Knight Lights Assignment 2  
Embedded systems Practicum  
Assignment 2

April 14, 2022.

**Group 12**

Students: Nick Mulders, July Trendafilov

# Abstract

The design and execution of a knight lights system are described in this technical report.   
This is the third assignment of the Embedded Systems 2 practicum. Our approach and conclusions are further elaborated below.

Table of Contents

[Introduction 3](#_Toc99791436)

[Procedure 3](#_Toc99791437)

[Design 4](#_Toc99791438)

[State machines design 4](#_Toc99791439)

[Circuit diagram 5](#_Toc99791440)

[Context Diagram 6](#_Toc99791441)

[Conclusion 7](#_Toc99791442)

[References 7](#_Toc99791443)

# Introduction

This assignment is about building a traffic light system with the use of 2 Arduino boards and 2 breadboards with components. In addition to building the system and writing this report we have also been assigned to program the Arduino’s in the C language. Further requirements and instructions are described in the assignment document (Hilderink, Assignments, 2022).

Our Procedure of building this traffic light system is further described in the ‘Procedure’ section.  
The design of the context diagram, state machines, circuit diagram and communication messages are described in the ‘Design’ section. And the conclusion of our research in the ‘Conclusion’ section.

# Procedure

The traffic light system was an assignment given by (Hilderink, Assignments, 2022). The needed hardware was given in the document. We first designed the system with a context diagram to get a nice overview of how the traffic lights system should work. Then we designed multiple state machines, one for the master and one for the slave. Lastly we created a circuit diagram.

We make two subsystems on Arduino’s that communicate with each other. Both systems simulate one traffic light and they communicate via the TX and RX pin on the board. These pins are handy because they can send and read messages from the connected board. At first we started designing the context diagram (see [Figure 4](#Figure4_context_diagram)). We did this so we could get a good general overview of the system. Then we started designing the state machine diagrams, so we could get a good understanding of how the master and slave should communicate and react in certain instances (see [Figure 1](#Figure1_state_machine_master) and [Figure 2](#Figure2_state_machine_slave)). It took multiple iterations to make sure the system was robust and could always recognise when communication fails. Then we designed the circuit diagram to make no mistakes during the building process (see [Figure 3](#Figure3_circuit_diagram)).

After determining how to build the system we first must make sure the system is safe. We did this by using Ohm’s law and calculating the power in Watts while using the (Semiconductors, 2009) datasheet.   
First we have to know to forward current and we calculate it by using the datasheet:

* Ur = 5 – 2,1 = 2,9V

With the forward current and the Amps we also retrieved from the datasheet we can use Ohm’s law to determine the minimal resistance needed:

* R = 2,9 / (30 / 1000) = 96,7 Ω

To know the circuit is safe for sure we have to know the power in Watts. We can also calculate this by using the forward current and Amps:

* P = 2,9 x 0,03 = 0,087 W

0,087 W < 0,250 W, so we can conclude that the circuit is safe. After verifying the circuit is safe we were able to build it. Then we started measuring the Volts and Amps. With these measurements we started calculating one more time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **V** | **mA** | **R** | **P** |
| **LED1** | 1,967 | 9,33 | 1,967 / 0,00933 = 210 Ω | 1,967 x 0,00933 = 0,0184 W |
| **LED2** | 1,972 | 8,97 | 1,972 / 0,00897 = 219 Ω | 1,972 x 0,00897 = 0,0177 W |
| **LED3** | 2,008 | 8,7 | 2,008 / 0,0087 = 230 Ω | 2,008 x 0,0087 = 0,0174 W |

And with the accurate values we can also confirm that the circuit is safe as seen in the power column.

Now that we are 100% certain our circuit is safe and reliable we are ready to start coding. While coding we had to keep the contracts of the state machine in our minds to ensure user safety.

The development of the code was going smoothly. We did not have any problems programming the boards separately and testing them. We tried optimising the boards with registers so that they work even faster. The only temporary issue we faced was the serial communication between the master and the slave Arduino as the format and type of the data were different in both boards. In the end we had to insert some delays to synchronize the lights so that the cars in both directions don’t move at the same time.

