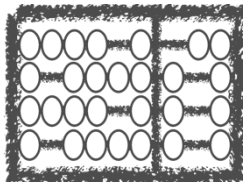
# INF-741: Embedded systems programming for IoT

Prof. Edson Borin

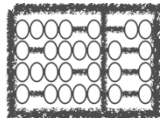


Institute of Computing

University of Campinas

# Timers and Counters





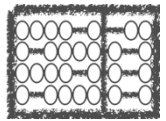
# Agenda

**Timers, Counters and Watchdog Timers**

ATMega328 Timers

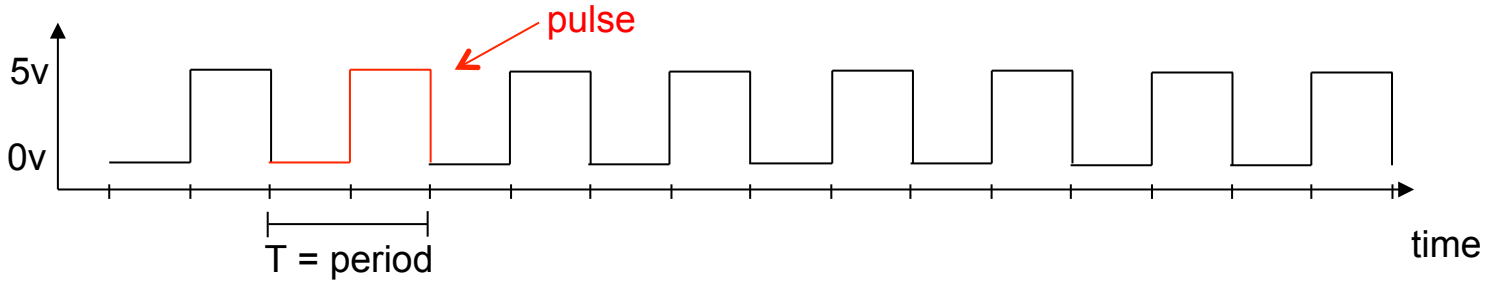
Course Work





# Timers, Counters and Watchdog Timers

**Clock signal**: periodic signal used to control the operation of digital circuits.



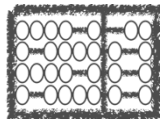

Periodic signal with period = T ==> x(t) = x(t + T)

**Pulse length = Period**

Usually a square wave generated by a **clock generator**.

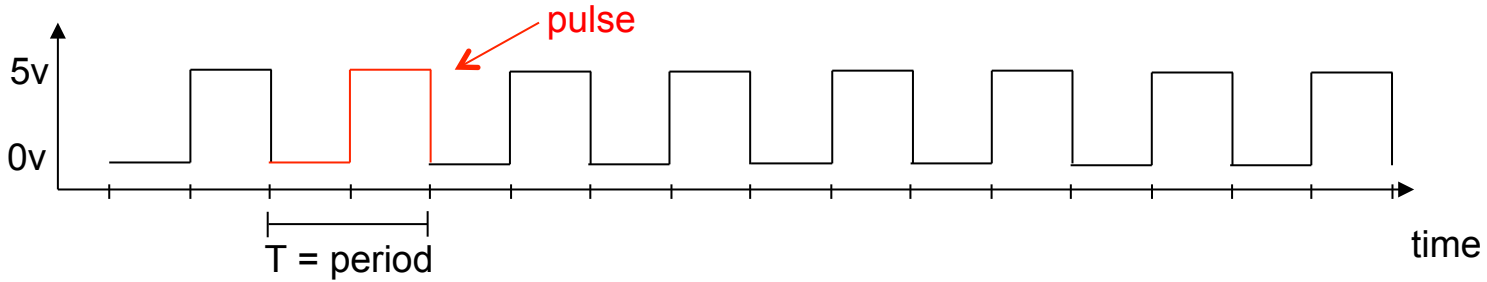
* Typically a signal generated by a crystal oscillator



# Timers, Counters and Watchdog Timers

**Clock signal**: periodic signal used to control the operation of digital circuits.



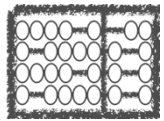

Clock **Rate** or Clock **Frequency** = Number of pulses per second.

