THE EQUINE DISTRESS MONITOR PROJECT

by

Luke B. Peacock

A report submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

Dr. Chris J. Winstead Major Professor	Dr. Donald L. Cripps Committee Member

Dr. Wei Ren
Committee Member

UTAH STATE UNIVERSITY
Logan, Utah

2012

Abstract

The Equine Distress Monitor Project

by

Luke B. Peacock, Master of Science

Utah State University, 2012

Major Professor: Dr. Chris J. Winstead
Department: Electrical and Computer Engineering

Colic is a very common symptom that effects many horses. Sometimes colic is an indicator of some very serious medical conditions that, if left untreated, could result in the death of the horse. A cast horse is a horse trapped in a prone position. Cast horses are also at risk of serious injury or even death. The causes of these two conditions are so numerous that a horse's risk of being affected, while very preventable, is significant. This illustrates a need for constant monitoring of high-risk, high-sentiment, or high-cost horses.

The Equine Distress Monitor (EDM) system is a non-evasive electronic long-term monitoring system that senses horse movements and analyzes them for indicators of colic and casting conditions. When colic or casting events are sensed, messages are sent through a mesh network to a base station computer where notification of the event can be sent to appropriate personnel. The EDM thesis project included the hardware and software development of mesh network devices and applications that accomplish the above mentioned monitoring and notification.

(91 pages)

Public Abstract

The Equine Distress Monitor Project

by

Luke B. Peacock, Master of Science

Utah State University, 2012

Major Professor: Dr. Chris J. Winstead
Department: Electrical and Computer Engineering

Imagine if the owner of a beloved horse enters the stable one morning and finds the horse laying on its side, struggling to breathe. The veterinarian is called immediately, but it is too late. The animal dies. It is determined that sometime during the night, the horse developed a serious bowel obstruction. If only the care-giver had been aware of this at the time, medical treatment could have been administered and the horse could have survived. Twenty-four-hour surveillance of animals is costly, and often impossible, but if some sort of device on the horse could alert a care-giver of a dangerous situation, many animal lives could be saved. The Equine Distress Monitor (EDM) project was to develop a system designed to give horse care-givers immediate notification of suspicious activity. This could allow the person adequate time to assess the situation. The EDM system consists of sensors and message routing devices. A sensor is placed on the horse's halter, detecting specific movements that have been researched to indicate conditions which are threating to the horse's life. A message is sent to the horse's caregiver informing of these conditions.

To my family...

Acknowledgments

I would like to thank a number of people for their contributions, advise, and support as I worked on this project.

First, I would like to thank USTAR for their financial contributions for the project. I feel that I had plenty of financial resources to aid me in furthering the development of the Equine Distress Monitor system.

I would also like to thank Dr. Winstead for allowing me the opportunity to turn this project, which started out as just a second source of income for me, into a fulfilling and educational experience. Dr. Winstead also contributed a lot of time helping test the system and in the software development aspects of the project, as well as extensive advisement in the engineering process. He also did all the leg work in getting funding for the project, facilitating technical support from the USU facilities department, and getting support from the Utah State University Equine Education department.

I would like to thank Mitchel Humpherys for all the work that he put into the software, the firmware, and some of the hardware development of the project. Mitch provided extensive expertise and hard work that was invaluable in progressing the project. I could not have done it without him.

I would like to thank those that developed the first version of the EDM system.

Lastly, I would like to thank my wife, Mary, for her patience, love, and support throughout the project. She also contributed by helping correct grammatical errors while I wrote this report.

Luke B. Peacock

Contents

Page

Abstract . . . . **iii**

Public Abstract . . . . **iv**

Acknowledgments . . . . **vi**

List of Figures . . . . **ix**

Acronyms . . . . **xii**

1 Introduction and Background **1**
1.1 Colic 1
1.1.1 Behavior Associated with Colic 1
1.1.2 Importance of Early Detection of Colic 2
1.2 Casting 2
1.2.1 Behavior Associated with Casting 3
1.2.2 Importance of Early Detection of a Cast Horse 3
1.3 Current Solution 3
1.3.1 Equine Inverted Posture Alarm 5
1.3.2 Center of Mass Movement and Mechanical Energy Fluctuation During Gallop Locomotion in the Thoroughbred Racehorse 5
1.3.3 Monitoring System for Animal Husbandry 5
1.3.4 Animal Instrumentation 6
1.3.5 Foaling Alarm 6
1.3.6 Livestock Breeding and Management System 7

2 The Equine Distress Monitor System **8**
2.1 The Solution 8
2.1.1 Horse Behavior Indicating Colic or Casting 8
2.1.2 Healthy Horse Behavior 9
2.2 Alpha Prototype of EDM 9
2.2.1 MeshBean Development PCBs 10
2.2.2 Testing 11
2.3 The EDM Project 11
2.3.1 Project Objectives and Contributions 12
2.3.2 Sensor Devices 13
2.3.3 Router Devices 14
2.3.4 Coordinator Devices 15
2.3.5 Base Station 15

3 Hardware Design **16**
3.1 Programming Boards 18
3.2 Sensor Boards 18
3.2.1 Sensor Version 1.1 20
3.2.2 Sensor Version 1.2 21
3.2.3 Sensor Version 1.3 22
3.3 Router Boards 23
3.4 Solar Charging Circuits 25
3.5 Coordinator Boards 27
3.6 Base Station Hardware 27

4 Embedded Firmware Design **29**
4.1 BitCloud Stack 29
4.1.1 ZigBee Device Objects (ZDO), Application Support Sub-Layer (APS), and Network Layer (NWK) 30
4.1.2 Board Support Package (BSP) and Hardware Abstraction Layer (HAL) 30
4.1.3 Task Manager 31
4.2 Shared Application Components 31
4.2.1 Source Code: edm_util.c 31
4.2.2 Configuration and Makefile 33
4.2.3 Source Code: EDM.c 33
4.3 Sensor Application Components 33
4.4 Router Application Components 35
4.5 Coordinator Application Components 35

5 Software Design **37**
5.1 ZigBee Base Station 38
5.2 SQLite Database Engine - EDMData.cpp 38
5.3 Web Server 39

6 Performance Analysis **43**
6.1 System Reliability 43
6.1.1 Radio Signal Reliability 44
6.1.2 Router Power Reliability 46
6.1.3 Sensor Power Reliability 46
6.1.4 Network Hand-offs vs Sensor Processing 47
6.1.5 Horse Behavior Detecting Algorithms 56
6.2 Colic Detection Analysis 58

References **59**

Appendices **62**
Appendix A Sensor Board Design Documents 63
Appendix B Router Board Design Documents 71
Appendix C Coordinator Board Design Documents 75

List of Figures

Figure Page

1.1 Example of a “belly-band” sensor device. 4
1.2 Injury risk posed by “belly-band” sensor. 4
2.1 Alpha prototype sensor. 9
2.2 MeshBean PCB. 10
2.3 EDM mesh network communication model. 11
2.4 Mounted sensor device. 13
2.5 Mounted router device. 14
2.6 Mounted base station and MeshBean coordinator. 15
3.1 Solder paste stencil mounted in stencil holder from Stencils Unlimited. . . . 17
3.2 Gold-Place manual pick-and-place system. 17
3.3 Re-flow oven. 18
3.4 Atmel AVR JTAGICE mkII programmer. 19
3.5 EDM programming board. 19
3.6 Sensor PCB mounted inside enclosure. 21
3.7 Sensor board V1.1 with modifications. 22
3.8 Sensor board V1.2 with a second tilt switch added for bi-directional tilt sensing. 23
3.9 Sensor board V1.3 (panelized). 24
3.10 Router board V1.0. 25
3.11 Solar charging circuit implemented with a Valor voltage regulator. 26
3.12 Solar charging circuit implemented with a Recom voltage regulator. 26
3.13 Coordinator board V1.0 with modification to power device from USB. . . . 28

4.1 Simplified architecture diagram of the BitCloud stack. 31

4.2 Original EDM data packet format. 34

4.3 State diagram describing general network flow of all EDM devices. 34

4.4 State diagram describing local operation of router devices application. . . . 36

4.5 State diagram describing local operation of coordinator device application. . 36

5.1 Web server user interface front page. 40

5.2 Web server user interface Data Explorer page. 41

5.3 Web server user interface Statistics page. 42

6.1 General map of the USUEEC router and coordinator/base station setup. . . 44

6.2 Router and coordinator/base station layout photos. 45

6.3 Original sensor state diagram. 49

6.4 Data illustrating data gaps found in hand-off testing. 51

6.5 Final sensor configuration power measurement while in "wait for end of tilt" state. 52

6.6 Original sensor configuration power measurement while in "wait for end of tilt" state. 53

6.7 Original sensor configuration power measurement searching for a mesh network host. 54

6.8 Final sensor state diagram. 57

6.9 Final EDM data packet format. 57

A.1 Sensor Schematic V1.0. 63

A.2 Sensor Layout V1.0. 64

A.3 Sensor Schematic V1.1. 65

A.4 Sensor Layout V1.1. 66

A.5 Sensor Schematic V1.2. 67

A.6 Sensor Layout V1.2. 68

A.7 Sensor Schematic V1.3. 69

A.8 Sensor Layout V1.3. 70

B.1 Router Schematic V1.0. 71

B.2 Router Layout V1.0. 72

B.3 Router Schematic V1.1. 73

B.4 Router Layout V1.1. 74

C.1 Coordinator Schematic V1.0. 75

C.2 Coordinator Layout V1.0. 76

C.3 Coordinator Schematic V1.1. 77

C.4 Coordinator Layout V1.1. 78

Acronyms

AI	Animal Instrumentation
API	Application Programming Interface
APS	Application Support Sub-Layer
BSP	Board Support Package
CPE	Canine Pose Estimation
DECT	Digital Enhanced Cordless Telecommunications
DIP	Dual In-Line Package
DSD	Dominant Spectral Density
EDM	Equine Distress Monitor
EIPA	Equine Inverted Posture Alarm
EMI	Electromagnetic Interference
FA	Foaling Alarm
FSD	Frequency of DSD Movement
GPIO	General Purpose Input/Output
GSM	Global System for Mobile Communication
HAL	Hardware Abstraction Layer
IC	Integrated Circuit
IP-65	International Protection Rating
IRQ	Interrupt Request
JTAG	Joint Test Action Group
LAN	Local Area Network
LBM	Livestock Breeding and Management
LDO	Low Drop Out
LQI	Link Quality Indicator
mAh	Milli-Ampere-Hour
MCU	Multi-Chip Unit
MMCX	Micro-Miniature Coaxial

NiMH	Nickel-Metal Hydride
NWK	Network Layer
PC	Personal Computer
PCB	Printed Circuit Board
RF	Radio Frequency
RFID	Radio Frequency Identification
RSSI	Received Signal Strength Indication
SDK	Software Development Kit
SMA	SubMiniature Version A
SMS	Short Message Service
TMC	Telemedical Center
UART	Universal Asynchronous Receiver/Transmitter
UID	Unique Identifier
USB	Universal Serial Bus
USUEEC	Utah State University Equine Education Center
US&R	Urban Search and Rescue
ZDO	ZigBee Device Objects

Chapter 1
Introduction and Background

1.1 Colic

The number one cause of natural death in horses is attributed to colic. Colic, by definition, means abdominal pain [1]. The abdominal pain is not necessarily the cause of death to the horse, but rather, it is a symptom of disease or distress to the animal. Causes of colic in horses are very diverse, ranging from twisted intestines, to worm infestation, or over-feeding, and so on. Most of these issues cause intestinal blockage, which results in colic [1]. Many of these conditions are very serious, and can cause permanent harm or even death.

1.1.1 Behavior Associated with Colic

- Excessive rolling or lying down [2]
- Restlessness [2]
- Inactive sweating [2]
- Pawing at the ground [2]
- Kicking at the belly [2]
- Looking at or biting at the belly [2]
- Change in disposition (usually dull, sleepy, or depressed) [2]
- Lack of appetite [2]
- Labored defecation (or inability to defecate) [2]

- Little-to-no sounds from belly region [2]
- Sitting like a dog [2]
- Stretching as if to urinate [2]

1.1.2 Importance of Early Detection of Colic

Early detection is the key to saving a horse from possible harm or death by the various ailments that cause colic. For several of the conditions, the Merck Veterinary Manual advocates early detection, as the actual cause of the colic episode can be difficult and lengthy to diagnose. Most cases of colic require some form of medical treatment [1]. Some colic episodes are serious enough that immediate surgery or medication are required, while other instances are so mild that simply walking the horse for fifteen minutes is adequate to relieve the horse's distress. Despite the fact that milder cases are not life-threatening to the horse, early detection and treatment are desirable to lessen the time the horse is suffering. Furthermore, early detection can save the owner money in veterinary costs by decreasing prolonged colic events which result in more severe damage to the animal. As many horse owners are emotionally attached to their animals, early detection and resolution becomes even more desirable.

1.2 Casting

Casting, or a cast horse, refers to a horse that is stuck on the ground on its side or back [2]. This can happen when a horse rolls and becomes tangled in, or trapped under an object such as a fence or building, or when a pregnant mare rolls and positions her legs uphill. Serious contusions can occur when a cast horse thrashes and entangles itself. Being cast for a long period of time is also dangerous because the weight of the horse's organs can cause suffocation. If found quickly, the care-giver can get the horse on its feet, reducing the risk of serious injury or death.

1.2.1 Behavior Associated with Casting

- Laying down for long periods of time
- Thrashing while laying down (fast, high energy movements)
- High level of activity while laying down (slow, labored, movements)

1.2.2 Importance of Early Detection of a Cast Horse

As with colic, time is a critical factor when dealing with a cast horse. The horse anatomy is such that laying down for long periods of time is harmful, and in some cases, fatal. If caught early, the care-giver can successfully remove the horse from the obstruction, preventing the horse from suffocating, as well as injuries caused by thrashing. In most cases where a cast horse thrashes, the more time that the animal spends in the casting position, the more injuries the horse will incur. In the event that the casting is caused by colic, it is still vital to return the animal to its upright position before further medical treatment can take place. In short, the sooner a cast horse is aided out of its position, the better.

1.3 Current Solution

For 100% detection and prevention of colic and casting, 24-hour surveillance of the horse is necessary. Using personnel for constant monitoring is impractical and expensive. Large facilities often employ staff that performs periodic checking, 24-hours a day. As colic and casting events can occur in between checking periods, the window of opportunity to save the animal from harm is often missed. Currently, there are few electronic systems that can detect colic and casting events. Most of them have originated from foaling monitors. Foaling monitors utilize devices that transmit alarms when a foaling mare begins labor. They are implemented in several different ways. Many of these systems have serious shortcomings that prevent them from being used on a large scale. Some monitors use a belly-band system similar to the one shown in Fig. 1.1 which can actually increase the chance of injury to the animal, as well as damage to the equipment, as shown in Fig. 1.2. Other monitoring systems use simple radio frequency (RF) transmitters to transmit an alarm when casting or colic related events occur. They are not adapted for long-term monitoring and usually cannot transmit data from multiple horses at one time. Each of the systems found in research during the Equine Distress Monitor project exhibited principles, components, and ideas that could be implemented into a large-scale, long-term monitoring system; however, none of them seemed to fill the current need.

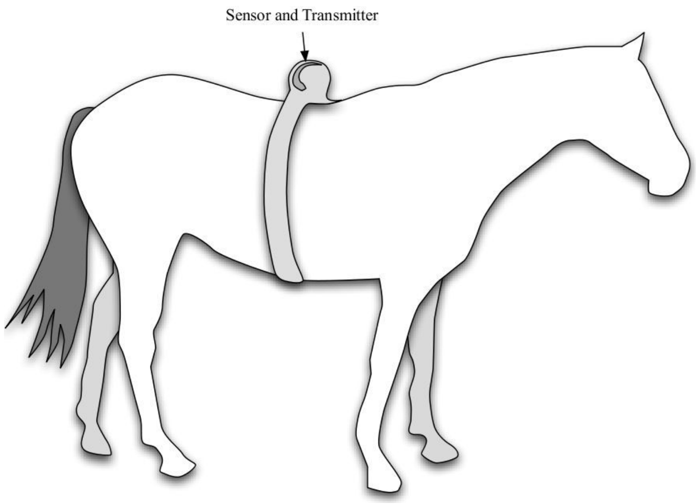


Fig. 1.1: Example of a “belly-band” sensor device.

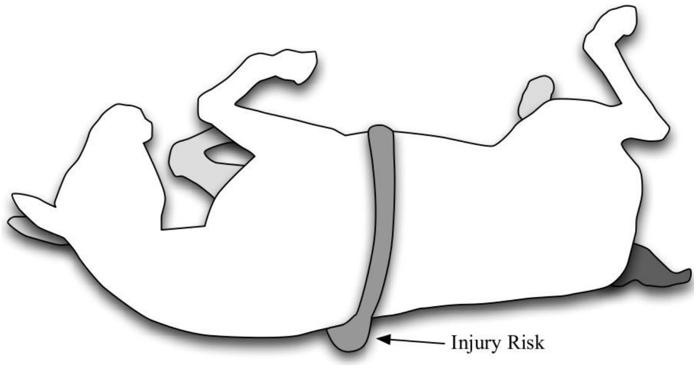


Fig. 1.2: Injury risk posed by “belly-band” sensor.

1.3.1 Equine Inverted Posture Alarm

The Equine Inverted Posture Alarm (EIPA) system was designed for detecting a horse's inverted posture which indicated foaling in pregnant mares [3]. It is mentioned in the patent paper that this system could be used to detect colic behavior as well. The system utilized a tilt switch activated sensor that transmitted an alarm from the remote location of the horse to the monitoring personnel. The EIPA system had very little resolution to determine whether the horse was actually healthy or not. The patent paper mentioned successful implementation in detecting a foaling mare; however, long-term monitoring for colic detection may not be as successful because there was very little event filtering. Another limitation to the EIPA system was that the range from sensor to monitor was comparatively limited.

1.3.2 Center of Mass Movement and Mechanical Energy Fluctuation During Gallop Locomotion in the Thoroughbred Racehorse

The kinetic and potential energy dynamics of horse locomotion is complicated to measure and analyze. "Centre of Mass Movement and Mechanical Energy Fluctuation During Gallop Locomotion in the Thoroughbred Racehorse" explained a system that utilized a series of accelerometers, an inertial sensor, gyroscope, a Global Positioning System (GPS) unit, and a Digital Enhanced Cordless Telecommunications (DECT) telemetry unit to gather motion data from seven thoroughbred horses [4]. The bulk of the article focused on data interpretation and what it meant for energy efficiency in the galloping movements. Elements of this system could be adapted for colic detection, but the main focus of the article was toward performance analysis.

1.3.3 Monitoring System for Animal Husbandry

Monitoring System for Animal Husbandry patent paper described an invention that monitors the conditions of a horse or group of horses, and relayed data gathered through a network to a base station where decisions were made about the well-being of the animal or animals [5]. The paper was written in very general terms regarding the invention. This paper primarily focused on sensors attached to the hooves and head of the monitored horses for detecting colic, yet it mentioned the use of the system to monitor different behaviors of interest in a variety of animals. The invention did not limit itself to any specific technologies in wireless communication or in data sensing. It was for this reason that a more specific and focused invention was needed.

1.3.4 Animal Instrumentation

The patented Animal Instrumentation (AI) system described in this paper was intended for use in veterinary diagnostics [6]. One of the best ways for a doctor to investigate and diagnose a problem is through communicating with the patient. In veterinary science, communication is very limited. As a result, it becomes necessary for the medical caregiver to find safe and effective ways of determining problems. The system in this paper described a series of radio communication electronic devices that used an array of sensors placed on different parts of a horses body to monitor movement, heart rate, perspiration, and other diagnostic information and transmitted the data to a work station to aid the veterinarian in the investigation of the horse's health. The patent paper did not describe itself in terms of a specific technology, but it did mention that it would be possible to use Bluetooth wireless technology, low-power ad-hoe, or ZigBee mesh networks. The AI system was used to further diagnose the problem the horse was known to have, which improved diagnostic understanding of the horse's health. The author also presented a method of correlating movement data with video of the same behavior to improve the way the system could process and interpret the data. This system was not intended for long-term monitoring; however, it could be adapted to be used for several days to monitor multiple horses.

1.3.5 Foaling Alarm

The Foaling Alarm (FA) was a "belly-strap" type of device, similar to the one shown in Fig. 1.1 and Fig. 1.2 that used tilt switches to detect the horizontal positioning of a foaling mare to initialize radio communication to a remote station where an alarm was sounded for attending personnel [7]. This paper was written in the 1970's, so the suggested wireless technology, Ace Wee1 Single Channel Tone Transmission, was quite outdated. The advantage this system had was that it did not consume power while the horse was in a vertical position, making it adaptable for long-term monitoring. The system's bulky size limited this application, especially in colic monitoring, due to the fact that if a horse rolled while wearing the device, it could cause injury. A possible scenario was shown in Fig. 1.2.

1.3.6 Livestock Breeding and Management System

The Livestock Breeding and Management (LBM) System was a mode of monitoring the timing, efficiency, and frequency of livestock breeding [8]. A system like this gave more resolution to livestock breeding patterns so intelligent decisions could be made by farmers. The system consisted of wireless transmitting devices attached to male breeding stock that read a radio frequency identification (RFID) tag attached to female breeding stock and transmitted breeding data through stationary and mobile data readers to a work station monitored by the breeding farmer. The specific wireless technology that was suggested for use in this patent was Bluetooth wireless technology. This suggested the range of the wireless communication would not exceed a hundred meters since class 1 Bluetooth was only guaranteed to that distance. This distance could be extended through routing devices placed on fence posts, or near watering troughs. The wireless transmitting devices attached to the males were activated when the animal assumed a breeding position on a female. This system could be modified to operate as a colic-detecting system; however, further development would be needed.

Chapter 2
The Equine Distress Monitor System

2.1 The Solution

The Equine Distress Monitor (EDM) was an electronic monitoring system that utilized Atmel Zigbit wireless modules communicating over a mesh network based on the Bitcloud stack instruction set for large-scale, long-term, twenty-four hour monitoring of horses. The EDM system overcame many of the drawbacks mentioned in the above article descriptions. It utilized a non-evasive sensor device that was placed in a non-obstructive location on the horse. The sensor was also very low power which adapted it for monitoring over months, or even years at a time without changing batteries. The integrated tilt switch and accelerometer in the sensor device enabled it to detect some of the symptoms of colic and casting while filtering out irrelevant movements. The sensor transmitted key distress movement detection messages through the Zigbit wireless mesh network to a base station computer where it was further analyzed and alarm messages could be sent out to relevant personnel.

2.1.1 Horse Behavior Indicating Colic or Casting

In order for long-term monitoring to occur, sensor devices must be small in size, rugged, and easy to maintain. A system that was capable of detecting all symptoms of colic would not fit in these constraints, therefore it was necessary for a system to choose certain colic indicating behaviors to focus detection. For the Equine Distress Monitor project, we chose to detect the following key behaviors that indicated colic or casting events.

- Lying down
- Large number of rolling or lying down events
- High level of activity while lying down

2.1.2 Healthy Horse Behavior

Healthy horses are also known to lay down and/or roll, but these events are almost always accompanied by certain behaviors. These key indicators of a healthy horse help to filter out unwanted alarms. They are: a small number of rolling events in short period of time, and shaking after a roll.

2.2 Alpha Prototype of EDM

Utah State University resources had already been invested into developing alpha prototypes of the EDM system prior to the graduate research project. Early prototypes that were developed for proof of concept testing consisted of MeshBean development PCBs produced by Meshnetics and accelerometer circuitry that communicated horse movement to the zigbit module. The MeshBean sensors were mounted inside enclosures attached to the test horse's chest. Figure 2.1 shows a horse with a sensor attached.

