IPRO 324
Power Measurements for Road Bicycles
"No strain, no gain…"

Advisors
Professor Dietmar Rempfer
Professor Sheldon Mostovoy

Team Members
Tarun Anupoju
Stephanus Halim
Bryan Kaminski
Brian Lam
Brandon Marcellis
Rebecca Martin
Edumaregbemiro Odunaiya
Stefan Stevanovic
Henrietta Tsosie
Ivan Voukadinov
Arkadiusz Ziomek

Table of Contents

Abstract

This semester, this IPRO group has worked to develop a system that measures the applied torque at a bicycle crankset. In contrast to current solutions, we are attempting to be able to retrofit our system to existing cranksets, obviating the need to abandon parts that the bicyclist already owns. In principle, according to preliminary tests performed at the MMAE department, this can be done using sets of quite inexpensive strain gages. However, being able to get accurate torque measurements requires some advanced processing of signals from the strain gages. These signals can then be transmitted wirelessly to a bicycle computer like the Garmin Edge 705 that the global positioning system corporation Garmin has released last spring. There is a defined wireless protocol, called ANT+Sport, which has been developed specifically for the purpose of transmitting exercise data, such as power output or heart rate, to small computers. After future semesters of this project, it is hoped that the market potential of a commercial product can be realized in a follow-up ENPRO project.

Background

The goal of the IPRO is to find an inexpensive, but accurate way of measuring the power output of a rider on a bicycle. Problems with systems currently available are: some products are not compatible with all bike systems causing the need to purchase new parts, the cost of the available products is expensive, and some of the available measuring systems are not very accurate.

There are four main ways in which systems measure the power output of a rider. They include crank set, free hub, chain, and opposing force systems. The crank set system uses strain gages to measure the strain in the crank set which can be related to torque from which the power is calculated. The free hub system works in much the same way except the strain gages are attached to the rear wheel of the bicycle. Chain systems detect the vibration frequency and the speed in the chain and convert that to a power reading. Opposing force systems calculate opposing forces to the rider and bicycle including: gravity, drag, acceleration of the bicycle, and wind speed. The system takes all this information and calculates the power using Newton's Third Law.

The crank set is what the pedals are directly attached to on the bicycle. The crank set includes the spider, which is attached to the crank arm, and the chain rings, which drive the chain. The freehub is used to connect the chain to the rear wheel.

Crank set systems can be very complicated and therefore are very expensive. Not only are the systems themselves expensive, but the system requires a new crank assembly, causing the replacement of an expensive part of the bicycle. The Quarq CinQo has an accuracy of ± 2%, but costs $1495 [(1)].

The free-hub systems have problems similar to the ones involved in crank set systems. Their measurement accuracy is diminished because the power output of the rider is not directly measured. The PowerTap Pro made by Saris has an accuracy of ±1.5%, but costs $1,199.00 with the computer [(2)].

Inaccuracy is a bigger problem with the chain systems because of power loss from the crank to the chain as in the free-hub systems as well as vibration in the chain caused by other factors including terrain. The Polar CS600 cycling computer with power output

sensor has an accuracy of ± 5%, and is less expensive than the crank set and free-hub systems at $419.95 [(3)] which also includes the bicycle computer.

While cost is not as much a factor in the opposing force systems as in the crank set systems the accuracy can be far less. The inaccuracy can be caused by drag which is affected by rider position, weight fluctuation of the rider, as well as the roughness of riding surface. The iBike Pro claims to have accuracies comparable to those of high end models, like the crankshaft and free-hub systems, but it becomes more inaccurate in sharp turns or long stretches of rough terrain. The cost of the iBike system is $689.00 [(4)].

Another side of the project is the interaction with the rider. This is done through the bicycle computer. The computer processes the information from the power measurement systems and displays it for the rider to see. Problems faced with the computers involve finding a way to relay the information wirelessly. The Garmin Edge 705 bicycle computer will be used to communicate the information to the rider. The ANT+Sport system will be used for communication between the computer and the rest of the system.

Results from the project's survey concluded that the cycling community wants a more affordable system than what is currently available, with the majority of the people surveyed selecting a $800-$1000 budget range. The cycling community also prefers a device which does not require either a crankset or hub replacement.

During the last semester (first semester of this IPRO), the team was able to implement the strain gage system in a crank set. However, they did not attain accurate results and found that the crank angle needed to be figured into the algorithm that allowed calculation of the power. A method of testing was also developed, but will be improved in order to speed up the process. For the electrical side the previous team was able to come up with a power circuit as well as make great progress on a reading circuit as well as a start on the wireless communication needed from the system to the bicycle computer.

Objectives

Electrical Team Objectives

The major objective of the electrical team is to develop an electrical system that can be used to measure the power usage of a cyclist. To develop this system, the team needs to:

1. Trigger measurement at specific angles using Reed switches
2. Amplify the voltage to a value that can be used in the calculations of the force
3. Minimize electronic noise
4. Convert the analog signal to digital for use in calculation
5. Develop code to transmit power output to the Garmin bike computer using the wireless ANT+ protocol

A problem encountered last semester was the inability to obtain a set of coefficients independent of the crank angle. As a solution, the team decided to use Reed switches to measure the angle of the crankset. This was to help determine which strain gages to read from and therefore, present a more accurate reading of the force being applied.

Mechanical Team Objectives

The mechanical team objectives are listed below:

1. Define strain gage position on crankset
2. Apply strain gages on the crankset
3. Design an experiment to measure the output of the strain gages under different load conditions
    a. Crank angle
    b. Point of force application
        i. Left pedal
        ii. Right pedal
    c. Outer chain ring vs. inner chain ring
4. Analyze data and implement an algorithm to calculate torque

The mechanical team decided to use Wheatstone bridges as opposed to quarter bridges. This makes the circuit less complicated as well as provides a stronger signal.

Methodology

Electrical Team Methodology

The Electrical team methodology was to plan, implement, and debug individual circuits, building on the results from the previous semester. Once complete, they are combined to form the final product. Individual circuits include strain gage amplification, ANT+ wireless communication, and RPM and Crank Angle monitoring. The strain gage circuit was designed to accept signal information from 4 individual Wheatstone bridges. These signals are passed individually though 4 low-resistance/low-noise switches, amplified by an instrumentation amplifier and finally passed to the microcontroller's Analog-to-Digital converter. This value can then be processed by an algorithm, from which the power output can be calculated and sent via wireless transmission to a Garmin Edge 705 cycling computer.

Changes from the proposed Project Plan

- The electrical team extended allowed time to work on ANT+ communication due to a bug in the microcontroller
- The team did not design the PCB layout because they needed to focus resources on other tasks
- Strain gage circuit required additional time to develop and refine

Mechanical Team Methodology

The initial research was done to see what was currently available for power measurement. It was decided that a system of strain gages was to be used to measure the strain in each spider arm (Figure 1). This data would then be used to create an algorithm which would relate the strain to the applied torque and would allow for a method of measuring the power of a cyclist. A total of 4 bridges of strain gages were used, one for each spider arm and a separate bridge for the two spider arms to which the crank attaches to (Figure 2). The mechanical team was responsible for gluing and calibrating the strain gages and subsequently collected the needed data.

Figure 1. Placement of strain gages inside spider.

