APRI0007 - Major project in electrical engineering

The V-LUXE - Final report

Academic year 2024-2025

Julien	Victoria	Guillaume	Matthieu	Arnaud
Boniver	Filée	Eyen	Gillard	Innaurato
s212020	s210904	s182398	s211883	s211234

Contents

1 Introduction **2**

2 Management **3**
2.1 Project charter . . . 3
2.2 Project execution plan . . . 4
2.3 Risk assessment and risk mitigation . . . 4
2.4 Backward analysis . . . 4

3 Mechanical design **7**
3.1 Structural frame . . . 7
3.2 Assembly of components on the structure . . . 7
3.3 Final assembly . . . 7

4 Electronics **10**
4.1 Microcontroller . . . 10
4.2 Printed Circuit Board . . . 10
4.3 Power delivery . . . 10
4.4 Buttons . . . 11
4.5 Temperature and humidity sensors . . . 11
4.6 QE circuit . . . 11
4.7 Light sensor . . . 12
4.8 Display . . . 13

5 Software **14**
5.1 I^2C management . . . 14
5.2 Tasks . . . 15
5.2.1 Initialization . . . 15
5.2.2 Main . . . 15
5.2.3 Security . . . 15
5.2.4 Buttons . . . 16
5.2.5 Sensors . . . 16
5.2.6 Decision . . . 17
5.2.7 Servo . . . 18
5.3 Schedulability . . . 18
5.4 Processor utilization . . . 20

6 DC motor control system **21**
6.1 Description of the control . . . 21
6.2 System analysis . . . 21
6.2.1 Laws governing DC motor . . . 21
6.2.2 Transfer function in the Laplace domain . . . 22
6.2.3 Determination of the physical parameters . . . 22
6.3 Controller design . . . 23
6.4 Implementation of the control system . . . 23

7 Improvements **26**

A Software **27**

B Videos **27**

1 Introduction

This project was carried out by a team of 5 students in the scope of their first year in MSc of Electrical Engineering. The goal of this year-long project is to develop a working electronic device from scratch, including software written in assembly on PIC microcontroller.

This project aims to design and build a scale model of a tilting window equipped with an electronically controlled blind. The system will support two modes of operation: automatic and manual.

In automatic mode, the window and blind will open or close based on sensor data and user-defined threshold values. The goal is to automatically regulate parameters such as temperature or brightness to maintain the desired indoor conditions.

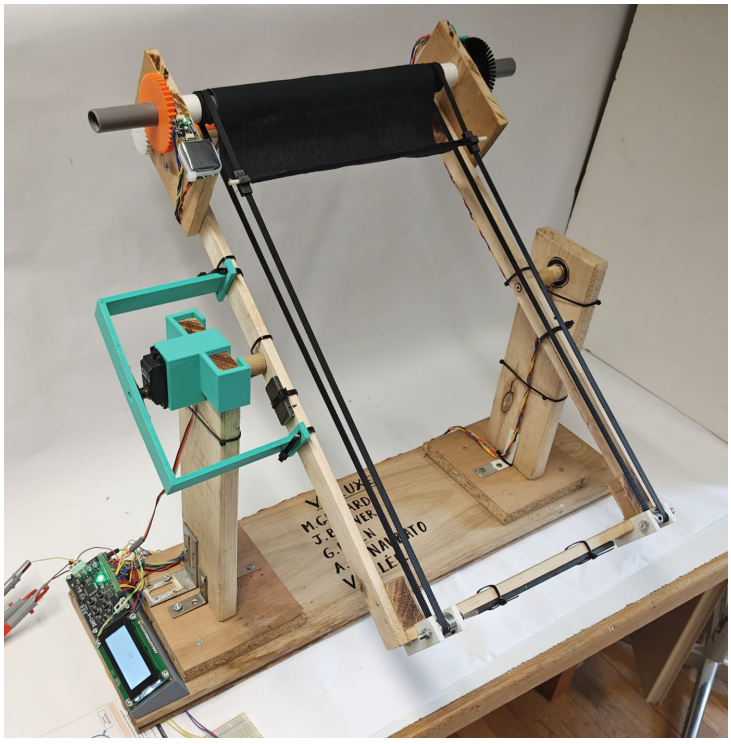
In manual mode, the user has full control over the window and blind, allowing them to be adjusted as desired. In this mode, the system will not respond to changes in the environment.

A control system has been implemented to have a blind which is not sensitive to external disturbances (in both modes).

The system will feature a control panel equipped with an LCD screen and buttons. This interface will allow the user to select the operating mode and customize settings as wanted.

This report outlines the whole project components including management, electronics, mechanics, software and control system. Some improvement paths are also developed. Associated with this project report, the software code and some videos are provided in respectively Appendix A and B.

We would like to express our gratitude to everyone who contributed their valuable insights and assistance to this project. This project made us slightly more comfortable with project organization and electronics development. This project also strengthened our bond as colleagues and friends.



2 Management

The management part of a project is essential and needs to be carried out from the start till the very end. Through its different parts, it helps the team having an overview of the objectives, the schedule, the cost and the risks. Managing a project can be performed by following different steps:

- **Project charter**: The project charter is the first step when initiating a project. The goal is to outline its objectives, scope and participants. This provides clarity to investors and is essential for project success.
- **Project execution plan**: The project execution plan is the apriori planning done at the beginning of the project. It shows a graphical visualization of the different tasks to be done and their desired execution date.
- **Risk assessment and risk mitigation**: During the creation of a project, some potential risks can be found. It is essential to list all the possible risks and some solutions to mitigate them. This process must be updated as soon as new risks are identified.

In the following sections, the different project management tasks carried out during this project are developed. The final section is dedicated to a backward analysis of the project progress and a comparison with the initial specifications.

2.1 Project charter

Project Purpose or Justification

Develop an automatic tilting window and blind in order to regulate a set of parameters according to different weather situations, both automatically and manually via remote control by the user. This project incorporates a feedback-controlled function that maintains the desired blind setting. The software will be written in assembly language. The software architecture will involve concurrent tasks to receive real-time measurements from the sensors and control both the window and the blind at the same time.

Project Description

This project aims to implement a scale model of a tilting window and its blind that can be remotely controlled and can achieve automatic operations according to different weather parameters. The lighting, temperature and humidity are continuously measured. The window angle and the blind opening status will be set accordingly.

Project and Product Requirements

- The project involves creating a scale model of a window, approximately 50 cm by 30 cm in size.
- The model will be alimented by using the DC generator of the laboratory.
- The model includes outdoor and indoor temperature and light sensors.
- The window will be moved by a servomotor roughly located abeam the center of gravity to minimize the needed torque.
- The blind will be moved by a motor monitored via QEI. The window and the blind should be able to move in a reasonable duration.

- Automatic control of the window and blind to achieve specific requirements set via the thermostat.
- The user can manually select the desired window angle and blind position no matter the measurements of weathers conditions.
- The blind should remain in its defined position despite undergoing external perturbations.

Secondary requirements

- Humidity sensor to detect external rain, internal steam due to cooking, etc.
- Noise sensor to close the window if it's too noisy outside.

2.2 Project execution plan

The project execution plan can be seen in Figure1. This figure shows the year calendar and the initial planning of the different tasks to be done.

