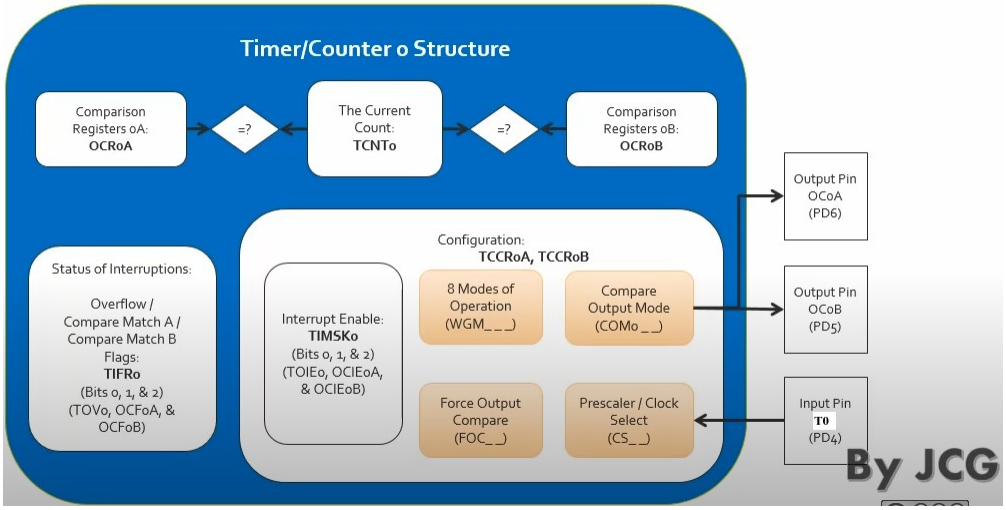
Timer/Counter0 CTC mode

https://www.youtube.com/watch?app=desktop&v=OReeWLpipmk&t=725s

- 8 bit-timer (can count from 0 to 255)
- Different modes of operation:
  - Can „count" time (or MCs – Machine Cycles) in two different ways(Normal vs. **CTC**- Clear Timer on Compare Match)
  - Can „count" external events (hence the counterpart in the name)
  - Can generate a Pulse With Modulation (PWM) Signal

- It can generate 3 interrupt sources:
  - When the counter „owerflows" (goes from 255 to 0)
  - When the counter matches a „desired" A value, or
  - When the counter matches a „desired" B value



The counter is „simple" but whit many possibilities of operation. But basically this is what happens in the Timer 0 peripherial:

- The current count is stored in **TCNT0** always.
- It can generate interrupt sources, enabled in **TIMSK0** - Timer/Counter Interrupt Mask Register (**TOIE0** - Timer/Counter0 Overflow Interrupt Enable, **OCIE0A**: Timer/Counter0 Output Compare Match A Interrupt Enable, **OCIE0B**: Timer/Counter Output Compare Match B Interrupt Enable)
- The mode of operation is configured in **TCCR0A, TCCR0B** – Timer/Counter Control Register A and B trough **WGM** - Waveform Generation Mode, **WGM2, WGM1, WGM0**
- It has two buffred registers to compare the current count vs. the compare register (**TCNT0 vs. OCR0A/OCR0B - O**utput **C**ompare **R**egisters)
- The frequency ot the timer depends on **CS - C**lock **S**elect **S**ettings (CS02, CS01, CS00). It can be either a fraction of the microcontroller frequency (prescaler), or an external pin (**T0**)in the microcontroller
- It has two **O**utput **C**ompare pins (**OC0A** and **OC0B**) which behave according to both WGM settings and **COM** (Compare Match Output Mode) settings

How do we make Timer/Counter 0 has performed its task?

1. Monitor its flags (polling).
   - This is not the best option. (Similar to the delay with processor since we would need to constantly be checking the flags with **SBIC/SBIS/SBRC/SBRS**, and also we need to manually reset flgas!)

2. Enable interrupts (event-drive).
   - The **I**nterupt **S**ervice **R**outine (**ISR**) will be executed once the timers flags are turned ON. (The Interrupt Mechanism in the microcontroller turns automatically the flags!)

We will review the CTC mode of operation of Timer0 to show how to SET-UP the Timer0 peripherial in this particular mode of operation.

- In this mode the current count (**TCNT0**) is cleared to **0x00** when the value of **TCNT0 = OCR0A**.
- The flags are set as:
  - **OCF0 -** Output Com**pare Flag (OCF0A or OCF0B) is set to 1, when TCNT0 = OCR0 + 1.**
  - **TOV0 -** Timer/Counter Overflow Flag is set to 1, when **TCNT0** goes from 0xFF to 0x00 (Overflow, although this should not happen, but CAN happen)
  - The output pins (**OC0A** and **OC0B**) can be used to generate a signal with the settings of **COM** bits.

If we want to generate a flashing LED at 20Hz, with Timer/Counter0 (ATmega2560 I/O frequency=16MHz).

1. **TCCR0A, TCCR0B** – Set **WGM** propertly to get CTC Mode (Note: Bit **WGM02** is located in **TCCR0B**).

   WGM2, WGM1, WGM0 = 0, 1, 0

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Mode	WGM2	WGM1	WGM0	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on[1][2]
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	TOP	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

2. Calculate/decide the prescaler value:
   We will chose N=1024, because it can count more time, thus generating a lower frequeny.

3. **OCR0A** – Calculate and load the **OCR0A**:

$$f_{OCnx} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnx)}$$

   The *N* variable represents the prescale factor (1, 8, 64, 256, or 1024).

   OCR0A = (16000000Hz)/(2 * 1024 * 20Hz)-1 = **389,625**

   !!!

   We cannot store 389,625 because these number is greater than 255, the register is 8 bit.
   We cannot generate 20hz signal.

   !!!

We will generate 40Hz signal.
OCR0A = (16000000Hz)/(2 * 1024 * 40Hz)-1 = **194,3125**
OCR0A = 194 (Note: Real frequency achived = 40.064 Hz)

4. **TIMSK0 -** Timer Interrupt Mask Register Enable interruptions of **OCR0A** and Global Interruption.
5. **TCCR0B** – Set the prescaler value. (Note: This will START the timer, hence why it is done last).
   CS02, CS01, CS00 = 1, 0, 1 (N = 1024)

6. Program the **ISR**, to divide the frequency by half and turn ON the LED.

atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf