Advanced Timers

ECE 362
https://engineering.purdue.edu/ece362/

Reading Assignment

- Reading assignment:
  - Textbook, Chapter 15, “General-purpose Timers”, pages 373 – 414.
  - Family Reference Manual, Chapter 17, "General purpose timers (TIM2 and TIM3)", pages 377 – 443.

Future Reading Assignment

- Textbook, Chapter 22, Serial Communication Protocols, pp. 527 – 598
  - It's a long chapter.
  - Let's first look at Section 22.3, SPI, pp. 568–577.
  - Don't worry so much about the USB section.
    - Read that only if you're curious.
    - Your development board has no USB interface.
      - There are some USB provisions in the STM32F091, but would take much work.
    - Other books are better for understanding USB.

GPIO Alternate Function

- Recall that the 2-bit fields in the GPIOx_MODER can take on four values to define a pin mode:
  - 00: Pin is an input.
  - 01: Pin is an output.
  - 10: Pin uses an alternate function.
  - 11: Pin is an analog input.
- Today, we'll start using the alternate function codes.
  - Configured by four-bit fields in the GPIOx_AFRL and GPIOx_AFRH.
  - Datasheet, page 41, table 14, Alternate functions for Port A.
  - Datasheet, page 42, table 15, Alternate functions for Port B.
  - ...
- Configuring a pin for AF means you cannot use that pin for general purpose I/O.
  - See FRM, Figure 20, page 163.

To set GPIO alternate function

9.4.9 GPIO alternate function low register (GPIOx_AFRL) (x = A..F)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[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
AFR3[3:0]				AFR2[3:0]				AFR1[3:0]				AFR0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 **AFRy[3:0]:** Alternate function selection for port x pin y (y = 0..7)
These bits are written by software to configure alternate function I/Os

AFRy selection:

0000: AF0
0001: AF1
0010: AF2
0011: AF3
0100: AF4
0101: AF5
0110: AF6
0111: AF7
1000: Reserved
1001: Reserved
1010: Reserved
1011: Reserved
1100: Reserved
1101: Reserved
1110: Reserved
1111: Reserved

- Two 32-bit AFR registers per port named AFRL/AFRH.
- Four bits per pin.
  - e.g., to set PA0 to use timer 2, channel 1:
    - GPIOA->AFRL_AFR0[3:0] = 0010
  - e.g., to set PB14 to use timer 15, channel 1:
    - GPIOB->AFRH_AFR6[3:0] = 0001
    - (14 – 8 == 6)
  - Where did these numbers come from?
    - Tables 14 – 19

Port A alt funcs: Datasheet pg 41

Table 14. Alternate functions selected through GPIOA_AFR registers for port A

Pin name	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PA0	-	USART2_CTS	TIM2_CH1_ETR	TSC_G1_IO1	USART4_TX	-	-	COMP1_OUT
PA1	EVENTOUT	USART2_RTS	TIM2_CH2	TSC_G1_IO2	USART4_RX	TIM15_CH1N	-	-
PA2	TIM15_CH1	USART2_TX	TIM2_CH3	TSC_G1_IO3	-	-	-	COMP2_OUT
PA3	TIM15_CH2	USART2_RX	TIM2_CH4	TSC_G1_IO4	-	-	-	-
PA4	SPI1_NSS, I2S1_WS	USART2_CK	-	TSC_G2_IO1	TIM14_CH1	USART6_TX	-	-
PA5	SPI1_SCK, I2S1_CK	CEC	TIM2_CH1_ETR	TSC_G2_IO2	-	USART6_RX	-	-
PA6	SPI1_MISO, I2S1_MCK	TIM3_CH1	TIM1_BKIN	TSC_G2_IO3	USART3_CTS	TIM16_CH1	EVENTOUT	COMP1_OUT
PA7	SPI1_MOSI, I2S1_SD	TIM3_CH2	TIM1_CH1N	TSC_G2_IO4	TIM14_CH1	TIM17_CH1	EVENTOUT	COMP2_OUT
PA8	MCO	USART1_CK	TIM1_CH1	EVENTOUT	CRS_SYNC	-	-	-
PA9	TIM15_BKIN	USART1_TX	TIM1_CH2	TSC_G4_IO1	I2C1_SCL	MCO	-	-
PA10	TIM17_BKIN	USART1_RX	TIM1_CH3	TSC_G4_IO2	I2C1_SDA	-	-	-
PA11	EVENTOUT	USART1_CTS	TIM1_CH4	TSC_G4_IO3	CAN_RX	I2C2_SCL	-	COMP1_OUT
PA12	EVENTOUT	USART1_RTS	TIM1_ETR	TSC_G4_IO4	CAN_TX	I2C2_SDA	-	COMP2_OUT
PA13	SWDIO	IR_OUT	-	-	-	-	-	-
PA14	SWCLK	USART2_TX	-	-	-	-	-	-
PA15	SPI1_NSS, I2S1_WS	USART2_RX	TIM2_CH1_ETR	EVENTOUT	USART4_RTS	-	-	-

Port B alt funcs: Datasheet, pg 42

Table 15. Alternate functions selected through GPIOB_AFR registers for port B

Pin name	AF0	AF1	AF2	AF3	AF4	AF5
PB0	EVENTOUT	TIM3_CH3	TIM1_CH2N	TSC_G3_IO2	USART3_CK	-
PB1	TIM14_CH1	TIM3_CH4	TIM1_CH3N	TSC_G3_IO3	USART3_RTS	-
PB2	-	-	-	TSC_G3_IO4	-	-
PB3	SPI1_SCK, I2S1_CK	EVENTOUT	TIM2_CH2	TSC_G5_IO1	USART5_TX	-
PB4	SPI1_MISO, I2S1_MCK	TIM3_CH1	EVENTOUT	TSC_G5_IO2	USART5_RX	TIM17_BKIN
PB5	SPI1_MOSI, I2S1_SD	TIM3_CH2	TIM16_BKIN	I2C1_SMBA	USART5_CK_RTS	-
PB6	USART1_TX	I2C1_SCL	TIM16_CH1N	TSC_G5_IO3	-	-
PB7	USART1_RX	I2C1_SDA	TIM17_CH1N	TSC_G5_IO4	USART4_CTS	-
PB8	CEC	I2C1_SCL	TIM16_CH1	TSC_SYNC	CAN_RX	-
PB9	IR_OUT	I2C1_SDA	TIM17_CH1	EVENTOUT	CAN_TX	SPI2_NSS, I2S2_WS
PB10	CEC	I2C2_SCL	TIM2_CH3	TSC_SYNC	USART3_TX	SPI2_SCK, I2S2_CK
PB11	EVENTOUT	I2C2_SDA	TIM2_CH4	TSC_G6_IO1	USART3_RX	-
PB12	SPI2_NSS, I2S2_WS	EVENTOUT	TIM1_BKIN	TSC_G6_IO2	USART3_CK	TIM15_BKIN
PB13	SPI2_SCK, I2S2_CK	-	TIM1_CH1N	TSC_G6_IO3	USART3_CTS	I2C2_SCL
PB14	SPI2_MISO, I2S2_MCK	TIM15_CH1	TIM1_CH2N	TSC_G6_IO4	USART3_RTS	I2C2_SDA
PB15	SPI2_MOSI, I2S2_SD	TIM15_CH2	TIM1_CH3N	TIM15_CH1N	-	-

