Version: PROD_ICC_REV3.3, For use with ICC Board Module Ver: 3.3
Last Modified: March, 31st, 2011

This document describes the functionality and details of the PIC24FJ128GA010 microcontroller firmware used in the Internal Crate Controller board, which is part of the Instrument Control System (ICS), or Slow Controls Subsystem for the Dark Energy Survey Project. This document is meant to be a reference to the source code for the user.

**Circuit Description Summary:**

The Internal Crate Controller (ICC) monitors the Detector Head Electronics subsystems for any over-voltage, over-humidity, over-temperature or fan rotation speed failure. The ICC board monitors an ‘nSHUTDOWN’ signal when installed in the heater controller crate. The ‘nSHUTDOWN’ signal is monitored through the +5 v analog voltage. The ‘nSHUTDOWN’ signal will be high (+5v) when functioning normally and be low (0v) when a shutdown is needed. If a failure is present, the ICC will turn off the Vicor power supply and notify the Instrument Control System (ICS) a failure has occurred. The ICC also saves fault information to non-volatile SRAM. This provides a history of failure and signal measurements which can help diagnose the cause of failure. Each board has its own firmware code. This is done because the gains in the measurement circuit between each board vary and require a calibration constant to adjust the value.

**ICS – ICC Communication Summary:**

The ICS communicates with the ICC by using Master SPI communication protocol. SPI protocol uses a master clock to synchronize serial data between a single master and slave(s). The ICS is the master SPI device while the ICC is the slave SPI device.

1 Common +5 volt signal, 4 SPI signals, and 2 control signals make up the communication signals between the master ICS and the slave ICC. Another signal is used to emulate the ICS being connected when it is not. These signals are driven at TTL levels (+5 volts) and are optically isolated to keep conducted noise through the returns isolated.

**Common +5 volt Signal:**

1. +5V – Constant +5 volts which drive the ICC_SDO_RTN & DHE_SYS_OK_RTN signals.

**SPI Signals:**

1. SCLK_IN - The master clock. Data (commands) coming from the ICS, is read during the rising edge of SCLK_IN. Data (response values.) coming from the ICC, is read by the ICS during the next low edge of SCLK_IN.
SCLK_IN is an active-low signal (starts high, then goes low) at the ICS end and an active-high signal (starts low, then goes high) at the ICC end.

SCLK_IN_RTN – The return signal for SCLK_IN. Is connected to NI9403 returns. Is isolated from ICC returns.

2. ICC_SDI – Serial data coming from ICS. ICC_SDI is an active-low signal at the ICS end and an active-high signal at the ICC end.

ICC_SDI_RTN – Return signal for ICC_SDI. Is connected to NI9403 returns. Is isolated from ICC returns.

3. ICC_SDO_RTN – Serial data going to ICS. ICC_SDO_RTN is an active-high signal at both ends. Connection to the NI9401 module is not done in the normal manner. Rather, ICC_SDO is driven to a static high (+5v), while ICC_SDO_RTN is being monitored. A pull-down resistor (of 2.1k) is added to drive the signal back to ground during a low signal.

ICC_SDO – Constant +5v signal coming from NI9403.

4. SPI_CS – SPI Chip Select/Slave Select. A high level selects the ICC board and Prepares it for SPI data communication.


**Control Signals:**

1. DHE_SYS_OK_RTN – Control signal coming from ICS. A static high level indicates there are no faults present or DHE system is ok. A static low level indicates a fault has occurred and further investigation is required. DHE_SYS_OK_OUT is is an active-high signal at both ends. Connection to the NI9401 module is not done in the normal manner. Rather, ICC_SDO is driven to a static high (+5v), while ICC_SDO_RTN is being monitored.

DHE_SYS_OK_OUT – Constant +5v signal coming from NI9403.

2. OK_TO_PWR_UP_RTN - Control signal coming from the ICS. It provides control over when the Vicor power supply can be turned on or off and when the value is written to NVRAM.

