SYNAPSE - UAV

Dariel Marlow – darielmarlow@hotmail.com
Michael DeLisi – delisi@eng.utah.edu

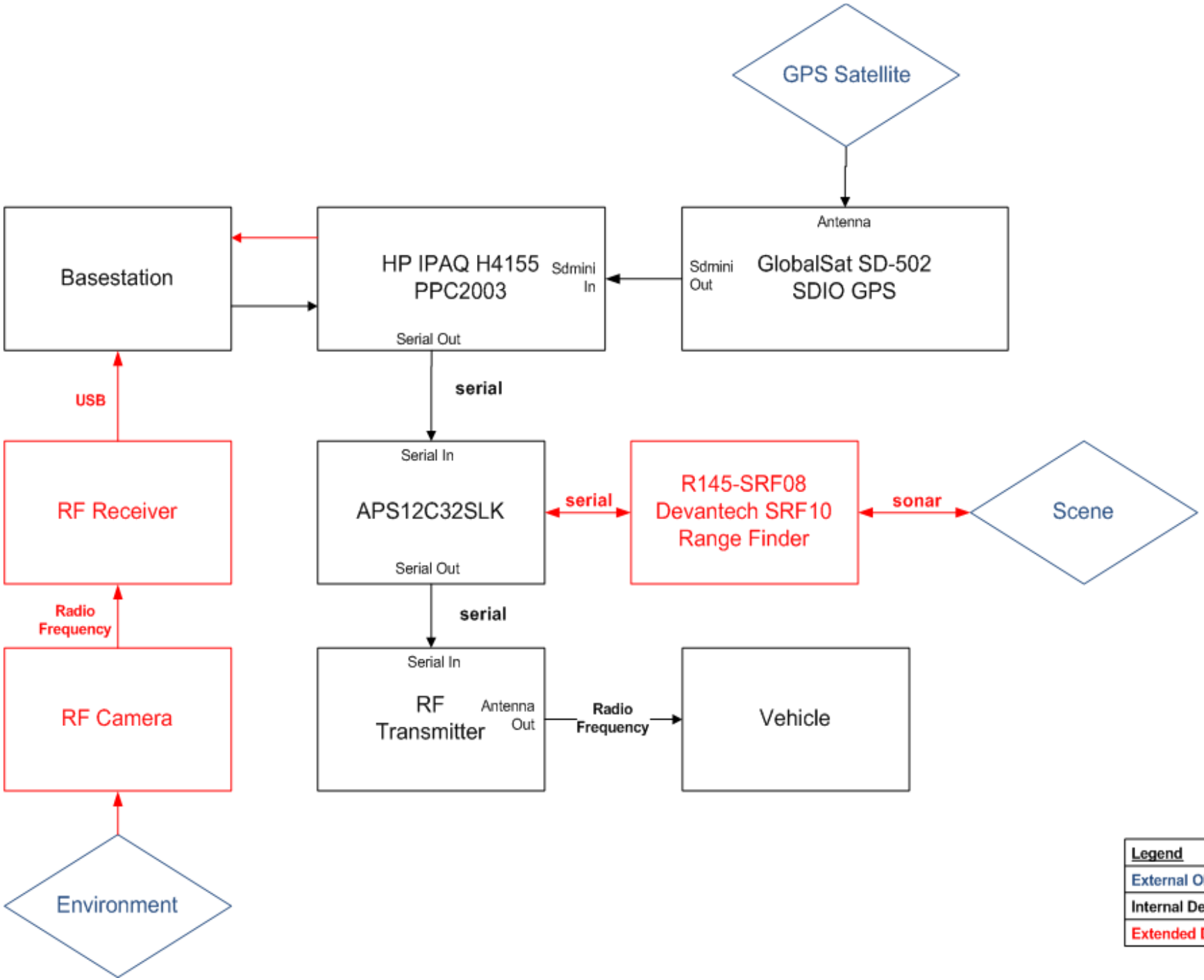
Toren Monson – nicmonson@gmail.com
Matt Stoker – matt.stoker@gmail.com

www.cs.utah.edu/~delisi/cs3992

Introduction

The Synapse UAV is an unmanned autonomous vehicle that is composed of several devices. The motivation behind this project is to learn about autonomous vehicles guided by GPS technology.