Accessing AFRx registers with C

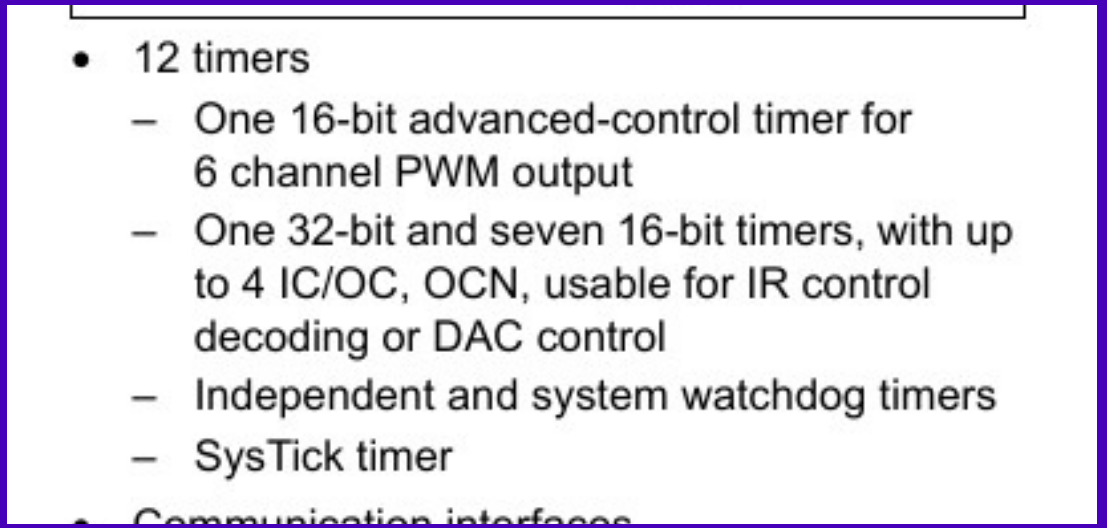
- It’s complicated.
- You can refer to values with symbols like this:

  GPIO_AFRL_AFRL0 == 0x0000000F

  GPIO_AFRH_AFRH5 == 0x00F00000
- It’s often easier to do it manually (e.g.: PB11: AF2):
  - `GPIOB->AFR[1] &= ~(0xf << (4*(11-8)))`
  - `GPIOB->AFR[1] |=   0x2 << (4*(11-8))`

Timers: There are many

- Start with the STM32F091xBC datasheet page 1.



- 12 timers
  - One 16-bit advanced-control timer for 6 channel PWM output
  - One 32-bit and seven 16-bit timers, with up to 4 IC/OC, OCN, usable for IR control decoding or DAC control
  - Independent and system watchdog timers
  - SysTick timer
- Communication interfaces

“How should I choose?”

- Start with the STM32F091xBC datasheet…
  - page 34, table 13, pin definitions.
    - See the "Alternate functions" column.
  - pages 41 – 44, table 14 – 19, Alternate functions for Ports A, B, C, D, and F.
  - "What about Port E?"
    - We don’t actually have a Port E on a 64-pin STM32F091.

“If I want a certain function, how do I chose a pin for it?”

Table 7. Timer feature comparison

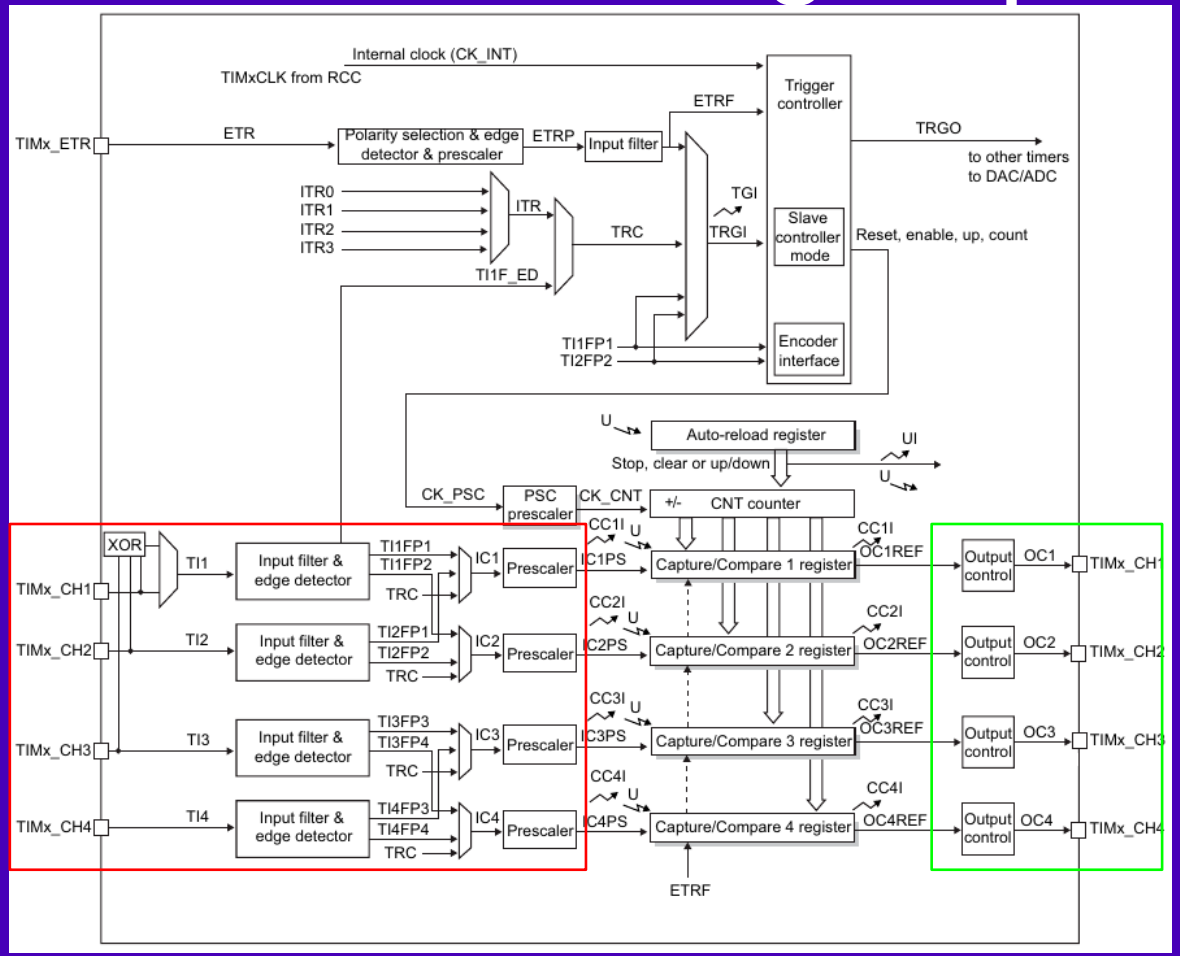
Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/compare channels	Complementary outputs
Advanced control	TIM1	16-bit	Up, down, up/down	integer from 1 to 65536	Yes	4	3
General purpose	TIM2	32-bit	Up, down, up/down	integer from 1 to 65536	Yes	4	-
	TIM3	16-bit	Up, down, up/down	integer from 1 to 65536	Yes	4	-
	TIM14	16-bit	Up	integer from 1 to 65536	No	1	-
	TIM15	16-bit	Up	integer from 1 to 65536	Yes	2	1
	TIM16 TIM17	16-bit	Up	integer from 1 to 65536	Yes	1	1
Basic	TIM6 TIM7	16-bit	Up	integer from 1 to 65536	Yes	-	-

- Start with the STM32F091xBC datasheet again.
  - page 21, table 7, timer feature comparison

“What pins should we start with?”

