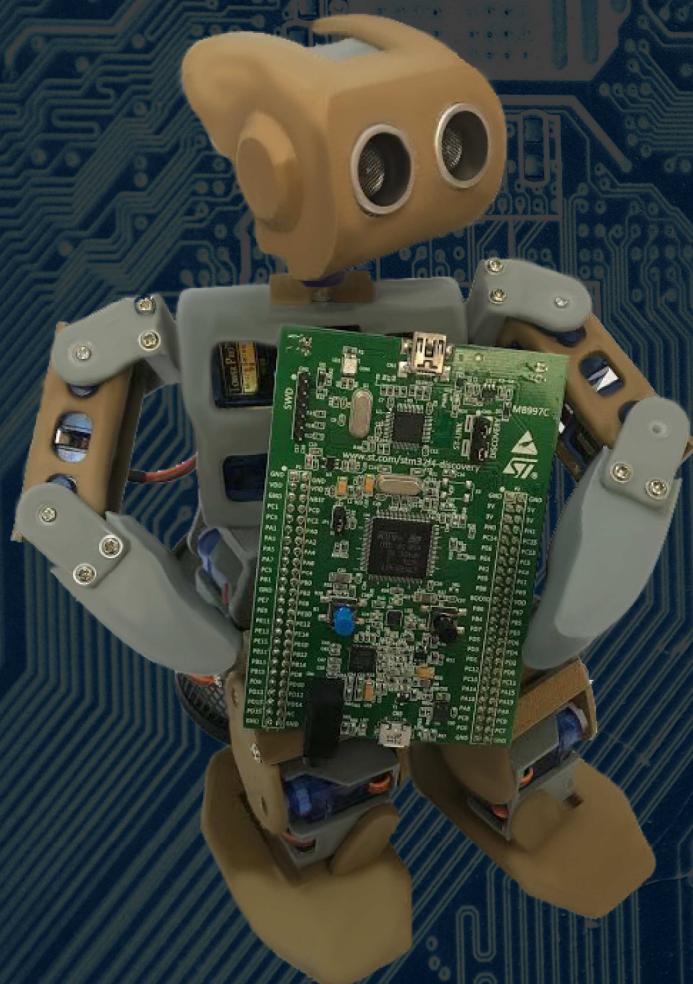




Fatih Sultan Mehmet
Vakıf Üniversitesi

Mikroişlemci Sistemleri Dersi Laboratuvar Uygulamaları



Öğr. Gör. Musa AYDIN

Araş. Gör. Ö. Faruk GÖKSU

İçindekiler

ÖNSÖZ	3
1. STM32F4 Discovery Geliştirme Kiti	4
2. Hafta: Kurulum	6
2.1 Amaç:	6
2.2 Keil MDK µVision IDE kurulumu	6
2.3 STM32CubeMX Code Generator Setup	8
3. Hafta: GPIO (Genel Amaçlı Giriş Çıkış)	10
3.1 Gerekli Malzemeler	10
3.2 Genel Bilgi	10
3.3 Sorular	10
3.4 Uygulama Sırasında Kullanılacak C Fonksiyonları	10
3.5 Devre Şeması	11
3.6 Deneyin Yapılışı	12
3.7 Ödev	16
4. Hafta: External Interrupt (Harici-Buton Kesmesi)	17
4.1 Gerekli Malzemeler	17
4.2 Genel Bilgi	17
4.3 Sorular	17
4.4 Devre Şeması	18
4.5 Deneyin Yapılışı	18
4.6 Ödev	22
5. Hafta: Analog - Dijital Çevirici (ADC)	23
5.1 Gerekli Malzemeler	23
5.2 Genel Bilgi	23
5.3 Sorular	24
5.4 Devre Şeması	25
5.5 Deneyin Yapılışı	25
5.6 Ödev	29
6. Hafta: Seri Haberleşme (Universal Synchronous Asynchronous Receiver Transmitter - USART)	30
6.1 Gerekli Malzemeler	30
6.2 Genel Bilgi	30
6.3 Sorular	30

6.4	Devre Şeması	31
6.5	Deneyin Yapılışı	31
6.6	Ödev	36
7.	Hafta: Zaman Kesmesi (Timer Interrupt)	37
7.1	Gerekli Malzemeler	37
7.2	Genel Bilgi	37
7.3	Sorular	39
7.4	Devre Şeması	39
7.5	Deneyin Yapılışı	39
7.6	Ödev	40
8.	Hafta: Pulse Width Modulation (PWM) with ADC	41
8.1	Gerekli Malzemeler	41
8.2	Genel Bilgi	41
8.3	Sorular	43
8.4	Devre Şeması	44
8.5	Deneyin Yapılışı	44

FATİH
SULTAN
MEHMET
VAKIF ÜNİVERSİTESİ
2010

Sekiller Tablosu

Şekil 1-1 STM32F4 Discovery Board Şematik Gösterimi-----	4
Şekil 2-1 Keil MDK download linki -----	7
Şekil 2-2 Keil MDK versiyon seçimi -----	7
Şekil 2-3 Keil Dowload -----	8
Şekil 2-4 CubeMX download link -----	9
Şekil 2-5 CubeMX Seçimi ve indirilmesi -----	9
Şekil 3-1 GPIO deneyi devre şeması -----	11
Şekil 3-2 CubeMX Programı Açılış Sayfası -----	12
Şekil 3-3 CubeMX programlanacak işlemcinin seçimi-----	12
Şekil 3-4 Mikroişlemci Pin Konfigürasyon Penceresi-----	13
Şekil 3-5 Konfigürasyonu yapılan işlemci için Keil Proje dosyasının oluşturulması -----	13
Şekil 3-6 Projenin Kod dosyasının hazırlanması -----	14
Şekil 3-7 Keil IDE' si main.c dosyası -----	14
Şekil 3-8 Projenin Derlenmesi -----	15
Şekil 3-9 Projenin derlenme sonrası hata ekranı -----	15
Şekil 4-1 External Interrupt deneyi devre şeması-----	18
Şekil 5-1 Analog sinyalin Dijital sinyale dönüştürülmesi-----	23
Şekil 5-2 ADC çevrim şematik göstergesi -----	23
Şekil 5-3 ADC deneyi devre şeması -----	25
Şekil 5-4 ADC deneyi için İşlemci Pin Konfigürasyonları-----	25
Şekil 5-5 ADC deneyi için ADC çevrim parametreleri ayar sayfası-----	26
Şekil 5-6 ADC deneyi için CubeMX ADC interrupt ayar sayfası-----	27
Şekil 6-1 Seri Haberleşme Temsili göstergesi -----	30
Şekil 6-2 USART deneyi devre şeması -----	31
Şekil 6-3 Seri haberleşme deneyi işlemci pin konfigürasyon ekranı-----	31
Şekil 6-4 Seri Haberleşme için işlemci pin seçimi-----	32
Şekil 6-5 USART modülü parametre seçimi -----	33
Şekil 6-6 USART modülü interrupt aktif etme -----	33
Şekil 7-1 STM32F4xx Serisi Mikrodenetleyilerin Timer özellikleri-----	37
Şekil 7-2 Timer Birimi Değişkenler Tablosu -----	38
Şekil 7-3 Timer Interrupt Devre Şemeası -----	39
Şekil 7-4 Timer Deneyi Parametre ayarları -----	40
Şekil 8-1 PWM deneyi Timer Parametre ayarları -----	42
Şekil 8-2 PWM deneyi Timer Compare Register şematik göstergesi-----	42
Şekil 8-3 Timer Counter Register şematik göstergesi-----	43
Şekil 8-4 Timer Counter Register şematik göstergesi-----	43
Şekil 8-5 PWM deneyi devre şeması -----	44

ÖNSÖZ

Bu deney kitabı mikroişlemci dersinin laboratuvar deneyleri için hazırlanmıştır. Bu deneylerde ARM çekirdek tabanlı mikrodenetleyicilere sahip STM32 serisi geliştirme kartları kullanılmaktadır.

Her deneyin başında deneye ilgili temel teorik ve pratik bilgiler verilmiştir. Öğrencinin konu ile ilgili daha detaylı bilgileri kitaplardan ve ders notlarından öğrenmesi ve sorular kısmında istenenleri yaparak laboratuvara gelmesi beklenmektedir. Hazırlıksız gelen öğrencilerin laboratuvar notları kırılacaktır.

Deneyler bilgisayar üzerinde gerekli programlama araçları kullanılarak yapılacaktır. Her öğrenci deney foyünde anlatılan adımlara uygun olarak deneyi gerçekleştirmekle yükümlüdür. Deney sonrası laboratuvar bilgisayarlarında oluşturulan proje dosyalarının saklanması öğrenci sorumluluğundadır. Deney sonrasında verilen ödevler bir sonraki hafta yapılacak laboratuvara kadar yapılmalı ve laboratuarın ilk 15 dakikasında teslim edilmelidir. Kopya olduğu anlaşılan ödevlere “0” verilecektir

Deneylerden bir şeyle öğrenemek için mutlaka konu ile ilgili ön çalışma yapmak gereği akıldan çıkarılmamalıdır.

Her öğrenci foyde bulunan bütün deneyleri yapmak zorundadır. Geçerli mazeretinden dolayı bazı deneyleri kaçırın öğrenciler son deney haftasında yapamadıkları deneyleri telafi ederler. İki deneyden fazla deneyi yapmayan öğrenci devamsızlıktan sınıfta kalır.

Bütün öğrencilerime faydalı ve yardımcı olması dileklerimizle,

Kapak tasarımı için öğrencimiz **Veysel Burak KELEŞ** 'e teşekkür ederiz.

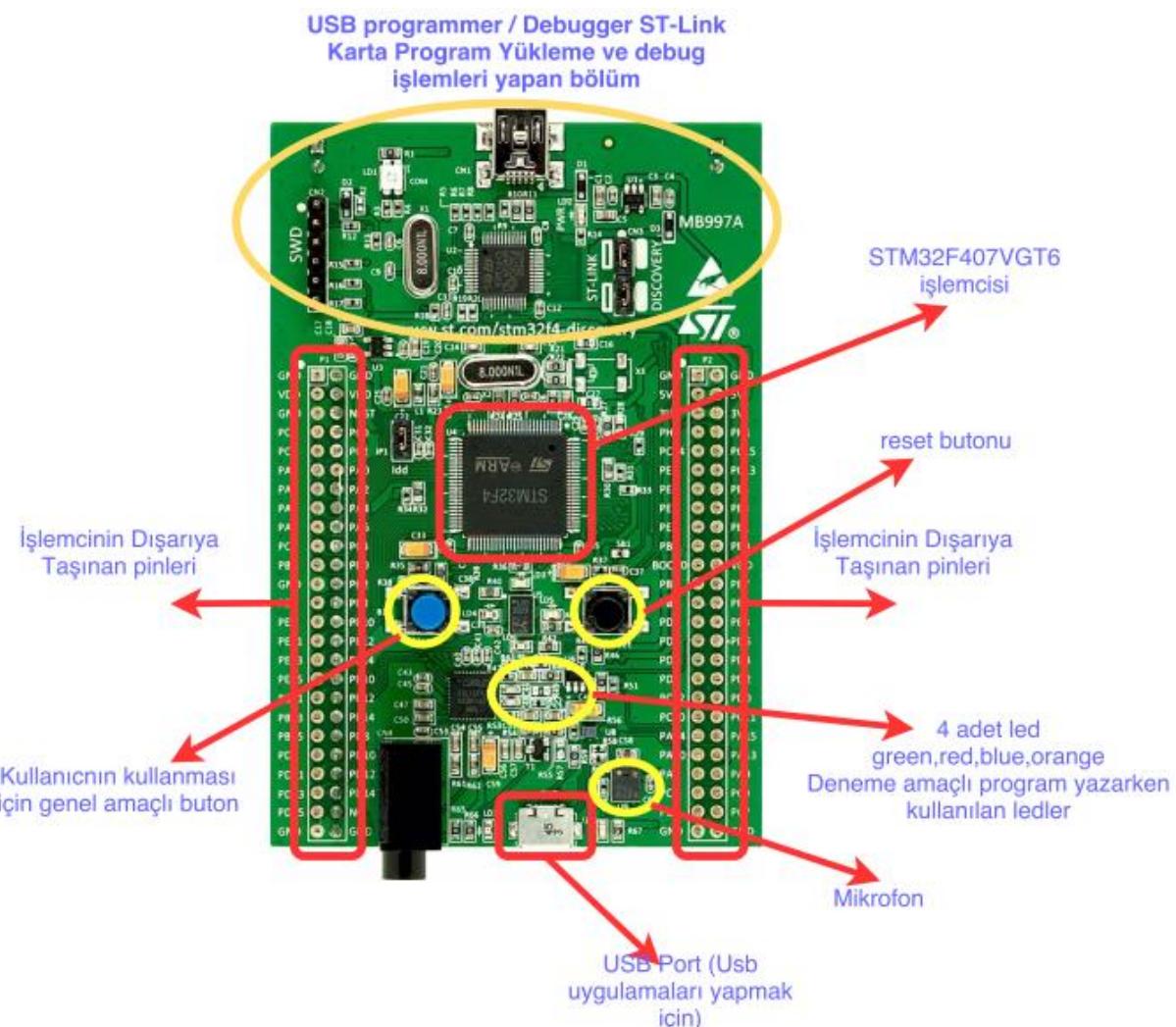
Öğr. Gör. Musa AYDIN

Araş. Gör. Ömer Faruk GÖKSU

1. STM32F4 Discovery Geliştirme Kiti

STM32F4 board'u STMicroelectronics firmasının üretmiş olduğu ve üzerinde ARM Cortex M4 tabanlı 168 MHz'lik bir mikrodenetleyici bulunduran oldukça kullanışlı bir kittir. ARM Cortex M4 çerkirdeğinin kullanılmış olması bu deney kitine ek bir özellik ve uygulamada bir artı katmış durumda. **Cortex M4 üzerinde DSP uygulamalarını yapabilmeye olanak tanıyan FPU(Floating Point Unit)** birimini barındırmaktadır.

İşlemci üzerinde FPU biriminin bulunması işlemciye ondalıklı sayılarla daha hızlı işlem yapabilme yeteneği kazandırmaktır. Özelliklerine geçmeden önce board üzerinde bulunan birimleri, giriş çıkışları vs. Şekil 1-1'de gösterilmiştir.



- STM32F407VGT6 mikroişlemci üzerinde 32-bit ARM Cortex-M4F çekirdek, **1 MB Flash, 192 KB RAM** bulunan **LQFP100 paketi**
- Kart üzerinde yer alan ST-LINK/V2 kullanım modunu değiştiribilen switch ile tek başına da kullanılabilen ST-LINK/V2 (SWD konnektör ile programlama ya da debug için)
- Kartın güç kaynağı: USB veri yolu üzerinden veya harici bir 5 V besleme gerilimi
- Harici uygulama güç kaynağı: 3 V ve 5 V
- LIS302DL veya LIS3DSH ST MEMS 3 eksenli accelerometer
- MP45DT02, ST MEMS ses sensörü, çok yönlü dijital mikrofon
- CS43L22, entegre D sınıfı hoparlör sürücüsü ile ses DAC
- Sekiz tane LED: USB iletişim için LD1 (kırmızı / yeşil), LD2 (kırmızı) 3,3 V güç ON, Dört adet kullanıcı LEDi, LD3 (turuncu), LD4 (yeşil), LD5 (kırmızı) ve LD6 (mavi), 2 USB OTG LED LD7 (yeşil) VBus ve LD8 (kırmızı) aşırı akım
- İki buton (kullanıcı ve reset)
- Mikro-AB konnektör ile USB OTG FS

STM32F4'ün özelliklerine bakacak olursak, en dikkat edilen özelliklerinden bir tanesi 1 MB flash hafıza, bir mikrodenetleyici için oldukça iyi bir hafıza olduğunu söyleyebiliriz.

Diğer özelliklerinden biri ise üzerinde ST MEMS sensörünün bulunması, 3 eksen bir accelerometer ile x,y,z eksenindeki hareketler algınarak uygulamalar yapılabilir.

Board dahili ST-link ile sunulduğu için programlamak için ayrıca bir aparata gerek yoktur.

2.Hafta: Kurulum

2.1 Amaç:

Mikroişlemci Sistemleri dersinin Laboratuvar uygulamasının ilk haftasında uygulamaların yapılması için gerekli olan yazılımlar ve donanım birimlerinin neler olduğu anlatılacaktır. Gerekli yazılımların temin edilmesi ve yükleme aşamaları gösterilecektir.

2.2 Keil MDK µVision IDE kurulumu

Keil™ ARM® firması tarafından sunulan ARM mikroişlemci kullanan tüm donanımlara destek veren bir geliştirme ortamıdır. Keil ücretli bir IDE olarak sunulmasının yanında, Code Limited (32KB) versiyonunu indirip belirli bir boyuta kadar(32KB) uygulamalarımızı geliştirebilmekteyiz. Bu ders kapsamında yapılacak tüm uygulamalar Keil IDE' si kullanılarak gerçekleştirilecektir. Keil geliştirme ortamında yapılan projeler C programlama dili ile geliştirilecektir. **Laboratuvar uygulamalarındaki projelerde kullanılacak C programları için herhangi bir programlama dilinin(Java,C#,C++) bilinmesi yeterli olacaktır.** Keil IDE geliştirme ortamının kurulum aşamaları aşağıda verilmiştir. Kurulum için öncelikle IDE'nin download edilmesi işlemini gerçekleştirmeliyiz.

Adım 1: Keil web sayfasından programımızı indiriyoruz. Google aramaya keil mdk download yazınca karşımıza gelen linklerden Şekil 2-1' de belirtilen link' e tıklanır.

