

Group 11

# GOBOSH G700S FLIGHT SIMULATOR

Chris Dlugolinski, Robert Gysi, Joseph Munera, Lewis Vail

Sponsored by: Mr. Dave Kotick, Grizzly Aviation

# Motivation

- Create a realistic cockpit-based flight simulator that can be used by our project sponsor to increase sales of both the aircraft and of his flight instruction business.

# Background (Aircraft)

- The GoBosh G700S is a import variant of the Polish-built Aero AT-3 and is considered to be in the Light Sport Aircraft (LSA) segment, which requires very little pilot training.

# Background (Project)

- To build a fully-integrated simulated cockpit environment which includes:
  - Standard Six-Pack gauges (Airspeed, VSI, Altimeter, Heading, Attitude and Turn Coordinator)
  - Flight Controls (Pedal, Stick, Throttle)
  - Projection system
- Integrate the above systems into an actual cockpit being shipped to our sponsor from the Czech Republic

# Outline

- ⦿ Simulator Software
  - Aircraft Model
- ⦿ Computer Hardware
- ⦿ SDK Software
- ⦿ Instrument/Control Software
- ⦿ Flight Controls
- ⦿ Flight Instruments
- ⦿ Administrative Information

# Simulator Software

# Flight Simulator Requirements

- Needs to be a low-cost software package
- Allow us the ability to interface with custom instruments and controls
- Provide a realistic environment
- Guarantee 30 Frames Per Second
- Method for creating/importing a custom aircraft model
- Two that meet these requirements:
  - Microsoft Flight Sim X (FSX)
  - Laminar Research X-Plane 9

# Microsoft FSX Pros

- The most popular desktop based flight simulator available on the market; large community of add-on developers
- Uses the FSUIPC and the SimConnect API for interfacing custom devices into the simulator
- Inclusion of many worldwide airports and accurate detailed scenery in large cities Computer-controlled (AI) based aircraft populate airspace automatically
- Cost: \$30



# Microsoft FSX Cons

- We can't deliver a guarantee a minimum of 30 FPS. A target can be set but the game will not auto-adjust settings to maintain the frame rate.
- No included model editor – require expensive 3<sup>rd</sup> party modeling tools.
- No built in Instructor Operator Station (IOS) functionality out of the box. Requires additional development.
- No longer in development – Microsoft closed the ACES studio in Jan. 2009.

# X-Plane 9

- ⦿ Decided to go with this software instead.
- ⦿ While there is not as an extensive community of add-on developers, the SDK documentation is very thorough – everything is written as a plugin for the simulator software.
- ⦿ Includes a built in model editor, meaning no need to purchase additional software.
- ⦿ Can maintain a frame rate of 30 FPS
- ⦿ With additional thumb stick from Laminar Research the simulator software can become FAA Certified and used for ground based training.
- ⦿ Cost: \$30

# FSX/X-Plane Graphics Comparison



◀ FSX

▼ X-Plane 9

Both screenshots are of a Cessna C172 over Innsbruck, Austria.



# Aircraft Model Requirements

- Model must match the actual Aircraft
  - Physics
  - Look
  - Flight Control
- ◉ Need a Model Built





# Simulator Computer

# Simulator PC Requirements

- CPU: 2GHz
- RAM: 4GB
- HDD: 120GB
- Graphics Card(s) powerful enough to output 120-Degree Simulated field of view on three monitors (Software requires minimum of 64MB onboard graphics RAM)
- Monitor: 24" or larger to created desired FOV (using equations above)

$$K = \frac{\sqrt{9^2 + 16^2}}{16} \quad (Eq. 1)$$

$$constant = 2K \tan 20^\circ = \frac{\tan 20^\circ \sqrt{9^2 + 16^2}}{8} \quad (Eq. 2)$$

$$monitor\ size = (distance)(constant) \quad (Eq. 3)$$

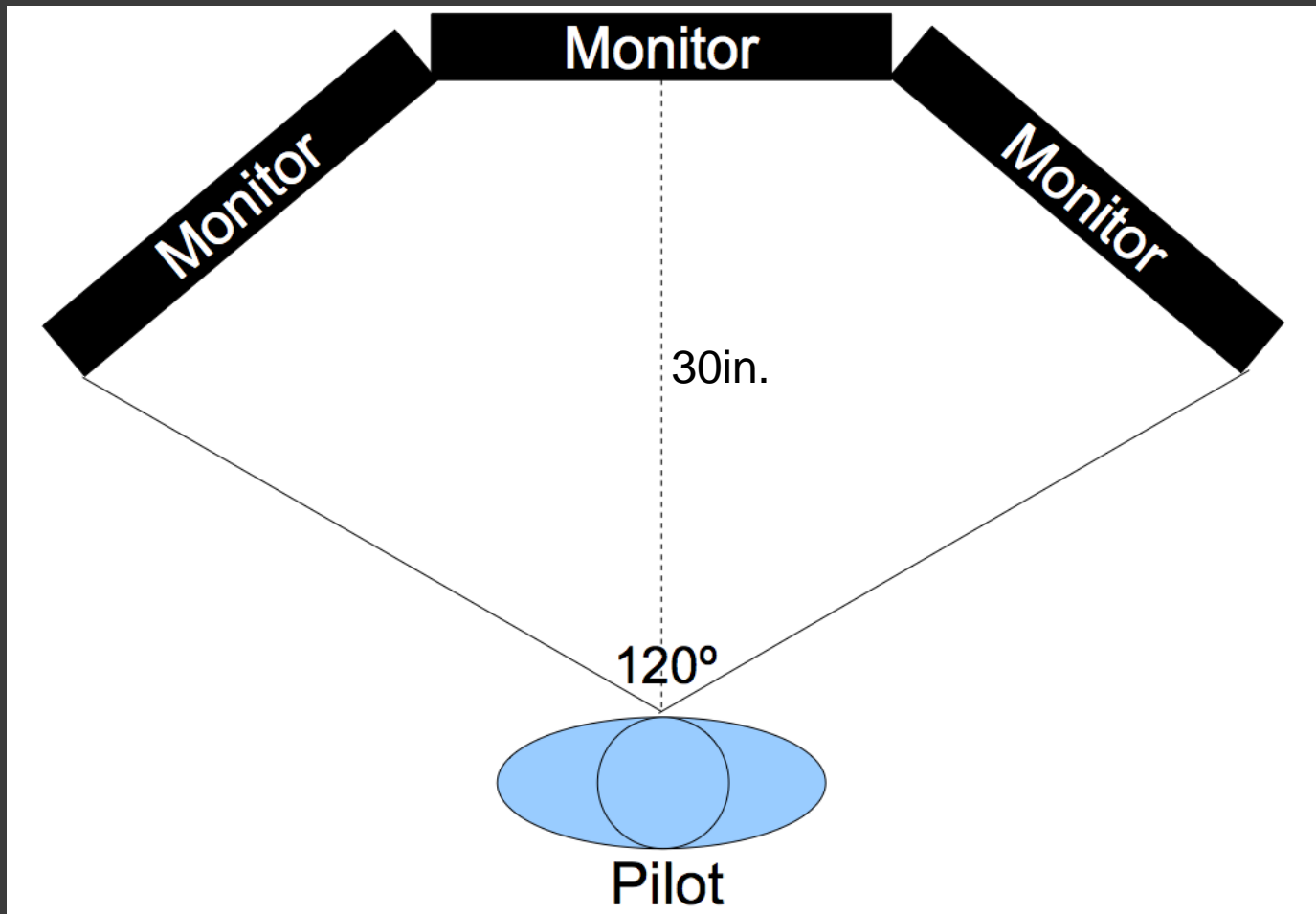
# Simulator PC Specifications

- Total Cost is roughly \$1200 to outfit the entire computer system, even using fairly low-cost components.

