

OTHER FEATURES

- Pseudo real time clock from 32khz watch crystal and GPS UTC time
- delay_ms function
- Macro defined functionality for most pins in config.h
- Basic clock system, configuration done in init.c
- Ring buffers (rbuffer.c/h)
- Checksums (fletcher16.c/h)

ERRORS:

For the most part error were captured haphazardly, there is an error containing structure with simply one byte variables to show which part of the code is breaking. This structure is contained in error.h Errors contained there are associated with:

- Particular sensor or device on the i2c bus
- GPS module
- SD card and file operations

In most of the software I've tried to output 0 if an operation has been successful, but whether or not that is the case everywhere is questionable.

TELEMETRY:

Most telemetry is captured in telem_points.h which contains a definition for a struct, an instance of which is used to capture all important data points.

- Five 3 axis measurements of magnetic field
- Five groups of 3 measurements of light intensity and temperature associated with each group
- 3 axis gyroscopic measurements from the SNS board frame
- 3 axis acceleration measurements from the SNS board frame
- Latitude, longitude, and altitude
- UTC time
- Real time to be associated with measurements.
- Pressure and temperature measurements

SALVO REAL TIME OPERATION SYSTEM:

- OS_Timer (used for task delay)
- Context Switching
- Semaphores (Binary, Counting)
- Queues
- Scheduling
- Message

TASK:

task_log_low_f
task_log_high_f

PRIORITY: 3
WAITS ON: LOG_HIGH_F_P
LOG_LOW_F_P
SIGNALS: None
DELAY: None
DESCRIPTION:

- Opens various log files, does chip select with GPIO
- Uses sprintf to write data as c-strings to be logged by the f_write function
- Provides error messages about the status of file operations.

TASK:

task_led

PRIORITY: 1
WAITS ON: None
SIGNALS: None
DELAY: None
DESCRIPTION:

- Provides a visual heartbeat

TASK:

task_measure_high_f

PRIORITY: 6
WAITS ON: None
SIGNALS: NEW_HIGH_F_P
DELAY: None
DESCRIPTION:

- Attitude data collection task:
 - Configures accelerometer (ADXL345), gyroscope (L3G4200D), magnetometers (hmc5883)
 - Samples high frequency measurements of acc., gyro., and mag.

TASK:

task_check_error

PRIORITY: 1
WAITS ON: CHECK_ERROR_P
SIGNALS: None
DELAY: None
DESCRIPTION:

- If an error has occurred on the I2C bus all the devices are powered off and back on. Then the devices are reconfigured.
- If the GPS experiences an error it is powered off and back on and is then reconfigured.

TASK:

task_measure_low_f

PRIORITY: 3
WAITS ON: COLLECT_LOW_F_P
SIGNALS: LOG_LOW_F
DELAY: None
DESCRIPTION:

- Configures GPS module (lea6) and pressure/temperature sensor (BMP085/BMP180)
- Samples pressure/temperature sensor via I2C, polls GPS for time, longitude/latitude, and altitude via UART
- Transmits attitude via UART to Raspberry PI

TASK:

task_stamp_comm

PRIORITY: NEW_HIGH_F_P
WAITS ON: None
SIGNALS: None
DELAY: None
DESCRIPTION:

- Configures UART at high baud rate to communicate attitude data to linux based flight computer system
- Transmits attitude data via UART to linux based system with additional bytes for identification and parsing
- TODO: Create latch with received uart transmission from linux system.

DEVICE:

micro SD card

Protocol: SPI
Driver: effs_thin_mmc_drv_ucb0.h/c
lib_effs_thin_msp430x-1.hza
Comment:

- 3rd party driver (Pumpkin Cubesat heritage) uses self contained SPI setup to communicate with an SD card.
- The lib_effs_thin_msp430x-1.hza contains the necessary functions to read and write from the sd card as well as other functionality. So far only basic writing has taken place. Documentation can be found in the effs_fat_@62.pdf file on redmine

ISOLATION:

The configurations for various states of power ons and offs as well as i2c and uart enables can be found bus_isolation.c/h and have appropriately named functions to handle certain common isolation tasks to enable proper function, not talking to multiple i2c devices with the same address, not talking to two UART devices at once, and so on.