2.3 Risk assessment and risk mitigation

All the possible risks are listed in Table 1. Potential problems have been associated with these risks, and a metric of importance has been used to measure them. This metric is a function of the severity, the occurrence and the ability to detect the given problem. Mitigation actions have been developed to reduce the importance of the highest-priority problems.

2.4 Backward analysis

At the end of this project, it makes sense to refer back to the project charter to ensure that all the specifications have been implemented. A comparison of the initial requirements and the final project shows that all of the specifications from a few months ago are included in the final project. The noise sensor was the only secondary requirement that was not included. It was deemed that this feature was irrelevant to this project.

Considering the initial planning, the final deadline for the report and working device has been met. Nevertheless, the different tasks were organized quite differently during the project progress. In fact, some tasks took much longer than expected (I^2C protocol implementation, getting started with assembly...) and some other essential tasks were not listed at the beginning of the project (design of the control system...). A preliminary plan was essential for ensuring that the project was started on time. However, the project's core organization was mainly done through physical meetings instead of following a given planning. Team communication was a key to get a complete project done in time.

The risk management table was followed throughout the implementation process. All of the actions that were mentioned to mitigate the different risks have been implemented for this project. The analysis shows that, after the actions taken, no risks with a Risk Priority Number (RPN) higher than 100 are listed. This simple condition is sufficient to confirm that the risks and their effects have been properly reduce.

Figure 1: Initial planning of the project progress

Block	Function	Problem	S	P	D	RPN	S'	P'	D'	RPN'	Actions Taken
Power Supply	Provide 5V DC	Overvoltage (>5V)	8	7	2	112	1	7	2	14	Add fuse and voltage regulator
		Undervoltage (<5V)	7	5	2	70	-	-	-	-	-
		Bouncing signals	8	7	7	392	1	7	7	49	Add fuse and voltage regulator
Microcontroller	Control logic	Crash during execution	8	3	3	72	-	-	-	-	-
		Current overload	8	4	6	192	2	4	6	48	Add fuse
		Insufficient memory	7	3	7	147	5	2	7	70	Change microcontroller model
Sensors (L/T/H)	Measure environment	Communication loss	7	3	8	168	7	3	2	42	Display error message
		Wrong measurement	7	1	8	56	-	-	-	-	-
		Obstructed sensor	3	5	6	90	-	-	-	-	-
		Overheating	8	4	8	256	8	4	3	96	Shut down system and display alert
		Too sensitive (sunlight)	6	9	2	108	6	7	2	84	Add tinted film sun shield
LCD Screen	Interface	No display	8	7	1	56	-	-	-	-	-
		Wrong characters	7	6	1	42	-	-	-	-	-
		Button bounce	5	7	3	105	2	7	3	42	Add debounce
DC Motor	Move blinds	Unstable speed	6	6	2	72	-	-	-	-	-
		Too fast	8	5	2	80	-	-	-	-	-
		Too slow	4	5	2	40	-	-	-	-	-
		No rotation	7	6	3	126	7	3	3	63	Add torque multiplier (gears)
		No motor found	9	3	1	27	-	-	-	-	-
Servo Motor	Tilt control	Too fast	8	7	2	112	8	4	2	64	Add software stepping
		Too slow	7	3	2	42	-	-	-	-	-
		No response	8	5	3	120	8	2	3	48	Check wiring and behavior
		Overheating	8	3	5	120	8	2	5	80	Turn off on high current
QEI Sensor	Track position	Wrong position	8	5	3	120	8	3	3	72	Test before integration
		Communication loss	8	4	3	96	-	-	-	-	-
Mechanics	Support structure	Break under weight	8	5	2	80	-	-	-	-	-
		Gears break	8	5	2	80	-	-	-	-	-
		Friction too high	7	4	3	84	-	-	-	-	-
PCB	Wiring	Poor solder joints	7	4	5	140	-	-	-	-	Recheck connections
Current Sensor	Safety stop	Hand stuck in mechanism	9	7	2	126	3	7	2	42	Stop window on current spike

Table 1: Risk Management Table

3 Mechanical design

This course does not cover the mechanical part. However, this project includes quite a significant part of mechanical design. As it is not the main goal of the project, the rigor used in this domain is sufficient to obtain a structure that meets the needs, but does not aim for excellence. The mechanical design can be decomposed into two parts, the structural frame of the V-luxe and the assembly of components on the structure.

3.1 Structural frame

The design consists of a rectangle wood frame fixed to a support via two ball bearings, allowing it to rotate freely around the horizontal axis. The material choice aimed for simplicity and was free of charge. Two wood panels are fixed on the top of the frame to serve as support for the blind system and the external sensors. The structural frame can be seen in Figure 2.

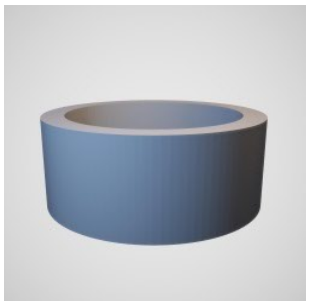
Figure 2: Wood structure

3.2 Assembly of components on the structure

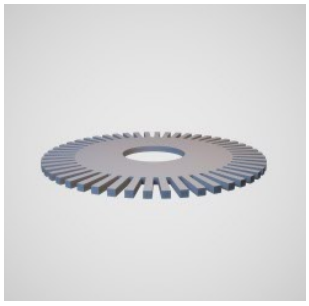
Most of the parts used to mount the sensors and actuators on the structure are 3D printed. The 3D printed parts were modeled using a CAD software and have been printed in ULiège's Fab52. These parts can be seen in Figure 3.

3.3 Final assembly

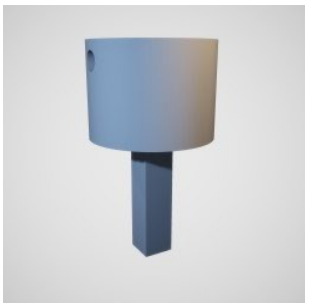
The blind system winds up around a PVC pipe and is guided by a 3D printed belt. Figure 4 shows the gear system linking the DC motor and the PVC pipe, which enables the blind movement.