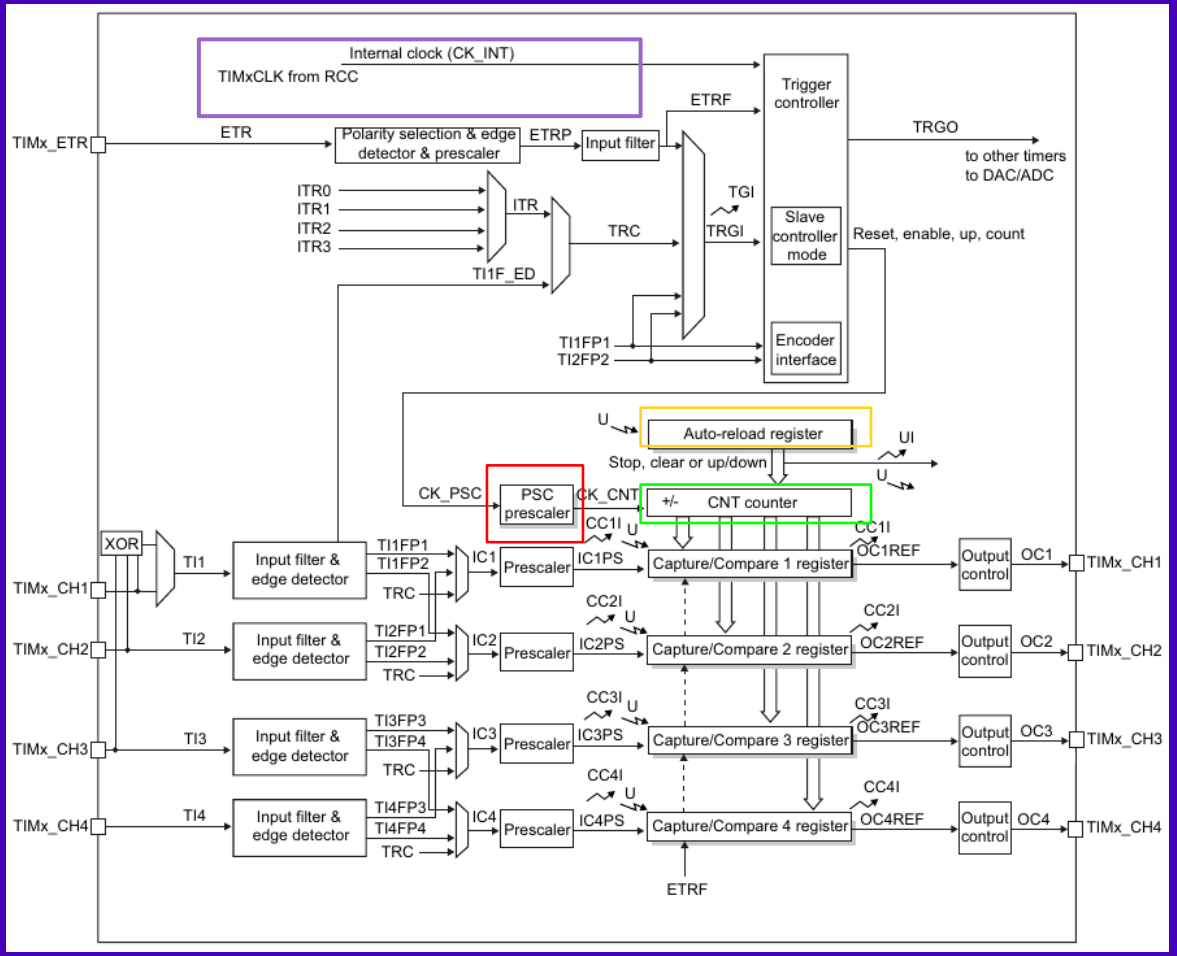
- From datasheet, table 13, pin definitions...
  - User push button SW2 (PA0) can be connected to TIM2_CH1_ETR [Timer 2, channel 1 (and external trigger)]
  - Red LED (PC6) can be connected to TIM3_CH1 [Timer 3, channel 1]
  - Yellow LED (PC7) can be connected to TIM3_CH2 [Timer 3, channel 2]
  - Green LED (PC8) can be connected to TIM3_CH3 [Timer 3, channel 3]
  - Blue LED (PC9) can be connected to TIM3_CH4 [Timer 3, channel 4]
- So, let’s look at Timers 2 and 3.

Timer groups 2 and 3



- This is what either timer 2 or timer 3 look like.
  - Each one contains four timer channels.
    - Each can:
      - measure input
        - count events
        - pulse length
      - generate output
        - toggle
        - PWM: Pulse Width Modulation
      - generate interrupts

How do TIM2/TIM3 work?

