CHAPTER 4
RESULTS AND ANALYSIS

4.1 Introduction

This chapter presents the results of the experiments that were conducted based on the methodology described in Chapter 3. The data obtained throughout the experiments were analyzed and interpreted. The analysis will focus on the performance of the robot, and also the characteristics of the robot using bang-bang controller, using PID controller (analog and digital), and using a PID controller with a Real Time Operating System (RTOS) ChibiOS.

4.2 Controller Design

To fulfill performance objectives desired loop, it should be added controller to the system. The selected controller is the controller Proportional Integral Differential (PID). Having obtained the transfer function of the system:

$$F(s) = \frac{1.02}{0.0093\,s^2+0.94s+1.224} \qquad \text{(Eq 4.1)}$$

Next is to determine the location of the closed-loop knot. System response to a step input function units can be seen in Figure 4.1.

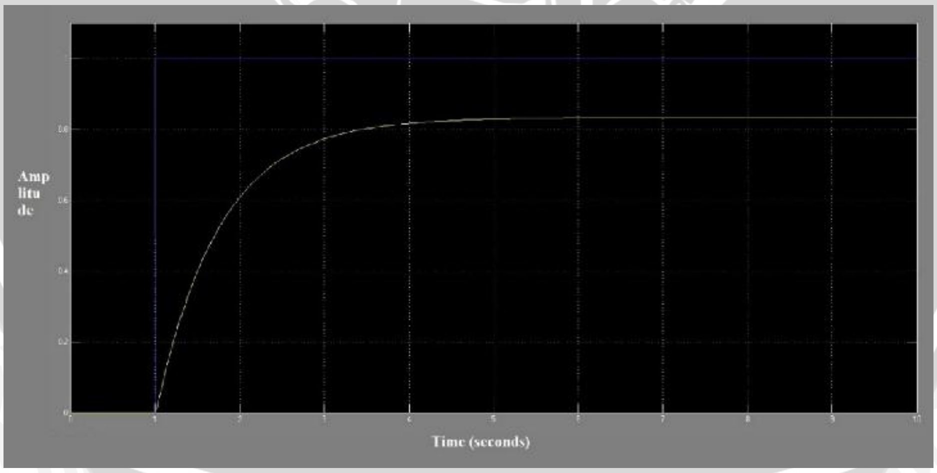

Figure 4.1 Graph simulation response system without controller.

From the responses obtained, the system has a large steady state error that required the controller to improve the system. Fixing the system can be obtained by modifying the loop - closed-loop. Based on the closed-loop transfer function can be determined system of order two. PID parameter value is determined by the selection of the pole on the root locus diagram. In this study used s = -4.43 Determination pole lies at the root locus diagram shown in Figure 4.2

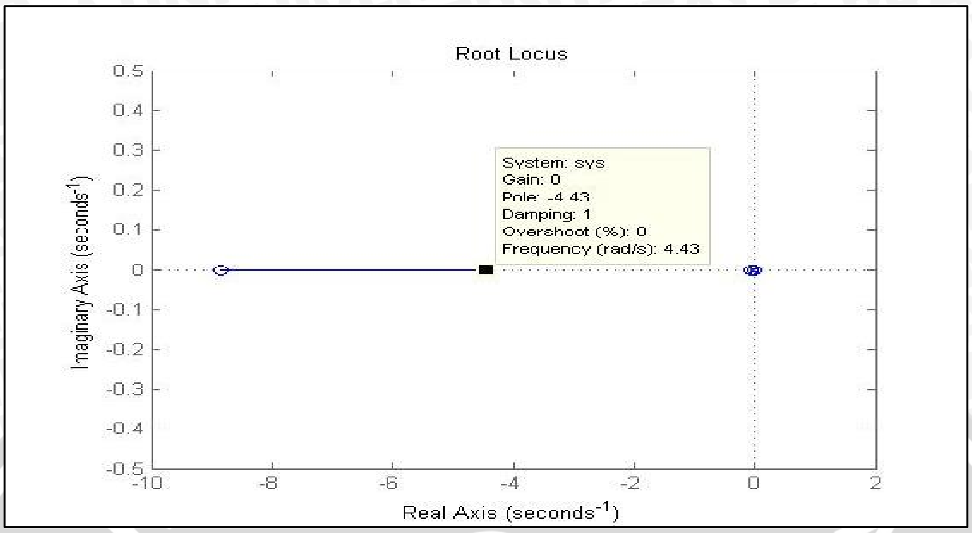

Figure 4.2 Pole position on root locus diagram.