Adım 2: Belirtilen link açıldıktan sonra Şekil 2-2'de gösterilen Keil versiyonu seçilir ve download aşamasına geçilir.

Adım 3: Download aşamasına geçmeden önce Şekil 2-3' deki bilgilerin doldurulması gerekmektedir, istenen bilgiler doldurulduktan sonra Submit diyerek Ürün indirme sayfasına yönlendiriliyoruz.

Keil IDE' si için indirme aşamaları yukarıda belirtilmiştir. Program indikten sonra ise **MDKXXX.exe** uzantılı dosyayı çalıştırıp kurulum işlemini başlatıyoruz ve standart bir program kurulum işlemi gerçekleştiriyoruz.

Keil IDE kurulum işlemi tamamlandıktan sonra STM32 serisi tüm mikroişlemciler için çeşitli ön ayarlamaların yapılmasına olanak tanıyan CubeMX programının kulum aşamasına geçilecektir.

The screenshot shows a search results page from a search engine. The search term 'keil mdk download' is entered in the search bar. Below the search bar, there are navigation tabs: 'Tümü' (selected), 'RESİMLER', 'VİDEO', 'Haritalar', 'Haberler', and 'Kaydettiklerim'. Below these tabs are filters: '9.770.000 Sonuçlar', 'Tarih ▾', 'Dil ▾', and 'Bölge ▾'. The main content area displays several search results:

- MDK Microcontroller Development Kit - Keil** [Bu sayfayı çevir](#)
[www2.keil.com/mdk5](https://www.keil.com/mdk5) ▾
MDK Microcontroller Development Kit. Keil ® MDK is the most comprehensive software development solution for ARM ... Download & Install.
- Keil Product Downloads** [Bu sayfayı çevir](#)
<https://www.keil.com/download/product> ▾
Keil makes C compilers, ... Download Products. Select a product from the list below to download the latest version. MDK-Arm
- Keil Downloads** [Bu sayfayı çevir](#)
<https://www.keil.com/download> ▾
Keil makes C compilers, ... Keil downloads include software products and updates, ... Download current and previous versions of the Keil development tools.

Şekil 2-1 Keil MDK download linki

The screenshot shows the 'arm KEIL' website. The top navigation bar includes links for 'Products', 'Download', 'Events', and 'Support', along with a search bar labeled 'Search Keil.' Below the navigation bar, the heading 'Download Products' is displayed. A sub-instruction 'Select a product from the list below to download the latest version.' is present. The page lists four products:

- MDK-Arm**
Version 5.24a (July 2017)
Development environment for Cortex and Arm devices.
- C251**
Version 5.59 (October 2016)
Development tools for all 80251 devices.
- C51**
Version 9.56 (August 2020)
Development tools
- C166**
Version 7.56 (October 2020)
Development tools

At the bottom of the page, a note states: 'Keil products use a License Management system - without a current license the product runs as a Lite/Evalu'

Şekil 2-2 Keil MDK versiyon seçimi

Şekil 2-3 Keil Dowload

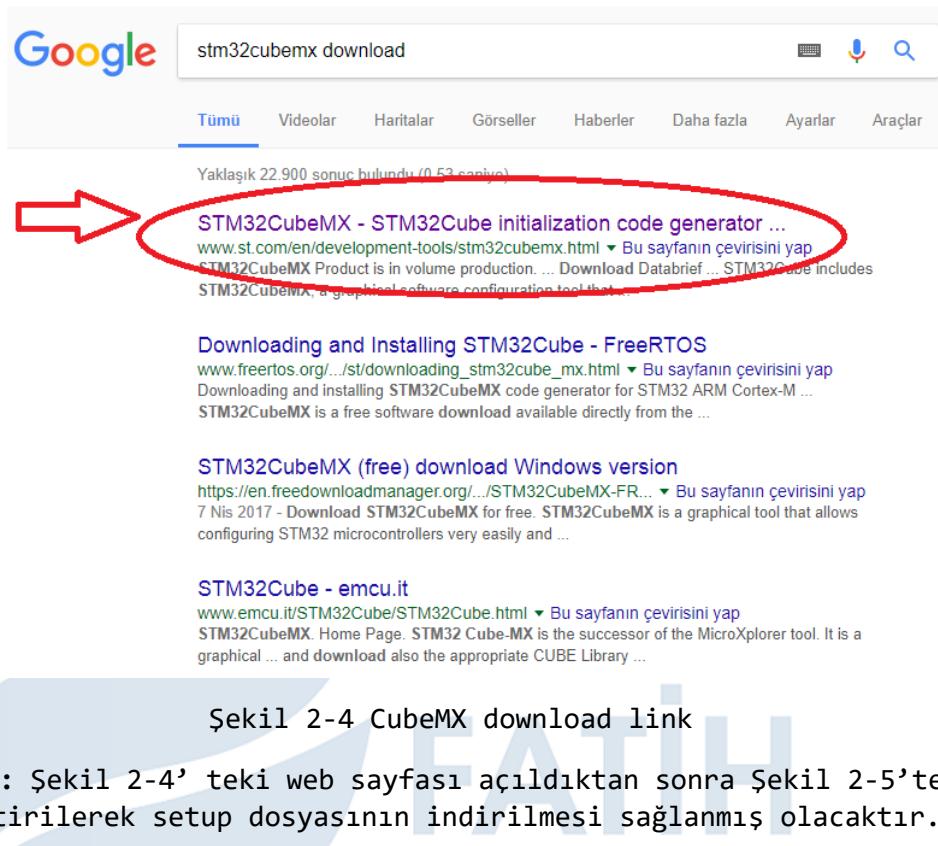
2.3 STM3CubeMX Code Generator Setup

STM3CubeMX, STMicroelectronics tarafından java programlama dili kullanılarak geliştirilmiş bir programdır. STM3CubeMX, mikroişlemci uygulamaları geliştirme aşamalarında karşılaşılan zorluklardan geliştiricileri kurtarmak için görsel bir yüz sunar.

Bu görsel ara yüz ile STMicroelectronics firmasının üretmiş olduğu tüm microdenetleyici ve mikroişlemci'ler için gerekli konfigürasyonlar yapılabilmektedir. Geliştirilmek istenen uygulamaya özgü olarak STM3CubeMX microdenetleyicinin port ayarları, clock frekansı, haberleşme birimi ayarları vb. gibi işlemciye dair her türlü ayar herhangi bir kod yazılmadan gerçekleştiriliyor.

STM3CubeMX download aşaması:

Adım 1: STM3CubeMX indirme işlemi için Google aramaya stm32cubemx download yazıyoruz ve Şekil 2-4' te belirtilen link seçili ilgili sayfaya yönlendiriliyoruz.



Şekil 2-4 CubeMX download link

Adım 2: Şekil 2-4' teki web sayfası açıldıktan sonra Şekil 2-5'teki adım yerine getirilerek setup dosyasının indirilmesi sağlanmış olacaktır.

QUICK VIEW	RESOURCES	TOOLS AND SOFTWARE	GET SOFTWARE
STM32CubeL1	ST	Embedded software for STM32 L1 series (HAL, Low-Layer APIs and CMSIS (CORE, DSP, RTOS), USB, File system, RTOS, Touch Sensing, Graphic - coming with examples running on ST boards: STM32 Nucleo, Discovery kits and Evaluation boards)	Get Software
STM32CubeL4	ST	Embedded software for STM32L4 series series (HAL, Low-Layer APIs and CMSIS (CORE, DSP, RTOS), USB, TouchSensing, File system, RTOS, Graphic - coming with examples running on ST boards: STM32 Nucleo, Discovery kits and Evaluation boards)	Get Software

GET SOFTWARE

Part Number	Software Version	Marketing Status	Supplier	Order from ST
STM32CubeMX	4.22.1	Active	ST	Get Software

Şekil 2-5 CubeMX Seçimi ve indirilmesi

3.Hafta: GPIO (Genel Amaçlı Giriş Çıkış)

3.1 Gerekli Malzemeler

- STM32 geliştirme kartı
- USB kablosu
- 1 x Led
- 1 x 560 ohm direnç

3.2 Genel Bilgi

Mikroişlemci sistemlerinin temel çevre birimlerinden biri olan GPIO (General Purpose Input Output – Genel Amaçlı Giriş Çıkış) portu, işlemci ile dış dünya arasında dijital haberleşmeyi sağlamaktadır. Bu pinler giriş ve çıkış olarak ayarlanabilmektedir.

Pinlerin kullanımı ile ilgili dikkat edilecek hususlar;

- Çıkış olarak kullanılan pinler için Lojik 0 - 0 V, Lojik 1 - 3 V seviyesindedir.
 - Giriş olarak kullanılan pinler alternatif fonksiyon olarak kullanılmıyor ise maksimum uygulanabilecek voltaj seviyesi 5 V'dur.
 - Çıkış olarak kullanılan pinlerin sink-source akımı maksimum 20 mA'dır. Pinlerden çekilen toplam akım değerinin 75 mA'i geçmemesi gerekmektedir.
 - Eğer giriş pinine uygulanan voltaj seviyesi 5V'dan büyük olacak ise seviye dönüştürücüsü kullanılması gerekmektedir.
 - İşlemci dahili pull-up ve pull-down dirençlerine sahiptir.

3.3 Sorular

1. Pull-up ve Pull-down dirençleri nedir ve neden kullanılır? Pull-up ve Pull-down bağlantısını çizerek gösteriniz.
2. Giriş ve çıkış pinlerinde seri bağlı direnç kullanımı neden gereklidir?
3. Butonlarda sıçrama (Bouncing) nedir ve neden olur? Sıçramayı engellemek için yazılımsal ve donanımsal olarak kullanılacak çözümlere örnek veriniz ve donanımsal çözümün bağlantılarını çizerek gösteriniz.

3.4 Uygulama Sırasında Kullanılacak C Fonksiyonları

`HAL_GPIO_Init(GPIO_TypeDef *GPIOx, GPIO_InitTypeDef *GPIO_Init)` fonksiyonu pin ayarlarının yapılması için kullanılır. Bu ayarlama CubeMX programı tarafından yapıldığı için, bizim kullanmamıza gerek kalmamaktadır.

`HAL_GPIO_ReadPin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin)` fonksiyonu, giriş olarak ayarladığımız pinde bulunan giriş lojik seviyesini okumak için kullanılır.

`HAL_GPIO_WritePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState)` fonksiyonu, çıkış olarak ayarlanan pinin set ya da reset edilmesi için kullanılır.

`HAL_GPIO_TogglePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin)` fonksiyonu, çıkış olarak ayarlanan pinin değerinin set ise reset, reset ise set edilmesini sağlar.

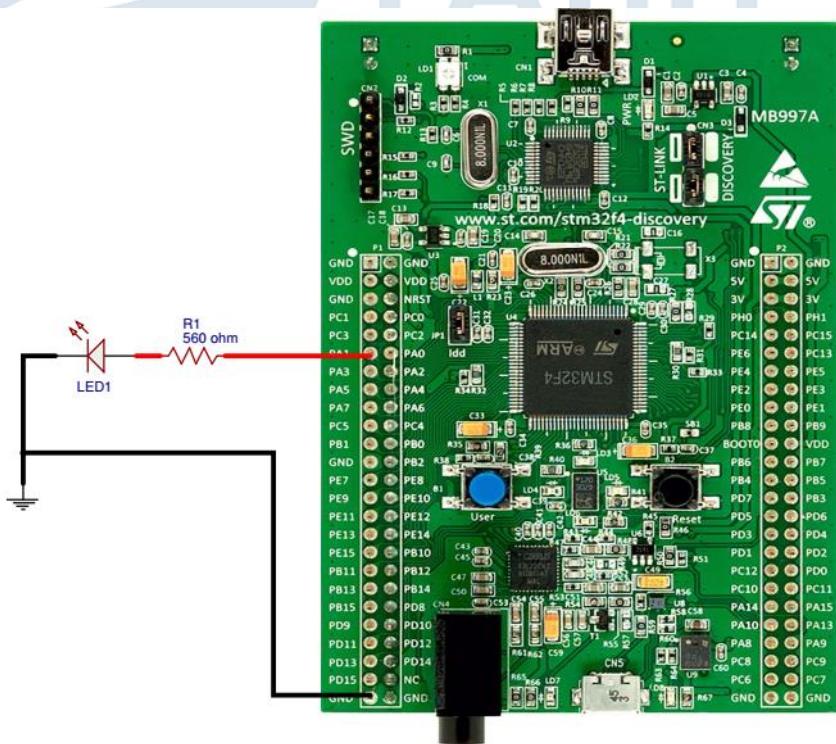
Örnek kullanım;

`GPIOx` hangi portun kullanıldığını ifade eder. Örneğin PA1 pinini kullanıyorsak, `GPIOx` yerine `GPIOA` yazarız. `GPIO_Pin` yerine de A portunun hangi bitini kullanacağımız onu yazmamız gerekmektedir. PA1 için, `GPIO_PIN_1` yazmamız gerekmektedir. Fonksiyonumuz düzenlenerek sonradan aşağıdaki gibi olmaktadır:

```
HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_1);
```

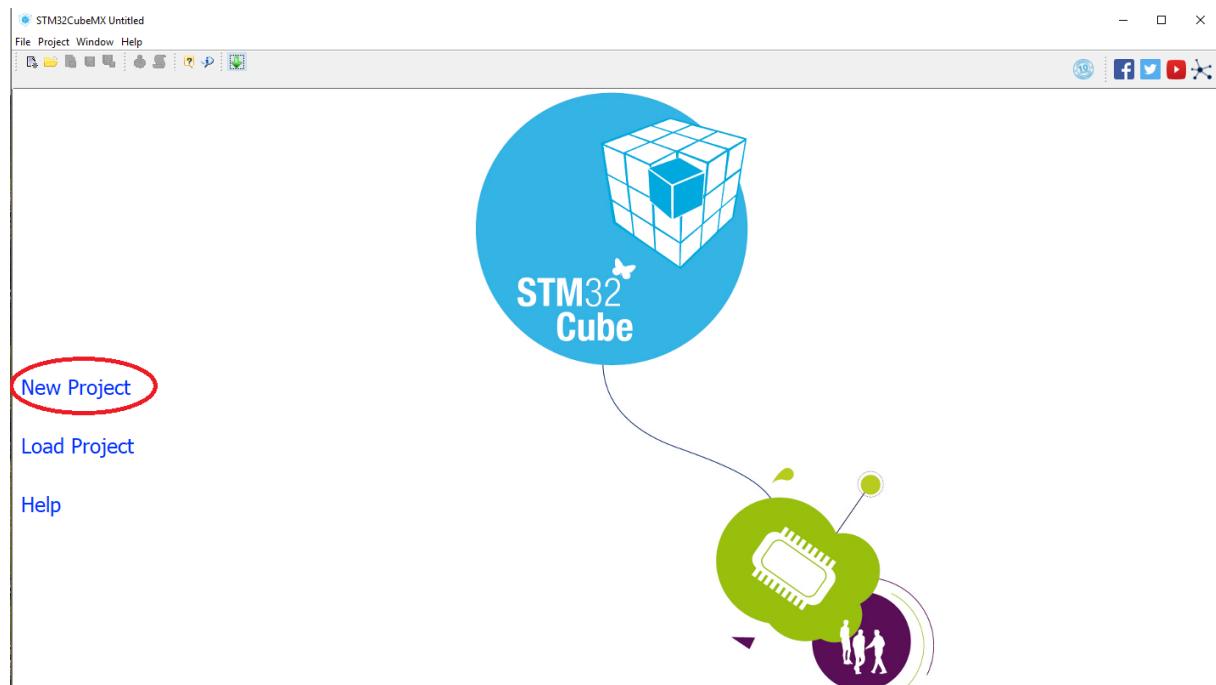
3.5 Devre Şeması

Deneyde oluşturulacak olan devre şeması Şekil 3-1' da görülmektedir, Şekil 6' ya uygun olarak devrenizin oluşturulması gerekmektedir.