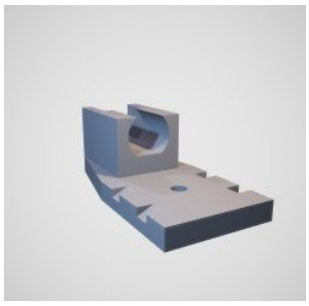
(a) Blind spacer



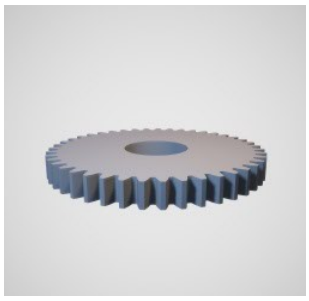
(b) Counting wheel



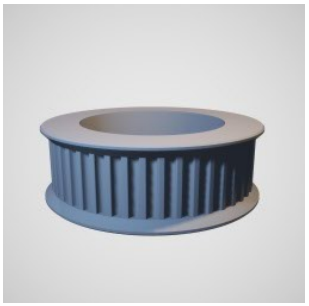
(c) Axle linking DC motor and gear



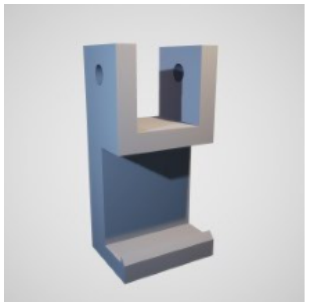
(d) DC motor support



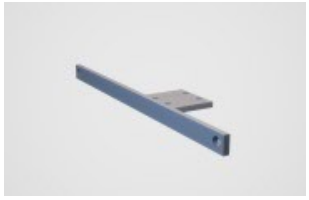
(e) DC motor gear



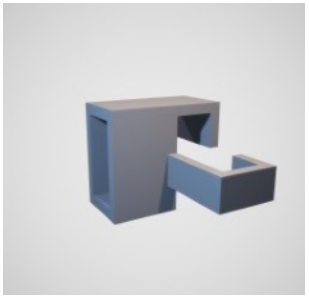
(f) Blind gear



(g) Pulleys support



(h) QE support



(i) Servomotor support

Figure 3: 3D printed objects present in the mechanical design

Figure 4: Gear system driving the blind, next to two external sensors

This project makes use of quite a lot of wires. They are arranged as cleanly as possible using zip ties, heat shrink tubing and hot glue. The window system is balanced using a metallic plate fixed on the wooden frame, to allow the servo to drive the system without too much effort.

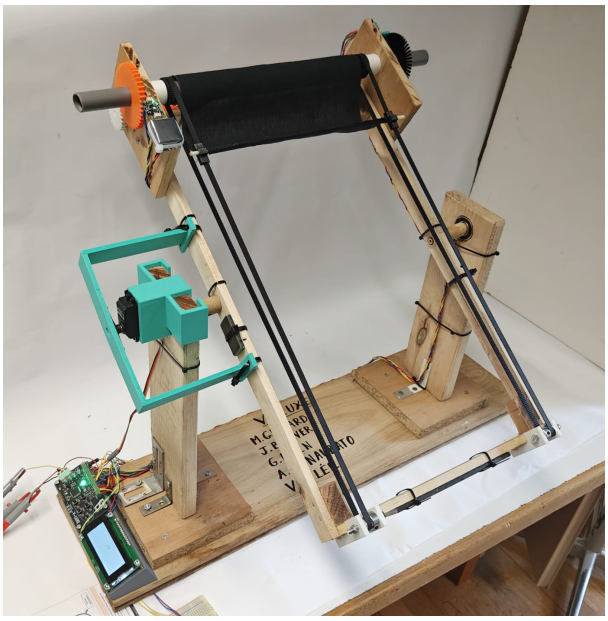


Figure 5: Final assembly

4 Electronics

This section details the motivations and reasoning that guided the electronic circuit design decisions throughout the project.

Only the aspects that are deemed the most relevant are discussed here; some standard practices and obvious implementations are omitted for the sake of clarity.

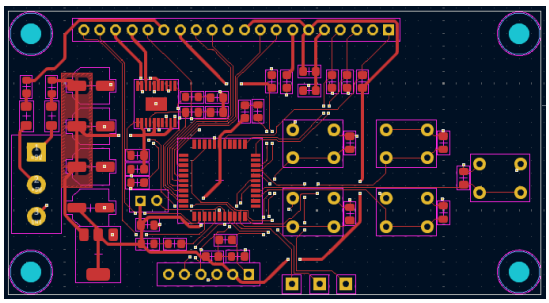
4.1 Microcontroller

The microcontroller used for this project is a PIC18F4331. It has been chosen for its embedded quadrature encoder interface (QEI) block. This microcontroller is responsible for coordinating all operations within the project.

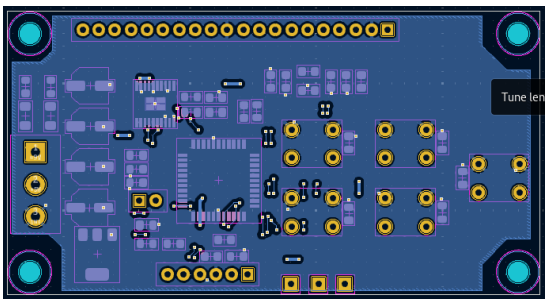
4.2 Printed Circuit Board

A printed-circuit board (PCB) has been designed to interconnect the microcontroller with the sensors and actuators. Due to the mechanical design of this project, the microcontroller is mostly connected to components located outside the PCB. These connections are made through an array of terminal block connectors.

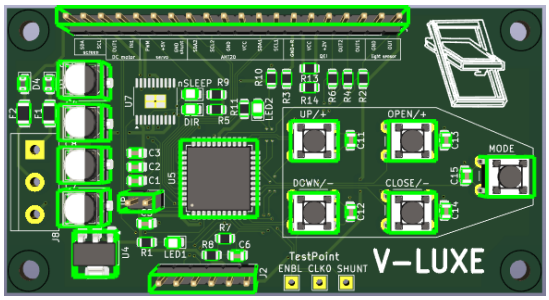
The PCB is composed of two conducting layers, one of which serves as ground plane. The PCB schematics and picture are shown in Figure 6.



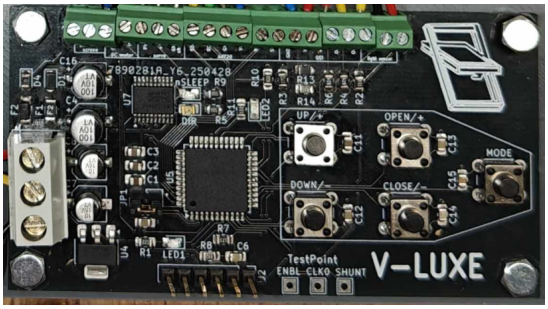
(a) Front copper layer



(b) Back copper layer



(c) 3D view from Kicad



(d) Real PCB after soldering all components

Figure 6: PCB Architecture.

4.3 Power delivery

The PCB is supplied with two DC voltage sources, one at 9V and the other at 5.5V. Both sources are connected to the same ground on the PCB.

A LM1117 linear voltage regulator (LDO), rated for a maximum output current of 800mA, is used to step down the 9V DC supply to a stable 5V, which serves as the VCC rail for the

entire PCB.

The 9V source is also used to power directly the DC motor via an H-Bridge, while all other components receive their supply through the regulated 5V output of the LM1117. To protect the circuit from damage due to reverse polarity or excessive current, a Schottky diode and a fuse have been placed at the input stage. These components ensure proper voltage polarity and provide overcurrent protection for sensitive electronics.

The 5.5V DC source directly supplies the servomotor that controls the tilt of the window. It has been chosen to use a second power source, since the servomotor is not compatible with 9V DC and it can draw a peak current that would not be supported by the LM1117 (see Table 2).

Component	Voltage	Peak current	Notes
PIC18F4331	5V	250mA	Via LM1117, likely <20mA in reality
Servomotor	5.5V	1400mA	External power
DC motor	9V	500mA	Through H-Bridge
LCD display (including I^2C module)	5V	100mA	Via LM1117
AHT20 (x2)	5V	50µA	Via LM1117
LDR + Voltage divider	5V	<1mA	Via LM1117, passive component
QE optical sensors (x2) + Voltage divider	5V	15mA	Via LM1117

Table 2: Power consumption of the different components

4.4 Buttons

To simplify the PCB design, the buttons are connected to the PORTB-group ports of the microcontroller, which can be configured as digital input with internal pull-up resistors. Physical pull-up resistors are therefore not needed, which saves space on the PCB. 100nF capacitors are placed in parallel with each button to prevent bouncing.

