**Elop Condor MS-C10A SBC Board**

**Software Requirements Specifications (SRS)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name and Title** | **Signature** | **Date** |
| Issued by: | Lior Yosef, R&D Engineer |  | First Issue |
| Michael Khomyakov, Firmware Engineer |  | 10/06/2024 |
| Approved by: | Bar Shpilman, R&D Team Leader |  |  |

**Effective Date:** 24/07/2024 **Last Updated By:** Michael K.

Date 15-04-2024

# Contents

[1. Contents 1](#_Toc172102986)

[2. General Description 3](#_Toc172102987)

[2.1. System Architecture Diagram 3](#_Toc172102988)

[2.2. Module Design and Requirements Description 4](#_Toc172102989)

[2.2.1. Sensing Module 4](#_Toc172102990)

[2.2.2. DC/DC Boards 11](#_Toc172102991)

[2.2.3. Relay Module 19](#_Toc172102992)

[2.2.4. Fans Control Module 20](#_Toc172102993)

[2.2.5. Power Status Indication Module 22](#_Toc172102994)

[2.2.6. External Interfaces 23](#_Toc172102995)

[2.2.7. MIU Interface 25](#_Toc172102996)

[3. Runtime Operation Specifications 27](#_Toc172102997)

[3.1. Start-Up Sequence 27](#_Toc172102998)

[3.1.1. Overview 27](#_Toc172102999)

[3.1.2. System Power ON Sequence Requirements 27](#_Toc172103000)

[3.2. System Reset 29](#_Toc172103001)

[3.2.1. Specific Requirements 29](#_Toc172103002)

[3.3. Shutdown Sequences 30](#_Toc172103003)

[3.3.1. Overview 30](#_Toc172103004)

[3.3.2. Standard Shutdown Specific Requirements 31](#_Toc172103005)

[3.3.3. Emergency Shutdown Specific Requirements 32](#_Toc172103006)

[3.4. Protection Specifications 33](#_Toc172103007)

[3.4.1. Over Voltage Protection (OVP) 33](#_Toc172103008)

[3.4.2. Under Voltage Protection (UVP) 34](#_Toc172103009)

[3.4.3. Over Temperature Protection (OTP) 36](#_Toc172103010)

[4. Elapsed time FLASH memory communication 37](#_Toc172103011)

[5. Communication Interface 39](#_Toc172103012)

[6. Log File functionality 40](#_Toc172103013)

[6.1. General description: 40](#_Toc172103014)

[6.2. PSU Logfile handling - general rules 41](#_Toc172103015)

[7. 29. FPGA Measurements 42](#_Toc172103016)

[SHUTDOWN\_OUT\_FPGA 52](#_Toc172103017)

# General Description

## System Architecture Diagram

The following diagram provides a complete description of Architectural design of the SBC board:

User LED [MIO47]

Program Flash Memory

DDR3 DRAM

ON/OFF

QSPI

DDR3

**Zynq 7000**

**System on Chip**

DC/DC Boards Interface

ON/OFF Logic

eMMC

Module

SD Card Interface

Power Good Logic

Power In Status

RS485 (RX, TX, DE)

Power Status

Module

Power Out Status

Fan Control Modules

FAN Control PWM

Aircraft Interface

Speed Sense PWM

Lamp Status

ON/OFF Logic

Pod Status

Current Sense

Reset

Full Duplex SPI

MIU Interface Module

Shutdown

Emergency

Voltage Sense

Spare Logic

Full Duplex SPI

MDC 100MbE

Ethernet PHY Module

Sensing Module

Time & S/N Storage Memory

QSPI

ON/OFF Logic

TCU Module

External Interfaces

ON/OFF Logic

Figure 1 - SBC Block diagram with specified interfaces

Relay Module

## Module Design and Requirements Description

### Sensing Module

#### Current Sensing Module

~~The module is based on ADS7951, 8-Channel 12-bit A/D converter, while communication with the module is performed by Full-Duplex SPI interface. (TBD In New Revision).~~

The module is based on ADS7950, 4-Channel 12-bit A/D converter, while communication with the module is performed by Full-Duplex SPI interface.

The SPI interface is implemented according to the following pinout diagram on the FPGA:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Package Pin | SPI Protocol Name | FPGA Direction |
| I\_sns\_ADC\_CS\_fpga | V7 | NSS (Active Low) | OUT |
| I\_sns\_ADC\_SCLK\_fpga | T9 | SCK | OUT |
| I\_sns\_ADC\_SDI\_fpga | U10 | MOSI | OUT |
| I\_sns\_ADC\_SDO\_fpga | Y7 | MISO | IN |

The following table describes the A/D module analog inputs alignment:

|  |  |  |
| --- | --- | --- |
| Name | A/D Channel | Description |
| DC\_PWR\_I\_sns | Channel 0 | Output DC Power Current |
| PH1\_I\_sns | Channel 1 | Phase 1 AC Current |
| PH2\_I\_sns | Channel 2 | Phase 2 AC Current |
| PH3\_I\_sns | Channel 3 | Phase 3 AC Current |
| ~~28V\_IN\_sns~~ | ~~Channel 4~~ | ~~28V Input Voltage (TBD In new rev.)~~ |

The ADS7950 is a 12-bit 4-channel Analog to Digital converter module, operated in a full-duplex SPI mode with a periodic sampling feature. The following is the sequence for ADS7951 operation:

1. **FPGA Initialization:**
   1. Rising Edge SCK for valid MOSI (CPOL= 0).
   2. Falling Edge SCK for valid MISO (CPHA = 0).
   3. Active LOW chip select (NSS).
   4. During the transactions: NSS = LOW.
   5. No transactions: NSS = HIGH (Deselected).
2. **ADS7950 Initialization:**
   1. Transmit Frame #1: 0x8000.
   2. Transmit Frame #2: 0xFFFF.
   3. Transmit Frame #3: 0x2800.
3. After the initialization frames are transmitted to the A/D module, periodic sampling can be performed as follows:
   1. Transmit Frame #3: 0x2800 and receive data on the same frame (16-bit frame).
   2. Decode the received 16-bit frame as follows: <Channel><Data>.
      1. <Channel> Represents the number of sampled channel [0...3].
      2. <Data> Represents the 12-bit sampled value [0...4095] from a specified channel.

**Example for data Reception frame-by-frame:**

1. Transmit = 0x2800 (MOSI), Receive = 0x1AAA
   1. Sampled Channel[0…3]: Channel 1 (0x10) – PH1\_I\_sns.
   2. Sampled Data[0:0xFFF]: AAA.
2. Transmit 0x2800 (MOSI), Receive = 0x2BBB
   1. Sampled Channel[0…3]: Channel 2 (0x20) – PH2\_I\_sns.
   2. Sampled Data[0:0xFFF]: 0xBBB.

#### Voltage Sensing Module

The module is based on ADS7951, 8-Channel 12-bit A/D converter, while communication with the module is performed by Full-Duplex SPI interface. The SPI interface is implemented according to the following pinout diagram on the FPGA:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Package Pin | SPI Protocol Name | FPGA Direction |
| HV\_ADC\_CS\_fpga | M15 | NSS (Active Low) | OUT |
| HV\_ADC\_SCLK\_fpga | L14 | SCK | OUT |
| HV\_ADC\_SDI\_fpga | L15 | MOSI | OUT |
| HV\_ADC\_SDO\_fpga | M14 | MISO | IN |

The following table describes the A/D module analog inputs alignment:

|  |  |  |
| --- | --- | --- |
| Name | A/D Channel | Description |
| OUT4\_sns | Channel 0 | Single-Phase Output Voltage |
| Vsns\_PH1 | Channel 1 | Phase 1 Input Voltage |
| Vsns\_PH2 | Channel 2 | Phase 2 Input Voltage |
| Vsns\_PH3 | Channel 3 | Phase 3 Input Voltage |
| OUT4\_Isns | Channel 4 | Single-Phase Output Current |
| Vsns\_PH\_C\_RLY | Channel 5 | Phase 3 Relay Output Voltage |
| Vsns\_PH\_B\_RLY | Channel 6 | Phase 2 Relay Output Voltage |
| Vsns\_PH\_A\_RLY | Channel 7 | Phase 1 Relay Output Voltage |

The ADS7951 is a 12-bit, 8-channel Analog to Digital converter module, operated in a full-duplex SPI mode with a periodic sampling feature. The following is the sequence for ADS7951 operation:

1. **FPGA Initialization:**
   1. Rising Edge SCK: MOSI (CPOL= 0).
   2. Falling Edge SCK: MISO (CPHA = 0).
   3. Active LOW chip select (NSS).
   4. During the transactions: NSS = LOW.
   5. No transactions: NSS = HIGH (Deselected).
   6. Each frame is 2 bytes (16-bit).
2. **ADS7951 Initialization:**
   1. Transmit Frame #1: 0x8000.
   2. Transmit Frame #2: 0xFFFF.
   3. Transmit Frame #3: 0x2800.
3. After the initialization frames are transmitted to the A/D module, periodic sampling can be performed as follows:
   1. Transmit Frame #3: 0x2800 and receive data on the same frame (16-bit frame).
   2. Decode the received 16-bit frame as follows: <Channel><Data>.
      1. <Channel> Represents the number of sampled channel [0...7].
      2. <Data> Represents the 12-bit sampled value [0...4095] from a specified channel.

**Example for data Reception frame-by-frame:**

1. Transmit = 0x2800 (MOSI), Receive = 0x2AAA
   1. Sampled Channel[0…7]: 2 – Vsns\_PH\_C\_RLY.
   2. Sampled Data[0:0xFFF]: AAA.
2. Transmit 0x2800 (MOSI), Receive = 0x3BBB
   1. Sampled Channel[0…7]: 3 – OUT4\_Isns.
   2. Sampled Data[0:0xFFF]: 0xBBB.

#### Sampled Data Conversion

Since all the sampled values represent the raw 12-bit A/D measurements, the values shall be converted to the valid data in accordance with the specified channel and value implemented in the design.