- Vehicle
- Guidance Controller
- Pocket PC & GPS Unit
- Base Station & Camera


GPS Satellite
Antenna
Basestation
HP IPAQ H4155
PPC2003
Sdmini In
Serial Out
Sdmini Out
GlobalSat SD-502
SDIO GPS
USB
serial
RF Receiver
Serial In
APS12C32SLK
Serial Out
serial
R145-SRF08
Devantech SRF10
Range Finder
sonar
Scene
Radio Frequency
serial
RF Camera
Serial In
RF Transmitter
Antenna Out
Radio Frequency
Vehicle
Environment
Legend
External Objects
Internal Devices
Extended Devices

VEHICLE

Vehicle

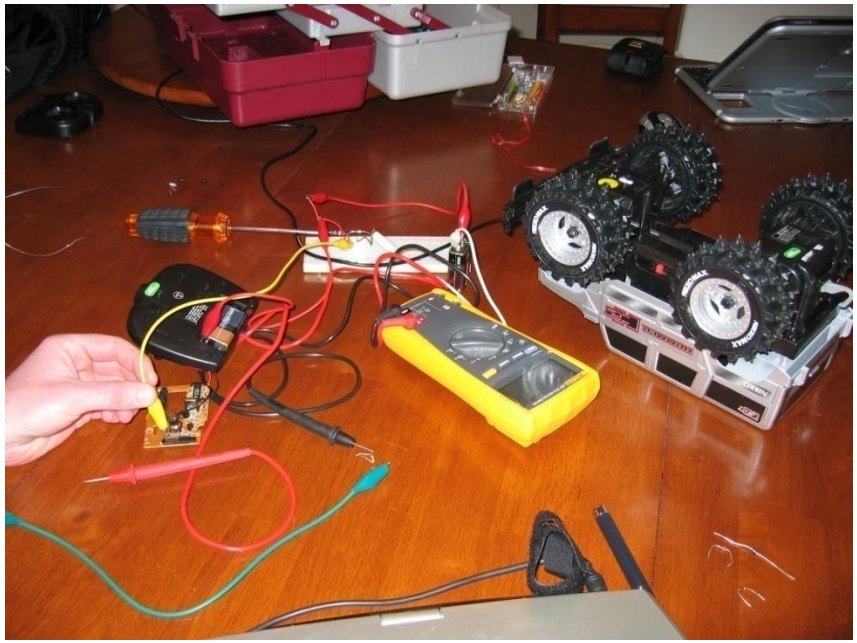
The Synapse UAV control system will be designed so any RF controlled vehicle can be used by loading a compatible software package onto the PocketPC. Our implementation vehicle will be an RF remote controlled car with forward/back and right/left controls.

Tasks:

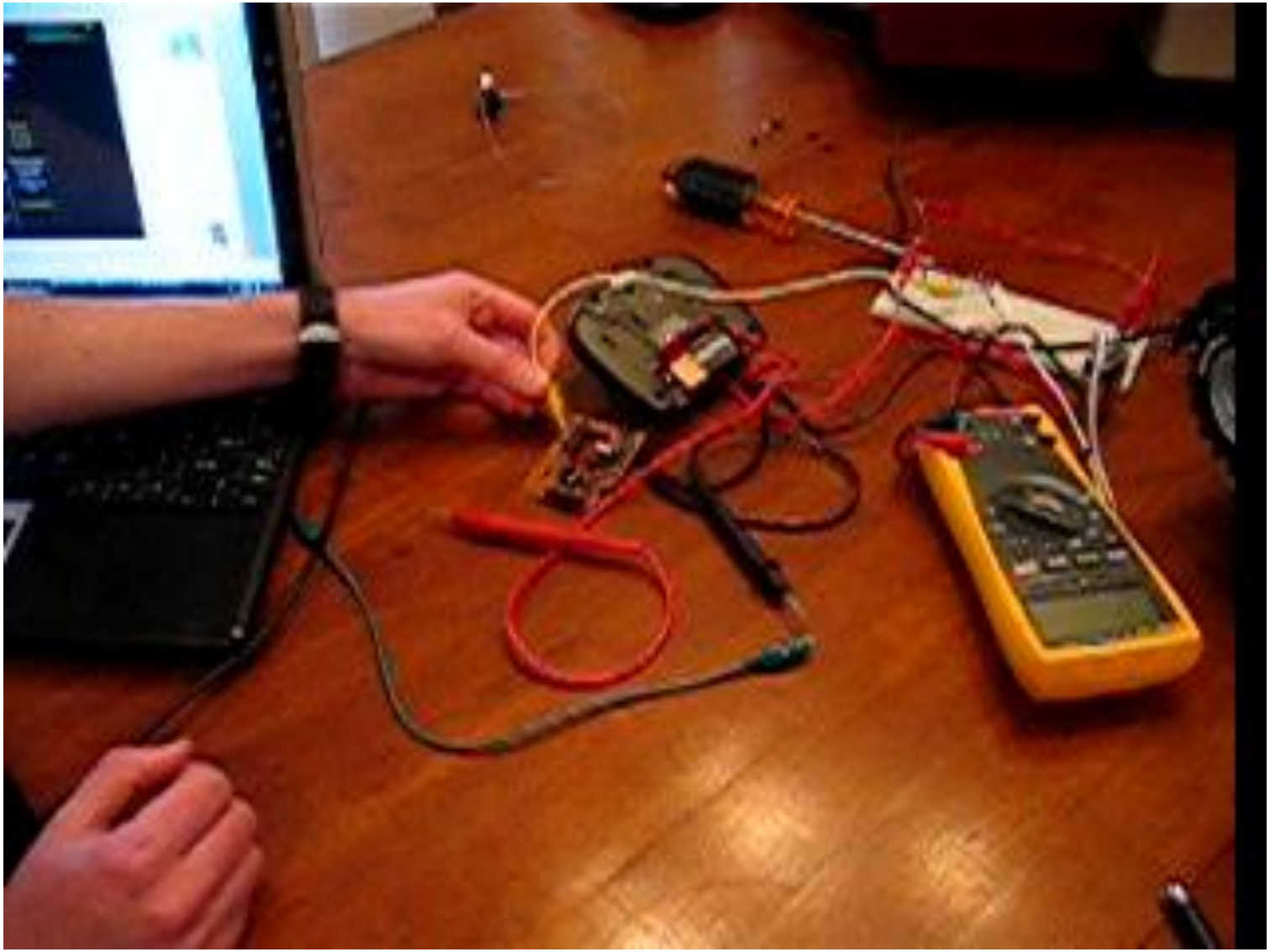
- Determine how the motor system will interface with guidance controller
- Define a software API flexible enough to admit several motor control systems.

Vehicle

The team has already begun to test methods of integrating the RF controllers that come with the vehicles into the rest of the system. The integration should be done with little or no permanent modification of the vehicle or it's controller.



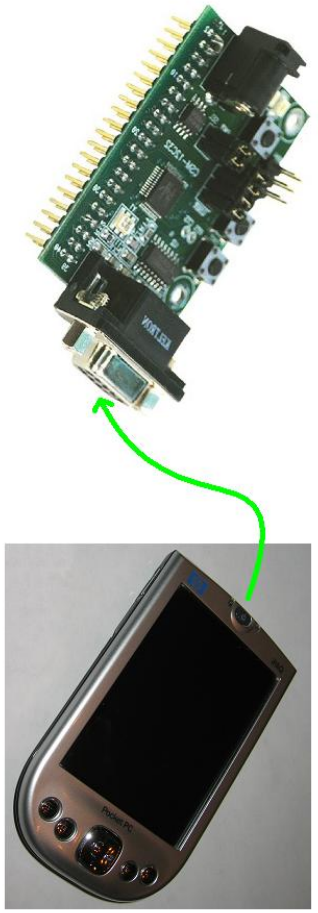
Vehicle



GUIDANCE CONTROLLER

Guidance Controller

The guidance controller interfaces between the motors and the Pocket PC. It takes in a serial input from the Pocket PC, interprets the data, and sends out signals to the vehicle's motors. The signal to the motors could either be an RF signal to take advantage of the current motor setup or we could bypass the RF system and send an analog signal directly to the motors. The guidance controller will be composed of a Motorola M9S12C32 microcontroller with an integrated serial interface. The M9S12C32 was provided free of charge by FreeScale Corporation.