Şekil 3-1 GPIO deneyi devre şeması

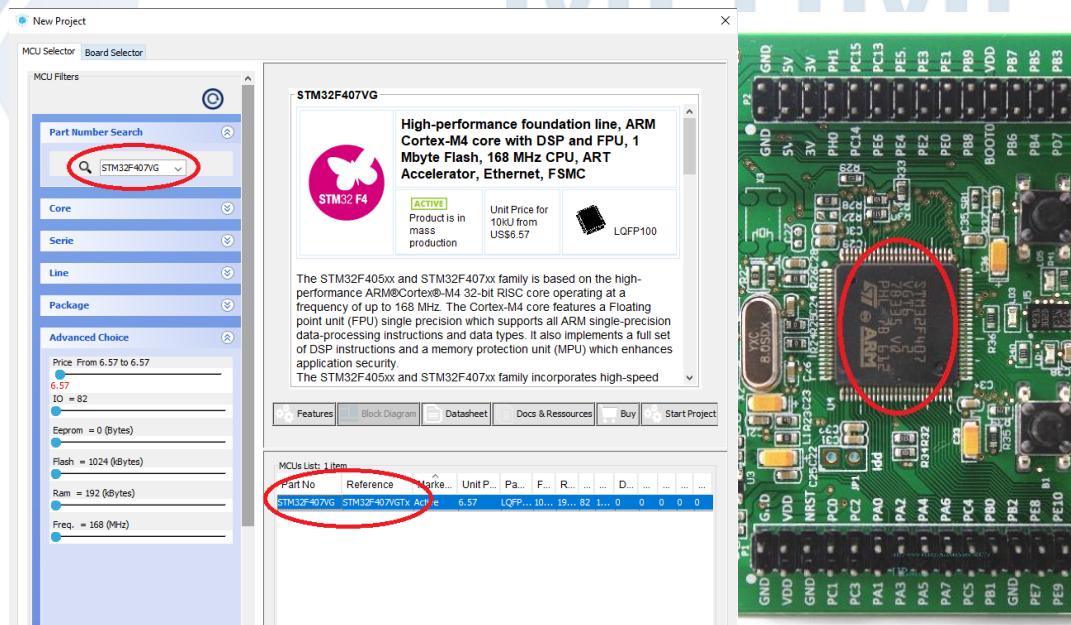
3.6 Deneyin Yapılışı



Şekil 3-2 CubeMX Programı Açılmış Sayfası

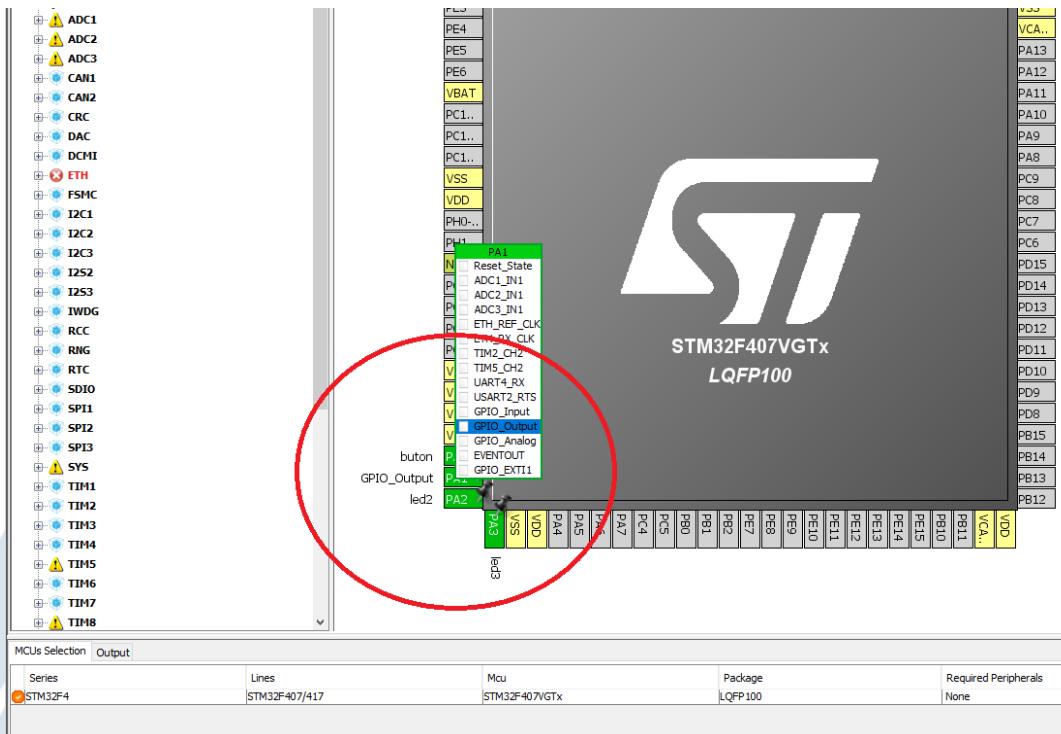
ADIM 1: Şekil 3-2’de, CubeMX programının açılış sayfası bizi karşılamaktadır. Kırmızı ile çevrili “New Project” butonu ile yeni proje oluşturuyoruz.

ADIM 2: Şekil 3-3’de karşımıza gelen pencere, projede kullanılacak işlemci seçimi için kullanılmaktadır. Discovery F4 kartında kullanılan işlemciyi seçiyoruz, “STM32F407VGTx” . Kullandığınız STM board üzerindeki işlemciye göre seçilecek model değişmektektir. İşlemci üzerindeki modeli kontrol ediniz.

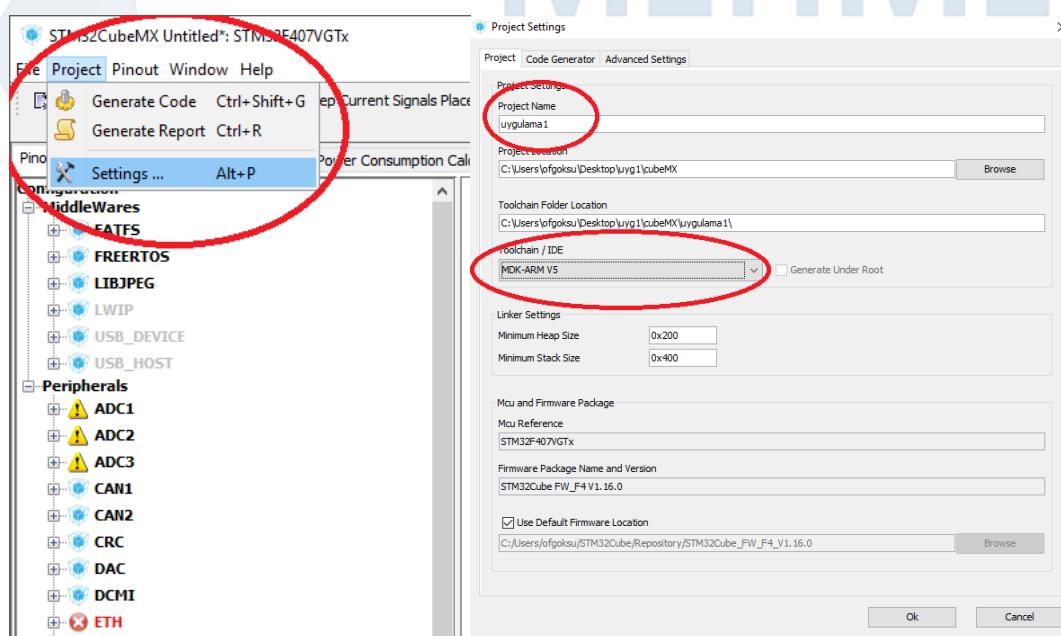


Şekil 3-3 CubeMX programlanacak işlemcinin seçimi

ADIM 3: Şekil 3-4’te oluşturulan projenin Pinout penceresi görülmektedir. Bu pencerede, işlemci pinlerinin ayarlanması görsel olarak yapılmaktadır. Uygulama için kırmızı ile gösterilen kısımda PA0 GPIO_Input, PA1,PA2,PA3 GPIO_Output olarak seçilerek ayarlanır. Ayarlanan pinlere etiket eklemek mümkündür. Bunun için, pin ayarlaması yapıldıktan sonra Mouse sağ tıklaması ile seçeneklerden “Enter User Label” seçeneği tıklanır.

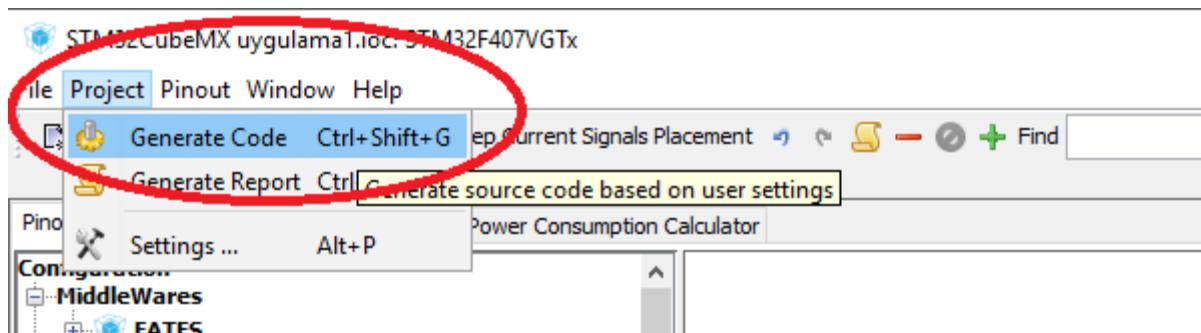


Şekil 3-4 Mikroişlemci Pin Konfigürasyon Penceresi



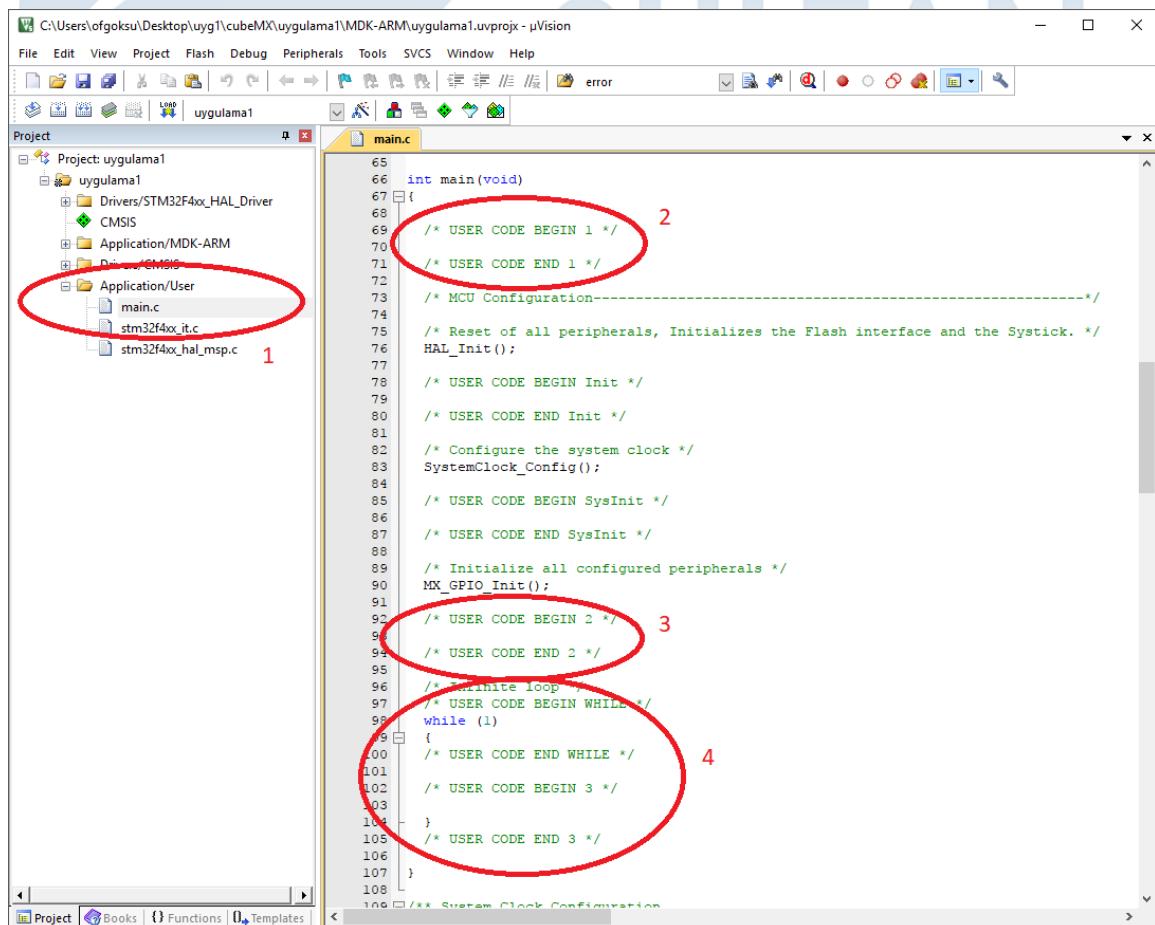
Şekil 3-5 Konfigürasyonu yapılan işlemci için Keil Proje dosyasının oluşturulması

ADIM 4: Şekil 3-5’te görülen pencerelerde, “Project” menüsünden “Settings” ile proje ayarları açılır. “Project Settings” penceresinde, “Project Name” projeye verilen ismi belirtir. Projenin nereye kaydedileceği “Project Location” da belirtilir. “Toolchain/IDE” seçeneği, kullanılacak IDE seçimi için kullanılır. Uygulama için MDK-ARM V5 kullanılmıştır.



Şekil 3-6 Projenin Kod dosyasının hazırlanması

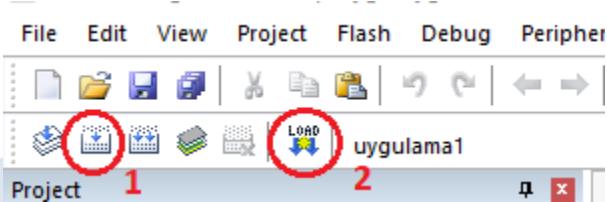
ADIM 5: Şekil 3-6’te görülen “Project” sekmesinden “Generate Code” ile ayarlanan işlemcimiz için MDK-ARM v5 için kod yazabileceğimiz yeni bir proje oluşturulur.



Şekil 3-7 Keil IDE’ si main.c dosyası

ADIM 6: Şekil 3-7'da, Keil IDE'sine ait pencere görülmektedir. 1 numaralı sekme, kodun yazılıacağı main.c dosyasının olduğu bölümü göstermektedir. Main.c dosyası içinde, main fonksiyonu 3 temel kısma ayrılmıştır. 2, 3 ve 4 numaralı alanlar kullanıcının kodunu yazması için ayrılmıştır. 2 numaralı alan, işlemciye ilk güç verildiğinde uygulanacak kodların yazılıacağı alandır. 3 numaralı alan, işlemci ayarlamaları tamamlandıktan sonra uygulanacak kodların, 4 numaralı alan da while döngüsünde yapılacak kodların olduğu alanı belirtir.

ADIM 7: Şekil 3-8'de kırmızı ile belirtilen 1. Buton kod derleme butonu, 2.buton derlenen kodu geliştirme kartına yükleme butonudur.



Şekil 3-8 Projelenmesi

Kod yazıldıktan sonra derleme işlemi yapılır ve “Build Output” penceresinde uyarı ve hata olup olmadığı kontrol edilir. Eğer hata ve uyarı varsa bunlar çözüldükten sonra derleme işlemi tekrar yapılır. Hata ve uyarı mesajı alınmadığı zaman geliştirme kartına yükleme işlemi yapılır.

A screenshot of the Keil IDE Build Output window. The window title is 'Build Output'. It shows the compilation process: 'compiling main.c...', 'compiling stm32f4xx_hal_msp.c...', 'linking...', and the final output: 'Program Size: Code=2680 RO-data=440 RW-data=8 ZI-data=1024 "uygulamal\uygulamal.axf" - 0 Error(s), 0 Warning(s). Build Time Elapsed: 00:00:13'. The entire output text is circled in red.

Şekil 3-9 Projelenme sonrası hata ekranı

Uygulama 1: Led yak/söndür

```
1. while (1)
2. {
3. /* USER CODE END WHILE */
4. /* USER CODE BEGIN 3 */
5.     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET);
6.     HAL_Delay(500);
7.     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET);
8.     HAL_Delay(500);
9. }
10. /* USER CODE END 3 */
```

Uygulama 2: Led yak/söndür - 2

```
1. while (1)
2. {
3. /* USER CODE END WHILE */
4. /* USER CODE BEGIN 3 */
5.     HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_1);
6.     HAL_Delay(500);
7. }
8. /* USER CODE END 3 */
```

Uygulama 3: Buton ile led

```
1. /* Infinite loop */
2. /* USER CODE BEGIN WHILE */
3. while (1)
4. {
5. /* USER CODE END WHILE */
6.
7. /* USER CODE BEGIN 3 */
8.     if( HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) ) {
9.         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET);
10.    }else{
11.        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET);
12.    }
13.
14. }
15. /* USER CODE END 3 */
```

3.7 Ödev

İstenen uygulama: 3 bit binary sayıcı.

Sayma işlemi geliştirme kartı üstünde bulunan buton ile sağlanmalıdır.

3 bitin gösterimi için 3 led kullanılmalıdır.

Bu ledler breadboard üzerine konulmalı, kartın üzerindeki ledler kullanılmamalıdır. Ledlere seri bağlı 560 ohm direnç bağlanmalıdır.

Ödev kontrolü laboratuvar dersi saatinde, ilk 15 dk içinde yapılacaktır.

Ödev teslimi yapmayanların deney notu 25 puan düşürülecektir.

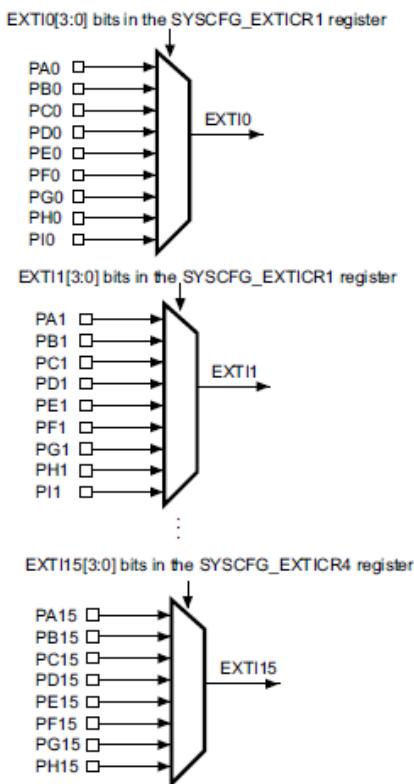
4.Hafta: External Interrupt (Harici-Buton Kesmesi)

4.1 Gerekli Malzemeler

- STM32 geliştirme kartı
- USB kablosu
- Push Button (Board Üstündeki User Button Kullanılacaktır)
- 4 adet LED (Board Üstündeki 4 adet LED kullanılacaktır)

4.2 Genel Bilgi

Interrupt (Kesme), yazdığımız kodun işleyişini aksatmadan, belirli zamanlarda gerçekleşen olayların takibini yapabilmemize imkan sağlayan bir olaydır. Örnek vermemiz gerekirse, eğer bir butonu döngü içinde okursak, eğer işlemcimiz başka bir kod parçasını işliyorsa buton okuma işlemini geciktirecektir. Eğer buton okuma işlemini kesme ile yaparsak, işlemci kesme işlemi gerçekleştiği zaman o an işlenen kod parçasını bekleterek buton okuma işlemi yapacaktır. Hızlı yapılması gereken işlemler kesme ile sorunsuz olarak gerçekleştirilebilmektedir.