Item	Description
CPU	AMD Phenom X2 550 @ 3.1 GHz
GPU (x2)	ATI Radeon 5750 1GB RAM onboard each
Motherboard	ASUS M4A785TD-V EVO
HDD	160GB
DVD-ROM	Yes
Power Supply	1000W ATX
Monitors (x3) – 24”	Gateway FHD2402



# Monitor Configuration

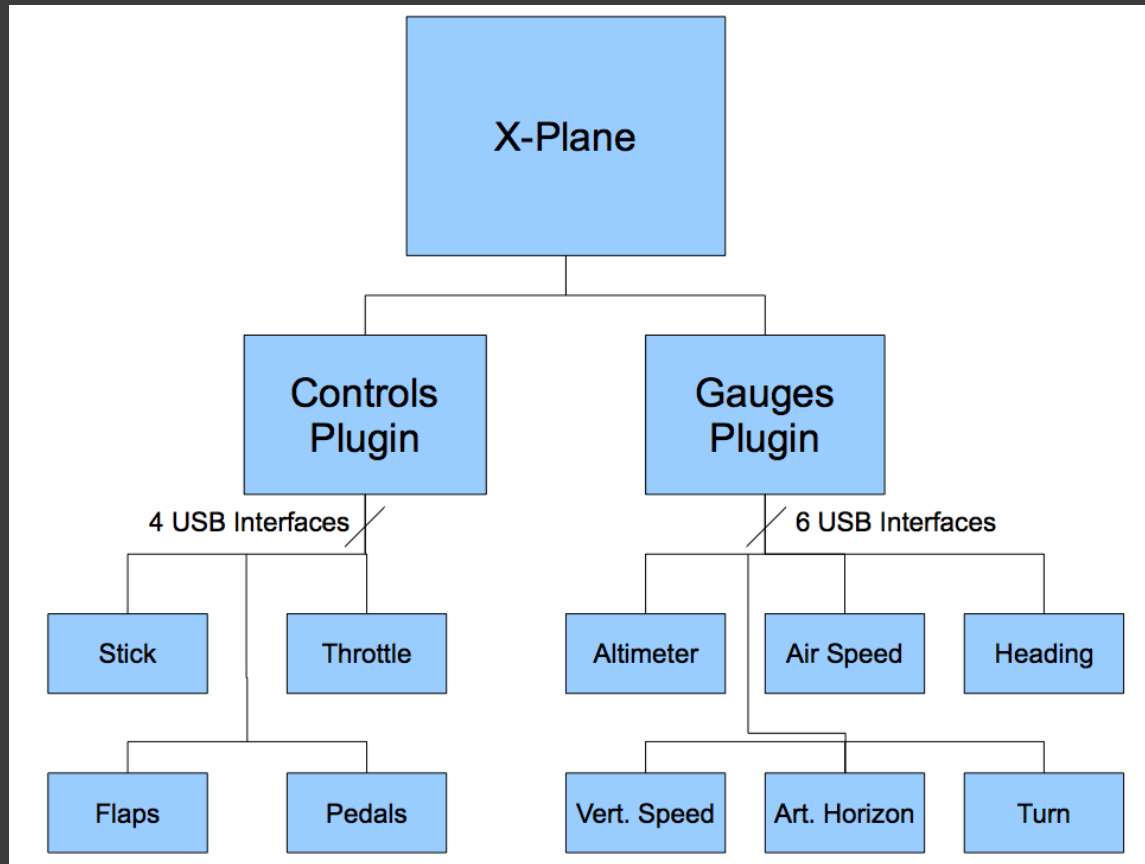


**SDK Software**

# Plugin Requirements

- ⦿ Wanted our development to be as modular as possible for future development
  - One plugin for input (controls) and one plugin for output (instruments)
  - Plugins are completely parametric driven
- ⦿ Need to be as realistic as possible
  - Aiming for 30fps for both sampling inputs and driving outputs pending performance testing

# High-level Plugin Architecture



# Plugin Design

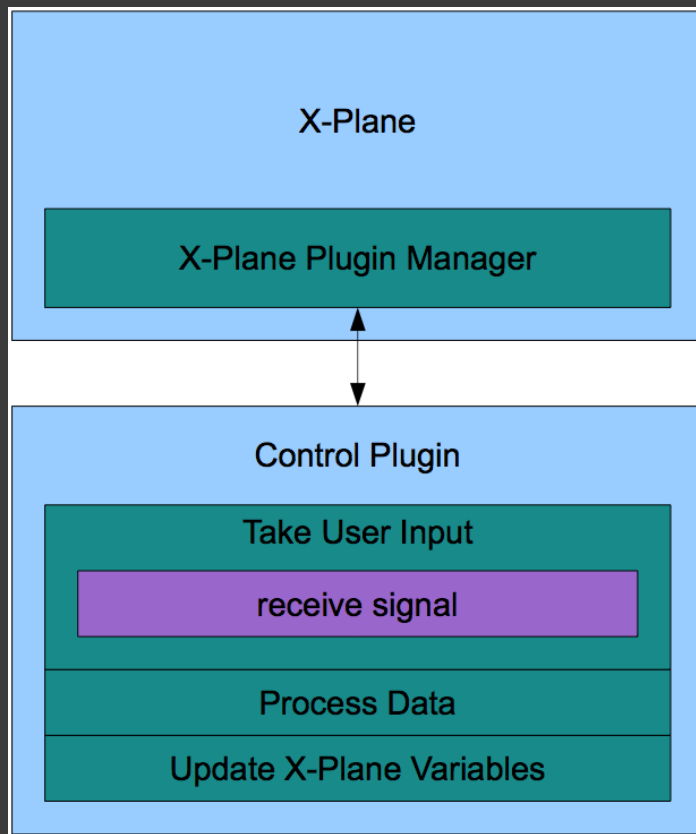
- ⦿ All plugins communicate with X-Plane through the X-Plane Plugin Manager (XPLM) via the plugin API
- ⦿ Controls:
  - Read position of controls, translate to X-Plane Value, and write new value back through XPLM
- ⦿ Instruments:
  - Read X-Plane values, translate to number of steps, and step through the difference