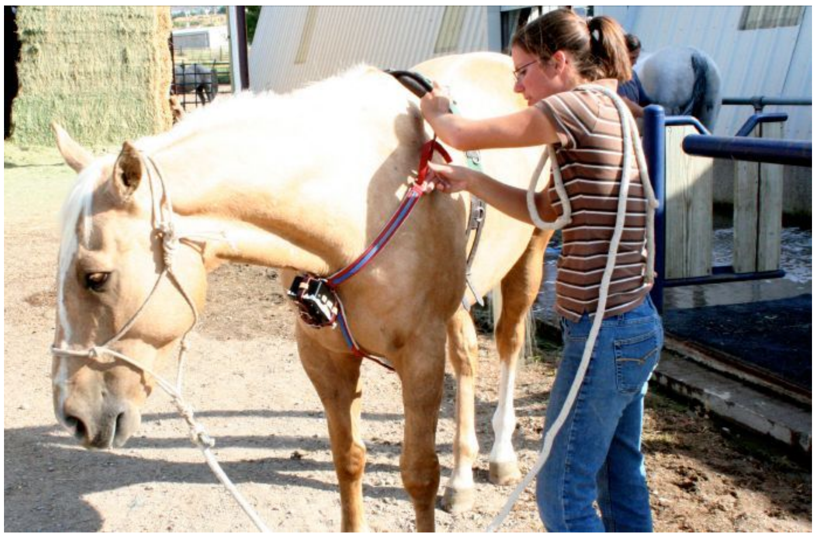


Fig. 2.1: Alpha prototype sensor.

2.2.1 MeshBean Development PCBs

Meshnetics was the original creator of the ZigBee and Zigbit radio protocol and radio modules. Since the creation of these RF modules, Atmel acquired the intellectual properties regarding them. Before this acquisition took place, Meshnetics produced a series of development printed circuit boards (PCBs) named the MeshBean series. One example of these PCBs is pictured in Fig. 2.2. These development platforms included the Zigbit A2 radio module with a series of switches and peripherals. The module was able to communicate through its universal asynchronous receiver/transmitter (UART) interface through a translator IC to a PC via universal serial bus (USB) protocol. It was able to be programmed and debugged through a joint test action group (JTAG) interface. The MeshBean incorporated its own power-delivery system via USB power or through two AA batteries. Other peripherals included dual in-line package (DIP) switches, push button switches, and an array of light-emitting diodes (LEDs). Meshnetics also distributed a series of software development packages to be sold with these PCBs. Many of the programs provided were used as examples in the development of the EDM system. These examples and the documentation for them can be found on the Meshnetics website [9].

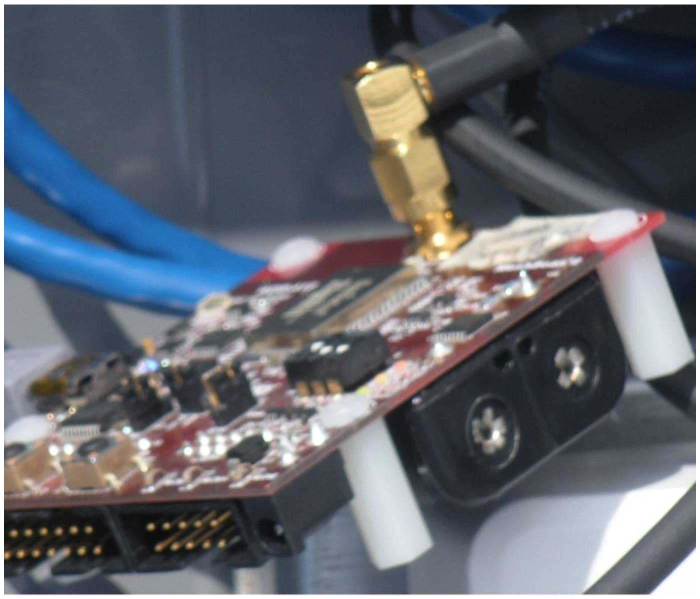


Fig. 2.2: MeshBean PCB.

2.2.2 Testing

Testing with the early prototypes revealed that it was possible to detect colic behavior in horses and transmit the sensed data through a mesh network to a PC base station for analysis. While there were not any colic events detected by the system, colic-like behavior had been simulated and detected for testing purposes. Even though the alpha versions of the sensor units were suitable for demonstrating the main concepts of the system, they were not intended as a feasible final design.

2.3 The EDM Project

The mesh network used for the EDM consisted of three types of devices: coordinator, routers, and end-devices. During the Alpha phase of the project, the EDM system utilized the MeshBean PCBs in each of these functions according to its programming. A basic communication model for the EDM system can be seen in Fig. 2.3.

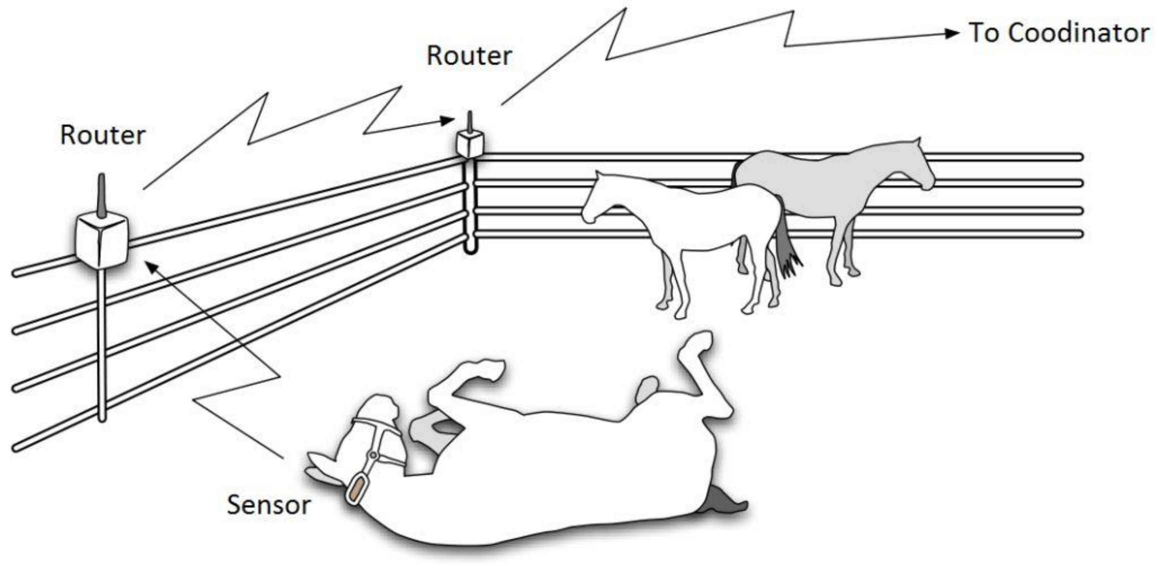


Fig. 2.3: EDM mesh network communication model.

The placement of the routers and coordinator was such that the radio communication between each of the three devices was reliable enough to ensure a stable communication between the sensor boards and the coordinator.

2.3.1 Project Objectives and Contributions

The main objective of the EDM project was to utilize the knowledge gained through the Alpha prototype design, and create a working Beta system that was more compact and had better performance.

The master of science degree canditate, Luke Peacock, began contributing to the EDM project in January of 2010. Dr. Chris Winstead hired him to assist in the hardware and software design. Luke worked closely with Mitchel Humpherys, another student, in developing a series of printed circuit boards with accompanying firmware to perform the various functions of the EDM system. By May 2010, Luke and Mitchel had developed the first sensor prototype and were able to program it to perform basic sensor tasks. However, there were many flaws that needed to be addressed in an immediate second revision. Throughout the summer of 2010, Luke focused mainly on hardware development, while Mitchel focused on progressing the firmware. It was found early on that in addition to the sensor PCB, they would need to develop router and coordinator PCBs. Much of Luke's time was spent researching different integrated circuits (IC's) that would perform the needed tasks. Luke also assisted Mitchel in researching the different firmware functions and hardware settings. By August 2010, the system was ready for initial system testing. That September, they deployed 5 EDM routers, 18 EDM sensors, along with a Meshbean coordinator and the Amos 3001 Ultra Compact Embedded System base station at the Utah State University Equine Education Center, located in Logan, Utah.

Testing of the deployed system revealed a few issues that are described later in this report. Throughout the Fall 2010 Semester, Dr. Winstead revised the EDM sensor state machine firmware to overcome these problems. Meanwhile, Luke tied up the loose ends with the hardware designs. From December 2010 to the present, he researched, documented, and assembled information to be used in writing this thesis project report.

2.3.2 Sensor Devices

The most important design constraints for the end-devices, or sensor boards, was power consumption and horse behavior monitoring. To meet these constraints, the sensor device included a tilt switch that detected when a horse laid down (possibly due to colic), rolls, shakes, or other types of movements. It was integrated into the sensor PCB to activate the Zigbit module which monitored accelerometer data from the on-board accelerometer chip to decide relevant movements of the horse. The sensor board and battery power supply were encased in a small leather pouch mounted on a standard horse halter or collar as shown in Fig. 2.4. This made it aesthetically pleasing, and reduced the risk of damage to the device, as well as to the horse. To further decrease chances of injury to the horse, a ”break-away” halter or collar may be used.

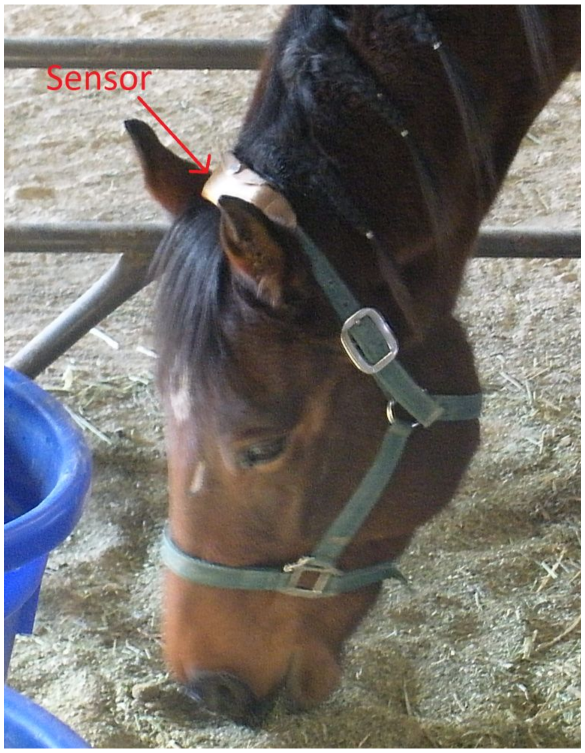


Fig. 2.4: Mounted sensor device.

2.3.3 Router Devices

The only real function of the router boards was to extend the network range of the coordinator from the location of the base station PC to the various pen locations. The only additional circuitry, apart from the Zigbit module, needed on this board was the power management and debugging circuitry. For long-term monitoring, it was necessary to integrate a solar-charging system for the router battery pack. This consisted of a DC/DC converter with a series-connected diode connected between a 12 volt, 5 watt, solar panel and the battery with bypass capacitors across each power rail. The router and battery assembly were placed in a weather-proof enclosure and mounted in the desired location on top of a fence post. To increase signal strength and connection reliability an external antenna was connected to a three foot coax cable and mounted on top of a PVC rod above the router device. A router station is pictured in Fig. 2.5.

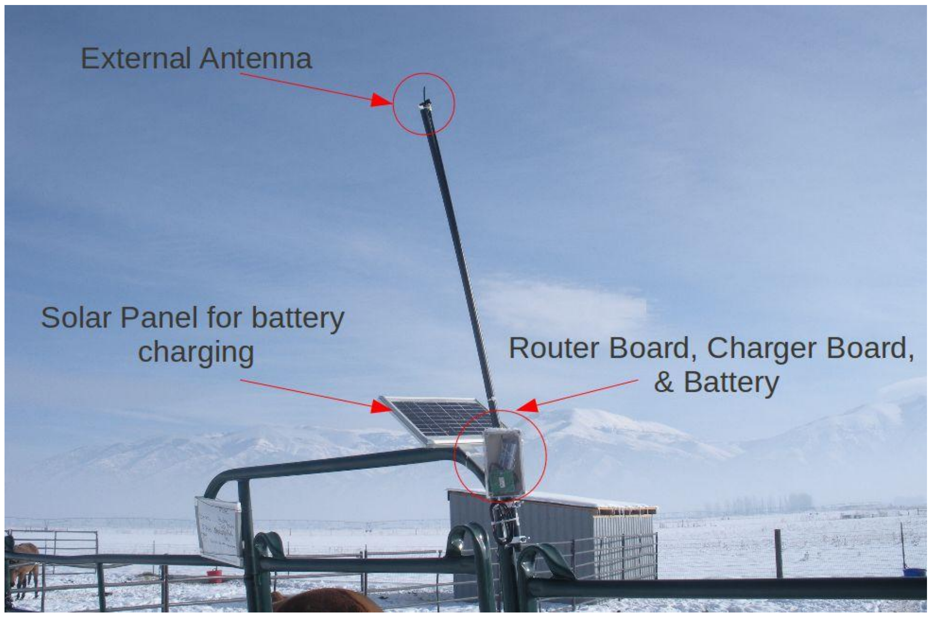

External Antenna
Solar Panel for battery charging
Router Board, Charger Board, & Battery

Fig. 2.5: Mounted router device.

2.3.4 Coordinator Devices

The coordinator devices were very similar to the router devices. There was no need for any extra peripherals since the main function was to relay the signal received from the sensor and router network, and relay them to the base station. The difference between the coordinator boards and the router boards was the USB interface circuitry. The coordinator communicated with the base station via a UART interface from the Atmel Zigbit module through a UART to USB transceiver IC to the PC base station. The coordinator was placed in an exterior electrical box with the base station computer.

2.3.5 Base Station

The base station computer was responsible for receiving all the alarm messages sent from the sensor devices through the mesh network and making final decisions on alarms sent to horse-care personnel. The base station was placed inside the exterior electrical box described in the coordinator section above, as shown in Fig. 2.6. Messages sent from the coordinator through the USB connection to the PC computer were all stored in a database file which was utilized to upload the sensor information to a web interface. At this point, the EDM system only uploads information to the web server which can be monitored both locally and remotely. Further development could be done to send out various types of alarm messages such as SMS or email to the responsible personnel.

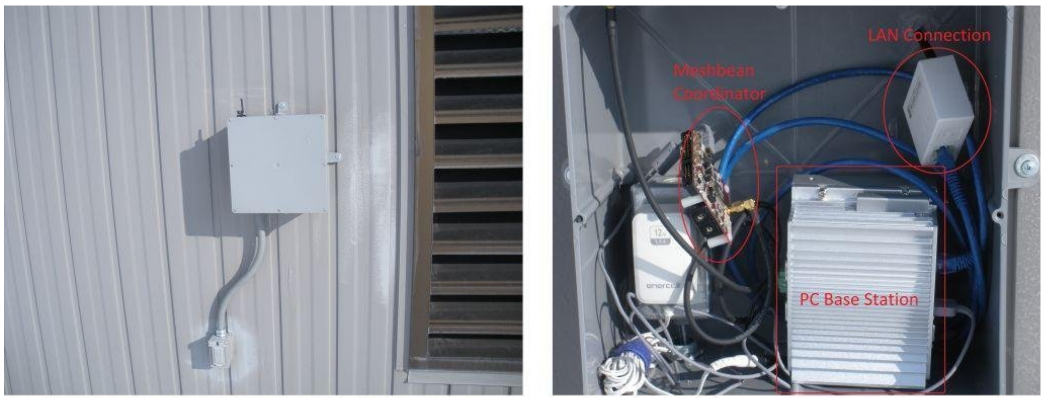


Fig. 2.6: Mounted base station and MeshBean coordinator.

Chapter 3
Hardware Design

The first prototype design of the EDM system was not feasible as the final product because the MeshBean development PCBs were costly, bulky, and not power efficient enough for long-term monitoring. Also, the placement of the original prototypes on the horse was undesirable from an aesthetic and safety standpoint. Furthermore, the MeshBean PCBs were no longer being manufactured. Due to these shortcomings, it was necessary that furthering the development of the EDM system included the design of a series of PCBs that incorporated similar functionality as the MeshBeans, but with more customization for optimizing specific functionality. It was desirable for this project to have three PCB designs that were optimized for each of these functions. Throughout the EDM project, much of the testing of software applications was done on the MeshBean platform PCBs.

Each individual device needed to have a unique identification number to distinguish between devices on the mesh network. To accomplish this goal, a Maxim silicon serial number IC [10], which communicated through the Dallas Semiconductor's 1-Wire protocol with the Zigbit module, was included in the design. Each device also included a 10-pin connector [11] to communicate with the JTAG interface of the Zigbit module for programming and debugging purposes.

Each of the EDM printed circuit boards was manufactured by PCBs Unlimited. Populating the components onto each of the EDM devices was done by placing solder paste onto each PCB using solder paste stencils ordered from Stencils Unlimited, shown in Fig. 3.1. Placing the components onto the wet solder paste was done using tweezers under a microscope and using a Gold-Place manual pick-and-place system [12] shown in Fig. 3.2. Once components were placed on the PCB, re-flowing the solder was done in the re-flow oven shown in Fig. 3.3. The re-flow oven consisted of a generic toaster oven with additional insulating material installed, and a Temp-Tell re-flow controller with temperature sensor feedback. After re-flowing each of the boards, touch up soldering was done using a standard solder-iron station and a hot air pencil.

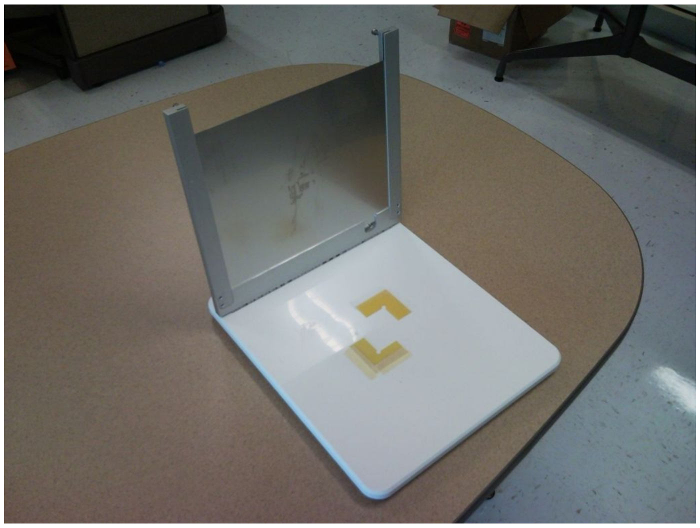


Fig. 3.1: Solder paste stencil mounted in stencil holder from Stencils Unlimited.

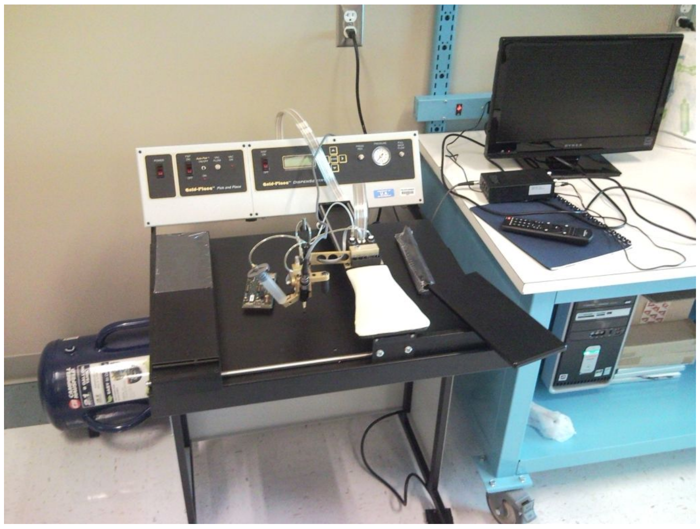


Fig. 3.2: Gold-Place manual pick-and-place system.

Fig. 3.3: Re-flow oven.

3.1 Programming Boards

In order to program the devices, it was necessary to design and build a PCB that would connect each device with the Atmel AVR JTAGICE mkII programmer as shown in Fig.3.4. These boards included just two connectors: a standard female JTAG connector [13] that mated with the JTAGICE programmer and a 10-pin male connector which connected to the 10-pin header found on each of the EDM modules. A picture of the programming board is shown in Fig. 3.5.

3.2 Sensor Boards

As mentioned earlier, the main constraints of the sensor devices were: small size, low-power consumption, and effective horse monitoring capabilities. The main component of this device was the Atmel Zigbit radio module [14]. One of the main reasons this particular Zigbit module was used was because of the on-board antenna included on each module. Also, the module had an internal voltage regulator enabling it to be powered directly from two 1.5 V AAA batteries. Activation of the module when a horse laid down was done using an interrupt signal which was triggered from the SignalQuest tilt-switch [15]. Once the module was activated, it would gather data about the horse's movements from the ST accelerometer [16] via I^2C communication protocol. The reason this accelerometer

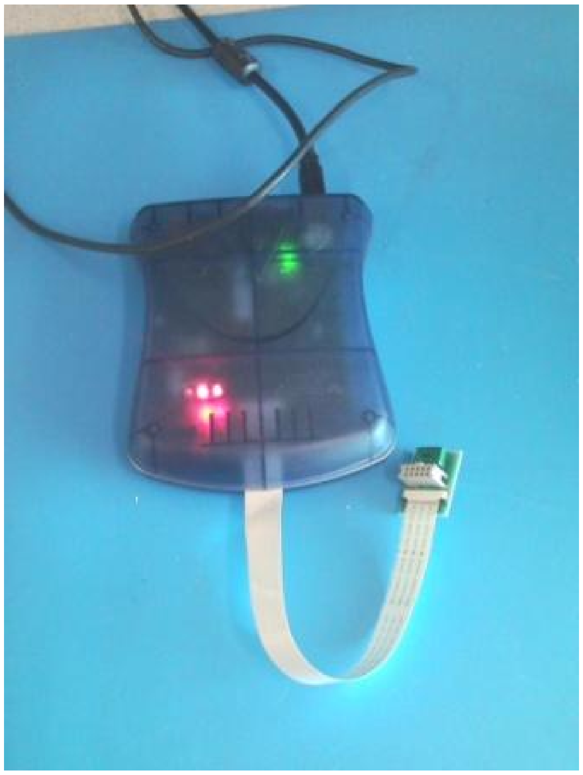


Fig. 3.4: Atmel AVR JTAGICE mkII programmer.

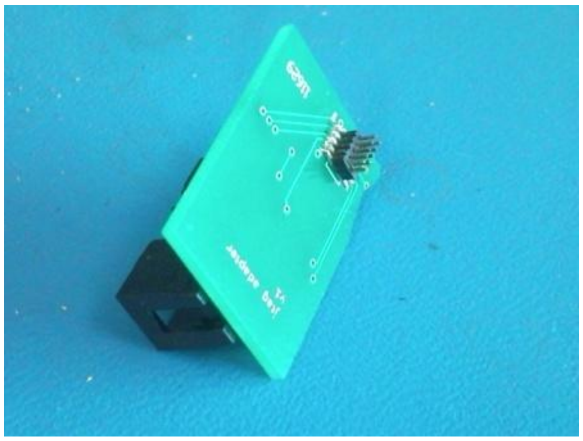


Fig. 3.5: EDM programming board.

was chosen was because of its low-power consumption and a variable range of acceleration detection. This accelerometer had a range that sufficiently detected the horse movements associated with colic, while still retaining enough resolution for data analysis. The range of acceleration detection fit within the requirements with sufficient resolution. Prior to this project development of a 1.0 version of the sensor board was designed with a different accelerometer. Schematic and layout drawings for this version are found Fig. A.1 and Fig. A.2 of Appendix A but will not be discussed in this report. Additional circuitry connected to the accelerometer was done according to the accelerometer manufacturer's specifications. Other components included on the sensor board design included various pull-up and pull-down resistors, filtering components, and bypass capacitors. The schematic drawings and layout images of the sensor board can be seen in Appendix A.

The physical dimensions of the sensor board were designed such that the board could fit inside an enclosure along with two AAA batteries. The plastic enclosure that was chosen for this design was slightly bigger than a traditional business card as shown in Fig. 3.6. This plastic enclosure was not sealed from moisture or dust, so in furthering the development of this system, a different enclosure would need to be investigated. The electrical component layout of the sensor was engineered so that the antenna of the Zigbit module would be placed at the highest point possible on the horses head when mounted inside the halter. The orientation of the tilt switch was designed to be perpendicular to the standing position of the horse. All other components on the sensor were not sensitive to specific placement, so they were placed according to geometrical convenience.

