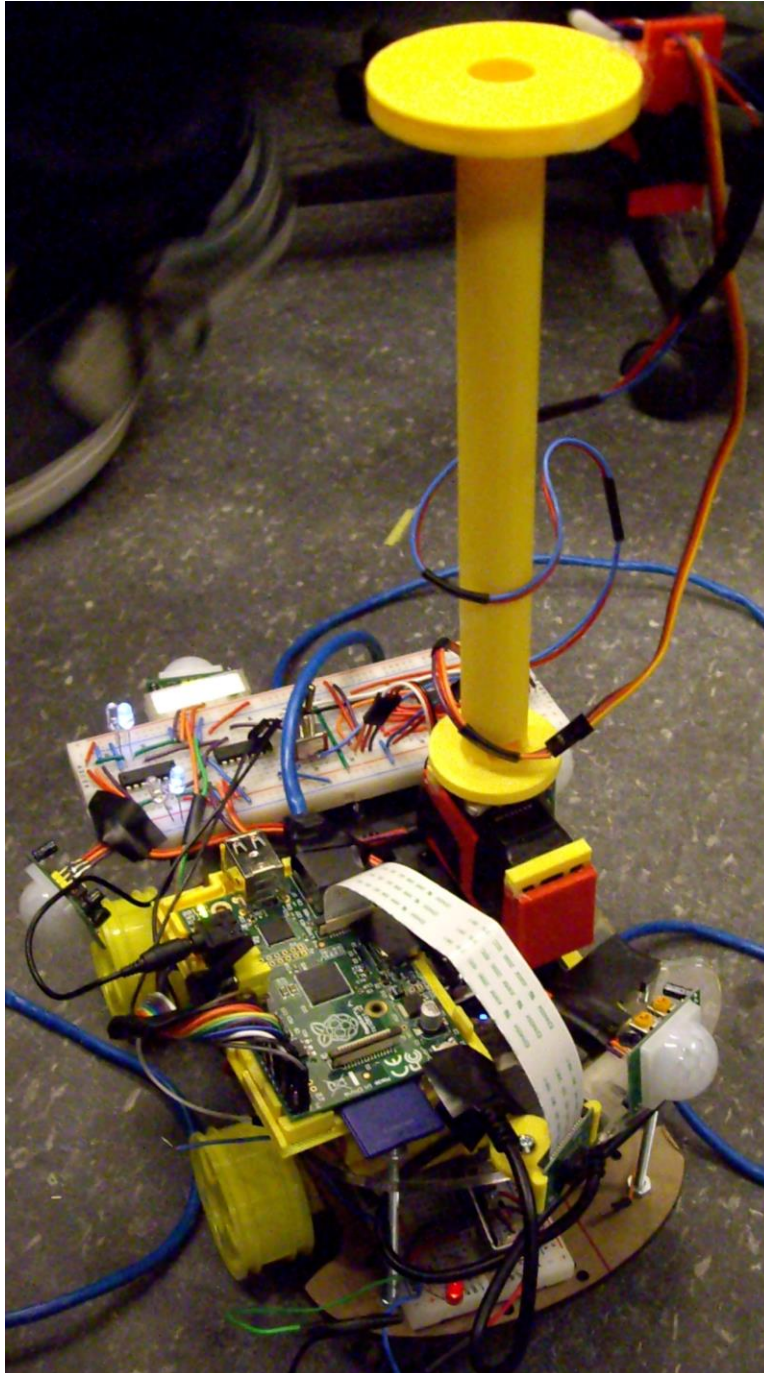


**EE 478 Capstone: Designing a Manual and Automatic Remote Cat
Exercise Device – CatErcise365
Serena Lam, Yanling He, Daniel Mueller**



Signatures:

Date:

Table of Contents

ABSTRACT	3
INTRODUCTION	3
DISCUSSION OF THE LAB	4
Requirements Specifications	4
Design Specifications	7
Design Procedure	13
System Description	15
Software Implementation	17
Hardware Implementation	25
TEST PLAN	27
TEST SPECIFICATION	27
TEST CASES	29
PRESENTATION, DISCUSSION, AND ANALYSIS OF THE RESULTS	31
ANALYSIS OF ANY ERRORS	34
ANALYSIS OF WHY THE PROJECT MAY NOT OF WORKED AND WHAT EFFORTS WERE MADE TO IDENTIFY THE ROOT CAUSE OF ANY PROBLEMS	35
SUMMARY	35
CONCLUSION	36
APPENDICES	36
CODE	39

ABSTRACT

The objective of this lab is to use skills learned from previous labs and classes to design an interesting and useful product to practice a full development life cycle of designing, building, and testing. For our lab we used the PIC18F25K22 Peripheral Interface Controller microprocessor as well as the Raspberry Pi in order to build a cat exercise machine that allows users to manually and automatically play and exercise their cats remotely, either from work or from wherever is needed. Our project had three main parts were implemented on the project, the movement/car portion which controlled the movement of the device mainly by the Raspberry Pi, another portion was laser control which was mainly controlled by the PIC18F25K22 controlling servos which moved a laser pointer in 360 degrees around the device as well as up and down to create an interesting pattern for the cat to chase, the last portion was the network portion which allows the user to have control of the device remotely and see the cat playing with the CatErcise365®. Other portions include the speaker which will make a sound when it turns on in order to let the cat know its time to play, there are also motion sensors implemented on the PIC18F25K22 which will turn on the laser when the cat walks by the device. The requirement specification describes and defines our system and design while the design specification is a more detailed version. The test plan, test specifications, and test cases are created to make sure that our implementation to the project is according to the project specifications. Our results were successful and worked accordingly to our design.

INTRODUCTION

For this final project we practice the full development life cycle of designing, building, and testing. We design a cat exercise machine with motion, laser, network, sound, video, motion sensor and command tasks. It allows users to manually and automatically play and exercise their cats remotely, either from work or from wherever is needed. We use the PIC18F25K22 Peripheral Interface Controller microprocessor as well as the Raspberry Pi as the main processor. Our project had three main parts were implemented on the project, the movement/car portion which controlled the movement of the device mainly by the Raspberry Pi. Another portion was laser control, which was mainly controlled by the PIC18F25K22 controlling one continuation rotation servo and one micro servo motor. This had the laser pointer rotating 360 degrees around the device as well as 180 degrees up and down to create an interesting pattern for the cat to chase. The last portion was the network portion which used CSS, PHP, HTML and JavaScript which allows the user to have control of the device remotely and see their cat playing with the CatErcise365® using a Raspberry Pi camera. Other portions include the speaker which will make a sound when the device turns on in order to let the cat know that it's time to play, this sound is currently the sound of food hitting the feeding bowl. Four Motion sensors are also used and implemented on the PIC18F25K22 which will detect movement and turn on the laser for the cat to play with as long as motion is detected. The tools used are, the Linux operating system for Raspberry PI, PI4J Java library to control Raspberry pi GPIO peripherals, Java, C, CSS, JavaScript, MPLAB X IDE V1.85, and the Pick it 3.

DISCUSSION OF THE LAB

Requirements Specifications

System Description

This specification describes and defines the basic requirements for an interactive laser pet toy. The system is able to allow the owner to see their pets through a video feed and be able to interact with their pet by controlling the system. The device will have a default setting to interact with the pet if the owner is out using the motion sensor detection to start the laser automatically. The device can be remotely controlled by the owner through a website.

Specification of External Environment

The system is to operate in household environments and can be controlled from all over the world. The unit is to be portable and battery operated.

System Input and Output Specifications

System Inputs

The system shall support the following types:

Camera – 5 Mega Pixel native resolution images and 640x480p60/90 video

Power – ON/OFF button controls power input

Commands from the Internet:

Video On/Off - turns the camera on or off.

Laser Off - turns the laser signal off.

Laser Pattern selection – the order to choose the laser movement pattern (wall, floor and mix) that preprogrammed in the system using servomotors.

Base Movement Direction – four directions inputs to direct where the device will go through the Internet.

Base Movement Speed Level – three speed levels (fastest, median and slowest) for the base movement

Speak – the order to output the feed food sound effect

Motion Sensor – movements around the system

System Outputs

Laser – laser signal to the pet based on the pattern selection.

Base Movement – directions and speed levels of the base movements.

Sound – feed food sound effect to the pet

Website Display:

Video – the real time video showing the pet's situation through network.

Control Panel – the webpage with the control buttons

The webpage is shown below:

CatErcise365

[Control Panel](#)

Cat Video



Figure 1 Video Webpage

The website control panel layout is shown below:

CatErcise365 Control Panel

Control buttons:



Camera:

ON

OFF



Laser:



Wall

Floor

Mix

OFF

Speed:



Figure 2 Control Panel Webpage

User Interface

The client is able to control the system locally using the buttons on the system and control the system remotely using the website through the Internet. The user can receive information from the system via the video displayed on the website.

The user shall be able to select the following using switches on the device

Power: On or Off

Motion Detection: On or Off

The user shall be able to controls the device using the buttons on the website. The user interface design graph is shown above as the control panel webpage.

Camera On/Off

Laser Patterns: Wall/Floor/Mix/OFF

Base Movement Directions: Up/Down/Left/Right

Base Movement Speed Levels: 1/2/3

Speaker: speak on

The user shall be able to see the video display shown on the website taken by the device camera.

System Functional Specification

The CatErcise365 is a small toy that will interact with a pet. This will allow the user to be able to play with their pet at work or at home. While the owner is at work, they will be able to access a website to input orders to manually control the system. If the owner is at home, the system can be manually turned on or off. The system supports remote controlling through the Internet. The system will then take a video and transmit the video feed through the internet to appear on a website the owner can access. The owner will then be allowed to use the video feed on the website to control the device to move throughout their home to locate their pet. The owner can also turn on and off the laser and input programmed actions to interact with their pet. The system can output feed food sound effect to attract the pet. The device has motions sensors that can detect the movements around it, once the detection is on and detects the movement the laser will be on automatically. The device requires four main hardware parts that are the laser, moving base, camera and motion sensor. The system also contains three software parts that are the website, command controlling and the laser signal controlling. The microcontroller that will be the main system is the Raspberry Pi. In the device, the Raspberry Pi would be located on a platform on top of the wheels of the device and the laser will be implemented a little higher on the device.

The device will be able to detect if the pet is near and will try to initiate action with the pet. The owner will also be able to turn on the device manually. The system supports remote controlling through Internet. The system can take the video and transmit the video to the Internet.

Memory Specifications

The system will only be able to display and hold default of the device settings. Device setting may be rewritten. The system memory should support taking the video and transmit the video to the Internet.

Display Specifications

The display will be on a website. The video will be displayed on the video website and control buttons are displayed on the control panel webpage.

Operating Specifications

The system is battery powered and should be replaced or recharged when dead. System will be used in a household environment. System will be powered on or off manually and automatically with proximity of the pet. Power supply shall supply 5 Volts to the system continuously. Network device should be in order to connect to the Internet and controlled by the humans through the Internet. Speaker shall be turned on manually if the users want to output sound effect.

Reliability and Safety Specification

The system shall supply the safe voltage range to the system. The voltage shall be safe for each parts of the system and safe for the user.

The system shall be able to measure the accurate data. The frequency of the clock system shall be within the operating range of all parts.

The system shall be also pet safe such as the laser will not be harmful the eye for humans and pets.

The system shall protect the security of the network connection by checking user name and password.

Design Specifications

System Description

This specification describes and defines the basic requirements for an interactive laser toy for a pet called CatErcise365. The hardware of the system contains five major parts, the base that includes the wheels to support the movement of the toy, the laser signal that the cat can play with, the camera that can take video of the cat during the remote control and sending the video through the Internet, the motion sensors that detects the movements around the system to enable the laser automatically and the speaker that output sound effect to attract the pet. The software of the system contains three major parts, a website where the user can see the video and inputs the commands to the toy, a command controlling system that can send live video to the user through Wi-Fi and accepting commands from a website and a laser signal controlling system that contains the default patterns of the laser movements.

The system is able to allow the owner to see their pets through a video feed and be able to interact with their pet by controlling base motion of the system and the select the laser patterns. The device will have a default setting to interact with the pet if the owner is out or can be remotely controlled by the owner through a website. The device can be turn on with motion detection mode that can automatically turns on and output the laser signals when it detect any movements around the system.

Specification of External Environment

The system is to operate in household environments and can be controlled from all over the world. The unit is to be portable and battery operated. The unit should be kept away from water and fire. The video function should be used in the daytime in order to take the video.

System Input and Output Specifications

System Inputs

The system shall support the following input types:

System Inputs

The system shall support the following types:

Camera – 5 Mega Pixel native resolution images and 640x480p60/90 video

Power – ON/OFF button controls power input

Commands from the Internet:

Video On/Off - turns the camera on or off.

Laser Off - turns the laser signal off.

Laser Pattern selection – the order to choose the laser movement pattern (wall, floor and mix) that preprogrammed in the system using servomotors.

Base Movement Direction – four directions inputs to direct where the device will go through the Internet.

Base Movement Speed Level – three speed levels (fastest, median and slowest) for the base movement

Speak – the order to output the feed food sound effect

Motion Sensor – movements around the system

System Outputs

Laser – laser signal to the pet based on the pattern selection.

Base Movement – directions and speed levels of the base movements.

Sound – feed food sound effect to the pet

Website Display:

Video – the real time video showing the pet's situation through network.

Control Panel – the webpage with the control buttons

The video webpage is shown in Figure 1.

The website layout is shown in Figure 2.

User Design Specification

User Interface

The client is able to control the system locally using the buttons on the system and control the system remotely using the website through the Internet. The user can receive information from the system via the video displayed on the website.

The user shall be able to select the following using switches on the device

Power: On or Off

Motion Detection: On or Off

The user shall be able to controls the device using the buttons on the website. The user interface design graph is shown above as the control panel webpage.

Camera On/Off

Laser Patterns: Wall/Floor/Mix/OFF

Base Movement Directions: Up/Down/Left/Right

Base Movement Speed Levels: 1/2/3

Speaker: speak on

The user shall be able to see the video display shown on the website taken by the device camera. The website display is shown in Figure 1 and Figure 2.

Use Cases

The use cases for the CatErcise365® are given in the following diagram.