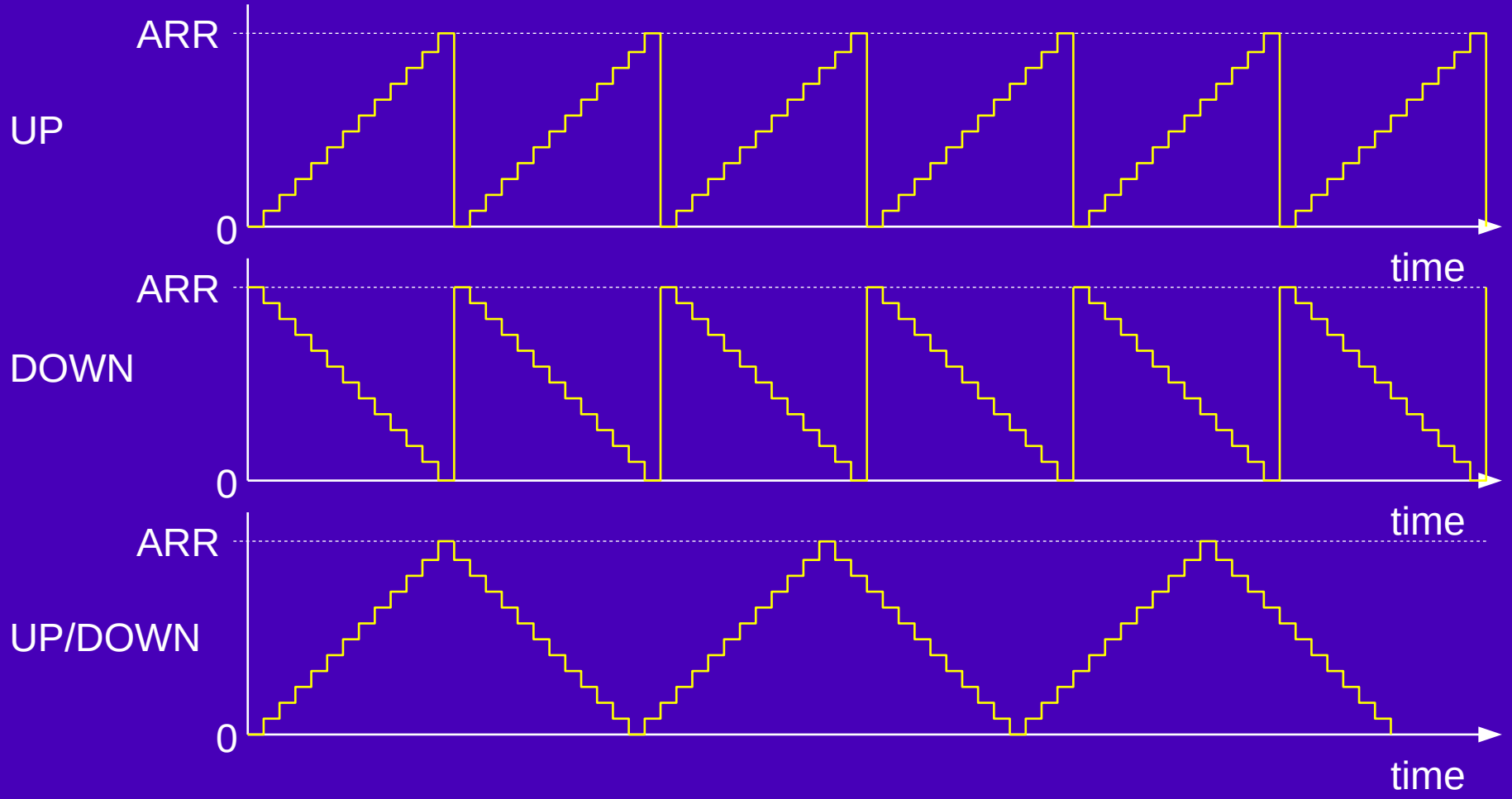


- Each has a free-running counter (CTR) which is 16-bit (TIM3) or 32-bit (TIM2).
  - 16-bit prescaler for input clock (PSC).
    - e.g. divide input clock by N.
    - System clock is default clk.
  - Auto-reload register (ARR)
    - Can count up from 0 to ARR
    - Can count down from ARR to 0
    - Can count up, down, up, down… 0 – ARR – 0 …

Configuring a timer for "output"

- Let’s look at output first. In this mode we:
  - configure a GPIO pin for an alternate function that connects it to a timer rather than its ODR.
  - set up the prescaler (PSC)
  - set up the auto-reload register (ARR)
  - set the direction/mode of the counter (CR1 DIR)
  - set the output compare mode (CCMR*n* OC*c*M)
  - enable the "channel" the pin is connected to (CCER)
  - enable the timer’s counter (with the CR1 CEN bit)

Counter modes



Counter mode configuration

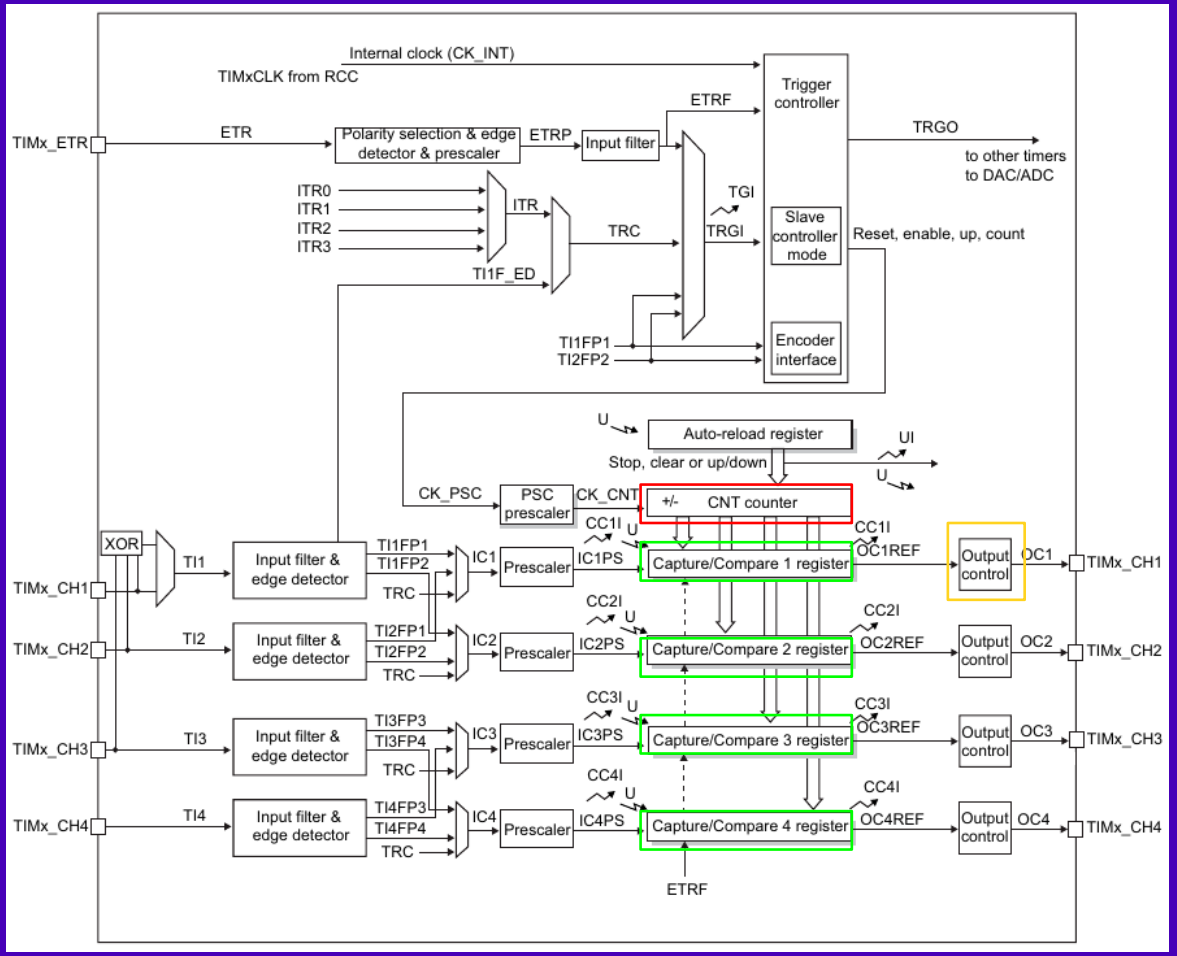
- Set with the TIMx_CR1 register:

9	8	7	6	5	4	3	2	1	0
CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- CMS: Center-aligned (up/down) mode selection.
  - Non-zero value enables up/down mode.
- DIR: Direction: 0: counts up, 1: counts down
- CEN: Counter enable: 1: enable timer.
  - Set this one last.

SET CEN LAST!

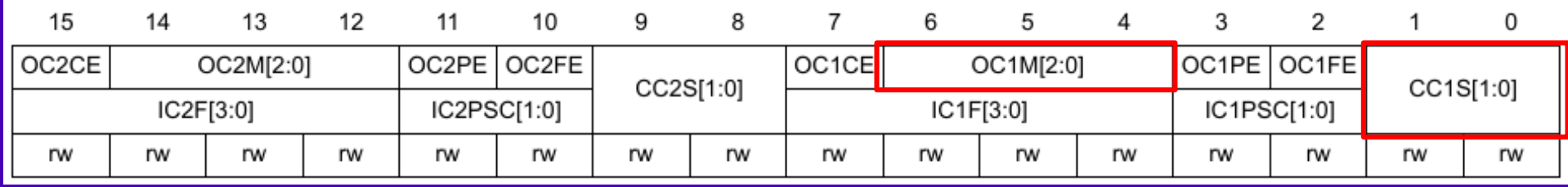
How do channels work?



