



# BLM4021 Gömülü Sistemler

## Laboratuvar Projesi Final Raporu

Grup No: 10

Proje Kategorisi: Gömülü Sistemler

Kişilerin Çalışma Yüzdesi:

Grup Sorumlusu:	<b>Burak Başol</b>	<b>60</b>	Sistem dizayn, gömülü sistem tasarımı, gömülü sistem kodlaması
	<b>Aiman Idris</b>	<b>40</b>	GUI tasarımı, rapor hazırlanması, video çekimi, fikir, sistem dizaynı, gömülü sistem tasarımı

# İçerik

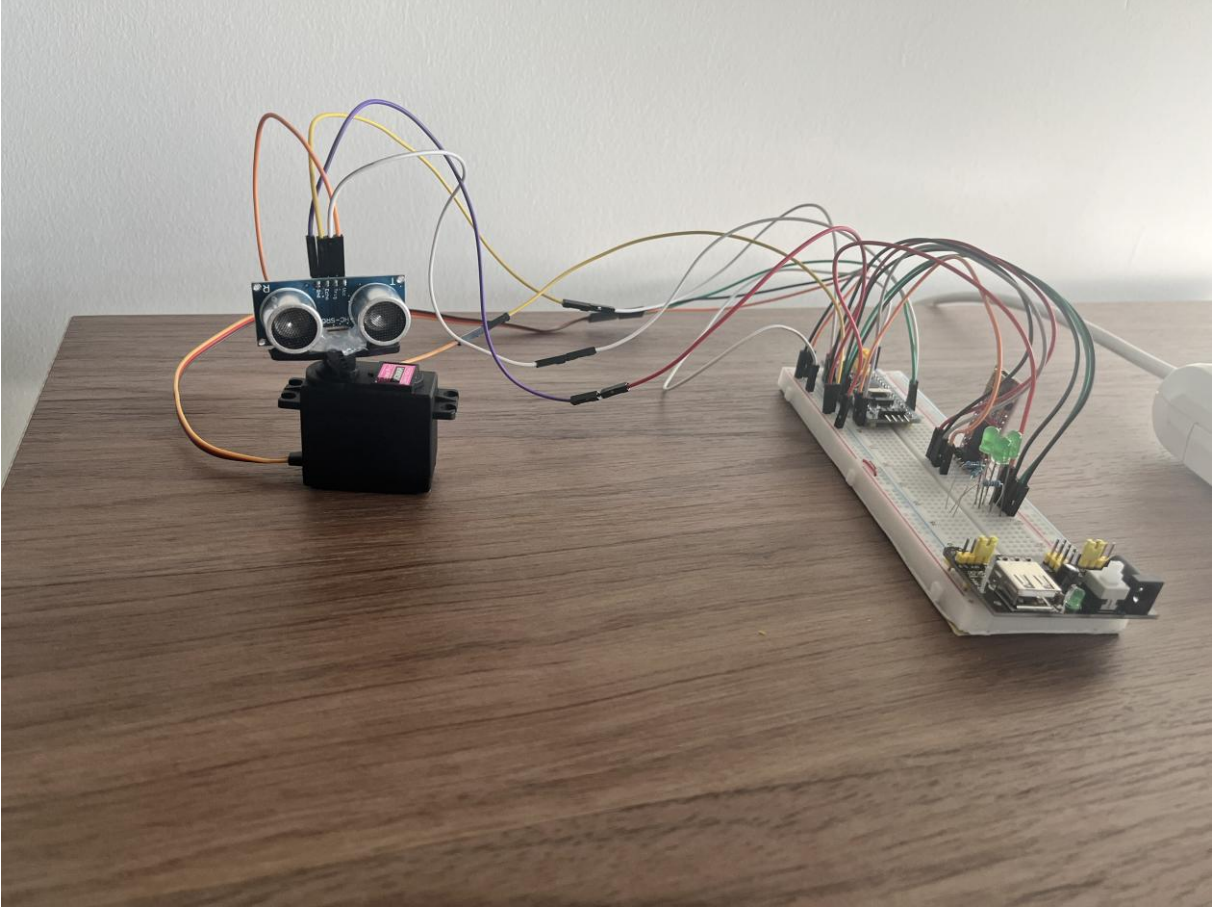
I.	Giriş ve Proje Tanıtımı.....	Sayfa: 3
II.	<i>Fritsing</i> ile Ön Tasarım.....	Sayfa: 5
III.	Kurulan Devre Detayları.....	Sayfa: 7
IV.	Yazılım Tasarımı.....	Sayfa: 11
V.	Sonuçlar, Demo Detayları ve Sunum Linki.....	Sayfa: 15
VI.	Referanslar.....	Sayfa: 18