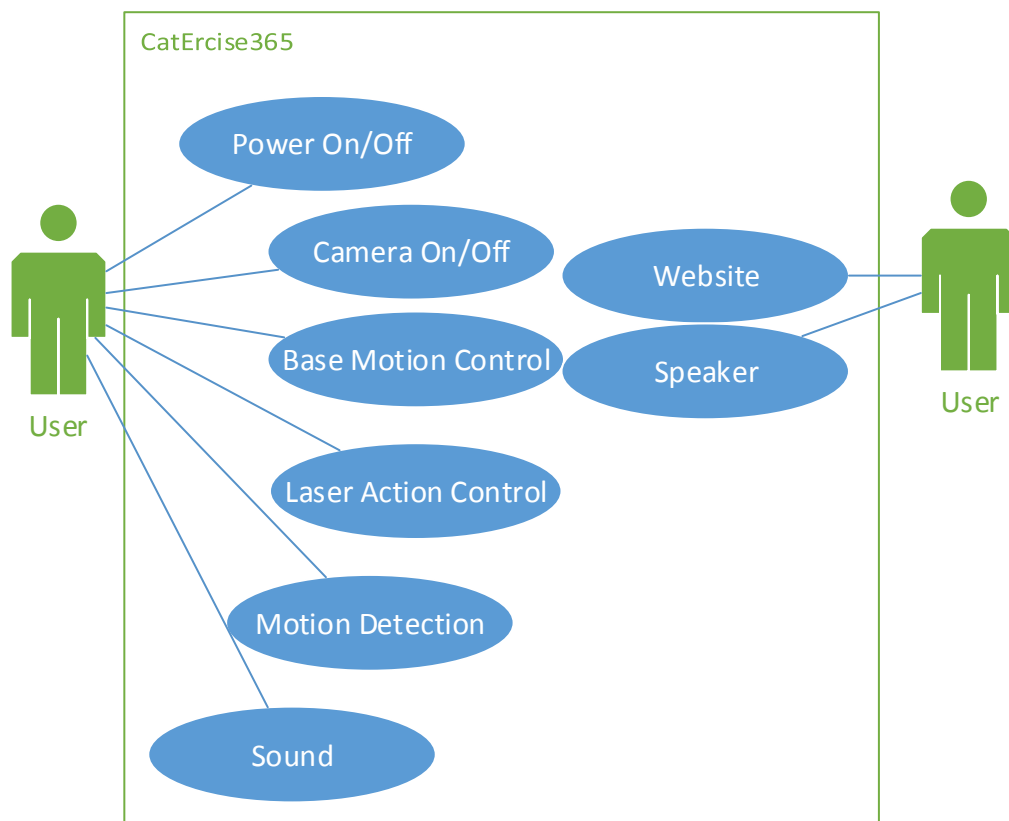


Figure 3 Use Case Diagram

User – System

- Power On/Off
 - The user controls the on/off of the device using the button on the device.
 - Exception – system is broken, button is broken, battery is dead.
- Motion Detection On/Off
 - The user controls the on/off of the motion detection of the device using the button on the device.

- Exception – system is broken, button is broken, motion sensor is broken.
- Camera On/Off
 - The user controls the on/off of the camera on the device through the Internet.
 - Exception – system is broken, system is not connected to Internet, the camera is broken.
- Base Motion Control
 - The user controls the directions (forward, backward, left and right) and speed levels (fastest, median and slowest) of the device movements through the Internet.
 - Exception – system is not connected to internet, wheels broken, system is broken.
- Laser Action Control
 - The user can turn the laser off.
 - The user can select the output laser patterns (wall, floor and mix) that preprogrammed in the system.
 - Exception - system is broken, system is not connected to the Internet, laser is broken, the system is not powered.
- Sound Control
 - The user can turn on the sound effect through the Internet, which output sound effect to the pet.
 - Exception - system is broken, system is not connected to the Internet, laser is broken, speaker is broken, the system is not powered.

System – User

- Website Display
 - The video shows on the website. The video website is shown in Figure 1.
 - The control buttons shown on the website. The control panel website is shown in Figure 2.
 - Exceptions - system is broken, system is not connected to the Internet, system is not powered, the camera is broken, monitor is broken.

System Functional Specification

The CatErcise365 is a small toy that will interact with a pet. This will allow the user to be able to play with their pet at work or at home. While the owner is at work, they will be able to access a website to input orders to manually control the system. If the owner is at home, the system can be manually turned on or off. The system supports remote controlling through the Internet. The system will then take a video and transmit the video feed through the Internet to appear on a website the owner can access. The owner will then be allowed to use the video feed on the website to control the device to move throughout their home to locate their pet. The owner can also turn on and off the laser and input programmed actions to interact with their pet. The system can output feed food sound effect to attract the pet. The device has motions sensors that can

detect the movements around it, once the detection is on and detects the movement the laser will be on automatically. The device requires four main hardware parts that are the laser, moving base, camera and motion sensor. The system also contains three software parts that are the website, command controlling and the laser signal controlling. The microcontroller that will be the main system is the Raspberry Pi. In the device, the Raspberry Pi would be located on a platform on top of the wheels of the device and the laser will be implemented a little higher on the device.

The device will be able to detect if the pet is near and will try to initiate action with the pet. The owner will also be able to turn on the device manually. The system supports remote controlling through Internet. The system can take the video and transmit the video to the Internet.

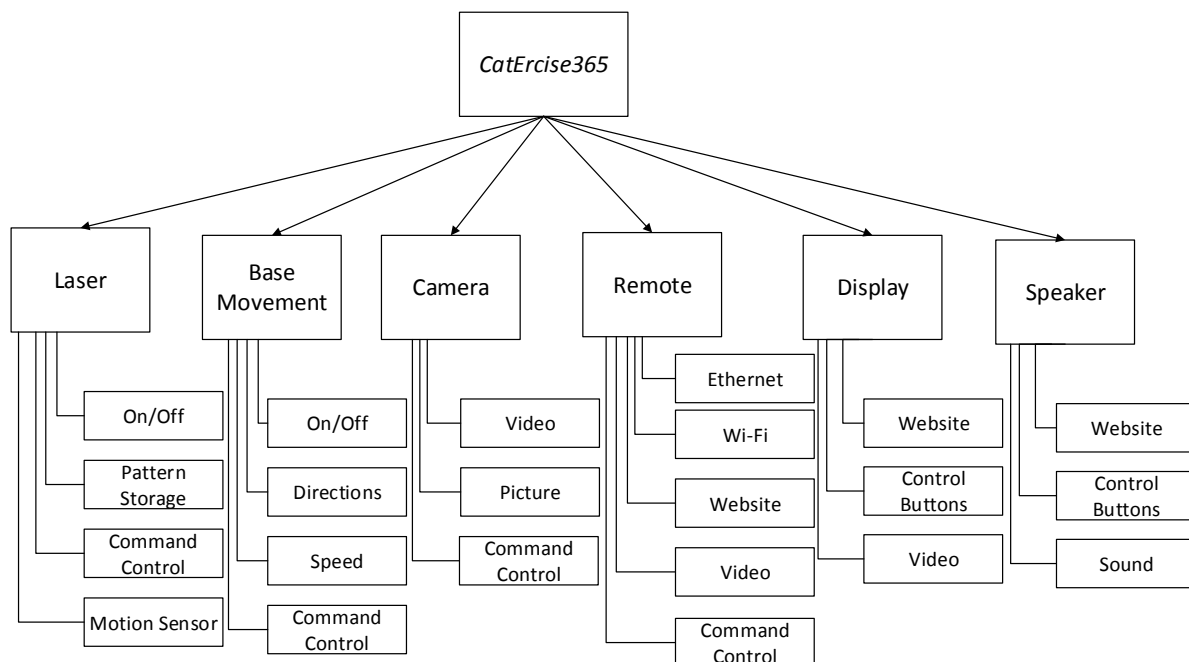


Figure 4 Functional Decomposition

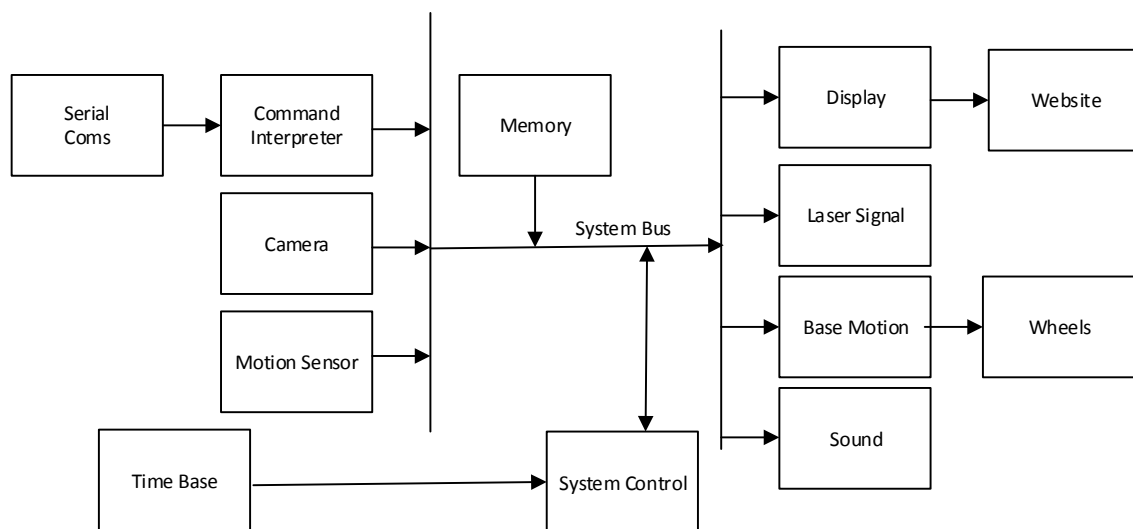


Figure 5 Block Diagram

• **Laser Subsystem** – Laser subsystem takes remote controlling commands from the network including off and pattern selections (wall, floor and mix) as the input and output the laser signal based on the pattern selection. On hardware part, two servomotors will control the laser actions. The servomotors allow the laser to be horizontally and vertically. The two main ideas of the programmed functions are to have the laser move on walls or on the floor. The servomotors will be controlled using pulse width modulators. The pulse width modulators will be created by using a PIC18F25K22 microcontroller. On software part, the action patterns are programmed and stored on a PIC and command input is transmit from the command task.

• **Base Movement Subsystem** – Base movement subsystem takes base motion directions and speed levels from the Internet as input and output the signals to the wheels to control the basement movement. The base movements are using motors and the Raspberry Pi. An H bridge circuit will be used to allow the wheels to be powered to move while the Raspberry Pi is used to control the direction the system should move towards.

• **Camera Subsystem** – Camera subsystem accept the camera on/off from the network and takes the video using camera and transmit the video data to the Internet. The camera is on the Raspberry Pi.

• **Sound Subsystem** – Sound subsystem accept the sound on from the network and outputs the feed food sound effect to the pet. The speaker is external speaker that connected to the Raspberry Pi.

• **Remote Subsystem** – Remote subsystem uses network to connect the website and the system. To be able to connect wirelessly, a Wi-Fi dongle was inputted in the Raspberry Pi. This will allow the control to be implemented wirelessly form the website to the device. The system sends the video to the client and takes the command inputs from the client. Some of the controls that would be implemented are turning on and off the system, changing the patterns of the laser output, and the direction and speed the device should travel. The video feed is assumed not be used in the darkness, allowing the pet to sleep and not receive an adrenaline rush that may keep the neighbors or roommates up.

• **Display Subsystem** – The display subsystem contains the website that displays the controlling buttons on the front panel and display the video to the user.

• **Motion Sensor Subsystem** – The motion sensor subsystem contains four motion sensors on the front, back, left and right of the device and has a on/off button to control the motion detection function on and off. It output motion-detected signal to the laser task if any movement detected and enable the laser.

Memory Specifications

The system will only be able to display and hold default of the device settings. Device setting may be rewritten. The system memory should support taking the video and transmit the video to the Internet.

Display Specifications

The display will be on a website. The video will be displayed on the video website and control buttons are displayed on the control panel webpage.

Operating Specifications

The system is battery powered and should be replaced or recharged when dead. System will be used in a household environment. System will be powered on or off manually and automatically with proximity of the pet. Power supply shall supply 5 Volts to the system continuously. Network device should be in order to connect to the Internet and controlled by the humans through the Internet. Speaker shall be turned on manually if the users want to output sound effect.

Reliability and Safety Specification

The system shall supply the safe voltage range to the system. The voltage shall be safe for each parts of the system and safe for the user.

The system shall be able to measure the accurate data. The frequency of the clock system shall be within the operating range of all parts.

The system shall be also pet safe such as the laser will not be harmful the eye for humans and pets.

The system shall protect the security of the network connection by checking user name and password.

Design Procedure

The design of our project started with the idea to make a remote cat exercise machine. Through this thought process we decided the best way was for the cat to chase a laser toy in a circular motion around the device. We also thought it would be good to control this from home so we made sure the device was capable of maintaining a server on the internet for control and that a camera would also need necessary to control the device. Of course, we needed to find the cat so it made sense to make the device mobile. These were the three original parts to our project that we split up between the group members which are network, motion, and laser. The ideas and actual design implementation were added and altered, as well as disregarded along the way and as the team got closer to demo, but the main idea behind the project prevailed, and eventually, the vision led to a successful device. Phase one consisted of these main ideas, and phase two of the design process would only be realized if the team had enough time, the team ended up finishing and adding both a speaker and audio signal to attract the cat, as well as motion sensors to let the device know when the cat was around

The network was implemented on the Raspberry Pi and controlled the on and off state of the camera as well, the website was to be designed to send commands to the device from any location around the world that had access to the internet. The website was written in HTML and CSS and what the group decided to do was update a .txt file with the button commands of the html website, this .txt file would then be read by the java code on the raspberry pi to control the

device. This worked well and the website also contained a video feed from the device, but if the team were to go back, they would like to have a way to update the website from the device itself, such as let the website know if the cat has tripped motion sensors.

