# Design of a 2-axis Solar Tracker: Sunflower System

## Design of Microcontroller-Based Systems

2020-21 Q2
May 24th, 2021


Victor Expósito Griñán

# Table of Contents

# 1 - Introduction

Nature knows best. When we look around we can see plenty of efficient natural processes made by animals and plants through years and years of evolution. Had not we observed the world around us, probably most of our human inventions would not have come true.

These processes can also help us to improve our existing mechanisms, especially when talking about obtaining energy. In the following piece of work I am going to develop the prototype design of a microcontroller-based system that emulates the natural behaviour of young sunflowers, which is to track the position of the Sun in order to maximize the energy obtained by a domestic solar panel.

# 2 - Characteristics of the System

## 2.1 - Functionalities

The first part of the design consists of describing the different tasks that the system will have to do to be effective. To begin with, it will have 2 different operating modes:

- STANDBY. The system is powered but deactivated, so it does not do anything despite the light source's position.
- CONTROL. The system is powered and activated. In this mode, the system will find the light source's position and will face the solar panel towards it.

To achieve the correct and optimal positioning, the system will dispose of two servo motors (horizontal and vertical axis) that will face different LDRs to the light in order to determine where the light source comes from. To control the status, it will have an LED that will be lighted on indefinitely when the system is in STANDBY mode, and will light intermittently when in CONTROL mode. The user will have a push button to change between modes. To do so, the button will have to be pressed for more than 2 seconds.

Also, the system will implement a remote control using an Android mobile device via Bluetooth. Through the phone, the user will be able to activate or deactivate the system and will be capable of consulting values such as the servo motors position (using an accelerometer), the LDRs voltage levels or the system status (STANDBY or CONTROL).

## 2.2 - Structural Diagram

Once the functionalities are clear, it is necessary to determine the subsystems that will make up the final system and how they need to be interconnected, with the goal of exchanging information. For this project I have used the structure shown in Figure 1, with the following subsystems:

- Power Supply Subsystem. It is constituted by a 5V voltage input, which will provide energy to the rest of subsystems.
- Input Subsystem. In charge of capturing the external signals, either they are produced by the environment (sunlight, position) or the user (push buttons, remote commands).
- Processing Subsystem. This subsystem will have to get the data obtained in the previously mentioned input subsystem and subsequently execute the algorithms that will generate the directives to control the servo motors.
- Output Subsystem. In this subsystem the data produced by the processing subsystem will move the servo motors to the desired position, as well as inform the user of the status of the system via the LED. This subsystem will also send to the Android mobile device the information it wants to consult.
- Communication Subsystem *. This subsystem's work will be to monitorize the communication with the accelerometer (it will probably use the I2C protocol) and with the remote controller device (Android mobile) through Bluetooth.

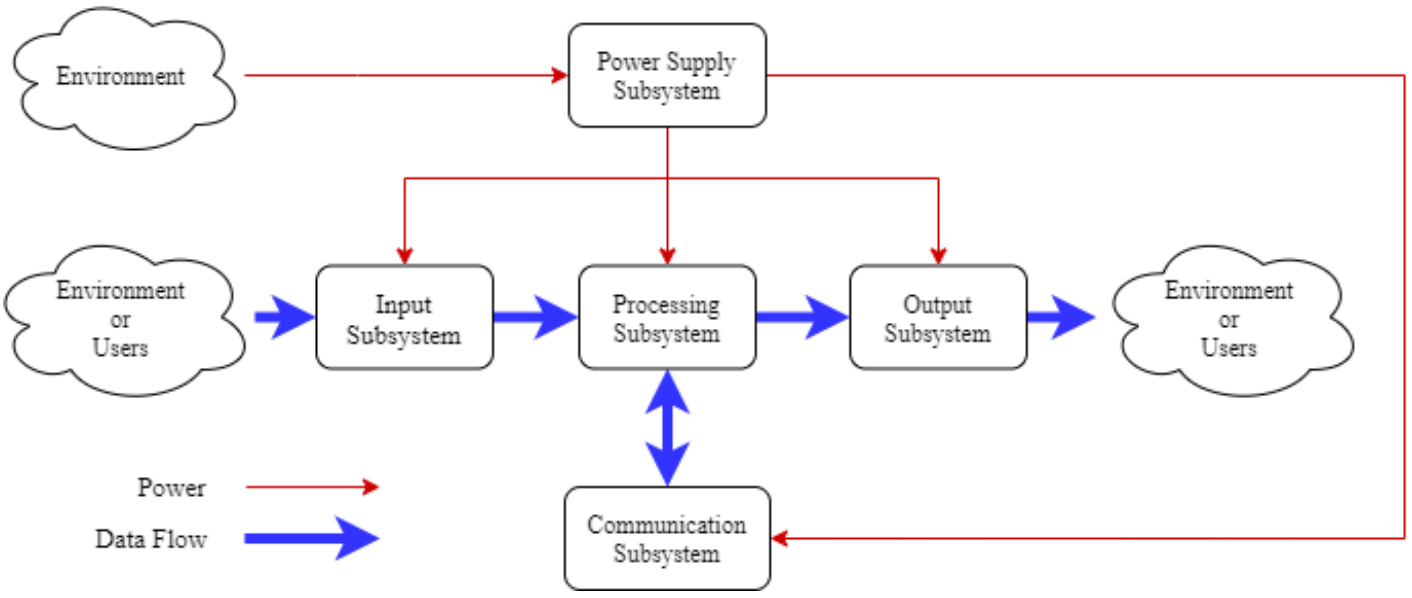


Figure 1: Structural Diagram of the System

* Actually, this subsystem also acts like the Input/Output subsystem, receiving information from the accelerometer and sending and receiving information from the remote controller.