3.2.1 Sensor Version 1.1

The initial design of this board included a P-Channel MOSFET [17] to connect main power to the board. This particular MOSFET had an internal resistor between the gate and source terminals of the FET. A horse tilt event would close the tilt-switch and pull the gate of the MOSFET low, connecting the main power to the PCB, activating the Zigbit module. Once powered, a GPIO from the Zigbit module would hold the gate of the MOSFET low while processing and signal transmission took place. When transmission and

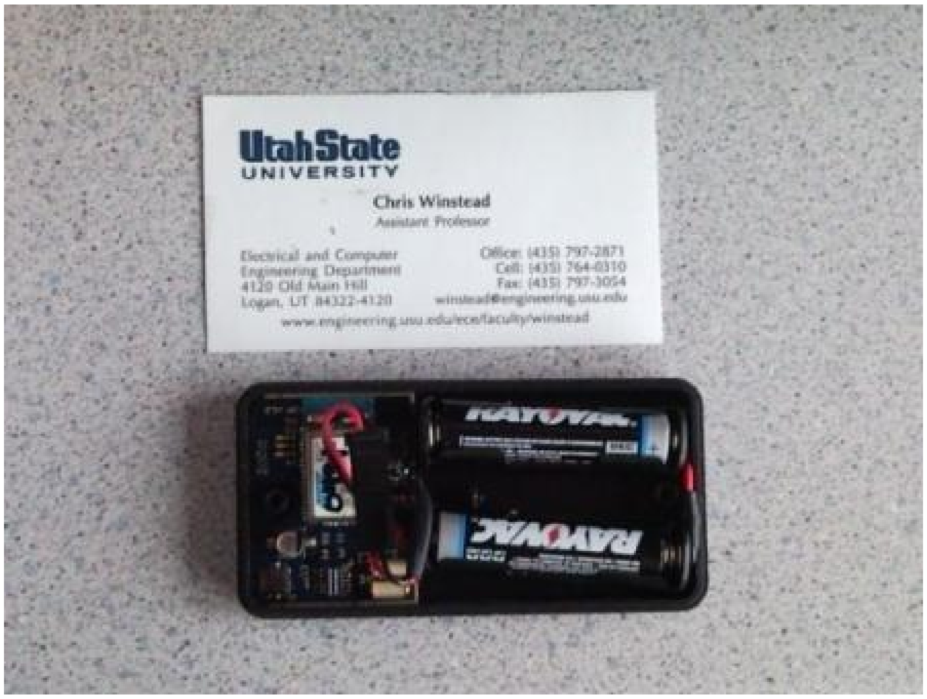


Fig. 3.6: Sensor PCB mounted inside enclosure.

processing finished, the Zigbit module would then pull the gate of the MOSFET high which disconnected power from the board. For more detailed review of the sensor board V1.1 design, please refer to Fig. A.3 and Fig. A.4 found in Appendix A. A photo of a prototype of V1.1 sensor board can be seen in Fig. 3.7. One will note the additional wires connecting to various parts of the board. This photo was taken after many tests and debugging had been performed on this particular prototype.

3.2.2 Sensor Version 1.2

It was found through testing of the V1.1 prototype that the mode of connecting and disconnecting power from the board through the MOSFET did not function as intended. The main power of the board would end up in an oscillating state. For V1.2, it was decided to utilize the low-power sleep-mode of the Zigbit module, remove the MOSFET from the design, leaving the power connected at all times. Reactivating the Zigbit module from the sleep-mode state was done by connecting the tilt-switch circuit to an interrupt pin on the Zigbit module, which was normally held high through a 1MΩ resistor. All other circuitry

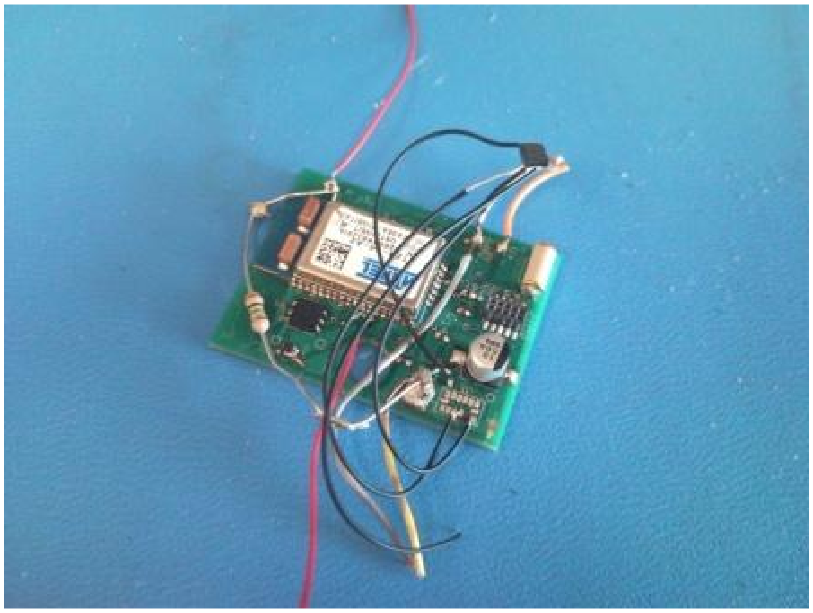


Fig. 3.7: Sensor board V1.1 with modifications.

was left the same as in V1.1. The PCB design drawings for V1.2 are shown in Fig. A.5 and Fig. A.6 found in the Appendix A. A photo of a prototype of V1.2 sensor board can be seen in Fig. 3.8.

3.2.3 Sensor Version 1.3

While testing and debugging V1.2 sensor devices, it was found that the tilt-switch chosen in the design was unidirectional, meaning that the sensor would only detect horse-tilt events on one side. Therefore, it was necessary to add an additional tilt-sensor to the design so tilt events on either side could be detected. Investigation of the Bitcloud instruction set revealed that battery gaging functionality was possible through the ADC inputs of the Zigbit module. It was decided that additional circuitry for this functionality should be added to the design. This circuitry included a high-precision ($+/-0.5\%$) resistor divider circuit connected between the battery voltage and ground through a digital switch [18]. Activation of the switch was done through a GPIO on the Zigbit module. It was also desirable for the sensor board to include a series of LEDs connected to GPIOs on the Zigbit module for debugging purposes. To add additional de-bounce on the tilt-switch circuitry, a capacitor

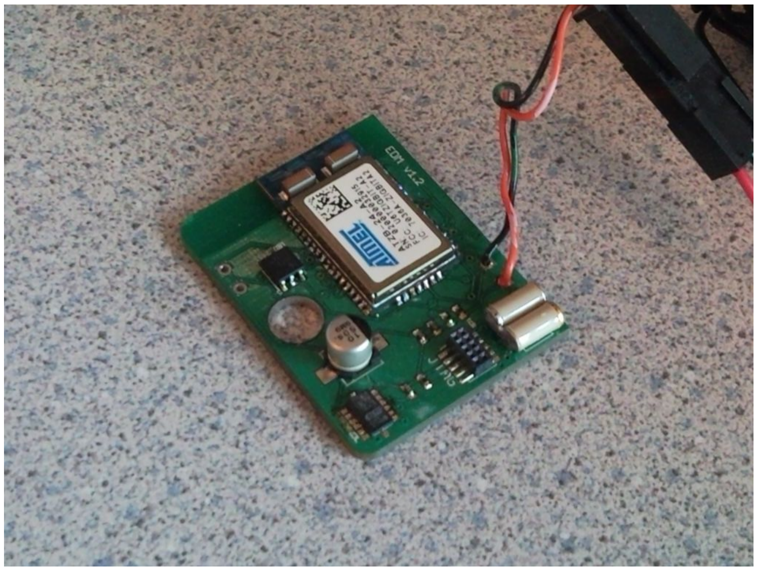


Fig. 3.8: Sensor board V1.2 with a second tilt switch added for bi-directional tilt sensing.

was connected between the interrupt signal and ground. All other circuitry on the sensor board remained unchanged from previous versions. Figure 3.9 shows a photo of a panel of V1.3 prototype PCBs. The PCB design drawings for V1.3 are shown in Fig. A.7 and Fig. A.8 found in Appendix A.

3.3 Router Boards

The main focus in designing the router PCB was to extend the range of the mesh network while maintaining radio signal strength and reliability between modules. To accomplish this, it was decided that the version of the Zigbit module that included an RF amplifier would be used [19]. This module increased reliability in the mesh network in transferring important data from the sensors to the base station computer. This Zigbit module did not include an internal voltage regulator; so in order to power the device a low-dropout [20] regulator was used to give the needed constant voltage of 3.3V. Decoupling capacitors were placed across the input and output of this LDO regulator to increase voltage reliability. The input voltage of the LDO regulator was provided from a 7.2V NiMH battery pack [21]. The PCB design drawings for V1.0 of the Router Board can be found in Fig. B.1 and Fig. B.2 found in Appendix B.

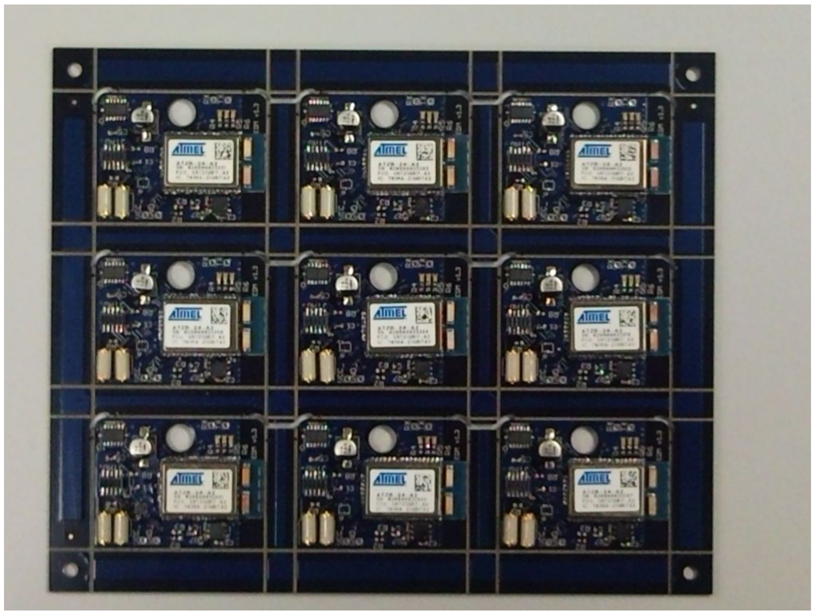


Fig. 3.9: Sensor board V1.3 (panelized).

To monitor the power usage of the battery pack, the same switch and resistor divider network used for battery voltage measurement on the later versions of the sensor board were also included in the first router board design. It was found during testing of V1.0 of the router board that the switch was not necessary, and that when measuring battery voltages greater than 3.3V it actually increased power consumption. In V1.1, the only change made was to remove the switch from the design, leaving the resister divider network always connected. Also included in the router board design was a series of LED circuits connected to GPIO pins on the Zigbit module. These LEDs communicated the network communication status of the router board. A photo of a V1.0 router board can be seen in Fig. 3.10. The PCB design drawings for V1.1 of the Router Board can be found in Fig. B.3 and Fig. B.4 found in Appendix B.

Because this board would be placed outdoors in various locations around the horse's environment, it was necessary to protect the router board from the elements. To accomplish this, an enclosure that was rated according to IP-65 standards for moisture and dust, was chosen to protect the devices [22]. The lid of the enclosure was made of a clear plastic material so that the LED communication of network status to the user was not disrupted.

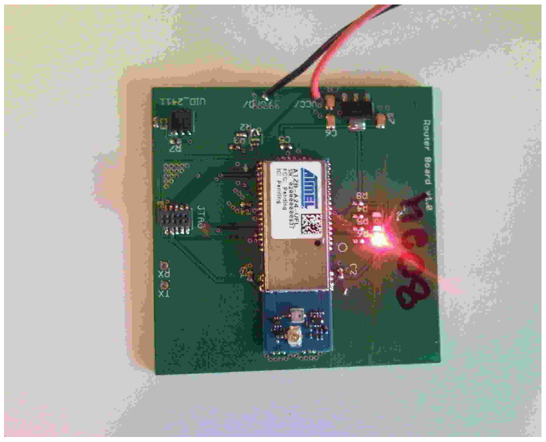


Fig. 3.10: Router board V1.0.

This enclosure housed the router board itself, along with the 7.2V battery pack.

The amplified Zigbit module required connection to an external antenna through an MMCX connector. This miniature coaxial connection transitioned to a standard polarity SMA female connector. This SMA connection was the means of transferring the RF signal to the outside of the enclosure through a small, sealed hole drilled in the outer wall of the enclosure. To increase network reliability, a 3-foot RG-58 coaxial cable connected the SMA connector to an elevated antenna. Elevation of this antenna was accomplished according to external resources available on each specific site. An example of a mounted router apparatus is pictured in Fig. 2.5.

3.4 Solar Charging Circuits

One will note that a solar panel is also depicted in Fig. 2.5. In deployment of the router devices, it became apparent that a means of recharging the battery packs would be needed. To accomplish this goal, two different implementations of solar charging circuits were executed. The first implementation included a Valor voltage regulator designed for solar-charging applications [23]. A PCB was not designed for this circuit because of an immediate need for a solution; therefore, connections were executed on a proto-board material. The battery chosen to keep the device powered during low to no sunlight hours for the commonly recommended three to five days was a Tenergy 5000mAh RC plane NiMH battery [21]. A picture of the solar circuit is shown in Fig. 3.11. The same circuit was implemented with a Recom voltage regulator [24] and is pictured in Fig. 3.12. For this implementation decoupling capacitors connected across the input and output terminals of the regulator were included as well. In future developments of this project a solar-charging circuit could be implemented on the router board itself to reduce complexity and increase power efficiency of the circuit.

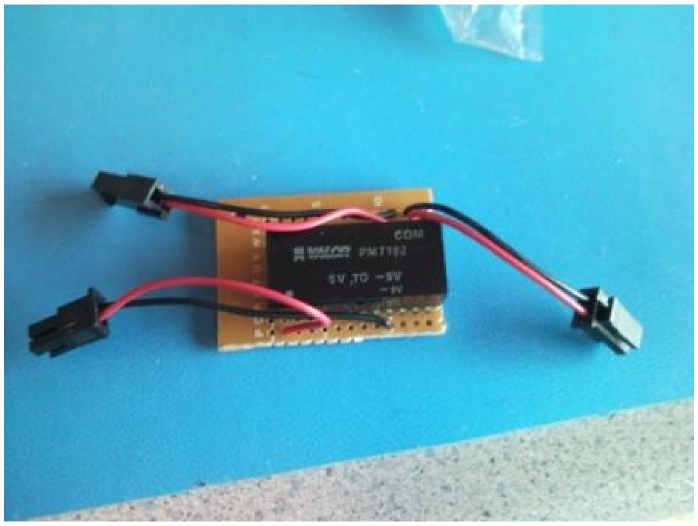


Fig. 3.11: Solar charging circuit implemented with a Valor voltage regulator.

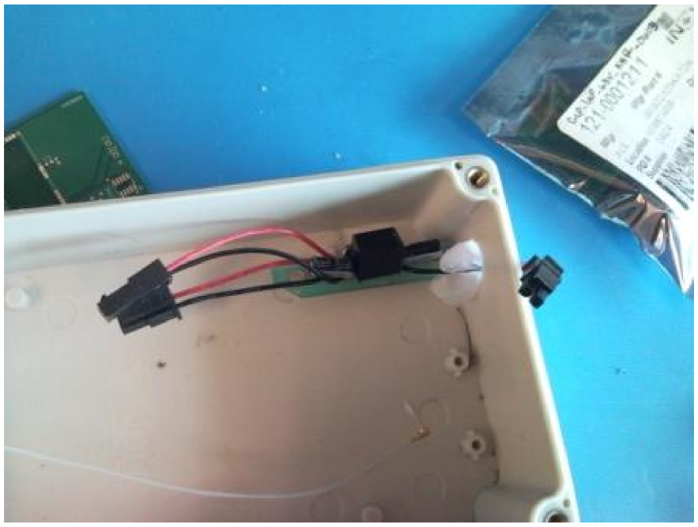


Fig. 3.12: Solar charging circuit implemented with a Recom voltage regulator.

3.5 Coordinator Boards

The coordinator board design originated from the router board, with additions for coordinator functionality. There was an added level of complexity from adding the USB communication capabilities to the coordinator design. First, a USB to UART bridge IC [25], with its manufacturers recommended circuitry, was added to the design so that communication to the base station computer could take place. To accomplish this, the design also included a USB mini-B connector [26] with EMI [27] and filtering [28] components. It was also desirable to power the PCB from the USB power connection from the base station computer. For this feature, it was necessary to connect the +5V USB input power to the input terminal of the LDO regulator. In V1.0 of the coordinator design, this was done through a regulator found on the USB to UART bridge IC. This proved to be problematic in application, so in V1.1, the USB power net was fed strait to the input terminal of the LDO regulator. To prevent current from feeding from a battery that could possibly be connected to the system on to the USB power net, a series diode was added. Alternatively, to prevent power from feeding from the USB power net to the battery, a series diode was added to the input ground terminal of the battery connector. For more details about the coordinator design, please refer to the schematic drawings and layout images of the coordinator in Appendix C in Fig. C.1, Fig. C.2, Fig. C.3, and Fig. C.4. Although it was not necessary for it to be protected from outside weather conditions, for convenience purposes, the same IP-65 enclosure used to protect and contain the router board and the MMCX and SMA RF connections was also used for the coordinator design. With the implementation of the EDM system at the Utah State Equine Education Center (USUEEC) this enclosure was omitted and the board was placed inside an exterior electrical box with the base station computer. The MeshBean coordinator shown in Fig. 2.6 was eventually replaced with the EDM coordinator shown in Fig. 3.13.

3.6 Base Station Hardware

The hardware chosen to function as the base station computer for the EDM system was the AMOS-3001 Ultra Compact Fanless Embedded System [29] manufactured by VIA

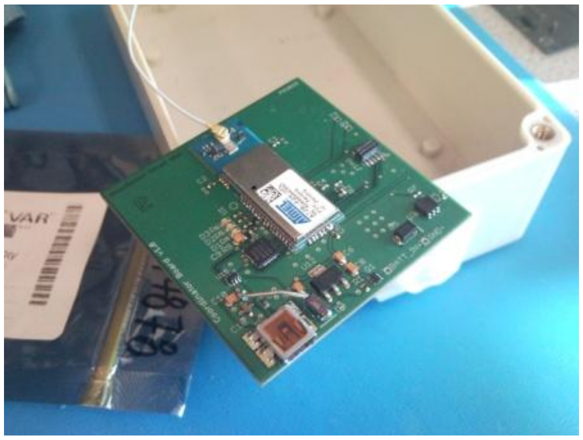


Fig. 3.13: Coordinator board V1.0 with modification to power device from USB.

Technologies. The compact size, with fewer moving parts, made this system especially desirable for the EDM project. The system was able to connect to a LAN connection on the outside of the Utah State University Equine Education Center office building for remote monitoring and database file upload, and to the USB connector of the coordinator PCB for mesh network communication. The base station computer was placed in an exterior electrical box with the MeshBean coordinator, as shown in Fig. 2.6.

Chapter 4
Embedded Firmware Design

Development of the firmware programmed onto the various Zigbit devices for the EDM project was done using a software development kit (SDK) written in the C programming language by AVR developers commissioned by Meshnetics and Atmel. The SDK included a set of example programs that could be referenced and mimicked in developing customized applications. These examples were written for deployment on a range of development hardware platforms, including the MeshBean platforms. The EDM applications contain components taken from several of these examples. The SDK included development tools for three different application types, as per with the ZigBee standard. These applications included: coordinator, router, and end-device (sensor) applications. The applications for each of these three types of devices were customized for the EDM project. During application development, aid from AVR experts employed for application development support on the http://www.avrfreaks.net/ website discussion forum was enlisted and utilized.

4.1 BitCloud Stack

Meshnetics and Atmel produced a full-featured embedded software stack that provided a software development platform for wireless applications written for Zigbit modules and named it the BitCloud stack [9]. The BitCloud stack implements a unique, cooperative scheduler for short callbacks for system and user generated events. The BitCloud stack was the part of the application program that operated the radio message transmission and performed network maintenance for each of the devices. This maintenance included updating the network information about each individual node. This included the network address, unique device identification number, RSSI, LQI, device parent information, and battery voltage level. It also managed data transmission from sensor devices all the way to the coordinator device. The architecture of the BitCloud stack consists of three main parts. The first level is the user application layer which utilizes components from the core stack and the shared, low-level services. A simplified diagram illustrating the architecture of the BitCloud stack is shown in Fig. 4.1.

4.1.1 ZigBee Device Objects (ZDO), Application Support Sub-Layer (APS), and Network Layer (NWK)

The components of the core BitCloud stack that handled network and hardware management were the ZDO, APS, and NWK base classes. The APS sub-layer interfaced between the network layer and the application layer so that data requests and confirmations could take place between devices. Initialization of the APS layer and the NWK layer was done by ZDO objects. The ZDO base class provided means of acquiring information about device hardware and the ability to control it by interfacing between the application layer and the hardware abstraction layer with the board support package. The ability to put the device in an ultra-low-power state (sleep) and recover from this state (wake) was achieved by utilizing functions from the ZDO class. The ZDO class also offered functions that enabled an application to obtain network status, signal strength, and battery voltage level through interfaces between the APS and NWK layers.

4.1.2 Board Support Package (BSP) and Hardware Abstraction Layer (HAL)

Outside the core stack, there were sets of functions that provided more direct access to the device hardware. These functions resided in the BSP and HAL classes. For the EDM project it was necessary to synchronize data acquired from the accelerometer to a global system timer for data analysis. Initialization, control, and queries to these device timers were done using the HAL function set. HAL also provided APIs for interfacing the application to external peripheral communications such as I^2C, IRQ, 1-wire, and UART; all of which were utilized in the EDM project. The BSP provided drivers for managing standard peripherals placed on the MeshBean development platform which were also integrated into the EDM devices.

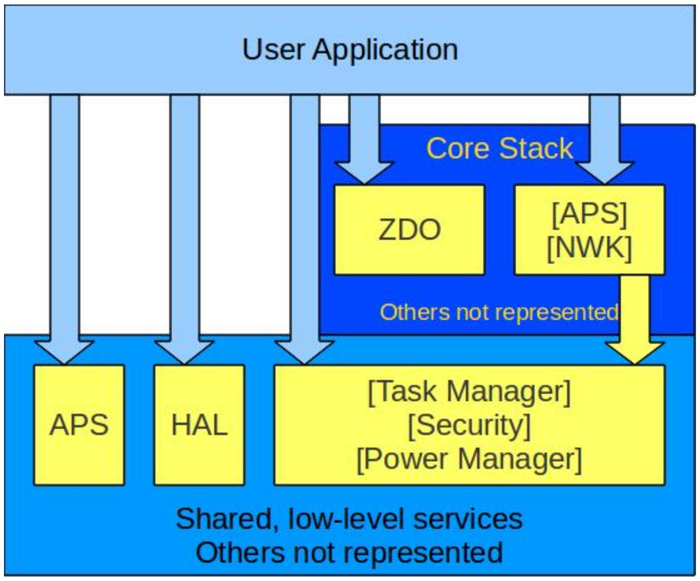


Fig. 4.1: Simplified architecture diagram of the BitCloud stack.

4.1.3 Task Manager

The main function of each of the device applications was executed through the application task handler. Scheduling and prioritizing tasks for the MCU from the user application was done by the task manager. Tasks were scheduled following a queue-based algorithm which was optimized for multi-layer stacks and time-critical network protocols [30]. Tasks sent to the task handler were executed asynchronously.

