Joshua Waugh

Title of the Thesis

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor’s Thesis

Date

|  |  |  |
| --- | --- | --- |
| Author  Title  Number of Pages  Date | First name Last name  Title of the Thesis  xx pages + x appendices  21 August 2017 | |
| Degree | Bachelor of Engineering | |
| Degree Programme | Information Technology | |
| Professional Major | Smart Systems | |
| Instructors | First name Last name, Title (for example: Project Manager)  First name Last name, Title (for example: Principal Lecturer) | |
|  | | |
| Keywords | |  |

Contents

List of Abbreviations

[1 Introduction 2](#_Toc491951623)

[2 Chapter Heading 2](#_Toc491951624)

[2.1 Subheading 2](#_Toc491951625)

[2.2 Subheading 2](#_Toc491951626)

[2.2.1 Subheading 2](#_Toc491951627)

[2.2.2 Subheading 2](#_Toc491951628)

[References 2](#_Toc491951629)

Appendices

Appendix 1. Title of the Appendix

Appendix 2. Title of the Appendix

List of Abbreviations

ORM Object-relational mapping. The set of rules for mapping objects in a programming language to records in a relational database, and vice versa.

DBMS Database management system. Software for maintaining, querying and updating data and metadata in a database.

PSoC Programmable System-On-Chip

LED Light Emitting Diode

# Introduction

This report is focused on the project of the first year Smart Systems Course for Information Technology students at Metropolia University or Applied Science. The project involved programming a robot to complete three tasks: sumo wrestling, line following and maze navigation. Each member of the group working on this task has no experience with programming prior to this project. As such, the task was approached largely as a learning exercise.

The project was approached by finding solutions to smaller side projects which would be transferrable to the main task. When discussing the final tasks in this report, it will be as the sum of the solutions to these tasks.

# Materials and Software

## PSoC® Creator™ Integrated Design Environment (IDE)

PSoC Creator is a program created by Cypress Semiconductor for writing programs for and compiling to PSoC hardware. This software was used for all code written during this project.

