Timers Low Layer

Hardware : SoC Stm32f4

Contact : Mustapha.hamdi@insat.rnu.tn

Institut national des sciences appliquées et de technologie, Tunisia

April 2017

Powered by STM32

References:

- AN4013 Application note :STM32 cross-series timer overview
- AN4776 Application note General-purpose timer cookbook
- ST onlinetraining : https://st-onlinetraining.s3.amazonaws.com/STM32L4_WDG_TIMERS_General_Purpose_Timer_(GPTIM)/index.html
- RM0090 Reference manual

Mustapha.hamdi@insat.rnu.tn

TIMERs

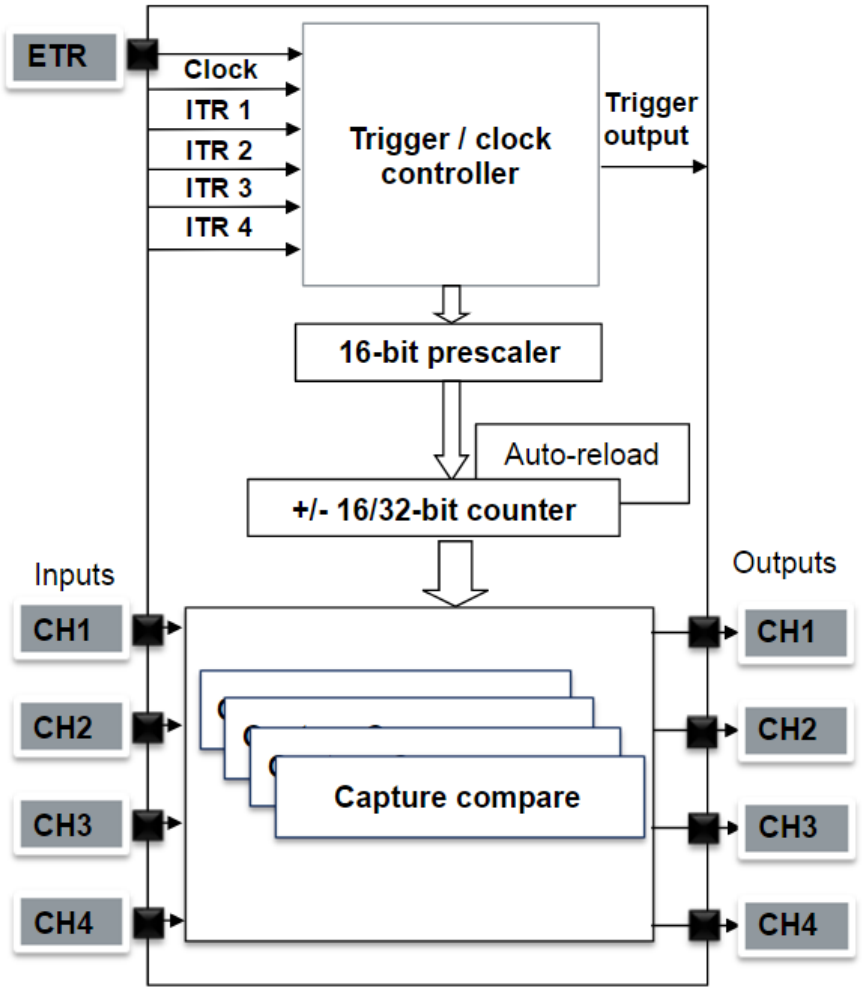
All the STM32 general-purpose timer peripherals share the same backbone structure. This section tears down the advanced configuration TIM1 timer peripheral, which is the timer peripheral with the most features. Next figure shows the block diagram for the TIM1 timer peripheral.

