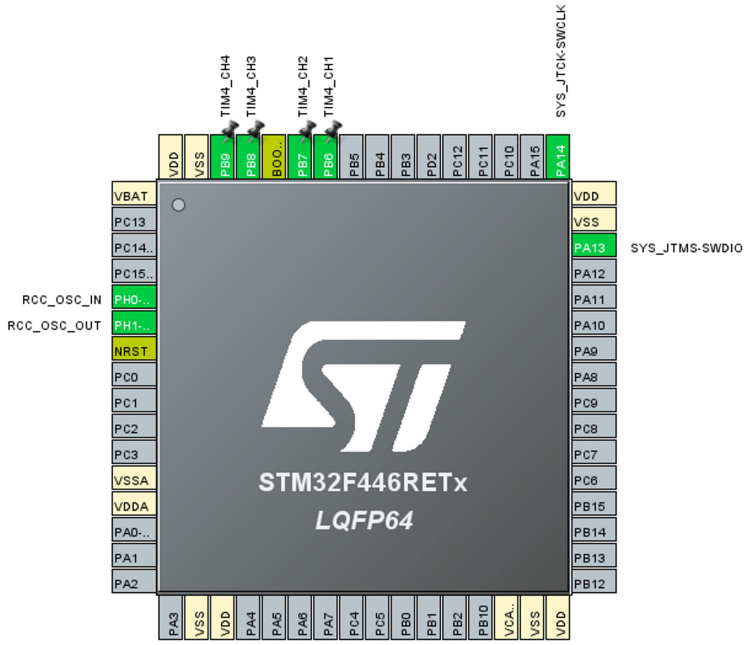
PWM Kullanımı

25 Aralık 2021 Cumartesi 00:59

PWM Kullanımı

- HAL

Konfigürasyon Kısmı



Pin Na...	Signal on Pin	GPIO outpu...	GPIO mode	GPIO Pull-u...	Maximum o...	User Label	Modified
PB6	TIM4_CH1	n/a	Alternate F...	No pull-up a...	Low		☐
PB7	TIM4_CH2	n/a	Alternate F...	No pull-up a...	Low		☐
PB8	TIM4_CH3	n/a	Alternate F...	No pull-up a...	Low		☐
PB9	TIM4_CH4	n/a	Alternate F...	No pull-up a...	Low		☐

- Timers kısmından TIM4 seçimi yapılır. Ardından Mod kısmından kanal seçimi yapılır.

Slave Mode: Disable
Trigger Source: Disable
☐ Internal Clock
Channel1: PWM Generation CH1
Channel2: PWM Generation CH2
Channel3: PWM Generation CH3
Channel4: PWM Generation CH4
Combined Channels: Disable
☐ XOR activation
☐ One Pulse Mode

- Period değerimize göre kanal çıkışlarına Pulse değeri yazacağız.
- Period kısmını Duty Cycle en fazla 100 olduğundan Period kısmına 100-1 olarak gireriz. Yani Period 100 ve Pulse değeri 50 girersek aslında %50 Duty Cycle olur.
- 10kHZ için işlem sonucunda Prescaler 90000 girilir.
- 1 kHZ=1000Hz

$$UpdateEvent = \frac{90.000.000}{(Prescaler + 1)(100)} = 10000\,Hz = 10\,kHz$$

$$Prescaler + 1 = 90$$

Counter Settings	
Prescaler (PSC - 16 bits value)	90-1
Counter Mode	Up
Counter Period (AutoReload Register - 16 b.	100-1
Internal Clock Division (CKD)	No Division
auto-reload preload	Disable
PWM Generation Channel 1	
Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High
PWM Generation Channel 2	
Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High
PWM Generation Channel 3	
Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High
PWM Generation Channel 4	
Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High

- Mode 1 ile Mode 2 arasındaki fark High ile Low durumların tersi olmasıydı.
- Mode 1 durumunda kanal %25 ise %25'te High %75'te Low oluyordu.
  %75 olduğunda %25'te Low %75'te High oluyor.
- Fast Mode durumun Disable olduğu zaman saymaya yaparken üst limite kadar saydıktan sonra 0'a doğru azala azala inerken Enabled olduğu zaman 0'a doğru birden iner.