DEVICE: pca9543

Protocol: I2C
Driver: pca9543.h/c
Comment: Addressable IO expander used to control the power and i2c enables of the various devices in the system. See documentation for more details.

DEVICE: tca9548

Protocol: I2C
Driver: tca9548.h/c
Comment: 8 channel i2c switch with reset reset used to switch the i2c bus that the msp430 can interface with.

DEVICE: ltc1393

Protocol: I2C
Driver: ltc1393.h/c
Comment:

- Four channel multiplexer used to select between the two UART channels. One connects the MSP430 to devices the other connects the MSP430 to they linux system. Driver is complete and used in bus_isolation.c

DEVICE: hmc5883

Protocol: I2C
Driver: hmc5883.h/c
Comment: 3 axis magnetometer with complete driver for almost all functionality. Much of the driver is untested. Depending on configuration measurements must be converted to gauss.

DEVICE: adf7994

Protocol: I2C
Driver: adf7994.h/c
Comment: Four channel analog to digital converter with 2.5 Volt reference. CEREBRO uses adf7994 which has 12 bits of resolution. CEREBRO uses it to determine voltages generated by 3 photodiodes and Im20 temperature sensor. Transfer functions are applied to the measurements to make them useful

DEVICE: adxl345

Protocol: I2C
Driver: adxl345.h/c
Comment: 3 axis accelerometer with measurements output in gravities (g's). Driver adapted from arduino library, contains extensive functionality much of which is untested and unused currently. This device is the accelerometer in the iphone.

DEVICE: l3g4200d

Protocol: I2C
Driver: l3g4200d.h/c
Comment: 3 axis gyroscope measurements in degrees per second. Driver adapted from an arduino library. Many of the driver's functions are untested.

DEVICE: bmp180/bmp085

Protocol: I2C
Driver: bmp085.h/c
Comment: Temperature and pressure sensor. Not a completely encapsulated implementation. Multiple calls and waits required beyond configuration.

DEVICE: lea6 GPS module

Protocol: UART/SPI
Driver: lea6.h/c
Comment: GPS module that understands UBLOX's UBX protocol over either UART or SPI. Driver is well abstracted for polling GPXGA NMEA sentences containing position and altitude information. Other kinds of sentences contains other information. Easy development of new functionalities and configurations can be done using the Ucenter software available from the manufacturer UBLOX. Can only be used at 9600 Baud.

Generic Linux system

Protocol: UART/I2C
Driver: develop.cpp/.hpp
Comment: Driver contains a state machine for receiving the data from CEREBRO. Once data is received then the attitude determination can be done and then used by control code to send a current to the motor controller (magento) board.

CEREBRO SOFTWARE MAP

Brief functional and connectivity description of the system dubbed CEREBRO which collects attitude and flight data for the Michigan eXploration Laboratory's project Strato. Software is currently located in the openlibraries git repository in the branch cerebro_dev.

Author: Joshua McCready
Email: jmccread@umich.edu
Date: January 22, 2015
Revision: 1.0

BUS PROTOCOL: I2C

Drivers: i2c.c/h
i2cdev.c/h

Capability:

i2c.h:
Contains basic msp430 register level hardware implementation for msp430 as master i2c communications. Standardized initialization of processor registers to communicate properly, read and write functions, and proper interrupts are defined here.

i2cdev.c/h:
These were adapted from an arduino library and similar functions are found elsewhere. These allow for easy writing and reading from devices by adding a layer of abstraction over the basic hardware level driver.

BUS PROTOCOL: UART

Drivers: uart.c/h
usci_a_uart.c

Capability:

uart.c/h:
Contains basic msp430 register level hardware implementation for msp430 for serial communications. However it is not a complete implementation because of a hardcoded initialization. It contains functions for encoding and decoding any standard type of array of data elements using the convention described in uart.h. The convention for sending data is to append a start byte, the type of data, the number of elements, an identifier, and a checksum. uart.c contains the interrupts used to send and receive bytes from the uart buffer.

usci_a_uart.c/h

TI provided drivers for the uart bus, contains necessary setup, read, and write functions.