TÉCNICO LISBOA

Distributed Real Time Control Systems

Electrical and Computer Engineering

Project First Stage - Report

Student:

Alexandre Reis (100123) alexandre.leite.reis@tecnico.ulisboa.pt

2023/2024 – Second Semester, Q3

1 Introduction

1.1 Problem Description

The objective of this project is to design and implement a distributed controller that maintains the correct lighting at each desk in an Office Building.

For the First Stage of this project, the controller will be designed for each desk independently. In this stage a CAN-BUS communication system will also be developed and tested. These are the two basic building blocks to build upon and achieve a real-time distributed controller for this system.

Each desk is made of a luminance sensor and a LED, both pointing at the desk. This not only ensures that the LED is mainly used to iluminate the desk, but also that the measured luminance is the luminance reflected from the desk, which is the relevant control parameter in this case.

In this First Stage of the project, a Serial Communication protocol between the Raspberry Pi Pico and the computer was developed. This enables the user to change and get several parameters and metrics relevant to the problem in real time.

For the Second Stage (not developed in this report), the above mentioned CAN-BUS communication protocol is used to communicate between the Raspberry Pi Picos and achieve distributed control.

In both stages, the control objective for the system is to follow a given Luminance Reference at a desk, while maximizing the user's confort and minimizing the energy consumtion.

1.2 Assembly

To simulate an Office Space, an opaque shoe box was used.

The measured luminance in each desk corresponds to the light reflected from the table (box's lid). To maximize this reflection, the box was cover with white paper.

Each desk consists of a Raspberry Pi Pico, a LED (actuator) and an LDR (sensor). These are connected using a Breadboard, resistors and a Capacitor (1a).

In order to see and evaluate the user confort while using this desk ilumination controller, a small windows was cut out on the side of the box. This window is movable, elabling for it to be closed when not nedded, ensuring proper outside light isolation. This window also enables for disturbances to be inputed in the system, with and external flashligh.

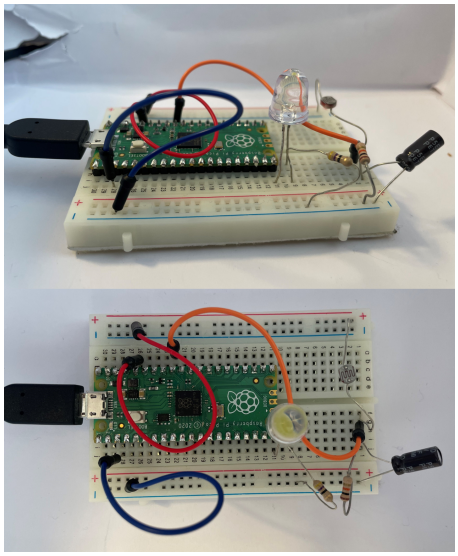
The CAN-BUS assembly will be shown and discussed later on in this report.

2 System Identification

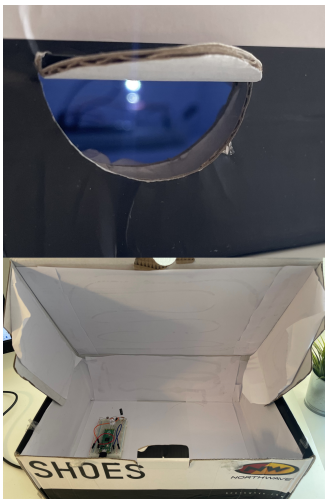
2.1 Luxmeter

The Luxmeter consists of an LDR (Light Dependent Resistor), connected as a voltage divider to an RC circuit. The capacitor in this circuit acts as a low pass filter, stabilizing the Analog Input value.

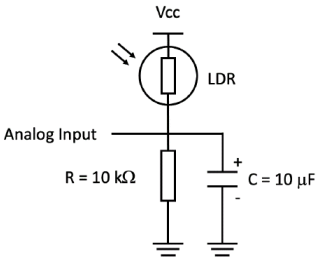
The relation between the LDR's resistance and the luminance (LUX) hitting it is defined by:



(a) Raspberry Pi Pico assembly with LED and LDR



(b) Shoe box covered with white paper and disturbances cut-out window



(c) LDR circuit

Figure 1: Assembly of one luminaire

$$log_{10}(R_LDR) = m * log_{10}(LUX) + b \tag{1}$$

Analysing the voltage divider circuit (2), and assuming steady state (constant LUX inside the box): $(\dot{v} = 0)$ we get the following equation:

$$\dot{v} \cdot \tau(x) = -V_{ss} + V_{cc} \cdot \frac{R}{R + R_{LDR}} \qquad R_{LDR} = R \cdot \frac{V_{cc} - V_{ss}}{V_{ss}} \tag{2}$$

As the value of $R = 10k\Omega$ and $V_{cc} = 3.3V$, it is possible to calculate the R_{LDR} value.