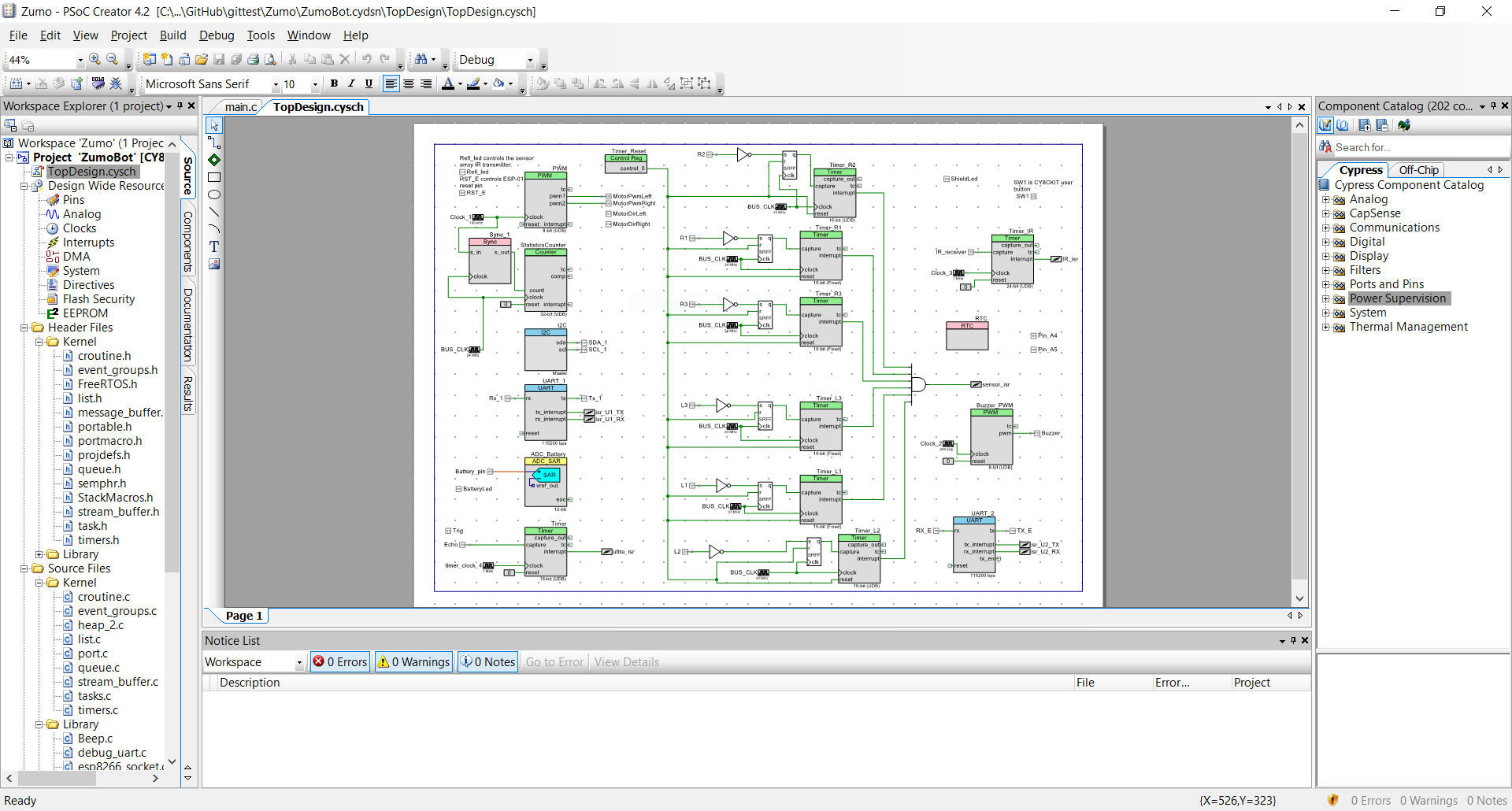


Figure 1

## CY8CKIT-059 PSoC® 5LP Prototyping Kit

The CY8CKIT-059 PSoC 5LP prototyping kit from Cypress Semiconductor was used to control the Zumo robot. Located on the PSoC is a USB connection, an LED, a button to reset the program and a programmable button that can execute programmed commands. Code was compiled and written to the PSoC by connecting it to a computer via USB and using PSoC Creator 4.2.



Figure 2.

## Polulu Zumo Robot

The Zumo robot hardware can be broken down into the following individual components: motors, 6 reflectance sensors, an accelerometer, an ultrasonic sensor, an infrared sensor and a Wi-Fi chip. A router acting as an MQTT broker is used to communicate between our computers and the robot. There is also a programmable button on the PSoC hardware itself. The libraries for controlling these components have been generously provided by Metropolia University of Applied Science.

PuTTY was used to see the output of the code written to the PSoC.

Zotero was used to track any sources used during the course of this project.

The project involved programming the robot so that it could perform 3 different tasks: fight in a sumo arena, follow a line and navigate the maze. Each task can be broken down into components that are common between all 3. These tasks were approached on a weekly basis in side projects. They are as follows: Following a line, collision detection, obstacle detection

# Method

HERE WE OUTLINE THE DIFFERENT THINGS WE WANT THE ROBOT TO DO. Eg line following,

The project involved solving smaller weekly tasks which allowed us to familiarize with the hardware. Each task provided some kind of benefit to the final project.

3.1 Starting and Stopping the Robot

The PSoC begins executing its code as soon as the robot is powered on. To prevent it from immediately driving away after being placed down, it was held in place by entering an infinite loop that is broken out of by pressing the button or by sending an infrared signal via remote.

The robot can similarly be stopped by these two inputs, however for several tasks during the project it was more sensible to designate a goal and to break the motor function loop when the goal condition is met. This usually meant whenever the robot senses a horizontal finish line.

## Line Sensing

The line following algorithm used throughout this project was a simple if/else statement. Additionally, a ratio was found between the two middle sensors so that as one started seeing less black than the other, the speed of the motor on the same side could be slowed down to bring the robot back to centre. Line sensing also involves recognizing a horizontal black line. The maps the robot needs to drive on are designed in such a way that the only time the robot should see black on both of its outside sensors (L3 and R3) is at a point it needs to stop or start at.