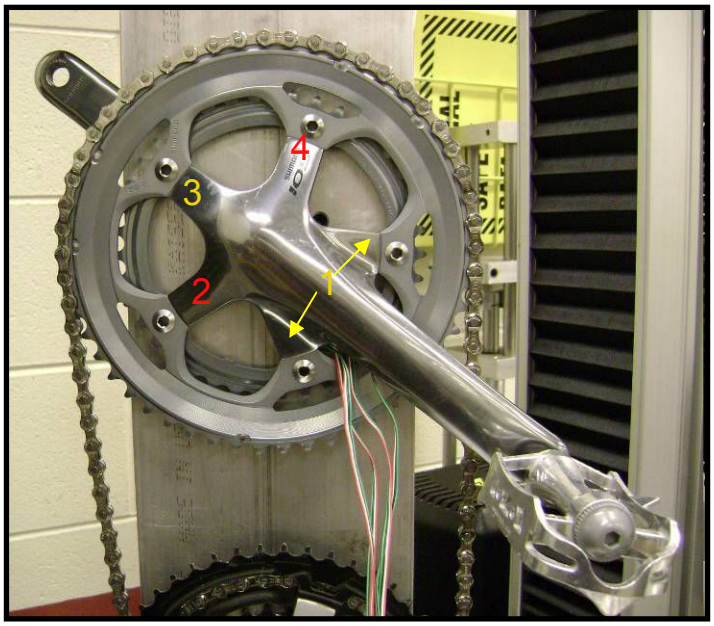


Figure 2. Placement of four Wheat stone bridges on crank set.

The mechanical setup involved loading the crank set apparatus under the Instron 5500 (Figure 3). A load of 90 lbf was applied at several angles ranging from 0-180 degrees. The zero degree reference is when the pedal is in a vertical position at the top. Measurements were taken for the inner chain ring as well as the outer chain ring. The strain data was recorded using the Vishay 6200 scanner (see Figure 4) as well as the Strain Smart software.

Figure 3. Measurement of angle before load is applied by the Instron 5500.

Figure 4. Vishay 6200 scanner.

Changes made to the project plan:

- The testing started about a month late because the strain gages took more time to glue than expected. Testing started on April 9th.
- The strain gage data acquisition machine had to be sent to the manufacturer for repair so extra time had to be allowed for this.
- The old crankset was never tested.

Team Structure and Assignments

Name	Major / Year	Subteam	IPRO Assignments	Contribution
Tarun Anupoju	Computer Engineering 4th Year	Electrical	Wireless Communication	Wireless Team: ANT+ development, helping out with the Stain gage circuit, Conducted survey for IPRO by sending questionnaire to websites and to major universities in Illinois which generated the maximum amount of responses possible for the online survey.
Stephanus Halim	Computer Engineering 3rd year	Electrical	Wireless Communication	Construction and debugging of strain measurement circuit, helped with ANT+ wireless development.
Bryan Kaminski	Electrical Engineering 4th Year	Electrical	Electrical Team Leader, Gage Interface	Team Leader Design and debug of strain measurement circuit; ANT+ development
Brian Lam	Mechanical Engineering/Physics 4th Year	Mechanical	Testing, MATLAB	Apply strain gages, acquire and analyze data
Brandon Marcellis	Aerospace Engineering 3rd Year	Mechanical	Mechanical Team Leader, Strain Gages Testing	Mechanical Team Leader, apply gages, work with measuring equipment, acquire and analyze data
Rebecca Martin	Mechanical Engineering 4th Year	Mechanical	Apply Strain Gages, Testing, Scribe	Prepared the spider for strain gages application, applied and tested strain gages. Project's scribe and webmaster.
Edumaregbemiro Odunaiya	ECE 4th Year	Electrical	Gage Interface	Design of strain measurement circuit; Helped out in biking team survey and in design of poster
Stefan Stevanovic	Mechanical Engineering 3rd Year	Mechanical	Testing, Apply Strain Gages	Designed reed switch ring, had testing apparatus modified for new crank set, acquire and analyze data, worked with measuring equipment
Henrietta Tsosie	Mechanical Engineering 4th Year	Mechanical	Apply Strain Gages, Testing	Project Leader, strain gages application, and data analysis, abstract, compiling final report
Ivan Voukadinov	Mechanical/Aerospace Engineering 4th Year	Mechanical	Pro/E, Apply strain gages, Testing	Applying strain gages, testing and analyzing data
Arkadiusz Ziomek	ECE 4th Year	Electrical	Gage Interface	Design and debugging of measurement circuit; work on ANT+;

Table 1. Team structure and team member assignments.

Budget

ITEM	UNIT PRICE	QTY	PRICE	PURPOSE	Vendor
Electrical					
MSP430 Dev Kit	$20.00	1	$20.00	Wireless	Multiple
INA 122 (Amplifier)	$5.00	1	$5.00	Amplifier	Texas Instruments
Voltage regulator	$5.00	1	$5.00	Regulate Voltage	National SemiConductor
Switches ADG 801	$4.00	4	$16.00	Switching Bridges	Analog Devices
Support Components	--		$54.00	Batteries & other electronics	Multiple
Soldering tips	$6.15	4	$24.60	Soldering	HMC Electronics
Mechanical					
Crank Set	$142.90	1	$142.90	R & D	Shimano
Bottom Bracket	$38.00	1	$38.00	Testing apparatus	Kozy's Cyclery
Half Bridge Gages (5 pack)	$50.00	6	$300.00		Vishay
Final Presentation, Materials					
Refreshments and food			$100.00	Team Building	Multiple
		Total	**$705.50**		

Table 2. Budget of IPRO 324.

Code of Ethics

The purpose of the project is to create a final power measuring device for road bicycles that is more affordable than current available products. This will be done while maintaining the principles of honesty and integrity for sake of both the biking community as well as the invested stakeholders.

7 Layers of Integrity

1) The Law
    i) Members shall respect and abide by the laws stated by the United States.
    ii) Intellectual Property
    b) Example
        i) Patent laws exclude other parties from using design of product. Members may feel the need to copy patent in order to get a working product with results in time.
2) Contracts and Agreements
    i) Members shall not engage in contracts or legal agreements by outside sources without proper approval.
    b) Example
        i) Members agreed to participate in discussion and take part in requirements of project.
        ii) Members also agreed to work towards to the objectives and goals of the team.
3) Professional Code of Ethics
    i) Members will abide by the engineering code of ethics provided by ASME and IEEE.

b) Example
    i) Members are expected to use their knowledge in their respective fields to benefit the project and protect the safety, health, and welfare of the public. Prevent faulty engineering and false data.

4) Industry Standards
    i) Members are not certified engineers therefore will not remake the product without the consent or supervision of the certified engineers involved with the project.

5) Social, Civic and Geographic Communities
    i) Members are to respect and abide by the social, civic, and geographic communities affected by the project.
    b) Example
        i) Members will try to understand the values and standards among the communities in which they operate.

6) Personal Relationships
    i) Relationships inside and outside the team are to be respected and will be treated with a professional manner. Mutual respect amongst team members is expected.
    b) Example
        i) The project is not the only one thing that any team member has and stress may sometimes engage members in arguments that lead to personal attacks. These should always be treated in a professional manner.
        ii) The educational opportunity is the most important factor and should be respected. Ideas and questions are never disregarded and some may feel disrespected when their ideas are dismissed.

7) Moral and Spiritual Values
    i) Team members will not accept credit for another team member's work. Assignments will not be divided unfairly amongst teams.
    b) Example
        i) Team members may feel the need to accept work for praise by sponsors or other members.
        ii) Members may assign more tasks to team members that have been getting them done which can be unfair to other team members.
        iii) Collected data can not be falsified or exchanged in order to complete tasks.