4.5 Temperature and humidity sensors

Two Adafruit AHT20 temperature and humidity sensors (see Figure 7) are used in this project. One is situated on the exterior of the window to detect changes in outside temperature and humidity. The other one is mounted on a moving support with long wires to simulate an indoor sensor.

These sensors communicate with the microcontroller via two separate I^2C buses, as they share the same fixed I^2C address (0x38), which cannot be modified. This separation is necessary to prevent address conflicts and ensure reliable data acquisition. Details about this communication mechanism, particularized for the AHT20 sensors, are given in section 5.1.

These sensors offer a typical accuracy of ±2% relative humidity and ±0.3°C, with a measurement range spanning from 0 to 100% RH and –40 to 85°C.

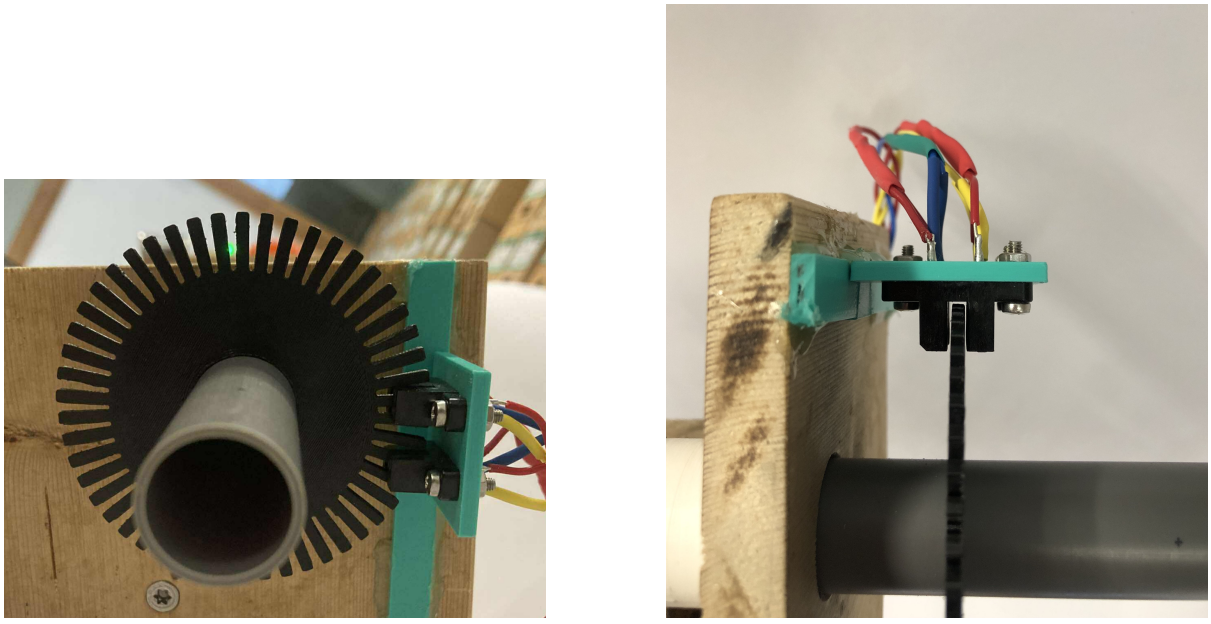
4.6 QE circuit

The quadrature encoder (QE) circuit is composed of two Vishay TCST2103 transmissive optical sensors and a 3D-printed counting wheel with 50 spokes (Figure 3b).
According to the TCST2103 datasheet, the maximum forward voltage provided to the emitting diode is 1.6V. A voltage divider is therefore used to provide the expected voltage. The theoretical resistors values are $R_1 = 500\Omega$ and $R_2 = 250\Omega$. However, in practice, the current going through the resistors of the voltage divider is not the same because of the LED present in the circuit. The theoretical resistor values were adjusted empirically to provide sufficient voltage to the LED. The resistors used in the final design have values of $R_1 = 100\Omega$ and $R_2 = 500\Omega$.

Figure 7: Adafruit AHT20 temperature sensor

The optical sensors are positioned in such a way that, when one is facing a slot, the other is aligned with the edge between a slot and a spoke. This configuration generates two square wave signals in quadrature, allowing the microcontroller to determine both the position and the direction of rotation of the wheel.
According to the TCST2103 datasheet, a 100Ω resistor must be placed between the output and ground. However, this was not successful. Empirical testing showed that a $1M\Omega$ resistor had to be placed between the ground and the output to read the output as expected.

The wheel is fixed on the shaft around which the blind is wound. This circuit forms the basis of the DC motor control system (Section 6). This assembly can be visualized in Figure 8



(a) Front view of the QEI

(b) Side view of the QEI

Figure 8: Front and side view of the QEI

4.7 Light sensor

The PDV-P5001 Light Dependent Resistor (LDR) is used to detect variations in ambient luminosity. Its resistance decreases with increasing light intensity. The LDR is part of a voltage

divider supplied with a fixed 5V. As the resistance varies, the voltage drop across the LDR changes accordingly. This voltage is measured at the divider's output and fed into one of the PIC's ADC channels, which converts the analog signal into a digital value for processing.

The sensor is placed in a 3D-printed enclosure, behind a translucent plastic screen to prevent direct exposure to sunlight, which could otherwise degrade its performance. The PDV-P5001 is designed to sense visible light in the 400–700nm wavelength range. The LDR can be seen in Figure 9.

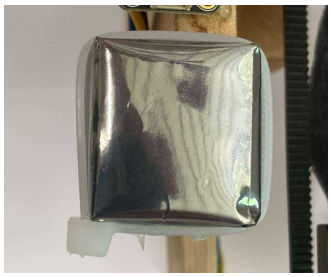


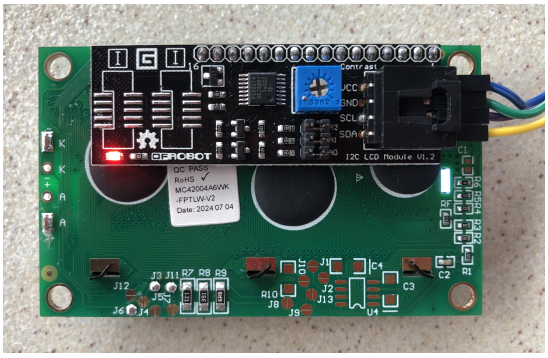
Figure 9: Light sensor protected with a plastic screen

4.8 Display

The display assembly is composed of a Midas Displays MC42004A6WK-FPTLW-V2 screen paired with a DFRobot I^2C LCD backpack, which is itself driven by a PCA8574 remote I^2C bus to 8-bit parallel port expander. This allows to communicate with the screen using the I^2C protocol.

In the scope of this project, the screen is used to display the desired indoor temperature. This temperature is initialized at 19 °C at power on and can be modified in automatic mode. When the variable is changed by the user, the screen is first cleared by an I^2C command, then its new value is displayed with the cursor approximately situated at the center of the screen.

