Autonomous Water Quality Sampler (AWQuSam)

Project Proposal
Florida State University Department of Oceanography

December 2, 2011

Agenda

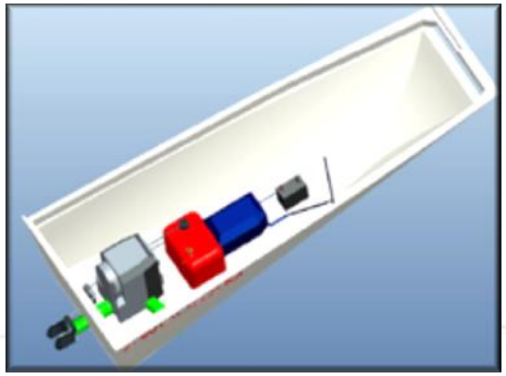
Friday, December 2, 2011

- Project Overview
- Data Acquisition
- Data Logging
- Data Handling
- Data Transmission
- Base Station Receiver
- Navigation
- Propulsion
- Steering
- Housing
- Financial Update
- Project Schedule

Project Overview

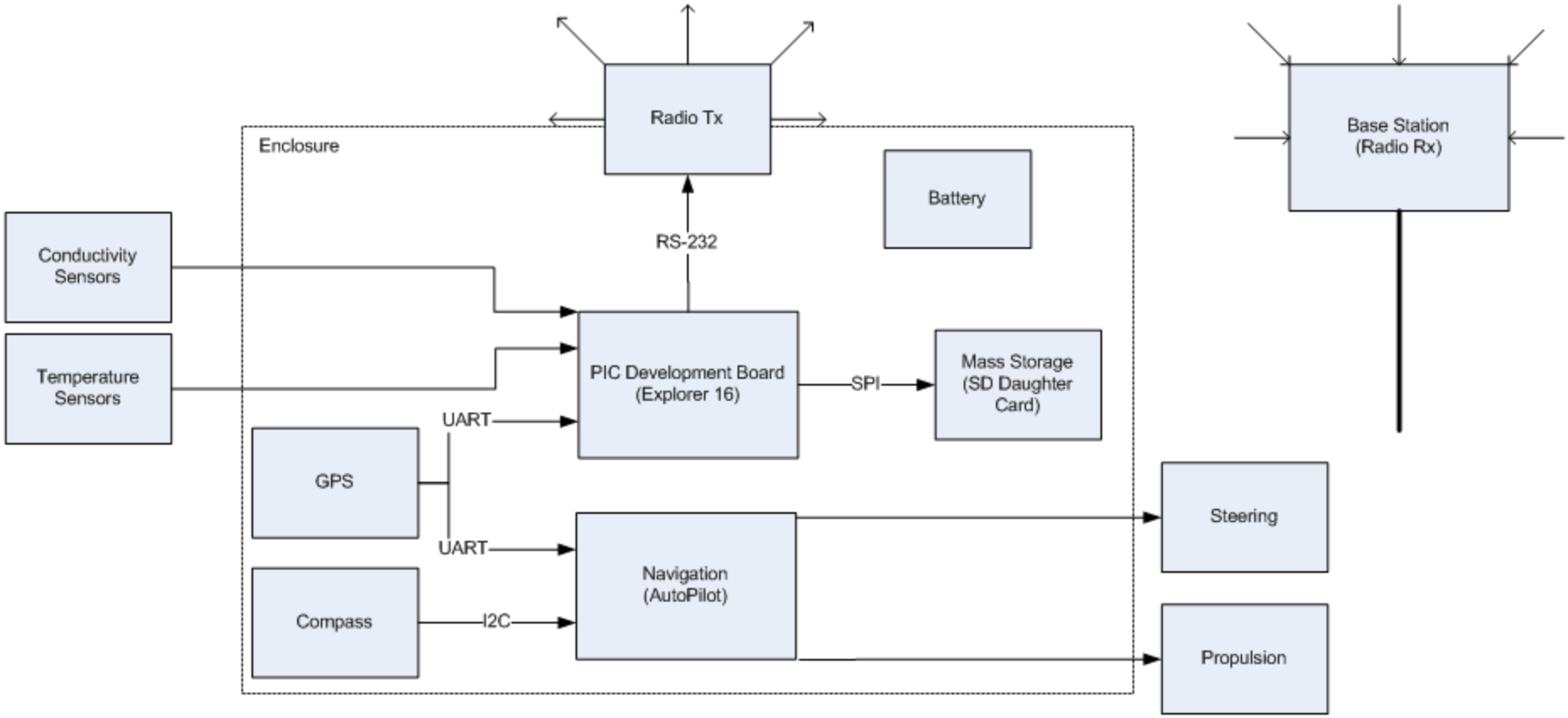
Brad Wells

Problem Description



- Gather Water Quality / Hydrographic Data
- Florida Shelf
  - Shallow Environment

Design Solution

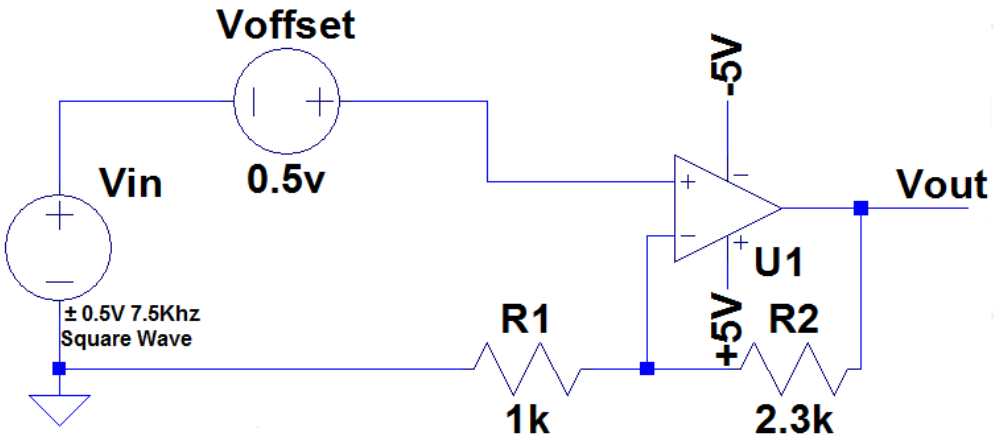


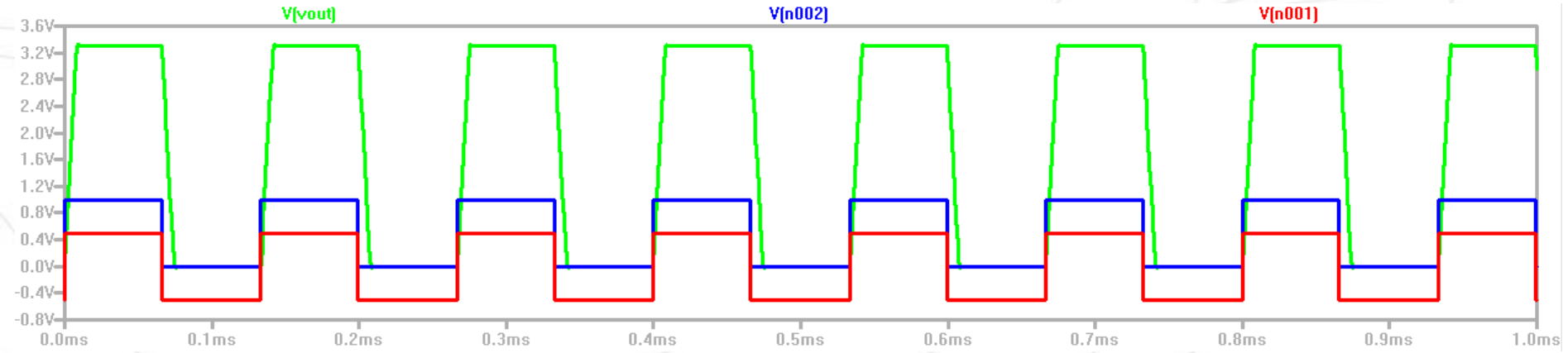
Data Acquisition

Brad Wells

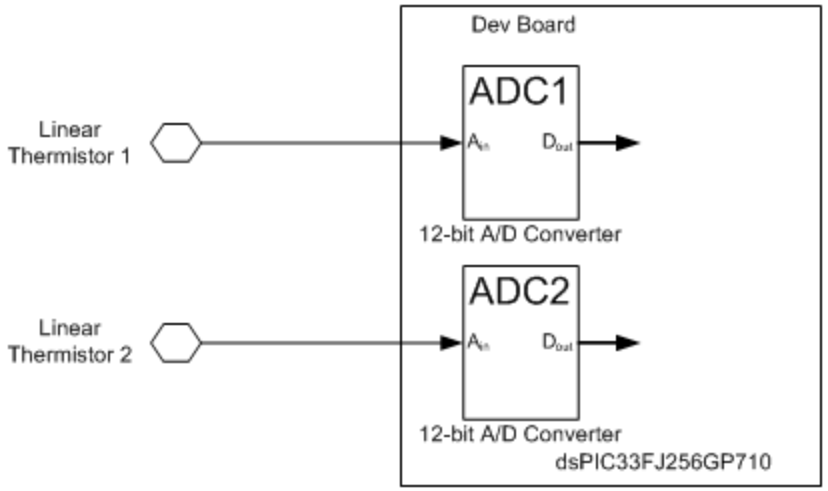
Conductivity Sensors

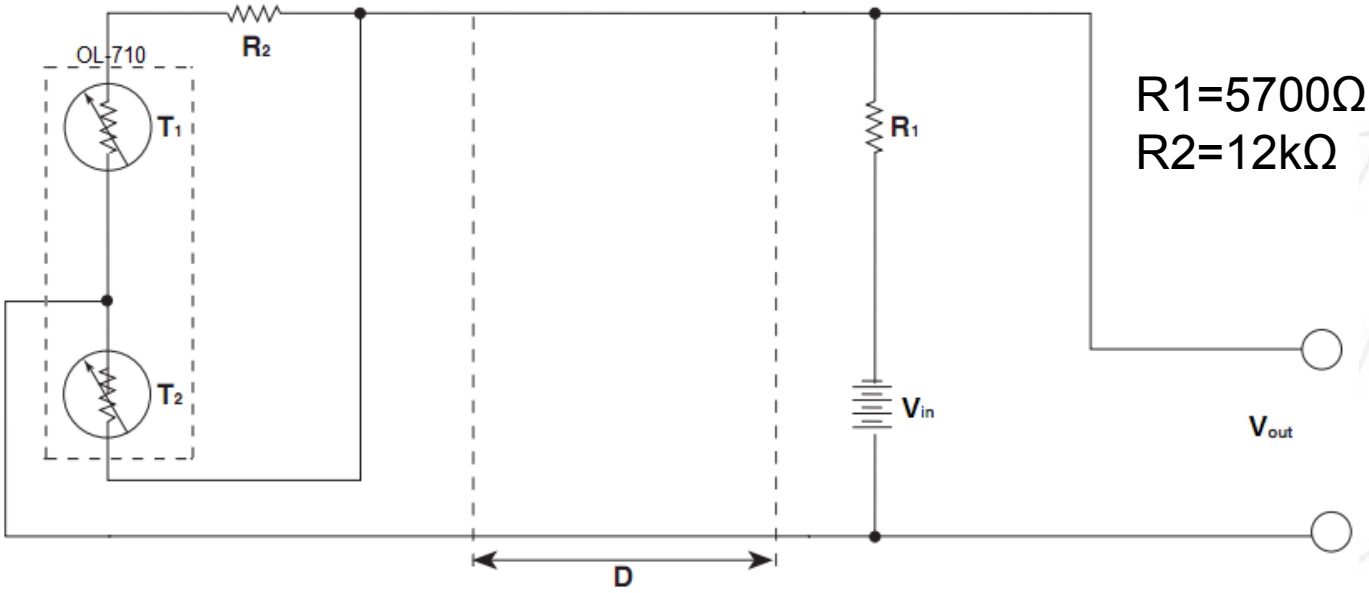
$$V_{out} = V_{in}(1 + \frac{R_2}{R_1})$$