Results

Electrical Team Results

The electrical team successfully designed and implemented a circuit which measures the revolutions per minute (RPM) as well as process strain gage signals and transmits data wirelessly. The results are described below:

1) Main Microcontroller

The microcontroller used for the final application this project was the PIC18F4331 [(5)]. This microcontroller was programmed using the Microchip ICD2 [(6)] development hardware, which interfaces with the MPLAB IDE [(7)]. The code for the

program is written in C and utilizes the Microchip C18 Libraries [8]. A note of caution: the datasheet for the PIC18F4331 lists two pairs of pinouts for the SPI data port, citing one set as "alternate" pins. The use of these "alternate" pins is not thoroughly described in the datasheet, but the correct pins are noted on the final schematic (Appendix A).

2.) Power supply

1V and 3V supplies were used in the circuit. A voltage regulator keeps the output at a constant 1V, which is then used to drive the strain gages. The team used a LP3878 [9] voltage regulator from National Semiconductor to achieve this (See Appendix A). The output voltage is currently fixed at 1V, but can be changed by selecting a different ratio of resistors R1 and R2. Consult the datasheet for more information. This eliminates error in calculating the corresponding strain value. The change from the quarter bridge to the full bridge required a change in the measurement circuitry. Two switches are being used for each bridge. This modification prevents saturation of the amplifier thereby allowing all the bridges to work efficiently.

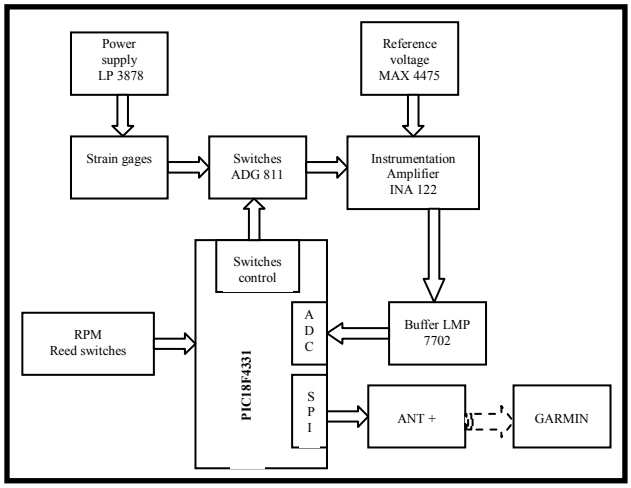


Figure 5. Circuit block diagram.

3.) RPM Circuit

In order to calculate the power generated by the rider, the torque must be multiplied by the RPM. RPM measurement is implemented by a simple Reed switch and magnet combination. When a magnet passes over the Reed switch, the circuit is closed, allowing for current to pass through. Eight Reed switches are placed at an angle of 45 degrees apart from each other. One switch is wired to a single pin on the microcontroller, while the other 7 are wired in parallel to another pin. These pins are set up as "Interrupt-on-Change", which means a change on the pin (caused by the magnet) will cause the microcontroller to jump to the interrupt routines. The single switch is taken as the reference point or 0 degrees. Once the reference switch is activated, a timer in software begins to count how long until the next time the reference switch is activated. This is how the RPM can be calculated. Every time one of the other switches is activated, a counter is incremented, from which the angle of the crank can be roughly estimated (0°,45°,90°,135°,180°, 225°, 270°, or 315°).

4.) Wireless Communication

One of the objectives of the electrical team was to implement the ANT+ chip to wirelessly transmit data to a handheld receiver which would display the power details to the cyclist. The team successfully communicated data from the circuit to the bicycle computer. This was implemented by following the specifications provided by ANT+, which are located in the data sheets on iGroups. (See Appendix B for Source Code).

5) Amplification Circuit

For amplification of the strain gauge signal, a Texas Instruments INA122 is used, which is a single-supply instrumentation amplifier. The signal from the strain gages can have both a positive and negative swing, so in order to be able to read the negative swing, the amplifier must be provided with a reference voltage which shifts the output of the amplifier within an acceptable range. For this design, the supply voltage is 3V, so the reference voltage is set to 1.5V, which allows the amplifier output to swing roughly +/-1.5V. Then, a voltage read below 1.5V can be interpreted as "negative" and higher than 1.5V can be interpreted as "positive". The reference voltage pin on the amplifier requires a low-impedance input, so a buffer (MAX4475) is used to set this voltage. The offset voltage can then be adjusted by using the potentiometer connected to pin 3 of the buffer [10], (Also See Appendix A).

Mechanical Team Results

It was decided that strain gages be placed on the crankset as full Wheatstone bridges each on an arm of the spider (Figure 2). It was found that the strain in the crankset was dependent on the angle and which chain ring the chain was attached. It was decided that the strain be measured at eight different angles of the crank arm each 45° with the angles offset from the directly vertical angle at which no torque is applied to the crankset. The crankset was tested at four angles on the left crank arm and the four opposite angles on the right arm. The inner and outer chain rings were tested at each angle. Each test gave the voltage drop across all four of the strain gage bridges, which the electrical team uses to find the torque. In order to find the torque from the voltage readings an algorithm was developed. The torque can be found using the expression,

$$T = C_1 \cdot V_1 + C_2 \cdot V_2$$

where T is the torque, C_1 and C_2 are coefficients, and V_1 and V_2 are voltages. Each angle has two different coefficients because of the effect of location of the chain. Since there are only two parameters for each angle only two strain gage bridges were needed. Referring to Figure 2, bridges 2 and 4 were used for the calculations. They were chosen because of their symmetry about 90 degrees (See graphs below). The coefficients were calculated by setting up a system of two equations with two voltages for each equation. The system of equations can be solved for the coefficients.

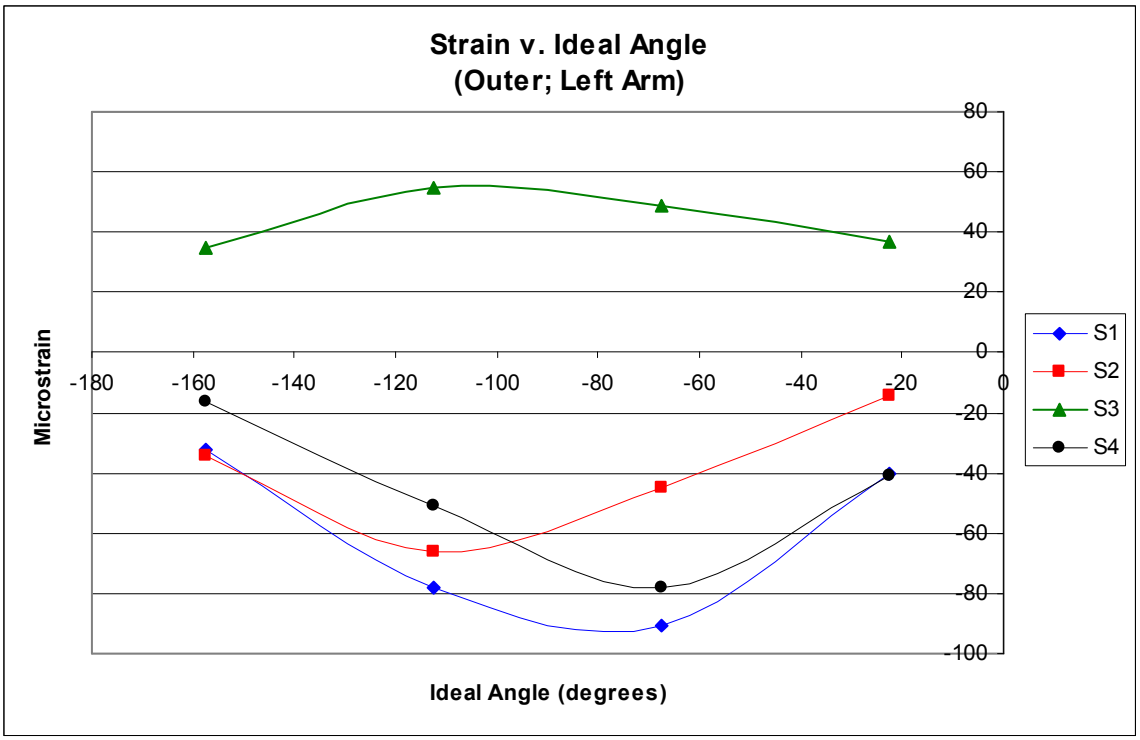


Figure 6. Strain versus angle results obtained for all bridges on the outer ring, left pedal arm.