The design for moving the device around the room did not change much over time. The original intent was to have four wheels and to move the car/device left right and back and forward. The control for this part took six pins, three to the digital to analog converter which used serial data communication, so it needed a clock, a chip select/enable as well as the data line to input a 10 bit level, as for the other three pins, one pin enabled and disabled the H-Bridge, two pins controlled forward and backward for left and right side of the car. The general concept for this design never changed. However, throughout the design and implementation procedure, there were many problems with the hardware design itself. Initial design spec was intended to use a digital to analog converter as well as an H bridge to drive the motors, which turned out to be four DC motors, not two as initially planned. This meant more power was needed to drive all four motors, because of this, a seven volt source was needed to start given the fact that some voltage is lost through each step of the way until reaching the H-Bridge such that it fell about one and a half to two volts down to five volts (which, for the original design idea would have been three volts) given that about four volts is necessary to drive the motors, and given the fact that the design called for three separate speeds, it was very necessary for the voltage to start at seven volts, and then have different voltages for different speeds for example 4.1V, 4.6V and 5.3 Volts. This ended up working as a proper driver for the motors and all four wheels ran forwards and backwards during these speeds. Also, our design called for small bursts of speed, meaning going forward at increments of two feet and turning at increments of 30 degrees at a time, this ended up being an issue because it was not enough time to get the motors running, and for this reason the bursts needed to be longer in order to start the motors, which act as inductors. It also became necessary to take the tires off of the wheels of the car, because this allowed for less torque on the wheels, this was because once the car was set down on the ground and had all its weight on it, the tires did not move for the lower two speeds. It also became obvious that a static turn (turning the car without moving forward first) was not the easiest thing to do for the car, so changing the code to have the car move forward first, then turn was the better move for the movement portion of the car. Our final design consisted of car that turned left and right, went forward and back, had three different speeds, (4.1V, 4.6V and 5.3 Volts to the H-Bridge) and no longer maintained the tires on its wheels.

The laser portion of the Lab consisted of two pins from the Raspberry Pi to control the state of the laser which are off, wall mode, floor mode and mix mode (which did both at once). These two pins went to the PIC18F25K22 (PIC) and the PIC took over from there to drive the pulse width modulation (PWM) of the servos that controlled the laser.

These three parts consisted of our first stage of our design, and since we finished with time left we were able to implement the second stage which consisted of the audio speaker which plugged into our Raspberry Pi auxiliary output and the motion sensor which outputted to LED's as well as back to the PIC to let it know the cat was near. We would have liked to send this signal all the way to the website to let the owner of the pet know that their pet is playing with the device, but unfortunately this was not possible with our website design. However, the laser is turned on

when the motion sensors detects movement, and the owner can turn on this capability with the flip of a switch just as he or she walks out the door to work.

System Description

The system is a remote cat toy and exercise machine, it even works autonomously with motion sensors. The design consists of two main microcontrollers which are the Raspberry Pi and the PIC18F25K22. The Raspberry Pi controls the network module, the sound module, the camera module, and the car movement module directly. It also sends state commands to the PIC18F25K22 (PIC) which controls the laser module and the motion sensor module. All of these modules consist of software as well as hardware, but the hardware that the team designed and did not buy was the power supply and voltage regulation, which was vital for the car motion module, laser module and motion sensor module, these modules had a vast amount of hardware built directly by the team. Six general purpose Input/output (GPIO) pins controlled the speed and direction of the car while two pins were sent to the PIC which controlled the laser and its servos using PWM (Pulse width modulation) as well as input from the motion sensor's detection to decide whether to turn on or off the laser when in motion detection mode (controlled manually). The biggest software module was the network module which did not contain hardware the team build themselves, it consisted of the website and interacting with that website based on commands from the website the device would react as such, the video feed was also connected with this module and the camera hardware was also bought and not built by the team. The sound module was a .WAV file sent to a store bought speaker and connected via auxiliary on the Raspberry Pi.

System Input:

- Motion sensor switch: switch for enabling this mode
- Motion sensors (4): The system intakes whether there is movement next to the system or not (note that the system needs to know whether or not it is moving given that this will also set off the sensors)
- website motion commands – The user controls where the device moves (up, down, left, and right)
- website Speed commands – The user controls the speed the device moves (slowest, median, and fastest)
- website laser commands – the user controls what mode the laser module is in (off, floor, wall, or mix)
- website sound commands - the user controls the speaker is on
- Power control: Manual switch on device powers it down
- Website camera command – on or off
- Camera: 5 Mega Pixel native resolution images

System Outputs:

- Website commands: Allows the user to see the commands for motion, speed, laser and camera
- Camera feed to website: Allows user to see their cat from the point of view of their device.
- Laser: System outputs an interactive laser for pet
- Base movement: Device will move around based on input commands
- Sound: feed food sound effect from the speaker

Side Effects:

- Wi-Fi causes the network slow on the Raspberry Pi
- Cat can scratch and destroy wiring to the laser as well as other parts, need some protection cover

Pseudo Algorithm, Function:

- Network: network takes data from server side and display to the user, and the user receive the data and send the data to the server by clicking the button on the website. Received data saved in the control.txt file.
- Command: command task read the data from control.txt file and parse the data
- Video: video task receive the command data from the command task and get data from the camera, then send the video to the network
- Sound: sound task receive the command data from the command task and output the sound to the pet
- Motion: If left flag turned on (on meaning “true”), turn left. If right flag turned on turn right, if forward flag turned on, move forward, if backward flag turned on, move backwards. Otherwise do nothing. If speed is enabled move into speed section: If fast mode flag is turned on, set analog to digital converter (ADC) to highest voltage, if medium mode is turned on set ADC to slightly lower than highest voltage, if slow mode is turned on set ADC to lower voltage (keep high enough that enough power goes to wheels to move forward with full device weight). Otherwise do nothing
- Laser: The laser takes a two bit input to determine which pattern should be outputted from the Raspberry Pi to the PIC. If wall pattern is chosen, the laser will be outputted, creating laser movements on a wall. If the flag pattern is chosen, the laser will be outputted, circling the device on the floor. If the laser pattern is mix, the output is a mix of the wall and floor pattern. If the pattern chosen is off, the laser is not outputted. Else do nothing.

Timing Constraints:

- The nature of our project did not need absolute timing
- The thing that slowed commands the most was over wifi, and even Ethernet cable would lag at certain points

- Video feed was delayed by five to ten seconds, this made it more difficult to drive the car, but would not be especially noticeable to a user when looking at their cat from work.

Error Handling:

- If not enough power is supplied to base movement motors, they will not move, if device is at all stuck or not responding increase speed.
- If not enough power is supplied to the system as a whole, please charge batteries
- If there is delay between commands when controlling from work, it may be due to video delay
- Using Wi-Fi is not advised at this time, it seems not enough power is supplied to USB port. Ethernet cable is faster and more reliable.
- If cat attacks device itself and is at risk of damage, we will introduce shielding to the device

Software Implementation

Function Overview:

The system has the following main functions: Motion, Laser, Video, Sound, Network, Command, and Motion Sensor.

- Motion function moves the device forward, backward, left and right with three speed levels.
- Laser function outputs the four laser patterns of off, wall, floor, and mix.
- Video function displays the video to the user.
- Sound function outputs the feed food sound effect to the pet.
- Network function displays the video and control panel to the user and takes the command inputs from user through the Internet.
- Command function allows the user to input the commands to control the system.
- Motion sensor function detects the movement around the device and enables the laser output.

The functional decomposition diagram is showing below:

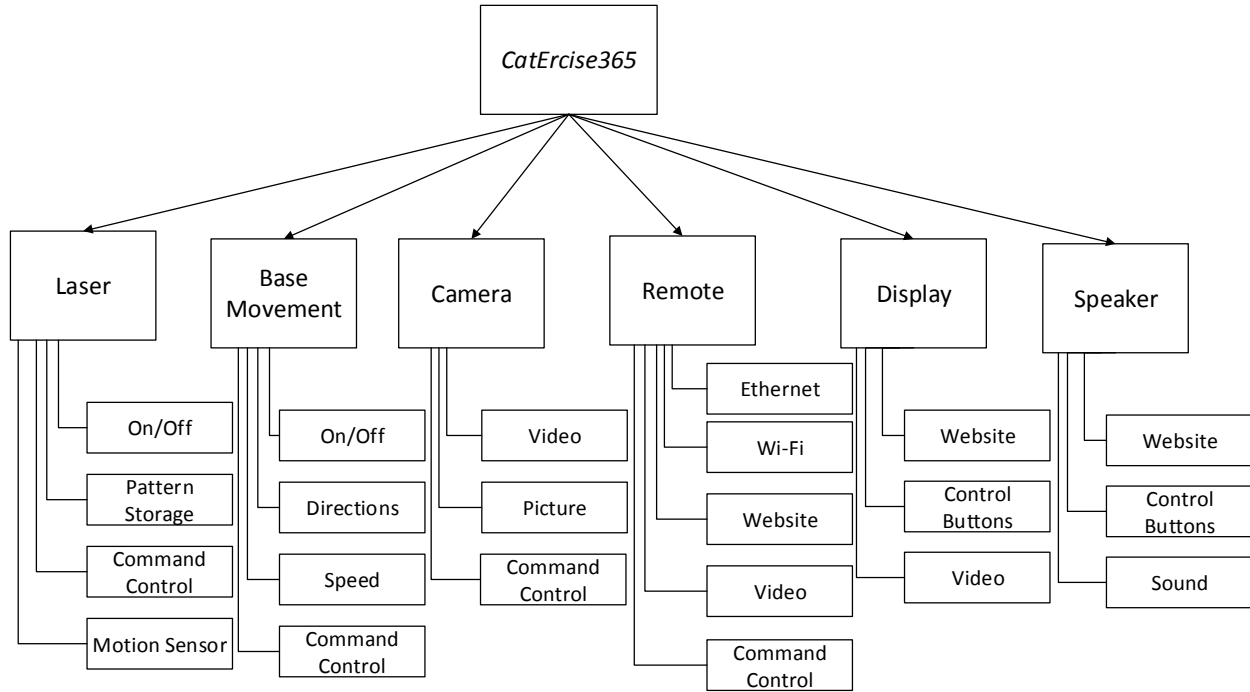


Figure 6 Functional Decomposition

Relationship between tasks:

- MotionTask is a dynamically task. It reads values from command task and outputs to GPIO on Raspberry Pi to control the base movements. It depends on Command Task.
- LaserTask is a dynamically task. It gets input from command task and motion detection. It depends on Command Task. It outputs to GPIO of Raspberry Pi to control the PIC outputs. PIC takes the input from GPIO and output state signals to the laser and servomotors.
- VideoTask is a dynamically task. It reads values from command task and takes video through the camera and output the video to the Internet. It depends on Command Task.
- SoundTask is a dynamically task. It reads values from command task and output sound effect to the pet. It depends on Command Task.
- NetworkTask is a static task. It gets input from user button clicks and save the input into a txt file. It outputs video and control panel on the websites.
- CommandTask is a static task. It reads the command line from control.txt file and deletes that line. It parses the command to operate the corresponding tasks.
- MotionSensorTask is a dynamically task. It detects the movements around it and enable the laser task if there is a movement detected.

The task/class diagram is shown below:

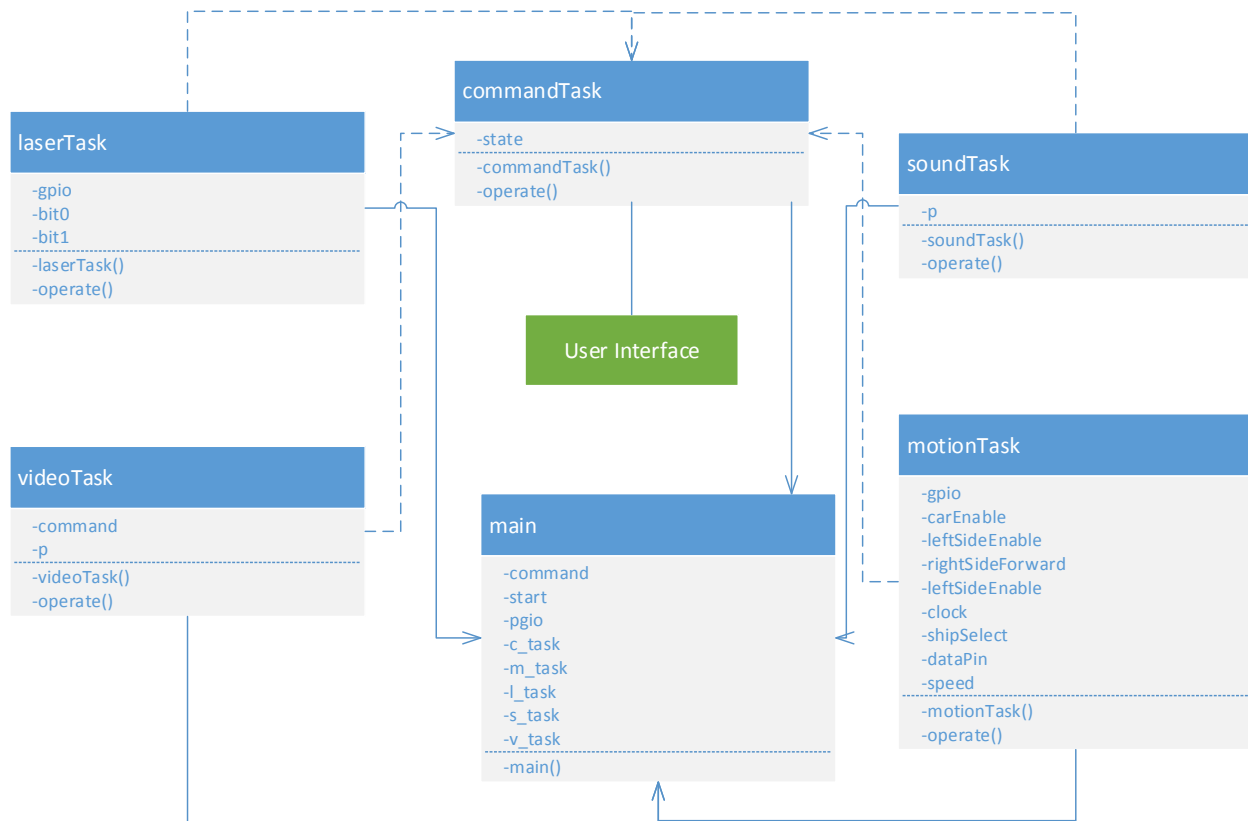


Figure 7 Class Diagram

Ordering of Tasks:

