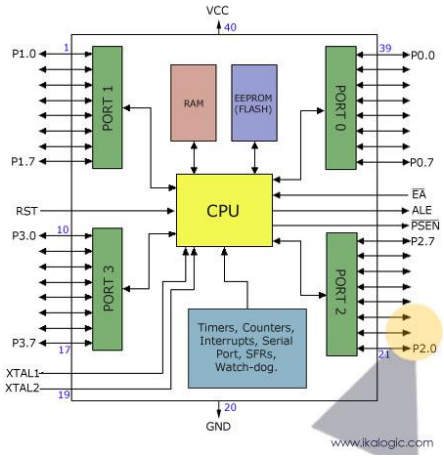
Embedded Systems Design, Spring 2025

Lecture 6



Timers

Outline

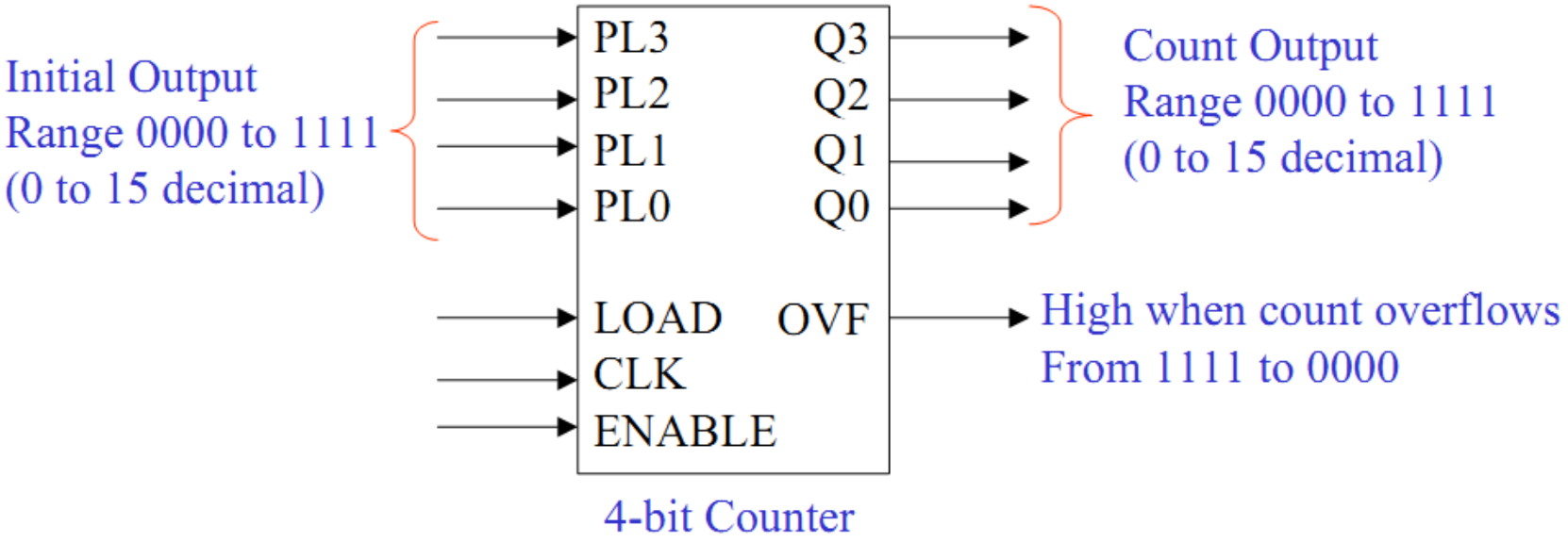
- 4bit counter
- Timer/Counter 0 and Timer/Counter 2
- Timer/Counter 1