After determining the location of the desired file and then by substituting the value s and the value of the system transfer function in Equation 2.26 and varying the value of Ki will get PID parameters in Table 4.1. Search parameters Kp, Ki, and Kd in Equation 2.26 equations using MATLAB 2013b shown in the following program listing:

```
% pole values determined from the image root locus

s1=-100.43

KI=[3 2.5 1 0.1 0.01]

plant_num=[0 0 1.02];
plant_den=[0.0093 0.94 1.224];

s1mag = abs(s1)
beta = angle(s1)

plant_al = polyval(plant_num,s1)/polyval(plant_den,s1);

plants1mag = abs(plant_al)
psi = angle(plant_al)
t=0:1:20:300;

for k =1:5

KP = -sin(beta+psi)/(plants1mag*sin(beta))-2*KI(k)*cos(beta)/s1mag
nilai_KI = KI(k)
KD = sin(psi)/(s1mag*plants1mag*sin(beta))+KI(k)/s1mag^2
```

```
Gcnum = [KD KP KI(k)];
Gcden = [0 1 0];

Tnum = conv(plant_num,Gcnum);
Tden = conv(plant_den,Gcden)+conv(plant_num,Gcnum);

r = roots(Tden)

step (Tnum,Tden,t)
hold on
end

hold off
figure, rlocus(Tnum,Tden)
```

The search results parameters Kp, Ki and Kd of calculations in the above program are shown in Table 4.1.

Table 4.1 PID parameters with s2 -4.43

No.	Kp	Ki	Kd	Pole 1	Pole 2	Pole 3
1	5.4117	0.01	0.6108	-163.6355	-4.4300	-0.0015
2	5.4524	0.1	0.6154	-164.1249	-4.4300	-0.0151
3	5.8587	1	0.6613	-169.0233	-4.4300	-0.1465
4	6.5359	2.5	0.7377	-177.2035	-4.4300	-0.3493
5	6.7616	3	0.7632	-179.9344	-4.4300	-0.4128

By modifying the value of the desired Ki generated response in Figure 4.3. Based on Figure 4.3 system response with Kp = 5.4117 Ki = 0.01 and Kd = 0.6108 has the smallest error with the fastest time setting.

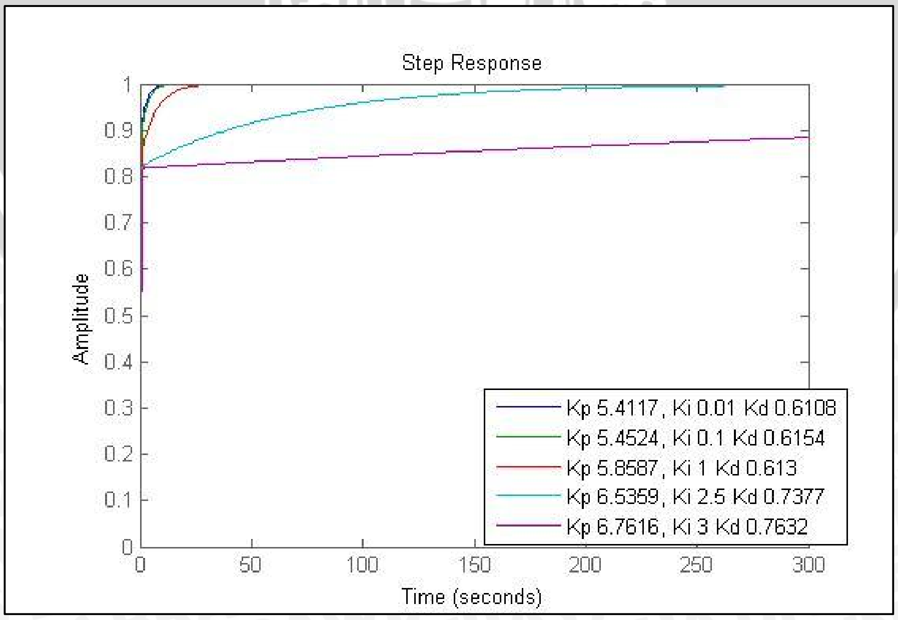

Figure 4.3 Response system based on PID parameters in Table 4.1.

The results of the PID parameter calculation result is then implemented on the system Arrow-bot using a PID controller (digital sensor concept) and the readings obtained sensor values against time is in Figure 4.4.