- Each has a counter compare register (CCRx).
  - Each is continually compared to the counter (CTR).
  - When CCRx = CTR, something happens to OCxREF (output channel reference).

Timer Output Modes

- Configured with the two CCMRs (Capture/Compare Mode Registers)



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

- CC1S: direction of channel:
  - 00: output, 01: input mapped on TI1, 10: input mapped on TI2, 11: input mapped on TRC.
- OC1M: output compare mode:
  - 001: Logic high if CNT = CCR
  - 010: Logic low if CNT = CCR
  - 011: Toggle if CNT = CCR
  - 110: PWM output mode 1: Logic high if CNT < CCR, else logic low

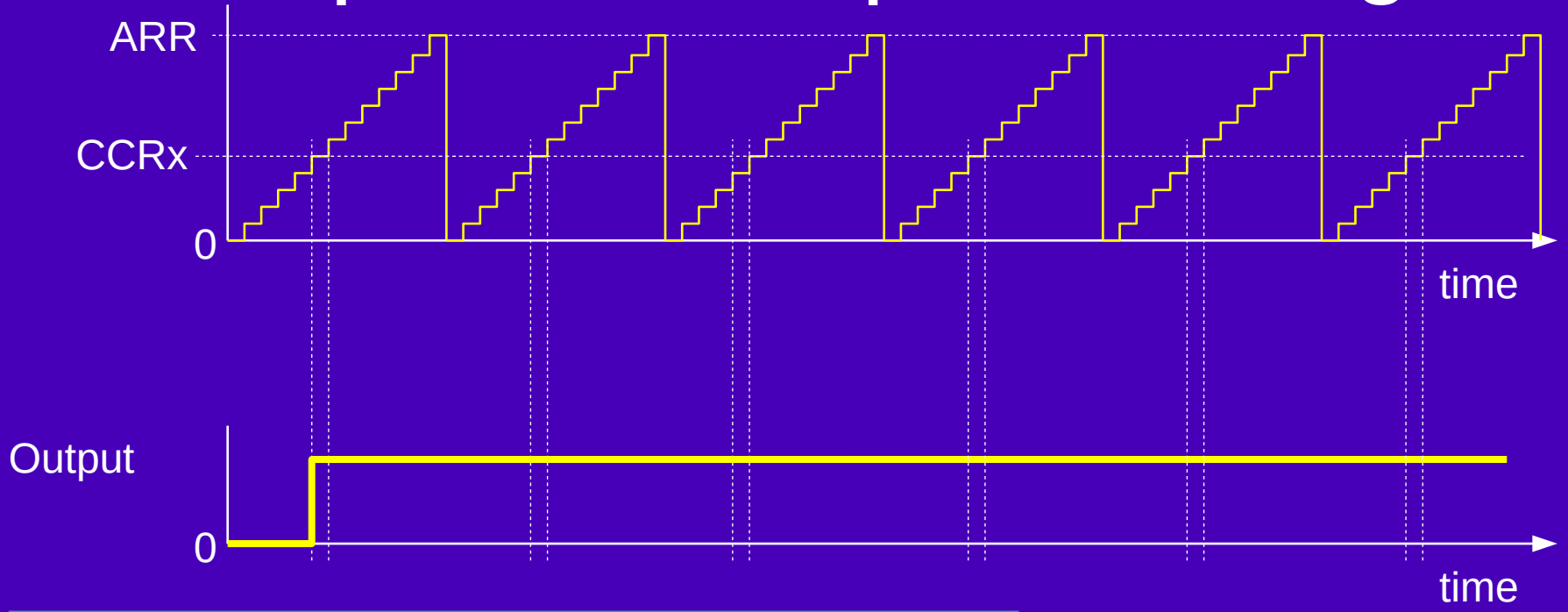
Remember to enable the channels

- Use the TIMx_CCER (capture/compare enable register)

