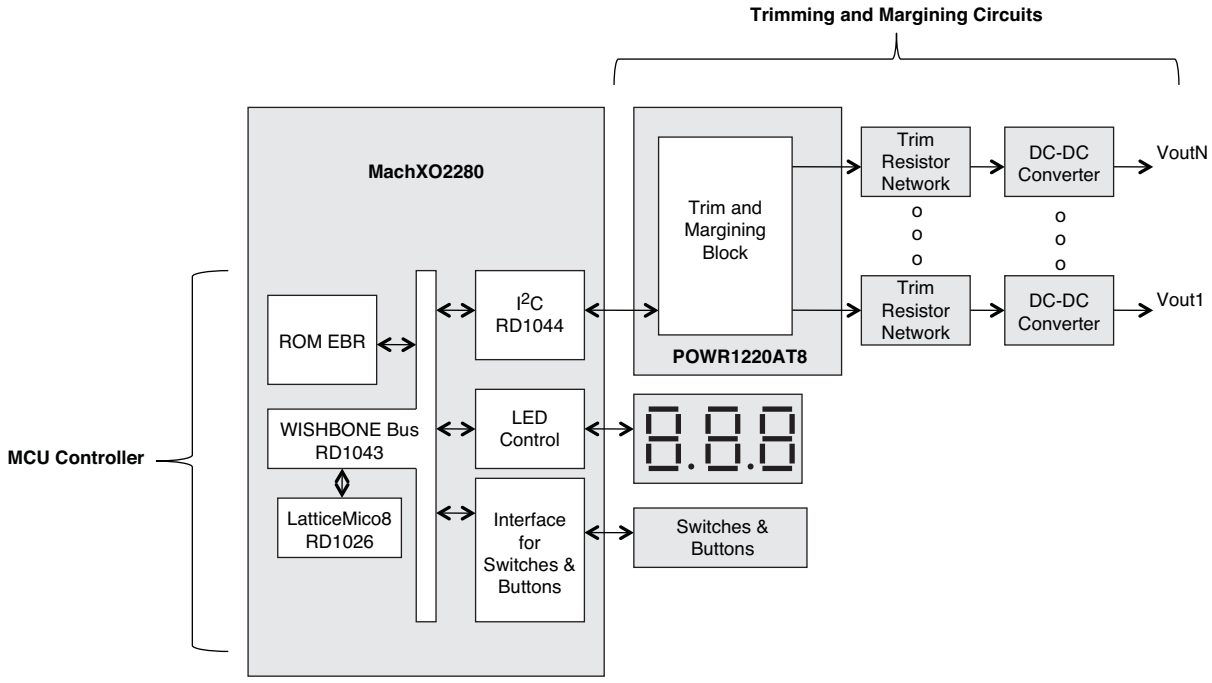
In the open loop mode (the mode switch, SW3, is pressed), the output voltage is set to a fixed (stored) digital value which is selected by the Voltage Profile Select pins of the POWR1220AT8. The VMON1 input value is measured by the ADC and the value (either 1.29V or 1.11V depending on the demo selected) is displayed on the LCD.

Overview of Key Components, Functions and Algorithms – Trim and Margin Up/Down Subsystem

Several parts of the Hercules Evaluation Board work together in the Quick Start Guide Demonstration Design to implement the Trim and Margin Up/Down subsystem (Hercules demonstration Commands 1, 2 and 2a). The key

subsystem functions, shown in Figure 24 can be categorized as the Trimming and Margining Circuits and the MCU Controller. The Trimming and Margining circuits implement fine control over the DC-DC converter voltage outputs by adjusting the DC-DC converter trim inputs. The MCU Controller implements the user interface by reading commands on the switches, decoding and executing the command and displaying results on the LCD display. In the closed loop trim mode the MCU is included in the feedback loop required to maintain the output voltage of the DC-DC converters at the user-selected value. The key elements and operation of the Trimming and Margining Circuits will be described first, followed by the MCU Controller.

Figure 24. Trimming and Margining Subsystem Key Elements

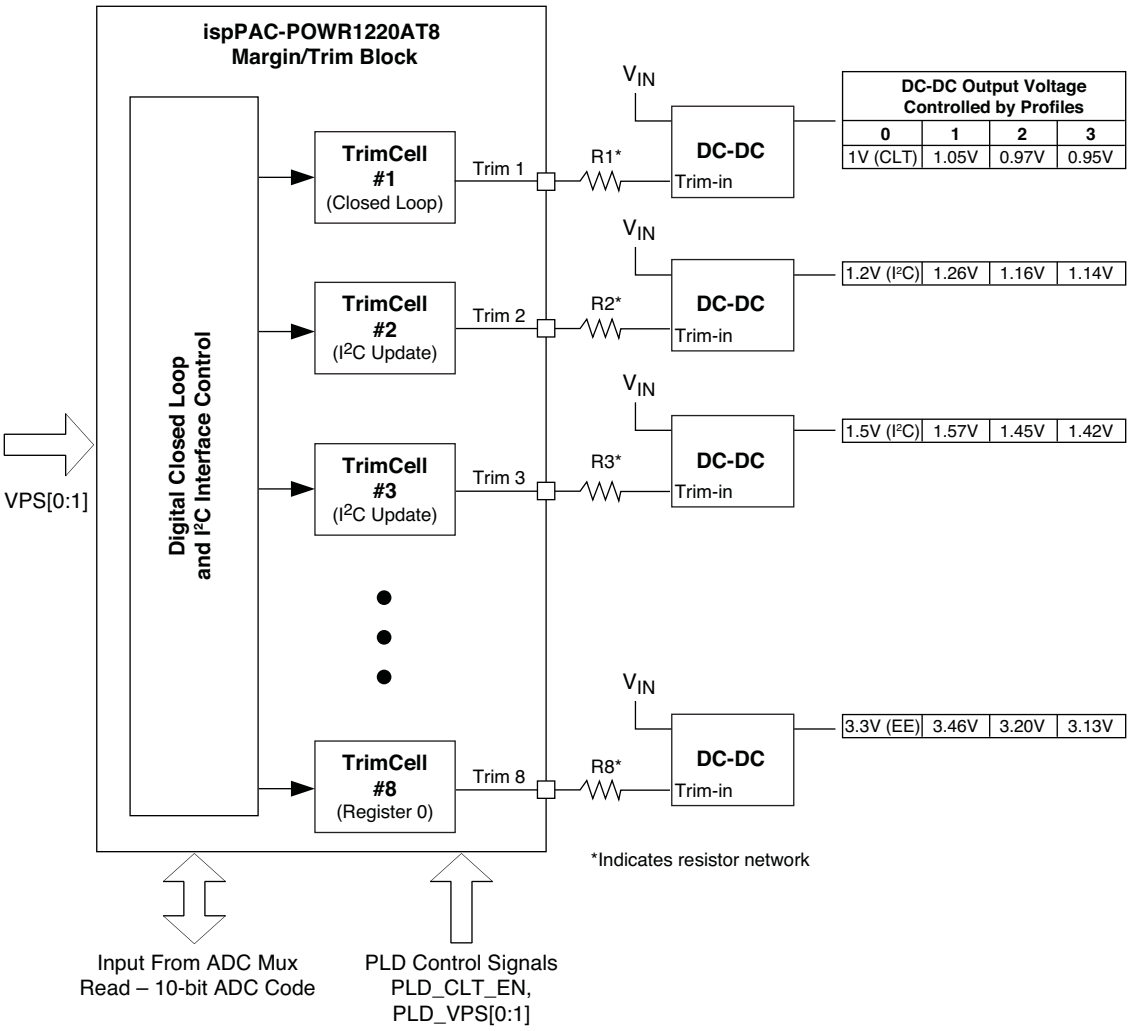


Hercules Trimming and Margining Circuits Overview

The Trimming and Margining Circuits portion of the subsystem contain the POWR1220AT8, the DC-DC converters and the required external trim resistor network support circuits. The main portion of the POWR1220AT8 used for this subsystem is the Trim and Margining Block.

POWR1220AT8 Trimming and Margining Block: One of the key features of the POWR1220AT8 is its ability to make adjustments to the power supplies that it may also be monitoring and/or sequencing. This is accomplished through the Trim and Margin Block of the device. The Trim and Margin Block contains eight voltage output DACs that can adjust voltages of up to eight different power supplies. These DACs are combined with control logic to create special-purpose TrimCells that implement the trimming and margining functions. Figure 25 illustrates these functions for an example design with four different voltage profiles for each DC-DC converter programmed into the POWR1220AT8 Trim and Margin block.

Figure 25. POWR1220AT8 Margin/Trim Block in an Example Design



The DC-DC blocks in the figure represent virtually any type of DC power supply that has a trim or voltage adjustment input. The interface between the POWR1220AT8 and the DC power supply is represented by a single resistor (R1 to R8) to simplify the diagram. Each of these resistors represents a resistor network. The resistor network is explained in more detail in the next section. Other control signals driving the Margin/Trim Block include:

- **VPS [0:1]** – Control signals from the device pins common to all eight TrimCells, which are used to select the active voltage profile for all TrimCells simultaneously.
- **PLD_VPS[0:1]** – Voltage profile selection signals generated by the PLD. These signals can be used instead of the VPS signals from the pins.
- **ADC input** – Used to determine the trimmed DC-DC converter voltage.
- **PLD_CLT_EN** – Only from the PLD, used to enable closed loop trimming of all TrimCells simultaneously.

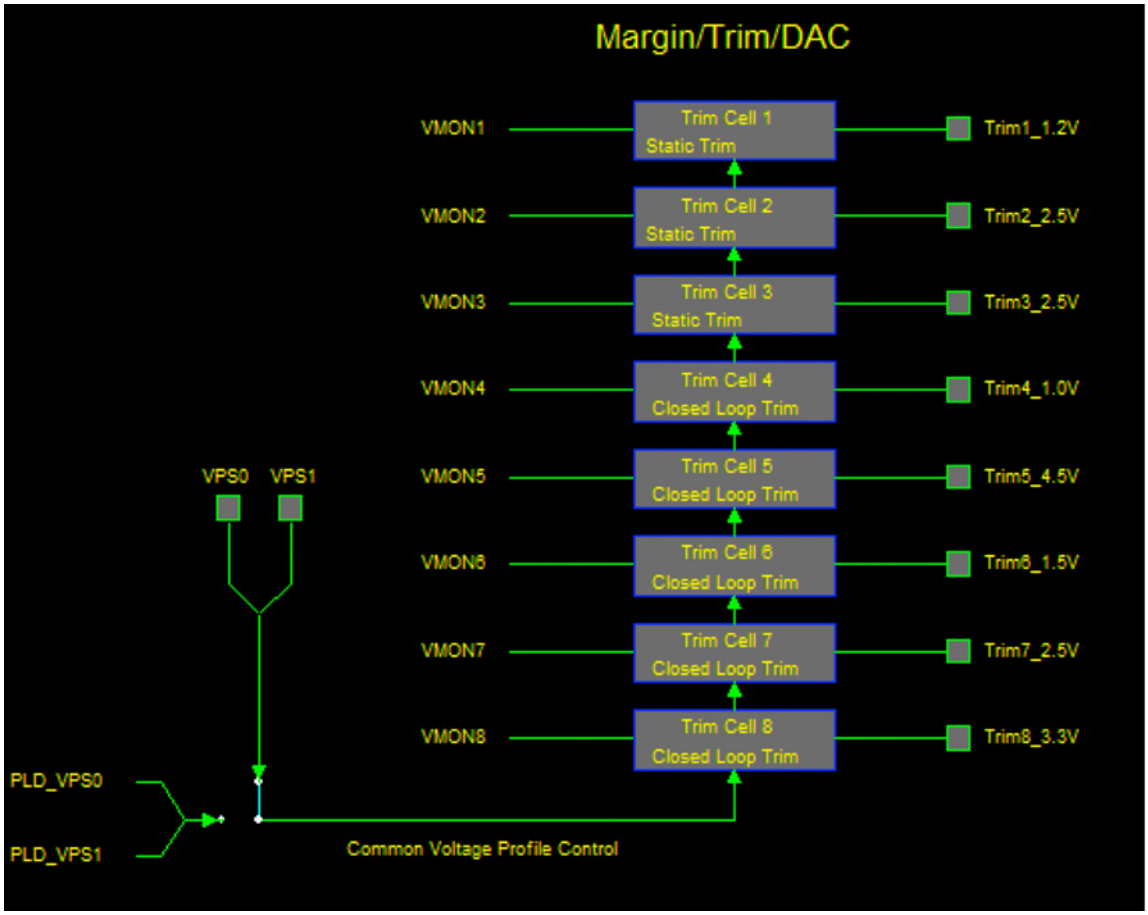
Next to each DC-DC converter, four voltages are shown. These voltages correspond to the operating voltage profile of the Margin/Trim Block for a particular design:

- When the VPS[0:1] = 00, representing Voltage Profile 0: The output voltage of the DC-DC converter controlled by the Trim 1 pin of the POWR1220AT8 will be 1V and that TrimCell is operating in closed loop trim mode. At the same time, the DC-DC converters controlled by Trim 2, Trim 3 and Trim 8 pins output 1.2V, 1.5V and 3.3V respectively.

- When the VPS[0:1] = 01, representing Voltage Profile 1 being active: The DC-DC output voltage controlled by Trim 1, 2, 3, and 8 pins will be 1.05V, 1.26V, 1.57V, and 3.46V. These supply voltages correspond to 5% above their respective normal operating voltage (also called as margin high).
- When VPS[0:1] = 11, all DC-DC converters are margined low by 5%.

These pre-programmed values allow the POWR1220AT8 to implement a variety of power supply profiles without the need for an external microcontroller. The screen shot of the Margin/Trim block from PAC-Designer is shown in Figure 26 and illustrates the shared use of the VPS0 and VPS1 signals between all eight TrimCells. In the Hercules design, the VPS1 and VPS0 pins are controlled by the MachXO using the XO_PWR2 and XO_PWR3 signals (Refer to the Hercules Schematic page 2 on the upper left).

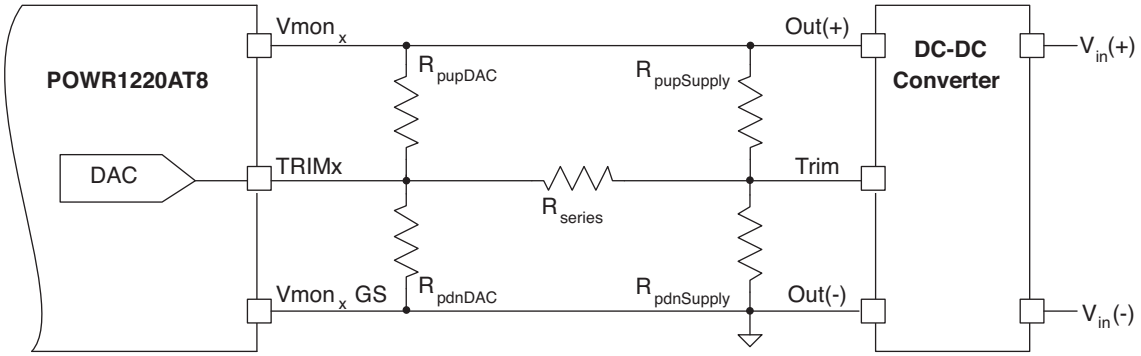
Figure 26. Margin Trim Block of the POWR1220AT8 Showing Common VP Signals



POWR1220AT8 Trim/Margin Operation Modes: The POWR1220AT8 device can implement a trim function for a DC-DC converter in three different operation modes: E^{2}CMOS®, I^{2}C and Internal Closed Loop. In the E^{2}CMOS mode the PAC-Designer software calculates the DAC code required to achieve the desired supply output voltage. This code is transferred to the device during JTAG programming. This is also called the Open Loop Trimming mode since no feedback adjustments are made to more precisely control the output voltage. In the I^{2}C mode the DAC automatically loads a hexadecimal value of 80 on power-up; this code corresponds to the bipolar zero voltage. An external microcontroller can write to or read from the I^{2}C register associated with the DAC. In this manner, the microcontroller can control the output voltage of the power supply, making adjustments as many times as necessary. This mode is also called External Closed Loop Trimming since the feedback loop is implemented with an external controller. In the Internal Closed Loop mode the POWR1220AT8 automatically (without external control) adjusts the output voltage to achieve the voltage defined by the DAC trim register. This closed loop system typically achieves accuracy of 0.75% or better.

Trimming and Margining Resistor Network: DC-DC converters usually provide a mechanism to allow the user to adjust the output voltage by adding one or more external components. Typically, this mechanism is a single pin which can be left floating to allow the supply to run at its nominal output voltage or this pin can be pulled toward ground or toward the output via a resistor in order to adjust the output voltage up or down. The POWR1220AT8 improves on a fixed implementation by using a DAC to control the trim input on the DC-DC converter. The DAC output can be adjusted as needed to push the supply voltage up or bring it down in small increments. An external resistor network is added between the POWR1220AT8 and the DC-DC converter as shown in Figure 27.

Figure 27. External Resistor Network for Trimming DC-DC Converters from POWR1220AT8



The values of the five resistor network shown in Figure 17 vary depending on the type of DC-DC converter, the voltage range of the trim input and the precision of the control desired. Lattice has included a useful calculator tool within the PAC-Designer software that determines which of the resistors need to be used and the values of these components. Actual trim networks contain between one and three resistors. Rseries is always needed to couple the DAC output to the trim pin of the DC-DC converter; in some cases, its value may be zero. For a given output voltage, the values of the resistors in this network will be different than those indicated by voltage programming tables or equations in the DC-DC converter data sheet.

MCU Controller Overview

The MachXO device controls most of the demo operations using an internal LatticeMico8 microcontroller. The microcontroller reads the VID-DIP switch (SW5) to decode the demo command, controls operation of the POWR1220AT8 and reports status to the user via the LEDs and LCD panel. The details of the command decode, reading switches and buttons and writing data to the LCD display are common functions to most of the demonstration commands and are described in detail in the Common Elements section of this document. The closed loop algorithm implemented by the MCU Controller is described in the implementation portion of this section.

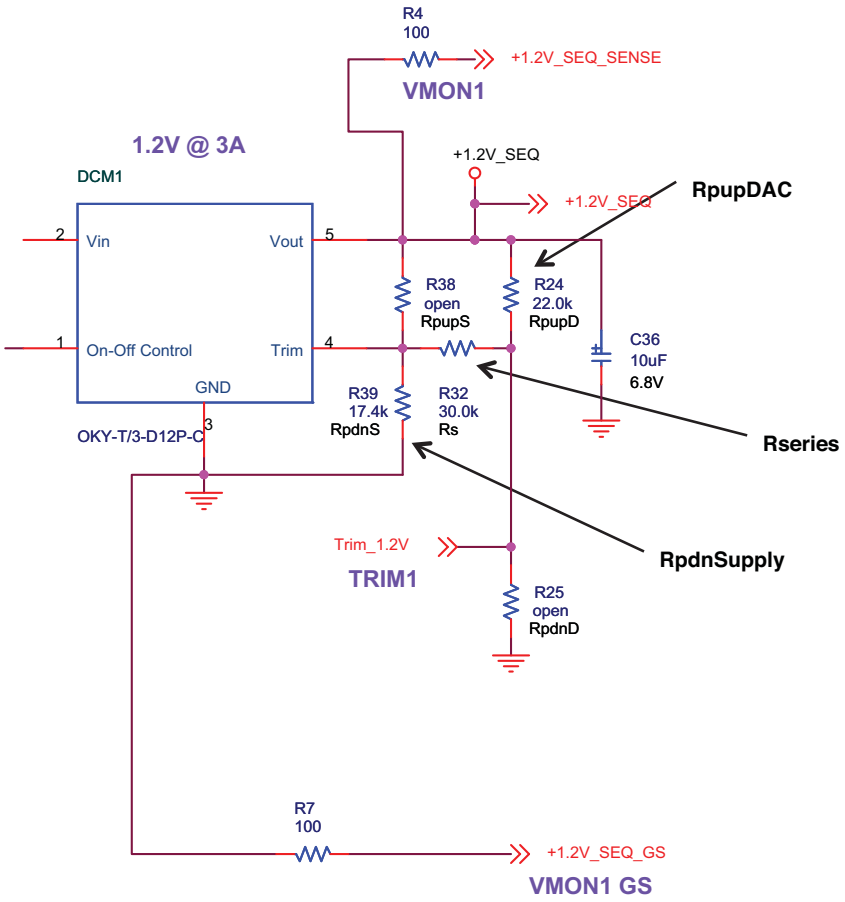
Implementation of the Trim Function in the Hercules Design

The Hercules design implements the trimming function to control the 1.2V supply voltage to better than 1% of its factory-set value by using either the external closed loop trimming algorithm or the open loop algorithm depending on the setting of SW3. The implementations for the Trimming and Margining Circuits and MCU Controller used in the Hercules design are described below.