4.2 Shared Application Components

The BSP API set was developed for the MeshBean hardware. During the hardware development of the EDM project, MeshBean design documentation was used as a guideline to minimize extra software development. Though most of the EDM hardware mimicked the MeshBean hardware, there was still a few components that had to be added to accomplish the specific goals of the EDM project. In order to establish communication between EDM specific hardware to the programmed application, a set of APIs was developed.

4.2.1 Source Code: edm_util.c

The edm_util.c file with its accompanying header file contained useful functions for specific operations performed on the EDM devices. It acted as a hardware abstraction layer to interface with the EDM device specific hardware. It also contained functions and macros that were used for debugging application errors. These functions included:

- Accelerometer Functions

  This set of APIs was produced to interface the Zigbit module through the I^2C bus to the accelerometer on the EDM sensor. The functions for this interface communication included: accelerometer initialization, accelerometer de-initialization, sending data requests to the accelerometer, receiving data from the accelerometer, reading from accelerometer registers, and writing to accelerometer registers.

- Battery Functions

  To read battery voltage levels on the EDM devices, another level of abstraction was added to interface EDM applications with the BSP functions of the BitCloud stack which managed the Zigbit ADC hardware. This subset of API functions provided means of initializing, de-initializing, and reading from the ADC that was connected to battery measurement hardware.

- Logging Functions

  During the application development for the EDM project, this subset of APIs was developed to aid in debugging application programs. It provided a means of communicating through the Zigbit universal asynchronous receiver/transmitter (USART) to the application development platform PC. Functions included in the API for this functionality included an initialization function, and a logging function. To interpret communication logged on the development PC, a Python Script was written to translate logged events to the developer.

- Other Useful Macros

  The edm_util.c file also included a set of macros used in converting timer data between different time units, i.e. milliseconds, seconds, and minutes. It also included other useful debugging functions such as a way to communicate binary numbers on the LED array integrated on the EDM devices. The last function included in this set of APIs was an extension of the LED communication function that placed the application in an infinite loop, displaying the same binary number on the LED arrays.

4.2.2 Configuration and Makefile

In order to compile applications specific to each of the different EDM device platforms, as well as the MeshBean hardware platform, a Configuration file was created with specific parameters that configured the application to a specific hardware platform and functionality. Parameters in this file included: network parameters, power management parameters, debugging parameters, device platform selection, version selection, battery measurement implementation parameters, communication profile configuration, and sensor state machine profile parameters. The Makefile extracted the parameters set in the Configuration file and initialized compilation of the object files from source code, accordingly.

4.2.3 Source Code: EDM.c

The EDM.c file with its accompanying header file contained global constants and the main application task handler function for each of the different device applications. One data type that was defined in the EDM.h file was the data packet that would be passed between devices. The message format is depicted in Fig. 4.2. Contained within the application task handler was a general state machine that initialized and managed network status for each of the devices as they implemented specific state machine logic and functionality according to configuration constraints. This task handler also implemented the device-specific task handlers. Figure 4.3 shows a state diagram describing different states and how they interact with each other.

4.3 Sensor Application Components

The sensor application task handler followed a complicated state machine logic to detect horse behavior and send data through the mesh network. This logic evolved over time to optimize performance, based on data obtained during testing. This state machine logic is

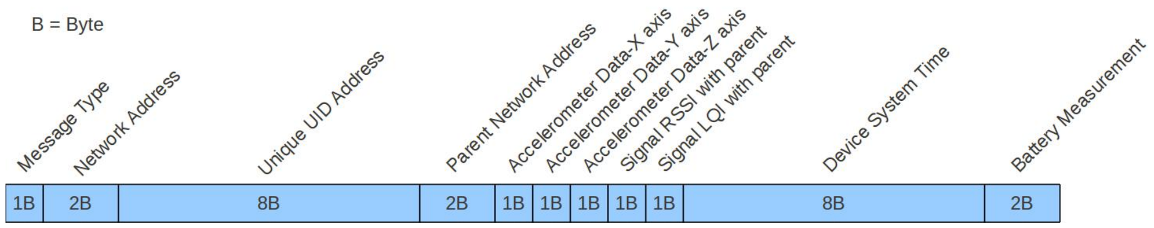


Fig. 4.2: Original EDM data packet format.

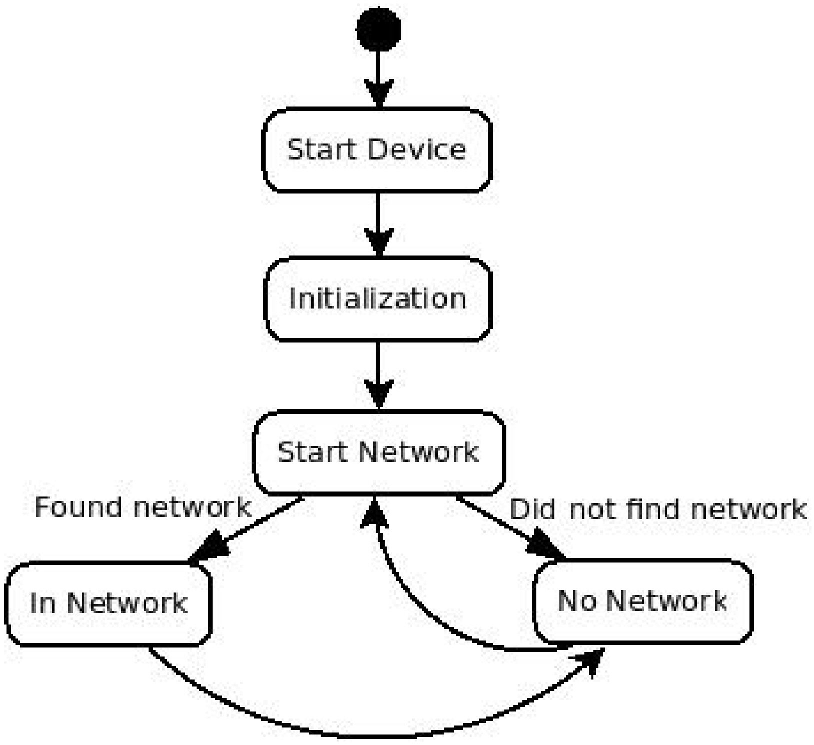


Fig. 4.3: State diagram describing general network flow of all EDM devices.

explained in detail in the Performance Analysis section of this report. See Section 6.1.4.

4.4 Router Application Components

The router device application performed regular message transfer duties conforming to ZigBee standard protocol. In addition, the router application also performed local functions to send status updates about the specific device to the base station. The functions gathered battery voltage measurements, signal strength measurements, and network status information, and sent them to the base station through the mesh network at periodic intervals set in the configuration file. These functions were executed through a state machine operating inside the router application task manager. Figure 4.4 depicts this state machine. After initialization, the router immediately sent a status message to the base station, then went into an idle state. This idle state was only for the local application. All messages sent from other devices were relayed to the router's parent node. Once confirmation that the status message was received, the application went into a scheduling state where it initialized a timer interrupt according to parameters set by the user in the configuration file. After the timer was set, the router went back into the idle state until the timer went off. At this point, the router sent a message then went back to the idle state. This cycle was designed to continue indefinitely.

4.5 Coordinator Application Components

The coordinator application was the simplest of the three devices. The only function that was performed locally, apart from the ZigBee standard functions, was handling communication over the Zigbit USART bus. Figure 4.5 shows the simple state machine implemented in the coordinator application task manager. After initialization the coordinator application cycled between idle and USART flushing states, sending cached messages received from the mesh network to the base station when the communication buffer was filled.

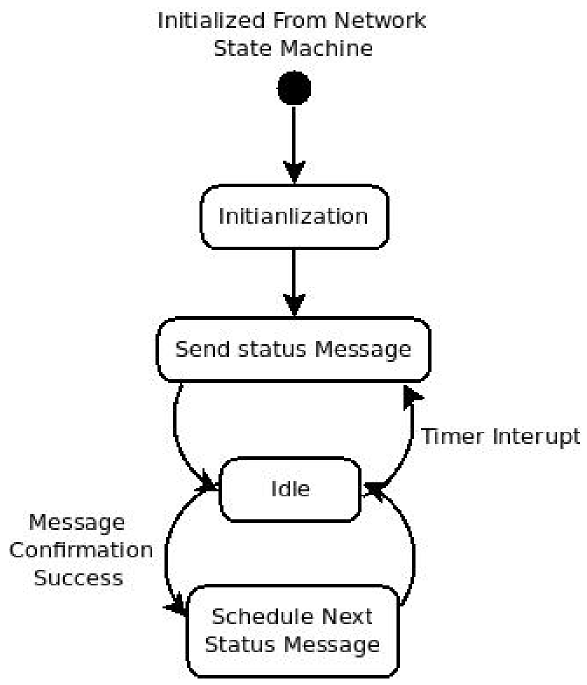


Fig. 4.4: State diagram describing local operation of router devices application.

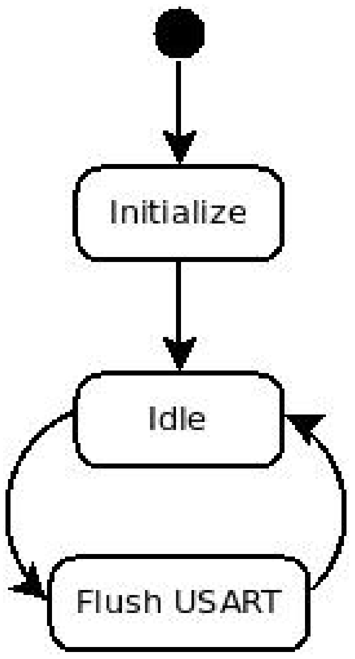


Fig. 4.5: State diagram describing local operation of coordinator device application.

Chapter 5

Software Design

As stated before, the hardware chosen to serve as the base station for the EDM project was the AMOS-3001 embedded computer system. This platform gave a very good fit to the EDM project goals; however, the software developed for the project was written such that it could be deployed on any generic PC-based system running a Linux Debian/Ubuntu operating system, set up with the correct installation packages. These packages included:

- build-essential,
- subversion (used for file version tracking during development),
- libboost-all-dev,
- libsqlite3-dev,
- sqlite-doc,
- python-pygame,
- python-matplotlib,
- python-flup,
- python-django,
- python-wxversion,
- apache2 (with needed adjustments to allow the web server to run from port 8000),
- libapache2-mod-python.

Development and testing for the EDM project was done with various platforms running these same packages with the application software developed for the EDM project. Once a platform was set up correctly, the base station would be executed in three different parts: the ZigBee base station, the SQLite database engine, and the http web server.

5.1 ZigBee Base Station

The ZigBee base station program was a server program that utilized functions from the boost C++ libraries to perform operations conforming the transmission control protocol (TCP) on the designated serial communication port. The boost library provided functions that enabled serial port communication, system error handling, file management, and other program operations [31]. The application initialized and performed standard maintenance functions on the serial port and watched for data being sent in from the coordinator. Once the data was received, the data was handed over to the SQLite database engine for further processing and analysis over a unidirectional communication socket.

5.2 SQLite Database Engine - EDMData.cpp

The SQLite C++ software library enabled the EDM application to perform the needed database archiving functions with a low-level of configuration [32]. The database application performed all the directory checks and file creation tasks. The engine then parsed incoming messages sent from the ZigBee base station program over the communication socket and archived them into the created database file. This application also utilized the boost library functions to manage data communication and attach time-stamps to the received data. The information stored in the database from each message included (not in any particular order):

- Message type,
- Sending node type (sensor or router),
- Local mesh network address of the sending node,
- UID of the specific device sending the message,

- Address of the parent node of the sending device within the network,
- Signal LQI between the sending device and its parent node,
- Signal RSSI between the sending device and its parent node,
- The relative system time of the sending device,
- Battery voltage level of the sending device,
- Accelerometer data form the sending device (sensor only),
- Horse or device name (added by user from web interface).

5.3 Web Server

The web server displayed the data stored in the database file to a web interface for user interaction. The web server displayed the information about each individual node and had tools to allow a user to download raw data for individual processing and analysis. This web interface was implemented over an Apache http server running on the PC platform [33]. The web interface was written in a combination of different languages, including: Python, Javascript, CSS, and HTML. A library of python scripts were written using python functions from the packages mentioned in Chapter 5 to perform processing functions on the data stored in the database file. The processed data, along with any processed plot images were then inserted into the web application directories to be displayed to the user. The user accessed the web interface through a web browser and entering the relative http:// address, at which point the user was taken to the server front page shown in Fig. 5.1. The user could then navigate to the Data Explorer page or Statistics page where nodes could be given horse names, data could be displayed, and data or plots of the data could be downloaded. These latter pages are shown in Fig. 5.2 and Fig. 5.3.

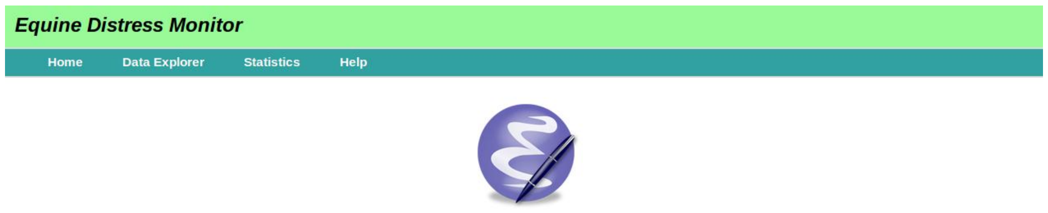
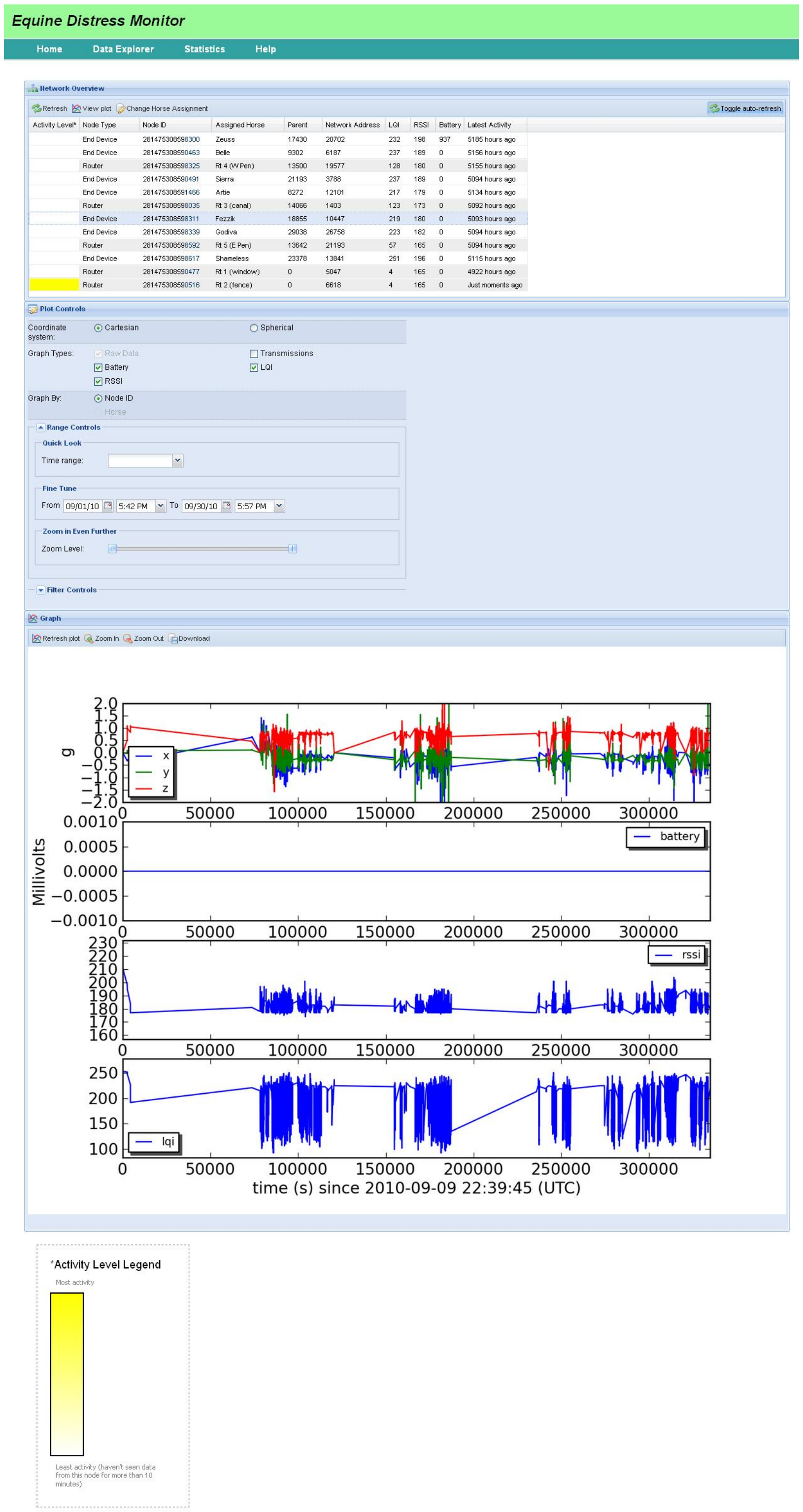


Fig. 5.1: Web server user interface front page.



Activity Level*	Node Type	Node ID	Assigned Horse	Parent	Network Address	LQI	RSSI	Battery	Latest Activity
	End Device	281475308598300	Zeuss	17430	20702	232	198	937	5185 hours ago
	End Device	281475308590463	Belle	9302	6187	237	189	0	5156 hours ago
	Router	281475308598325	Rt 4 (W Pen)	13500	19577	128	180	0	5155 hours ago
	End Device	281475308590491	Sierra	21193	3788	237	189	0	5094 hours ago
	End Device	281475308591466	Artie	8272	12101	217	179	0	5134 hours ago
	Router	281475308598035	Rt 3 (canal)	14066	1403	123	173	0	5092 hours ago
	End Device	281475308598311	Fezzik	18855	10447	219	180	0	5093 hours ago
	End Device	281475308598339	Godiva	29038	26758	223	182	0	5094 hours ago
	Router	281475308598592	Rt 5 (E Pen)	13642	21193	57	165	0	5094 hours ago
	End Device	281475308598617	Shameless	23378	13841	251	196	0	5115 hours ago
	Router	281475308590477	Rt 1 (window)	0	5047	4	165	0	4922 hours ago
🟨	Router	281475308590516	Rt 2 (fence)	0	6618	4	165	0	Just moments ago

Coordinate system: [✓ Cartesian] Spherical

Graph Types: [✓ Raw Data] [✓ Battery] [✓ RSSI] Transmissions [✓ LQI]

Graph By: [✓ Node ID] Horse

Quick Look — Time range:

Fine Tune — From 09/01/10 5:42 PM To 09/30/10 5:57 PM

Zoom in Even Further — Zoom Level:

Fig. 5.2: Web server user interface Data Explorer page.

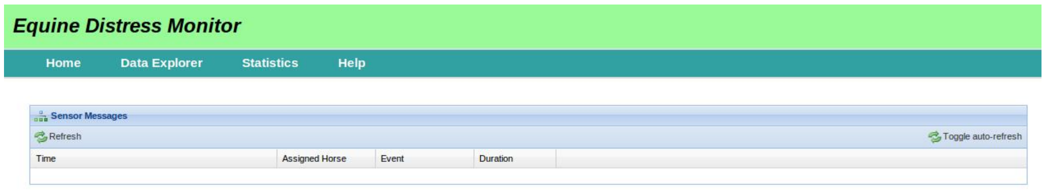


Fig. 5.3: Web server user interface Statistics page.

Chapter 6

Performance Analysis

Throughout the development of the before mentioned sub-systems, there were many tests that took place to validate the different components of the system. Deployment of a complete system took place late in the summer of 2010. This first deployment revealed many discoveries and some issues that have since been analyzed and addressed. These issues were influential in the transformation of the methods of data processing and communication and are discussed in detail in Section 6.1. Analysis of the data and the effectiveness of the system in detecting colic is discussed further in Section 6.2.

The layout of the routers and the coordinator/base station in the initial deployment setup at the Utah State University Equine Education Center (USUEEC) is shown in the map in Fig. 6.1. The labels found in this map will be referenced to describe the test setup throughout the rest of this report. This map shows an even distribution of routers to cover the outdoor areas where horses resided. This was to ensure adequate coverage of the horses that were outside, and to ensure adequate signal reliability between routers to create a stable radio transmission path from each sensor to the coordinator. During summer months, horses are placed in the "Horse Pens" and "Outdoor Stalls" depicted in the map. Figure 6.2 shows annotated photos of the router and coordinator layout described in Fig. 6.1.

6.1 System Reliability

As stated before, there were a number of problems experienced in the deployment of the EDM system. These problems ranged in frequency and diversity; however, solutions were found for the major ones, and the rest could be resolved through further analysis and design.

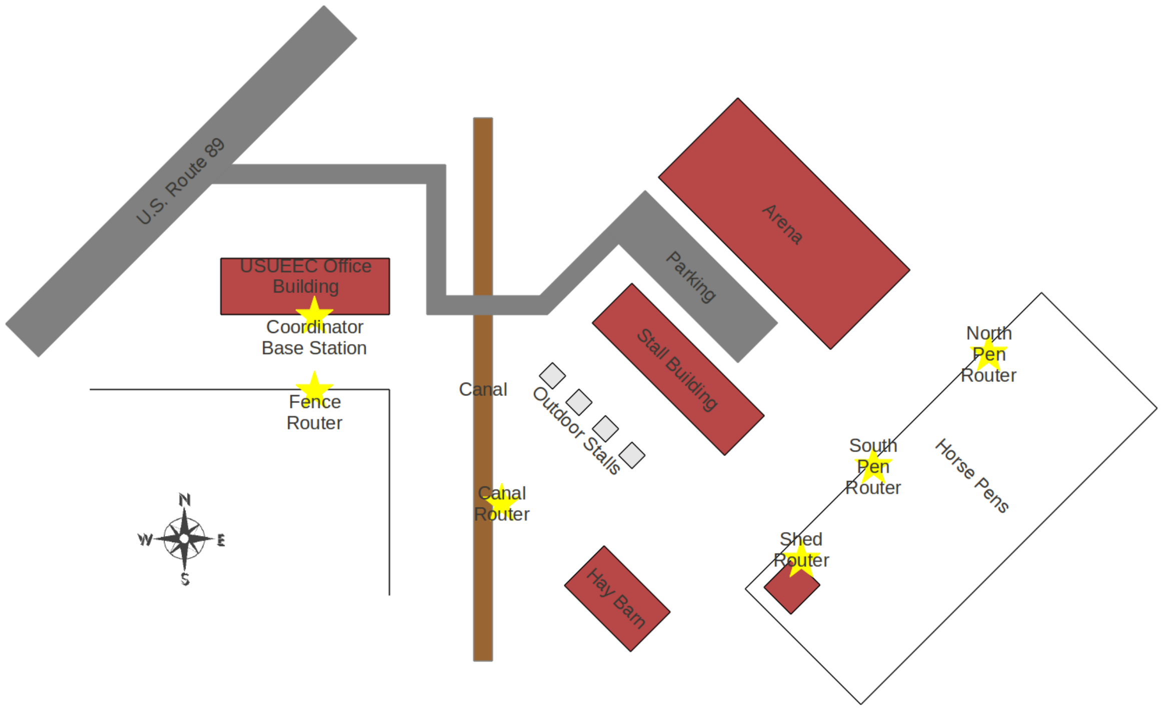
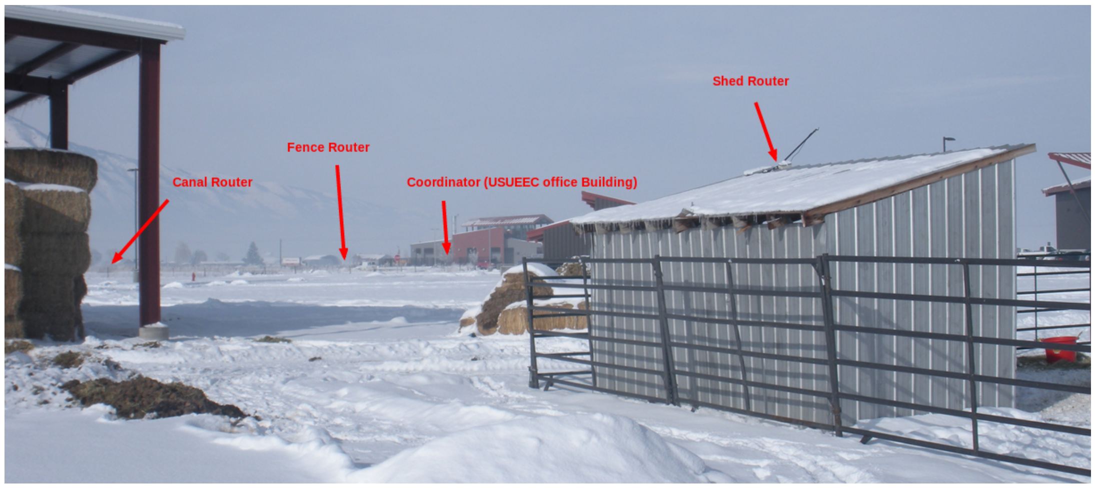


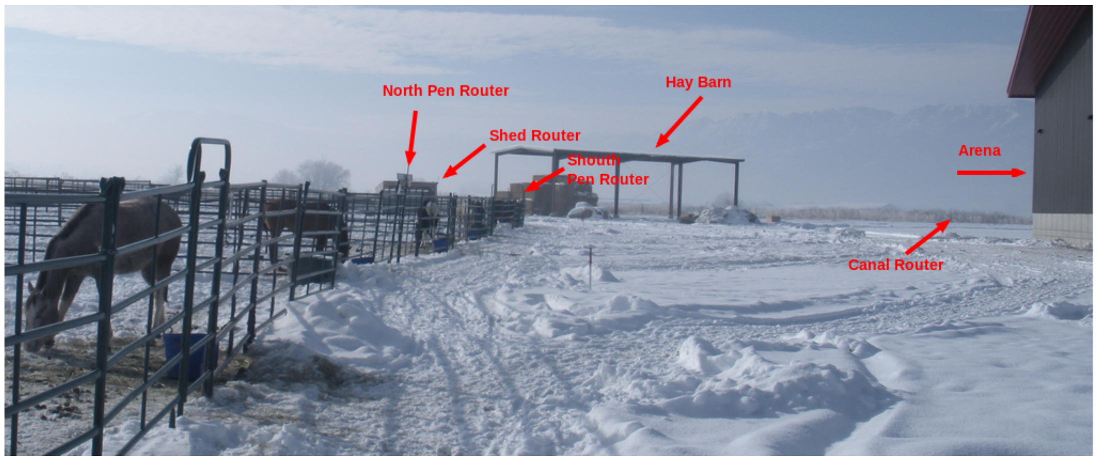
Fig. 6.1: General map of the USUEEC router and coordinator/base station setup.

