Light seeking robot swarm

Embedded Systems Design and Development: Start report



Teacher: Kotkansalo Jouko

Team: Esa Jääskeläinen

Philipp Brogli

Place, Date: Jyväskylä, 08.09.2017

Title image:

URL: <http://ak8.picdn.net/shutterstock/videos/8752507/thumb/1.jpg>

Table of content

[Goal and function of the project 4](#_Toc500001036)

[Block Diagram 5](#_Toc500001037)

[Hardware 6](#_Toc500001038)

[Software 6](#_Toc500001039)

[General function 8](#_Toc500001040)

[List of robots 12](#_Toc500001041)

[Data of the own robot 12](#_Toc500001042)

[Data of the other Robots 13](#_Toc500001043)

[Protocol for the radio 14](#_Toc500001044)

[Overview 14](#_Toc500001045)

[Not connected 14](#_Toc500001046)

[ID conflict 14](#_Toc500001047)

[Measuring 14](#_Toc500001048)

[Standing still 15](#_Toc500001049)

[Wants to move 15](#_Toc500001050)

[Moving 16](#_Toc500001051)

[Beacon Mode 16](#_Toc500001052)

[Stopped moving but waiting 16](#_Toc500001053)

[Spreading out 17](#_Toc500001054)

[Motor 18](#_Toc500001055)

[Photo diode 18](#_Toc500001056)

[Infrared LED and Infrared photodiode 18](#_Toc500001057)

[Status LED 18](#_Toc500001058)

[Shift register 19](#_Toc500001059)

[Radio 19](#_Toc500001060)

[Display 19](#_Toc500001061)

[Test concept 20](#_Toc500001062)

# Goal and function of the project

For fishes in the shallow water a quick detection of a shadow can decide between live and death. With that in mind we build five identical robots which simulate similar behaviour.