# Plugin Implementation

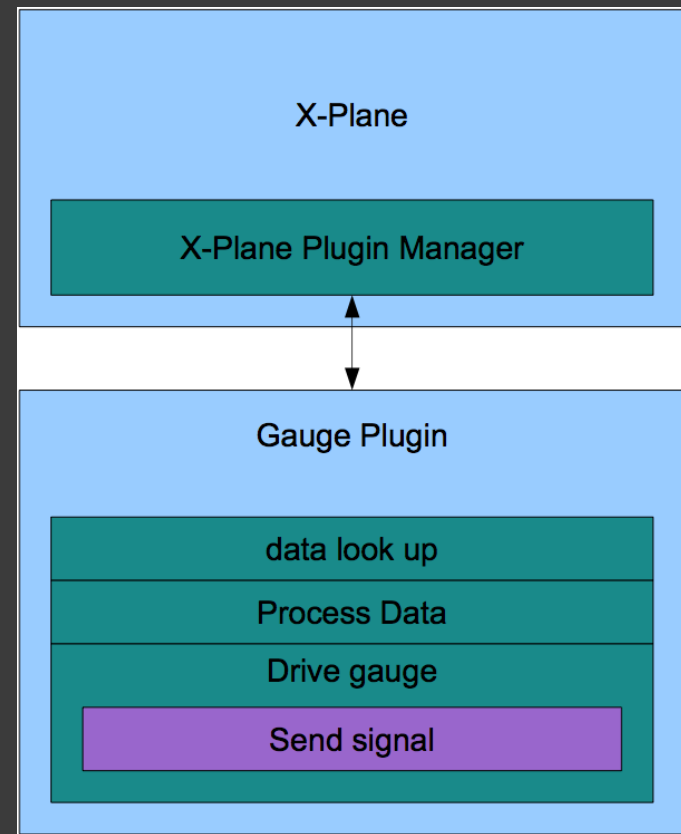
- ◉ Writing everything in C++ because this what the X-Plane SDK supports
- ◉ Internal plugin logic is all implemented in one thread to avoid any read/write hazards within the XPLM
- ◉ Conversions between X-Plane values and number of steps will all be done using stored minimum and maximum values for each device
  - This data will be stored in a flat config file
  - One config file per FTDI chip
- ◉ Control plugin and instrument plugin will use much of the same code to ease development

# Plugin Design (Continued)

## Control Plug-in Architecture

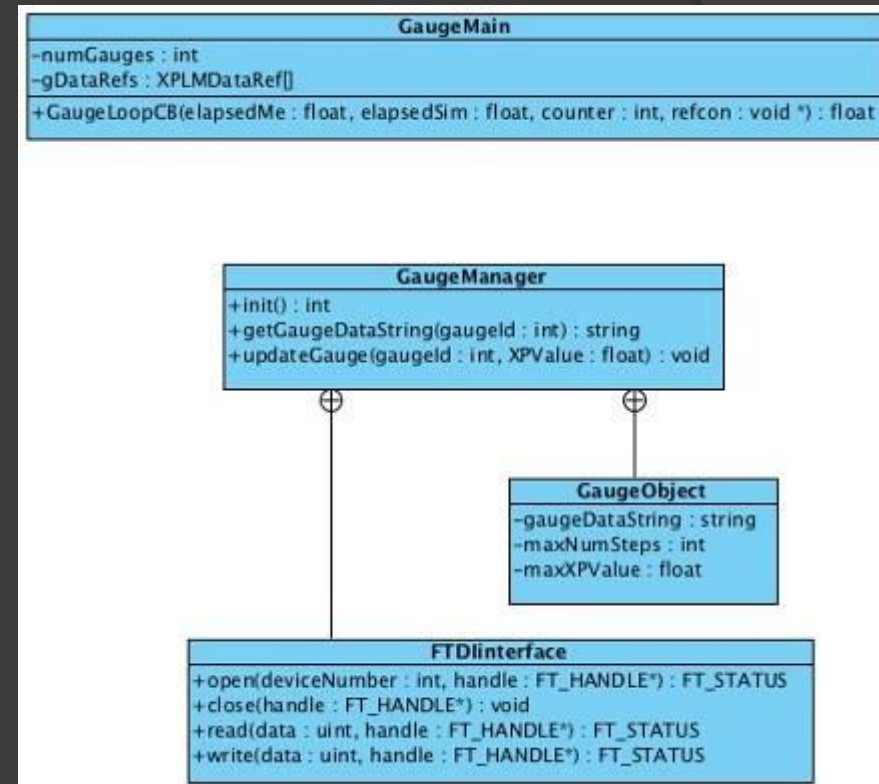


## Gauge Plug-in Architecture



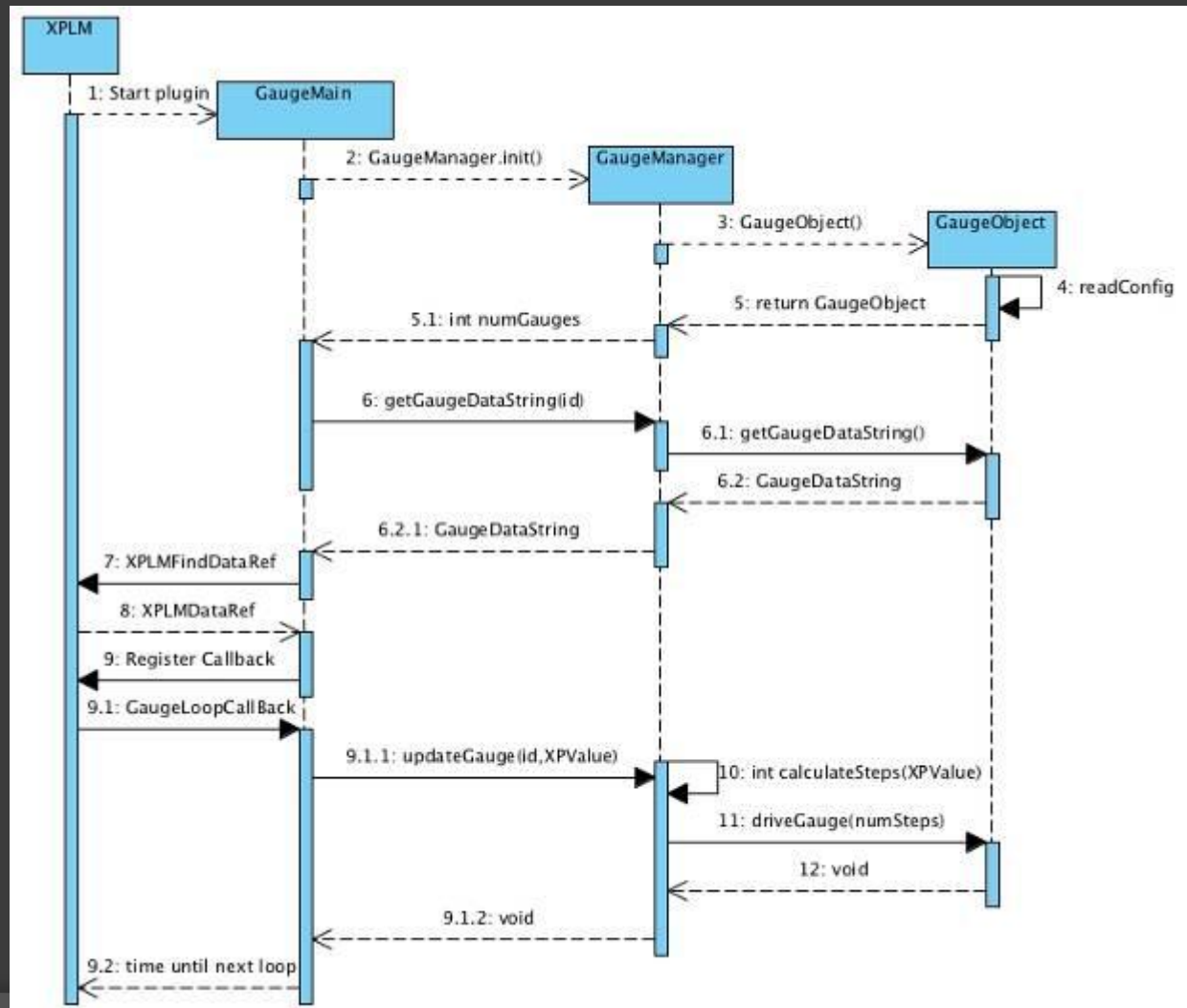
# Class Diagram

- Four classes:
  - GaugeMain** interfaces with X-Plane
  - GaugeManager** is the container class for FTDIinterface and GaugeObject
  - FTDIinterface** interfaces with the FTDI chips
  - GaugeObject** stores the FTDI data