# 3 - Hardware Architecture

## 3.1 - List of Components

After some research, I have found the necessary components to reach the complete functionality of the prototype, which are the following:

| Product | Model | Manufacturer | Prod. Ref. |
|---|---|---|---|
| Push Button | B3F-Series | Omron Electronics | B3F-1022 |
| LED | Kingbright | Kingbright | L-53SYD |
| Servo Motor | Micro Servo 9G SG-90 | Tower Pro | SRV-0016 |
| Servo Motors Bracket | Tiendatec | Tiendatec | ARD-SERVOBRACKET |
| LDR | GL-5528 | Tiendatec | GL5528 |
| Accelerometer | MPU-6050 | Sunfounder | RB-Suf-16 |
| Bluetooth Module | HC-05 | Bricogeek | BLT-0008 |
| Microcontroller | Mega 2560 Rev 3 | Arduino | A000067 |
| 220 Ω Resistor | TE Connectivity | TE Connectivity | CFR100J220R |
| 1 kΩ Resistor | Stackpole Electronics | Stackpole Electronics | CF14JT1K00 |
| 2 kΩ Resistor | Yageo | Yageo | CFR-12JB-52-2K |
| 10 kΩ Resistor | TE Connectivity | TE Connectivity | CFR100J10K |
| Wire Jumpers | MikroElektronika | MikroElektronika | MIKROE-512 |
| Protoboard | Sunhayato | Sunhayato | SAD-101 |

In addition to that list, and to make the remote control possible, an Android mobile device is required. For the testing section (see Chapter 5) a Bluetooth communication App will be needed. It will be used on my personal phone, which is a OnePlus 7 with Oxygen OS, an Android based Operating System.

In the table below, the total cost of the prototype creation can be found. It might be a little expensive, but it has to be considered that half of the total price is dedicated only to the processing board, and that there are some components, like the accelerometer, that are relatively expensive but necessary.

| Product | Quantity | Price per unit | Total Price ** |
|---|---|---|---|
| Push Button | 1 | 0,28 € | 0,28 € |
| LED | 5 * | 0,361 € | 1,805 € |
| Servo Motor | 2 | 2,96 € | 5,92 € |
| Servo Motors Bracket | 1 | 2,95 € | 2,95 € |
| LDR | 4 | 0,25 € | 1 € |
| Accelerometer | 1 | 8,19 € | 8,19 € |
| Bluetooth Module | 1 | 5,45 € | 5,45 € |
| Microcontroller | 1 | 41,26 € | 41,26 € |
| 220 Ω Resistor | 10 * | 0,119 € | 1,19 € |
| 1 kΩ Resistor | 1 | 0,08 € | 0,08 € |
| 2 kΩ Resistor | 1 | 0,1 € | 0,1 € |
| 10 kΩ Resistor | 10 * | 0,174 € | 1,74 € |
| Wire Jumpers 10-Piece Pack | 2 | 3,34 € | 6,68 € |
| Protoboard | 1 | 4,71 € | 4,71 € |
| **Total Price of the System** | | | **81,36 €** |

* Minimum number of units that can be purchased, the prototype will need less
** All prices include taxes

## 3.2 - Architecture Diagram

Before starting to build the circuit I wanted to make a conceptual model of how the prototype will look like in terms of hardware architecture. The diagram can be seen in Figures 2 and 3.

The idea behind the whole system is to divide it into different structural and functional blocks, which are 4: Status Control System, Servo Motors System, Luminosity Control System and Wireless Communication System.

The first block belongs to the local status control, basically, the push button and the LED. In the second block the servo motors bracket can be found, along with the 2 servo motors that will move in the horizontal and vertical axis. This block will move the structure that forms the third block, which is a plate that contains 4 LDRs and an accelerometer. This structure will be explained in detail in Chapter 4.2. Finally, the fourth block will consist of the HC-05 Bluetooth communication module.