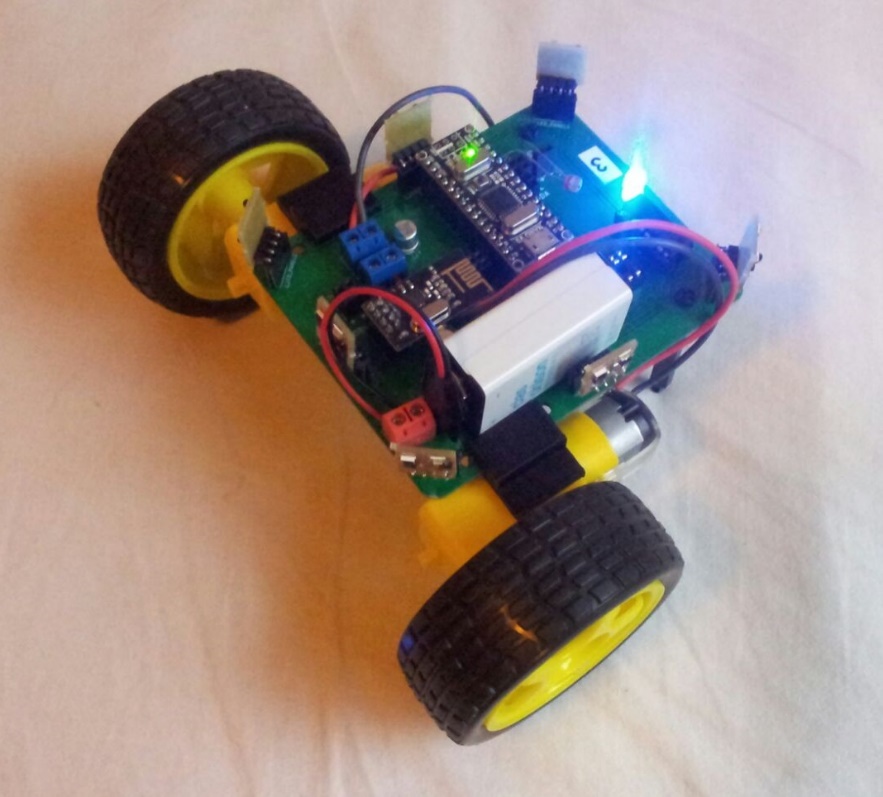
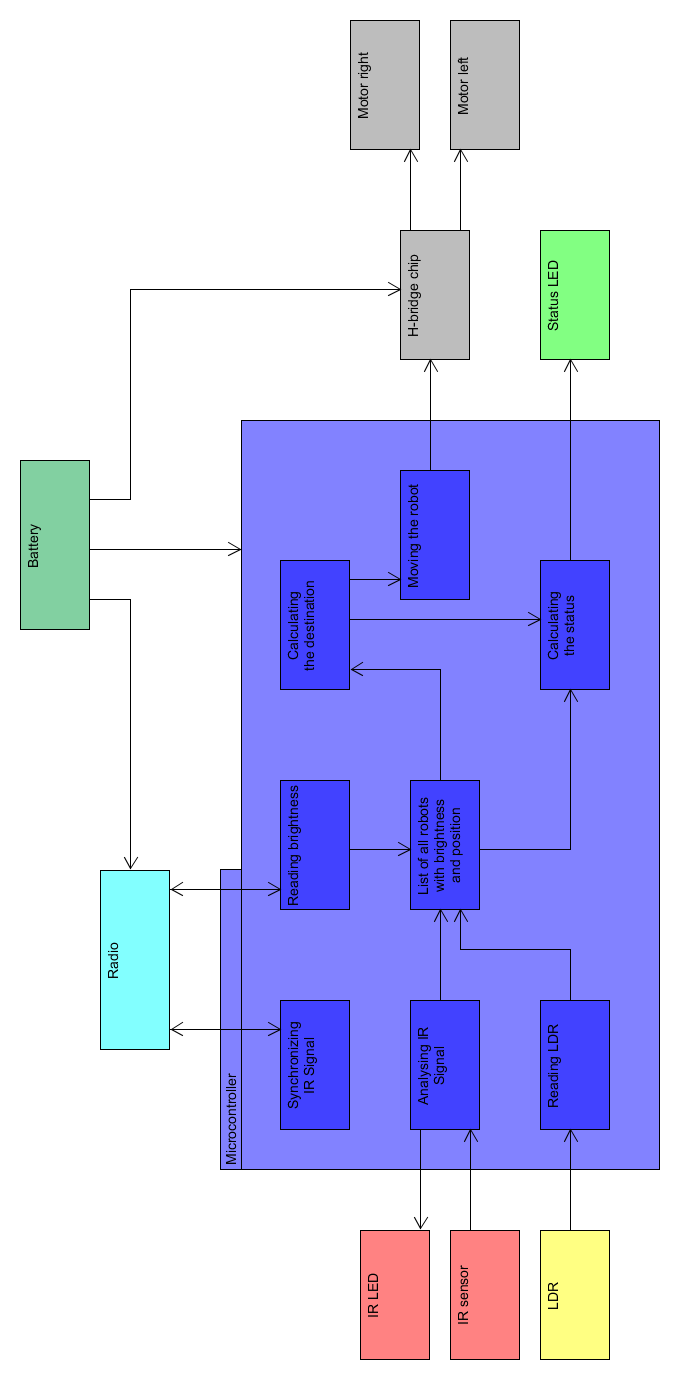
The robots all get placed onto a flat surface. With a LDR each robot measures the brightness above it. For that to work best there has to be a ceiling with a homogeny brightness. With two motors they can move on top of the surface. A RGB LED can be used to show the status of the robot. There are seven small chips mounted on each robot. They contain a IR LED and a IR Sensor. With those the robots measure the rough distance and direction between each other. The robots communicate with each other with the use of a radio. The Microcontroller controls every move of the robot. And the robot is powered by a battery.

Figure 1: A picture of the robot.

As a final result, the robots compare the brightness of themselves with the one of the others. If theirs is noticeable darker than one of the others, they move to the brighter one. So if you light with a flashlight onto one of the robots all of them should move to that spot. If the brightness of only one robot is darker than the others it moves to the closest one with a brighter value. Therefore if you cover one or more robots with a black object they move out of the dark spot to another robot. If all the brightness’s are the same they stay where they are.

This project is deliberately too big for us to complete in the time give. We might continue it after the Semester has ended.

