STM32F103 Programlama

23 Şubat 2021 Salı 00:18

- https://www.youtube.com/playlist?list=PL-p1dQWEJtWbHbnawz64RFk0JwpOSS4jD
- https://www.youtube.com/playlist?list=PLre4S1H8v3bQRzjCMNY1BoAihAKKgvWQ_

- [x] 01 Dahili Led Yakma
- [x] 02 Harici Led Yakma
- [x] 03 Buton ile Led Yakma
- [x] 04 Timer Interrupt
- [x] 05 External Interrupt
- [x] 06 Tek Kanal ADC
- [x] 07 ADC DMA
- [x] 08 Çok Kanal ADC
- [x] 09 ADC Interrupt
- [ ] 10 PWM

CENGİZHAN TOPÇU

MEKATRONİK MÜHENDİSİ

01 Dahili Led Yakma

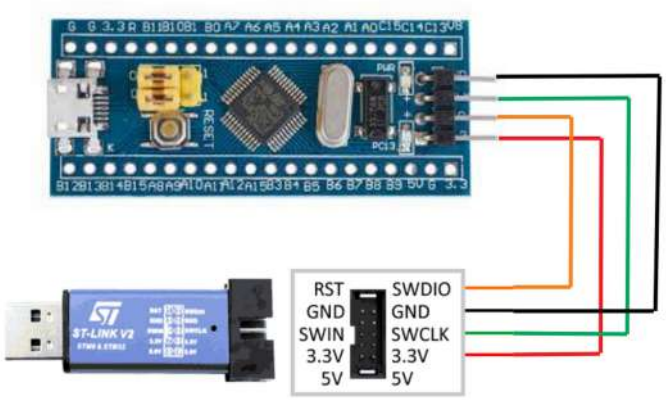
23 Şubat 2021 Salı 01:07

- https://youtu.be/-d6WD1UBiPI
- https://youtu.be/bJhIfAfpWC0
- https://youtu.be/d9XVKVo50oc

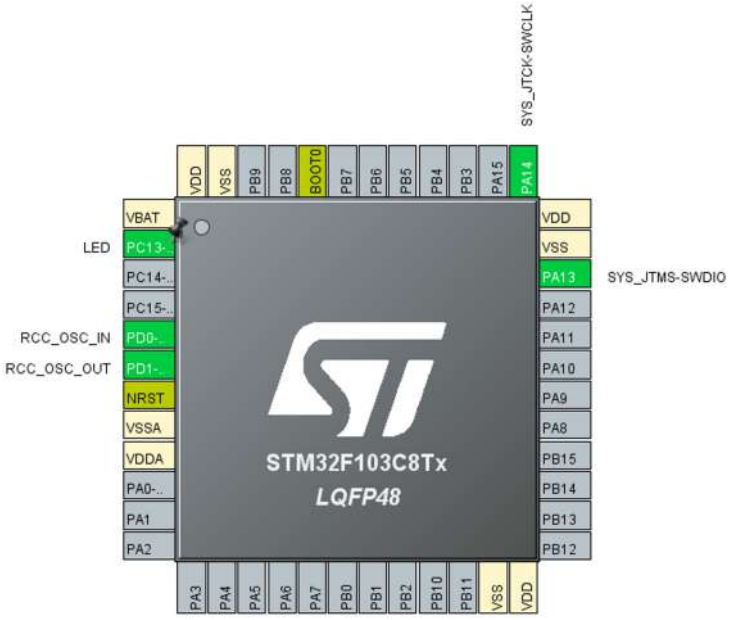
01 Dahili Led Yakma

Teorik Kısmı

- Kodun nasıl yazıldığını bilmeyebiliriz. Bunun için Keil'de Functions kısmından CubeIDE'de Core ve Drivers dosya kısmından bakabiliriz. Burdaki .h dosyalarında fonksiyonların tanımlamaları olur. .c dosyasında kullanabileceğimiz fonksiyonlar gözükür. Nasıl kullanıldığını öğrenmek için çift tıklarız.



Konfigürasyon Kısmı



- System Core kısmından SYS tıklıyoruz ve Debug'da Serial Wire diyoruz. Çünkü St-Link kullanabilmek için bu işlemi yapmamız gerekiyor.

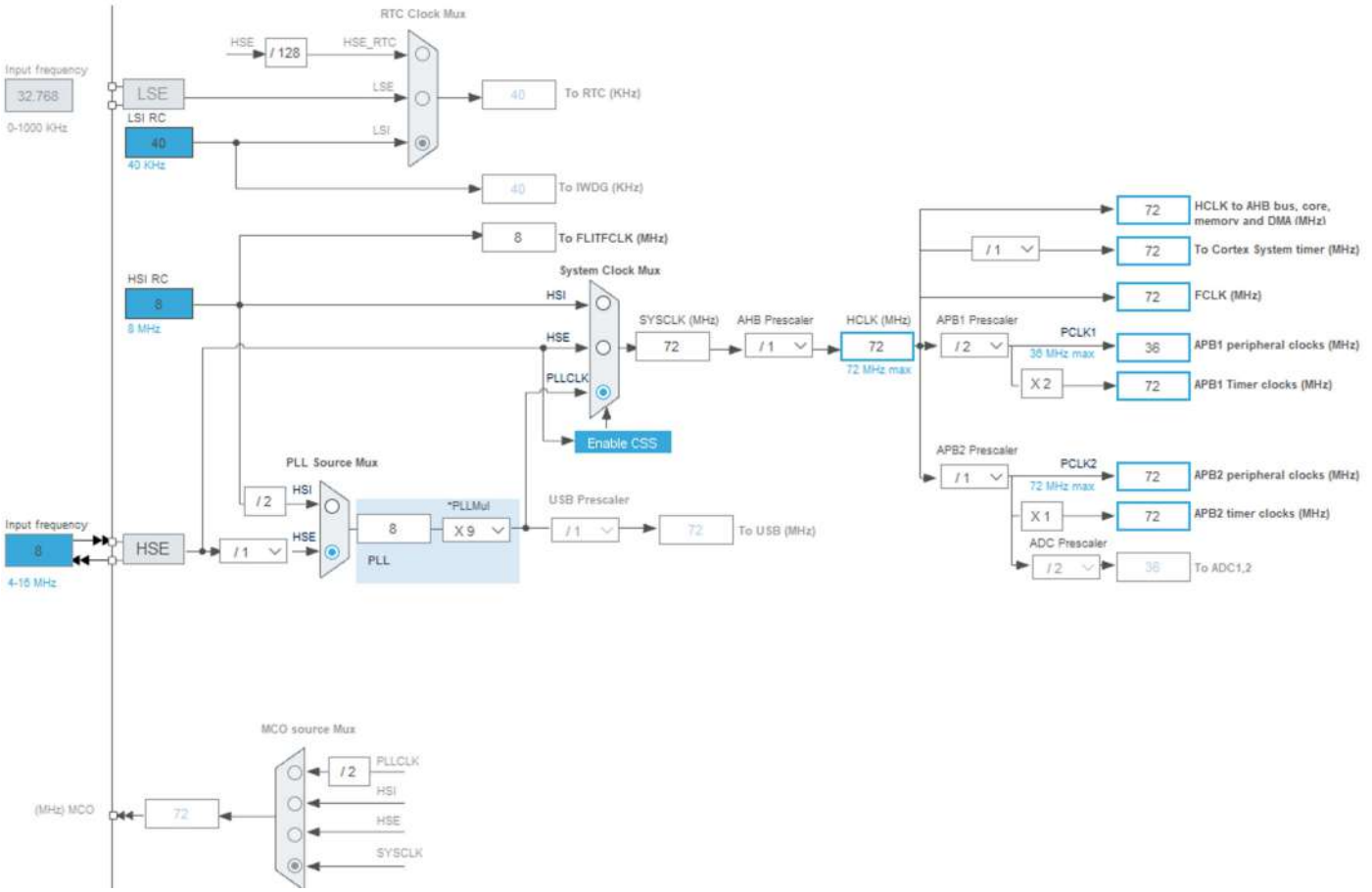
Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-...	Maximum out...	User Label	Modified
PA13	SYS_JTMS-SWDIO	n/a	n/a	n/a	n/a		☐
PA14	SYS_JTCK-SWCLK	n/a	n/a	n/a	n/a		☐

- RCC tıklıyoruz HSE'de Crystal/Ceramic Resonator işaretliyoruz.