In the simple the ordering of the task are scheduled in the following activity diagram. In the scheduler the order of the tasks is: Network Task, Command Task. All other tasks including MotionTask, LaserTask, VideoTask, SoundTask and MotionSensorTask are operated on demand. Command task will enable MotionTask, LaserTask, VideoTask and SoundTask based on the input command from the user. MotionSensorTask is enabled by the motion detection button. Therefore, the activity and sequence diagrams are shown below:

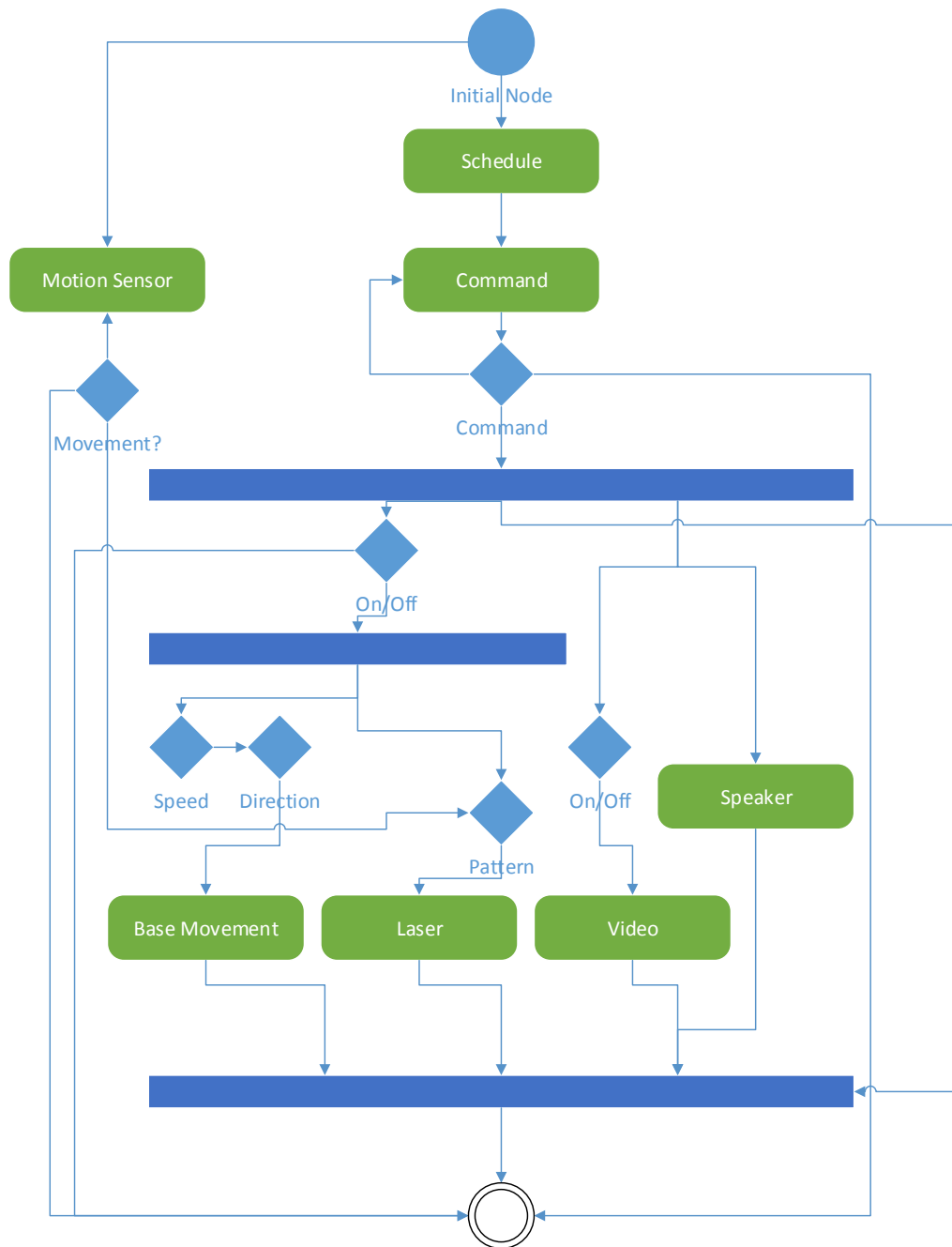


Figure 8 Pi Activity Diagram

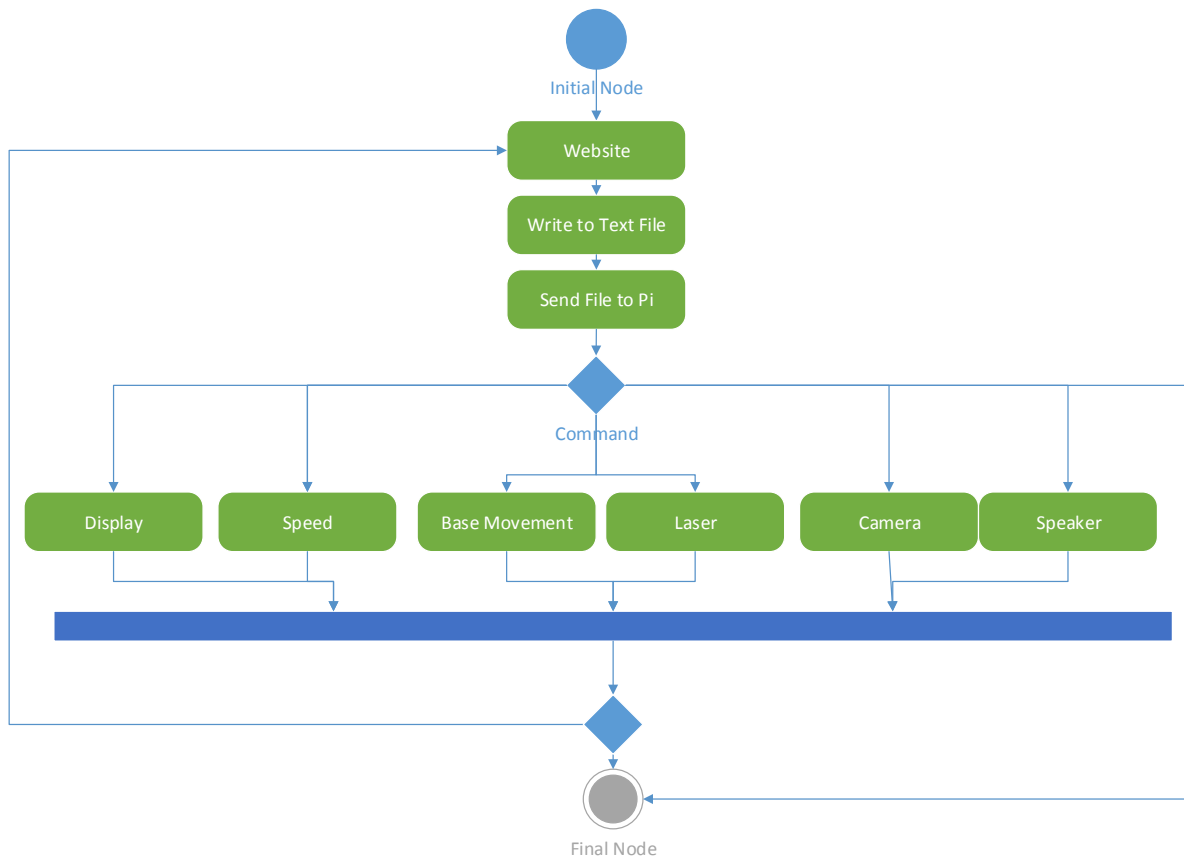


Figure 9 Website Activity Diagram

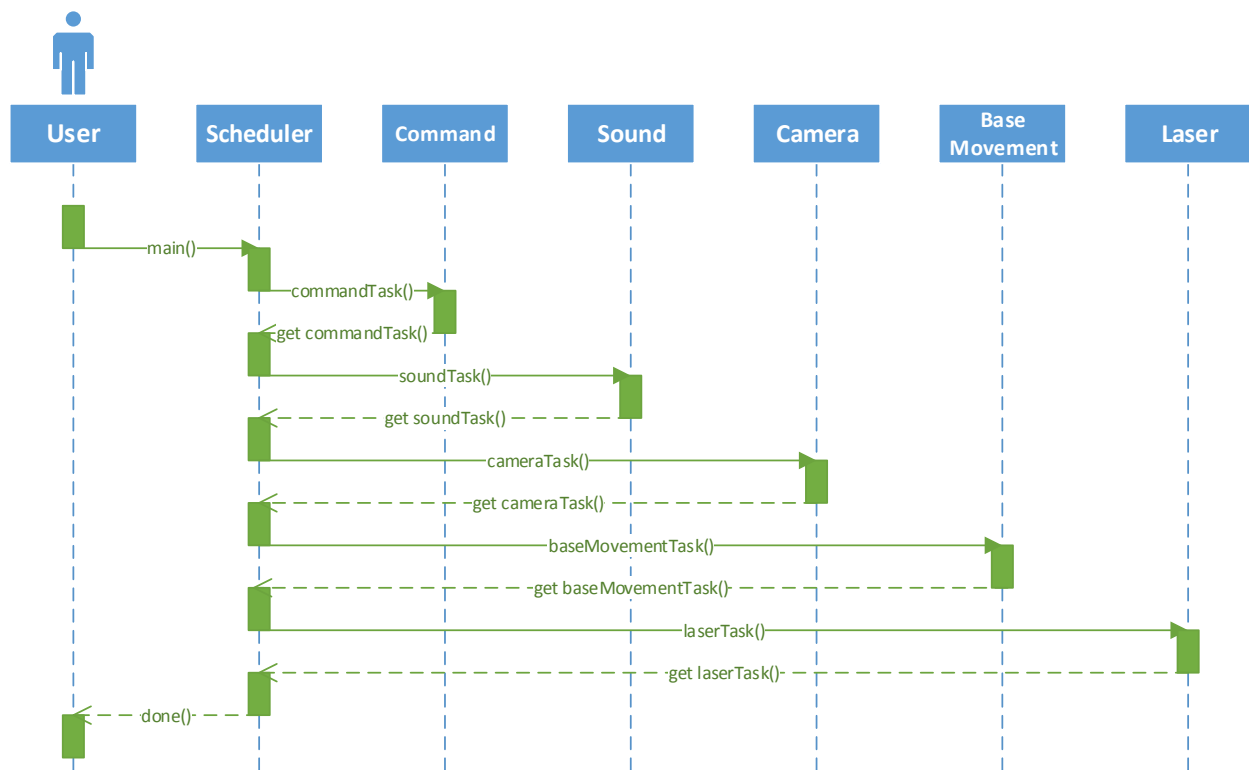


Figure 10 Sequence Diagram

Data Flow:

Command data are collected from network task through website using PHP, HTML and JavaScript then converted into the string stored in a control.txt file on the Raspberry Pi. The command task will read the control.txt file and get the command line saved as a command and delete this line in control.txt file. This command will be parsed in the command task. If the command line starting with “m_” the system will pass this command to motion task and operate the motion task. The motion task will read the command and give the corresponding output. If the command line starting with “l_” the system will pass this command to laser task and operate the laser task. The motion task will read the command and output laser pattern.

If the command line starting with “v_” the system will pass this command to video task and operate the video task. The video task will take video input from the camera and send the video data to the website through the Internet.

If the command line starting with “s_” the system will pass this command to sound task and operate the sound task. The sound task will take the feeafood.wav sound as the input and output to the pet using the speaker.

All the output data will be shown on the webpage taken by the video. The data flow control diagram is shown below:

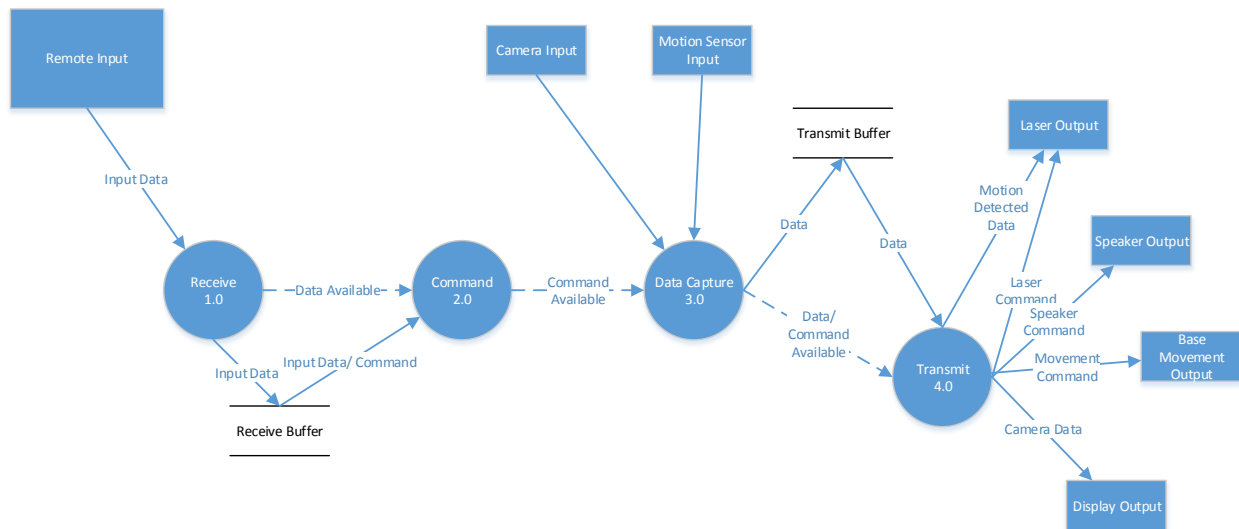


Figure 11 Data Flow Control

Tasks Implementation:

1. Main:

The main function initializes the system, creates and initializes the tasks and runs the tasks in order.

2. Motion:

When the command task reads the command starts with “m_” the system will run the motion task. In the motion task it check the input each time, it reads the input command string. If it reads the input as “m_1”

If it reads the input as “m_2”

If it reads the input as “m_3”
 If it reads the input as “m_up”
 If it reads the input as “m_down”
 If it reads the input as “m_left”
 If it reads the input as “m_right”

3. Laser:

When the command task reads the command starts with “l_” the system will run the laser task. In the laser task it check the input each time, it reads the input command string. If it reads the input as “l_wall”

If it reads the input as “m_floor”
 If it reads the input as “m_mix”
 If it reads the input as “m_off”

4. Video:




When the command task reads the command starts with “v_” the system will run the video task. In the video task it check the input each time, it reads the input command string. If it reads the input as “v_on” it turns on the video on the device and taking the video data and saved into a buffer, then sending this buffer data using html to the website. If it reads the input as “v_off” it turns off the video on the device and there is no data goes to the video window on website so the video window will be off.






