SAE Baja Data Acquisition System

Team E#5

Milestone #4
DETAILED DESIGN REVIEW AND TEST PLAN

http://www.eng.fsu.edu/~willidew/baja
https://github.com/hp09d/SAE-Baja-Data-Acquisition

February 13, 2015

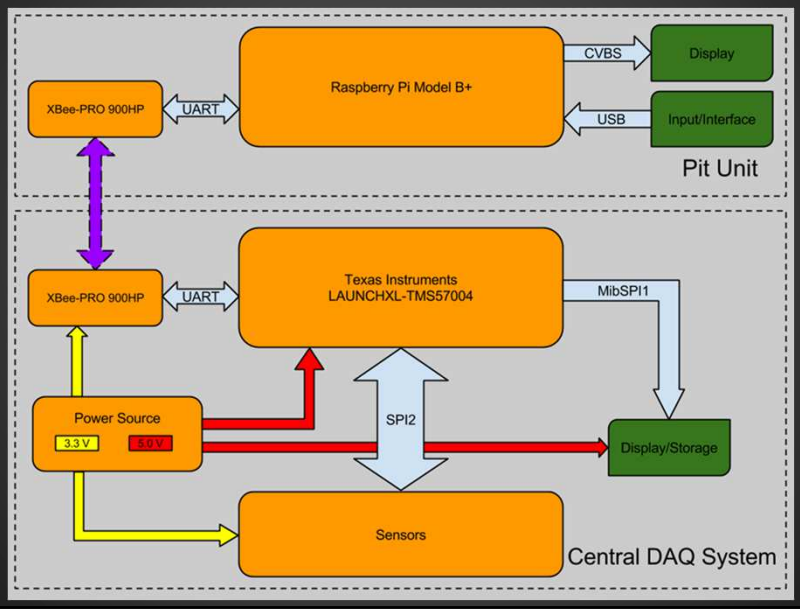
Project Overview

- Goal: To develop a comprehensive and reliable data acquisition system for use in SAE's Baja Series off road competition
- Collects data such as:
  - Speed
  - Acceleration
  - Tire Pressure
- Stores data to a log file and transmits readings wirelessly to a remote display
- Applicant in Texas Instruments' North America Innovation Challenge contest
  - Received free ICs and a store voucher for development tools

1

Christopher Riker

Top Level Block Diagram



Christopher Riker

Design Updates

Christopher Riker

TI Innovation Challenge

Design Changes

- **TI Innovation Challenge requires the use of 2 TI ICs as well as a TI microprocessor**
- **Team received a $200 TI e-Store voucher**

MSP430G2553
- Similar to G2332 (not available in TI e-Store)
  - More FLASH
  - Dual USCI (vs. 1)

TL2575 Switching Regulators (3.3V and 5V)
- Step-down Buck regulator
- 50mV - 150mV ripple typical

TI DRV5053 Bipolar Hall Effect Sensor
- Linear output
- Vout = 1V @ B = 0mT

4

Christopher Riker

Accelerometer

ST Micro LSM303DLHC

Features
- Digital Output
  - Communication via I2C
  - Onboard 16-bit ADC
- Smaller range
  - +/- 2G minimum
- Onboard Magnetometer

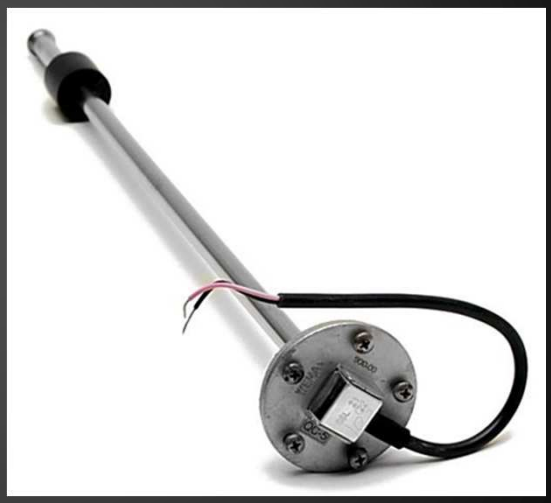
Reason for Change
- ADXL335 range too high for accurate measurements using 10-bit ADC
  - 1024 steps / 6g range = 170 steps/g = .0059 g/step
  - At 2g, Sensitivity = .001 g/step
  - Average automobile acceleration: 0.44g