6.1.1 Radio Signal Reliability

The first problems that were discovered in the initial testing of the entire system were related to signal reliability. It became apparent very quickly that the steal-covered buildings that were built at the USUEEC caused reflection and shielding effects on the RF signals of the mesh network. The initial placement of the coordinator and base station was just inside a south window in the USUEEC office building. The reliability of the signal link between the coordinator and the fence router was found to be very weak and unreliable. Signal links between the other routers varied; however, extending the antennas with the 3-foot RG-58 coaxial cable with PVC pipe stabilized them enough for the EDM project application. Efforts to rearrange the coordinator in various poses and locations inside the building did not improve link quality. Another router was placed just outside the window where the coordinator was placed in an effort to improve the link. This did improved signal strength; however, it was still not sufficiently stable for this application. At that point it was theorized that a coating on the windows, in conjunction with the steel material that



(a) South side looking west



(b) East side looking south

Fig. 6.2: Router and coordinator/base station layout photos.

was used to build the office building, were impeding signal projection between the router and coordinator. This theory was further substantiated when the base station application was deployed on a laptop computer and taken outside the building. The signal reliability improved dramatically. Further proof of this theory was seen when the laptop used as the test base station, which was equipped with a WiFi radio transceiver, could not connect to the wireless Internet setup inside the office building. It was concluded that in order for the system to work properly, the coordinator would have to be placed outside the steel building. The problem with this solution was that there would not be a way for the base station to connect to the Internet to upload the web interface application. USU facilities services were then employed to route a wired LAN connection and power outlet to an exterior electrical box to the south-facing outer wall of the building. Placing the coordinator and base station inside the exterior electrical box resolved the signal reliability problem.

6.1.2 Router Power Reliability

In designing the EDM system, it was realized that the ability to power the router devices for extended periods of time needed to be addressed. The battery packs chosen to power the router devices had capacity enough to power them for three to five days. This estimation was verified through testing; however this amount of time did not fulfill the project goals. These battery packs met design requirements for the solar application. This solar charging application was effective in keeping the routers powered until temperatures in the Logan, Utah area dropped below freezing, at which point the NiMH batteries depleted and the routers turned off. To address this problem, lead acid batteries were purchased to replace the NiMH battery packs.

6.1.3 Sensor Power Reliability

The solution to keep the sensor devices active for long periods of time was an application of conservation rather than regeneration. Testing the EDM system revealed that when the sensors were placed on the horses, they were activated more often than was expected. This frequent activation caused the sensor batteries to deplete very rapidly. Resolving this issue was done by changing the behavior of the device through applying a different behavior model in the sensor application which is described in Section 6.1.4.

6.1.4 Network Hand-offs vs Sensor Processing

The behavior of the sensor device was the most critical in accomplishing the project objectives. It was imperative that the timing and modes of operation of the sensor be correct in order for reliable and valid colic detection to be accomplished. It was also important that the behavior be as conservative as possible in consuming power. Because of this, the sensor application changed and evolved many times throughout the project. Most of the iterations were not deployed in large scale and will not be discussed. The sensor application included in the first large-scale deployment of the system is described below as the "Original State Behavior of the Sensors" and represents the culmination of all the previous iterations to that point. This application offloaded the processing of raw accelerometer data to the base station which had greater processing resources and could conduct more intelligent processing and analysis. Upon testing the original design, it was found that some redesign of the application was needed. After further development, the sensor application described below as "Final State Behavior of the Sensors" was deployed and tested. This final application moved the processing of raw accelerometer data to the sensor device where more primitive means of detecting colic and casting events were utilized.

Original State Behavior of the Sensors

The behavior of the original state machine programmed onto the sensor devices is described graphically in Fig. 6.3. Upon initialization, the device sent one message to the base station then went immediately into the low-power sleep state. The sensor was meant to spend the majority of time in this state to ensure power conservation for long-term monitoring. Radio transmission only occurred when a casting or colic event was detected by the sensor through certain state machine logic transition paths described below. At this point, the sensor began transmitting raw data from the accelerometer through the mesh network to the base station for further processing and analysis. Transmission would occur for a set period of time, then the sensor returned to the sleep state. Exiting the sleep state only happened when an interrupt signal form the tilt switch indicated that the horse had tilted its head more than forty five degrees for a set period of time. When this occurred, the sensor would then move to a wake state that acted as a low-pass filter to verify that a tilt event really occurred. If a tilt was deemed as invalid the sensor would go back to the sleep state. Confirmation of a true tilt event caused the sensor to activate a timer interrupt and wait for one of two things to happen: first, the sensor periodically monitored the tilt switch for an inactive signal when, upon confirmation of the inactive signal, would wait for a second tilt event to occur. The second condition causing the sensor to transition from this state was if the preset timer interrupt occurred before an end of tilt was detected. This would indicate a possible casting behavior and the sensor would move into the data transmission state. If the sensor had transitioned from the confirmation of an end of the first tilt event to the second tilt detection state, it would periodically check the tilt switch for a second tilt event. If no event occurred after a set period of time, this indicated that the horse did not have colic and the sensor went to the sleep state. If a second tilt event was confirmed to have happened the sensor would transition to the transmission state.

Data Gaps Due to Network Hand-offs

While trying to analyze the data obtained in the first deployment of the EDM system, there were many gaps that impeded effective analysis. We found that sensors placed in proximity to two multiple routers tended to switch routing between routers when signal strength in the current parent router became too weak. This hand-off scenario made colic and casting events difficult to detect. To support our conclusion concerning hand-off data loss, a test was performed under a controlled setting. Multiple routers were set up in a fashion that a sensor that would pass by each of the routers would be forced to switch between parent routers. The sensor was programmed to transmit accelerometer data continuously through the Mesh network to the base station. Once set up correctly, the sensor was walked past each router several times. When a router was passed by, the sensor was shaken vigorously so that the event could be easily identified in the data. The strait, clean

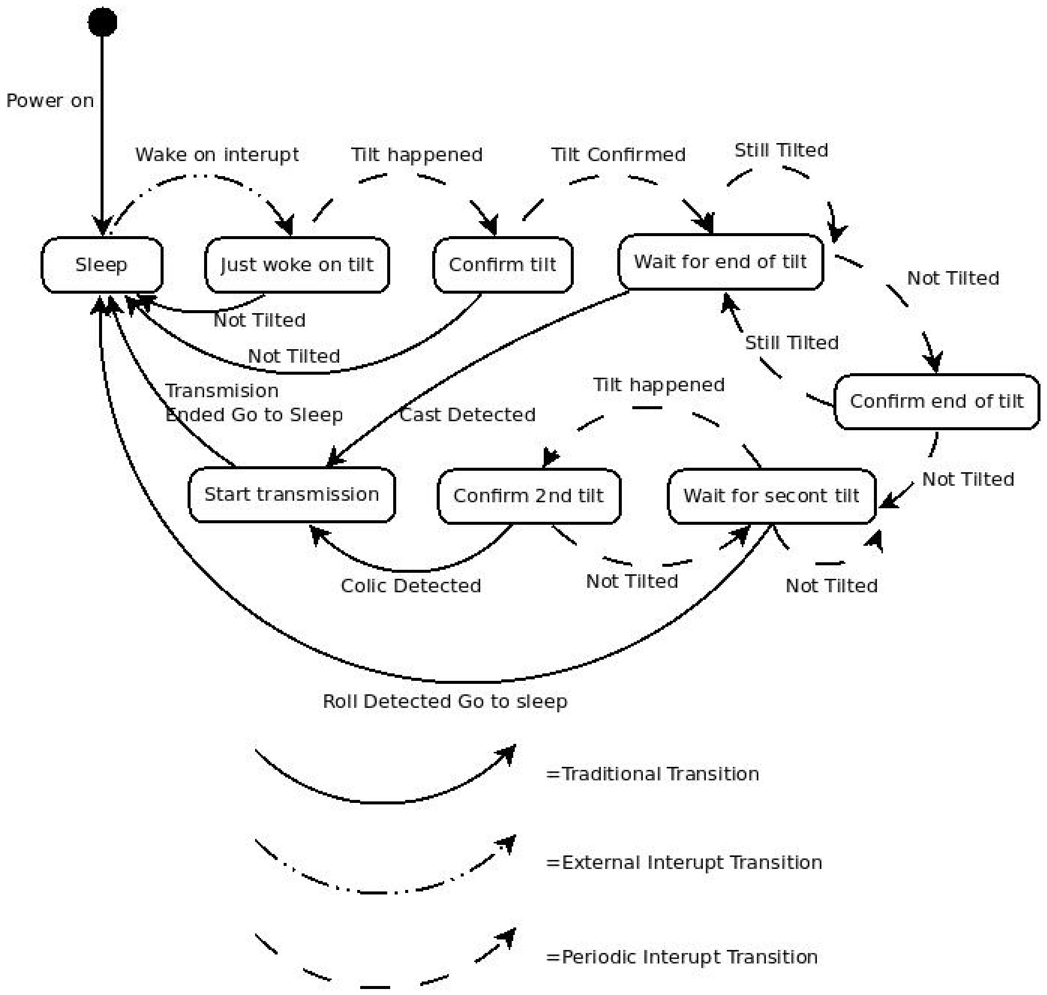


Fig. 6.3: Original sensor state diagram.

lines in Fig. 6.4 show the data gaps as the sensor changed from one parent router to another. These gaps posed challenges for automated data analysis. The solution developed to overcome this issue was to move data processing algorithms to the sensors, rather than trying to reliably analyze incomplete data at the base station.

Power Consumption Analysis

Another way that sensor power consumption was reduced was to decrease the amount of time that certain parts of the Zigbit module were activated. An example of this achievement was in the way the radio module in the Zigbit was kept inactive until the time it was actually used. In order to investigate and illustrate how much power was used by the sensor, a power supply was connected through a series resistor to sensors programmed with the different algorithms. An oscilloscope was then used to measure the voltage-drop across the series resistor at which point the amount of current being consumed by the sensor at different points in the algorithms were calculated using the measured voltage-drop and resistance of the series resistor. Figures 6.5 and 6.6 show a comparison of oscilloscope screen shots measuring the power consumption of the two main algorithms in a similar state. The state at which the power was measured was shortly after a tilt event was detected by each device and was waiting for the tilt event to end. Testing revealed that in this particular state the algorithm that performed data analysis on the sensor processor consumed more power. This was not unexpected since the processor was doing more work in this configuration, but it showed the trade-off that was needed to have adequate performance while conserving energy.

Figure 6.7 shows a case where the final configuration algorithm consumed less power. Figure 6.7 shows a measurement of power while the sensor programmed with the original algorithm was searching for a mesh network host. A comparison for this power consumption was not shown because the device programmed with the final algorithm went into an ultra-low-power state where insignificant amounts of energy were consumed when no network host was found within a short period of time.

Other states of the devices gave very similar power consumption measurements in both

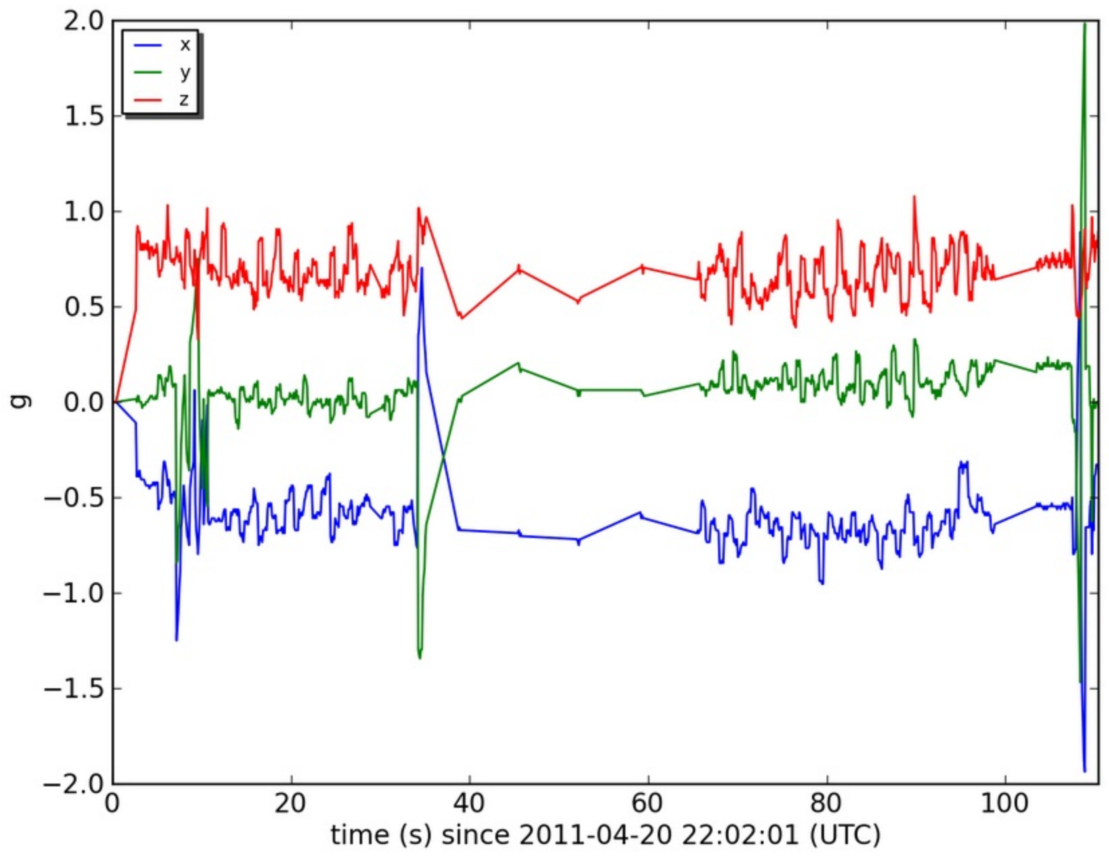


Fig. 6.4: Data illustrating data gaps found in hand-off testing.

algorithms. This told us that more testing was needed to discover other ways the power consumption could be reduced. One key factor that made the most difference was the frequency of sensor activation over extended periods of time. In order to have a valid comparison for this, more time was needed before the end of the project time line for more focused testing and code iteration.

In the commercial configuration of the sensor circuitry and programming, it will be important of exact power consumption measurements to be made so that scheduled periodic battery changes and/or recharging can be planned in the final deployment. Over the course of this project, relative measurements were made to find methods to reduce power consumption. Further testing was needed after the project to find an acceptable balance between performance and efficiency.

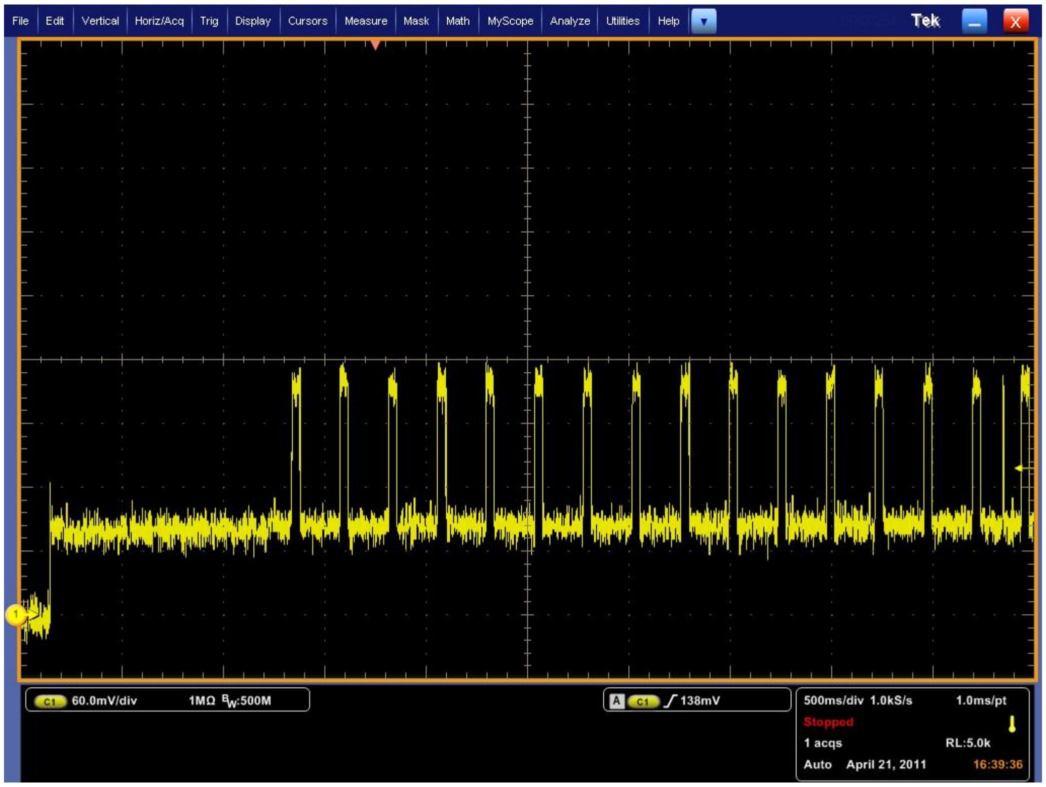


Fig. 6.5: Final sensor configuration power measurement while in “wait for end of tilt” state.

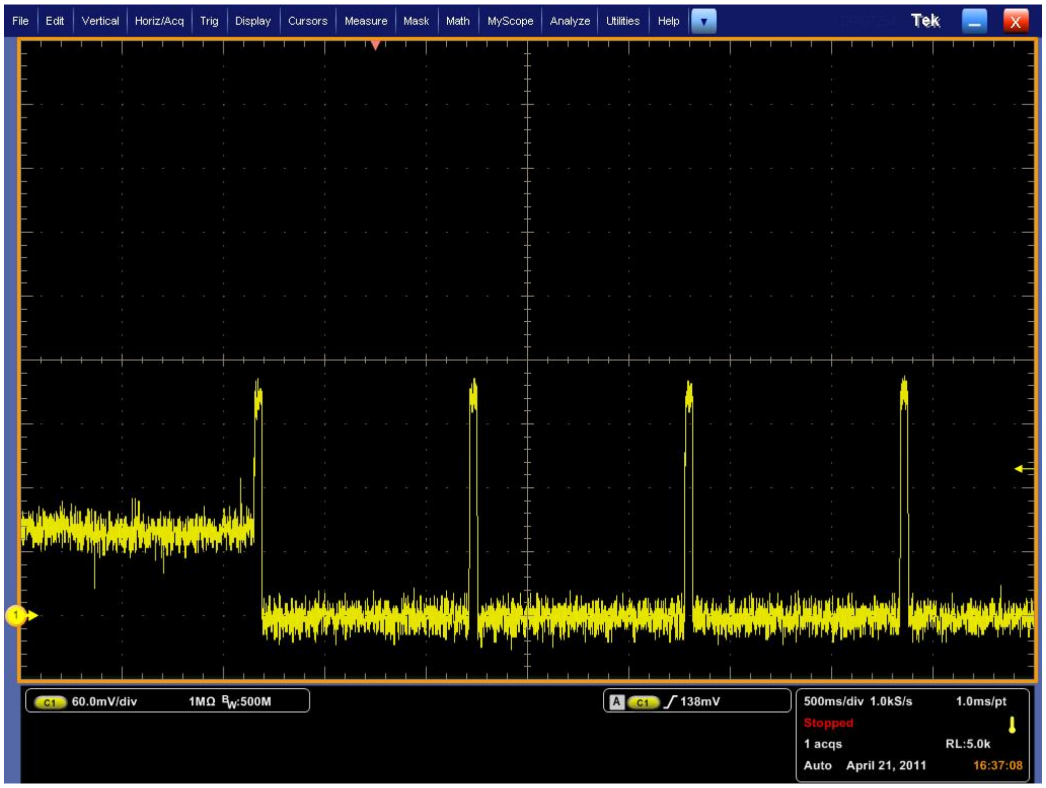


Fig. 6.6: Original sensor configuration power measurement while in “wait for end of tilt” state.