## I. Giriş ve Proje Tanıtımı

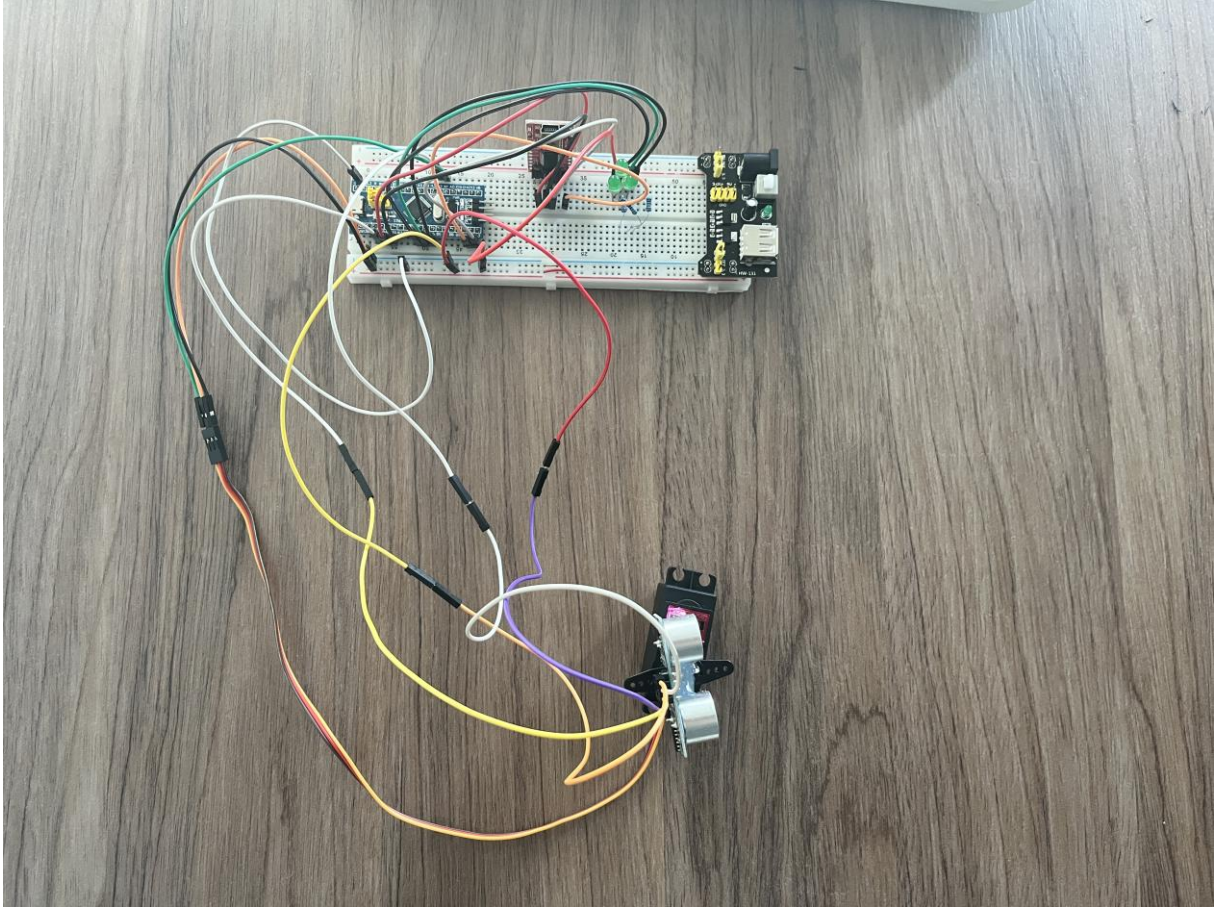
As Burak, i take place in the design of the system and programming the microcontroller and also i helped in other fileds too.

As Aiman, i was responsible for the GUI design and real-time data visualization. I also managed the final report preparation, video demonstration, and assisted in the physical system design and hardware integration.

