



Politechnika
Wrocławska

Intermediate Project Report

Fleeing Alarm CloCk – FACK

date: January 31, 2019

author: Krzysztof Dąbek

faculty: Embedded Robotics (AER)

department: Department of Electronics

university: Wrocław University of Science and Technology

instructor: dr inż. Witold Paluszyński

Abstract

Description

The project focuses on designing and building a robot using **Raspberry Pi** which would operate on the basis of **Robotic Operating System (ROS)**. The robot will serve as an **Alarm Clock** with the ability to flee from the owner.

Assumptions

- The robot is constructed as a (2,0) platform with one wheel axis
- The robot is constructed around **Raspberry Pi 3B+** or equivalent
- The robot uses third-party modules as extensions

Goals and results

Obligatory

- Simple mechanical design of the robot (**DONE**)
- Design of a power module for the robot (**NOT NEEDED**)
- Integration with Raspberry Pi 3B+ (**DONE**)
- Linux and ROS up and running on Raspberry Pi (**DONE**)
- Integration with LED screen and simple keyboard (**DONE**)
- Integration with optical distance sensors and buzzer (**DONE**)
- Implementation of ROS nodes for low-level control (**DONE**)
- Implementation of ROS nodes for operational control (**PARTIALLY DONE**)
- Operational and corner case testing (**PARTIALLY DONE**)
- Real-life scenario testing (**NOT DONE**)

Extensions

- Design and creation of PCB for the power module and input/output peripherals (**NOT DONE**)
- Optimisation of control and algorithms (**NOT DONE**)
- Implementation of OROCOS based Real Time System (**NOT DONE**)
- Component based program structure (**NOT DONE**)
- Implementation of Machine Learning algorithm (**NOT DONE**)
- Event-Based Robot control algorithm (**PARTIALLY DONE**)
- Advanced path optimisation algorithms (**NOT DONE**)
- Integration with ROS2 (**NOT DONE**)
- Migration from Raspberry Pi to low-power STM microcontroller (**NOT DONE**)

Troubles

- Unsupported system version for **Raspberry Pi 3B+**
- Connections between modules
- Burnt module with Buzzer and KTIR sensor
- Boot with portable power source faulty on **Raspberry Pi 3B+**

1 Introduction

1.1 Background

Back in the day (as many others) I had trouble getting out of bed in the morning. I had to set 2 – 3 or even more alarms not to sleep over all of them. After that long period of interrupted sleep every morning I was exhausted before going to school or university. To prevent oversleeping to work I had to think of something. Then the idea came to my mind that I could build a robot that I had to chase to even disable the irritating alarm. That amount of workout would surely wake anybody up immediately. I have decided to join this idea with the intermediate project. For a long time there have been two things I wanted to learn and use: **ROS** and **Raspberry Pi**. Therefore I started designing this platform to be my solution and learning playground.

1.2 Assumptions

- The robot is constructed as a (2,0) platform with one wheel axis
- The robot is constructed around **Raspberry Pi 3B+** or equivalent
- The robot uses third-party modules as extensions
 - Tact-Switch Keyboard as input module
 - Two-channel Motor Controller module
 - I2C module with LCD screen
 - I2C extension board with at least 3 ADC channels
 - 3 Analogue Sharp distance sensor modules
 - 1 KTIR proximity sensor module
 - Simple Buzzer module
 - Portable Power Source (LiPol battery) and step-down voltage converter module
- The modules are connected with wires and pins

1.3 Constraints

- Use of every possible already in possession module
- Low cost of the construction
- Use of only Open Licensed or free software
- Effectiveness in real-life scenarios

2 Description

2.1 Hardware

The robot was designed to be a simple (2,0) platform with two wheels, additional support point at the tail and no joints. Also it was constructed as lightweight and containing all the required modules. The electronic design was developed and updated during the construction phase. The base of the construction is glass-fibre board cut to the desired shape and size. The modules were mounted using hot glue (easiest in making and fixing). The modules have soldered gold-pin connectors and are wired with simple connecting wires. There are two ways of powering the robot: micro-USB port on the Raspberry Pi board, portable LiPol Battery with DC/DC step-down voltage converter. Pololu Motors are mounted to the board with screws and original casing. The mechanical design is presented in figure (1) and the construction in figure (4).