## Detecting objects

The robot can detect objects ahead of it using an ultrasonic sensor. Careful consideration was made when using the sensor as its reliability rapidly decreases as objects get very near and very far. It is necessary to detect objects in the maze task so that the robot can navigate around 4 wooden blocks placed on the grid. The below figure outlines when the robot is asked to search for obstacles and how it avoids them.

Obstacle detection was initially considered for the Sumo Wrestling task also; however it was ultimately decided that it provided no benefit.

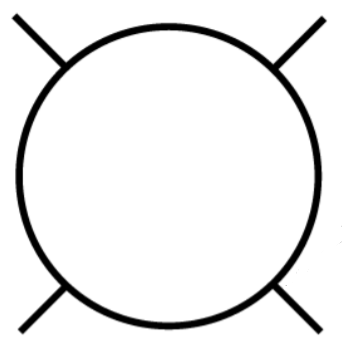
## Detecting changes in acceleration

The robot can detect changes in acceleration using an accelerometer, most notably from a collision with an obstacle or another robot. Initially it was presumed that the robot should change its behavior during Sumo Wrestling if a collision was recorded. However, this seemed unlikely to improve the robot’s chance at victory. As such, the robot knows when it has been hit, but its only reaction is printing a message that it has been hit.

# Results

## Sumo Wrestling

The arena for this task can be seen in the figure below.



The Sumo wrestling competition involves robots being placed at three of the lines perpendicular to the circle. After pressing the button and powering on the motors, the robot approaches the edge of the arena. Here the robot and its opponents all wait for an infrared signal, at which point they enter the battle.

Initially it was presumed that the robot should react when it has been struck by an opponent. After testing, it was found that too many factors were involved to reliably improve our chances of victory with a simple victory. As such when the robot is in the ring, it will simply drive in a straight line until it sees the edge, turn around for some period of time and then repeat.

The robot must know the time when it has reached the edge of the ring, when it has received a signal from the remote, when it has been struck by an opponent and when it has been powered off.

The flowchart of the sumo wrestling code looks like this:

## Line Following Race

The line following project was a tournament between all teams to see whose robot is the fastest. Each race was between two robots on two identical 10-meter tracks. The race would begin after the robots had approached the starting line and been given the infrared signal. Timestamps were recorded at the starting line and the finish line and the difference between the two would be the robot’s completion time.



## The Maze

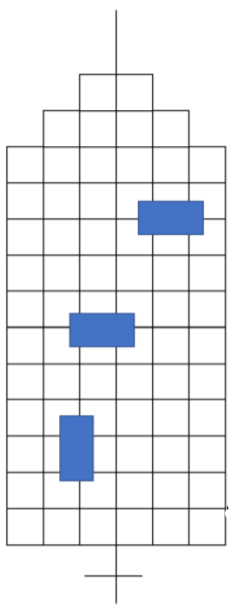
The maze is a grid of 14 rows and 7 columns. Four wooden blocks were placed at random locations around the grid between rows 3 and 10. The problem was approached with a co-ordinate system, with each intersection on the grid being a number in an array of equal size to the grid. Then variables were created to hold the robot’s current direction and position. The robot’s direction was initialized as 0 and would decrement by 1 on a left turn or increment by 1 on a right turn. However, the value would never exceed -1 or 1 as it was assumed the robot would never have to turn backwards towards the entrance of the maze.

As the robot entered each intersection, it would update its position based on the direction it was facing. If it was facing forwards when it entered the intersection, then it must be on a new row. If it was facing sideways, then it must be on a new column. When a wooden block was detected by the ultrasonic sensor, the robot would update that intersection to be inaccessible and turn to avoid it. The direction of the turn depended on the robot’s current position, with it always prioritizing returning to the centre. However, if its path to the centre was also blocked, it would turn 180 degrees and find a clear path in that direction.

Between intersections the robot would only execute line following code. At each intersection, it would go through these steps:

Update the robot’s x or y coordinate, depending on the direction it was facing when entering the intersection. Check to see if there is an obstacle in the intersection directly in front of the robot. If there is no obstacle, continue driving forward. If there is an obstacle, check to find a pathway on the array.

Additional logic was required to solve this project, beyond what had been solved in the weekly tasks.