# Sequence Diagram



# Instrument/Control Software

# Control Design Decisions/Requirements

## ⦿ Data Speed

- Must use USB for communications
- Smooth gauge motion

## ⦿ Modularity

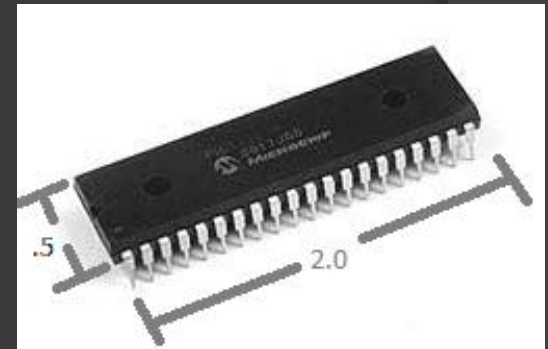
- Adding gauges
- Adding controls



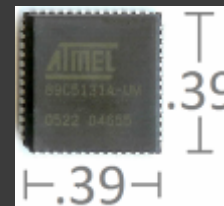
# Microcontroller Requirements

- Need to support USB
- Can control Stepper Motors and switches
- Work with A/D converters
- Use less than 5v and
- $< 100\text{mA}$  at startup and
- $< 500\text{mA}$  fully functioning
- Fit in a 3.25 inch tube
- At least 8 I/O pins

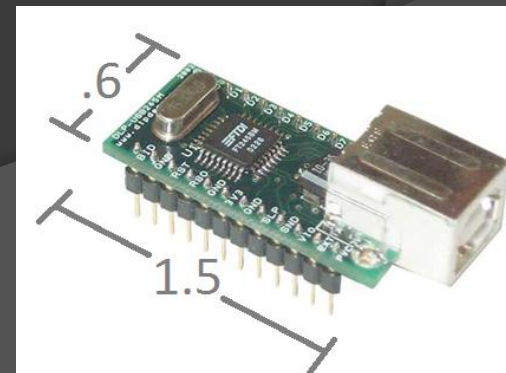
PIC-USB-4550



Atmel AT89C5131



FT245BM



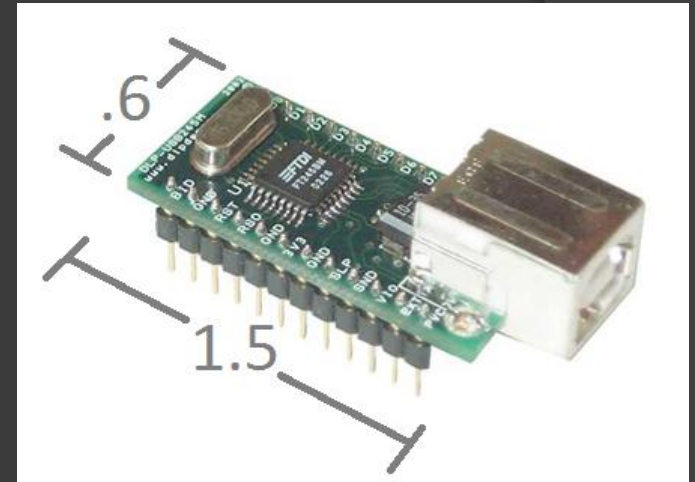
# Microcontroller Comparison

Microcontroller	Atmel AT89C5131	PIC18F4550	FTDI FT245BM
Dev Board	futurlec.com	futurlec.com	FTDI
Cost	Dev Board \$35.90 Chip \$10.11	Dev Board \$46.90 Chip \$14.99	Dev Board \$30 Chip \$5.00
Usb driver	<a href="http://www.atmel.com/dyn/resource/s/prod_documents/doc7646.pdf">http://www.atmel.com/dyn/resource/s/prod_documents/doc7646.pdf</a>	CDC Firmware <a href="http://www.microchip.com/sourcecode/index.php#pic18f4550usb">http://www.microchip.com/sourcecode/index.php#pic18f4550usb</a> <a href="http://microcontrollershop.com/product_info.php?products_id=2125">http://microcontrollershop.com/product_info.php?products_id=2125</a>	Free from FTDI to download and no programming on chip unless really necessary. <a href="http://www.futurlec.com">www.futurlec.com</a>
speed	24 MHz	48 MHz	USB 1.1 or USB 2.0 (compatible)
Examples	Come with dev board	Come with dev board <a href="http://www.create.ucsb.edu/~dano/CUI/">http://www.create.ucsb.edu/~dano/CUI/</a> <a href="http://www.edaboard.com/topic313796.html">http://www.edaboard.com/topic313796.html</a>	<a href="http://electronicdesign.com/Articles/index.cfm?AD=1&amp;ArticleID=16125">http://electronicdesign.com/Articles/index.cfm?AD=1&amp;ArticleID=16125</a>
Memory	32k	32k	External EEPROM
Memory RAM	1k	2k	
I/O	34	35	8 pin
Languages	c, assembler	c, c++, assembler	any
Voltage and Current Ratings	3.0V to 3.6V 30 mA Max Operating Current	3.3V detached 25mA	All usb self contained may need to do something for control of external parts
Thoughts			
Atmel AT89C5131	This seems to be a better choice all around including the fact that we could have pre made USB communications cutting out some of the hassle of that. After speaking with Dr. Richie and discussions with the rest of the group this option is there only if we need to actually do some programming on chip that is greater than necessary to make the hardware work		
PIC18F4550	Most hobbyists and a lot of projects on the net use this controller, which means we will have many examples to use or go from		
FT245BM	This is a chip that requires no extra programming on chip for the USB communication. It can be used in conjunction with the other chips or on its own in a special mode that allows for direct transfer of the info from the cpu to the ports on the chip....		