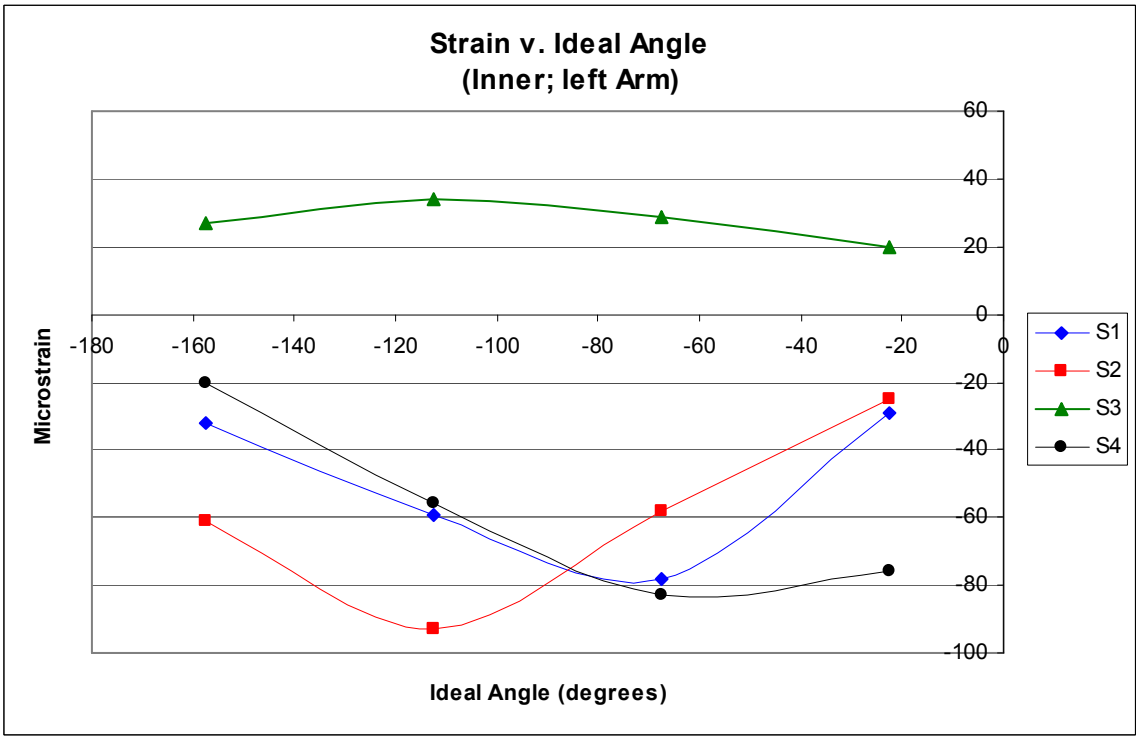


Figure 7. Strain versus angle results obtained for all bridges on the inner ring, left pedal arm.

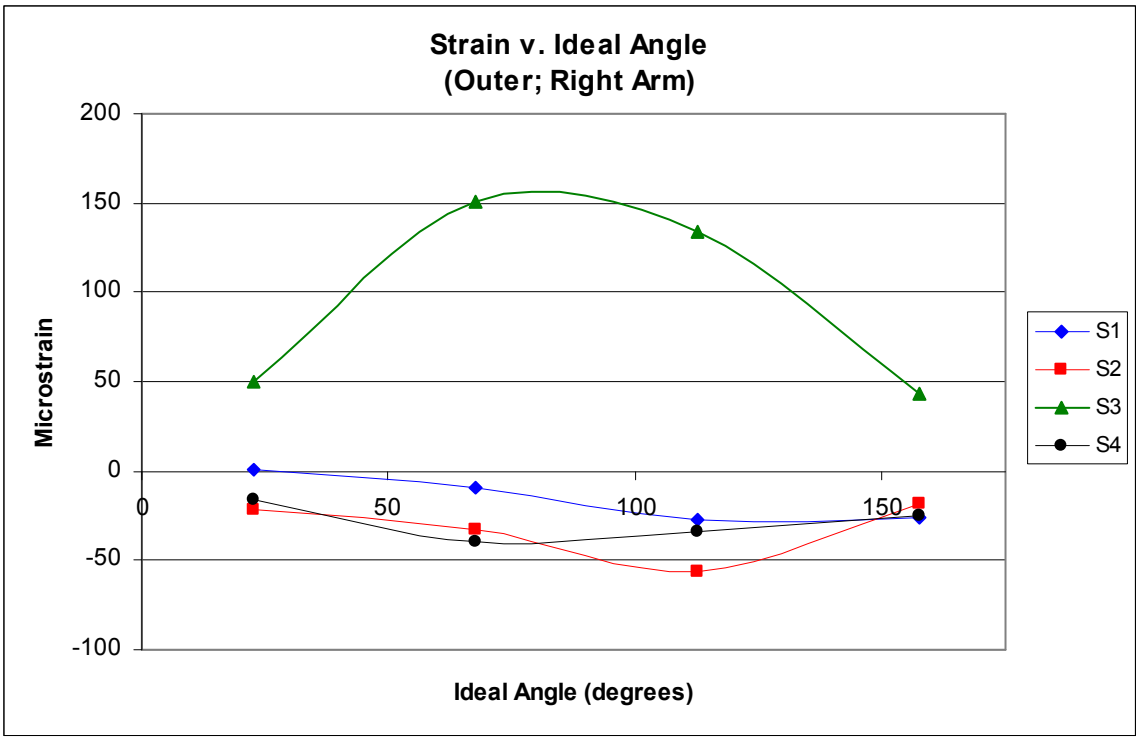


Figure 8. Strain versus angle results obtained for all bridges on the outer ring, right pedal arm.