1. **A/D Sample to Instantaneous Voltage Conversions:**

To convert the values from unsigned A/D raw 12-bit data to 16-bit signed value, the following equation is used:

Since the IRS document provided by customer provides a specific set of requirements for the scale factor and range, the following table shall be implemented:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | A/D Channel | Variable Length | ADC Module | Description | Parameter A |
| OUT4\_sns | Channel 0 | 2-bytes | ADS7951 | Single-Phase Output Voltage | 1.08881 |
| Vsns\_PH1 | Channel 1 | 2-bytes | Phase 1 Input Voltage |
| Vsns\_PH2 | Channel 2 | 2-bytes | Phase 2 Input Voltage |
| Vsns\_PH3 | Channel 3 | 2-bytes | Phase 3 Input Voltage |
| Vsns\_PH\_C\_RLY | Channel 5 | 2-bytes | Phase 3 Relay Output Voltage |
| Vsns\_PH\_B\_RLY | Channel 6 | 2-bytes | Phase 2 Relay Output Voltage |
| Vsns\_PH\_A\_RLY | Channel 7 | 2-bytes | Phase 1 Relay Output Voltage |

**Example:** A/D Channel 3 (Vsns\_PH3) 12-bit unsigned sample is 3000[d], hence the stored 16-bit value shall be:

1. **A/D Sample to Current Conversions:**

Since all the sampled values represent the raw 12-bit A/D measurements, the values shall be converted to the valid data in accordance with the specified channel and value implemented in the design. The current samples obey the following rule:

The following table is the values of C and D parameters required for conversion:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | A/D Channel | Variable Length | ADC Module | Description | Parameter K |
| **OUT4\_Isns** | **Channel 3** | **2-bytes** | **ADS7951** | **Single-Phase Output Current** | 0.29304 |
| DC\_PWR\_I\_sns | Channel 0 | 2-bytes | ADS7950 | Output DC Power Current |
| PH1\_I\_sns | Channel 1 | 2-bytes | ADS7950 | Phase 1 AC Current |
| PH2\_I\_sns | Channel 2 | 2-bytes | ADS7950 | Phase 2 AC Current |
| PH3\_I\_sns | Channel 3 | 2-bytes | ADS7950 | Phase 3 AC Current |
| ~~28V\_IN\_sns~~ | ~~Channel 4~~ | 2-bytes | ~~ADS7951~~ | ~~28V Input Voltage~~ | ~~TBD in new Rev.~~ |

**Example**: A/D Channel 1 (PH1\_I\_sns) sample is 2400[d], hence the measured current value:

1. **Power Calculation:**

The calculated power is a sum of all the three (3) phases of current multiplied by voltage. The following is an example for power calculation:

|  |  |  |  |
| --- | --- | --- | --- |
| Time | Instantaneous Voltage  [100mV] | Instantaneous Current [50mA] | Instantaneous Power |
| 100us |  | 20 |  |
| 200us | 200 | 40 |  |
| 300us | 300 | 60 |  |
| 400us | 400 | 80 |  |
| 500us | 500 | 100 |  |
| 600us | 600 | 120 |  |
| 700us | 700 | 140 |  |
| 800us | 800 | 160 |  |
| 900us | 900 | 180 |  |
| 1ms | 1000 | 200 |  |

#### A/D Measurements and Log File compatibility

The following table provides information about the values that will be reported to the log file

|  |  |  |  |
| --- | --- | --- | --- |
| # | Type | Name in Log File | Name in FPGA |
| ~~2~~ | ~~S2~~ | ~~VDC\_IN~~ | ~~TBD~~ |
| 3 | S2 | VAC\_IN\_PH\_A | Vsns\_PH1 |
| 4 | S2 | VAC\_IN\_PH\_B | Vsns\_PH2 |
| 5 | S2 | VAC\_IN\_PH\_C | Vsns\_PH3 |
| 6 | S2 | I\_DC\_IN | DC\_PWR\_I\_sns |
| 7 | S2 | I\_AC\_IN\_PH\_A | PH1\_I\_sns |
| 8 | S2 | I\_AC \_IN\_PH\_B | PH2\_I\_sns |
| 9 | S2 | I\_AC \_IN\_PH\_C | PH3\_I\_sns |
| 12 | S2 | V\_OUT\_3\_ph1 | Vsns\_PH\_A\_RLY |
| 13 | S2 | V\_OUT\_3\_ph2 | Vsns\_PH\_B\_RLY |
| 14 | S2 | V\_OUT\_3\_ph3 | Vsns\_PH\_C\_RLY |
| 15 | S2 | V\_OUT\_4 | OUT4\_sns |
| 24 | S2 | I\_OUT\_3\_ph1 | PH1\_I\_sns |
| 25 | S2 | I\_OUT\_3\_ph2 | PH2\_I\_sns |
| 26 | S2 | I\_OUT\_3\_ph3 | PH3\_I\_sns |
| 27 | S2 | I\_OUT\_4 | OUT4\_Isns |
| 34 | S2 | AC\_Power | See specific Requirements section **2.2.1.3.C** |

#### Sensing Module Specific Requirements

1. The firmware shall sample the values with a sample rate of at least **10KSamples/sec.** (25 Points on the complete 400[Hz] AC cycle).
2. The firmware shall calculate the RMS values in accordance with the calculated zero crossings to determine the cycles timing (**See appendix A: Zero Crossing Detection**):
   1. **Voltage Calculation:** Three (3) consecutive cycles of AC voltage.

RMS value that is being transferred through Running average.

* 1. **Current Calculation:** Three (3) consecutive cycles of AC current.

RMS value that is being transferred through Running average.

* 1. **Average Power Calculation:** Total power calculation for 3-Phase input. The power shall be calculated by taking an average of 10 instantaneous Current and voltage points per 1[millisecond] for **each phase**: The following is the equation to use in the implementation:

1. The firmware shall provide the main application manager (Runtime) access to all the input data received from the Sensing Module with a sampling rate of at least **1KSamples/sec**. for all the measurements.
2. The Sensing Module shall perform periodic self-test and provide an output to the Runtime application for the following events:
   1. Voltage sensing module does not communicate through SPI correctly.
   2. Current sensing module does not communicate through SPI correctly.

**Sequence Example:**

The number of points for the sampling frequency was set to 25 points per 2.5mS cycle: Hence the number of points for the calculation (N) shall be implemented as a function of input frequency as described in the following:

1. Determine the signal period (Refer to Appendix A: Zero Crossing Detection):
2. Determine the number of points:
3. Calculate the , and values.

### DC/DC Boards

#### Interface Description

|  |  |  |
| --- | --- | --- |
| Name | Description | FPGA Direction |
| RS485\_RxD\_i | UART Transmit Data | IN |
| RS485\_TxD\_i | UART Receive Data | OUT |
| RS485\_DE\_i | UART Data Direction | OUT |
| EN\_PSU\_i | Board Enable (Active High) | OUT |
| PG\_PSU\_i | Board Power Good (Active High) | IN |

The system contains 9 operating board units with implementation of the logical control and RS485 communication features. The following is the RS485-based interface and logical control for each board:

The “i” character in the line alias name represents the index of the DC/DC board, whereas by implemented design.

#### Logical Communication

1. **Logical output EN\_PSU\_i:** The Board Enable signal is active HIGH logical output from the FPGA, that provides Turn ON / Turn OFF commands to the DC/DC or Main Board outputs from the System.
2. **Logical input PG\_PSU\_i:** The Board Power Good signal is active HIGH logical input to the FPGA. This input is an indication for a FPGA, that the corresponding board status is OK (Driven HIGH) or FAIL (Driven LOW).

#### RS485 Communication

1. RS485\_DE\_i: The Data Enable signal is a RS485 data direction selection FPGA output used in communication handling.
2. RS485\_RxD\_i: The signal is the UART data receive line input to the FPGA.
3. RS485\_TxD\_i: The signal is the UART data transmit line output from the FPGA.

|  |  |  |
| --- | --- | --- |
| Data | State | FPGA Direction |
| RS485\_DE\_i = HIGH | Data can be transmitted by UART TxD, Reception on UART RxD is disabled | FPGA to Board |
| RS485\_DE\_i = LOW | Data can be received by UART RxD, Transmission on UART TxD is disabled | Board To FPGA |

The following is the description of data flow selection implemented in the hardware design:

#### Complete Pinout and Board Arrangement Description

The following describes the complete description for the DC/DC Boards & Main Board Interface:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FPGA Line** | **Board Index** | **Package Pin** | **Direction** | **Actual Board** |
| RS485\_RxD\_1 | 0 | R19 | IN | DCDC\_1 |
| RS485\_TxD\_1 | T11 | OUT |
| RS485\_DE\_1 | T10 | OUT |
| EN\_PSU\_1\_FB | Y19 | OUT |
| PG\_PSU\_1\_FB | Y18 | IN |
| RS485\_RxD\_9 | 1 | U19 | IN | DCDC\_2 |
| RS485\_TxD\_9 | N18 | OUT |
| RS485\_DE\_9 | P19 | OUT |
| EN\_PSU\_2\_FB | W16 | OUT |
| PG\_PSU\_2\_FB | V16 | IN |
| RS485\_RxD\_2 | 2 | T12 | IN | DCDC\_5 |
| RS485\_TxD\_2 | U12 | OUT |
| RS485\_DE\_2 | N20 | OUT |
| EN\_PSU\_5\_FB | R17 | OUT |
| PG\_PSU\_5\_FB | R16 | IN |
| RS485\_RxD\_3 | 3 | V13 | IN | DCDC\_6 |
| RS485\_TxD\_3 | V12 | OUT |
| RS485\_DE\_3 | W13 | OUT |
| EN\_PSU\_6\_FB | R18 | OUT |
| PG\_PSU\_6\_FB | T17 | IN |
| RS485\_RxD\_4 | 4 | T14 | IN | DCDC\_7 |
| RS485\_TxD\_4 | T15 | OUT |
| RS485\_DE\_4 | P14 | OUT |
| EN\_PSU\_7\_FB | V18 | OUT |
| PG\_PSU\_7\_FB | V17 | IN |
| RS485\_RxD\_5 | 5 | R14 | IN | DCDC\_8 |
| RS485\_TxD\_5 | Y16 | OUT |
| RS485\_DE\_5 | Y17 | OUT |
| EN\_PSU\_8\_FB | W19 | OUT |
| PG\_PSU\_8\_FB | W18 | IN |
| RS485\_RxD\_6 | 6 | W14 | IN | DCDC\_9 |
| RS485\_TxD\_6 | Y14 | OUT |
| RS485\_DE\_6 | T16 | OUT |
| EN\_PSU\_9\_FB | P18 | OUT |
| PG\_PSU\_9\_FB | N17 | IN |
| RS485\_RxD\_7 | 7 | U17 | IN | DCDC\_10 |
| RS485\_TxD\_7 | V15 | OUT |
| RS485\_DE\_7 | W15 | OUT |
| EN\_PSU\_10\_FB | P16 | OUT |
| PG\_PSU\_10\_FB | P15 | IN |
| RS485\_RxD\_8 | 8 | U14 | IN | Main Board |
| RS485\_TxD\_8 | U15 | OUT |
| RS485\_DE\_8 | U18 | OUT |
| EN\_PFC\_FB | U20 | OUT |
| PG\_BUCK\_FB | T20 | IN |