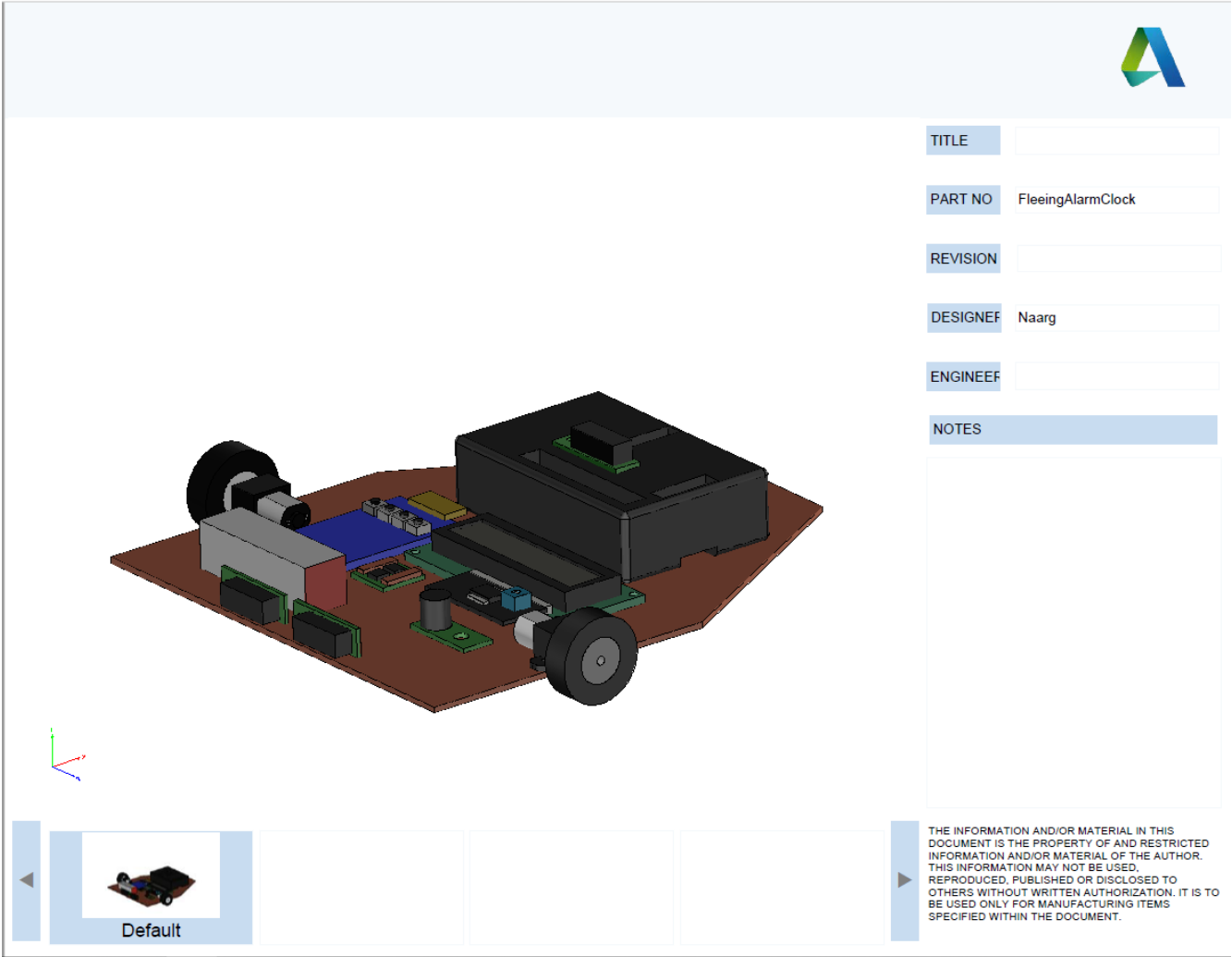


Figure 1: Picture of the final Mechanical Design of the robot (without wires)

The modules were bought in Botland. Modules used:

- **Raspberry Pi:** Raspberry Pi 3B+ with casing
- **Motor Controller:** Pololu TB6612FNG
- **DC Converter:** Pololu S9V11F5
- **Buzzer:** MOD-04526
- **LCD Screen:** 16x2 HD44780
- **LCD I2C Controller:** PCF8574
- **I2C extension board:** Adafruit ATSAMD09 Seesaw
- **Distance Sensor:** Sharp GP2Y0A60SZLF
- **Proximity Sensor:** KTIR0711SS
- **Keyboard:** WSR-04491 4x4 Matrix

The electronics design containing all the listed modules is presented in the figure below (2):

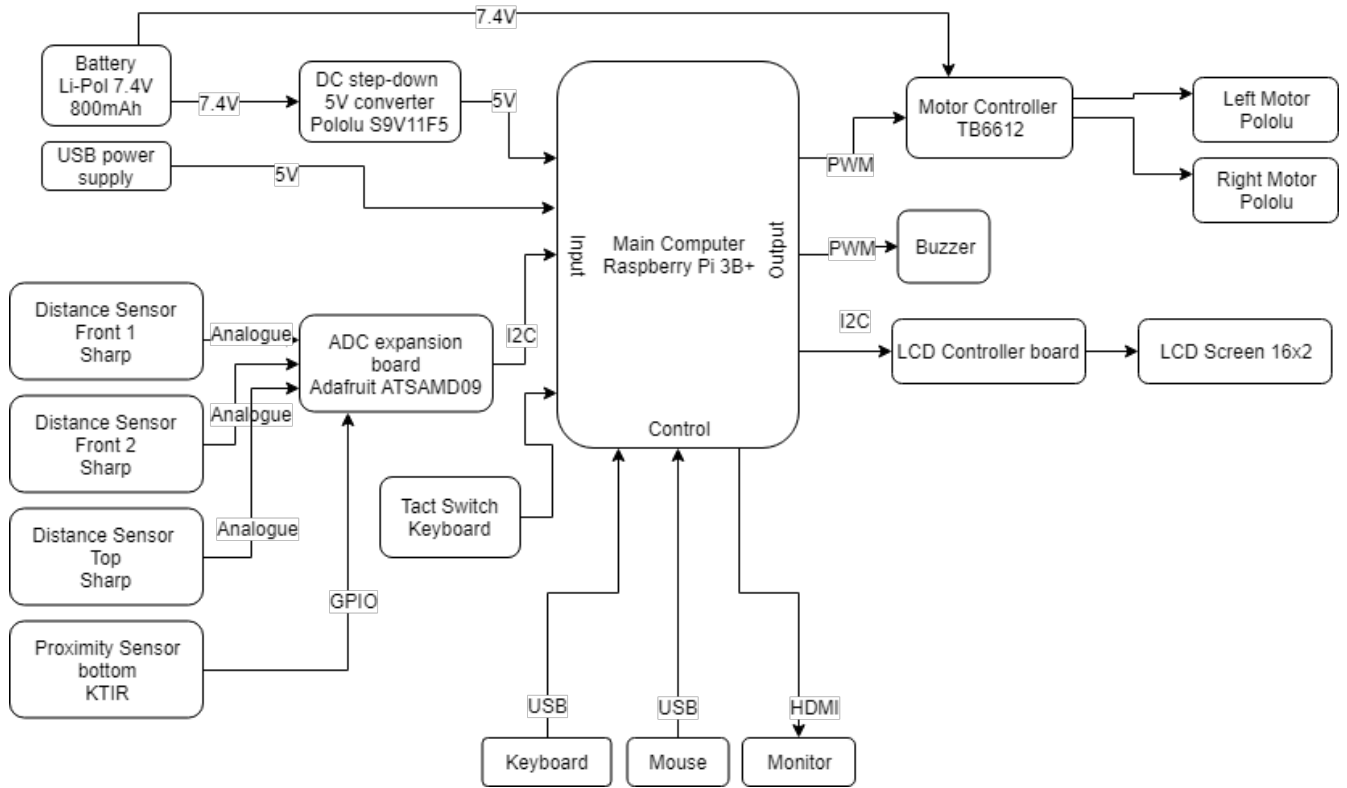


Figure 2: Electronic Design of the board connections

2.2 Software

The robot was programmed using ROS, python with rospy and a couple of external libraries and drivers. The whole program works using node structure and publisher/subscriber or service/client communication between nodes. Node Diagram has been shown below