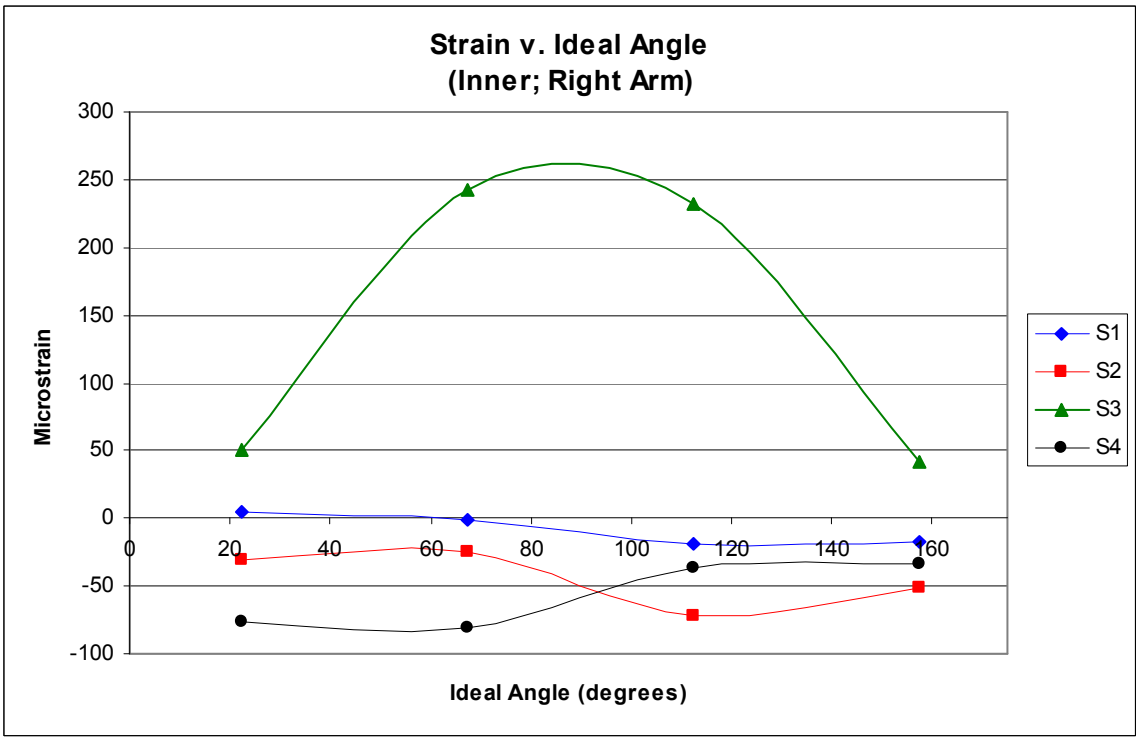


Figure 9. Strain versus angle results obtained for all bridges on the inner ring, right pedal arm.

Angle (°)	22.5	67.5	112.5	157.5	202.5	247.5	292.5	337.5
C_2	-5559	-6533	2027	1215	1295	567.6	656.8	-39803
C_4	711.5	-1302	-11251	-5350	-9459	-2920	-1806	11456

- C_2 and C_4 refer to our bridge two and bridge four respectively
- The angle is the angle between the right crank arm and the vertical position

Obstacles

Electrical Team Obstacles

The main obstacle in this project was balancing strain gages in quarter-bridge configuration. Because of the off-balance, the amplifier was saturated at all times. Even though the team installed switches with better characteristics, the amplifier was still saturated. Due to this, the team changed the measuring circuit from a quarter-bridge to a full-bridge configuration. The full-bridge configuration, due to presence of four strain gages, performs better because each of those strain gages compensate for the other three gages. However, it was still hard to balance the bridge. It turned out that additional long wires, to connect measuring bridge with electric circuit, could cause the problem. Thus, the team used a new set of strain gages designed for full-bridge configuration, by this we have decreased the number of long wires in the circuit.

The change from quarter-bridge to full-bridge required a redesign of measuring circuitry. This was done because the objectives of the team changed from the previous semester. Last semester, the team used 10 switches with 3 fixed resistors. This semester only used 4 Wheatstone bridges therefore the resistors were not needed. In the first attempt to get a signal from the four measuring bridges through the use of only one amplifier, the team used analog switches to disconnect only one signal line of each of these bridges. The problem with this was that disconnecting only one signal line, while the second line was still connected to amplifier, caused saturation of the amplifier. As a result two switches for each bridge were used to disconnect both signal lines from the amplifier.

When connecting the instrumentation amplifier (op-amp) to an Analog to Digital Converter (ADC) the signal was very low close to 0. After reading datasheets for both devices it turned out that output impedance of op-amp must be smaller than 2.5kΩ, otherwise the ADC will not operate correctly. Thus, a voltage follower, which has output impedance smaller than 1kΩ, was implemented between those devices.

During initialization, there was an interruption of the microcontroller which caused it to not operate correctly. In the errata to the datasheet of this microcontroller the team found that a physical bug existed in the internal structure of this microcontroller. To alleviate this problem, the team had to set one more register not related to the interrupt. The problem was figured out beside the bug in the interrupt system there is a problem with port A of this microcontroller. In particular, two of the pins that were being used to connect the microcontroller and ANT+, were damaged inside the microcontroller. As a result the team had to use a new microcontroller.

Mechanical Team Obstacles

The mechanical aspect of this project encountered several difficulties. The first challenge was placing the strain gages in a discreet location without sacrificing their effectiveness in measuring strain. Based on previous studies, it was decided that they should be placed on the inner surface of the spider arms (which have a C-channel cross-section) and midway along the length of the arms. Each arm would contain a full Wheatstone bridge (4 strain gages) with two gages placed on the flat surface and the other two on curved surfaces. This arrangement would minimize damage to the gages and keep them hidden without sacrificing the quality of data they obtain.

Another obstacle was establishing communication between the Vishay scanner and computer software. The software was upgraded in hopes of restoring communication. However, the firmware in the device also needed upgrading. The team decided to send the instrument back to Vishay for the necessary update which would take roughly two weeks.

The chosen strain gage arrangement created several challenges. Gluing the gages in place became difficult because of their extremely small size and fragile nature. There was a tendency for the gages to lift off of the curved surface, in which case they had to be re-applied. Soldering introduced another challenge. New soldering tips had to be ordered that were fine enough to properly solder wires to the gages. On occasion, the solder would break and a wire would have to be soldered again. It was also extremely easy to accidentally solder two tabs together, rendering the strain gage useless. The difficulties encountered in preparing the crank for testing set the team back several weeks.

In addition, the new crank set was not compatible with the existing test apparatus, which further delayed testing. One piece required modification to allow it to clamp onto the new bottom bracket. A tool also had to be fabricated to thread one side of the crank holder.

Another set of obstacles was anticipating whether it would be necessary to measure crank angle, and predicting the effect of inner and outer chain rings on torque measurements. The team decided early on that it was unlikely that the strains measured would be independent of crank angle. Thus, the crank angle had to be tracked somehow. This obstacle was overcome by creating a carrier ring for reed switches which would be triggered by the passing motion of the crank arm. The strains measured could also be a function of which chain ring is used. This meant that both inner and outer chain rings had to be tested in order to verify their relationship to the strains.

Possibly the most challenging obstacle involved creating an algorithm to convert the strains to torque and ultimately power measurements. According to previous studies, reading the gages as full bridges would create a different relationship than reading the gages individually. The team decided to further investigate reading the gages as Wheatstone bridges, which would reduce power consumption and therefore improve battery life.

Recommendations

The following is a list of recommendations given by the team:

- Research market and provide business case
- Conduct a Finite Element Analysis on the spider to find optimal locations for strain gage placement as well as locating areas unaffected by applied loads

- It was found that using unshielded wires for the bridge connections introduces electrostatic interference into the circuit. For presentation and demonstration purposes, shielded connection wires should be used to minimize this interference.
- It is recommended to include additional switches to disconnect any unused bridges from the excitation voltage. This will allow the circuit to use less battery power. Additionally, the voltage regulator IC has a 'shutdown' pin which can be utilized to turn the regulator off, saving additional battery power.
- In order to ensure accurate power values, a method of calibrating should be implemented by future teams. This calibration might also implement a method for self-balancing bridges.
- The electrical schematics for this project are located on IGROUPS in Protel format.