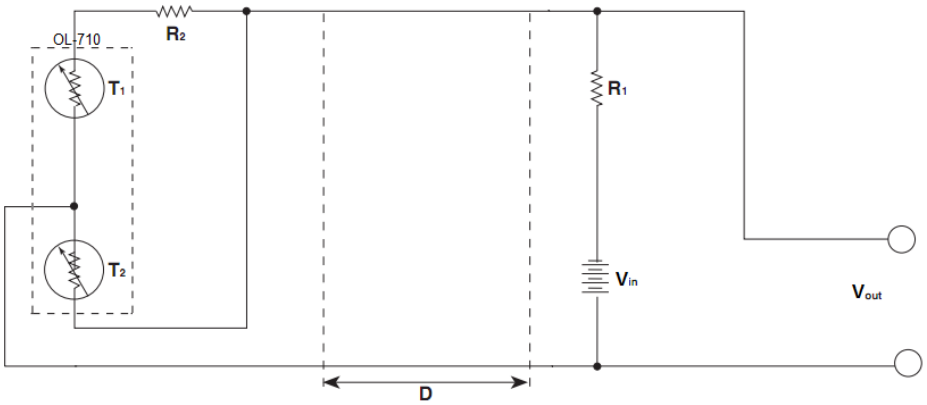
Temperature Sensors

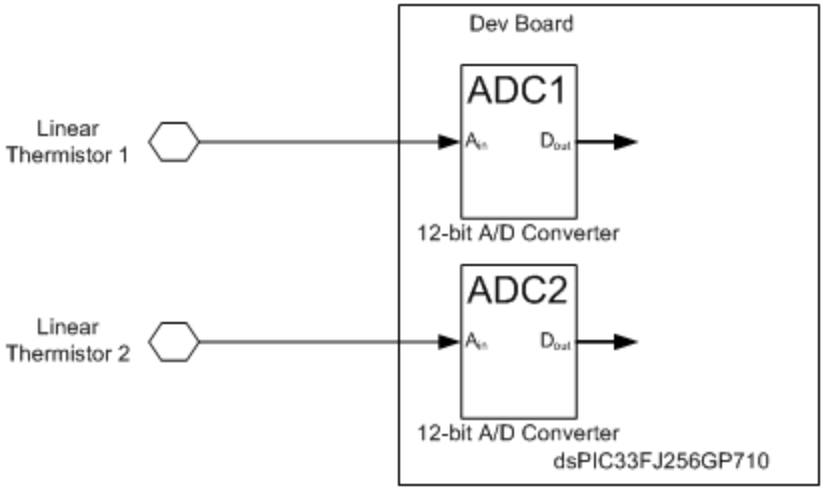


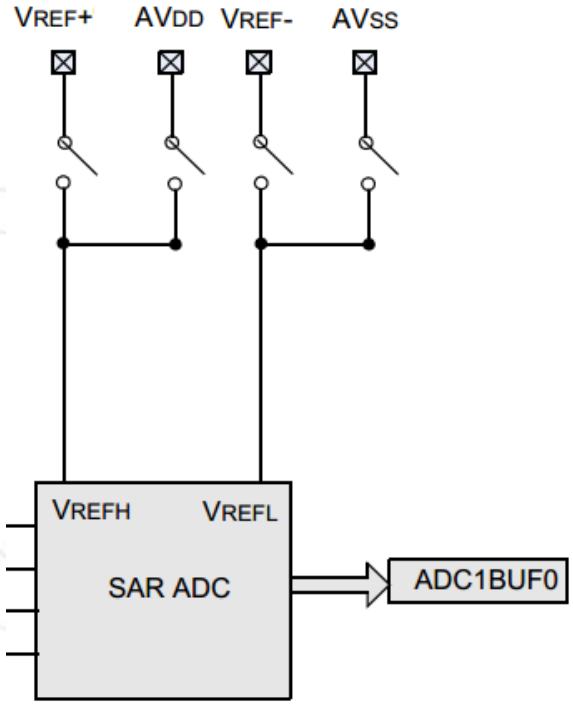


R1=5700Ω
R2=12kΩ

Temperature Sensors







$$V_{out} = (-0.0056846 \cdot V_{in})\,T + 0.805858 \cdot V_{in}$$

0°C	40°C
2.65933 14V	1.90896 42V

Data Logging And Handling

Francisco Schroeder

Data Logging

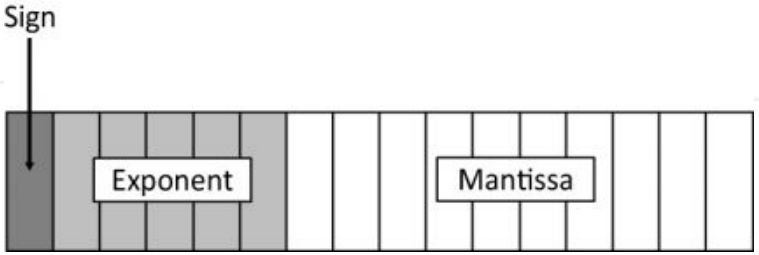
2-2 Dimensional Arrays
- 6 * 345600 (double)
- 2 temperature sensors
- Conductivity
- Latitude and Longitude

- 3 * 345600 (integer)
- Seconds
- Minutes
- Hours

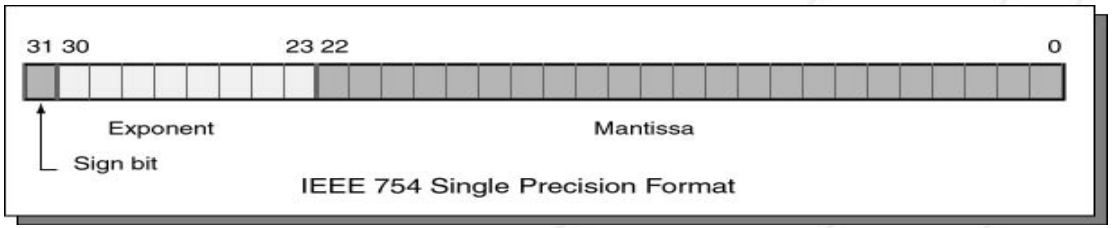
$$\frac{Samples}{Mission} = \frac{8\ Samples}{Second} \times \frac{60\ Seconds}{Minute} \times \frac{60\ Minutes}{Hour} \times \frac{12\ Hours}{Mission}$$

Data Handling

•**Purpose:** Convert Data from array into a bit stream.
•**Length:** 145 Bits
• 64 bits: half-precision IEEE-754 floating point standard
•Temperature and Conductivity sensors.



• 64 bits: single-precision IEEE-754 floating point standard
•Latitude and Longitude



• 17 bits: unsigned binary equivalent
•12 bits: minutes and seconds
•5 bits: hours

Data Handling **Pseudocode**

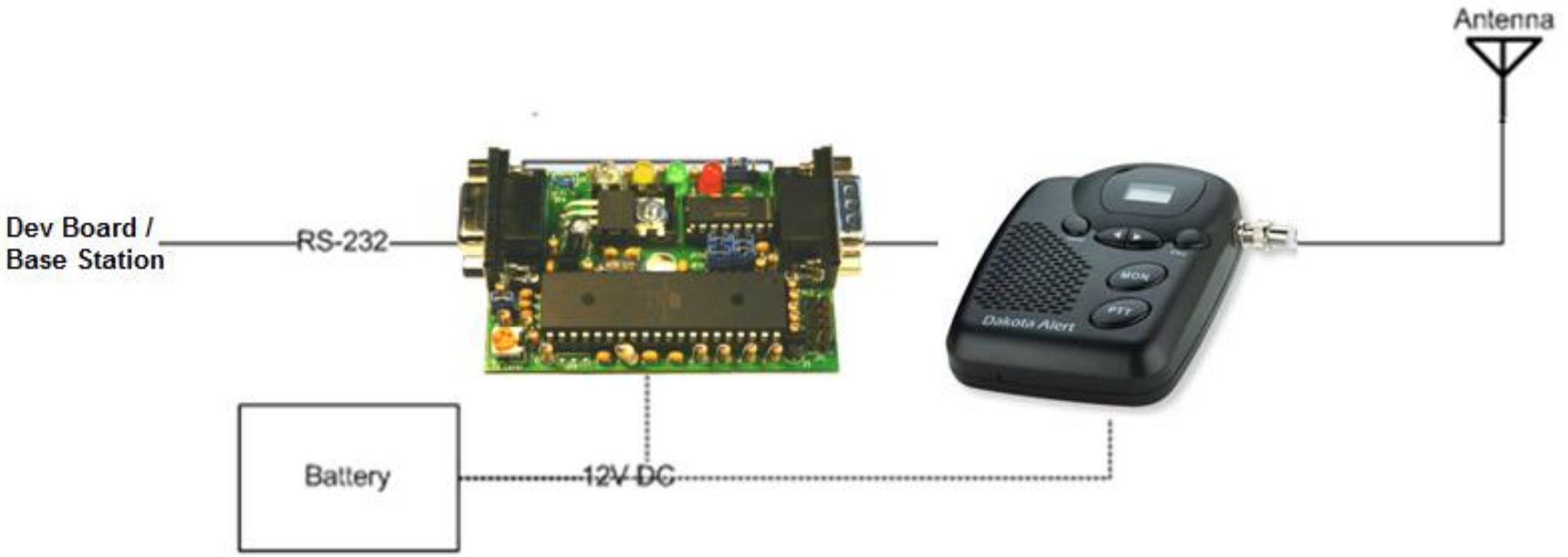
```
typedef struct
{
unsigned int bit : 1;
} Bit;

void main (double c1, double c2, double t2,
double t2, double lat, double long, int h, int m, int s)
{
Bit bitstream[145], con1[16], con2[16], tem1[16], tem2[16],
latitude[32], longitude[32], hour[5], minute[6], second[6];

con1 = half(c1);
con2 = half(c2);
tem1 = half(t1);
tem2 = half(t2);
latitude = single(lat);
longitude = single(long);
hour = integer(h);
minute = integer(m);
second = integer(s);
```

```
for (int i =0; i<145: i++)
{
    if (i < 16)
        bitstream[i] = con1[i];
    else if (i <32)
        bitstream[i] = con2[i-16];
    else if (i <48)
        bitstream[i] = tem1[i-32];
    else if (i <64)
        bitstream[i] = tem1[i-48];
    else if (i<96)
        bitstream[i] = latitude[i-64];
    else if (i <128)
        bitstream[i] = longitude[i-96];
    else if (i <133)
        bitstream[i] = hour[i-128];
    else if (i <139)
        bitstream[i] = minute[i-133];
    else
        bitstream[i] = second[i - 139];
}
}
```