4 bit counter example

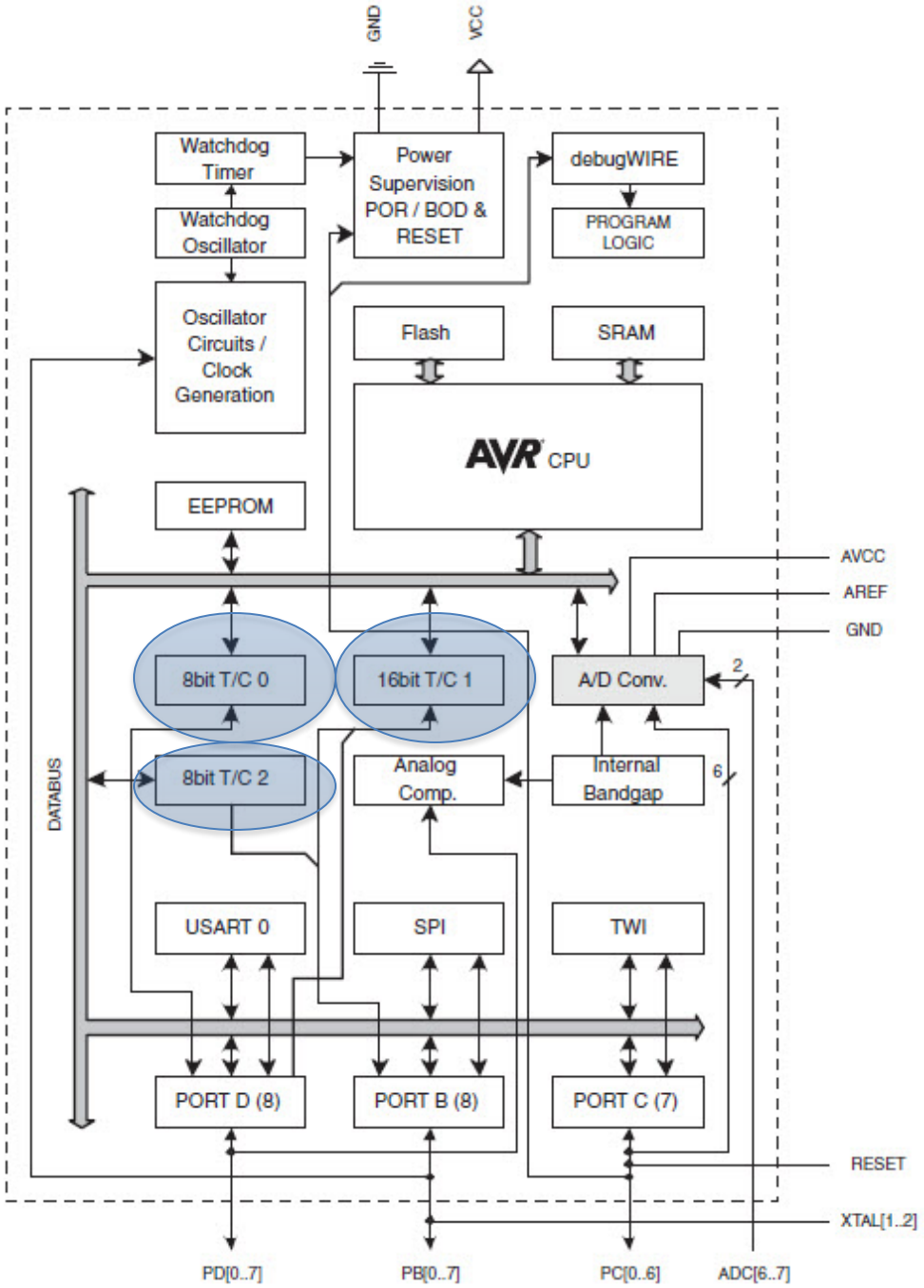
- Count range: 0000 – 1111 (0x0 – 0xF)
- After reaching 1111, the counter will overflow to 0000
- When this happens, an overflow flag is set
- For a specific delay, load an initial value and then start
  - Every counter clock, the timer will increment
  - Maximum possible delay: 16 clock cycles
  - To get a shorter delay: change initial value

