

**e-Yantra Robotics Competition - 2015**

**Implementation Analysis – Pizza Delivery Service**

**<Team ID>**

|  |  |
| --- | --- |
| **Team leader name** | Pranjal Aswani |
| **College** | Thakur College of Engineering and Technology |
| **Email** | aswani.pranjal@gmail.com |
| **Date** | 11/1/2016 |

**Preparing the Arena (5)**

<Prepare the arena according to the steps given in Section 3: Arena, of the rulebook. Take a photo of the completed arena such that the entire arena is clearly visible in the photo. Insert the image here>

**Design Analysis**

**Q-1. What do you understand by “Internal Timer”? and what is the significance of it in this theme. (5)**

An internal timer is a timer controlled by the microcontroller using interrupts in order to separate the execution of time-sensitive code against code that runs and evaluates the functions of the bot that allow it to move, make decisions, and understand its surroundings.

The internal timer is very important because the bot needs to accurately represent how much time it has left for any given order, and the order timeline's solution is optimized using this time. The bot will be scored based on this time too, so it is paramount to ensure that it is reliable.

**Q-2. Make a video in which time displays from 000 to 030 seconds on the Seven Segment Module. (10)**

[All files required to compile task 2's code are available in this zip.](https://drive.google.com/file/d/0B659He5qyintRHhCY00zUDloVVU/view?usp=sharing)

The C code is replicated below. The three files must be linked together and the final hex file is then burned to the atmega2560 processor. This is done using the Makefile(s) seen in the zip.

**SevenSegment.c:**

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

#include "SevenSegment.h"

void seven\_segment\_pin\_config() {

// Port D upper nibble for CA connections

// Port J for a,b,c,d, e,f,g,DEC as per manual

// Both things above are in decreasing order

// i.e.

// Port J = Pin 7, 6, 5...0

DDRD = DDRD | 0xF0; // set upper nibble of port D to output

DDRJ = DDRJ | 0xFF; // set all bits to output

}

void seven\_port\_init() {

seven\_segment\_pin\_config();

}

void seven\_init\_devices() {

cli(); //Clears the global interrupt

seven\_port\_init(); //Initializes all the ports

sei(); // Enables the global interrupt

}

int seven\_convert\_to\_hex(int num) {

int ret = 0x00;

int values[10];

values[0] = 0x03;

values[1] = 0x9F;

values[2] = 0x25;

values[3] = 0x0D;

values[4] = 0x99;

values[5] = 0x49;

values[6] = 0x41;

values[7] = 0x1F;

values[8] = 0x01;

values[9] = 0x09;

if (num >= 0 && num <= 9) {

ret = values[num];

}

return ret;

}

void seven\_display\_num(int num) {

int digit;

int MIN\_SELECT\_VALUE = 0x10;

int MAX\_SELECT\_VALUE = 0x40;

int select\_value = MIN\_SELECT\_VALUE;

PORTD = PORTD & 0x0F; // Reset upper nibble to 0

PORTD = PORTD | select\_value;

while (select\_value <= MAX\_SELECT\_VALUE) {

digit = num % 10;

PORTJ = seven\_convert\_to\_hex(digit);

\_delay\_ms(5);

select\_value \*= 2;

if (select\_value <= MAX\_SELECT\_VALUE) {

PORTD = PORTD & 0x0F; // Reset upper nibble to 0

PORTD = PORTD | select\_value; // Set upper nibble value to select apt CA

}

num /= 10;

}

}

**timer.c:**

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

#include "timer.h"

#include "../buzzer/buzzer.h"

// TIMER4 initialize - prescale:1024

// WGM: 0) Normal, TOP=0xFFFF

// desired value: 1Hz

// actual value: 1.000Hz (0.0%)

void timer\_timer4\_init() {

TCCR4B = 0x00; // stop

TCNT4H = 0x1F; // Counter higher 8 bit value

TCNT4L = 0x01; // Counter lower 8 bit value

OCR4AH = 0x00; // Output Compair Register (OCR)- Not used

OCR4AL = 0x00; // Output Compair Register (OCR)- Not used

OCR4BH = 0x00; // Output Compair Register (OCR)- Not used

OCR4BL = 0x00; // Output Compair Register (OCR)- Not used

OCR4CH = 0x00; // Output Compair Register (OCR)- Not used

OCR4CL = 0x00; // Output Compair Register (OCR)- Not used

ICR4H = 0x00; // Input Capture Register (ICR)- Not used

ICR4L = 0x00; // Input Capture Register (ICR)- Not used

TCCR4A = 0x00;

TCCR4C = 0x00;

TCCR4B = 0x04; // start Timer

}

// This ISR can be used to schedule events like refreshing ADC data, LCD data

ISR(TIMER4\_OVF\_vect) {

// TIMER4 has overflowed

TCNT4H = 0x1F; // reload counter high value

TCNT4L = 0x01; // reload counter low value

timer\_1s\_magic();

}

void timer\_init\_devices() {

cli(); // Clears the global interrupts

timer\_timer4\_init();