The ICC will continue to evaluate the subsystem for faults while waiting for the ‘OK_TO_PWR_UP’ to come up. The Vicor power supplies will be allowed to come up to voltage, or turn on, once the control signal is raised. The Vicor power supplies can also be immediately turned off by dropping this signal low. This control signal also controls when to
clear the values in NVRAM. When the signal is low, indicating to turn off, or not allow the power supply to come up, the ICC doesn’t clear NVRAM. To clear NVRAM the OK_TO_PWR_UP signal must be set to turn on the Vicor p.s.

The idea behind the stored values to NVRAM is that after a fault has occurred, the ICS leaves the Vicor p.s. off (OK_TO_PWR_UP signal) and sends a 'Get Values from NVRAM' command to get the stored fault data. Once the fault is understood, and the unit is ready for normal operation, then turn on Vicor p.s. (OK_TO_PWR_UP signal) and issue a 'Reset ICC' command. This will clear the memory so it will be ready to save new fault values.

One condition associated with the power supply shutdown interlock circuitry is that if the ‘OK_TO_PWR_UP’ signal is low the ICC board will shut off the Vicor supply and the ICC board will set an over-voltage fault condition. Over-voltage fault condition can occur when the monitored voltages are above a nominal value or when the monitored voltages are off. To get the ICC board functioning properly after a low ‘OK_TO_PWR_UP’ signal has been given, the ‘OK_TO_PWR_UP’ signal should be raised to high and the ICS should send a reset command to the ICC board to clear the over-voltage fault condition and let the power supplies come up in the right sequence.

OK_TO_PWR_UP_IN_RTN – Return signal for OK_TO_PWR_UP_IN. Is connected to NI9403 returns. Is isolated from ICC returns.

Emulated Signal: This signal is NOT INTENDED to be connected to NI9403 module.
1. OK_PWR_UP_ECO – Control signal that allows the ICC board to run and power up the Vicor power supply without the ICS cable connected. An emulator plug is used to emulate the ICS.

OK_PWR_UP_RTN_ECO – Return signal for OK_PWR_UP_ECO. Is connected to ICC returns.

ICS – ICC Commands Summary:
One of four commands can be sent to the ICC by the ICS:

Command #1: ‘Initiate a software RESET’ (‘0x1111’):
1. The ICC will respond to indicate it is receiving the command with ‘0xCCCC’.
2. The ICC will send ‘0x1111’ to indicate a software reset is about to occur.
3. The ICC will initiate a software RESET and begin program execution at the initial state.
**Command #2: Clear all fan speed fault signals on fan detection circuit** (‘0x5555’):
1. The ICC will respond to indicate it is receiving the command with ‘0xCCCC’.
2. The ICC will send ‘0x5555’ to indicate the fan fault signals have been cleared.
3. The ICC will clear all faults coming from the fan rotation speed fault circuitry. This is done to see if there might be a glitch in the ICC subsystem which may have caused a pre-mature failure to be set. A valid fault will set the appropriate fan rotation speed fault again once the ICC resumes operation.

**Command #3: ‘Send current values’ (‘0x9999’):**
1. The ICC will respond to indicate it is receiving the command with ‘0xCCCC’.
2. The ICC will send ‘0x9999’ to indicate the current values were sent.
3. The ICC will send the current value, or one group of 11, 16-bit values to the ICS. Each value represents the measured values of voltages, temperature and humidity; plus the fault condition which indicates what caused the fault.

**Command #4: ‘Send saved NVRAM values’ (‘0xAAAA’):**
1. The ICC will respond to indicate it is receiving the command with ‘0xCCCC’.
2. The ICC will send ‘0xAAAA’ to indicate the saved NVRAM values were sent.
3. The ICC will send 3 groups of 11 (or 528), 16-bit values to the ICS:
   - Pre-fault NVRAM values are values that were saved to NVRAM before a fault was set.
   - Current-Fault NVRAM values are values saved to NVRAM during the fault.
   - Post-Fault NVRAM values are values saved right after the fault occurred.
   The values represent the measured values of voltages, temperature and humidity; plus the fault condition which indicates what caused the fault.

**Command #5: ‘Get Serial Number ’ (‘0xB BBBB’):**
1. The ICC will respond to indicate it is receiving the command with ‘0xCCCC’.
3. The ICC will send ‘0xB BBBB’ to indicate the serial number was sent.
4. The ICC will send the serial number to the ICS.