- Measured in Hertz (Hz)

Clock **Frequency** = 1 / Clock **Period** or f = 1/T

Clock **period** of a 16 MHz signal = 1/16.000.000 s = 0.0625 microseconds



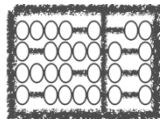


# Timers, Counters and Watchdog Timers

**Timer**

- Very common peripheral device that can measure time intervals.
- Measures time by counting **pulses** that occur on an input clock signal having a known period.
  - Ex: counting 2000 pulses of a 1MHz clock signal = 2 milliseconds
    - 1MHz => 1000000 pulses per second.
- May be used to:
  - generate events at specific timer;
  - determine the duration between two external events;
  - etc.

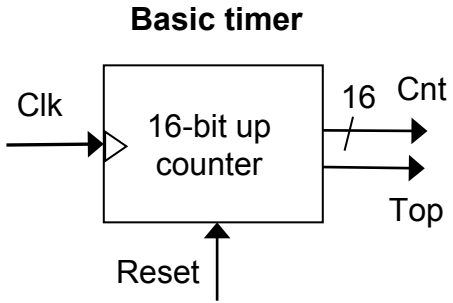




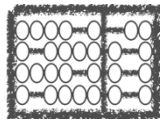
# Timers, Counters and Watchdog Timers

## Timer

- A counter: register + control circuit to increment or decrement its value based on clock pulses (Clk).
- Reset: zeroes the counter.
- Top: output signal that indicates that the counter reached its maximum value:
  - 16-bits => 0xFFFF => 65535
  - Register value wraps around when top is reached



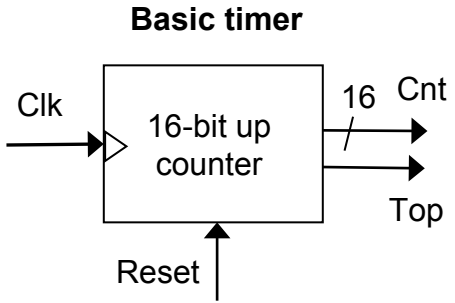


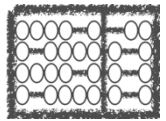
# Timers, Counters and Watchdog Timers

## Timer

- **Range**: Maximum time interval the timer can measure
  - Range = $2^N$ – 1 * Clock Period
    - where N = number of bits.
  - Ex: 16-bit counter and 1MHz Clk
    - $2^{16}$-1 x 1/1.000.000 s = 65535 * 0.000001 = 0.065535s
- **Resolution**: minimum time interval the timer can measure
  - Resolution = Clock Period
  - Ex: 1 MHz Clk – Resolution = 0.000001 second (1 microsecond)



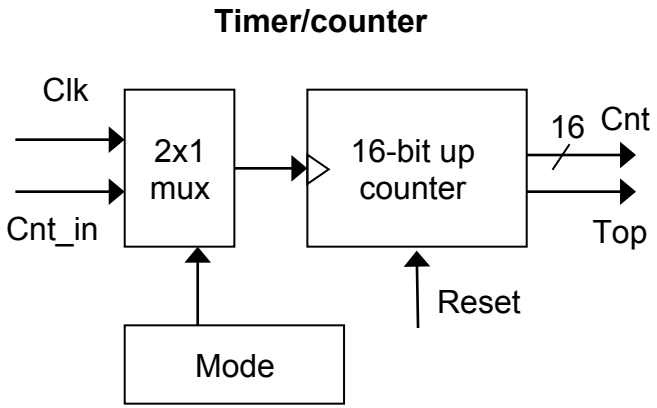


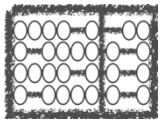
# Timers, Counters and Watchdog Timers

## Counter

- Similar to a timer, but counts pulses on a general input signal rather than clock.
  - e.g. count cars passing over a sensor
- Timer devices are usually flexible and can be configured to act as a counter (Mode).
- Mode: controls whether the counter is incremented based on clock pulses (Clk) or a general input signal (Cnt_in).