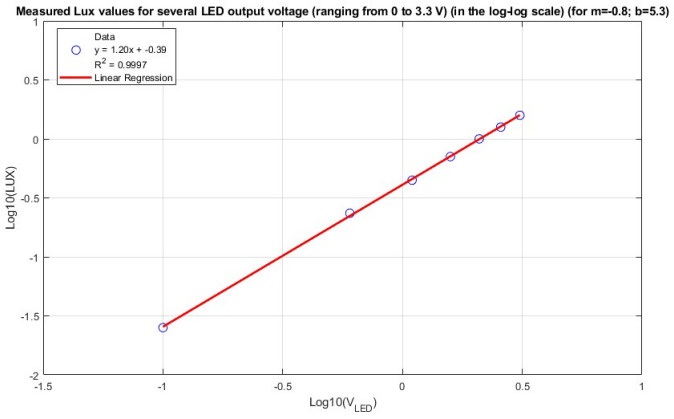


Figure 2: Measured Lux values for several LED output voltage/PWM (ranging from 0 to 4095) (in the log-log scale) (Graph is for m=-0.8; b=5.3; b is not the final selected value)

In order to use the circuit to estimate the luminance value (LUX), the m and b parameters are needed. The most important aspect is to select an m value that ensures a direct proportional

relation (in the log-log scale) between the control action voltage outputted to the LED and the measured LUX. This relation comes from the 2 facts: the LUX in the box is directly proportional to the voltage at the LED; equation 1. Several values of m were tried, and the value $m = -0.8$ was chosen, as it ensures a very high R^2 (figure 2), ensuring the wanted relation between variables.

The b constant was chosen using the LDR's datasheet [3] (PGM5659D). This datasheet says that at 10 LUX, the LDR should have resistance of 150-300$k\Omega$. As there was no Luxmiter avaliable, the median value of this interval was chosen (225$k\Omega$) and b was computed so that the measured LUX in this situation would be 10 LUX. The selected b value is: $b = 6.15$.

2.1.1 Time Constant τ

As stated in equation 2, the Luximeter's response in a transient state depends on the time. This is caused by the delay induced by the capacitor.

This parameter can be estimated theoretically using the RC circuit formula $\tau = R_{eq} * C$. Where C is the capacitor's capacitance and $R_{eq} = \frac{R1 * R_{LDR}}{R1 + R_{LDR}}$.

This parameter can also be measured experimentally using the method described in Module 5 of the Lecture Slides [1]. These slides state that τ is the time the sensor's system takes to reach 63% of the final measured voltage.

Tau values (s)		**Variation of u (PWM)**						
		500	**1000**	**1500**	**2000**	**2500**	**3000**	**3500**
Initial u (PWM)	**0**	0.341	0.400	0.361	0.320	0.270	0.195	-
	500	0.141	0.170	0.165	0.160	0.150	0.150	0.146
	1000	0.135	0.135	0.130	0.125	0.120	0.105	-
	1500	0.111	0.130	0.105	0.120	0.125	-	-
	2000	-	0.110	0.126	0.120	-	-	-

Table 1: Experimental values for τ

Theoretical Tau values (s)	0.0937	0.0890	0.0850	0.0811
Final u value (PWM)	1000	2000	3000	4000

Table 2: Theoretical values for τ

The measured values in table 1 differ significantly from the theoretical ones in table 2. But it is possible to identify that in both cases, for higher u values, the Tau decreases. Besides this, from the experimental data, when the LED is initially turned off, the LDR takes a lot more time to react (bigger τ) than otherwise.

As most of the relevant reference changes in the controller are between non-zero control values, for the rest of this report, $\tau = 0.13$(s) is the assumed value.

2.2 LED actuation

The LED actuation is done thought a PWM signal (Pulse width modulation). This means that, in order to control the average voltage received by the LED, the Raspberry Pi Pico outputs

"variable-width pulses to represent the amplitude of an analog input signal". In practise, the output is defined by a Duty Cycle value u (ranging from 0-4095 for this case), which corresponds to the percentage of time that the LED is receiving 3.3V (and to the percentage of 3.3V that the LED circuit receives).