STM32 serisi mikrodenetleyicilerde, GPIO pinleri kesme özelliğine sahiptir. Pinler dışında çevre birimleri de kesme özelliğine sahiptir. GPIO pinleri, 16 adet kesme hattına bağlanmıştır.

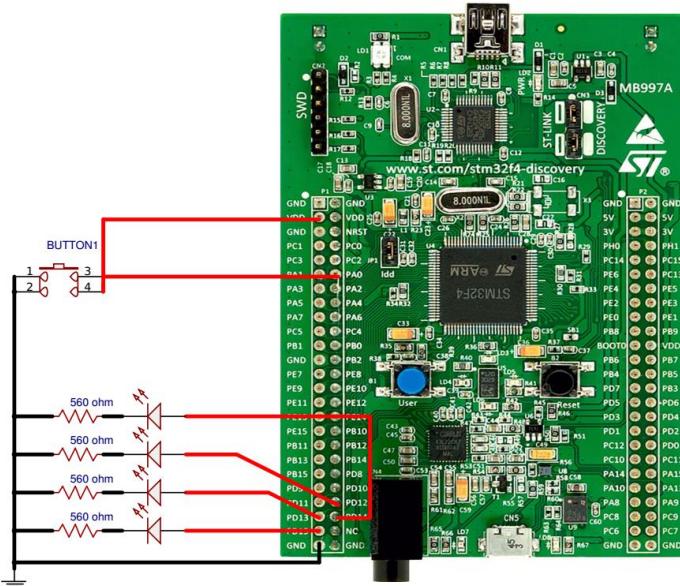
Şekil 1'de görüldüğü üzere, her pinin 0. Bitleri EXTI0 hattına, 1. Bitleri EXTI1 hattına bağlanmıştır. Bu her portun bitleri için geçerlidir. Yani PA0 ve PB0 pinlerinde kesme aktif edilmiş olsun, iki girişten de kesme geldiği zaman, mikrodenetleyicide aynı kesme rutinine gitmektedir. Bu yüzden eğer aynı hatta bağlı kesmeler kullanılıyorsa, kesme rutininde hangi pin üzerinden kesme olduğu kullanıcı tarafından belirtilmelidir.

4.3 Sorular

1. Interrupt kullanımı bize ne sağlar? Örnek veriniz?
2. Buton interrupt kullanımında sıçrama (bouncing) meydana gelir mi? Eğer böyle bir durum varsa engellemek için ne yapılabilir?
3. Yükselen kenar ve düşen kenar tetikleme nedir?

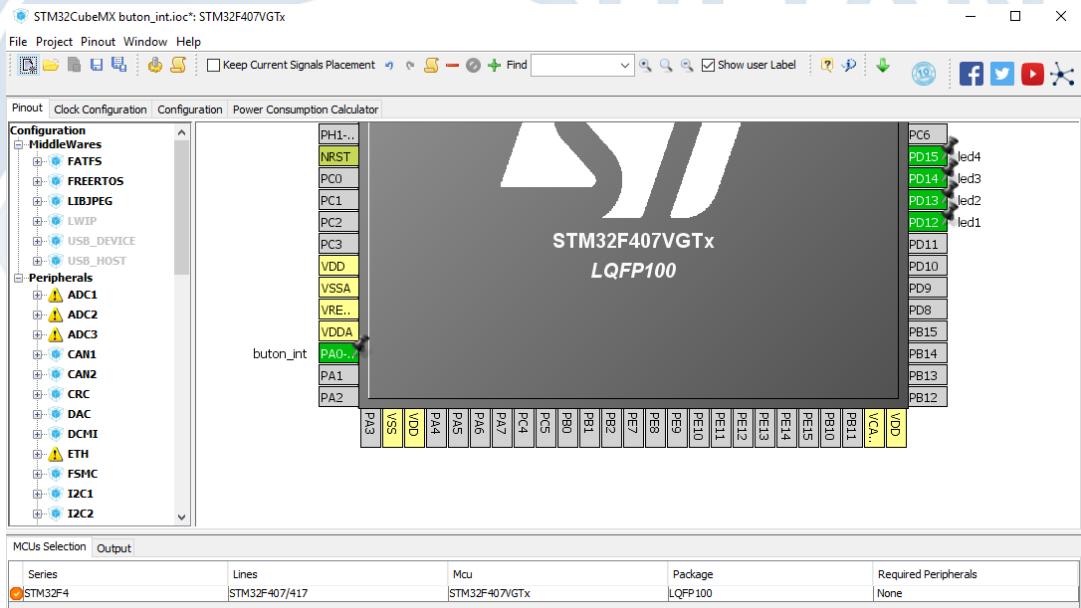
4.4 Devre Şeması

Deneyde oluşturulacak olan devre şeması Şekil 14' te görülmektedir, şekil 14' e uygun olarak devrenizin oluşturulması gerekmektedir.



Şekil 4-1 External Interrupt deneyi devre şeması

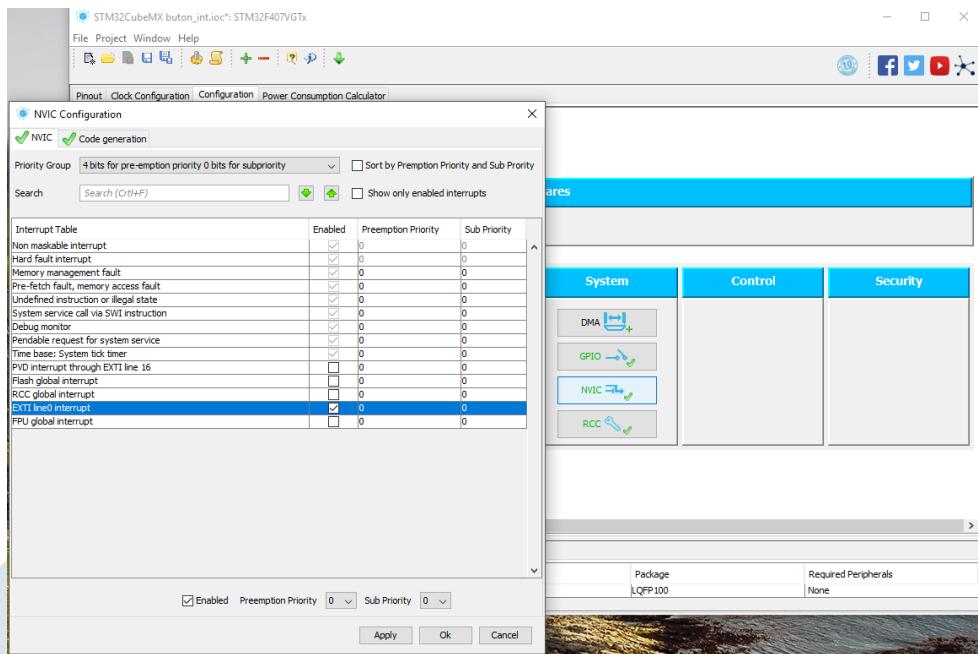
4.5 Deneyin Yapılışı



ADIM 1: CubeMX ile kullanılan geliştirme kartı/İşlemciye uygun yeni bir proje dosyası oluşturulur. Proje üzerinde istenilen ayarlamalar yapılmadan, proje kayıt edilmeli ve hangi IDE kullanılacaksa doğru şekilde seçilmelidir. Saat frekansı default halde bırakılabilir. PA0 pinı GPIO_EXTI0 olarak ayarlanır. Kolaylık olması açısından “button_int” olarak etiketlenmiştir. PD12, PD13, PD14, PD15 pinleri, Discovery F4 geliştirme kartında default olarak kartın

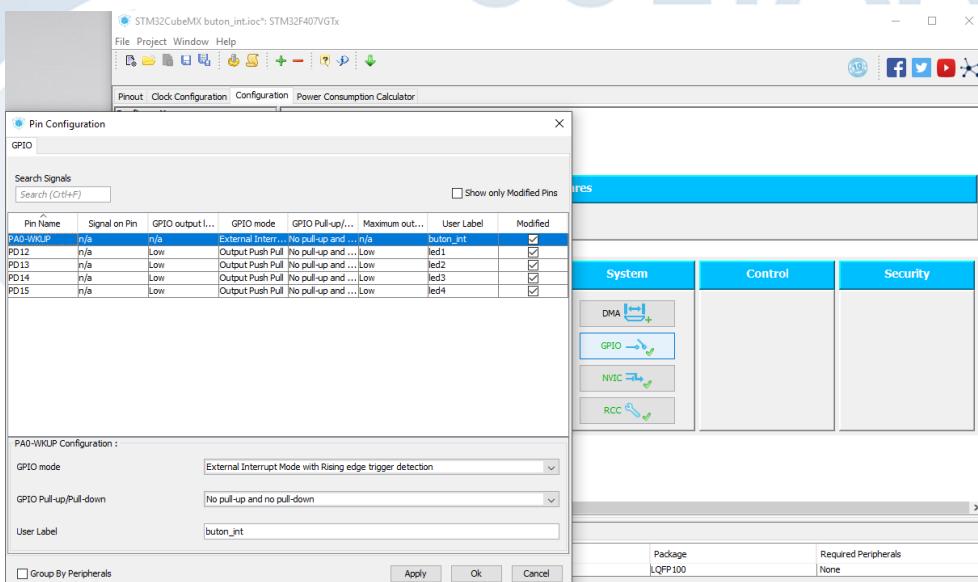
üzerinde bulunan ledlere bağlıdır. Bu pinler GPIO_Output olarak ayarlanmış ve “led1,2,3,4” olarak etiketlenmiştir.

ADIM 2:



Configuration sekmesinde, System parametrelerinden NVIC menüsü açılır ve EXTI line0 interrupt aktif edilir.

ADIM 3:



GPIO menüsünden, PA0 pinine ait ayarlar sekmesi açılır. GPIO mode ile, harici kesme olacağı ve kesme işleminin sinyalin düşen kenarında gerçekleşeceği belirlenmiştir. Dahili pull-up ya da pull-down direnci kullanılmayacağı belirtilemiştir. İstenirse bu menüde ayarlanabilir.

ADIM 4: Ayarlamalar bitirilir ve kod oluşturulur. Keil IDE açıldıktan sonra ilk derleme işlemi gerçekleştirilir ve herhangi bir hata olup olmadığı

kontrol edilir. Hata ve uyarı alınmadı ise, `stm32f4xx_it.c` dosyasından `"EXTI0_IRQHandler"` fonksiyonu `main.c` içerisindeki `User Code Begin 0` bölümüne taşınır. Bu işlem, oluşturulan değişkenlerin interrupt içinde kullanılabilmesini sağlamaktadır.

ADIM 5:

The screenshot shows the MDK-ARM IDE interface with the project 'button_int' open. The main window displays the `main.c` file, which contains the EXTI0_IRQHandler function. The code is annotated with comments indicating the flow from the EXTI0_IRQHandler entry point through the EXTI_Callback and HAL_GPIO_EXTI_Callback functions down to the EXTI0_IRQHandler function itself. The code also includes sections for SystemClock_Config, HAL_Init, and the main() function. The left sidebar shows the project structure, and the bottom pane shows the build output, which indicates a successful build with no errors or warnings.

```

56 void SystemClock_Config(void);
57 static void MX_GPIO_Init(void);
58 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin);
59 /* USER CODE BEGIN PPP */
60 /* Private function prototypes */
61
62 /* USER CODE END PPP */
63
64 /* USER CODE BEGIN 0 */
65
66 /**
67 * @brief This function handles EXTI line0 interrupt.
68 */
69 void EXTI0_IRQHandler(void)
70 {
71     /* USER CODE BEGIN EXTI0_IRQHandler 0 */
72     HAL_GPIO_TogglePin(GPIOD, led2_Pin|led3_Pin|led4_Pin);
73     /* USER CODE END EXTI0_IRQHandler 0 */
74     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
75     /* USER CODE BEGIN EXTI0_IRQHandler 1 */
76
77     /* USER CODE END EXTI0_IRQHandler 1 */
78 }
79
80 /* USER CODE END 0 */
81
82 int main(void)
83 {
84     /* USER CODE BEGIN 1 */
85
86     /* USER CODE END 1 */
87
88     /* MCU Configuration-----*/
89
90     /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
91     HAL_Init();
92
93     /* USER CODE BEGIN Init */
94
95     /* USER CODE END Init */
96
97     /* Configure the system clock */
98     SystemClock_Config();
99
100    /* USER CODE BEGIN SystemInit */
101
102    /* USER CODE END SystemInit */

```

Adım 4'te yapılan işlemler sonunda kod tekrar derlenerek hata ve uyarı olup olmadığı kontrol edilmelidir. Hata ve uyarılar varsa giderildikten sonra, aşağıda bulunan uygulama kodları gerçekleştirilecektir.

//Aşağıda bulunan değişken uygulamalarda kullanılacak durum değişkeni olacaktır.

```

1. /* Private variables ----- */
2.
3. /* USER CODE BEGIN PV */
4. /* Private variables ----- */
5.
6. int durum=0;
7.
8. /* USER CODE END PV */

```

Uygulama 4: Interrupt ile Led toggle

```
1. /* USER CODE BEGIN 0 */
2. void EXTI0_IRQHandler(void)
3. {
4.     /* USER CODE BEGIN EXTI0_IRQn 0 */
5.     HAL_GPIO_TogglePin(GPIOB, led1_Pin|led2_Pin|led3_Pin|led4_Pin);
6.     HAL_Delay(100);
7.     /* USER CODE END EXTI0_IRQn 0 */
8.     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
9.     /* USER CODE BEGIN EXTI0_IRQn 1 */
10.
11.    /* USER CODE END EXTI0_IRQn 1 */
12. }
13./* USER CODE END 0 */
```

Uygulama 4: Interrupt ile Kayan Işık

```
1. /* USER CODE BEGIN 0 */
2. void EXTI0_IRQHandler(void)
3. {
4.     /* USER CODE BEGIN EXTI0_IRQn 0 */
5.     durum++;
6.     if(durum>3) {
7.         durum=0;
8.     }
9.     ledYak(); //uygulama 2 alt fonksiyon
10.    HAL_Delay(100);
11.    /* USER CODE END EXTI0_IRQn 0 */
12.    HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
13.    /* USER CODE BEGIN EXTI0_IRQn 1 */
14.
15.    /* USER CODE END EXTI0_IRQn 1 */
16. }
17. //uygulama 2 alt fonksiyon
18. void ledYak() {
19.
20.     switch(durum) {
21.         case 0:
22.             HAL_GPIO_WritePin(GPIOB, led1_Pin, GPIO_PIN_SET);
23.             HAL_GPIO_WritePin(GPIOB, led2_Pin|led3_Pin|led4_Pin, GPIO_PIN_RESET);
24.         break;
25.
26.         case 1:
27.             HAL_GPIO_WritePin(GPIOB, led2_Pin, GPIO_PIN_SET);
28.             HAL_GPIO_WritePin(GPIOB, led1_Pin|led3_Pin|led4_Pin, GPIO_PIN_RESET);
29.         break;
30.     }
```

```

31.     case 2:
32.         HAL_GPIO_WritePin(GPIOD, led3_Pin, GPIO_PIN_SET);
33.         HAL_GPIO_WritePin(GPIOD, led1_Pin|led2_Pin|led4_Pin, GPIO_PIN_RESET);
34.     break;
35.
36.     case 3:
37.         HAL_GPIO_WritePin(GPIOD, led4_Pin, GPIO_PIN_SET);
38.         HAL_GPIO_WritePin(GPIOD, led1_Pin|led2_Pin|led3_Pin, GPIO_PIN_RESET);
39.     break;
40.     default:
41. HAL_GPIO_WritePin(GPIOD, led1_Pin|led2_Pin|led3_Pin|led4_Pin, GPIO_PIN_RESET);
42.     break;
43. }
44. }
45. //uygulama 2 alt fonksiyon bitis
46. /* USER CODE END 0 */

```

4.6 Ödev

İstenen uygulama: 7 segment led ekran ile 0-9 sayıcı

Sayma işlemi geliştirme kartı üzerinde bulunan buton ile sağlanmalıdır.

0'dan 9'a kadar rakamlar butona basıldıkça birer birer artacak, 9'dan sonra 0'dan tekrar sayacaktır. (0,1,2,.....,8,9,0,1,...)

- Buton harici interrupt kaynağı olarak kullanılacaktır.
- Rakamlar arası geçişte atlama olmamalıdır. Artış birer birer olmalıdır.

Ödev kontrolü Laboratuvar dersi saatinde, ilk 15 dk içinde yapılacaktır.

Ödev teslimi yapmayanların deney notu 25 puan düşürülecektir.