References

(1) - http://www.quarq.us/cinqo
(2) - http://www.cycle-ops.com/p-362-powertap-pro.aspx
(3) - http://www.polar.fi/us-en/products/cycling/CS600/
(4) - http://www.ibikesports.com/detail.aspx?ID=55
(5) - http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010288
(6) - http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en010046
(7) - http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002
(8) - ww1.microchip.com/downloads/en/devicedoc/MPLAB_C18_Libraries_51297f.pdf
(9) - http://www.national.com/mpf/LP/LP3878-ADJ.html
(10) - http://www.maxim-ic.com/quick_view2.cfm/qv_pk/3085

Personal Communication - Darryl Peterson, Vishay Micromeasurements
Personal Communication – Dynstream Corporation

Resources

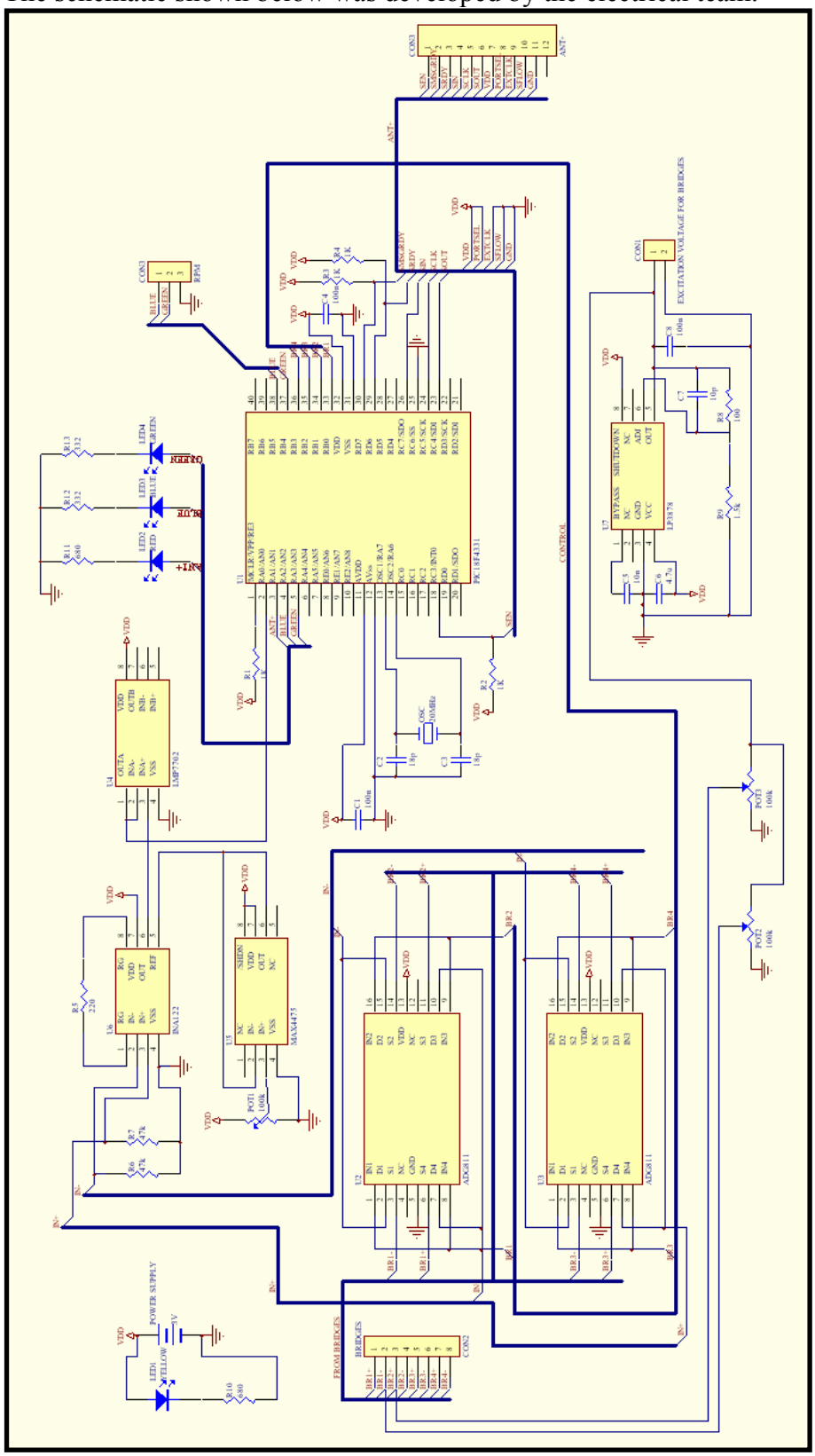
Interfacing with ANT modules, ANT protocol information, ANT dev Kit Manual
Garmin Edge 705
National Semiconductor (Voltage Regulator)
Texas Instruments
Vishay Micromeasurements

Acknowledgements

Russell Janota
Professor Dietmar Rempfer
Professor Sheldon Mostovoy
Craig Johnson

Appendix A

The schematic shown below was developed by the electrical team.