2.2 The components

### Following a line

Lines are followed by using the Zumo robot’s infrared reflectance sensors. The sensors can return a RAW number or a digital number that is flipped depending on some threshold. An if/else statement was constructed that influenced the speed of the motors based on the sensor readings. The below figure illustrates the conditions considered for this task and the reactions to them.

There must always be text or a new subheading below each heading. Do not place a figure or table below a heading with no text in between.

Label each figure and table appropriately. Provide a number, title and reference (if needed) below each figure and above each table. Make sure to mention all figures and tables in the text. Each figure and table must be explained in the text and referred to by its number (… as figure 1 illustrates. /as summarized in table 1.).

Apply the Figure style for each image. This is necessary in order to prevent a page break from occurring between the figure and its caption. The Figure caption style is applied for the figure’s caption. This causes the figures to be numbered automatically.

1. Virtual studies completed by Metropolia students in the academic year 2009-2010.

There must always be text between a figure or table and a new heading.

## Subheading

### Subheading

There must always be text or a new subheading below each heading. Do not place a figure or table below a heading with no text in between.

1. Virtual studies completed by Metropolia students in the academic year 2009-2010.

|  |  |
| --- | --- |
| Field of study | Studies completed, ECTS |
| Culture | 131 |
| Technology, Communication and Transport | 552 |
| Health Care and Social Services | 175 |
| Business and Administration | 52 |
| Not bound to a field of study | 18 |
| Metropolia total | 928 |

There must always be text between a figure or table and a new figure or table or a new heading.

### Subheading

There must always be text or a new subheading below each heading.

Use the Quotation style for an indented quotation. For the last paragraph immediately before the quotation, use the Body Text before Quotation or List style.

If a direct quotation is several lines long, indent the quotation and use single (1.0) line spacing. No quotation marks are used then. Always provide a reference to the source. If the direct quotation is shorter than two lines, include it in the body of the text in quotation marks, and provide a reference to the source.

After an indented direct quotation, continue the text at the left margin using the Body text style.

Use the Bulleted list style for an in-text list:

* This is the first list item.
* The second item of the list contains a long text that spans more than one row. The left margin will be automatically justified.
* This is the third list item.
* This is the fourth list item.

The items on the bulleted list begin with a capital letter. An item ends in a full stop if each item on the list is a full sentence.

The list items begin with a lower-case letter if the list items are not sentences. The last item is followed by a full stop. Thus, a thesis consists of

* words
* sentences
* paragraphs
* sections.

After the list, the text continues from the left margin in the Body text style.

You can insert numberChapter Heading

References

Details of the references are given here. Use the referencing system required in your degree programme or as agreed with your supervisor.

Layout of this page in the author-date (Harvard) referencing system:

Details of the reference Details of the reference Details of the reference Details of the reference Details of the reference.

Details of the reference Details of the reference Details of the reference Details of the reference Details of the reference.

Layout of this page in the number (Vancouver) referencing system:

1. Details of the reference Details of the reference Details of the reference Details of the reference Details of the reference.
2. Details of the reference Details of the reference Details of the reference Details of the reference Details of the reference.

**Title of the Appendix**

The contents of the appendix are placed here. Below are the instructions for removing and adding appendices in a way that maintains the headers and footers in their correct form.

Instructions for removing an unwanted appendix:

1. Select the entire page(s) that form the appendix and delete the contents by hitting the Delete key.
2. As you are in the beginning of the empty appendix page (see the image below), double-click the header of the empty page and press Link to Previous button in the ribbon. The following dialogue window opens:



Click Yes.

1. If necessary, make hidden format information visible by pressing .
2. Delete the section break immediately before the appendix to be removed (see image below).



Instructions for adding a new appendix:

1. Place the cursor at the end of the last appendix.
2. Select Page Layout from the menu bar. From the ribbon, select Breaks/Section Breaks/Next Page. This causes a new appendix to appear, but the appendix number in the header is not yet correct.
3. Double click the header of the new appendix with the wrong appendix number. If the option “Link to Previous” is selected, click the corresponding button to deselect it.
4. Replace the appendix number with the correct number.

Note that the appendices need to be updated in the table of contents manually.

**Title of the appendix**

The contents of the appendix are placed here.