# Hardware strategy

From the challenge brief, the final Derbot design needs to be able to:

* Follow a white line around the course and stop in the finish zone
* Be small enough to fit through a gate way at the start
* Detect if the object in front of it is lit or unlit
* Move the object to the left if unlit and to the right if lit
* Display the number of lit and unlit object there were around the course on a seven-segment display once inside the finish zone

Figure 1 shows a diagram of the proposed Derbot design. This design should meet all the needed criteria mentioned above.

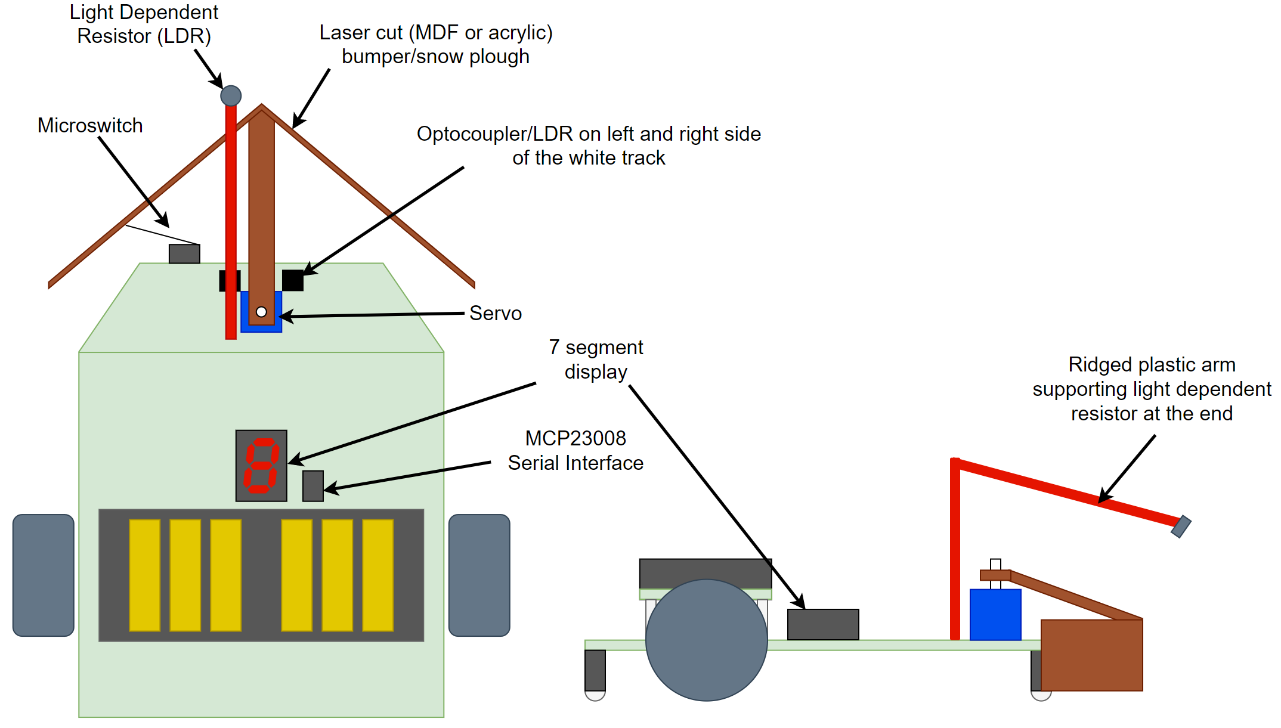


Figure 1 - Diagram showing the plan view (left) and the side elevation (right) of the Derbot design.

The two optocouplers (or LDRs) will be used to keep the Derbot on the track, allowing the code to adjust the two rear motors depending on their values.

It will have an arrow shaped bumper to push the objects to the left or right as it moves forward; an arm extending from the bumper will be connected to a servo allowing it to be moved left and right.

The default position of the bumper will be in the right position thus pushing everything to the left unless, the Light Dependent Resistor (LDR) at the end of an extended arm detects high level of light from the object, where the servo will move the bumper to the left position pushing the lit object to the right as the Derbot carries on moving forward along the line.