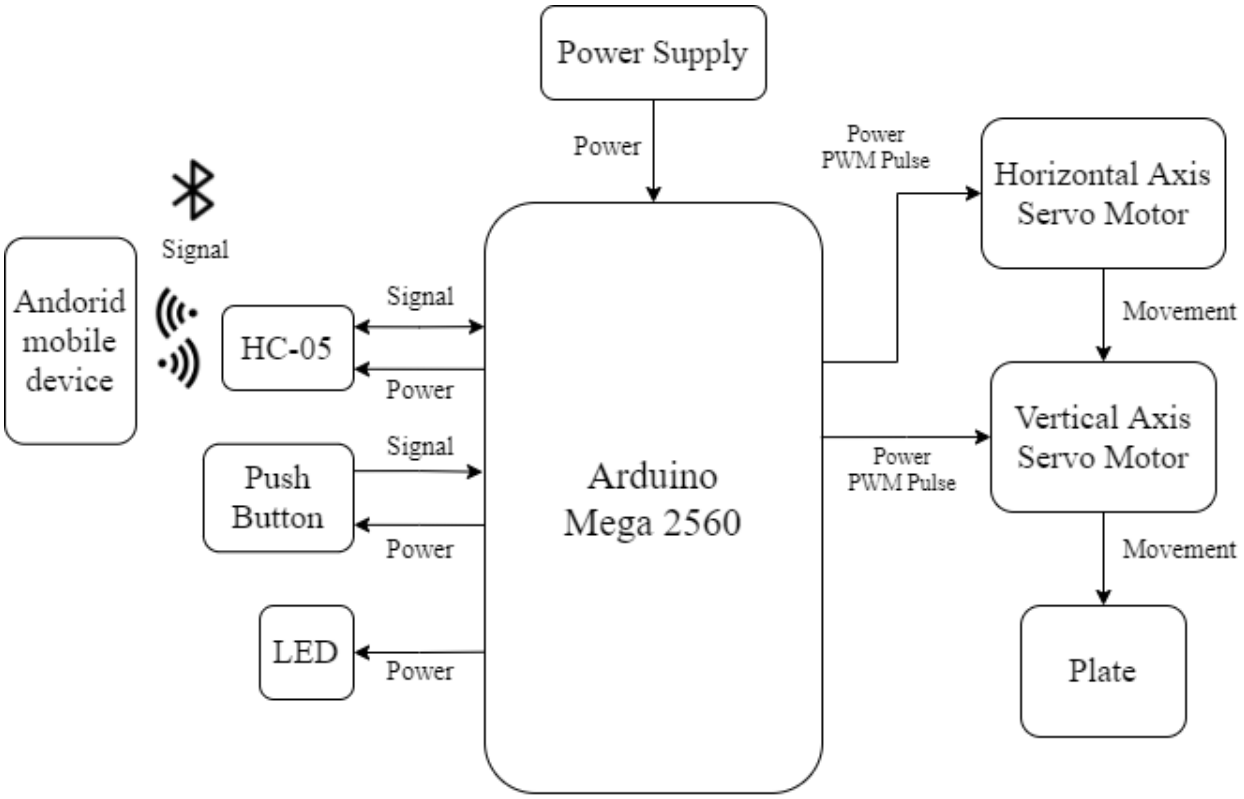


Figure 2: Architecture Diagram of the System

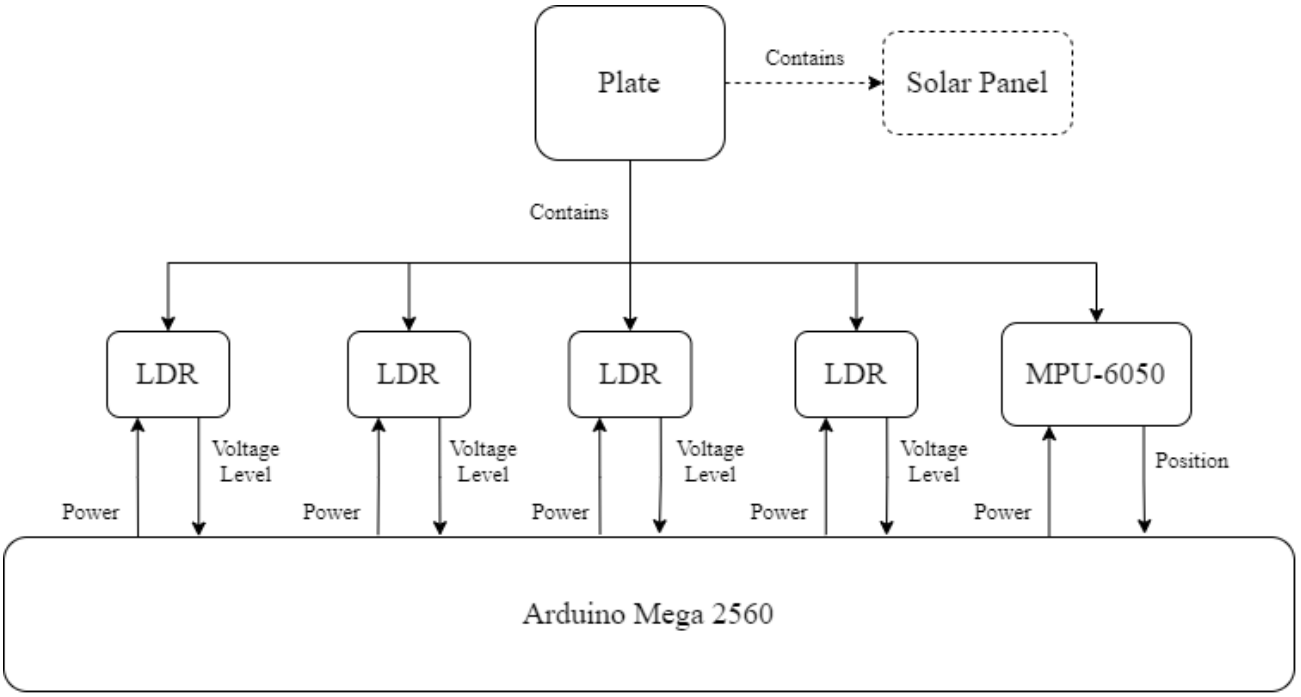


Figure 3: Architecture Diagram of the System (Plate Zoom-In)

## 3.3 - Electronic Circuits

At this point, I can model the electronics of the project having in mind the particular characteristics of the chosen components. In the following subchapters, the four functional blocks mentioned in last Chapter are going to be described (block 2 is separated in servo motor mechanism and position control).

### 3.3.1 - Status Control: Activation and Deactivation

In this section, there are only two components, the push button and the LED. The button will be connected to 5V with a 10 kΩ resistor, and its signal will be captured by the D2 pin of the board. The push button is of type SPST Normally Open, so the measured pulse will be 5V if the button is pressed and 0V if it is not.

The yellow LED will be powered by the D3 pin of the Arduino, and it will be connected in series with a 220 Ω resistor.

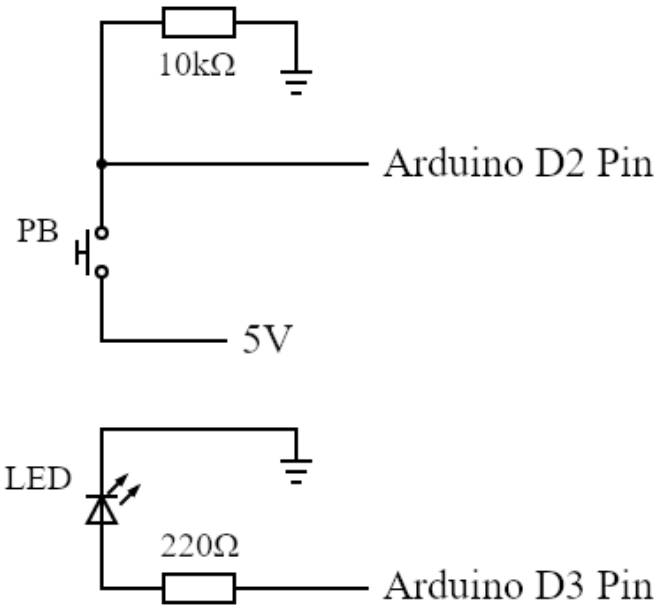


Figure 4: Status Control Circuit

### 3.3.2 - Servo Motors Mechanism

For this circuit, the 2 servo motors are required. These have 3 connections, VCC, GND and the pulse that will move the servo. Each one will be controlled by a digital pin of the board, D9 and D10. The model I have chosen needs approximately 5V of voltage (4,8V to be precise) to work, so the mechanism will look like the one shown in Figure 5. Also, the servo motor SM1 will have to be connected to SM0 to represent the 2-axis, horizontal (SM0) and vertical (SM1).