The objective of the project was designing a basic radar system. For this project we used STM32 Blue Pill as a microcontroller. HC SR04 for the meaningful data. This project uses timers and interrupts to get and send data.



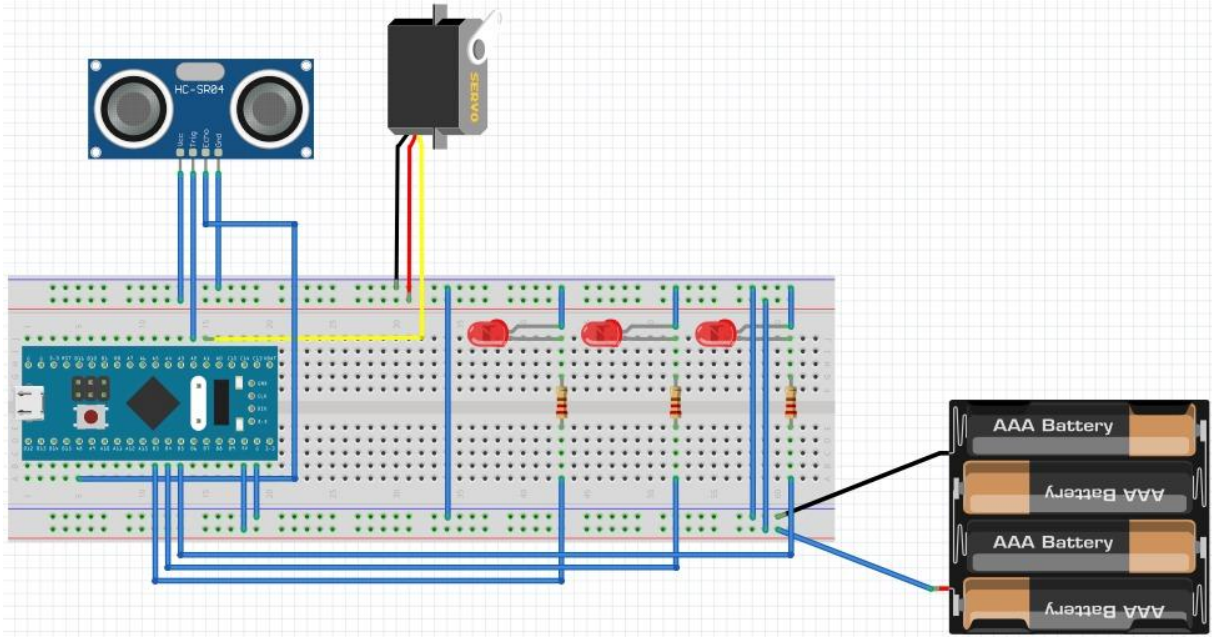
Şekil 1 Sistem Görünümü



*Şekil 2 Sistem Görünümü*

As seen in the figure 1 and 2, for system demo we built it on the breadboard. With jumper cables and external power source.

## II. *Fritsing* ile Ön Tasarım



Şekil 3Fritzing Design

The figure 3 is a representation of the actual demo design from Fritzing.

Servo motors works with 5V power supply because servo motors draw too much current when working. If you supply the current from the microcontroller it will cause a drop in the microcontroller and microcontroller is gonna reset.

Microcontroller uses USB-UART brigde to power and plot the serial data. Because STM32 Blue Pill does not come with a programmer. HC SR04 sensor works with 5V and also it uses interrupt and timer to produce valid data.

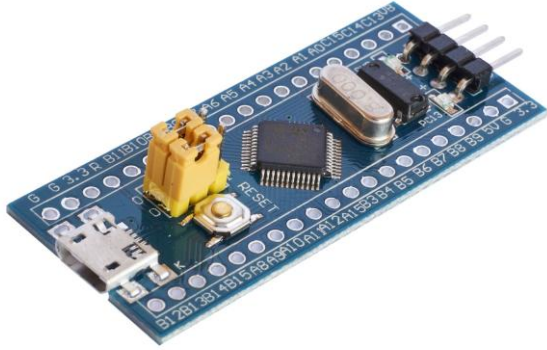
Module	Pin	Function	Logic Level	STM32 Pin
HC SR04	Trigger	Trigger Output	5V	A2
	Echo	Input Capture	5V	A8
Servo Motor	Pwm	Control Output	3.3V	A1
USB UART Bridge	Rx-Tx	Program and print serial	3.3V	A10 A9

Some pins are working with 5V because Stm32 Blue Pill pins are resistant to 5V too. But some pins only accept 3.3V.