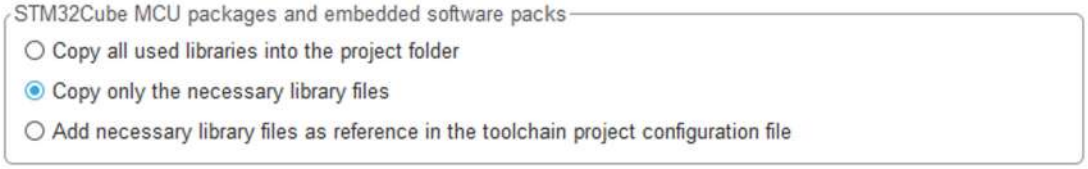
Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-u...	Maximum ou...	User Label	Modified
PD0-OSC_IN	RCC_OSC_IN	n/a	n/a	n/a	n/a		☐
PD1-OSC_O...	RCC_OSC_OUT	n/a	n/a	n/a	n/a		☐

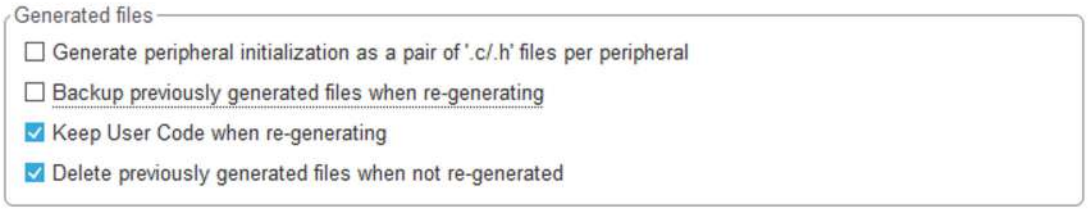
- GPIO kısmında PC13 için GPIO Mode durumunu Push Pull ya da Open Drain olarak iki türlü çıkış ayarlayabiliriz. Birincisi doğrudan çıkış verir, ikincisi için pull-up direnci bağlamamız gerekir.

Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up/P...	Maximum ou...	User Label	Modified
PC13-TAMP...	n/a	Low	Output Push...	No pull-up and ...	Low	LED	☑



- Code Generator kısmından işimize yarayacak olanları işaretliyoruz.





Kod Kısmı

```
20 /* Includes -------
21 #include "main.h"
```

- 49.satırda CubeMX kısmında yaptığımız sistem clock konfigürasyonunda yaptıklarımız kod içerisine kendi dahil olur.
- 50.satırda GPIO başlatma fonksiyonlarını barındırır.

```
48 /* Private function prototypes -
49 void SystemClock_Config(void);
50 static void MX_GPIO_Init(void);
```

- Kod aslında int main(void)'den başlar while'ın sonunda biter.

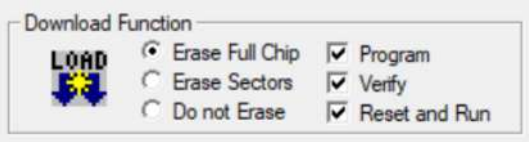
```
64 int main(void)
65 {
66   /* USER CODE BEGIN 1 */
67
68   /* USER CODE END 1 */
69
70   /* MCU Configuration--------------------------------------------------------*/
71
72   /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
73   HAL_Init();
74
75   /* USER CODE BEGIN Init */
76
77   /* USER CODE END Init */
78
79   /* Configure the system clock */
80   SystemClock_Config();
81
82   /* USER CODE BEGIN SysInit */
83
84   /* USER CODE END SysInit */
85
86   /* Initialize all configured peripherals */
87   MX_GPIO_Init();
88   /* USER CODE BEGIN 2 */
89
90   /* USER CODE END 2 */
91
92   /* Infinite loop */
93   /* USER CODE BEGIN WHILE */
94   while (1)
95   {
96     /* USER CODE END WHILE */
97
98     /* USER CODE BEGIN 3 */
99   }
```

```
465 void HAL_GPIO_WritePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState)
```

- Bu fonksiyona hal_gpio.c dosyasından ulaşırız.
- Write pin komutun üç parametresi sırasıyla GPIO portu, pini ve pinin durumudur.
- PC13 pini için SET ile 0, RESET ile 1 yapılır.

```
94   while (1)
95   {
96     /* USER CODE END WHILE */
97
98     /* USER CODE BEGIN 3 */
99         HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_SET);
100        HAL_Delay(1000);
101        HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_RESET);
102        HAL_Delay(1000);
103
104  }
```

- Kod yükleme işlemi yaparken önce hatamız var mı öğrenmek için önce Build yapılır.
- Keil'de Options and Target kısmından ayarlamalar yaparız. Output'tan Hex kodunu oluşturabiliriz. Debug kısmından St-Link Debugger seçilir ve Setting kısmından ayarlamalar yapılır ardından load yapılır.



- Kodu St-Link Utility programı ile yüklemek istersek önce aygıta bağlanmak için Connect to the Target diyoruz.
  Hex kodu yüklemek için Program Verify tıklıyoruz. Debug kısmında Keil için hex dosyası, CubeIDE içim bin dosyasını seçiyoruz.
- CubeIDE'de Run Configurations kısmından Debbugger tıklanır ve Debug Probe kısmından ST-LINK (OpenOCD) seçilir. Burda Show generator options tıklanır Hide generator options yapılır ayrıca Reset Mode Software system reset seçilir ve Apply deyip OK denir.