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

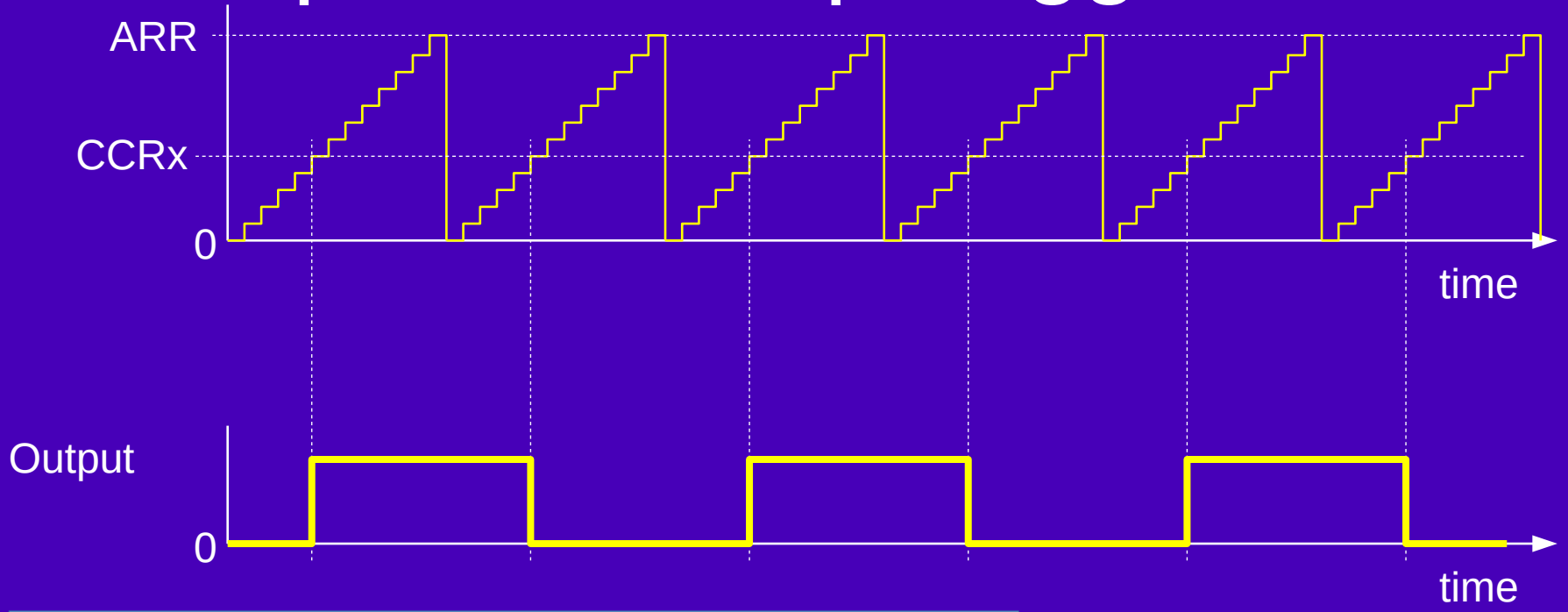
- CCxE: enable the channel

Example: count up, active high



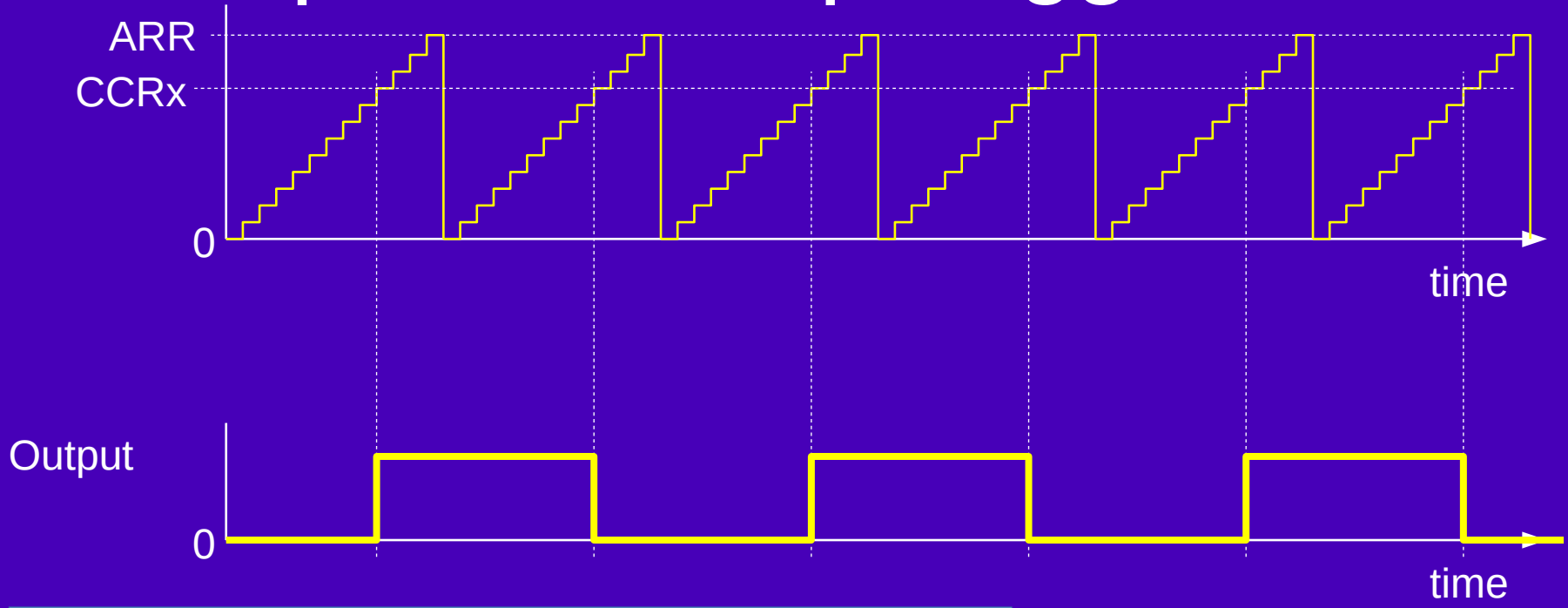
Each CCR is continually comparing its value
with the free-running counter.
When it matches, the operation (go high) happens

Example: count up, toggle mode



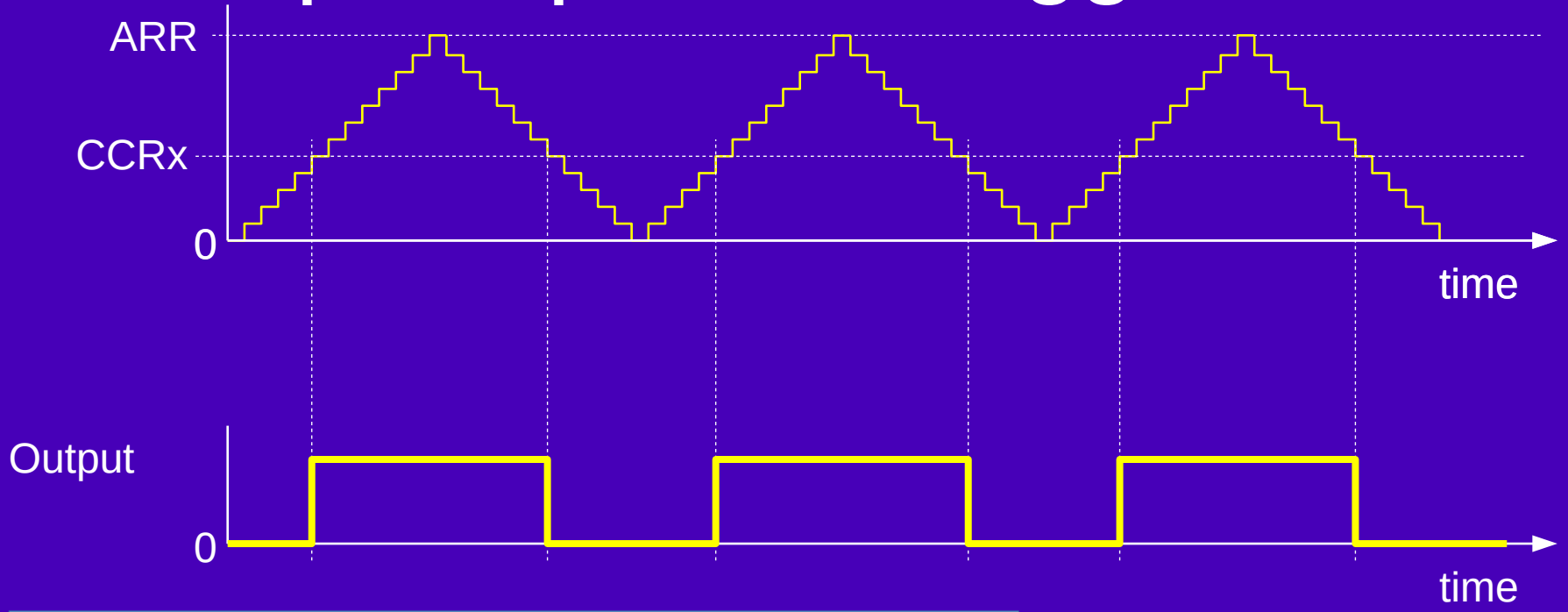
Each CCR is continually comparing its value
with the free-running counter.
When it matches, the operation (toggle) happens