Operation

- If EN is high, the counter will count each rising edge in the clock
  - Range: 0000b – 1111b
- If LOAD is high, PL0 – PL3 will be loaded as the initial value
- When the counter overflows, OVF will go high



ATmega 328p Block Diagram



Timers in ATmega328p

- The ATmega328 has 3 timers
  - Timer 0, Timer 1 and Timer 2
- Timer 0 and Timer 2
  - **8 bit** counter
  - Count range: **0x00 – 0xFF** (255)
- Timer 1
  - **16 bit** counter
  - Count range: **0x0000 – 0xFFFF** (65535)

- Operation modes:
  - **Timer mode: count internal clock pulses**
    - Timer register updated every clock cycle
    - Delays, real time clocks
  - **Counter mode: count external events**
    - Timer register updated every time a transition is present on the pin
    - Pulse width or frequency measurements
  - PWM generation

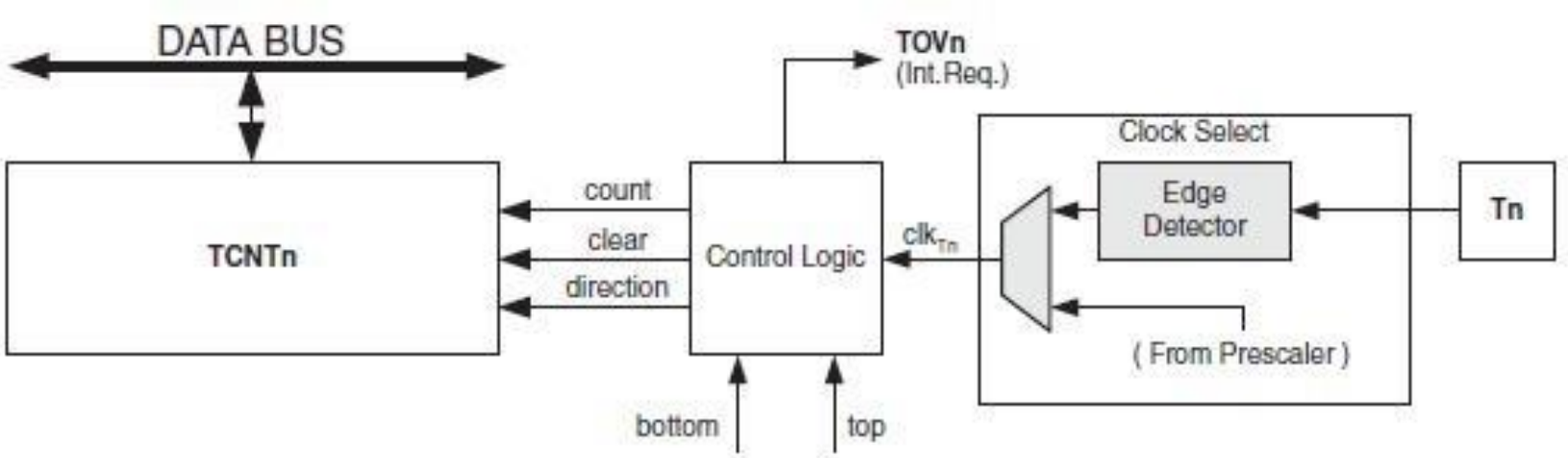
Timer operation

- Operation:
  - The clock source sends pulses to the prescaler
  - The prescaler divides this with a certain amount
  - The input is sent to the control circuit which increments TCNTn
  - If TCNTn reaches TOP LIMIT, it resets to 0 and triggers a timer overflow

TOP LIMIT value either

- max value on 8 bit / 16 bit (normal mode)
- user defined value (CTC mode)