5.Hafta: Analog – Dijital Çevirici (ADC)

5.1 Gerekli Malzemeler

- STM32 geliştirme kartı
- USB kablosu
- 1 x Buton
- 1 x 560 ohm direnç
- Potansiyometre

5.2 Genel Bilgi

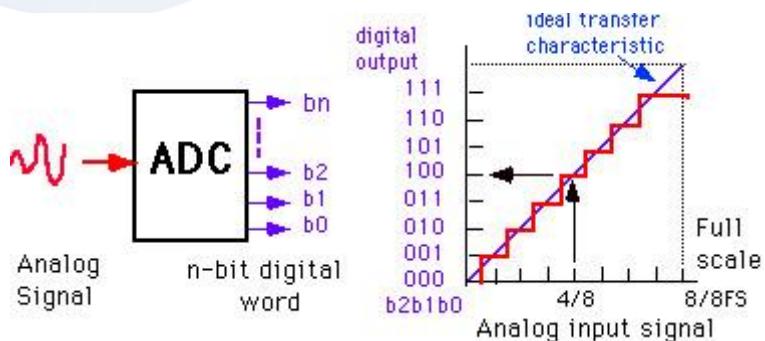
Mikrodenetleyicilerde ADC birimi, analog sinyallerin dijital sinyallere dönüştürülmesini sağlamaktadır. Dijital sistemlerde lojik 0 ve lojik 1 seviyeleri olmasına karşın, analog sinyaller teorik olarak sonsuz değerden oluşmaktadır. Bu değerleri mikrodenetleyicilerin anlayacağı şekilde dönüştürmek için kullanılan çevre birimlerinden biri de ADC birimidir. Şekil’ X te, örnek bir analog sinyalin ADC birimine input olarak verildikten sonra, output’tan alınan dijital sinyalin temsili gösterilmektedir.



Şekil 5-1 Analog sinyalin Dijital sinyale dönüştürülmesi

Mikrodenetleyicilerde bulunan ADC birimlerinin analog-dijital çevrim işlemi hassasiyet ile doğru orantılıdır. ADC biriminin çözünürlüğü(resolution) ne kadar fazla ise, analog sinyaldeki küçük değişimleri seviyelendirecektir.

Aşağıdaki şekilde Analog sinyalin digital sinyale dönüştürülmesi aşaması gösterilmektedir. 3 bit resolution’ a sahip olan bir ADC birimi için en büyük seviye step_size(111)(1 LSB) 7 değeri verildiği görülmektedir.



Şekil 5-2 ADC çevrim şematik gösterimi

ADC birimleri farklı çözünürlükte olabilmektedir. Bu çözünürlük, ADC ölçümünün hassasiyetini belirlemektedir.

- ❖ ADC için referans voltaj değerleri belirlenmelidir. Genel değerler, 3.3 V ve 5V olarak kullanıldığını düşünürsek, Referans voltaj değeri, ADC biriminin maksimum ölçüm yapabileceği analog sinyal sınırlarını belirlemektedir.
- ❖ Bit sayısı, referans voltaj değerini kaç parçaya böleceğimizi belirlemektedir. 3 bit ADC, referans voltaj değerini 8 eşit parçaya bölmektedir.
- ❖ ADC hassasiyeti, dijital değerde 1 birimlik (1 step size) değişme gerçekleşmesi için, analog sinyaldeki gereken değişim miktarıdır.

Örnek:

10 bit çözünürlüğe sahip bir ADC biriminin referans voltajı 3.3 V olsun;
ADC çözünürlük (resolution) = 10 bit, $2^{10}-1 = 1023$ alacağı maksimum değer,
 $1 \text{ LSB} = 3.3 \text{ V} / 1023 = 0,0032258064516129 \text{ V}$ hassasiyete sahiptir,
Yukarıdaki hesaplanan 1LSB(1 step size) değeri şu anlama gelmektedir, ADC birimi, input sinyalindeki her 0,00322 V' luk değişimi bir seviye(sayısal bir değer) ile anlamlandıracaktır.

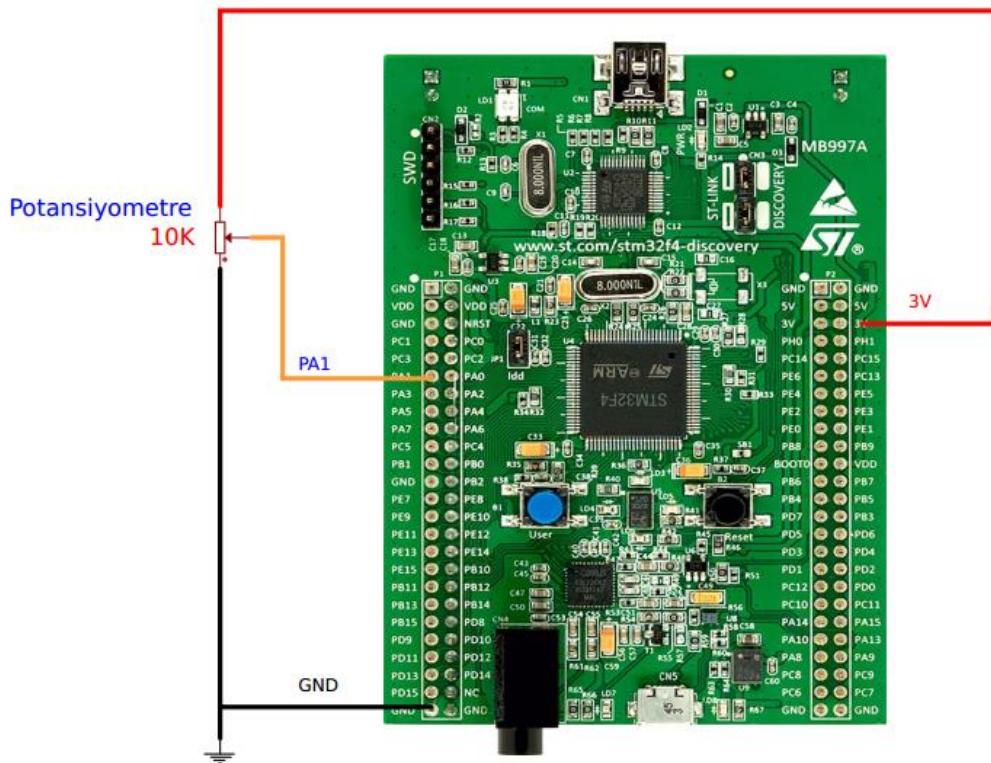
STM32 genel ADC özellikleri;

- ❖ Analog sinyal giriş aralığı = 0-3.3 V
- ❖ Referans Voltaj değeri = 2.4 V - 3.3 V
- ❖ DMA
- ❖ Interrupt

5.3 Sorular

1. Örnekleme frekansı, kuantalama seviyesi, çevrim zamanı terimleri nedir?
2. Örnekleme frekansı ile çevrim zamanı arasında nasıl bir ilişki kurulabilir? Yorumlayınız.
3. 0.75 mv hassasiyete sahip bir ADC için çözünürlük en az kaç bit olmalıdır? ADC referans voltaj değerini 3V alınız.

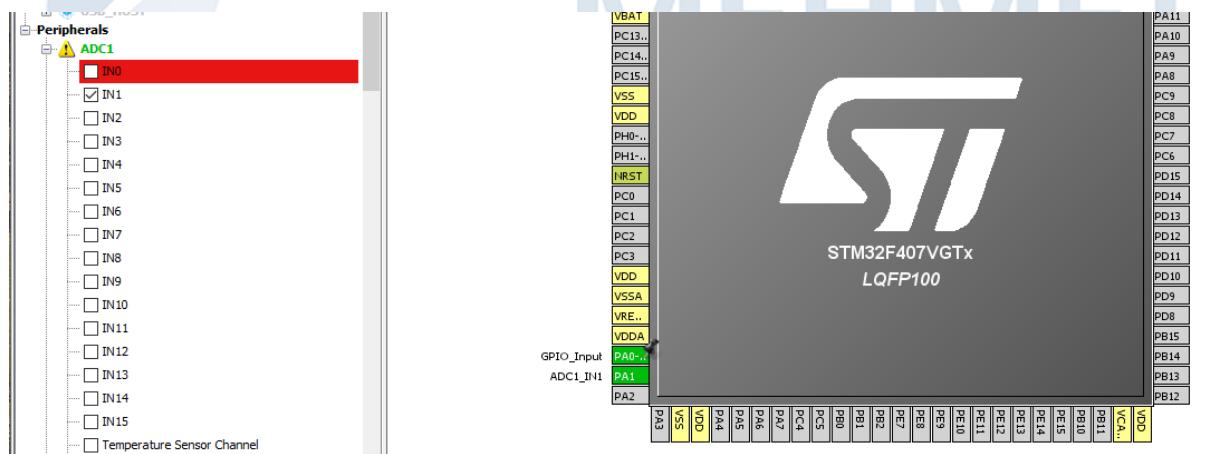
5.4 Devre Şeması



Şekil 5-3 ADC deneyi devre şeması

5.5 Deneyin Yapılışı

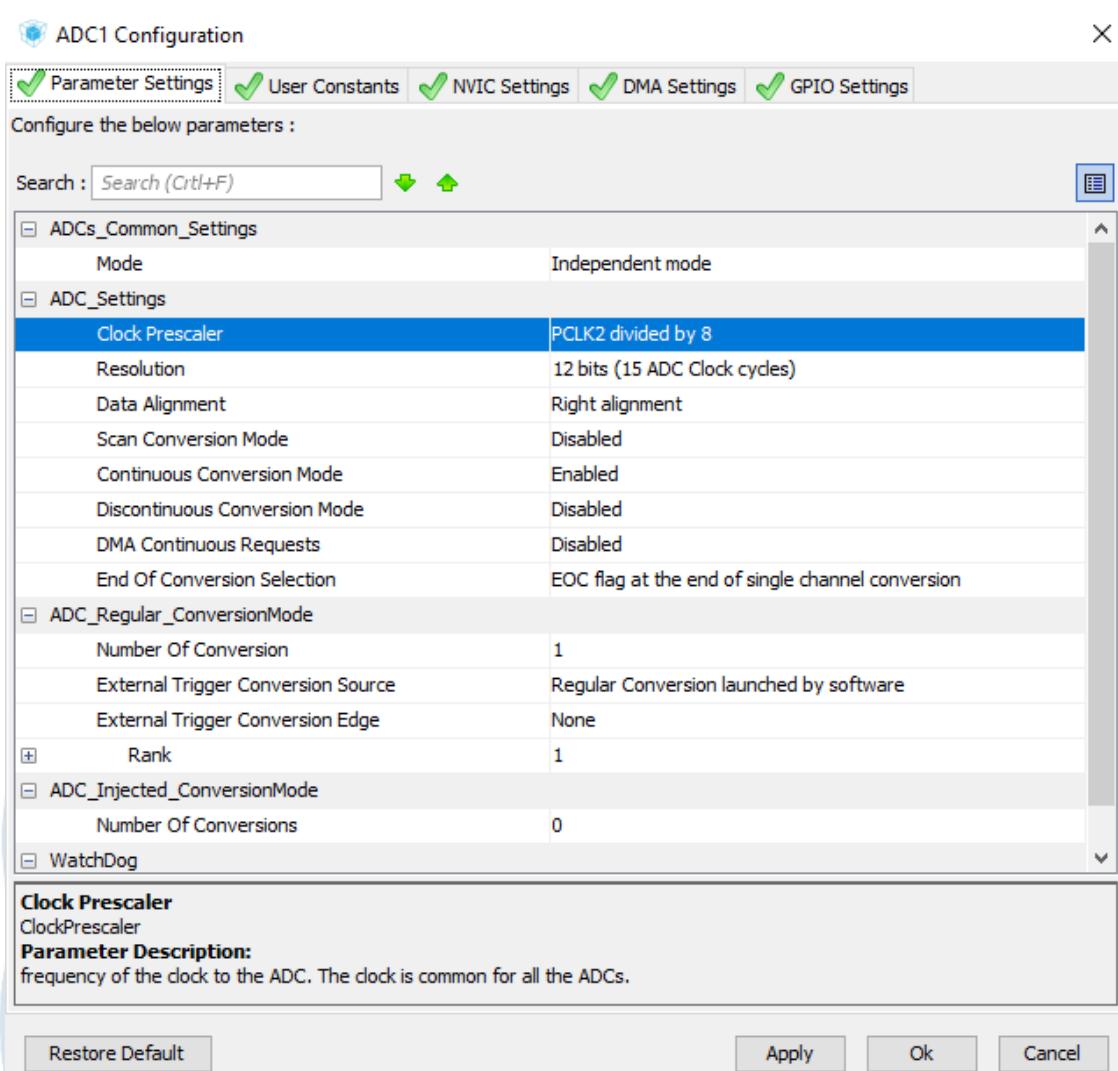
ADMİ 1:



Şekil 5-4 ADC deneyi için İşlemci Pin Konfigürasyonları

Yeni CubeMX projesi oluşturulur ve STM32F407VGTx mikrodenetleyicisi seçilir. PA0 pinı GPIO_input olarak ayarlanır. “Peripherals” altında ADC1 çevre biriminin “IN1” girişi aktif edilir. Otomatik olarak PA1 pinı ADC1 modülünün IN1 girişine bağlanacaktır.

ADMİ 2:



Sekil 5-5 ADC deneyi için ADC çevrim parametreleri ayar sayfası

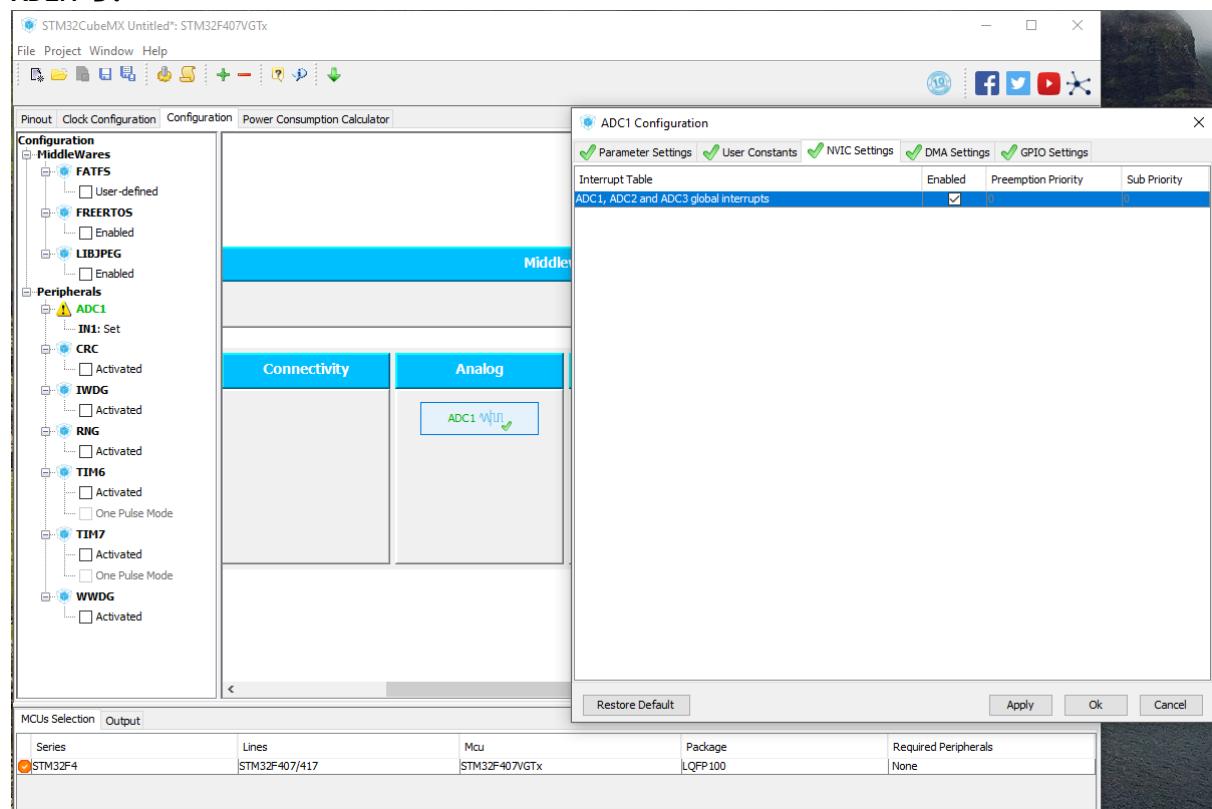
“Configuration” sekmesinden ADC1 modülünün ayar sayfası açılır. “Parameter Settings” sekmesinde ADC modülüne ait ayarlar şekildeki gibi düzenlenir.

Scan Conversion Mode – Birden çok ADC kanalı okunacak ise bu seçenek kanalların otomatik olarak değiştirilmesini sağlar.

Continuous Conversion Mode – ADC çevrimi tamamlandıktan sonra otomatik olarak çevrimin tekrar başlatılmasını sağlar.

End Of Conversion Selection – Interrupt’ın her kanal çevrimi bittiğinde ya da tüm kanalların çevrimi tamamlandıktan sonra oluşturulmasının seçimi yapılır.

ADIM 3:



Şekil 5-6 ADC deneyi için CubeMX ADC interrupt ayar sayfası

“NVIC Settings” sekmesinden “ADC1, ADC2 and ADC3 global interrupts” seçeneği işaretlenir ve ADC kesmesi aktif edilir.