Tasks:

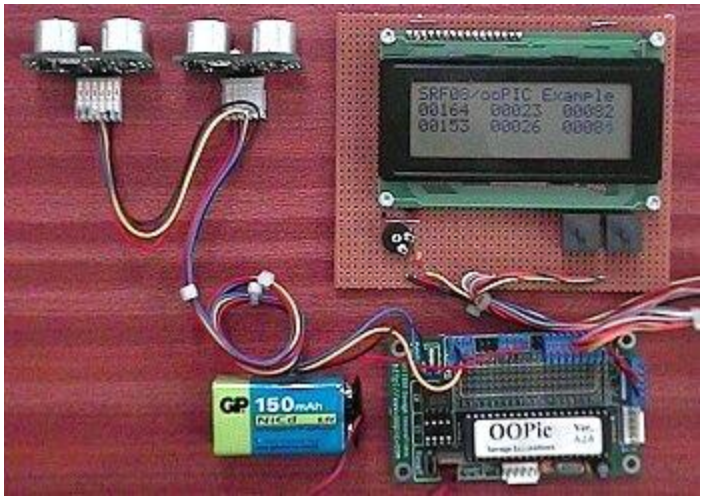
- Combine serial interface and microcontroller and come up with messaging standard.

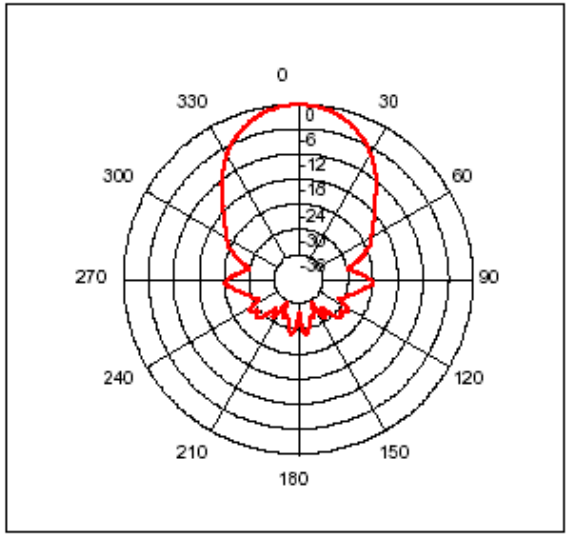
Guidance Controller

SRF08 – Devantech

High Performance Ultrasonic Range Finder.

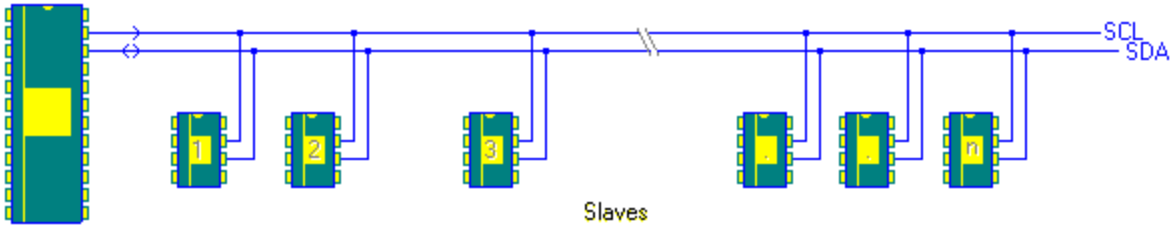
I²C Protocol
65mS
6 meter range (up to 11 meters)
Light Sensor

