The Kwan-Truc

Todd Hummel
todd.hummel@utah.edu

Travis Reed
twreed83@gmail.com

Project Webpage
www.eng.utah.edu/~thummel/cs3992/index.html

Table of Contents

1 Introduction

The basic idea of the Kwan-Truc is to display streaming video from an RC helicopter on any Windows-based laptop. This will allow the user to control the RC helicopter using the live video feed or record a flight from the helicopter's perspective (or both at once). This could be used for something as serious as a surveillance/reconnaissance system in hazardous situations, or for something as simple as a hobbyist's coolest new toy.

The complete system will consist of a camera that is mounted forward-facing on an RC helicopter, along with the other modules that will manage the wireless transmission of the video from the helicopter to the laptop. These other modules are a microcontroller and transmitter module, and they will be the only additional hardware necessary (aside from the helicopter, camera, and the laptop) to complete the Kwan-Truc system. This is because the RC helicopter itself will still use the normal RC controller for its control system, and the laptop will be using its normal 802.11b interface to receive the video feed.

In order to display the video feed, the Kwan-Truc software on the laptop will provide the necessary functionality to redirect the video feed from the 802.11 interface card to the vendor-supplied Windows driver that comes with the camera. This should allow any camera to be chosen, as long as it has a vendor-supplied Windows driver. By achieving this connectivity, full controllability of the remote camera will be possible from the laptop. This will streamline the process of configuring the camera for a specific resolution and frame rate, as well as the process of displaying and recording the video (because the camera's software can be used).

As a future addition to the project, we may also make it so that you can control the helicopter using the mouse and keyboard on the laptop (like a flight simulator); however, this would require an extra interface between the microcontroller on the helicopter and the motor controls, as well as considerable tuning of the software. Given the time restrictions on this project, this functionality will only be developed if everything else goes very smoothly.

2 Motivation

There are systems readily available which transmit video wirelessly, but many of them are purely analog, and require a television for viewing the video feed. It would be more portable and user-friendly if the system were configurable to work with a laptop, and also eases the digital recording process. Also, this project gives us the chance to learn about the USB, I2C, and SPI interfaces, as well as Windows device driver programming.

3 System Overview

A block diagram of the Kwan-Truc system can be seen in Figure 1. The specific interfaces and the tasks involved in implementing them are described in the sections below. The most important thing to notice now is what all of the hardware components are and how they interact. Also note that all of the hardware for the Kwan-Truc system is located on the helicopter.