Trimming and Margining Circuits Implementation in the Hercules Design

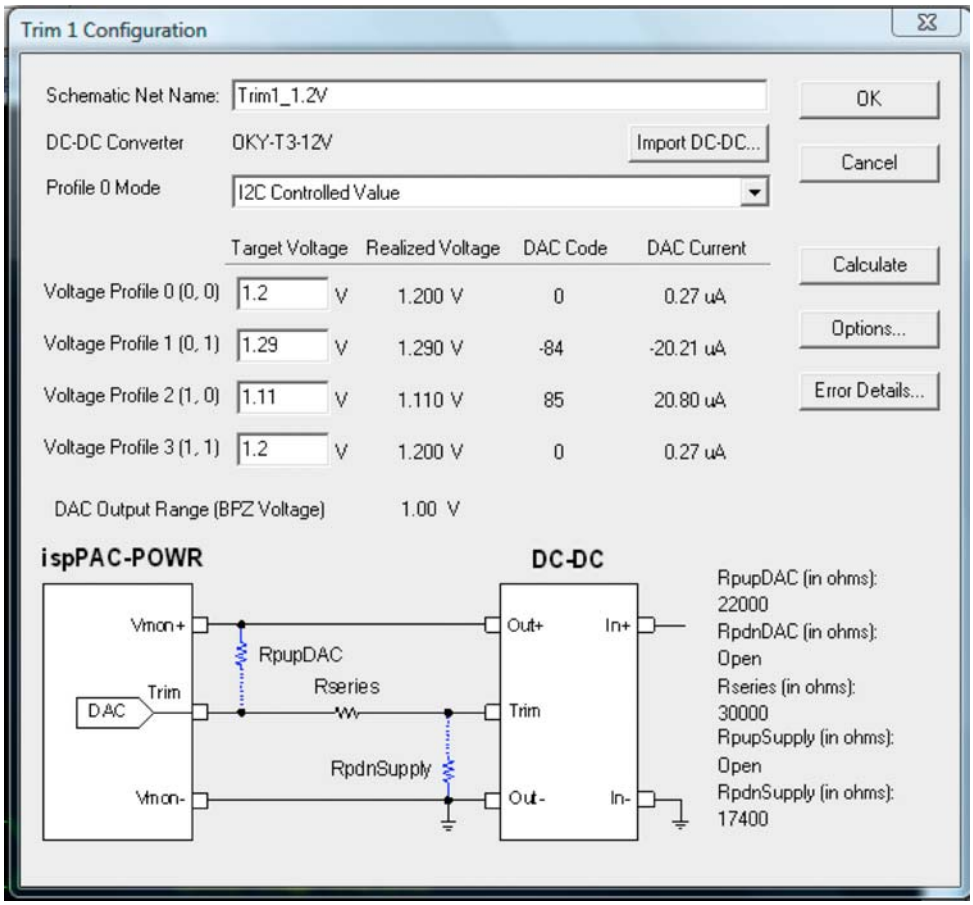
The circuit used to create the trim input for the 1.2V DC-DC converter is shown in the Hercules Evaluation Board schematic in Appendix A. The relevant portion is shown in Figure 28.

Figure 28. 1.2V Trim Circuit



The trim output from the POWR1220AT8 is labeled at the middle of the figure as TRIM1. This is the Trim_1.2V signal shown in the PAC-Designer Trim Dialog box in Figure 29. The voltage monitored by the POWR1220AT8 is VMON1. The three resistors populated in the Hercules design (R24, R32, R39) correspond to the resistors automatically calculated by PAC-Designer and called out in the Trim Dialog Box (RpupDAC, Rseries and RpdnSupply). The Hercules resistor values also correspond (22K, 30K and 17.4K) to those shown in the Trim Dialog Box.

Figure 29. 1.2V Trim Circuit Configuration Dialog Box in PAC-Designer



Note: The POWR1220AT8 also has an internally controlled closed loop trim function that does not require external control. Refer to the ispPAC-POWR1220AT8 Data Sheet for more details on this function.

The resistors R4 and R7 in Figure 28, support off-board differential sensing or on-board differential sensing and help reduce sensing errors due to variations in board trace impedance. The 100 ohms is in-series with the 65k (typical) VMON input impedance.

Note: Other DC-DC converters used in the Hercules design will have trim resistor networks implemented based on the associated PAC-Designer Trim Dialog boxes. The user can run PAC-Designer to verify this correspondence.

MCU Controller Implementation of the 1.2V Supply Trimming Functions in the Hercules Design

In this design the 1.2V supply output voltage is controlled to less that 1% of its factory value. The MCU Controller runs either the open loop trim function or the closed loop trim function depending on the setting of SW3. The selection of either the open loop or closed loop POWR1220AT8 functions are done via control of the Voltage Profile pins.

In the Hercules design the Voltage Profile Select pins (VPS0, VPS1) of the POWR1220AT8 are controlled by the MachXO and thus the LatticeMico8 code. For external closed loop trim VPS0 and VPS1 are both set low (logic 0) to select Trim-DAC control register zero. This register is preconfigured in PAC-Designer to be controlled by the I^2C interface as shown in Figure 29.

In the open loop mode (the mode switch, SW3, is pressed), VPS0 and VPS1 are both set high (logic 3) to select Trim-DAC control register 3. This register is pre-configured in PAC-Designer to store a preset value of zero, which corresponds to a DC-DC output of 1.2V as shown in Figure 29.

The implementation descriptions of both the open loop and closed loop functions within the MachXO device using the LatticeMico8 microcontroller are given below.

Open Loop Trimming Implementation Description in the Hercules Design: The LatticeMico8 microcontroller implements the 1.2V supply open loop trimming design in the main_demo_trim_nominal portion of the LatticeMico8 assembly code as shown in Listing 13. The code first checks to see if the new trim value stored in r4 is different from the last trim value stored in r17. If it is not, control jumps to the demo_trim_nominal_sw3 location at Line 14. If it is, the Trim DAC is set to near target value to prevent a slow slewing of the output voltage when changing from one mode to another. Slow slewing can result since the DAC is being updated once every 30mS.

Once at the demo_trim_nominal_sw3 location at Line 14 the program checks if SW3 is pressed. If SW3 is not pressed, then control jumps to the External Closed Loop demo_trim_nominal_i2c routine at Line 30 (which is explained in the following External Closed Loop description section). If SW3 is pressed then control continues and sets up the call to the Sub_1220_Margin subroutine. The CLT register is set to 0x00 to clear the register. The CLT register is used in the Closed Loop Trim routine and is described later in that section. The voltage profile pins are moved to r5 to select the pre-programmed 1.2V value at location 1:1 (refer to Figure 29 if needed to see the various voltage profile voltage settings displayed). The subroutine writes the selected Voltage Profile value to the MachXO output pins which drive the Voltage Profile inputs on the POWR1220AT8. Refer to the subroutine code at the bottom of the Listing 13 for more details. Control is then transferred to demo_trim_done at Line 41.

Listing 13. LatticeMico8 Open Loop Trimming Assembly Routine

```
1  ; Trim-Nominal Demo Mode
2  main_demo_trim_nominal:
3     ; If last Trim was different then update the DAC
4     ; to prevent slow slewing voltage
5     Cmp    r4,  r17
6     bz     demo_trim_nominal_sw3
7     ; update the "last" trim value
8     mov    r17, r4
9     ; set DAC mid scale
10    movi   r3,  0x00 ; Trim DAC1
11    movi   r4,  0x60 ; near mid scale value
12    Call   Sub_i2c_Write_DAC
13
14  demo_trim_nominal_sw3:
15    ; If SW3 is pressed, use Voltage Profile
16    call   Sub_i2c_Read_SW3
17    andi   r5,  0x01
18    bnz    demo_trim_nominal_i2c
19
20    ; clear the CLT register, can't be done if in open loop
21    Movi   r18, 0x00
22
23    ; set the Voltage Profile Pins to 1:1 Margin Nominal
24    Movi   r5,  0xc0
25    call    Sub_1220_Margin
26
27   ; end of Open Loop Trim Nominal: repeat main loop
28    b       demo_trim_done
29
30 demo_trim_nominal_i2c:
31   ; set the Voltage Profile Pins to 0:0 I2C External Closed Loop Trim
32   Movi    r5,  0x00
33   call    Sub_1220_Margin
```

```
34
35   ; load r8 & r9 with ADC target value
36   ; shift up by 4-bits to match I2C format
37   Movi    r8,  0x8f ; 1.20 V low nibble + 0x0f (111 and Done Bit)
38   movi    r9,  0x25 ; 1.20 V high byte
39   call    Sub_1220_Trim
40
41  demo_trim_done:
42   ; Enable 1.2V supply by setting Power Manager IN5 pin High
43   movi    r5,  0x10      ; Power Manager IN5 pin
44   call    Sub_LED_set
45
46   ; update the Power OR-ing LEDs
47   call    Sub_1220_Power_OR
48
49   ; update the LDC display with selected VMON
50   ; Re-read VID Dip Sw Setting and save in r3
51   movi    r15, LED_SW
52   import  r3,  REG_DIP_SW
53   andi    r3,  0x0f
54   call    Sub_1220_Display_VMON
55
56   ; provide a delay in the I2C traffic to the 1220AT8
57   ; and debounce the VID dip switch
58  main_delay:
59   ; 30 ms  = 0x3c67 @ 8 MHz
60   ; 30 ms  = 0xf19c @ 33 MHz
61   movi    r1, 0x9c
62   movi    r2, 0xf1
63   call    Sub_delay_small
64
65   call    Sub_uart_check4data
66   andi    r5, 0x01
67   bz      main
68
69   call    Sub_uart_parse_input
70
71   ; Restart the main loop
72   b       main
73 ;=====================================================================#
74 ; Subroutine: Sub_1220_Margin                                         #
75 ; Input : r5 = Voltage Profile Pins value                            #
76 ; Ouput : r4 = LED status register value                             #
77 ; Used  : R15 = LED_SW Device address                                #
78 ; Description: Read and modify the LED status register.             #
79 ;    The upper 2 bits control the Power Manager Voltage Profile pins #
80 ;    VPS0 = 0x40                                                     #
81 ;    VPS1 = 0x80                                                     #
82 ;=====================================================================#
83 Sub_1220_Margin:
84   movi    r15, LED_SW
85   import  r4,  REG_LED
86   andi    r5,  0xc0    ; mask all but the VPS bits
87   andi    r4,  0x3f    ; AND Off the Voltage Profile Pins
```

```
88   or      r4,  r5      ; OR  On the passed value
89   export  r4,  REG_LED
90
91   ret
```

The demo_trim_done routine at Line 41 executes several clean-up functions. It enables the 1.2V supply, makes a call to the Redundant Power-OR'ing routine to check for and make updates if any changes occurred, updates the LCD display with the VMON output, provides a delay to control the I^2C traffic and to debounce the VID switch. Finally, it checks the UART routine to see if any command has arrived for the Fault Logging routine, if it has control transfers to that routine. If not, it branches to the top of the main program loop to check for a new command.

Closed Loop Trimming Implementation Description in the Hercules Design: In the Hercules design the closed loop trimming of the 1.2V DC-DC converter is implemented using an external closed loop trim function. The Closed Loop routine demo_trim_nominal_i2c is shown in the middle of Listing 13 at Line 30. It sets the profile pins to 0:0 (the external closed loop Voltage Profile setting) by calling the Sub_1220_Margin subroutine. It then sets up the call to the Sub_1220_Trim subroutine by loading the target voltage value (1.2V) in r8 and r9. The listing for the Sub_1220_Trim subroutine is given in Listing 14. A quick review of the architecture of the POWR1220AT8 is given in the VID-Trimming Operational Description section. Users unfamiliar with the architecture of the ADC block might want to read this description prior to exploring the details of the Sub_1220_trim subroutine.

The Sub_1220_Trim subroutine reads the current value of VMON and compares it to the target value stored in registers r9 and r8. If the actual voltage is over the target then the routine bumps the Trim DAC up (DC-DC has inverted control). If the actual voltage is less than the target the routine bumps the Trim DAC down. It then updates the CLT Status register to return either that the algorithm is slewing (all 1's or all 0's = slewing) or that it is locked (mixed 1's and 0's = dithering = locked). Note that the CLT register is updated by shifting the carry bit into the register. If after several loops through the routine r18 is all 1's or all 0's the comparison is always the same. The returned value can be used by a later routine to adjust the DAC level to bring the comparison in range.

Listing 14. LatticeMico8 Sub_1220_Trim Assembly Routine

```
    ;=============================================================================#
    ; Subroutine: Sub_1220_Trim                                                  #
    ; Input : r8  = ADC target value low byte                                    #
    ;         r9  = ADC target value high byte                                   #
    ;         r18 = Closed Loop Trim Status (all 1's or all 0's = slewing)       #
    ; Output: r5 = Final Trim DAC value                                          #
    ;         r18 = Closed Loop Trim Status (mixed 1's and 0's = locked)         #
    ; Used  : r15 = By I2C subroutines                                           #
    ;         r10 = Trim DAC register + / - one bit based on VMON value          #
    ;         r5  = ADC result high byte                                         #
    ;         r4  = ADC result low byte                                          #
    ;         r3  = VMON and Trim DAC number (0 - 7).                            #
    ; Description: Read current value of VMON and compare to target value       #
    ;(r9,r8). If actual is over target then bump Trim DAC up (DC/DC has          #
    ; inverted control). If acutal is less than target bump Trim DAC down.      #
    ; Update the CLT Status register                                             #
    ; (all 1's or all 0's = slewing: mixed = dithering = locked).               #
    ;=============================================================================#
1   Sub_1220_Trim:
2      ; Read and save current Trim DAC value
3      movi     r3,  0x00                  ; Trim DAC1
4      call     Sub_i2c_Read_VMON   ; r5 = hi, r4 = lo
5      mov      r10, r5       ; save DAC value for later update
6
7      ; Read actual VMON value
```

```
8     movi     r3,  0x00           ; VMON1 no attenuation
9     call     Sub_i2c_Read_VMON   ; r5 = hi, r4 = lo
10
11    ; Lowest nibble is undefined so force it to match the target values
12    ori   r4,  0x0f
13
14    ; Subtract the acutal voltage from target
15    sub      r8, r4
16    subc     r9, r5
17    bnc   Trim_DAC_down
18
19    ; Carry: then Actual > Target : Adjust DAC up
20    rolc       r18, r18 ; update the CLT register
21    addi     r10, 0x01
22    bnc      Trim_DAC_update
23    movi     r10, 0xff
24    b        Trim_DAC_update
25
26  Trim_DAC_down:
27    ; No carry: then Actual < Target : Adjust DAC down
28    rolc   r18, r18 ; update the CLT register
29    subi     r10, 0x01
30    bnc      Trim_DAC_update
31    movi     r10, 0x00
32
33  Trim_DAC_update:
34    movi     r3,  0x00
35    mov      r4,  r10
36    call  Sub_i2c_Write_DAC
37
38   ret
```

Implementation of the Margin Up/Down Functions

The Hercules design implements the Margin Up/Down functions to trim the 1.2V supply voltage to either +7.5% or -7.5% of its factory set value by using either the external closed loop trimming algorithm or the open loop algorithm depending on the setting of SW3. The implementations for the Trimming and Margining Circuits and MCU Controller used in these function of the Hercules design are described below.

Trimming and Margining Circuits for Margin Up/Down Function in the Hercules Design

The Trimming and Margining Circuits used in the Margining Up/Down functions are the same as those used in the previously described Trimming function. Refer to that section for details.

MCU Controller Implementation of Margin Up/Down Function in the Hercules Design

In the Margin Up/Down demo, the MCU Controller runs either the open loop trim function or the closed loop trim function depending on the setting of SW3. The implementation description of this function within the MachXO device using the LatticeMico8 microcontroller is given below.

In the Hercules design the control of the 1.2V DC-DC converter in the Margin Up/Down functions is implemented by the LatticeMico8 microcontroller. The LatticeMico8 code for implementing the Trim-Up Command is given in Listing 15. First the trim value is checked to see if it is different than the previous value. If it is different the DAC will be updated setting it to near full scale to prevent slow slewing of the voltage. Slow slewing can result since the DAC is being updated once every 30ms.

Once at the demo_trim_up_sw3 location the program checks if SW3 is pressed. If SW3 is not pressed, then control jumps to the External Close Loop demo_trim_up_i2c routine (described in the following External Closed Loop description section). If SW3 is pressed then control continues and sets up the call to the Sub_1220_Margin subroutine. The CLT register is set to 0x00 to clear the register. The CLT register is used in the Closed Loop Trim routine and is described later in that section). The voltage profile pins are moved to r5 to select the pre-programmed 1.29V value at location 0:1 (refer to Figure 19 if needed to see the various voltage profile voltage settings displayed). The subroutine writes the selected Voltage Profile value to the MachXO output pins which drive the Voltage Profile inputs on the POWR1220AT8. Refer to the subroutine code at the bottom of the Figure 13 for more details. Control is then transferred to demo_trim_done.

Listing 15. LatticeMico8 Open Loop Trimming Assembly Routine

```
1; Trim-Up Demo Mode
2main_demo_trim_up:
3   ; If last Trim was different then update the DAC
4   ; to prevent slow slewing voltage
5   Cmp     r4,  r17
6   bz      demo_trim_up_sw3
7   ; update the "last" trim value
8   mov     r17, r4
9   ; set DAC full scale (DC-DC has negative gain on trim pin)
10   movi    r3,  0x00 ; Trim DAC1
11   movi    r4,  0x18 ; negative near full scale value
12   Call    Sub_i2c_Write_DAC
13
14 demo_trim_up_sw3:
15   ; If SW3 is pressed, use Voltage Profile
16   call    Sub_i2c_Read_SW3
17   andi    r5,  0x01
18   bnz     demo_trim_up_i2c
19
20   ; clear the CLT register, can't be done if in open loop
21   Movi    r18, 0x00
22
23   ; set the Voltage Profile Pins to 0:1 Margin High
24   Movi    r5,  0x40
25   call    Sub_1220_Margin
26
27  ; end of Open Loop Trim Up: repeat main loop
28   b       demo_trim_done
29
30 demo_trim_up_i2c:
31   ; set the Voltage Profile Pins to 0:0 I2C External Closed Loop Trim
32   Movi    r5,  0x00
33   call    Sub_1220_Margin
34
35   ; load r8 & r9 with ADC target value
36   ; shift up by 4-bits to match I2C format
37   Movi    r8,  0x5f ; 1.29 V low nibble + 0x0f (111 and Done Bit)
38   movi    r9,  0x28 ; 1.29 V high byte
39   call    Sub_1220_Trim
40
41  ; end of Closed Loop Trim Up: repeat main loop
42   b       demo_trim_done
```

Closed Loop Trimming Implementation- Margin Up/Down Function in the Hercules Design

In the Hercules design the closed loop trimming of the 1.2V DC-DC converter is implemented using an external closed loop trim function. The closed Loop routine demo_up_i2c is shown at the end of Listing 15. It sets the profile pins to 0:0 (the external closed loop Voltage Profile setting) by calling the Sub_1220_Margin subroutine. It then sets up the call to the Sub_1220_Trim subroutine by loading the target voltage value (1.29V) in r8 and r9. The listing for the Sub_1220_Trim subroutine was given in Listing 14 previously. The detailed description of the subroutine operation is given in that section. It is not repeated here.

Operational Description – VID Trimming

This section describes, in detail, the operation of the VID Trimming function of the demonstration design. It shows how the design works in sufficient detail so that it can be used as a reference design and easily modified, if required, for a target application. The section is organized around the following key topics:

- Overview of VID Trimming
- Review of the VID Trimming Demonstration
- Overview of Key Components, Functions and Algorithm of the VID Trimming Design
  - VID Trimming Circuits
  - ADC Block Architecture
  - MCU Controller
- Implementation of the VID Trimming Function in the Hercules Design
- Implementation of the 1.2V Supply Turn ON/Off Function in the Hercules Design

Overview of VID Trimming

VID trimming is very similar to the previously-described Trimming and Margining functions. In VID Trimming the target voltage can be selected by the user. It is not a "hard-wired" value stored in the POWR1220AT8 as it was in the Trimming and Margining demonstration. The VID trimming function is useful when a power supply value needs to be changed based on operation conditions. For example, the voltage to a processor might be adjusted based on the frequency of operation. For a higher frequency operation, a higher voltage is required. For a lower performance operation a lower voltage can be used. This can reduce power consumption in power-critical applications.