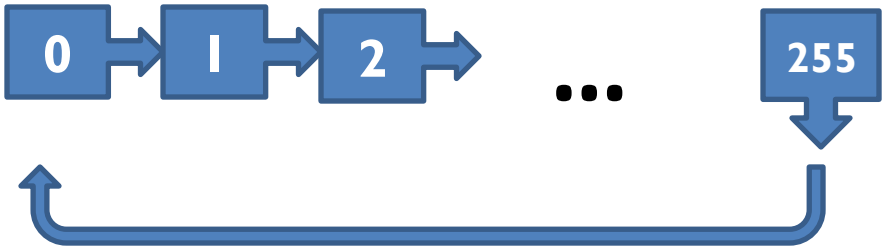
Timer 0 (8 bits)



From prescaler-> TCNT0

Normal mode

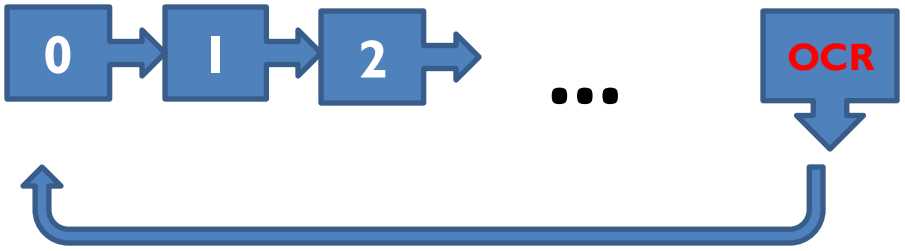
- Clock cycle > Prescaler > Control Logic > increment TCNTn
- When it reaches TOP = 255 it overflows to 0 and sets TOVn bit
- Hard to use for exact interval
- Good for a no specific time interval



Setting TOV (Timer overflow)

CTC Mode

- Clear Timer on Compare
- Uses the OCR (on compare register with desired TOP value)
- Clock cycle > Prescaler > Control Logic > increment TCNTn
- TCNTn compared with OCRn register
  - If equal: OCFn bit is set in the TIFR register
- Good for an exact time interval

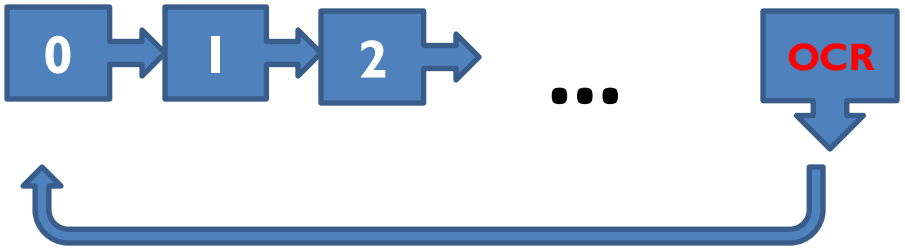


Setting OCF (on compare flag)

CTC Mode

OCRn = [(clock_speed/prescaler) * time_in_seconds] - 1

- **Must be an integer value**
- **Must fit into 8 bits/16 bits**



Setting OCF (on compare flag)

Registers: 1. Set Outputs

TCCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	COM0A1	**COM0A0**	COM0B1	COM0B0	-	-	WGM01	WGM00

These bits control the Output Compare pin (OC0A) behavior. (B7 output)

Table 15-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

Registers 1. Set Outputs

TCCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	COM0A1	COM0A0	**COM0B1**	**COM0B0**	-	-	WGM01	WGM00

These bits control the Output Compare pin (OC0B) behavior. (D0 output)

Table 15-5. Compare Output Mode, non-PWM Mode

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Toggle OC0B on Compare Match
1	0	Clear OC0B on Compare Match
1	1	Set OC0B on Compare Match

Registers: 2. Choose mode

TCCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	COM0A1	COM0A0	COM0B1	COM0B0	-	-	**WGM01**	**WGM00**

TCCR0B	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	FOC0A	FOC0B	-	-	**WGM02**	CS02	CS01	CS00

These bits control the Waveform Generation Mode
Can be used to control Normal Mode, CTC Mode or 2 PWM Modes