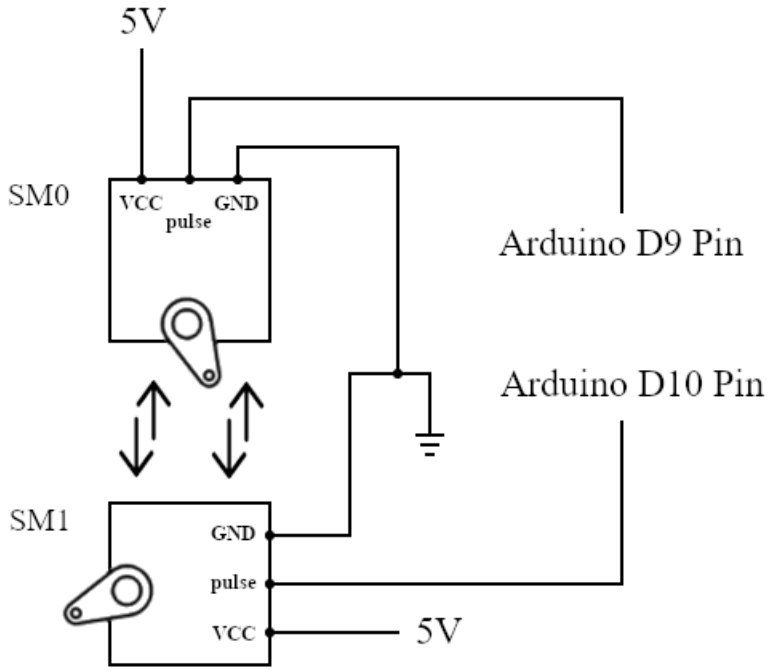


Figure 5: Servo Motors Mechanism

### 3.3.3 - Servo Motors Position Control

In order to know at any time which is the position of the servo motors, the prototype will need an accelerometer, specifically, the MPU-6050. The reason I have chosen this is because I have used it before and it gives really good results. For the device to work, the connections are very simple (see Figure 6), it needs a power input that can be 3,3V or 5V (which is the level used) and the SCL and SDA pins will be connected to the homonymous pins in the Arduino.

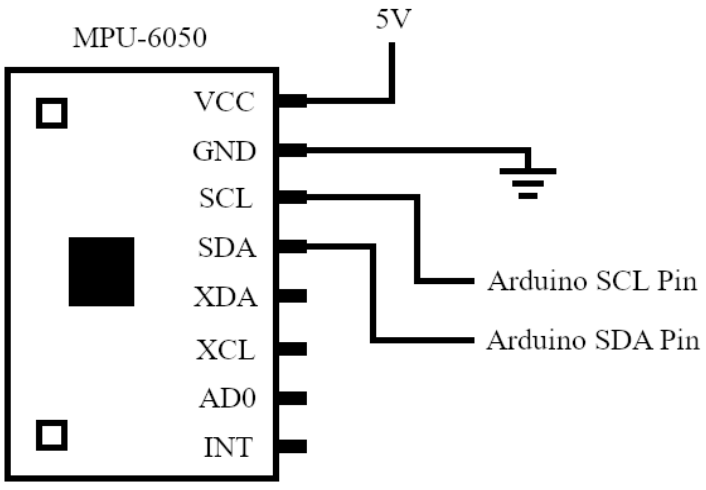


Figure 6: Servo Motors Position Control Circuit

### 3.3.4 - Luminosity Control

This is probably the most important part of the system. Here is where the decision of moving the servo motors will be taken and where the direction of that movement will be determined. This circuit will be formed by 4 LDRs through which a 5V voltage will be applied. The LDRs will need as well a 10 kΩ resistor (specified by the manufacturer). With those components forming a voltage divider, the Arduino will be able to read the voltage levels that the LDR permits, depending on the light it is exposed to. These readings will be done by the A0-A3 analog pins of the board.