Review of the VID Trimming Demonstration

In this demonstration the Trim DAC output of the POWR1220AT8 is used to fine-tune (trim) the 1.2V DC-DC supply based on the value of the VID-DIP switches. Positions 3-0 of SW5 provide a 4-bit unsigned binary value ranging from 0-15. This value is multiplied by 10mV to create the trim value. For example, 0000b=0*10mV=0V, 0100b=4*10mV=40mV, 1000b=8*10mV=80mV; 1100b=12*10mV=120mV.

In Demo 5 (selected with VID-DIP switches set to 0,1,1,1) the VID value is added to the 1.2V supply voltage. In Demo 5a (selected with VID-DIP switches set to 0,1,1,0) the VID value is subtracted from the 1.2V supply voltage.

Note: the trimming of the 1.2V supply is limited to +/- 180mV of 1.2V based on Trim DAC range and interface circuits. Not all positive VID values inside this range can be realized. The guaranteed absolute VID range is +/- 110mV, or a setting of 1011b.

When the mode switch (SW3) is pressed the value is computed from the DIP switch settings and stored as a target value for the closed loop trim function. The closed loop trim function will control the target voltage and will keep it stable even with load variations. The voltage measured is shown on the LCD display.

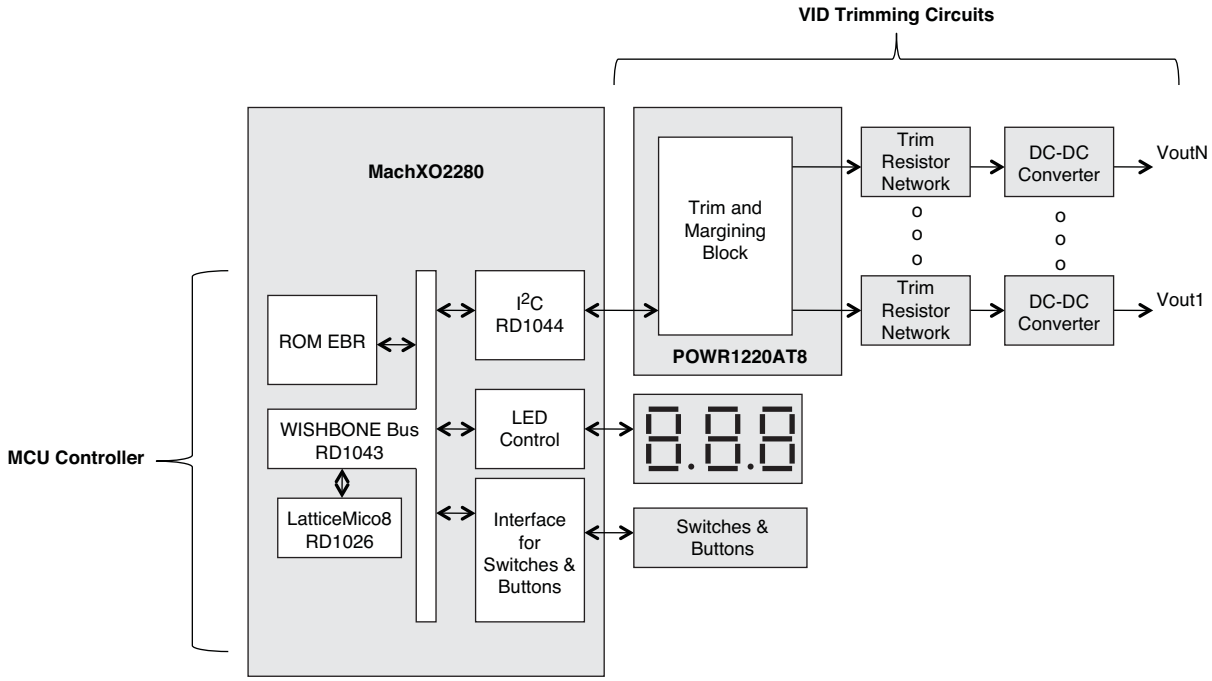
In Demo 5b (selected with the VID-DIP switches set to 0,1,0,x) and pressing switch SW3 the 1.2V supply is turned off. To turn the 1.2V supply back, on set the Demo DIP switch (SW5) position 5 to a logic 1 and press the mode switch (SW3).

Note: Without an external load, the output of the 1.2V DC-DC supply, when disabled, will float up to 0.7V.

Overview of Key Components, Functions and Algorithm of the VID Trimming Design

Several parts of the Hercules Evaluation Board work together in the demonstration design to implement the VID Trimming subsystem (Hercules demonstration Commands 5, 5a and 5b). The key subsystem functions shown in Figure 30, can be categorized as the VID Trimming Circuits and the MCU Controller. The VID Trimming Circuits implement the fine control over the DC-DC converter voltage outputs by adjusting the DC-DC converter trim inputs. The MCU Controller implements the user interface by reading commands on the switches, decoding and executing the command and displaying results on the LCD display. In the external closed loop trim mode the MCU also implements the feedback loop required to maintain the output voltage of the DC-DC converters at the user-selected value from the VID switch settings. The key elements and operation of the VID Trimming Circuits will be described first, followed by the MCU Controller.

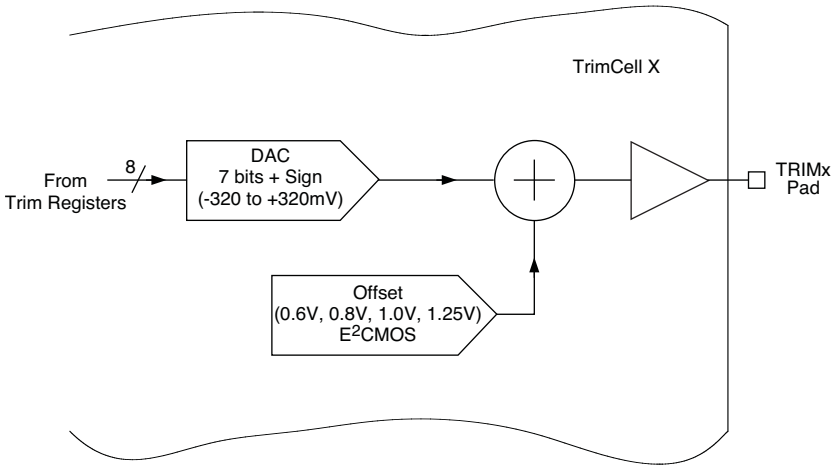
Figure 30. VID Trimming Subsystem Key Elements



VID Trimming Circuits

Much of the general operation of the trimming circuits have previously been described (Trim Resistor Network and POWR1220AT8 Trim and Margining Block) and need not be duplicated here, but the fine control offered by the POWR1220AT8 device when using the I^2C interface can be better understood with a closer look at the detailed architecture of the Trim DAC output and ADC block. Figure 31 shows the main blocks and interconnections of one of the eight Trim-DAC cells previously discussed (refer to Figure 26).

Figure 31. Trim Output and Trim DAC Architecture

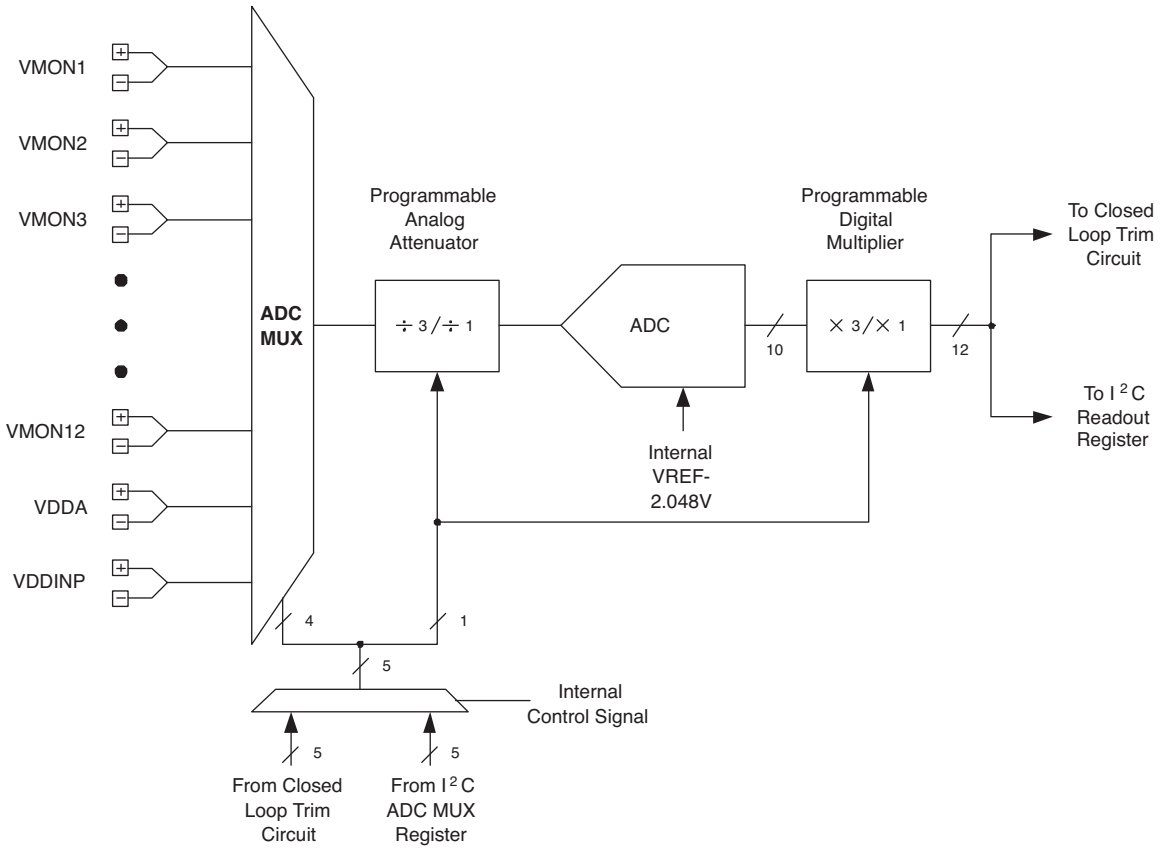


The value stored in the trim register, sourced from the left side of Figure 31, is an 8-bit value and drives the DAC in the middle of the figure. The full-scale output voltage of the DAC is +/-320mV. A code of 80H results in the DAC output set at its bi-polar zero value. The voltage output from the DAC is added to a programmable offset value and the resultant voltage is then applied to the trim output pin. The offset voltage is typically selected to be approximately equal to the DC-DC converter open circuit trim node voltage. This results in maximizing the DC-DC converter output voltage range. The programmed offset value can be set to 0.6V, 0.8V, 1.0V or 1.25V. This value selection is stored in E^2CMOS memory and cannot be changed dynamically.

ADC Block Architecture

Figure 32 shows the architecture of the POWR1220AT8 ADC block. The ADC input comes from one of the VMON inputs as selected by the I^2C ADC mux register (since this implementation uses the external close loop trimming algorithm) and adjusted by an optional 3/1 attenuator. The 12-bit digital value generated by the ADC is available for reading via the I^2C interface. In the external closed loop trimming algorithm used in the Hercules design the LatticeMico8 MCU will be responsible for writing and reading all the appropriate ADC registers to measure the 1.2V supply value so the trimming function can adjust the trim output either up or down as needed.

Figure 32. POWR1220AT8 ADC Block Architecture



MCU Controller

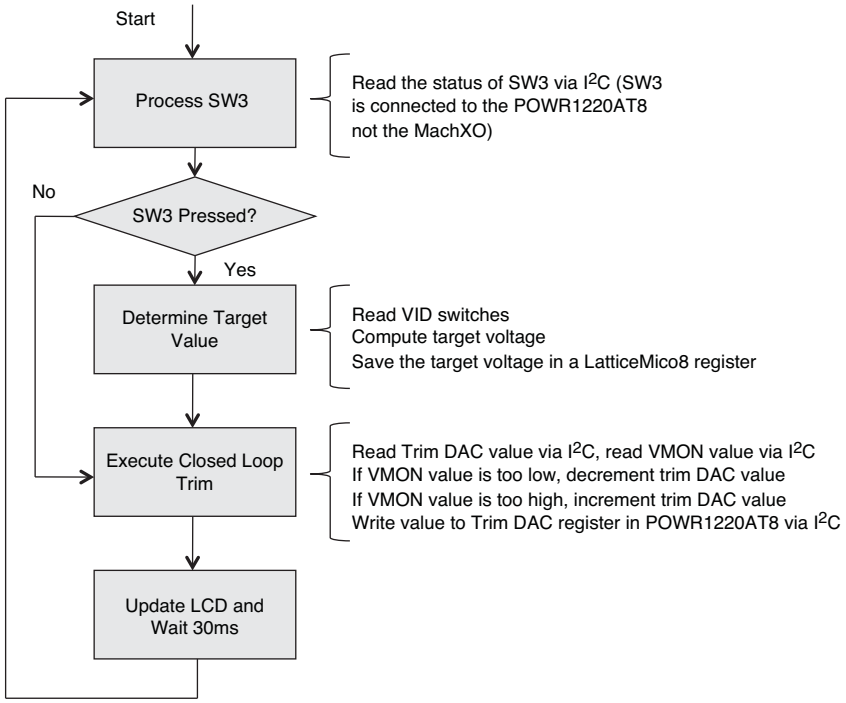
In the VID Trimming subsystem the MachXO device, using the LatticeMico8 microcontroller, either holds the current voltage using external closed loop control or updates the voltage to a new value depending on the state of SW3. To update the voltage the LatticeMico8 reads the values on the VID-DIP switches and then computes the DAC value required to change the 1.2V DC-DC converter voltage output to the target voltage level. To hold the applied voltage the LatticeMico8 implements the external closed loop trimming function, using a similar algorithm as described previously in the Trimming and Margining section. The DC-DC converter output level is read by the LatticeMico8 over the I^2C port and then output to the LCD display.

Implementation of the VID Trimming Function in the Hercules Design

The VID Trimming function uses the same external circuits as those described in the Trimming and Margining section to interface to the 1.2V DC-DC converter trim input. The implementation description of these circuits will not be duplicated here. The main implementation differences between the VID Trimming subsystem and the Trimming and Margining subsystem have to do with the process of updating the trim register based on the VID_DIP switch setting. A flow chart of the process implemented by the LatticeMico8 for the VID Trimming function is show in Figure 33. The process starts with a check on SW3 to see if the voltage needs to be updated. The SW3 setting is read by the LatticeMico8 over the I^2C interface since SW3 is connected to the POWR1220AT8 not the MachXO. If SW3 is high the LatticeMico8 then determines the target voltage by reading the selected value from the VID-DIP switches. The voltage is computed (since the digital value on the DIP switched needs to be multiplied by 10mV to get the target voltage) and saved in an internal register. If SW3 is not pressed, control jumps to the closed loop trim section. The closed loop trim section reads the Trim DAC and VMON values in the POWR1220AT8 via I^2C. If the VMON value is too low the Trim DAC value is decremented. If the VMON value is too high the Trim DAC value is incremented. The net gain of the DC-DC converter from trim input to the output is negative. Thus, increasing the trim bias results in a decrease in output voltage. The Trim DAC value is then written to the POWR1220AT8 via I^2C.

The VMON voltage is updated to the LCD display and a 30ms delay is added prior to returning to the start of the program.

Figure 33. VID Trimming Function Flow Chart



The code used to implement the above process is given below in Listing 16. It is very similar to the above flow chart with a few extra housekeeping steps and the detailed implementation of the target voltage conversion and comparison. A few of the key points will be covered here. The rest of the details can be easily understood by reading the code comments.

In the code block starting on line 8 the DIP switch is re-read to see if bit 6 is set. If it is set the supply is turned on. If it is not set the supply is turned off and the program jumps to the end of the routine at demo_VID_update_LCD at Line 115. The routine demo_VID_turn_on at line 23 enables the 1.2V supply by setting the POWR1220AT8 IN5 pin high. It sets the starting target value for the VMON ADC at 1.2V using 2mV/bit (the nominal 1.2V setting). The value for the 1.2V Trim DAC (Trim-DAC1) is also set at 1.2V. Note the value is adjusted for the slight bias in the resistor network and the DC-DC converter.

The target voltage is now computed from the VID switch settings. The 4-bit VID value is extracted from the DIP switch read value by masking off the upper bits. The ADC value is positive if DIP switch 4 is a high and negative if it is a low. The VID value is thus either added or subtracted five times to get the right value (10mV per VID bit / 2mV per ADC bit = 5 * VID). The r0 register is initialized as a loop counter to execute either an add or subtract five times. Control branches to either the positive 5x routine, demo_VID_pos_5x at Line 48 or the negative 5x routine, demo_VID_neg_5x at Line 61.

In demo_VID_pos_5x the VID amount stored in r4 is added to the accumulator in r8 and r9 (with carry so r9 will contain the upper bits). The loop counter r0 is decremented and checked for zero. After the last pass through the loop the r4 value is adjusted (rotated left twice) and then subtracted from r7 to create the Trim DAC value. The demo_VID_neg_5x routine is the same as the positive version except the opposite arithmetic operations are used. Control passes to the demo_VID_ADC_shift routine at Line 75. This routine formats the data to be sent to the POWR1220AT8 by shifting the value four bits to the left and appending four trailing bits (111 and Done).

Listing 16. VID Trimming Code for LatticeMico8 MCU

```
1   ; VID Demo Mode
2 main_demo_VID:
3   ; If SW3 is pressed, update the VID setting
4   call    Sub_i2c_Read_SW3
5   andi    r5,  0x01
6   bnz     demo_VID_trim
7
8   ; SW3 is pressed, Re-read VID Dip Sw Setting and mask only VID code
9   movi    r15, LED_SW
10   import  r3,  REG_DIP_SW
11   andi    r3,  0x3f
12
13   ; DIP Switch 0x20 = Turn Supply On
14   mov     r4,  r3
15   andi    r4,  0x20
16   bnz     demo_VID_turn_on
17
18   ; Disable 1.2V supply by setting Power Manager IN5 pin Low
19   movi    r5,  0x10                  ; Power Manager IN5 pin
20   call    Sub_LED_clear
21   b       demo_VID_update_LCD
22
23 demo_VID_turn_on:
24   ; Enable 1.2V supply by setting Power Manager IN5 pin High
25   movi    r5,  0x10                  ; Power Manager IN5 pin
26   call    Sub_LED_set
27
28   ; start with nominal 1.2V in ADC value (2mv/bit)
29   movi    r8,  0x58 ; 1.20 V low byte
30   movi    r9,  0x02 ; 1.20 V high nibble
31
32   ; start with nominal 1.2V in Trim-DAC1
33   ; 0x80 is bi-polar zero but, the resistors and DC-DC have a slight bias
34   ; so use 0x60 for 1.2V nominal ouptut
35   movi    r7,  0x60
36
37   ; convert lower 4 bits to ADC value, mask off polarity and enable bits
38   ; 10mV per VID bit / 2mV per ADC bit = 5 * VID
39   ; so add or subract value 5 times
40   mov     r4,  r3
41   andi    r4,  0x0f
42   movi    r0,  0x05 ; loop count
43
44   ; 0x10 = Positive Polarity
45   andi    r3,  0x10
46   bz      demo_VID_neg_5x
47
48 demo_VID_pos_5x:
49   add     r8,  r4
50   addic   r9,  0x00
51   subi    r0,  0x01
52   bnz     demo_VID_pos_5x
```

```
53
54   ; preset the Trim-DAC close to final value
55   rol     r4,  r4
56   rol     r4,  r4
57   sub     r7,  r4
58
59   b       demo_VID_ADC_shift
60
61 demo_VID_neg_5x:
62   sub     r8,  r4
63   subic   r9,  0x00
64   subi    r0,  0x01
65   bnz     demo_VID_neg_5x
66
67   ; preset the Trim-DAC close to final value
68   rol     r4,  r4
69   rol     r4,  r4
70   add     r7,  r4
71
72  ; store the target voltage in ADC format
73  ; r11 = low nibble + 0x0f (111 and Done Bit)
74  ; r12 = high byte
75 demo_VID_ADC_shift:
76   movi    r0,  0x04        ; set loop counter for ADC shift
77
78demo_VID_ADC_shift_loop:
79   setc
80   rolc    r8,  r8
81   rolc    r9,  r9
82   subi    r0,  0x01
83   bnz     demo_VID_ADC_shift_loop
84
85   mov     r11, r8 ; VID low nibble + 0x0f (111 and Done Bit)
86   mov     r12, r9 ; VID high byte
87
88   ; preset Trim-DAC close to final value
89   movi    r3,  0x00 ; Trim DAC1
90   mov     r4,  r7
91   Call    Sub_i2c_Write_DAC
92
93 demo_VID_trim:
94   ; If 1.2V supply is disabled skip the update
95   Movi    r15, LED_SW
96   import  r5,  REG_LED
97   andi    r5,  0x10        ; VID Supply Enable switch
98   bnz     demo_VID_trim_2_target
99
100   ; Can't be CLT if supply is Off
101   Movi   r18, 0x00
102   b      demo_VID_update_LCD
103
104 demo_VID_trim_2_target:
105   ; set the Voltage Profile Pins to 0:0 I2C External Closed Loop Trim
106   Movi    r5,  0x00
```

```
107   call    Sub_1220_Margin
108
109   ; load r8 & r9 with ADC target value
110   ; shift up by 4-bits to match I2C format
111   Mov     r8,  r11 ; VID low nibble + 0x0f (111 and Done Bit)
112   mov     r9,  r12 ; VID high byte
113   call    Sub_1220_Trim
114
115 demo_VID_update_LCD:
116   ; update the Power OR-ing LEDs
117   call    Sub_1220_Power_OR
118
119   ; update the LDC display with selected VMON1
120   Movi    r3,  0x01
121   Call    Sub_1220_Display_VMON
122
123  ; end of VID: repeat main loop
124   b       main_delay
```

The demo_VID_trim function at Line 93 implements the closed loop trim portion of the algorithm. It checks to see if the supply is disabled and skips the trim function if it is. If the supply is enabled the demo_VID_trim_2_target routine at Line 104 is executed. The voltage profile pins are set to 0:0 for the external closed loop setting with a call to Sub_1220_Margin. The properly formatted for I^2C VID values in r12 and r11 and transferred to r8 and r9 and a call to Sub_1220_Trim executes the closed loop trim operation. The details of this subroutine are described in the Trimming Operational Description section and are not repeated here. After the trimming function is completed the (previously described in the Trimming Operational Description section) calls to the Sub_1220_Power_OR and Sub_1220_Display_VMON subroutines are made prior to the return to the main program loop.