To detect the unlit objects being pushed to the left, the built in microswitch on the left of the Derbot will be connected to the bumper, the bumper will slightly flex when the object is push passed, clicking the microswitch and detecting the object. Both scenarios are shown below in figure 2. By setting the default position of the bumper to the left, the Derbot will not have to slow down when approaching each object, but only change the servo position when it detects high light levels.

There will also be a seven-segment display fitted to the Derbot to show the number of lit and unlit objects detected at the end of the course. This will be via I2C and a MCP23008 serial interface chip.

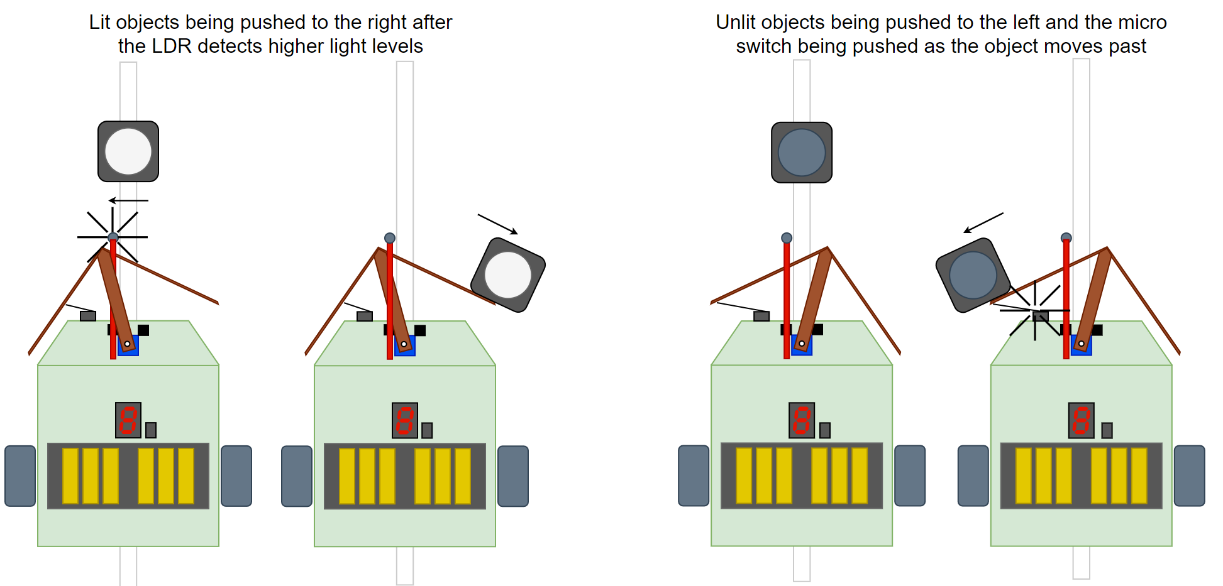
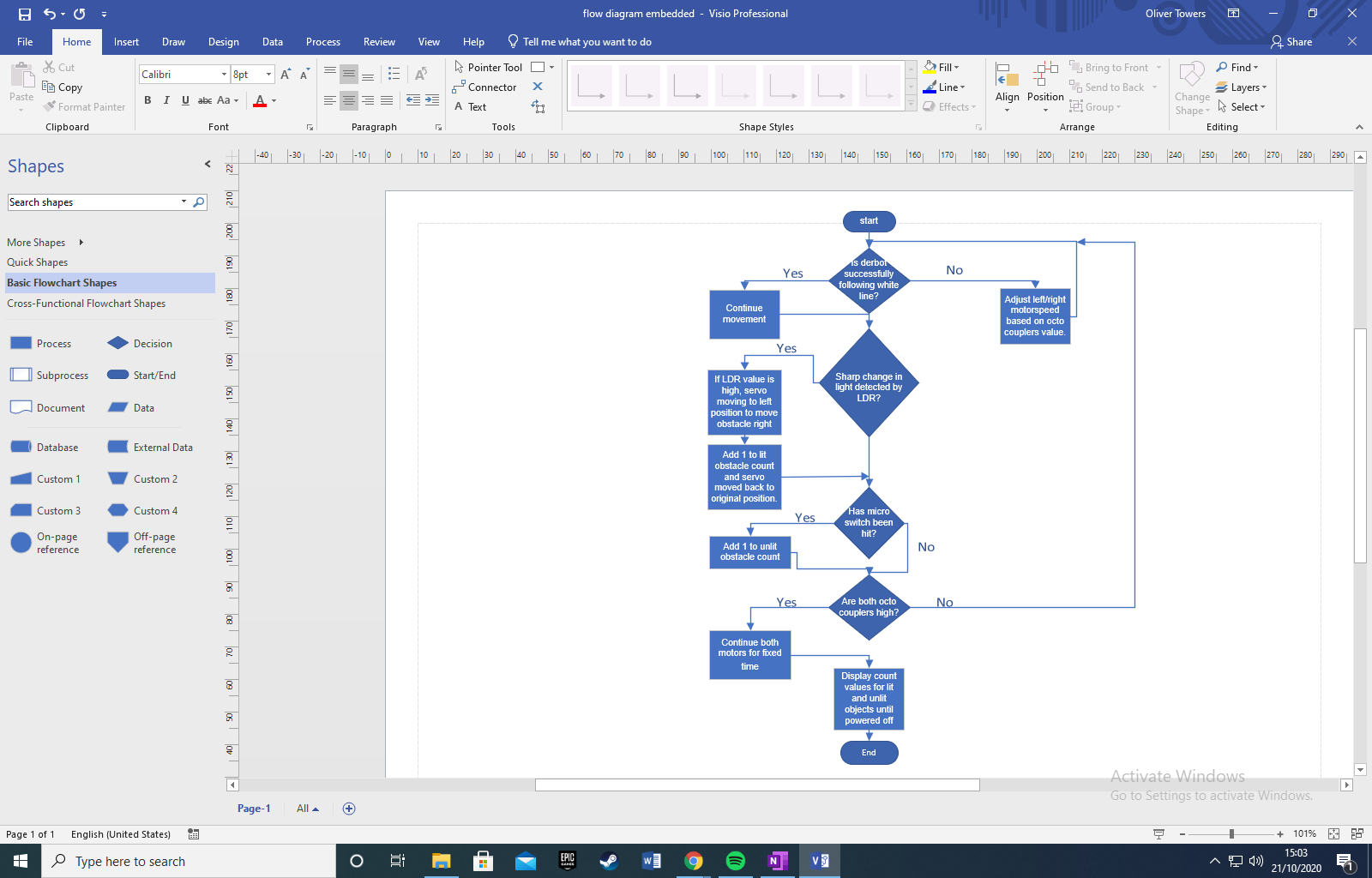