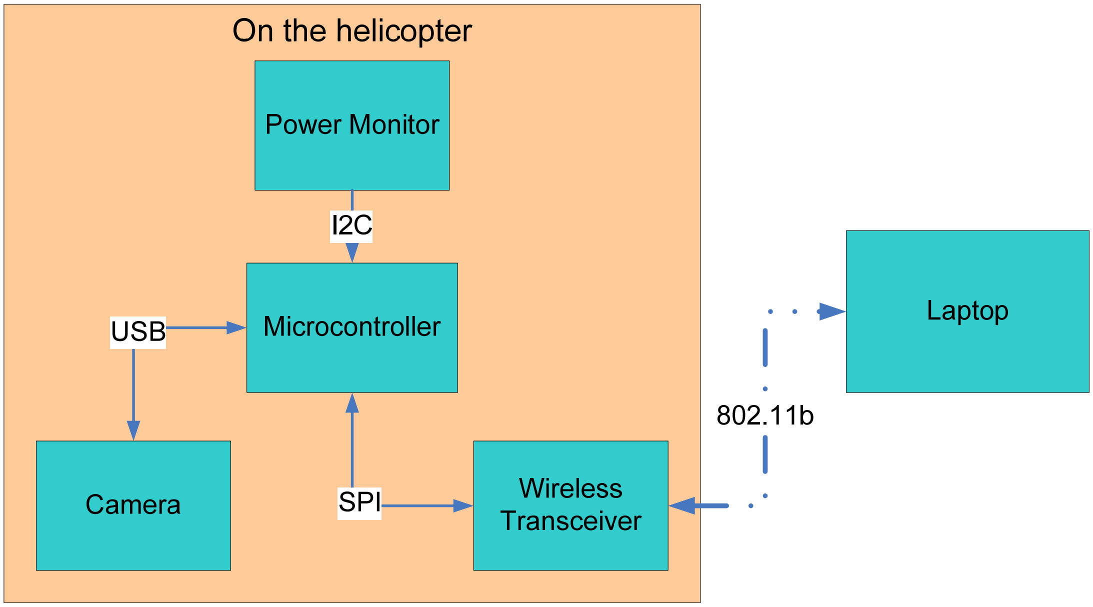


Figure 1 Block Diagram

4 Components and Tasks

Each of the following sections describes a major component of the Kwan-Truc system in terms of its inputs, outputs, and functions performed. Where applicable, specific hardware requirements such as physical plugs/wires, current/voltage specs, etc. are also described here. The task leader, manpower assigned, and estimated time to completion are shown in section 8.

4.1 Camera

The camera interface is USB, and our microcontroller project board has a mini-USB jack built onto it. So for the physical connection, a mini-USB male to USB A female adapter (see Bill of Materials/Vendors List) will be needed. The voltages will be regulated by the USB port on microcontroller project board. The USB standard states that the camera cannot draw more than 500 mA at 5 V, so we will need to supply a maximum of 2.5 Watts for the camera.

The logical connection will be a bit more difficult. In order to initialize the camera to a certain resolution/frame rate and give it the command to begin transmitting video, we will need a USB driver on the microcontroller that knows the correct commands for the camera. Fortunately, we will not have to know these commands because they will ultimately come from the vendor-supplied Windows driver that is located on the laptop.

4.2 Microcontroller/Project Board

The microcontroller that we have chosen is Freescale's ColdFire MCF52221. This part was chosen because it provides the three necessary interfaces: USB, SPI, and I2C. To simplify the construction process, Freescale’s MCF52221 Demonstration Board will be used, which provides the physical port for the USB and also a 40 pin I/O port which gives access to the SPI and I2C ports. Also included is a 3-axis accelerometer that could potentially be used to provide the acceleration of the helicopter to the laptop. There are also several LED's and push-button switches which may be useful for debugging.

The demo board takes anywhere from +5 to +16 VDC, and performs its own voltage regulation. The documentation isn't specific on the current draw of the board, but it cannot exceed 500 mA since the board has an option of being powered from a USB cable (which has a limit of 500 mA).

The microcontroller will have the task of controlling the flow of data between all of the components on the helicopter (as seen in Figure 1). The specific interfaces will be described in section 5 below.

4.3 Wireless Transceiver

The Quatech WLNG-AN-DP102 wireless transceiver module was chosen for our project because it greatly simplifies the task of integrating an 802.11b interface into our embedded application. It is configurable using a built-in web server which can be accessed from the browser of any PC or laptop with a wireless card. All that is needed to communicate with the microcontroller is to use the built-in SPI port of the transceiver.

The transceiver requires 3.3 VDC with enough current to provide for the maximum peak inrush during startup (1900 mA). After that, it only requires 420 mA during transmission. The transceiver provides its own internal voltage regulation for its 2.5 V parts, so there is no need for an external regulator.

4.4 Power System

The power system consists of the batteries and a power monitor. Four common Energizer NH15-2200 NiMH AA batteries connected in series will provide about 4.8 V nominal, which is sufficient for the microcontroller, camera, and wireless transceiver (as the USB standard specifies 4.75 V as the minimum). The data sheet for these batteries indicates that they should operate for approximately 1.5 hours at the typical current draw of 1420 mA (see Table 1 for analysis).

Table 1 System Power Analysis

Part	Maximum current draw (mA)	Typical current draw (mA)	Voltage
Camera	500	<500	5
Microcontroller	500	<500	5-16
Transceiver	1900	420	3.3
Total	2900	1420	5

The Maxim DS2746 power monitor will be used to detect when the batteries are running low, so that the operator will have some warning before the video feed is broken. This way he may return the helicopter to somewhere in view or land it safely.

The DS2746 will communicate with the microcontroller by way of a standard I2C interface. Voltage, bidirectional current, and accumulated current measurement data can be provided to the host processor in either 8 or 16 bit words.