Implementation of the 1.2V Supply Turn ON/Off Function in the Hercules Design

This demonstration utilizes the MachXO to turn off the 1.2V supply. When it detects the turn-off command, by decoding the Demo DIP switch settings, it sets XO_PWR0 (the IN5_X_PWR0 signal inside the POWR1220AT8) to a logic low to disable the 1.2V supply. The XO_PWR0 signal is an input to the POWR1220AT8 and is used in the Supervisory Logic Equations section of the Sequence Controller block. An extract of the Supervisory Logic Equations is shown in Listing 17.

Listing 17. Hercules Demonstration Design Supervisory Logic Equation- OUT6_Enable_1_2V.D Signal

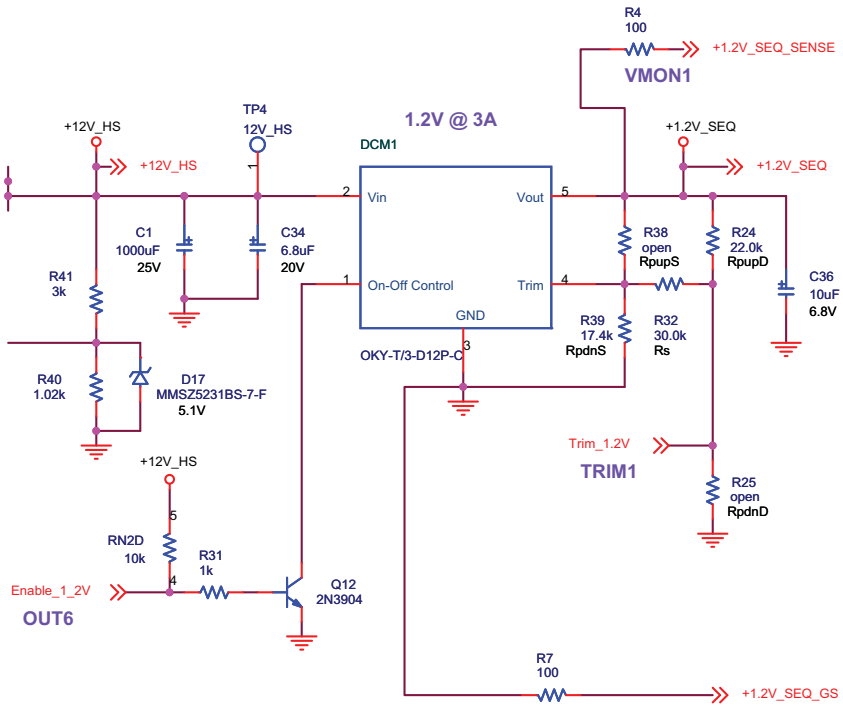
```
//Enable 1.2V Supply and Loads when Power is Good and XO has enabled

EQ17OUT6_Enable_1_2V.D = NOT (IN5_XO_PWR0 AND NOT OUT15_PWR_GOOD)
```

The OUT6_Enable_1_2V.D output signal from the POWR1220AT8 is used to turn on or off the 1.2V supply. This signal is a D-type registered (as shown by the .D suffix) open drain output and it is set low (enabled) when the OUT15_PWR_GOOD signal (created by the Sequencer when board power is working) is a logic 0 (it is active low) and the IN5_XO_PWR0 signal is a logic 1 (active high). The 1.2V supply can be disabled if board power is not good (OUT_PWR_GOOD is a logic 1) or if the MachXO desires the 1.2V supply to be shut down (IN5_XO_PRW0 is a logic 0).

Figure 34 is an extract from the Hercules Schematics from Appendix A. It shows the circuit (isolated in the black box) used to control the On-Off state of the DC-DC converter. This DC-DC converter is disabled when the On-Off Control input is Low (or grounded) or enabled when the On-Off Control Input is High (or Floating). The OUT6 signal is used to control the 1.2V enable feature. The base of the NPN transistor (Q12) is pulled up to 12V with two resistors. The POWR1220AT8 digital outputs are open drain outputs which require an external pull-up resistor so this arrangement is compatible with the output. However the 12V level would exceed the POWR1220AT8 operating specifications if the output was actually pulled up to this level. In this design the transistor (Q12) effectively creates

a diode drop to ground so the actual voltage level on the POWR1220AT8 output will not be much over 0.7V. In order to insure that this circuit works correctly the output of the POWR1220AT8 device is configured to use a reset state of high and is driven low when the power supply is scheduled to be turned on.

Figure 34. OUT6 Control of 1.2V DC-DC Converter On-Off Input Signal



Operational Description – Shut Down

The shut down sequence controls the turn-off of the power supplies on the board when a non-recoverable fault occurs. The POWR1220AT8 enters the shutdown state either from the main Sequence Logic or via an Exception condition. The Hercules sequencer instructions for the shutdown routine are shown in Listing 18. Step 12 is the start of the shutdown sequence and provides a branch location from other parts of the sequencer logic. Step 13 begins the shutdown process by initiating a fault event so the shutdown even will be logged in the SPI memory. Step 14 sets both OUT11 and OUT15 high.

Listing 18. Hercules Demonstration Design Supervisory Logic Equation – Shutdown Sequence

```
Step 12  Begin Shutdown Sequence

Step 13  DUMP_STATUSn = 0

Step 14  OUT11_Shut_12V_Down = 1, OUT15_PWR_GOOD = 1
```

The OUT11_Shut_12V_Down signal controls the charge pump circuit (explained in detail in the Hot Swap Operational Description section. A high on this signal turns off the charge pump circuit and thus disables the Q1 MOSFET that allows the external 12V supply into the Hercules circuitry (refer to Figure 7). The OUT15_PWR_GOOD signal is used by the sequence to enable the 1.2V DC-DC converter. If the signal is high, the converter is turned off (see Listing 5 for the equation using OUT15_PWR_GOOD to turn off the 1.2V supply). OUT15_PWR_GOOD is also used to turn on the LED that indicates power on the board is operating correctly. Setting the signal high will turn off the LED.

The other way to enter the shutdown sequence is with an exception condition. An exception condition is a special logic equation that transfers control to a location within the sequencer on an interrupt basis. If the exception is true control is transferred unless the instruction is market as not interruptible. Listing 19 shows the exception condition

in the Hercules design. It jumps to Step 12 if it is true. The shut-down sequence is entered when both 12V supplies are off (Primary and External) or an over-current occurs (Primary current is high or External current is high).

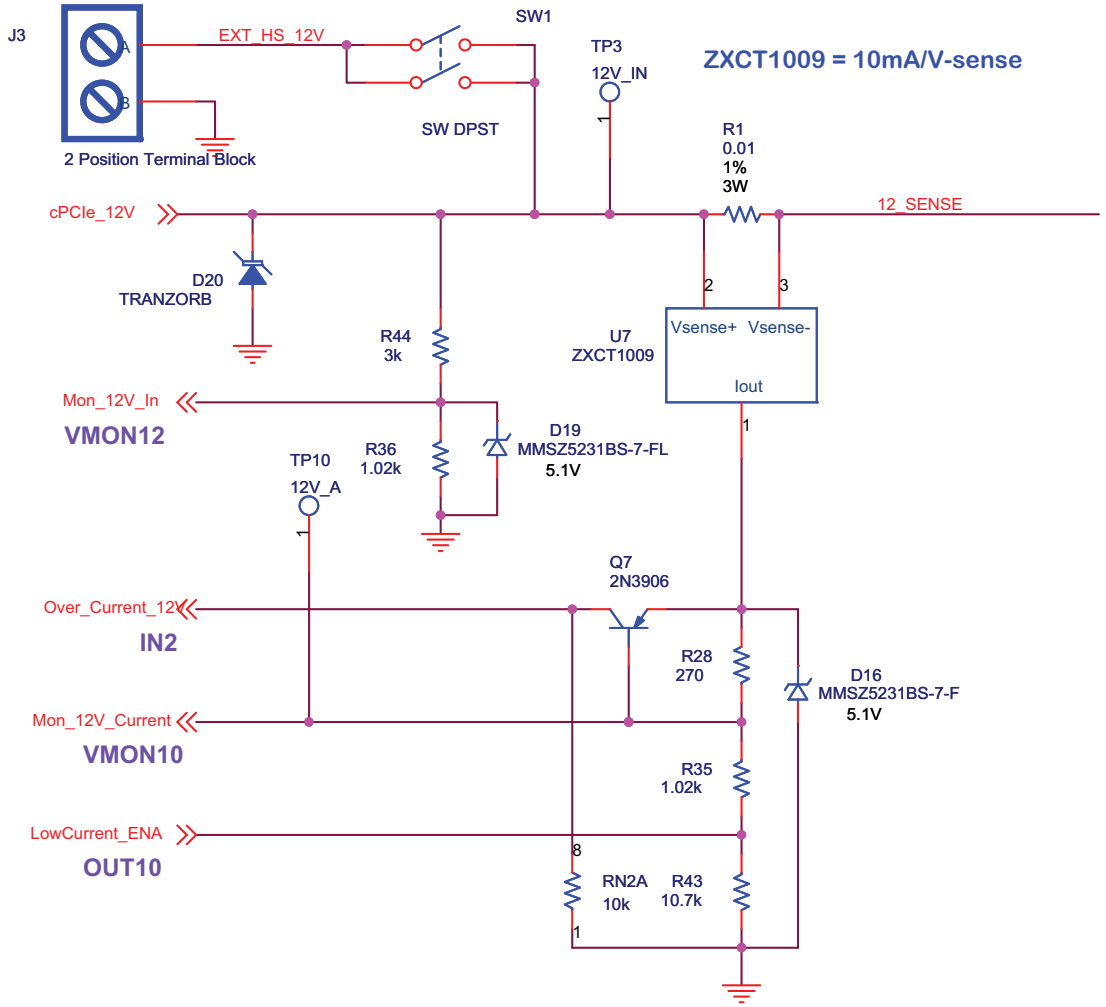
Listing 19. Hercules Demonstration Design Exception Condition Equation- Shutdown Sequence

```
E 0      If  ( NOT INPUT_12V_OK AND NOT V7_EXT_IN_12V_OK )    OR
         V10_PRI_CUR_HIGH    OR
         V8_EXT_CUR_HIGH
```

INPUT_12V_OK uses the VMON12 input and is connected to the Mon_12V_In signal on the schematic. This signal was shown previously in Figure 1 and the resistor divider used to generate the VMON12 voltage is described in detail in that section. V7_EXT_IN_12V_OK uses the VMON7 input and is connected to the Mon_Ext_12V_In signal on the schematic. This signal uses a similar resistor divider as on VMON12.

V10_PRI_CUR_HIGH is connected to the VMON10 input. The current sense circuit is shown in Figure 4 and its use is explained in detail in that section. V8_EXT_CUR_HIGH is connected to the VMON8 input. It uses the circuit shown in Figure 35. The resistor divider is used to allow two different ranges to be measured, a high current range when OUT9 is low (since only the 1K Ohm resistor is used to develop a voltage) and a low current range when OUT9 is high/floating (since both resistors are connected providing 11K Ohms to develop a voltage).

Figure 35. VMON8 Current Sense Input Signal



Operational Description – Common Elements

There are several common elements to the Hercules Demonstration Design that are best described in a separate section so the Operational Description is not needlessly repeated over several parts of this document. The common elements described in this section are the power subsystem design which provides power to all the various portions of the Hercules Evaluation Board, the interface and control of the LCD display, and the interface to the buttons, switches and LEDs used by the MachXO to read user commands and provide simple status outputs.

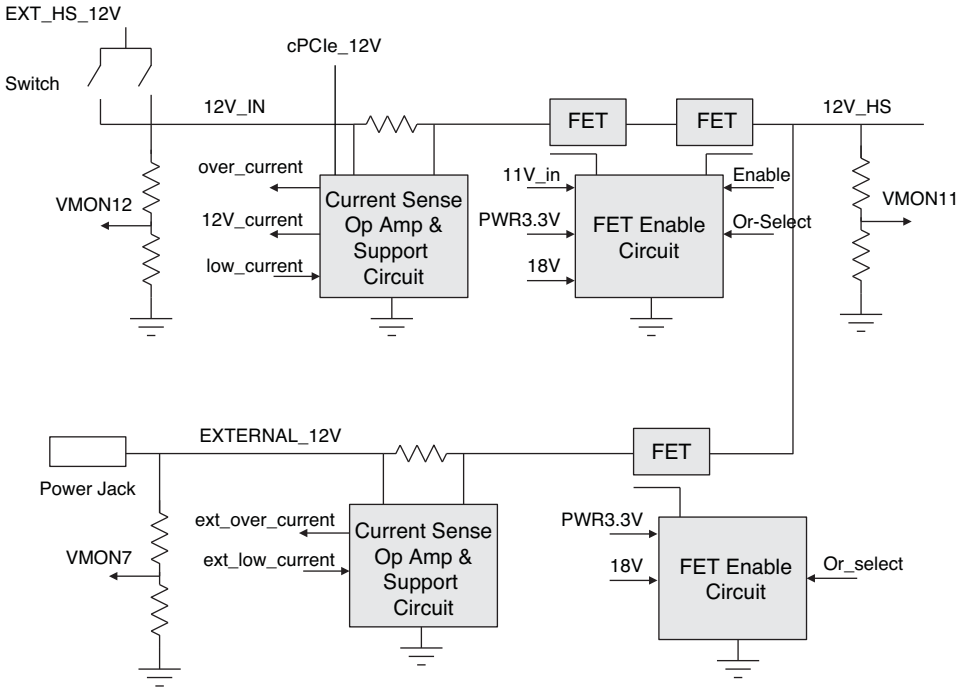
Power Subsystem Architecture

The power subsystem design on the Hercules Evaluation Board contains several useful features that can be copied for many different applications. The main power rails developed by the power subsystem can be categorized as the Main 12V power rail or the Critical Circuits 3.3V power rail. The Main 12V power rail provides power to the bulk of the board and is used to demonstrate Hot Swap, Redundant Supply operation. It drives the 1.2V DC-DC converter that is used in the Trimming, Margining and Fault Logging demonstrations. The Critical Circuits 3.3V power rail is used to supply current to the circuits that are required to operate during the bring-up of the Main 12V power rail. They must also operate prior to the 12V supply being enabled) as well as during a fault on the main 12V supply. Therefore, the Critical Circuits need to run long enough for the fault logging event to be successfully written to the SPI memory even while the 12V supply is failing.

Note: The figures used in this section represent high-level versions of the actual circuits in the Hercules schematics. In some cases the detailed implementation of the circuits within a block have been described previously and will not be described in detail in this section as well. Some simplification has resulted from the high-level conversion. The schematic in Appendix A should be considered the more accurate if any differences are identified.

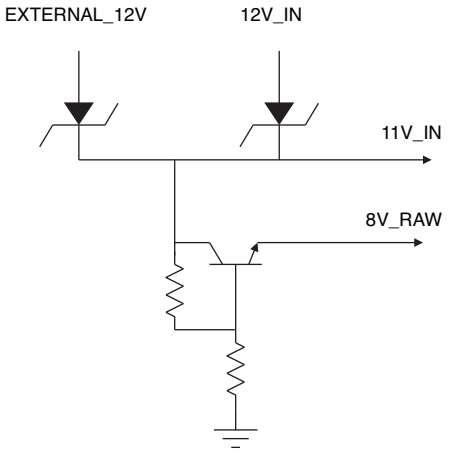
Figure 36 shows the high-level architecture for the Main 12V power rail. There are two main legs of the rail- one (12V_IN) is sourced from the external switch or the cPCI_12V power inputs and the other (EXTERNAL_12V) comes from the 12V wall socket power jack. The 12V_IN leg voltage is monitored by VMON12 and current is sensed by the Over_current and 12V_current signals under control of the Low_current enable signal. The detailed implementation is explained in detail in the Hot Swap Operation Description section. The FET enable circuit for the 12V_IN leg controls both the main power ET and the Power-OR'ing FET from the Enable and OR_Select signals. The implementation of the circuit is explained in detail in the Redundant Power Operational Description section. The main output of this leg, 12V_HS, is used to feed the 1.2V DC-DC converter and its support circuits used in many of the demonstration program functions. Refer to Figure 34 to see the connection from 12V_HS to the DC-DC converter and its support circuits.

Figure 36. High-Level Diagram for the Main 12V Power Rail



shows the first stage of developing the Critical Circuits 3.3V power rail. The two Main 12V power sources, EXTERNAL_12V and 12V_IN, shown in Figure 36 are used to create an 11V_IN voltage source through two diodes. If either (or both) of the 12V power sources is available the 11V_IN voltage will be at around 11V. This voltage is then converted to an 8V_RAW voltage through a transistor. The transistor is biased such that the voltage drop from the collector to emitter is about 3V which results in an 8V_RAW output voltage. <Is this description accurate?>

Figure 37. First Stage of Critical Circuits Power Rail Implementation

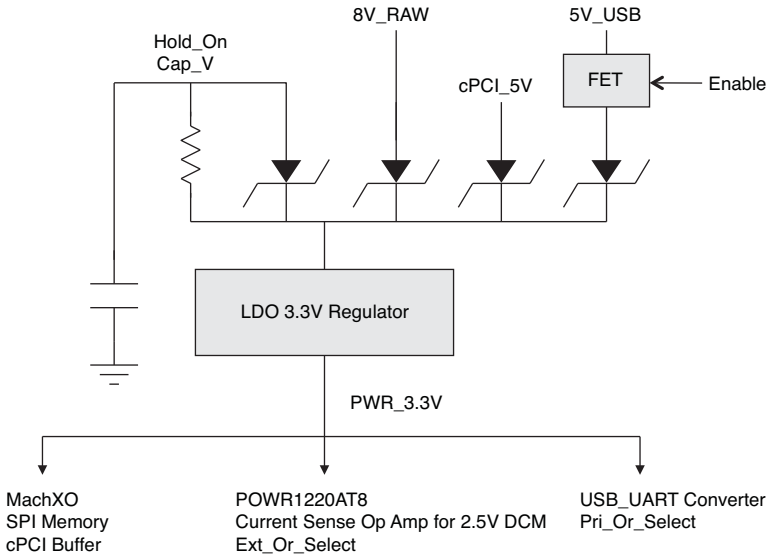


Once the8V_RAW voltage is available it can be used to source circuits which need to be on prior to the main 12V_HS voltage being available. The 8V_RAW voltage is sourced through a diode to a 3.3V LDO voltage regulator as illustrated in Figure 38. The output of this LDO drives all the critical circuits needed to control the 12V_HS bring-up (MachXO, POWR1220AT8, SPI Memory, Current Sense Op amps, and FET control support circuits). The power to the LDO must be 'held on' while the main 12V power is interrupted long enough to store fault data to the SPI memory. This is accomplished by using the 'hold on' capacitor on the left side of Figure 38. This capacitor will be

charged from the 8V_RAW voltage source and stores enough charge so that when the 12V power source is removed, the LDO will continue to supply voltage and current to all the critical devices.