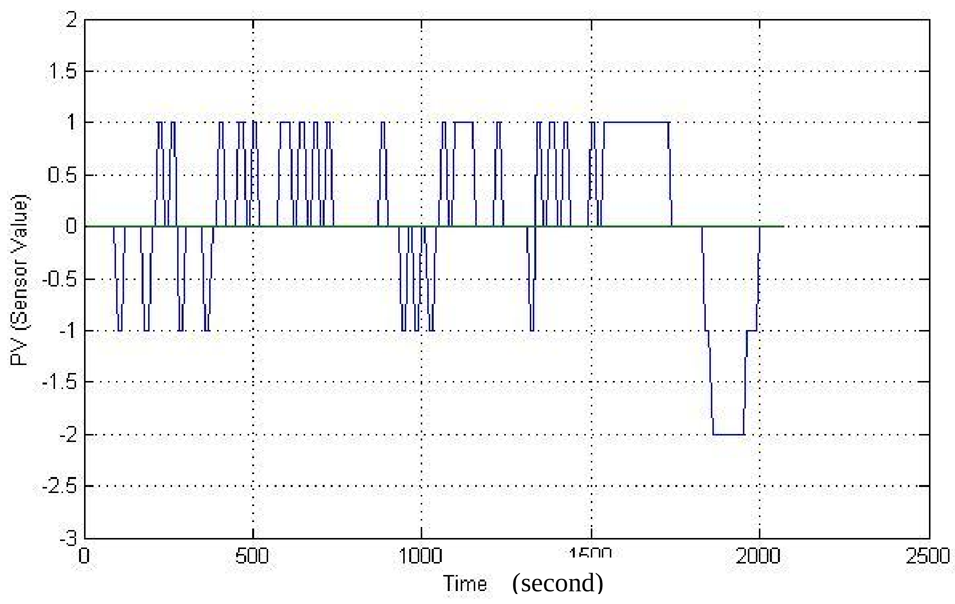

Figure 4.4 Graph reading sensor values against time (Arrow- bot using PID controller with digital sensor concept).

The results of the PID parameter calculation result is then implemented on the system Arrow-bot using a PID controller (analog sensor concept) and the readings obtained sensor values against time is in Figure 4.5.

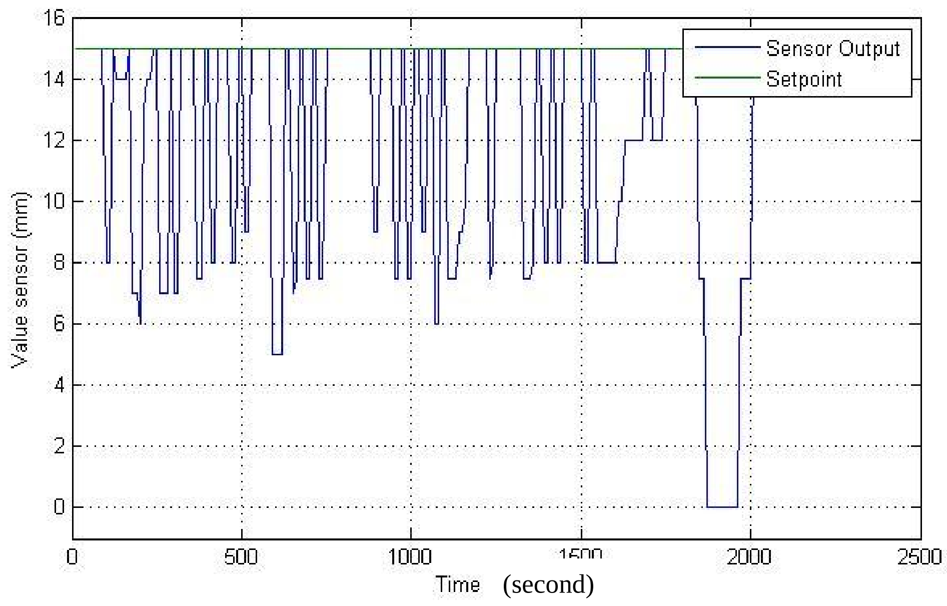

Figure 4.5 Graph reading sensor values against time (Arrow- bot using PID controller with analog sensor concept).

And the results of the PID parameter calculation result is then implemented on the system Arrow-bot using a PID controller with RTOS (digital sensor concept) and the readings obtained sensor values against time is in Figure 4.6.