5. Sound:

When the command task reads the command starts with “s_” the system will run the sound task. In the sound task it check the input each time, it reads the input command string. If it reads the input as “s_on” it reads the feedfood.wav file and output it through the speaker.

6. Network:

When the system is on it runs the network task which enables the server so the user can see the website. It outputs the control panel website and video website to the user using HTML, PHP, CSS and JavaScript. When the user click the button on the website is enable PHP page and write a command line into control.txt file. The written command line corresponding to each button is based on the following table:

Button	Command
Video: ON	v_on
Video: OFF	v_off
	m_up
	m_down
	m_left

	m_right
	m_1
	m_2
	m_3
Laser: Wall	l_wall
Laser: Floor	l_floor
Laser: Mix	l_mix
Laser: OFF	l_off
Speaker 	s_on

7. Command:

When the main task is running the system runs the command task each time cycle. The command task reads the data in the control.txt file. If it reads a line it saved the command data and delete that line from the control.txt file, so it won't read that next time. Then parse the command data and call the corresponding task based on the command data. The task enabled based on the commands are shown as the following table:

Command	Operating Task	Function
v_on	Video	Video turns on
v_off	Video	Video turns off
m_up	Motion	The CatErcise365 moves forward for one unit
m_down	Motion	The CatErcise365 moves backward for one unit
m_left	Motion	The CatErcise365 moves to the left for one unit
m_right	Motion	The CatErcise365 moves to the right for one unit
m_1	Motion	The motion movement changes to slowest mode
m_2	Motion	The motion movement changes to median mode
m_3	Motion	The motion movement changes to fastest mode
l_wall	Laser	Laser turns on and the pattern is wall
l_floor	Laser	Laser turns on and the pattern is floor
l_mix	Laser	Laser turns on and the pattern is mix of floor and wall
l_off	Laser	Laser turns off
s_on	Sound	Sound turns on and output feed food sound effect

Side Effects:

If the inputs are invalid then system won't take it.

Hardware Implementation

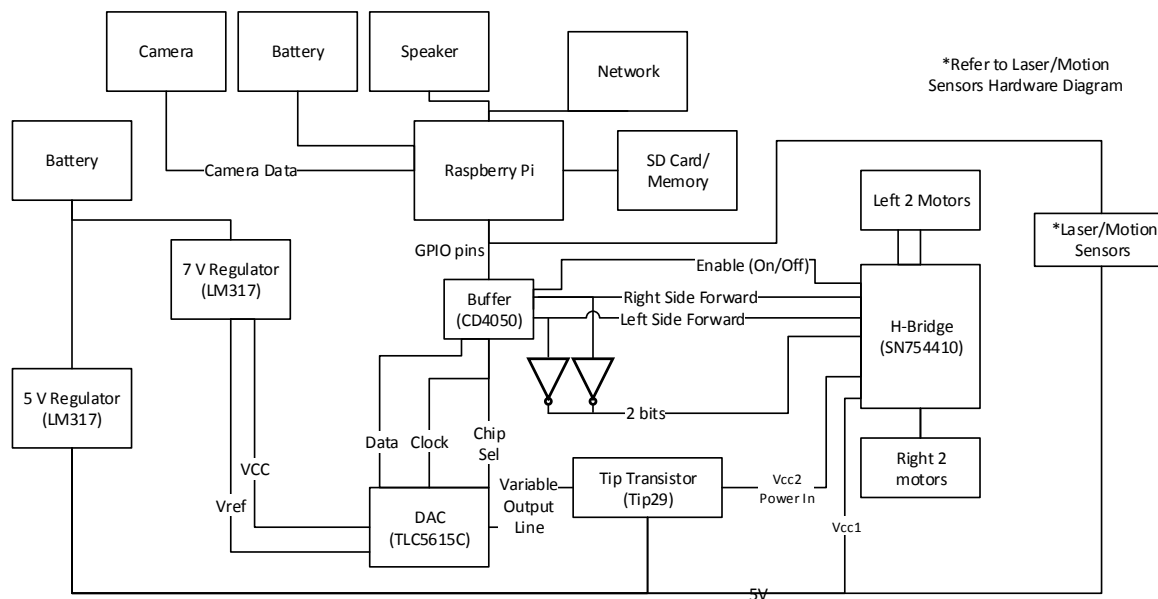


Figure 12 Overall Hardware Block Diagram

Overview:

The hardware to our project consisted of two main controllers, the Raspberry Pi (Rpi) and a PIC 18F25K22. The Raspberry Pi supported the server for the internet control of the device and display of the video feed, control the camera and output to a speaker as well as handling the main components of the entire device with its General Purpose Input/output (GPIO) pins.

1. Main Processor - Raspberry Pi as the main controller controlling

The Raspberry Pi is the main processor of the system. It receives data from network and save it into a text file. And the catercise program reads the file and operate each task based on the command. Raspberry Pi sends commands to the PIC which controls these portions of the hardware. The Raspberry Pi also directly controls some of the hardware itself using GPIO pins. It directly controls the movement/motion control of the device using its GPIO pins allowing for 360 degree motion.

2. Laser and servos:

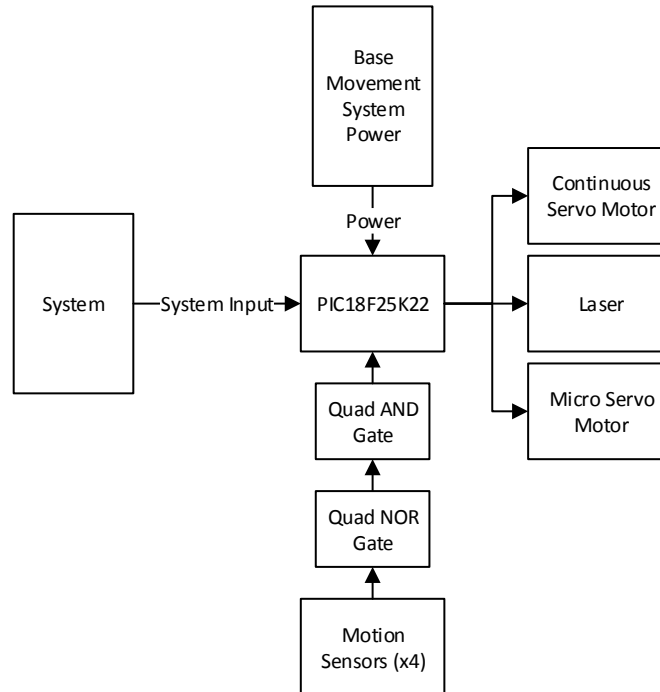


Figure 13 Laser Hardware Block Diagram

The Raspberry Pi controls the state of the PIC18F25K22 allowing it to control the two laser servo and the laser diode, as well as the four peripheral motion sensors. The laser goes 365 degrees around the device with a full rotation servo and about 30 degrees up and down adjacent to the device with a micro servo, both of these, as well as the on or off state of the diode, are controlled by the microcontroller, which has Pulse Width Modulation (PWM) outputs for the servos. The PIC receives the same power source as the base movement system.

3. Motion Sensors:

As for the four motion sensors, when one detects motion it send a 2.5V “on” voltage to a series of logic gates (NOR gate [CD4001B] and AND [DM74LS08] gates) when it senses a moving object. This, then lights up LED’s corresponding to the tripped motion sensor, this also sends a signal to the 18F25K22 in order to turn on the laser and run the servo patterns in order to allow the cat play with it when it passes the device.

4. Speaker and camera:

The speaker and camera is directly connected on the Raspberry Pi. The Raspberry Pi sends audio signals to its speaker as well as sends signals to turn on and off the camera.

5. Motor Driver:

Controlling this motor driver consists of sending three pins to the series Digital to Analog (TLC5615) converter which outputs varying voltages based on the speed the user wants to go. It controls this speed using these voltage levels as the output of the digital to analog converter goes first to a TIP29 power transistors’ base since the logic output would not be enough power to drive the motors. This transistors collector draws current from the five volt rail then the current

amplified voltage level is sent out the emitter and is sent to the H-Bridge power input (quadruple half-H Driver SN754410). Based on the signals from the Rpi, the H-Bridge sends power signals to all four wheel motors, flipping polarities, telling them to go forward, backward, and at what speed, slow, medium, fast, and stop. The Rpi sends three control pins to the H-Bridge to control the direction of each wheel allowing the device to turn left and right, move forward and backward. Other parts of the motor driver are the not gates (Hex Inverters HD74S04) used to save on pins on the Rpi, exactly two not gates are used since polarities need to be switched such that one input to the H-Bridge is exactly opposite the other. Another piece is the MOSFET buffers CD4049UBC (HEX non-inverting buffer) for the six motor driver pins from the Raspberry Pi, this buffers the Rpi from back current and voltage spiking over 3.3 volts that the Rpi can handle.

6. Powering the system:

The system as a whole runs on two distinct voltage levels, a five volt rail which powers the motor driver as well as the laser servo motors and the motion sensors. The other one is a 7 volt supply for the motor driver (first to the digital to analog converter), these voltages are regulated by two voltage regulators LM317, which two voltage regulators down step from the battery pack. The battery pack is a 1200 mAH 8.4V RC car supply voltage. The Rpi is powered directly from a big battery pack designed specifically to charge phones and outputs high power (NAME).

TEST PLAN

The system consists of several components, which needs to be tested both separately and together. Tests will cover the scheduler, and all the tasks implemented: motion, laser, video, sound, command, network and motion sensor.

The tasks need to be tested for correct functionality separately. Then, inter-task communication (variable passing), correct execution order, and correct execution time need to be tested for the system.

Exception and error handling is not a priority for these tests. The system is generating well-defined values, and the command inputs received during execution time are defined in the websites. The tasks at this point are NOT required to distinguish between correct and nonsense values.

TEST SPECIFICATION

Scheduler tests:

- Every task are executed
- Main functions of tasks are called with correct data passed in

System Initial tests:

- System initial task runs only once
- Tasks are executed in correct order

- All tasks are disabled on the beginning until there is a command input

Network task tests:

- Network should be enabled and provide a website through the server
- Ensure the video and control panel webpage displayed on the given IP address
- Get the command saved when button on the control panel clicked

Video task tests:

- Displays the real time video (couple seconds delay is allowed) on the website
- Video can be turned on, off and suspend
- Video is controlled by the ON/OFF buttons on the website

Sound task tests:

- Output the feed food sound effect to the pet
- Sound is controlled by “speaker” button on the website

Motion task tests:

- Output 4 directions – forward, backward, left and right to control the system move to these four directions
- Output 3 speed levels for the movement
- Directions and speed levels are controlled by the buttons on the website

Laser task tests:




- Outputs the correct laser pattern (wall, floor or mix) to the pet
- Laser pattern is controlled by the button on the website or motion sensor






Motion sensor task tests:

- Detects the movement around the system to enable the laser
- Motion sensor detection can be turn on and off by the switch

Command task tests:

- For each button on the website ensure
- Sending the correct value and operating correct task when click the button according to the following table:

Button	Command	Operating Task	Function
Video: ON	v_on	Video	Video turns on
Video: OFF	v_off	Video	Video turns off
	m_up	Motion	The CatErcise365 moves forward for one unit
	m_down	Motion	The CatErcise365 moves backward for one unit
	m_left	Motion	The CatErcise365 moves to the left for one unit

	m_right	Motion	The CatErcise365 moves to the right for one unit
	m_1	Motion	The motion movement changes to slowest mode
	m_2	Motion	The motion movement changes to median mode
	m_3	Motion	The motion movement changes to fastest mode
Laser: Wall	l_wall	Laser	Laser turns on and the pattern is wall
Laser: Floor	l_floor	Laser	Laser turns on and the pattern is floor
Laser: Mix	l_mix	Laser	Laser turns on and the pattern is mix of floor and wall
Laser: OFF	l_off	Laser	Laser turns off
Speaker 	s_on	Sound	Sound turns on and output feed food sound effect

TEST CASES

Scheduler tests:

- Breakpoints at task entry instructions for correct order and verifying data passing.

System Initial tests:

- Breakpoint in scheduler to ensure system initial task is only called once, and the motion and laser are off initially.


Network task tests:

- Turn on the system the network should be enabled and user can see the website at given IP address
- Click the “control panel” button the control panel webpage should be displayed
- The command data can be send to the server using network and saved the into control.txt file








Video task tests:

- When the system starts the video task should be off
- When the command input starting with “v_” the video task should run
- If the video “ON” button is clicked the camera should turn on and display the real time (couple seconds delay is allowed) video on the video webpage
- If the video “OFF” button is clicked the camera should turn off and display the video display on the video webpage should turned off

Sound task tests:

- When the system starts the sound task should be off
- When the command input starting with “s_” the sound task should run
- If the sound “” button is clicked the speaker should output feed food sound effect to the pet
- When feed food sound effect is end the sound task should terminate

Motion task tests:

- When the system starts the motion task should be off
- When the command input starting with “m_” the sound task should run
- If the speed level button “” is clicked the speed level is set to slowest level
- If the speed level button “” is clicked the speed level is set to median level
- If the speed level button “” is clicked the speed level is set to fastest level
- If the direction up button “” is clicked the CatExercise365 moves forward for one unit based on the current speed setting
- If the direction down button “” is clicked the CatExercise365 moves forward for one unit based on the current speed setting
- If the direction left button “” is clicked the CatExercise365 moves to the left for one unit based on the current speed setting
- If the direction right button “” is clicked the CatExercise365 moves to the right for one unit based on the current speed setting
- When one unit movement is finished the food motion task should terminate

Laser task tests:

- When the system starts the laser task should be off
- When the command input starting with “v_” the video task should run
- If the laser “**Wall**” button is clicked the laser should turn on and output the wall pattern to the pet which is the laser movements on the wall
- If the laser “**Floor**” button is clicked the laser should turn on and output the floor pattern to the pet which is the laser movements on the floor
- If the laser “**Mix**” button is clicked the laser should turn on and output the mix pattern to the pet which is the laser movements on the floor and wall
- If the laser “**OFF**” button is clicked the laser should turn off if the motion sensor detection is off. Otherwise the laser is controlled by the motion detection.
- When motion detection is on, if there is any motion detected around the system the laser should turn on.
- When motion detection is on, if no motion detected around the system the laser should turn off.