The LDO can also be supplied from other voltage sources: the 5V USB connector input (through a FET), or the cPCI 5V connector. These sources allow the critical circuits to run off of these sources for development purposes, but are not designed to support successful fault logging if they are the only source of power to the board and they are interrupted.

Figure 38. High-Level Diagram for the Critical Circuits 3.3V Power Rail



LCD Display

The LCD display is used by all the demonstration design functions to output status information. The LatticeMico8 typically calls a subroutine that displays the desired value on the LCD. The example shown in Listing 20 is the Sub_1220_Display_VMON subroutine. In this routine the voltage on VMON1 is read over the I^2C bus and then converted to a 3-digit hexadecimal value. The value is exported to the LCD holding registers LCD_REG_0 and LCD_REG_1.

Listing 20. Display VMN Results on LCD Assembly Language Routine

```
;=============================================================================#
; Subroutine: Sub_1220_Display_VMON                                          #
; Input : r16 = Update Delay Counter                                         #
; Output: r16 = Update Delay Counter-1 or VMON_DISPLAY_DELAY_COUNT           #
; Used  : r15 = Used by I2C routines                                         #
;         r3  =                                                              #
;         r6  =                                                              #
;         r7  =                                                              #
; Description: Decrement delay count and display VMON voltage on LCD and     #
; reload delay counter if zero.  The delay prevents LCD "flicker."           #
; Calls the I2C routine to read the VMON value and the Sub_vmon2hex          #
; routine to format the ADC result for the LCD.                              #
;=============================================================================#
Sub_1220_Display_VMON:
   ; decrement the display delay
   ; and update if zero
   subi    r16, 0x01
   bz     update_VMON_display
```

```
19       ret
20
21     update_VMON_display:
22       ; reset the display delay counter
23       movi     r16, VMON_DISPLAY_DELAY_COUNT
24
25       ; adjust DIP-SW to mux value
26       subi    r3,  0x01
27       ; Turn on attenuate bit
28       addi    r3,  0x10
29
30       ; Address the MUX and convert the VMON
31       call    Sub_i2c_Read_VMON
32
33      ; Convert the 10-bit ADC value into 3-hexidecimal nibbles
34       call    Sub_vmon2hex
35
36       ; display voltage on LCD digits
37       movi    r15, LCD
38       export  r6,  REG_LCD_0 ; write Tens & Ones
39       export  r7,  REG_LCD_1 ; write DPs & Hundreds
40
41       ret
```

The LCD display is driven by the MachXO and it requires 24 of the MachXO output signals. To simplify the job of the LatticeMico8 a small logic block is added to the MachXO. This block takes the hexadecimal value from the LCD holding registers (LCD_REG_0 and LCD_REG_1 loaded by the LatticeMico8 over the WISHBONE bus), and converts it to the 7-segment format required by the LCD display. The Verilog code for this block is in the lcd_wb.v file in the MachXO source folder in the project directory. The listing for lcd_wb.v is too long to include in this document, but the beginning of the file is shown in Listing 21. The standard WISHBONE interface is used to connect the module to the LatticeMico8 system and the output is the 24-bit connection to the LCD display. The details of the rest of the routine are left to the user (who can open the routine and follow the detailed comments).

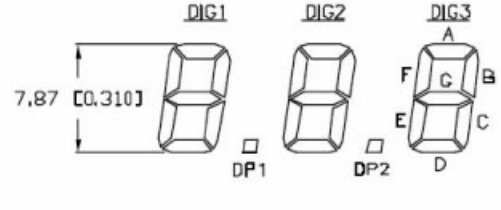
Listing 21. Conversion of Hexadecimal to 7-Segment Display Format

```
//--------------------------------------------------------------------------
// Name:  lcd_wb.v
//
// Description: This module is to control LCD from the Wishbone Bus
//              The low address register holds the Ones and Tens
//              and the high address reg holds the Hundreds and decimal
//              points.
//
//--------------------------------------------------------------------------
// Code Revision History:
//--------------------------------------------------------------------------
// Ver: | Author|Mod. Date |Changes Made:
// V1.0 | CWD |14 May 2010 |Init ver
// V1.1 | CWD       | 7 Jun 2010 |Added leading zero blanking
//
//--------------------------------------------------------------------------

module lcd_wb(
// wishbone interface
inputrst_i,// reset
```

```
inputclk_i,// WB bus clock
input   stb_i,// strobe
inputadr_i,// lsb address bit
input   we_i,// write enable
input   cyc_i,// cycle
output  ack_o,// acknowledge
input   [7:0]dat_i,// 8-bit data in

output reg[7:0]dat_o,// 8-bit data out

// external connections
output[23:0]LCD_pin// Three Digit LCD display with 2 decimal points
);

// User provided parameters
parameter LCD_CLOCK_TAP = 15;// 100Hz should = Bus clock / 2^(LCD_CLOCK_TAP +1)
// 122Hz = 8MHz / 2^16

// internal signals
wire display_clk;// LCD common pin 50% square wave at 100Hz
```

The mapping of the 24-bit LCD display input is shown in Figure 39. Each of the outputs of the module drives a segment on the LCD display except Pin 1, which is a common signal. The common signal is driven from a 120Hz clock generated from the 33MHz WISHBONE clock. The segments are driven 180 degrees out of phase with the common signal to "turn on a segment" and in phase to turn off a segment.

Figure 39. LCD Display Segment Definitions



PIN NO.	1	2	3	4	5	6	7	8
SEGMENT	COM	E1	D1	G1	DP1	E2	D2	C2
PIN NO.	9	10	11	12	13	14	15	16
SEGMENT	DP2	E3	D3	C3	B3	A3	F3	G3
PIN NO.	17	18	19	20	21	22	23	24
SEGMENT	B2	A2	F2	G2	B1	A1	F1	G1

Buttons, Switches and LEDs

The buttons, switches and LEDs are all read or written to using the WISHBONE interface. The WISHBONE interface for these devices is instantiated in the top-level Hercules demonstration Verilog code. The code for the instantiated logic is contained in the led_sw_wb.v file in the project/XO/source directory. The source for led_sw_wb.v is given in Listing 22. The input and output signals are primarily for the WISHBONE interface. The LED outputs and switch inputs are LED[7:0] (registered) and sw[7:0] respectively. An internal register (data[7:0] is used to hold the LED value or the switch value on a read instruction.

Listing 22. Switch and LED Wishbone Interface for LatticeMico8

```
// Wishbone bus interface to VID dip switches and XO_LEDx

`include "timescale.v"
module led_sw_wb(
  input      clk_i,
  input      rst_i,
  input      [7:0] dat_i,
```

```
    input      [7:0] sw,
    input      stb_i,
    input adr_i,
    input      we_i,
    input      cyc_i,
    output     [7:0] dat_o,
    output reg [7:0] led,
    output     ack_o
    );

  // Internal register
    reg[7:0] data;

  always@(posedge clk_i or posedge rst_i) begin
    if (rst_i)
      // LED outputs are active Low
      led <= 8'b00001111;

    // write to either adx is a write to LED reg
    else if (stb_i && cyc_i && we_i)
       led <= dat_i;

    // adx = 0 = LED register
    else if (stb_i && cyc_i && ! we_i && ! adr_i)
   data <= led;

    // adx = 1 = Dip Switch
    else if (stb_i && cyc_i && ! we_i && adr_i)
   data <= sw;
  end

    assign dat_o = data;
    assign ack_o = stb_i;

  endmodule // led_sw_wb
```

The exception to the above design is the handling of SW3. SW3 is connected to the POWR1220AT8 and not the MachXO. The state of SW3 is thus detected by doing an I^2C read of the Input1 on the POWR1220AT8. This is handled by the Sub_i2c_Read_SW3 subroutine in the LatticeMico8. The listing for this subroutine is given in Listing 23. The subroutine writes individual bytes to the POWR1220AT8 to start the transmission, address the status register within the POWR1220AT8 and end the transmission. Next a read of the SW3 input is executed using a similar sequence with a read replacing the previous write. The value of the input bits is returned in r5. These detailed read and write subroutines can be useful for a variety of designs that need access to the POWR1220AT8 resources over I^2C.

Listing 23. Reading SW3 via an I^2C Interface

```
 ;=====================================================================#
; Subroutine: Sub_i2c_Read_SW3                                         #
; Input : None                                                         #
; Output: r5  = POWR1220AT8 Input Status Register read by I2C          #
; Used  : r0  = I2C subroutines                                        #
;         r15 = I2C subroutines                                        #
; Description: Calls I2C routines to read back the value of the Input  #
;    Status Register of the Power Manager.  Used to check the sataus   #
```

```
;    of IN1 (push button SW3).                                          #
;=======================================================================#
Sub_i2c_Read_SW3:
  ; Start and Address 1220AT8
  movi  r5,  POWR_1220AT8_ADX
  call    Sub_i2c_write_start_byte
  ; Input Status Register
  movi    r5,  REG_1220AT8_IN_STAT
  call    Sub_i2c_write_stop_byte

  ; Start and Address 1220AT8 with Read bit
  movi  r5,  POWR_1220AT8_ADX_READ
  call    Sub_i2c_write_start_byte
  ; Read back Input Status and Stop
  call    Sub_i2c_read_byte
  ; return the result in r5
  ret
```

Ordering Information

Description	Ordering Part Number	China RoHS Environment-Friendly Use Period (EFUP)
Power Manager II Hercules Development Kit - Advanced Version	PAC-POWR1220AT8-HA-EVN	
Power Manager II Hercules Development Kit - Standard Version	PAC-POWR1220AT8-HS-EVN	

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
December 2010	01.0	Initial release.

