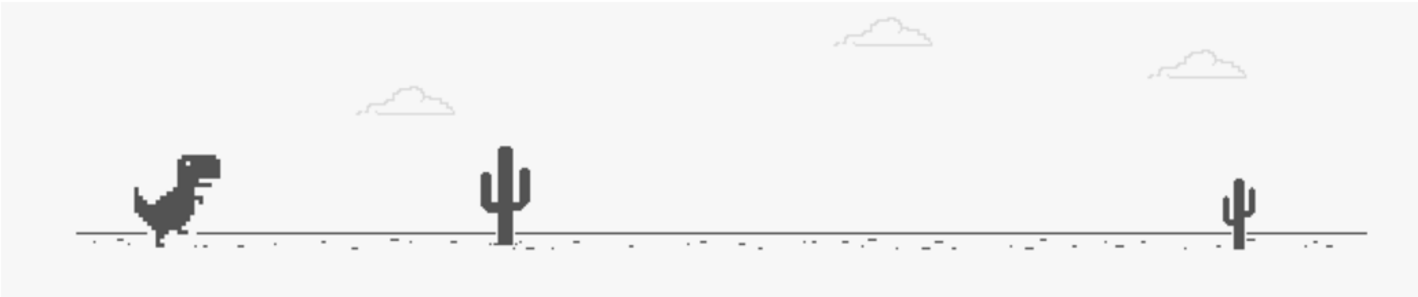
How to control a Dinosaur?



Type of Actuator used:

- *SERVO MOTOR 9G 1 PIECE*
- *Potentiometer 10kohm 1 PIECE*
- *Photoresistor 1 PIECE*

Hardware used

Type of hardware

Ex. ST Nucleo F411RE

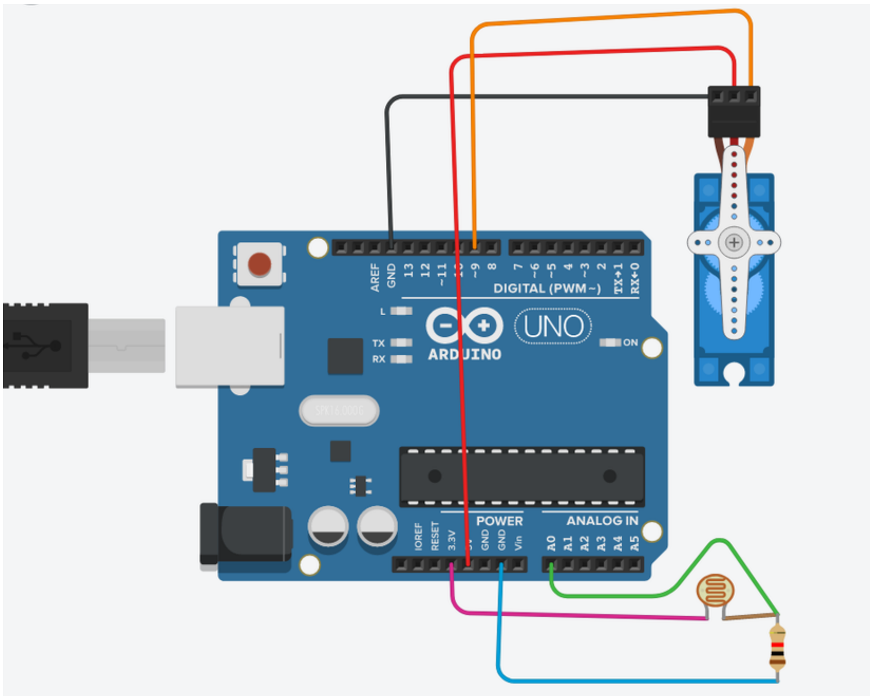
Software used

Type of software → *Ex. STM32 CubeIDE*

Group composed by:

YASIN SHADROUH - *0780715*

HAMED NAHVI - *0780707*



Description of the project idea:
It is not only controlling the Dino and pressing the button, when it comes to control these items, the timing and configuring details. The hardest part is the best spot and best value read by the sensor and the best angle and movement angle for the servo motor according to its speed and the keyboard essential movement. Also It is better to adjust the speed of read and press the button but as the speed of servo limited and it designed to have more power than speed, after some points as the speed of Dino increases, the servo motor's limits comes in the game and we can't go further more.