Mode	WGM02	WGM01	WGM00	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	BOTTOM	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

Registers: 3. Select prescaler

TCCR0B	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	FOC0A	FOC0B	-	-	WGM02	**CS02**	**CS01**	**CS00**

These bits control the clock source
Can be used to control Normal Mode, CTC Mode or 2 PWM Modes

Table 15-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)

Registers: 4. Others

Timer/Counter 0 Interrupt Mask Register

TIMSK0	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	-	-	-	-	-	OCIE0B	**OCIE0A**	**TOIE0**

Timer/Counter 0 Interrupt Flag Register

TIFR0	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	-	-	-	-	-	OCF0B	**OCF0A**	**TOV0**

Timer/Counter 0 Register (stores the current counter value)

TCNT0	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit

Timer/Counter 0 Output Compare Register (desired value)

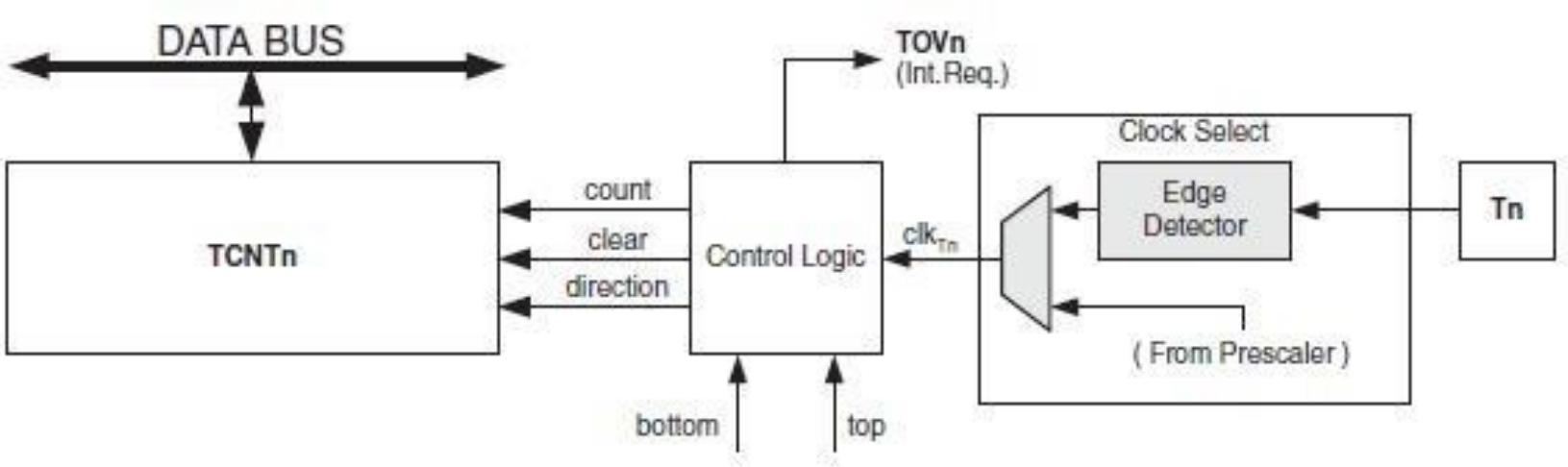
OCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit

Use timer0 to generate a 1ms delay

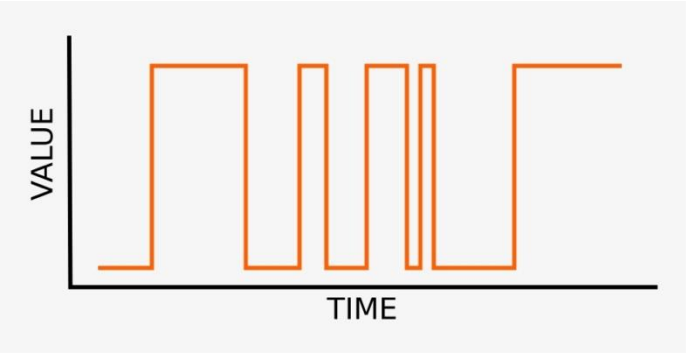
Example:

```
// this code sets up a timer0 for 1ms @ 16Mhz clock cycle
#include <avr/io.h>

int main(void)
{
    while (1)
    {
        // Set the Timer Mode to CTC
        TCCR0A |= (1 << WGM01);
        // Set the value that you want to count to
        OCR0A = 0xF9;
        // start the timer
        TCCR0B |= (1 << CS01) | (1 << CS00);// set prescaler to 64 and start
the timer
        while ( (TIFR0 & (1 << OCF0A) ) == 0)  // wait for the overflow event
        {
        }

        // reset the overflow flag
        TIFR0 = (1 << OCF0A);


    }
}
```

Counter 0 (8 bits)



T0:



Edge detector -> TCNT0

Counting external inputs

- **Normal mode:**
  - Counter will count until it hits TOP value (0xFF)
  - Then starts from 0 and sets TOV0 flag

- **CTC mode**(Clear Timer on Compare):
  - Counter will count until it hits OCR0 value
  - Then starts from 0 and sets OCF0 flag

Registers: 1. Set Outputs

TCCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	COM0A1	**COM0A0**	COM0B1	COM0B0	-	-	WGM01	WGM00

These bits control the Output Compare pin (OC0A) behavior.

Table 15-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

Registers: 1. Set Outputs

TCCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	COM0A1	COM0A0	**COM0B1**	**COM0B0**	-	-	WGM01	WGM00

These bits control the Output Compare pin (OC0B) behavior.

Table 15-5. Compare Output Mode, non-PWM Mode

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Toggle OC0B on Compare Match
1	0	Clear OC0B on Compare Match
1	1	Set OC0B on Compare Match

Registers: 2. Choose Mode

TCCR0A	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	COM0A1	COM0A0	COM0B1	COM0B0	-	-	**WGM01**	**WGM00**

These bits control Waveform Generation Mode.
Can be used to control Normal Mode, CTC Mode or 2 PWM Modes
Any mode works, but Normal Mode recommended

Mode	WGM02	WGM01	WGM00	Operation	TOP
0	**0**	**0**	**0**	Normal	0xFF
2	**0**	**1**	**0**	CTC	OCRA

Registers: 3. Select clock type

TCCR0B	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	FOC0A	FOC0B	-	-	WGM02	**CS02**	**CS01**	**CS00**

These bits control the clock source
Can be used to control which type of edge (falling/rises) triggers the counter

CS02	CS01	CS00	Description
0	**0**	**0**	Timer/Counter 0 disabled
1	**1**	**0**	External Clock Source on T0 pin (falling edge)
1	**1**	**1**	External Clock Source on T0 pin (rising edge)

Registers: 4. Others

TIMSK0	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	-	-	-	-	-	OCIE0B	OCIE0A	**TOIE0**

Timer/Counter 0 Interrupt Mask Register

TIFR0	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit
	-	-	-	-	-	OCF0B	OCF0A	**TOV0**