Appendix A. Schematic

Figure 40. Table of Contents

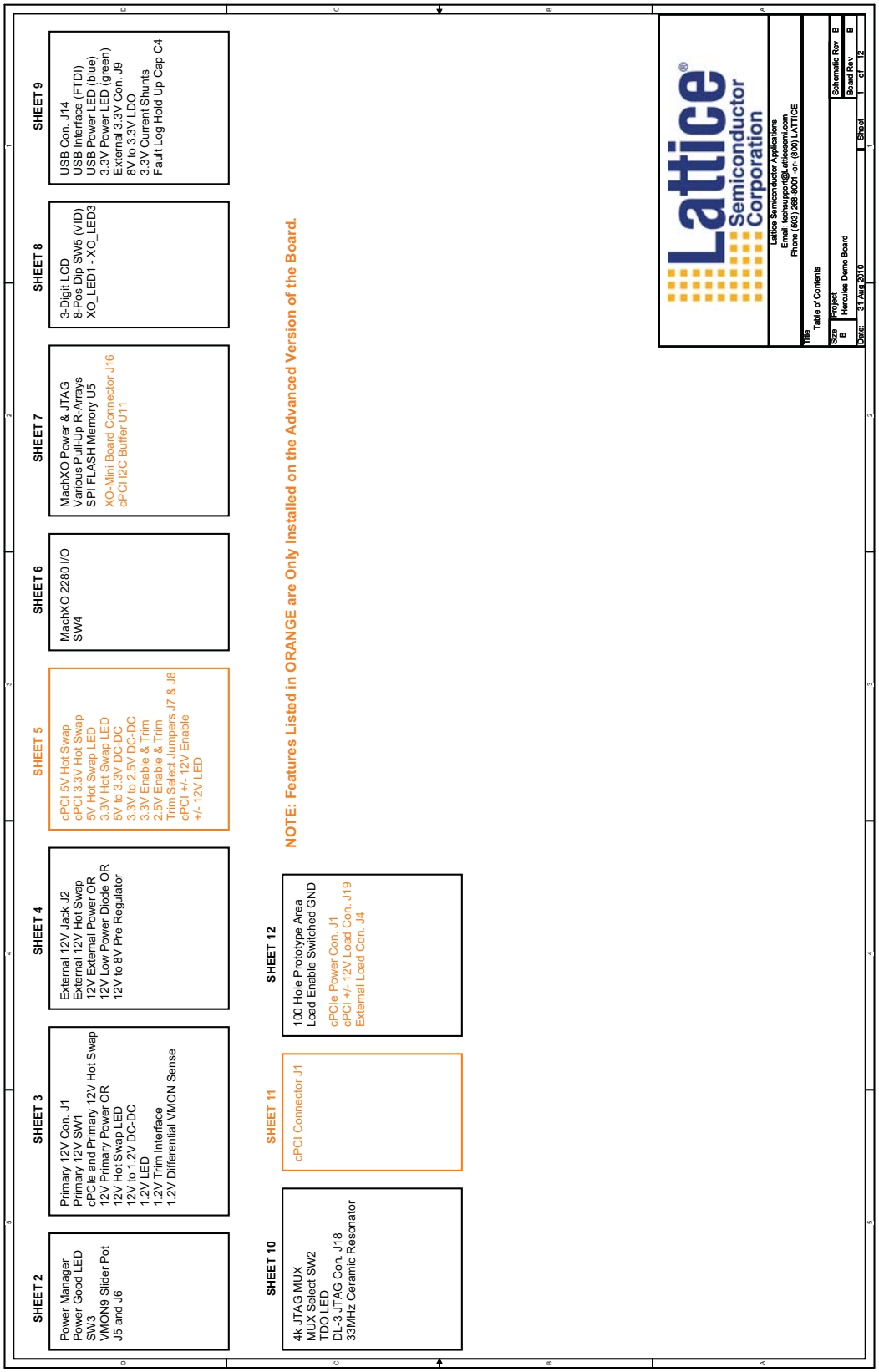


Figure 41. POWR1220AT8

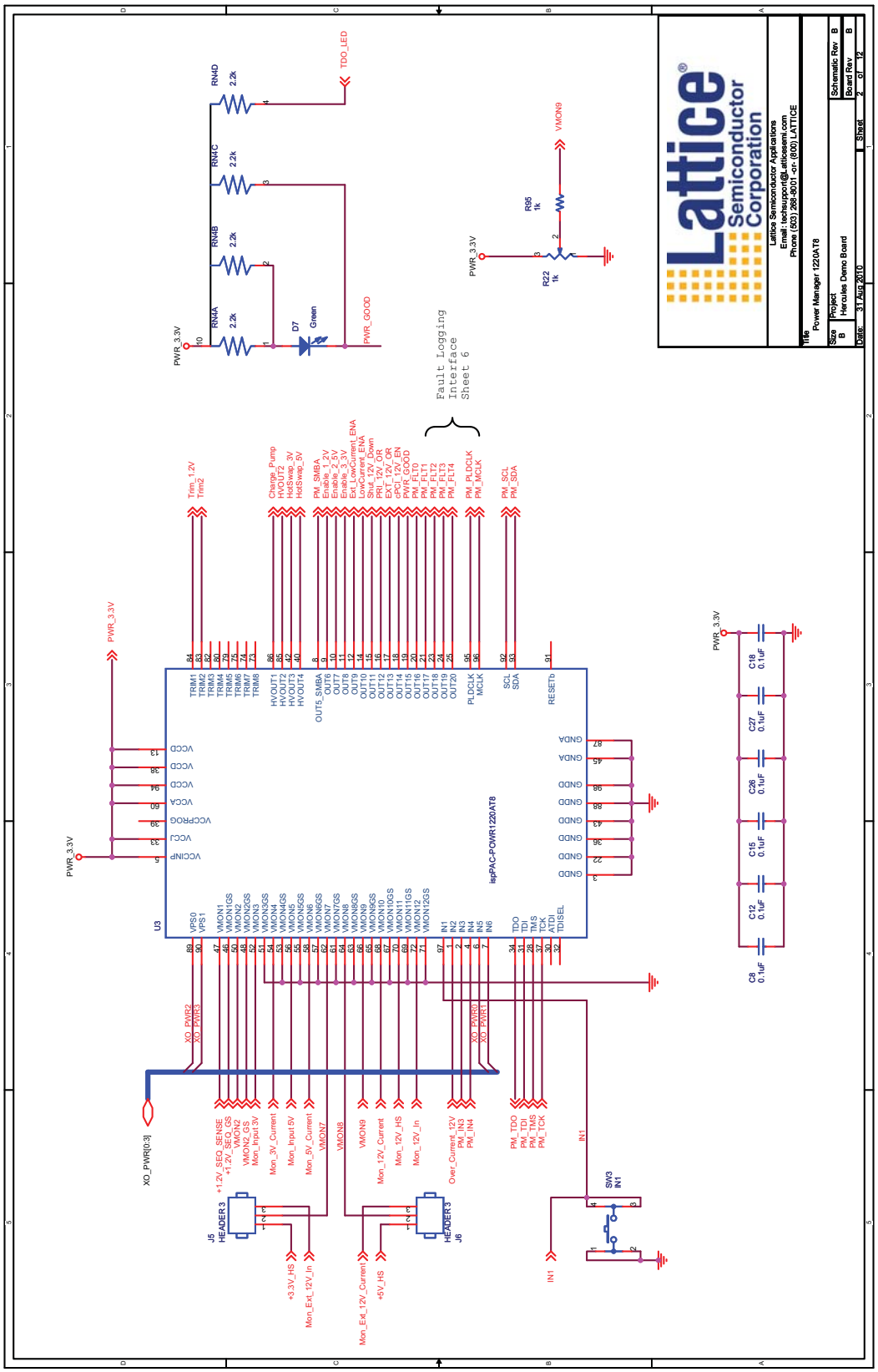


Figure 42. Hot Swap 12V

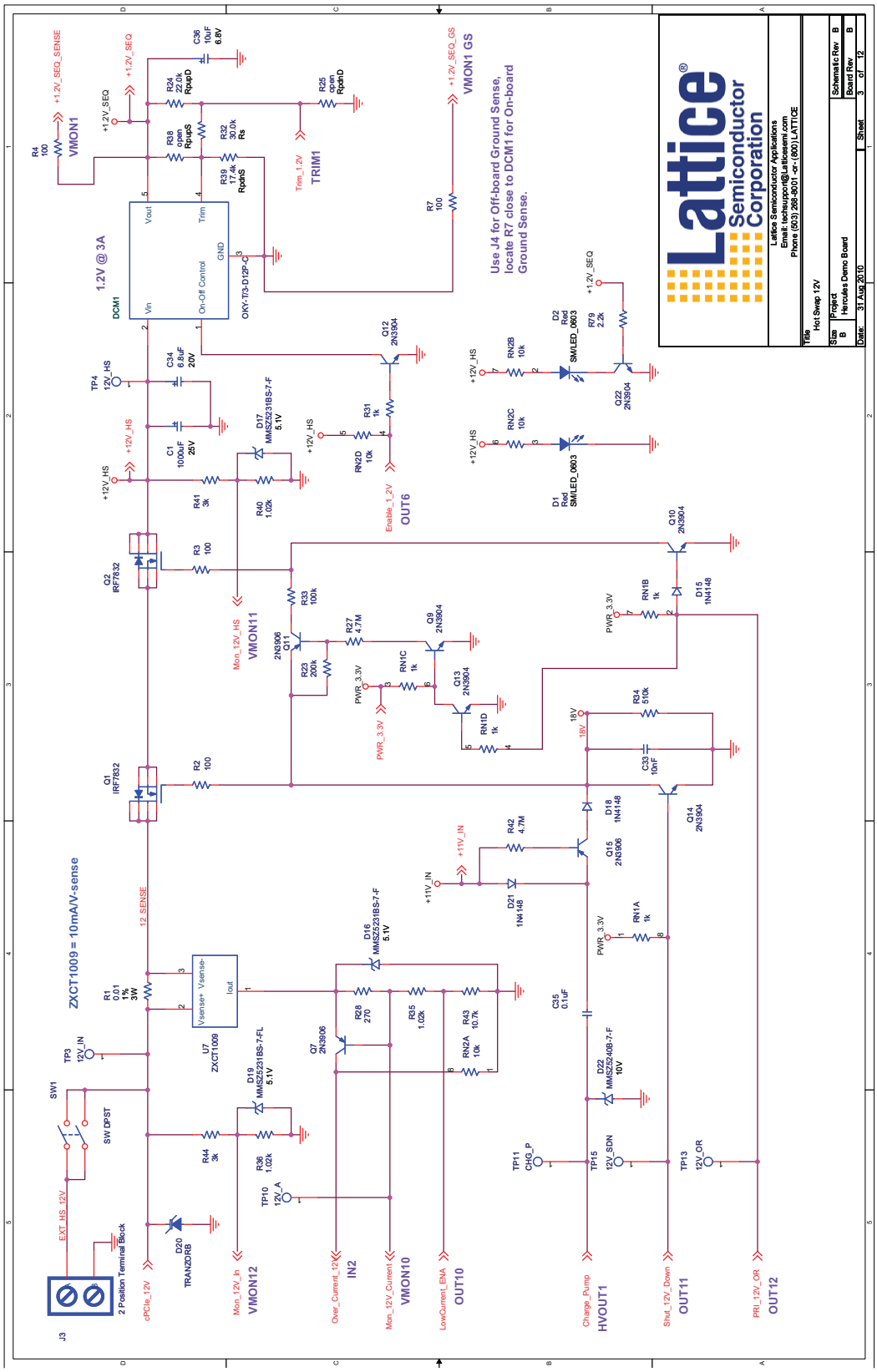


Figure 43. External 12V Power-OR'ing

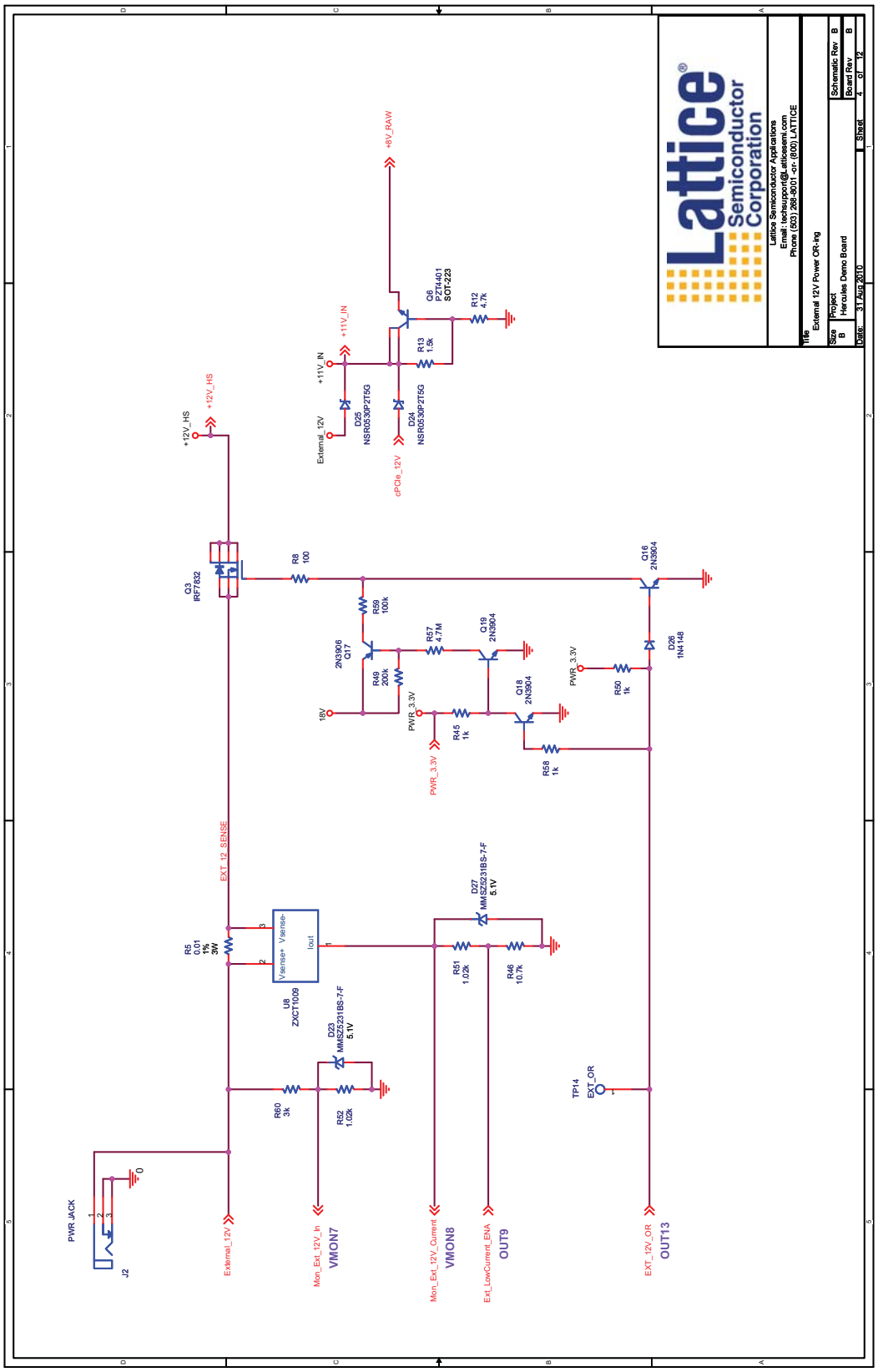


Figure 44. Hot Swap 3.3V and 5V

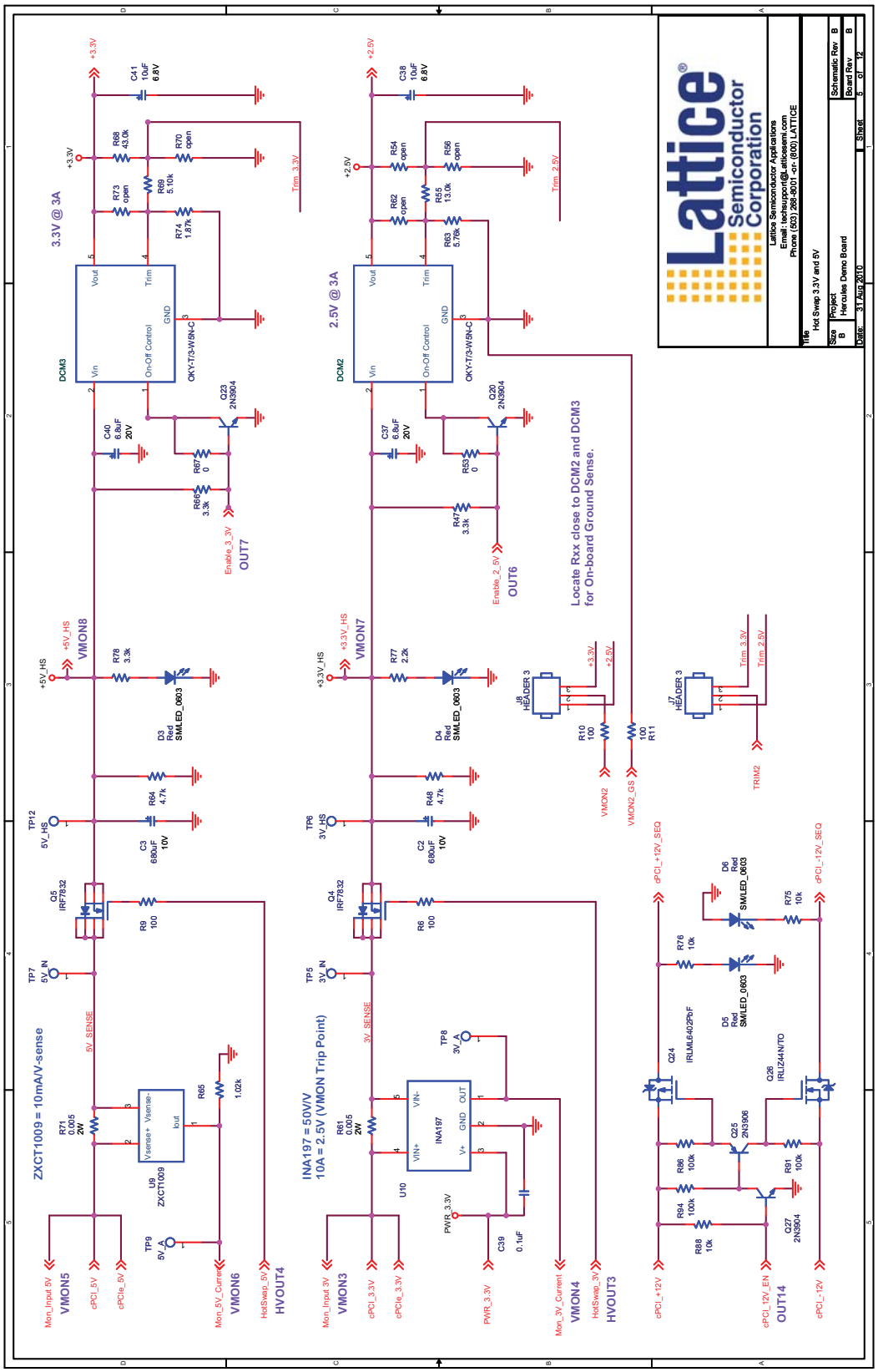


Figure 45. MachXO I/Os

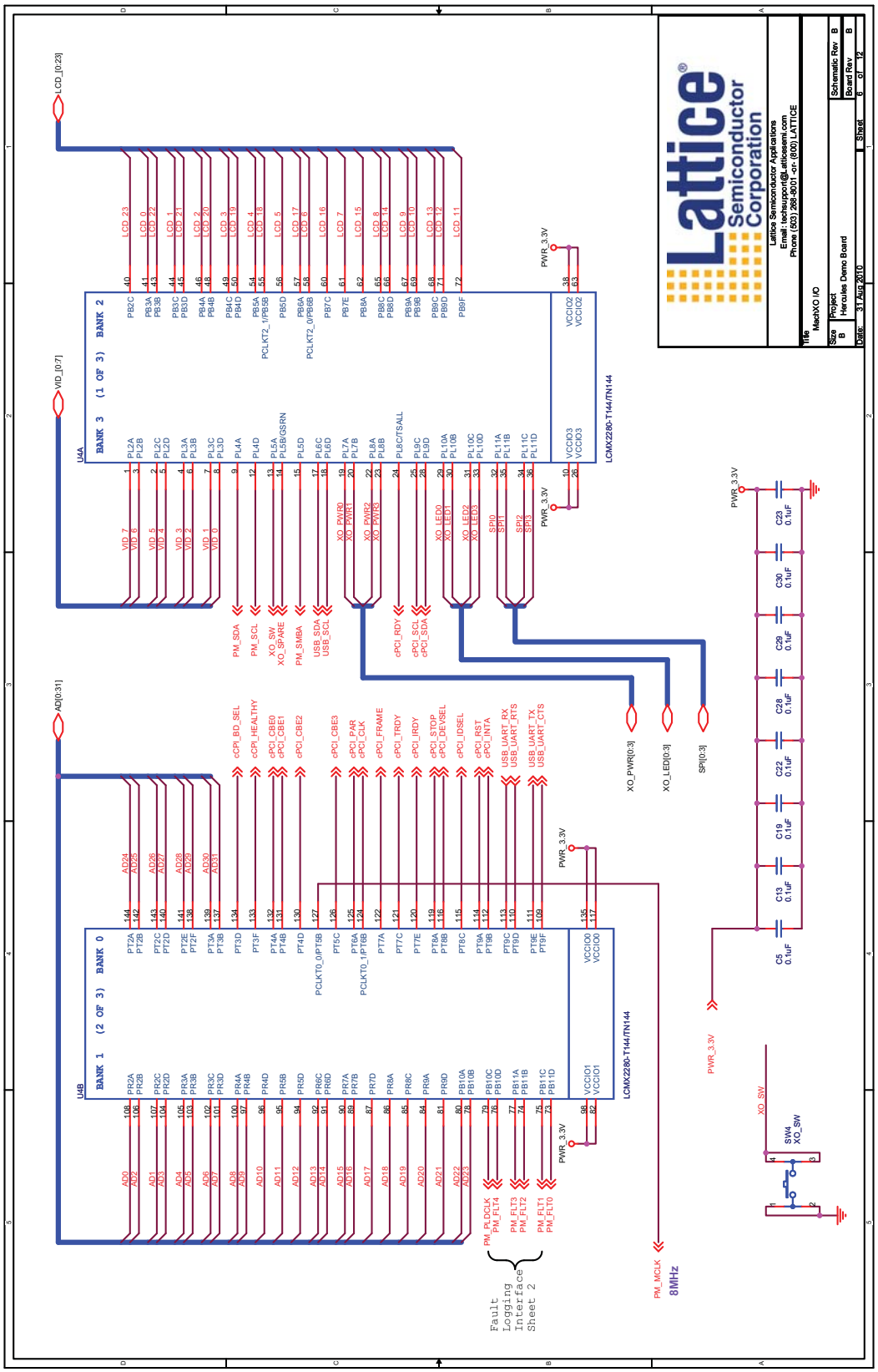


Figure 46. MachXO Bottom JTAG and Power

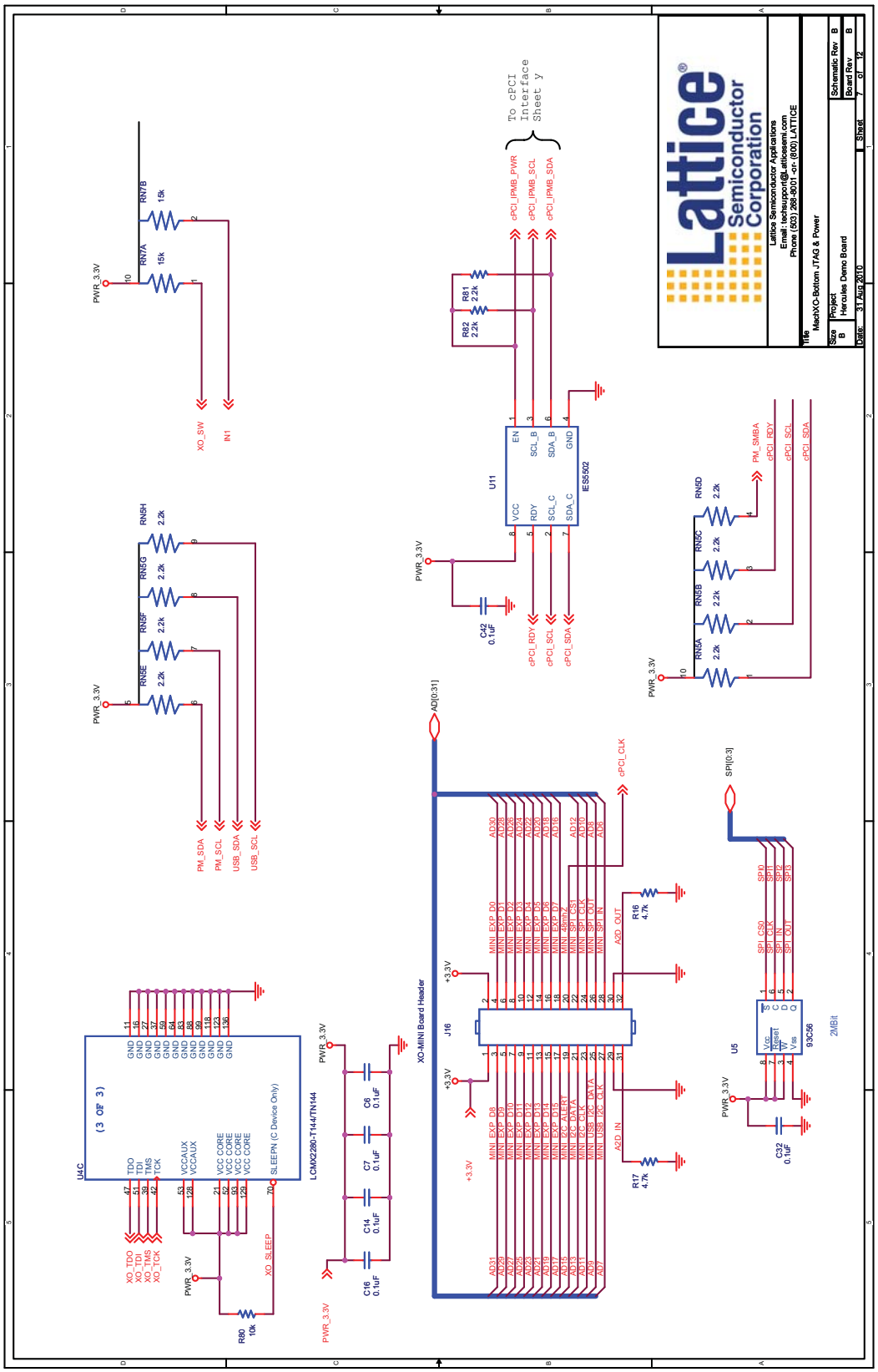


Figure 47. LED and LCD Display and Voltage Identification Digital

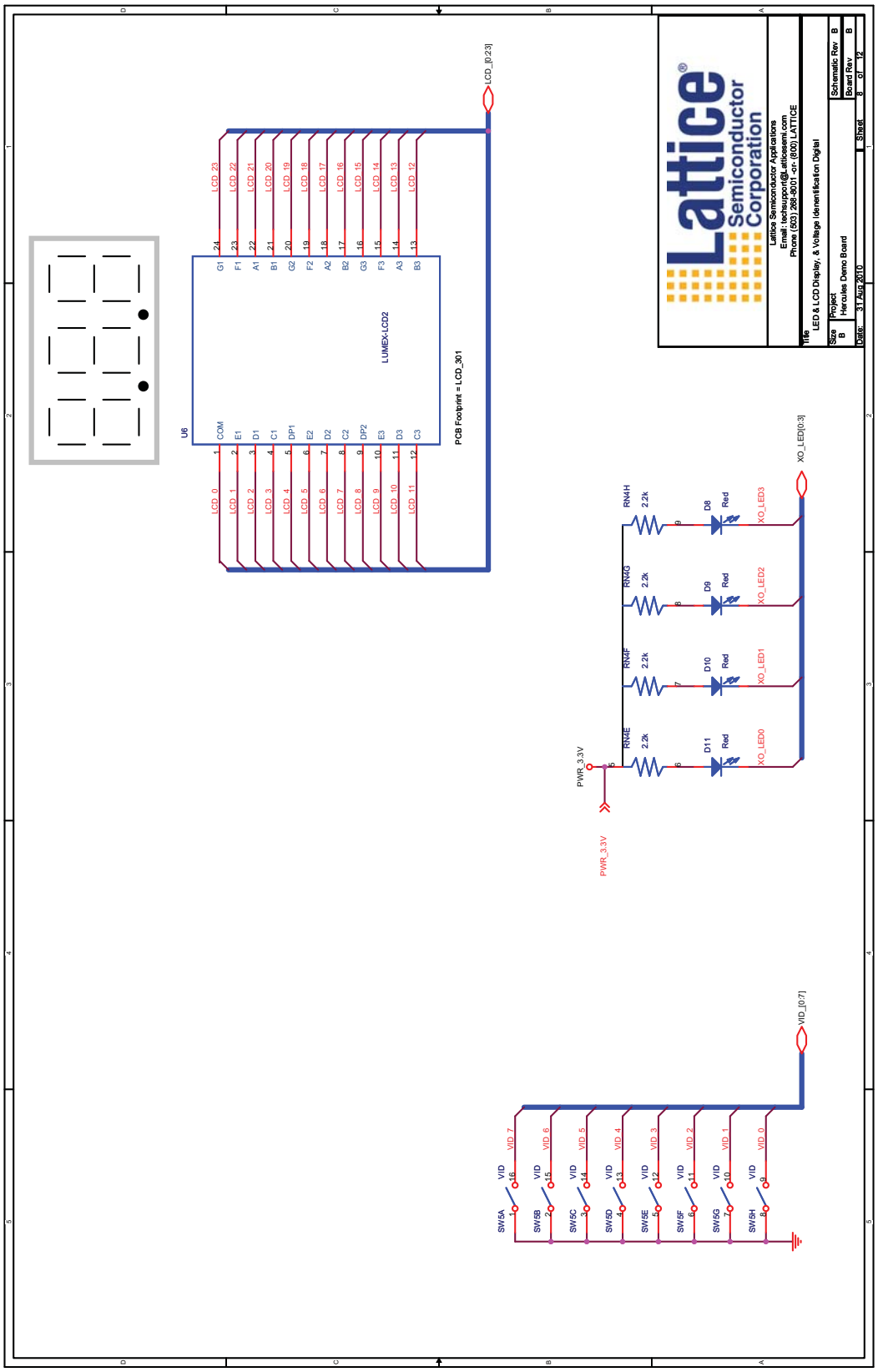


Figure 48. USB to JTAG, I²C and UART for MachXO and POWR1220AT8 and 3.3V Registers