Motion sensor task tests:

- If the switch is on the motion detection function is on.
- If the switch is off the motion detection function is off.

- When motion detection is on, if there is any motion detected around the system the laser should turn on.
- When motion detection is on, if no motion detected around the system the laser should turn off.
- When motion detection is off, laser is not controlled by the motion detection.

Command task tests:

- Read one line from control.txt file, operate the corresponding task and delete this line from the control.txt file
- Command s_on – operate sound task to output feed food sound effect
- Command v_on – operate video task to turn the camera on
- Command v_off – operate video task to turn the camera off
- Command m_1 – operate motion task to change the speed level to slowest
- Command m_2 – operate motion task to change the speed level to median
- Command m_3 – operate motion task to change the speed level to fastest
- Command m_up – operate motion task to move forward one unit
- Command m_down – operate motion task to move backward one unit
- Command m_left – operate motion task to move to the left one unit
- Command m_right – operate motion task to move to the right one unit
- Command l_wall – operate the laser task to output wall laser pattern
- Command l_floor – operate the laser task to output floor laser pattern
- Command l_mix – operate the laser task to output mix laser pattern
- Command l_off – operate the laser task to turn laser off

PRESENTATION, DISCUSSION, AND ANALYSIS OF THE RESULTS

Our project worked according to our design. Each task executed correctly by being able to activate our system through a website. The website is designed so the user is able to control the device from work. The website has a network that the user can controls the system and be able to choose the device's laser patterns, movement the device will take, video, sound, and motion sensors.

Network:

The network works accordingly as the message gets sent to the device form the website. When a button is clicked on the website, the device will implement the command attached to the button. The different commands that are on the website are the laser patterns, movement the device will take, activate sound, and enabling/disabling video feed. The capture of the website is shown below:

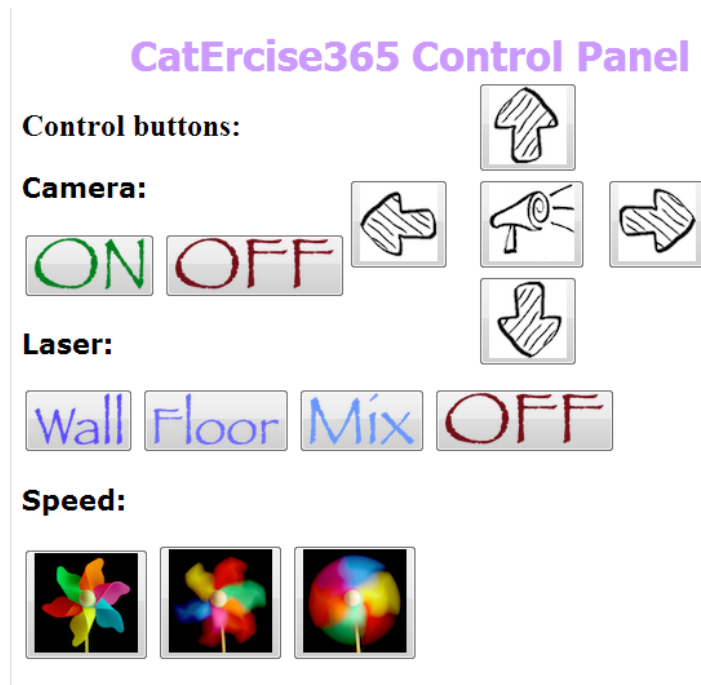


Figure 14 Control Panel Webpage

Laser:

The laser tasks works as expected. The website controls the laser by choosing patterns that are preprogrammed into a PIC microcontroller. There are four choices the user can choose from including an off function. The other functions are wall, floor, and mix. The wall will have the laser do a motion on the wall while the floor will have the laser circle around the device, and lastly the mix will do a mixture of the wall and floor patterns. Mix will circle around the device while moving up and down.

A problem that occurred was the wiring. The laser requires the use of two servo motors. Since we designed the laser to be able to circle around the device, one of the servo motors is a continuous rotational motor while the other one is a micro servo motor. The continuous rotational motor controls the horizontal direction while the micro servo motor controls the vertical direction. Both servo motors are attached to a PIC microcontroller. The problem that occurred is that the micro servo motor's wires gets twisted due to the continuous rotational motor. The solution was to have the continuous motor go back and forth a full rotation in one direction and then go in the opposite direction for another full rotation.

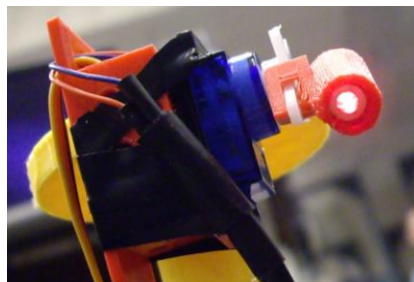


Figure 15 Laser On

Motion:

The movement task works as expected. There are three speeds the device can take and each speed can have the device go forwards, backwards, right, and left. The speed of the device is chosen by the user through the website as well as the directions. The device will only move a short distance instead of having continuous movement.

A slight difficulty with the movement is. The movement part was tested to move while the system was not on the floor and on the floor. The movement worked as designed. However, when all the parts were integrated together and was tested on the floor, the device did not move. We concluded that there was too much weight on the car to move as they were weighted down with the other parts of the device. The movement design was weighted down its batteries, the battery powering the Raspberry Pi, the Raspberry Pi, the speaker, and the circuit designed for the laser and PIR motion sensors.

Video:

The video task works. There are two buttons on the website to turn off and on the camera. The camera turns on and off when their corresponding button is pressed on the website. There is a slight delay between the camera and the video being outputted on the website. After some research on forums and consulting with other groups with cameras connected to their Pi, we concluded that there will be a lag due to the design. This is due to our design where the video is being written to a buffer which is causing a delay in our video stream. To compensate for that, we redesigned our movement to only move a certain distance instead of continuous driving, allowing the video streaming to catch up to the system. The video webpage is shown below:

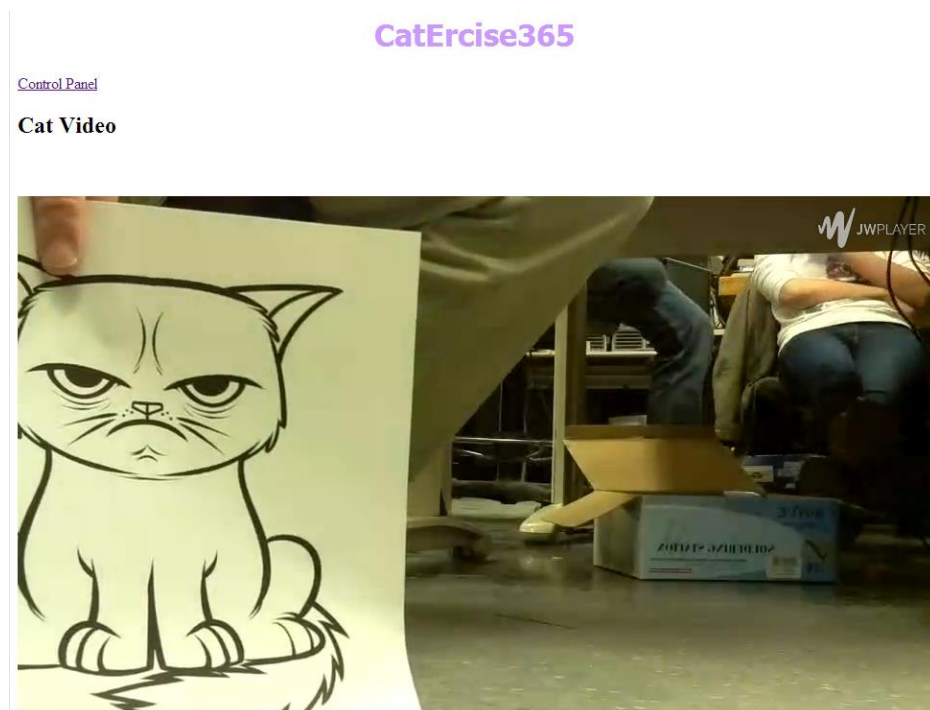


Figure 16 Video Webpage

Sound:

The sound task works as expected. There is one button to activate the sound. When the button is clicked on the website, a speaker will output the sound. The sound is prerecorded onto the device which is of a pet food being opened and poured.

Motion Sensors:

The motion task works as expected. This task is not implemented through the website. Instead, there are motion sensors on the device that will detect if there is movement nearby. If there is movement nearby, the sensors will notify a PIC microcontroller which will then turn on the laser and do a floor pattern as long as there is movement. Otherwise, the laser will be off. The LEDs on the image below shows the motion sensors are activate:

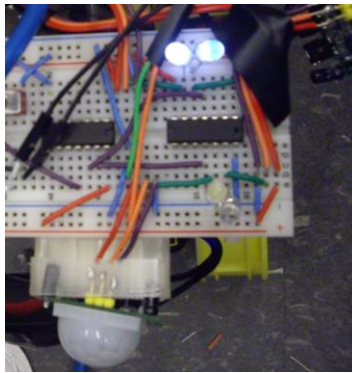


Figure 17 Motion Sensor

ANALYSIS OF ANY ERRORS

There were small problems that occurred within the project that were solved or redesigned later on throughout the development period explained above. Below is the difficulties that occurred with the system, the Raspberry Pi.

Raspberry Pi:

In general, we believe the Raspberry Pi we received was not well made. Throughout the project, there were problems. The problems that occurred was the USB ports (explained below), and the connections between our circuit and Raspberry Pi. There were times that we were not able to turn on the Raspberry Pi or the camera. We believed the problem with not being able to turn on the Raspberry Pi and camera was due to the SD card not being properly mounted to the Pi. They had a place to hold the SD card but it was very easy to knock the card out of place and make the Pi malfunction. So to fix this problem, we 3D printed out a bottom Pi case. Also the pi was easily affected by the GPIO pins when connected with our circuits. To fix this problem, buffers were added to buffer the Pi and to raise their logic level from 3.3 Volts to 5 Volts.

ANALYSIS OF WHY THE PROJECT MAY NOT OF WORKED AND WHAT EFFORTS WERE MADE TO IDENTIFY THE ROOT CAUSE OF ANY PROBLEMS

A problem we encountered is the Wi-Fi network. We concluded that it was the University of Washington's Wi-Fi network. Once we were connected to the Wi-Fi, we would eventually be disconnected and would not be able to reconnect unless the Pi was restarted. This hypothesis was concluded when other groups with Pi and Wi-Fi had the same problem. Another situation that concluded our hypothesis was other groups had spotty Wi-Fi connections with the network on their laptops.

Another problem we encountered was using the USB ports on the Raspberry Pi. We concluded that it was the system's hardware which is the Raspberry Pi. When we first received the Raspberry Pi, we were able to use a Bluetooth and Wi-Fi dongle and they worked perfectly. The Bluetooth dongle was used to connect to a keyboard and mouse. However over time, the Bluetooth dongle would not work anymore but the Wi-Fi dongle did. We then used an external powered USB hub which allowed the keyboard and mouse connect to the Pi. But later on, the Wi-Fi dongle would not work anymore. At first, the Wi-Fi connection would be lost and it would take a while before Wi-Fi was enabled again. In the end, the Wi-Fi connection barely works. We analyzed and concluded the root cause to be the Raspberry Pi's hardware by borrowing other group's Wi-Fi dongle and tried to use them, which did not work. This concluded that the Pi has some type of hardware malfunction.

SUMMARY

The objective of this lab is to use skills learned from previous labs and classes to design an interesting and useful product to practice a full development life cycle of designing, building, and testing. For our lab we used the PIC 18F25K22 Peripheral Interface Controller microprocessor as well as the Raspberry Pi in order to build a cat exercise machine that allows users to manually and automatically play and exercise their cats remotely, either from work or from wherever is needed. Our project had three main parts were implemented on the project, the movement/car portion which controlled the movement of the device mainly by the Raspberry Pi, another portion was laser control which was mainly controlled by the PIC18F25K22 controlling servos which moved a laser pointer in 360 degrees around the device as well as up and down to create an interesting pattern for the cat to chase, the last portion was the network portion which allows the user to have control of the device remotely and see the cat playing with the CatErcise 365®. Other portions include the camera which will allow the user to see the cat as it plays with the device and be able to control the base movement of the device, the speaker which will make a sound when it turns on in order to let the cat know it's time to play, there are also motion sensors implemented on the PIC18F25K22 which will allow the cat to play with the device when the owner is not in control. The control of the user is done on the website which is upheld by the network section of the project. The control works very well controlling speed, direction, laser patterns and sound, it even gets a video signal, but unfortunately it is delayed by 5-7 seconds. Other problems we had are detailed in problem section but include powering the movement section and making sure all three speed levels worked. The requirement specification describes and defines the design of our project while the design specification is a more detailed version.

The test plan, test specifications, and test cases are created to make sure that our implementation to the project is according to the laboratory specifications. Our results were successful and worked accordingly to the laboratory specifications.

CONCLUSION

In conclusion, after completing this lab, we learned the 18F25K22 PIC microprocessor, Raspberry Pi, servos, motors, motion sensors and used our knowledge from the previous lab to create a portable remote controlled cat exercise device system. In this project we learn how to build server on the Raspberry Pi and how to run our program on it combined with other hardware connections. We also learned about the production procedure in embedded system including design, implementing, testing and improving. Network enabled in our system to allow the user to communicate with the system through a computer over the Internet. We worked through the problems and fixed them accordingly to make the system work. The outcome of the project was a success and it worked to the project's specification.