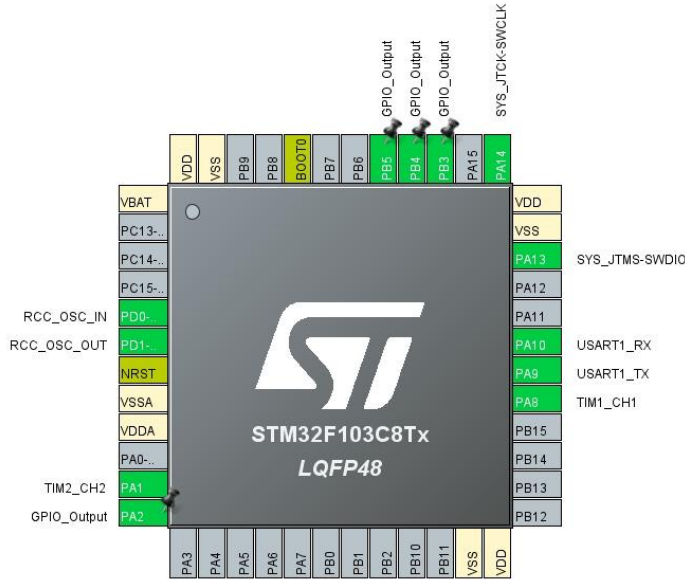


### III. Kurulan Devre Detayları



Şekil 4 STM32 Blue Pill

STM32 Blue Pill Arm M3 çekirdeğine sahip bir mikrodeneleyicidir. Projemizde tercih edilmesinin sebebi küçük ve hızlı olmasıdır. Kendisinde programlayıcı olmadığı için Şekil 8’de yer alan USB UART köprü entegresi kullanılmıştır. Sistem tasarımında ST markasına ait bir IDE olan CubeIDE kullanılmıştır.



Şekil 5 Çipe Ait Görünüm

Öncelikle kartın bağlantıları şekil 5’te görüldüğü üzere yapılmıştır. USART1 programlamak ve verilerin yazdırılması için kullanılmıştır.

Timer 1 ve A2 gpio pini sensör bağlantısı için kullanılmıştır.

Timer 2 ise servo motorun kontrolü için kullanılmıştır.

GPIO çıkış pinleri B3 B4 B5 ise LED’ler için kullanılmıştır.



Şekil 6 MG996R Servo Motor

Şekil 6'daki servo motor MG996R 180 derece dönme açısına sahip ve PWM sinyali ile kontrol edilen bir motordur.



Şekil 7 HC SR04 Ultrasonic Sensor

Şekil 7'deki sensör ultrasonik sensördür. 2cm ile 400 cm arası ölçüm yapabilmektedir.



Şekil 8 FT232R USB UART Bridge

Şekil 8'deki FT232R USB UART köprüsü seri port üzerinden programlama ve çıktı almak için kullanılmaktadır.

UART protokolü ile STM32'ye bağlanmıştır.



Şekil 9 HW131 BreadBoard Power Circuit

Şekil 9'daki modül breadboarda güç verebilmek için kullanılmıştır.

Servo ve sensör gücü bu modül ile karşılanmaktadır.

Sistemin düzgün çalışabilmesi için tüm toprak bağlantısı ortaktır.