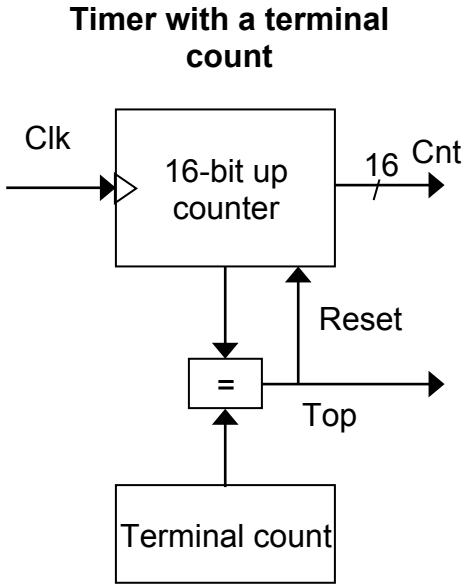



# Timers, Counters and Watchdog Timers

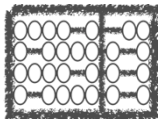
## Interval Timer

- Indicates when desired time interval has passed.
- We set **terminal count** to desired interval

  Number of clock cycles = Desired time interval / Clock period
- Ex: obtain a duration of 3 microseconds from a clock cycle of 10 nanoseconds (100MHz).
  - Terminal count = (3 x $10^{-6}$s) / (10x$10^{-9}$s) = 300 cycles
- Top is asserted after 300 cycles (3 microseconds)



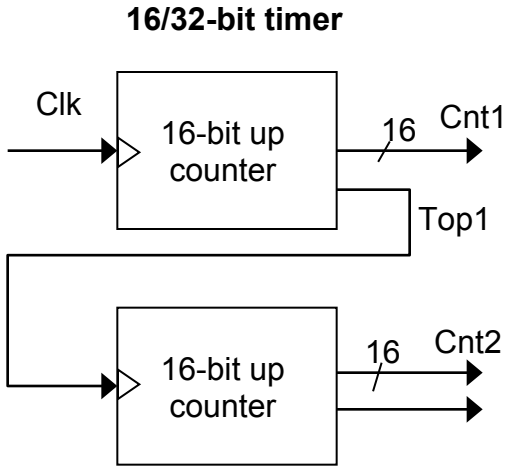


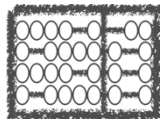
# Timers, Counters and Watchdog Timers

## Cascaded counters

- Multiple timers can be combined to extend range.
  - Ex: The top output of the first 16-bit counter (Top1) can be used as the clock input of the second 16-bit counter.
  - The second counter is only incremented when the first one reaches the top value (*carry-out*).



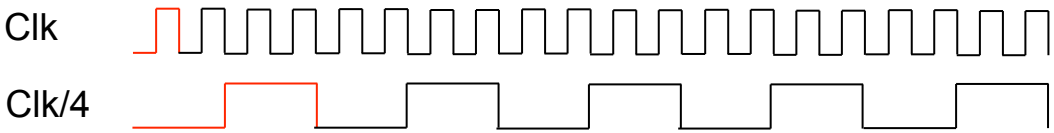


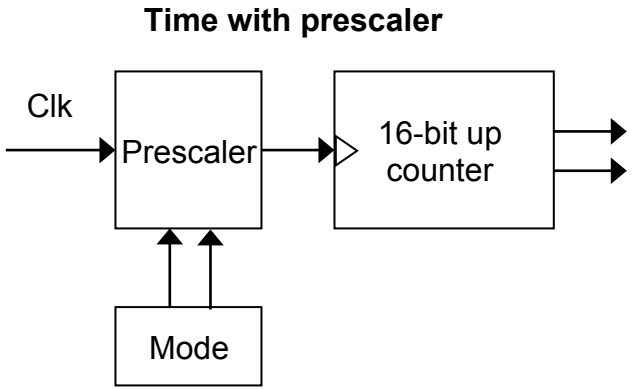
# Timers, Counters and Watchdog Timers

## Timer with a prescaler

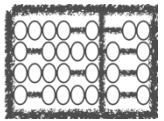
- Prescaler: configurable clock-divider circuit
- clock
- May divide the input clock by 1, 2, 4, 8, etc...




- May be used to increase a timer's range, at the cost of decreasing resolution.



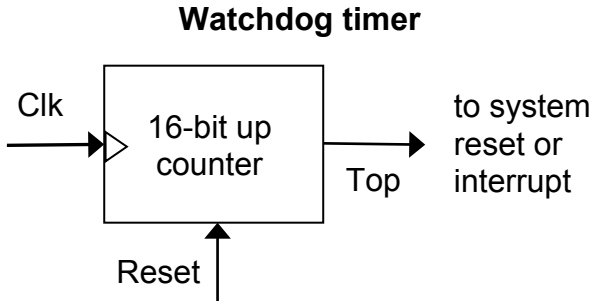


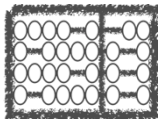
# Timers, Counters and Watchdog Timers

## Watchdog Timers

- Output Top signal is connected to interrupt or reset pin.
  - Counter overflow resets the system
- User must restart the counter from time to time to prevent reset.
  - Failing to restart the counter in due time will cause the system to reset.
- May be used to recover from system failures:
  - Ex: Undesired infinite loops. Waiting for an input event that never arrives. Etc.