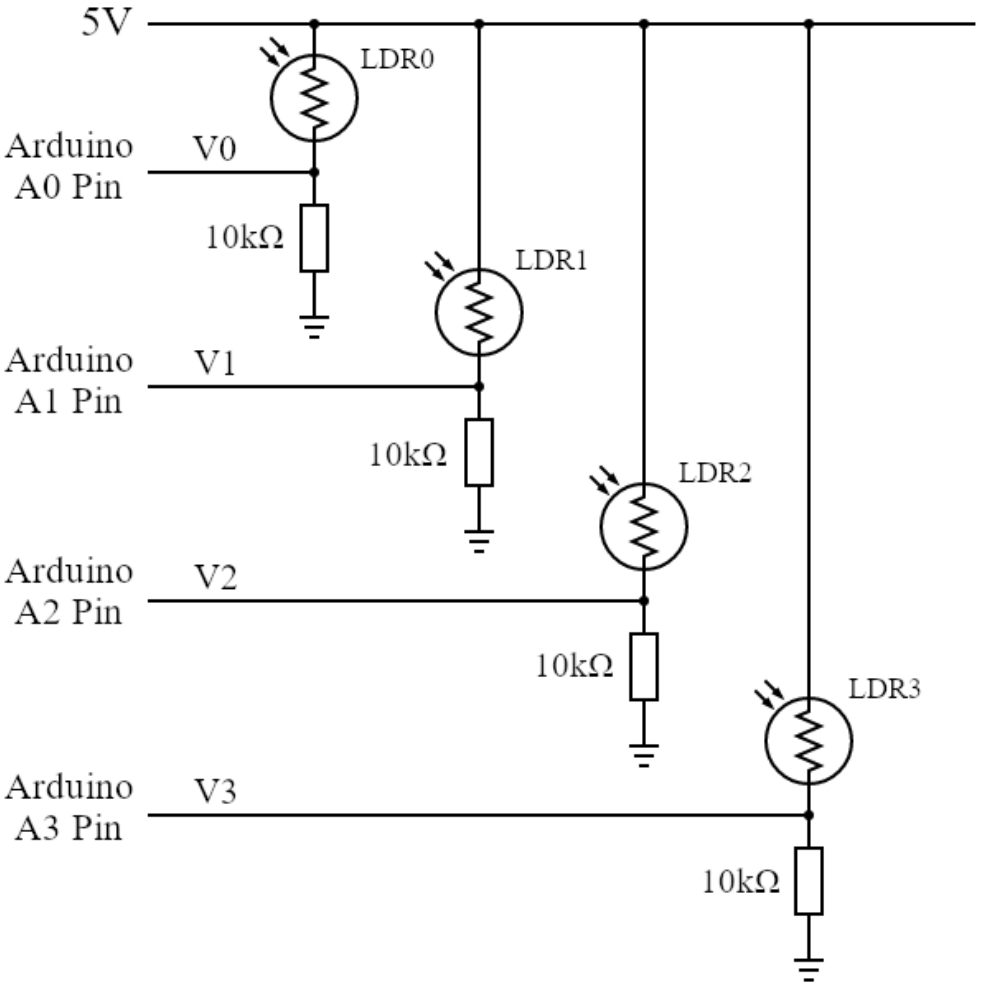


Figure 7: Luminosity Control Circuit

(Note: the configuration of this circuit makes that the higher the luminosity, the higher the voltage in V0-V3. If the resistors were switched in position, the result would be the opposite, the higher the luminosity, the lower the voltage output).

### 3.3.5 - Wireless Communication for Remote Control

Lastly, the circuit that will provide the system with remote control is the one in Figure 8. Consists of the HC-05 Bluetooth module connected to the Transmission and Reception pins (Tx0 and Rx0 respectively) of the Arduino board. Collecting information about this module I found that it can be powered by a voltage in the range between 3,3V - 6V, and that the TxD and RxD pins in the module use a 3,3V voltage level, so connecting it directly to a 5V powered board (like the Arduino) could damage the device. To solve this problem a voltage divider with a 1 kΩ and a 2 kΩ resistor in the RxD pin will be used in order to reduce the voltage to 3,3V.

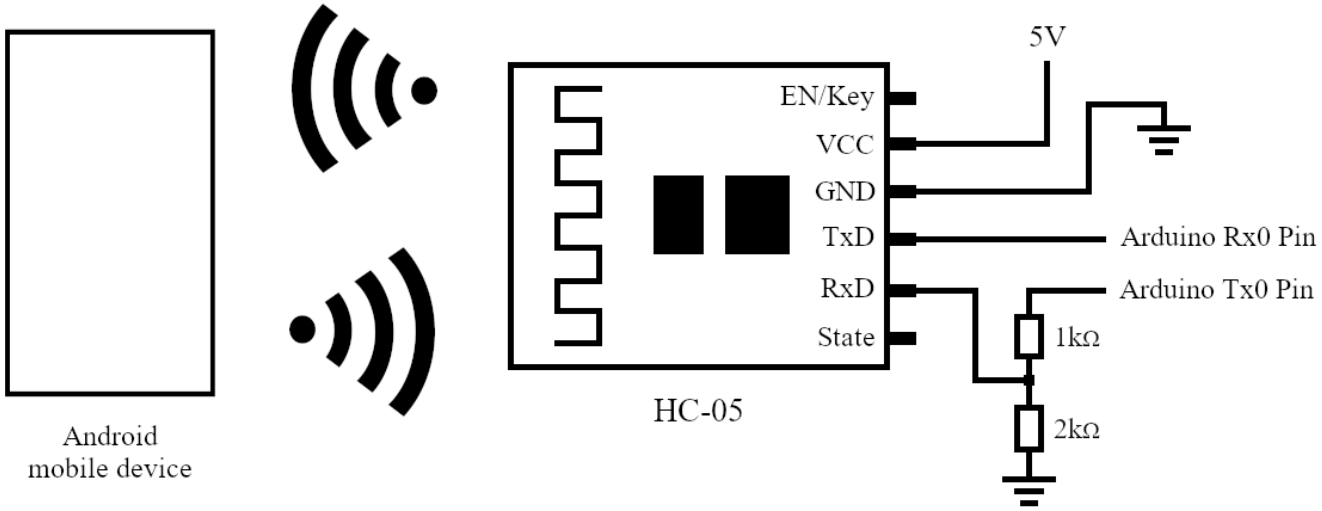


Figure 8: Wireless Communication Circuit

## 4 - Software Architecture

Once the physical structure of the prototype is clear, now it is time to develop the necessary software to create a fully functional system. This section will explain the behaviour of the system and the algorithms that will be implemented in the different blocks.

### 4.1 - Status Control: Activation and Deactivation

In this part, the control of the status of the system will follow the flux diagram shown in Figure number 9. Basically, I will need a system variable that stores the current status and will toggle when the push button is pressed for more than 2 seconds or when a Bluetooth command is received (Calculation of the time the button is pressed can be easily implemented using the *millis()* function).

Assuming that the system is powered on:

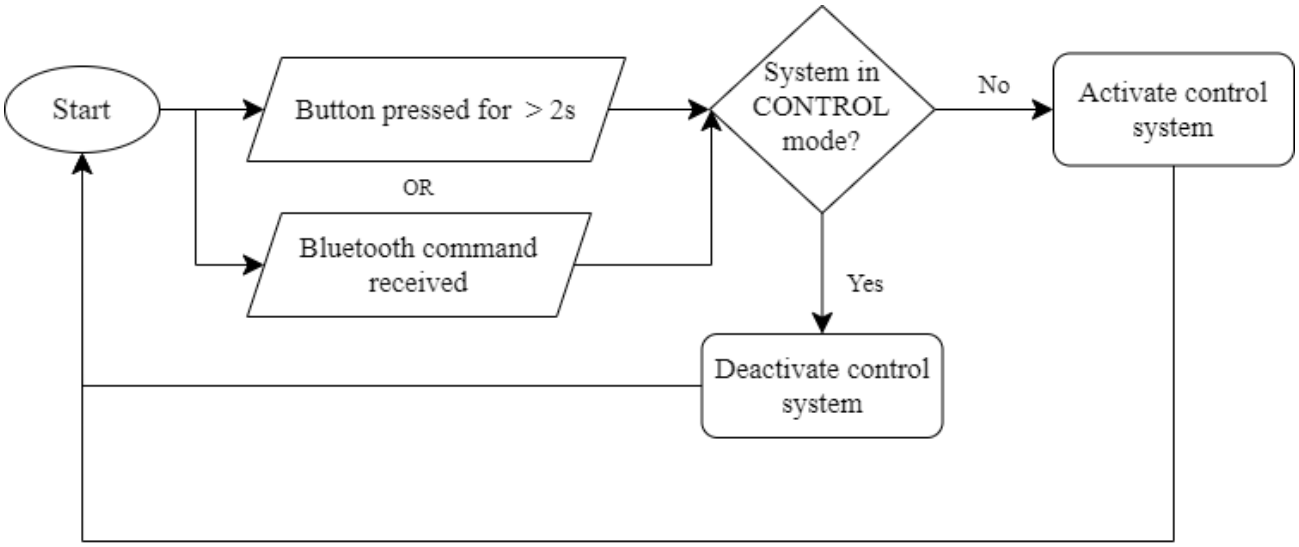


Figure 9: Activation/Deactivation Flux Diagram

Activating the control system consists in:

- Putting the LED in intermittence
- Changing status variable from STANDBY to CONTROL
- Moving servo motors to CONTROL starting position *
- Performing the luminosity control and moving servos if necessary

Deactivating the control system consists in:

- Lighting on the LED indefinitely
- Changing status variable from CONTROL to STANDBY
- Putting servo motors in initial inoperative position *

* The initial inoperative position for the servo motors will be a “looking down” position at an angle of 20 degrees below the XY plane, like shown in Figure 10. When activating, the starting position for light seeking will be a “looking above” position at an angle of 30 degrees above the XY plane (Figure 11). From then on, the mechanism starts to do its job. This is done with the goal of easing the luminosity control, as the Sun (or light source) will be located above.

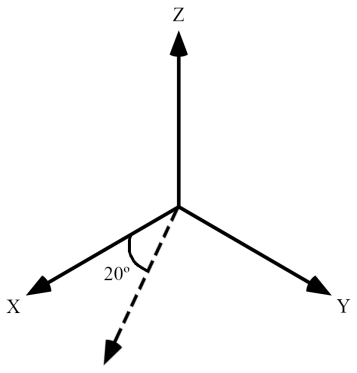


Figure 10: Initial inoperative position

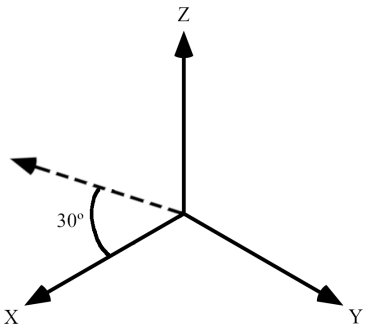


Figure 11: Starting active position

## 4.2 - Luminosity Control and Servo Motors Mechanism

To understand how I will implement this part, first I have to explain the idea behind the light detection. It consists of a plate with 4 LDRs and an accelerometer just like it can be seen in Figure 12. Those walls between the resistors will be the element that will make the difference.

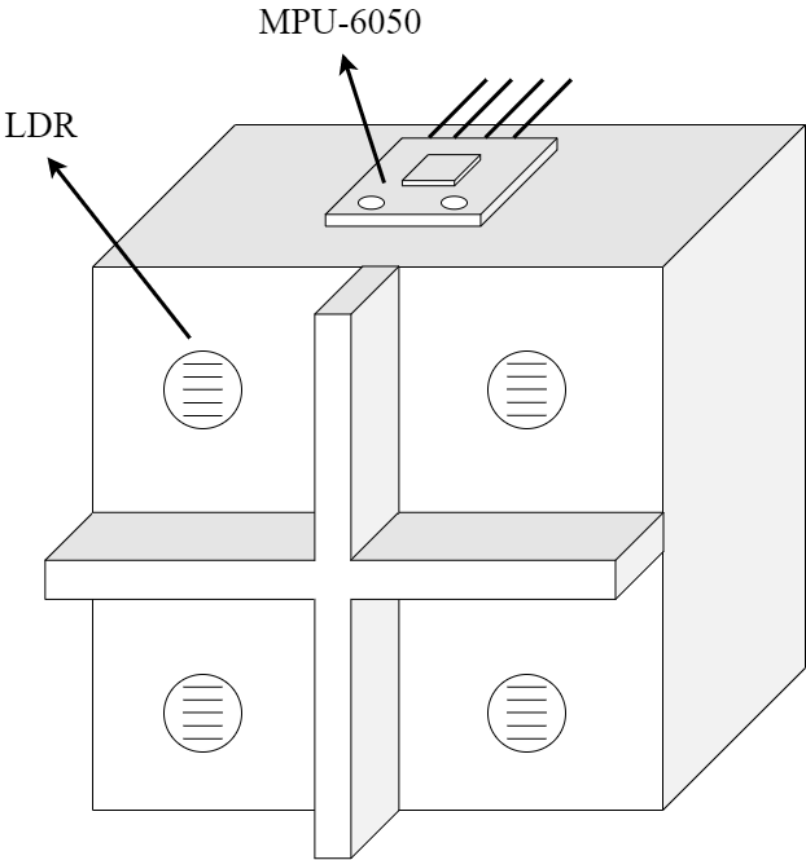


Figure 12: Plate Diagram

When the light comes from a frontal position, the walls provoke that all the LDRs get approximately the same amount of it, which will produce a similar voltage level in all of them. If all of the values are within the same range of values (to be determined in experimental tests), the servo motors will not move.

In case the LDRs in the right produce a higher voltage than their counterparts in the left, it means that the light comes from the right, and that will make the SM0 servo motor get the necessary pulse to rotate to that position. Obviously, this will also work when the light comes from the left.