## IV. Yazılım Tasarımı

The software architecture is based on the STM32 HAL library and uses interrupt driven state machine structure.

```
uint16_t pwm;
int angle;

for (angle = 0; angle <= 180; angle += 2) {
    HAL_GPIO_TogglePin(LED_PORT, LED_STM_PIN);
    stm_led = HAL_GPIO_ReadPin(LED_PORT, LED_STM_PIN);
    pwm = SERVO_MIN + ((uint32_t)(SERVO_MAX - SERVO_MIN) * angle) / 180;
    __HAL_TIM_SET_COMPARE(&htim2, TIM_CHANNEL_2, pwm);

    HAL_GPIO_TogglePin(LED_PORT, LED_SERVO_PIN);
    servo_led = HAL_GPIO_ReadPin(LED_PORT, LED_SERVO_PIN);

    HAL_Delay(40);

    HCSR04_Read();
    HAL_Delay(40);
    if (Distance > 0 && Distance < 400) {
        HAL_GPIO_TogglePin(LED_PORT, LED_SENSOR_PIN);
        sensor_led = HAL_GPIO_ReadPin(LED_PORT, LED_SENSOR_PIN);
    }

    printf("%d,%d,%d,%d,%d\r\n", angle, Distance, stm_led, servo_led, sensor_led);
}

for (angle = 180; angle >= 0; angle -= 2) {
    HAL_GPIO_TogglePin(LED_PORT, LED_STM_PIN);
    stm_led = HAL_GPIO_ReadPin(LED_PORT, LED_STM_PIN);
    pwm = SERVO_MIN + ((uint32_t)(SERVO_MAX - SERVO_MIN) * angle) / 180;
    __HAL_TIM_SET_COMPARE(&htim2, TIM_CHANNEL_2, pwm);

    HAL_GPIO_TogglePin(LED_PORT, LED_SERVO_PIN);
    servo_led = HAL_GPIO_ReadPin(LED_PORT, LED_SERVO_PIN);

    HAL_Delay(40);

    HCSR04_Read();
    HAL_Delay(40);
    if (Distance > 0 && Distance < 400) {
        HAL_GPIO_TogglePin(LED_PORT, LED_SENSOR_PIN);
        sensor_led = HAL_GPIO_ReadPin(LED_PORT, LED_SENSOR_PIN);
    }

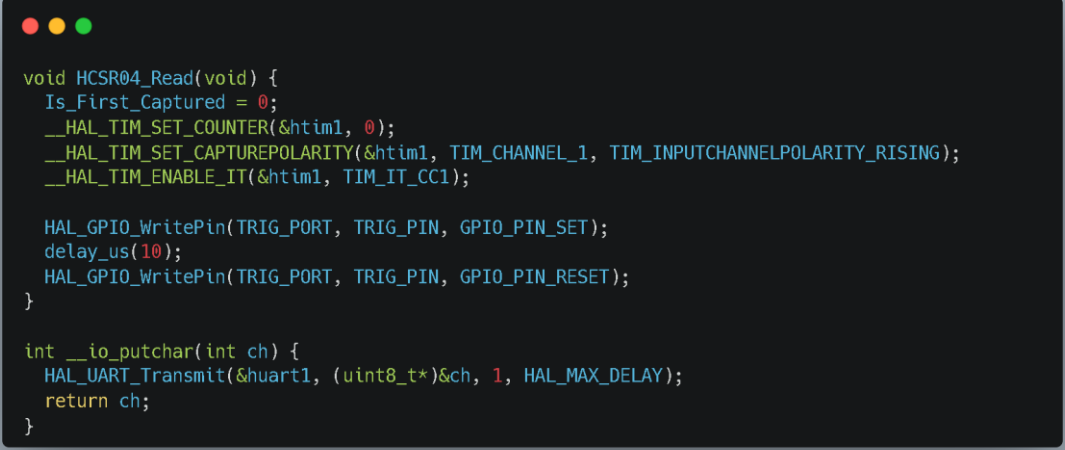
    printf("%d,%d,%d,%d,%d\r\n", angle, Distance, stm_led, servo_led,
        sensor_led);
}
```

Şekil 10 Ana Mantık Bloğu

Figure 10 represents where the main control happens.

The system works in this order:

1. The for loops are deciding the angle variables value.
2. If for loop is working then STM32 is working so LED is blinked.
3. Calculate the pwm variables value according to the angles value.
4. Set the servo motor to that pwm value.
5. Toggle the LED that represents the servo.
6. Read value from HC SR04.
7. If distance is meaningful value toggle the LED that's responsible for the sensor.
8. Print everything according to their values to the serial monitor.



```
void HCSR04_Read(void) {
    Is_First_Captured = 0;
    __HAL_TIM_SET_COUNTER(&htim1, 0);
    __HAL_TIM_SET_CAPTUREPOLARITY(&htim1, TIM_CHANNEL_1, TIM_INPUTCHANNELPOLARITY_RISING);
    __HAL_TIM_ENABLE_IT(&htim1, TIM_IT_CC1);

    HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_SET);
    delay_us(10);
    HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);
}

int __io_putchar(int ch) {
    HAL_UART_Transmit(&huart1, (uint8_t*)&ch, 1, HAL_MAX_DELAY);
    return ch;
}
```

Şekil 11 Sensor Read and Print Function

Read function reads the data from the sensor .