(a) Front of the screen



(b) Back of the screen assembly with the I^2C backpack

Figure 10: Front and back view of the screen

5 Software

This section is dedicated to explaining the software at a high level of abstraction. It is mainly composed of previous reports sections as the final implementation reflects well the initial solution that was imagined.

Since this project deals with room temperature and relatively slow actuators, it does not require short response times and critical organization of tasks. The software architecture does not require preemption and the order of execution of the tasks can be constant. This is why the initial choice was to use a round-robin without interrupt software architecture. The tasks are sorted in two categories, the ones done at 50Hz and the ones done once every second.

However, soon after the start of the project, it was discovered that the PIC18F4331 does not support I^2C communication in master mode. Since communication with the temperature and humidity sensor is required, the protocol had to be implemented from scratch. Such a protocol needs to be fast and precise, and should therefore be executed within an interrupt routine. This is why the final implementation is a round-robin, with interrupts handling the I^2C protocol.

5.1 I^2C management

The fundamental principle of I^2C communication is that the clock (SCL) and serial data (SDA) lines are high by default and can be pulled low by either the master or the slave. To reproduce this behavior as accurately as possible, the microcontroller generates the clock at a frequency of 1.7kHz, and the SDA pin is configured as an input when the slave is sending data, and as an output otherwise.

The AHT20 Temperature and Humidity Sensor operates in two phases: first, a command is required to initiate measurement, after 80ms, it responds with the temperature and humidity values. Therefore, the microcontroller must first write to the sensor and then read from it.

To respect this delay and because the values are not expected to change rapidly, the communication timing is based on a slow timer set to 1Hz. Each time the timer flag is raised, a communication is initiated, alternating between write and read commands.

During communication, if the master is sending data, the SDA line is set during the second half of the clock's low level, so that the sensor can read the value on the following rising edge. In reading mode, with the line now configured as an input, the value is sampled on each rising edge of the clock. Moreover, in write mode, after every 8 bits sent, the SDA line is released to allow the slave to acknowledge the byte. Conversely, in read mode, the master takes control of the line again to send the acknowledgment bit.

Several critical issues were encountered during the implementation phase of the code:

- **Fake START or STOP**: if the value of the SDA line is modified during a high level of the clock, this will be interpreted as a start or stop signal be the slave and the communication will collapse. This was thus highly monitored during the process.
- **ACK timing**: because the SDA line changes direction before an acknowledgment the timing is crucial. This causes many issues.

5.2 Tasks

5.2.1 Initialization

```
Task INITIALIZATION
    Variables initialization
    ADC, DC PWM, Servo PWM, QEI and Buttons initialization
    Interrupts initialization
end Task
```

5.2.2 Main

The Round-Robin architecture's infinite while loop is implemented here. As mentioned earlier, the tasks are organized into two categories. This is implemented using a single timer at 3.4kHz that is divided into a the 50Hz and 1Hz flags. The controller part is explained later in Section 6.

```
Task MAIN
    while 1 do
        if flagFast = 1 then
            Security();
            Controller();
            Buttons();
        end if
        if flagSlow = 1 then
            Sensors();
            Decision();
            Servo();
        end if
    end while
end Task
```

5.2.3 Security

This task was originally designed to detect potential obstructions blocking the servomotor by monitoring the current drawn through a shunt resistor. The intended response was to fully open the window upon detecting an overcurrent event, indicative of a blockage. The voltage drop across the shunt was read via the corresponding ADC result register.

However, after carrying multiple test, it was found that the servomotor uses the same current when rotating as when it is blocked. This behavior suggests that the servomotor lacks sufficient power and therefore operates at near-maximum load continuously. A more powerful servomotor would likely allow for more accurate blockage detection through current sensing. Due to time and budget constraints, the motor was not replaced. Nevertheless, the associated detection code has been retained (although currently non-functional) to facilitate future integration of a suitable replacement.

Task SECURITY
  if $i > i_{th}$ **then**
    NOP (to be changed with a new servomotor);
  end if
end Task

5.2.4 Buttons

This task checks if a button is pressed by the user and actualizes the global variables accordingly. There are three types of buttons on the control panel:

- The "Automatic" and "Manual" buttons, for the user to change the mode of utilization.
- The "↑" and "↓" (resp. up and down) buttons to either change the desired temperature of the user (in automatic mode) or control the blind (in manual mode).
- The "+" and "−" buttons (resp. plus and minus) buttons to change the opening of the window in manual mode.

Task BUTTONS:

automatic_button is pushed:
if $Mode = 1$ **then**
  $Mode \leftarrow 0$
end if
....
minus_button is pushed:
if $Mode = 0$ **then**
  Return
end if
if $Mode = 1$ **then**
  Update $ref_{position\_serv}$
end if

5.2.5 Sensors

This enables reading of all the sensors. The luminosity sensor being a variable resistor, its value is read in an ADC result and if this value exceed a threshold, a flag is raise to say "it is currently sunny outside".

The temperature and humidity are obtained as explained in section 5.1. This function sets the flag to launch the I^2C communications.

Task SENSORS
  Launch I^2C for temperature and humidity;
  if $Ligth > Light_{th}$ **then**
    Raise sunny_flag;
  end if
end Task

5.2.6 Decision

This task drives the automatic mode. Given the user preferences and the outside/inside conditions, this function controls the window and the blind. For the blind, the flag set in the previous task for the luminosity is used. Figure11 represents how the decision is made. By default, if it's not sunny outside, the blind is open to avoid darkness inside.

About the window, humidity and temperature are considered. Figure 12 represent the decision making. First, humidity extreme case is checked, for example if the user is cooking and the inside humidity is high, the window should open if the external humidity isn't high too. Then if humidity is not critic or can't be changed, the internal, external, and user temperature are compared to find out if the best solution to approach the user desired temperature is to either open or close the window.