Actuator description (how it works):
We used the servo motor 9g for doing the operation of pressing the "space button" of the keyboard that allow the precise control of angular motion. The output torque of the motor is 1.8kg/cm means it can lift 1 kg weight at a distance of 1cm, but its speed is 0.1s/60 degrees which it is low in the higher speeds.

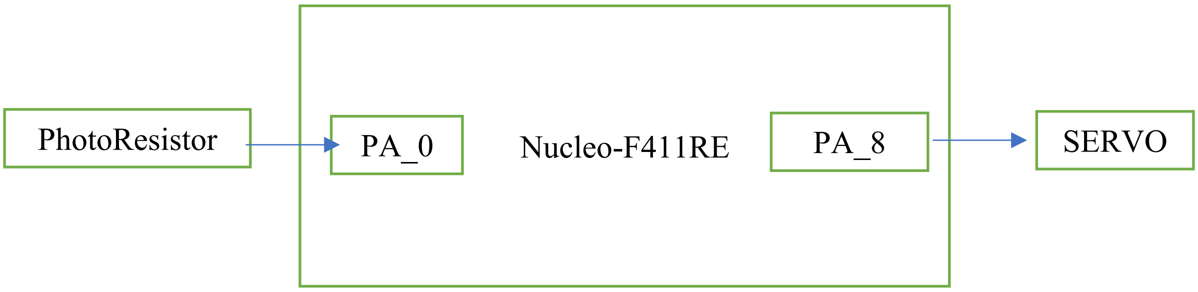
Simply this project work by using the photoresistor to identify the "cactuses" and send the signal if it is darker or brighter area of display, then microcontroller decide to move the servomotor with the correst angle to press the button of keyboard and then move back to the angle which releases the space button but it did not move so far to rich faster to the button for the next jump.

In order control the servo motor we will supply the pulse frequency of 50 Hz and we will change its width which will results in rotation of servo motor.
we will set the timer clock to 45MHz in clock configuration settings and we need frequency output 50Hz we will change its width result in rotation of servo motor.
Divide the 45Mhz by 50 we will get the value nine hundred thousand (900000)

we will distribute this value to prescalar value and ARR value
Prescalar = 900
ARR value =1000

We know the formula t=1/f → t=1/50= 20ms
So 0.5ms is the 2.5% of 20ms
So if we move the value to CCR1 25 then servo move to the position 0 degrees, it means at 0.5 seconds the position of servo is 0 degrees
Likewise at 1.5ms the value of CCR1 will be 75 and the position of servo will be 90 degrees and so on.
For the CubeIDE values are 10000 for 20ms, so it is 1250 for 2.5ms(180 degrees) and 250 for 0.5ms(0 degrees)
And the values for the photoresistor is 0V for 0 lux and 3.3V for 4095 (2 in power with 12 - 1).

Wiring diagram of signal conditioning:



Description of used hardware:
F411RE ELECTRONIC CONTROLLER
PHOTORESISTOR USED FOR ADC CONVERSION
SERVO MOTOR 9G FOR BUTTON PRESS
Peripherals initialized in CubeIDE:
A0 → PA0→ PHOTORESISTOR CHANNEL ADC1_IN0
D7 → PA_8→ SERVO TIM1_CH1

CLOCK CONFIGURATION HCLK = 90MHZ
Prescaler value of Servo is = 180 - 1
ARR value of Servo is = 10000 – 1

$$f_{\text{timer}} = \frac{f_{\text{HCLK}}}{(\text{Prescaler} + 1)}$$

Code written in CubeIDE:

```
/* USER CODE BEGIN 3 */
	HAL_ADC_Start(&hadc1);
	HAL_ADC_PollForConversion(&hadc1, 10);
	lux = HAL_ADC_GetValue(&hadc1);

	if(lux<2300){
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);
		__HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1, 650);
		HAL_Delay(50);
	}else{
		HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
		__HAL_TIM_SET_COMPARE(&htim1,TIM_CHANNEL_1, 560);
	}

	HAL_Delay(20);

  }
  /* USER CODE END 3 */
```