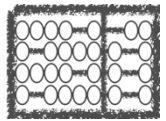
# Agenda

Timers, Counters and Watchdog Timers

**ATMega328 Timers**

Lab Activities





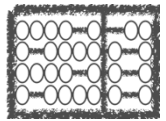
# ATMega328 Timers

3 timers:

- TIMER0 and TIMER2: 8 bits
- TIMER1: 16 bits

Used by Arduino runtime software to implement functionalities that require timing, such as the ones offered by the functions delay(), millis(), micros(), tone() and analogWrite();



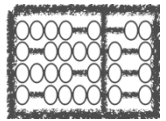


# ATMega328 Timers

TIMER0 - Registers:

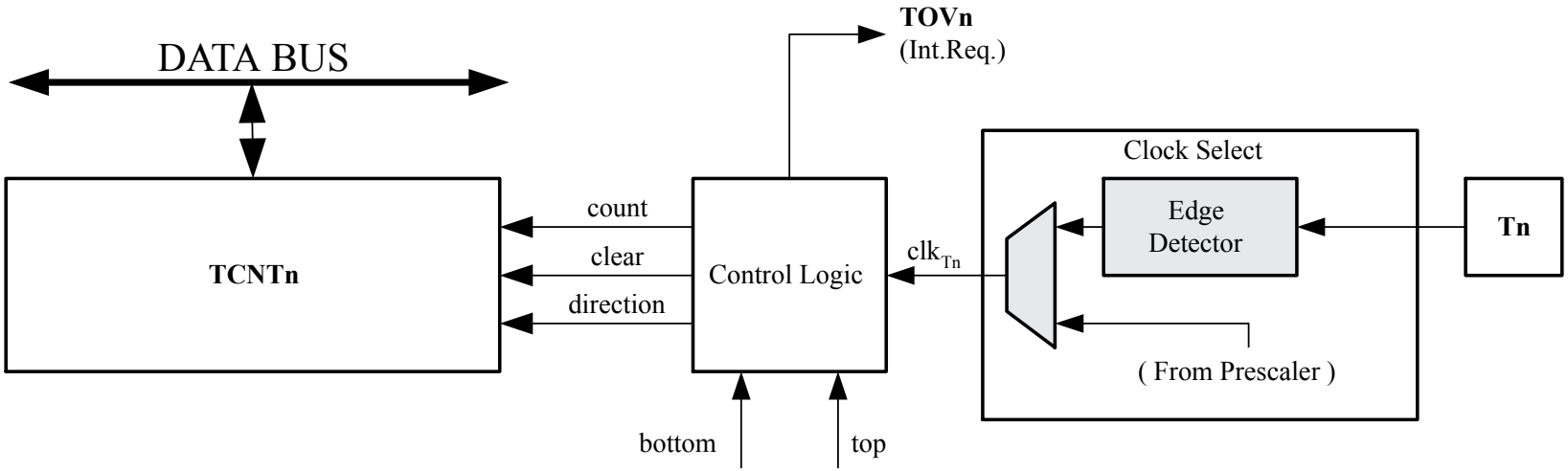
- **TCNT0**: Timer/Counter Register. The actual timer value is stored here.
- **TCCR0 A/B**: Timer/Counter Control Registers. The pre-scaler can be configured here.
- **OCR0 A/B**: Output Compare Registers
- **TIMSK0**: Timer/Counter Interrupt Mask Register. To enable/disable timer interrupts.
- **TIFR0**: Timer/Counter Interrupt Flag Register. Indicates a pending timer interrupt.