This will apply to the upper and lower halves of the structure as well. If the voltage levels of the LDRs located above are higher, the same logic is used, using the SM1 servo motor to move the plate upwards. If this happens for the lower resistors, the servos will move the plate downwards.

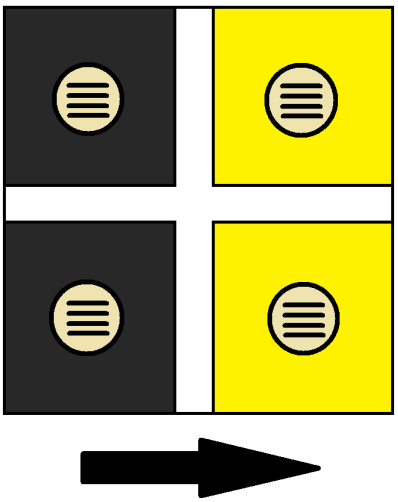

Figure 13: Plate Moving Right

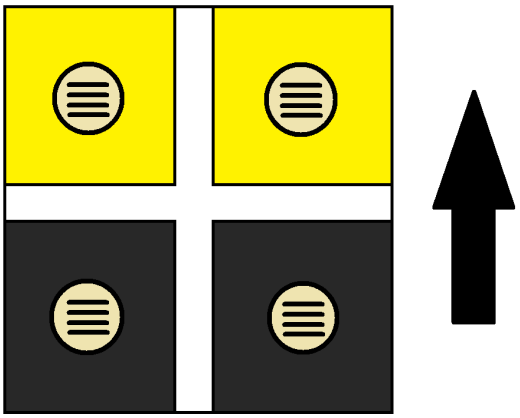

Figure 14: Plate Moving Upwards

The flux diagram of this part of the system is the following:

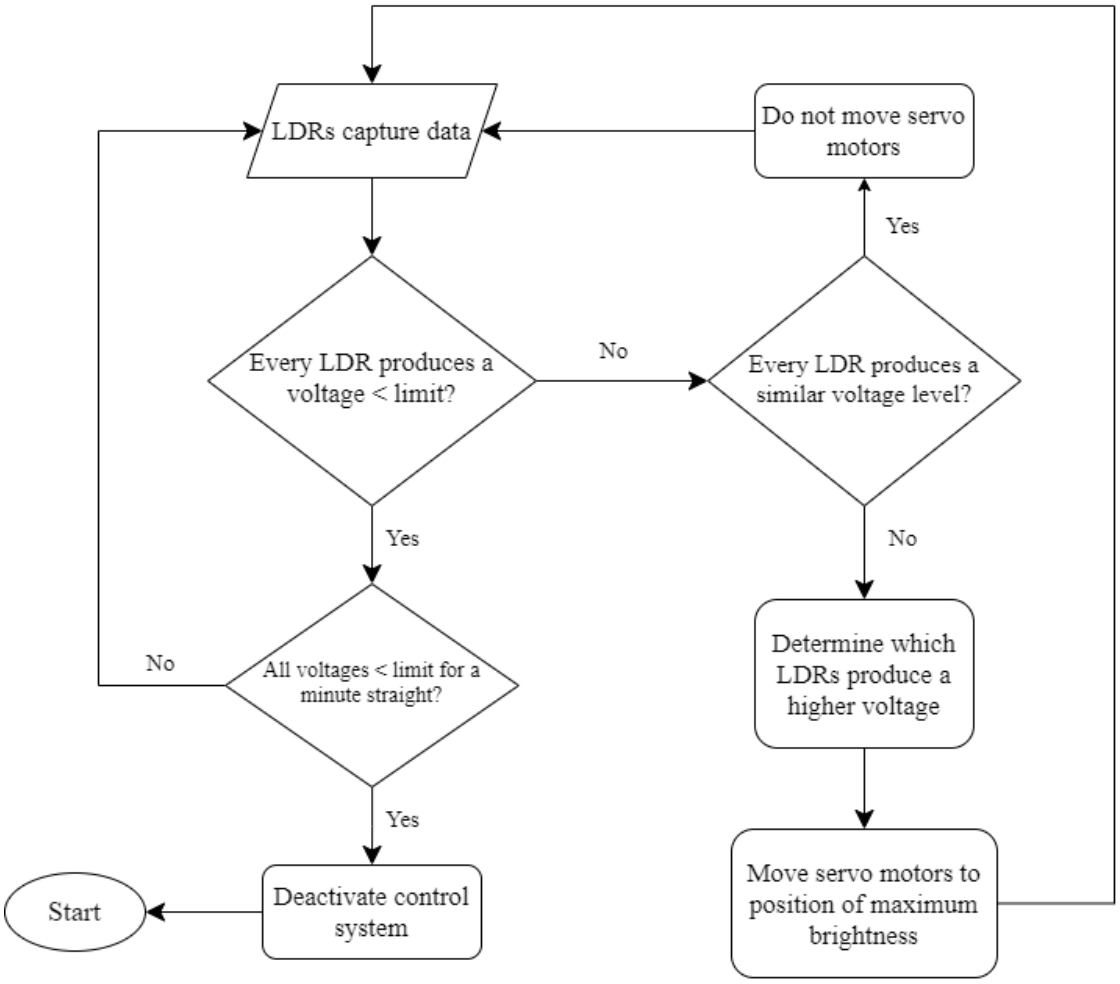


Figure 15: Luminosity Control and Servo Motors Mechanism Flux Diagram

Another detail of the system is that if all the LDRs are exposed to a light that is so dark that makes them produce a voltage level lower than a certain limit (to be determined in experimental tests), the system is deactivated automatically. This condition has to be true for a minute, which is an arbitrary value that might be enough (even though it can be modified easily if required).

The accelerometer built on top is put there to control the movement and position of the plate and, consequently, the servo motors. The device is able to determine the Pitch, Roll and Yaw angles (basics in navigation, see Figure 16), and in this system, I will only focus on 2 of them, the Pitch (angle of SM1 respect to the XY plane) and Yaw (angle of SM0 respect to the XZ plane).