In order to Identify the System in analysis, it is important to refer that the LED (actuator) has an approximately instantaneous response to a the given duty cycle (control variable u).

It is also important to refer that the LUX (x) inside the box have a directly proportional relation to the given duty cycle (u):

$$X(t) = G * u(t) + o(t) \tag{3}$$

Where G is the Gain that describes the Box and LED system behaviour, and $o(t)$ is the external luminance.

Measured Gain (G)	0.0096	0.0110	0.0119	0.126
u (PWM)	1000	2000	3000	4000

Table 3: Measured Gain values using

These 2 value can be estimated by measuring 2 points (one measurement taken with LED turned off, and one with LED turned on with some value u). These measurements need to be made after the system reached steady state, in order for the measured LUX value from the LDR to equal the LUX value inside the box. Some measurements of G are shown in table 3.

As the Gain value is very dependent on the reflection conditions of the box and the direction of the LED and LDR, this value is measured in the setup process when the controller boots up. This measurement is done with u=0 and a big value of u (for example u =4000), in order to increase it's the accuracy.

One final relevant parameter to take into account is the impact of the direction of the LED and LDR in the system's Gain value. As the amplitude of the light emitted by the LED is very limited (50% Power Angle of 8^{o}), it needs to be perfectly perpendicular to the reflection surface and parallel to the LDR sensor in order to maximize the Gain inside the box.

3 Individual Luminaire Controller

3.1 Control Requirements

The defined requirements for this system's controller are:

- Very fast response to changes in reference;
- Slow response to disturbances (as to not blink the LED when the sensor is momentarily covered or hit with an external light source).
- Null Steady State Error between reference and measurements

3.2 Feedfoward control

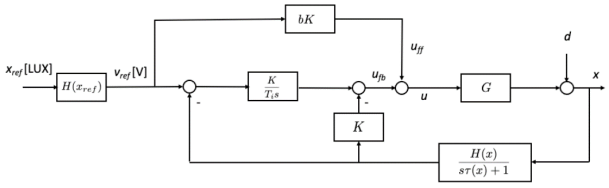


Figure 3: Luminaire system with Feedback PI controller

Using figure 3, it is possible to deduce the Feedfoward set-point transfer function:

$$\frac{X}{X_{ref}} = \frac{1}{G * bK * H(x_{ref})} \tag{4}$$

In order to make x (LUX) inside the box follow the given reference, it is obvious, from the given TF, that the control variable must be defined as: $bK = \frac{1}{G*H(x_{ref})}$.

As this is a feedfoward controller, it should have a non-null Steady State Error , as the system's model is not perfect and cannot respond accordingly to external disturbances.

3.3 Control loop variables

A feedback controller uses measurements from the sensor to achieve a null Steady State Error. In order to compare these measurements with the reference and compute a control action, these two should be in the same units.

For this reason, as the sensor's output is a value in volts corresponding to the ADC port's measurements, the reference X_{ref}(LUX) should be converted into this corresponding value. This means that v_{ref}[V] should be the measurement in volts from the sensor if the box would be at X_{ref}(LUX).

$$H(x_{ref}) = \frac{v_{ref}}{X_{ref}} = \frac{\frac{V_{cc}*R_1}{R_1+10^{m*log_{10}(x_{ref})+b}}}{X_{ref}} \tag{5}$$

Using a similar way of thinking, $H(x)$ is given by:

$$H(x) = \frac{V_{ss}}{X} = \frac{V_{ss}}{10^{\frac{log_{10}(R_1*(\frac{3.3}{V_{ss}}-1))-b}{m}}} \tag{6}$$

V_{ss} is the measurement in the Rasberry Pi's ADC port; R_1 is the resistance of the LDR's voltage divider; $V_{cc} = 3.3V$.

Finally, it is important to mention that the LED's dimming is controlled by a Duty Cycle value u (ranging from 0 to 4095)m, which is the control variable. Besides this, G is the Gain of the LED-BOX system, which means that X should be the actual luminance measure in LUX, inside the box.

3.4 Feedback control

A feedback controller should solve the steady state error problem. For this system, a PI controller was used, as there is no delay in the control action that justifies to consider a derivative controller (LED and LUX inside the box respond instantaneously to control action u).