5

Christopher Riker

Fuel Level Measurement
Removed from Design

- After a rule review, the team discovered that the gas tank must not have holes drilled in it, even if patched.
- Many other solutions implemented by other DAQ teams failed technical inspection
- Ultrasonic sensor possible, but too costly to implement this year



6

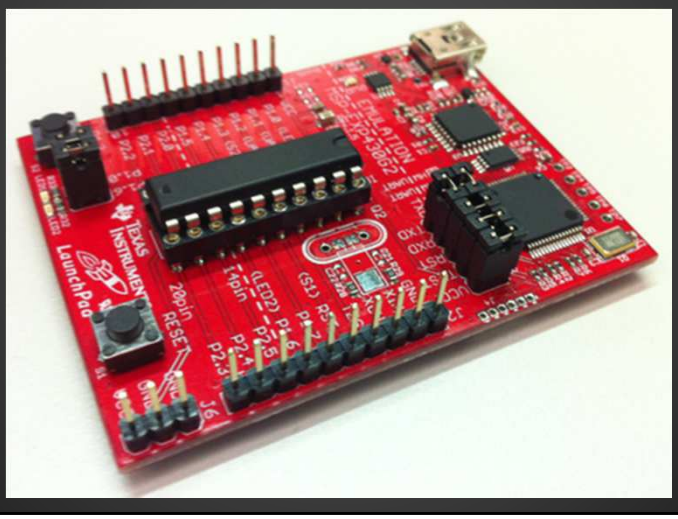
Christopher Riker

Design Review

7

Christopher Riker

Sensor Microcontrollers



8

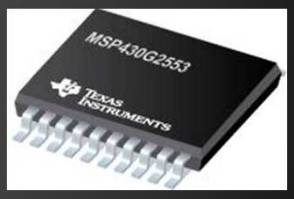
Christopher Riker

Sensor Microcontrollers
Development

Completed:
- Configuring SPI slave mode
- Sending multi-byte data over SPI
- ADC configuration and capture
- Embedded circuit design

In Progress:
- I^2C Master Mode
- Circuit board design and milling

Future Development:
- Configuring timers
- Final sensor code

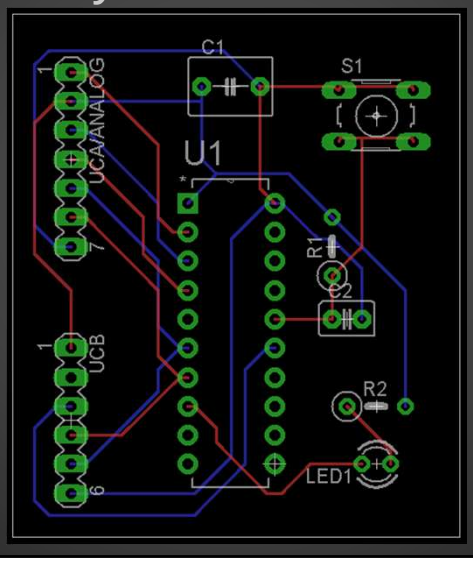


9

Christopher Riker

Sensor Microcontrollers

Preliminary Circuit Board Design



10

Christopher Riker

Sensor Microcontrollers

Code Samples

```
while ( 1 ) {
        ADC10CTL0 |= ENC + ADC10SC;
        while( !(ADC10CTL0 & ADC10IFG) ) {};
        pot = ADC10MEM;

        if ( betweenBytes == 0 ) {
                /*  !!!!!  BEGIN DANGER ZONE  !!!!!   */
                sampleArray[ byteNum ] = ( pot >> 8 );
                UCB0TXBUF = sampleArray[ byteNum ];
                matchingLSB = byteNum + 1;
                byteNum = (++byteNum) % 4;
                sampleArray[ byteNum ] = ( pot & 0x00FF );
                byteNum = (++byteNum) % 4;
                /*  !!!!!  END DANGER ZONE   !!!!!   */
        }
        ADC10CTL0 &= ~(ADC10SC);
}
```

```
#pragma vector=USCIAB0RX_VECTOR
__interrupt void USCI0RX_ISR_HOOK (void)
{
        betweenBytes ^= 1;
        while (!(IFG2 & UCB0RXIFG)) {};
        data = UCB0RXBUF;
        while (!(IFG2 & UCB0TXIFG)) {};
        UCB0TXBUF = sampleArray[ matchingLSB ];
        __delay_cycles(50);
}
```