**Master ICS to slave ICC Communication Timing Diagram:**

The clock timing diagram describes the relationship between clock levels and other communication signals coming from the Instrument Control System (or Master SPI). The Master SPI clock has a period of 3.0mS which all commands and responses are dependant upon. ICC communications timing diagram information can be found at the end of this document, under **Appendix A**.
Hardware/Development Platform:
The Microchip PIC24FJ128GA010 microcontroller is used to control the ICC system. The microcontroller is programmed and tested using Microchip MPLAB IDE, version 8.10 Integrated Development Environment. The firmware was written in C using MPLAB C30 v3.00 or later compiler and MPLAB LINK30 linker. The linker uses the linker script file p24FJ128GA010.gld. Header File p24FJ128GA010.h is also needed to build the project.

State Machine:
This program uses a state machine to run the tasks.

The states are:
state_init → state_check_for_faults → state_measure_values →
state_is_humidity_high → state_is_temp_high → state_is_voltage_high →
state_store_values → state_process_faults → state_spi_request → state_check_for_faults

Function Calls:
These states are written as function calls. Function calls in the firmware.

void state_init(int *) // Initialize & Configure PIC24 uController PORTS & Peripherals

void state_check_for_faults(int *) // Set bit in Fault_Condition register based on fault
// that occurred

void state_measure_values(int *) // Measures voltages on A/D inputs and saves
// them to ADCValueArray

void state_is_humidity_high(int *) // Set 'Measured Humidity Fault' bit in Fault
// Condition register if the measured humidity value
// in ADCValueArray is above a certain level

void state_is_temp_high(int *) // Set 'Measured Temperature Fault' bit in Fault
// Condition register if the measured temperature in
// ADCValueArray is above a certain level

void state_is_voltage_high(int *) // Set 'Measured Voltage Fault' bit in Fault
// Condition register if any of the measured voltages
// in ADCValueArray is above a certain level

void state_store_values(int *) // Gets values from ADCValueArray and stores
// them into NVRam
void state_process_faults(int *)  // Looks at Fault_Condition register,  
// Drops, DHE_SYS_OK signal

void state_spi_request(int *)  // Sends values to Master SPI when clocked

void state_get_values()  // Gets values from NVRam and saves them into  
// ADCValueArray

void state_clear_fan_faults()  // Clear fan faults

void state_wait_for_fans();  // A delay loop used to allow PIC24 to wait until  
// fans come up to full speed before checking for  
// any fan alerts

void state_wait_for_Vicor();  // A delay loop used to allow the Vicor power  
// supply to come up and allow the Volts_Good  
// circuitry to take over.

void state_xfer_spi_data();  // Send data to/receive commands from ICS using  
// SPI protocol. 16 bits of data is sent/received  
// at a time.

**Global Variables:**
There are 6 global variables used in the firmware:

Fault_Condition – Register which holds integer values that describe which fault has occurred.

PMP_Addr – Address Register used to read and write integer values from/to the  
NVRam Parallel Master Port.

ICC_SDI– An integer structure used to hold 16 bits. This variable is used to get the serial  
data from the ICS, one bit at a time, starting with the most significant bit.

Fan_CNTRL – An control variable which turns the fans off and on.

IsPostFaultValueStored – A control variable that indicates if the post-fault values have  
been saved to NVRAM.

IsFaultPresent – A control variable that indicates if a fault was set.

IsCurrentValueStored – A control variable that indicates if the current fault values have  
been saved to NVRAM.
ADCValueArray[11] – An array which holds the measured values and the fault condition value. These values are: measured temp, measured -5VA, measured +3.3VD, measured +5VA, measured +15VA, measured +48VA, measured -28VA, measured -15VA, measured humidity, measured +5VD, and the Fault Condition register value.

SPIValueArray[11] -- A temporary array which holds current ADCValueArray value, read from NVRAM, to be sent to the ICS via SPI.

ICC_SDO_BITS – A integer structure used to hold 16 bits. This variable is used to send the serial data to the ICS, one bit at a time, starting with the most significant bit.

ICC_SDI_BITS – An integer structure used to hold 16 bits. This variable is used to receive the serial data from the ICS, one bit at a time, starting with the most significant bit.