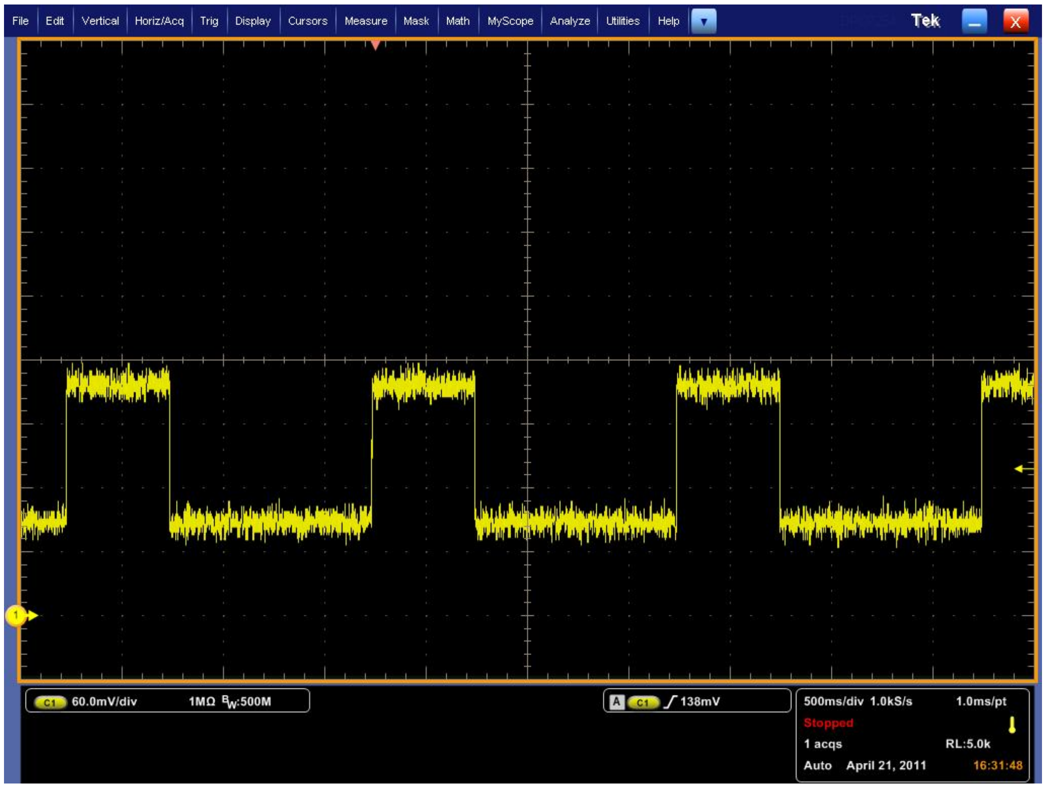


Fig. 6.7: Original sensor configuration power measurement searching for a mesh network host.

Network Traffic Analysis

Another problem found in analyzing the initial deployment was that there was a significant amount of background network activity on the mesh network found using a network activity measurement probe. This network activity was theorized to be part of the cause of the sensor battery depletion problem as well as an additional cause of gaps in sensor data transmitted to the base station.

Final State Behavior of the Sensors

The changes in state behavior in the sensor application helped minimize the number of transmissions. Figure 6.8 shows the state flow diagram for the final sensor application. The basic logic for detecting colic and casting events remained mostly unchanged; however, the method used to switch the sensor between states changed dramatically. The greatest change was the number of messages transmitted from the sensor device to the base station. Instead of transitioning into a transmission state, where all the data acquired from the accelerometer was transferred to the base station for processing, the accelerometer data was processed by the sensor and event messages were sent when significant events were detected in the data processing. Upon activation through an interrupt from the tilt switch, the sensor began filling a buffer with the data from the accelerometer and processed the data using the algorithms described in Section 6.1.5. This gave the sensor more intelligence in detecting tilt, shake, casting, and colic events. This state machine logic essentially followed the same basic logic applied in the original state machine. After awakening from an interrupt event from the tilt switch, the tilt event was confirmed using tilt switch state and accelerometer data. If a tilt was deemed invalid from this analysis, the sensor went back into its sleep state. If the tilt event was confirmed, the sensor waited for the tilt to end. If the tilt event did not end within a set period of time, the sensor sent a casting message to the base station, then went back to sleep. If an end of tilt was detected and confirmed, the sensor then waited for a second tilt event, at which point it transitioned back to the first tilt confirmation state. If a second tilt event did not occur within a set period of time, this indicated that the horse was healthy and the sensor went back to sleep. This method of detecting significant horse behavior resulted in much fewer radio transmissions which helped improve the sensor's battery life. Utilizing the accelerometer data to detect horse behavior, instead of just the tilt switch, also helped in reducing the number of message transmissions, and it allowed the processing of data without any significant data gaps. The reduced number of messages being sent through the mesh network reduced the risk of collisions and missed messages. As a result of this change in message format, the accelerometer data fields were used to transfer information about the duration of a roll event as well as the number of times the device had awakened since the last transmission. This modified data packet format is represented in Fig. 6.9.

Another major change to the sensor application was the management of the radio transceiver. As explained before, the power consumption of the sensor, with the original application, was too high. One reason for this was the fact that the application was keeping the radio transceiver enabled whenever the device was not in a sleep state. The final application program kept the radio transceiver turned off whenever it was not being used.

6.1.5 Horse Behavior Detecting Algorithms

In order to verify that a tilt event had occurred, a function was written to process the accelerometer data stored in the buffer mentioned previously. Each sample contained data from the x, y, and z axis of the accelerometer. Each axis sample was compared to preset thresholds in relation to thresholds of the other axis samples. If a significant number of samples were found to be within these threshold conditions, the function determined the tilt event as valid. Similar to the above mentioned function, another function was written that checked if the horse was in an upright position. A similar algorithm determined whether this function would return an upright posture condition.

Detecting horse shake events required more intelligent processing. This algorithm analyzed all the data from the Z axis of the accelerometer that was stored in the buffer. Each sample was compared to the previous sample to detect significant differences between the samples. Significant differences indicated high energy movements, consistent with the shaking movements of a horse. If a significant number of samples comparisons were found to

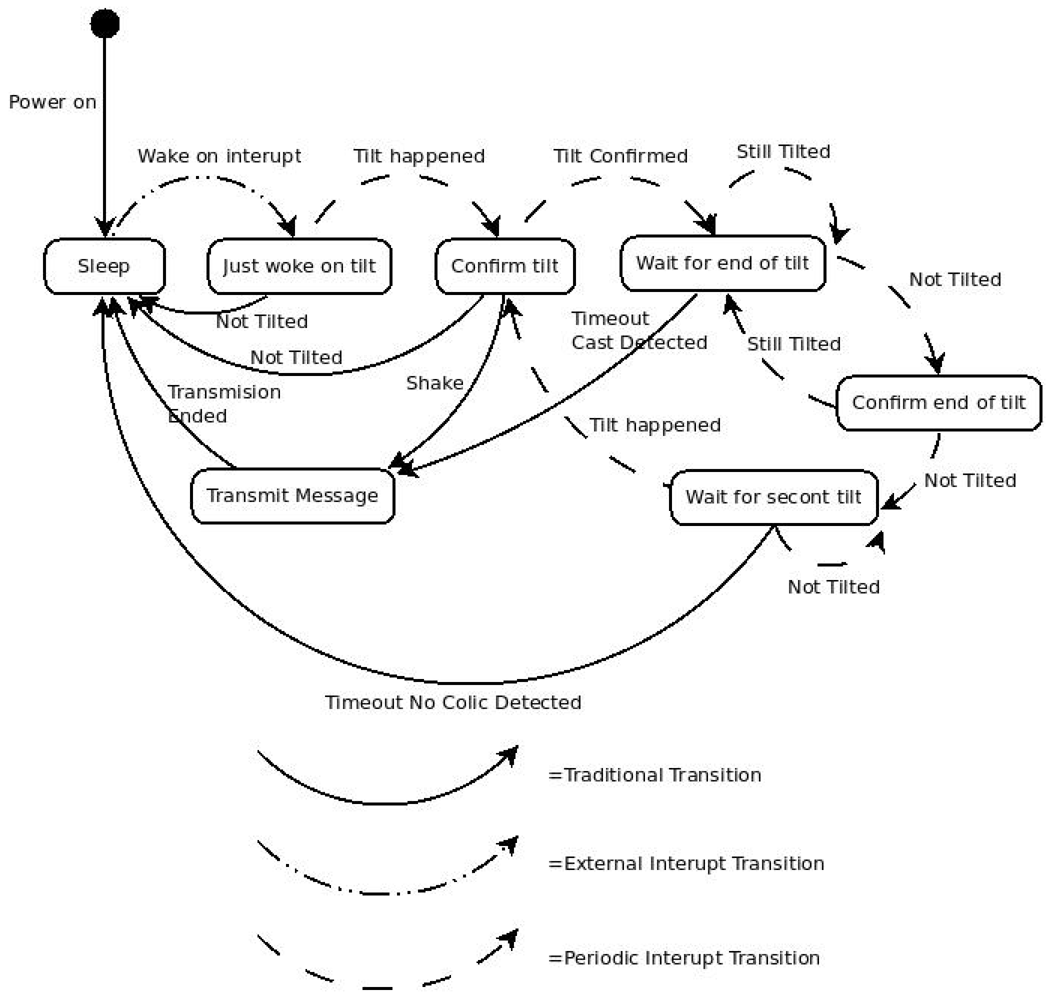


Fig. 6.8: Final sensor state diagram.

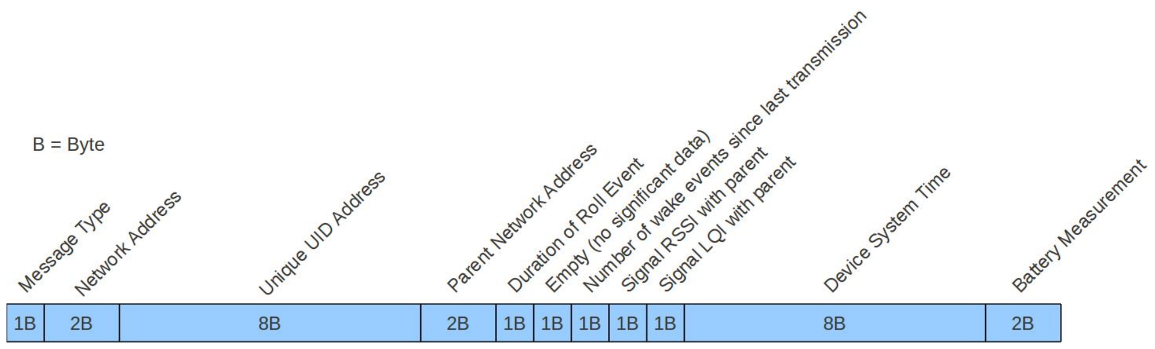


Fig. 6.9: Final EDM data packet format.

indicate this high energy behavior, the function returned an indication that a shake event had occurred.

6.2 Colic Detection Analysis

During initial and followup testing, there was not a confirmed colic or casting event detected. It was confirmed to detect instances when horses rolled and shook. In addition there were still a few fine tuning adjustments in configuration parameters that needed to be made in order to filter out erroneous data. There were plans made to coordinate with local veterinarians to measure movements of horses during euthanization. This would allow the system to be analyzed under more controlled conditions on horses that perform colic indicating movements. Future activities are beyond the scope of this report.

References

[1] C. M. Kahn, S. Line, and S. E. Aiello, *The Merck Veterinary Manual*, 9th ed., Merck & Co., Inc., Merial Limited, Whitehouse Station, NJ, USA, 2008. [Online]. Available: http://www.merckvetmanual.com

[2] A. Tadlock. The ultimate horse site. internet. Jan. 2011. [Online]. Available: http://ultimatehorsesite.com/

[3] D. D. Beaver, “Equine inverted posture alarm,” United States of America Patent 09/766,240, 2002.

[4] T. Pfau, T. H. Witte, and A. M. Wilson, “Centre of mass movement and mechanical energy fluctuation during gallop locomotion in the thoroughbred racehorse,” *The Journal of Experimental Biology*, vol. 209, pp. 3742–3757, July 2006.

[5] G. B. Rugg, “Monitoring system for animal husbandry,” United States of America Patent 7 335 168, 2008.

[6] M. A. M. Davis, “Animal instrumentation,” United States of America Application US 2007/0 130 893 A1, 2007.

[7] L. T. Skeggs, “Foaling alarm,” United States of America Patent 4 055 839, Oct 25, 1977.

[8] R. Lowe, “Livestock breeding and management system,” United States of America Patent US 8 066 179, Nov. 29, 2011.

[9] Meshnetics website. Internet. Meshnetics. [Online]. Available: http://www.meshnetics.com/

[10] Silicon serial number with vcc input ds2411. Maxim. [Online]. Available: http://datasheets.maxim-ic.com/en/ds/DS2411.pdf

[11] Header, female 1mm cc, 2 rows, smt sfm210-lpse-d05-sp-bk. Sullens. [Online]. Available: http://www.sullinscorp.com/drawings/174_SFM210-LP_E-D___-S_-BK,_11200-A.pdf

[12] Mpp-21pick and place system. APS Novastar. [Online]. Available: http://www.apsgold.com/pick-and-place/manual-systems/mpp-21-pick-and-place-system

[13] 3m header n2510-6002-rb. 3M. [Online]. Available: http://multimedia.3m.com/mws/mediawebserver?66666UuZjcFSLXTtMXM6MxfcEVuQEcuZgVs6EVs6E666666--

[14] Zigbit 2.4 ghz wireless module atzb-24-a2. Atmel. [Online]. Available: http://www.atmel.com/dyn/resources/prod_documents/doc8226.pdf

[15] On/off tilt sensor, horizontal normally open sq-sen-845. Singal Quest. [Online]. Available: http://www.signalquest.com/datasheets/SQ-SEN-8xx%20On-Off%20Tilt%20Sensor,%20Horizontal%20Normally%20Open%20Datasheet.pdf

[16] Mems motion sensor 3-axis - 2g/ 8g smart digital output piccolo accelerometer lis302dl. ST. [Online]. Available: http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/DATASHEET/CD00135460.pdf

[17] 84m ohm p-channel mosfet mic94052. Micrel. [Online]. Available: http://www.micrel.com/_PDF/mic94052-53.pdf

[18] Low-r-on low voltage spst analog switch fsa1156l6x. Fairchild Semiconductor. [Online]. Available: http://www.fairchildsemi.com/ds/FS/FSA1156.pdf

[19] Zigbit 2.4 ghz amplified wireless module atzb-a24-uf. Atmel. [Online]. Available: http://www.atmel.com/dyn/resources/prod_documents/doc8228.pdf

[20] 800ma low dropout positive regulator lt1117cst-3.3. Linear Technology. [Online]. Available: http://cds.linear.com/docs/Datasheet/1117fd.pdf

[21] Tenergy-72-5000-nimh. Tenergy. [Online]. Available: http://www.batteryjunction.com/tenergy-72-5000-nimh-pack.html

[22] Ip65 rated outdoor enclosure sodct6235-2.5. Pac Tec. [Online]. Available: http://www.pactecenclosures.com/product-detail.php?productid=219&seriesid=38&classid=35

[23] Pm7102 valor ethernet unregulated dc/dc converter. Valor. [Online]. Available: http://www.ben.cz/_d/datasheet/pm7102.pdf

[24] The r-78xx-series high efficiency switching regulator r-789.0-0.5. Recom. [Online]. Available: http://www.recom-international.com/pdf/Innoline/R-78xx-0.5.pdf

[25] Single-chip usb to uart bridge cp2102. Silicon Labs. [Online]. Available: https://www.silabs.com/Support%20Documents/TechnicalDocs/cp2102.pdf

[26] Usb mini-b connector 54819-0572. Molex. [Online]. Available: http://www.molex.com/pdm_docs/sd/548190572_sd.pdf

[27] Single circuit type for large current nfm21pc104r1e3d. Murata. [Online]. Available: http://search.murata.co.jp/Ceramy/image/img/PDF/ENG/L0111S0111NFM21P.pdf

[28] Chip ferrite bead for ghz noise blm18hg471sn1d. Murata. [Online]. Available: http://search.murata.co.jp/Ceramy/image/img/PDF/ENG/L0110S0101BLM18H.pdf

[29] Amos-3001ultra compact fanless embedded system with epia-p820 or epia-p720 pico-itx board. [Online]. Available: http://www.via.com.tw/en/products/embedded/ProductDetail.jsp?productLine=2&id=1051

[30] Bitcloud sdk for atzb-dk-24 / atzb-dk-a24 / atzb-dk-900 v. 1.4.1. Internet. Meshnetics/Atmel. [Online]. Available: https://intranet.ee.ic.ac.uk/t.clarke/projects/Resources/BitCloud/BitCloud_ZDK_1_4_1/Documentation/HTML%20help/main.html

[31] boost c++ libraries. Internet. [Online]. Available: http://www.boost.org/

[32] Sqlite software libraries. Internet. Bloomberg, Mozilla, Oracle, Adobe. [Online]. Available: http://www.sqlite.org/

[33] Apache documentation website. Internet. The Apache Software Foundation. [Online]. Available: http://www.apache.org/

Appendices

Appendix A

Sensor Board Design Documents

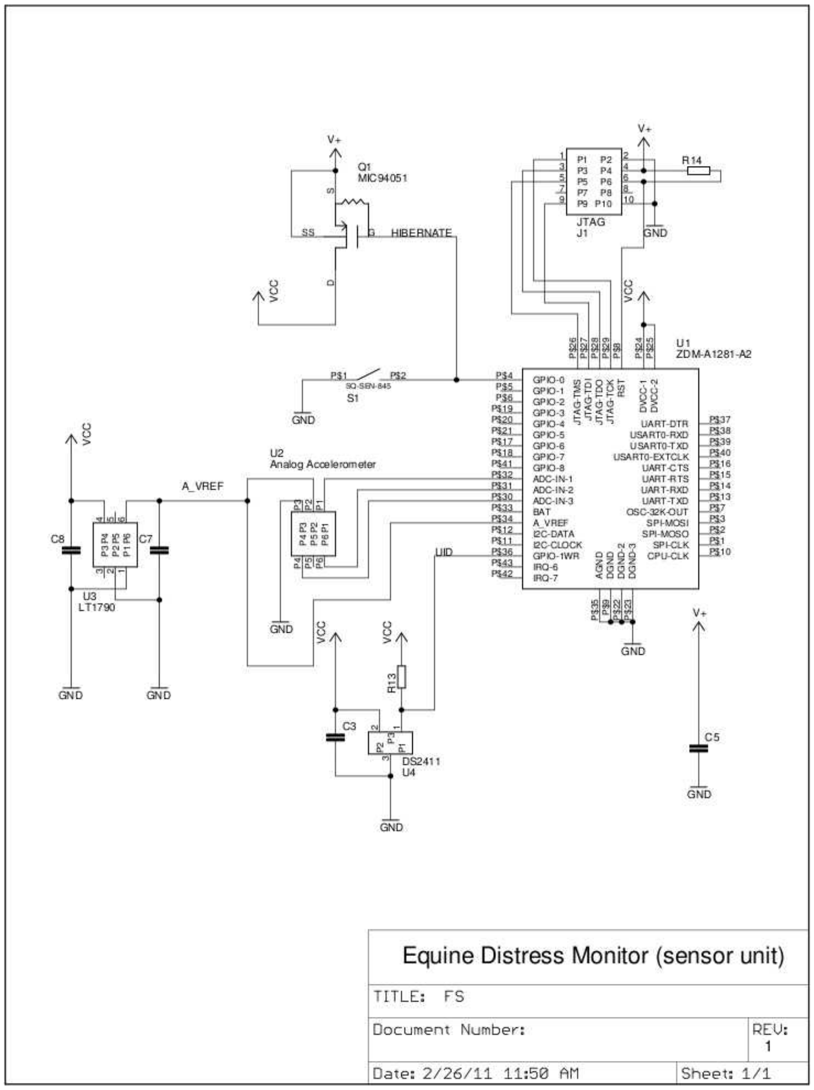


Fig. A.1: Sensor Schematic V1.0.

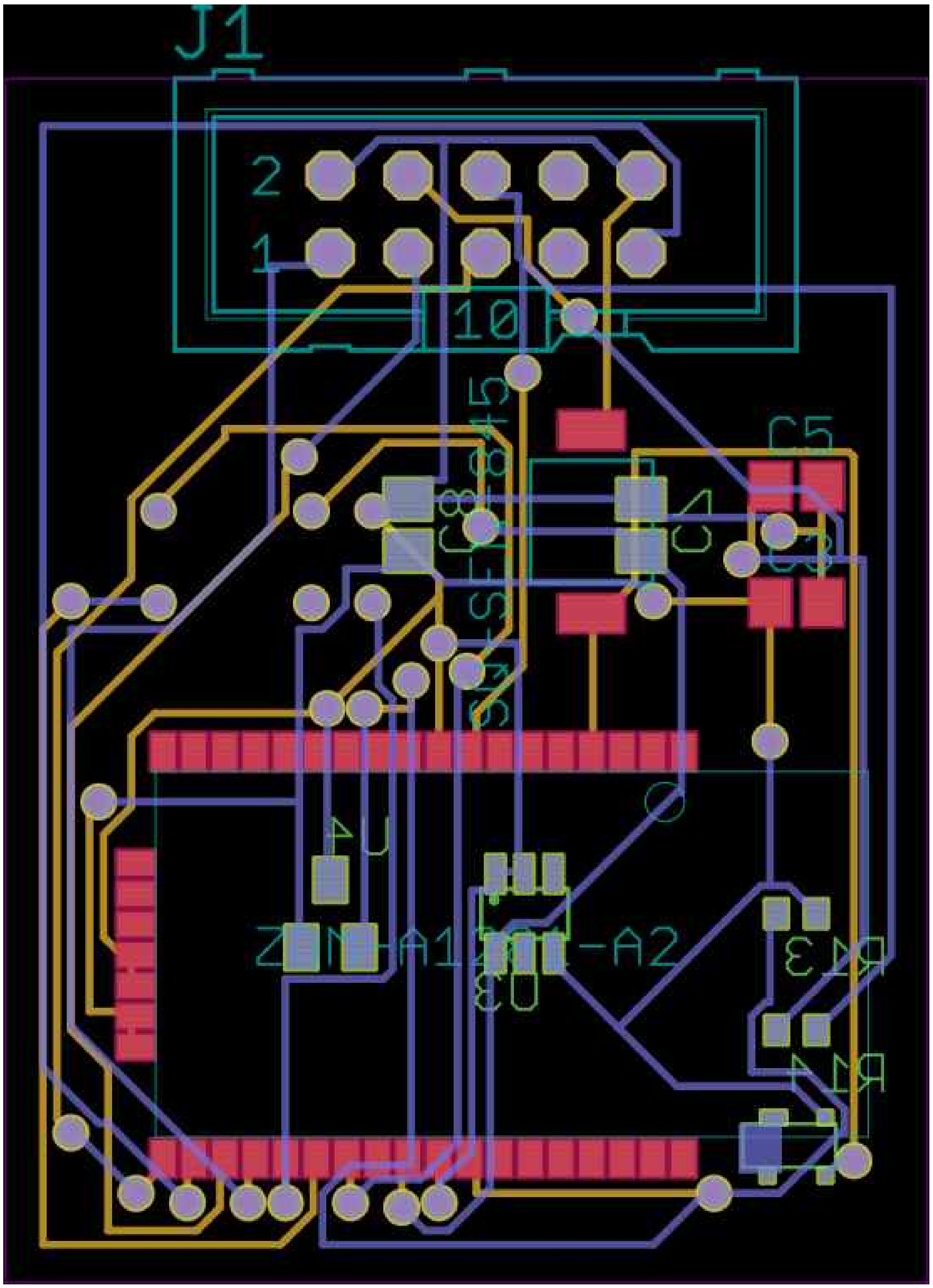


Fig. A.2: Sensor Layout V1.0.