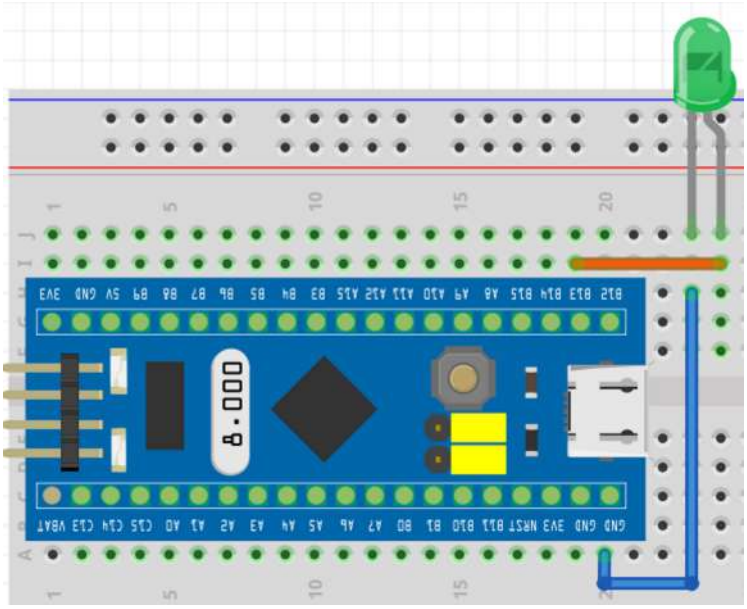
02 Harici Led Yakma

23 Şubat 2021 Salı 01:07

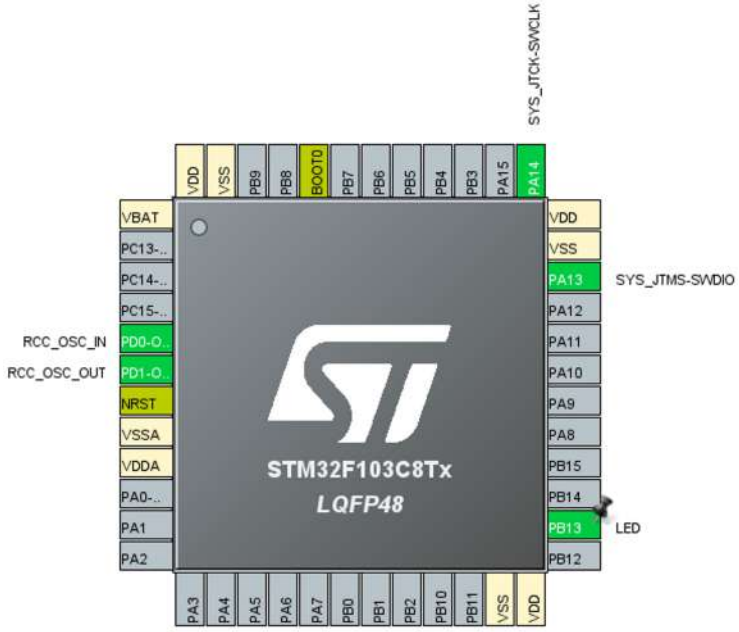
- https://youtu.be/s5_FLd9RKVA

02 Harici Led Yakma

Devre Kısmı



Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PB13	n/a	Low	Output Push...	No pull-up an...	Low	LED	☑

Kod Kısmı

- Toggle 0 olan durumu 1, 1 olan durumu 0 yapar.
- Port yazarken pine verdiğimiz isim şekli olan "LED_GPIO_Port" yerine direk portu "GPIOB" olarak yazabiliriz. Bu tanımlamayı main.h metninden bakarak yazıyoruz. Aynı şekilde aynı yerden pin adını da "LED_Pin" yerine "GPIO_PIN_13" şeklinde yazabiliriz.
- Bu tanımlamalara üzerine gelip Keil'de F12 ile CubeIDe'de F3 ile bakabiliriz. Aşağıda

F3'e tıkladığımızda main.h kütüphanesindeki bu satırları gösteriyor.

```
#define LED_Pin GPIO_PIN_13
#define LED_GPIO_Port GPIOB
```

- Bu fonksiyona hal_gpio.c dosyasından ulaşırız.

```
487 void HAL_GPIO_TogglePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)
```

```
 94   while (1)
 95   {
 96     /* USER CODE END WHILE */
 97
 98     /* USER CODE BEGIN 3 */
 99       HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin);
100       HAL_Delay(200);
101
102   }
```

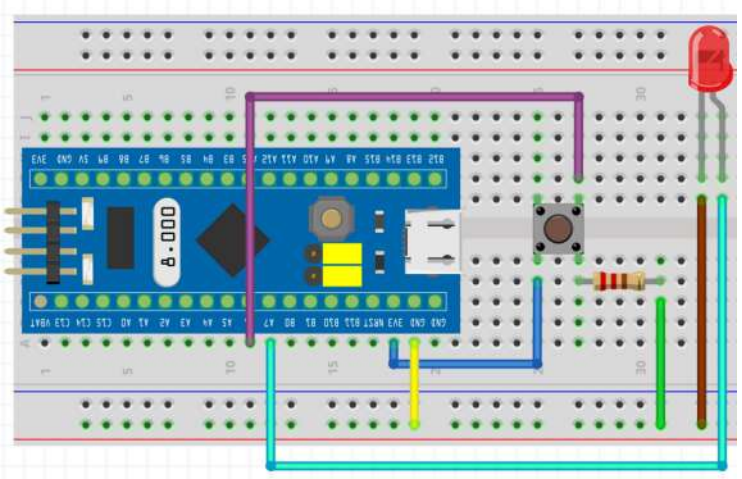
03 Buton ile Led Yakma

27 Şubat 2021 Cumartesi 18:36

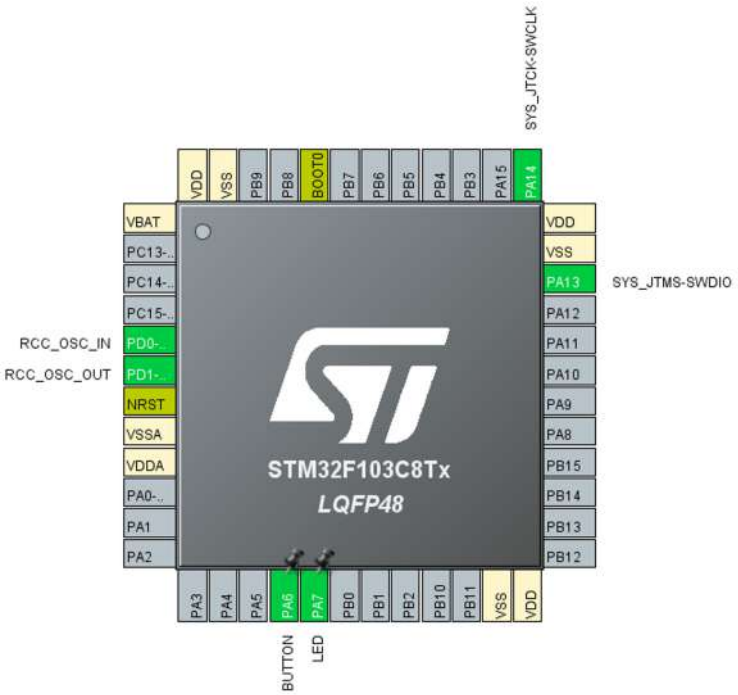
- https://youtu.be/XimVOa4tQGI

03 Buton ile Led Yakma

Devre Kısmı



Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA6	n/a	n/a	Input mode	Pull-down	n/a	BUTTON	☑
PA7	n/a	Low	Output Push...	No pull-up an...	Low	LED	☑

Kod Kısmı

- Bu fonksiyona hal_gpio.c dosyasından ulaşırız.

```
431 GPIO_PinState HAL_GPIO_ReadPin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)
```

```
94  while (1)
95  {
96    /* USER CODE END WHILE */
97
98    /* USER CODE BEGIN 3 */
99      if(HAL_GPIO_ReadPin(GPIOA,BUTTON_Pin)==1)
100     {
101           HAL_GPIO_WritePin(GPIOA,LED_Pin,GPIO_PIN_SET);
102     }
103     else
104     {
105           HAL_GPIO_WritePin(GPIOA,LED_Pin,GPIO_PIN_RESET);
106     }
107
108 }
```

