Design with Microprocessors

Lecture 4

Year 3 CS
Academic year 2023/2024
1st Semester

Lecturer: Radu Dănescu

Timers

AVR timers

- 8 bit timers/counters
- 16 bit timers/counters

Characteristics

- Input clock prescaler
- Read / write counter status
- Waveform generator using a comparator (register)
    - Frequency tuning, PWM generator (pulse width modulation)
- Generation of interrupts at regular time intervals
- Triggered by external events (capture)

Usage

- Waveform generator
- Program synchronization with regular time intervals
- Time intervals measurement

Timers

Atmega 328P

1x 8 bit Timer0 with PWM, 1x 8 bit Timer2 PWM and Async. Operation,
1x 16 bitTimer1 with PWM

Atmega 2560

1x 8 bit Timer0 with PWM, 1x 8 bit Timer2 PWM and Async. Operation,
4x 16 bitTimer(1,3,4,5) with PWM

8 bit Timers specific features

- 2 Independent Output Compare Units
- 3 Independent Interrupt Sources (TOVx, OCFxA, and OCFxB)

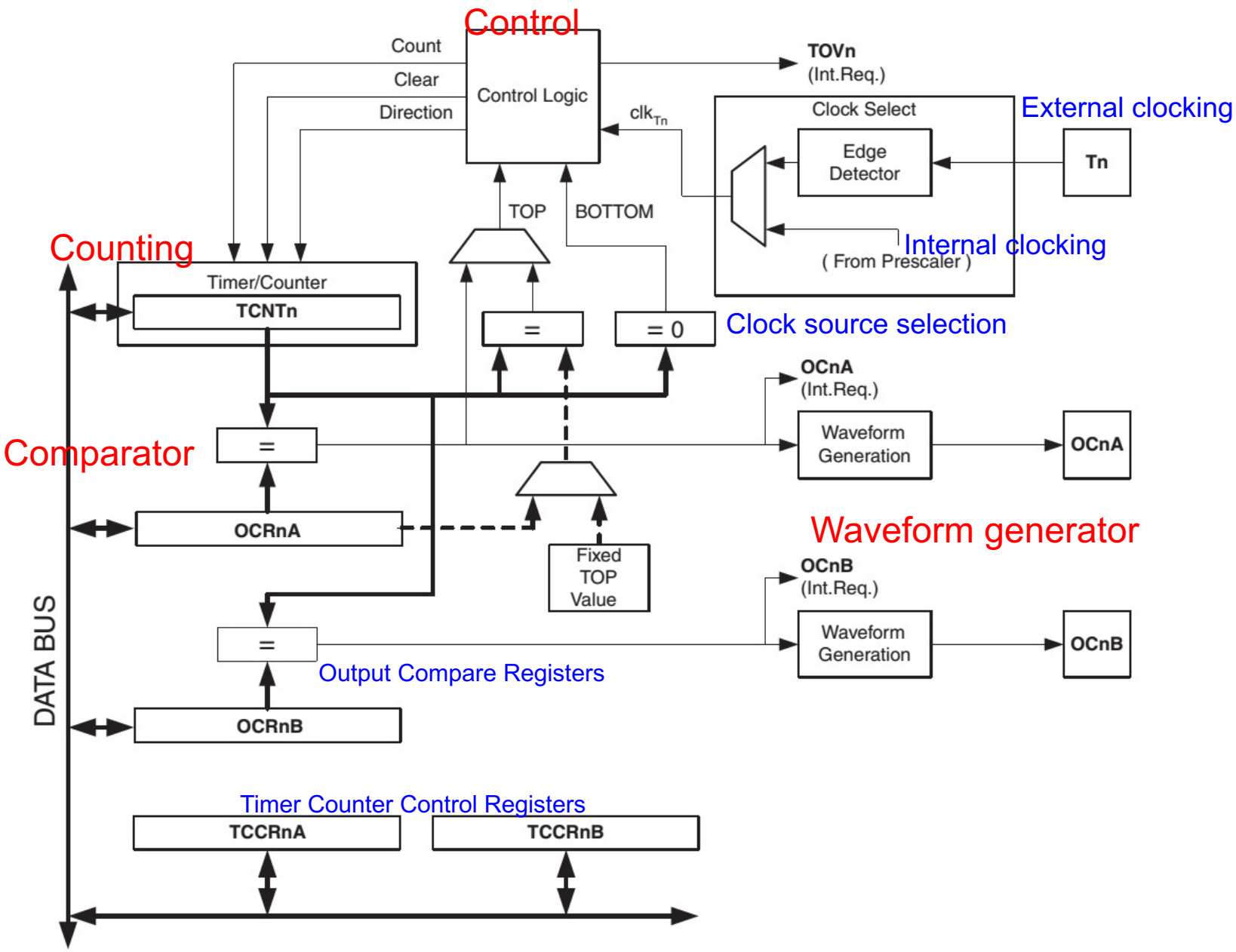
16 bit timers specific features

- 3 independent Output Compare Units
- 4 independent interrupt sources (TOVx, OCFxA, OCFxB, OCFxC, ICFx,
- 1 Input Capture Unit
- External Event Counter