Kod Kısmı

- hal_tim.c dosyasından Timer'ı PWM ile başlatmamız gerekiyor.

```
1439 /**
1440   * @brief  Starts the PWM signal generation.
1441   * @param  htim TIM handle
1442   * @param  Channel TIM Channels to be enabled
1443   *          This parameter can be one of the following values:
1444   *            @arg TIM_CHANNEL_1: TIM Channel 1 selected
1445   *            @arg TIM_CHANNEL_2: TIM Channel 2 selected
1446   *            @arg TIM_CHANNEL_3: TIM Channel 3 selected
1447   *            @arg TIM_CHANNEL_4: TIM Channel 4 selected
1448   * @retval HAL status
1449   */
1450 HAL_StatusTypeDef HAL_TIM_PWM_Start(TIM_HandleTypeDef *htim, uint32_t Channel)
```

```
91   /* USER CODE BEGIN 2 */
92   HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
93   HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_2);
94   HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_3);
95   HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_4);
96   /* USER CODE END 2 */
```

```
45 /* USER CODE BEGIN PV */
46 int i;
47 /* USER CODE END PV */
```

```
/**
  * @brief  Set the TIM Capture Compare Register value on runtime without calling another time ConfigChannel function.
  * @param  __HANDLE__ TIM handle.
  * @param  __CHANNEL__ TIM Channels to be configured.
  *          This parameter can be one of the following values:
  *            @arg TIM_CHANNEL_1: TIM Channel 1 selected
  *            @arg TIM_CHANNEL_2: TIM Channel 2 selected
  *            @arg TIM_CHANNEL_3: TIM Channel 3 selected
  *            @arg TIM_CHANNEL_4: TIM Channel 4 selected
  * @param  __COMPARE__ specifies the Capture Compare register new value.
  * @retval None
  */
#define __HAL_TIM_SET_COMPARE(__HANDLE__, __CHANNEL__, __COMPARE__) \
  (((__CHANNEL__) == TIM_CHANNEL_1) ? ((__HANDLE__)->Instance->CCR1 = (__COMPARE__)) :\
   ((__CHANNEL__) == TIM_CHANNEL_2) ? ((__HANDLE__)->Instance->CCR2 = (__COMPARE__)) :\
   ((__CHANNEL__) == TIM_CHANNEL_3) ? ((__HANDLE__)->Instance->CCR3 = (__COMPARE__)) :\
   ((__HANDLE__)->Instance->CCR4 = (__COMPARE__)))
```

- Compare kısmına Pulse değerini yazıyoruz.

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

    /* USER CODE BEGIN 3 */
      for(i=0;i<=1999;i++)
      {
          __HAL_TIM_SET_COMPARE(&htim4,TIM_CHANNEL_1,i);
          __HAL_TIM_SET_COMPARE(&htim4,TIM_CHANNEL_2,i);
          __HAL_TIM_SET_COMPARE(&htim4,TIM_CHANNEL_3,i);
          __HAL_TIM_SET_COMPARE(&htim4,TIM_CHANNEL_4,i);
          HAL_Delay(100);
      }

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

➢ REGİSTER

Konfigürasyon Kısmı

- RCC için yazdığımız fonksiyonlar aşağıdaki gibidir.

```
void RCC_Config(void)
{
    RCC->CR |= 0x00010000;              //HSEON
    while(!(RCC->CR & 0x00020000));     //HSERDY
    RCC->CR |= 0x00080000;              //CSSON
    RCC->CFGR = 0x00000000;
    RCC->PLLCFGR |= 0x00400000;         //PLLSRC
    RCC->PLLCFGR |= 0x00000004;         //PLLM 4
    RCC->PLLCFGR |= 0x00002A00;         //PLLN 168
    RCC->PLLCFGR |= 0x00000000;         //PLLP 2
    RCC->CR |= 0x01000000;              //PLLON
    while(!(RCC->CR & 0x02000000));     //PLLRDY
    RCC->CFGR |= 0x00000001;            //SW
    while(!(RCC->CR & 0x00000001));     //SWS
    RCC->CFGR |= 0x00000000;            //HPRE AHB 1
    RCC->CFGR |= 0x00001400;            //PPRE1 APB1 4
    RCC->CFGR |= 0x00008000;            //PPRE2 APB2 2
    RCC->CIR |= 0x00080000;             //HSERDYC
    RCC->CIR |= 0x00800000;             //CSSC
}
```

- GPIO için çıkışlarını alternatif fonksiyon yapıyoruz.