The STM32 timer peripheral is made by the assembly of four units:

1. Clock source
2. A prescaler
3. Resolution (autoreload)
4. Capture/compare function.

Mustapha.hamdi@insat.rnu.tn

Timer-peripheral block diagram



Mustapha.hamdi@insat.rnu.tn

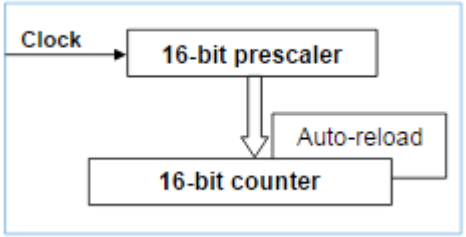
TIMERs

The timers are based on the some architecture and scalable in terms of:

- Number of inputs/outputs
- Resolution (16 or 32 bits)
- Features (PWM, DMA, counting..)

Each timer embeds a 16-bit linear prescaler (1,2,3… 65536)

- An Auto-reload register defines the counting period
- An update event (interrupt or DMA)



Prescaler and auto-reload parameters are configured by respectively, PSC and ARR register) and used for example for high resolution PWM configuration

Mustapha.hamdi@insat.rnu.tn

TIMERs

All STM32 timers have independent I/O channels. These can be used as follows:

- Input Capture
- Output Compare or PWM
- One Pulse Mode

Timers can be clocked by:

- Internal Clock
- External Clock Sources
  - External Mode 1 (TI1 and TI2 pins)
  - External Mode 2 (ETR pin)
  - Internal Trigger (ITRx)

Interrupt and DMA events occur when the followings occur:

- Update
  - Counter overflow/underflow
  - Counter initialized
  - Others
- Trigger (For ADC event for example)
  - Counter Start
  - Counter Stop
  - Counter Initialize
  - Others
- Input Capture / Output Compare
- Others

Mustapha.hamdi@insat.rnu.tn

TIMERS

The table below summarizes some capabilities of STM32 timers:

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
	TIM13, TIM14	16-bit	Up	Any integer between 1 and 65536	No	1	No	42	84
Basic	TIM6, TIM7	16-bit	Up	Any integer between 1 and 65536	Yes	0	No	42	84

Mustapha.hamdi@insat.rnu.tn

PWM mode

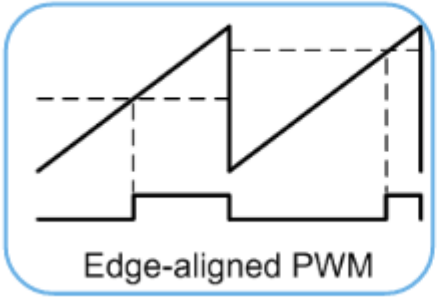
PWM is a way of digitally modifying analog signal levels. Through the use of high resolution (autoreload) the duty cycle of a square wave is modulated to change a specific analog signal level.

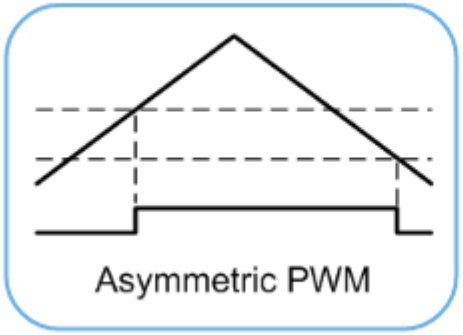
The stm32 timer is able to generate PWM in edge-aligned mode or in center-aligned mode with a frequency determined by the value of the TIMx_ARR register, and a duty cycle determined by the value of the TIMx_CCRx register.