Figure 2 - How the Derbot will react to a lit object (left) and how it will react to an unlit object (right).

Programming Strategy

Figure 3 - Flow chart showing the logical steps the code will make.



# Pic resources

|  |  |  |
| --- | --- | --- |
| **Peripheral** | **Function** | **Use in Derbot** |
| ACD – Analog to Digital Converter | Convert analogue signals into digital values | Read analogue voltage from pins 2, 3 and 5 (the LDRs) |
| FVR – Forward Voltage Reference | Set the voltage reference for the ADC (1.024V, 2.048V or 4.096V) | Setting the voltage reference of the ADC to 1.024V. the ADC has a 10bit resolution which will result in a resolution of 1mV |
| ECCP1 – Enhanced capture compare module 1 | Enhanced PWM mode 1 | This will generate the PWM signal for the left motor of the DERBOT |
| ECCP2 – Enhanced capture compare module 2 | Enhanced PWM mode 2 | This will generate the PWM signal for the right motor of the DERBOT |
| TMR2 – internal Timer 2 of the PIC | This timer will be used for both PWM pulses for the left and right motor | The PWM values are sent into the IN pins of the L293D |
| TMR4 – internal Timer 4 of the PIC | This timer will be used to generate a PWM signal in-code (instead of using a CCP module) | The in-code-generated PWM signal will be used as the data for the SERVO |
| TMR6 – internal Timer 6 of the PIC | This timer will be used at the end of the program to determine the stopping time | When the ending white line is detected, the DERBOT will stop when TMR6 overflows (1s) |
| MSSP1 | I2C – PIC will be the MASTER | The PIC will be the Master |
| MSSP2 | I2C – external I/O will be the SLAVE | The I/O expansion will be the SLAVE |

Table 1 - Table showing the peripherals to be used from the PIC microcontroller, their functions, and their purpose on the Derbot.