GPIO port mode register (GPIOx_MODER) (x = A..I/J/K)

Address offset: 0x00

Reset values:

- 0xA800 0000 for port A
- 0x0000 0280 for port B
- 0x0000 0000 for other ports

GPIO port mode register (GPIOx_MODER) (x = A..I/J/K)

Address offset: 0x00

Reset values:

- 0xA800 0000 for port A
- 0x0000 0280 for port B
- 0x0000 0000 for other ports

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- Kart üzerindeki ledleri kullanıyoruz. 24., 26., 28. ve 30.biti 1 yapıyoruz.

Bits 2y:2y+1 **MODERy[1:0]:** Port x configuration bits (y = 0..15)
These bits are written by software to configure the I/O direction mode.
00: Input (reset state)
01: General purpose output mode
10: Alternate function mode
11: Analog mode

```
GPIOD->MODER |= 2 << 24 | 2 << 26 | 2 << 28 | 2 << 30;
```

- Alternatif fonksiyon ile kast edilen pinin çevresel birimlerden I2C, SPI olarak kullanılacağını belirtiyor. Biz burada TIM4 için kullanacağımızı belirteceğiz.

GPIO alternate function low register (GPIOx_AFRL) (x = A..I/J/K)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFRL7[3:0]				AFRL6[3:0]				AFRL5[3:0]				AFRL4[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFRL3[3:0]				AFRL2[3:0]				AFRL1[3:0]				AFRL0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

GPIO alternate function high register (GPIOx_AFRH) (x = A..I/J)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFRH15[3:0]				AFRH14[3:0]				AFRH13[3:0]				AFRH12[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFRH11[3:0]				AFRH10[3:0]				AFRH9[3:0]				AFRH8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- Low ile 0 ile 7.pinler arası iken high 8 ile 15.pinler arasıdır. Biz 12, 13,14 ve 15. pinleri kullandığımızdan high olanı kullanıyoruz.

Bits 31:0 **AFRHy:** Alternate function selection for port x bit y (y = 8..15)
These bits are written by software to configure alternate function I/Os

AFRHy selection:

0000: AF0	1000: AF8
0001: AF1	1001: AF9
0010: AF2	1010: AF10
0011: AF3	1011: AF11
0100: AF4	1100: AF12
0101: AF5	1101: AF13
0110: AF6	1110: AF14