Mustapha.hamdi@insat.rnu.tn

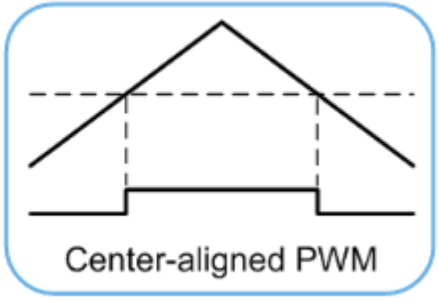
PWM mode

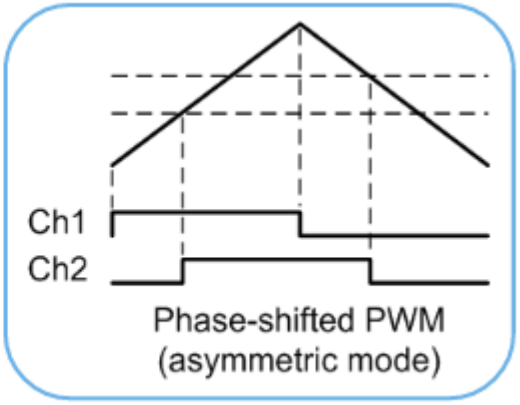
- Basic PWM, edge or center-aligned





- Asymmetric center-aligned PWM





- More information's here :ST onlinetraining : https://st-onlinetraining.s3.amazonaws.com/STM32L4_WDG_TIMERS_General_Purpose_Timer_(GPTIM)/index.html

Mustapha.hamdi@insat.rnu.tn

PWM mode

Timer Period:

The timer can be used as a time base generator.
Depending on the clock, prescaler, auto reload coefficient and repetition counter (if present) parameters, the 16-bit timer can generate an update event from a **nanosecond** to a few minutes. For the 32-bit timer, the range is larger.

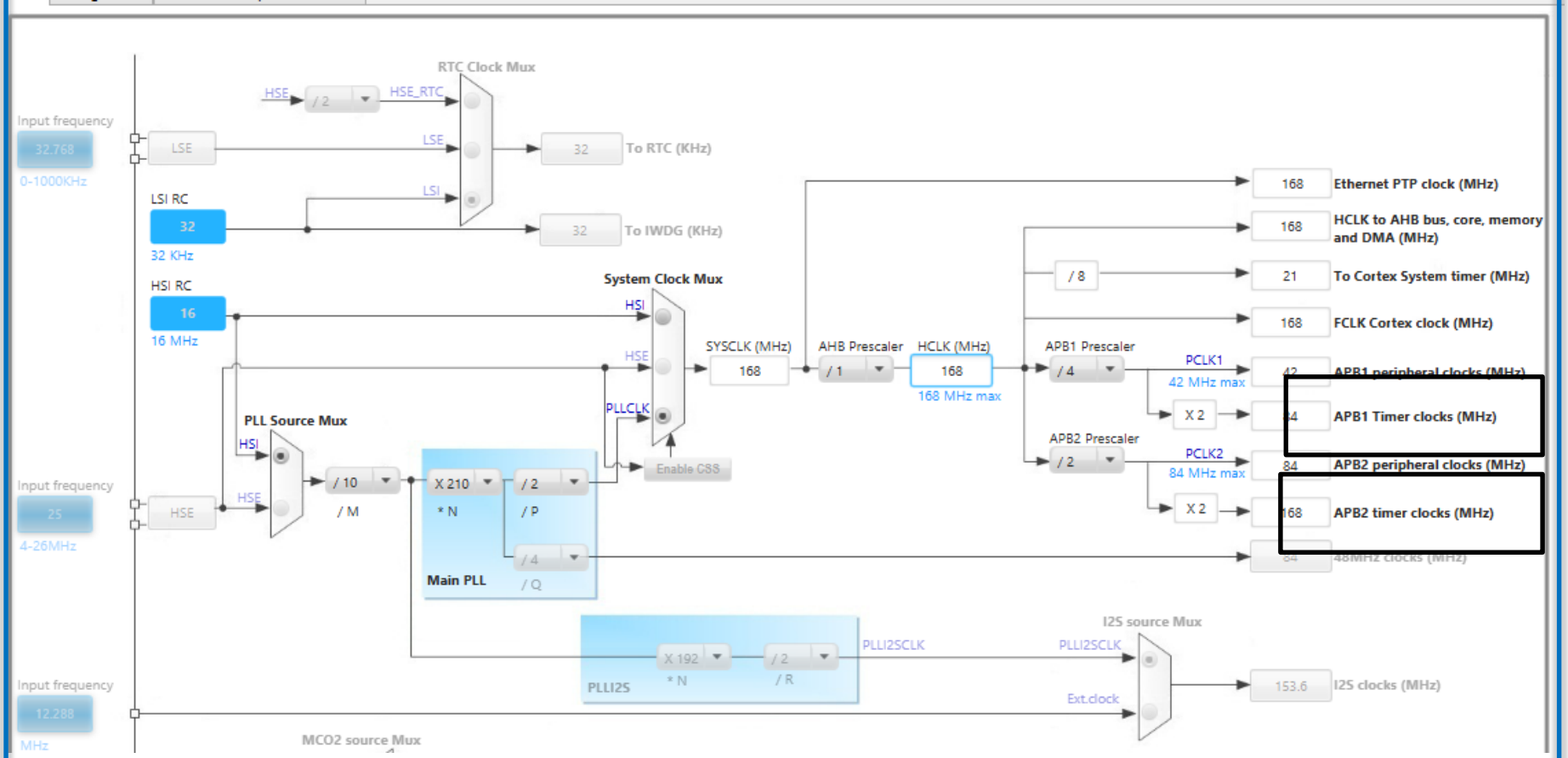
Example update event period :