# Do we really need a Microcontroller?

FT245BM

- To  $\mu$ C or not?
- 1M Byte/sec using D2 driver
- Can be self powered and power other devices up to 5V and 500mA
- USB 1.1 and 2.0 compatible
- ★ ○ Bit-Bang Mode allows the data bus to be used as an 8 bit general purpose I/O port without the need for MCU or other support logic



If it doesn't provide good enough control can be USB Comm chip


# Data Speed Calculation

$$200 \frac{\text{steps}}{\text{rev}} * \frac{8 \text{ bits}}{.5 \text{ step}} = 3200 \text{ bits/rev}$$

Max Motor speed FTDI chip

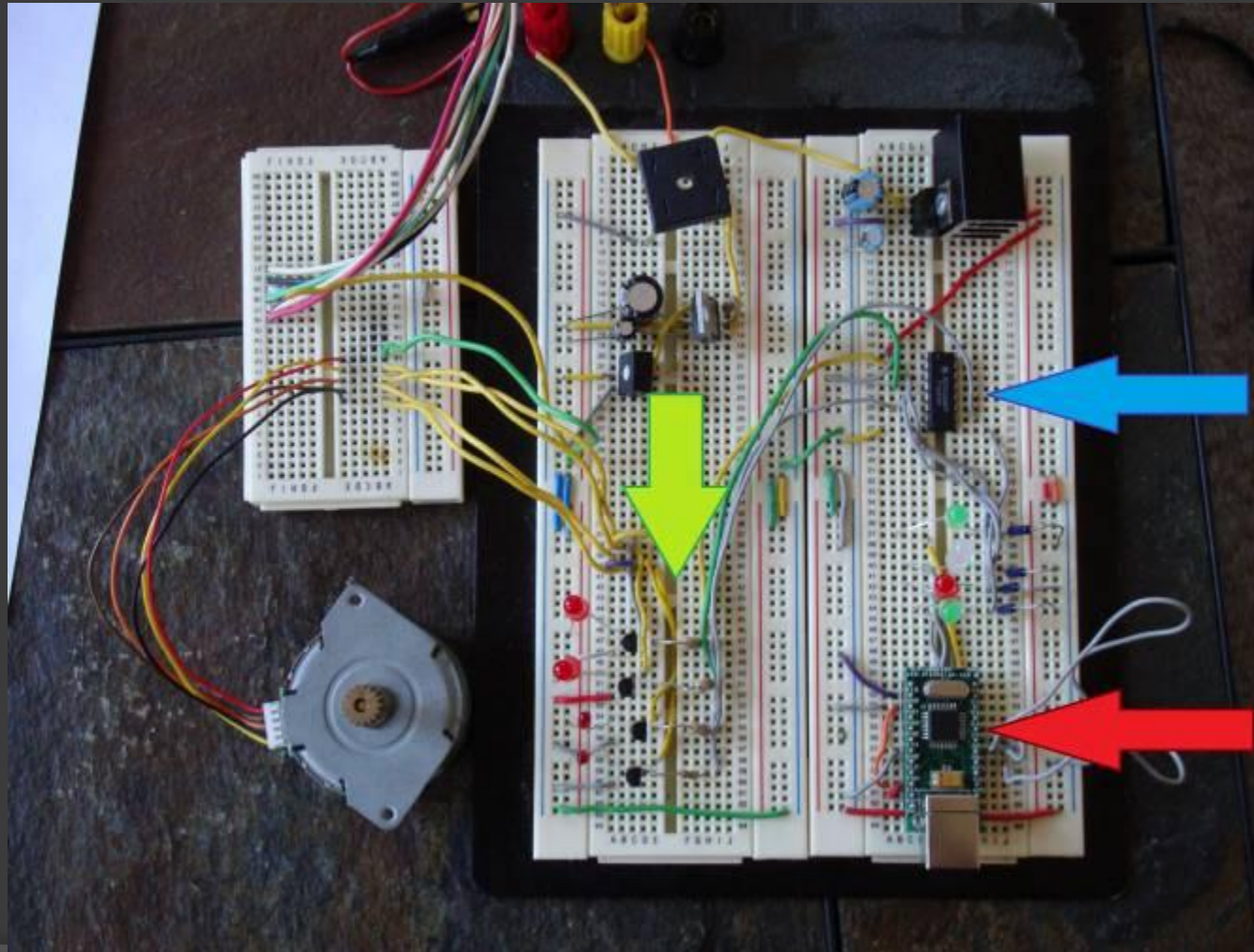
$$\frac{8,000 \frac{\text{kb}}{\text{s}}}{3.2 \frac{\text{kb}}{\text{rev}}} = 2500 \frac{\text{rev}}{\text{s}}$$

Actual Motor speed FTDI chip


$$200 \frac{\text{step}}{\text{rev}} * 2 \frac{\text{ms}}{\text{step}} = 400 \frac{\text{ms}}{\text{rev}} \quad \frac{1}{.4 \text{ s/rev}} = 2.5 \text{ rev/s}$$



# Prototype



Motor control

Buffer 4050

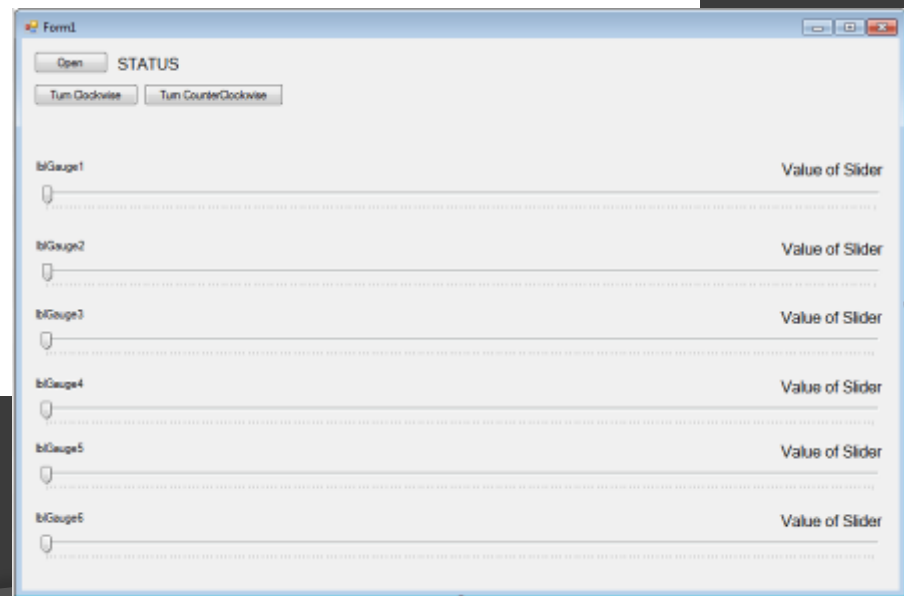
FTDI CHIP



# Implementation

```
private: System::Void gaugeSlider_Scroll(System::Object^ sender, System::EventArgs^ e)
{
    lblValue->Text = gaugeSlider->Value.ToString();
    if(previousValue < gaugeSlider->Value)
    {
        if(m_obUSBComm->GetAvailableGaugesList()[0].type == "FTDI Chip")
            m_obUSBComm->GetAvailableGaugesList()[0].status = m_obUSBComm->MoveClockwise(m_obUSBComm->FindHandle(0));
        else
            m_obUSBComm->GetAvailableGaugesList()[0].status = m_obUSBComm->MoveClockwise(m_obUSBComm->FindHandle(0));
    }
    else if(previousValue > gaugeSlider->Value)
    {
        if(m_obUSBComm->GetAvailableGaugesList()[0].type == "FTDI Chip")
            m_obUSBComm->GetAvailableGaugesList()[0].status = m_obUSBComm->MoveCounterClockwise(m_obUSBComm->FindHandle(0));
        else
            m_obUSBComm->GetAvailableGaugesList()[0].status = m_obUSBComm->MoveCounterClockwise(m_obUSBComm->FindHandle(0));
    }
    previousValue = gaugeSlider->Value;
    if(gaugeSlider2->Enabled && gaugeSlider2->Visible)
    {
        gaugeSlider2->Value = previousValue;
    }
}

FT_STATUS USBComm::Open(int deviceNumber, FT_HANDLE* handle)
{
    FT_STATUS ret;
    ret = FT_Open(deviceNumber, handle);
    if(ret != FT_OK)
    {
        // FT_Open failed
        ShowError("Open Failed");
    }
    return ret;
}
```