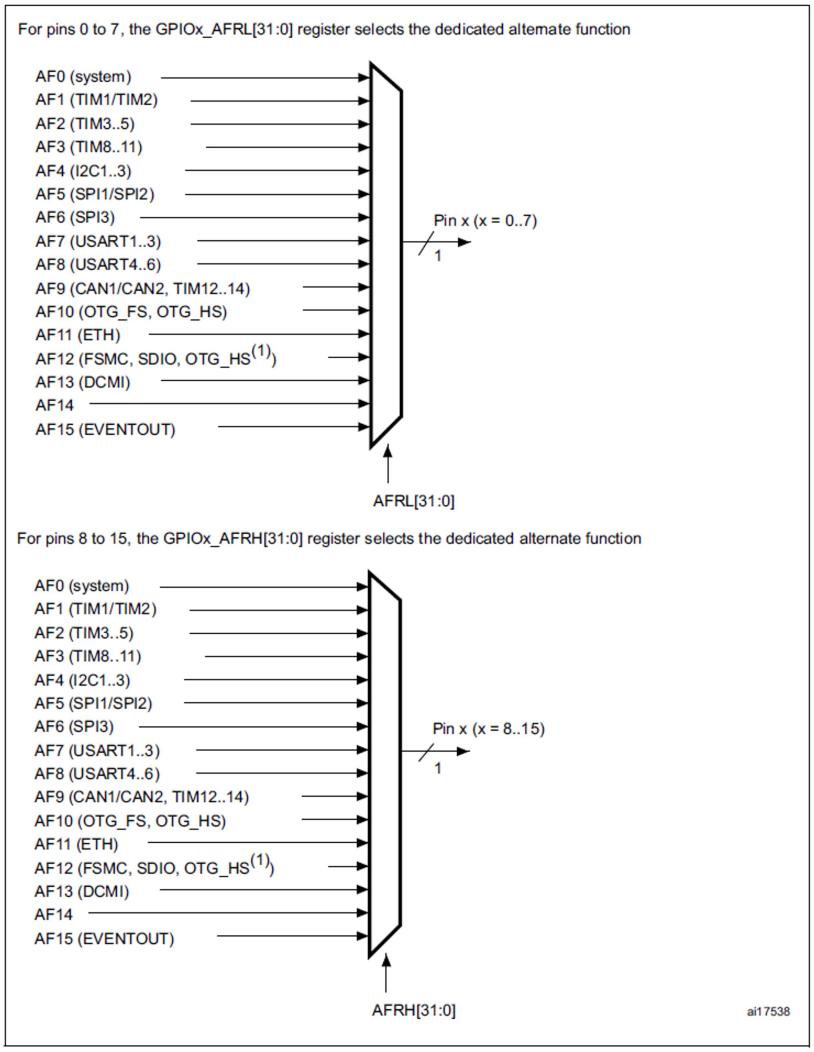
Bits 31:0 **AFRHy:** Alternate function selection for port x bit y (y = 8..15)
These bits are written by software to configure alternate function I/Os

AFRHy selection:

0000: AF0	1000: AF8
0001: AF1	1001: AF9
0010: AF2	1010: AF10
0011: AF3	1011: AF11
0100: AF4	1100: AF12
0101: AF5	1101: AF13
0110: AF6	1110: AF14
0111: AF7	1111: AF15

- Seçtiğimiz TIM4 çevresel birimi hangi AF'de olduğunu bilmek için Reference Manuel kitapçığına bakıyoruz.

Selecting an alternate function



- TIM4, AF2'de bulunuyor. Böylece pinlere AF2 için olan 0010 tanımlaması yapacağız.
- AFR'nin High ve Low olduğunda belirtmemiz gerekiyor. AFR'ye Ctrl ile sağ tıklarız. AFR'nin parantez içinde 2 elemanlı dizi olduğunu gösterir. 0.eleman low, 1.eleman high temsil eder. Biz 1 yazıyoruz.

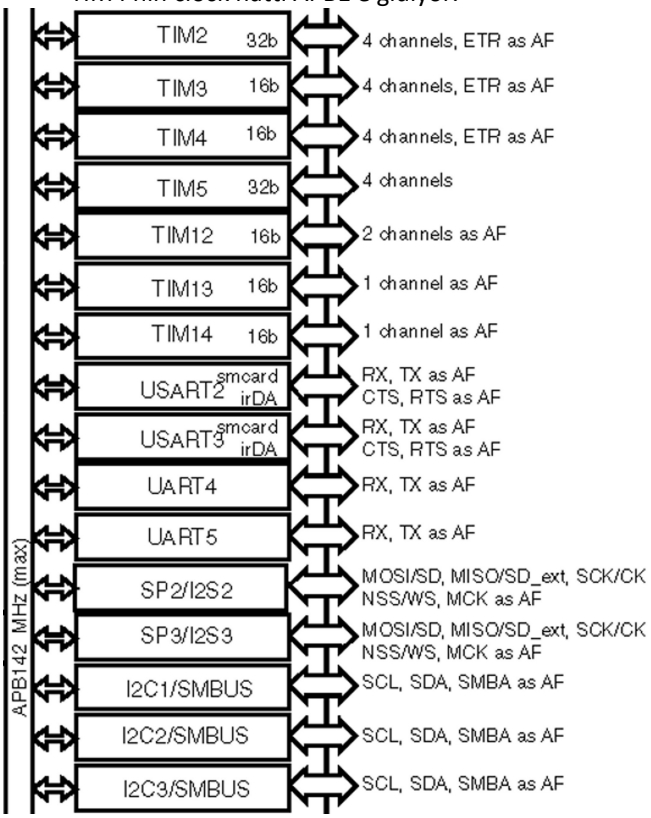
```
682   __IO uint32_t AFR[2];   /*!< GPIO alternate function registers,     Address offset: 0x20-0x24 */
```

```
GPIOD->AFR[1] |= 2 << 16 | 2 << 20 | 2 << 24 | 2 << 28;
```