The update event period is calculated as follows:
Update_event = TIM_CLK/((PSC + 1)*(ARR + 1)*(RCR + 1))
TIM_CLK = 72 MHz, Prescaler = 1 , Auto reload = 65535 and No repetition counter RCR = 0
Update_event = 72*(10^6)/((1 + 1)*(65535 + 1)*(1))
Update_event = 549.3 Hz.

Mustapha.hamdi@insat.rnu.tn

Stm32 System Clock

GUI Clock configuration from CubeMX



Mustapha.hamdi@insat.rnu.tn

Case Study : Timer 3 as PWM

1. Select Timer and outputs
   - Ex. TIM3-Ch1 (gpioC6)…TIM3-ch4(gpioC9)
2. GPIO initialization (AF mode)
3. Alternate Function config (ex. TIMER3 as AF)

Mustapha.hamdi@insat.rnu.tn

Case Study : Timer 3 as PWM

4
- Timer initialization : Clock APB1ENR, Prescaler

5
- PWM configuration

6
- Duty cycle definition
- TIM3->CCR1, …. TIM3->CCR4

Mustapha.hamdi@insat.rnu.tn

Registers

GPIO configuration

clock	AHB1ENR
AF mode	MODER
Max freq	OSPEEDER
Config AF	AF[0] AF[1]

Mustapha.hamdi@insat.rnu.tn

Timer configuration

Clock	APBx
Max timer Clock prescaler	PSC
Timer auoreload	ARR
Update generation	EGR
Config channels as pwm	CCMR1,2

CR1	CCER	CCR1,2,,,
Counter enable	**Enable PWM1,PWM2,PWM3,andPWM4**	**Duty cycle**

Mustapha.hamdi@insat.rnu.tn

GPIO configuration

Alternate functions

Pins of microcontroller offer many extra different functions. The conventional function would refer to GPIO, *General Purpose Input/Output*. In that case, pins can be used directly by writing to and reading from the relevant registers such as Input Data register IDR and Output Data Register ODR.