# Block Diagram



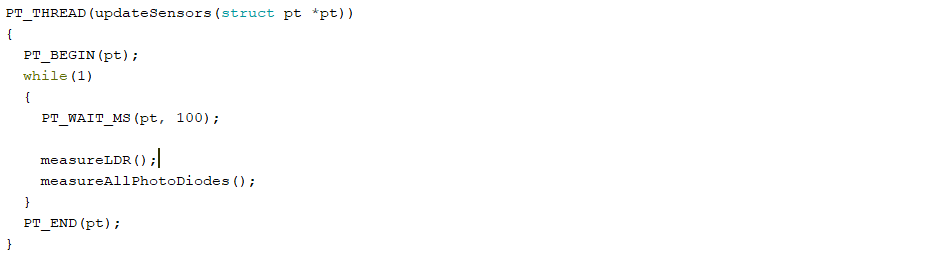
# Hardware

# Software

The entire Project is deliberately chosen to be too big. The documentation will describe the executed parts and the goal. It will be written what is implemented and what isn’t.

The Software was written in c. For the sake of making it easier to read it is split into several header and .cpp files. The header files contain the initialisation of the classes, the #defines and some of the variables. In the .cpp files the actual functions of the program is implemented. Each part of the program contains a header and a .cpp file. To use functions of a different .cpp file you need to include the header file into your .cpp file.

To further improve the readability the library protothreads is used. The following image shows an example on how it’s used.



During the setup you need to initialise the protothread (sensorPt). While the program is running the thread is called. You can set the time on how often the code inside the thread is executed with the following line PT\_WAIT\_MS(pt, 100); In this case it is 100ms.

For testing purposes some of the .cpp files can be disabled. If you want to work on the robot and because of some reason it always drives away you can just disable the motors.

All the disable functions are placed in the defines.h file. Just by making the line into a comment or into program code you change it from enabled and disabled.



Inside the .cpp files the code looks if the motors are enabled or disabled with the following lines.



## Flow chartGeneral function

The current status of the software is that all the basic functions are implemented. The robots can drive, control the LEDs, measure the brightness and Infrared and they can communicate with the radio. The only thing that is left is to implement the way everything is controlled. That is basically what this chapter is about.

Not connected to any robot via radio:

Upon start up the robot first tries to find other robots to communicate with. It sends in random intervals a message into the world. If there is no group around but only a single robot they react to each other and form a group. Groups are a lot more active with the radio than a single robot. So it’s highly likely for the robot to spot a group before there is even a need to send the first message.

ID conflict:

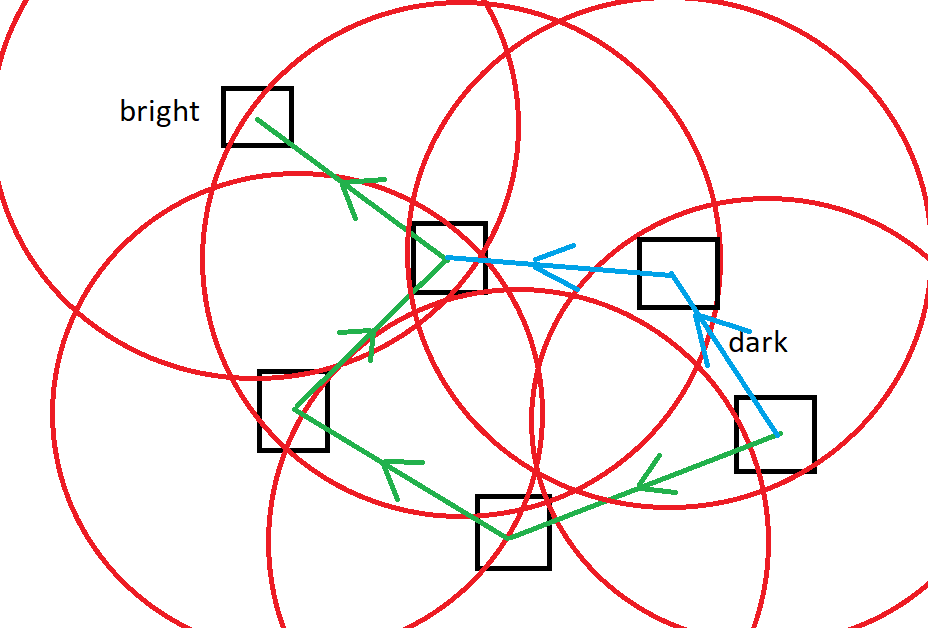
After a robot joined or left the group the robots reorder the ID. Each robot has an own ID number whereas the first robot has the ID number 1. The next one has an ID of 2 and then 3 until every robot has an ID.

There is no plan jet on how to implement that feature.