#### Communication Protocol: DC/DC Boards and Main Board

The communication between the FPGA and the DC/DC boards is implemented as a three (3) step process:

1. FPGA sets the Data Enable (DE) High.
2. FPGA Transmits the data request frame (0xAA).
3. FPGA sets the Data Enable (DE) Low.
4. FPGA Receives the 9-byte data response from the DC/DC board.

The data request frame from the FPGA to the DC/DC board shall be implemented as follows:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data Request Frame from Master to Slave (Table 1) | | | | | | | | |
| Bytes: 1 | Bit 7  (MSB) | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 (LSB) |
| Request Frame | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

#### Received Frame Description: DC/DC Boards

The data response from the **DC/DC board** to the FPGA is implemented as follows:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data Frames from Slave to Master (Table 2) | | | | | | | | |
| Bytes:  Total: 9 | Bit 7 (MSB) | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 (LSB) |
| Byte 1: | Temp.  bit 7 | Temp.  bit 6 | Temp.  bit 5 | Temp.  bit 4 | Temp.  bit 3 | Temp.  bit 2 | Temp.  bit 1 | Temp.  bit 0 (LSB) |
| Byte 2: | Vin  bit 11 (MSB) | Vin  bit 10 | Vin  bit 9 | Vin  bit 8 | 0  0 | 0  0 | 0  0 | 0  0 |
| Byte 3: | Vin  bit 7 | Vin  bit 6 | Vin  bit 5 | Vin  bit 4 | Vin  bit 3 | Vin  bit 2 | Vin  bit 1 | Vin  bit 0 (LSB) |
| Byte 4: | Vout bit 7 | Vout bit 6 | Vout bit 5 | Vout bit 4 | Vout bit 3 | Vout bit 2 | Vout  bit 1 | Vout  bit 0 (LSB) |
| Byte 5: | Iin  bit 11 (MSB) | Iin  bit 10 | Iin  bit 9 | Iin  bit 8 | Vout  bit 11 (MSB) | Vout  bit 10 | Vout  bit 9 | Vout  bit 8 |
| Byte 6: | Iin  bit 7 | Iin  bit 6 | Iin  bit 5 | Iin  bit 4 | Iin  bit 3 | Iin  bit 2 | Iin  bit 1 | Iin  bit 0 (LSB) |
| Byte 7: | Iout bit 7 | Iout bit 6 | Iout bit 5 | Iout bit 4 | Iout bit 3 | Iout bit 2 | Iout  bit 1 | Iout  bit 0 (LSB) |
| Byte 8: | OVP | OCP | OTP | VinP | Iout  bit 11 (MSB) | Iout  bit 10 | Iout  bit 9 | Iout  bit 8 |
| Byte 9: | crc8 bit 7 (MSB) | crc8 bit 6 | crc8 bit 5 | crc8 bit 4 | crc8 bit 3 | crc8 bit 2 | crc8 bit 1 (LSB) | crc8 bit 0 |

#### Data Description – DC/DC Boards

1. Temp[7:0]: The raw 8-bit [0…255] DC/DC board temperature sample.
2. Vin[12:0]: The raw 12-bit [0…4095] DC/DC board input voltage sample.
3. Vout[12:0]: The raw 12-bit [0…4095] DC/DC board output voltage sample.
4. Iin[12:0]: The raw 12-bit [0…4095] DC/DC board input current sample.
5. Iout[12:0]: The raw 12-bit [0…4095] DC/DC board output current sample.
6. res: Reserved bit.
7. crc8[8:0]: The 8-bit CRC of the eight (8) consecutive bytes (Not including the CRC byte).
8. OVP, OCP, OTP, VinP: The 1-bit status data for each one of the corresponding events in the selected DC/DC board.

Since all the sampled values represent the raw 12-bit A/D measurements, the values shall be converted to the 16-bit signed data in accordance with the specified channel and value implemented in the design. The following table is the values of parameters required for conversion for each DC/DC board (“i” represents DC/DC board number):

|  |  |  |
| --- | --- | --- |
| Name | Value Type | Expression |
| Vin[i] | Voltage | Vin[i] = 0.56 \* Sample |
| Vout[i] | Voltage | Vout[i] = 0.28 \* Sample |
| Iin[i] | Current | Iin[i] = 0.32234 \* Sample |
| Iout[i] | Current | Iout[i] = 0.32234 \* Sample |
| Temp[i] | Temperature | Temp[i] = Sample - 60 |

#### Received Frame Description: Main board

The data response from the **Main board** to the FPGA is implemented as follows:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data Frames from Slave (Buck micro) to Master (SBC FPGA) | | | | | | | | |
| Bytes:  Total: 9 | Bit 7  (MSB) | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 (LSB) |
| Byte 1: | Temp.  bit 7 | Temp.  bit 6 | Temp.  bit 5 | Temp.  bit 4 | Temp.  bit 3 | Temp.  bit 2 | Temp.  bit 1 | Temp.  bit 0 (LSB) |
| Byte 2: | IphA  bit 11 (MSB) | IphA  bit 10 | IphA  bit 9 | IphA  bit 8 | 0 | 0 | 0 | 0 |
| Byte 3: | IphA  bit 7 | IphA  bit 6 | IphA  bit 5 | IphA  bit 4 | IphA  bit 3 | IphA  bit 2 | IphA  bit 1 | IphA  bit 0 (LSB) |
| Byte 4: | IphB bit 7 | IphB bit 6 | IphB bit 5 | IphB bit 4 | IphB bit 3 | IphB bit 2 | IphB  bit 1 | IphB  bit 0 (LSB) |
| Byte 5: | IphC  bit 11 (MSB) | IphC  bit 10 | IphC  bit 9 | IphC  bit 8 | IphB  bit 11 (MSB) | IphB  bit 10 | IphB  bit 9 | IphB  bit 8 |
| Byte 6: | IphC  bit 7 | IphC  bit 6 | IphC  bit 5 | IphC  bit 4 | IphC  bit 3 | IphC  bit 2 | IphC  bit 1 | IphC  bit 0 (LSB) |
| Byte 7: | OVP | OCP | OTP | VinP | res | res | res | Cap\_EOL |
| Byte 8: | crc8 bit 7 (MSB) | crc8 bit 6 | crc8 bit 5 | crc8 bit 4 | crc8 bit 3 | crc8 bit 2 | crc8 bit 1 (LSB) | crc8 bit 0 |
| Byte 9: | 0  bit 7 (MSB) | 0  bit 6 | 0  bit 5 | 0  bit 4 | 0  bit 3 | 0  bit 2 | 0  bit 1 (LSB) | 0  bit 0 |

#### Data Description – Main Board

1. Temp[7:0]: The raw 8-bit [0…255] DC/DC board temperature sample.
2. IphA[12:0], IphB[12:0], and IphC[12:0]: The raw 12-bit [0…4095] RMS values of phase voltage input current samples on the Main Board.
3. crc8[8:0]: The 8-bit CRC of the eight (8) consecutive bytes (Not including the CRC byte).
4. OVP, OCP, OTP, VinP: The 1-bit status data for each one of the corresponding events in the selected DC/DC board.
5. Cap\_EOL: the 1-bit status data for The Capacitor end-of-life.

The following table is the values of parameters required for conversion for Main Board:

|  |  |  |
| --- | --- | --- |
| Name | Value Type | Expression |
| IphA | Current | 0.16117 \* (Sample) |
| IphB | Current | 0.16117 \* (Sample) |
| IphC | Current | 0.16117 \* (Sample) |
| Capacitor EOL | Logic | HIGH or LOW |
| Temp | Temperature | Temp[degC] = Temp - 60 |

#### Data Collection Example – DC/DC Boards and Main Board

The following is an example flow of the data collection from all the DC/DC boards:

1. Set RS\_485\_DE\_x = HIGH.
2. Send “Data Request Character” 0xAA to the DC/DC board or Main board.
3. Set RS\_485\_DE\_x = LOW.
4. Receive and store the acquired packets (9-byte sequence).
5. Convert the values for all the boards in accordance with value conversion tables.
6. Make all the values accessible to Runtime.