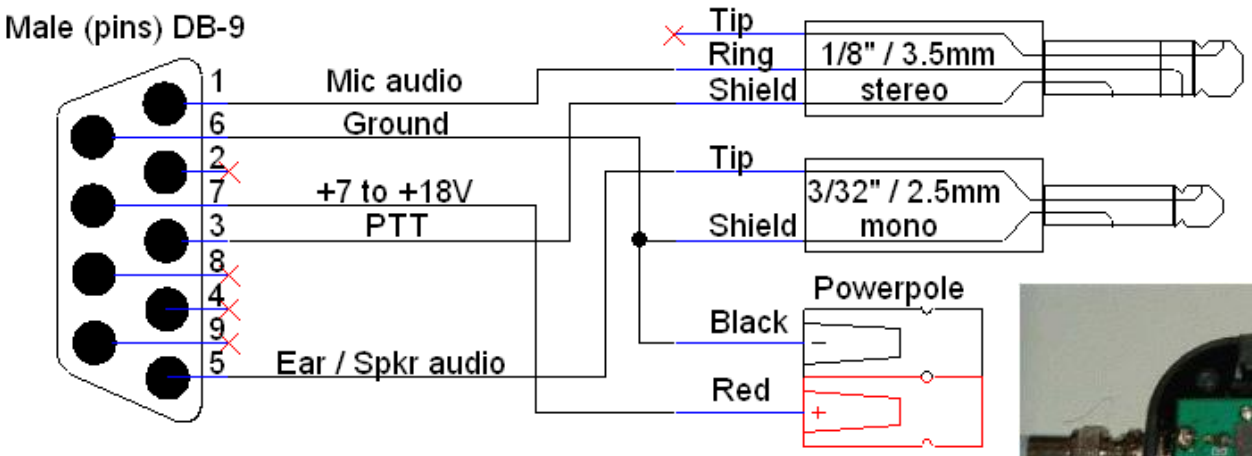
Data Transmission

Francisco Schroeder

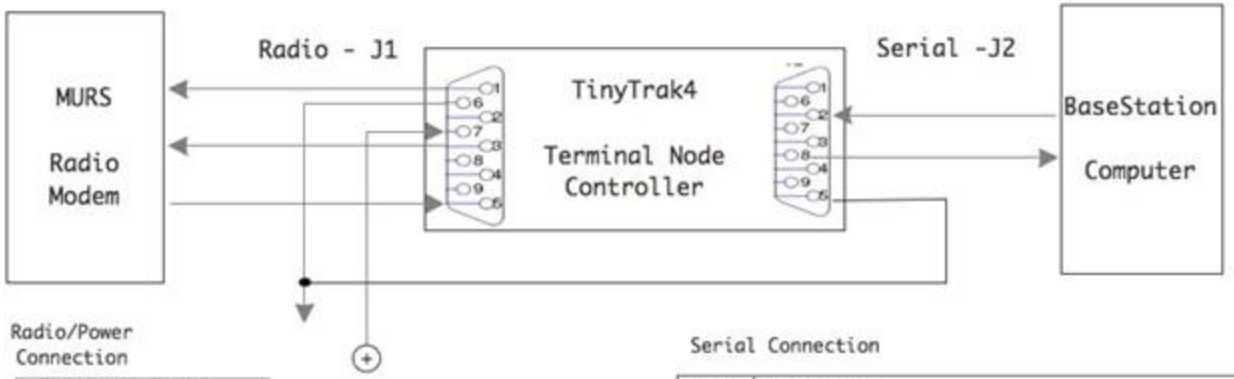
Data Transmission



Transmission Integration



Transmission Integration

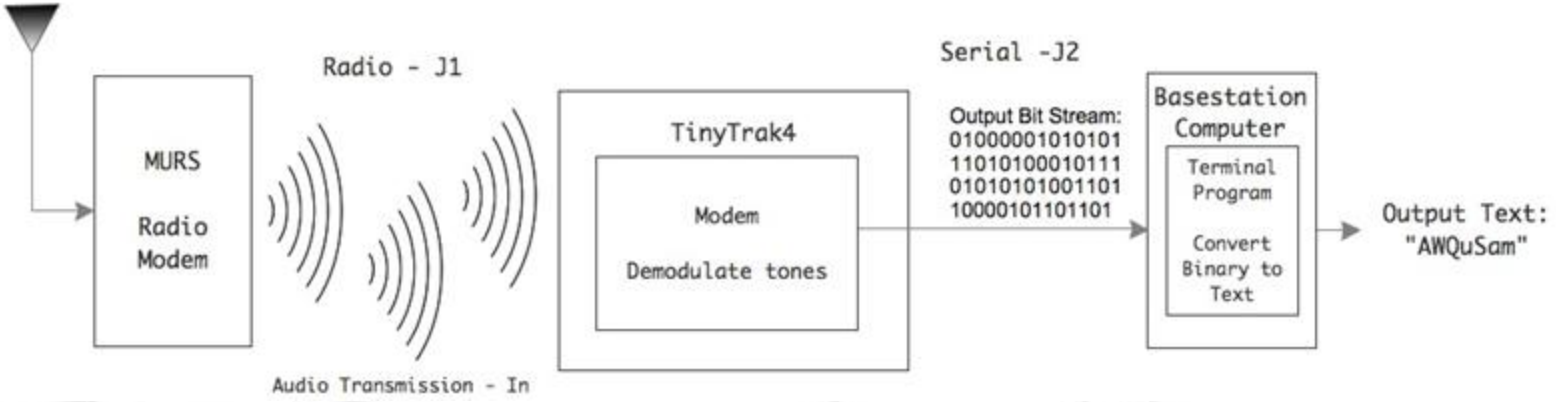


Radio/Power Connection

Pin	Function
1	Audio out
2	Carrier Detect
3	PTT Out
4	JP1
5	Audio in
6	Ground
7	Power In
8	PTT In
9	No connection

Serial Connection

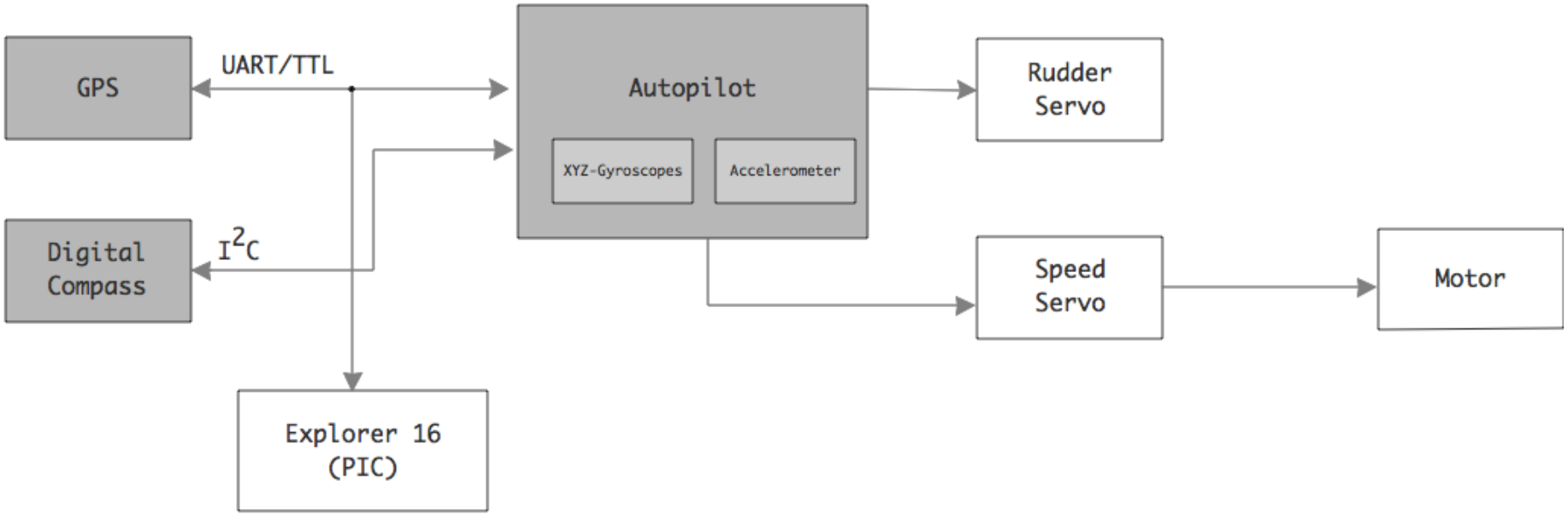
PIN	Function
1	No Connection
2	Primary Serial data in from a GPS or computer
3	Primary Serial data out to a GPS or computer
4	Power out for GPS (Vin or 5V), or alternate power input
5	Ground
6	No Connection
7	Secondary Serial data out to a GPS or computer
8	Secondary Serial data in from a GPS or computer
9	No Connection

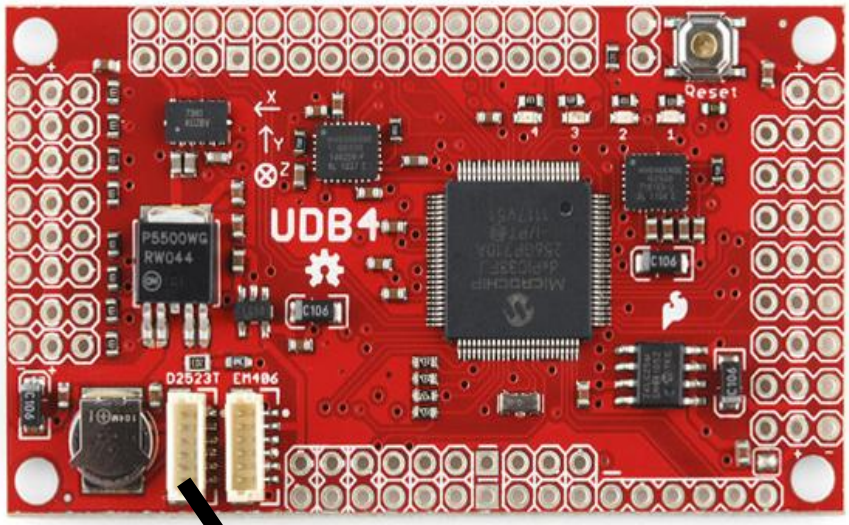


Navigation

Triesha Fagan

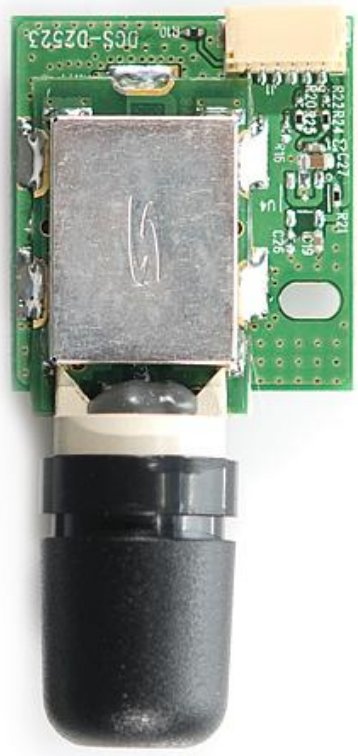
Top-level Architecture of Navigation System:

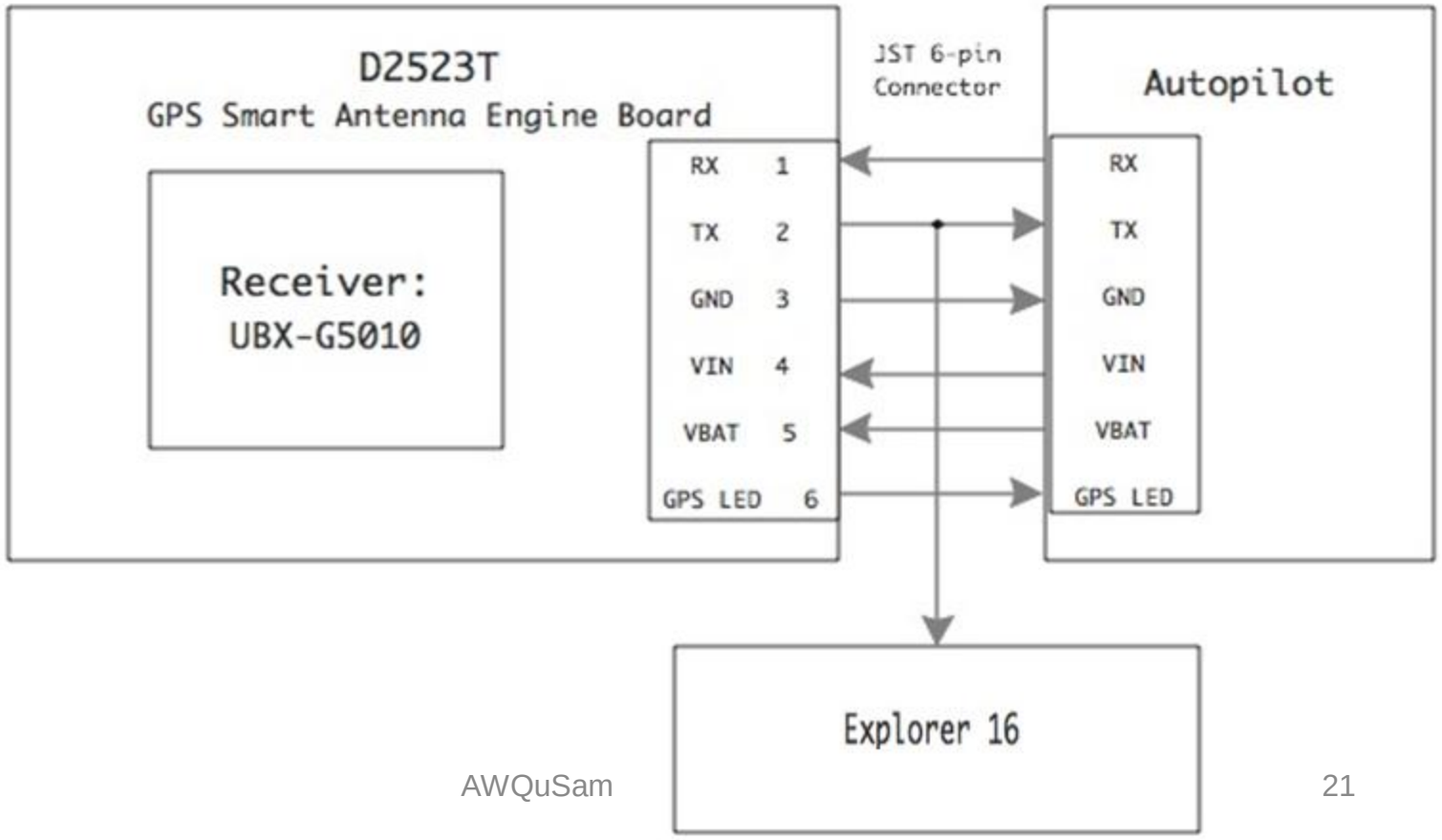




Integrate:
Autopilot and GPS module

Interface:
6-pin JST cable





GPS NMEA: $GPRMC

$GPRMC, hhmmss.ss, A, llll.ll, a, yyyyy.yy, a, x.x, x.x, ddmmyy ,x.x ,a ,*hh

Data to be parsed from GPRMC string:

- Latitude position
- Longitude
- Speed of Ground in Knots
- True Course

Name	Example Data	Description
~~Sentence Header~~	~~$GPGGA~~	~~Global Positioning System Fix Data~~
~~UTC of Position~~	~~hhmmss~~	~~Time Stamp~~
~~Data Status~~	~~A~~	
Latitude position	llll.ll	Degrees, decimal minutes
N or S	a	
Longitude	yyyyy.yy	Degrees, decimal minutes
E or W	a	
Speed of Ground in Knots	x.x	
True Course	x.x	
~~UTC Date~~	~~ddmmyy~~	~~Date Stamp~~
~~Magnetic Variation Degrees~~	~~x.x~~	~~Easterly variation subtracts from true course~~
~~E or W~~	~~a~~	
Checksum	*hh	

GPS NMEA: $GPGGA

$GPGGA, hhmmss.ss, llll.ll, a, yyyyy.yy, a, x, xx, x.x, x.x, M, x.x, M, x.x, xxxx, *hh

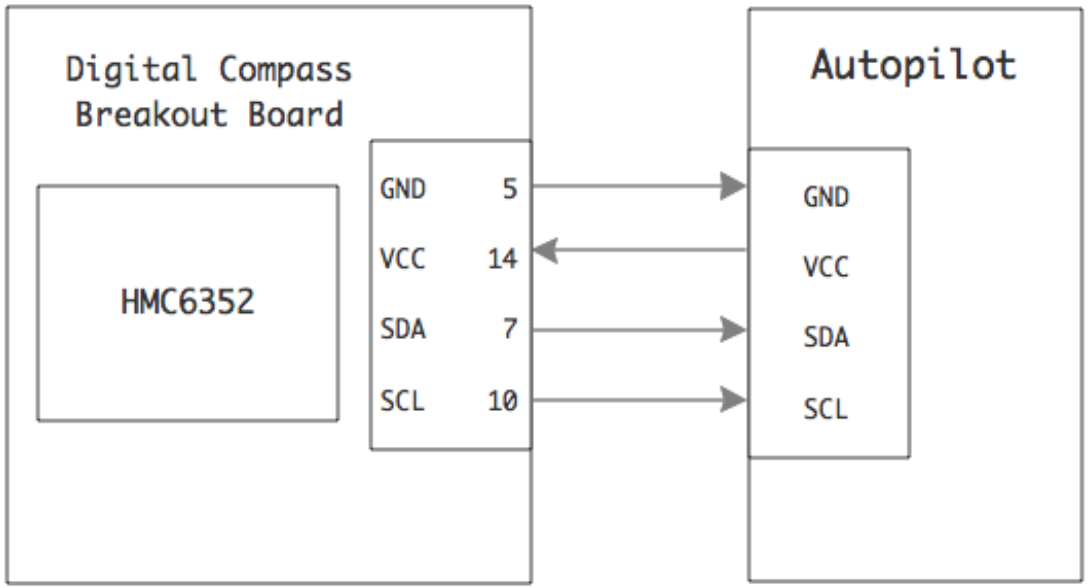
Data to be parsed from GPGGA string:

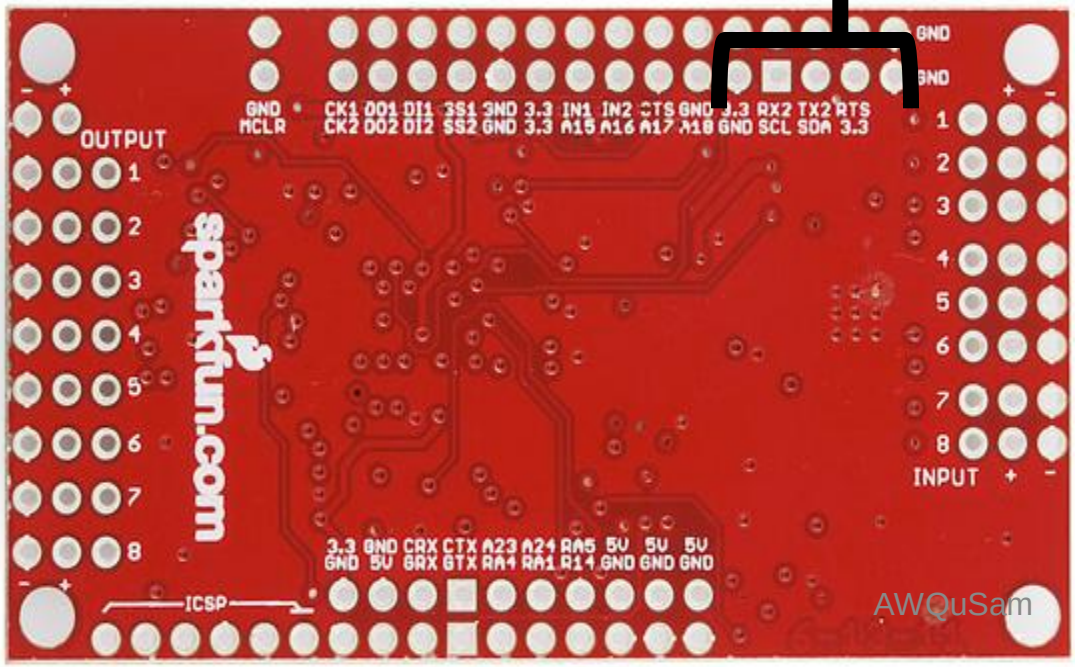
- ❖GPS Quality Indicator
- ❖Altitude

Name	Example Data	Description
~~Sentence Header~~	~~$GPGGA~~	~~Global Positioning System Fix Data~~
~~UTC of Position~~	~~hhmmss~~	~~Time Stamp~~
~~Latitude position~~	~~llll.ll~~	~~Degrees, decimal minutes~~
~~N or S~~	~~a~~	
~~Longitude~~	~~yyyyy.yy~~	~~Degrees, decimal minutes~~
~~E or W~~	~~a~~	
GPS Quality Indicator: **- 0 = Invalid** **- 1 = GPS fix** **- 2 = DGPS fix**	x	- 0 = Invalid - 1 = GPS fix - 2 = DGPS fix
~~Number of Satellites in Use~~	~~xx~~	
~~Horizontal Dilution of Precision (HDOP)~~	~~x.x~~	~~Relative accuracy of horizontal position~~
Altitude	x.x	Antenna altitude above mean-sea-level
~~Units of antenna altitude, meters~~	~~M~~	
~~Geoidal separation~~	~~x.x~~	
~~units of geoidal separation, meters~~	~~M~~	
~~Age of Differential GPS data (seconds)~~	~~x.x~~	~~Age in seconds since last update from diff. reference station~~
~~Diff. reference station ID#~~	~~xxxx~~	
Checksum	*hh	