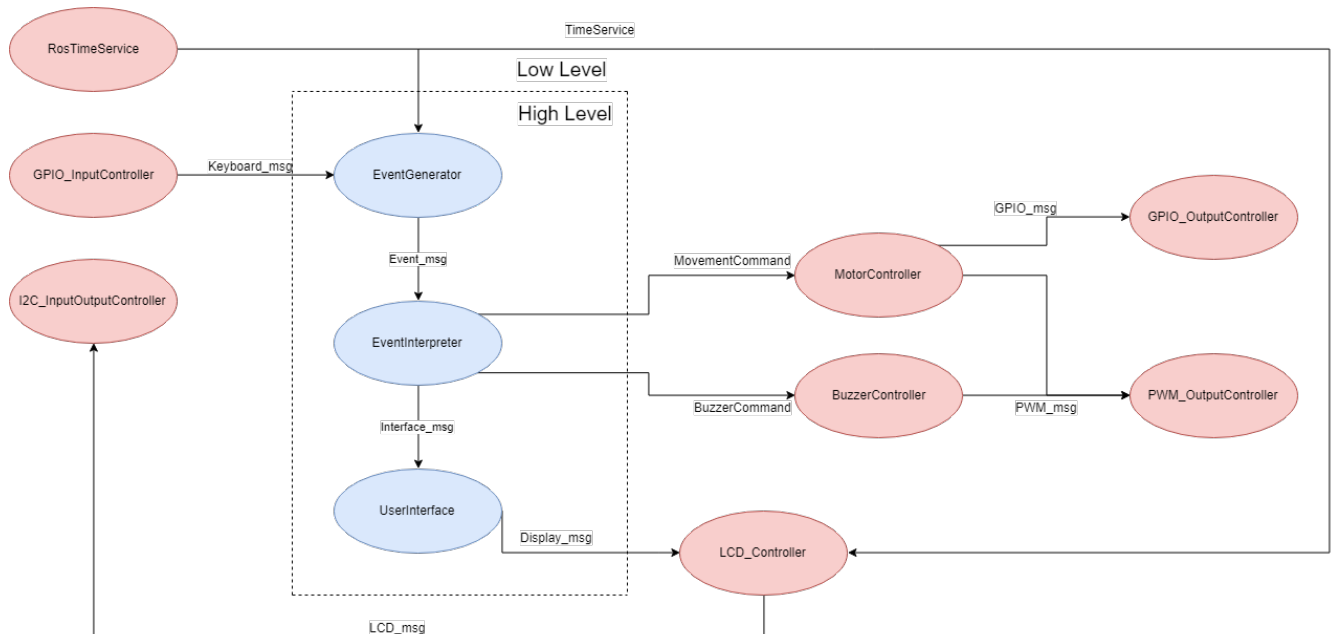


Figure 3: Ros Node Design Diagram

2.2.1 Nodes – Low-Level

- **RosTimeService** – Works as a service server which returns `currentTime`, `currentDate` and `timestamp` since the start of the node.
It is using **rospy**, **datetime**
- **BuzzerController** – A controller which upon events `AlarmStart`, `AlarmStop` decides to start or stop the Buzzer from going off, sends message to the PWM topic with buzzer's `DutyCycle`.
It is using **rospy**, **RPi.GPIO**
- **GPIO_InputController** – Recognizes the input from the tact-switch keyboard and propagates the message to Keyboard topic.
It is using **rospy**, **RPi.GPIO**
- **GPIO_OutputController** – Translates the commands for the motor and sets the rotation direction pins of the Motor Controller.
It is using **rospy**, **RPi.GPIO**

- I2C_InputOutputController – Reads ADC values from the sensors via I2C and propagates the data to ADCData topic It is using **rospy, Adafruit_Seesaw library**
- MotorController – Reads the MovementCommand from the topic and publishes the correct PWM info for the PWM controller
It is using **rospy**
- PWM_OutputController – Reads the PWM info from the topic, sets-up all relevant PWM pins and controls them accordingly
It is using **rospy, RPi.GPIO**
- LCD_Controller – Calls the time service and writes time and date to the screen via I2C. Intentionally it should also display the user interface according to the keyboard input.
It is using **rospy, I2C_LCD_driver library**