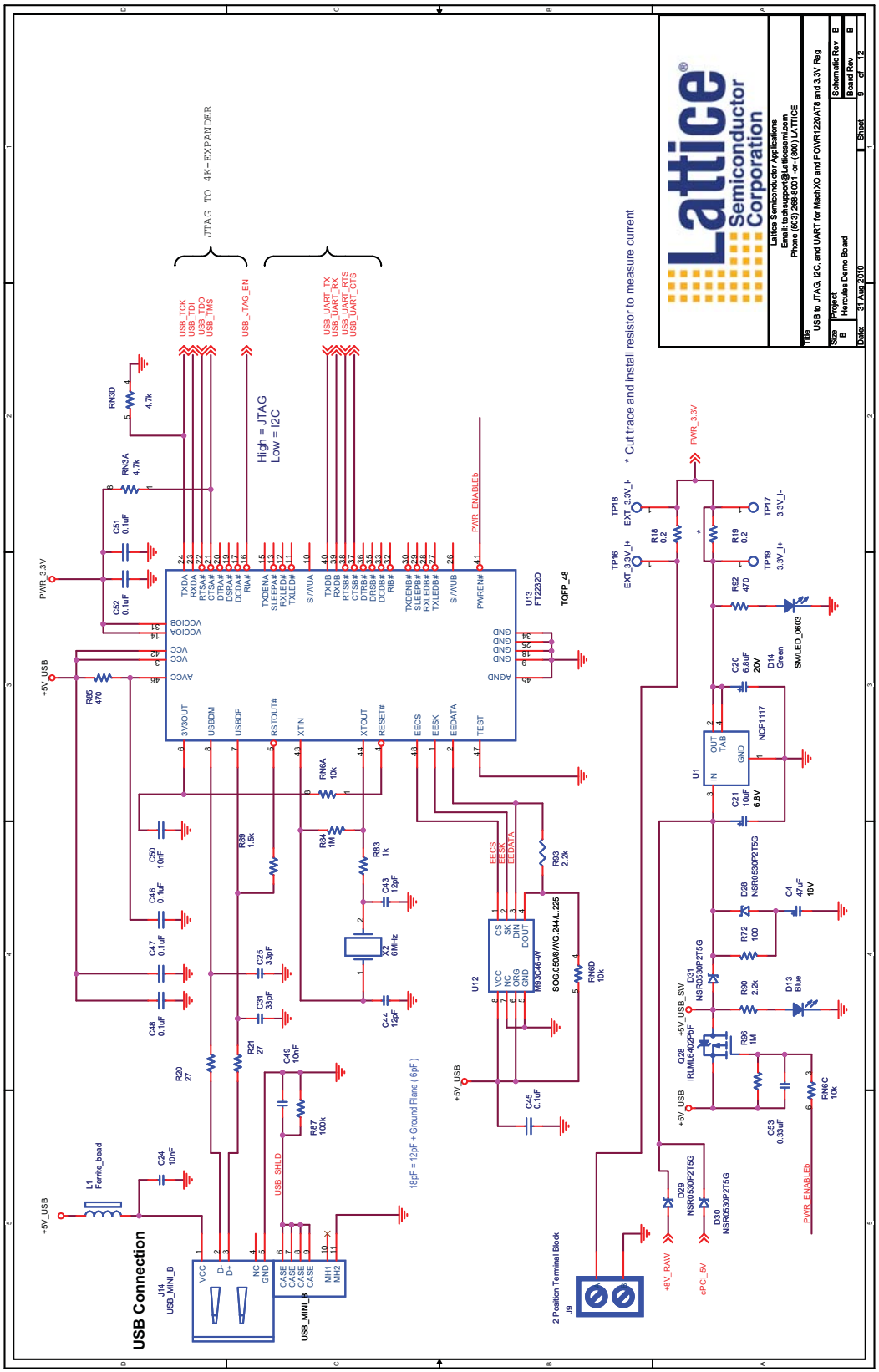


Figure 49. JTAG Chain Configuration

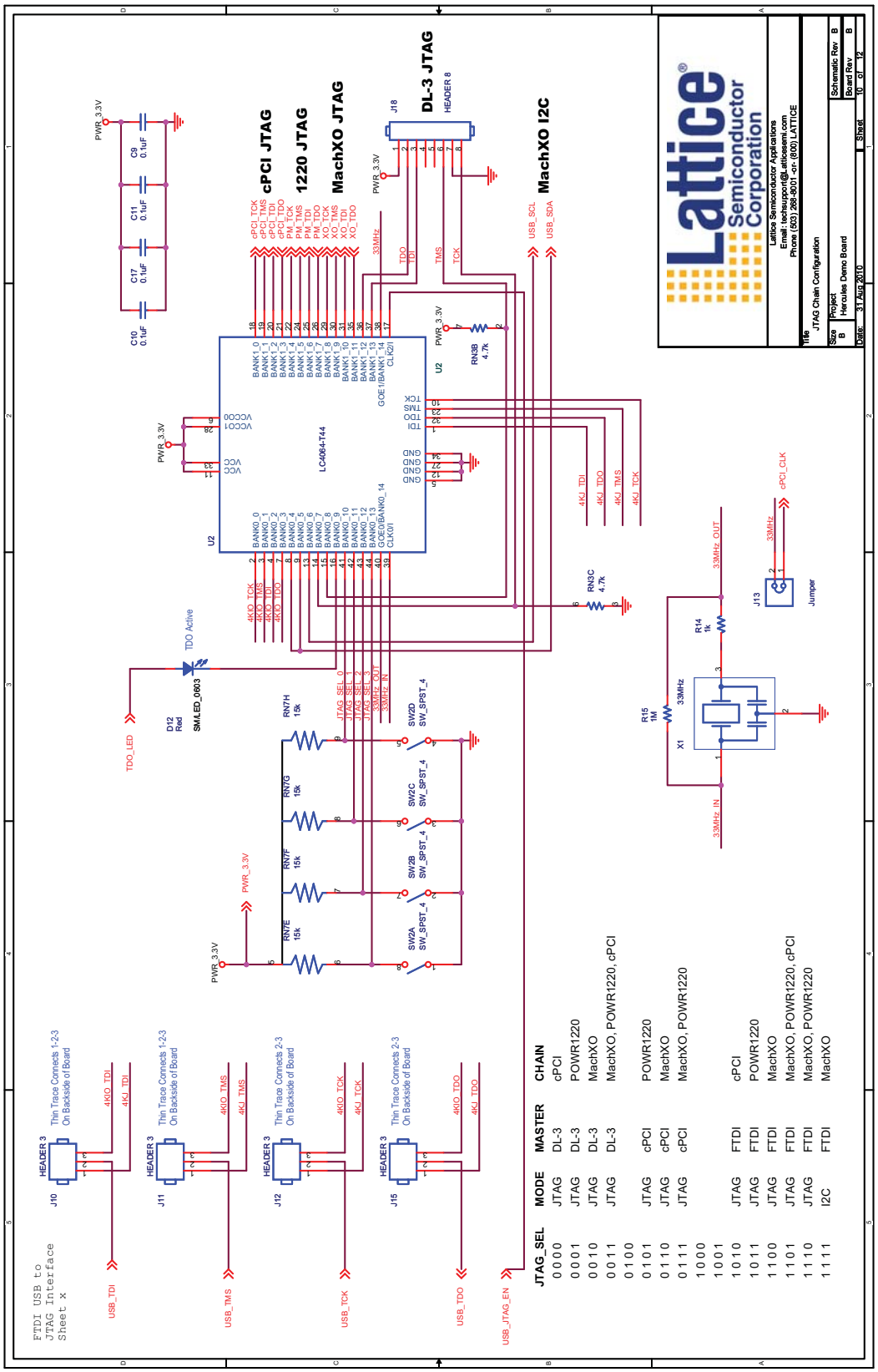


Figure 50. CompactPCI Connector

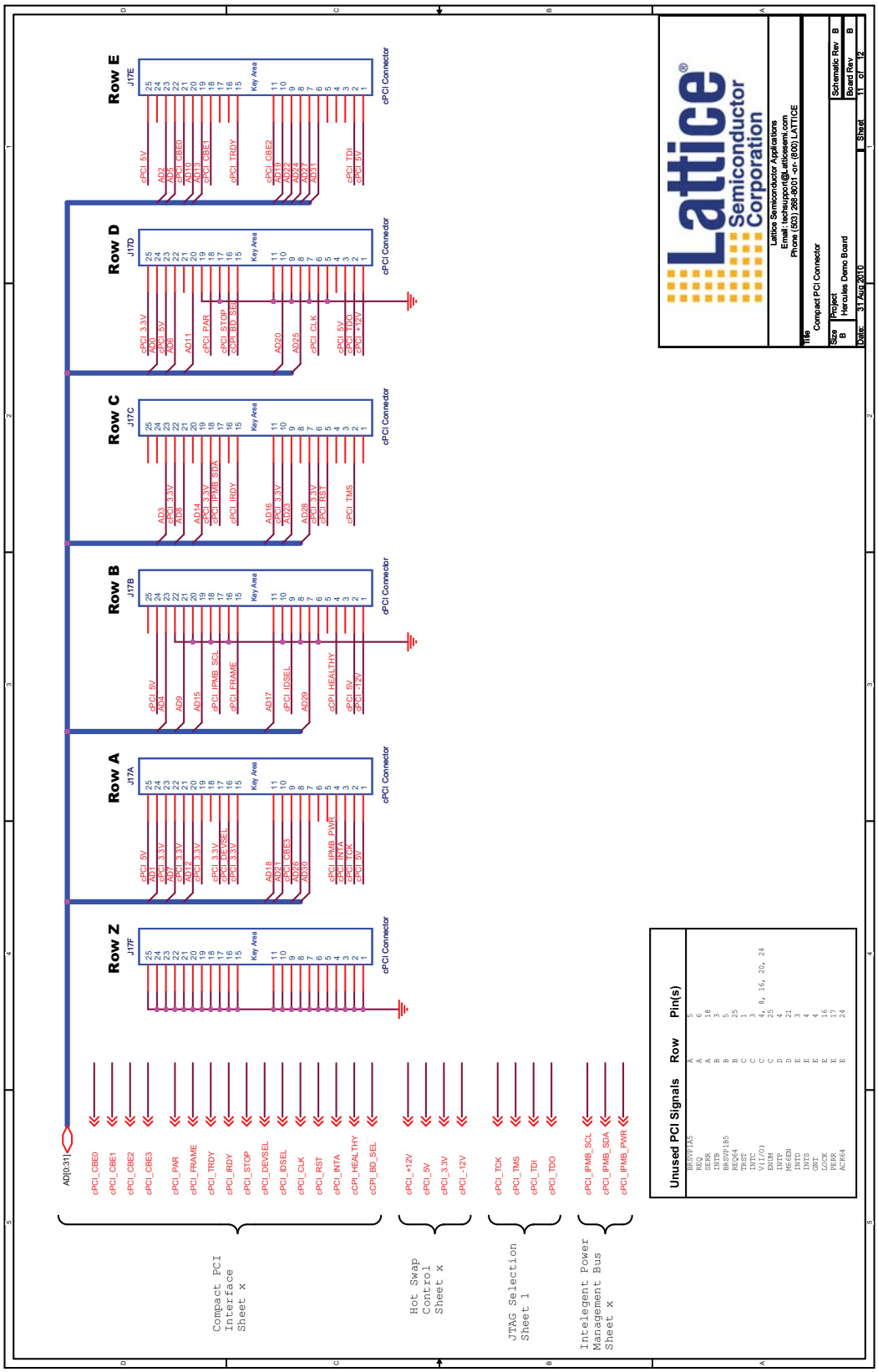
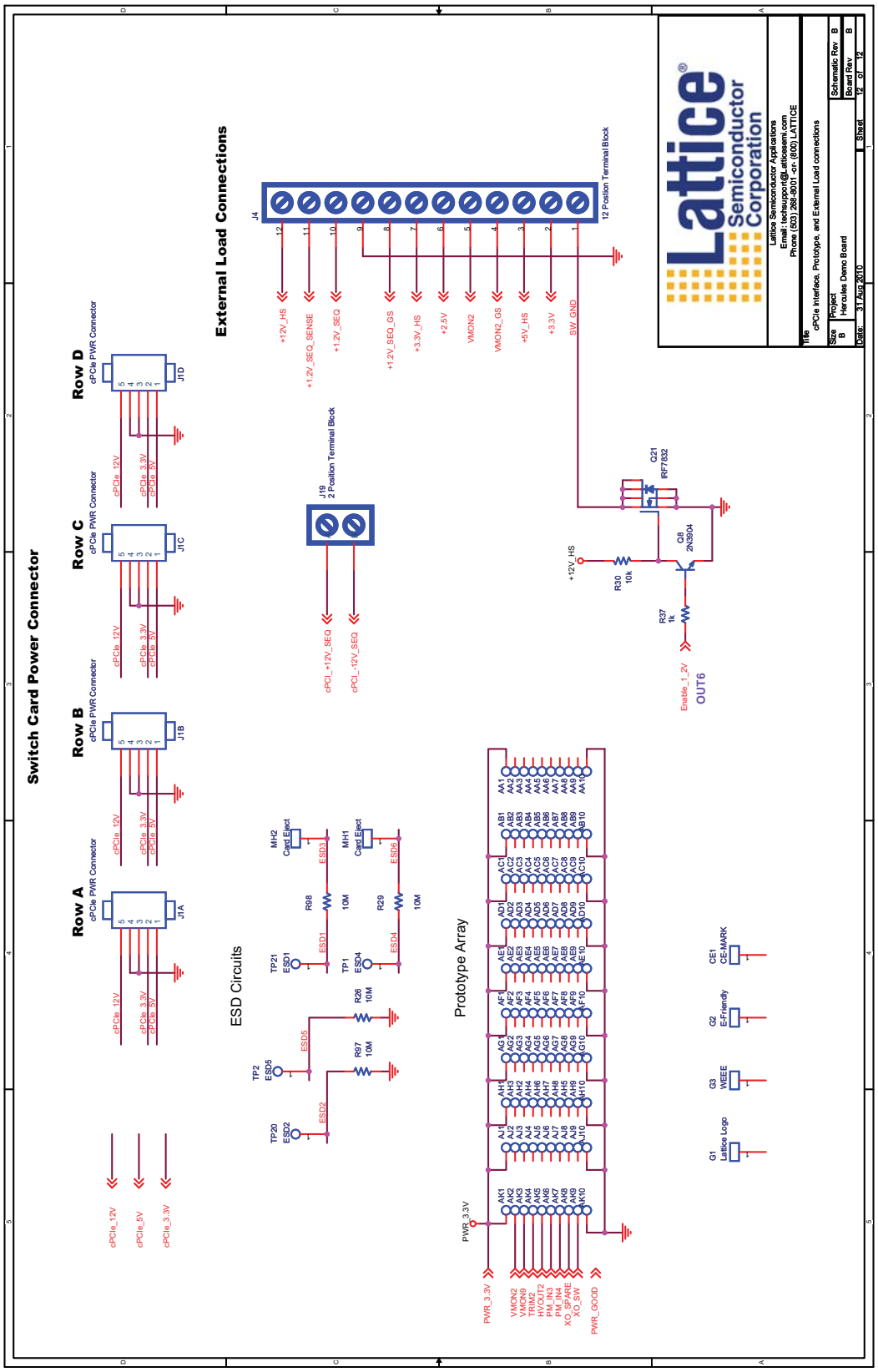


Figure 51. cPCIe Interface, Prototype and External Load Connections



Appendix B. Bill of Materials

Revision B

Advance Board Only

Quantity	Reference Designator(s)	Description	Package	Source	Part Number
ICs					
1	U3	POWR1220AT8	100-Pin TQFP	Lattice	ispPAC-POWR1220AT8
1	U4	MachXO2280	144-Pin TQFP	Lattice	
1	U2	ispMACH 4000	44-Pin TQFP	Lattice	
1	U13	IC USB to Serial (RS-2232)	48-Pin TQFP	FTDI (Future)	FT2232D
1	U12	Serial Memory	8-Pin SOIC	STMicro	M93C46-WMN6TP
1	U5	SPI Flash 2M-bit	8-Pin SOIC 3.9W	STMicro	M25PE20-VMN6TP
2	U7,U8	IC Current Sense Amp.	SOT-23	Zetex	ZXCT1009
1	U9	IC Current Sense Amp.	SOT-23	Zetex	ZXCT1009
1	U6	LCD Display, 3 Digits	T-Hole 12x2	Lumex	LCD-S301C31TR
1	U1	LDO Votage Reg.	SOT-223	ON Semi	NCP1117ST33T3G
1	U10	IC Current Sense Amp.	SOT-23-5	TI	INA197
1	U11	IC I^2C Buffer	SOIC-8	Hendon Semi.	IES5502
Capacitors					
2	C43, C44	Cap 12pF	SMD 0603	Murata Electronics	ERB1885C2E120JDX5D
2	C25, C31	Cap 33pF	SMD 0603	Murata Electronics	ERB1885C2E330JDX5D
4	C24, C33, C49, C50	Cap 10nF 16V	SMD 0603	Panasonic ECG	ECJ-1VB1C103K
30	C5, C6, C7, C8, C9, C10, C11, 12, C13, C14, C15, C16, C17, C18, C19, C22, C23, C26, C27, C28, C29, C30, C32, C35, C45, C46, C47, C48, C51, C52	Cap 0.1uF 16V	SMD 0603	Panasonic ECG	ECJ-1VB1C104K
2	C39, C42	Cap 0.1uF 16V	SMD 0603	Panasonic ECG	ECJ-1VB1C104K
1	C53	Cap 0.33uF	SMD 0603	Panasonic ECG	ECJ-1VB0J334K
2	C20, C34	Cap Tant. 6.8uF 20V	6032-28 (EIA)	Vishay/Sprague	293D685X9020C2TE3
2	C37, C40	Cap Tant. 6.8uF 20V	6032-28 (EIA)	Vishay/Sprague	293D685X9020C2TE3
2	C21,C36	Cap Tan. 10uF 6.3V	SMD 0805	Nichicon	F920J106MPA
2	C38, C41	Cap Tan. 10uF 6.3V	SMD 0805	Nichicon	F920J106MPA
1	C1	Cap 1000uF 25V	0.532" Sq.	Panasonic ECG	EEV-FK1E102Q
2	C2, C3	Cap 680uF 10V	0.327" Sq.	Panasonic ECG	EEE-FK1A681P
1	C4	Cap 47uF 16V	0.25" Qq.	Panasonic ECG	EEE-HAC470WAR
Diodes					
5	D16, D17, D19, D23, D27	Zener 5.1V	SOD-323	Diodes Inc	MMSZ5231BS-7-F
1	D22	Zener 10V	SOD-323	Diodes Inc	MMSZ5240B-7-F
4	D15, D18, D21, D26	Diode 1N4148	SOD-123	Micro Commercial	1N4148W-TP
1	D20	Transzorb 24V	DO-214AA	Littelfuse	SMBJ22CA
7	D1, D2, D8, D9, D10, D11, D12	LED Red	SMD 0603	Lite-On	LTST-C190CKT
4	D3, D4, D5, D6	LED Red	SMD 0603	Lite-On	LTST-C190CKT
2	D7, D14	LED Green	SMD 0603	Lite-On	LTST-C190KGKT
1	D13	LED Blue	SMD 0603	Lite-On	LTST-C190TBKT
5	D24, D25, D28, D29, D31	Rectifier Schottky	SOD-923	ON Semi	NSR0530P2T5G
1	D30	Rectifier Schottky	SOD-923	ON Semi	NSR0530P2T5G
Transistors					
4	Q1, Q2, Q3, Q21	Transistor N-MOSFET	SOIC-8	IR	IRF7832