- GPIO için yazdığımız fonksiyon aşağıdaki gibidir.

```
24 void GPIO_Config(void)
25 {
26     RCC->AHB1ENR |= 0x00000008;                                  //D clock enable
27
28     GPIOD->MODER |= 2 << 24 | 2 << 26 | 2 << 28 | 2 << 30;       //PD12, PD13, PD14, PD15
29     GPIOD->AFR[1] |= 2 << 16 | 2 << 20 | 2 << 24 | 2 << 28;      //TIM4 AFRH12, AFRH13, AFRH14, AFRH15
30     GPIOD->OTYPER |= 0x00000000;                                 //Output push-pull
31     GPIOD->OSPEEDR |= 0xFF000000;                                //Very high speed
32     GPIOD->PUPDR |= 0x00000000;                                  //No pull-up, pull-down
33 }
```

- TIM4'nin clock hattı APB1'e gidiyor.



RCC APB1 peripheral clock enable register (RCC_APB1ENR)

Address offset: 0x40

Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DAC EN	PWR EN	Reser-ved	CAN2 EN	CAN1 EN	Reser-ved	I2C3 EN	I2C2 EN	I2C1 EN	UART5 EN	UART4 EN	USART 3 EN	USART 2 EN	Reser-ved
		rw	rw		rw	rw		rw	rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Reserved		WWDG EN	Reserved		TIM14 EN	TIM13 EN	TIM12 EN	TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN
rw	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw

- 2.biti 1 yapıyoruz.

Bit 2 **TIM4EN:** TIM4 clock enable
Set and cleared by software.
0: TIM4 clock disabled
1: TIM4 clock enabled

```
RCC->APB1ENR |= 0x00000004;          //TIM4EN
```

TIMx control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- Sayma yapacağımdan saymayı başlatmak için 0.biti aktif ediyoruz. Sayma başlayacağından bu işlem fonksiyonun en son satırında olmalı.

Bit 0 **CEN**: Counter enable

0: Counter disabled
1: Counter enabled

Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.

CEN is cleared automatically in one-pulse mode, when an update event occurs.

```
TIM4->CR1 |= 1 << 0;
```

- Saymanın yönünü belirliyoruz. Biz yukarı doğru saymasını istiyoruz.

Bit 4 **DIR**: Direction

0: Counter used as upcounter
1: Counter used as downcounter

Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.

```
TIM4->CR1 |= 0 << 4;
```

- 5. ve 6.biti 0 yapıyoruz.

Bits 6:5 **CMS**: Center-aligned mode selection

00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR).
01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting down.
10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting up.
11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set both when the counter is counting up or down.

Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1)

```
TIM4->CR1 |= 0 << 5;
```

- Sistem clock 168MHz ayarlamıştık ve TIM4 clock veri yolu ise 42MHz'dir fakat bunun 2 katı değeri alıyorlardı yani 84MHz'dir. Biz bunu bölmek istiyorsak ayarlayabiliyoruz. Biz bölmüyoruz.

Bits 9:8 **CKD**: Clock division

This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, TIx),

00: $t_{DTS} = t_{CK\_INT}$
01: $t_{DTS} = 2 \times t_{CK\_INT}$
10: $t_{DTS} = 4 \times t_{CK\_INT}$
11: Reserved

- 8. ve 9.biti 0 yapıyoruz.

```
TIM4->CR1 |= 0 << 8;
```

- PWM aslında capture/compare mode kısmına giriyor.