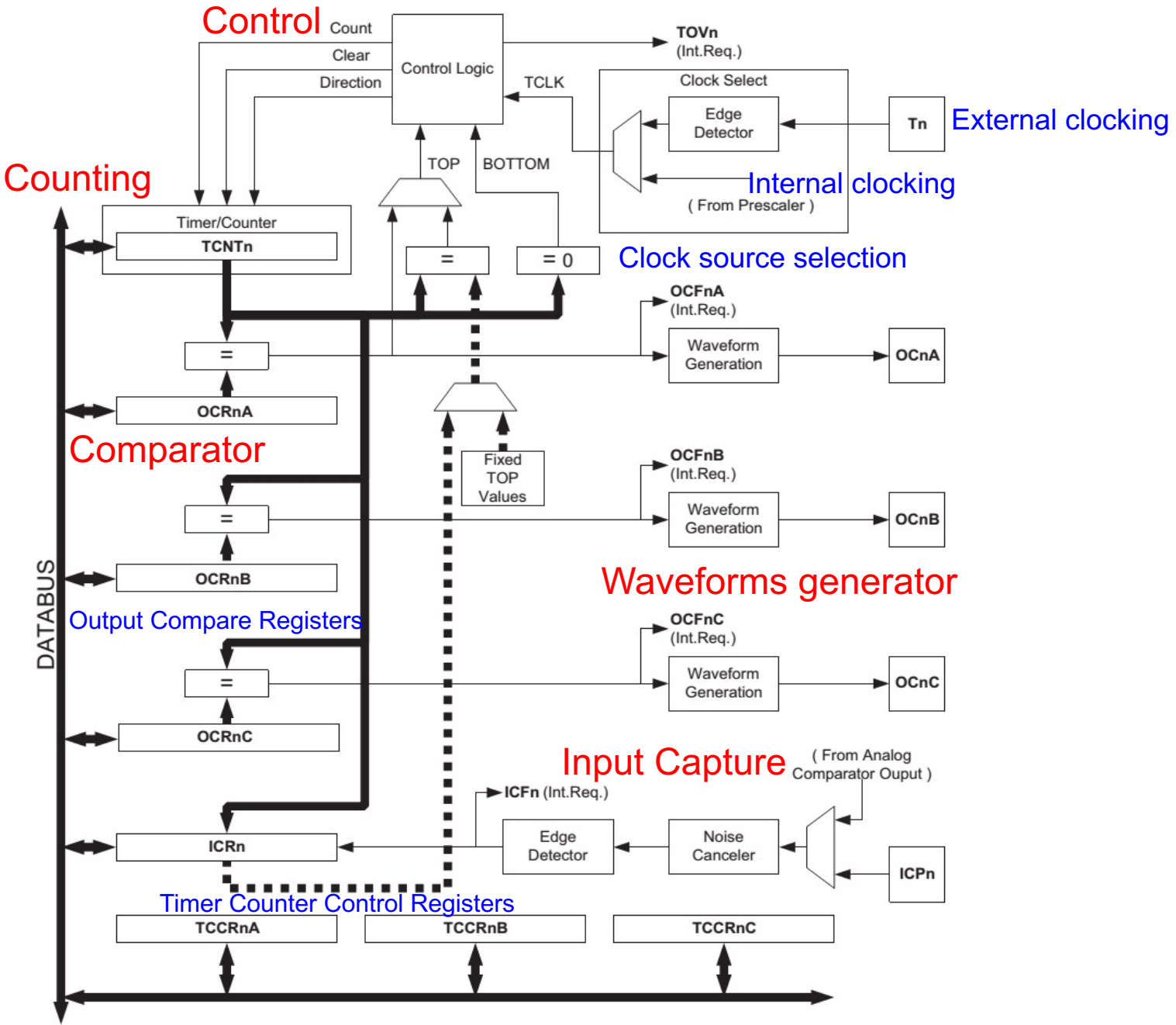
Common features

- Double Buffered Output Compare Registers
- Clear Timer on Compare Match (Auto Reload)
- Glitch Free, Phase Correct Pulse Width Modulator (PWM)
- Variable PWM Period
- Frequency Generator

Structure of 8 bit timers



Structure of 16 bit timers



8 bit timer configuration

TCCRnX registers control the timer's working mode

Force Output Compare

Clock signal configuration

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	

Compare Output Mode:
Controls the way in which the result of the comparison unit is used – depends on the wave generator mode

