INTEGRATED DESIGN PROJECT (IDP)

FIRST REPORT

M105

Tiger Woods’ Ball Handlers

Tiger Woods’ Ball Handler

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **CRSid** | **Lab Group** | **College** | **Sub-Team** |
| Dan Hyman |  | 65 | Selwyn | Mechanical |
| Sam Chapman |  |  |  | Mechanical |
| Cheryn Ng | cn397 |  | Magdalene | Electrical |
| Nathan Zhu | hz325 |  | Magdalene | Electrical |
| Karthik Suresh | ks800 | 56 | Wolfson | Software |
| Jeremi Miller  (Coordinator) | jrm224 | 65 | Selwyn | Software |

**Table Of Contents**

[**Problem Statement**](#_8nomup8erdn1) **3**

[**Requirement Specification**](#_lszqg981q5bm) **3**

[**Concept Evaluation**](#_y5e9zwd2e0qo) **3**

[**Integration Specifications**](#_rjl7nmm763ys) **3**

[**Project Management**](#_lu0euww46lxs) **3**

[**Project Timetable**](#_im48w12znb0j) **3**

[**Work Allocation**](#_r2a86rf7wxao) **3**

[**Calculations for Design Decisions**](#_uhlkcj8nyiol) **3**

[**Mechanical**](#_lq6g4xna0vd7) **3**

[**Drawings**](#_69lhub8oi6h6) **3**

[**Initial Experiments**](#_ki6kt51o5kw4) **3**

[**Initial BoM**](#_u5h9kz51twb6) **3**

[**Electrical**](#_mvup52p7sq2t) **4**

[**Initial Experiments**](#_wndqv1m9xk8s) **4**

[**List of Components**](#_nuq34l437nqe) **4**

[**Hardware-vs-Software Tradeoff**](#_i3rkwfun8z32) **4**

[**Initial BoM**](#_kft6qabl2wz2) **4**

[**Software**](#_eoti4d9hqiiz) **5**

[**Initial Experiments**](#_5hv0rx3lucmz) **5**

[**Software Layout**](#_7vrwjcs1py2r) **5**

[**Timing Issues**](#_dzh1yj26afac) **5**

[**Recovery Strategies**](#_ekbsivoypioa) **5**

[**Initial BoM**](#_za212dpfcx7t) **5**

# **Problem Statement**

Solution neutral problem statement

# **Requirement Specification**

The requirements should be quantified whenever possible

# **Concept Evaluation**

Sketches of concepts + evaluation charts + brief description of ads & disads

# **Integration Specifications**

Specifications of the interfaces between subsystems & outline of the framework for their integration

|  |  |
| --- | --- |
| **Interface** | **Key Integration Considerations** |
| Software + Electrical | * A decision was made to process the outputs of all sensors used for the robot as binary signals and not to use analogue readings. This was for the sake of simplicity and optimal processing time for both software and electrical teams. * The output from the line following IR sensors and distance sensors is passed directly to the operating program. These signals are first processed within the program’s action functions. * The output from the two LDRs and micro switch will first be processed through logic circuits to determine which type of ball has been picked up by the robot at a given time. This will then allow the identifying LEDs on the robot to output results without the intervention of software. Furthermore, this saves the time that would have been expended had the signals of each sensor been passed to the operating program to determine the type of ball picked up. * The software and electrical teams need to carry out tasks concurrently and ensure that at any given time both teams are working towards the same function section of the robot (e.g. line following circuit should be constructed while software team writes code for line following algorithm). This will ensure that there is minimal delays to test key functions of the robot. |
| Software + Mechanical | * The mechanical design of the robot influences the sensitivity factors that are coded within the robot’s operating program. The construction of the robot should be concluded more than a week before the competition date to give the software team sufficient time to ensure the program’s functions have suitable parameters for the robot to complete its tasks precisely. * The configuration of the IR sensors, delivery strategy and wheel numbers/types/positions were communicated between the software and mechanical teams to ensure that the action functions for the robot could be written appropriately. For example, it was decided that the robot should pick-up all six balls and then successively deliver them one by one to their respective bins. This requires the operating program to have a function that can calculate the critical path of the robot based on the order and types of balls it is carrying. |
| Mechanical + Electrical |  |

# **Project Management**

# **Project Timetable**

(Put Gantt Chart here)

# **Work Allocation**

(Explain Gantt Chart here, the work is allocated in there)