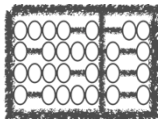


# ATMega328 Timers

TIMER0 – Counter Unit:

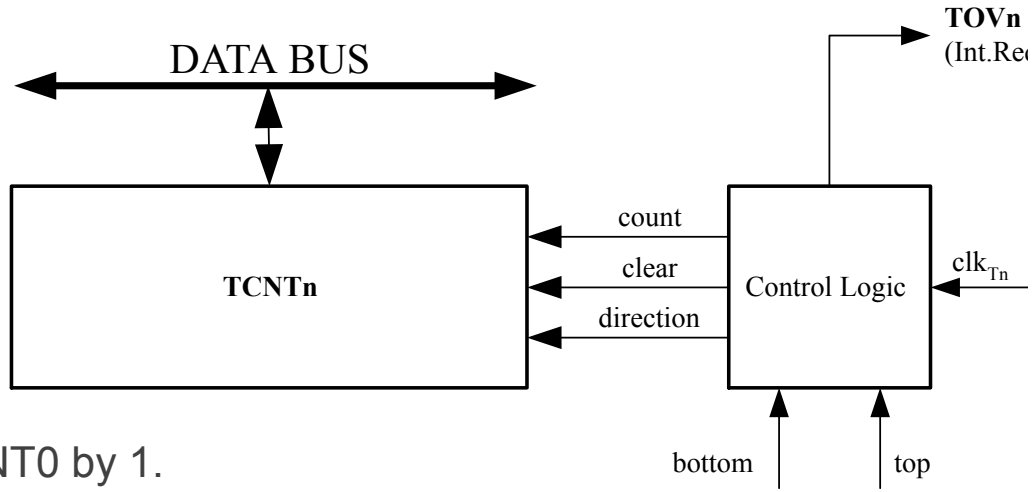




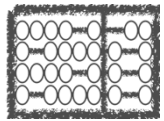
# ATMega328 Timers

TIMER0 – Counter Unit:



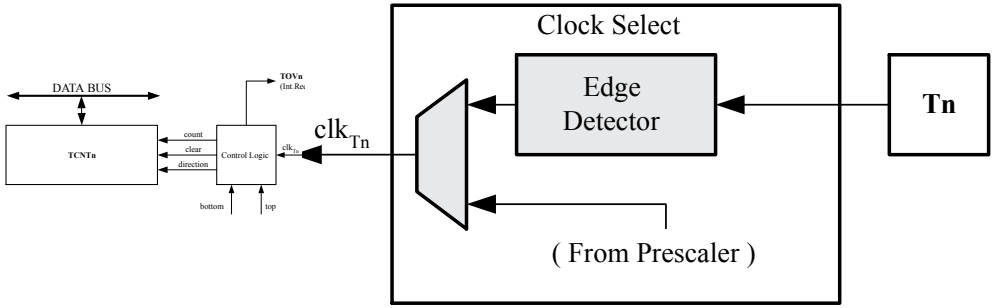

- **count**: Increment or decrement TCNT0 by 1.
- **direction**: Select between increment and decrement.
- **clear**: Clear TCNT0 (set all bits to zero).
- **clkTn**: Timer/Counter clock
- **top**: Signalize that TCNT0 has reached maximum value.
- **bottom**: Signalize that TCNT0 has reached minimum value (zero).





# ATMega328 Timers

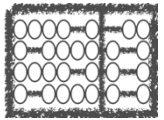
## TIMER0 – Input Clock Select




**Name:** TCCR0B
**Offset:** 0x45
**Reset:** 0x00

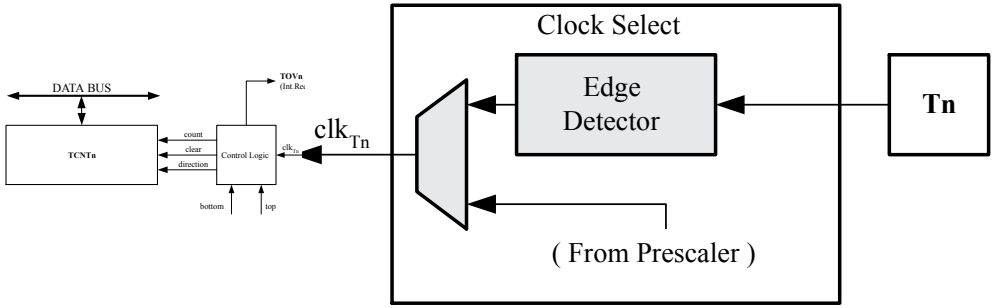
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FOC0A | FOC0B | | | WGM02 | CS0[2:0] | | |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| CS02 | CS01 | CS00 | Description |
|---|---|---|---|
| 0 | 0 | 0 | No clock source (Timer/Counter stopped). |
| 0 | 0 | 1 | $clk_{I/O}$/1 (No prescaling) |
| 0 | 1 | 0 | $clk_{I/O}$/8 (From prescaler) |
| 0 | 1 | 1 | $clk_{I/O}$/64 (From prescaler) |
| 1 | 0 | 0 | clkI/O/256 (From prescaler) |
| 1 | 0 | 1 | $clk_{I/O}$/1024 (From prescaler) |
| 1 | 1 | 0 | External clock source on T0 pin. Clock on falling edge. |
| 1 | 1 | 1 | External clock source on T0 pin. Clock on rising edge. |