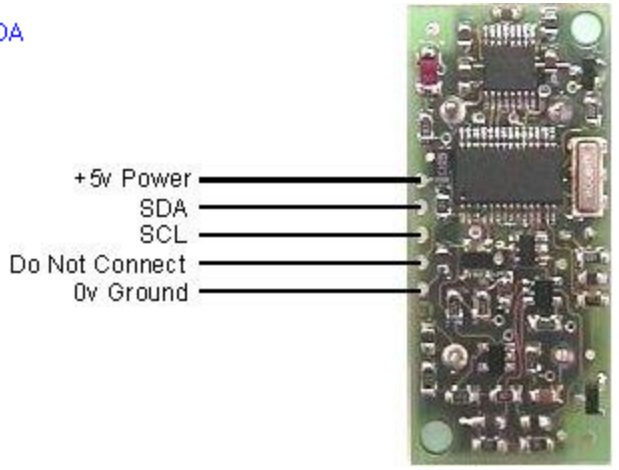




Guidance Controller

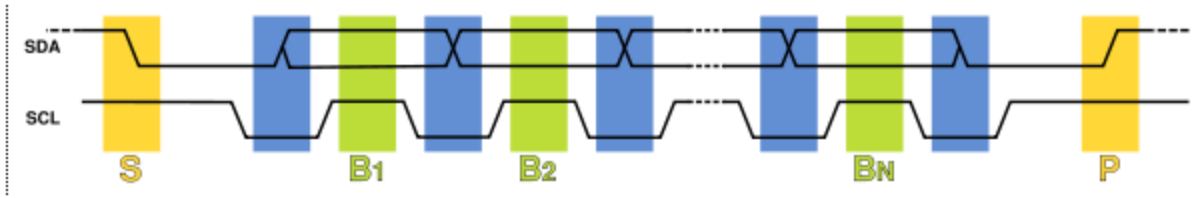
I²C Communication Protocol





Steps

1. Send start info and begin clock on SCL
2. Send address a certain slave device (sonar)
3. Send bit for send or receive and get an acknowledge back
4. Send/receive 8 bit data on SDA and get an acknowledge back
5. Go to step 3 or send end info



Guidance Controller

Command		Action
Decimal	Hex	
80	0x50	Ranging Mode - Result in inches
81	0x51	Ranging Mode - Result in centimeters
82	0x52	Ranging Mode - Result in micro-seconds
83	0x53	ANN Mode - Result in inches
84	0x54	ANN Mode - Result in centimeters
85	0x55	ANN Mode - Result in micro-seconds
160	0xA0	1st in sequence to change I2C address
165	0xA5	3rd in sequence to change I2C address
170	0xAA	2nd in sequence to change I2C address

Input & Output Sonar Bits

Location	Read	Write
0	Software Revision	Command Register
1	Light Sensor	Max Gain Register (default 31)
2	1st Echo High Byte	Range Register (default 255)
3	1st Echo Low Byte	N/A
~~	~~~~	~~~~
34	17th Echo High Byte	N/A
35	17th Echo Low Byte	N/A

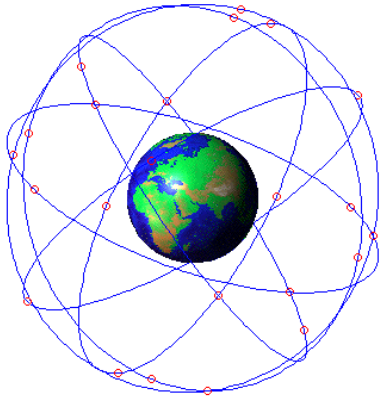
Steps

1. To initiate a ranging, the microcontroller sends a command 0x51.
2. The microcontroller sends a read to the sonar and will repeatedly read oxFF until the sonar is done ranging and sends out the distance in centimeters.
3. The microcontroller reads the data and stores it another location for other parts of the microcontroller to use
4. Go to 1.

POCKET PC & GPS UNIT

Pocket PC & GPS Unit

The Pocket PC is used to make calculations based on a predefined path by using the data from the attached GPS unit. It will not only store the main program, but also the travel path of the UAV. The SD port will be used to connect the GPS unit to the Pocket PC and also provide power to it. Once it receives this data, the Pocket PC will then be able to calculate the path that the UAV must travel. It will relay this this information to the guidance controller via serial interface.



Pocket PC & GPS Unit

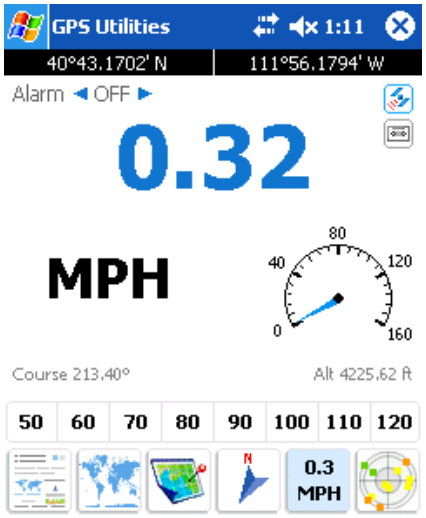
The GPS data will be parsed with the use of GeoFrameworks GPS utilities. We will use Visual Studio and C# to write all of the Pocket PC software.

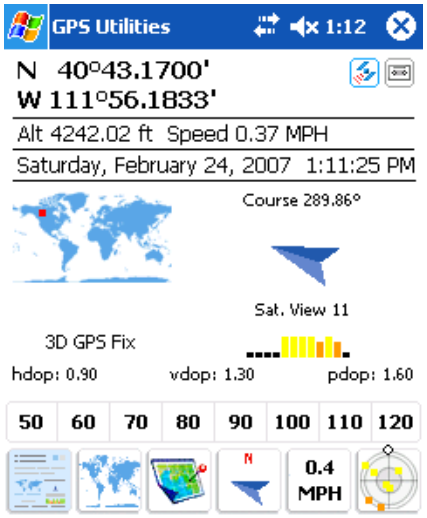
The Pocket PC comes from one of the group members.