ADIM 4: Ayarlamalar bitirildikten sonra kod derlenir ve proje açılır. “stm32f4xx_it” dosyası içinde yazılı olan “ADC_IRQHandler” fonksiyonu “main.c” dosyası içindeki “USER CODE BEGIN 0” bölgesine taşınır ve fonksiyon içindeki “USER CODE BEGIN ADC_IRQHandler 0” bölgesine çevrimi tamamlanan verinin okunmasını sağlamak için “adc_value=HAL_ADC_GetValue(&hadc1)” kodu eklenir.

```
1. /* USER CODE BEGIN 0 */
2. void ADC_IRQHandler(void)
3. {
4.     /* USER CODE BEGIN ADC_IRQHandler 0 */
5.     adc_value=HAL_ADC_GetValue(&hadc1);
6.     /* USER CODE END ADC_IRQHandler 0 */
7.     HAL_ADC_IRQHandler(&hadc1);
8.     /* USER CODE BEGIN ADC_IRQHandler 1 */
9.
10.    /* USER CODE END ADC_IRQHandler 1 */
11. }
12.
13. /* USER CODE END 0 */
```

“USER CODE BEGIN 2” bölgесine ADC modülünün kesme ile çalıştırılması için gerekli kodu yazarak analog-dijital çevrim başlatılır.

```
1. /* USER CODE BEGIN 2 */
2.     HAL_ADC_Start_IT(&hadc1);
3. /* USER CODE END 2 */
```

ADIM 5: İlk ADC çevrimi tamamlandıktan sonra, interrupt fonksiyonu içindeki kodlar çalıştırılır ve okunan değer “adc_value” değişkenine kaydedilir. ADC çevriminin tekrar başlatılması için “HAL_ADC_Start_IT(&hadc1);” kodunun tekrar çalıştırılması gerekmektedir.

Uygulama 1: while döngüsünde ADC çevrimi başlatmak

```
1. /* Infinite loop */
2. /* USER CODE BEGIN WHILE */
3. while (1)
4. {
5. /* USER CODE END WHILE */
6.
7. /* USER CODE BEGIN 3 */
8.         HAL_ADC_Start_IT(&hadc1);
9.         HAL_Delay(500);
10.    }
11. /* USER CODE END 3 */
```

“while” döngüsü içinde, her 500 ms zaman periyodunda ADC çevrimi başlatılır. Çevrim tamamlandıktan sonra, interrupt rutininde ADC datası adc_value değişkenine kayıt edilir.

Uygulama 2: Interrupt içinde ADC çevriminin başlatılması

```
1. void ADC_IRQHandler(void)
2. {
3. /* USER CODE BEGIN ADC IRQn 0 */
4.
5.     adc_value=HAL_ADC_GetValue(&hadc1);
6.     HAL_ADC_Start_IT(&hadc1);
7.
8. /* USER CODE END ADC IRQn 0 */
9. HAL_ADC_IRQHandler(&hadc1);
10. /* USER CODE BEGIN ADC IRQn 1 */
11.
12. /* USER CODE END ADC IRQn 1 */
13. }
```

“USER CODE BEGIN 2” bölgesinde başlatılan ADC çevrimi tamamlandıktan sonra, interrupt rutininde adc datası adc_value değişkenine kayıt edilir ve ADC çevrimi tekrar başlatılarak interrupt rutini tamamlanır. Her çevrim sonrası yeni çevrim otomatik olarak başlatılmaktadır.

5.6 Ödev

İstenen uygulama: Potansiyometreden okunan değere göre yanın Ledler.

Geliştirme kartı üzerinde bulunan kırmızı, turuncu, mavi ve yeşil ledlerin, potansiyometreden okunan değer 0 ile 255 arasında ise kırmızı ledin, 256 ile 511 arasında ise turuncu ledin, 512 ile 767 arasında ise mavi ledin, 767 den büyük ise yeşil ledin yanması istenmektedir.

ADC çözünürlüğü 10 bit olarak ayarlanması istenmektedir. ADC çevrim frekansının 5 Hz'den düşük olmaması gerekmektedir. Ledlerin bağlı olduğu port ve pin numaraları geliştirme kartının şematik dosyasında bulunmaktadır. Şematik dosyasını firmanın web sayfasında bulabilirsiniz.

Ödev kontrolü laboratuvar dersi saatinde, ilk 15 dk içinde yapılacaktır.

Ödev teslimi yapmayanların deney notu 25 puan düşürülecektir.

6.Hafta: Seri Haberleşme (Universal Synchronous Asynchronous Receiver Transmitter - USART)

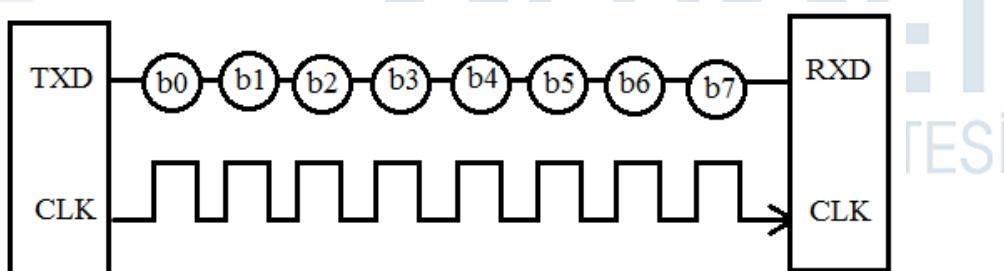
6.1 Gerekli Malzemeler

- STM32 geliştirme kartı
- USB kablosu
- 1 x Led
- 1 x 560 ohm direnç
- 1 x buton
- 3 x dişi-dişi jumper kablo

6.2 Genel Bilgi

Seri haberleşme, bir sistemin başka bir sistemle haberleşmesinde ilk akla gelen haberleşme yöntemlerinden biridir. Kolay kullanım ve bağlantıya sahip olması, hızlı uygulanabilmesi seri haberleşmeyi tercih sebebi olarak sunmaktadır. Seri veri iletimi, bir veri içindeki bitlerin aynı hat üzerinden ard arda gönderilmesi şeklinde gerçekleşir. Şekil 6-1'de örnek olarak bir terminalden başka bir terminalde seri veri gönderilmesi gösterilmiştir. Seri haberleşmede 8 bitlik paketler halinde ve ard arda tüm veri aktarılmaktadır.

Seri haberleşmeye Örnek olarak iki adet bluetooth cihazı arasında seri haberleşme ile veri aktarımı gerçekleşir.



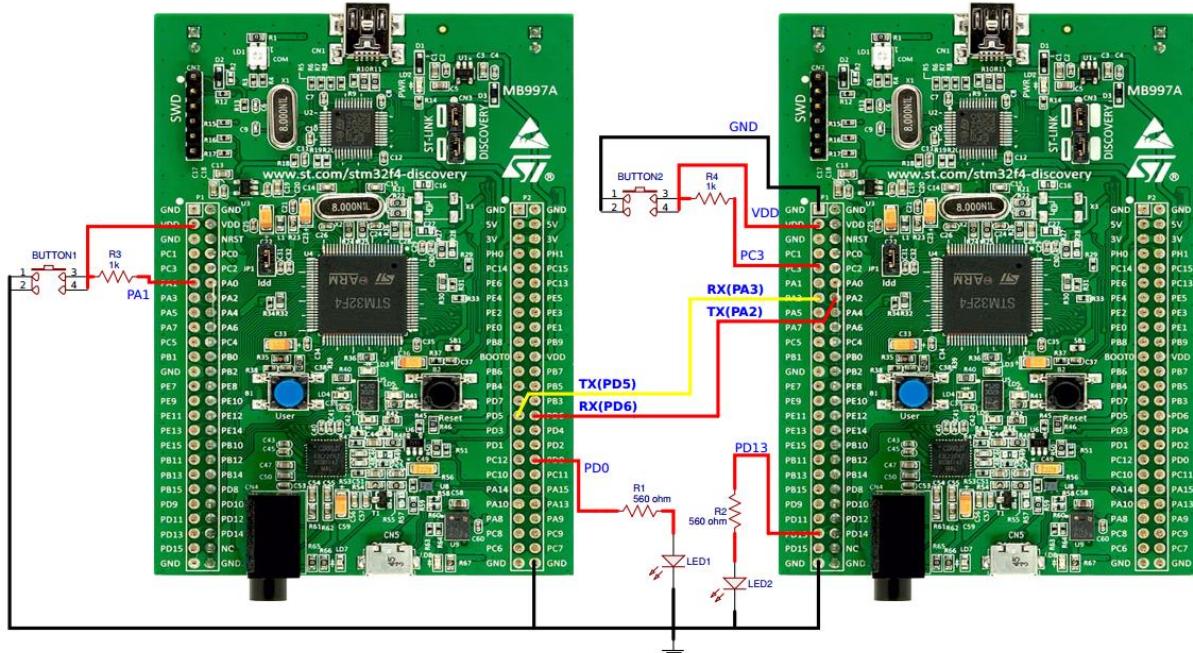
Şekil 6-1 Seri Haberleşme Temsili gösterimi

6.3 Sorular

1. Baudrate nedir? Baudrate oranının değiştirilmesi sonucu ne olmaktadır?
2. Senkron, asenkron seri haberleşme farkları nelerdir?
3. Full-duplex ve half-duplex haberleşme nasıl çalışır? Farkları nelerdir?
4. Uygulama 1 Buton ile veri yollama - alma için yazılan kodları inceleyiniz. Program akışını sözlü olarak anlatınız.

6.4 Devre Şeması

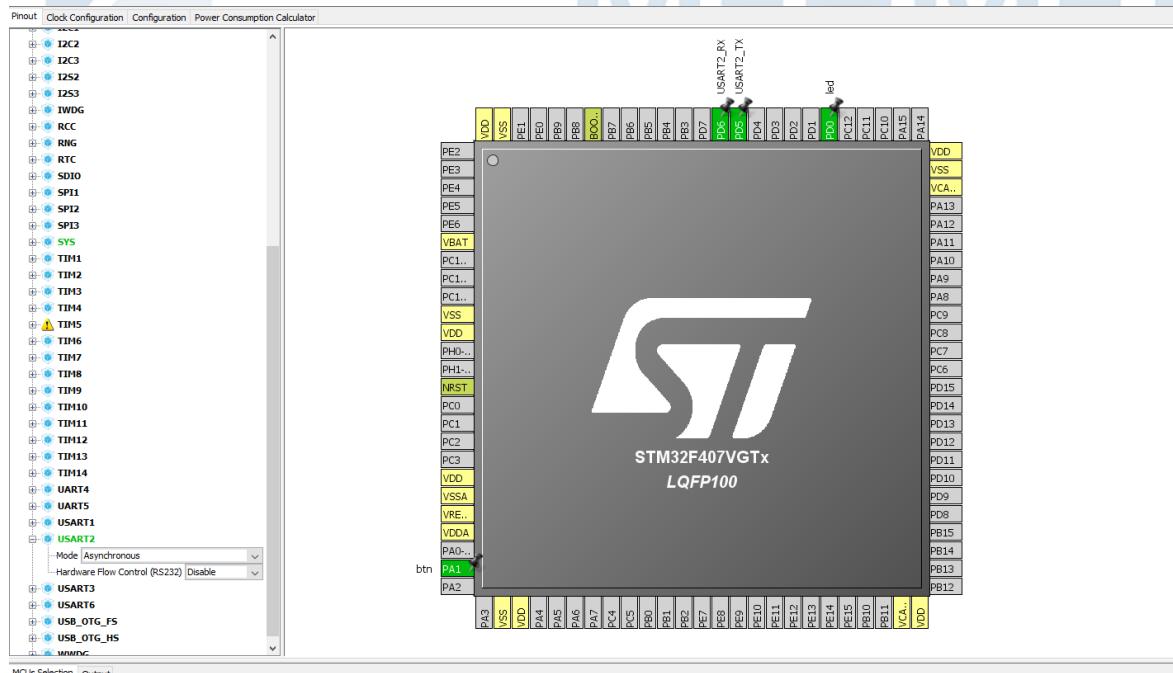
Deneyde oluşturulacak olan devre şeması Şekil 6-2' de görülmektedir, şekil 6'



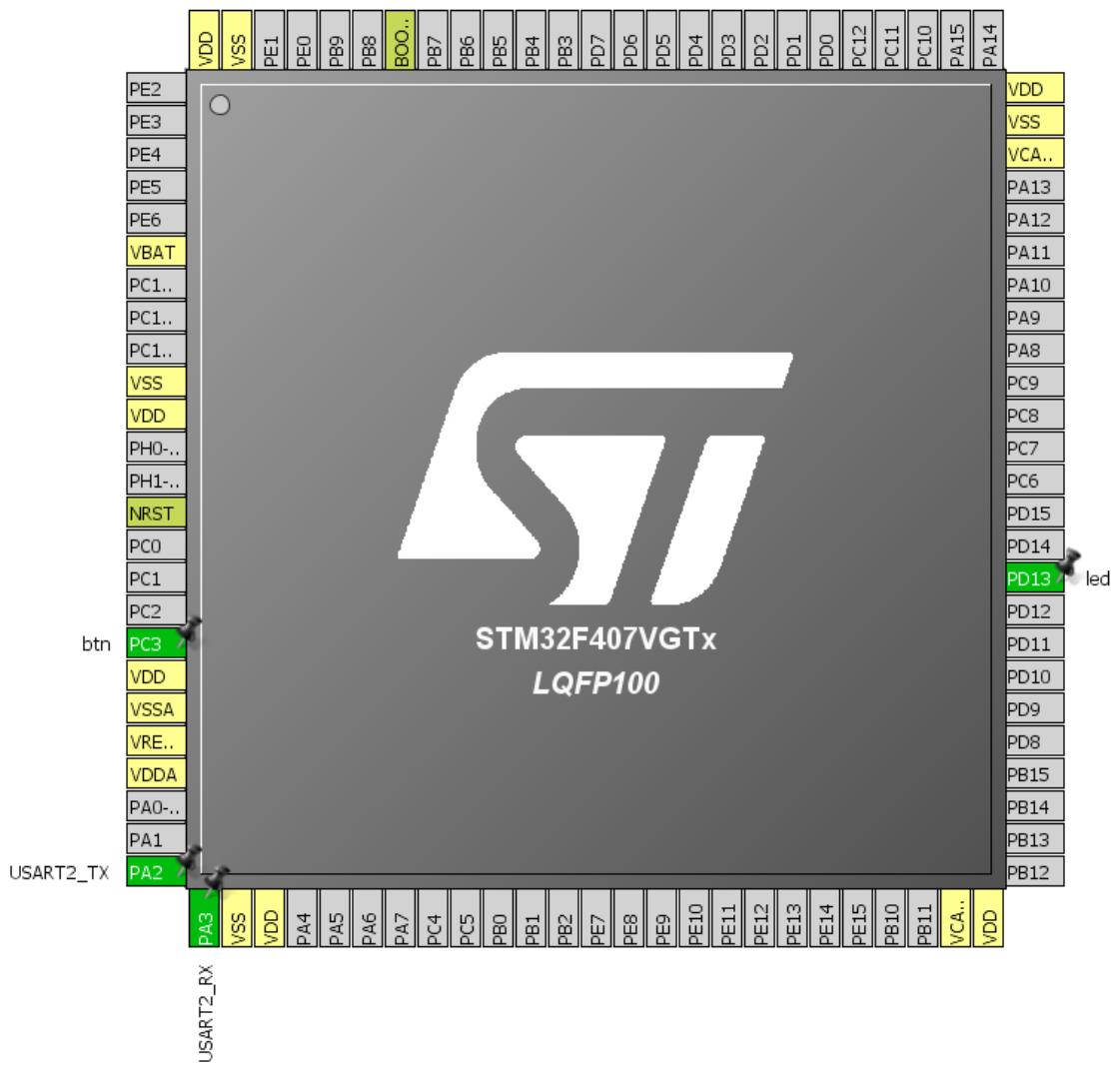
Şekil 6-2 USART deneyi devre şeması

6.5 Deneyin Yapılışı

ADIM 1: CubeMX ile yeni proje oluşturulur ve kullanılacak işlemcimiz (STM32F407VG) seçilir.



Şekil 6-3 Seri haberleşme deneyi işlemci pin konfigürasyon ekranı



Şekil 6-4 Seri Haberleşme için işlemci pin seçimi

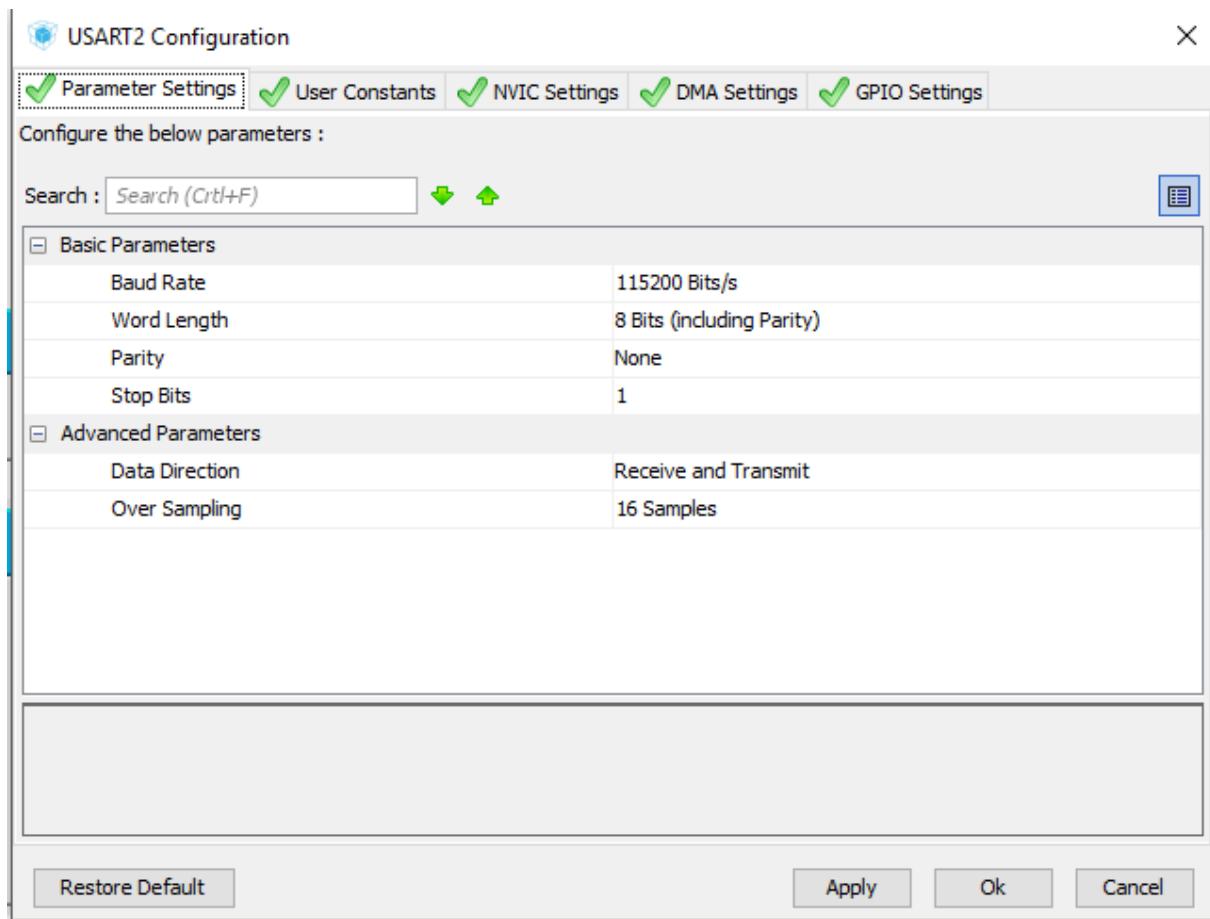
ADIM 2: USART2 birimi asynchronous modu seçilerek aktif edilir.