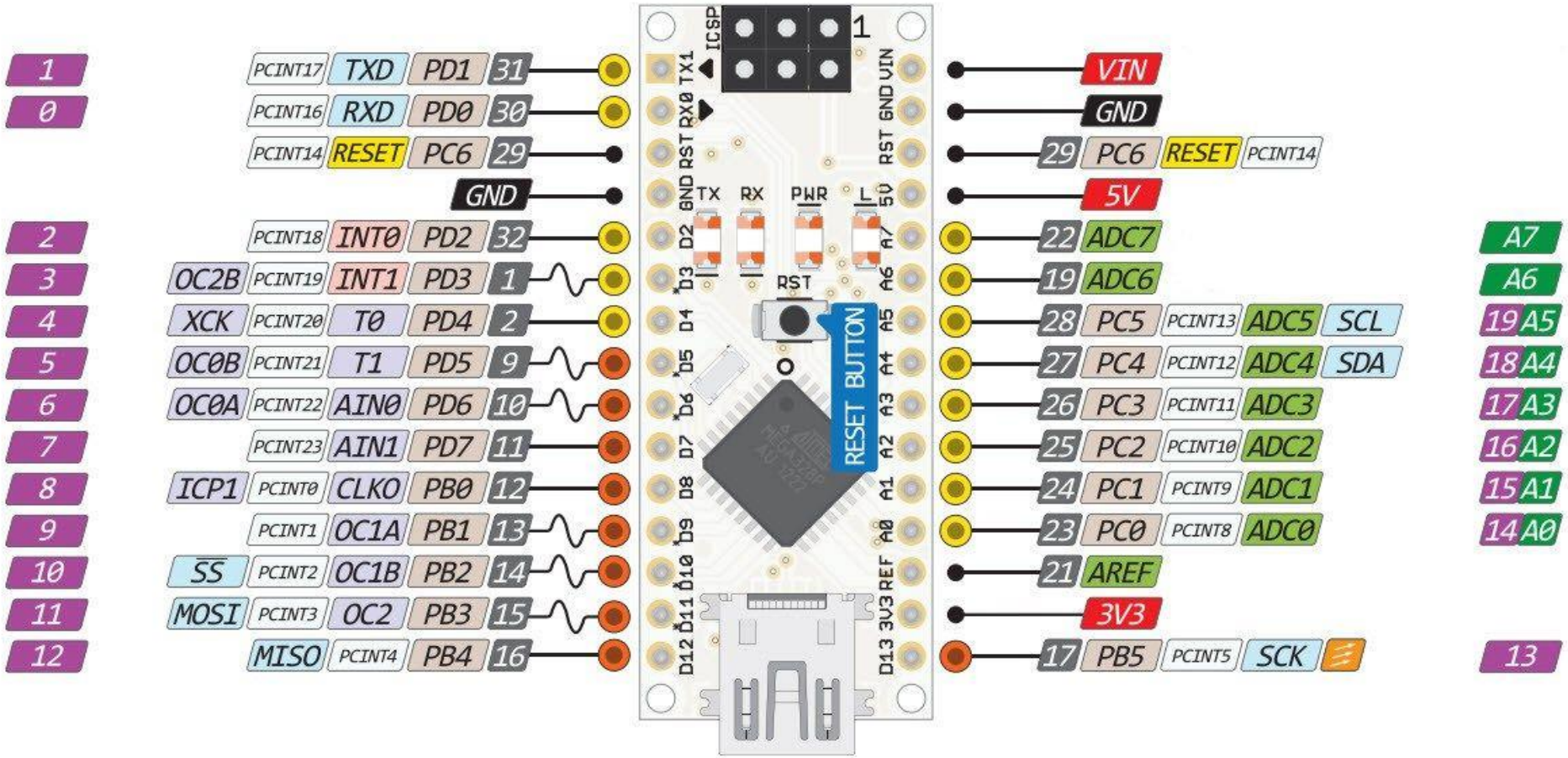
Timer/Counter 0 Interrupt Flag Register

TCNT0	7 bit	6 bit	5 bit	4 bit	3 bit	2 bit	1 bit	0 bit

Timer/Counter 0 Register (stores the counter value)

Use counter0 for counting edges (on T0)

Arduino nano pinout



Example:

```
#include <avr/io.h>

int main(void)
{
    DDRD &= ~(1 << DDD4);     // Clear the PD4 pin
    // PD4 is now an input

    PORTD |= (1 << PORTD4);   // turn On the Pull-up
    // PD4 is now an input with pull-up enabled

    TCCR0B |= (1 << CS02) | (1 << CS01) | (1 << CS00);
    // Turn on the counter, Clock on Rise



    while (1)
    {
        // we can read the value of TCNT0 (print every 1 second for
example)
    }
}
```