11

Christopher Riker

Sensor Microcontrollers

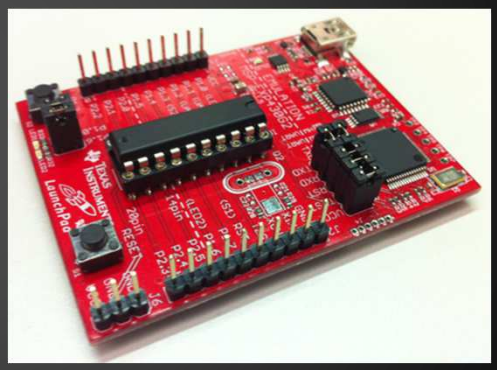
Test Plan

Completed Tests:

- ADC10 **PASS**
- SPI Slave Mode **PASS**
- SPI Slave Mode w/ ADC10 **PASS**

Planned Tests:

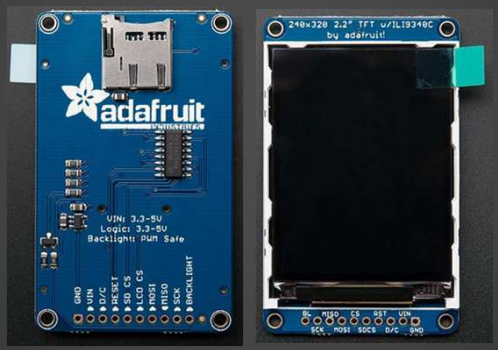
- I²C Master Mode
- I²C Master Mode w/ SPI Slave Mode
- PCB Functionality



12

Christopher Riker

Driver Display



13

Dewey Williams

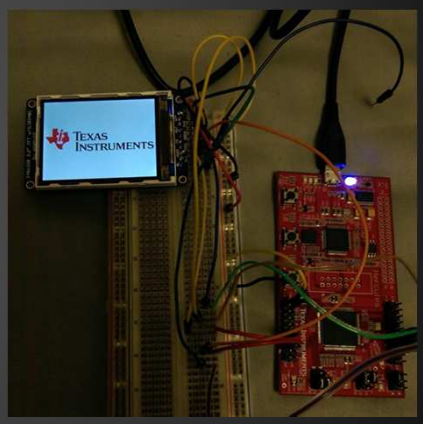
Driver Display
Development

Completed:
- Hercules communication via mibSPI
- Proof of concept (displaying bitmap)

In Progress:
- Driver software development
  - Port of Adafruit ILI9340 library
  - Will display text and bitmaps for simple GUI
  - Optimized for mibSPI

Future Development:
- Enclosure and mounting
- Final GUI design

Completed Test:
- Hercules Display Test



14

Dewey Williams

Driver Display
Code Sample

```
void ili9340DrawChar(uint16_t x, uint16_t y, uint8_t size, char c, uint16_t color){
    uint8_t row,col;
    uint16_t pix, mask, x0, y0, blob;
    for(row = 0; row < 8; ++row)
    {
        pix = console_font_9x8[16*c + 2*row];
        pix = pix<<8;
        pix |= console_font_9x8[16*c + 2*row + 1];
        mask = 0x8000;
        for(col = 0; col < 9;){
            if(pix & mask){
                blob = 0;
                x0 = x+col*size;
                y0 = y+row*size;
                while(pix & mask && col < 9){       //Find "blobs" in the row to optimize drawing
                    ++blob;
                    mask = mask >> 1;
                    ++col;
                }
                ili9340FillRect(x0, y0, blob*size, size, color);
            }else{
                ++col;
                mask = mask >> 1;
            }
        }
    }
}
```

15

Dewey Williams

Data Storage

16

Dewey Williams

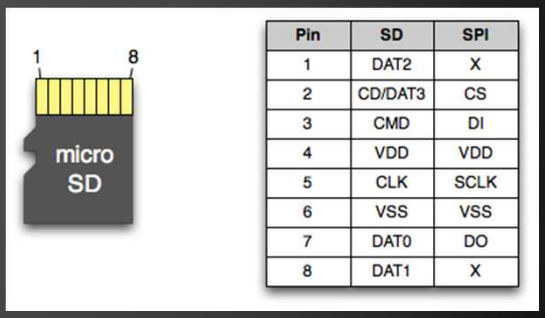
Data Storage
Development

Completed:
- Proof of concept (reading root directory)