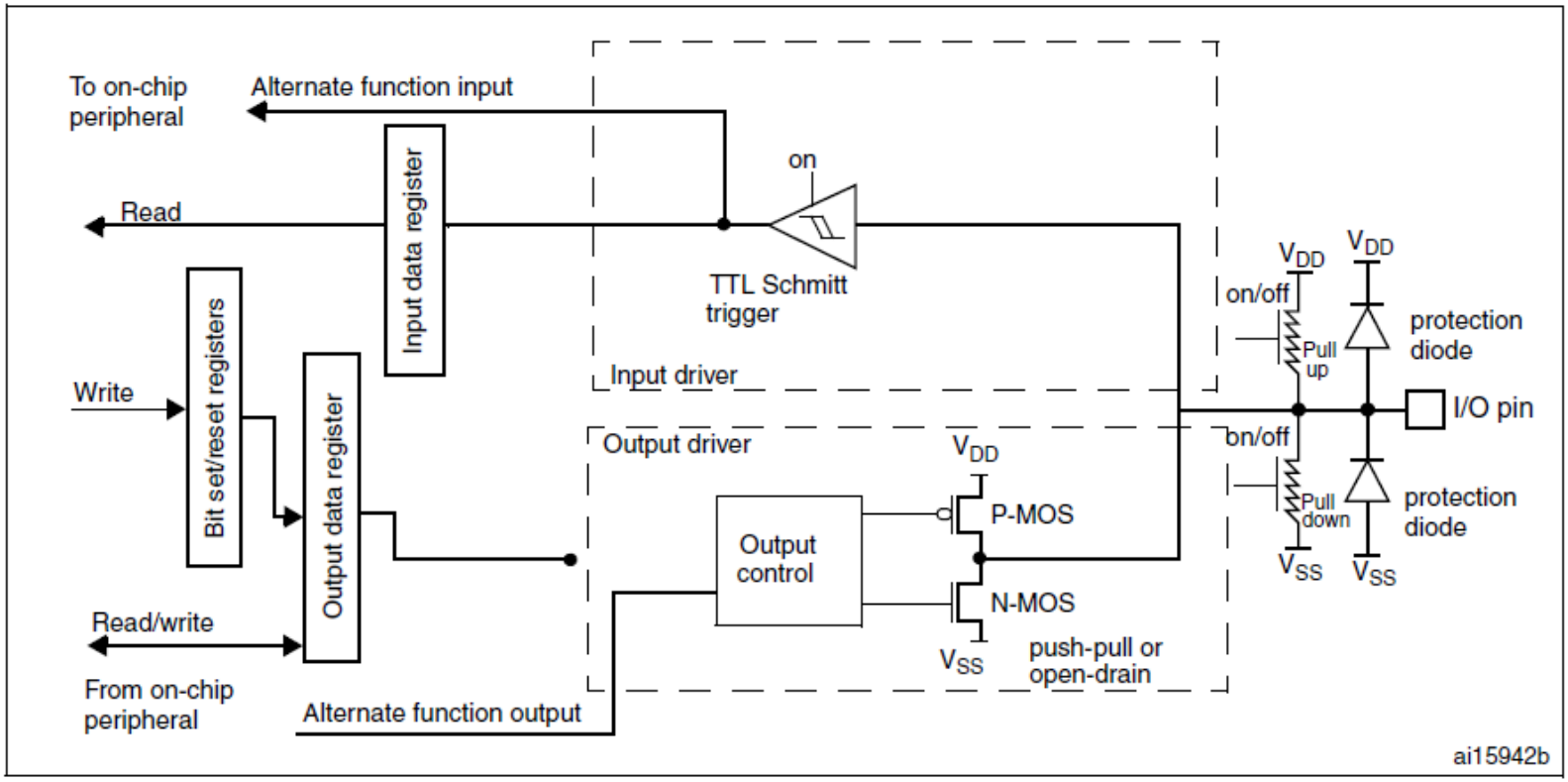
Alternate functions would refer to other 'extra' functions, that may include for example I^2C, SPI, USART, CCP, PWM, Clock, ADC.