The GPS unit has been procured through an online vendor.

Tasks:

- Write main application to calculate direction based on GPS information





C#

```
// Creates a new instance using the specified value and unit type.
Speed Speed1 = new Speed(
          // value (System.Double)
          1.0,
          // units (SpeedUnit)
          SpeedUnit2);

// Create a new Speed object.
Speed Speed2 = new Speed(
          // value (System.String)
          "value");
```

Pocket PC & GPS Unit

$GPRMC,POS_UTC,POS_STAT,LAT,LAT_REF,LON,LON_REF,SPD,HDG,DATE,MAG_VAR,MAG_REF*CC<cr><lf>

$GPGLL,LAT,LAT_REF,LONG,LONG_REF,POS_UTC,POS_STAT*CC<cr><lf>

$GPGGA,POS_UTC,LAT,LAT_REF,LONG,LONG_REF,FIX_MODE,SAT_USED,HDOP,ALT,ALT_UNIT,GEO,G_UNIT,D_AGE,D_REF*CC<cr><lf>

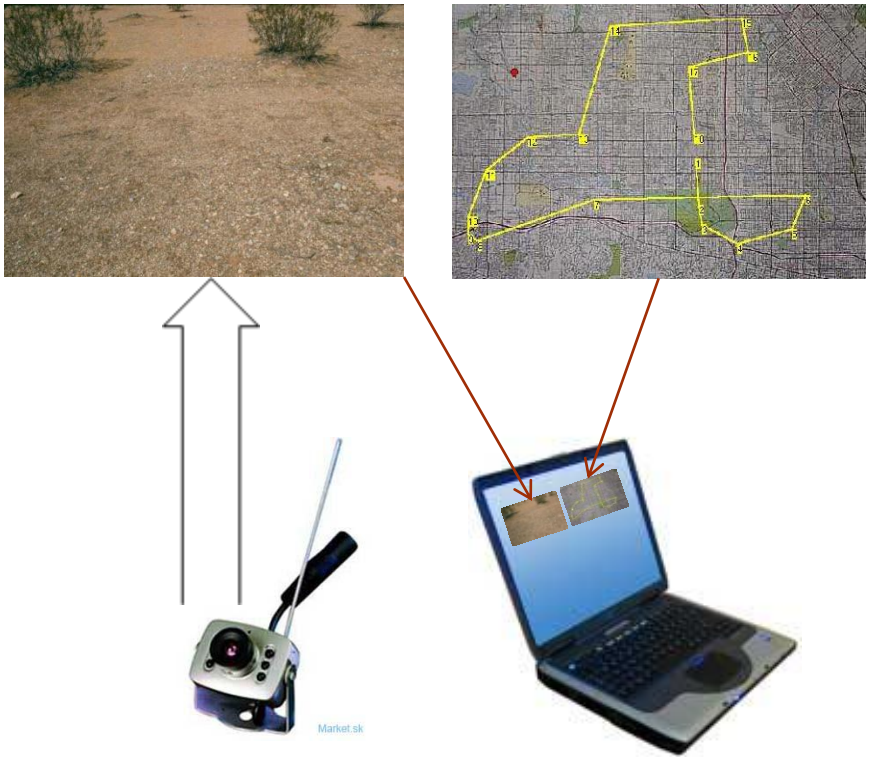
POS_UTC - UTC of position. Hours, minutes and seconds [fraction (opt.)]. (hhmmss[.fff])
POS_STAT - Position status. (A = Data valid, V = Data invalid)
LAT - Latitude (llll.ll)
LAT_REF - Latitude direction. (N = North, S = South)
LON - Longitude (yyyyy.yy)
LON_REF - Longitude direction (E = East, W = West)
SPD - Speed over ground. (knots) (x.x)
HDG - Heading/track made good (degrees True) (x.x)
DATE - Date (ddmmyy)
MAG_VAR - Magnetic variation (degrees) (x.x)
MAG_REF - Magnetic variation (E = East, W = West)
FIX_MODE - Position Fix Mode (o = Invalid, >o = Valid)
SAT_USED - Number Satellites used in solution
HDOP - Horizontal Dilution of Precision