In Progress:
- Low level driver development
  - FatFS used for high level file system

Future Development:
- Data formatting and scheduling
- Safe power down

Planned Test:
- Hercules SD Card Test



Pin	SD	SPI
1	DAT2	X
2	CD/DAT3	CS
3	CMD	DI
4	VDD	VDD
5	CLK	SCLK
6	VSS	VSS
7	DAT0	DO
8	DAT1	X

17

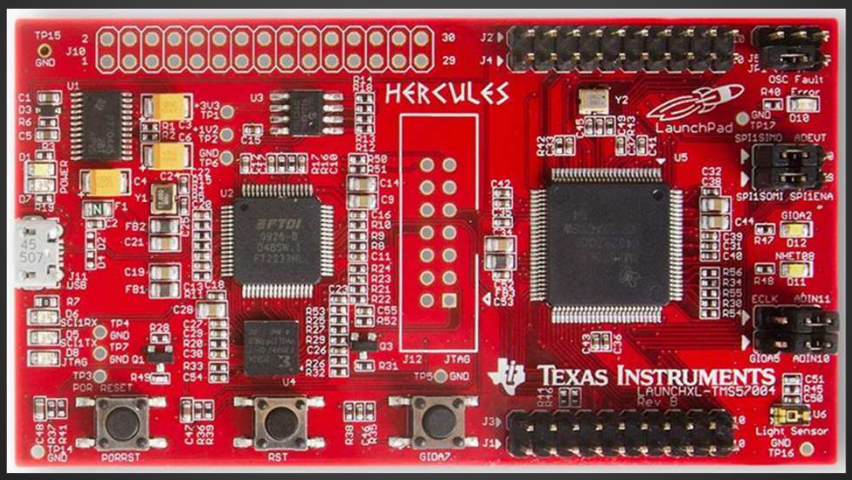
Dewey Williams

Data Storage
Code Samples

```
263  uint16_t sendCmd(uint8_t cmd, uint32_t arg){
264          uint16_t i = 0;
265          uint16_t src[5];
266
267          while(getByte() != 0xFF){
268                  //Wait for ready
269          }
270
271          src[0] = cmd | 0x40;
272          for( i = 0; i < 4; ++i)
273          {
274                  src[i+1] = (arg >> 8*(3-i)) & 0xFF;
275          }
276
277          uint16_t resp = 0x80; //Init to some invalid response
278          uint16_t crc = 0xFF;
279          if (cmd == CMD0) crc = 0X95;  // correct crc for CMD0 with arg 0
280          if (cmd == CMD8) crc = 0X87;  // correct crc for CMD8 with arg 0X1AA
281
282          spiTransmitData(spiREG1, &dataConfig1, 5, &src[0]); //send cmd and arg
283          spiTransmitAndReceiveData(spiREG1, &dataConfig1, 1, &crc, &resp); //send CRC
284
285          while( resp & 0x80 ){ //Send clocks until a response is received
286                  resp = getByte();
287          }
288
289          return resp;
290  }
```

```
305  uint16_t dataRead(uint32_t block, uint16_t offset, uint16_t count, uint16_t * dest){
306          uint16_t status = sendCmd(CMD17,block);
307          uint16_t i;
308
309          if( count + offset > 512 ){
310                  return 1;               //Out of bounds
311          }
312
313          if( status != 0 ){
314                  return 1;               //Fail
315          }else{
316                  status = 0xFF;
317                  while(status == 0xFF){
318                          status = getByte();
319                  }
320                  if( status != DATA_START_BLOCK ){
321                          return 1;       //Fail
322                  }
323          }
324
325          for( i = 0; i < offset; ++i ){
326                  getByte();                      //Skip to the offset
327          }
328          for( i = 0; i< count; ++i ){
329                  dest[i] = getByte();            //Read the data
330          }
331          for( i = count+offset; i < 512; ++i ){
332                  getByte();                      //Throw away  the rest
333          }
334
335          return 0;
336  }
```