# **Calculations for Design Decisions**

# **Mechanical**

# **Drawings**

# **Initial Experiments**

# **Initial BoM**

See Appendix A

# **Electrical**

# **Initial Experiments**

Initial experiments were performed to familiarise

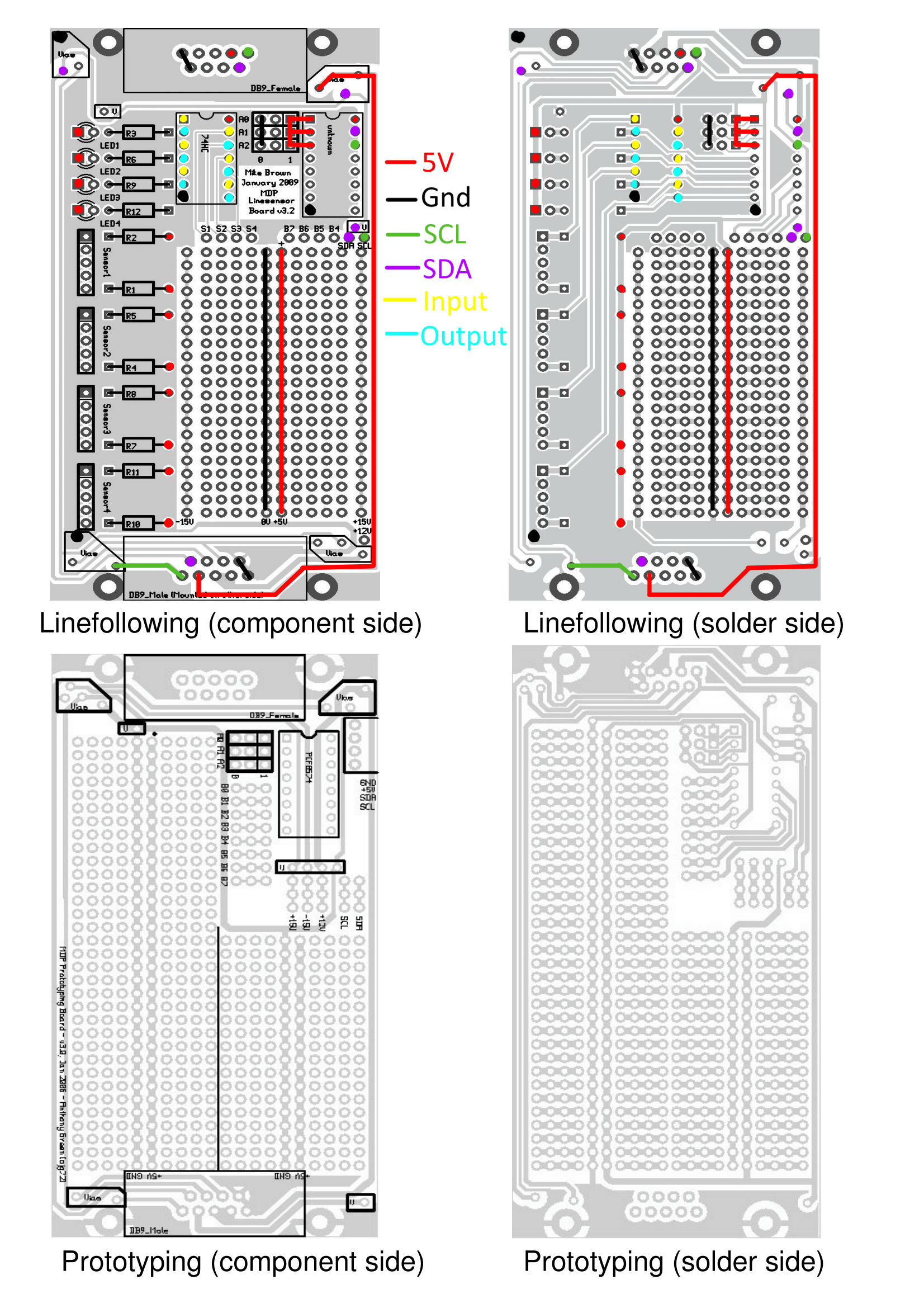
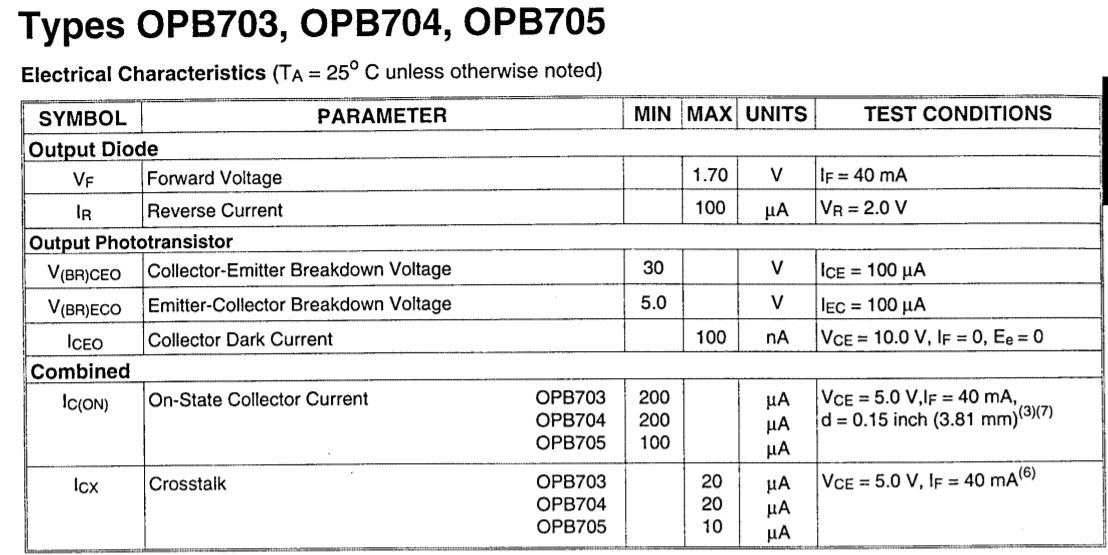
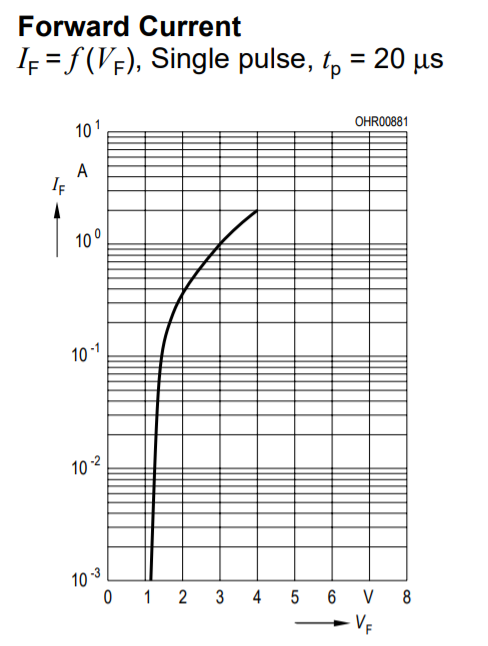
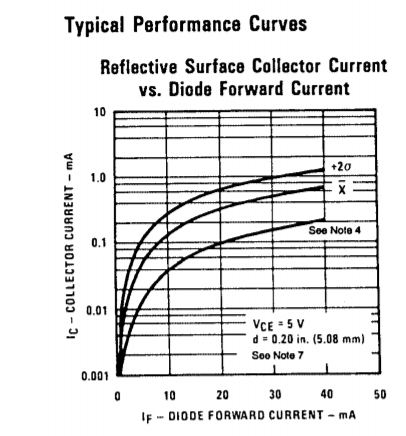