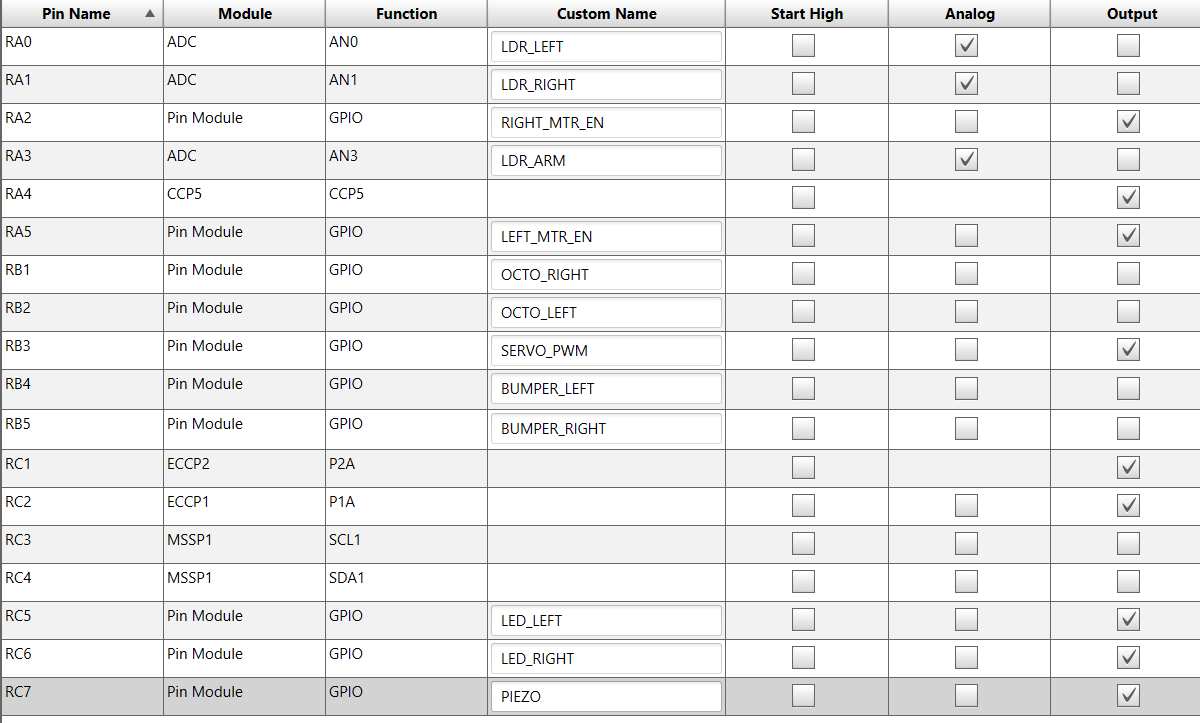


Figure 4 - Screenshot from MPLABX showing the pin names and custom names which will be used.

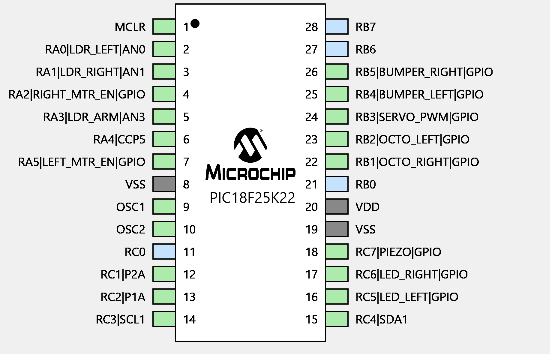


Figure 5 - Screenshot from MPLABX showing a graphical representation of the same data from figure 4.

# Allocation of work and time

## Allocation of work

Charlie

* Object detection and moving

Oliver

* I2C communication and seven segment display

Mateo

* White line following/starting and stopping

## Gannt chart

Figure 6 - The Gannt chart for the design and build of the Derbot system.