Mustapha.hamdi@insat.rnu.tn

GPIO configuration

Alternate functions

Figure 30. Alternate function configuration



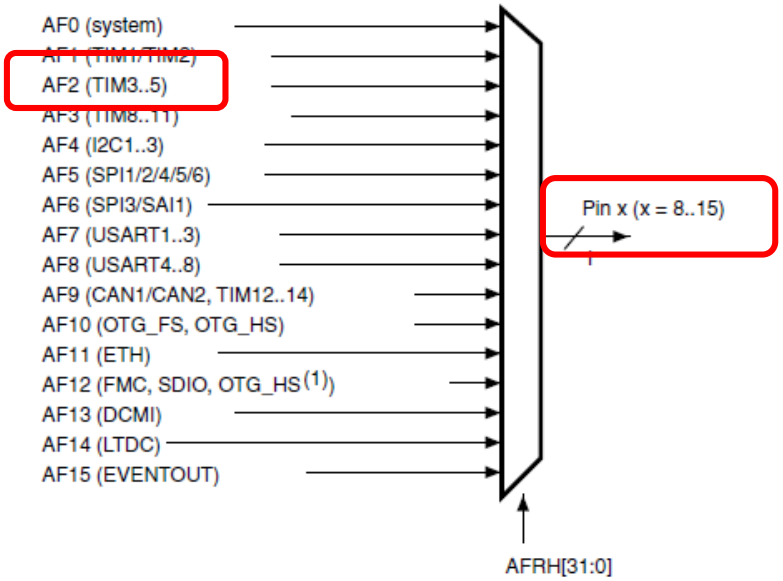
GPIO configuration

Alternate functions:

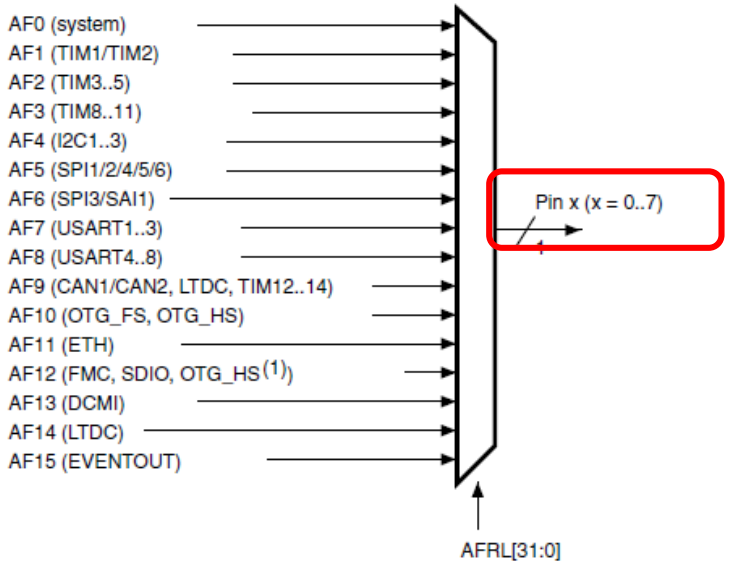
AF1: TIM1/TIM2, AF2: TIM3..5, AF9: CAN1/CAN2,TIM12

Each AF requires 4 bits config via AFRL (for pin0->7) and AFRH (for pin 8->14) registres.

For pins 8 to 15, the GPIOx_AFRH[31:0] register selects the dedicated alternate function



For pins 0 to 7, the GPIOx_AFRL[31:0] register selects the dedicated alternate function



GPIO configuration

Alternate functions AFRL

8.4.9 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

Bits 31:0 **AFRLy:** Alternate function selection for port x bit y (y = 0..7)

These bits are written by software to configure alternate function I/Os

AFRLy selection:

0000: AF0
0001: AF1
0010: AF2 **TIM3..5**
0011: AF3
0100: AF4
0101: AF5
0110: AF6
0111: AF7
1000: AF8
1001: AF9
1010: AF10
1011: AF11
1100: AF12
1101: AF13
1110: AF14
1111: AF15

```
GPIOC->AF[0] |= (2<<4*6)
;// PC6 as TIM3 AF
```