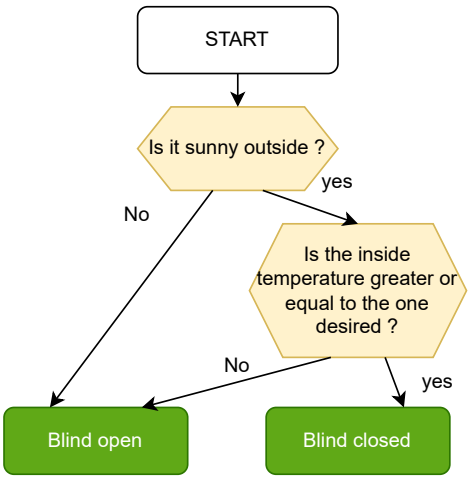


Figure 11: Decision-making flowchart for the blind control

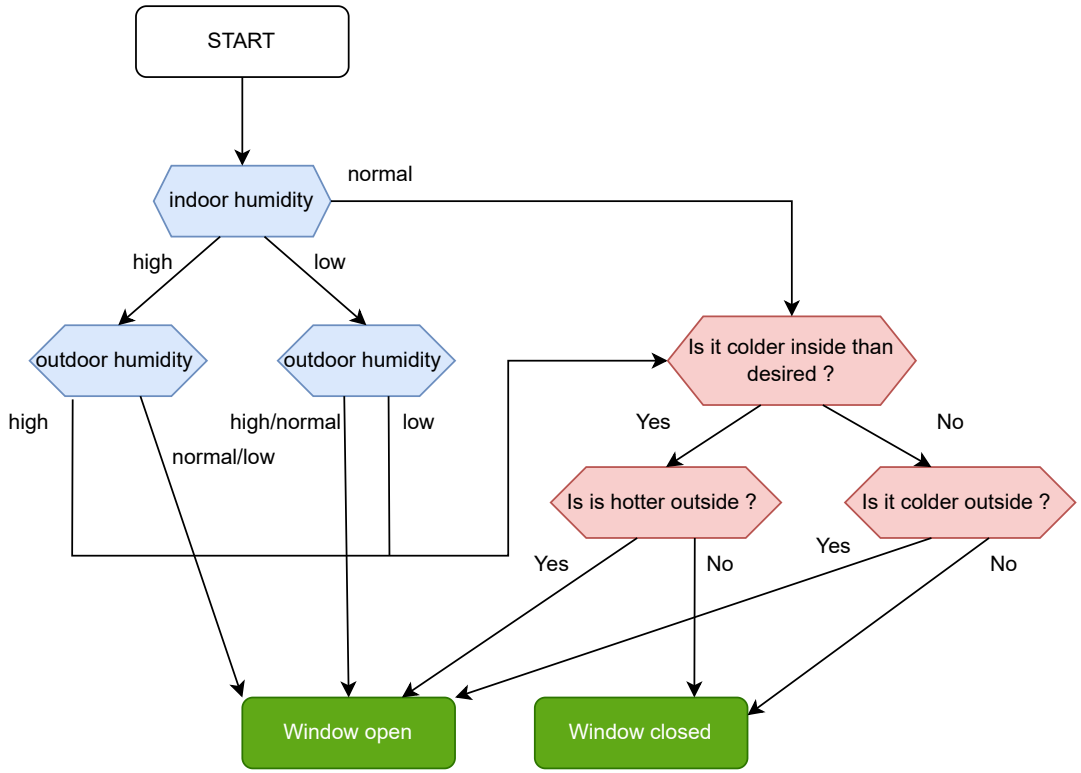


Figure 12: Decision-making flowchart for the window control

5.2.7 Servo

This task drives the servomotor given a desired position in degrees. It actualizes the PWM voltage fed to the servo. To avoid useless computation, waste of processor time and a bad behavior of the servomotor, the supposed new desired position is compared to the current desired position and the function only carries on if the desired position has changed.

Task SERVO(targetPosition)
  if $targetPosition \neq targetPosition_{current}$ **then**
    Output this PWM
  end if
end Task

5.3 Schedulability

In order to perform a schedulability analysis, the execution time of the tasks explained above have been measured. The process used to measure short execution time was the following :

- Set a test point output to 5V at the beginning of the task
- Clear this test point back to 0V at the end of the task
- Using the oscilloscope, measure the maximum high time of the signal over a few seconds

of execution.

Period	Task	Maximum Execution Time	Subtotal Execution Time
1s (1Hz)	Decision Servo Sensors	50 µs $\approx 1\,\mu s \leq \mathcal{E}$ 19 µs	70 µs
20ms (50Hz)	PID Buttons	38 µs 26 µs	64 µs
290 µs (3400Hz)	I^2C Sensor TH I^2C Screen	80 µs 4 µs	84 µs

Table 3: Task Execution Time Results

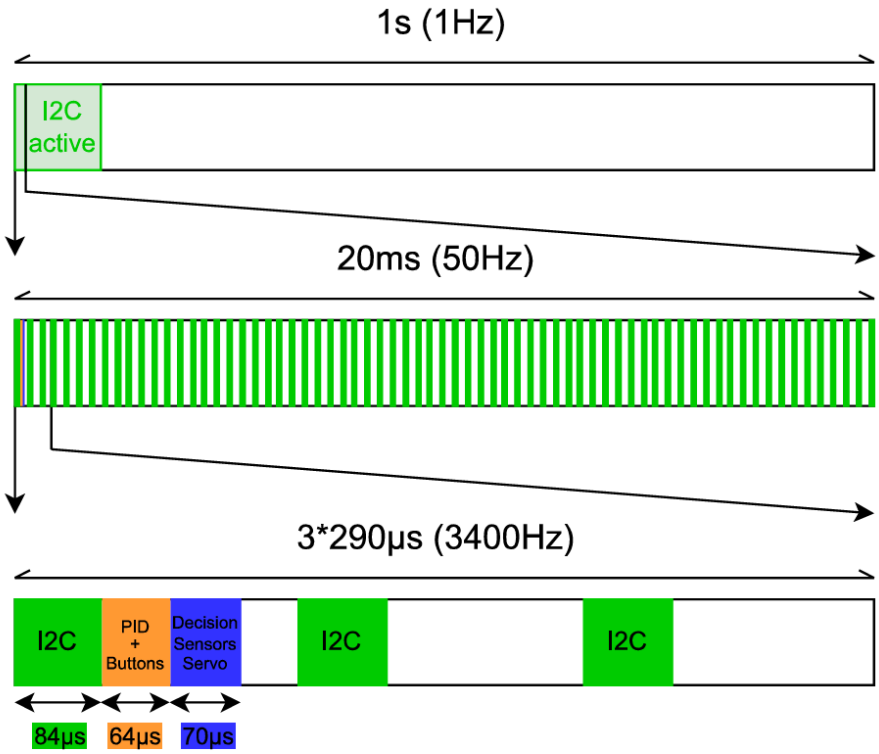


Figure 13: Visual analysis of the schedulability

The visual task execution order and schedulability analysis (Figure13) is based on the results shown in Table 3. This represents the worst possible scenario where all the tasks are initially called at the same time. Note that the I^2C is performed at 3.4kHz only during communications, which last for a maximum of 1000ms and happen every second.

Unfortunately, this analysis alone cannot prove that the tasks are schedulable. Indeed, a lot of processes are initialized once and constantly running in background. Among others one can cite the QEI, PWM, ADC and Timers modules. However, since the analysis reveals that the processor still has plenty of idle time and since no error appears during testing, it seems reasonable to suppose that the whole software is schedulable

If the background modules execution time are supposed negligible, one can compute the processor load factor to be :

$$U = \frac{70\mu s}{1s} + \frac{64\mu s}{20ms} + \frac{84\mu s}{290\mu s} * 10\% = 3,2\%$$

5.4 Processor utilization

As mentioned earlier, this project is build on a PIC18F4331 micro-controller. The final version of the software uses 38% of the available Program space and 11% of the Data space. These utilization could for sure be optimized but will not since the project is not intended to grow anymore.

6 DC motor control system

6.1 Description of the control

To include a control system in this project, it has been chosen to implement a controller for the blind system. The blind is driven by a DC motor and the position is measured using a coding wheel and the quadrature encoder interface (QEI) of the microcontroller. The scope of the control system is to control the blind position with respect to external disturbances. The design of this control was carried out in several steps that are presented in the following subsections.