04 Timer Interrupt

27 Şubat 2021 Cumartesi 00:03

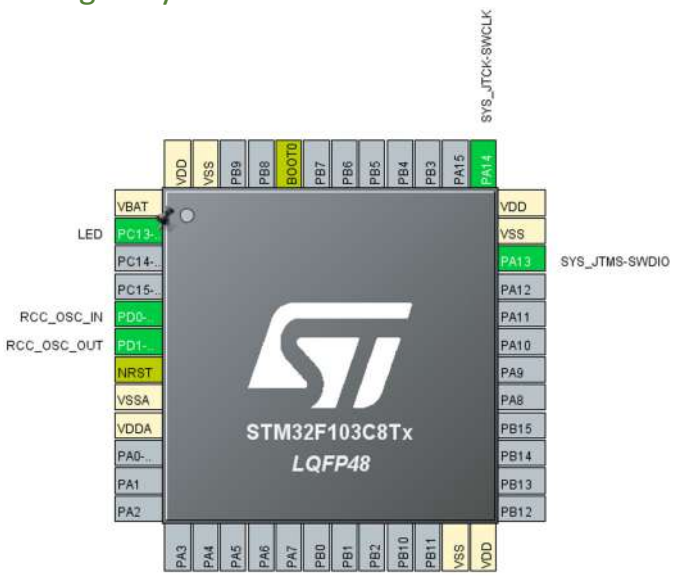
- https://youtu.be/-FO_jc_jiag
- https://youtu.be/KLtSiVqjKkk

04 Timer Interrupt

Teori Kısmı

- Timer saymaya başlar belirttiğimiz sayıdan sonra taşma işlemi oluşuyor ve başa dönüyor. Limite göre de istediğimiz işlemi yapıyoruz. Taşma işlemine kesme adını veriyoruz.
- Timer, delaydan farkı arka tarafta çalışmasıdır.

Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up/P...	Maximum ou...	User Label	Modified
PC13-TAMP...	n/a	Low	Output Push...	No pull-up and ...	Low	LED	☑

- Timer1 için ayarlama yapıyoruz. Timer için girilen değerlere göre saniye hesabını aşağıdaki formül ile yaparız. Bizim timer 1 saniye aralıklarla tekrarlı şekilde çalışır.
- Timers kısmında TIM1 tıklarız ve öncelikle Mode kısmında Clock Source'dan Internal Clock seçilir. Bu seçim bizim dahili olanı seçtiğimizi gösterir. ETR2 ise harici olanı kullancağımız anlamına gelir.
- Daha sonra Parameter Settings'den aşağıdaki değerler girilir. Prescalaer bizim sayısının en üst seviyesini belirler. Sayma işlemi 0'dan başlamayıp 1'den başladığından 1 eksiğini alarak yazarız. Sayma şeklini Up belirleyip yukarı doğru sayıyor. Counter Period kısmında her seferinde taşma işlemi bittikten sonra tekrar bunu yükler. Yükleyeceği değeri yazarız. Auto-reload preload kısmında Enable diyerek sayma bittiğinde başa dönmesini sağlarız.

Counter Settings

Prescaler (PSC - 16 bits value)	36000-1
Counter Mode	Up
Counter Period (AutoReload Register - 16 bits val...	2000-1
Internal Clock Division (CKD)	No Division
Repetition Counter (RCR - 8 bits value)	0
auto-reload preload	Enable

$$UpdateEvent = \frac{Timer_{clock}}{(Prescaler + 1)(Period + 1)}$$

$$UpdateEvent = \frac{72.000.000}{(36000)(2000)} = 1\,Hz = \frac{1}{1}s = 1\,s$$

- NVIC Settings kısmından TIM1 update interrupt Enabled yapılır. Bununla her güncellemede, sayıyı bitirmede bir interrupt oluşmasını sağlıyoruz.

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
TIM1 break interrupt	☐	0	0
TIM1 update interrupt	☑	0	0
TIM1 trigger and commutation interrupts	☐	0	0
TIM1 capture compare interrupt	☐	0	0

Kod Kısmı

- 92.satırdaki kod ile interrupt modunda Timer'ı çalıştırır.
- Bu fonksiyona hal_tim.c dosyasına baktığımızda htim önünde yıldız görürüz. Bu önüne "&" işareti kullanılır.

```
458 HAL_StatusTypeDef HAL_TIM_Base_Start_IT(TIM_HandleTypeDef *htim)
```

```
88   /* Initialize all configured peripherals */
89   MX_GPIO_Init();
90   MX_TIM1_Init();
91   /* USER CODE BEGIN 2 */
92       HAL_TIM_Base_Start_IT(&htim1);
93
94   /* USER CODE END 2 */
```

- Az önce yazılan kod ile it.c dosyasında fonksiyon altına otomatik gelir. Gelmez ise timer'ın interrup kısmı başlamaz.

```
205 void TIM1_UP_IRQHandler(void)
206 {
207   /* USER CODE BEGIN TIM1_UP_IRQn 0 */
208
209   /* USER CODE END TIM1_UP_IRQn 0 */
210   HAL_TIM_IRQHandler(&htim1);
211   /* USER CODE BEGIN TIM1_UP_IRQn 1 */
212
213   /* USER CODE END TIM1_UP_IRQn 1 */
214 }
```

- Timer1 için yapması gereken işlemi yazıyoruz. Öncesinde değişkeni yazıyoruz.

```
45 /* USER CODE BEGIN PV */
46 uint16_t sayi=0;
47 /* USER CODE END PV */
```

- hal_tim.c dosyasından HAL_TIM_PeriodElapsedCallback fonksiyonunu çekeriz. Timer'ın yaptığı işlemler buradan çağırılır.

```
5511  __weak void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
```

- Hangi timer'ın yapısını kullanıyorsam önce onu gösteriyorum. 224.satırda eğer taşma yapan timer TIM1 ise sayi değişkenini 1 arttır diyorum. Yani taşma işlemi olursa istediğimiz işlemleri yapabiliyoruz.
- Sadece 226.satırdaki kod çalışsaydı 1 saniyelik gecikme yapıp ledimiz yanıp sönecekti. Fakat 1 saniye'de bir sayi değişkenini 1 arttırıp 5 olduğu zaman ledi söndürüyor.

```
222 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
223 {
224     if(htim->Instance==TIM1)
225     {
226         //HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin);
227         sayi++;
228     }
229 }
```

```
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
         if(sayi==5)
         {
            HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin);
         }
  }
  /* USER CODE END 3 */
}
```

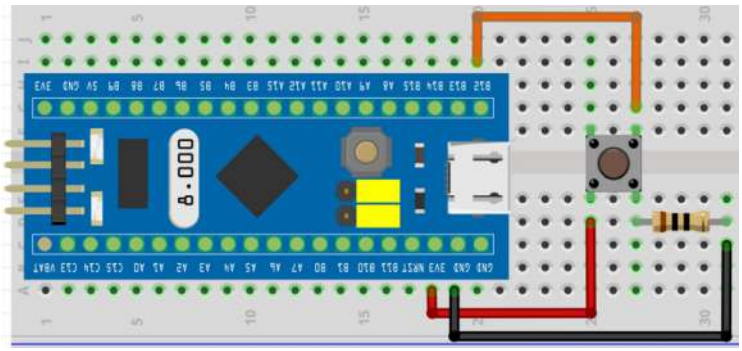
05 External Interrupt

1 Mart 2021 Pazartesi 19:18

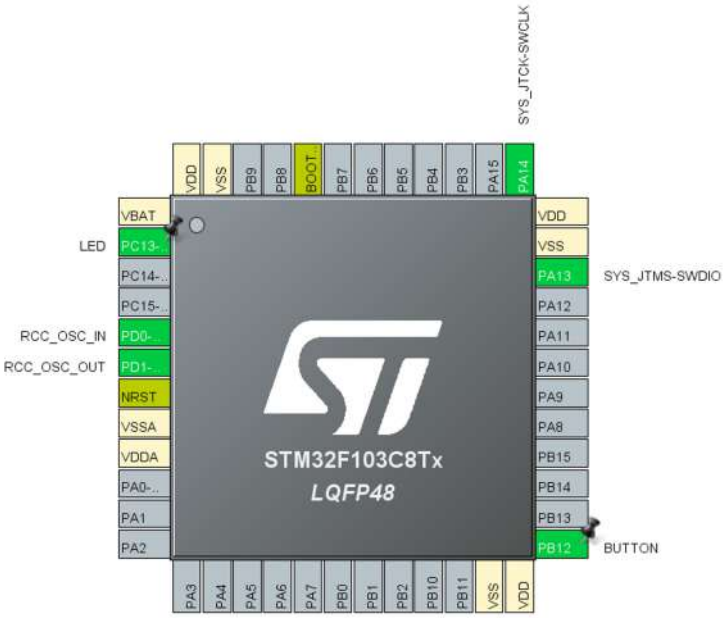
- https://youtu.be/QN16t6J1VKA
- https://youtu.be/hikauabT0bM