Fig. **X** Annotated Circuit Diagram of Line Sensor PCB

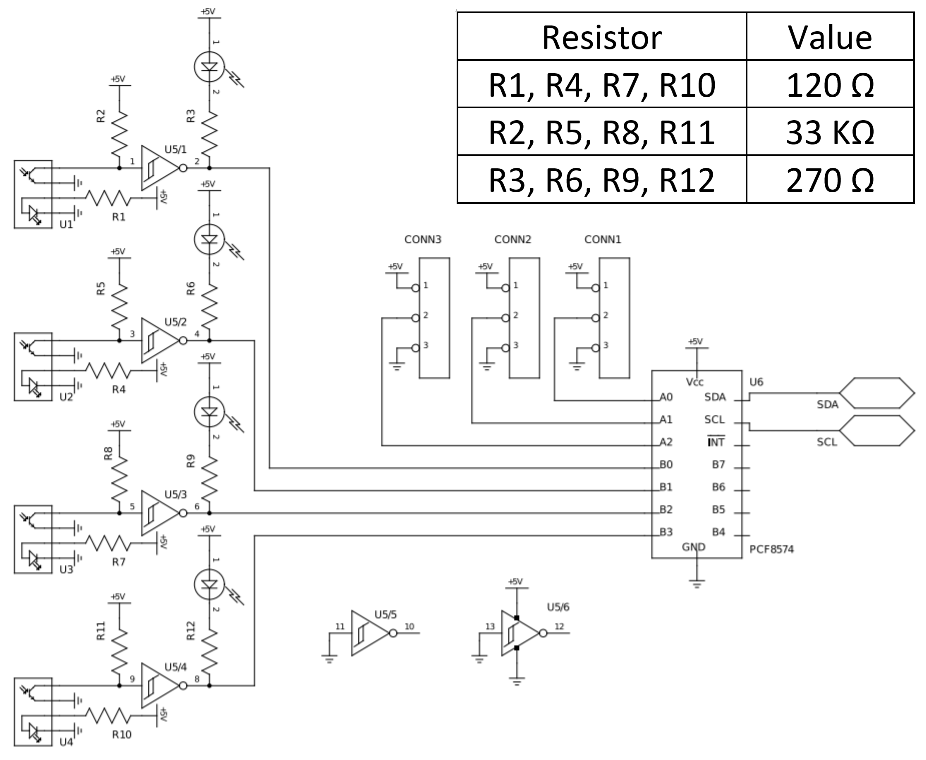
Component Layout Exercise 1-4

Exercise 5



Based on the datasheet, the max operating point for the built-in LED is VF=1.7V, IF=40mA, IC=20uA.