Appendix B

Source code provided by electrical team.

```
/***********************************
IPRO324 BIKE POWER METER CODE
LAST REVISION: 05/06/2009, Bryan Kaminski

BACKGROUND INFORMATION: ANT+ DATA IS PUT INTO THE BUFFER ARRAY (send_buffer[]) AND THE
MESSAGE LENGTH VARIABLE IS UPDATED. THEN THE UPDATE FLAG IS CLEARED, AND SMSGRDY
SIGNAL IS DRIVEN LOW WHICH CAUSES THE HIGH PRIORITY INTERRUPT CODE TO RUN, WHICH SENDS
OUT THE DATA THROUGH THE SPI PORT TO THE ANT+ CHIP. WHEN THE PIC RECEIVES A
NOTIFICATION FROM ANT+ TO UPDATE DATA (through the INT0 interrupt pin), THE UPDATE FLAG
IS SET. WHEN THE UPDATE FLAG IS SET, WE ARE CLEAR TO UPDATE NEW DATA.
THIS OCCURS IN THE MAIN LOOP, WHICH EXECUTES FOREVER.

ALSO, DATA MUST BE TRANSMITTED LSB FIRST TO ANT+ CHIP, WHICH IS EASILY OVERLOOKED IN THE
DATASHEET.
THE HARDWARE SPI PORT ON THE PIC DOES NOT SUPPORT THIS NATIVELY SO THE MACRO/FUNCTION
flip_byte IS USED TO REVERSE MSB/LSB IN SOFTWARE.
*/

#include<p18f4331.h>
#include<delays.h>
#include<math.h>
#include<spi.h>

//SERIAL CONTROL PINS
/*
SCLK = PIN 14
SIN = PIN 15
SOUT = PIN 16
SEN = Interrupt on RB0
*/
#define SMSGRDY                                 LATDbits.LATD7
#define SRDY                              LATDbits.LATD6
#define SEN                                     PORTCbits.RC3

#define mask(byte, pos) ((byte) & ~(1<<(pos)))
#define get_bit(byte, pos) ((byte) & (1<<(pos)))

#define program_bit(byte, pos, low_high)    \
        byte = mask(byte, pos);             \
        if (low_high)                       \
            byte  = byte | (1<<pos);

#define UCHAR unsigned char

UCHAR flip_byte(UCHAR);
void low_isr(void);
void high_isr(void);
void DO_SRDY(void);

//ANT+ Functions
void ANT_AssignChannel();
void ANT_ChannelID();
void ANT_OpenChannel();
void ANT_SyncReset();
void ANT_SetNetworkKey(UCHAR netnumber, UCHAR key0, UCHAR key1, UCHAR key2, UCHAR key3,
UCHAR key4, UCHAR key5, UCHAR key6, UCHAR key7);
void ANT_ChannelRF_Frequency(UCHAR channumber, UCHAR frequency);

//GLOBAL VARIABLES
//SOME VARIABLES MAY BE UNNEEDED HERE
//OPTIMIZATION COULD BE USED FOR THE FLAG VARIABLES, INSTEAD OF USING ONE BYTE PER FLAG
UCHAR send_buffer[20];
UCHAR rx_buffer[20];
UCHAR msg_length = 0;
UCHAR flag_updateOK = 0x00;
UCHAR flag_responseWait = 0x00;
UCHAR counter = 0x00;
UCHAR counter_mfg = 0x00;
```

```
UCHAR counter_product = 0x00;

unsigned int inst_power = 0;
UCHAR inst_power_lsb;
UCHAR inst_power_msb;
unsigned int acc_power = 0;
UCHAR acc_power_lsb;
UCHAR acc_power_msb;

UCHAR switch_counter = 0x00;

//COEFFICIENTS FOUND BY MECHANICAL TEAM
unsigned int C1 = 6533;
unsigned int C2 = 1302;

float bridge_1_voltage = 0;
float bridge_2_voltage = 0;
float adc_test = 0;

#pragma code lowvector=0x18
void lowvector(void)
{
_asm goto low_isr _endasm
}
#pragma code

#pragma interruptlow low_isr
void low_isr(void)
{
        if(INTCONbits.RBIF == 1)
        {
                //one of the bits RB4 - RB7 have changed: find which one
                if(PORTBbits.RB4 == 0)
                {
                        //GREEN - ZERO POINT
                        PORTAbits.RA3 = 1;
                //      Delay10KTCYx(10);
                }
            else if (PORTBbits.RB5 == 0)
                {
                        PORTAbits.RA2 = 1;
                //      Delay10KTCYx(10);
                }

        INTCONbits.RBIF = 0; //Clear interrupt flag
        }
}

#pragma code highvector=0x08
void highvectorvector(void)
{
_asm goto high_isr _endasm
}
#pragma code

#pragma interruptlow high_isr
void high_isr(void)
{
        UCHAR result_byte;
        UCHAR j;

        if(INTCONbits.INT0IF == 1)
        {

                DO_SRDY();
                result_byte = ReadSPI();
                if(flip_byte(result_byte) == 0xA5)
                {
                        flag_updateOK = 0x00;
                        PORTAbits.RA1 = 0;
```

```
                        SMSGRDY = 1;
                        //we are sending to ANT
                        //pulse SRDY as many times as needed to clock all bytes to ANT+

                                for(j=0; j < msg_length; j++)
                                {
                                        DO_SRDY();
                                        WriteSPI(send_buffer[j]);
                                }
                                //do we wait for response?
                                if(flag_responseWait == 0x01)
                                {
                                        while(SEN);
                                        DO_SRDY();
                                        result_byte = ReadSPI();
                                        flag_responseWait = 0x00;
                                }
                }

                if(flip_byte(result_byte) == 0xA4)
                {
                        //data from ANT...read and do whatever...?
                        DO_SRDY();
                        msg_length = flip_byte(ReadSPI()); //# of following data bytes
                        for(j=0; j< msg_length+2; j++)
                        {
                                DO_SRDY();
                                rx_buffer[j] = flip_byte(ReadSPI());
                        }


                if(rx_buffer[0] == (0x40) && rx_buffer[1] == (0x00) && rx_buffer[2] ==
(0x01) && rx_buffer[3] == (0x03))
                        {
                                //EVENT_TX msg
                                //set OK to update data flag

                                flag_updateOK = 0x01;
                                PORTAbits.RA1 = 1;
                }
                }
        }


        INTCONbits.INT0IF = 0; //Clear interrupt flag
}

void DO_SRDY(void)
{
        SRDY = 1;
        Delay10TCYx(1);
        SRDY = 0;
        Delay10TCYx(2); //Delay > 2.5us
        SRDY = 1;
    return;
}

void ANT_SyncReset()
{
                //ANT+ SYNC
        SMSGRDY = 1;
        SRDY = 1;
        Delay10KTCYx(20);
        SRDY = 0;
        Delay1KTCYx(2);
        SMSGRDY = 0;
        // delay 45 microsec
        Delay10TCYx(20);
        SMSGRDY = 1;
}
```

```
void ANT_SetNetworkKey(UCHAR netnumber, UCHAR key0, UCHAR key1, UCHAR key2, UCHAR key3,
UCHAR key4, UCHAR key5, UCHAR key6, UCHAR key7)
{
        send_buffer[0] = flip_byte(0x09);
        send_buffer[1] = flip_byte(0x46);
        send_buffer[2] = flip_byte(netnumber);
        send_buffer[3] = flip_byte(key0);
        send_buffer[4] = flip_byte(key1);
        send_buffer[5] = flip_byte(key2);
        send_buffer[6] = flip_byte(key3);
        send_buffer[7] = flip_byte(key4);
        send_buffer[8] = flip_byte(key5);
        send_buffer[9] = flip_byte(key6);
        send_buffer[10] = flip_byte(key7);
        send_buffer[11] = flip_byte(0xA5 ^ 0x09 ^ 0x46 ^ netnumber ^ key0 ^
key1^key2^key3^key4^key5^key6^key7);
        msg_length = 12;
        flag_responseWait = 0x01;
        SMSGRDY = 0;
}

//REFER TO ANT+ DOCUMENTION FOR INFORMATION ON THESE COMMAND MESSAGES
void ANT_AssignChannel()
{
        //ASSIGN CHANNEL COMMAND
        send_buffer[0] = flip_byte(0x03);
        send_buffer[1] = flip_byte(0x42);
        send_buffer[2] = flip_byte(0x00);
        send_buffer[3] = flip_byte(0x10);
        send_buffer[4] = flip_byte(0x00);
        send_buffer[5] = flip_byte(0xA5 ^ 0x03 ^ 0x42 ^ 0x00 ^ 0x10 ^ 0x00);
        msg_length = 6;
        flag_responseWait = 0x01;
        SMSGRDY = 0;
}

void ANT_ChannelID()
{
        //CHANNEL ID
        send_buffer[0] = flip_byte(0x05);
        send_buffer[1] = flip_byte(0x51);
        send_buffer[2] = 0x00; //channel number