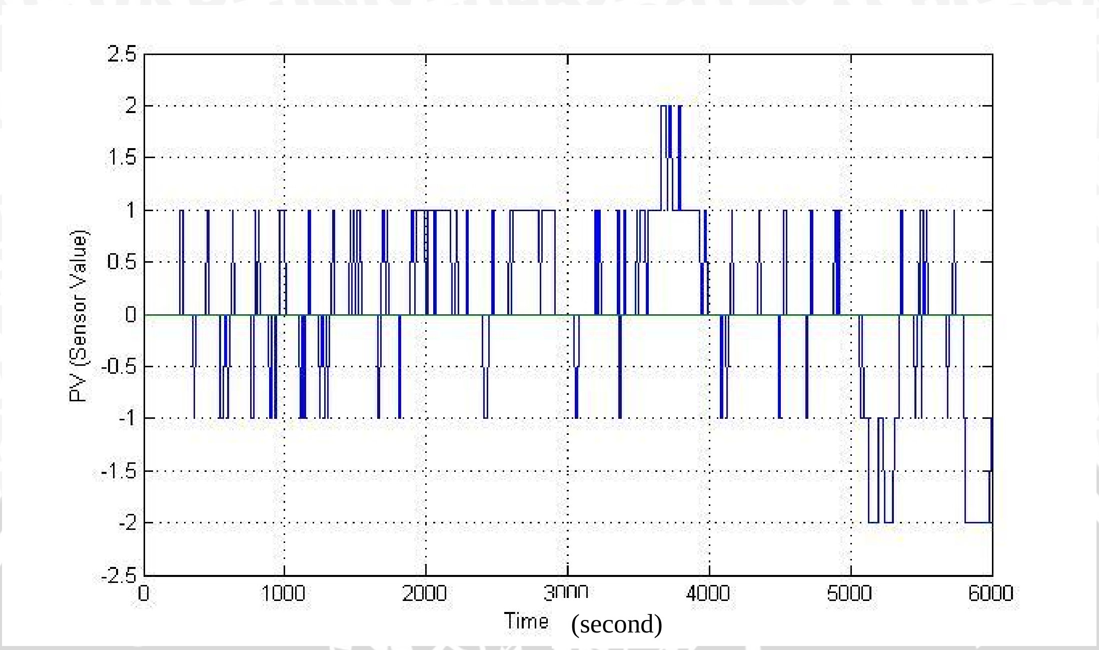

Figure 4.6 Graph reading sensor values against time (Arrow- bot using PID controller with RTOS ChibiOS).

4.3 System Interfacing Testing

There are four system tests are needed to be tested in real data testing through salae USB of logic analyser. There are five parameters to be tested on each system. These parameters are overall system, reading sensors, sensor config, PID compute, and motor command. Table 4.2 show the parameters for testing. The results of this test are very important to know the speed of system performance.

Table 4.2 Parameters testing

Parameters	**Arrow-Bot using bang-bang controller**	**Arrow-bot using PID controller (Digital sensor concept)**	**Arrow-bot using PID controller (Analog sensor concept)**	**Arrow-bot using PID controller with RTOS Chibios (Digital sensor concept)**
Overall system	×	✓	✓	x
Sensor reading	✓	✓	✓	✓
Sensor config	✓	×	×	×
PID compute	×	✓	✓	✓
Motor Command	✓	✓	✓	✓

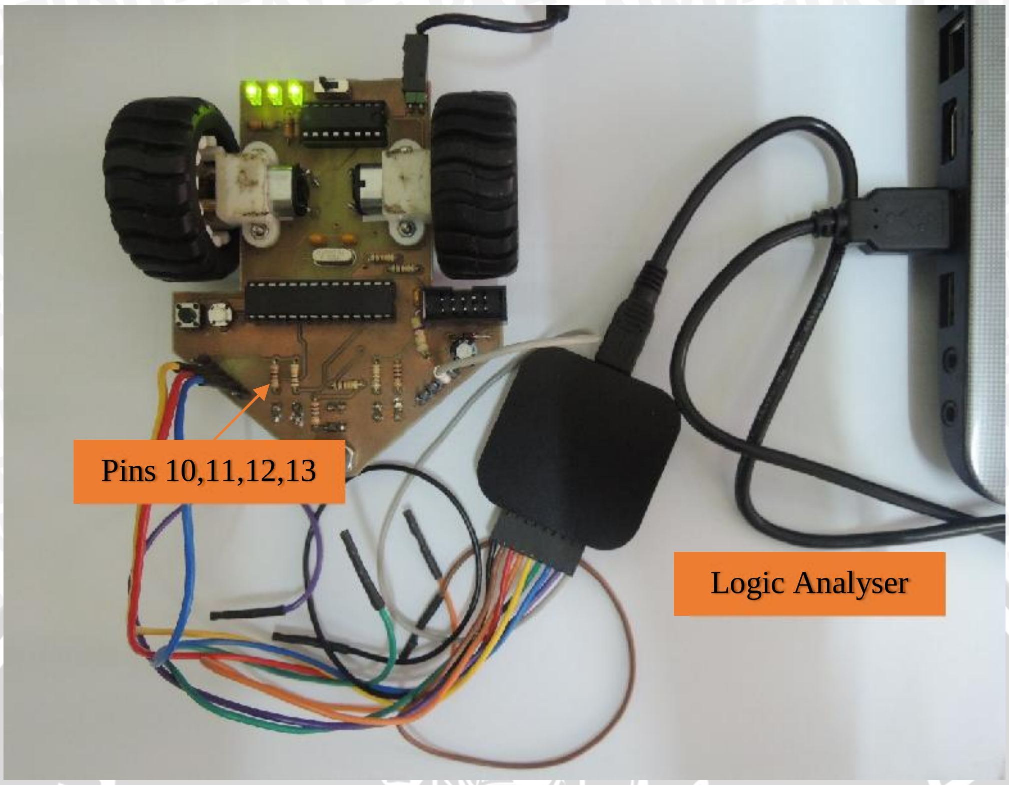

Figure 4.7 Test system using Logic Analyser.