Example: count up, toggle mode



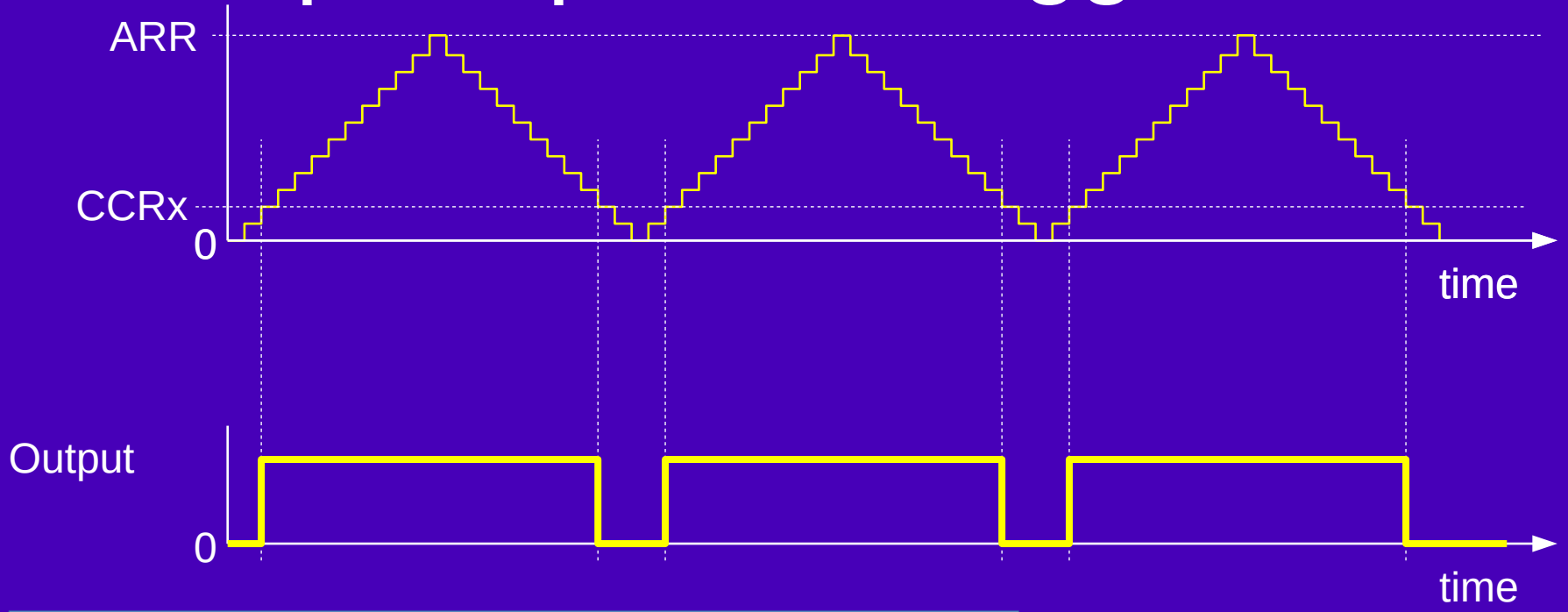
Changing the CCRx value changes the offset into the count when the output toggles.

Example: up/down, toggle mode



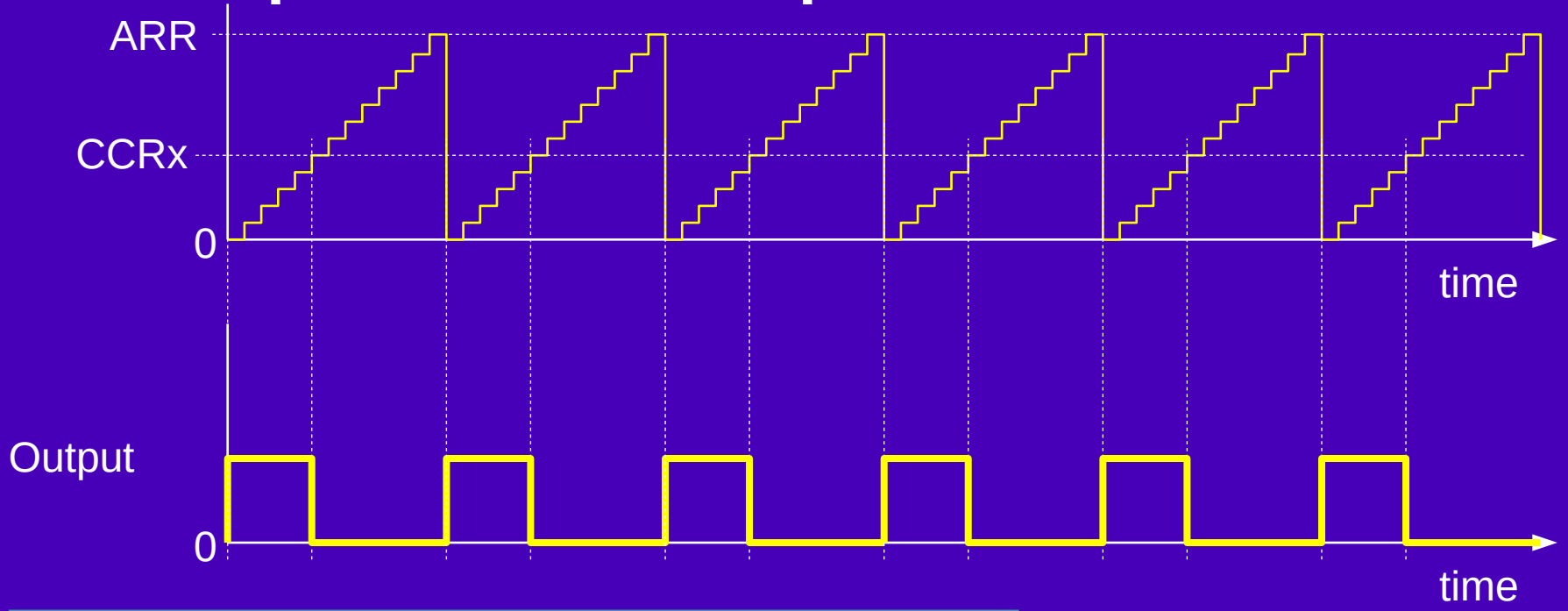
Up/down mode is referred to as "Centered Mode"

Example: up/down, toggle mode



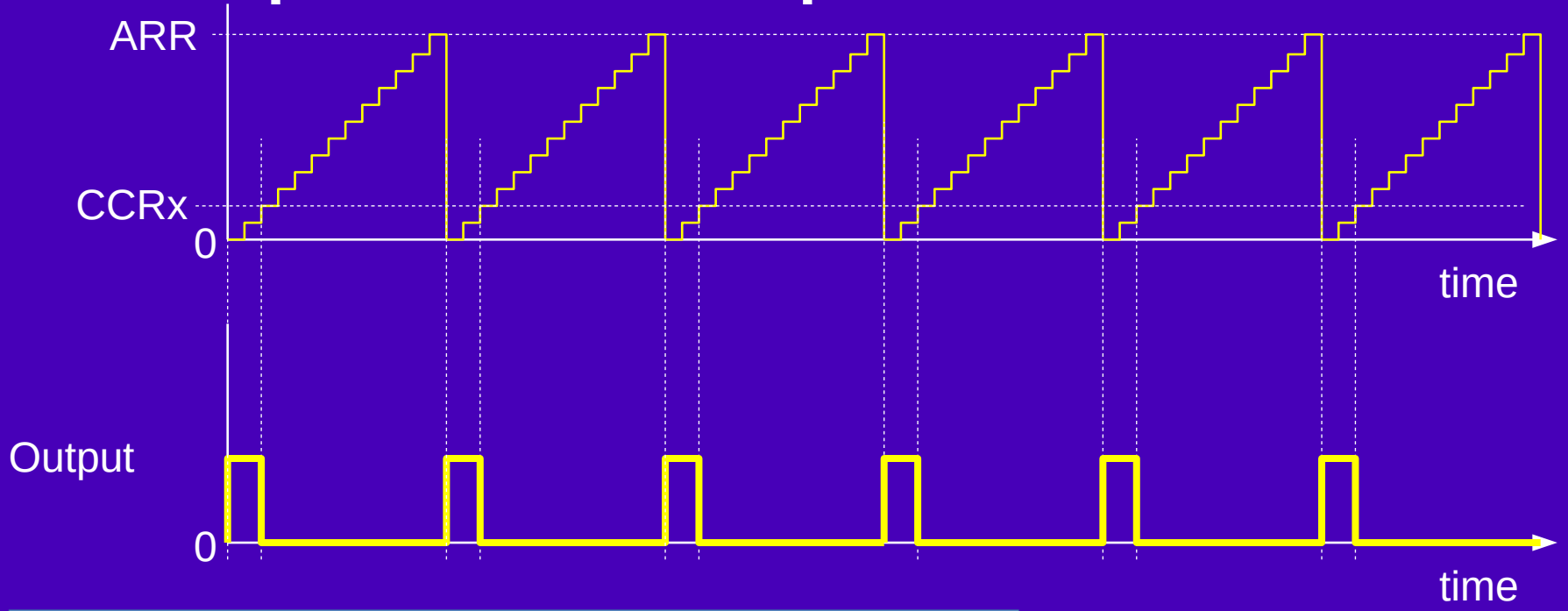
Changing the CCRx value widens the active cycle of the wave, but it stays "centered"