Main Microcontroller

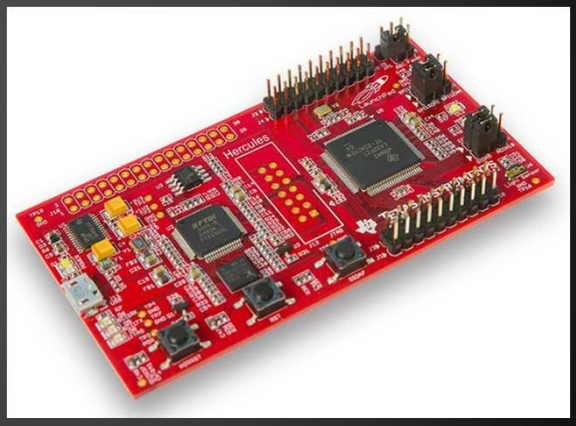


Main Microcontroller
Development

Completed
- SPI proof of concept w/ MSP430
- Display proof of concept w/ mibSPI
- SD card proof of concept w/ legacy SPI
- One-way XBee interfacing

In Progress
- Display driver development
- SD card driver development
- Two-way XBee interfacing

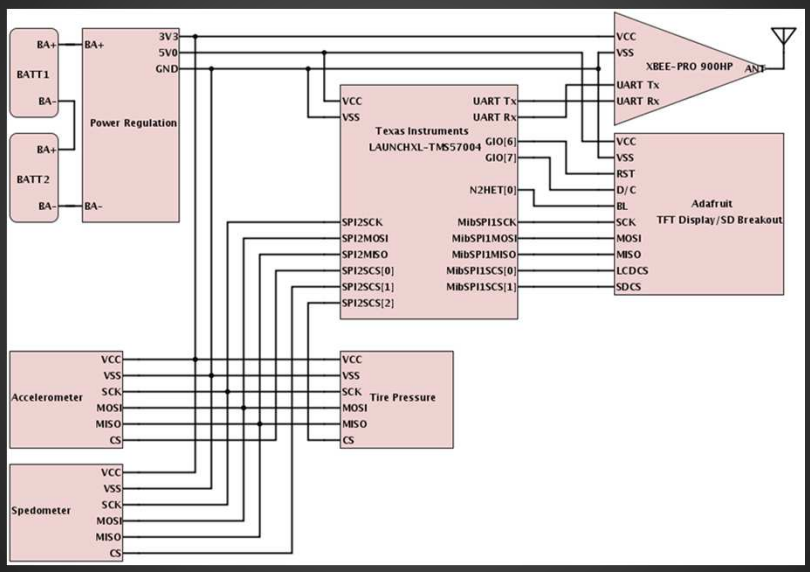
Future Development
- Data formatting
- Control flow management
- Optimization for real time operation