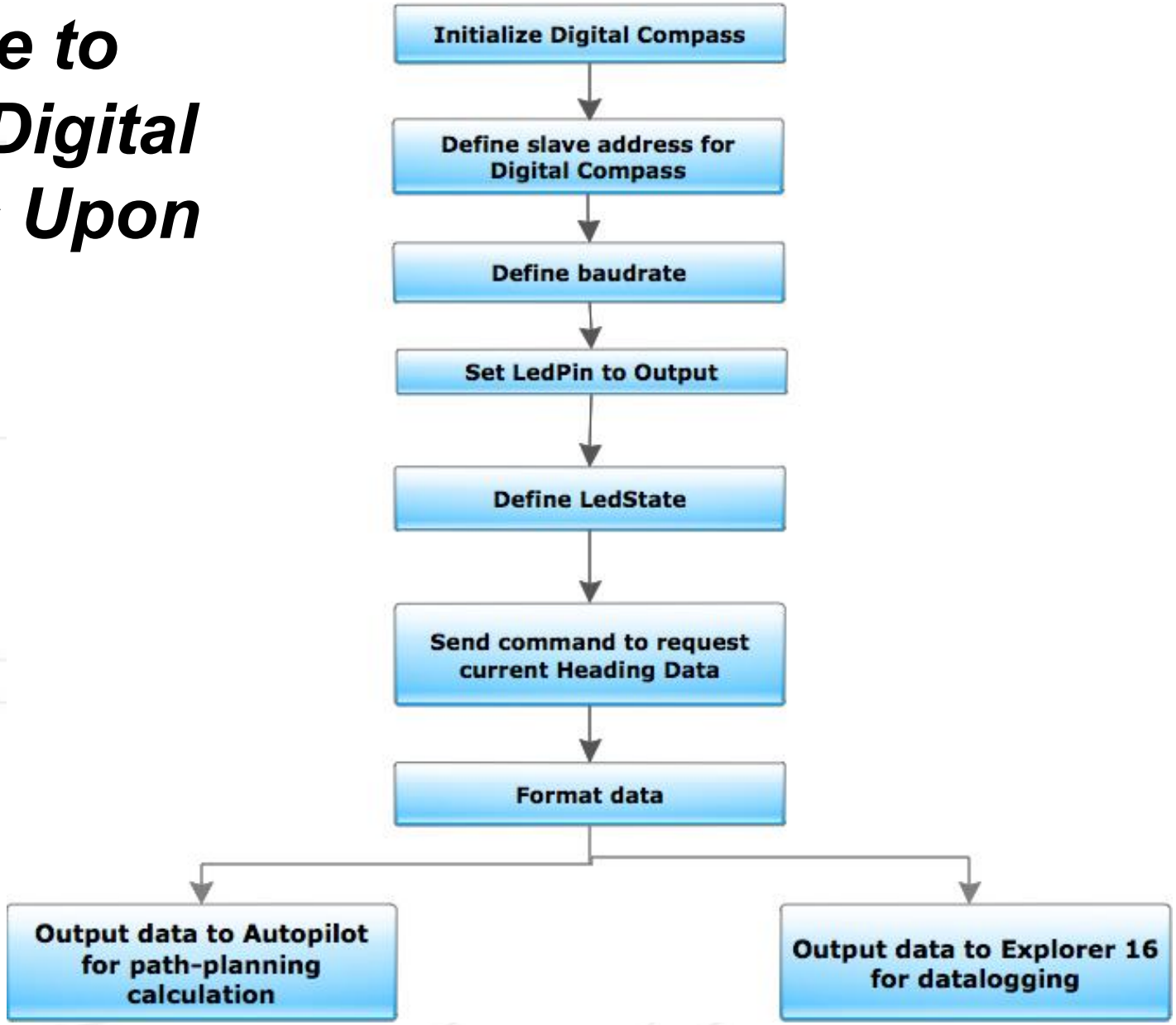
Integrate:

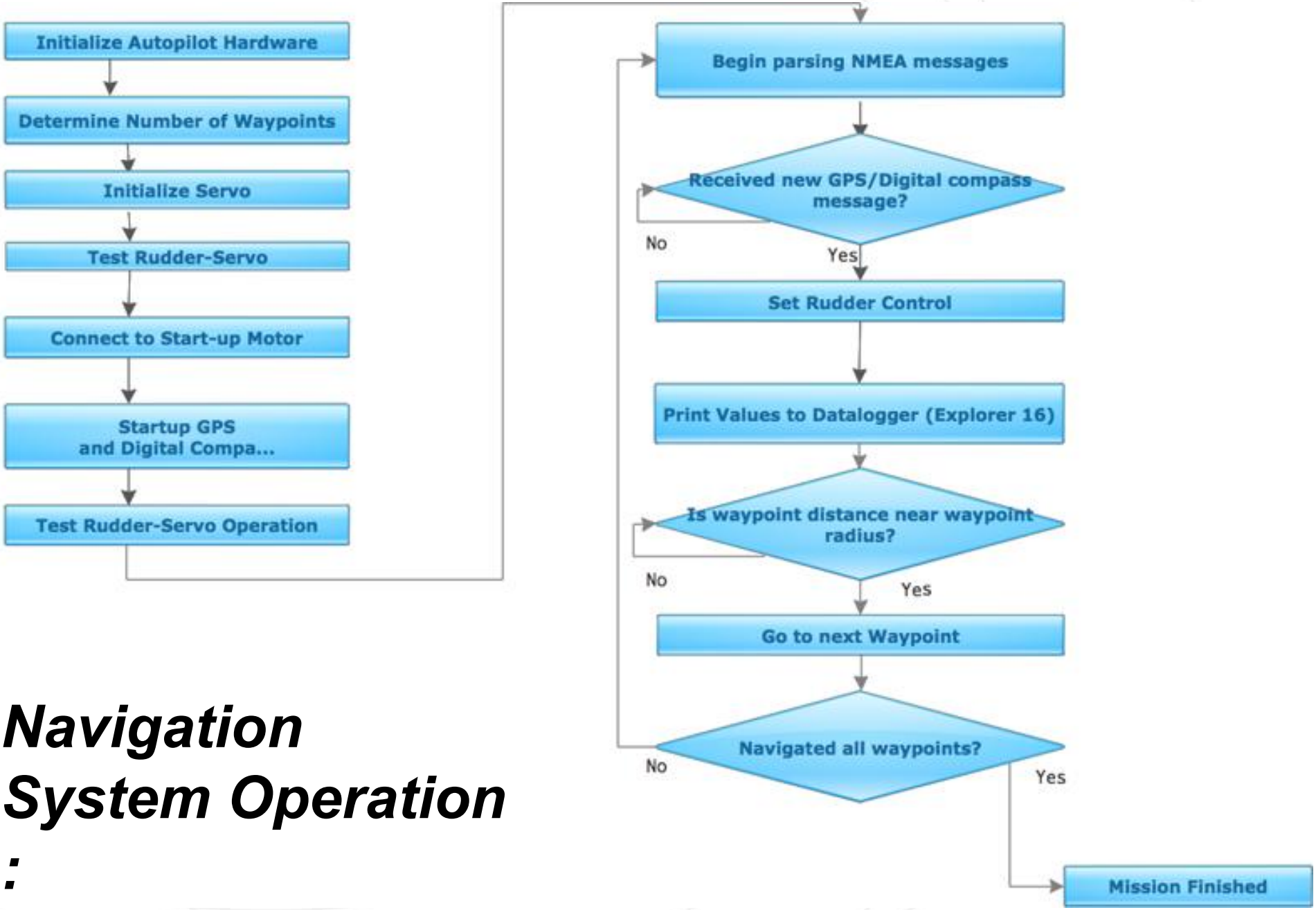
Autopilot and Digital Compass module





Procedure to Initialize Digital Compass Upon Startup:





Navigation System Operation :

Mechanical Components

3.1 Hull

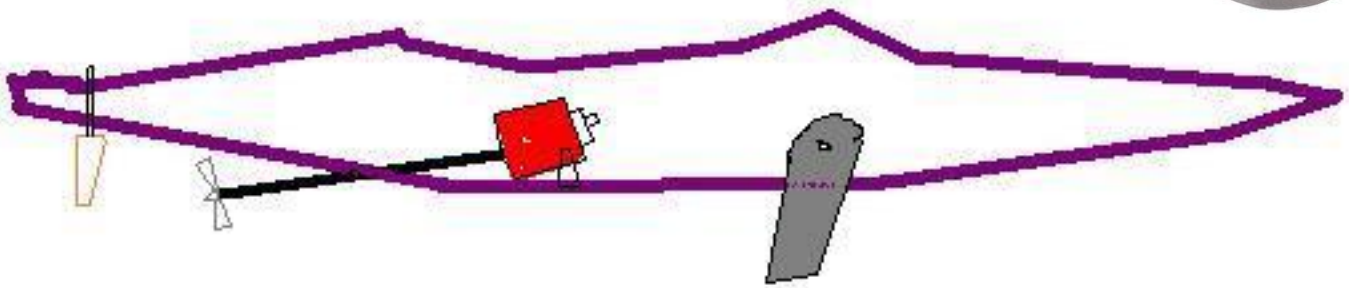
- Three different options were considered
  1. Custom design for an aluminum hull.
  2. Using 8 ft. jon boat.
  3. Purchasing a used kayak.

Decision Matrix

Option	Cost [20%]	Hydrodynamics [10%]	Bouyancy [10%]	Stability [10%]	Durability [15%]	Time to build [20%]	Weight [7%]	Ease of Transportation [8%]	Total
Aluminum Hull	0.4	0.3	0.2	0.4	0.15	0.4	0.175	0.24	2.27
Used jon boat	0.6	0.2	0.5	0.4	0.6	1	0.07	0.32	3.69
Used kayak	0.8	0.5	0.05	0.4	0.6	0.8	0.28	0.32	3.75
*Ranked from 1-5									

Kayak Hull

- fastest, lightest, and cheapest of all the three options.
- It needs to be refitted to meet requirements.

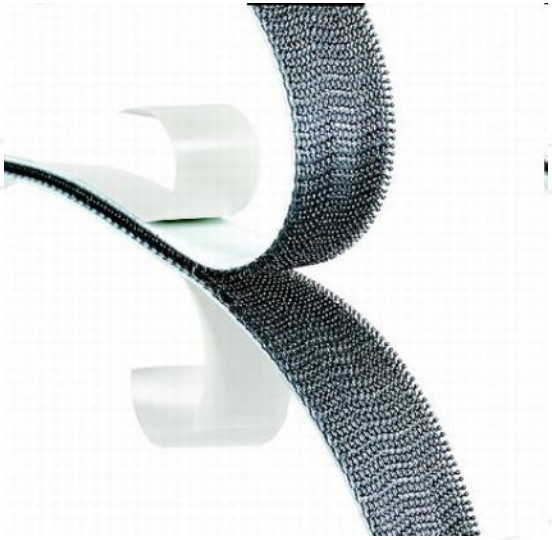


Kayak Hull

- Other important considerations are: sensor placement and internal structure

Electronics Housing

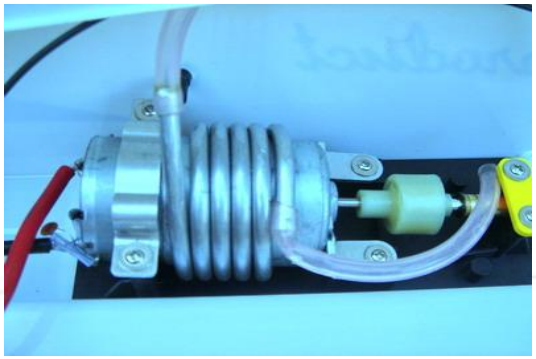
- Enclose all electronic components.
- Dimensions shall not surpass 0.6m x 0.9m x 0.3m



3M Velcro

Tight seal box

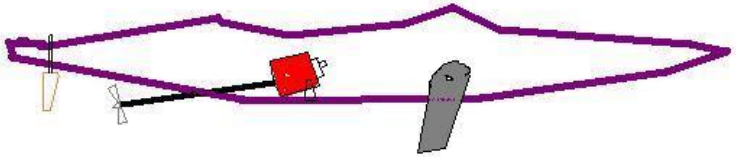
Cooling for Electronics



1. Flowing seawater through pipes running to electronics.

2. Electronics submersed in Mineral oil to help dissipate heat.

3.2 Propulsion



- Three design options:
  1. Electric Trolling
  2. Water Pump Jet drive
  3. Prop driven gas engine.