Position measuring:

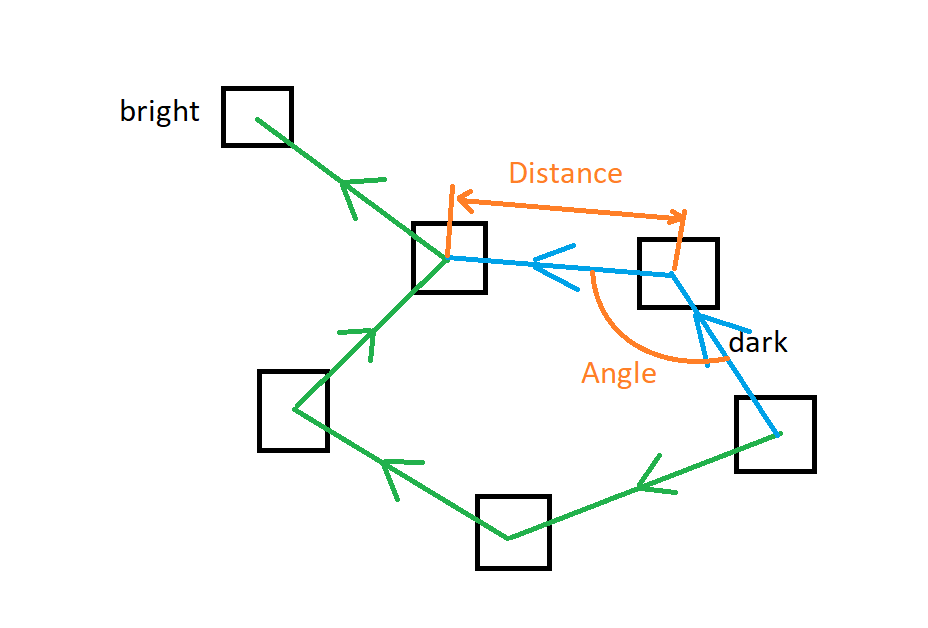
After the group composition changed or after some of the robots moved they have to evaluate which robots are in visible range and where they are relative to the each other. After every robot has finished measuring they share the data and store it into a list on each robot ([see chapter List of robots](#_List_of_robots)).

Standing & Brightness compare:

After the position is known the robots go into some sort of waiting modus. The only thing they do is share some data with each other. In the end the robots want to drive to a bright spot. The brightness of each robot, which is constantly shared, is constantly compared to the own brightness. If the own brightness is low compared to the others they try to go to that one. If they are in close proximity they can just drive to it directly. That is what the current program does. But if the group is spread out more widely they can’t see each other and therefor they can’t drive to the intended place. That is why they store the data on which robot sees which one. With that information they can calculate if there is a path between the goal and the current position. The following picture illustrates a possible scenario.

In that picture the robot in the dark place doesn’t see the goal directly but with the knowledge about the IR measurement of the others there is a way to calculate it.

Wants to move:

With the brightness difference the robot decides to move. But before it can start driving the path has to be calculated first. For that the pathfinding algorithm A\* is used. With it there can be calculated if there is a path and which one is the shortest. A\* basically tries every possibility until it find the goal. It can each step of the path can have its weight. The more weight it has the longer the path is in the end. In this case the robots can measure a rough distance and a rough angle. The longer the distance and the sharper the angle is the higher the weight.

Gets called out to be a beacon:

As long as the robots stay still every position is known. As soon as one moves the list of each robot is faulty and the same path might be impossible. If robot A want’s to go to robot C via robot A both robot B and C have to wait until robot A has reached its goal. In the protocol ([see chapter Protocol for the radio](#_Protocol_for_the)) robot A mentions B and C. As soon as the end goal C is visible B is no longer on the list in the protocol. Therefor B can move as well if it wants to.

Moving:

Inside the list there is a bit which sates on which robot has moved since the last measurement.

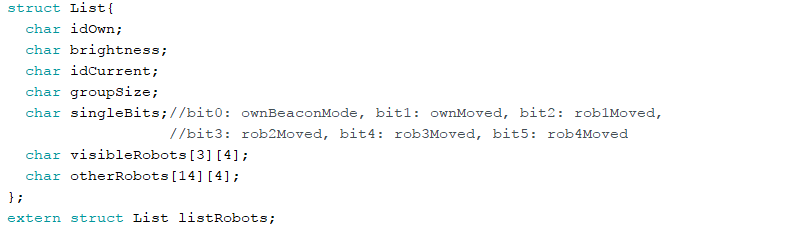
Stops and waits until others stop:

If the robot reached its goal and other robots are still moving it has to wait. As soon as every robot stopped moving they measure the distance between each other and everything starts all over again.