20

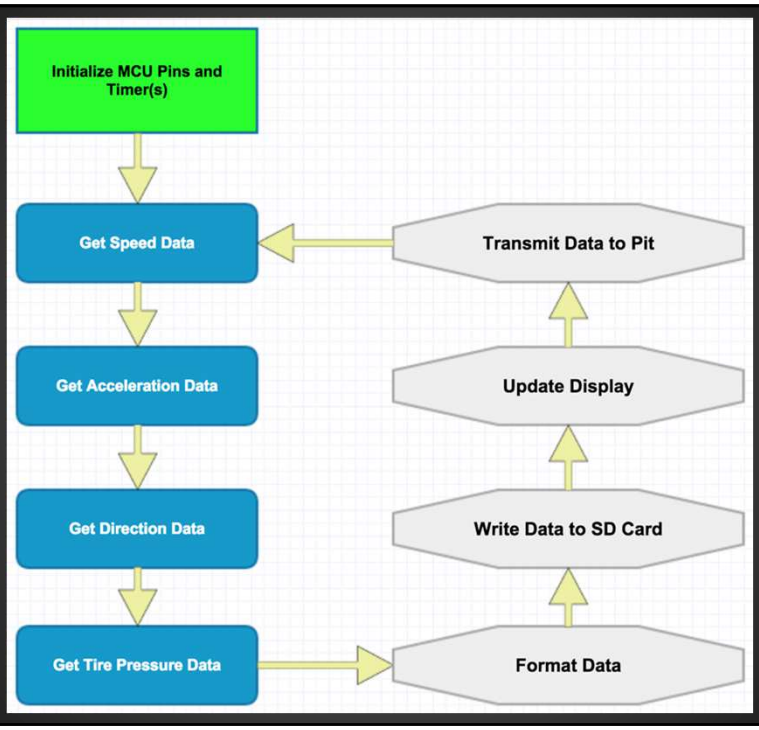
Dewey Williams

Main Microcontroller



21

Dewey Williams



22

Dewey Williams

Sensors

23

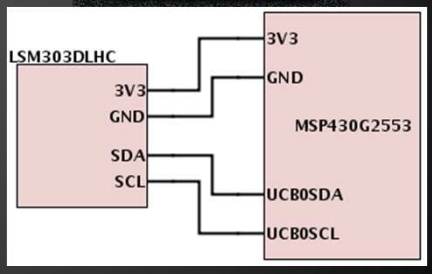
Hebe Perez

Accelerometer
Development

In Progress:
- Configuration and data access over I²C

Future Development:
- Sample filtering
- Getting direction from magnetometer
- Interrupt usage (rollover detection)

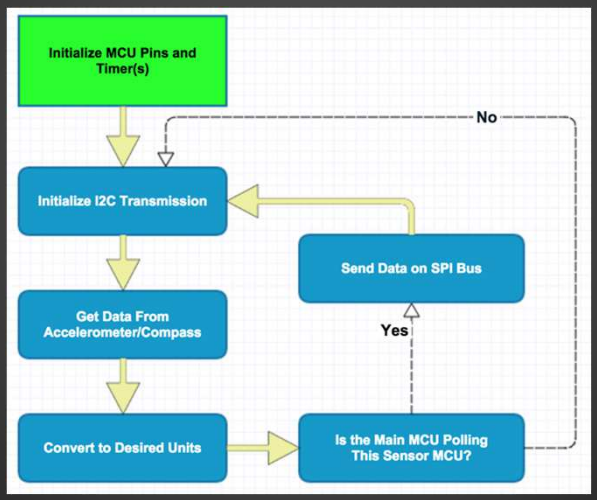
Planned Tests
- Accelerometer calibration test



24

Hebe Perez

Accelerometer
MCU Control Flow



25

Hebe Perez

Hall Effect Sensor
Development

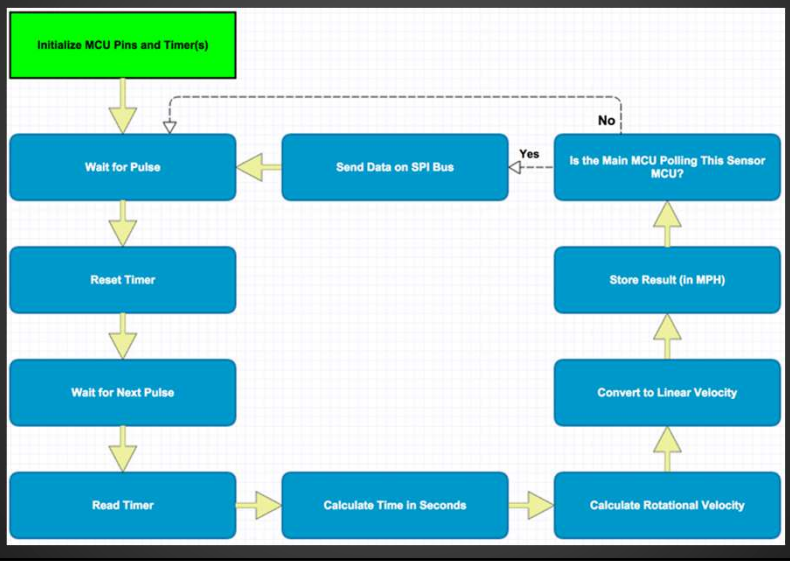
Future Development:
- MSP430 timer configuration and pulse detection
- Mounting options

Planned Tests
- Baseline functionality test
- Speed accuracy test
- Materials test