Example: count up, PWM mode 1



PWM output mode is an easy way of configuring a variable "duty cycle" output.

Example: count up, PWM mode 1



As CCRx value is decreased, the
"duty cycle" is decreased proportionally.

How do I program this???

```c
#include "stm32f0xx.h"

int main(void)
{
    // Enable Port C
    RCC->AHBENR |= RCC_AHBENR_GPIOCEN;

    // Set the mode for PC8 for "alternate function"
    GPIOC->MODER &= ~(3<<16);
    GPIOC->MODER |= 2<<16;

    // Set the alternate function for PC8
    // PC0-7 are on AFR[0], 8-15 are on AFR[1]
    GPIOC->AFR[1] &= ~0xf;
    GPIOC->AFR[1] |= 0;

    // Enable the system clock for timer 3
    RCC->APB1ENR |= RCC_APB1ENR_TIM3EN;

    // Counting direction: 0=up, 1=down
    TIM3->CR1 &= ~TIM_CR1_DIR; // clear it to count up

    // Set prescaler output to 4kHz (48MHz/12000)
    TIM3->PSC = 12000 – 1;

    // Auto-reload 4000
    TIM3->ARR = 4000 – 1;

    // Any value between 0 and 4000.
    TIM3->CCR3 = 3456;

    // Channel 3 of the timer is configured in CCMR2.
    // Set the bits to select toggle mode (011)
    TIM3->CCMR2 &= ~TIM_CCMR2_OC3M_2; // Turn off bit 2.
    TIM3->CCMR2 |= TIM_CCMR2_OC3M_1 | TIM_CCMR2_OC3M_0;

    // Enable output for channel 3 active-high output
    TIM3->CCER |= TIM_CCER_CC3E;

    // Enable timer 3
    TIM3->CR1 |= TIM_CR1_CEN;

    for(;;)
        asm(“wfi”);

    return 0;
}
```

Adapted from Textbook, page 380.

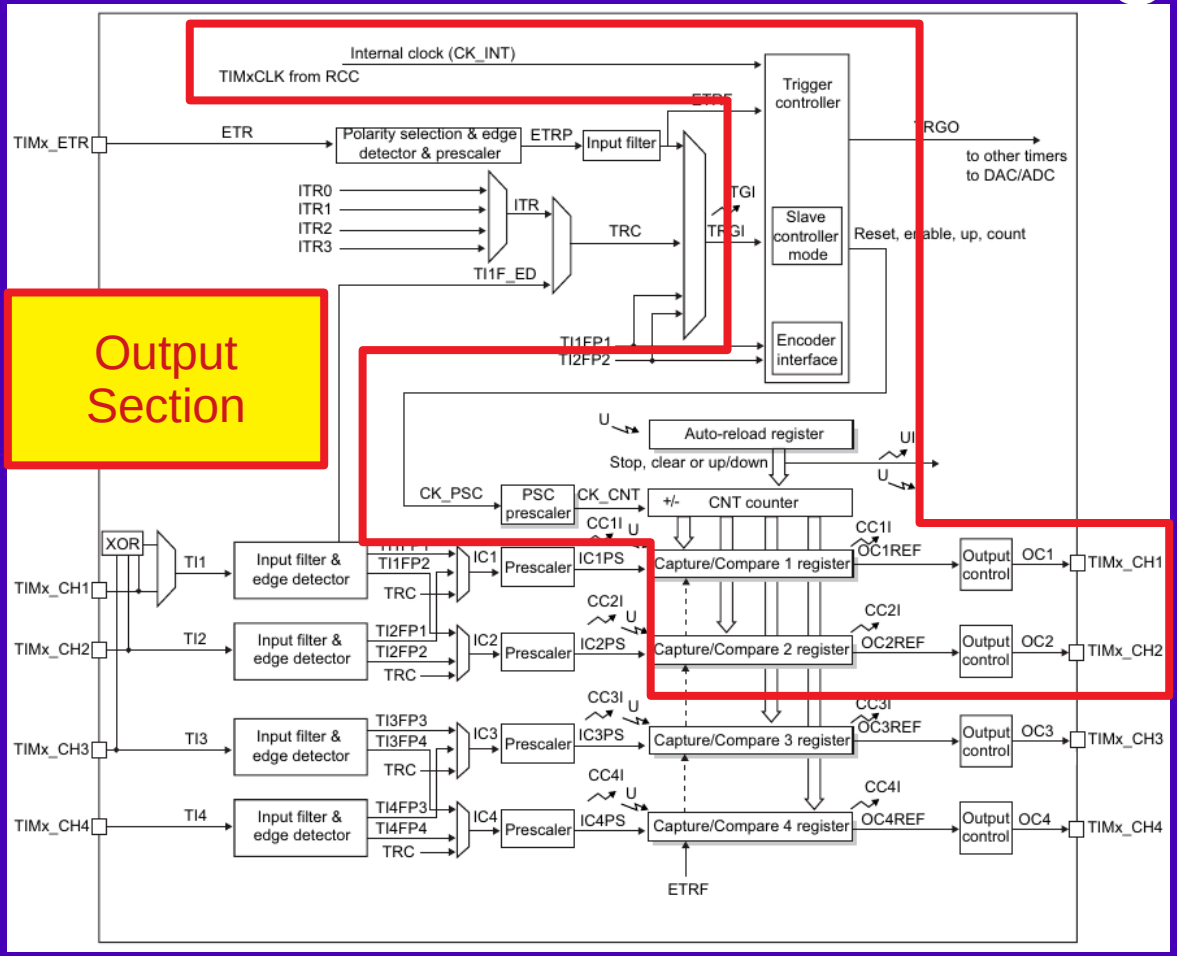
This will blink the green LED.

Register summary

- See Family Reference Manual, table 60, page 442...
- TIMx_CR1: Control Reg
- TIMx_CCMR1/2: Capture/Compare Mode Reg
- TIMx_CCER: Capture/Compare Enable Reg
- TIMx_CNT: 16- or 32-bit Free-running counter
- TIMx_PSC: 16-bit Prescaler Reg
- TIMx_ARR: Auto-Reload Reg
- TIMx_CCRx: Capture/Compare Reg

- RCC_APB1ENR: Location of the TIM3EN bit.

Full view of Timer 2/3 again



Output config register arrangement