05 External Interrupt

Devre Kısmı



Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PB12	n/a	n/a	External Inte...	Pull-down	n/a	BUTTON	☑
PC13-TAMP...	n/a	Low	Output Push...	No pull-up an...	Low	LED	☑

- PB12 pinini buton adına GPIO_EXTI12 olarak ayarlıyoruz.
- Bunu yaptıktan sonra kesme ayarı yapmamız gerekiyor. NVIC kısmından ilgili satır olan en alttakini işaretliyoruz ve Preemption Priority değerini 15 yapıyoruz. Bunun anlamı öncelik sırasını belirlemek için kullanıyoruz ve biz en son değeri seçiyoruz.

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priorit
Non maskable interrupt	☑	0	0
Hard fault interrupt	☑	0	0
Memory management fault	☑	0	0
Prefetch fault, memory access fault	☑	0	0
Undefined instruction or illegal state	☑	0	0
System service call via SWI instruction	☑	0	0
Debug monitor	☑	0	0
Pendable request for system service	☑	0	0
Time base: System tick timer	☑	0	0
PVD interrupt through EXTI line 16	☐	0	0
Flash global interrupt	☐	0	0
RCC global interrupt	☐	0	0
EXTI line[15:10] interrupts	☑	15	0

- NVIC birimi mikrodenetleyicideki interrupt işlemesini konrol eden birimdir.

Kod Kısmı

- İlk önce söndürme işlemi yapıyoruz.

```
88   /* USER CODE BEGIN 2 */
89   HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_SET);
90   /* USER CODE END 2 */
```

```
44 /* USER CODE BEGIN PV */
45 uint16_t sayi=0;
46 /* USER CODE END PV */
```

- Kesme işlemi olduğunda nasıl davranacağını aşağıdaki gibi belirleriz.
- Bu fonksiyona hal_gpio.c dosyasından ulaşırız.

```
561 __weak void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
```

```
178 /* USER CODE BEGIN 4 */
179 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
180 {
181     HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin);
182     sayi++;
183 }
184
185 /* USER CODE END 4 */
```

- Interrupt için Callback fonksiyonlarını kullandık ayrıca Handler fonksiyonlarını da kullanabilirdik fakat bu fonksiyon Callback fonksiyonlarını arkada tarafta çağırıyor.
- Handler fonksiyonuyla yaparsak it.c fonksiyonunda kesme olduğunda EXTI15_10 _IRQHandler fonksiyonu altında yazdığımız kodlar çalışmış olacak.

```
205 void EXTI15_10_IRQHandler(void)
206 {
207   /* USER CODE BEGIN EXTI15_10_IRQn 0 */
208     HAL_GPIO_TogglePin(GPIOC,GPIO_PIN_13);
209     while (HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_12))
210
211   /* USER CODE END EXTI15_10_IRQn 0 */
212   HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_12);
213   /* USER CODE BEGIN EXTI15_10_IRQn 1 */
```

```
205 void EXTI15_10_IRQHandler(void)
206 {
207   /* USER CODE BEGIN EXTI15_10_IRQn 0 */
208     HAL_GPIO_TogglePin(GPIOC,GPIO_PIN_13);
209     while (HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_12))
210
211   /* USER CODE END EXTI15_10_IRQn 0 */
212   HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_12);
213   /* USER CODE BEGIN EXTI15_10_IRQn 1 */
214
215   /* USER CODE END EXTI15_10_IRQn 1 */
216 }
```

- Kod çalıştığında led yanıyor daha sonra butona bastığımızda led sönüyor.

06 Tek Kanal ADC

1 Mart 2021 Pazartesi 23:26

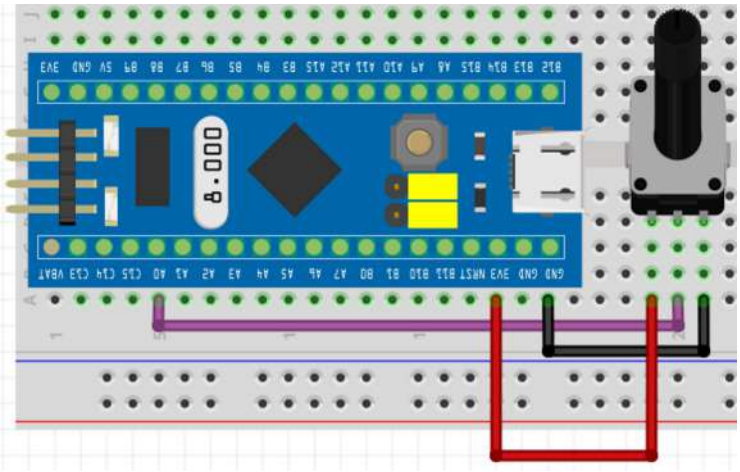
- https://youtu.be/3uDl-H8o7Xg
- https://youtu.be/hsMzpDI1hx0

06 Tek Kanal ADC

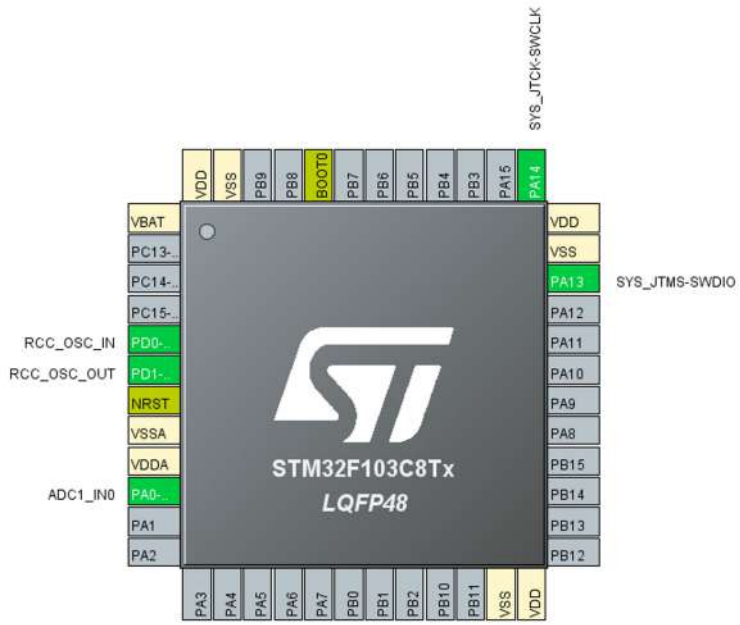
Teori Kısmı

- Mikrodenetleyicimizde 12 bitlik ADC yapısı var.

Devre Kısmı



Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA0-WKUP	ADC1_IN0	n/a	Analog mode	n/a	n/a		☐

- Analog değer okumak için potansiyometreyi PA0'a bağladık. Bunun için Analog kısmından ADC1'den IN0 tıklarız.
- Daha sonra Parameter Settings'den Continuous Conversion Mode Enabled yapılır. Sürekli çevrim modu anlamına gelir ve tekrar tekrar okuma durumu yapar eğer çalıştırmazsak bir kere okur daha sonra değer okumaz.