Options	Weight [lb]	Cost	Thrust [lb]	Speed [knots]	Run time [hr]	Type/amount of fuel
Pacer Self-priming transfer pump	80	449.99	45	6.5	12	unleaded gasoline, 4 gallons
Trolling Motor Minn Kota	167	168.99	30	6.5	12	12V 100Ah battery, 3
Prop driven Honda engine 35cc	22	290	29	7.8	12	unleaded gasoline, 2 gallons
Prop driven Predator engine 99cc	58	180	76	11.3	12	unleaded gasoline, 4 gallons

Real tests

For 33lbs of mass (hull only)	
Thrust (lbs)	Speed (mph)
5	4.6
10	6.4
15	7

For 78lbs of mass	
Thrust (lbs)	Speed (mph)
5	3.8/3.4
10	4.8/4.7
18	6.4

Honda GX35

GX 35

Engine Type	Air cooled 4 stroke OHC petrol engine
Cylinder Sleeve Type	Aluminium Cylinder
Bore x Stroke	39 x 30mm
Displacement	35.8 cm3
Compression	8.0 : 1
Net Power	1.0 Kw (1.3 HP) / 7000 rpm)
Max net torque	1.6 Nm / 0.16 Kgm / 5500 rpm
Ignition System	Transistorised
Starting System	Recoil
Fuel tank Capacity	0.63 l
Fuel cons. at rated power	0.71 L/hr - 7000 rpm
Lubrication	Crankcase Pressure Driven
Engine Oil Capacity	0.1 l
Dimensions (L x W x H)	198 x 234 x 240 mm
Dry Weight	3.33 kg (w/o clutch)

Fuel Tank

- Since the factory fuel tank is too small for our requirements, we will utilize an external plastic fuel tank to drain safely to the engine tank

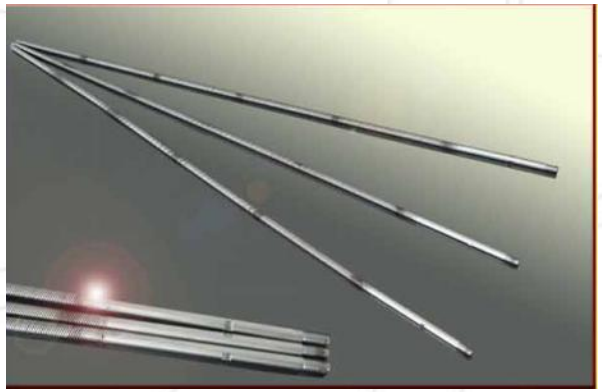
Propeller

- Prather S280 Metal Racing Props (Stainless Steel)

3.10" Diameter 4.5“ Pitch

Shaft

- **Hughey Flex Drive Cable, 1/4" x 24" 3/16 Step**



- **¼” straight metal shaft cut to size**

Coupler links

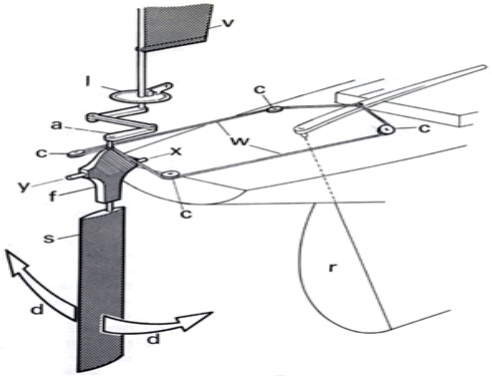
- We may have to machine a custom coupler to connect the engine shaft to the drive shaft.

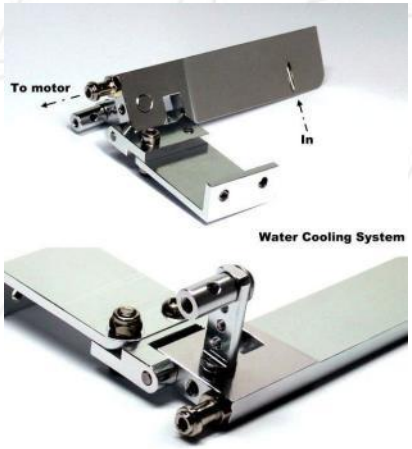
Clutch Drive

Enables the prop to be neutral when engine is at Idle.

3.3 Steering System

- Design shall have nimble turning radius when needed and a straight path as well.
- Waterproof high torque digital servo.

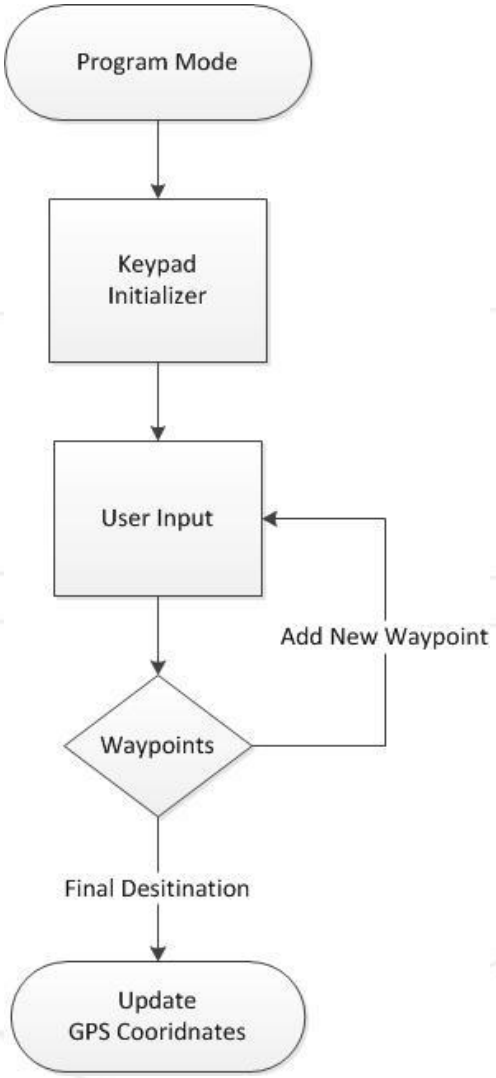




User Interface

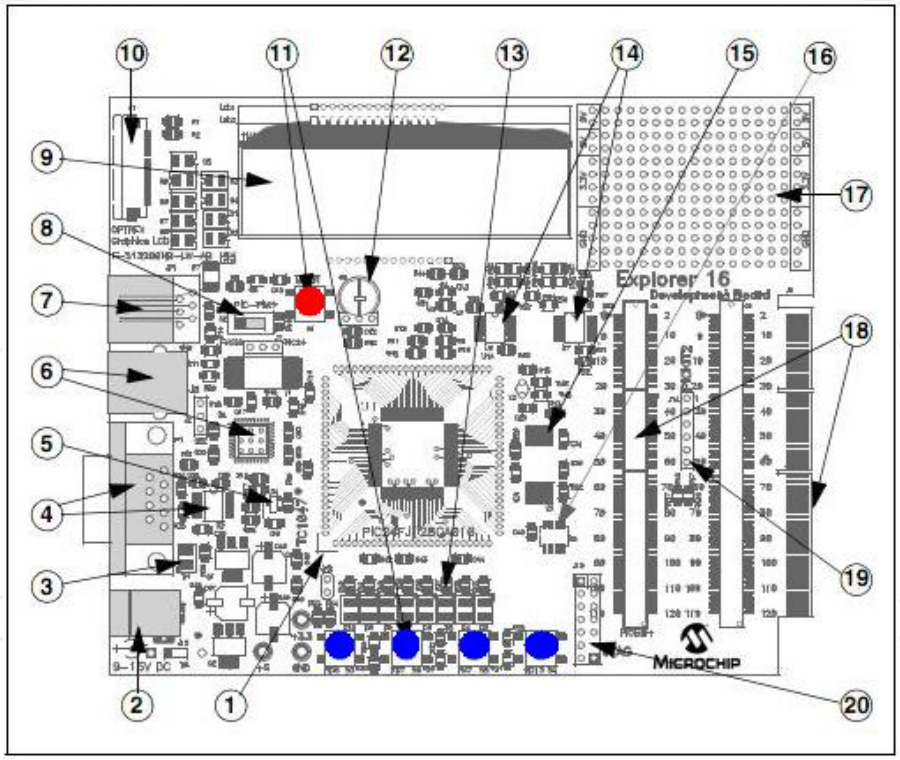
Steven Golemme

User Interface



- The keypad interface will allow the end-user to program paths into the AWQuSam using GPS coordinates of waypoints and final destination.
- The flowchart shows the process for developing the user interface which can be broken into four phases; Program Mode, Keypad Initialization, User Input, and GPS Coordinates.

Program Mode



The push-button would throw an interrupt and cause the processor to save its state of execution and begin executing in “Program Mode”. This occurs when the program
polls the switches and discover that one is active-low.

- S3: Active-low switch connected to RD6 (user-defined)
- S4: Active-low switch connected to RD13 (user-defined)
- S5: Active-low switch connected to RA7 (user-defined)
- S6: Active-low switch connected to RD7 (user-defined)

Keypad Initializer

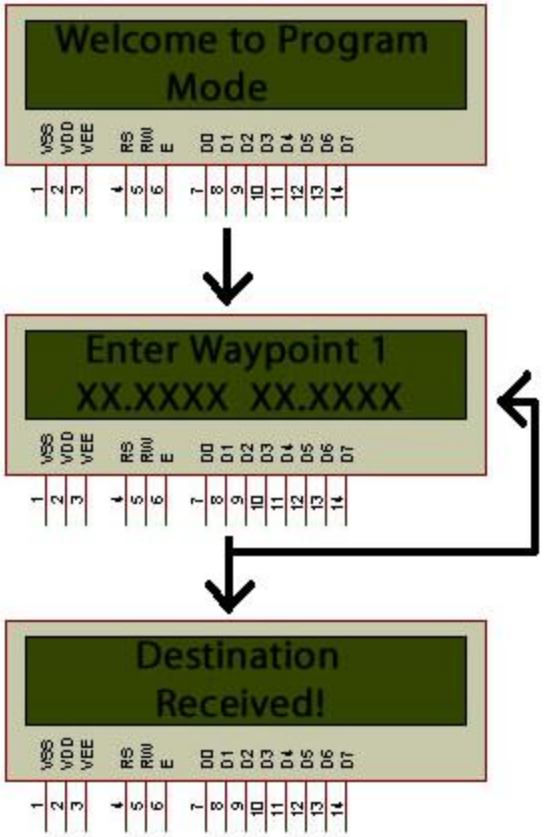
```
rom char* KeyPadMatrix = {
'1', '2', '3', 'A',
'4', '5', '6', 'B',
'7', '8', '9', 'C',
'*', '0', '#', 'D',
   0xFF
    };
```

Then the keypad connections would be defined as follows:

```
volatile bit row1port  @PORTA.1
volatile bit row1tris   @TRISA.1
volatile bit row2port  @PORTA.2
volatile bit row2tris   @TRISA.2
```

- void ScanKeypad(){
  char key = 0, row;
  for(row = LSB; row < binary 0001000; row shift left 1)
  {
- //Turns the rows output
- row1port = row (bit 1)
- row2port = row (bit 2)
- row3port = row (bit 3)
- row4port = row (bit 4)
- //Read columns and break if key is pressed
- if(col1port is pressed)
- break;
- increment key;
- if(col2port is pressed)
- break;
- increment key;
- …
- return KeyPadMatrix[key];
- }

User Input



- Pseudocode:
- /* Explorer 16 Dev board provides lcd libraries and functions that can be called to output to the lcd therefor there will be no need to develop code for initializing and writing to the lcd display*/
- #include<lcd.h>
- Init_LCD(); //will initialize lcd display
- lcd_cmd(LCD_CURSOR_OFF); //turns the cursor off
- lcd_cmd(CLEAR_LCD); //clears lcd of any previous display
- char* text = “Enter Waypoint 1”;
- puts_lcd(text, 16); //prints text to the first line of lcd
- char userinput[16]; //character array to store user input
- int i = 0;
- while(i < 15) { //for 16 keystroks
- char keyentry =keypadScan(); //scan
- lcd_data(keyentry); //prints keystroke to LCD
- userinput[i] = keyentry;

GPS Coordinates

- The user input will be stored as floating point numbers that will be sent to the AutoPilot module in source files to indicate the direction and coordinates for the AWQuSam to begin traveling. This information will be stored in multiple source files for each waypoint. Once the AWQuSam has reached the waypoint indicated by the GPS, it will then be updated with the next waypoint and destination. This routine will be the format for how the AWQuSam will continue on its course.