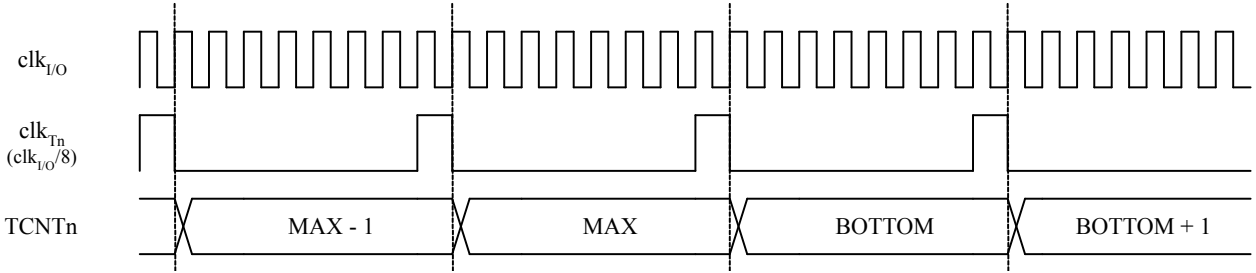
# ATMega328 Timers

## TIMER0 – Prescaler Example



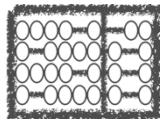

**Name:** TCCR0B
**Offset:** 0x45
**Reset:** 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FOC0A | FOC0B | | | WGM02 | CS0[2:0] | | |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |




| CS02 | CS01 | CS00 | Description |
|---|---|---|---|
| 0 | 1 | 0 | $clk_{I/O}/8$ (From prescaler) |