Alternate functions

Register AF[1]: AFH for pin 8->15
Register AF[0]: AFL for pin 0->7

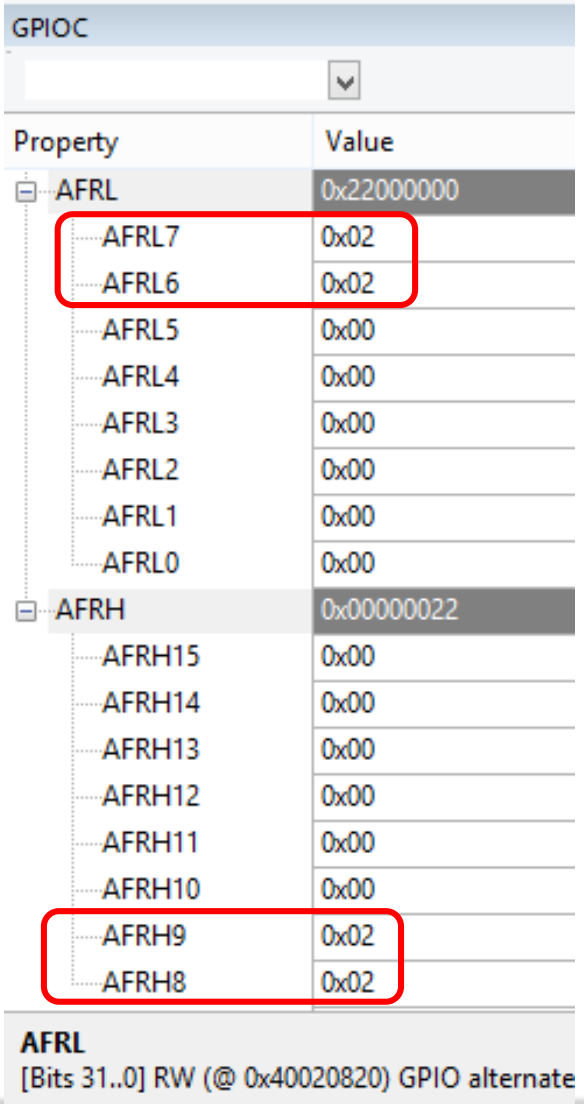
```
GPIOC->AF[0] |=(2<<4*6)+(2<<4*7);  //   pin 6 and 7
GPIOC->AF[1] [=(2<<4*0)+;(2<<4*1); //  pin 8 and 9
```


Mustapha.hamdi@insat.rnu.tn

Alternate functions

Keil MDK debug session :

PC6,PC7,PC8 and PC7 : TIM3 AF



Mustapha.hamdi@insat.rnu.tn

Timer configuration

6.3.13 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
UART8 EN	UART7 EN	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	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

Mustapha.hamdi@insat.rnu.tn

Timer configuration

18.4.11 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

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").

```
TIM3->PSC = 100;        // Prescaler FClockmax/100 (Set Frequency PWM)
                        // FClockMax Timer3= 84 MHz
```

Timer configuration

18.4.12 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 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.

```
TIM3->ARR = 1000;      // Counter/Resolution/(Set Frequency PWM)
                       // PWM resolution =1000
```

Timer 3 Frequency : F = 84MHz/(Psc+1)(ARR+1)

Timer configuration

18.4.6 TIMx event generation register (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									w		w	w	w	w	w

Bit 0 UG: Update generation

This bit can be set by software, it is automatically cleared by hardware.

0: No action

1: Re-initialize the counter and generates an update of the registers

```
TIM3->EGR |= 1;        // Up Counter
```

Timer configuration

OC1M: Output compare 1 mode
110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNTTIMx_CCR1 else active (OC1REF=1).

18.4.7 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
0	**1**	**1**	**0**	**0**	**0**	**0**	**0**	0	1	1	0	0	0	0	0

=0x6060

Channel 2 Channel 1

TIM3->CCMR1 |= 0x6060; //Set PWM channel 1 and PWM Channel 2

Timer configuration

18.4.13 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

TIM3->CCR1 = 750; // Set Duty PWM1, ARR=1000 => Duty cycle = 75% for channel 1

Timer configuration

18.4.1 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

```
TIM3->CR1 |= 1;        // Enable TIM3
```