APPENDICES

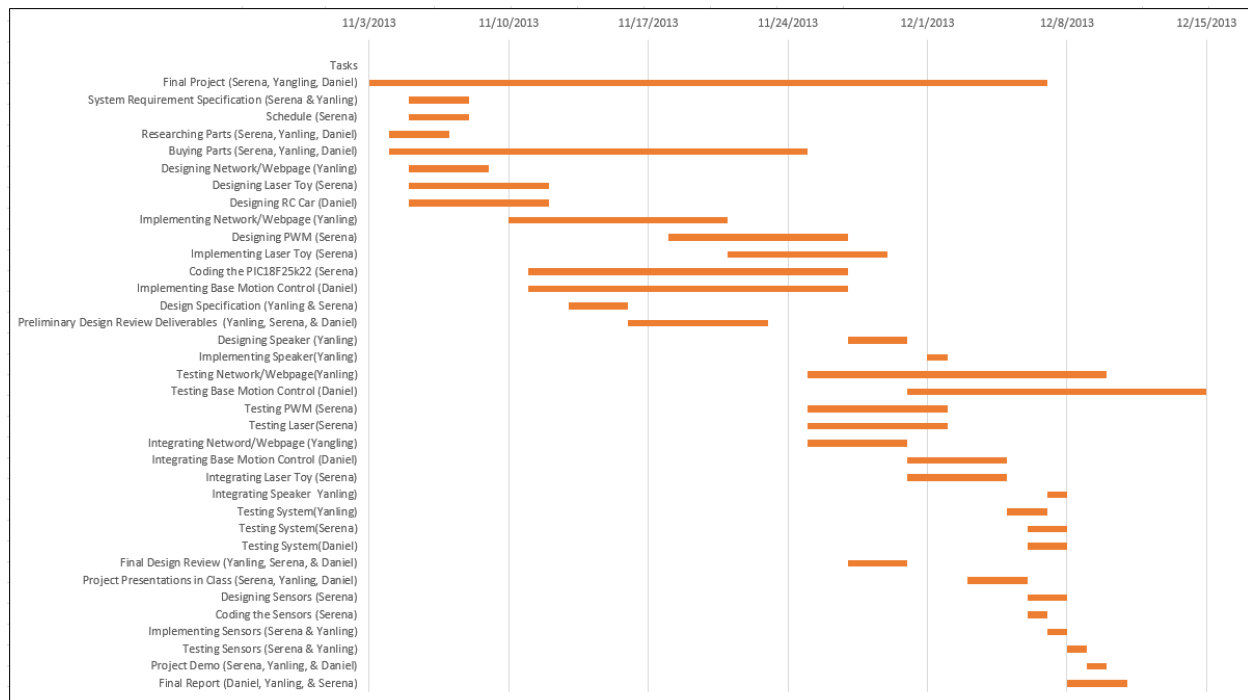


Figure 18 Final Project Grant Chart

Part	# of units	Cost	Total
Raspberry Pi 5MP Camera	1	\$34.43	\$34.43
Raspberry Pi	1	\$40.50	\$40.50
Wi-Fi Dongle	1	\$10.00	\$10.00
HDMI to DVI cable	1	\$5.00	\$5.00
PIC Microprocessor 18F25K22	1	\$7.70	\$7.70
Continuous Rotation Servo	1	\$20.99	\$20.99
Micro Servo Motor	1	\$4.49	\$4.49
5mW laser diode	1	\$4.99	\$4.99
PIR Motion Sensors	5	\$3.00	\$15.00
White LEDs	4	\$0.20	\$0.80
Quad 2-Input AND gates (74LS08)	1	\$0.40	\$0.40
Voltage Regulators	2	\$0.40	\$0.80
RC Car	1	\$23.00	\$23.00
H bridge	1	\$2.58	\$2.58
10 Bit Digital-to-Analog Converters (TLC5615C)	1	\$5.07	\$5.07
Hex Non-Inverting Buffer (CD4050BC)	1	\$0.50	\$0.50
Hex Inverter	1	\$0.40	\$0.40
Red LEDs	2	\$0.20	\$0.40
Power Transistor, PNP (Tip29)	2	\$0.50	\$1.00
Slide Switch	2	\$0.40	\$0.80
Toggle Switch	1	\$1.50	\$1.50
240 ohm resistor	2	\$0.10	\$0.20
1k ohm resistor	1	\$0.10	\$0.10
1.5k ohm Resistors	1	\$0.10	\$0.10
Voltage Regulator (LM317T)	1	\$0.60	\$0.60
Heat Sinker	1	\$0.40	\$0.40
SD Card	1	\$10.70	\$10.70
Spool of Wires	4	\$1.20	\$4.80
Total			\$186.55

Figure 19 Bill of Materials

Table Contribution
Requirement Specs (Serena & Yanling)
Design Specs (Yanling)
Deliverables (Serena and Yanling)
Bill of Materials (Serena)
Grant Chart (Serena)
UML Use Case (Serena)
Functional Decomposition (Daniel)
Block Diagram (Yanling)
Network Task Implementation (Yanling)
Motion Task Implementation (Daniel)
Video Task Implementation (Yanling)
Laser Task Implementation (Serena)
Sound Task Implementation (Yanling)
Motion Sensor Implementation (Yanling & Serena)
Integration (Daniel, Serena and Yanling)
Project Presentation (Daniel, Serena and Yanling)
Full Project Demo (Daniel, Serena and Yanling)
Final Report Abstract & Intro (Daniel)
Final Report Requirement Spec (Yanling)
Final Report Design Spec (Yanling)
Final Report Design Procedure (Daniel)
Final Report System Description (Daniel)
Final Report Software Implementation (Yanling)
Final Report Hardware Implementation (Daniel)
Final Report Test Plan & Test Cases (Yanling)
Final Report Result Analysis (Serena)
Final Report All the UMLs (Serena)
Final Report Summary (Daniel)
Final Report Conclusion (Yanling)

Figure 20 Table Contribution

CODE

Catercise.java

```
/*
 * *****
 * PROJECT: EE 478 Final Project Catercise
 * GROUP MEMBERS: Yanling He, Serena Lam, Daniel Mueller
 * FILENAME: cattersise.java
 *
 * This file is a part of the CatErcise365 project.
 * *****
 * %%
 * Copyright (C) 2013 CatErcise365
 * %%
 * Author: Yanling He
 */

import java.io.*;
import java.lang.InterruptedExceoption;
import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinState;
import com.pi4j.io.gpio.RaspiPin;

public class catercise {

    public static void main(String [] args) throws IOException, InterruptedException{

        System.out.println("CatErcise365 starting...");
        String command = null;
        char start = 'n';

        final GpioController gpio = GpioFactory.getInstance();

        commandTask c_task = new commandTask();
        motionTask m_task = new motionTask(gpio);
        laserTask l_task = new laserTask(gpio);
        soundTask s_task = new soundTask();
        videoTask v_task = new videoTask();

        System.out.println("CatErcise365 running...");
        while(true) {
            command = c_task.operate("start");
        }
    }
}
```

```

        if (command != null) {
            start = command.charAt(0);
            if (start == 'm') {
                System.out.println("Running motion task " + command + "...");
                m_task.operate(command);
            } else if (start == 'l') {
                System.out.println("Running laser task " + command + "...");
                l_task.operate(command);
            } else if (start == 's') {
                System.out.println("Running sound task " + command + "...");
                s_task.operate(command);
            } else if (start == 'v') {
                v_task.operate(command);
            }
        }
    }
}
}
}

```

commandTask.java

```

/*
 * *****
 * PROJECT: EE 478 Final Project Catercise
 * GROUP MEMBERS: Yanling He, Serena Lam, Daniel Mueller
 * FILENAME: commandTask.java
 *
 * This file is a part of the CatErcise365 project.
 * *****
 * %%
 * Copyright (C) 2013 CatErcise365
 * %%
 * Author: Yanling He
 */

```

```

import java.io.*;

public class commandTask {
    private String state;

    public commandTask() {
        state = "sleep";
    }

    public String operate(String newSt) throws IOException{
        state = newSt;
        if (state.equals("start")) {

```



```

File inputFile = new File("/home/pi/catercise/control/control.txt");
File tmpFile = new File("/home/pi/catercise/control/tmpcontrol.txt");

BufferedReader reader = new BufferedReader(new FileReader(inputFile));
BufferedWriter writer = new BufferedWriter(new FileWriter(tmpFile));

    String command = null;
    String line = null;

    tmpFile.setReadable(true, false);
    tmpFile.setWritable(true, false);

    command = reader.readLine();
    while ((line = reader.readLine()) != null) {
        writer.write(line);
        writer.newLine();
    }
    writer.close();

    boolean successful = tmpFile.renameTo(inputFile);
    if (successful && command != null) {
        return command;
    } else {
        return null;
    }
} else {
    return null;
}
}
}

```

videoTask.java

```

/*
 * *****
 * PROJECT: EE 478 Final Project Catercise
 * GROUP MEMBERS: Yanling He, Serena Lam, Daniel Mueller
 * FILENAME: videoTask.java
 *
 * This file is a part of the CatErcise365 project.
 * *****
 * %%
 * Copyright (C) 2013 CatErcise365
 * %%
 * Author: Yanling He
 */

```

```

import java.io.IOException;
import java.lang.Runtime;
import java.lang.Process;

public class videoTask {
    private String command;
    Process p;

    public videoTask() {
        command = null;
        p = null;
    }

    public void operate(String command) throws IOException{
        System.out.println("Running video task " + command + "...");
        if (command.equals("v_on")) {
            try {
                p = Runtime.getRuntime().exec("./camera");
            } catch (Exception err) {
                err.printStackTrace();
            }
        } else if (command.equals("v_off")){
            p.destroy();
        }
    }
}

```

soundTask.java

```

/*
 * *****
 * PROJECT: EE 478 Final Project Catercise
 * GROUP MEMBERS: Yanling He, Serena Lam, Daniel Mueller
 * FILENAME: soundTask.java
 *
 * This file is a part of the CatErcise365 project.
 * *****
 * %%
 * Copyright (C) 2013 CatErcise365
 * %%
 * Author: Yanling He
 */

```

```

import java.io.IOException;
import java.lang.Runtime;
import java.lang.Process;

```

```

public class soundTask {
    Process p;

    public soundTask() {
        p = null;
    }

    public void operate(String command) throws IOException{
        System.out.println("Running sound task " + command + "...");
        if (command.equals("s_on")) {
            try {
                p = Runtime.getRuntime().exec("aplay /home/pi/catercise/feedfood.wav");
            } catch (Exception err) {
                err.printStackTrace();
            }
        }
    }
}

```

motionTask.java

```

/*
 * *****
 * PROJECT: EE 478 Final Project Catercise
 * GROUP MEMBERS: Yanling He, Serena Lam, Daniel Mueller
 * FILENAME: commandTask.java
 *
 * This file is a part of the CatErcise365 project.
 * *****
 * %%
 * Copyright (C) 2013 CatErcise365
 * %%
 * Author: Daniel Mueller
 */
import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinState;
import com.pi4j.io.gpio.RaspiPin;

public class motionTask {
    private GpioController gpio;
    private GpioPinDigitalOutput carEnable;
    //private GpioPinDigitalOutput leftSideEnable;
    private GpioPinDigitalOutput rightSideForward;
    private GpioPinDigitalOutput leftSideForward;
    private GpioPinDigitalOutput clock;

```

```

private GpioPinDigitalOutput chipSelect;
private GpioPinDigitalOutput dataPin;

boolean speed;

public motionTask(GpioController g) {
    gpio = g;
    carEnable = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_00, "34ER",
PinState.LOW);
    //leftSideEnable = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_03, "12ER",
PinState.LOW);
    rightSideForward = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_01, "34AR",
PinState.HIGH);
    leftSideForward = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_02, "12AR",
PinState.HIGH);
    speed = true; //will be turned to true once hardware is done

    dataPin = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_03, "Clock", PinState.HIGH);
    clock = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_04, "chipSel", PinState.HIGH);
    chipSelect = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_05, "data",
PinState.HIGH);
}

public void operate(String command) throws InterruptedException {

    carEnable.low();
    carEnable.low();
    Thread.sleep(100);

    if(command.equals("m_right")) { //Right
        rightSideForward.high();
        leftSideForward.high();
        carEnable.high();
        carEnable.high();
        Thread.sleep(1000);
        rightSideForward.low();
        leftSideForward.high();
        carEnable.high();
        carEnable.high();
        Thread.sleep(1200);
    } else if (command.equals("m_left")) { //Left
        rightSideForward.high();
        leftSideForward.high();
        carEnable.high();
        carEnable.high();
        Thread.sleep(1000);
    }
}

```

```

    rightSideForward.high();
    leftSideForward.low();
    carEnable.high();
    carEnable.high();
    Thread.sleep(1200);
} else if (command.equals("m_up")) { //up/Forward
    rightSideForward.high();
    leftSideForward.high();
    carEnable.high();
    carEnable.high();
    Thread.sleep(2000);
} else if (command.equals("m_down")) { //down/Backward
    rightSideForward.low();
    leftSideForward.low();
    carEnable.high();
    carEnable.high();
    Thread.sleep(2000);
}
carEnable.low();
carEnable.low();
//Thread.sleep(100);

if(speed)
{
    //put speed stuff here once hardware for it is finalized
    chipSelect.high();
    if(command.equals("m_3"))
    { //loads all ones for max voltage
        chipSelect.low();
        dataPin.high();
        for(int i = 0; i < 1; i++) //12 since there are 2 dont care's at the end
        {
            clock.low();
            Thread.sleep(10);
            clock.high();
            Thread.sleep(10);
        }
        clock.low();

        Thread.sleep(5);
        dataPin.high();
        for(int i = 1; i < 12; i++) //12 since there are 2 dont care's at the end
        {
            clock.low();
            Thread.sleep(10);
            clock.high();
        }
    }
}

```

```

        Thread.sleep(10);
        clock.low();

    }

    Thread.sleep(10);
    dataPin.low();
    Thread.sleep(10);
    chipSelect.high();
} else if(command.equals("m_2"))
{
    chipSelect.low();
    dataPin.low();
    for(int i = 0; i < 1; i++)//12 since there are 2 dont care's at the end
    {
        clock.low();
        Thread.sleep(10);
        clock.high();
        Thread.sleep(10);
        clock.low();
    }
    Thread.sleep(10);
    dataPin.high();

    for(int i = 1; i < 2; i++)//12 since there are 2 dont care's at the end
    {
        clock.low();
        Thread.sleep(10);
        clock.high();
        Thread.sleep(10);
        clock.low();
    }
    Thread.sleep(10);
    dataPin.low();

    for(int i = 2; i < 3; i++)//12 since there are 2 dont care's at the end
    {
        clock.low();
        Thread.sleep(10);
        clock.high();
        Thread.sleep(10);
        clock.low();
    }
    Thread.sleep(10);
    dataPin.high();
}