ADC_Settings	
Data Alignment	Right alignment
Scan Conversion Mode	Disabled
Continuous Conversion Mode	Enabled
Discontinuous Conversion Mode	Disabled

Kod Kısmı

- "&" işaretinden sonra yazdığımız bizim kaynağımız oluyor.
- Önce ADC işlemini başlattık.

```
88    /* Initialize all configured peripherals */
89    MX_GPIO_Init();
90    MX_ADC1_Init();
91    /* USER CODE BEGIN 2 */
92    HAL_ADC_Start(&hadc1);
93    /* USER CODE END 2 */
```

```
45  /* USER CODE BEGIN PV */
46  uint16_t adc_deger=0;
47  /* USER CODE END PV */
```

```
97    while (1)
98    {
99      /* USER CODE END WHILE */
100
101     /* USER CODE BEGIN 3 */
102       adc_deger=HAL_ADC_GetValue(&hadc1);
103   }
104   /* USER CODE END 3 */
105 }
```

- 2^12=4096 olduğundan en fazla bu değeri gösterebilir.

07 ADC DMA

1 Mart 2021 Pazartesi 23:26

- https://youtu.be/tsZrPbWYKfI

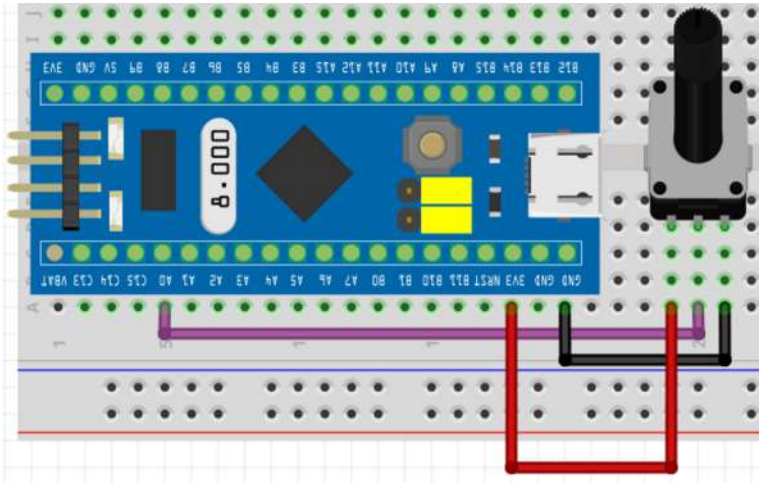
07 ADC DMA

Teori Kısmı

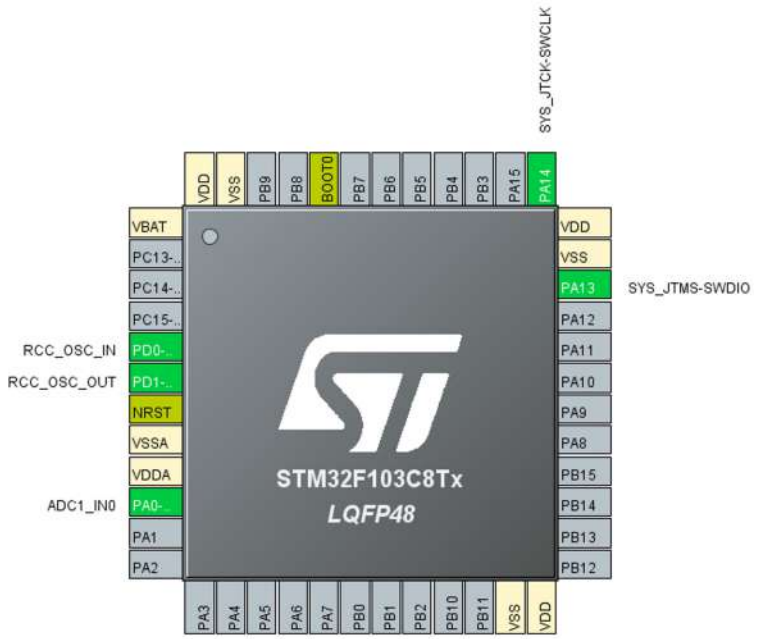
- İşlemci yaptığı işlemleri bellekte saklarlar. O bellekteki bilgiyi tekrar isteyerek tekrardan işleriz.
- DMA yönetimi ile ilgili tablo bize hangi kanalın hangi DMA'da olduğunu gösteriyor.

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
ADC	ADC	ADC	-	-	-
SPI	-	SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX
USART	-	USART1_TX USART3_TX	USART1_RX USART3_RX	USART1_TX USART2_TX	USART1_RX USART2_RX
I2C	-	I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX
TIM1	-	TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_CH3 TIM1_UP
TIM3	-	TIM3_CH3	TIM3_CH4 TIM3_UP	TIM3_CH1 TIM3_TRIG	-
TIM6	-	-	TIM6_UP	-	-
TIM7	-	-	-	TIM7_UP	-
TIM15	-	-	-	-	TIM15_CH1 TIM15_UP TIM15_TRIG TIM15_COM
TIM16	-	-	TIM16_CH1 TIM16_UP	TIM16_CH1 TIM16_UP	-
TIM17	TIM17_CH1 TIM17_UP	TIM17_CH1 TIM17_UP	-	-	-

Devre Kısmı



Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA0-WKUP	ADC1_IN0	n/a	Analog mode	n/a	n/a		☐

- Önceki uygulamamızdan ekstra olarak ADC1 ayarları kısmından DMA Settings tıklayıp Add işleminden ADC1 seçilir ve burdan Mode kısmı sürekli olan Circcular ile Memory kısmındaki Word seçilir.

DMA Request	Channel	Direction	Priority
ADC1	DMA1 Channel 1	Peripheral To Memory	Low

Add Delete

DMA Request Settings

		Peripheral	Memory
Mode	Circular		
	Increment Address	☐	☑
	Data Width	Half Word	Word

Kod Kısmı

- Değişkenimizi dizi olarak tanımladık. Çünkü, bizden dizi olarak istenilecek.

```
/* Private macro --------
/* USER CODE BEGIN PM */
uint32_t adc_deger[1];
/* USER CODE END PM */
```

- ADC'yi başlatırken DMA kullandığımızdan buna göre yazıyoruz. Bunu yazarken bizden ekstra iki parametre istiyor. Birincisi dizi halindeki değişken adı, ikincisi dizinin uzunluğu yani kaç data alacaksın, kaç diziye kaydedeceksin anlamındadır. Biz burada bir kanal kullandığımızdan bir yazdık.

```
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_DMA_Init();
  MX_ADC1_Init();
  /* USER CODE BEGIN 2 */
  HAL_ADC_Start_DMA(&hadc1,adc_deger,1);
  /* USER CODE END 2 */
```