Using the test box, 33kΩ and 35mA gives the best reading. 33kΩ corresponds to R2, while 35mA is the current that flows through LED. Operating point of LED at 35mA is 1.3 V, so voltage across R1 is 3.7 V. Resistor R1 = 3.7 / 35mA = 105.7 Ω. 120 Ω is used as a safety factor.



# **List of Components**

# **Hardware-vs-Software Tradeoff**

# **Initial BoM**

See Appendix B

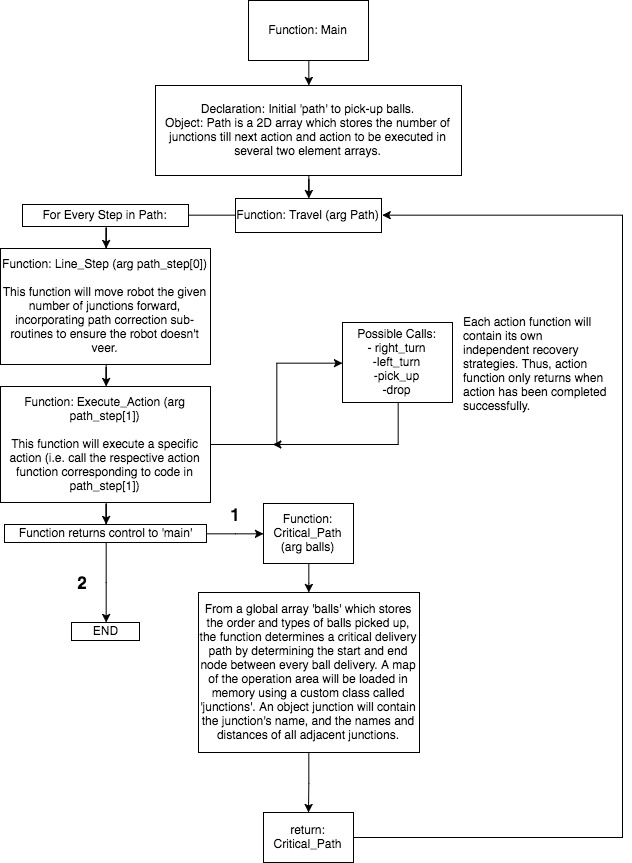
# 

# **Software**

# **Initial Experiments**

* All initial tests relating to: establishing communication links with the robot; correctly reading the output signals of the sensors through the PCF8574 chip; and sending operating commands to various motors, were completed successfully.
* The test to determine the time taken to send and receive signals from the microprocessor revealed that the variance in results was greater when the program was run remotely through the workstation compared with when it was run locally on the microprocessor. In the first case the time taken for a 100 instructions varied between 105ms and 10000ms. In the second case the result was between 200 and 300ms.
* In the first week, a prototype for the line following algorithm and turning function was coded, and the test robot was used to test the initial code. From the physical LED’s on the robot, reflecting the output of all the three IR sensors at any given time, it could be seen that the robot could detect junctions and identify when it was veering off a straight path. In some cases the software was able to record the number of junctions crossed and execute the correction function in the case of veering, however it was unable to do so precisely in most of the tests. Test sequences reveal that the data transmission between the robot and software is lossless, accurate and without delay suggesting the prototype algorithm needs revision.

# **Software Layout**



# 

# **Timing Issues**

* In our initial test of the line following algorithm, one of our main sources of error was coordinating timing between the robot’s motion and the software’s execution. For example, a loop in the program would execute too quickly thus registering a single junction as crossing several junctions. This will need to be rectified by placing time restrictions at appropriate points within the code.

# **Recovery Strategies**

* Regarding line following, our main idea behind recovery is prevention. We are spending most of our time working on and extensively testing the line following function, which has a ‘veering’ function built in, which corrects any possible mistakes. The plan is to perfect it, so that the actual recovery function does not have to be called at all. The recovery function is called when no sensors are on the line. The idea is to make the robot go backwards until it finds the line with one of its front sensors and then position itself on the line. However, this might prove to be costly timewise and difficult to rely on, so in case where the problem occurred before having dropped off any balls, we think that restarting would be a better idea.

# **Initial BoM**

See Appendix C