4.5 Laptop application

The laptop will be any standard laptop with an 802.11b interface card. We will be writing a software application that will receive the incoming data stream from the camera and pipe it to the Windows device driver (see section 4.6 below). After this, the computer should be able to access the remote camera in the same way as if it were physically plugged in to the computer (therefore all further display/record capabilities may be deferred to the vendor-supplied camera software).

4.6 Laptop driver

In order for the camera to be initialized correctly, it must be configured and turned on, just as if it were plugged in to the computer directly. To easily achieve this, we have chosen to write a Windows device driver that will sit between our application and the camera's vendor-supplied device driver. This driver will have the purpose of creating a "virtual USB cable" between the remote camera and the laptop, greatly simplifying the need for internal working knowledge of the camera. To create this "virtual USB cable," our application will read the data from the wireless card, decide if it is meant to go to the USB driver (battery status messages won't for example), and then pass the information to our driver. This will then redirect the information to the USB camera driver, where results and other commands will then flow back through our driver to our application and back across the wireless interface. This full-duplex connection will be difficult to write, but it has the benefit of allowing us to use any USB camera with a Windows driver without having to know its interface.

4.7 Helicopter(s)

Many of the above parts have no specification as to how much they weigh, but a conservative estimation is shown in Table 2. Therefore, the only requirement for the helicopter is that it must be able to lift about 270 grams worth of equipment without becoming completely unstable. Also, because neither us have any experience flying RC helicopters, we have included a practice helicopter in the Bill of Materials/Vendors List so that we can have something a little cheaper to abuse.

Table 2 System Weight Analysis

Part	Weight (g)
Camera	<30
Microcontroller	<60
Transceiver	<60
Batteries	120
Total	270

We are still unsure of exactly which helicopter models we are going to use, but we will be working with the staff at Some Dude's Hobbies in West Jordan, UT to determine what we need.

5 Specific Task Interfaces

This section will describe some of the specific details of the interfaces between the components. These details will include information such as the hardware provided capabilities, logical interface to the hardware, and commands/instructions the software will control.

5.1 Camera-Microcontroller (USB)

The camera to microprocessor interface will be a standard USB 2.0 full-duplex link. The physical connection will require a mini-USB male to USB A female adapter because the project board we have chosen only has a mini-USB port available. Once the physical connection is made, it should be fairly simple to configure the microcontroller to use the USB interface. A preliminary look at the MCF52221's Reference Manual shows that the USB interface is controlled as a memory-mapped I/O device. Thus, the USB "driver" on the microcontroller will only have to set control registers and read memory in order to communicate over the USB bus. The manual also claims to support full speed USB transfers of 480 Mbps.

5.2 Microcontroller-Wireless Transceiver (SPI)

The microcontroller and wireless transceiver communicate over a Serial Peripheral Interface Bus (SPI) Bus. SPI is a four-wire serial bus that allows for full duplex communication between the two parts. The microcontroller will be the bus master and the wireless transceiver will be a bus slave. The microcontroller controls the data transfers on the bus. Our data rate across this bus is determined by the SPI interface on the microcontroller, which allows for high-speed clocking of up to 16.6 Mbps.

The implementation of this interface will be done in three parts. First, the physical wires must be correctly hooked up. Second, the wireless transceiver must be configured correctly for using its built-in SPI port. This is done through the web server interface of the wireless transceiver. Third, the microcontroller must be programmed to send data to the wireless transceiver. This will be done using the QSPI (Queued SPI) module that is integrated into the microcontroller. Once the data is sent to the wireless transceiver, it is automatically packaged and sent across the 802.11 interface.

5.3 Power Monitor- Microcontroller (I2C)

The microcontroller and power monitor chip will communicate over a serial interface known as I2C. I2C "is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange, minimizing the interconnection between devices." The microcontroller will be the bus master and the power monitor chip will be a bus slave. As such, the microcontroller initiates all data transfers. The driver we write on the microcontroller will simply have to configure the correct control registers, wait, and read the result back from memory. Speed is not a concern for this interface as it will only have to be read intermittently (every 1 second at most).

5.4 Wireless Transceiver-Laptop application (802.11b)

Given the hardware we have chosen, this is probably the simplest interface. There is no hardware connection necessary (assuming the wireless transceiver is powered correctly), so most of the other work just involves making a connection and reading the data off the connected socket. Because the wireless transceiver conforms with the 802.11b standard, everything should work that easily.