From figure 3, the following Transfer functions were deduced (set-point and disturbance TFs):

$$\frac{X}{X_{ref}} = \frac{(bT_i s + 1) * (s\tau + 1)}{(s\tau + 1) * \frac{T_i s}{G*H*K} + T_i s + 1} \tag{7}$$

$$\frac{X}{D} = \frac{\frac{T_i}{G*H*K} s}{\frac{T_i}{G*H*K} s + 1} \tag{8}$$

Note: $\tau = T_i$ was assumed for equation 8. This is a common starting point for the T_i value in systems with dominant first-order dynamics.

Analysing the set-point TF, for it to be equal to one in steady state, $bK = \frac{1}{G*H(x_{ref})}$ (as was the case with the FeedFoward controller).

Analysing the disturbances TF, it is possible to conclude that (when $\tau = T_i$) it is a first order TF, with a single pole on $s = -\frac{G*H*K}{T_i}$. This means that, to achieve a slower response to disturbances, the controller should have: small K and big T_i.

3.5 Simulink analysis

With an initial idea of what the controller parameters should be, a Simulink analysis of the system was performed.

This simulation used: reference changes at t=4s and t=8s; step disturbance of 10 LUX at t=10s; sinusoidal disturbances with 5 LUX of amplitude all the time (only used for comparing K and T_i values).

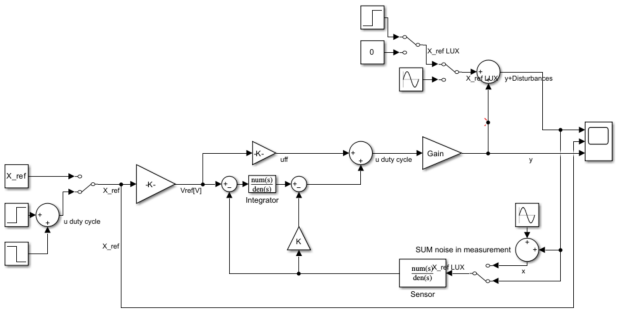
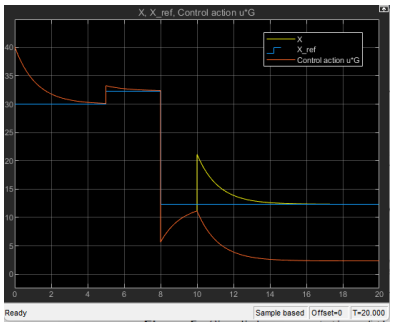


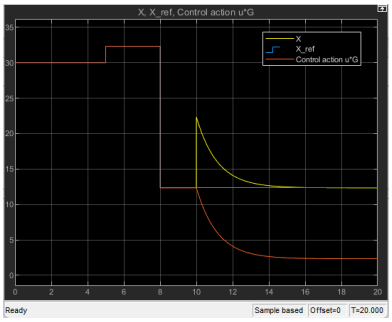
Figure 4: Simulink representation of the control loop

Conclusions from simulation:

- In order for step response to be instantaneous and very good, b parameter should be exactly equal to $\frac{1}{GKH(x_{ref})}$ (making the set-point TF equal to 1)
- Smaller K values should be used to decrease the effect of small disturbances in sensor reading

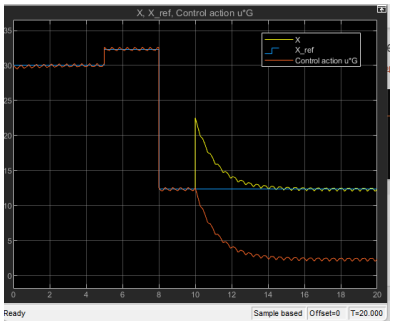


(a) Using control parameter b different from $\frac{1}{GKH(x_{ref})}$

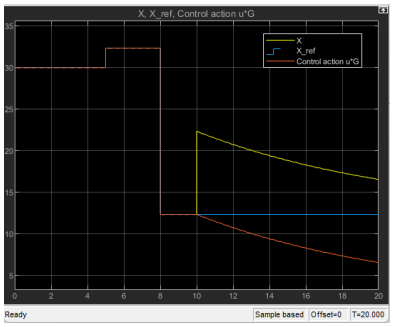


(b) Using control parameter b equal to $\frac{1}{GKH(x_{ref})}$

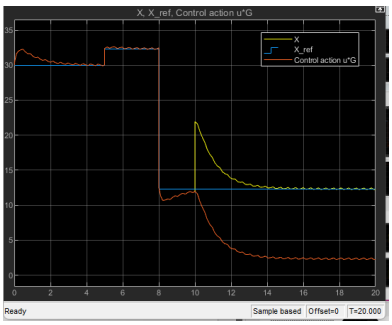
Figure 5: Simulation result comparison for control parameter b