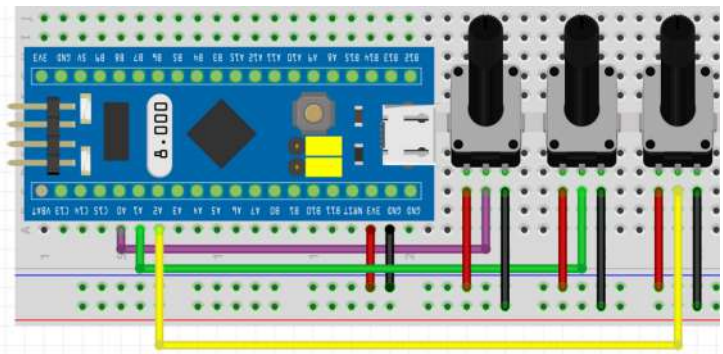
08 Çok Kanal ADC

5 Mart 2021 Cuma 02:36

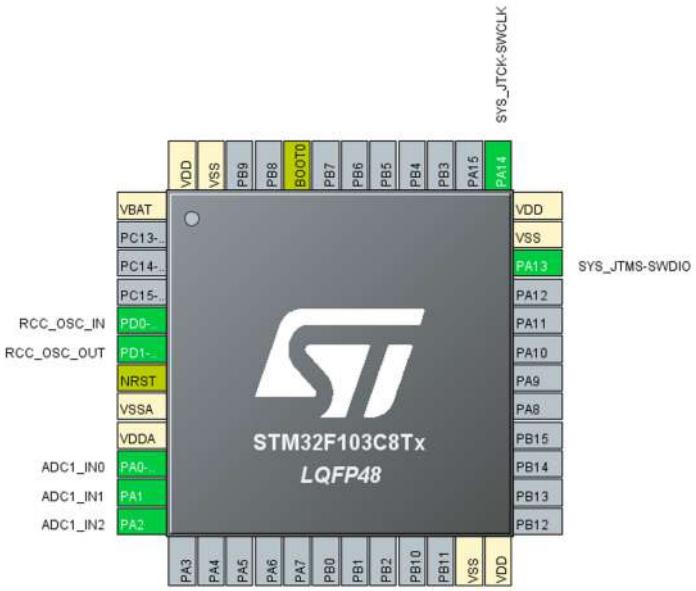
- https://youtu.be/tsZrPbWYKfI
- https://youtu.be/oTO_kRq2SmA

08 Çok Kanal ADC

Devre Kısmı



Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA0-WKUP	ADC1_IN0	n/a	Analog mode	n/a	n/a		☐
PA1	ADC1_IN1	n/a	Analog mode	n/a	n/a		☐
PA2	ADC1_IN2	n/a	Analog mode	n/a	n/a		☐

- Number of Conversion kısmında toplam kanal sayısını giriyoruz. Biz IN0, IN1 ve IN2 seçtiğimiz için 3 adet olarak giriyoruz ve Scan Conversion Mode otomatik Enabled oluyor. Bu da tarama işlemini gerçekleştiriyor.
- Rank ile kanalları seçebiliyoruz. Tekrarlanma zamanlarını belirleyebiliyoruz.
- Tekrardan DMA ayarlarını yapıyoruz.

ADC_Settings

Data Alignment	Right alignment
Scan Conversion Mode	Enabled
Continuous Conversion Mode	Enabled
Discontinuous Conversion Mode	Disabled

ADC_Regular_ConversionMode	
Enable Regular Conversions	Enable
Number Of Conversion	3
External Trigger Conversion Source	Regular Conversion launched by software
Rank	1
Rank	2
Rank	3

Rank	1
Channel	Channel 0
Sampling Time	1.5 Cycles
Rank	2
Channel	Channel 1
Sampling Time	1.5 Cycles
Rank	3
Channel	Channel 2
Sampling Time	1.5 Cycles

Kod Kısmı

- Toplam üç kanal olduğundan 3 yazdık.

```
59 /* Private user code ---
60 /* USER CODE BEGIN 0 */
61 uint32_t adc_deger[3];
62 /* USER CODE END 0 */
```

```
90   /* Initialize all configured peripherals */
91   MX_GPIO_Init();
92   MX_DMA_Init();
93   MX_ADC1_Init();
94   /* USER CODE BEGIN 2 */
95   HAL_ADC_Start_DMA(&hadc1,adc_deger,3);
```

09 ADC Interrupt

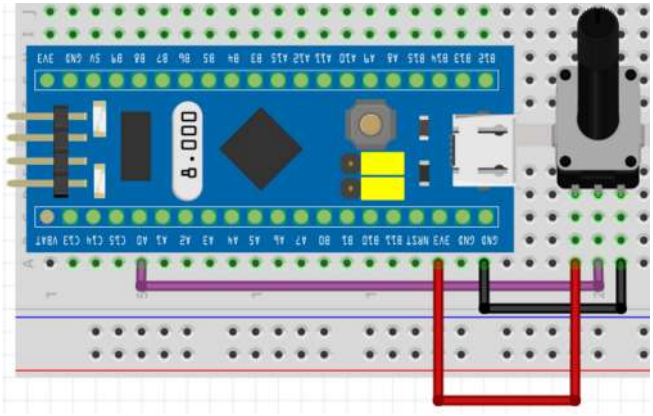
1 Mart 2021 Pazartesi 23:26

09 ADC Interrupt

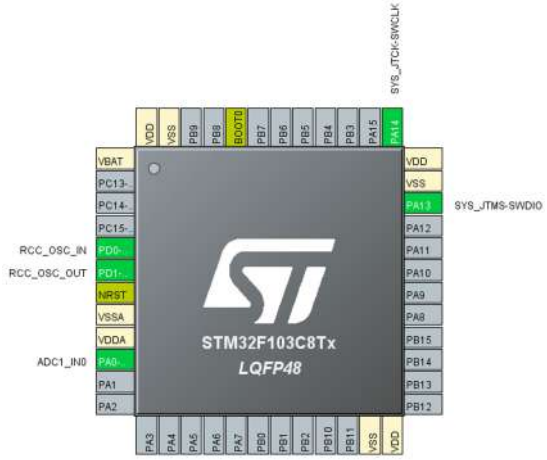
Teori Kısmı

- ADC oluşturduktan sonra Interrupt oluşturuyor. Oluşturduğu Interrupt içerisinde kayıt oluşturacağız.

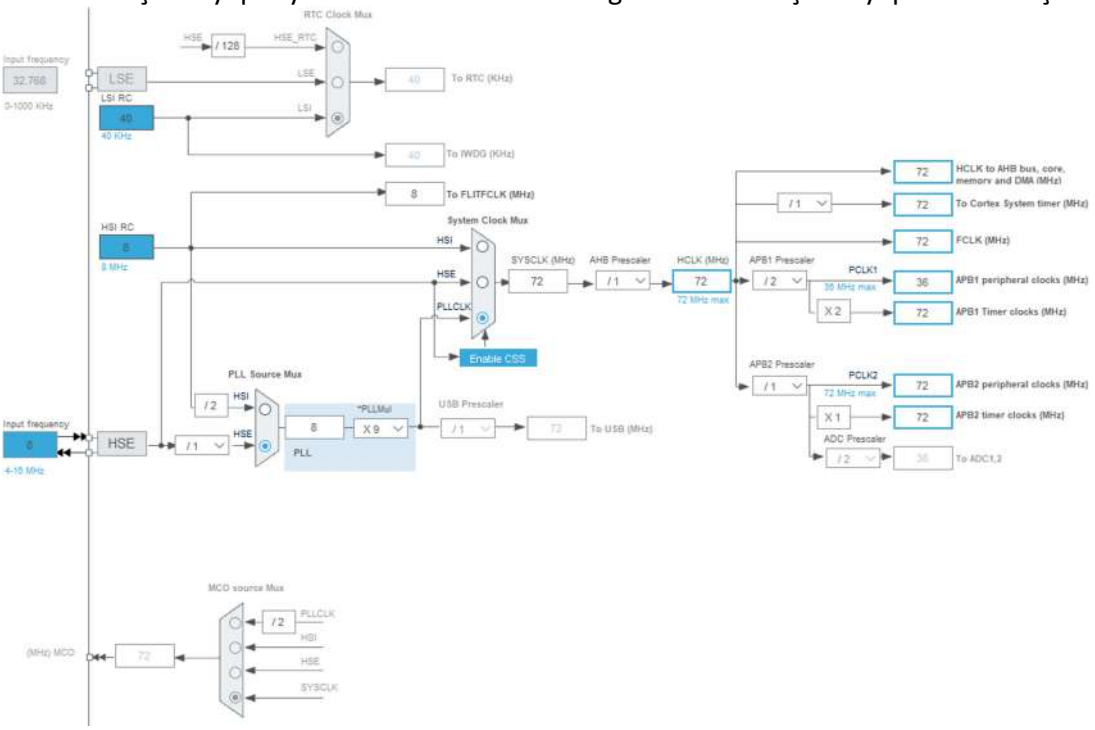
Devre Kısmı



Konfigürasyon Kısmı



- DMA işlemi yapmıyoruz. Sadece NVIC Settings'den kesme işlemi yapılabilmesi için ilk satırdakini işaretliyoruz.