Financial Update

Steven Golemme

Expenditures to Date

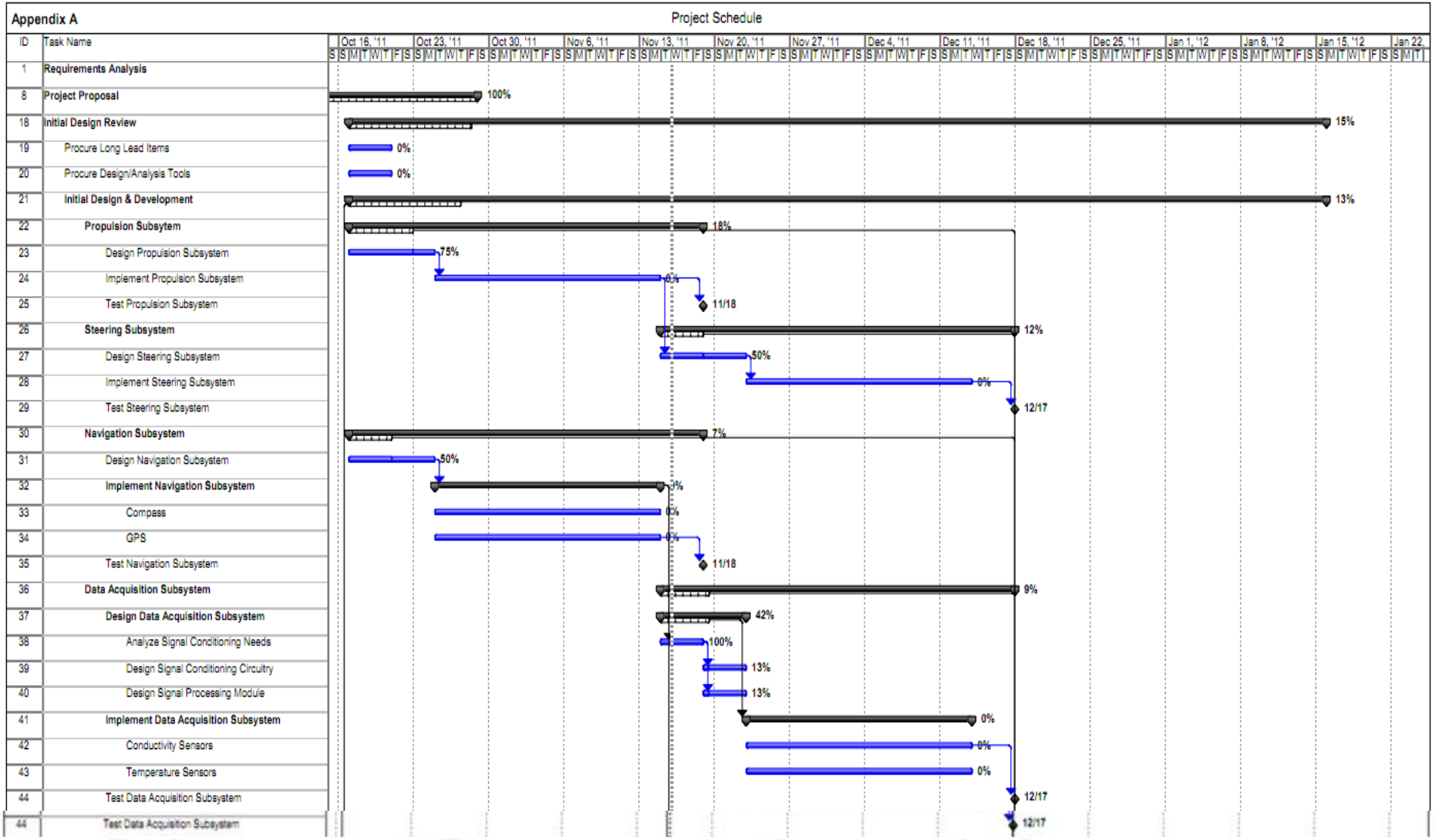
Item	P/N	Manufacturer	Distributor	Qty	Projected Cost	Actual Cost
PIC Development Board w/ Programmer	DV164037	Microchip	Microchip	1	$225	$225.00
SD Daughter Card	AC164122	Microchip	Microchip	1	$28.50	$28.50
Conductivity Sensor	SBE-4	Seabird	Ocean Dept	2	-	-
Temperature Sensor	OL-710	Omega	Omega	2	$200	$173.00
Compass Module	SEN-07915	Honeywell	Sparkfun	1	$34.95	$34.95
GPS	GPS-09566	ADH Technology Co. Ltd	Sparkfun	1	$79.95	$99.95
microSD 1GB Memory Card	COM-08163	A-Data	Sparkfun	1	$9.95	$9.95
MURS Radio	M538-BS	Dakota	Dakota	2	$260	$113.00
Terminal Node Controller	TinyTrak4	Byonics	Byonics	2	-	$65.00
Hull	-	-	Craig's List	1	$98	$140.00
Total Expenditures					$936.35	**$889.35**

Risk Register

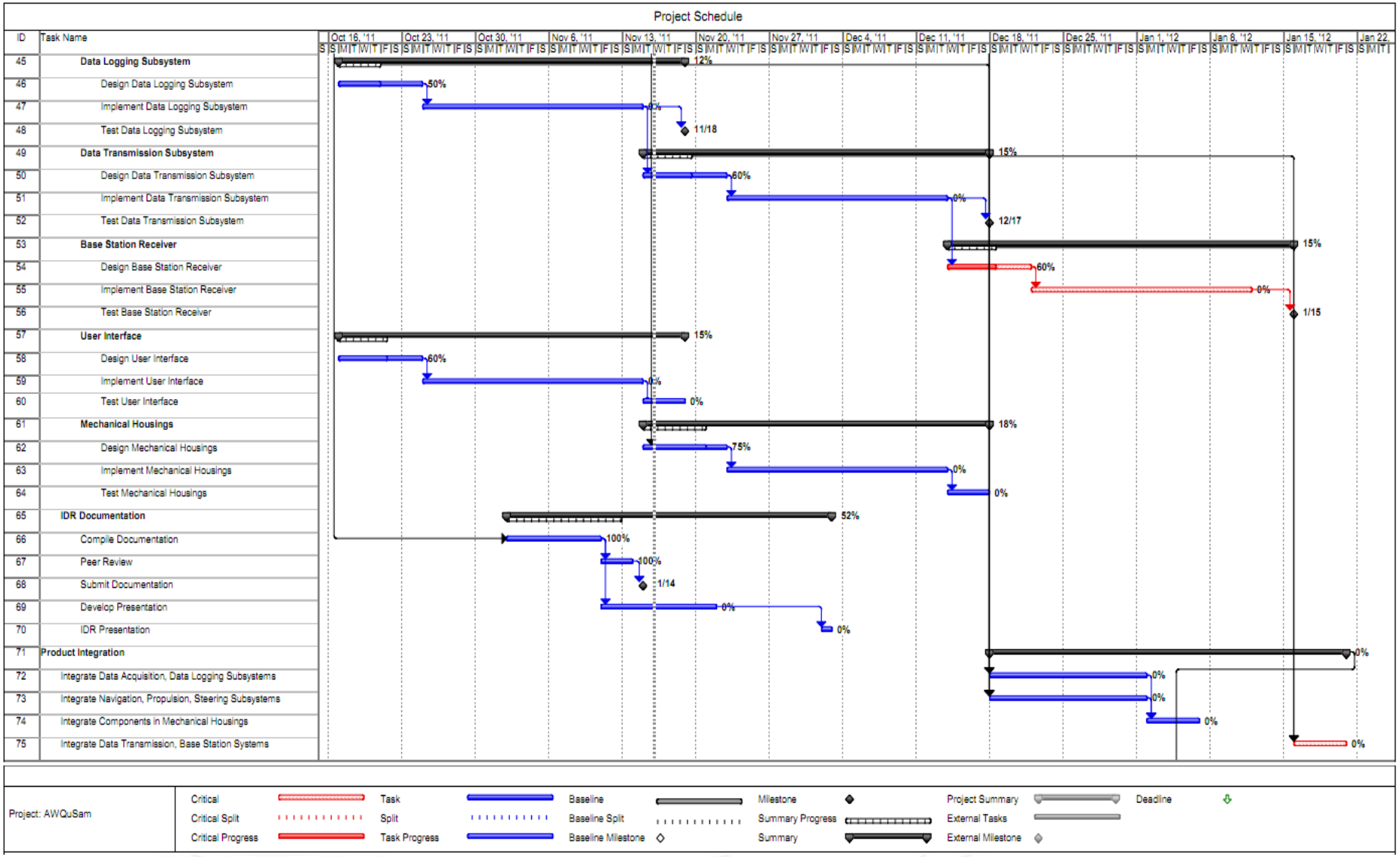
Budget Risks

ID	Risk Item	Probability	Impact	Mitigation Plan
1	Increase Product Expenses - Price estimates in the original proposal could possibly be lower than what the team actually pays	Low	Moderate	To make sure that this never occurs, the team makes budget talks a priority at every meeting.
2	Unforeseen Expenses -As the team continues moving forward in the design and development of the AWQuSam, there may be devices or products that the team was initially unaware of, that are a necessity to the completion of this product.	Moderate	High	If an engineer can identify a component that may be needed in the future early in the process, it becomes much less expensive to solve compared to the final stages of the design.
3	Safety Risk - AWQuSam impacting a diver in the head is a grave safety concern. Addressing this issue requires budget modification	Moderate	Moderate	A strobe / revolving beacon can be added to address the safety concern. Reserve budget allotment must be used for this.