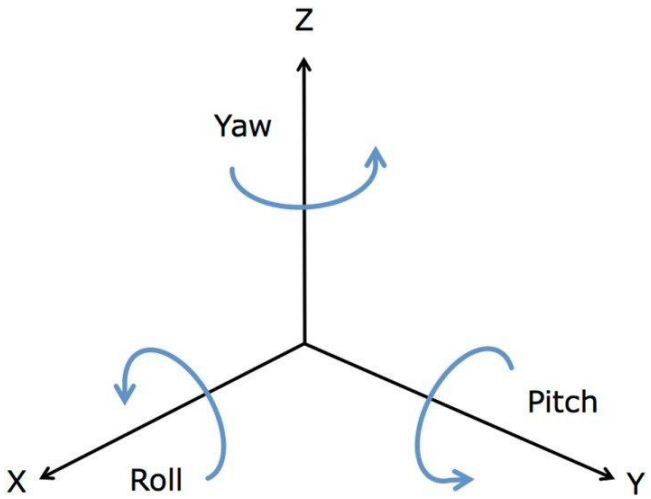


Figure 16: Navigation Angles

To make the accelerometer work, I will need an I2C library, like *I2Cdev*, and also a library called *MPU6050_light*, a very useful one when calculating those angles. It is important to mention that the Yaw angle can not be calculated solely with an accelerometer, because it uses gravitational differentials, not present when moving in that axis. This is no problem for the MPU-6050, because it also can behave like a gyroscope. Consequently, the device needs to be calibrated in order to calculate this value, but the library provides this task and it is not that much of an obstacle.

## 4.3 - Wireless Communication for Remote Control

For this part of the prototype, there are a few things to mention. Wireless communication will take place in 2 directions, from Android mobile device to Arduino and from Arduino to Android mobile device. For both ways, I will need an App on my mobile device, which will be *Bluetooth Chat*. In all cases, first I will need to pair the Android device with the HC-05 module, whose default password is 1234.

### 4.3.1 - Android mobile device to Arduino

The code responsible for capturing commands from the remote control will need to read from the serial port, so the idea is to process the data coming from there. When the user wants the system to activate, it will have to send a '1' to the module, and a '0' if it wants to deactivate it. A visual interface could be developed in order to make this process more "user-friendly", but I want to keep things simple.

### 4.3.2 - Arduino to Android mobile device

In this second type of communication, the code will send information to the user on demand. This means that the data sending will take place when the user makes a petition. The following table contains the different petition codes:

| Code sent by the user | Information provided in response |
|---|---|
| '2' | Status of the system (CONTROL/STANDBY) |
| '3' | Position of the servo motors (Pitch and Yaw angles) |
| '4' | LDRs voltage levels |

For the '2' petition, the response will be the system status variable mentioned in Chapter 4.1. For the petition made with '3' the result will be the calculation of the Pitch and Yaw angles in the Arduino code with the data provided by the MPU-6050. Finally, for the '4' petition, the response will be the voltage levels of pins A0-A3.

## 5 - Tests

The final part of this design consists of a battery of tests to know if the functionalities work properly and as expected. Before explaining the different tests I have to mention 2 important things. The first is that the Arduino Mega 2560 will be connected to a PC, which automatically powers the board with 5V. This means I will not need a power supply device of any type.

The second clarification is that for the prototype tests, a real solar panel is not really necessary in my opinion, because the important thing is that the mechanism works well. Once it does its job correctly, the panel can be installed. Anyway, I found a domestic solar panel that could be used to test how much energy the system produces in comparison to a static solar panel.

| Product | Model | Manufacturer | Prod. Ref. |
|---|---|---|---|
| Solar Panel | Voltaic Systems | Voltaic Systems | 3809 (Adafruit ID) |

| Product | Quantity | Price per unit | Total Price |
|---|---|---|---|
| Solar Panel | 1 | 21,45 € | 21,45 € |

This panel will be allocated in a foam plastic sheet alongside the plate structure (Figure 12) and glued to the servo motors bracket. Another detail is that the chosen bracket will need to have a slight modification in order to install this foam plastic sheet.

Now, the tests can take place.

- **Test 1: Activation/Deactivation Test 1:**

  This test consists of activating and deactivating the system using the push button, pressing it for more than 2 seconds. The expected response is seeing the servos move either to the starting operative position or to the initial inoperative position, depending on the mode. Also, the LED should light indefinitely (STANDBY) or intermittently (CONTROL).

- **Test 2: Activation/Deactivation Test 2:**

  This test is the same as the previous one, but instead of using the button, I will use the Android mobile device. The expected result should make no difference.

- **Test 3: Activation/Deactivation Test 3:**

  The third test consists of checking if the automatic deactivation works. The idea is to turn off the lights for more than a minute with the goal of producing a low voltage in A0-A3 (lower than the determined limit). The system should be in CONTROL mode and the expected result is to see how the system deactivates.

- **Test 4: Light Seeking:**

  It is time to check if the system seeks for light correctly. For this I will need a flashlight that will represent the sunlight. The manufacturer of the LDRs communicates that the resistors are quite slow, so they are not suitable for rapid and constant changes of light. This is no problem because in reality sunlight does not move fast, but the flashlight light should not be moved quickly. The expected result is to see the plate facing the light source thanks to the servo motors movement.

- **Test 5: Data Collection 1:**

  The first test of data collection will consist of getting the status of the system using the Android mobile device. To see if the wireless communication works, I will change the status with the button, and the expected result is to see a string that communicates in which state the system is (“standby” or “control”).

- **Test 6: Data Collection 2:**

  The second and final test of data collection will consist of making petitions to the system via the wireless communication while moving the light source. The different values consulted will be the position of the servo motors (Pitch and Yaw) and the LDRs voltage levels, which are the expected output. Obviously, when the light source moves, these values should be different.

- **Test 7: Solar Panel Efficiency:**

  This final test is a little bit offtopic, but it will be interesting to do the comparative. This test will consist of what I have mentioned above, see the difference between having the solar panel in a static position and to have it in a solar tracker mechanism. The result should be that a solar tracker should perform better as it aligns the solar panel in the best possible position.