Kod Kısmı

```
37  /* Private macro -------
38  /* USER CODE BEGIN PM */
39  uint32_t adc_deger;
40  /* USER CODE END PM */
```

```
88    /* Initialize all configured peripherals */
89    MX_GPIO_Init();
90    MX_ADC1_Init();
91    /* USER CODE BEGIN 2 */
92  HAL_ADC_Start_IT(&hadc1);
93    /* USER CODE END 2 */
```

- Interrupt olursa bu bölüme gelip buradaki kodları çalıştırır. Eğer ADC1 ise değeri değişkene yazdırır.

```
210  /* USER CODE BEGIN 4 */
211  void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
212  {
213    if(hadc->Instance==ADC1)
214    {
215    adc_deger=HAL_ADC_GetValue(&hadc1);
216    }
217  }
218  /* USER CODE END 4 */
```

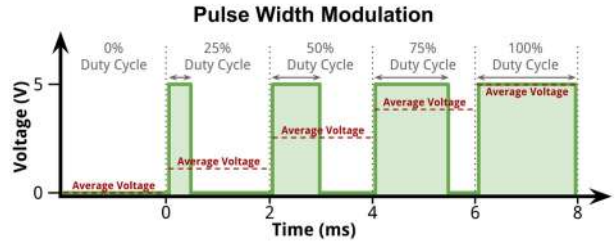
10 PWM

1 Mart 2021 Pazartesi 23:26

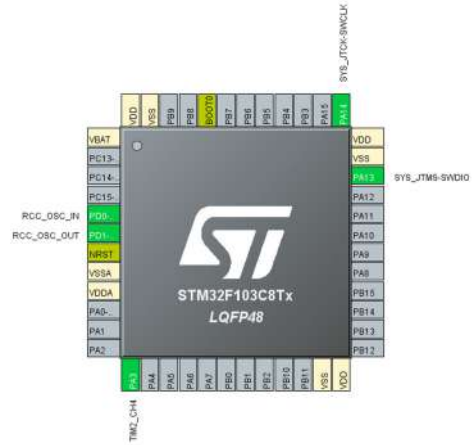
- https://youtu.be/keS-JecLk7U
- https://youtu.be/bXo1TbHVfR4

10 PWM

Teori

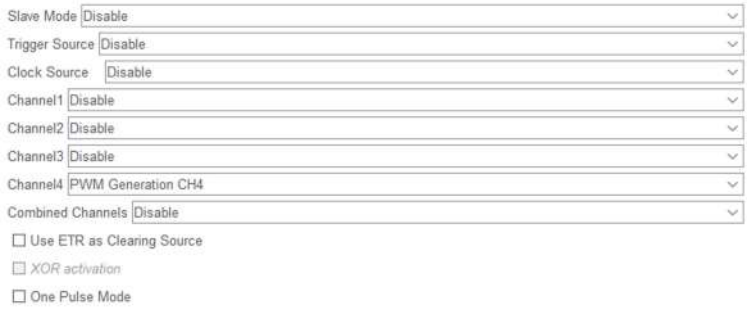


Konfigürasyon Kısmı



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA3	TIM2_CH4	n/a	Alternate Fu...	n/a	Low		☐

- Pwm için TIM2'den Channel 4 kısmından PWM Generation CH4 seçilir.



- 50Hz'lik sinyal üretebilmek için Parameter Settings kısmından Prescaler işlem sonucunu gireriz. Period kısmını Duty Cycle en fazla 100 olduğundan Period kısmına 100-1 olarak gireriz. Pulse kısmı için işlemde 1 kullanırız fakat 1-1'den 0 olarak gireriz. İşlem sonucunda Prescaler 14400 girilir.
- Pulse değeri 50 girilirse doluluk oranı %50 olur.

$$UpdateEvent = \frac{72.000.000}{(Prescaler + 1)(100)} = 50\ Hz$$

$$Prescaler + 1 = 14400$$

Counter Settings

Prescaler (PSC - 16 bits value)	14400-1
Counter Mode	Up
Counter Period (AutoReload Register - 1...	100-1
Internal Clock Division (CKD)	No Division
auto-reload preload	Disable

- Bunun için bir değişiklik yapmıyoruz.

PWM Generation Channel 4

Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High

- NVIC Settings'den TIM2 global interrupt'ın Enabled kısmı açık yapılır.

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
TIM2 global interrupt	☑	0	0

- Pin olarak PA3 seçilmiş. Bunu üzerine Ctrl tuşu ile basarsak bize farklı renkte PB11 üzerindede kullanabileceğimizi gösteriyor.

Kod Kısmı

- PWM işlemini başlatıyoruz. Başlatmak için hal.tim.c dosyasından fonksiyonu çekiyoruz.

```
1428 HAL_StatusTypeDef HAL_TIM_PWM_Start(TIM_HandleTypeDef *htim, uint32_t Channel)
```

```
 98   /* Initialize all configured peripherals */
 99   MX_GPIO_Init();
100   MX_TIM2_Init();
101   /* USER CODE BEGIN 2 */
102   HAL_TIM_PWM_Start(&htim2,TIM_CHANNEL_4);
103   /* USER CODE END 2 */
```

- Kodumuza fonksiyon ekliyoruz. Burda CubeMX'de PWM için yer alan ayarları yazdık.

```
 57⊖/* Private user code ---------------------------------------
 58 /* USER CODE BEGIN 0 */
 59⊖void SetPwm(uint16_t pulseValue)
 60 {
 61     TIM_OC_InitTypeDef sConfigOC;
 62
 63     sConfigOC.OCMode=TIM_OCMODE_PWM1;
 64     sConfigOC.Pulse=pulseValue;
 65     sConfigOC.OCPolarity=TIM_OCPOLARITY_HIGH;
 66     sConfigOC.OCFastMode=TIM_OCFAST_DISABLE;
 67     HAL_TIM_PWM_ConfigChannel(&htim2,&sConfigOC,TIM_CHANNEL_4);
 68     HAL_TIM_PWM_Start(&htim2,TIM_CHANNEL_4);
 69 }
 70 /* USER CODE END 0 */
```

```
 45 /* USER CODE BEGIN PV */
 46 uint32_t pwm_deger=0;
 47 /* USER CODE END PV */
```

- SetPwm komutu ile değişkene değer göndererek atama yapıyoruz.

```
106   /* USER CODE BEGIN WHILE */
107   while (1)
108   {
109     /* USER CODE END WHILE */
110
111     /* USER CODE BEGIN 3 */
112         SetPwm(pwm_deger);
113         pwm_deger++;
114         if( pwm_deger>100)
115         {
116             pwm_deger=0;
117         }
118         HAL_Delay(100);
119   }
120   /* USER CODE END 3 */
```

@cengizhantopcu53

https://linktr.ee/cengizhantopcu53