26

Hebe Perez

Hall Effect Sensor
Control Flow

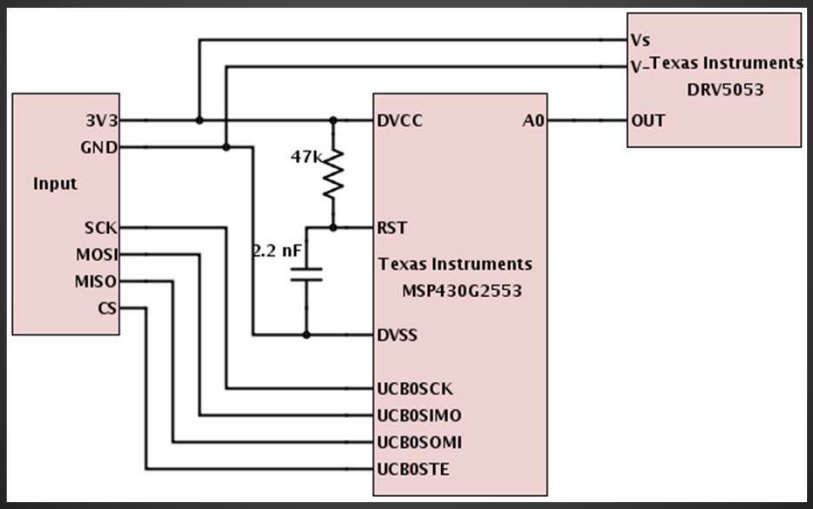


27

Hebe Perez

Hall Effect Sensor

Circuit Diagram



28

Hebe Perez

Tire Pressure Monitoring System

Schrader C4N3MF9 TPMS Sensors

Completed

- Behavioral research
  - Automatically transmits temperature and pressure data every 60 seconds

In Progress

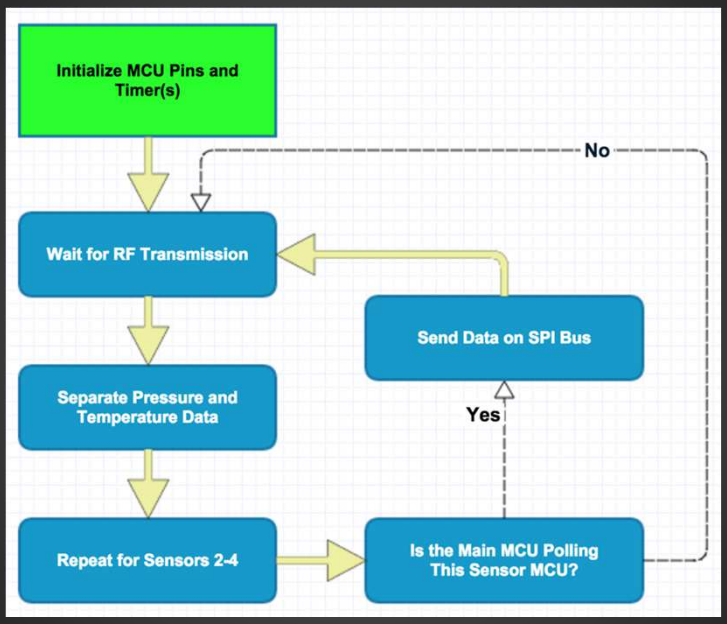
- RF communication research

Planned Test:

- Tire Pressure Sensor Test

29

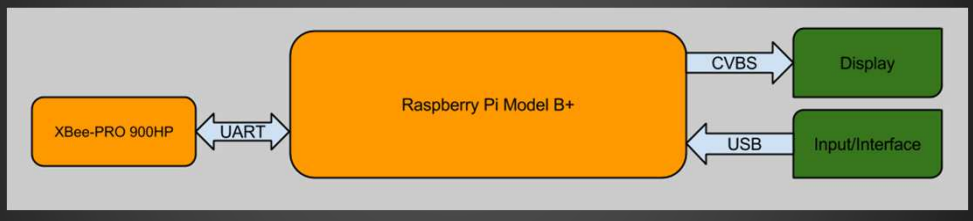
Hebe Perez



30

Hebe Perez

Pit Unit



31

Hebe Perez

Pit Unit CPU

Development

Completed:

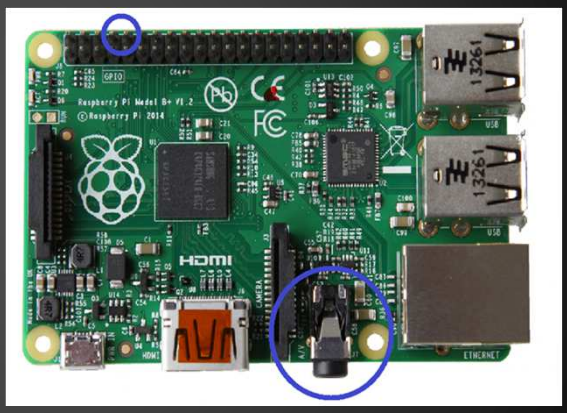
- Raspberry pi operating system installation.
- UART Port Initialization
- Communication via XBee