# Flight Controls

# Flight Controls

- We need to build a simulated stick, pedals and throttle controls

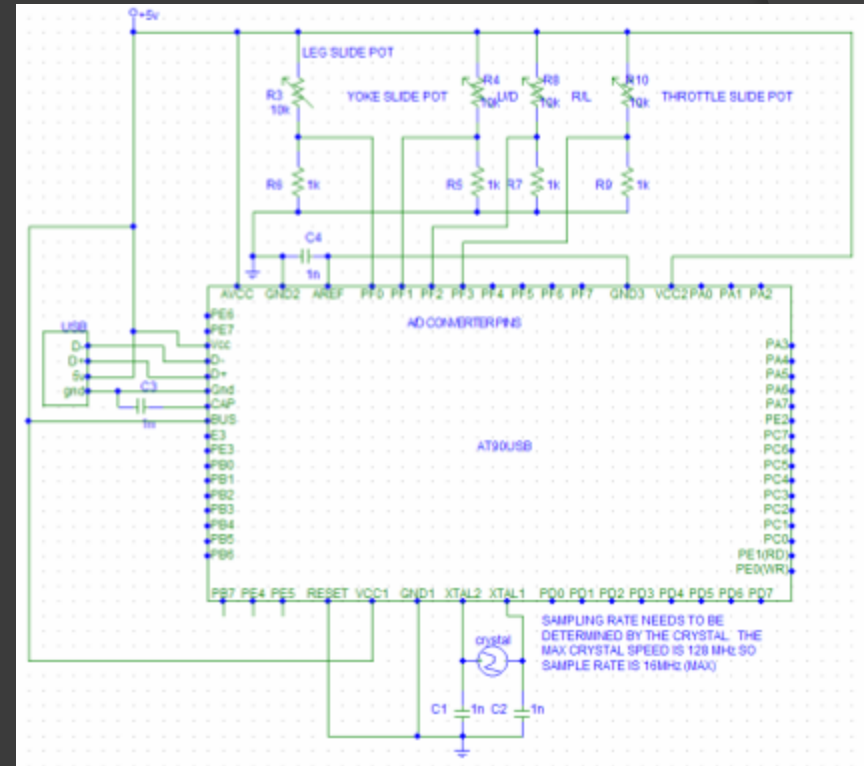


# Control Design

- ⦿ All three controls will share the same basic design characteristics – electrically and mechanically.
- ⦿ The basic design element for each of the devices is the use of a 10k-ohm slide potentiometer.
- ⦿ Four slide potentiometers will be used to create our controls. One for the throttle, one for the pedals (they are already mechanically tied together), and two for the stick (X and Y)

# Controls Interface (Option 1)

- Option 1: Our initial design has us using an Atmel AT90USB microcontroller with onboard USB and A/D pins. There are many good references on writing the code for implementing human interface devices.



## Atmel AT90USB1287

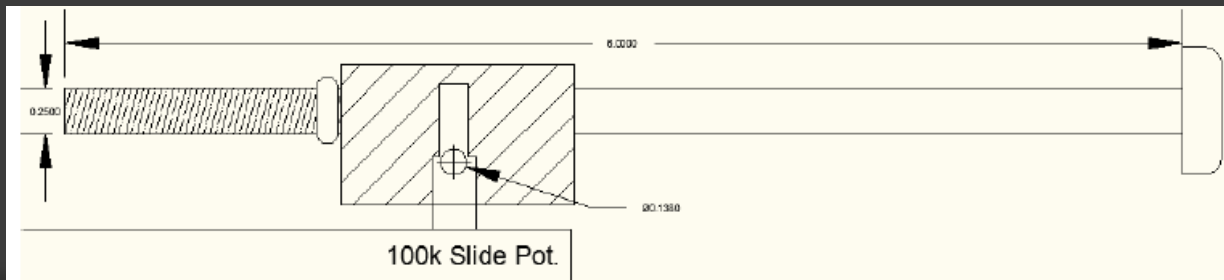
A/D Pins	8
Flash Memory	128 Kbyte
Voltage	2.7-5.5v (Use at 5V)
Onboard USB	Yes

# Controls Interface (Option 2)

- Option 2: Use existing FTDI FT245BL Development board.
- Pros: Lower development time since we won't need to learn an additional development environment, keeping design consistent.
- Cons: Need more external circuitry (ADC and MUX), increased cost due to need to use two FTDI chips.

# Mechanical Design

- ◉ Aside from the throttle, the complete pedals and stick design is largely dependent of what we see once the cockpit arrives (delivery date unknown).
- ◉ The throttle will consist of the following configuration with the pedals and having similar configurations



# Flight Instruments



# Cockpit Overview

- Actual aircraft view of Flight Instrument placement behind instrument panel in the GoBosh G700S.



# Motor Selection

- Two options: Servo or stepper motor
- Servos: Only turns 180-Deg., uses Pulse Width Modulation, use of a 555-timer circuit cannot give precise control over motor position.
- Steppers: Allows us to step through our rotations with no limit on number rotations, very inexpensive

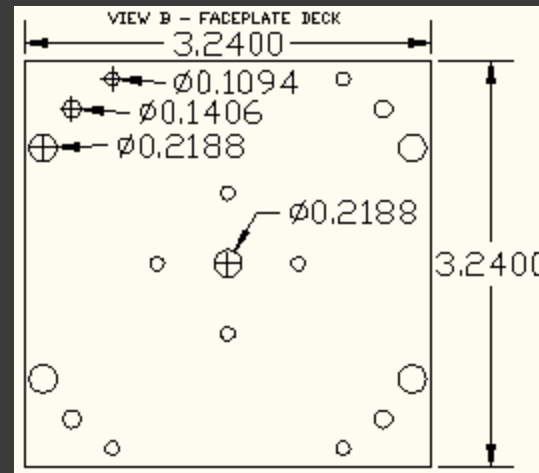
# Servo Motor Prototype

- ◎ It appears however that the Futaba servo does not possess the right response curve in terms of the rotation angle, therefore causing problems with gauges that require extreme movements of the gears (such as an airspeed indicator or altitude indicator)

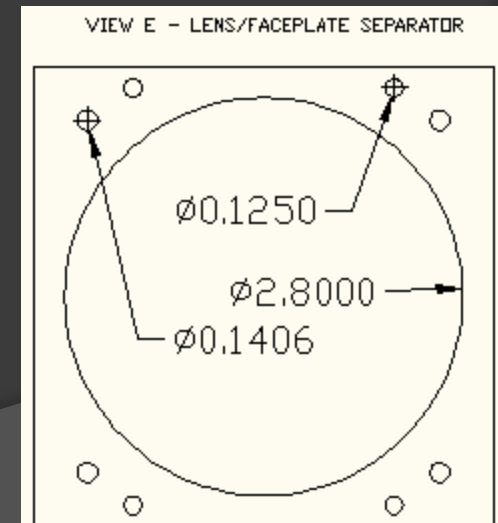


# Common Materials

- 200 step/rev, 1.8 degrees per step, unipolar
- .050" sheet aluminum
- .125" clear acrylic sheet
- OWCP 4537 CDS Photocell
- FTDI FT245BL USB Communication Board



Two of the decks that are common to many gauges



# Airspeed Indicator

- Displays airspeed of the simulated aircraft.
- Due to the nature of the GoBosh, our displayed airspeed range will be 0-160 Knots.
- Requires nearly 360-Deg. range of motion from a single stepper motor.