(a) K=750, $T_i = \tau$



(b) K=7.5, $T_i = \tau$



(c) K=75, $T_i = \tau/10$

Figure 6: Simulation result comparison for control parameters K and T_i

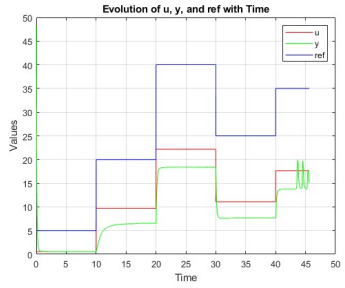
- If smaller K and bigger T_i values are used, the response of the system to steady state errors is slower. This decreases the performance of the controller with step responses if the b parameter is not well tuned.
- Finally, when fine tuning the controller to achieve a practical good performance only K and T_i should be tuned (mainly K). Taking care to control overshoot that can appear if T_i =!τ.

4 Experimental results and PI tuning

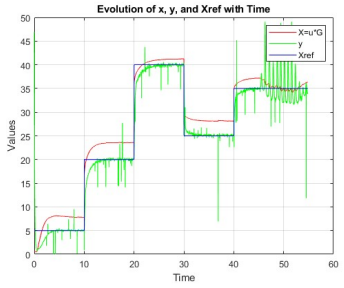
Experiments were made with different controllers. For all of them, changes in reference were made in t=10,20,30 and 40 (s). After 45s, external disturbances were induced in the system with a flashlight using the box's window.

4.1 Feedfoward control

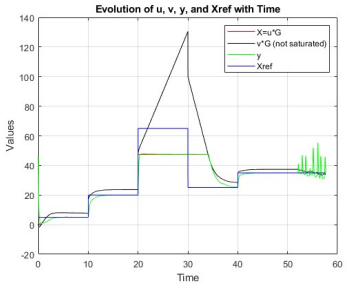
As expected, when not using feedback control, there exists a non-null steady-state error(section 3.2), due to imperfection in the system identification. (figure 7a)



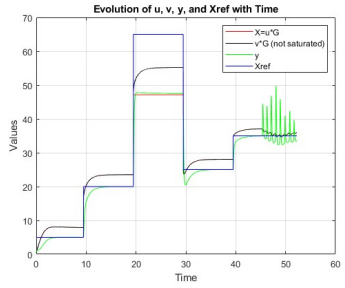
(a) Basic Feedfoward controller



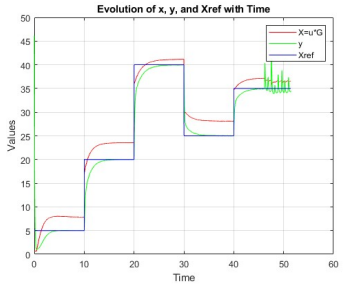
(b) Feedback controller with no digital filter



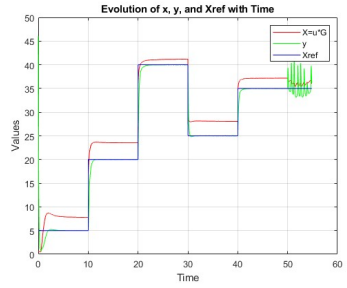
(c) Feedback controller with no anti-windup



(d) Feedback controller with anti-windup



(e) Feedback controller with $T_i = \tau$ and $K = 750$



(f) Feedback controller with $T_i = \frac{\tau}{100}$ and $K = 15$

Figure 7: Controller results for 1 luminaire (experimental) - y is the measured LUX value by the LDR; Xref is the defined reference in LUX; X=u*G is used to show the stability of the control action

4.2 Digital filter

As suggested in the project assignment [2], a digital filter was used to implement a low-pass filter on the LDR readings (this further stabilizes the LDR readings, in conjunction with the use of the RC circuit).

This filter measures 20 readings and takes the median out of them.

Analysing figures 7b with all other figures, it is possible to see that the filter made a big difference in the stability of the measurements.

4.3 PI Controller

The values that were estimated theoretically and using Simulink (figure 7e) showed very good rejection of disturbances, as the control action is almost constant. The problem with this controller is that the system takes too much time to reach steady state after a change in reference. This means that:

- The b coefficient is not completely correct, which makes the step response not instantaneous (unlike the simulation's results). This was to be expected as the system identification is not perfect
- the integral part of the controller $\frac{K}{T_i}$ is too low.

A lot of tuning was done to K and T_i. The best controller was the one in figure 7f, using $T_i = \frac{\tau}{100}$ and $K = 15$.