#### DC/DC Boards Measurements and Log File compatibility

The following table provides information about the values that will be reported to the log file

|  |  |  |  |
| --- | --- | --- | --- |
| # | Type | Name in Log File | Name in FPGA |
| 10 | S2 | V\_OUT\_1 | DCDC1\_VOUT |
| 11 | S2 | V\_OUT\_2 | DCDC1\_VOUT |
| 16 | S2 | V\_OUT\_5 | DCDC5\_VOUT |
| 17 | S2 | V\_OUT\_6 | DCDC6\_VOUT |
| 18 | S2 | V\_OUT\_7 | DCDC7\_VOUT |
| 19 | S2 | V\_OUT\_8 | DCDC8\_VOUT |
| 20 | S2 | V\_OUT\_9 | DCDC9\_VOUT |
| 21 | S2 | V\_OUT\_10 | DCDC10\_VOUT |
| 22 | S2 | I\_OUT\_1 | DCDC1\_IOUT |
| 23 | S2 | I\_OUT\_2 | DCDC2\_IOUT |
| 28 | S2 | I\_OUT\_5 | DCDC5\_IOUT |
| 29 | S2 | I\_OUT\_6 | DCDC6\_IOUT |
| 30 | S2 | I\_OUT\_7 | DCDC7\_IOUT |
| 31 | S2 | I\_OUT\_8 | DCDC8\_IOUT |
| 32 | S2 | I\_OUT\_9 | DCDC9\_IOUT |
| 33 | S2 | I\_OUT\_10 | DCDC10\_IOUT |
| 40 | S1 | T1 | DCDC1\_Temp |
| 41 | S1 | T2 | DCDC2\_Temp |
| 42 | S1 | T3 | DCDC10\_Temp |
| 43 | S1 | T4 | Main Board Temp |
| 44 | S1 | T5 | DCDC5\_Temp |
| 45 | S1 | T6 | DCDC6\_Temp |
| 46 | S1 | T7 | DCDC7\_Temp |
| 47 | S1 | T8 | DCDC8\_Temp |
| 48 | S1 | T9 | DCDC9\_Temp |

#### DC/DC and Main Board Specific Requirements

1. The firmware shall request the data from the DC/DC boards and Main Board the values with a sample rate of at least **1KSamples/sec for each board**:
   1. The baud rate for the communication shall be 115200 Baud/sec.
   2. Number of stop bits shall be 1.
   3. Parity shall be NONE.
2. Each data transfer shall be done in accordance with RS485 Hardware flow (RS485\_DE\_x) terminal shall be driven HIGH on data request and driven LOW on data reception for each board.
3. The firmware shall provide the main application manager (Runtime) access to all the input data received from the Sensing Module with a sampling rate of at least **1KSamples/sec**. for each channel.
4. The boards in the firmware shall be mapped in accordance with their RS485 connection in the schematic diagram implemented in hardware (refer to mapping table between the FPGA UART interface and the DC/DC boards).
5. The logical lines (PG and EN) shall be the **same** as the actual board number (Only the communication is remapped).
6. The calculation constants shall be implemented in an upper application layer, so they can be modified outside the module:
   1. Each board that is can be used in firmware. For Example:
      1. DCDC\_1\_USED = ‘1’, DCDC\_2\_USED = ‘0’, etc.
   2. Explanation: Since the integration process is evolutionary, the number of boards used with RS485 communication support will rise sequentially.

### Relay Module

#### General Description

The Relay Module is responsible for enabling the AC output to be used by the customer. The module is controlled by a single logical signal as follows:

3 Phase Input

3-Phase Output

Relay Module

1-Phase Output

Relay Enable: Package Pin **K16**

#### Relay Module Specific Requirements

1. The relay module shall be turned ON or OFF by driving the line Relay\_3ph\_fpga HIGH or LOW.
2. In the case of error, the firmware shall turn OFF the relay module.
3. If the relay module is ON (Relay\_3ph\_fpga = HIGH), The firmware shall compare the following measurements:
   1. Vsns\_PH\_A\_RLY RMS and Vsns\_PH1 RMS.
   2. Vsns\_PH\_B\_RLY RMS and Vsns\_PH2 RMS.
   3. Vsns\_PH\_C\_RLY RMS and Vsns\_PH3 RMS.
   4. ~~OUT4\_sns and TBD.~~
4. If the compared values provide an absolute value if difference of more than 20[V], the Relay Module shall be turned off (Relay\_3ph\_fpga = LOW).

### **Fans Control Module**

#### General description:

The module contains three (3) fans controlled by integrated circuits (ICs), while the control and sensing are provided according to the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Package Pin | Fan Number | Description | Direction |
| FAN\_CTRL1\_fpga | K17 | 1 | Fan 1 Control Signal | OUT |
| FAN\_HALL1\_fpga | L16 | Fan 1 Speed Frequency Sense | IN |
| FAN\_CTRL2\_fpga | E18 | 2 | Fan 2 Control PWM | OUT |
| FAN\_HALL2\_fpga | D19 | Fan 2 Speed Frequency Sense | IN |
| FAN\_CTRL3\_fpga | E17 | 3 | Fan 3 Control PWM | OUT |
| FAN\_HALL3\_fpga | B19 | Fan 3 Speed Frequency Sense | IN |

#### Control Description

All the three (3) fans can be controlled separately by providing the PWM signal to the control lines. The fan speed is controlled by generating a PWM signal on FAN\_CTRLx\_fpga in accordance with the following equation:

The typical curve for the implemented fan implicates that in the frequency region of around 900[Hz] the controlled fan speed has a linear behavior, as it can be seen in the following figure:



#### Sensing Description

The fans module sensing mechanism is based on frequency measurement on the FAN\_HALLx\_fpga line. The following expression provides the measured fan speed derived from the implemented hardware:

#### Fans Control Module Specific Requirements

1. The module shall receive the speed setting from the Runtime with a period of at least **100 milliseconds**.
2. The module shall update each fan speed value in the units of RPM with a period of at least 100 milliseconds.
3. Each fan shall be able to receive an independent PWM signal through FAN\_CTRLx\_fpga lines with a fixed frequency of 900[Hz], while the pulse width of each signal defines the fan speed.
4. Each fan speed output FAN\_HALLx\_fpga shall be treated as frequency output.

### Power Status Indication Module

#### General description:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Package Pin | Direction | Description |
| P\_in\_status\_fpga | G18 | OUT | PSU input voltage indication line:  a. 115[V] AC 3-Phase Input.  b. 28[V] DC Input. |
| P\_out\_status\_fpga | J20 | OUT | PSU output voltage indication line:  a. 115[V] AC Output.  b. 28[V] DC/DC boards outputs. |

The module provides 2 status lines to the customer LED panel. The following is the description of the lines implemented:

#### Specific Requirements

* 1. P\_in\_status\_fpga shall be HIGH if the following conditions are met:
     1. 108[V] < Vsns\_PHx RMS < 118[V].
     2. 22[V] < 28V\_IN\_sns < 30[V].
  2. If one or more conditions are not met – the Pin\_status\_fpga shall be LOW.
  3. P\_out\_status\_fpga shall be HIGH if the following conditions are met:
     1. 108[V] < Vsns\_PH\_x\_RLY RMS < 118[V].
     2. 108[V] < OUT4\_sns RMS < 118[V].
  4. DC/DC boards output range is defined in accordance with the following table:

|  |  |  |
| --- | --- | --- |
| DC/DC Board | Output voltage | DC Ranges |
| 1 | 36VDC | 34.92V-37.08V |
| 2 | 30.5VDC | 30.0425V-30.9575V |
| 5 | 28VDC | 27.16V-28.84V |
| 6 | 28VDC | 27.16V-28.84V |
| 7 | 28VDC | 27.16V-28.84V |
| 8 | 28VDC | 27.16V-28.84V |
| 9 | 28VDC | 27.16V-28.84V |
| 10 | 28VDC | 27.16V-28.84V |

* 1. If one or more conditions are not met – the P\_out\_status\_fpga shall be LOW.

### External Interfaces

#### Overview

A screenshot of a computer

Description automatically generatedThe following diagram provides an overall view of the PSU external interfaces, while the blue interface connection lines are referred to as FPGA connectivity:

#### Aircraft Interface

The following table illustrates the FPGA interface with the controls from the Aircraft Interface:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Package Pin | Direction | Description |
| PowerON\_fpga | V5 | IN | Dry contact switch from the cockpit (PSU ON/OFF Switch) |
| pod\_status\_fpga | G17 | IN | The pod status is the logic to the cockpit lamp. The lamp status is measuring the actual voltage at the cockpit lamp to make sure the lamp and lamp driver are not shorted. |
| lamp\_status\_fpga | J15 | OUT | The lamp indication on the control panel in the cockpit |
| ECTCU\_INH\_fpga | H20 | OUT | TCU External Inhibit Line 1 |
| CCTCU\_INH\_fpga | J19 | OUT | TCU External Inhibit Line 2 |

#### Aircraft Interface Specific Requirements

1. The Firmware shall include an internal checking mechanism for the AC and the DC inputs, while the following is the definition:
   1. VIN\_DC\_28V\_STATUS = HIGH if 18[V] < 28V\_IN < 32[V], otherwise LOW.
   2. VIN\_AC\_115V\_STATUS = HIGH, if all the Vsns\_PHx[0…3] RMS measurements are between 95[V] and 125[V], otherwise LOW.
2. The PSU shall provide a discrete indication to a lamp on the control panel in the cockpit, the following table summarizes the required behavior of the output discrete to the LED according to the required inputs:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | | Output |
| VIN\_DC\_28V\_STATUS | VIN\_AC\_115V\_STATUS | PowerON\_fpga | MIU Ethernet Communication | lamp\_status\_fpga |
| LOW | LOW | N/A | N/A | LOW |
| HIGH | LOW | N/A | N/A | Square Wave 1[Hz] |
| LOW | HIGH | N/A | N/A | LOW |
| HIGH | HIGH | LOW | N/A | HIGH |
| HIGH | HIGH | HIGH | No communication | Square Wave 4[Hz] |
| HIGH | HIGH | HIGH | Communication established | LOW |

1. The firmware shall periodically check the pod\_status\_fpga each 10 milliseconds of the Runtime.
2. ~~If pod\_status\_fpga state differs from lamp\_status\_fpga, the system shall TBD.~~

### MIU Interface

#### General Description

This module integrates a critical component of the system. The MIU interface is consists of the following:

MIU

(Customer)

Reset\_Out\_fpga

MIU Interface Module

Shutdown\_Out\_fpga

EShutdown\_Out\_fpga

Spare\_Out\_fpga

Ethernet PHY Module

MDC 100MbE

The module consists of 4 logical lines, that can be driven HIGH or LOW and an 100Mbit/s Ethernet interface (MDC). The following is the description of the module lines:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Package Pin | Direction | Description |
| Reset\_Out\_fpga | H15 | OUT | Reset request output to MIU. The terminal provides reset request signals to MIU according to the Runtime Specification. |
| Shutdown\_out\_fpga | G15 | OUT | Standard shutdown indication to MIU. The terminal provides an indication of a **standard shutdown** procedure to the MIU, according to Runtime Specification. |
| EShutdown\_Out\_fpga | K14 | OUT | Emergency shutdown indication to MIU. The terminal provides an indication of an **emergency shutdown** procedure to the MIU, according to Runtime Specification. |
| Spare\_Out\_fpga | J14 | OUT | Spare logical output to the MIU, that was requested by customer and its operation can be implemented in the future. |
| MDC 100MbE | PS | I/O | Standard 100Mbit/s Ethernet communication interface with the MIU. The Ethernet communication enables the control over the data stream of the PSU, the requests from the PSU and the operation definitions according to the Runtime specification. |

#### Ethernet Interface

The PSU implements an Ethernet communication interface with the MIU interface, while the PSU shall be able to maintain continuous UDP connections with responsiveness to the MIU interface commands. The PSU IP address shall be 192.168.1.30 and the specific requirements are described in the Customer IRS document: “**IRS MIU-PSU-V1**”.