2.2.2 Nodes – high-level

- EventGenerator (unfinished) – Reads keyboard inputs, ADCData topics and calls time service to generate events like AlarmStart, Wall, Caught, etc It is using **rospy**
- EventInterpreter (unfinished) – Reads events from the topic and sends commands to the peripheral controllers
It is using **rospy**
- UserInterface (unfinished) – Reads Keyboard commands and decides which information to display on the screen It is using **rospy**

3 Results

3.1 Accomplishments

The construction of the robot has been presented in figure (4) below:

figures/robot.png

Figure 4: Picture of already constructed robot

- Installation of all required packages, os, libraries done on Raspberry Pi
- Construction and wiring of the robot platform has been done successfully
- All modules, infrastructure and integration of low-level nodes has been finished and tested
- High-level nodes have been created and started to work but not finished
- Functionality of the alarm clock has been implemented but cannot be presented yet
- Automatic roslaunch on system startup has been implemented
- Testing with USB power source and additional portable power source done

3.2 Problems

- Unsupported version of Raspberry board for Ubuntu Mate OS (**solved**)
- Unsupported version of system (Raspbian) for ROS (**solved**)
- Buzzer and KTIR modules not working (probably burnt)
- Amount of wires needed for connection way bigger than previously anticipated
- Raspberry Pi 3B+ is not supporting boot from 3.3V input

4 Summary

The platform has been constructed according to the mechanical design. During construction and implementation many problems have been encountered some of which have already been solved. The project is not finished as the high-level control, optimization and testing still needs to be done. The mandatory goals of the project have mostly been met and the platform is ready for implementing the extensions. ROS and Raspberry Pi have been used as intended. Robot has got modular control and is fairly easy to modify.

5 References

- <https://ubuntu-mate.org/>
Ubuntu Mate – Linux distribution to install on Raspberry Pi, easy to install ROS on it:
- <https://www.raspberrypi.org/forums/viewtopic.php?t=208538>
Forum topic about of problems with Ubuntu Mate installation, the information about how to make working Ubuntu mate image using Raspbian image for this board
- <http://wiki.ros.org/kinetic/Installation/Ubuntu>
ROS installation and configuration on Ubuntu mate
- <https://forbot.pl/blog/kurs-raspberry-pi-podstawy-python-a-gpio-id26099>
Tutorial on how to use GPIO on Raspberry Pi with python
- <https://forbot.pl/blog/kurs-raspberry-pi-pwm-wejscia-kamera-w-pythonie-id26930>
Tutorial on how to use PWM on Raspberry Pi with python
- <http://www.circuitbasics.com/raspberry-pi-i2c-lcd-set-up-and-programming/>
Tutorial on how to control the LCD with python driver
- <https://gist.github.com/DenisFromHR/cc863375a6e19dce359d>
GitHub repository with the I2C LCD python driver library
- <https://learn.adafruit.com/adafruit-seesaw-atsamd09-breakout/using-the-seesaw-platform>
Adafruit's library for Adafruit Seesaw board in python
- https://botland.com.pl/pl/index.php?controller=attachment&id_attachment=83
TM6612 Motor controller datasheet
- <https://www.pololu.com/product/2836>
Manufacturer website of the Voltage Regulator
- <https://botland.com.pl/pl/buzzery-generatory-dzwieku/4526-modul-z-buzzerem-aktywnym-z-generatorem-zie>
html?search_query=mod+04526&results=1
Description of the Buzzer module
- https://botland.com.pl/pl/index.php?controller=attachment&id_attachment=210
LCD I2C converter module datasheet
- <https://botland.com.pl/pl/wyswietlacze-alfanumeryczne-i-graficzne/224-wyswietlacz-lcd-2x16-znakow-niebieski.html>
LCD HD44780 16x2 description and datasheet
- <https://www.adafruit.com/product/3657>
Adafruit Seesaw I2C Expansion board manufacturer website
- <http://www.waveshare.com/4x4-keypad.htm>
Waveshare 4x4 Keyboard manufacturer website
- https://botland.com.pl/pl/index.php?controller=attachment&id_attachment=9
KTIR sensor datasheet
- https://botland.com.pl/pl/index.php?controller=attachment&id_attachment=850
Sharp sensor datasheet