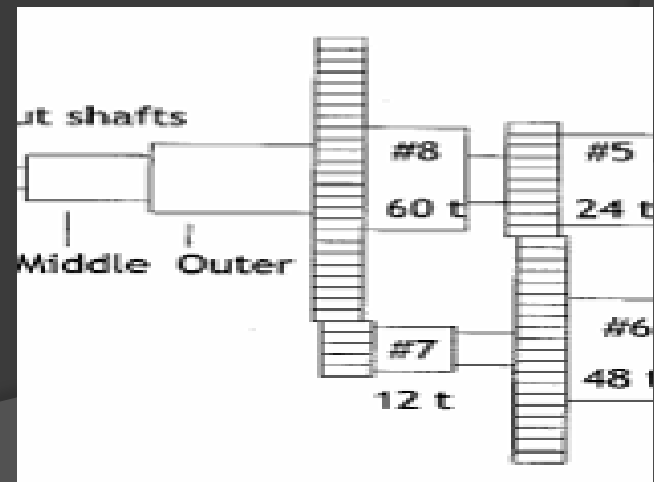
# Vertical Speed Indicator

- Displays vertical speed of the simulated aircraft.
- Displayed output range is 0 to +/-20
- Requires nearly 360-Deg. range of motion from a single stepper motor.



# Altimeter

- Displays the aircraft's altitude above sea level (in feet).
- Requires over 360-Deg. range of motion from a single stepper motor.
- Requires the gearing of the shaft to accommodate the dual needles representing 100 ft and 1000ft increases.
- 1:10 Gear ratio is required.





# Turn Coordinator

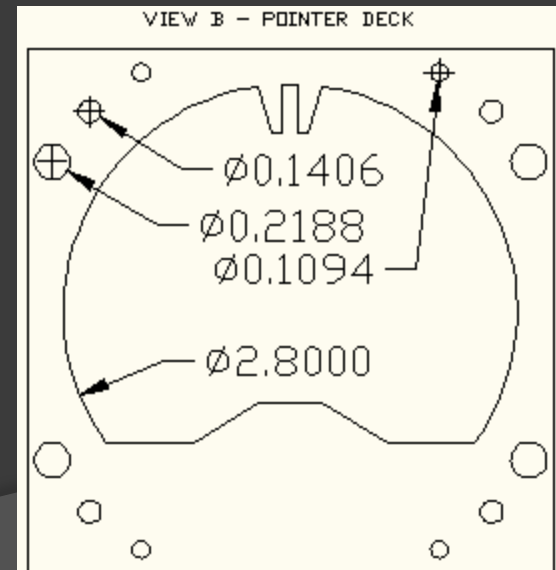
- ⦿ Displays the rate of yaw (turn), roll, and the coordination of the turn.
- ⦿ Requires two stepper motors. One for the level indicator and one of the ball.
- ⦿ The wings on the level indicator are limited to +/- 90-Deg.
- ⦿ The ball moves within 50-Deg. in the ball track.





# Attitude Indicator

- Displays aircraft relative to the horizon.
- Gyroscopic instrument - in order to simulate we will use a moving plate configuration.
- Build or Buy? It's a relatively complex gauge and is available as a kit from manufacturers. Buying a kit would be expensive and require shipping from Europe.



# Heading Indicator

- Displays aircraft heading (compass)
- Requires 360-Deg. of movement with no mechanical stops (flying in a circle) - stepper motor



# Prototype Build



# Power

The power needed to run one stepper motor

$$12 \text{ volt} * 150\text{mA} = 1.8\text{W}$$

We will have at least 6 stepper motors

$$8 \text{ motors} * 1.8 \frac{\text{W}}{\text{motor}} = 14.4 \text{ W}$$

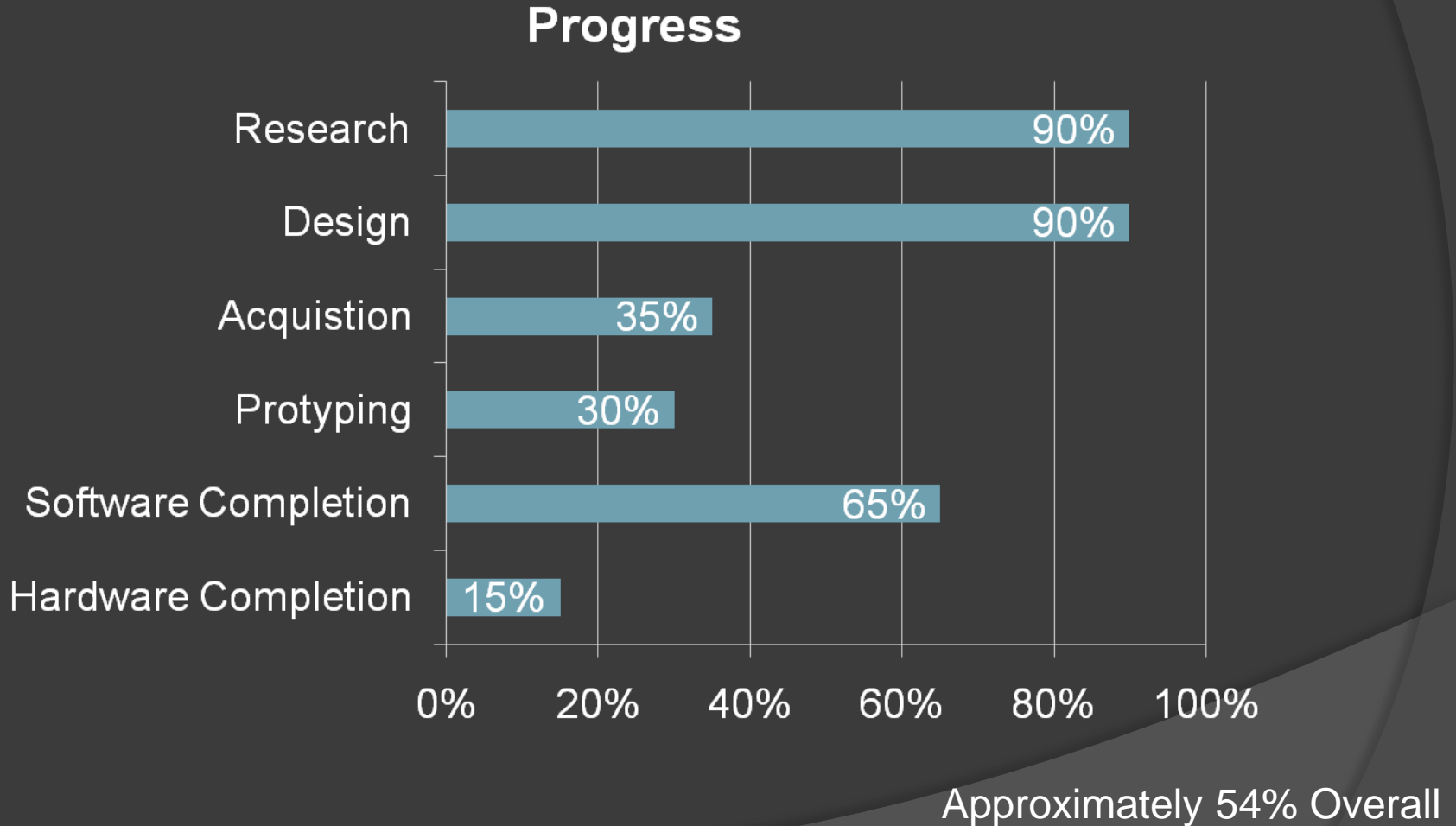
Indicator lights and Slide pot control will run within the USB power rating limits