4.3.1 Arrow- bot Using Bang- Bang Controller

In Figure 4.8 shows the data write and read on Arrow- Bot without controller. From these data, the time required to read the sensor is 8,1 ms. Then the data is processed to drive the motor, the data processing time is 17.7 µs. Timing of the motor to work is 0,1ms. So when totalized whole time this system from sensor readings in each one sample to the execution motors, it takes at 10.1 ms. So, Arrow-bot without controller response is classified as very fast. Although Arrow-bot has limitations of the sensor but with a quick response, Arrow-bot can follow the line with a variety of terrain. Although the accuracy and the results are lacking.

The time takes the system is very fast. Because a little calculation in the program. After reading the sensor data, then the data is given the command to change the motor PWM. Where the value of the motor PWM had prearranged.

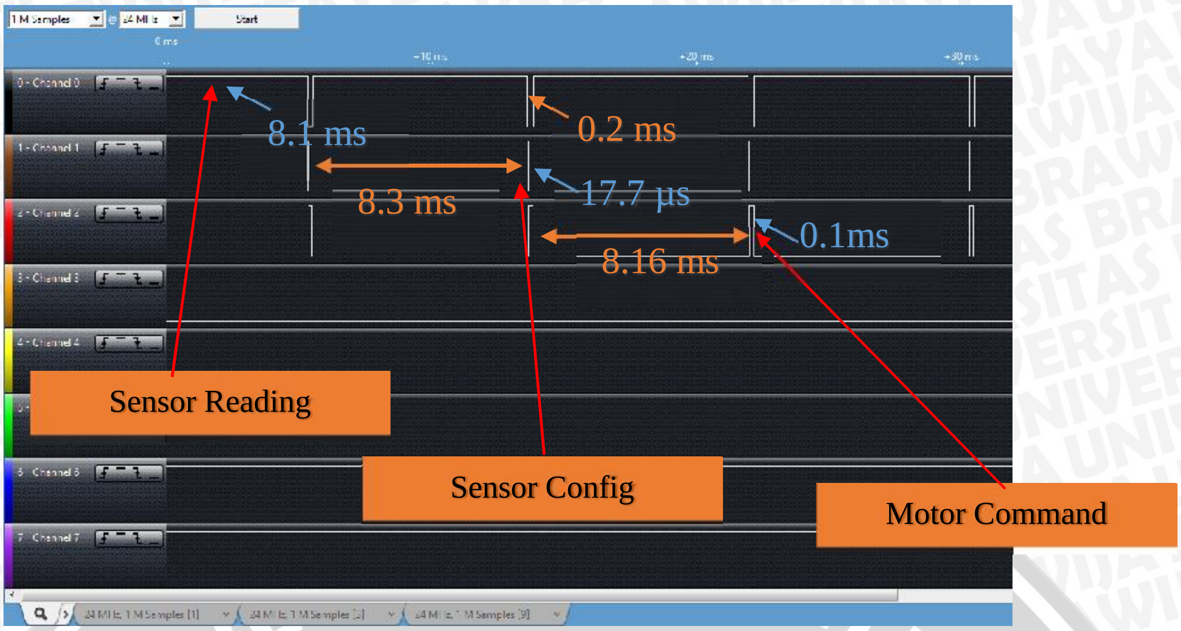

Figure 4.8 Data write and read on Arrow-bot using bang-bang controller.

4.3.2 Arrow-bot using PID Controller (Digital Sensor Concept)

In Figure 4.9 shows the data write and read on Arrow-bot using PID Controller (digital sensor concept). From these data, the time required to read the sensor is 0.36 ms. Then the data is processed to PID compute, In this section the error of data obtained and processed by the PID controller to produce a form of PWM command. PID compute time required is 16.573 ms. Then the PWM data is processed by the overall system. Time taken this command is 17.01ms. So when totalized whole time this system from sensor readings in each one sample to the execution motors, it takes at 17.01 ms.

In this system sometimes if there is a sharp turn the robot does not follow it, it is because the longer calculation time than Arrow-bot using bang-bang controller. The solution is given by adding the number of sensors in the Arrow-bot. So if there is a sharp turn sensors with outer distance can reach.