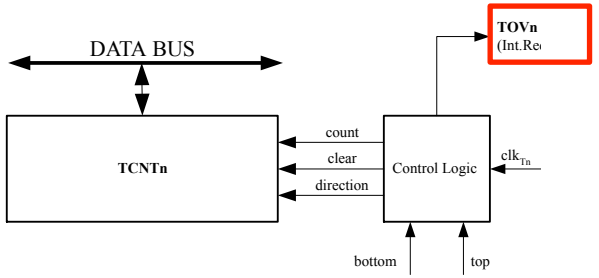


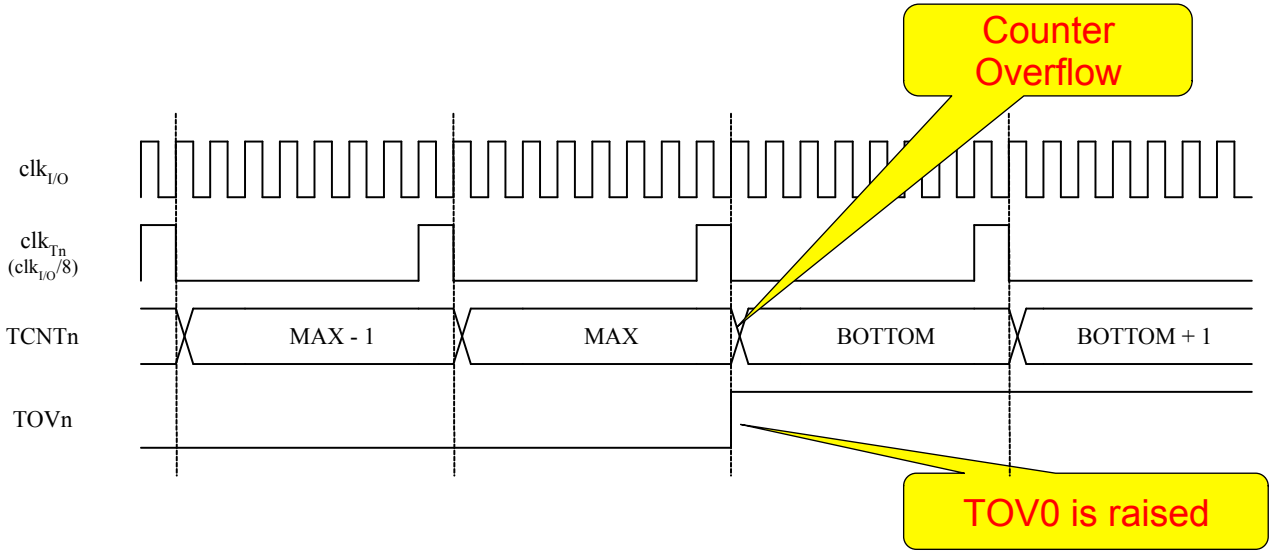


# ATMega328 Timers

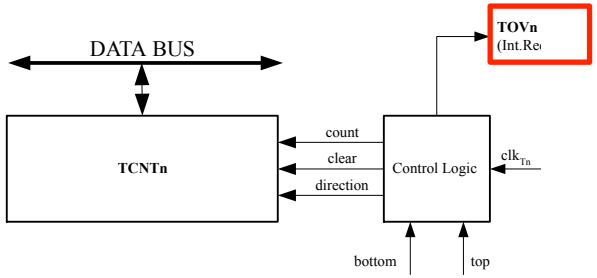
TIMER0 – Interrupt on Overflow

- Interrupt generated when counter overflows (TOV0)



# ATMega328 Timers




TIMER0 – Interrupt on Overflow

- Interrupt generated when counter overflows (TOV0)
- In case the TOIE0 bit (register TIMSK0) and the Global Interrupt Enable bit (register SREG) are set, the processing unit will handle the interrupt.
- A jump to the ISR must be registered on address 0x20.

**ATMega328 Interrupt Address Table**

| Vector No | Program Address | Source | Interrupts definition |
|---|---|---|---|
| ... | ... | ... | ... |
| 15 | 0x001C | TIMER0_COMPA | Timer/Counter0 Compare Match A |
| 16 | 0x001E | TIMER0_COMPB | Timer/Coutner0 Compare Match B |
| 17 | 0x0020 | TIMER0_OVF | Timer/Counter0 Overflow |

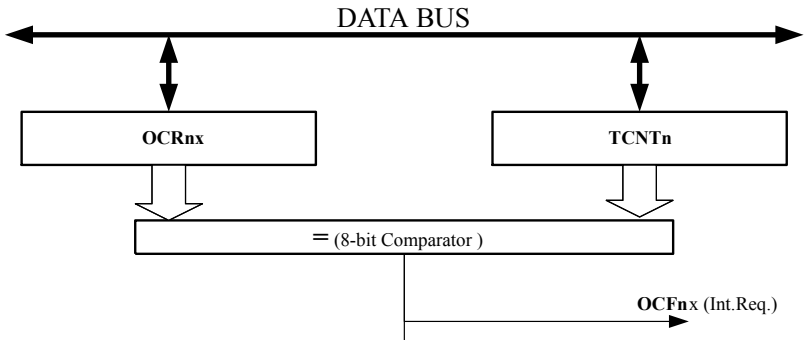
**Name:** TIMSK0
**Offset:** 0x6E
**Reset:** 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | OCIEB | OCIEA | TOIE |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

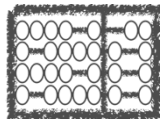
# ATMega328 Timers

TIMER0 – Output Compare Unit

- Timer counter (TCNT0) continuously compared to registers OCR0A and OCR0B
- Whenever TCNT0 equals to OCR0A or OCR0B, the comparator signals a match
- If the corresponding interrupt bit (OCIEB or OCIEA of TIMSK0) and the Global Interrupt Enable bit (SREG register) are enable, then an interrupt is generated.