Quantity	Reference Designator(s)	Description	Package	Source	Part Number
2	Q4, Q5	Transistor N-MOSFET	SOIC-8	IR	IRF7832
4	Q7, Q11, Q15, Q17	Transistor PNP	SOT-23	Fairchild	MMBT2907A
1	Q25	Transistor PNP	SOT-23	Fairchild	MMBT2907A
10	Q8, Q9, Q10, Q12, Q13, Q14, Q16, Q18, Q19, Q22	Transistor NPN	SOT-23	Fairchild	MMBT2369A
3	Q20, Q23, Q27	Transistor NPN	SOT-23	Fairchild	MMBT2369A
1	Q6	Transistor NPN 3A	SOT-223	Diodes/Zetex	FZT649TA
1	Q28	Transistor P-MOSFET	SOT-23	IR	IRLML6402PbF
1	Q24	Transistor P-MOSFET	SOT-23	IR	IRLML6402PbF
Resistors					
1	RN1	Resistor array 1k x 4 Iso.	SMD	CTS Corp.	741X083102JP
2	RN4, RN5	Resistor array 2.2k x 8 Bus	SMD	CTS Corp.	746X101222JP
1	RN3	Resistor array 4.7k x 4 Iso.	SMD	CTS Corp.	741X083472JP
2	RN2, RN6	Resistor array 10k x 4 Iso.	SMD	CTS Corp.	741X083103JP
1	RN7	Resistor array 15k x 8 Bus	SMD	CTS Corp.	746X101153JP
1	R22	Trim Pot 1k Slider	Thru Hole	Alpha	RA2043F-20-10EB1-B1K
2	R20, R21	Resistor 27	SMD 0603	Panasonic ECG	ERJ-3GEYJ270V
6	R2, R3, R4, R7, R8, R72	Resistor 100	SMD 0603	Panasonic ECG	ERJ-3GEYJ101V
4	R6, R9, R10, R11	Resistor 100	SMD 0603	Panasonic ECG	ERJ-3GEYJ101V
1	R28	Resistor 270	SMD 0603	Panasonic ECG	ERJ-3GEYJ271V
2	R85, R92	Resistor 470	SMD 0603	Panasonic ECG	ERJ-3GEYJ471V
8	R14, R31, R37, R45, R50, R58, R83, R95	Resistor 1k	SMD 0603	Panasonic ECG	ERJ-3GEYJ102V
2	R13, R89	Resistor 1.5k	SMD 0603	Panasonic ECG	ERJ-3GEYJ152V
3	R79, R90, R93	Resistor 2.2k	SMD 0603	Panasonic ECG	ERJ-3GEYJ222V
3	R77, R81, R82	Resistor 2.2k	SMD 0603	Panasonic ECG	ERJ-3GEYJ222V
3	R47, R66, R78,	Resistor 3.3k	SMD 0603	Panasonic ECG	ERJ-3GEYJ332V
1	R12	Resistor 4.7k	SMD 0603	Panasonic ECG	ERJ-3GEYJ472V
4	R16, R17, R48, R64	Resistor 4.7k	SMD 0603	Panasonic ECG	ERJ-3GEYJ472V
2	R30, R80	Resistor 10k	SMD 0603	Panasonic ECG	ERJ-3GEYJ103V
3	R75, R76, R88	Resistor 10k	SMD 0603	Panasonic ECG	ERJ-3GEYJ103V
3	R33, R59, R87	Resistor 100k	SMD 0603	Panasonic ECG	ERJ-3GEYJ104V
3	R86, R91, R94	Resistor 100k	SMD 0603	Panasonic ECG	ERJ-3GEYJ104V
2	R23, R49,	Resistor 200k	SMD 0603	Panasonic ECG	ERJ-3GEYJ204V
1	R34	Resistor 510k	SMD 0603	Panasonic ECG	ERJ-3GEYJ514V
3	R15, R84, R96	Resistor 1M	SMD 0603	Panasonic ECG	ERJ-3GEYJ105V
3	R27, R42, R57	Resistor 4.7M	SMD 0603	Panasonic ECG	ERJ-3GEYJ475V
4	R26, R29, R97, R98	Resistor 10M	SMD 0603	Panasonic ECG	ERJ-3GEYJ106V
5	R35, R36, R40, R51, R52	Resistor 1.02k 0.5%	SMD 0603	Susumu Co.	RR0816P-1021-D-02H
1	R65	Resistor 1.02k 0.5%	SMD 0603	Susumu Co.	RR0816P-1021-D-02H
1	R74	Resistor 1.87k 0.5%	SMD 0603	Susumu Co.	RR0816P-1871-D-27H
3	R41, R44, R60	Resistor 3k 0.5%	SMD 0603	Susumu Co.	RR0816P-302-D
1	R69	Resistor 5.10k 0.5%	SMD 0603	Yageo	RT0603DRD075K1L
1	R63	Resistor 5.76k 0.5%	SMD 0603	Susumu Co.	RR0816P-5761-D-74H
2	R43, R46,	Resistor 10.7k 0.5%	SMD 0603	Susumu Co.	RR0816P-1072-D-04C
1	R55	Resistor 13.0k 0.5%	SMD 0603	Yageo	RT0603DRD0713KL
1	R39	Resistor 17.4k 0.5%	SMD 0603	Yageo	RT0603DRD0717K4L
1	R24	Resistor 22.0k 0.5%	SMD 0603	Susumu Co.	RR0816P-223-D
1	R32	Resistor 30.0k 0.5%	SMD 0603	Susumu Co.	RR0816P-303-D

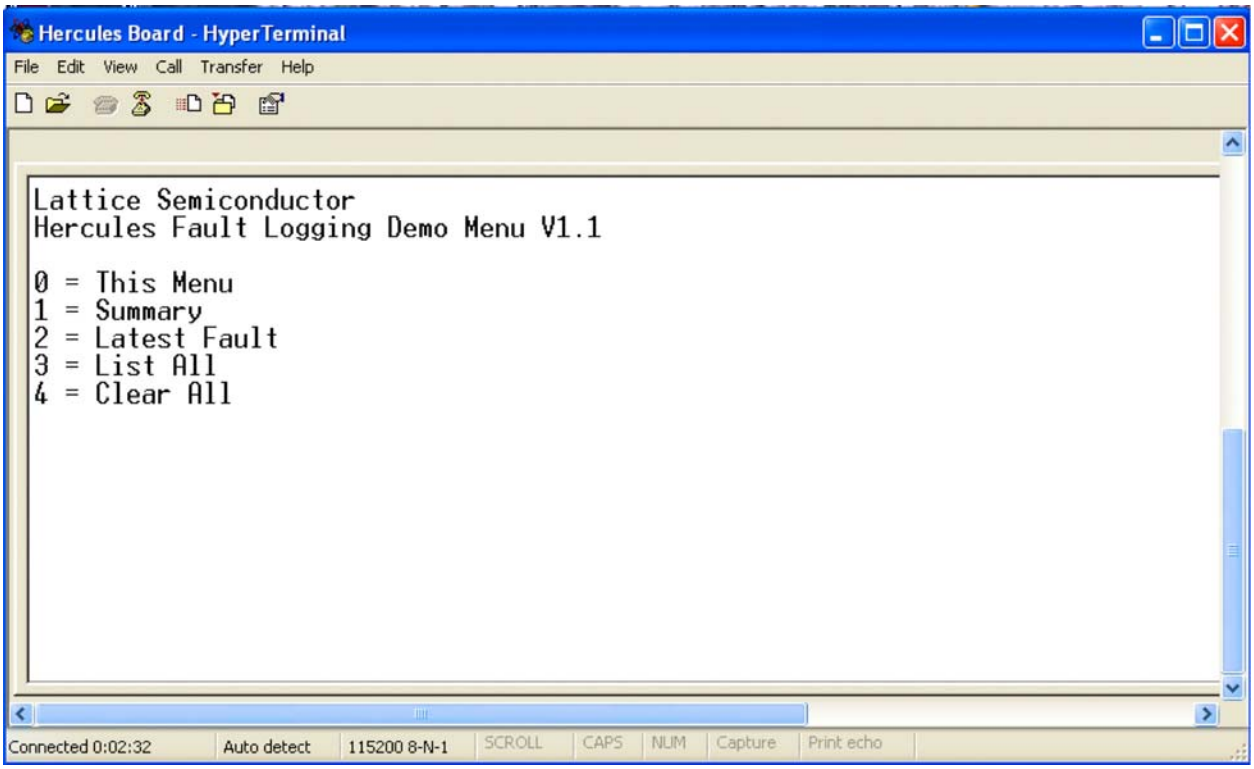
Quantity	Reference Designator(s)	Description	Package	Source	Part Number
1	R68	Resistor 43.0k 0.5%	SMD 0603	Susumu Co.	RR0816P-433-D
2	R61, R71	Resistor 0.005 1% 2W	SMD 2512	Ohmite	MCS3264R005FER
2	R1, R5	Resistor 0.01 1% 3W	SMD 2512	Bourns	CRA2512-FZ-R010ELF
Connectors, Jumpers, Miscellaneous					
1	J1	cPCIe Power Connector	Thru Hole	Tyco Electronics	120955-1
1	J17	cPCI Connector	Thru Hole	AVX Connectors	277200110000002
1	J2	2.1mm DC power connector	Thru Hole	CUI, Inc.	PJ-102A
1	J14	Male USB Type-B Mini Connector	SMD-USB	Hirose Electric Co.	UX60-MB-5ST
1	X2	Crystal 6 MHz SMD 10x4.5	SMD-10x4.5	Citizen America Corp.	HCM49 6.000MABJ-UT
1	X1	Resonator 33MHz	SMD	Abracon Corp.	AWSCR-33.00CW-T
1	L1	Ferrite Bead	SMD 0603	Stewart	HI0603P600R-10
4	J5,J6,J12,J15	Header 3x1 0.1"	Thru Hole	Molex Connector	22-28-4364
2	J7,J8	Header 3x1 0.1"	Thru Hole	Molex Connector	22-28-4364
1	J16	Recepticle 16x2 0.1"	Thru Hole	Samtec Inc.	SLW-116-01-G-D
1	J19	Screw Terminal Header 2-pos	Thru Hole	Phoenix Contact	1725656
2	J3,J9	Screw Terminal Header 2-pos	Thru Hole	Phoenix Contact	1727010
1	J4	Screw Terminal Header 12-pos	Thru Hole	Phoenix Contact	1725753
1	SW1	Switch DP-DT Power	Thru Hole	NKK Switches	M2022SS1W03-RO
2	SW3,SW4	Push Button SW	SMD	Panasonic ECG	EVQ-Q2K03W
1	SW5	8-Position Switch VID	SMD	CTS	195-8MST
1	SW2	4-Position Switch JTAG	SMD	CTS	195-4MST
1	DCM1	DC-DC In=12V Out=1.2V @ 3A	DOSA-SMD	MuRata	OKY-T/3-D12P-C
2	DCM2,DCM3	DC-DC In=5V Out=3A	DOSA-SMD	MuRata	OKY-T/3-W5N-C
4		3M Rubber Bump-ons (P416)	N/A	3M	SJ-5003
Not Installed; Factory Testing					
	J13	Header 2x1 0.1"	Thru Hole	Molex Connector	22-28-4364
	J18	Header 8x1 0.1"	Thru Hole	Molex Connector	22-28-4364
	J10, J11	Header 3x1 0.1"	Thru Hole	Molex Connector	22-28-4364
	R18, R19	Resistor 0.2	SMD 0805	Susumu Co	RL1220S-R20-F
	R53, R67	Resistor 0.0	SMD 0603	Panasonic ECG	ERJ-3GEY0R00V

Appendix C. Hercules Hyper-Terminal User Interface

The LatticeMico8 supports a series of simple commands, for use with the Fault Logger demonstration, via Hyper-Terminal (or equivalent) running on a PC using the UART reference design and the external USB-to-UART converter. A list of the available commands is shown in the HyperTerminal screen capture given below in Figure 52. The user types the number for the corresponding command.

0: Displays the Menu

Figure 52. Demonstration Design Fault Logger Commands Controlled via HyperTerminal Window on PC



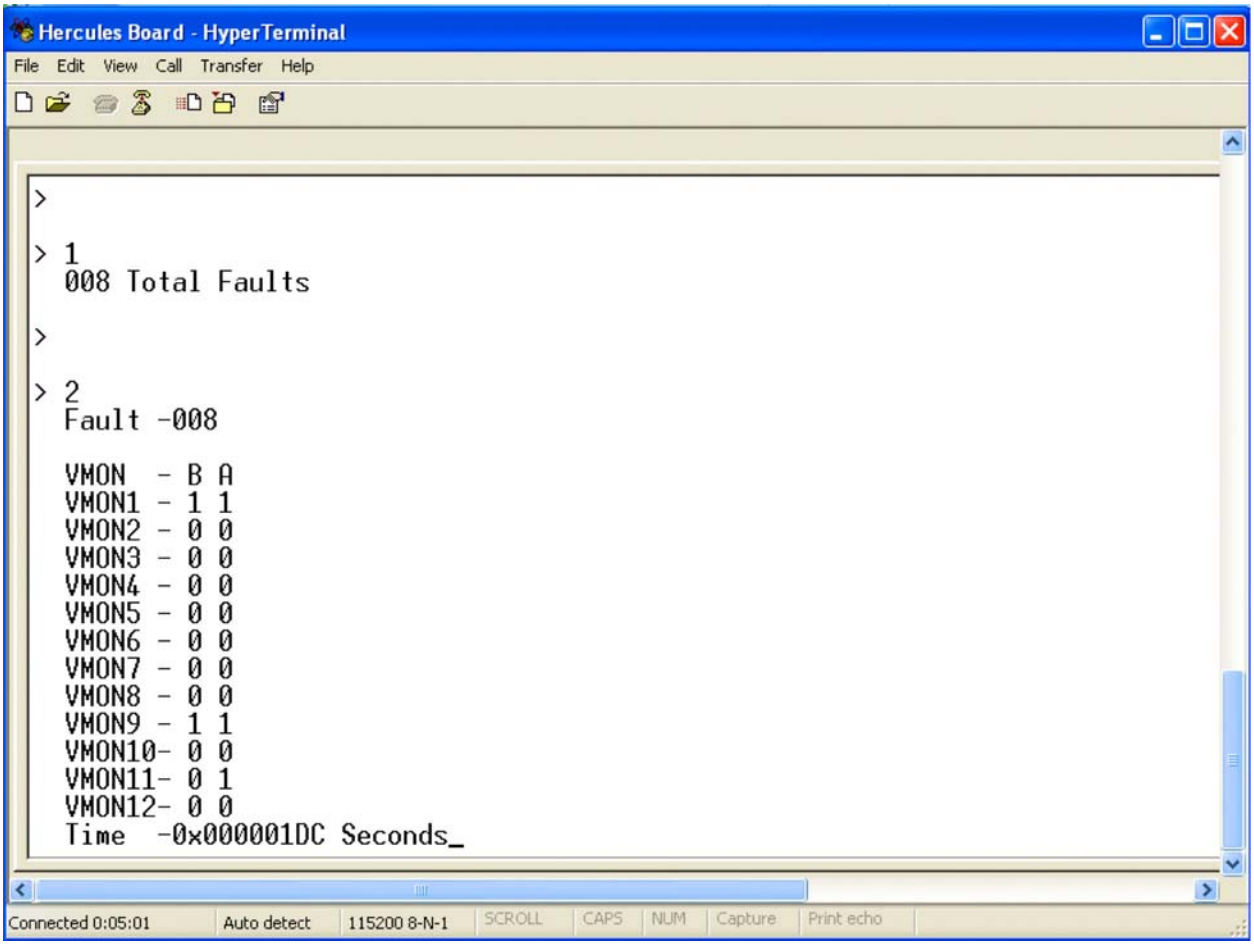
1: Displays the Fault Summary

The fault summary is the total number of faults detected since the last Clear All command. An example is shown at the top of Figure 53.

2: Displays the Data Stored for the Last Logged Fault

The data stored is the set of all VMON signals captured by the Fault Logging command as described previously. The last logged fault data is displayed on the screen, as shown in Figure 7, with the fault number listed first 008 in this example), followed by the detailed listing of each VMON value. Note that the VMON A and B outputs are displayed in the associated column (e.g., VMON1_B is the first signal in the VMON1 line and VMON1_B is the second).

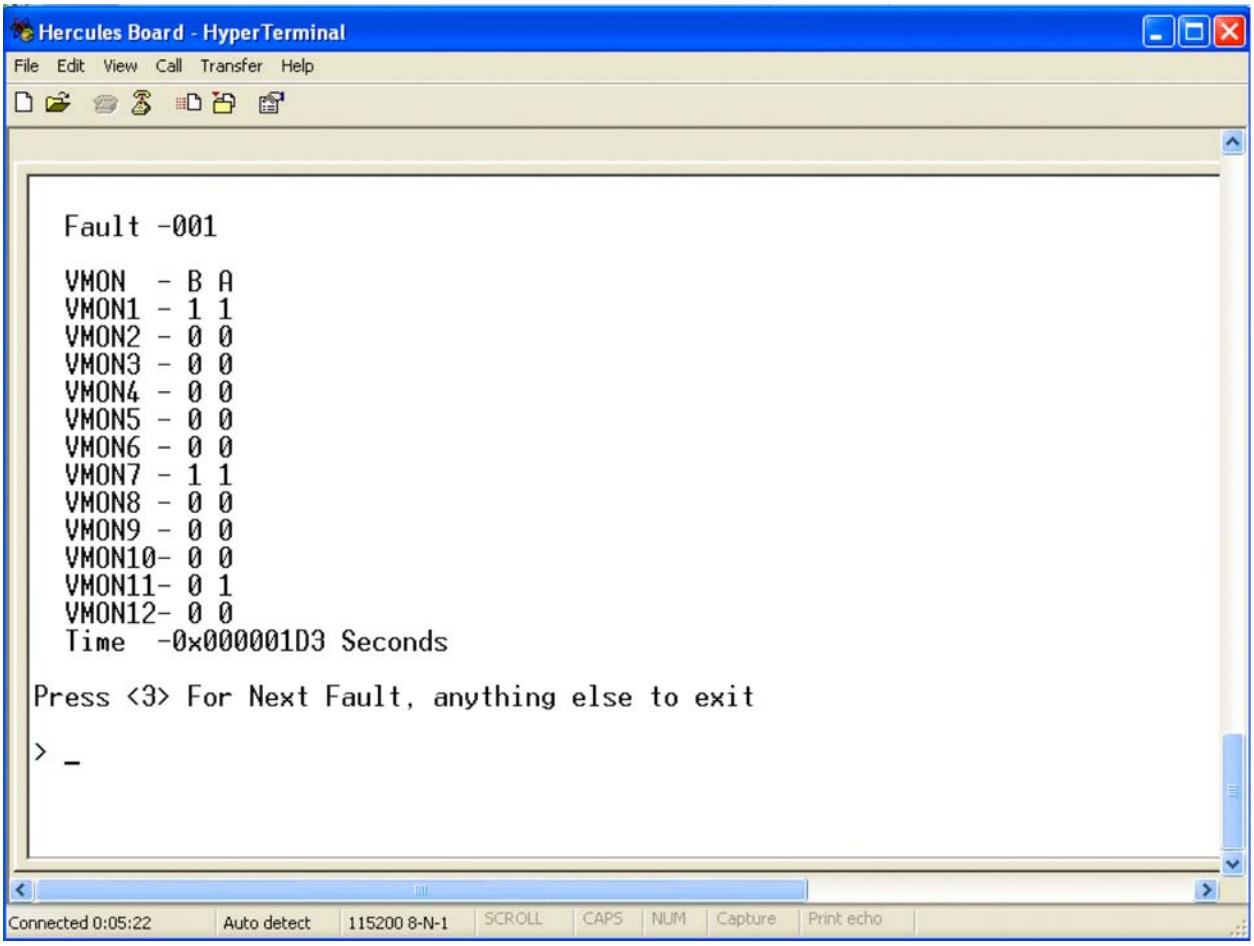
Figure 53. Screen Capture of Fault Summary and Last Fault Example Display



3: Lists all the fault data, starting with fault 001

An example display is shown in Figure 54. The display format is the same as described above. If the command is repeated the display refreshes with the next fault. You can continue through the entire list by repeatedly typing 3.

Figure 54. Screen Capture of List All Example Display



4: Clears all the fault data from the fault logging memory

As mentioned previously the Hercules_demo.asm file included in the design project folder contains several useful subroutines that can be used directly or customized for your design. The code to initialize the UART, get data/send data to/from the UART, and read or write to/from the SPI memory are all particularly useful for implementing a fault logging subsystem. Refer to the Hercules_demo.asm file for the detailed description in the comments fields for each routine.

Using HyperTerminal with the Hercules Demonstration Design

To use the HyperTerminal User Interface described above the following steps show be completed in the order given. These steps are for Windows XP. For other operating systems, check the Lattice web site for updates.

1. Download USB drivers for the Hercules Evaluation Board. They are located on the Hercules web page on the Lattice web site. The file folder is named "CDM 2.04.06 WHQL Certified.zip". Place the unzipped folder in a convenient location.
2. Power-up the Hercules Evaluation Board by plugging in the wall adaptor.
3. Plug the USB cable into the Hercules Evaluation Board and then into your PC.
4. Drivers will be found automatically in some systems. If not, direct the installer to the unzipped driver download folder on the PC. The drivers should install and will indicate the new hardware is now ready to use.

5. Open the Device Manager. You should find two new USB Serial Converter entries under the **Universal Serial Bus Controller** category. Expand the category if necessary to see the entries. Check the **Ports** category and not the existing COM ports being used. A new one will be added for the Hercules USB connection.

6. Select the properties for the USB Serial Converter B (A is used for the JTAG function and should be left alone). In the **Advanced** tab check the **VCP** box. Click **OK**.

7. Unplug the USB cable and then plug it back in.

8. In Device Manager, check the **Ports** category (expand if needed). A new COM port will be listed (probably with a USB Serial Port heading). Note the COM port associated with it. You will use this port number with HyperTerminal.

9. Open HyperTerminal. Select a name and icon. Select the **Connect Using** pull-down menu and pick the COM port identified previously as the Hercules port.

10. Select the following settings and then click **OK**:

    a. BPS 115200
    b. Data Bits 8
    c. Parity None
    d. Stop Bits 1
    e. Flow Control None

11. A terminal window will open.

12. Press the **Reset** button (SW4) on the Hercules Evaluation Board. The command window (as seen in Figure 6) should be displayed.