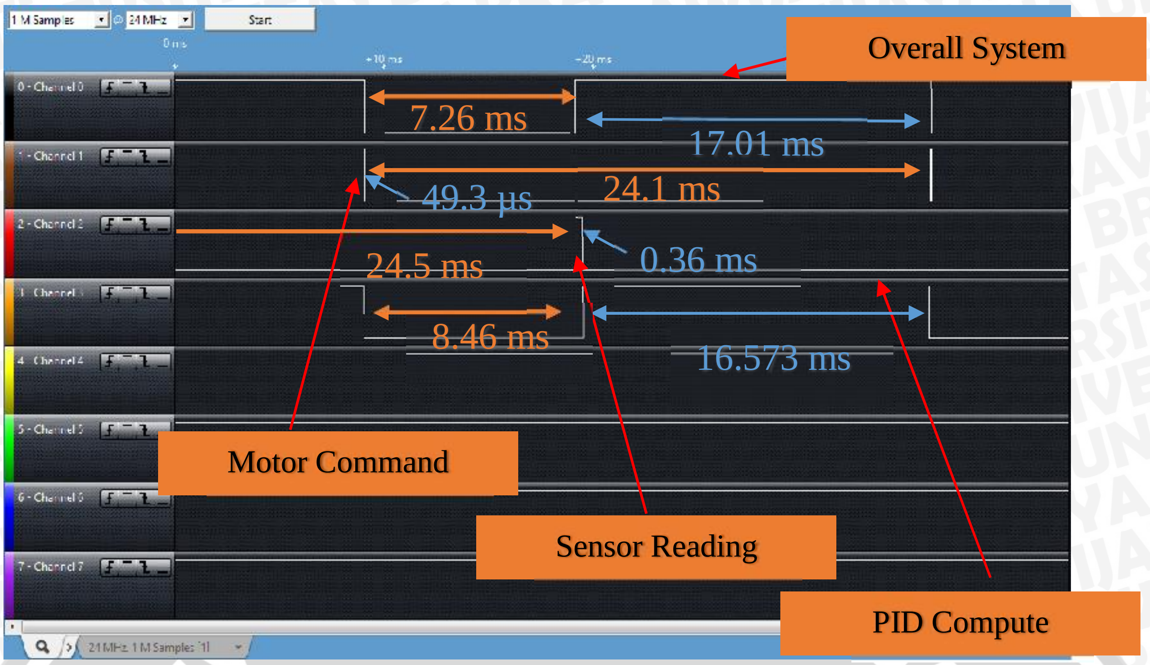

Figure 4.9 Data write and read on Arrow-bot using PID controller (digital sensor concept)

4.3.3 Arrow-bot Using PID Controller (Analog Sensor Concept)

In Figure 4.10 shows the data write and read on Arrow-bot using PID Controller (analog sensor concept). From these data, the time required to read the sensor is 0.554 ms. Then the data is processed to PID compute, In this section the error of data obtained and processed by the PID controller to produce a form of PWM command. PID Compute time required is 16.63 ms. Then the PWM data is processed by the overall system. Time taken this command is 17.31ms. So when totalized whole time this system from sensor readings in each one sample to the execution motors, it takes at 17.31 ms.

In this system, time to takes the sensor in the calculation of the error value is longer than the Arrow-bot using PID controller (digital sensor concept) because there are a lot of calculations. But the results of the accuracy in detecting the line is better because there are many possibilities. In outline, the characteristics of the robot is same as the Arrow-bot using PID Controller (digital sensor concept). This robot also sometimes if there is a sharp turn the robot does not follow.