5.5 Laptop application-Camera driver (Windows Device Driver)

The laptop application will receive information from the airborne system through the wireless device on the laptop (see above). This information will be fed to a Windows device driver that is capable of finding, loading, and communicating with the camera's vendor-supplied device driver (assuming it is installed on the laptop). This supplied driver will then be able to function as it normally does when the USB camera is plugged in directly to the laptop. Any commands given by the vendor-supplied driver will then be able to flow the other direction as well (back to the laptop application).

6 Integration Testing Strategy

This section describes our basic integration plan: how the tasks are brought together to form larger components and how they are incrementally tested.

6.1 Microcontroller-Wireless Transceiver-Laptop

Once the SPI interface to the wireless transceiver is working, a connection can easily be made to the laptop by way of the 802.11 interface. This should be simple to test, as the application on the microcontroller can simply send text strings to the laptop application, which can easily be matched up when they are received. Another possibility would be to connect one of the spare serial ports on the microcontroller to a serial port on the laptop. Then you could send the information through both interfaces and confirm that they match. The downside with that method is it requires writing an unneeded port driver on the microcontroller.

6.2 Camera-Laptop application

Having already tested the interface from the processor to the laptop, adding the camera would mean that we can see information on the laptop (probably just as a bit stream) that comes from the camera. At this phase, we will not be displaying video, so the biggest trick of this portion will be to know if the information we are looking at is what was indeed sent from the camera; however, if the interface from section 6.1 above is working correctly, there shouldn't be any concern.

6.3 Camera-Camera driver

By this point we should know that we are getting information from our camera to the laptop. We will then need to incorporate the camera driver into the mix. Information from the wireless transmission will be fed into the camera driver. Debugging this interface will probably be very difficult, but testing will be rather simple. If we see the video when we activate the camera, then it is working…otherwise we have more work to do.

6.4 Power Monitor-Laptop application

Once the interface from section 6.1 above is complete, it should be fairly simple to integrate it with the I2C interface. This will be testable in the same way as 6.1; however, a better option would be to connect everything and wait until you observe the power readings start to drop and that the application notifies the user appropriately.

7 Group Communication Plan

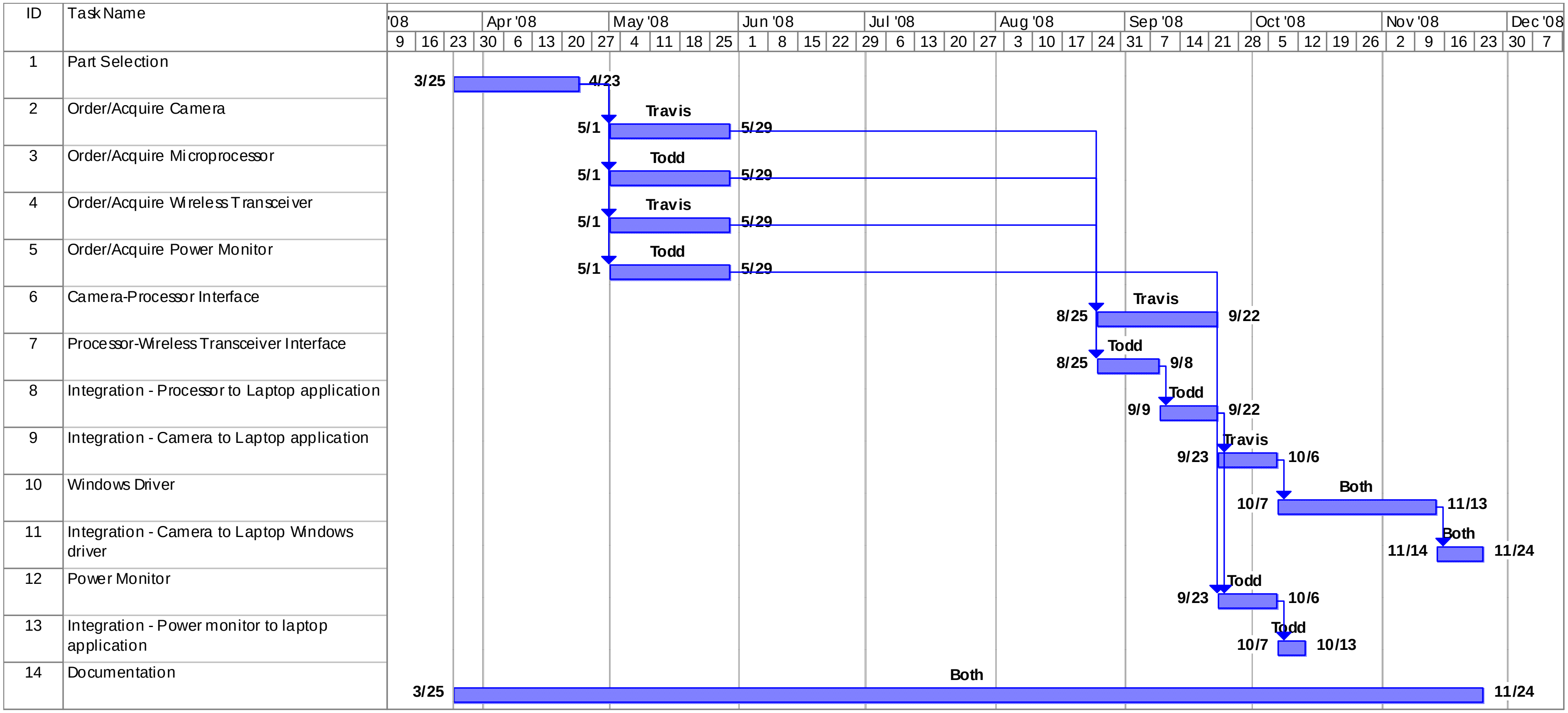
We will meet officially once a week. Todd will update the website with meetings. This log will include the following:

- Date, time, and duration of meeting (attendance is not necessary, if only one of us is there, it's not a meeting)
- Current tasks (referenced from our Microsoft Project schedule)
- Tasks completed since last meeting
- Progress on current tasks
- New tasks (if any)
- Main discussion points
- Decisions made
- Rating of overall progress
- Number of hours spent on project since last meeting

We work together at L-3 Communications, so we will likely have very frequent progress updates. Will be using Microsoft Project to manage scheduling and tracking of we're doing on the overall schedule. Group communication will be achieved via phone, email, and meeting in person.

8 Schedule

Below is a Gantt chart showing the full tasking breakdown and schedule. For specific information about each task, see sections 4, 5, or 0 above.



ID	Task Name
1	Part Selection
2	Order/Acquire Camera
3	Order/Acquire Microprocessor
4	Order/Acquire Wireless Transceiver
5	Order/Acquire Power Monitor
6	Camera-Processor Interface
7	Processor-Wireless Transceiver Interface
8	Integration - Processor to Laptop application
9	Integration - Camera to Laptop application
10	Windows Driver
11	Integration - Camera to Laptop Windows driver
12	Power Monitor
13	Integration - Power monitor to laptop application
14	Documentation

Figure 2 Project Gantt chart

9 Risk Assessment

This section describes the risk level of each task, and a mitigation plan in case of failure.
§ risk level: high, medium, low, none
§ risk nature: lack of knowledge, lack of experience, complexity, …
» sometimes you aren't in control – e.g. dependent on a vendor
» need to minimize this with a back up plan
• which is why vendor 2nd sourcing is important
§ mitigation plan
» what happens when the risk causes failure
» you will be required to have one unless risk = none
• e.g. it would be cool to do this but if we can't pull it off then the overall project still works but is just missing a neat feature
• or an alternate method that is more sure but less optimal for some reason.>

9.1 Camera-Microcontroller Interface

Risk Level: low. The risk is mainly with the unfamiliarity with the USB protocol and controlling the USB bus using the microcontroller interface. Communication will probably be fairly simple to establish, but we will need to know what messages to send. These will ultimately come from the vendor-supplied driver on the laptop, but for initial testing it may be difficult to know if communication is truly established.
Mitigation Plan: see the mitigation plan of section **Error! Reference source not found.** for details of how this interface would be revised in case of failure.

9.2 Microcontroller-Wireless Transceiver Interface

Risk Level: none. We each have experience using the SPI protocol, so writing the driver for this should not be a problem. The only potential reason for risk in this case is unfamiliarity with the wireless transceiver that we will be using (e.g. Is it really as easily configurable as the company claims?); however, there is no reason to believe that it won't work as advertised.