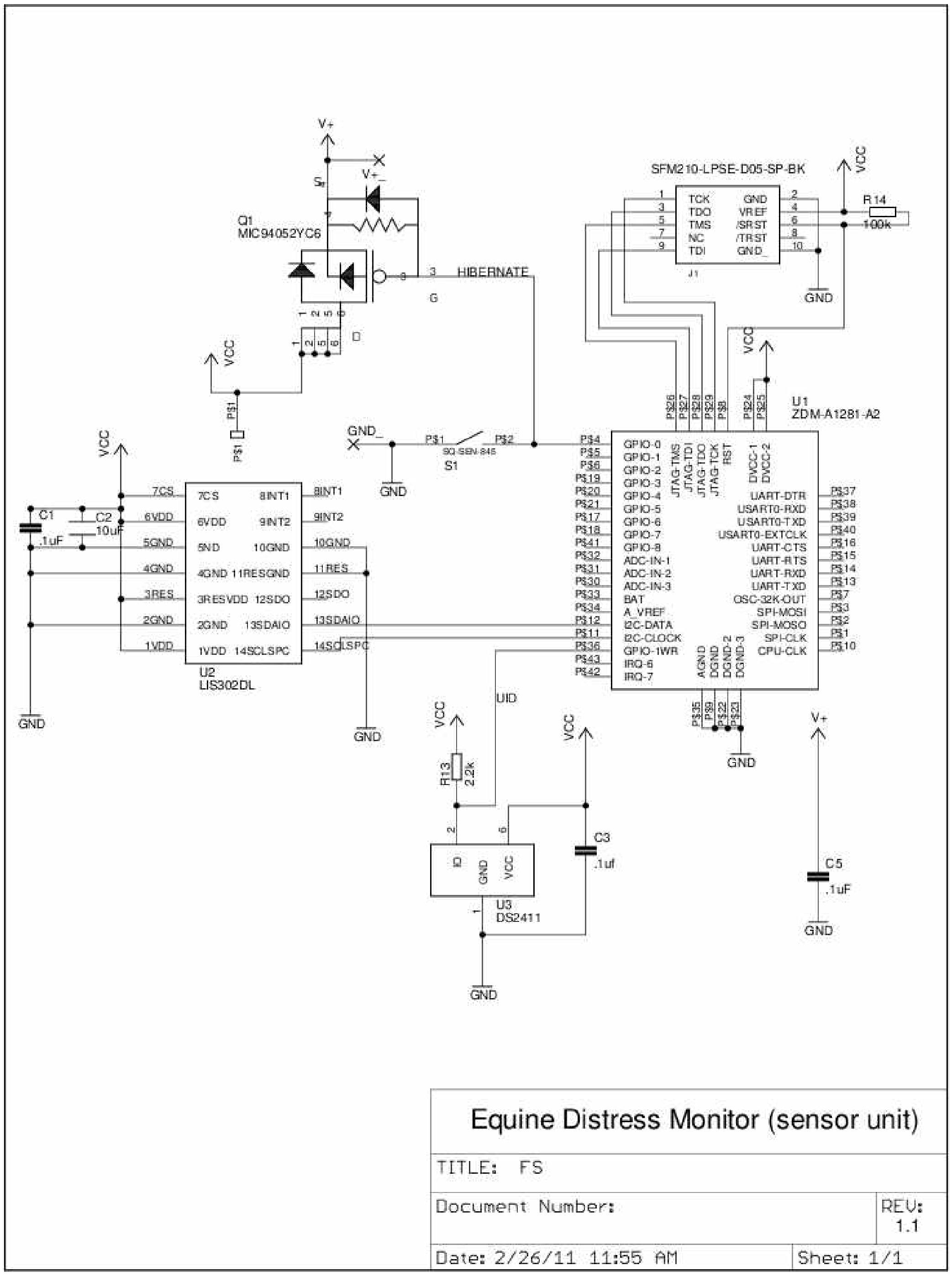


Fig. A.3: Sensor Schematic V1.1.

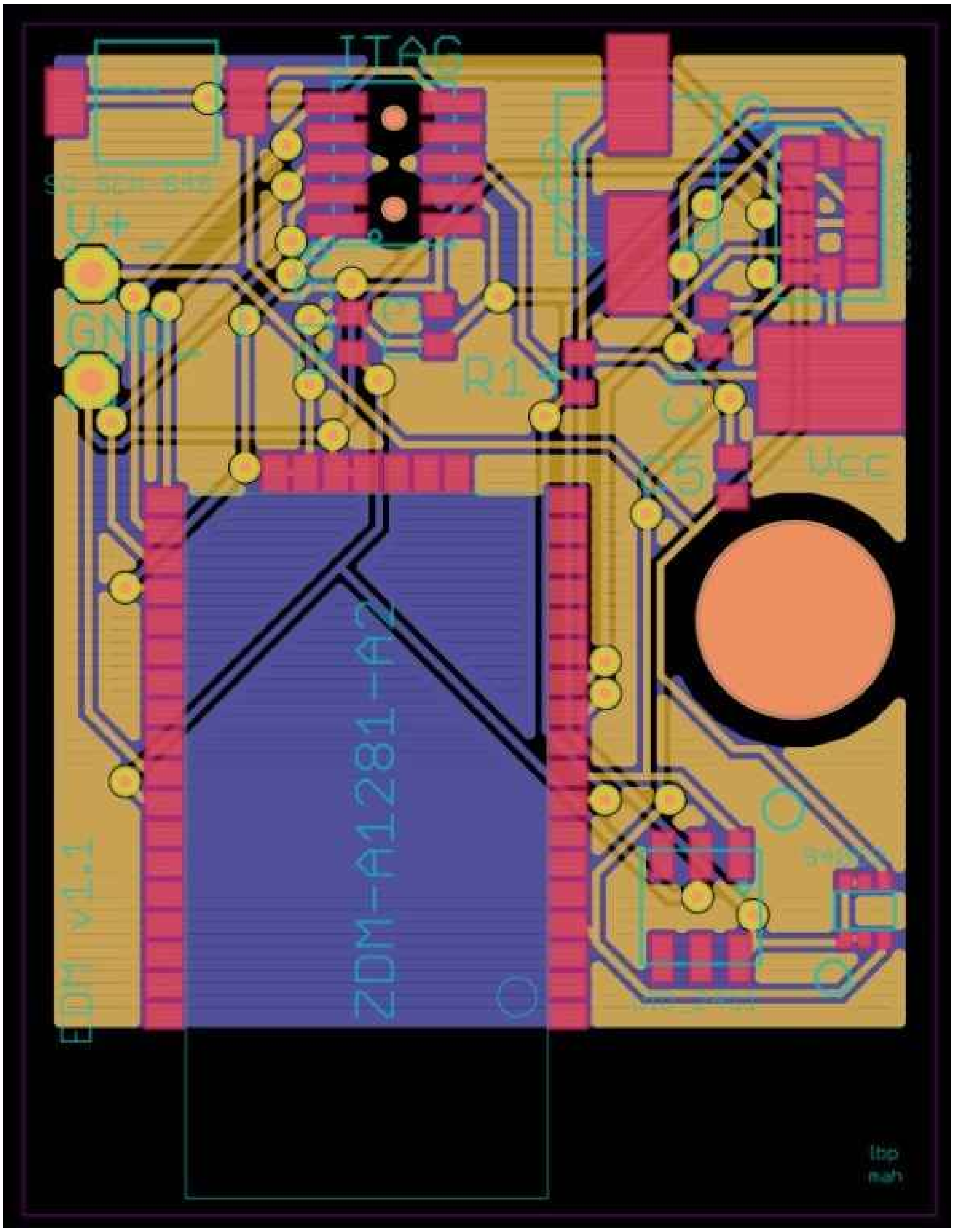
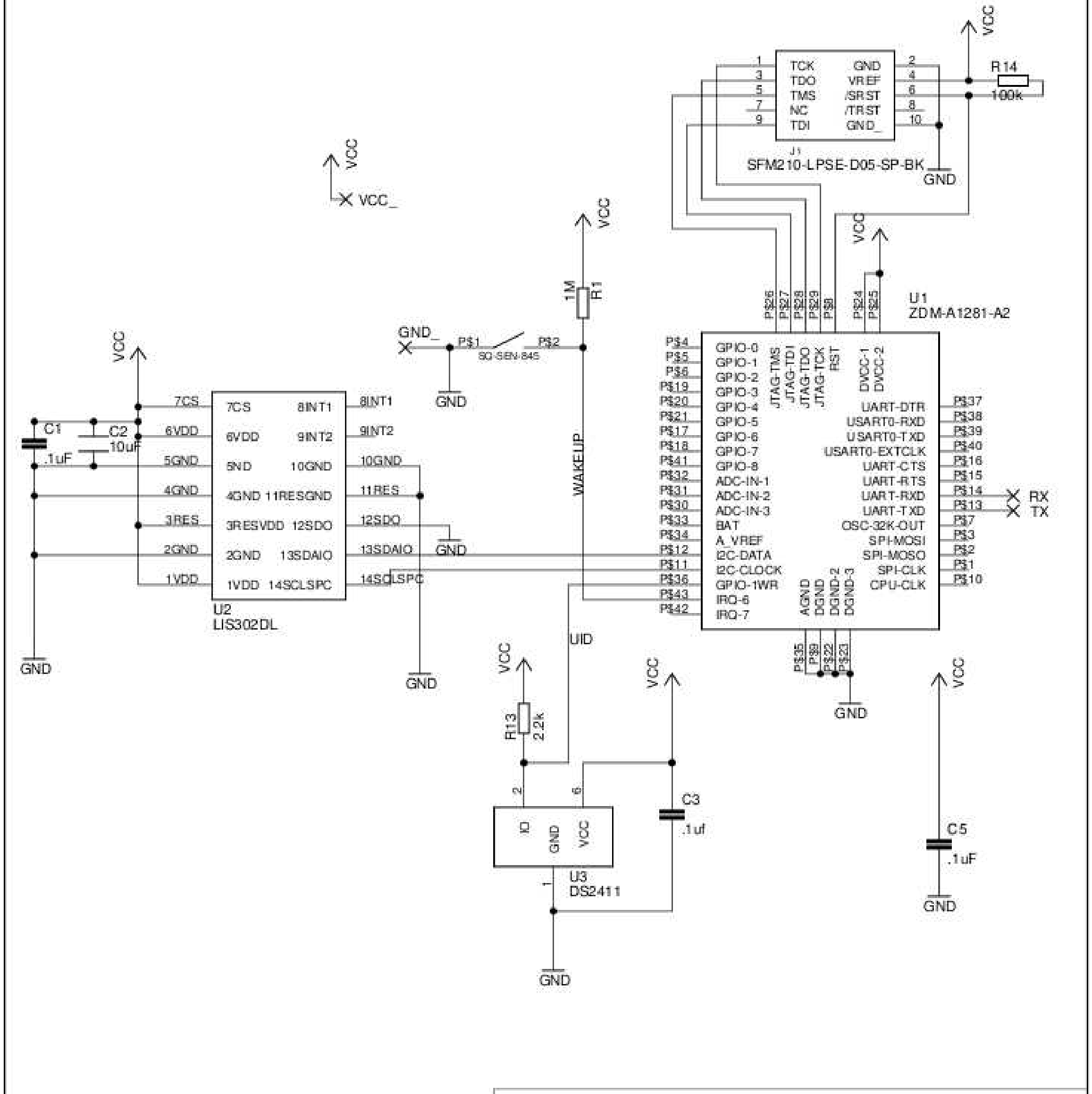


Fig. A.4: Sensor Layout V1.1.



Equine Distress Monitor (sensor unit)

TITLE: FS	
Document Number:	REV: 1.2
Date: 2/26/11 11:22 AM	Sheet: 1/1

Fig. A.5: Sensor Schematic V1.2.

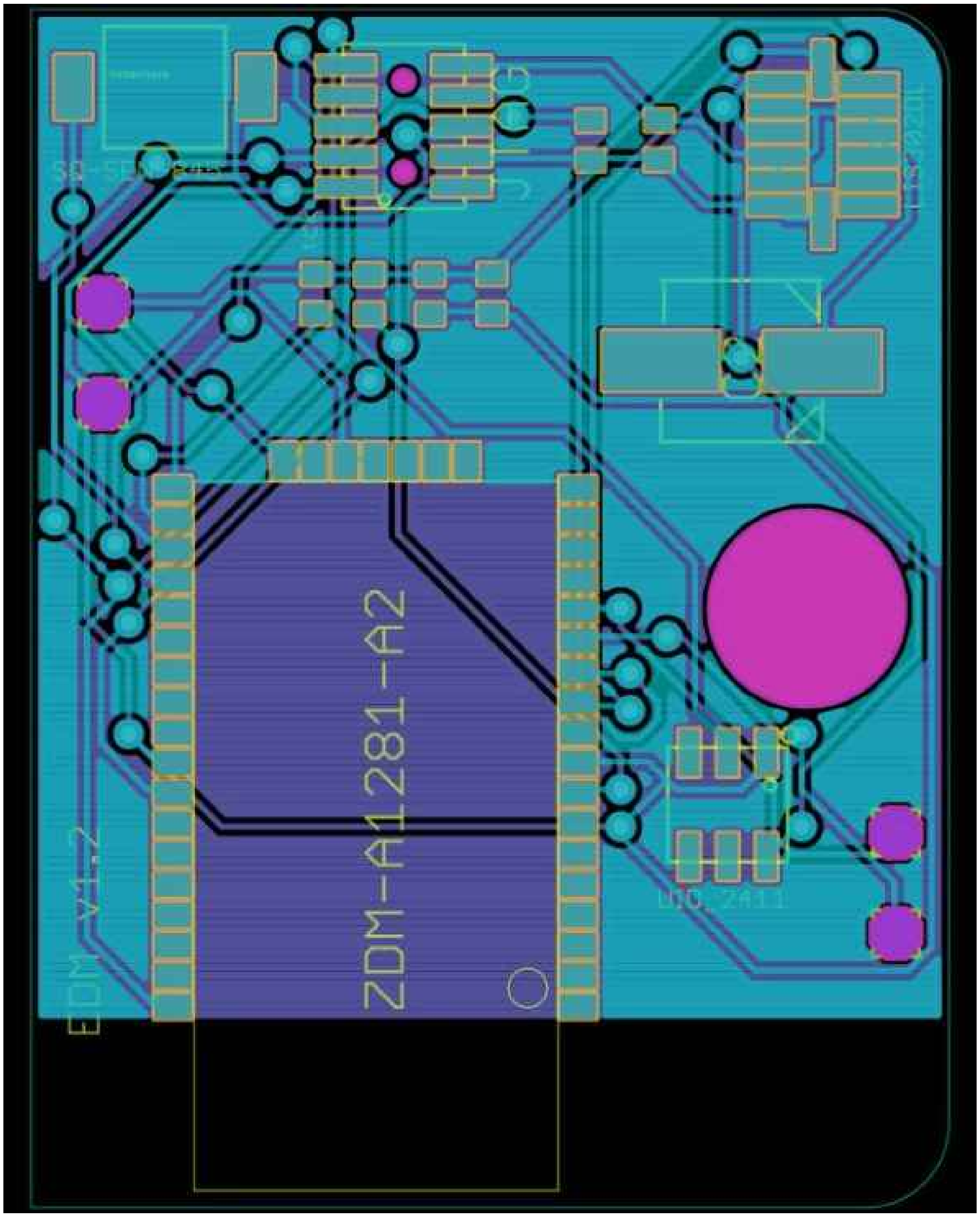


Fig. A.6: Sensor Layout V1.2.

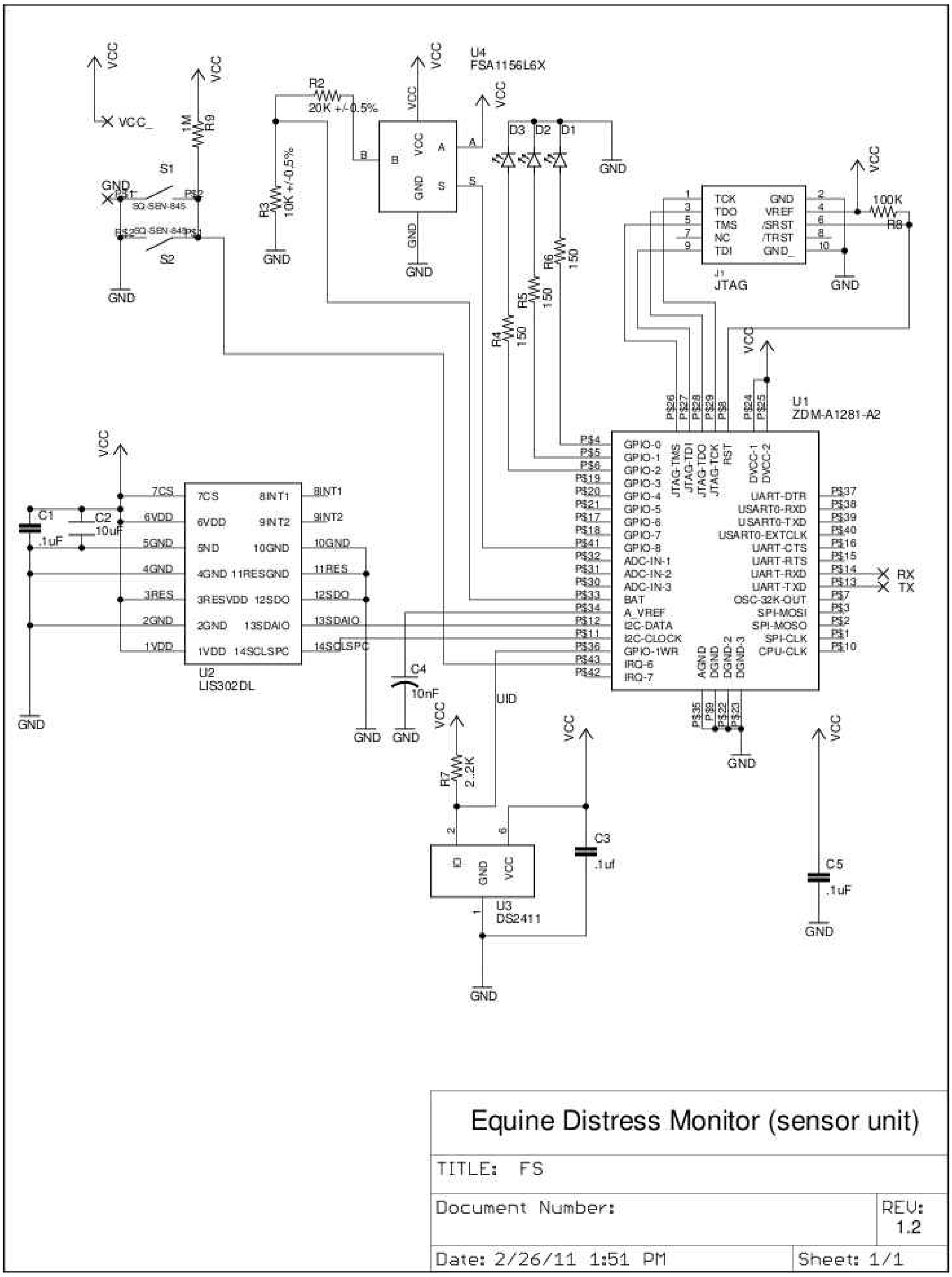


Fig. A.7: Sensor Schematic V1.3.

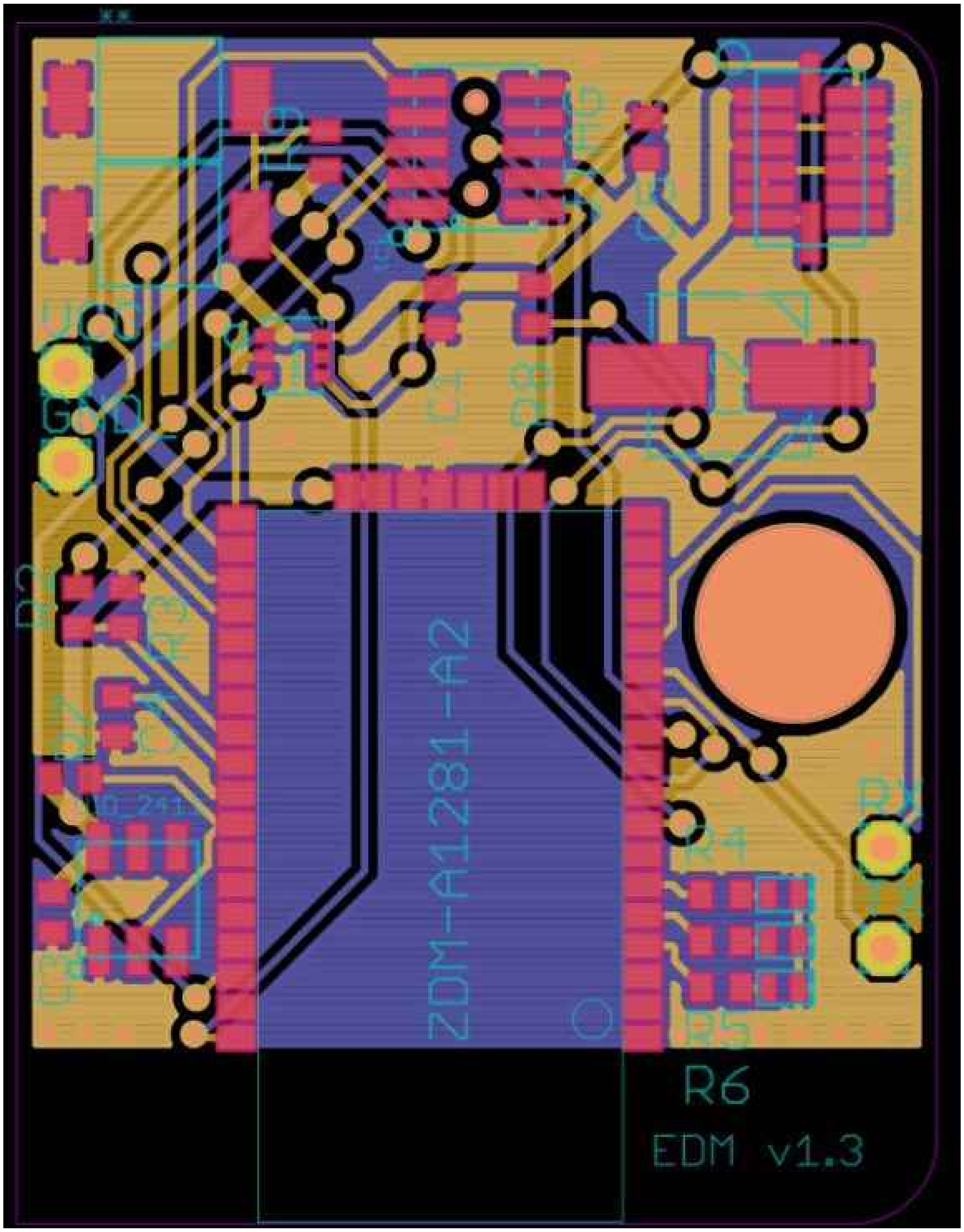


Fig. A.8: Sensor Layout V1.3.

Appendix B

Router Board Design Documents

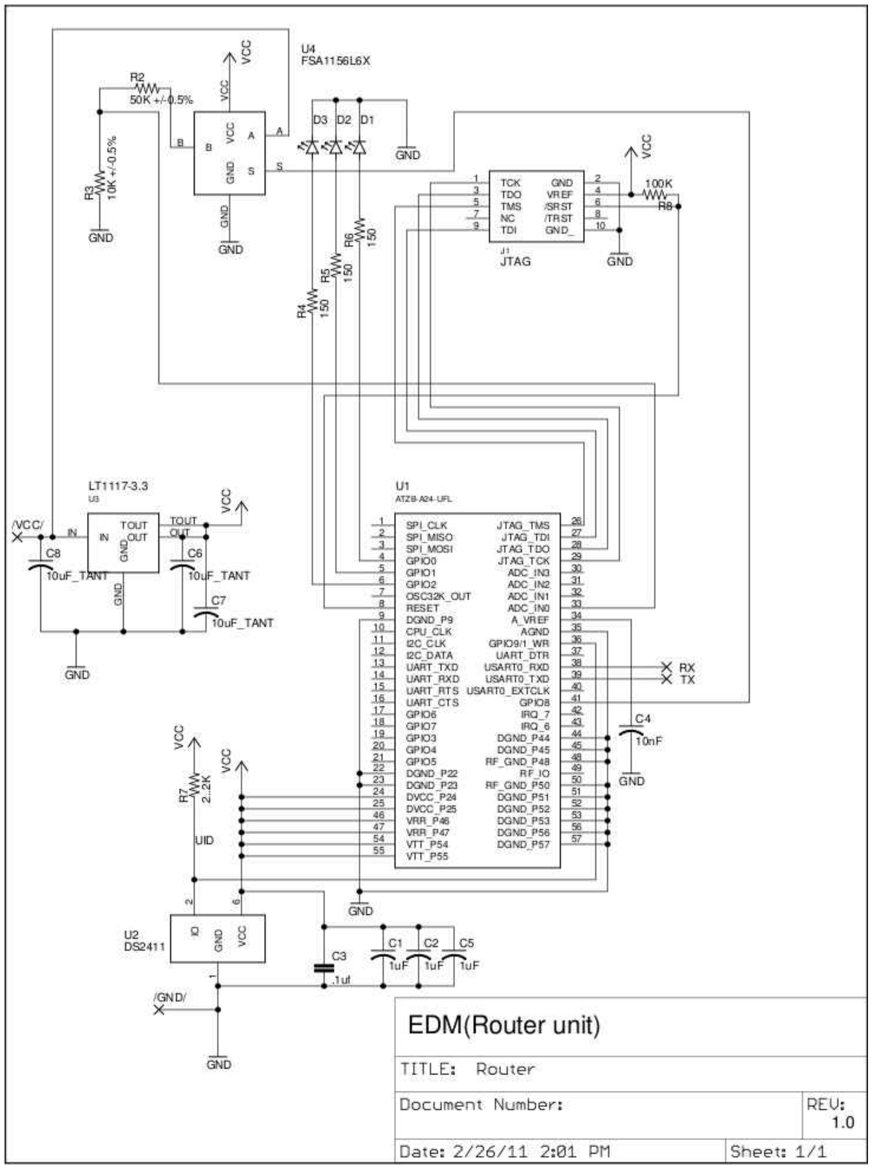


Fig. B.1: Router Schematic V1.0.