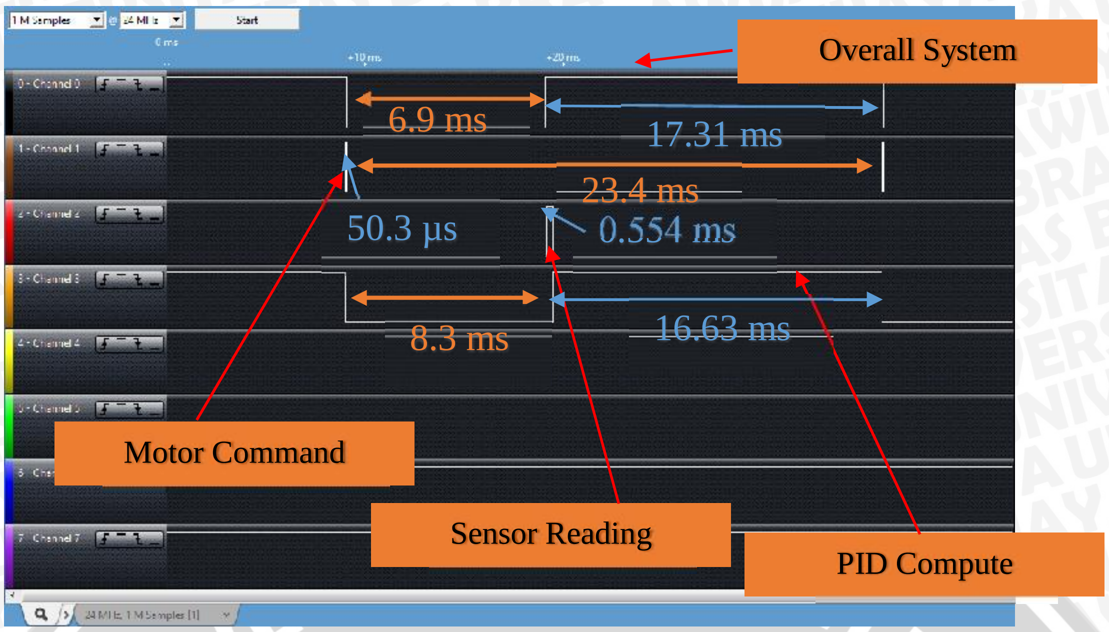

Figure 4.10 Data write and read on Arrow-bot using PID controller (analog sensor concept).

4.3.4 Arrow-bot Using PID Controller with RTOS ChibiOS (Digital Sensor Concept)

In Figure 4.11 shows the data write and read on Arrow-bot using PID Controller with RTOS ChibiOS (digital sensor). In the figure below shows, the system has different characteristics than three other systems. Each task has its own memory allocation path, so that in every command for each task is independent of another command. If seen in the Figure 4.11 when the sensor reading line needs 0.359 ms and data time reading is same continuously. Because the sensors do not wait for the command to read another command. While the time required to compute PID is 17.515 ms. Time motors move takes 42.65 µs and 0.95 ms delay. In the figure also explained that when the motor moves depends on compute PID, because the movement of the motor is set according to the value of output controller. It is becoming the system faster than use Arrow-bot using PID (digital sensor concept) or Arrow-bot using PID (analog sensor concept) due to the use of real-time operating system. So that each command can be multitasking.

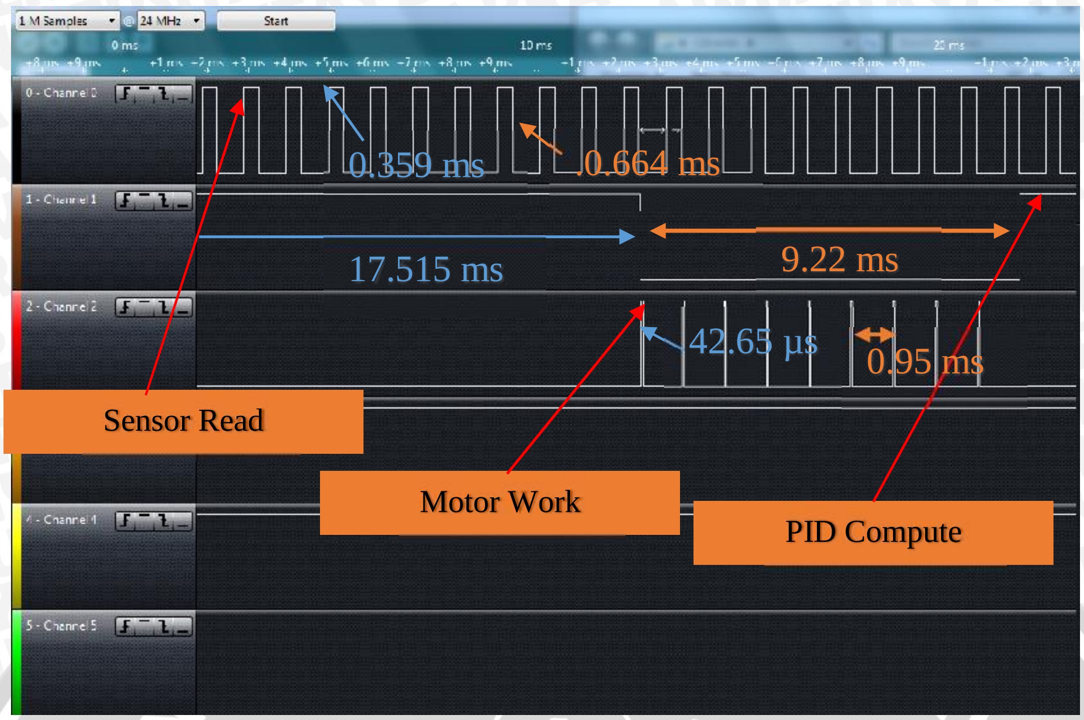

Figure 4.11 Data write and read on Arrow-bot using PID controller with RTOS ChibiOS (digital sensor concept).

4.4 Maze Specification

Specifications maze designed as shown in Figure 4.9 below. Strategies and algorithms used in this is a line tracking robot, that robot will always follow a dark line. In this maze has a width of 1.5 cm. Since the three sensors are used to detect the line, then the robot has limitations in condition sharp turn.