Putchar function is used for printing data to serial monitor.

```

void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef* htim) {
    if (htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1) {
        if (Is_First_Captured == 0) {
            IC_Val1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);
            Is_First_Captured = 1;
            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1,
                                           TIM_INPUTCHANNELPOLARITY_FALLING);
        }

        else if (Is_First_Captured == 1) {
            IC_Val2 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);
            __HAL_TIM_SET_COUNTER(htim, 0);

            if (IC_Val2 > IC_Val1) {
                Difference = IC_Val2 - IC_Val1;
            }

            else if (IC_Val1 > IC_Val2) {
                Difference = (0xffff - IC_Val1) + IC_Val2;
            }

            Distance = Difference * .034 / 2;
            Is_First_Captured = 0;

            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1, TIM_INPUTCHANNELPOLARITY_RISING);
            __HAL_TIM_DISABLE_IT(&htim1, TIM_IT_CC1);
        }
    }
}

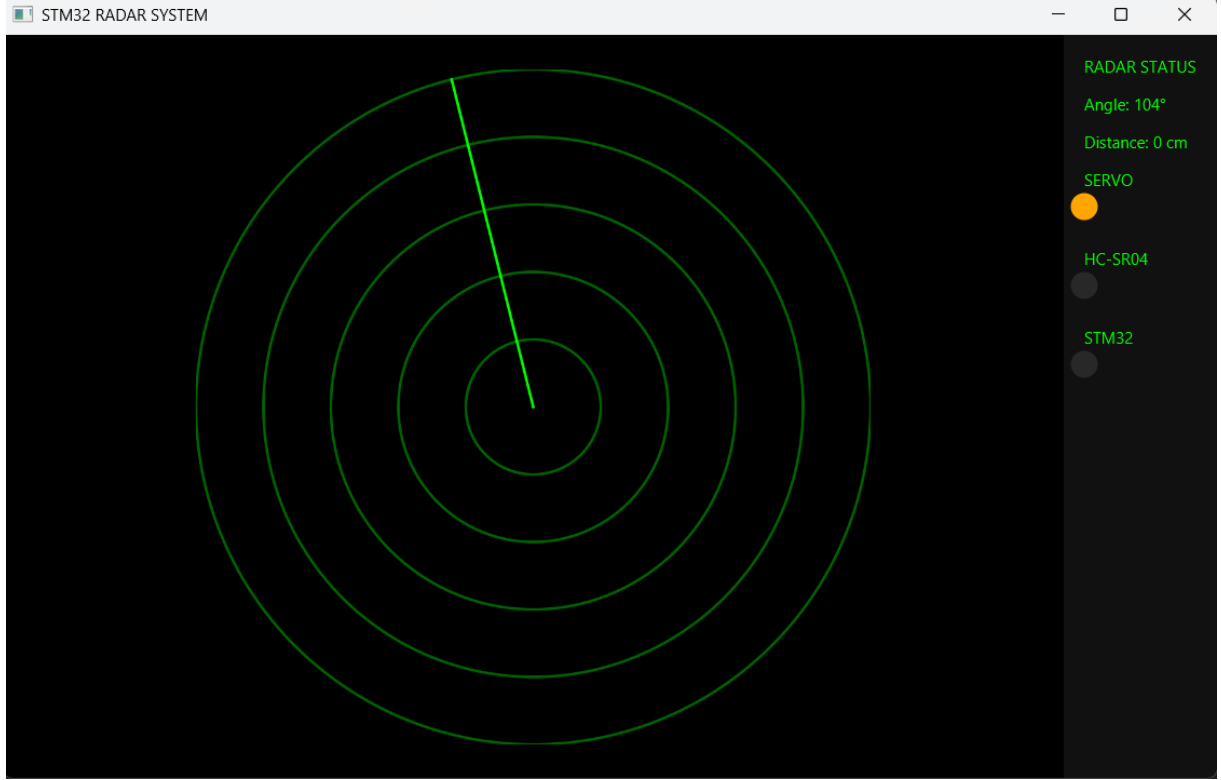
```

Şekil 12 Timer Control Block

Capture Callback function puts the data to the Distance variable and sets the clears the interrupt.