TIMSK4 = 0x01; // timer4 overflow interrupt enable

sei(); // Enables the global interrupts

}

**task2.c:**

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

#include "../timer/timer.h"

#include "../SevenSegment/SevenSegment.h"

int count = 0;

void timer\_1s\_magic() {

count++;

if (count >= 30) {

count = 30;

}

}

int main() {

timer\_init\_devices();

seven\_init\_devices();

while(1) {

seven\_display\_num(count);

\_delay\_ms(1);

}

}

**Q-3. Draw a labeled diagram to explain how you have planned to place the sensors on/around the robot? (5)**

<Draw a neat diagram to show the positions of sensors on and around the robot. Show all the sensors you are using in designing the theme. Justify placement of the sensors shown in your diagram>

**Q-4. Teams have to make the robotic arm for picking up/placing the packages in the arena.**

1. **Choose an option you would like to use to position the robotic arm on the robot and why? (2)**

1. **Front 2. Back 3. Right/Left 4. On both sides**

We chose to mount the robotic arm on the **front** of the robot because of the following reasons:

* The bot cannot traverse sideways directly; it would need to stop in the correct position, and rotate 90 degrees if the arm was placed on either side. This would add wasteful time to the bot's deliveries in a time-sensitive competition.
* If we mount the arm on either sides, we risk toppling over other pizza boxes at the pizza store because of restrictions in space.
* We have mounted our sharp sensor on the side, which also restricts the space we have to mount the arm on the side.
* All the expansion board ports are near the back of the bot, which needs to be accessible to allow us to control components using the bot.
* If we mounted 2 arms, on both sides of the bot, we'd require more motors, ports, power and circuitry, in addition to more complex algorithms dealing with carrying multiple pizza boxes at the same time. This is in addition to the disadvantage of not being able to traverse sideways directly, and of risking toppling over pizza boxes.

All the reasons above leave us with a preference to mounting the arm on the front of the bot. This is only a slight preference over mounting it on the back, but it makes more intuitive sense to us.

**b. Draw a diagram to show the robotic arm and how it is mounted on the robot. Also show the mounting of the color sensor. (5)**

<Draw figure(s) to show how you are planning to mount the robotic arm and color sensor on the robot. >

**Q-5. Choose the actuator you will use to design the robotic arm. (3)**

1. **DC-Motor 2. Servo Motor 3. Stepper Motor 4. Others**

**Answer:** 2. Servo Motor

We have chosen the servo motor because they are the most accurate and reliable motors for the specific purpose of building an arm. A DC motor is an open-loop system, which means we'd require specific sensors (like position encoders) to track how far the motor had moved. A servo motor on the other hand, is a closed-loop system, and allows us to move the motor with the kind of precision required to control a robotic arm. We don't use stepper motors because stepper motors are generally very expensive, and they generally provide a lot more precision and torque than we require for the arm.

**Q-6**

**a. What is the principle of operation of the color sensor? (5)**

< Please explain the answer in your own words. The answers copied directly from tutorials/datasheet will not be considered for evaluation.

Answer format: Text

Word-limit: 150 words

>

**b. Explain frequency scaling. Why is it necessary? (5)**

< Please explain the answer in your own words. The answers copied directly from tutorials/datasheet will not be considered for evaluation.

Answer format: Text

Word-limit: 100 words

>

**c. How will you identify Red, Blue and Green colors from the values you get from the color sensor? Explain your algorithm to identify the three colors (Red, Blue and Green). (5)**

<

Answer format: Bulleted form

Step 1:

Step 2:

Step 3…etc.

>

**Please note: You have to burn the demo code on Firebird V (the robot in your kit) to test the color sensor. If you do not test the color sensor and report any problem in this document, we will assume that your color sensor is working fine. We will not entertain any queries related to faulty color sensor later on; Teams will have to buy the color sensor on their own.**