The maze can be used in other forms. This robot can be used on other models of the maze. Test was also performed to detect the width line that can be achieved. Line width of 1,5-2 cm for Arrow-bot using bang-bang controller, width of 0.7- 2 cm for Arrow-bot using PID controller (digital sensor concept) and width of 0.5- 2 cm for Arrow-bot using PID controller (analog sensor concept).

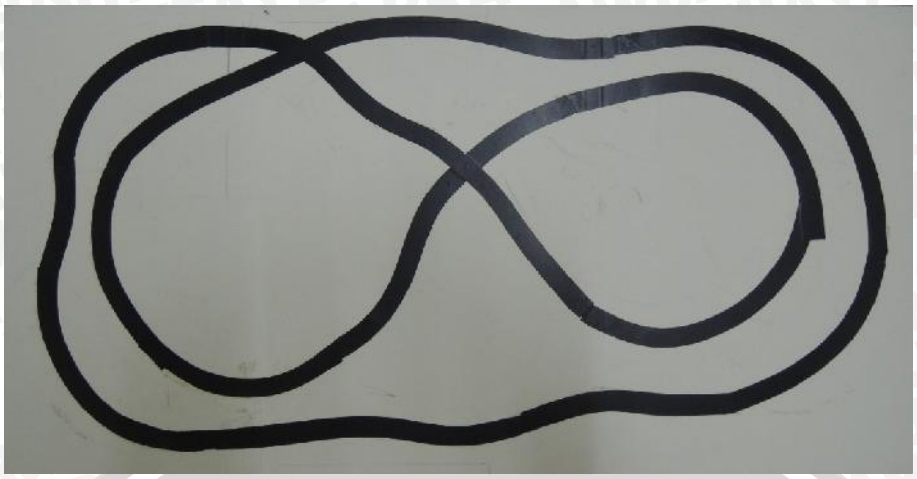
Figure 4.12 Maze design for Arrow-bot (34 x 67 cm).

4.5 Velocity Arrow-bot

In testing this robot is done by calculating time it takes the robot to achieve one lap. Testing with a stopwatch and count the start time from start line until finish line. In Table 4.3 describes time required Arrow-bot to achieve time with different system.

From these data the fastest time is the Arrow-bot using PID controller with RTOS Chibios (digital sensor concept), the second is Arrow-bot using PID controller (digital sensor concept), the third is Arrow-bot using PID controller (analog sensor concept), and the last is Arrow-bot using bang-bang controller.

Table 4.3 Time required robot to achieve one lap.

Parameters	**Arrow-bot using bang-bang controller**	**Arrow-bot using PID controller (digital sensor concept)**	**Arrow-bot using PID controller (analog sensor concept)**	**Arrow-bot using PID controller with RTOS Chibios (digital sensor concept)**
Time required to achieve one lap	13.6 seconds	13.2 seconds	13.4 seconds	12 seconds

Test was also carried out at what velocity of the robot is 22.3 $\frac{cm}{s}$ = 2,2333 $\frac{mm}{ms}$. Test was performed by measuring the time taken at PWM 255 with a specified distance.

4.6 Overall Performance Testing

Of all the data obtained it can be concluded that any system that is used in line follower robot has advantages and disadvantages of each. In Arrow -bot using bang-bang controller time required for the one-time processing of data faster than most other system. Although accuracy is bad, width- line is limited, can not move with any turn and the time spent to reach the first lap longest among others, because this mode is not smooth movement. In Arrow-bot using PID controller (digital sensor) has good accuracy, width –line is wider, fixed error value continuously. But this system can not turn if on a sharp turn because the slower response time. In Arrow-bot using PID controller (analog sensor) has the best accuracy than most other system, because using analog sensors and calculation. But this system needs more time to one-time processing data. In outline this system is same with Arrow-bot using PID controller (digital sensor). In Arrow-bot using PID controller with RTOS ChibiOS is a system derived from improve the preceding systems. Because the system uses a real-time operating system, the system is working in multitasking. So that the robot will be improved response and has the fastest time to reach one lap.Table 4.4 below shows a comparison of performance on each system

Table 4.4 Overall performance testing.

	Arrow-bot using bang-bang controller	**Arrow-bot using PID controller (digital sensor concept)**	**Arrow-bot using PID controller (analog sensor concept)**	**Arrow-bot using PID controller with RTOS ChibiOS (digital sensor concept)**
Error	×	✓	✓	✓
Time 1 lap	Fast (13.6 seconds)	Fast (13.2 seconds)	Fast (13.4 seconds)	Very fast (12 seconds)
Width -Line	1,5 – 2 (cm)	0,7 – 2 (cm)	0,5 – 2 (cm)	0,7 – 2 (cm)
Time 1 processing data	Very fast	Fast	Fast	Fast
Accuracy	Depend on maze	Accurate	Very accurate	Accurate
Sharp turn	✓	×	×	×