Spreading out:

Because the robots only move closer to each other there is a need for something to drive them apart. But they shouldn’t move too far because then they couldn’t see each other again. If some of the other robots get cut off from the group this step could also help them get them back. But there is no plan jet on how to solve that.

# List of robots



This list contains the most important knowledge the robot. The data in there is needed for the protocol and the calcullation of the path with the A\* algorithm.

Visible robot:

In total there can be 4 robots visible with the infrared photodiodes. With the array visibleRobots[3][4] there is space to store the ID, the distance (abs) and the direction (ang) of all of them.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **On IR visible robot with lowest ID** | | | **On IR visible robot with second lowest ID** | | | **On IR visible robot with third lowest ID** | | | **On IR visible robot with forth lowest ID** | | |
| ID | abs | ang | ID | abs | ang | ID | abs | Ang | ID | abs | Ang |

Other robot:

Basically the same information as you can find in visibleRobots is useful to know about the other robots.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID of robot** | **Brightness** | **On IR visible robot with lowest ID** | | | **On IR visible robot with second lowest ID** | | | **On IR visible robot with third lowest ID** | | | **On IR visible robot with forth lowest ID** | | |
| ID | Brightness | ID | abs | ang | ID | abs | ang | ID | abs | ang | ID | abs | ang |
| ID | Brightness | ID | abs | ang | ID | abs | ang | ID | abs | ang | ID | abs | ang |
| ID | Brightness | ID | abs | ang | ID | abs | ang | ID | abs | ang | ID | abs | ang |
| ID | Brightness | ID | abs | ang | ID | abs | ang | ID | abs | ang | ID | abs | ang |

# Protocol for the radio

The protocol helps the robots communicate between each other. It is build up in Starting info, Message and End info. The Starting info of every protocol are these four bytes.

|  |  |  |  |
| --- | --- | --- | --- |
| IDown | Group size | Message Size | Task |

The IDown and groupsize is there to identify the robot and help solving ID conflict. The message size and the task is there to tell the receivers what the message contains. Whenever the robot has sent all data which it wanted to send it ends the protocol with the byte

|  |  |
| --- | --- |
| Brightness | IDnext |

IDnext has normally the value IDown+1. If IDown has the same value as group size then IDnext is 1. Of course you can turn of a robot while the others are still running. If that is the case this robot won’t respond to the previous robot and the Robot sends an ID conflict.

Not connected:

If the robot is not connected the protocol contains only the starting information. The robot always switches between a longer period of random length of reading and sending a short message. As soon as it receives a message from another robot it goes into the ID conflict state.

|  |
| --- |
| Starting Info |

Task: 0001’0000 (16)

ID conflict:

There is nothing planned jet on how to solve that Task.

Task: 0010’0000 (32)

Measuring

Here the g:oal is for every robot to know where which robot is. So every Robot lights up its IR LEDs one at a time. That way the surrounding robots know its position and the power consumption is low enough that the battery can support it.

When it is the robot turn, it first sends just the starting info.

|  |
| --- |
| Starting Info |

Task: 0010’1000 (40)

After that it lights up all the LEDs one after the other. When it has finished with that it sends the Starting info again and its brightness and the ID of the next robot.

|  |  |
| --- | --- |
| Starting Info | Ending Info |

Task: 0010’1001 (41)

This step of the flow chart gets executed by every robot of the same group. When all the robots finished this task they make another round to share their measurement with the following protocol.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Starting Info | ID1 | abs1 | ang1 | ID2 | abs2 | ang2 | … | Ending Info |

Task: 0010’1010 (42)

IDn, absn and angn: transmit the data of the visible robots. The content of the data can be found in chapter Data of the own robot.