BASE STATION & CAMERA

Base Station & Camera

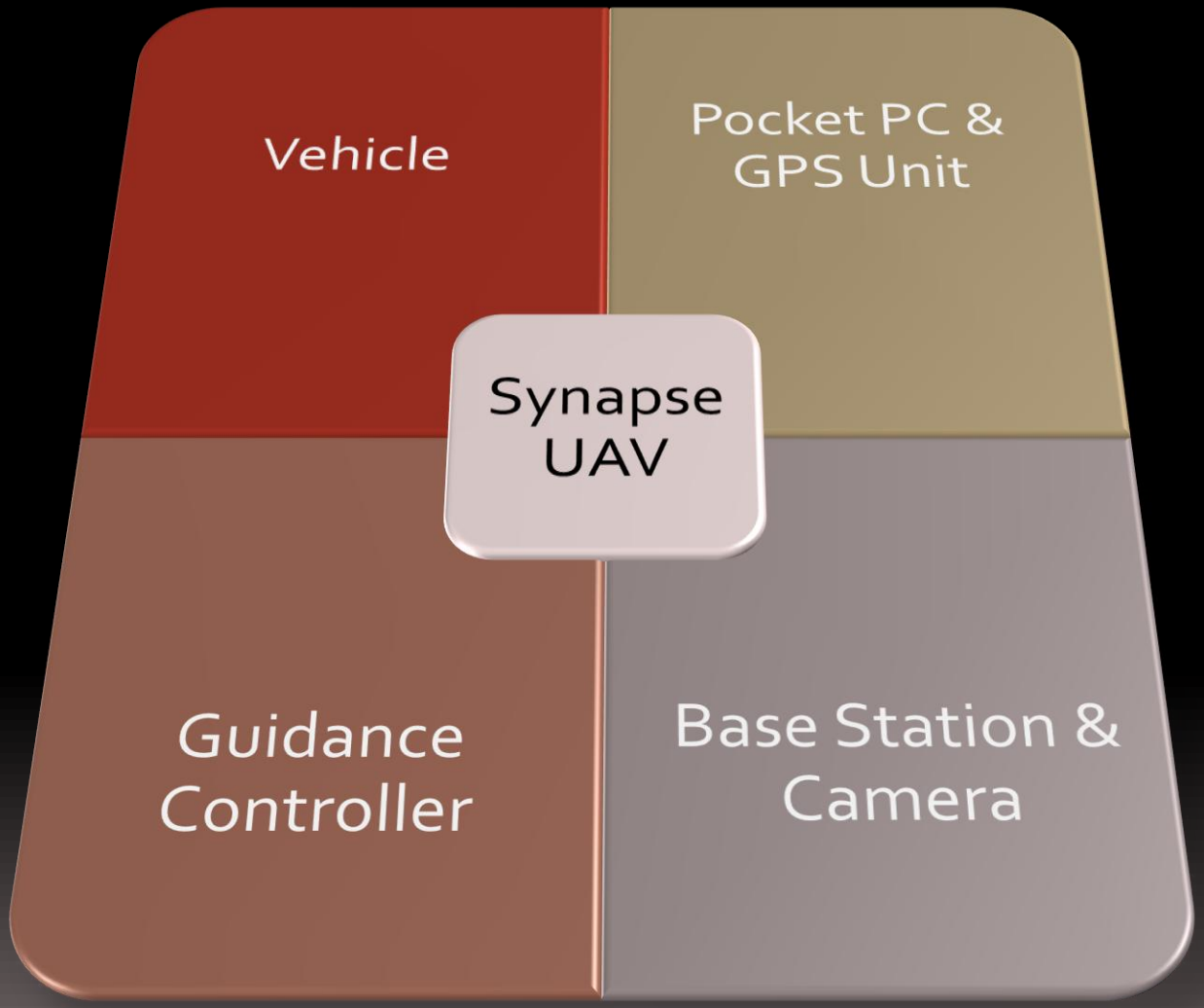
The base station software will also be written in C# and run from a laptop. The travel path of the UAV will be displayed on the base station along with any other relevant data, such as GPS coordinates, altitude, etc. An RF camera on the vehicle with an RF receiver on the laptop is also being considered

The RF receiver for the laptop, along with the camera to be mounted onto the vehicle, will need to be purchased.



Tasks:

- Write base station application with video display module from RF receiver.


Vehicle
Pocket PC & GPS Unit
Synapse UAV
Guidance Controller
Base Station & Camera