ADIM 3: Bu deneyde geliştirme kartının UART2 pinleri iki ayrı şekilde konfigüre edilecektir. Yanyana olan iki gruptan sol tarafta bulunanlar, UART2 pinlerini şekilmekle'de gösterildiği gibi (PD5,PD6), sağ tarafta bulunanlar da mekilşekilde'de gösterildiği gibi (PA2,PA3) ayarlayacaklardır.

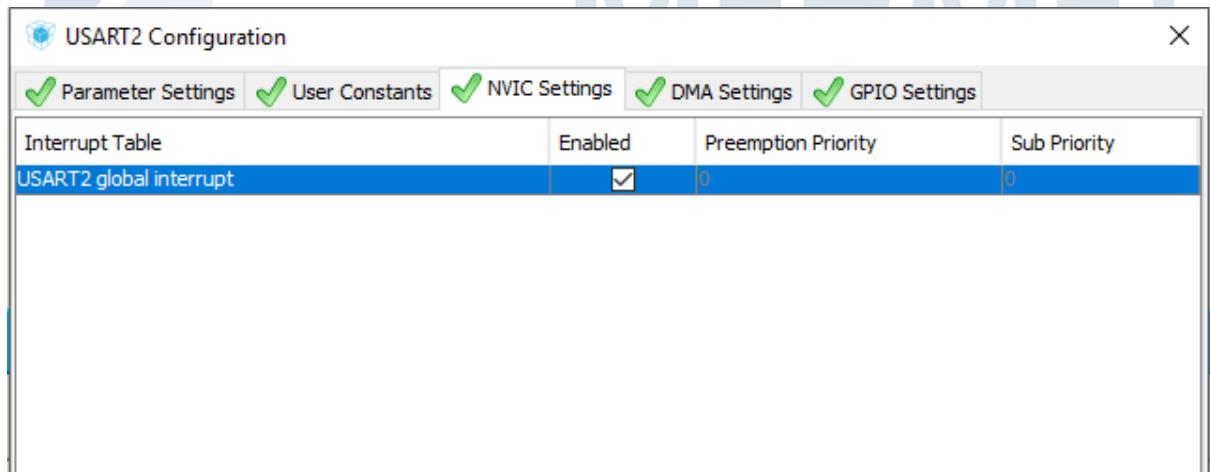
ADIM 4: USART ayarları Şekil 6-5'de görüldüğü gibi ayarlanacaktır.

-Baudrate 115200 Bits/s

-8bit/None parity/1 stop bit



Şekil 6-5 USART modülü parametre seçimi



Şekil 6-6 USART modülü interrupt aktif etme

ADIM 5: USART2 global interruptı aktif edilir, böylece data alma ve gönderme işlemleri interrupt ile gerçekleştirilir.

ADIM 6: Code Generate ile proje dosyası oluşturulur ve Keil IDE ortamına geçiş yapılır.

ADIM 7: Uygulama için kullanılacak değişkenler belirlenir. Uygulama 1 için, gönderilecek ve gelen datanın (verinin) saklanacağı değişkenler tanımlanmalıdır.

```
1. /* Private variables --*/
2.
3. uint8_t r_data;      //giden data
4. uint8_t t_data=0;    //gelen data
5.
6. /* USER CODE END PV */
```

ADIM 8: Kullanılan USART 2 birimi, lobal interrupt hattına sahiptir. USART 2 çevre biriminde aktif olan tüm interruptlar (veri yollama, veri alma, hata vb.) bu interrupt fonksiyonunu çalıştırır. Bu yüzden, interrupt fonksiyonu içerisinde veri alma fonksiyonu yazılır. Eğer interrupt veri alma sebebiyle oluştuysa, gelen verinin alınması sağlanır. Gelen verinin alınması, "HAL_UART_RxCpltCallback" fonksiyonunun çağrılmasına sebep olur. Gelen verinin nasıl kullanılacağı bu fonksiyon içerisinde yazılır.

Uygulama 1: Buton ile veri yollama - alma

```
1. /* USER CODE BEGIN 0 */
2.
3. void USART2_IRQHandler(void)
4. {
5.     /* USER CODE BEGIN USART2_IRQn_0 */
6.
7.     HAL_UART_Receive_IT(&huart2, &r_data, 1);
8.
9.     /* USER CODE END USART2_IRQn_0 */
10.    HAL_UART_IRQHandler(&huart2);
11.    /* USER CODE BEGIN USART2_IRQn_1 */
12.
13.    /* USER CODE END USART2_IRQn_1 */
14. }
15.
16. void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
17.
18.     t_data=r_data+1;
19.
20. }
21.
22. /* USER CODE END 0 */
```

Interrupt ve HAL_UART_RxCpltCallback fonksiyonları hazırlandıktan sonra, while döngüsü içinde, butona basıldığı zaman t_data içindeki veriyi yollayacak kod yazılacaktır.

```
1. /* USER CODE BEGIN 3 */
2. if( HAL_GPIO_ReadPin(btn_GPIO_Port, btn_Pin) ){
3.     HAL_UART_Transmit_IT( &huart2, &t_data, 1 );
4.     HAL_GPIO_WritePin(led_GPIO_Port, led_Pin, GPIO_PIN_SET);
5.     HAL_Delay(250);
6.     HAL_GPIO_WritePin(led_GPIO_Port, led_Pin, GPIO_PIN_RESET);
7. }
```

Uygulama 2: Buton ile uzaktaki modülün kontrolü

Uygulama 2, karta bağlı olan ledin kontrolünün başka bir karta bağlı olan buton ile seri haberleşme üzerinden yapılması amaçlanmıştır.

Veri alma ve led kontrolü

```
1. void USART2_IRQHandler(void)
2. {
3.     /* USER CODE BEGIN USART2_IRQn_0 */
4.
5.     HAL_UART_Receive_IT(&huart2, &r_data, 1);
6.
7.     /* USER CODE END USART2_IRQn_0 */
8.     HAL_UART_IRQHandler(&huart2);
9.     /* USER CODE BEGIN USART2_IRQn_1 */
10.
11.    /* USER CODE END USART2_IRQn_1 */
12. }
13.
14. void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
15.
16.     HAL_GPIO_WritePin(led_GPIO_Port, led_Pin, (uint8_t ) r_data);
17.
18. }
```

Buton bilgisinin seri haberleşme ile yollanması

```
1. /* USER CODE BEGIN WHILE */
2. while (1)
3. {
4.     /* USER CODE END WHILE */
5.     /* USER CODE BEGIN 3 */
6.     if( HAL_GPIO_ReadPin(btn_GPIO_Port, btn_Pin) ){
7.         t_data=1;
8.     }else{
9.         t_data=0;
```

```
10.     }
11.     HAL_UART_Transmit_IT( &huart2, &t_data, 1 );
12.     HAL_Delay(150);
13. }
14. /* USER CODE END 3 */
15. }
16.
```

6.6 Ödev

İstenen uygulama: Seri haberleşme ile led kontrolü.

Geliştirme kartı üzerinde bulunan kırmızı, turuncu, mavi ve yeşil ledlerin, seri haberleşmeden gelen veriye göre yanması istenmektedir. Gelen değer 0 ile 255 arasında ise kırmızı ledin, 256 ile 511 arasında ise turuncu ledin, 512 ile 767 arasında ise mavi ledin, 767 ile 1000 arasında ise yeşil ledin yanması istenmektedir. Acil durum olarak gelen veri 1000'den büyük ise, 4 led birlikte blink (yanıp sönme) yapacaktır. Bu blink göz ile farkedilmesi gerekmektedir.

Verinin, potansiyometreden okunan değerden alınarak seri haberleşme ile yollandası gerekmektedir.

ADC çözünürlüğü 10 bit olarak ayarlanması istenmektedir. ADC çevrim frekansının 5 Hz'den düşük olmaması gerekmektedir. Ledlerin bağlı olduğu port ve pin numaraları geliştirme kartının şematik dosyasında bulunmaktadır. Şematik dosyasını firmanın web sayfasında bulabilirsiniz.

Ödev kontrolü laboratuvar dersi saatinde, ilk 15 dk içinde yapılacaktır.

Ödev teslimi yapmayanların deney notu 25 puan düşürülecektir.

7.Hafta: Zaman Kesmesi (Timer Interrupt)

7.1 Gerekli Malzemeler

- STM32 geliştirme kartı
- USB kablosu
- 2 x Led
- 2 x 560 ohm direnç

7.2 Genel Bilgi

Timer yani zamanlayıcı, bir mikroişlemcinin donanımsal olarak barındırdığı birimlerinden bir tanesidir. Temel olarak bir zamanlayıcıyı bir sayıcı olarak kullanabiliriz. Bir sayıcı belirli bir sayıdan aşağı birer birer düşer ya da istenilen bir değere kadar artırabiliriz. Timer'lar sayma işlemini clock hızına göre yapabildikleri gibi dışarıdan verilen harici bir sinyale göre de yapabilirler.

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/compare channels	Complementary output	Max interface clock (MHz)	Max timer clock (MHz)
Advanced-control	TIM1, TIM8	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	84	168
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	84	168
	TIM10, TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	84	168
	TIM12	16-bit	Up	Any integer between 1 and 65536	No	2	No	42	84
Basic	TIM13, TIM14	16-bit	Up	Any integer between 1 and 65536	No	1	No	42	84
	TIM6, TIM7	16-bit	Up	Any integer between 1 and 65536	Yes	0	No	42	84

Şekil 7-1 STM32F4xx Serisi Mikrodenetleyilerin Timer özellikleri

Bir Timer yüklenen değere ulaştığında (up veya down counter) bir zaman kesmesi (interrupt) üretirler ve interrupt kod bloğundaki programı çalıştırırlar. STM32F4xx serisi mikrodenetleyicilerde yaklaşık 14 adet Timer

bulunmaktadır. Şekil 7-1'de STM32F4 serisi mikrodenetleyicilerde bulunan Timer'lar ve genel özelliklerini bulunmaktadır. Bu özelliklerden;

Counter Resolution = Zamanlayıcının sayacı的最大値を表す。Counter Resolution = Zamanlayıcının sayacı的最大値を表す。

Counter Type = Zamanlayıcının artarak ya da azalarak sayacığını temsil eder.

Prescaler Factor = Zamanlayıcının kaç saat durbesinde bir artış göstereceğini temsil eder.

DMA Request Gener. = Timer çevre biriminin, DMA modülü ile çalışıp çalışamayacağını gösterir.

Capture/Compare Ch. = Timer çevre biriminin, çıkış portlarına direkt bağlı olabilecek ve bazı fonksiyonlar için kullanılabilen kanal sayısını belirtir (PWM, Encoder gibi).

Complementary outs. = Alternatif çıkış oalrak ayarlanan pinlerin çıkış polaritesini ters çevirme fonksiyonudur (Aktif 1 ya da aktif 0 olarak ayarlanması).

Counter Settings
Prescaler (PSC - 16 bits value)
Counter Mode
Counter Period (AutoReload Register - 32 bits value)
Internal Clock Division (CKD)

Şekil 7-2 Timer Birimi Değişkenler Tablosu

Timer çevre biriminin istenilen şekilde kullanılması için, ayarlanabilir değişkenlere sahiptir. Bu özelliklerden temel olanları Şekil 7-2'de verilmiştir.

Prescaler = Timer birimine giren saat kaynağının istenilen sayıya bölünmesini sağlar. Örnek olarak saat frekansı 16 Mhz, prescaler değeri de 8 olsun, timer çevre biriminin sayma frekansı 2 MHz olacaktır.

Counter Mode = Timer çalışırken yukarı, aşağı, yukarı-aşağı modlarından hangisini kullanacağını seçmeyi sağlar.

Counter Period = Timer biriminin kaça kadar sayacığını belirler. Cnt = 10 ise, 10dan sonra counter değeri 0'a dönecektir.

Internal Clk Div = Prescaler ile aynı fonksiyona sahiptir.

Bu değişkenler ile, timer biriminin çalışma frekansı ayarlanabilir,

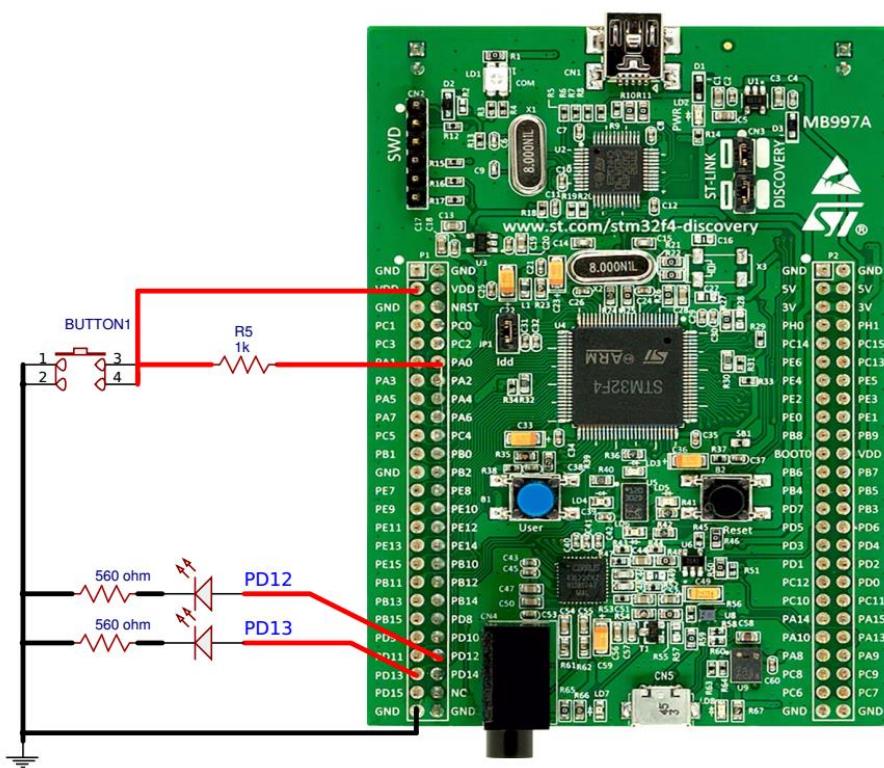
$$Freq = Clk_Source_Freq / (PSC * (Counter_Per + 1))$$

7.3 Sorular

1. Timer çevre birimini neler için kullanabiliriz? 2 örnek veriniz.
2. Verdiğiniz örneklerden birinin çalışma mantığını açıklayınız.
3. 16 MHz saat frekansına sahip Timer çevre biriminin 1 ve 2 saniye periyotlu interrupt sinyali oluşturma için gereken "counter" ve "prescaler" değerleri ne olmalıdır? Hesaplayınız.
4. 16 MHz saat frekansına sahip Timer çevre biriminin 100 Hz frekanslı interrupt sinyali oluşturma için gereken "counter" ve "prescaler" değerleri ne olmalıdır? Hesaplayınız.

7.4 Devre Şeması

Deneyde oluşturulacak olan devre şeması Şekil 3-1' da görülmektedir, şekil



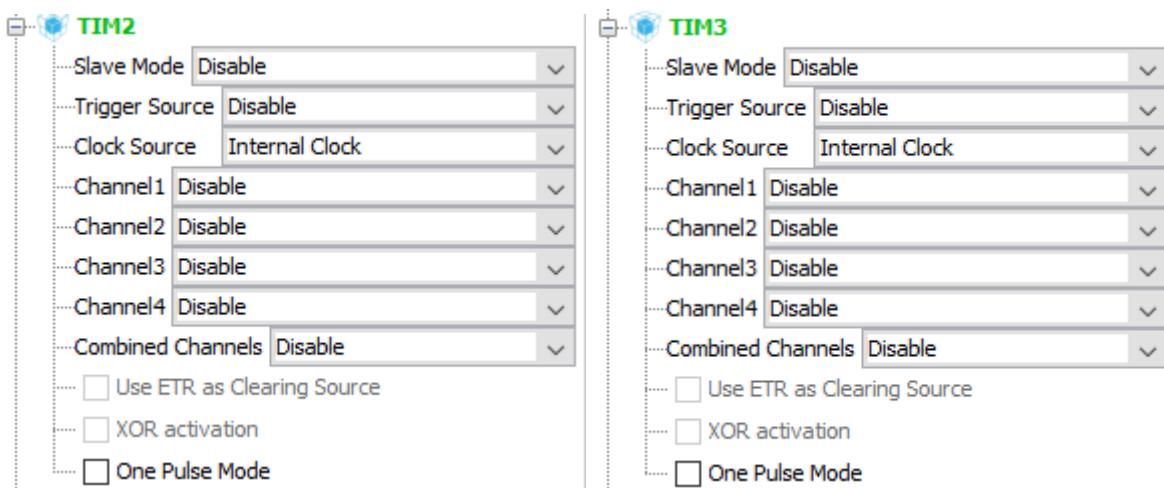
Şekil 7-3 Timer Interrupt Devre Şemeası

7.5 Deneyin Yapılışı

ADIM 1: STM32CubeMX ile yeni proje oluşturulur ve PD12 ile PD13 çıkış olarak ayarlanır. "led1" ve "led2" olarak etiket ismi verilir.