Standing still:

In this step the robots basically only share their brightness and show that they are still active. That is done with the following protocol.

|  |  |
| --- | --- |
| Starting Info | Ending Info |

Task: 0011’0000 (48)

Wants to move:

As soon as there is another robot at a much brighter spot this robot wants to move to the brighter one. The A\* Algorithm calculates the shortest possible path to the other robot.

|  |  |
| --- | --- |
| Starting Info | Ending Info |

Task: 0100’0000 (64)

Moving:

Shortly after the path to a brighter spot is calculated the robot starts to move. It also tells the other robots where it wants to move to and which robot it has to follow to get to that destination. The robot always moves to the left most visible robot on the protocol. All the robots right to that on the protocol get removed from that list. All the robots in the List aren’t allowed to move because then the position after moving isn’t known anymore.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Starting Info | IDdestination | IDsecond last | IDthird last | … | Ending Info |

Task: 0100’0001 (65)

Beacon Mode:

When in beacon mode the robot is not moving. This step has priority over the step robot wants to move and standing still. As long as a robot that wants to move or is moving has this robot on its protocol, this step is active for this robot. The other robots must know where this one is and therefor this robot lights up its IR LEDs one at a time. As soon as it isn’t on the protocol anymore it stops the beacon mode.

Before the LEDs light up this protocol gets send via the radio.

|  |
| --- |
| Starting Info |

Task: 0100’0010 (66)

After that this protocol gets send.

|  |  |
| --- | --- |
| Starting Info | Ending Info |

Task: 0100’0011 (67)

Stopped moving but waiting:

When a robot moves the position has to be measured again afterwards. When several robots are moving the position can only be measured again when all robots stopped moving again. The robots which have finished moving and are waiting for the others are in this step. When all robots have finished they change into the step position measuring.

|  |  |
| --- | --- |
| Starting Info | Ending Info |

Task: 0100’0100 (68)

Spreading out:

There is nothing planned jet on how to solve that Task.

0101’0000 (80)

## Motor

The motors are controlled by two values. Direction and speed. The speed is controlled with a pwm signal on an H bridge. Since the robot has two Motors which are drive individually they can make the robot drive forwards, backwards, in a curve or make it turn on spot.

## Photo diode

### Infrared LED and Infrared photodiode

The IRLED are positioned around the robot to show the other robots its position. At the moment the IRLED are turned off when the brightness level is low. If it gets brighter they get turned on. The robots measure the IR reading every 100ms. As soon as they see a robot turning on their IR they start driving to it.

There are seven IR photodiode one in the front, three to the left and three to the right. Because of that the robot knows where the IR light source is. With that knowledge it can turn either left or right to adjust itself correctly and afterwards it drives to this IR source. One problem that exist at the moment is that the normal light sources have enough infrared in them to make the robot drive into a random direction. If the room lighting is done with LED lamps the background noise is not big enough for that to occur. Instead of replacing the room lighting there is also the possibility to reduce the sensitivity of the IR photodiodes. Since the entire project is not done jet we didn’t bother with fine tuning of this detail.

### Status LED

The status LED is a RGB LED and its goal is to show the user what the robot is currently doing.

At the moment the LED is blue if the brightness reading is very low. When the infrared reading gets higher than a certain level the LED turns red.

When the program is more finished and has a more complex function the status LED has to signal the user with more information. The following list describes the status LED with of the finished program.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Action** | **Blue**  Radio offline | **Green**  Idle state | **Red**  Too dark | **Yellow**  Driving to the light |
| If there is no robot in visible range (IR) it is always light up. |  | **X** | **X** |  |
| Blinks with 2.5Hz for the amount of robots in visible range (IR). It’s dark for a second afterwards. |  | **X** |  | **X** |
| While turning on spot it is 0.2 s on and 0.8s off. |  | **X** | **X** |  |
| Waiting for others doing something:  Diming on and of 0.5Hz |  | **X** | **X** |  |
| Measuring distance or other stuff |  | **X** | **X** |  |

### Shift register

## Radio

The radio library used here is called RF24. The radio has six different channels. This Software only uses one of them. Most of the time the robots are listening for incoming messages. The robot only switches to the sending modus when it send the protocol. Afterwards it immediately switches back. While in a sending modus the radio can’t listen for incoming messages. But if the radio is in a sending modus it stores the message until it is overwritten by the next incoming message. This message can get called at any time.

In this project there are several participant in the communication via radio. The sending one is the master and the rest are Slaves. Normally the radio expects a response from the receiver because messages can get lost. Since there are an undefined number of listeners around that doesn’t work. The only place where a check is crucial is at the handover of the master position to the next robot. If the previous master receives the message of the next master then the handover was successful. If there is no reply the previous master retries again.

## Display

# Test concept