9.3 Wireless Transceiver-Laptop Interface

Risk Level: none. As this interface is simply a Windows application that can communicate over a normal TCP/IP socket, there is no risk involved. Both of us have done this several times.

9.4 Power System/Battery Monitor-Microcontroller Interface

Risk Level: low. The power system will not take much effort to connect together properly with the correct voltages. The only thing we are unsure of is how difficult it will be to supply the peak inrush current (1.9 A) that the wireless transceiver requires at startup. The I2C protocol is very standard and well documented, so there should be no problem at all in communicating with the battery monitor.

9.5 Laptop application-Camera driver (Windows Device Driver)

Risk Level: High. Neither of us have any experience whatsoever with writing Windows device drivers. The upside is that we know that it is possible to do what we have proposed because there are existing commercial software packages that do it. But as of now, we don’t really know how we are going to do this. The plan is just to spend a lot of time this summer learning how to program device drivers so that we can hopefully succeed in this part of our project.

Mitigation Plan: Should this prove to be too difficult, we have two options. The first option is to find a camera that has a documented interface specification so that we can develop our own embedded driver for the camera which would reside on the microcontroller. That would be a much simpler task than the Windows device driver we are hoping to develop. The second option would be to 'record' the initialization routine for the camera by plugging the camera into a computer and observing the USB data transfer wires with a logic analyzer. We could then store this routine in the microcontroller and run it each time the system is started in order to initialize the camera as desired. The downside with either of these options is that we would no longer be able to use just any USB camera with a Windows driver. We would be limited to the one camera for which we choose to implement this.

10 Bill of Materials/Vendors List

Component	**Vendor**	**Contact Info**	**Part Number**	**Unit Cost**	**Qty.**	**Total Cost**
Microcontroller Project Board	Freescale	1-800-521-6274	M52221DEMO	119.00	1	119.00
	Future Electronics	1-800-FUTURE1 1-800-675-1619	M52221DEMO	119.00	1	119.00
Wireless Transceiver	Quatech	1-800-553-1170	WLNG-AN-DP102	129.00	1	129.00
	Mouser	1-800-346-6873	WLNG-AN-DP102	109.06	1	109.06
Antenna	Quatech	1-800-553-1170	ACH2-AT-DP003	11.00	1	11.00
	Digikey	1-800-DIGIKEY	590-1007-ND	12.38	1	12.38
Camera	Local store…		TBD	~25.00	1	~25.00
USB adapter	Vernier	1-888-837-6437	USB Standard-A to Mini-A Adapter	5.00	1	5.00
Power Monitor	Maxim	1-888-629-4642	DS2746	Sample/ 1.94	1	Sample/ 1.94
	Digikey	1-800-DIGIKEY	DS2746G+-ND	3.82	1	3.82
Batteries	Local store…		NIH-2200	7.50	4	30.00
Practice Helicopter	Some Dude’s Hobbies	801-233-6565	TBD	~40.00	1	~40.00
Final Helicopter	Some Dude’s Hobbies	801-233-6565	TBD	~250.00	1	~250.00

11 References

Freescale Semiconductor. M52221DEMO Product Summary Page <http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=M52221DEMO>

Freescale Semiconductor. MCF5222X Product Summary Page <http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MCF5222X>

Freescale Semiconductor. MCF52223 ColdFire® Integrated Microcontroller Reference Manual. <http://www.freescale.com/files/32bit/doc/ref_manual/MCF52223RM.pdf>

DPAC Technologies. AirborneTM Wireless LAN Node Module Data Book. <http://www.dpactech.com/docs/wireless_products/Airborne_WLN_DataBook_RevE.pdf>

Dallas Semiconductor. DS2746 Data Sheet.<http://datasheets.maxim-ic.com/en/ds/DS2746.pdf>

Vernier Software and Technology. USB Standard-A to Mini-A Adapter. <http://www.vernier.com/accessories/access.html?usb-mini&template=basic.html>

Energizer Battery Company. Energizer NH15-2200 Product Datasheet. <http://data.energizer.com/PDFs/nh15-2200.pdf>