# Runtime Operation Specifications

## Start-Up Sequence

### Overview

The FPGA shall be able to turn the entire PSU system ON or OFF in response to:

* 1. Dry contact switch in the cockpit (Aircraft Interface) position.
  2. Emergency Shutdown Event.

The following is the states defined as a source for initialization of the PSU system

Short = ON (POWERON\_FPGA = HIGH).

Open = OFF (POWERON\_FPGA = LOW).

### System Power ON Sequence Requirements

1. If POWER\_ON\_FPGA = HIGH:
   1. 50ms Switch debounce mechanism.
   2. Set the Main Board enable: EN\_PFC\_FB = HIGH and start an elapsed time measurement.
   3. Turn on the Fans module, by providing a PWM signal to all the three (3) fans.
   4. Start measurement of the fan speed.
   5. Start an elapsed time measurement.
   6. Measure the line PG\_BUCK\_FB.
   7. If more than 4 seconds elapsed and PG\_BUCK\_FB was driven LOW - the system shall enter emergency shutdown state.
   8. If fans are at the correct speed and PG\_BUCK\_FB is driven HIGH, the following lines shall be set as follows:
      1. EN\_PSU\_1\_FB = HIGH
      2. EN\_PSU\_2\_FB = HIGH
      3. EN\_PSU\_5\_FB = HIGH
      4. EN\_PSU\_6\_FB = HIGH
      5. EN\_PSU\_7\_FB = HIGH
      6. EN\_PSU\_8\_FB = HIGH
      7. EN\_PSU\_9\_FB = HIGH
      8. EN\_PSU\_10\_FB = HIGH
      9. Relay\_3ph\_fpga = HIGH
   9. The System Power ON Sequence is treated as finished, if the following conditions are met:
      1. The power good (PG) logical lines are driven HIGH by all the used DC/DC boards and Main Board in accordance with the following:
         1. PG\_PSU\_1\_FB = HIGH
         2. PG\_PSU\_2\_FB = HIGH
         3. PG\_PSU\_5\_FB = HIGH
         4. PG\_PSU\_6\_FB = HIGH
         5. PG\_PSU\_7\_FB = HIGH
         6. PG\_PSU\_8\_FB = HIGH
         7. PG\_PSU\_9\_FB = HIGH
         8. PG\_PSU\_10\_FB = HIGH
      2. The Fan speed is at least at 80[%] of the setpoint value.
      3. The Relay Module measurements are valid (Section 2.2.3).
   10. If System Power ON Sequence is finished and successful:
       1. Wait for 1 millisecond.
       2. Enable the TCUs:
          1. ECTCU\_INH\_fpga = HIGH
          2. CCTCU\_INH\_fpga = HIGH

## System Reset

### Specific Requirements

The following sequence describes the system reset performed by the Aircraft Interface switch:

1. The POWERON\_FPGA = LOW detected.
2. Start elapsed time measurement.
3. POWERON\_FPGA = HIGH detected.
4. If the POWERON\_FPGA = HIGH was detected within the time frame of

**50[milliseconds] < Elapsed Time < 6[seconds]:**

* + 1. The PSU shall not disable the output voltages.
    2. The PSU shall generate a reset sequence by driving the MIU interface line reset\_out\_fpga line LOW for 5ms ± 10%.

1. The following diagram describes the system reset implementation:

## **Shutdown Sequences**

### Overview

System shutdown implementation consists of two different sequences: standard shutdown and emergency shutdown.

#### Standard Shutdown

If the POWERON\_FPGA = LOW was detected for more than 6 seconds, the PSU shall enter a Power OFF sequence. The diagram below illustrates the logic and timing diagram:



#### Emergency Shutdown

Emergency shutdown shall be initiated, if one or more of the following conditions are met:

1. Power Good signal (PG) from one or more DC/DC boards is driven LOW.
2. Power Good signal (PG) from the Main board is driven LOW.
3. Relay module is turned ON and the measurements of Relay Module are valid (See section 2.2.3).
4. Sensing module is not communicating correctly.
5. DC/DC boards or Main board do not communicate correctly.
6. Fan speed is turned ON and its speed is below than an 80% of the setpoint speed.
7. Input OVP/UVP condition.
8. Output OVP condition.

### Standard Shutdown Specific Requirements

1. If the System Shutdown procedure was initiated, the POWERON\_FPGA line shall be ignored for 10 seconds after the PSU finishes the procedure (effectively 26 seconds after the switch has been turned off since the first 6 seconds are ignored and then an additional 10 seconds are needed to power down).
2. Standard shutdown procedure shall be performed when POWERON\_FPGA was driven LOW for a time above 6[seconds] or after an OTP event.
3. shutdown\_out\_fpga shall be driven HIGH for 10[seconds].
4. After the step (C), the shutdown\_out\_fpga shall be driven LOW.
5. All the DC/DC boards shall be disabled by driving the following lines LOW:
   1. EN\_PSU\_1\_FB = LOW
   2. EN\_PSU\_2\_FB = LOW
   3. EN\_PSU\_5\_FB = LOW
   4. EN\_PSU\_6\_FB = LOW
   5. EN\_PSU\_7\_FB = LOW
   6. EN\_PSU\_8\_FB = LOW
   7. EN\_PSU\_9\_FB = LOW
   8. EN\_PSU\_10\_FB = LOW
6. The relay module shall be disabled: Relay\_3ph\_fpga = LOW
7. Main Board shall be disabled: EN\_PFC\_FB = LOW.
8. CCTCU = LOW and ECTCU = LOW.
9. 1 minute after step (H), the fans control module shall be disabled.

### Emergency Shutdown Specific Requirements

1. If the Emergency Shutdown Procedure was initiated, the **EShutdown\_Out\_fpga line shall be driven HIGH for 51[milliseconds].**
2. The PSU Shall disable the power supplies according to the following hold-up sequence:
   1. Immediately:
      1. EN\_PSU\_1\_FB = LOW.
      2. EN\_PSU\_9\_FB = LOW.
      3. Relay\_3ph\_fpga = LOW.
   2. 10[milliseconds] after EShutdown\_Out\_fpga = HIGH:
      1. EN\_PSU\_2\_FB = LOW.
      2. EN\_PSU\_6\_FB = LOW.
      3. EN\_PSU\_8\_FB = LOW.
      4. EN\_PSU\_10\_FB = LOW.
   3. 20[milliseconds] after EShutdown\_Out\_fpga = HIGH:
      1. EN\_PSU\_7\_FB = LOW.
   4. 50[milliseconds] after EShutdown\_Out\_fpga = HIGH:
      1. EN\_PSU\_5\_FB = LOW.
      2. EN\_PFC\_FB = LOW.
      3. ECTCU\_INH\_fpga = LOW.
      4. CCTCU\_INH\_fpga = LOW.

## **Protection Specifications**

### Over Voltage Protection (OVP)

#### Overview

|  |  |  |
| --- | --- | --- |
| Voltage | O.V. Trip Point | Detection Response Time |
| 115VAC | 125VAC | 100msec |
| 28VDC | 35VDC | 2ms |
| 30.5VDC | 31.5VDC | 2ms |
| 36VDC | 45 VDC | 2ms |

The following is the Over Voltage Protection trip point specifications from the customer SPEC, generalizing the input and output voltages from the PSU, that are continuously measured:

#### Over Voltage Protection Specific Requirements

1. The following is the Over Voltage Protection **Input** trip point specifications, that shall be implemented:

|  |  |  |
| --- | --- | --- |
| Measured Lines | Voltage Range | Tripping Time  [milliseconds] |
| Vsns\_PH\_A\_RLY RMS  Vsns\_PH\_B\_RLY RMS  Vsns\_PH\_C\_RLY RMS | Vsns\_PHx RMS > 125[V] | 100 |
| Vsns\_28V\_IN | Vsns\_28V\_IN > 35[V] | 2 |

1. If the Input OVP event occurs, the system shall initiate the Standard Shutdown sequence.
2. If the system Input voltages are within the specified range for at least 1 second, the system shall initiate the Power On sequence.
3. The following is the Over Voltage Protection **Output** trip point specifications, that shall be implemented:

|  |  |  |
| --- | --- | --- |
| Measured Lines | Voltage Range | Tripping Time  [milliseconds] |
| DCDC1\_VOUT | DCDC1\_VOUT > 45[V] | 2 |
| DCDC2\_VOUT | DCDC2\_VOUT > 31.5[V] |
| DCDC5\_VOUT  DCDC6\_VOUT  DCDC7\_VOUT  DCDC8\_VOUT  DCDC9\_VOUT  DCDC10\_VOUT | DCDCx\_VOUT > 35[V] |

1. If the output OVP event occurs, the system shall initiate the Emergency Shutdown procedure.
2. If the dry-contact switch (PowerOn\_fpga) line was driven LOW and driven HIGH back again, the system shall initiate the Power On sequence.