6.2 System analysis

The system to control is represented in Figure14. The reference provided to the system is the desired position of the blind. The output of the system is the duty cycle provided to the H-Bridge of the DC motor. Modeling of the system behavior can be done using electromagnetic

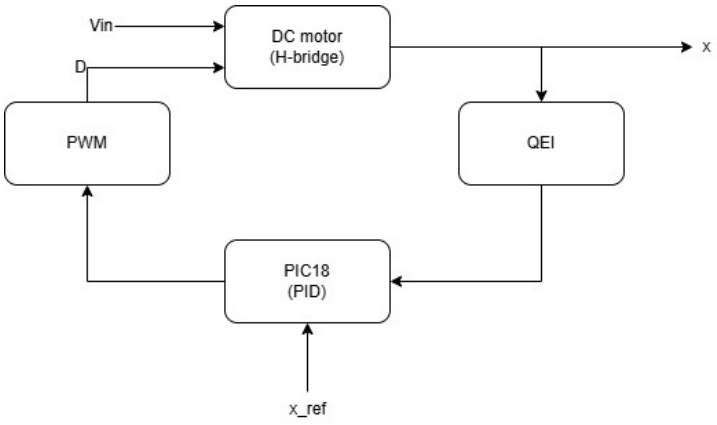


Figure 14: Closed loop system of the DC motor

laws of DC motors. The DC motor used in this project is a DC motor with a permanent magnet rotor. The excitation current of the rotor can therefore not be modified. Various equations must be combined in order to obtain a transfer function of the system. The procedure followed is based on MATLAB control tutorials.

6.2.1 Laws governing DC motor

- Kirchhoff applied to DC motor :

$$V(t) = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + e(t) \tag{1}$$

  where $V(t)$ is the applied voltage, R_a is the resistance of the armature, L_a is the inductance of the armature and $e(t)$ is the electromotive force created in the motor.

- The electromotive force is linked to the angular velocity of the rotor by the relation

$$E = K_{e1}\omega\phi(I_e) = K_e\omega = K_e\frac{d\theta}{dt}. \tag{2}$$

  The relation is proportional due to the constant excitation current in the rotor.

- The torque produced by the motor is linked to the armature current by the relation

$$\tau = K_{t1} I_a \phi(I_e) = K_t I_a. \tag{3}$$

- The torque is related to the angular velocity and acceleration using the relation

$$\tau(t) = J\frac{d^2\theta}{dt^2} + b\frac{d\theta}{dt}, \tag{4}$$

where J is the inertia momentum of the rotor and b is the friction coefficient of the rotor.

Neglecting the losses, the coefficients K_e and K_t can be considered equal in SI units. The electrical power provided to the stator can be equalized to the mechanical power at the rotor. In the following developments, these coefficients will be called K.

6.2.2 Transfer function in the Laplace domain

In the Laplace domain, these equations can be written

$$V(s) - E(s) = I_a(s)[R_a + sL_a], \tag{5}$$
$$E(s) = Ks\theta(s), \tag{6}$$
$$C(s) = KI_a(s) \text{ and} \tag{7}$$
$$C(s) = s^2J\theta + bs\theta. \tag{8}$$

Combining these equations gives a transfer function of the system

$$\theta(s) = V(s)\frac{K}{s(sJ+b)(sL_a+R_a)+K^2s} \tag{9}$$

In the practical configuration used in this project, the controller should output the duty cycle of the PWM that controls the DC motor. $V(s)$ can therefore be expressed as $V(s) = DV_{in}$ where V_{in} is the DC input voltage of the DC motor, that is, $V_{in} = 9\text{V}$.

$$\theta(s) = \frac{K}{s(sJ+b)(sL_a+R_a)+K^2s}DV_{in} \tag{10}$$

The control of the system is done on the blind position $x(t)$. The blind position can be linked to the angular DC position by the relation $x(t) = r\theta(t)$ where r is the radius of the gear that drives the blind axis.

The overall transfer function of the control system is therefore,

$$x(s) = \frac{rK}{s(sJ+b)(sL_a+R_a)+K^2s}DV_{in} \tag{11}$$

6.2.3 Determination of the physical parameters

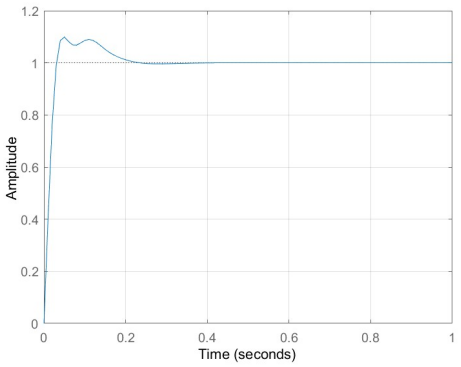
- In a first-order approximation, friction of the motor can be neglected. b is therefore considered null.
- J is the inertia momentum of the rotor. This value is not provided in the datasheet of the DC motor. An approximation based on the formula of the inertia momentum of a DC motor ($J = \frac{1}{2}mr^2$, with m and r the mass and the radius of the rotor) and the dimension of the rotor give a value of $1.10^{-8}[\text{kg.m}^2]$ that will be considered in the following sections.
- K is the torque coefficient of the DC motor. This value is also unknown and should be approximated. A typical value considered in the following sections will be 0.0589, given that $K_e = K_t = K = \frac{V - R_aI_a}{\omega}$ with $V = 6\text{V}$, $I_a = 0.3\text{A}$ and $\omega = 73.3\text{r/s}$ considering a gear ratio of 10 at the motor output.
- R_a is the resistance of the armature. This resistance was measured using an ohmmeter. A value of $R_a = 5.6\Omega$ has been found.

- L_a is the inductance of the armature. Knowing R_a, the value of the inductance has been evaluated by measuring the time constant of an RL circuit. A value of $L_a = 52.8\text{mH}$ has been measured.

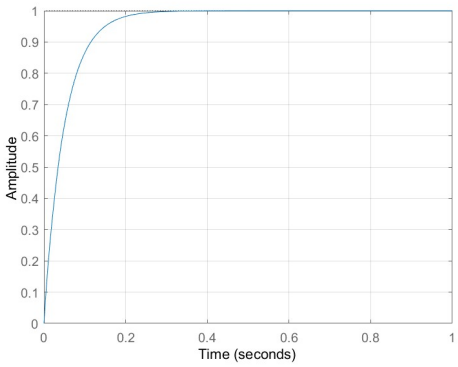
6.3 Controller design

The first idea was to design a PID controller. The design was done on MATLAB Simulink. Its input will be the error between the blind position, computed with the QEI, and the target position. It should reach the target position in a reasonable period of time. In practice, the DC motor has a limited speed, so the time needed to reach the target is expected to be greater than in simulation.

With the idea of designing a PID controller, the method used to determine K_p, K_i and K_d was the *Ziegler-Nichols* method. This method consists of setting all gains to zero and then increasing K_p until oscillations appear. The other coefficients are determined based on this value of K_p. It appeared that the system response was already very good with only a proportional controller. Results with PID controller were less satisfactory than the results with only a proportional controller, as seen in Figure 15a and 15b.