Project Schedule



Project Schedule



Risk Register

Schedule Risks

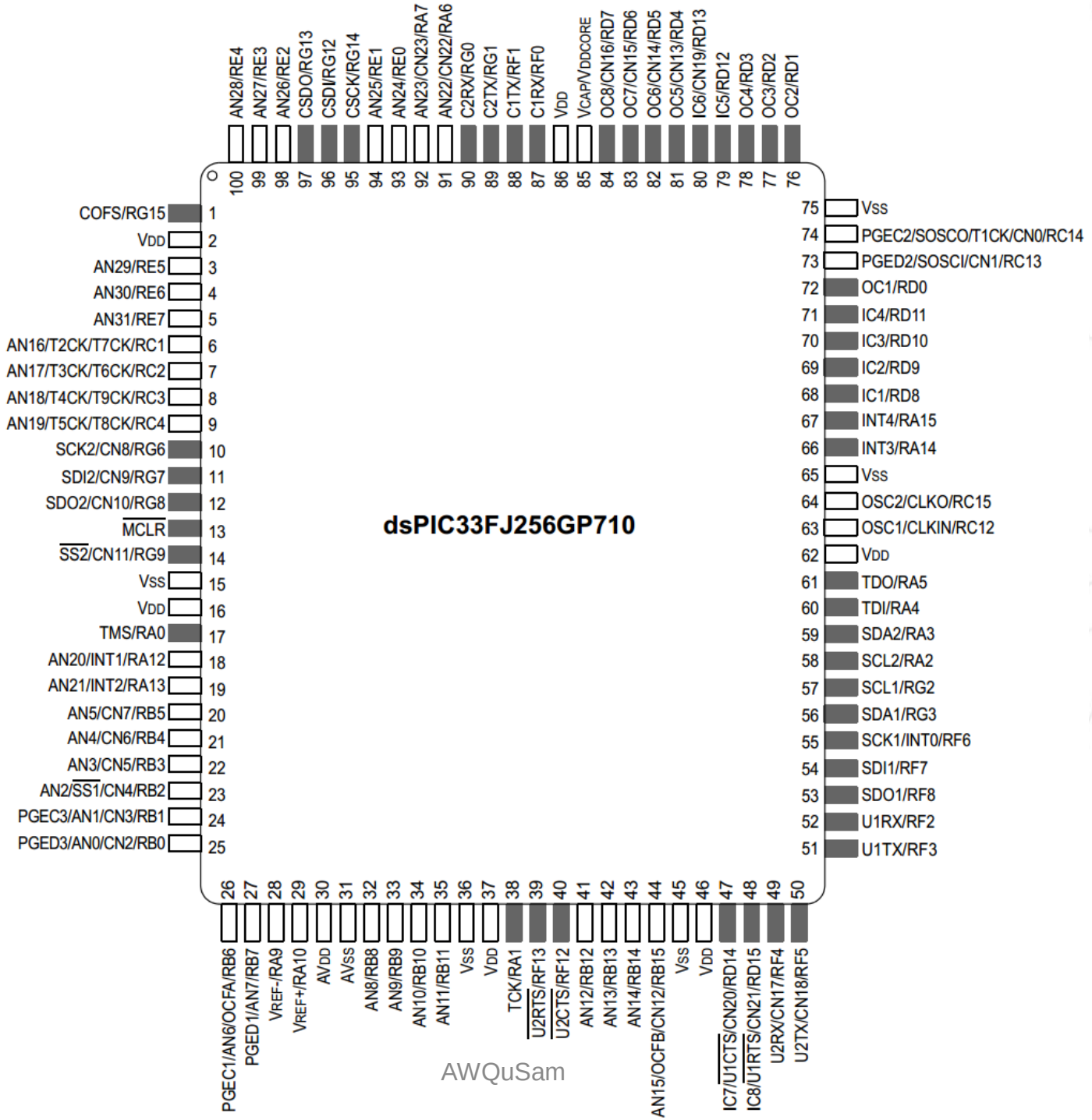
ID	Risk Item	Probability	Impact	Mitigation Plan
1	Personnel Leave - Illness or absence of any engineer would require over-allocating other resources	Medium	High	Some flexibility has been built into schedule. Many subsystems have multiple engineers assigned. One engineer can divide time to cover responsibilities of absent engineer.
2	Centralized Design - Majority of design is on the same microcontroller development board, but this board is not available to all engineers simultaneously	High	Low	Programmer will be installed on multiple computers to allow for independent programming and parallel design work. Module testing time will be coordinated.

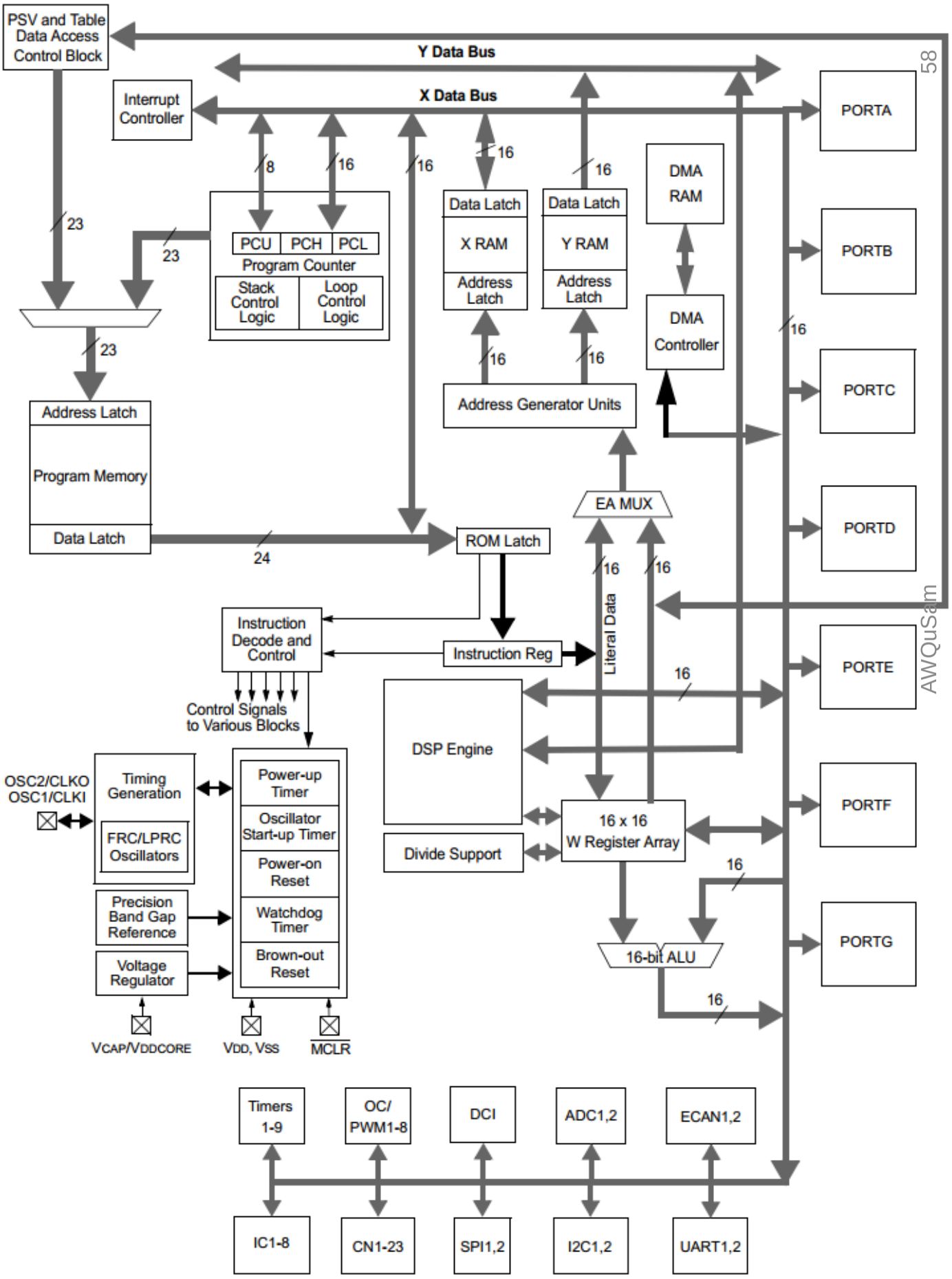
Questions?

AWQuSam Engineering

Thanks Each of You

Additional Slides


dsPIC33FJ256GP710
1 COFS/RG15
2 VDD
3 AN29/RE5
4 AN30/RE6
5 AN31/RE7
6 AN16/T2CK/T7CK/RC1
7 AN17/T3CK/T6CK/RC2
8 AN18/T4CK/T9CK/RC3
9 AN19/T5CK/T8CK/RC4
10 SCK2/CN8/RG6
11 SDI2/CN9/RG7
12 SDO2/CN10/RG8
13 MCLR
14 SS2/CN11/RG9
15 VSS
16 VDD
17 TMS/RA0
18 AN20/INT1/RA12
19 AN21/INT2/RA13
20 AN5/CN7/RB5
21 AN4/CN6/RB4
22 AN3/CN5/RB3
23 AN2/SS1/CN4/RB2
24 PGEC3/AN1/CN3/RB1
25 PGED3/AN0/CN2/RB0
26 PGEC1/AN6/OCFA/RB6
27 PGED1/AN7/RB7
28 VREF-/RA9
29 VREF+/RA10
30 AVDD
31 AVSS
32 AN8/RB8
33 AN9/RB9
34 AN10/RB10
35 AN11/RB11
36 VSS
37 VDD
38 TCK/RA1
39 U2RTS/RF13
40 U2CTS/RF12
41 AN12/RB12
42 AN13/RB13
43 AN14/RB14
44 AN15/OCFB/CN12/RB15
45 VSS
46 VDD
47 IC7/U1CTS/CN20/RD14
48 IC8/U1RTS/CN21/RD15
49 U2RX/CN17/RF4
50 U2TX/CN18/RF5
51 U1TX/RF3
52 U1RX/RF2
53 SDO1/RF8
54 SDI1/RF7
55 SCK1/INT0/RF6
56 SDA1/RG3
57 SCL1/RG2
58 SCL2/RA2
59 SDA2/RA3
60 TDI/RA4
61 TDO/RA5
62 VDD
63 OSC1/CLKIN/RC12
64 OSC2/CLKO/RC15
65 VSS
66 INT3/RA14
67 INT4/RA15
68 IC1/RD8
69 IC2/RD9
70 IC3/RD10
71 IC4/RD11
72 OC1/RD0
73 PGED2/SOSCI/CN1/RC13
74 PGEC2/SOSCO/T1CK/CN0/RC14
75 VSS
76 OC2/RD1
77 OC3/RD2
78 OC4/RD3
79 IC5/RD12
80 IC6/CN19/RD13
81 OC5/CN13/RD4
82 OC6/CN14/RD5
83 OC7/CN15/RD6
84 OC8/CN16/RD7
85 VCAP/VDDCORE
86 VDD
87 C1RX/RF0
88 C1TX/RF1
89 C2TX/RG1
90 C2RX/RG0
91 AN22/CN22/RA6
92 AN23/CN23/RA7
93 AN24/RE0
94 AN25/RE1
95 CSCK/RG14
96 CSDI/RG12
97 CSDO/RG13
98 AN26/RE2
99 AN27/RE3
100 AN28/RE4


PSV and Table Data Access Control Block
Y Data Bus
X Data Bus
Interrupt Controller
PORTA
PORTB
PORTC
PORTD
PORTE
PORTF
PORTG
DMA RAM
DMA Controller
PCU
PCH
PCL
Program Counter
Stack Control Logic
Loop Control Logic
Data Latch
X RAM
Address Latch
Data Latch
Y RAM
Address Latch
Address Generator Units
Address Latch
Program Memory
Data Latch
ROM Latch
EA MUX
Literal Data
Instruction Decode and Control
Instruction Reg
Control Signals to Various Blocks
DSP Engine
Divide Support
16 x 16 W Register Array
16-bit ALU
OSC2/CLKO
OSC1/CLKI
Timing Generation
FRC/LPRC Oscillators
Power-up Timer
Oscillator Start-up Timer
Power-on Reset
Watchdog Timer
Brown-out Reset
Precision Band Gap Reference
Voltage Regulator
VCAP/VDDCORE
VDD, VSS
MCLR
Timers 1-9
OC/ PWM1-8
DCI
ADC1,2
ECAN1,2
IC1-8
CN1-23
SPI1,2
I2C1,2
UART1,2
58
AWQuSam