### Under Voltage Protection (UVP)

#### Overview

All main inputs shall be under-voltage protected. If any input voltage is in an under-voltage state, for more than the rated duration, the PSU shall handle the operation accordingly.

#### Under Voltage Protection AC Specific Requirements

1. All three phases of input AC voltages shall be under-voltage protected according to the following table:

|  |  |  |
| --- | --- | --- |
| Measured Lines | Voltage Range | Tripping Time  [milliseconds] |
| Vsns\_PH1 RMS  Vsns\_PH2 RMS  Vsns\_PH3 RMS | Vsns\_PHx RMS < 90[V] | 0 (Immediate UVP Event) |
| 90[V] < Vsns\_PHx RMS < 95[V] | 500 |
| Vsns\_PHx RMS > 95[V] | Normal Operation |

1. the PSU shall issue Emergency Shutdown procedure, if UVP event occurs.
2. After the voltage returns to the nominal value for at least 1 second, the PSU shall initiate the Power ON Procedure in accordance with PowerOn\_fpga line status.

#### Under Voltage Protection DC Specific Requirements

1. The 28V input voltage shall be under-voltage protected according to the following table:

|  |  |  |
| --- | --- | --- |
| Measured Line | Voltage Range | Tripping Time  [milliseconds] |
| Vsns\_28V\_IN | Vsns\_28V\_IN < 17[V] | 0 (Immediate UVP Event) |
| 17[V] < Vsns\_28V\_IN < 18[V] | 500 |
| Vsns\_28V\_IN > 18[V] | Normal Operation |

### Over Temperature Protection (OTP)

#### Overview

The PSU shall implement a protection mechanism to protect itself from the possibility of overheating.

#### Over Temperature Protection Specific Requirements

#### The firmware shall detect the OTP event by continuous measurements of DC/DC boards and Main Board temperature. The following table provides the OTP event values for each board separately:

|  |  |  |
| --- | --- | --- |
| **Board number** | **Over Temperature Protection** | **Turn on temperature** |
| Main board | 100°C | 80°C |
| LLC | 105°C | 85°C |
| DCDC1 | 110°C | 90°C |
| DCDC2 | 110°C | 90°C |
| DCDC5 | 110°C | 90°C |
| DCDC6 | 110°C | 90°C |
| DCDC7 | 110°C | 90°C |
| DCDC8 | 110°C | 90°C |
| DCDC9 | 110°C | 90°C |
| DCDC10 | 110°C | 90°C |

#### If OTP event occurs, The PSU shall initiate the Emergency Shutdown procedure, while the order of operations is as follows:

#### The PSU shall notify the MIU via communication interface.

#### Shutdown\_out\_fpga line must be driven HIGH.

#### After the Shutdown\_out\_fpga line was driven HIGH, the PSU shall wait 10 seconds, so the MIU can finish its own shutdown sequence.

#### The system shall initiate Standard Shutdown procedure.

#### If the temperature drops back below a safe level (20[degrees] below OTP value), the PSU shall initiate a start-up sequence.

1. The temperatures shall be sampled from DC/DC boards and Main Board according to DC/DC Boards specifications: the resolution of the temperature is the integer value in degrees [0…255], while the temperature is calculated as follows:

# Elapsed time FLASH memory communication

General description:

Each of the PSU shall provide a digital Elapsed Time Indication with resolution of hours. This information shall be provided along with the serial number indication. The indication shall be accessible by MIU via communication and recorded in the PSU internal log file.

The elapsed Time indication (ETI) will be saved in dedicated memory, and it will be written to the Flash Memory in parallel.

The elapsed Time indication (ETI) can be updated / restored via dedicated pins on PSU connector.

The time indication can only be reset separately as part of the PSU manufacturing process (can be in final stages, during check-out). The time reset shall not be accessible when the PSU box is closed, only when it is opened and connected via specific port to test equipment at developer facility.

The ETI shall be accessible / data shall be restored in case of most types of PSU malfunctions.

The FPGA has an A/D reading of the 12V\_Redundant supply that creates all the aux power for the SBC Board. When this supply drops below 8V for more than 1ms (level and duration may be modified according to testing), then the FPGA should immediately stop writing to the eMMC to prevent the possibility of trying to write to the eMMC while the power is going down. This test should be done frequently enough that there is no possibility for the SBC power to go down in the middle of a write operation to the eMMC (e.g. every 10ms). Then the file should be closed.  
Note that when the JTAG is programming the eMMC, the 12V\_Redundant supply may be as low as 5V.

12v \_redundant is sampling by internal A2D at VP\_0 (VN\_0 at GND). when the result shows that this voltage is low, meaning it is going down. We need to avoid writing to FLASH. The input to the VP0 is 12v \_redundant divided by 23.1 🡪 8volt is 0.346v

   12v is 0.519v



# 

# Communication Interface

The PSU shall have an Ethernet communication interface with MIU, which shall provide the following capabilities (detailed PSU-MIU communication protocol shall be defined in a separate IRS document, PSU IP address shall be 192.168.1.60):

1. Providing periodic statuses and telemetry (@100Hz) with time tag, including:
   * Input/output voltages (for each phase separately and for DC voltage)
   * Current consumptions (also for each phase & DC voltage, not including Neutral)
   * Total input (AC+DC) power consumption
   * Internal temperature of every internal card
   * Missing phase or missing neutral indication
   * PSU Fans speed for each of the 3 fans
   * Software/firmware version (format shall be “Major.Minor.Build.Hotfix”)
   * ETI (Elapsed Time Indication)
   * PSU serial number
   * Additional bit statuses such as:
     1. Reset or Shutdown indications.
     2. Input voltage OK indication.
     3. TCU ON/OFF status
     4. MIU communication OK/NOK
     5. Overvoltage & overcurrent indications
     6. Capacitor end-of-life indications

Since status changes can occur intermittently, in order not to miss the occurrence, the faulty notification shall be latched until sent to the MIU via communication channel. Only after the faulty notification was sent to MIU, it will be cleared.

1. Log file downloading from PSU (using SFTP) via dedicated command.
2. Control of TCUs ON/OFF by MIU via PSU
3. MIU shall provide the time at power-up for log file time stamp synchronization. The purpose is to correlate between events in the MIU and events in the PSU, during a flight.

# Log File functionality

## General description:

1. The PSU shall gather telemetry and status data at a high rate and store it internally. This is required to capture any transient anomalies in case they occur. Detailed logfile parameters shall be defined in the IRS. This data includes (but not limited to):

* Input voltages & currents per line (no need to record neutral line input)
* Output voltages & currents per line.
* Power consumption (VA)
* Date & time (synchronized to MIU time stamp)
* PSU Fans speed for each of the 3 fans
* Internal temperature of every internal card reading
* Logic statuses (incl. missing phase indications)
* Software/firmware version
* ETI
* Serial number
* ON/OFF signal occurrences
* Reset/power down indications.
* All indications are also sent periodically to the MIU via communication channel.

1. The log file shall be cyclic and capable of recording at a frequency of 1KHz with a minimum memory allocation of 10GBytes.

## PSU Logfile handling - general rules

* Logfile shall be recorded in the PSU at 1KHz rate.
* A Logfile is defined as a sequence of recorded entries (entry format detailed in par 6.2 below).
* Every time input minimum voltage is supplied to the PSU (see PSU PIDS paragraph 3.1.10 for minimum voltage conditions for log operation), a new log file will be started with the following name:
  + First the name of the file will be based on the starting date of 01/01/1970 and a sequential cyclic counter. E.g. **1970\_01\_01\_00\_00\_00\_PSU\_0000.log**
  + Once receiving the actual time from the MIU, the file name will be updated according to the actual date and time, using the following format: YYYY\_MM\_DD\_HH\_MM\_SS\_PSU\_XXXX.log (YYYY-Year, MM-Month, DD-Day, HH-Hour, MM-Minutes, SS-Seconds, XXXX – stays as before with the sequential counter).
  + the last 4 digits of the file name are a decimal number, but always 4 digits (for example 0000, 0001, etc.). Also, the counter is at the start of the name.
  + In case no communication with the MIU is established, then the file name shall remain with the date of 1970 and the sequential counter.
  + The suggested solution of saving in the eMMC the last number as a file. If eMMC card is replaced, then the counter is reset.
  + Another option to consider – instead of storing the counter in the eMMC, upon power up – look for the last logfile name counter and continue from there.
  + If the MIU commands the PSU to erase logfiles, then the counter should start from 0000 again.
* The log file will be recorded if the 12V redundant supply remains above 8V. When going below 8V, the log file will be closed, and recording stopped.
* Log files will be recorded if there is memory capacity for the recording. When the memory is completely full, the recording will stop.
* The logfile shall be downloaded using SFTP client.
* The log file shall be cyclic and capable of recording at a frequency of 1KHz with a memory allocation of 10GBytes.
* eMMC 64GB
* FPGA built-in controller (50MHz)
* When voltage is applied to the system, we assume that we start a new log file and do not erase any existing log files.