TIMx capture/compare mode register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. Take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- 1.kanal için output yapıyoruz yani ilk 2 bit 0 yapıyoruz.

Bits 1:0 **CC1S**: Capture/Compare 1 selection

This bit-field defines the direction of the channel (input/output) as well as the used input.
00: CC1 channel is configured as output.
01: CC1 channel is configured as input, IC1 is mapped on TI1.
10: CC1 channel is configured as input, IC1 is mapped on TI2.
11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register)

Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

- Mod olarak PWM mode 1 seçiyoruz. Bunun için bite 110 yazıyoruz.

Bits 6:4 **OC1M**: Output compare 1 mode

These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits.
000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs.(this mode is used to generate a timing base).
001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1).
010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1).
011: Toggle - OC1REF toggles when TIMx_CNT=TIMx_CCR1.
100: Force inactive level - OC1REF is forced low.
101: Force active level - OC1REF is forced high.
110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT<TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0) as long as TIMx_CNT>TIMx_CCR1 else active (OC1REF=1).
111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 else inactive.

Note: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode.

- 2.kanal için output yapıyoruz. 8. ve 9.biti 0 yapıyoruz.

Bits 9:8 **CC2S[1:0]**: Capture/Compare 2 selection

This bit-field defines the direction of the channel (input/output) as well as the used input.
00: CC2 channel is configured as output
01: CC2 channel is configured as input, IC2 is mapped on TI2
10: CC2 channel is configured as input, IC2 is mapped on TI1
11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register)

Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIMx_CCER).

- 2.kana'lın modunu 1.kanal'da yaptığımız gibi PWM Mode 1 yapıyoruz.

Bits 14:12 **OC2M[2:0]**: Output compare 2 mode

```
TIM4->CCMR1 |= 0 << 0 | 6 << 4 | 0 << 8 | 6 << 12;
```

TIMx capture/compare mode register 2 (TIMx_CCMR2)

Address offset: 0x1C

Reset value: 0x0000

Refer to the above CCMR1 register description.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- CCMR2 ile bu sefer kanal 3 ve 4 için yapıyoruz. Kanal 1 ve 2 ile yaptıklarımızın aynısını yapıyoruz.

Bits 1:0 **CC3S**: Capture/Compare 3 selection

This bit-field defines the direction of the channel (input/output) as well as the used input.
00: CC3 channel is configured as output
01: CC3 channel is configured as input, IC3 is mapped on TI3
10: CC3 channel is configured as input, IC3 is mapped on TI4
11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register)

Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

Bits 6:4 **OC3M**: Output compare 3 mode

Bits 9:8 **CC4S**: Capture/Compare 4 selection

This bit-field defines the direction of the channel (input/output) as well as the used input.
00: CC4 channel is configured as output
01: CC4 channel is configured as input, IC4 is mapped on TI4
10: CC4 channel is configured as input, IC4 is mapped on TI3
11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register)

Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIMx_CCER).

Bits 14:12 **OC4M**: Output compare 4 mode

```
TIM4->CCMR2 |= 0 << 0 | 6 << 4 | 0 << 8 | 6 << 12;
```

TIMx capture/compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
rw		rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

- Çıkışları Enabled yapıyoruz.

0., 4., 8. ve 12. bitleri 1 yapıyoruz.

Bit 0 **CC1E**: *Capture/Compare 1 output enable.*

CC1 channel configured as output:
0: Off - OC1 is not active
1: On - OC1 signal is output on the corresponding output pin

CC1 channel configured as input:
This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not.
0: Capture disabled
1: Capture enabled

Bit 4 **CC2E**: *Capture/Compare 2 output enable.*

refer to CC1E description

Bit 8 **CC3E**: *Capture/Compare 3 output enable.*

refer to CC1E description

Bit 12 **CC4E**: *Capture/Compare 4 output enable.*

refer to CC1E description

```
TIM4->CCER |= 1 << 0 | 1 << 4 | 1 << 8 | 1 << 12;
```

TIMx prescaler (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- Prescalaer bizim sayısının en üst seviyesini belirler.
- Sayma işlemi 0'dan başlamayıp 1'den başladığından 1 eksiğini alarak yazarız.
- TIM4 clock hızı 84MHz olduğundan bunu kaça bölmeliyim diye soruyoruz. Bu clock hız için 42000'e bölüyoruz. Bu sayı da 41999 sayısı yapıyor.