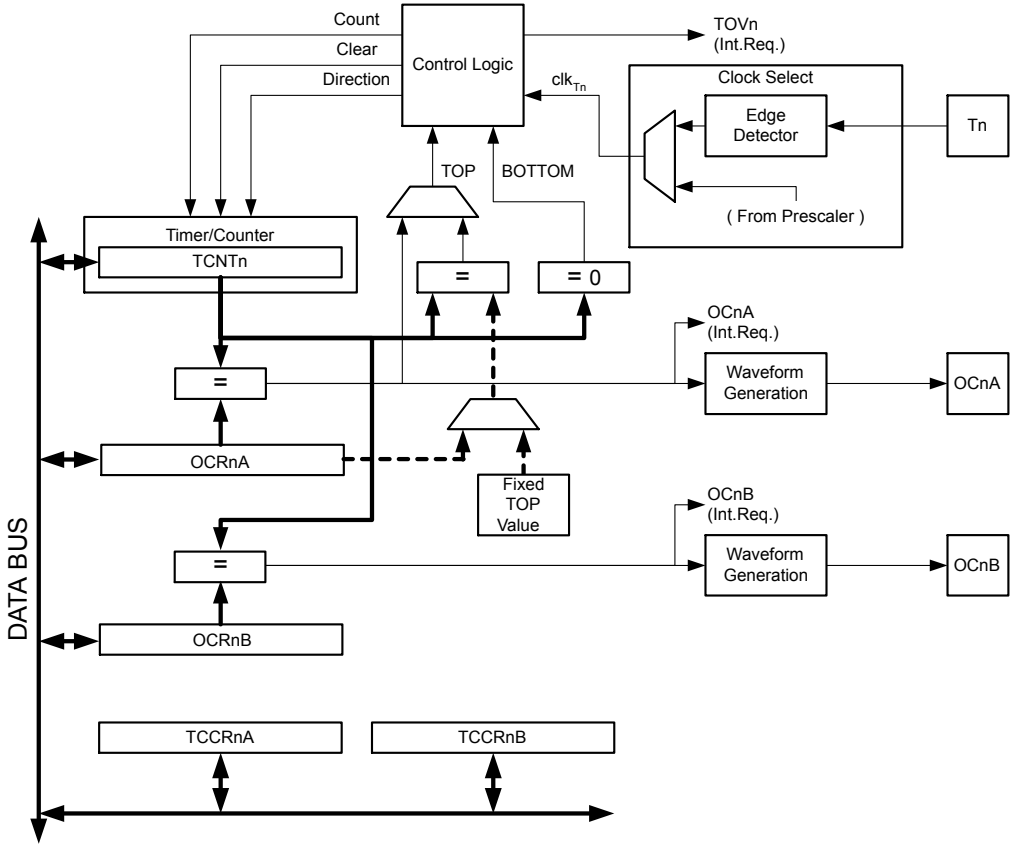



# ATMega328 Timers

TIMER0: Overall structure

Other features:

- Waveform generation (PWM)



# ATMega328 Timers

## TIMER0: Several modes

- Normal mode:
  - Timer increments
  - Wraps around at TOP = 0xFF
  - Starts again at 0
  - TOV0 interrupt flag set when TCNT0 reset to 0
  - Useful for generating interrupts every N time units

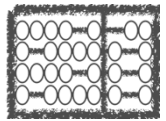
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COM0A1 | COM0A0 | COM0B1 | COM0B0 | | | WGM01 | WGM00 |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

**Name:** TCCR0A
**Offset:** 0x44
**Reset:** 0x00

| Mode | WGM02 | WGM01 | WGM00 | Timer/Counter Mode of Operation | TOP | Update of OCR0x 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 |

# ATMega328 Timers - Example

```
/* Code to blink LED when the TIMER0 counter overflows. */
#include <avr/io.h>
#include <stdbool.h>
#include <avr/interrupt.h>

ISR(TIMER0_OVF_vect)
{
  PORTB=PORTB ^ 0xFF;// Toggle PORTB output pins
}

int main( void )
{
  DDRB = 0xFF;        // Configure PORTB as output
  PORTB = 0xFF;       // Set output pins as HIGH (LED ON)
  TIMSK0=(1<<TOIE0); // Enable the interrupt on overflow for Timer0
  TCNT0=0x00;         // Set timer0 counter initial value to 0

  TCCR0B = (1<<CS02) | (1<<CS00); // Set timer0 with /1024 prescaler

  sei();              // Set the Global Interrupt Enable bit

  while(true) { /* do nothing forever. */ }
}
```



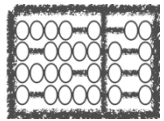
# Agenda

Timers, Counters and Watchdog Timers

ATMega328 Timers

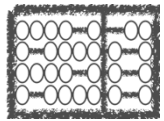
**Course Work**





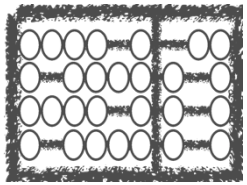
# Course Work 1: Reaction Timer





# INF-741: Embedded systems programming for IoT

Prof. Edson Borin



Institute of Computing

University of Campinas