# 29. FPGA Measurements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Measurement | Source | Details | net name at source schematic | RS485 protocol |
| VDC\_IN | DCDC1 RS485 | DCDC1 primary micro measures VDC\_IN and reports to DCDC1 secondary micro. Secondary micro reports VDC\_IN to FPGA through RS485. | Vin\_sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Vin)  1LSB 0.028V |
| VAC\_IN\_PH\_A | A/D on SBC  (HV\_ADC CH1) | SBC A/D U48 CH1 represents instantaneous VAC\_IN\_PH\_A. Must calculate RMS. | Vsns\_PH1 |  |
| VAC\_IN\_PH\_B | A/D on SBC  (HV\_ADC CH2) | SBC A/D U48 CH2 represents instantaneous VAC\_IN\_PH\_B. Must calculate RMS. | Vsns\_PH2 |  |
| VAC\_IN\_PH\_C | A/D on SBC  (HV\_ADC CH3) | SBC A/D U48 CH3 represents instantaneous VAC\_IN\_PH\_C. Must calculate RMS. | Vsns\_PH3 |  |
| I\_DC\_IN | A/D on SBC  (I\_sns\_ADC CH0) | SBC A/D U54 CH0 represents I\_DC\_IN | DC\_PWR\_I\_sns |  |
| I\_AC\_IN\_PH\_A | A/D on SBC  (I\_sns\_ADC CH1) | SBC A/D U54 CH1 represents instantaneous phase A input current. Must calculate RMS. | PH1\_I\_sns |  |
| I\_AC\_IN\_PH\_B | A/D on SBC  (I\_sns\_ADC CH2) | SBC A/D U54 CH2 represents instantaneous phase B input current. Must calculate RMS. | PH2\_I\_sns |  |
| I\_AC\_IN\_PH\_C | A/D on SBC  (I\_sns\_ADC CH3) | SBC A/D U54 CH3 represents instantaneous phase C input current. Must calculate RMS. | PH3\_I\_sns |  |
| V\_OUT\_2 | 50WDCDC RS485 | 50WDCDC secondary micro measures V\_OUT\_2 and reports to FPGA through RS485. | Vout\_LLC\_sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Vout)  1LSB 0.014V |
| V\_OUT\_3\_ph1 | A/D on SBC  (HV\_ADC CH7) | SBC A/D U48 CH7 represents instantaneous V\_OUT\_3\_ph1. Must calculate RMS. | Vsns\_PH\_A\_RLY |  |
| V\_OUT\_3\_ph2 | A/D on SBC  (HV\_ADC CH6) | SBC A/D U48 CH6 represents instantaneous V\_OUT\_3\_ph2. Must calculate RMS. | Vsns\_PH\_B\_RLY |  |
| V\_OUT\_3\_ph3 | A/D on SBC  (HV\_ADC CH5) | SBC A/D U48 CH5 represents instantaneous V\_OUT\_3\_ph3. Must calculate RMS. | Vsns\_PH\_C\_RLY |  |
| V\_OUT\_4 | A/D on SBC  (HV\_ADC CH0) | SBC A/D U48 CH0 represents instantaneous V\_OUT\_4. Must calculate RMS. | OUT4\_sns |  |
| V\_OUT\_x  x = 1,5-7,9,10 | DCDCx RS485 | DCDCx secondary micro measures V\_OUT\_x and reports to FPGA through RS485. | Vout\_LLC\_sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Vout)  1LSB 0.014V |
| V\_OUT\_8 | LLC RS485 | LLC secondary micro measures output voltage and reports to FPGA through RS485. | Vout\_LLC\_sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Vout)  1LSB 8.86mV |
| I\_OUT\_2 | 50WDCDC RS485 | 50WDCDC secondary micro measures I\_OUT\_2 and reports to FPGA through RS485. | LLC\_Iout\_Sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Iout)  1LSB 4mV |
| I\_OUT\_3\_ph1 | Calculation  I\_OUT\_3\_ph1(RMS)=  PH1\_I\_sns-OUT4\_Isns-I\_PFC\_PH\_A | See note 1. Subtract instantaneous current I\_OUT\_4 and I\_PFC\_PH\_A from I\_AC\_IN\_PH\_A and calculate rms value. | Calculation example: FPGA measures Vph\_A(t), FPGA calculates Vph\_A\_rms, Main board reports Ipfc\_A\_rms, FPGA assumes ipfc\_A=(Vph\_A(\*Ipfc\_A\_rms)/(Vph\_A\_rms) iph\_A-ipfc\_A-iout4=iout3\_phA calculate rms (iout\_ph\_A) for phase B, C, don't include iout4. |  |
| I\_OUT\_3\_ph2 | Calculation  I\_OUT\_3\_ph2(RMS)=  PH2\_I\_sns-I\_PFC\_PH\_B | See note 1. Subtract instantaneous current I\_PFC\_PH\_B from I\_AC\_IN\_PH\_B and calculate rms value. |  |
| I\_OUT\_3\_ph3 | Calculation  I\_OUT\_3\_ph3(RMS)=  PH3\_I\_sns-I\_PFC\_PH\_C | See note 1. Subtract instantaneous current I\_PFC\_PH\_C from I\_AC\_IN\_PH\_C and calculate rms value. |  |
| I\_OUT\_4 | A/D on SBC  (HV\_ADC CH4) | SBC A/D U48 CH4 represents instantaneous current through Output 4. Must calculate RMS. | OUT4\_Isns |  |
| I\_OUT\_x  x = 1,5-7,9,10 | DCDCx RS485 | DCDCx secondary micro measures I\_OUT\_x and reports to FPGA through RS485. | LLC\_Iout\_Sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Iout)  1LSB 0.00805V |
| I\_OUT\_8 | LLC RS485 | LLC secondary micro measures I\_OUT\_x and reports to FPGA through RS485. | Iout\_LLC\_sns | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Iout)  1LSB 0.0032V |
| AC\_Power | Calculation  Average of Vsns\_PH1(line3) \* PH1\_I\_sns (line 7) + Vsns\_PH2(line4) \* PH2\_I\_sns(line 8) + Vsns\_PH3(line 5)\*PH3\_I\_sns(line 9) | Running average of VAC\_IN\_PH\_A\* I\_AC\_IN\_PH\_A + VAC\_IN\_PH\_B\* I\_AC\_IN\_PH\_B + VAC\_IN\_PH\_C\* I\_AC\_IN\_PH\_C | rolling average 400Hz 3 cycles 6 half cycles.  adjust according to line frequency.  refer power calculations. |  |
| Fan\_Speed\_1 | Hall sensor from Fan1 | Frequency of hall sensor3 same as fan speed. Change Hz to RPM. | FAN\_HALL1\_FPGA | Refer to FAN control in FPGA specifications file |
| Fan\_Speed\_2 | Hall sensor from Fan2 | Frequency of hall sensor2 same as fan2 speed. Change Hz to RPM. | FAN\_HALL2\_FPGA |
| Fan\_Speed\_3 | Hall sensor from Fan3 | Frequency of hall sensor3 same as fan3 speed. Change Hz to RPM. | FAN\_HALL3\_FPGA |
| T\_x  x = 1,5-7,9,10 | DCDCx RS485 | Temperature of DCDCx reported from DCDCx secondary through RS485. See note 2. | NTC\_b  (Temperature from micro) | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Temp.) |
| T\_2 | 50WDCDC RS485 | Temperature of 50WDCDC reported from 50WDCDC secondary through RS485. See note 2. | NTC\_b  (Temperature from micro) | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Temp.) |
| T\_8 | LLC RS485 | The temperature of LLC reported from LLC secondary through RS485. See note 2 | temp\_sns\_LLC\_SR\_filt | refer: Protocol\_RS485\_  DC-DC\_Control\_  SEC.docx  (Temp.) |
| T\_main | HV\_BUCK RS485 | Temperature of Main Board reported from Buck through RS485. See note 2 | main\_board\_temp\_sns |  |
| DC\_IN\_Status | Calculation | If 28VDC from Aircraft is within correct range, then logical 1; otherwise, logical 0. See note 3. |  | check if VDC\_IN (line 2) is in range 18-32VDC.  from DCDC1 RANGE 18-32VDC RS485 every 1msec.  if v<18 for 0.5sec, shutdown. If v>=20, turn on |
| AC\_IN\_Status | Calculation | If RMS values of VAC\_IN\_PH\_A, B, and C are in correct range, then logical 1; otherwise, logical 0. See note 4. |  | check if VAC\_IN\_PH\_A,  B, C (lines 3,4,5) are in the range 95-125VAC.  if v<95 for 0.5sec, shutdown. If v>=95, turn on |
| Power\_Out\_Status | Calculation | If all modules produce PG=1, then 1, otherwise 0. |  | check if PG\_PSU\_1, PG\_PSU\_2, PG\_PSU\_5, PG\_PSU\_6, PG\_PSU\_7, PG\_PSU\_8, PG\_PSU\_9, PG\_PSU\_10 and PG\_BUCK are all equal 1 |
| OUTx\_OC  x = 1,5-7,9,10 | DCDCx RS485 | If PG goes low on any power supply unit, then the secondary of that unit must send status information to the FPGA via RS485 indicating if the shutdown occurred due to output overcurrent, output overvoltage, overtemperature, input voltage out-of-range, or another reason. |  | 4 BIT STATUS:  (Refer protocol  \_RS485\_DC-DC\_Control\_SEC  .docx)  VinP Input voltage protection - Byte 8, bit 4 OTP -  Byte 8, bit 5 OCP -  Byte 8, bit 6 OVP -  Byte 8, bit 7 pc number: 00 main board 01 DCDC1 02 DCDC50W 05 DCDC5 06 DCDC6 07 DCDC7 08 LLC 09 DCDC9 |
| OUT2\_OC | DCDC2 RS485 |  |
| OUT8\_OC | LLC RS485 |  |
| DC\_IN\_OV | DCDC1 RS485 |  |
| OUTx OV  x = 1,5-7,9,10 | DCDCx RS485 |  |
| OUT2 OV | DCDC2 RS485 |  |
| OUT8 OV | LLC RS485 |  |
| DC\_IN\_UV | DCDC1 RS485 |  |
| AC\_IN\_UV | Calculation | See note 4. |  |
| PHx\_Status  x = 1, 2, 3 | Calculation | See note 4. Report according to voltage on each phase. |  | RANGE 95-125VAC |
| Neutral\_Status | Calculation | Normally a logical 1. Average of instantaneous sum of VAC\_IN\_PH\_A, VAC\_IN\_PH\_B, and VAC\_IN\_PH\_C should always be close to 0. If departing more than some threshold amount from 0, that indicates that the neutral fuse opened in which case the logical status will change to 0. |  | check if the sum is close to zero [-10,10] |
| OVER\_TEMP\_Status | RS485 all PSUs | Logical 1 unless a PSU reports an overtemperature through the RS485 bus. |  | OTP |
| Capacitor\_end\_of\_life | Buck RS485 | See note 5. |  | file: "protocol RS485\_Main\_brd" ---> Byte 7: cap EOL (Must send this value from previous operation since it is only updated at shutdown.) |