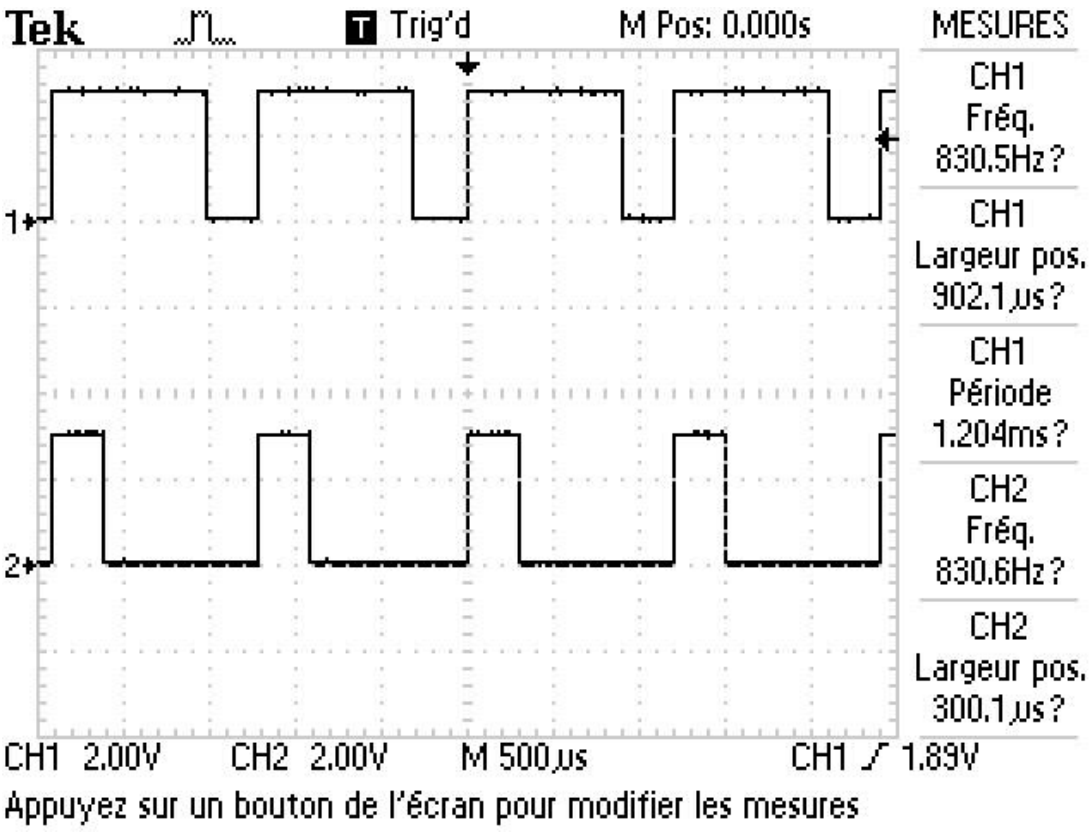
Timers

Timer Configuration code :

```
RCC->APB1ENR |= (1<<1); // Open clock Timer3
TIM3->PSC = 100;       // Prescale
TIM3->ARR = 1000;      // Autoreload : Counter Resolution

TIM3->EGR |= 1;        // Up Counter

//  Timer 3 Gen PWM  config

TIM3->CCMR1 |= 0x6060;  // Set PWM channel 1 and PWM Channel 2
TIM3->CCMR2 |= 0x6060;  // Set PWM channel 3 and PWM Channel 4

// Duty Cycle :

TIM3->CCR1 = 750;
TIM3->CCR2 = 250;
TIM3->CCR3 = 750;
TIM3->CCR4 = 100;

TIM3->CCER |= 0x1111;   // Enable PWM1,PWM2,PWM3,andPWM4

TIM3->CR1 |= 1;        // Enable TIM3
```

Timers

Timer 3 Frequency: F = 84MHz/(Psc+1)(ARR+1)=830,85 Hz, Period= 1,2 ms
Channel 1: Duty Cycle = 75% = 902 us
Channel 2: Duty Cycle = 25% = 300 us



Mustapha.hamdi@insat.rnu.tn