```

```

for(int i = 3; i < 12; i++)//12 since there are 2 dont care's at the end
{
    clock.low();
    Thread.sleep(10);
    clock.high();
    Thread.sleep(10);
    clock.low();
}

    Thread.sleep(10);
    dataPin.low();
    chipSelect.high();
}else if(command.equals("m_1")) {
    chipSelect.low();
    dataPin.low();
    for(int i = 0; i < 1; i++)//12 since there are 2 dont care's at the end
    {
        clock.low();
        Thread.sleep(10);
        clock.high();
        Thread.sleep(10);
        clock.low();
    }
    Thread.sleep(10);
    dataPin.high();

for(int i = 1; i < 2; i++)//12 since there are 2 dont care's at the end
{
    clock.low();
    Thread.sleep(10);
    clock.high();
    Thread.sleep(10);
    clock.low();
}
    Thread.sleep(10);
    dataPin.low();

for(int i = 2; i < 3; i++)//12 since there are 2 dont care's at the end
{
    clock.low();
    Thread.sleep(10);
    clock.high();
    Thread.sleep(10);
    clock.low();
}

```

```

        Thread.sleep(10);
        dataPin.high();

for(int i = 3; i < 4; i++)//12 since there are 2 dont care's at the end
{
    clock.low();
    Thread.sleep(10);
    clock.high();
    Thread.sleep(10);
    clock.low();
}
    Thread.sleep(10);
    dataPin.low();

for(int i = 4; i < 5; i++)//12 since there are 2 dont care's at the end
{
    clock.low();
    Thread.sleep(10);
    clock.high();
    Thread.sleep(10);
    clock.low();
}
    Thread.sleep(10);
    dataPin.high();

for(int i = 5; i < 12; i++)//12 since there are 2 dont care's at the end
{
    clock.low();
    Thread.sleep(10);
    clock.high();
    Thread.sleep(10);
    clock.low();
}
    Thread.sleep(10);
    dataPin.low();
    chipSelect.high();
}
}

}

```

```

private void clockDAC(int start, int end, GpioPinDigitalOutput clock) throws
InterruptedException
{

```



```

        for(int i = start; i< (end + 1); i++)//12 since there are 2 dont care's at the end
        {
            clock.low();
            Thread.sleep(10);
            clock.high();
            Thread.sleep(10);
            clock.low();
        }
    }
}

```

laserTask.java

```

import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinState;
import com.pi4j.io.gpio.RaspiPin;

public class laserTask{
    GpioController gpio;
    GpioPinDigitalOutput bit0;
    GpioPinDigitalOutput bit1;

    public laserTask(GpioController g){
        gpio = g;
        bit0 = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_06, "Bit0", PinState.LOW);
        bit1 = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_07, "Bit1", PinState.LOW);
    }

    public void operate(String command) throws InterruptedException{

        if(command.equals("l_wall")){ //if string action == Wall Action
            bit0.low();
            bit1.high();
        } else if(command.equals("l_floor")){ // if string action == Floor Action
            bit0.high();
            bit1.low();
        } else if (command.equals("l_mix")){ //if string action == Mix Action
            bit0.high();
            bit1.high();
        } else { //OFF

            bit0.low();
            bit1.low();
        }
    }
}

```

```
}  
}
```

camera.bash

```
#!/bin/bash  
raspivid -t 999999 -w 960 -h 540 -fps 25 -b 5000000 -vf -o - | ffmpeg -i - -vcodec copy -an -r 25 -f  
flv -metadata streamName=myStream tcp://0.0.0.0:6666
```

index.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"  
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
  <title>CatErcise365 Video Site</title>  
  
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
  <meta name="description" content="Web page that displays ASCII art." />  
  <meta name="keywords" content="web programming, CSE 190 M, Washington, UW,  
computer science, Marty Stepp, HTML, XHTML, JavaScript" />  
  
  <meta http-equiv="Cache-Control" content="no-cache" />  
  <meta http-equiv="Pragma" content="no-cache" />  
  <meta http-equiv="Expires" content="0" />  
  
  <link href="style.css" type="text/css" rel="stylesheet" />  
  <script src="http://ajax.googleapis.com/ajax/libs/prototype/1.6.1.0/prototype.js"  
type="text/javascript"></script>  
  
</head>  
  
<body>  
  <h1>CatErcise365</h1>  
  <a href="/control.php"> Control Panel</a>  
  <h2>Cat Video</h2>  
  <div>  
    <div id="video-jwplayer_wrapper" style="position: relative; display: block; width: 960px;  
height: 540px;">  
      <object type="application/x-shockwave-flash" data="/jwplayer/jwplayer.flash.swf"  
width="100%" height="100%" bgcolor="#000000" id="video-jwplayer" name="video-jwplayer"  
tabindex="0">  
        <param name="allowfullscreen" value="true">  
        <param name="allowscriptaccess" value="always">  
        <param name="seamlesstabbing" value="true">  
        <param name="wmode" value="opaque">  
      </object>  
    <div id="video-jwplayer_aspect" style="display: none;"></div>
```

```

    <div id="video-jwplayer_jwpsrv" style="position: absolute; top: 0px; z-index: 10;"></div>
</div>

<script src="/jwplayer/jwplayer.js"></script>
<script type="text/javascript">
    jwplayer('video-jwplayer').setup({
        flashplayer: "/jwplayer/jwplayer.flash.swf"
        , file: "rtmp://" + window.location.hostname + "/flvplayback/flv:myStream.flv"
        , autoStart: true
        , rtmp: {
            bufferlength: 0.1
        }
        , deliveryType: "streaming"
        , width: 960
        , height: 540
        , player: {
            modes: {
                linear: {
                    controls: {
                        stream: {
                            manage: false
                            , enabled: false
                        }
                    }
                }
            }
        }
        , shows: {
            streamTimer: {
                enabled: true
                , tickRate: 100
            }
        }
    });
</script>
</div>
</body>
</html>

```

control.php

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>CatErcise365 Control Panel</title>
    <link href="style.css" type="text/css" rel="stylesheet" />

```

```

</head>

<body>
  <h1>CatErcise365 Control Panel</h1>
  <h2>Control buttons:</h2>
  <h3>Camera:</h3>
  <div>
    <button id="video_on" onclick="alert('Camera is
ON!');window.location='write.php?data=v_on';"><img src=on.png></button>
    <button id="video_off" onclick="alert('Camera is
OFF!');window.location='write.php?data=v_off';"><img src=off.png></button>
  </div>
  <h3>Laser:</h3>
  <div>
    <button id="wall" onclick="alert('Laser pattern is
WALL!');window.location='write.php?data=l_wall';"><img src=wall.png></button>
    <button id="floor" onclick="alert('Laser pattern is
FLOOR!');window.location='write.php?data=l_floor';"><img src=floor.png></button>
    <button id="mix" onclick="alert('Laser pattern is
MIX!');window.location='write.php?data=l_mix';"><img src=mix.png></button>
    <button id="laser_off" onclick="alert('Laser is
OFF!');window.location='write.php?data=l_off';"><img src=off.png></button>
  </div>
  <div>
    <button id="up" style="position:fixed; left: 360px; top: 60px;"
onclick="window.location='write.php?data=m_up';"><img src=Up.png></button>
    <button id="down" style="position:fixed; left: 360px; top: 210px;"
onclick="window.location='write.php?data=m_down';"><img src=Down.png></button>
    <button id="left" style="position:fixed; left: 260px; top: 135px;"
onclick="window.location='write.php?data=m_left';"><img src=Left.png></button>
    <button id="right" style="position:fixed; left: 460px; top: 135px;"
onclick="window.location='write.php?data=m_right';"><img src=Right.png></button>
    <button id="sound" style="position:fixed; left: 360px; top: 135px;"
onclick="alert('sound');window.location='write.php?data=s_on';"><img
src=Sound.png></button>
  </div>
  <h3>Speed:</h3>
  <div>
    <button id="speed1" onclick="window.location='write.php?data=m_1';"><img
src=f1.png></button>
    <button id="speed2" onclick="window.location='write.php?data=m_2';"><img
src=f2.png></button>
    <button id="speed3" onclick="window.location='write.php?data=m_3';"><img
src=f3.png></button>
  </div>
</body>

```

```
</html>
```

write.php

```
<html>
<body>
<?php $file = '/home/pi/catercise/control/control.txt';
$dold_content = file_get_contents($file);
    file_put_contents($file, $dold_content.$_GET["data"]."\n");
?>
<script>
window.location='/control.php';
</script>
</body>
</html>
```

style.css

```
h1{
    font-family:Tahoma, Verdana, sans-serif;
    font-size:24pt;
    text-align: center;
    color:#CC99FF;
}
h3{
    font-family:Comic Sans, Verdana, sans-serif;
    font-size:16pt;
    text-align: left;
}
legend{
    font-family:Comic Sans, Verdana, sans-serif;
    font-size:14pt;
    text-align: left;
}
```

PIC18F25K22 Laser Code:

```
/*
 * File:  main.c
 * Author: serena18
 *
 * Created on November 22, 2013, 5:43 PM
 */
```

```
#include <stdio.h>
#include <stdlib.h>
#include "pwm.h"
#include <p18f25k22.h>
#include <delays.h>
```

```
/*
```

```

*
*/
#pragma config FOSC = INTIO67
#pragma config PLLCFG = OFF
#pragma config BOREN = OFF
#pragma config WDTEN = OFF
#pragma config MCLRE = EXTMCLR
#pragma config LVP = OFF
#pragma config XINST = OFF

void main(void)
{
    int count=0;
    TRISAbits.RA1=1;
    TRISAbits.RA2=1;
    ANSELA=0x00;
    TRISAbits.RA3=1;

    while(1)
    {
        if(!PORTAbits.RA2 && !PORTAbits.RA1) //turn off everything
        {
            TRISCbits.RC1=1;
            PORTCbits.RC1=1;
            TRISCbits.RC2=1;
            PORTCbits.RC2=1;
            PORTC=0x00;

        }
        else
        {
            TRISC=0b00000000;
            PORTC=0xFF;
            PR2=124;
            PR4 = 50 ;
            T2CON = 0b00000111 ;
            T4CON = 0b00000111 ;
            CCP1CON = 44;
            // CCPR1L = 66;
            CCP2CON = 44;

            if(!PORTAbits.RA2 && PORTAbits.RA1) //wall
            {
                // WALL code
                if(count ==5)
                {
                    CCPR1L = 30;
                    CCPR2L = 30;
                    Delay10KTCYx(5);
                    CCPR1L = 30;
                }
            }
        }
    }
}

```

```

        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 35;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 35;
        CCPR2L = 10;
        Delay10KTCYx(5);
        CCPR1L = 25;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 25;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 30;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 30;
        CCPR2L = 21;

    }else if(count==10){
        CCPR1L = 30;
        CCPR2L = 21;
        Delay10KTCYx(15);
        count=0;

    }else{

        CCPR1L = 30;
        CCPR2L = 28 ;
        Delay10KTCYx(5);
        CCPR1L = 30;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 35;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 35;
        CCPR2L = 10;
        Delay10KTCYx(5);
        CCPR1L = 25;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 25;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 30;
        CCPR2L = 21;
        Delay10KTCYx(5);
        CCPR1L = 30;
        CCPR2L = 21;
    }
}

```

```

        count++;
    }

}
else if(PORTAbits.RA2 && !PORTAbits.RA1) // floor
{
    if(count==5)
    {
        CCPR1L = 37;
        CCPR2L = 21;
        Delay10KTCYx(33);
        CCPR2L = 21;
        Delay10KTCYx(33);
        count=0;

    }else{

        CCPR1L = 37;
        CCPR2L = 23;
        Delay10KTCYx(33);
        CCPR2L = 19;
        Delay10KTCYx(33);
        CCPR2L = 23;
        Delay10KTCYx(10);
        CCPR2L = 19;
        Delay10KTCYx(10);

        CCPR2L = 23;
        Delay10KTCYx(33);
        CCPR2L = 23;
        Delay10KTCYx(10);
        CCPR2L = 19;
        Delay10KTCYx(10);
        CCPR2L = 19;
        Delay10KTCYx(33);
        count++;

    }

}
else if(PORTAbits.RA2 && PORTAbits.RA1) //mix
{
    CCPR1L = 30;
    CCPR2L = 23;
    Delay10KTCYx(11);
    CCPR1L = 35;
    Delay10KTCYx(11);

    CCPR1L = 25;
    Delay10KTCYx(11);

```



```

        CCPR2L = 19;
        CCPR1L = 30;
        Delay10KTCYx(11);
        CCPR1L = 35;
        Delay10KTCYx(11);
        CCPR1L = 25;
        Delay10KTCYx(11);

    }
    else // turn off everything
    {
        PORTC=0x00;
        CCPR1L = 35;
        CCPR2L = 21;

    }
}
if(! PORTAbits.RA3 && !PORTAbits.RA2 && !PORTAbits.RA1)
{
    TRISC=0b00000000;
    PORTC=0xFF;
    PR2=124;
    PR4 = 50 ;
    T2CON = 0b00000111 ;
    T4CON = 0b00000111 ;
    CCP1CON = 44;
    // CCPR1L = 66;
    CCP2CON = 44;
    if(count==5)
    {

        CCPR1L = 37;
        CCPR2L = 21;
        Delay10KTCYx(33);
        CCPR2L = 21;
        Delay10KTCYx(33);
        count=0;

    }
    else{

        CCPR1L = 37;
        CCPR2L = 23;
        Delay10KTCYx(33);
        CCPR2L = 19;
        Delay10KTCYx(33);
        CCPR2L = 23;
        Delay10KTCYx(10);
        CCPR2L = 19;
    }
}

```

```

        Delay10KTCYx(10);

        CCPR2L = 23;
        Delay10KTCYx(33);
        CCPR2L = 23;
        Delay10KTCYx(10);
        CCPR2L = 19;
        Delay10KTCYx(10);
        CCPR2L = 19;
        Delay10KTCYx(33);
        count++;
    }
}
}

```