(a) $K_p = 90, K_i = 1500, K_d = 1.35$



(b) $K_p = 25$

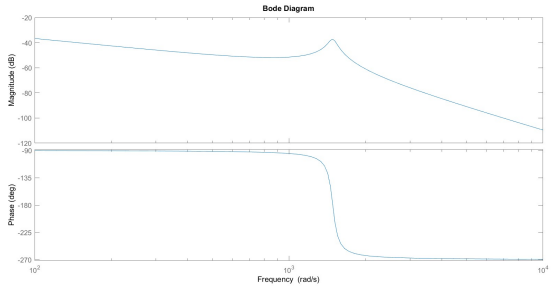
Figure 15: Comparison of the unit step response with the PID controller tuned using *Ziegler-Nichols* method (15a) and a proportional controller (15b).

Therefore, a proportional controller with $K_p = 25$ has been chosen for the controller design. The Bode plots of the compensated and uncompensated system are shown in Figures 16a and 16b.

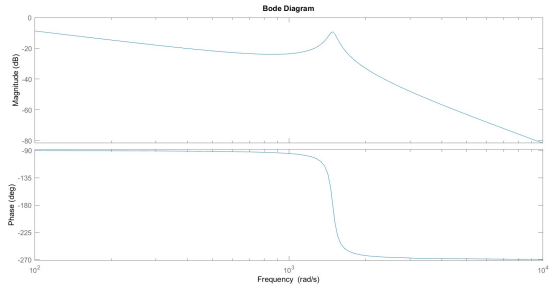
The switching frequency considered in this project will be of 50Hz. Analog control design loop theory says that the cross-over frequency should be around 1/10 of the switching frequency. The uncompensated bode plot shows that at the cross-over frequency (around 5Hz), the phase margin is equal to 90°. The controller that should be used can therefore be an proportional controller. This theoretical result support the simulation that have been carried out.

6.4 Implementation of the control system

As it has been previously analyzed, the control is done with a proportional controller. The higher the error, the higher the motor reaction desired. The controlled parameter is the duty cycle of the PWM module that is provided to the H-Bridge. The direction of the motor has also to be controlled by setting or clearing the digital output pin connected to the direction of the H-Bridge.



(a) Uncompensated



(b) Compensated ($K_p = 25$)

Figure 16: Comparison of the system bode plot uncompensated (16a) and compensated with proportional controller of $K_p = 25$ (16b)

This update procedure is done at a rate of 50Hz. This frequency has been chosen to fit a trade-off between rapid reaction for smooth trajectories and to not be influenced by the measurement noise.

Experiments have been conducted to understand how the motor responds to a given duty cycle. It showed that the motor could only be activated with a 7-bit duty cycle. A duty cycle of `0b00000000` and `0b01111111` gives the motor the highest and lowest possible power, respectively. Since the reaction of the motor should be the highest when the error is the highest, the result of $k_p \times error$ is complemented.

Some choices had to be made because of the limited resolution of the duty cycle. In fact, the actual and the target positions are encoded on 8-bits. The error, which is the difference between the target and the actual position, is encoded on 8-bits as well. Since k_p has a value of 25 (`0x19`), the controller output is encoded on 13-bits ($k_p \times error$).

The first idea should be to consider only the 7 LSB of the controller output and if it is greater than `0x80` to provide the maximum power to the DC motor. This solution leads to a very accurate control system in terms of position. The main drawbacks were oscillations around the target position, since the reaction for small error was too high and a limited distance of control in terms of velocity. A change of one bit in the encoded position is less than 1mm of the blind position. The motor speed was therefore the highest when the actual position was about 2cm from the target. This small range of control was not suitable for this application, as the location of the blind does not need to be monitored with a precision of millimeters.

An other way to consider this limited duty cycle resolution is to remove a given number (4) of the least significant bits of $k_p \times error$. It is evident that this solution leads to a static error of the blind position since the tiny error will not be overcome. This static error is considered reasonable in this application. In fact, the idea behind the blind control is to obtain an automatic blind that

returns to its position after disturbances. The user does not require a millimeter accuracy. The advantages of this solution is to get rid off oscillations without considering the implementation of an integrative error term and to get a greater range of velocity control. The trajectory to reach the target location is, therefore smoother than with the first solution.

This static error is also the consequence of the finite number of gear teeth. All the positions are not accessible due to the discretizations of the mechanical gears. The static error of this proportional controller has been estimated to be less than 5mm. For this application, it has been considered to be satisfying. If one wants to remove this static error, adding an integrative term would solve this issue. Since this project, does not require millimeter control precision, this solution has not been implemented.

The final implementation follows this procedure.

Task UPDATE PWM DC MOTOR
  $error \leftarrow position_{target} - position_{measurement}$
  $sign_{error} \leftarrow \mathbf{1}(error > 0)$
  $error \leftarrow abs(error)$
  if $error = 0$ **then**
    $Motor \leftarrow deactivate$
    return
  end if
  $Motor \leftarrow activate$
  $pid_{out} \leftarrow k_p \times error;$
  $dutyCycle \leftarrow \overline{pid_{out} < 10 : 4 >}$
  if $pid_{out} < 15 : 11 > \geq$ `0b00001` **then**
    $dutyCycle \leftarrow$ `0x00`
  end if
  if $sign_{error} = 1$ **then**
    $Direction \leftarrow Down$
  else
    $Direction \leftarrow Up$
  end if
end Task

---

7 Improvements

Since this project has been carried during the whole academic year, the choices made at the initiation of this project had an important impact during the whole process. From this first complete project carried out in the engineering curriculum, one can reflect on advice and decisions that could have been made differently. Below is a list of topics that would be approached differently if the project were to be started again.

- This project choice did not implement analog or power electronics blocks which would have been nice to develop practical skills on these electronics topics.
- Components were chosen without full knowledge of all the problems that they would cause. Notably, the microcontroller was chosen because of its quadrature encoder interface. During the elaboration process, it has appeared that it does not provide any I^2C master mode module. The implementation of the whole I^2C protocol to be able to interact with the screen and with the sensors was the main difficulty of this project. This difficulty might have been avoided by a better reading of the PIC18F documentation.
- The choice of using a servomotor to tilt the window didn't take into account the inertia of the latter and the speed at which a servomotor tries to reach its target position, which results in oscillations in this project. A better choice would have been to use a simple or stepper DC motor with a reducing gearbox.

A Software

The complete embedded software in assembly can be found in this GitHub repository.

B Videos

Some videos showing the system in different mode of operations are provided.

Video 1 - Automatic Window Opening (Humidity)

This video demonstrates the automatic opening of the window in response to increased indoor relative humidity.

Video 2 - Blind Control System

This video illustrates the blind control system in action. When external disturbances attempt to shift the blind from its set position, the control system actively counteracts the disturbance to maintain its target.

Video 3 - Control Panel in Automatic Mode

This video showcases the system operating in automatic mode. The user sets a desired indoor temperature, which is displayed on the LCD. The system automatically adjusts conditions to maintain that temperature.

Video 4 - Automatic Window Opening (Temperature)

This video demonstrates the automatic opening of the window due to elevated indoor temperature. When the measured temperature exceeds the user-defined threshold, the window is automatically opened to regulate the environment.

Video 5 - Automatic Blind Closing (Luminosity)

This video shows the automatic closing of the blind in response to high outdoor luminosity. The system reacts by closing the blind to reduce incoming light intensity.

Video 6 - Manual Control Mode

This video displays the manual control mode for both the window and the blind. In this mode, the user can operate both devices independently of environmental conditions.