TABLE I – Pinout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pin name | PS/PL | Dir I/O/IO | FW/SW control | Description | pull down/up |
| POWERON\_FPGA | PL | in | FW | in from system to FPGA to turn on the power supplies | down |
| Relay\_3ph\_fpga | PL | out | FW | RELAY 3PH OPERATE, part of system turns on |  |
| P\_in\_status\_fpga | PL | out | FW | indication for 28v, 115v, power input measured by HV\_ADC using Mio comm.  OK is when all input voltages are within range:  115VAC range: 108V-118V 28VDC range: 22-30V |  |
| P\_out\_status\_fpga | PL | out | FW | after all is good, set this to on.  OK is when all output voltages are within range indicated by receive PG from all unit |  |
| RESET\_OUT\_FPGA | PL | out | FW | normally high. 5ms low following power on\_fpga low for t<6s and t>50ms.  send information to the system |  |
| SHUTDOWN\_OUT\_FPGA | PL | out | FW | After power supplies are down send this indication to the system. |  |
| POD\_STATUS\_FPGA | PL | out | FW | After all is good, set this on.  Inputs: PG [9:1] &PG\_BUCK are all HIGH |  |
| lamp\_status\_fpga | PL | in |  | in indication of problem with lamps | down |
| EN\_PSU\_1\_FB | PL | out | FW | Enable power supply 1, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_1\_FB | PL | in | FW | FB from power supply 1, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_1 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_1 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_1 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_2\_FB | PL | out | FW | Enable power supply 2, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_2\_FB | PL | in | FW | FB from power supply 2, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_2 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_2 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_2 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
|  |  |  |  | The PFC\_EN signal operates the PFC/Buck |  |
| PG\_BUCK\_FB | PL | in | FW | power good of BUCK supply, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_3 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_3 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_3 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_10\_FB | PL | out | FW | Enable power supply 10, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_10\_FB | PL | in | FW | FB from power supply 10, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_4 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_4 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_4 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_5\_FB | PL | out | FW | Enable power supply 5, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_5\_FB | PL | in | FW | FB from power supply 5, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_5 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_5 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_5 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_6\_FB | PL | out | FW | Enable power supply 6, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_6\_FB | PL | in | FW | FB from power supply 6, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_6 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_6 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_6 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_7\_FB | PL | out | FW | Enable power supply 7, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_7\_FB | PL | in | FW | FB from power supply 7, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_7 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_7 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_7 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_8\_FB | PL | out | FW | Enable power supply 8, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_8\_FB | PL | in | FW | FB from power supply 8, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_8 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_8 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_8 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| EN\_PSU\_9\_FB | PL | out | FW | Enable power supply 9, part of power on sequence (POWERON\_FPGA) |  |
| PG\_PSU\_9\_FB | PL | in | FW | FB from power supply 9, part of power on sequence (POWERON\_FPGA) | down |
| RS485\_DE\_9 | PL | out | SW | Active-high driver data out enable, SW will ask for info from microcontroller and get answers to log them. Drive to high whenever writing to the UART and drive it to low whenever reading from the UART. |  |
| RS485\_TXD\_9 | PL | out | SW | Mosi, SW will ask for info from microcontroller and get answers to log them |  |
| RS485\_RXD\_9 | PL | in | SW | Miso, SW will ask for info from microcontroller and get answers to log them | down |
| ECTCU\_INH\_fpga | PL | out | FW | ECTCU OFF (0) or ON (1) indication, part of the power good sequence (POWERON\_FPGA) and shutdown sequences  tcu identification 0x00 |  |
| CCTCU\_INH\_fpga | PL | out | FW | CC\_Inhibit CCTCU OFF (0) or ON (1) indication part of the power good sequence (POWERON\_FPGA) and shutdown sequences. tcu identification 0x01 |  |
| HV\_ADC\_CS\_FPGA | PL | out | SW | spi for current sense of in supply, SW will do some calculation based on this in |  |
| HV\_ADC\_SCLK\_FPGA | PL | out | SW | spi for current sense of in supply, SW will do some calculation based on this in |  |
| HV\_ADC\_SDI\_FPGA | PL | out | SW | out of A/D (Mosi), spi for current sense of in supply, SW will do some calculation based on this in |  |
| HV\_ADC\_SDO\_FPGA | PL | in | SW | input of A/D (Miso), spi for current sense of in supply, SW will do some calculation based on this in | down |
| I\_sns\_ADC\_CS\_fpga | PL | out | SW | SPI for current sense, SW will do some calculation based on this in |  |
| I\_sns\_ADC\_SCLK\_fpga | PL | out | SW | SPI for current sense, SW will do some calculation based on this in |  |
| I\_sns\_ADC\_SDI\_fpga | PL | out | SW | SPI for current sense, SW will do some calculation based on this in |  |
| I\_sns\_ADC\_SDO\_fpga | PL | in | SW | SPI for current sense, SW will do some calculation based on this in | down |
| ZCR\_sns\_ADC\_CS\_fpga | PL | out | SW | SPI for reference current sense, SW will do some calculation based on this in |  |
| ZCR\_sns\_ADC\_SCLK\_fpga | PL | out | SW | SPI for reference current sense, SW will do some calculation based on this in |  |
| ZCR\_sns\_ADC\_SDI\_fpga | PL | out | SW | SPI for reference current sense, SW will do some calculation based on this in |  |
| ZCR\_sns\_ADC\_SDO\_fpga | PL | in | SW | SPI for reference current sense, SW will do some calculation based on this in | down |
| QSPI\_1\_CS# | PS |  |  | program flash |  |
| QSPI\_DQ0 | PS |  |  | program flash |  |
| QSPI\_DQ1 | PS |  |  | program flash |  |
| QSPI\_DQ2 | PS |  |  | program flash |  |
| QSPI\_DQ3 | PS |  |  | program flash |  |
| QSPI\_1\_CLK | PS |  |  | program flash |  |
| FAN\_HALL1\_fpga | PL | in | FW | in square wave, measure the frequency of hall sensor1 same as fan speed. Change the FAN\_CTRL to bring to correct speed. Change Hz to RPM. | down |
| FAN\_PG1\_fpga | PL | in | SW | PG sent to the log file requires both PG for the fan voltage and fan speed OK. | down |
| FAN\_CTRL1\_fpga | PL | out | FW | in square wave, measure the frequency. Change the FAN\_CTRL to bring to correct speed. |  |
| FAN\_EN1\_fpga | PL | out | FW | in square wave, measure the frequency. Change the FAN\_CTRL to bring to correct speed. |  |
| FAN\_HALL2\_fpga | PL | in | FW | in square wave, measure the frequency of hall sensor2 same as fan speed. Change the FAN\_CTRL to bring to correct speed. Change Hz to RPM. | down |
| FAN\_PG2\_fpga | PL | in | FW | PG sent to the log file requires both PG for the fan voltage and fan speed OK. | down |
| FAN\_CTRL2\_fpga | PL | out | FW | in square wave, measure the frequency. Change the FAN\_CTRL to bring to correct speed. |  |
| FAN\_EN2\_fpga | PL | out | FW | in square wave, measure the frequency. Change the FAN\_CTRL to bring to correct speed. |  |
| FAN\_HALL3\_fpga | PL | in | FW | in square wave, measure the frequency of hall sensor3 same as fan speed. Change the FAN\_CTRL to bring to correct speed. Change Hz to RPM. | down |
| FAN\_PG3\_fpga | PL | in | FW | PG sent to the log file requires both PG for the fan voltage and fan speed OK. | down |
| FAN\_CTRL3\_fpga | PL | out | FW | in square wave, measure the frequency. Change the FAN\_CTRL to bring to correct speed. |  |
| FAN\_EN3\_fpga | PL | out | FW | in square wave, measure the frequency. Change the FAN\_CTRL to bring to correct speed. |  |
| ETH\_TX\_CLK | PS |  |  |  |  |
| ETH\_TXD0 | PS |  |  |  |  |
| ETH\_TXD1 | PS |  |  |  |  |
| ETH\_TXD2 | PS |  |  |  |  |
| ETH\_TXD3 | PS |  |  |  |  |
| ETH\_TX\_CTRL | PS |  |  |  |  |
| ETH\_RX\_CLK | PS |  |  |  |  |
| ETH\_RXD0 | PS |  |  |  |  |
| ETH\_RXD1 | PS |  |  |  |  |
| ETH\_RXD2 | PS |  |  |  |  |
| ETH\_RXD3 | PS |  |  |  |  |
| ETH\_RX\_CTRL | PS |  |  |  |  |
| ETH\_MDC | PS |  |  |  |  |
| ETH\_MDIO | PS |  |  |  |  |
| SPI\_CLK | PS | out |  | elapsed time FLASH MIO28 |  |
| SPI\_MISO | PS | in |  | elapsed time FLASH MIO29 | down |
| SPI\_CSN | PS | out |  | elapsed time FLASH MIO30 |  |
| SPI\_MOSI | PS | out |  | elapsed time FLASH MIO33 |  |
| SD\_CLK | PS | out |  | eMMC |  |
| SD\_CMD | PS | out |  | eMMC |  |
| SD\_D0 | PS | inout |  | eMMC |  |
| SD\_D1 | PS | inout |  | eMMC |  |
| SD\_D2 | PS | inout |  | eMMC |  |
| SD\_D3 | PS | inout |  | eMMC |  |
| UART\_RXD\_PL | PL | in | FW | FPGA debug UART(MISO) |  |
| UART\_TXD\_PL | PL | out | FW | FPGA debug UART(MOSI) |  |
| UART\_RXD | PS | in | FW | debug UART for PS(MISO) |  |
| UART\_TXD | PS | out | FW | debug UART for PS(MOSI) |  |

**Appendix A: Zero Crossing Detection**

All the RMS voltages and currents are calculated as a rolling average of exactly three 400Hz cycles and the 400Hz line frequency can vary at least up to 10%. The rms calculations need to use the correct number of samples to get close to three complete line cycles even if the line frequency varies from 400Hz. Note that the calculation of line frequency (based on Phase A voltage) can be used for all rms measurements in the system (even for phases B and C).

When the polarity changes, ignore for 10 samples (1msec) to provide a debounce.