Will all be possible with the computer power supply using extra molex connectors



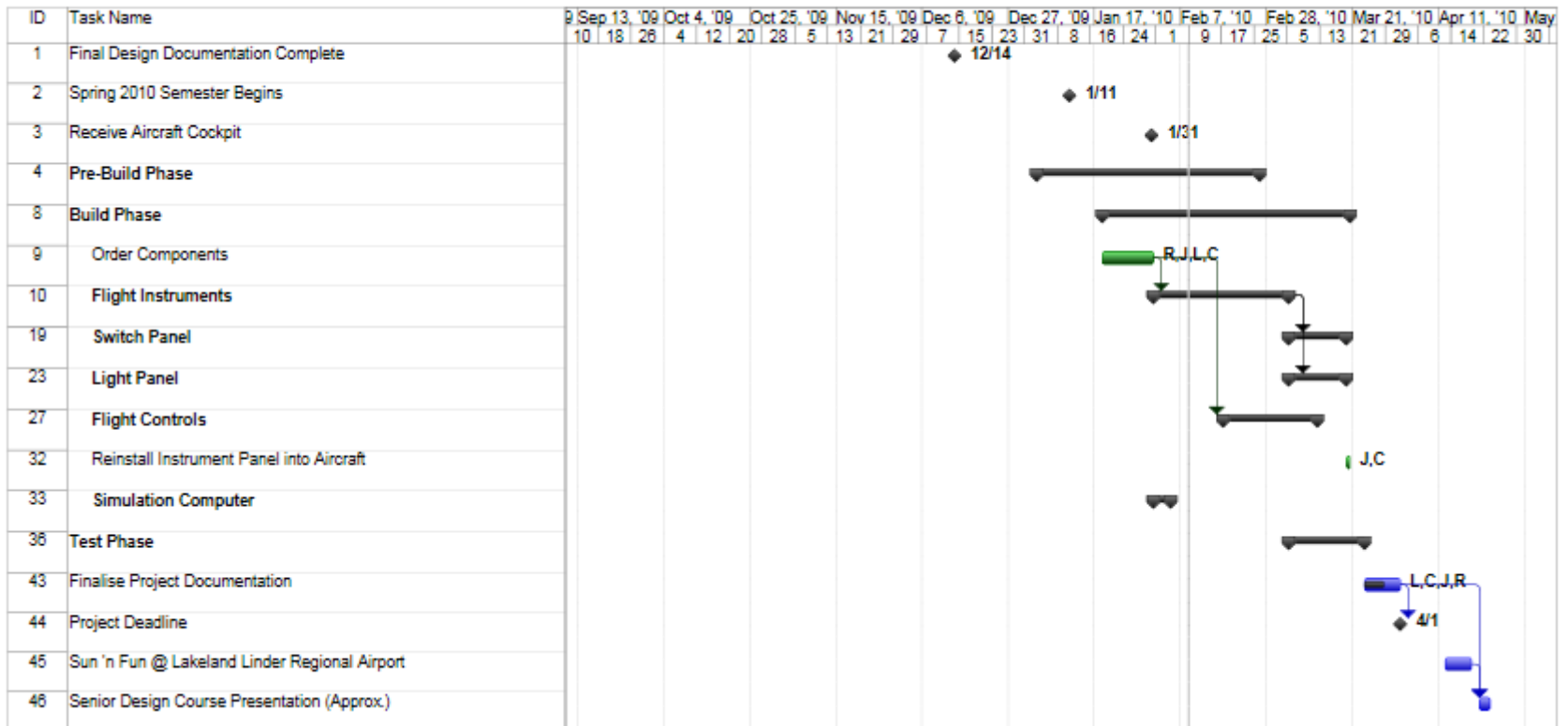
# Administrative Information

# Progress



# Milestone Chart

Senior Design Group 11 - Spring Semester Schedule and Resources



# Budget

Item	Part Number	Quantity Required	Unit Cost	Total Cost
X-Plane 9	N/A	1	\$29.99	\$29.99
Gateway 24" 5ms LCD Display	FHD2402	3	\$199.99	\$599.97
ATI/AFX ATI Radeon HD 5750 1GB	AFX HD-575X-ZNFC	2	\$139.99	\$279.98
ASUS Socket AM3 Motherboard	M4A785TD-V EVO	1	\$73.99	\$99.99
AMD Phenom II X2 Black Edition	HDZ550WFGIBOX	1	\$90.99	\$90.99
4GB DDR3 1066 RAM Kit	OCZ3G10664GK	1	\$72.99	\$72.99
HITACHI Deskstar 160GB HDD	0A38005	1	\$38.99	\$38.99
Lite On IDE DVD-Rom Drive	iHDP118-04	1	\$17.99	\$17.99
1000W ATX Power Supply	EP-1000SC	1	\$99.99	\$99.99
Mid-Tower ATX Case - Black	313-06-C2228	1	\$20.99	\$20.99
Atmel USB Microcontroller	AT90USB1287	1	\$15.05	\$15.05
Slide Potentiometer - 10k, 100mm	N/A	5	\$5.00	\$25.00
FTDI USB Communication Board	FT245BL	4	\$30.00	\$120.00
USB AB Cord	N/A	6	\$2.00	\$12.00
Transistors	2N2222	10	\$0.29	\$2.90
Digital Potentiometer	AD5220	5	\$1.00	\$5.00
Buffer	CD4049	5	\$0.30	\$1.50
Assorted Hardware	Various	-	\$87.00	\$87.00
			Total	\$1,617.04
			Initial Budget	\$1,500.00
			Difference	(\$117.04)



# Expansion Items

- If there is enough time at the conclusion of our project and the sponsor allocates additional funding, there are two features that can be implemented
- Lights: Add panel indicator lights
- Switches: Add panel switches

Questions?