`       send_buffer[3] = flip_byte(0x01); //\\\\\\\\\\\\\\\\\\\\ DEVICE ID
        send_buffer[4] = flip_byte(0x00); /// - little endian

        send_buffer[5] = flip_byte(0x0B); // pairing bit + device type ID
        send_buffer[6] = flip_byte(0x05);  // transmission type
        send_buffer[7] = flip_byte(0xA5^ 0x05 ^ 0x51 ^0x00 ^0x01 ^0x00 ^0x0B^0x05);
        msg_length = 8;
        flag_responseWait = 0x01;
        SMSGRDY = 0;
}

void ANT_OpenChannel()
{
        //OPEN CHANNEL
        send_buffer[0] = flip_byte(0x01);
        send_buffer[1] = flip_byte(0x4B);
        send_buffer[2] = flip_byte(0x00);
        send_buffer[3] = flip_byte(0xA5 ^ 0x01 ^ 0x4B ^ 0x00);
        msg_length = 4;
        flag_responseWait = 0x01;
        SMSGRDY = 0;
}

void ANT_ChannelRF_Frequency(UCHAR channumber, UCHAR frequency)
{
        send_buffer[0] = flip_byte(0x02);
        send_buffer[1] = flip_byte(0x45);
        send_buffer[2] = flip_byte(channumber);
```

```
        send_buffer[3] = flip_byte(frequency);
        send_buffer[4] = flip_byte(0xA5 ^0x02 ^0x45 ^channumber ^ frequency);
        msg_length = 5;
        flag_responseWait = 0x01;
        SMSGRDY = 0;
}

UCHAR flip_byte(UCHAR byte)
{
    UCHAR i, j;
    UCHAR bit_i, bit_j;

    for(i=0, j=7; i<j; i++, j--)
    {
        bit_i = get_bit(byte, i);
        bit_j = get_bit(byte, j);
        program_bit(byte, i, bit_j);
        program_bit(byte, j, bit_i);
    }

    return (byte);
}

void main()
{
        //CONFIGURE DIRECTION BITS
        TRISA = 0x00;
        PORTA = 0x00;
        TRISB = 0x00;
        PORTB = 0x00;
        TRISD = 0x00;

        TRISCbits.TRISC3 = 1; //SEN
        TRISDbits.TRISD2 = 1; //SERIAL DATA IN
        TRISDbits.TRISD3 = 1; //SERIAL CLOCK IN
        TRISDbits.TRISD1 = 0; //SERIAL DATA OUT
        TRISCbits.TRISC6 = 1; //SS pin
        TRISCbits.TRISC5 = 1;

        TRISBbits.TRISB5 = 1;
        TRISBbits.TRISB4 = 1;

        TRISAbits.TRISA0 = 1;
        TRISAbits.TRISA2 = 0;
        TRISAbits.TRISA3 = 0;

        ANSEL0  = 0x01;                      //RA0 Analog Input
        ADCON0  = 0b00000001;  //single-shot conversion on ADC group A
        ADCON1  = 0b00001110;  //Use AVdd and AVss for references, 1110 for A/D pin select
        ADCON2  = 0b10100110;  //R-justified, 20TAD aquisition time, TAD=64*TOSC

        //INTERRUPT CONFIGURATION
        RCONbits.IPEN = 1;
        INTCONbits.INT0IF = 0;
        INTCON = 0b11011000;
        INTCON2 = 0b00000000;

        //WE USE SS PIN, BUT IT IS ALWAYS GROUNDED
        //OPEN SPI PORT
        OpenSPI(SLV_SSON, MODE_11, SMPMID);

        ANT_SyncReset();
        //SET UP FOR BIKE POWER METER CONFIGURATION - THIS INFORMATION IS FOUND IN ANT+
DEVICE PROFILES DATASHEET
        ANT_SetNetworkKey(0, 0xB9, 0xA5, 0x21, 0xFB, 0xBD, 0x72, 0xC3, 0x45);
        //THE ORDER OF OPERATIONS HERE IS IMPORTANT
        ANT_AssignChannel();
`       ANT_ChannelID();
        ANT_ChannelRF_Frequency(0, 57);
        ANT_OpenChannel();
```

```
	while(1)
	{
		//SWITCH 1 ON THEN TAKE A/D CONVERSION
		PORTB = 0xFE;
		ADCON0bits.GO_DONE=1;		//Start A/D conversion process
		while(ADCON0bits.GO_DONE){}	//wait for process to finish
		bridge_1_voltage = (.0029)*((unsigned int)ADRESH*256 + (unsigned
int)ADRESL);
		Delay10KTCYx(50);
		//SWITCH 2 ON THEN TAKE A/D CONVERSION
		PORTB = 0xFD;
		ADCON0bits.GO_DONE=1;		//Start A/D conversion process
		while(ADCON0bits.GO_DONE){}	//wait for process to finish
		bridge_2_voltage = (.0029)*((unsigned int)ADRESH*256 + (unsigned
int)ADRESL);
		Delay10KTCYx(50);

		//CALCULATE VOLTAGE DIFFERENCE FROM OFFSET FOR EACH BRIDGE
		bridge_1_voltage = fabs(1.5 - bridge_1_voltage);
		bridge_2_voltage = fabs(1.5 - bridge_2_voltage);

		//ROUGH ESTIMATE TO CALCULATE HP FROM TORQUE; WILL NEED TO BE REDONE
		adc_test = .1*(C1*bridge_1_voltage + C2*bridge_2_voltage);

		if(flag_updateOK == 0x01)
		{
			counter_mfg++;

			//RESET DISPLAY LEDS (DEMONSTRATION ONLY)
			PORTAbits.RA2=0;
			PORTAbits.RA3=0;

			//PER ANT DOCUMENTATION, MFG AND PRODUCT DATA PAGES MUST BE
INTERLEAVED 121,242 MSGS RESPECTIVELY
			if(counter_mfg == 121)
			{
				send_buffer[0] = flip_byte(0x09);
				send_buffer[1] = flip_byte(0x4E);
				send_buffer[2] = flip_byte(0x00);
				//start data packet
				send_buffer[3] = flip_byte(0x50);
				send_buffer[4] = flip_byte(0x00);
				send_buffer[5] = flip_byte(0x00);
				send_buffer[6] = flip_byte(0x01);
				send_buffer[7] = flip_byte(0x01);
				send_buffer[8] = flip_byte(0x00);
				send_buffer[9] = flip_byte(0x01);
				send_buffer[10] = flip_byte(0x00);
				//end data packet
				send_buffer[11] = flip_byte(0xA5 ^ 0x09 ^ 0x4E ^
0x00^0x50^0x00^0x00^0x01^0x01^0x00^0x01^0x00);
				msg_length = 12;
				flag_updateOK = 0x00;
				SMSGRDY = 0;
				///Send mfg page
			}

			if(counter_mfg =242)
			{
				counter_mfg = 0x00;
				send_buffer[0] = flip_byte(0x09);
				send_buffer[1] = flip_byte(0x4E);
				send_buffer[2] = flip_byte(0x00);
				//start data packet
				send_buffer[3] = flip_byte(0x51);
				send_buffer[4] = flip_byte(0x00);
				send_buffer[5] = flip_byte(0x00);
				send_buffer[6] = flip_byte(0x02);
				send_buffer[7] = flip_byte(0xFF);
				send_buffer[8] = flip_byte(0xFF);
				send_buffer[9] = flip_byte(0xFF);
```

```
                                send_buffer[10] = flip_byte(0xFF);
                                //end data packet
                                send_buffer[11] = flip_byte(0xA5 ^ 0x09 ^ 0x4E ^ 0x00 ^
0x51^0x00^0x00^0x02^0xFF^0xFF^0xFF^0xFF);
                                msg_length = 12;
                                flag_updateOK = 0x00;
                                SMSGRDY = 0;
                                //Send prod page
                        }
                        if(counter_mfg != 121 || counter_mfg != 242)
                        {
                                counter++;
                                //RPM = 45
                                //JANKY FUDGE FACTOR; JUST FOR DEMO
                                adc_test = adc_test - 37;


                                acc_power = acc_power + adc_test;

                                acc_power_lsb = acc_power%256;
                                acc_power_msb = acc_power/256;
                                inst_power_lsb = (unsigned int)adc_test%256;
                                inst_power_msb = (unsigned int)adc_test/256;

                                send_buffer[0] = flip_byte(0x09);
                                send_buffer[1] = flip_byte(0x4E);
                                send_buffer[2] = flip_byte(0x00);
                                //start data packet
                                send_buffer[3] = flip_byte(0x10);
                                send_buffer[4] = flip_byte(counter);
                                send_buffer[5] = flip_byte(0xFF);
                                send_buffer[6] = flip_byte(0x3C);
                                send_buffer[7] = flip_byte(acc_power_lsb);
                                send_buffer[8] = flip_byte(acc_power_msb);
                                send_buffer[9] = flip_byte(inst_power_lsb);
                                send_buffer[10] = flip_byte(inst_power_msb);
                                //end data packet
                                send_buffer[11] = flip_byte(0xA5 ^ 0x09 ^ 0x4E ^ 0x00 ^
0x10 ^ counter ^ 0xFF ^0x3C ^acc_power_lsb ^acc_power_msb ^inst_power_lsb
^inst_power_msb);
                                msg_length = 12;
                                flag_updateOK = 0x00;
                                SMSGRDY = 0;
                        }
                }
        }
}
```