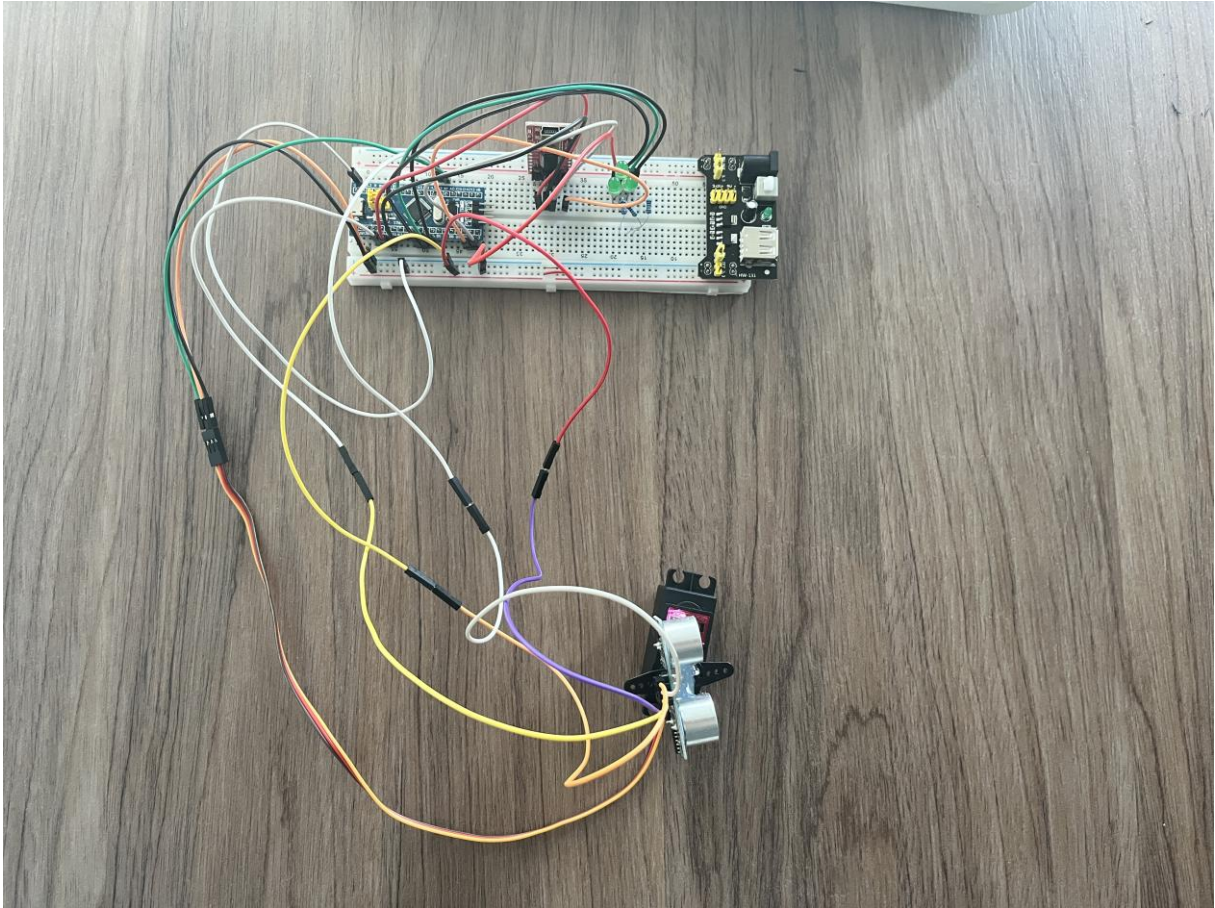
## V. Sonular, Demo Detayları ve Sunum Linki



Şekil 13 GUI

In the figure 13, we show the radar systems work. GUI demonstrates the LED's working in the right side. Andalso it gives written information. If sensor gets a distance change it shows on the main side of the GUI.





Şekil 14 Demo

Figure 14, shows the demo of the project.

What we have learnt from this project is that the problem could be on the hardware or on the software that's why working in between hardware and software takes time and exhausts people. Being patient and trying new things and checking for the problems is the key to success.

If we talk about the project itself, this demo shows the workflow correctly. But upgrading the hardware to better ones is gonna give more accurate smooth results.

Video : <https://youtu.be/RxD9rPxmH8g>



## VI. Referanslar