Bits 15:0 **PSC[15:0]**: Prescaler value

The counter clock frequency CK_CNT is equal to f_{CK_PSC} / (PSC[15:0] + 1).
PSC contains the value to be loaded in the active prescaler register at each update event

Bits 15:0 **PSC[15:0]**: Prescaler value

The counter clock frequency CK_CNT is equal to f_{CK_PSC} / (PSC[15:0] + 1).
PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in "reset mode").

```
TIM4->PSC |= 41999;
```

- 84MHz'i 42000'e böldüğümüzde 2000 sayısı yapıyor. Bu değer auto-reload oluyor. Bunun 1 eksiğini yazıyoruz.

TIMx auto-reload register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16] (depending on timers)															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:16 **ARR[31:16]**: High auto-reload value (on TIM2 and TIM5).

Bits 15:0 **ARR[15:0]**: Auto-reload value
ARR is the value to be loaded in the actual auto-reload register.
Refer to the *Section 18.3.1: Time-base unit* for more details about ARR update and behavior.
The counter is blocked while the auto-reload value is null.

```
TIM4->ARR |= 1999;
```

- Böylece 1 sn'de 2000'e kadar sayıyor.
- Pinlere atayacağımız paslar CCR1 12.pin, CCR2 13.pin, CCR3 14.pin ve CCR4 15.pin içindir.

TIMx capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16] (depending on timers)															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

TIMx capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2[31:16] (depending on timers)															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

TIMx capture/compare register 3 (TIMx_CCR3)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3[31:16] (depending on timers)															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

TIMx capture/compare register 4 (TIMx_CCR4)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[31:16] (depending on timers)															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro	rw/ro

- Bunlara atayacağımız değerler en fazla Auto-reload değeri kadar olabilir.

```
TIM4->CCR1 |= 500;
TIM4->CCR2 |= 1000;
TIM4->CCR3 |= 1500;
TIM4->CCR4 |= 1999;
```

- TIM için yazdığımız fonksiyonlar aşağıdaki gibidir.

```
35 void TIM_Config(void)
36 {
37     RCC->APB1ENR |= 0x00000004;                          //TIM4EN
38
39     TIM4->CR1 |= 0 << 4;                                 //DIR Counter used as up counter
40     TIM4->CR1 |= 0 << 5;                                 //CMS Edge-aligned mode
41     TIM4->CR1 |= 0 << 8;                                 //tDTS = tCK_INT 84MHz
42     TIM4->CCMR1 |= 0 << 0 | 6 << 4 | 0 << 8 | 6 << 12;   //Capture/Compare selected output, PWM mode 1 (1 & 2)
43     TIM4->CCMR2 |= 0 << 0 | 6 << 4 | 0 << 8 | 6 << 12;   //output, PWM mode 1 selected (3 & 4)
44     TIM4->CCER |= 1 << 0 | 1 << 4 | 1 << 8 | 1 << 12;    //output enable (1, 2, 3 & 4 )
45     TIM4->PSC |= 41999;                                  //PSC Prescaler value
46     TIM4->ARR |= 1999;                                   //ARR Auto-reload value
47     TIM4->CCR1 |= 500;                                   //PD12
48     TIM4->CCR2 |= 1000;                                  //PD13
49     TIM4->CCR3 |= 1500;                                  //PD14
50     TIM4->CCR4 |= 1999;                                  //PD15
51     TIM4->CR1 |= 1 << 0;                                 //CEN Counter enabled
52 }
```

Kod Kısmı

```
54 int main(void)
55 {
56     RCC_Config();
57     GPIO_Config();
58     TIM_Config();
59
60   while (1)
61   {
62
63   }
64 }
```

- Pindeki ledlerin durumu 12.pin %25, 13.pin %50, 14.pin %75, 15.pin %100 parlaklıkta yanıyor.