ADIM 2: TIM2 ve TIM3 birimleri Pinout sekmesinden Şekil 7-4'deki gibi aktif edilir.

ADIM 2: İşlemci saat frekansı 16 MHz olarak ayarlanır ve Timer 2 ile Timer 3 ayarları Configuration sekmesindeki ayar sekmesinden Timer 2 2 sn, Timer 3 1 sn periyoda sahip olacak şekilde ayarlanır. Değişkenler Soru 3'de hesaplanan değerler ile ayarlanır.



Şekil 7-4 Timer Deneyi Parametre ayarları

ADIM 3: Ayarlar tamamlandıktan sonra Keil v5 için gerekli kod oluşturularak proje dosyası açılır. Interrupt kodları “main.c” içine taşınır.

ADIM 4: Timer birimlerinin interrupt üretebilmesi için, `HAL_TIM_Base_Start_IT(&htim)` fonksiyonunun çalıştırılması gerekmektedir. Bu kod timer biriminin ve interrupt’ın aktif edilmesini sağlar.

```

1. /* USER CODE BEGIN 2 */
2. HAL_TIM_Base_Start_IT(&htim2);
3. HAL_TIM_Base_Start_IT(&htim3);
4. /* USER CODE END 2 */

```

ADIM 5: Interrupt fonksiyonları içine, ledleri toggle eden kodu yazınız. Her interrupt bir ledi kontrol edecektr. *Ledlerin toggle olmasını sağlayan kodu istediğiniz şekilde yazabilirsiniz.*

7.6 Ödev

İstenen uygulama: Timer ile 3 bit binary sayıcı.

3 bitin gösterimi için 3 led kullanılmalıdır.

Bu ledler breadboard üzerine konulmalı, kartın üzerindeki ledler kullanılmamalıdır. Ledlere seri bağlı 560 ohm direnç bağlanmalıdır.

1,5 sn periyod ile sayma işlemi yapan bir uygulama yazınız.

Ödev kontrolü Laboratuvar dersi saatinde, ilk 15 dk içinde yapılacaktır.

Ödev teslimi yapmayanların deney notu 25 puan düşürülecektir.

8.Hafta: Pulse Width Modulation (PWM) with ADC

8.1 Gerekli Malzemeler

- STM32 geliştirme kartı
- USB kablosu
- 1 x Servo motor
- 1 x Osiloskop
- 1 x 10 K ohm potansiyometre

8.2 Genel Bilgi

PWM (Pulse Width Modulation - Sinyal genişlik modülasyonu) dijital modülasyon tekniklerinden biridir. Modülasyon tekniği, analog sinyal üretilmesi yerine lojik-0 ve lojik-1 sinyallerinin zamanlaması ile gerçekleştirilir. Bu modülasyon tekniği ile oluşturulan sinyalin özellikleri, sinyalin periyodu ve doluluk oranı ile belirlenir.

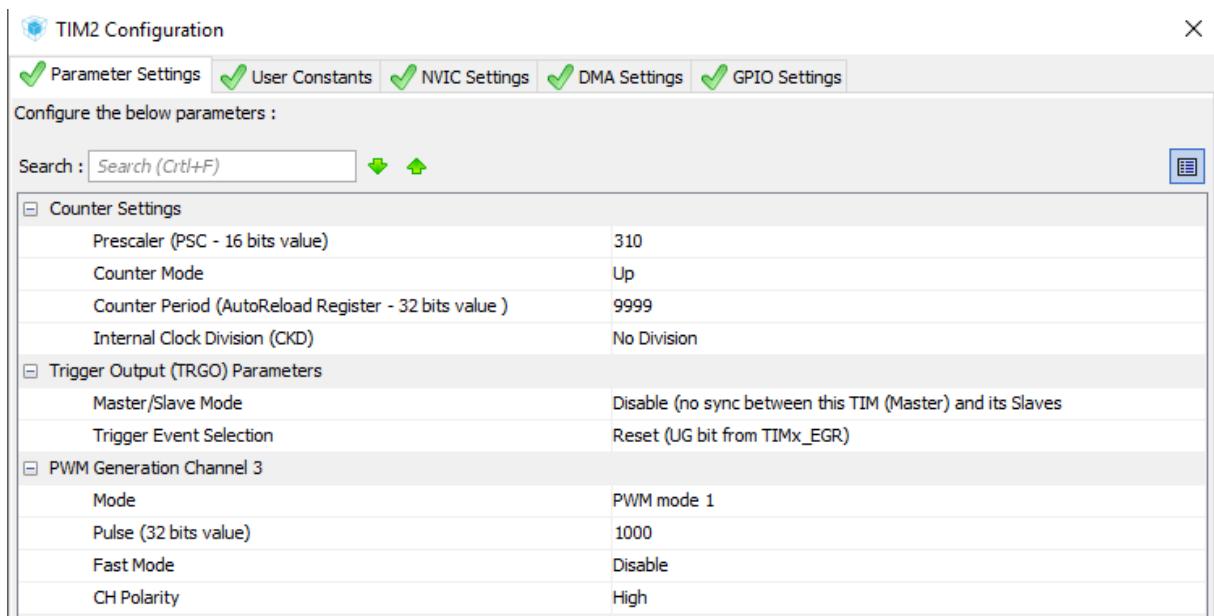
PWM tekniğinin avantajları arasında, anahtarlama kaybının düşük olması, tamamen dijital bir teknik olmasından dolayı dijital sistemler ile uyumlu çalışması sayılabilir. İletişim ve haberleşme sistemlerinde sıkça kullanılmasının yanı sıra temel anahtarlama devrelerinde ve çeşitli motor sürücülerinde, hobi elektroniki devrelerinde de kullanılmaktadır.

STM32 işlemcileri ile PWM sinyali üretmek için timer birimi kullanılabilir. Böylece PWM sinyali donanımsal çevre birimi tarafından üretilerek, işlemci üstüne düşen yük azaltılmaktadır. PWM sinyali timer birimi tarafından üretildiğinden dolayı, PWM frekans değeri;

$$Freq = Clk_Source_Freq / (PSC * (Counter_Per + 1))$$

formülü ile hesaplanabilir.

2010

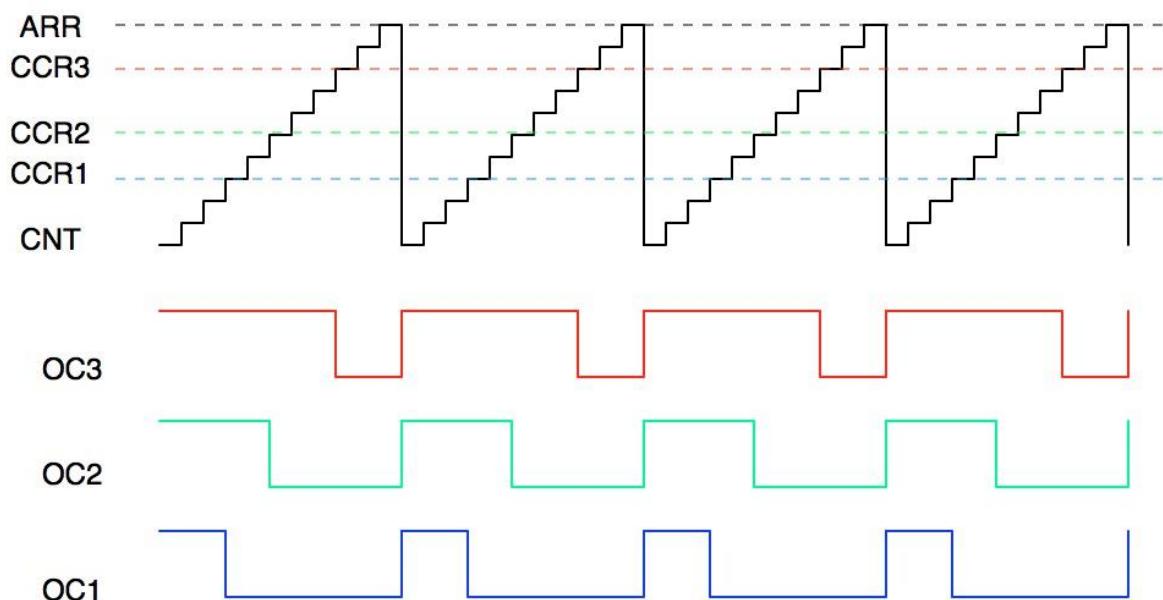


Şekil 8-1 PWM deneyi Timer Parametre ayarları

Şekil 8-1 incelenirse, Timer uygulamasından farklı oalrak, PWM çıkışları aktif edildiğinde “Configuration” penceresine “PWM Generation Channel x” menüsü eklenmiştir. Eklenen seçeneklere bakıldığında,

- Mode = PWM sinyalinin kenar veya merkez hizalamalı olmasının seçilmesini sağlar.
- Pulse = PWM sinyalinin lojik-0 ve lojik-1 seviyelerinde kaç birim süresince aktif olacaklarının ayarlanması sağlar.
- CH Polarity = PWM sinyalinin aktif-1 veya aktif-0 çalışmasını belirler.

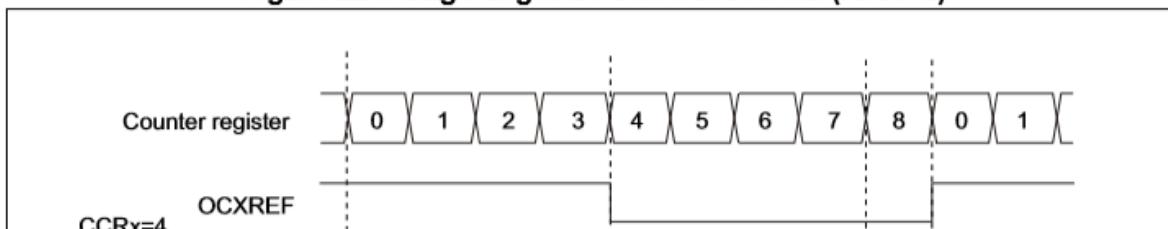
Three PWM signals from the Output Compare Channels of a general purpose timer



Şekil 8-2 PWM deneyi Timer Compare Register şematik gösterimi

Şekil 8-2, PWM sinyalının timer birimi tarafından nasıl oluşturulduğunu göstermektedir. ARR değeri Timer biriminin “Counter Period” değeri tarafından belirlenir. Belirlenen değere kadar 0'dan sayılır ve tekrar başa dönülür. CCR değerleri, PWM kanallarının “Pulse” değerleri ile belirlenir. “CH Polarity” high (aktif 1) olarak seçilirse, counter değeri belirlenen pulse değerine ulaşana kadar PWM sinyali lojik 1 seviyesinde, pulse değerinden counter period değerine kadar da lojik 0 seviyelerinde sinyal üretir.

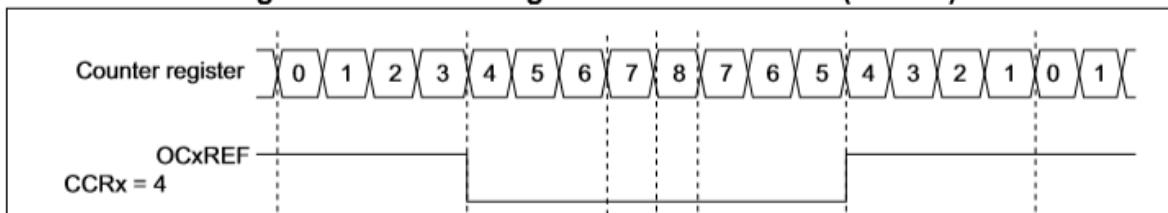
Figure 221. Edge-aligned PWM waveforms (ARR=8)



Şekil 8-3 Timer Counter Register şematik gösterimi

PWM Mode 1 seçilirse, timer birimi 0'dan belirlenen değere kadar sayı ve tekrar başa döner.

Figure 222. Center-aligned PWM waveforms (ARR=8)



Şekil 8-4 Timer Counter Register şematik gösterimi

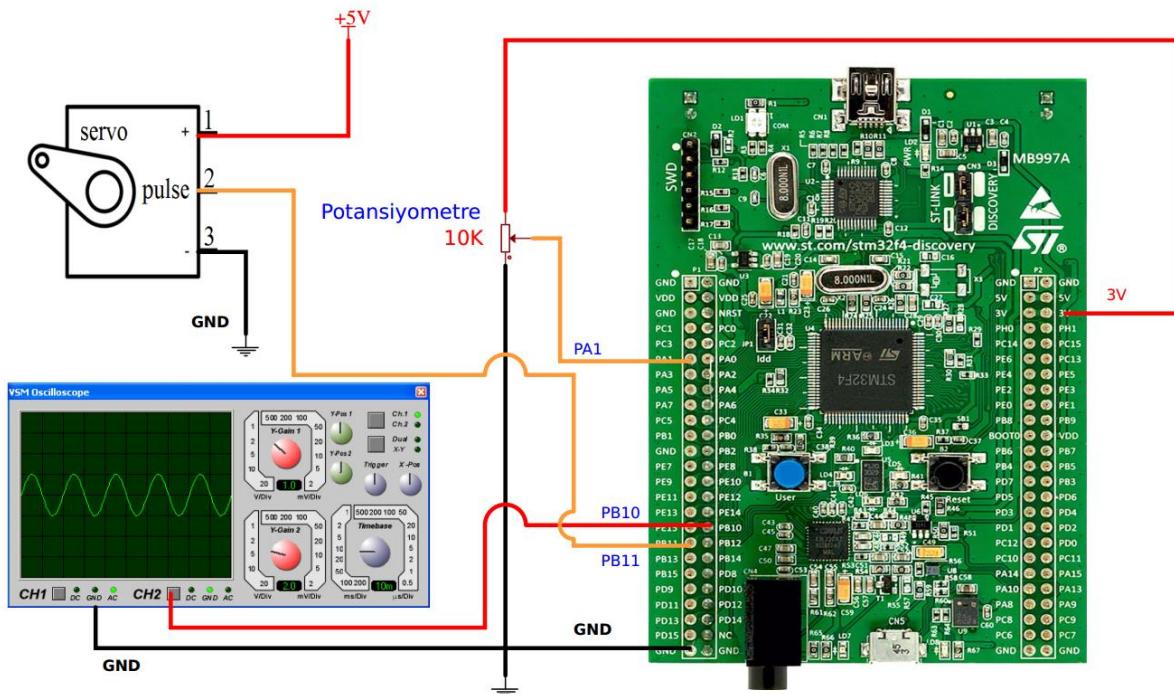
PWM Mode 2 seçilirse, timer birimi 0'dan belirlenen değere kadar sayı ve o değerden 0'a geri sayılır. Bu döngü sürekli tekrarlanır.

8.3 Sorular

1. PWM sinyalının doluluk oranı neyi ifade etmektedir?
2. 50 Hz frekansa sahip %10 doluluk oranına sahip PWM sinyali oluşturmak için Timer ve PWM output değerleri ne olmalıdır?
3. PWM ile analog çıkış oluşturmak için PWM sinyali çıkışına nasıl bir analog devre kurulmalıdır?
4. PWM sinyali ile HC-SR04 ultrasonik mesafe sensörü kontrol edilmek isteniyor. Gerekli PWM sinyalının nasıl ayarlanması gereğini belirleyiniz. Gerekli değerleri bulunuz.

8.4 Devre Şeması

Deneyde oluşturulacak olan devre şeması Şekil 3-1' da görülmektedir, şekil



Şekil 8-5 PWM deneyi devre şeması

8.5 Deneyin Yapılışı

ADIM 1: STM32CubeMX ile devre şemasında kullanılacak pinlere ve aşağıda belirtilen özelliklere göre pinout sekmesinden gerekli ayarlar yapılır.

- PB10 ve PB11 Timer2 biriminin PWM çıkışları olarak ayarlanmalı
- PA1 analog giriş olarak ayarlanmalı
- Timer2 frekansı sorular kısmında bulunan değerler ile ayarlanmalı
- ADC 10 bit olarak ayarlanmalı

ADIM 2: Gerekli kodu oluşturarak, kullanılacak Timer ve PWM kanallarını aktif edecek kodları yazınız.

Örnek kullanım -> `HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);`

ADIM 2: ADC çevrimini başlatarak geliştirme kartına bağladığınız potansiyometreden değer okumaya başlayınız.

ADIM 3: Okunan adc değeri ile, PB10 çıkışını 0 ile belirlediğiniz “counter period” değeri arasında değiştirecek kodu oluşturun.

ADIM 4: Okunan adc değeri ile, PB11 çıkışını servo motorun çalışma aralığında ayarlayacak şekilde değiştiren kodu oluşturun.

ADIM 5: Hazırladığınız kod ile PB10 çıkışını osiloskop ile gözlemleyiniz. PB11 çıkışına bağladığınız PWM kontrollü servo motorun dönme hareketini gözlemleyiniz.