This controller has a maximum settling time of 2s, with most settling times being bellow 1.5s.

This controller also has almost non-existing overshoot.

This controller uses bumpless transfer, which enables for there to not be a step when the control parameter b changes in real time (which happens when the reference changes). This is performing well and no step is detectable.

4.4 Anti-Windup

Anti-windup is implemented in the final version of the controller.This enables the Integral part of the controller to not accumulate over time when the control action saturates. The chosen T_t parameter for this function was 1 (s). This enables the Integral part of the controller to discharge in up to a second.

Anti-windup performed very well when comparing to not using it. (figures 7d and 7c)

4.5 Possibility of improvement

There are 2 factors that could improve the controller:

- Adaptable T_i. This would improve the behaviour of the system as it depends on τ which is not constant (as shown in 2.1.1). Ideally, depending on the reference step, different values for τ would be used.
- Adaptable K based on the current reference.

These were not implemented due to: the final controller being sufficiently good given the defined requirements; lack of time.

4.6 Metrics

The 3 metrics are computed using a class that is updated at each loop iteration. This uses the formulas given on the Project Assigment.

For estimation P_{max}, the LED was turned on with it's maximum value (PWM is a continous 3.3V signal) and the voltage drop of the resistor was measured. With this value, the Current Intensity was calculated, and multiplied by the 3.3V. $P_{max} = 0.021W$.

For the final controller and routine (figure 7f): Energy consumption= 1.043J ; Visibility Error=0.333 LUX; Flicker= 0.0068 s^{-1} .

4.7 Real time calculations, Jitter and Sampling Time

The control loop was implemented in a non-blocking manner.

The sampling time was defined in the Project Assignment (100Hz). This sampling time is followed without Jitter as all the computations in the loop take less time than the sampling time.

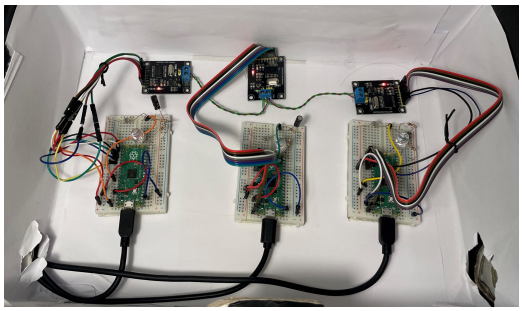
Function	Measure LUX, Compute control and Actuate LED	Metrics	Serial input
Time (ms)	0.410 to 0.450	0.063 to 0.068	1.2 to 1.9

Table 4: Computation time of control loop

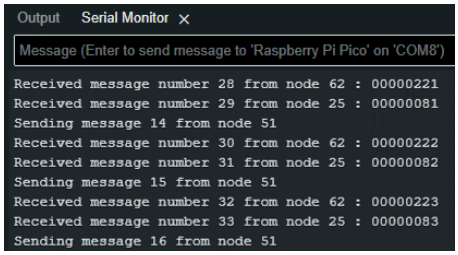
4.8 Serial Interface

A serial interface was implemented using object oriented programming, following the commands defined in the Project Assignment. This enables the user to: set most variables relevant in the controller; get their values; and get metrics. This also enable the user to set the desk as occupied or not occupied (changing the luminance reference).

5 CAN-BUS



(a) Assembly



(b) Messages sent and received with 3 nodes at 1 Mbit/s

Figure 8: CAN-BUS implementation

CAN-BUS test were made. As figure 8b shows, the communication between the 3 nodes is reliable using the maximum bit rate of 1 Mbit/s. This is expected as the length of the cables is shorter than 30cm [1].

Tests were also made to try to understand the delay evolved with sending and receiving messages. For that a program was run, where a node sent a message, another node received it and responded. The total elapsed time was calculated. This corresponds to files "CANBUS_test_latency.ino" and "CANBUS_test_latency_node2.ino".

The time it takes for the message to be sent and received is: 0.510 to 0.570 (ms). This is longer than expected ([1] expects 0.150ms*2 for this operation), but it can be justified with the processing time of the Raspberry Pi Pico and the wait time in the can-bus buffers.

References

[1] Alexandre Bernardino. Distributed real-time control systems - lecture slides, 2023-2024.

[2] Alexandre Bernardino. Real-time cooperative control of a distributed illumination system - project assignment, 2023-2024.

[3] Token. Pgm5506 datasheet - light-dependent photoresistors for sensor applications.