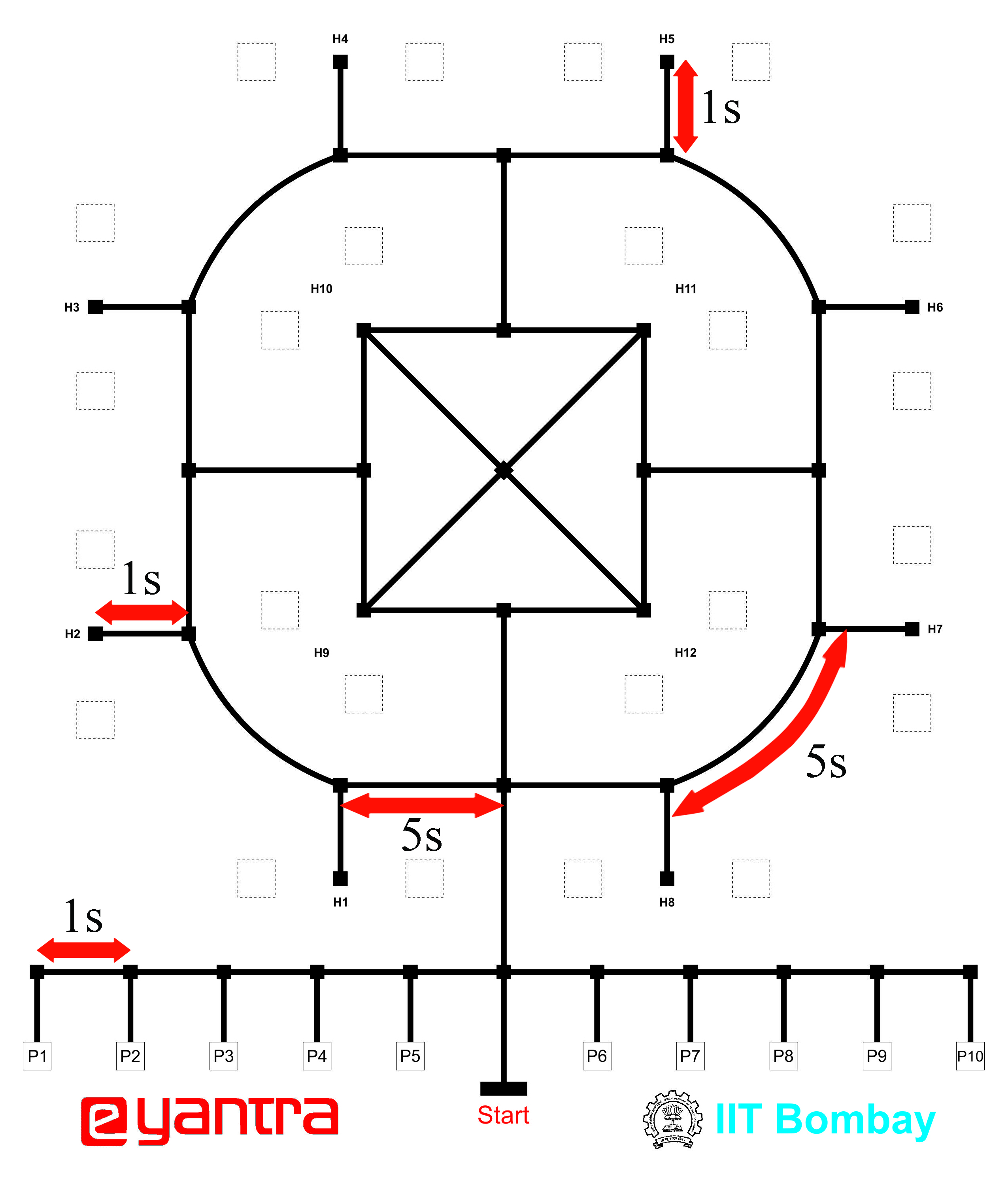
**Note: The timings mentioned below are purely hypothetical. Your robot timings may be shorter or longer than given below.**

**Consider the following assumptions:**

* **The time taken by the robot to travel between the Nodes of Pizza Shop in arena is 1 second.**
* **The time taken by the robot to travel between the Nodes above Pizza Shop in arena is 5 seconds.**
* **The time taken by the robot to travel from Node before the Front Door to Front Door is 1 second.**

**Refer to Figure 1.**

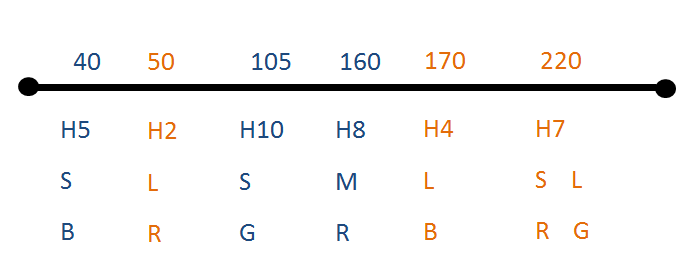
**Note: Time for picking and placing the Pizza is not considered in the traversal time.**



**Figure 1: Time for Traversal between Nodes**

**Q-7 Given an Order Timeline as shown in Figure 2 how you will deliver each and every Pizza such that you score the maximum points as per the Formula in “Scoring System” in the rulebook. (20)**

Step 1: Go from “Start” to P8



**Figure 2: Order Timeline**

**The placement of Pizzas at Pizza Shop is given below:**

**Small Blue Pizza that is to be delivered at H5 is placed at P8 (S, B is at P8).**

**Similarly: L, R is at P6 S, G is at P5 M, R is at P1 L, B is at P7 S, R is at P2 L, G is at P3**

**Algorithm Analysis**

**Q-1 Draw a flowchart illustrating the major functions that are used. (20)**

<

The flowchart should elaborate on every possible function that you will be using for completing the assigned theme. Example : lineFollowing(), colorDetection() etc.

Follow the standard pictorial representation used to draw the flowchart.

>

**Q-2 Draw a flowchart illustrating main function of your code. (10)**

<

The flowchart should explain how you will be using the functions defined in Q-1 in the main program for completing the theme assigned to you.

Follow the standard pictorial representation used to draw the flowchart.

>