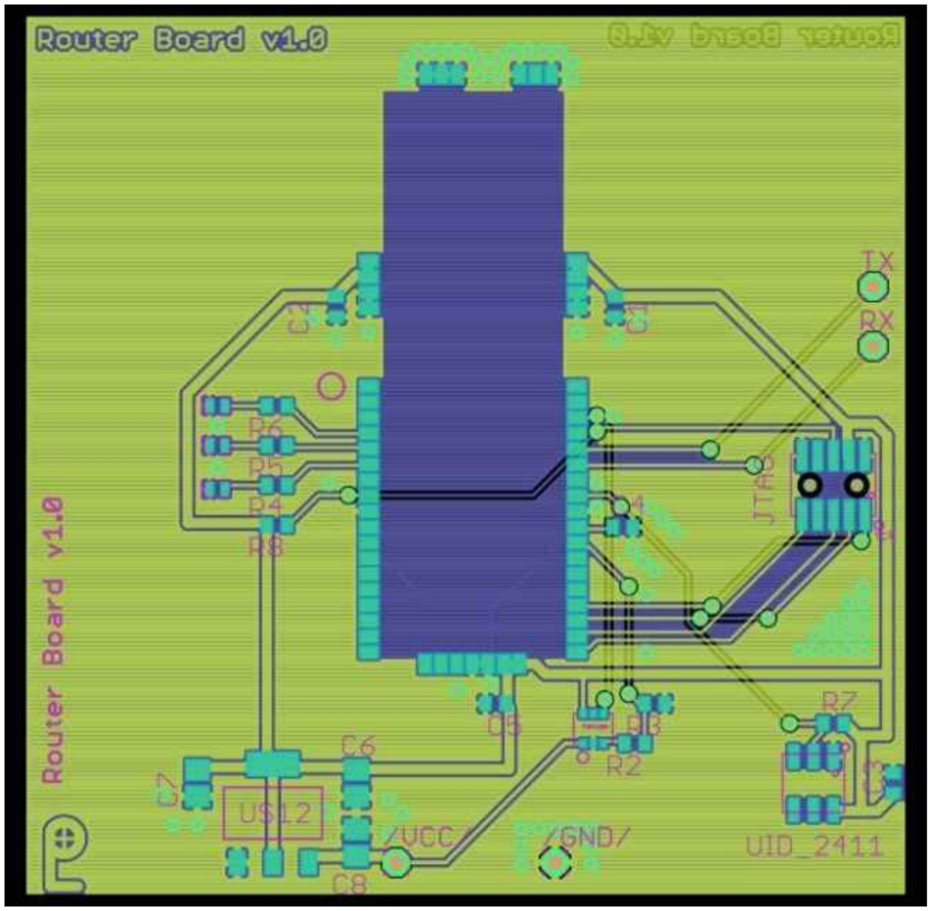


Fig. B.2: Router Layout V1.0.

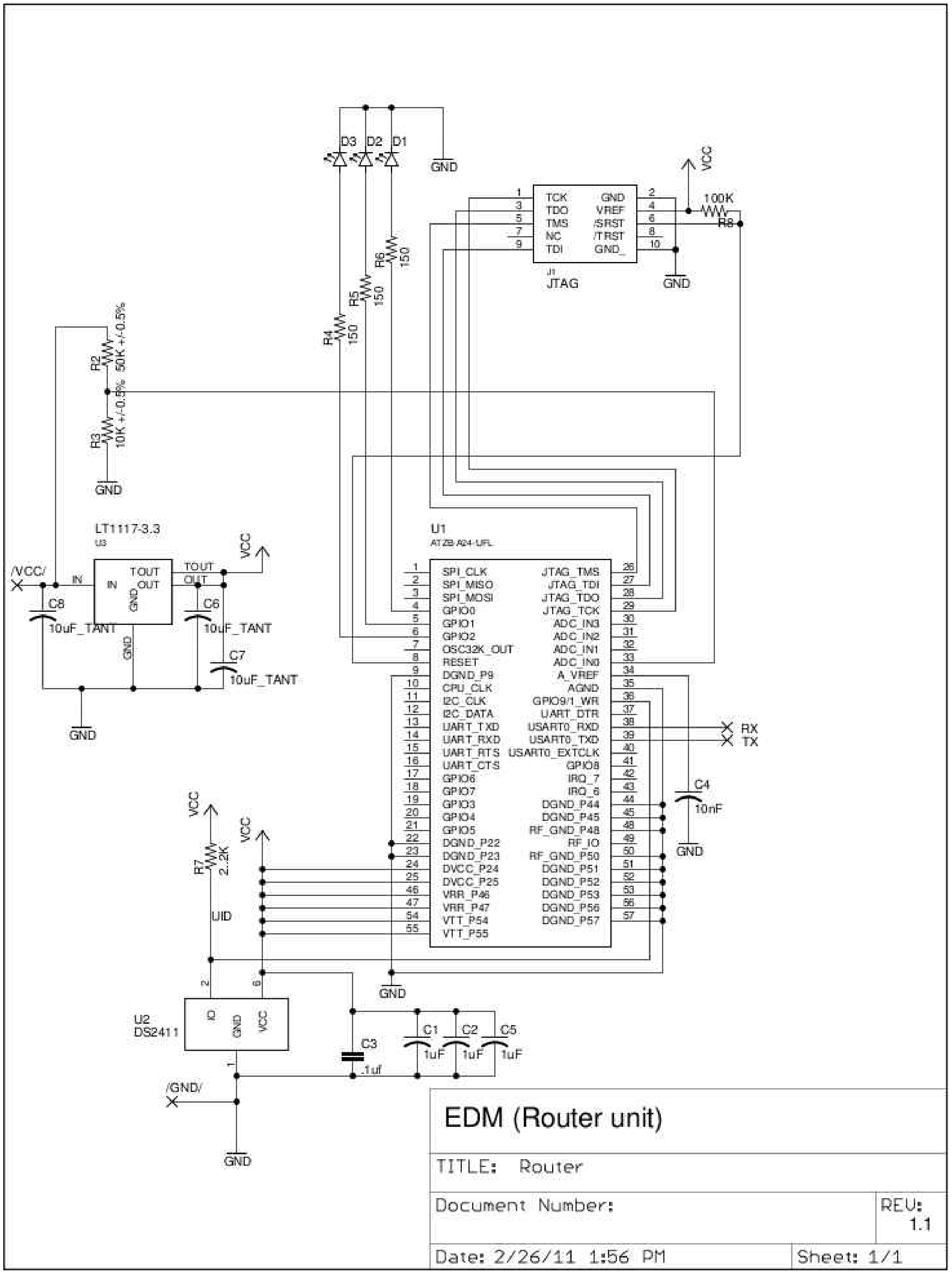

D3 D2 D1
GND
VCC
TCK TDO TMS NC TDI
GND VREF /SRST /TRST GND_
100K
R8
J1
JTAG
R4 150
R5 150
R6 150
R2 50K +/-0.5%
R3 10K +/-0.5%
LT1117-3.3
U3
/VCC/
IN
TOUT
OUT
C8 10uF_TANT
C6 10uF_TANT
C7 10uF_TANT
U1
ATZB-A24-UFL
SPI_CLK SPI_MISO SPI_MOSI GPIO0 GPIO1 GPIO2 OSC32K_OUT RESET DGND_P9 CPU_CLK I2C_CLK I2C_DATA UART_TXD UART_RXD UART_RTS UART_CTS GPIO6 GPIO7 GPIO3 GPIO4 GPIO5 DGND_P22 DGND_P23 DVCC_P24 DVCC_P25 VRR_P46 VRR_P47 VTT_P54 VTT_P55
JTAG_TMS JTAG_TDI JTAG_TDO JTAG_TCK ADC_IN3 ADC_IN2 ADC_IN1 ADC_IN0 A_VREF AGND GPIO9/1_WR UART_DTR USART0_RXD USART0_TXD USART0_EXTCLK GPIO8 IRQ_7 IRQ_6 DGND_P44 DGND_P45 RF_GND_P48 RF_IO RF_GND_P50 DGND_P51 DGND_P52 DGND_P53 DGND_P56 DGND_P57
RX
TX
C4 10nF
R7 2.2K
UID
U2
DS2411
IO GND VCC
C3 .1uf
C1 1uF
C2 1uF
C5 1uF
/GND/
EDM (Router unit)
TITLE: Router
Document Number:
REV: 1.1
Date: 2/26/11 1:56 PM
Sheet: 1/1

Fig. B.3: Router Schematic V1.1.

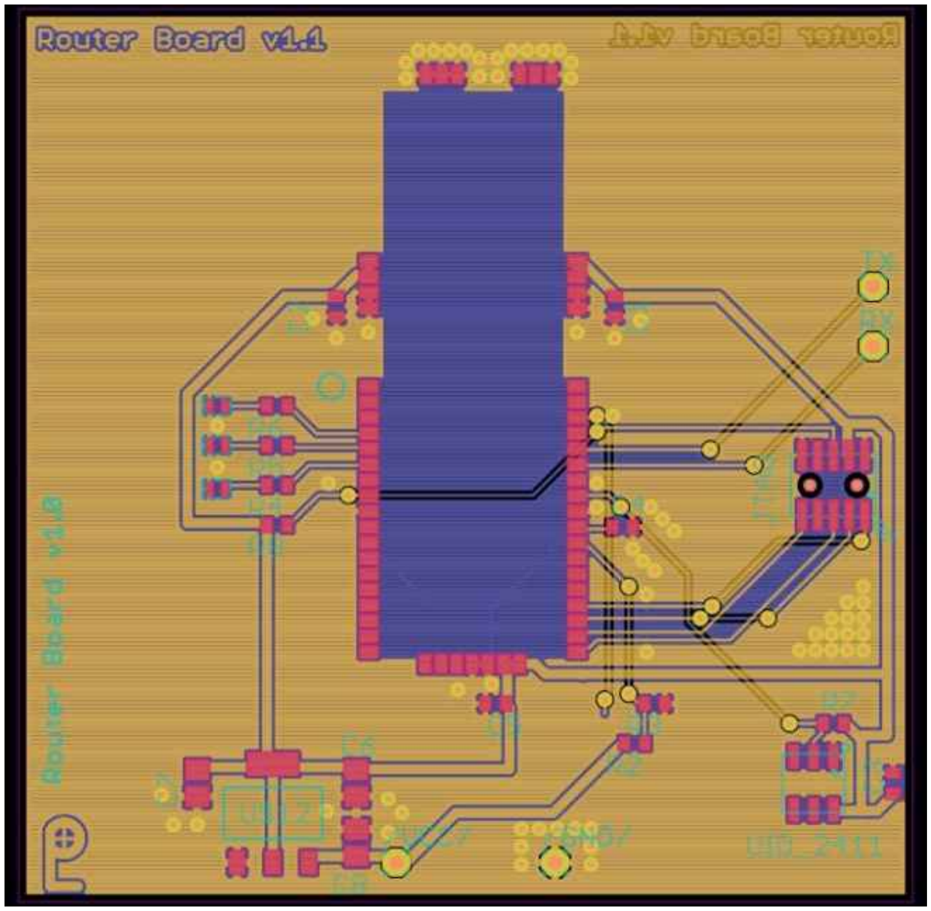


Fig. B.4: Router Layout V1.1.

Appendix C

Coordinator Board Design Documents

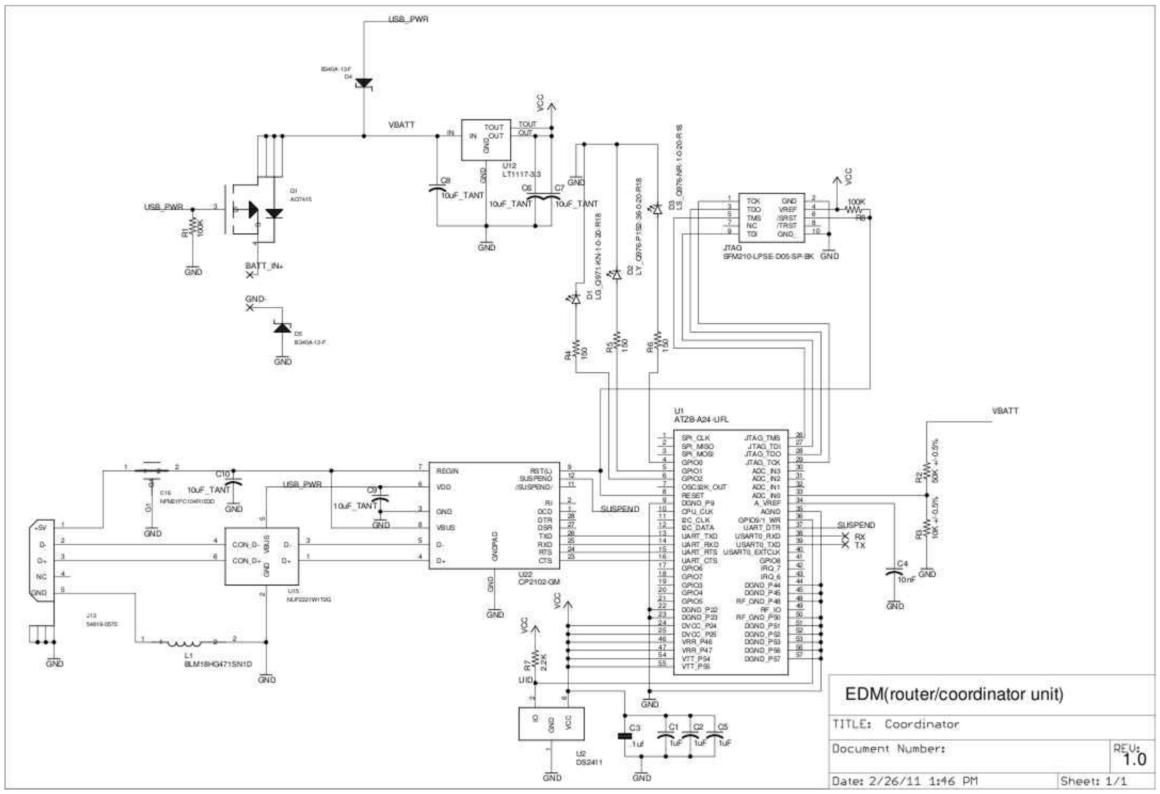


Fig. C.1: Coordinator Schematic V1.0.

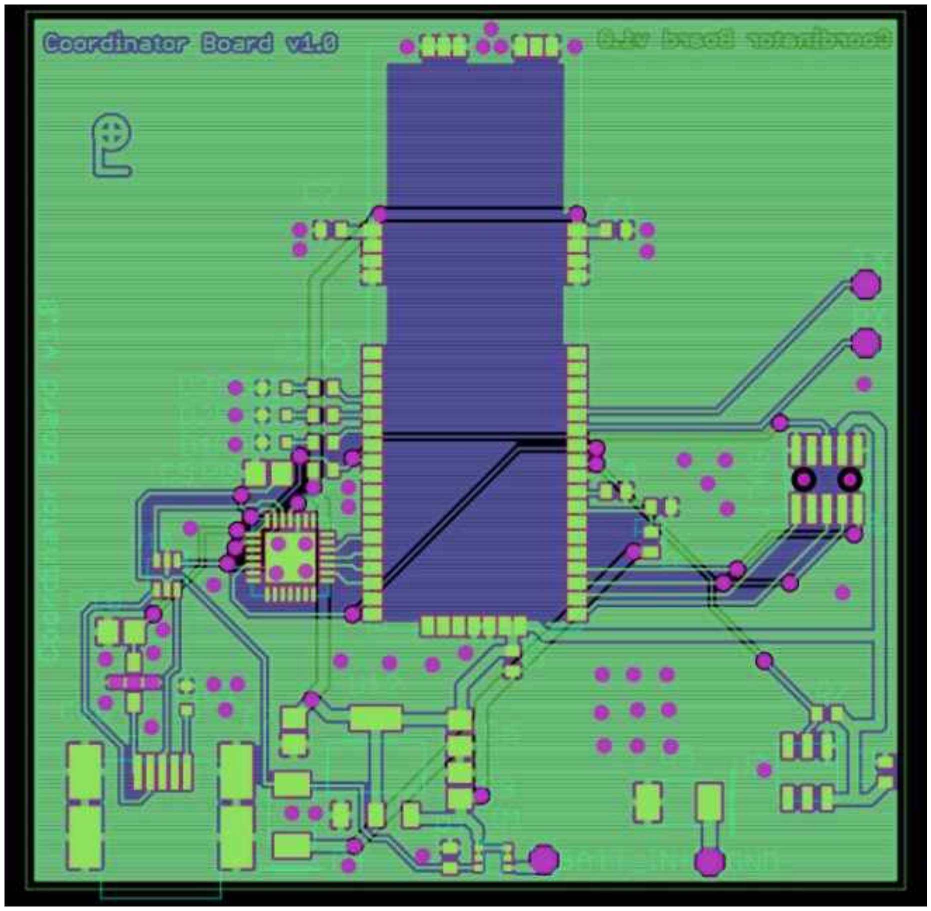


Fig. C.2: Coordinator Layout V1.0.

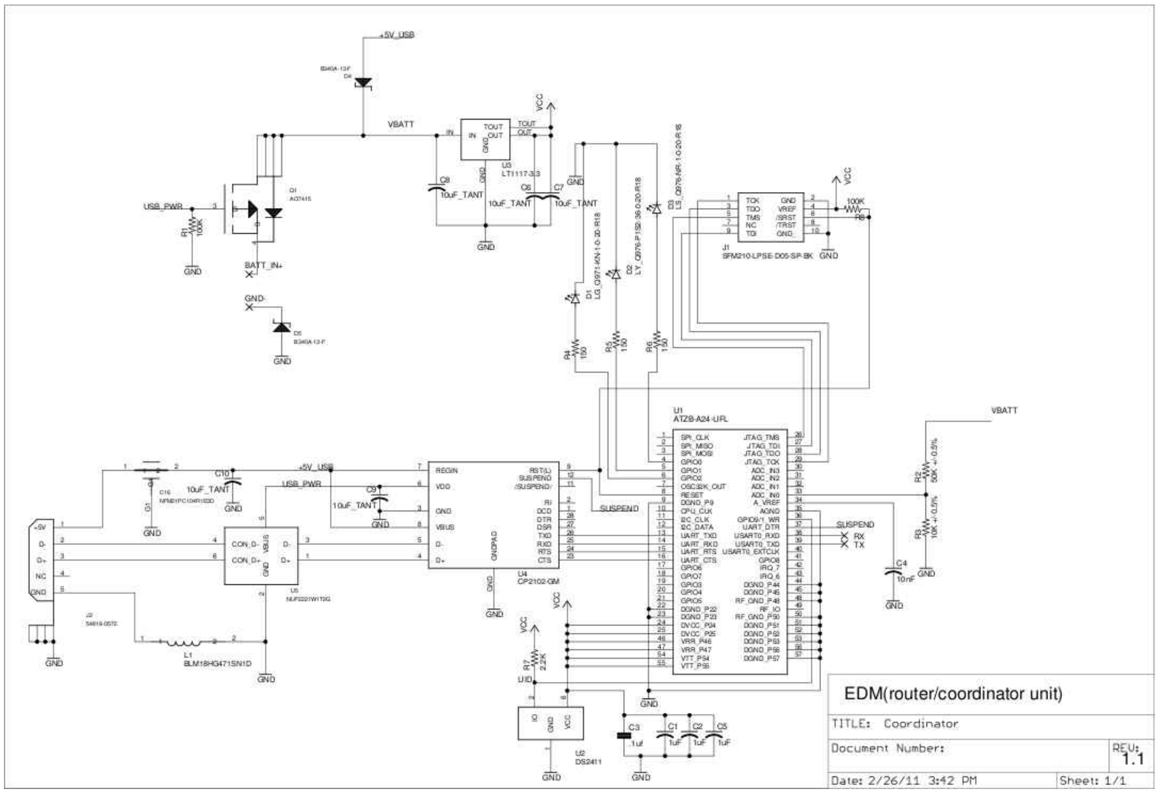


Fig. C.3: Coordinator Schematic V1.1.

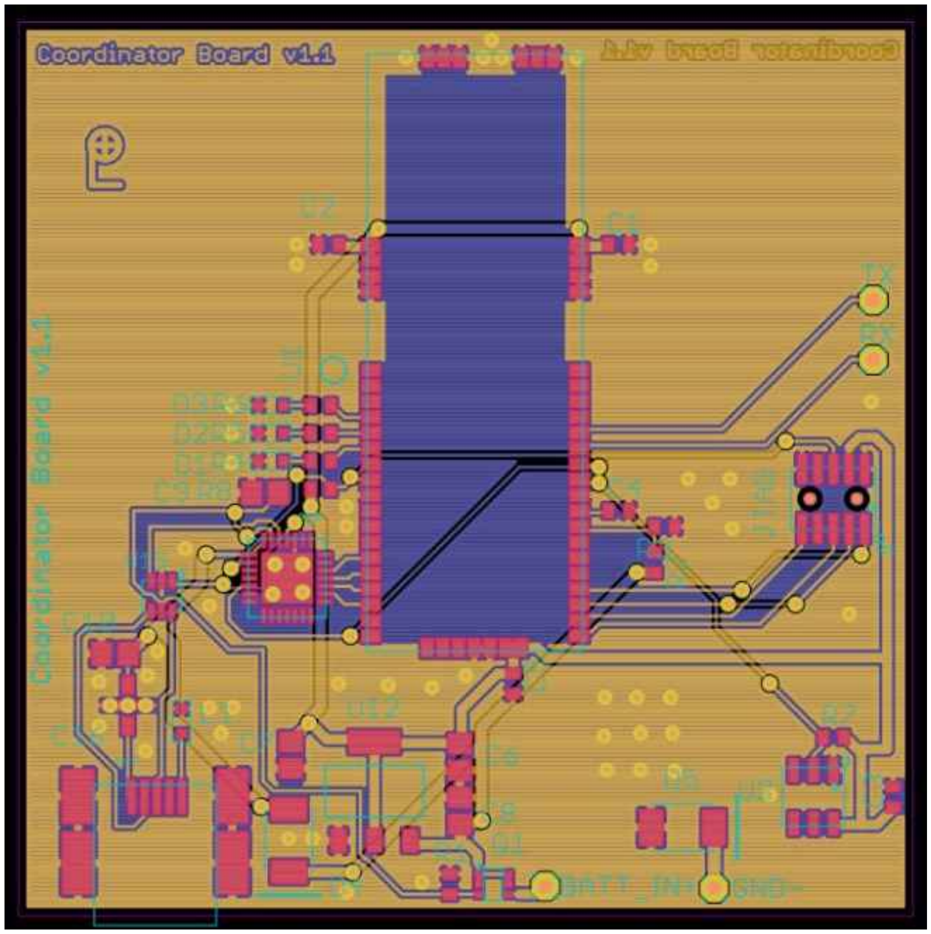


Fig. C.4: Coordinator Layout V1.1.