Future:

- User interface

Completed Test:

- XBee Raspberry Pi Test **PASS**



32

Hebe Perez

Pit Unit CPU

Code Sample

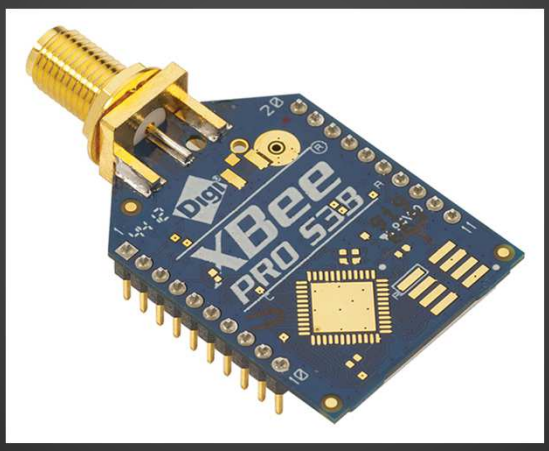
```
p_tx_buffer = &tx_buffer[0];
*p_tx_buffer++ = 'H';
*p_tx_buffer++ = 'e';
*p_tx_buffer++ = 'l';
*p_tx_buffer++ = 'l';
*p_tx_buffer++ = 'o';

if (uart0_filestream != -1)
{
        int count = write(uart0_filestream, &tx_buffer[0], (p_tx_buffer - &tx_buffer[0]));
        if (count < 0)
        {
                printf("UART TX error\n");
        }
}
unsigned char rx_buf[256];
int rx_len = read(uart0_filestream, (void*)rx_buf, 255);
while(1){
        if(rx_len <= 0){
                rx_len = read(uart0_filestream, (void*)rx_buf, 255);
        }else{
                rx_buf[rx_len] = '\0';
                printf("%s",rx_buf);
                fflush(stdout);
                rx_len = read(uart0_filestream, (void*)rx_buf, 255);
        }
}
```

33

Hebe Perez

Wireless Data Transmission



34

Tyler Dudley

Wireless Data Transmission
Development

Completed:
- Initial configuration and testing
- Communicates with Raspberry Pi via UART pins
  - Also works over USB via FTDI

In Progress:
- Communication with Hercules via UART
- Hercules-to-Pi communication

Future Development:
- Data formatting and timing

35

Tyler Dudley

Wireless Data Transmission

Test Plan

Completed Tests

- XBee Baseline Test **PASS**
- XBee Short Range Test **PASS**

Planned Tests

- XBee Long Range Test
- Communication between Hercules and Raspberry Pi
  - In-Lab
  - On-Vehicle

36

Tyler Dudley

Power System

37

Tyler Dudley

Power System
Development

Completed
- Batteries, regulators received
- Batteries tested for charge

In Progress
- Regulator PCB design
- Regulator testing
- Wiring options

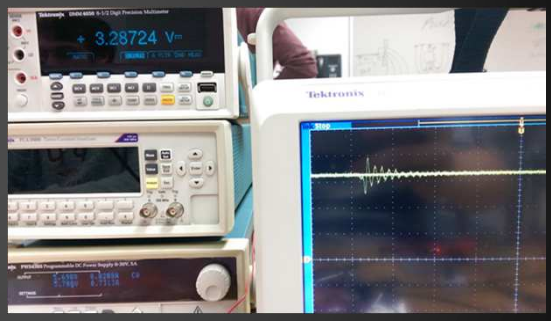
38

Tyler Dudley

Power System
Testing

Completed
- Voltage Regulator Test **FAIL**
- Battery Charge Test **PASS**

Planned
- Voltage Regulator Test #2
- Battery Life



39

Tyler Dudley

Weatherproofing

40

Tyler Dudley

Weatherproofing
Testing/Development

To Do:

- Test enclosures for waterproofing integrity
- Test “waterproof” cable grommets for waterproofing integrity
- Determine mounting positions/methods

41

Tyler Dudley

fin

42