# Design

## State machines design

The final design of the state machines are given in figure 1 and figure 2. The design of the master in figure 1 mainly sends messages and only receives acknowledgements from the slave. These acknowledgements are necessary to ensure the safety of the users. If a system doesn’t receive an acknowledgement it forces itself in a blinking mode where the yellow lights start blinking.

Diagram, schematic

Description automatically generated

Figure 1: State machine of master Arduino

The design of figure 2 mainly receives messages from the master and only sends acknowledgements back. The slave Arduino also has a built-in time-out timer because in case of a communication error the system itself should also be able to recognize and turn itself into a hazard mode.

Diagram

Description automatically generated

Figure 2: State machine of slave Arduino

## Circuit diagram

The circuit diagram gives us a nice overview of how the components should be connected to the two Arduino’s. With this diagram we already know how the system will look like before even building it. It is also good to have names of the components to refer to them in the measurements table for example.

Diagram, schematic

Description automatically generated

Figure 3: Circuit diagram of Traffic light system

## Context Diagram

// two separate contexts for master and slave in one context diagram

// In state diagram we specify the sequence of the states

// figure 5 messages should be on the context diagram lines

The context diagram is the starting point of this assignment. While creating the diagram we started brainstorming about how the system should look like and what were the key elements of a traffic light system.

Diagram

Description automatically generated

Figure 4: Context diagram of Traffic light system

Furthermore we also made a message protocol for the system to make no communication mistakes and to give the reader a clear view of what certain messages mean and what they do.

Figure 5: Messages

|  |  |  |
| --- | --- | --- |
| **Message** | **Description** | **Specs** |
| RED | The Master sends this signal to the Slave for it to change to RED light. | HIGH |
| YELLOW | The Master sends this signal to Slave for it to change to YELLOW light. This signal is for the short blink between the transition from RED to GREEN. | HIGH |
| GREEN | The Master sends this signal to the Slave for it to change to GREEN light. | HIGH |
| LONG\_YELLOW | The Master sends this signal to the Slave for it to change to YELLOW light. The difference with the “YELLOW” message is that this one is longer as it comes right after the green in the sequence from GREEN to RED. | HIGH |
| ACK | This signal is returned by the Slave to the Master every time it receives a specific signal from the ones described above. When the Master receives this acknowledgement signal, it proceeds to the execution of the opposite instruction (Master changes to RED as Slave changes to GREEN). | Event |

A specific feature of this system is the TIMEOUT check. If the Master or/and the Slave don’t receive a signal for a specific amount of time, both systems TIMEOUT and therefore change to blinking yellow light.

# Conclusion

The collection of diagrams gave us a clear view on how the system should work and how it should communicate. This made the building process a lot easier.

The creation of the state machines took us a long time with multiple iterations to ensure bottlenecks could not occur. The context diagram on the other hand was simple since there was not that much to say about it.

Before building we could ensure the circuit was safe and after building we could also verify it, both with some simple calculations using Ohm’s law and by calculating the power.

Coding with keeping the diagrams in mind is more smooth and less error-prone than just starting and changing the code later. After finishing the code and perfecting the communication we could ensure the system worked, even when a hazardous situation occurred. For example, if the connection would fail.

# References

Hilderink, G. (2022). *Assignments.* Retrieved from Canvas: https://fhict.instructure.com/courses/12214/assignments/198481

Hilderink, G. (2022). *Basic Electrical Concepts.* Retrieved from Canvas: https://fhict.instructure.com/courses/12214/files/1609799?module\_item\_id=759914

technologies, S. (2008, July 28). *sunrom-443700.* Retrieved from Canvas: https://fhict.instructure.com/courses/12214/files/1586961?module\_item\_id=759955

Vishay Semiconductors. (2008, July 18). LED TLHx640. Retrieved from Canvas: https://fhict.instructure.com/courses/12214/files/1586967?module\_item\_id=759959