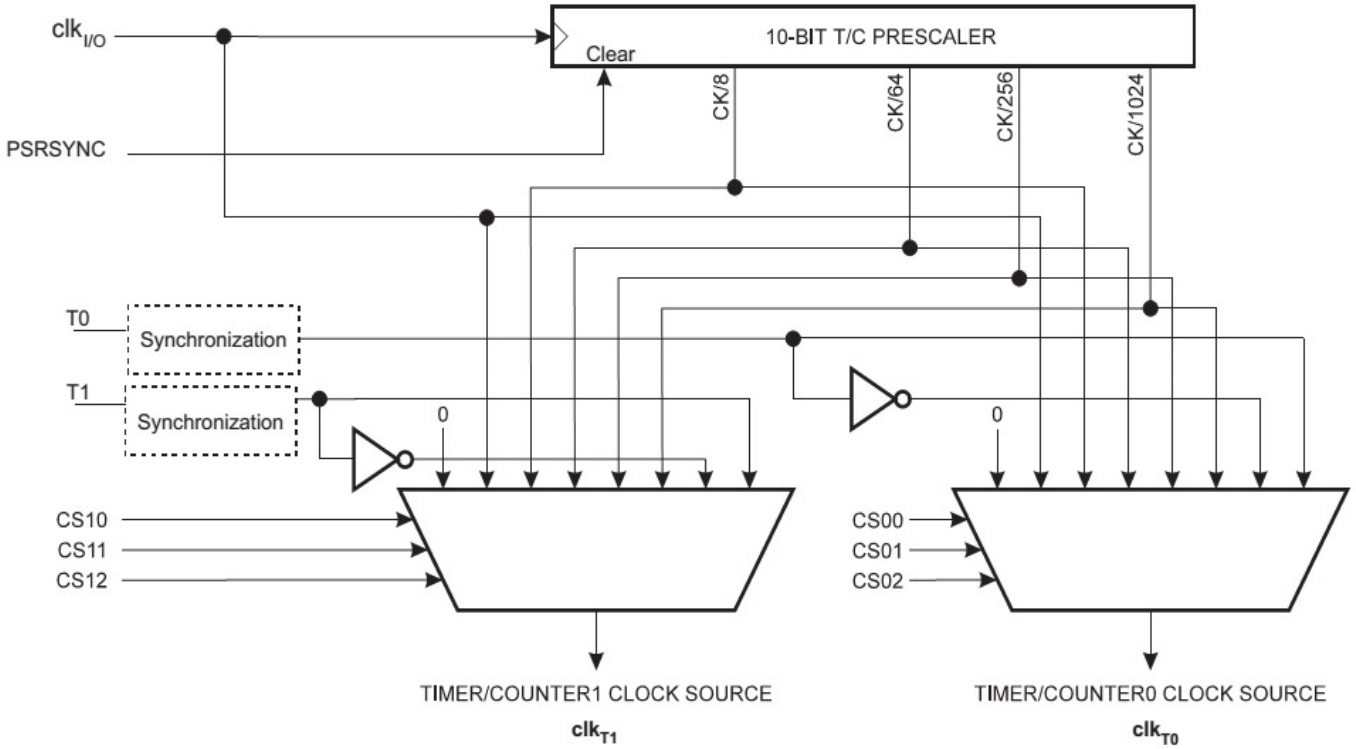
Wave Generator Mode control

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	

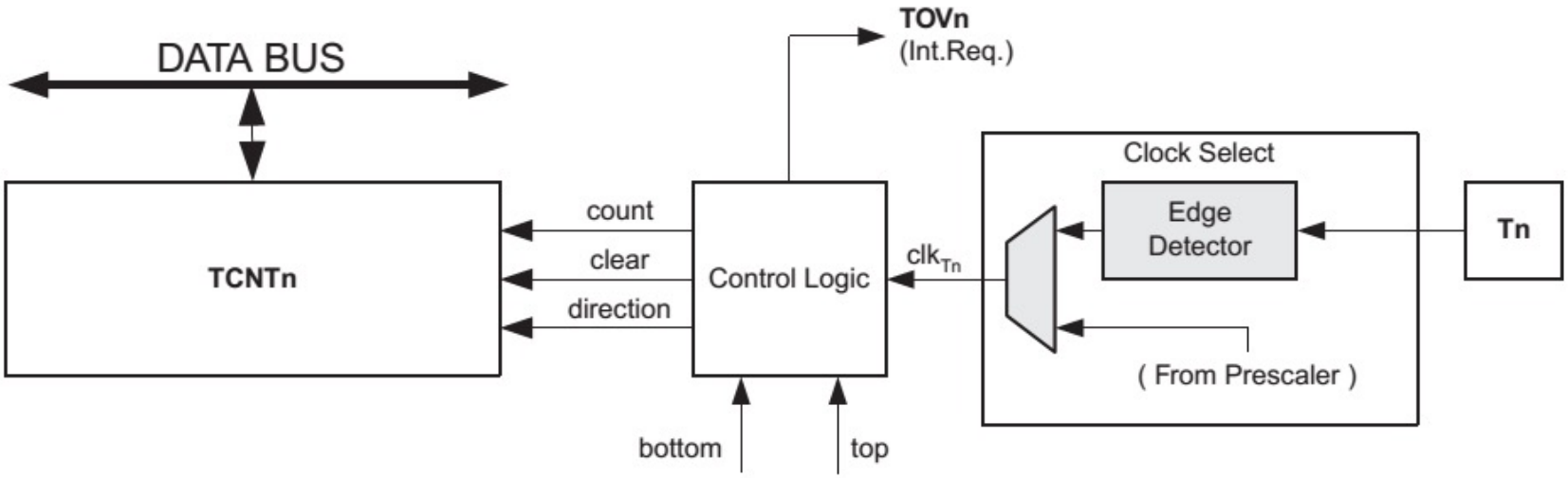
Clock signal selection

- Bits CS02 .. CS00 are controlling the clk_{TOS} division at the counter's input
- Tuning of the count speed (frequency)

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)
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.



Counter unit

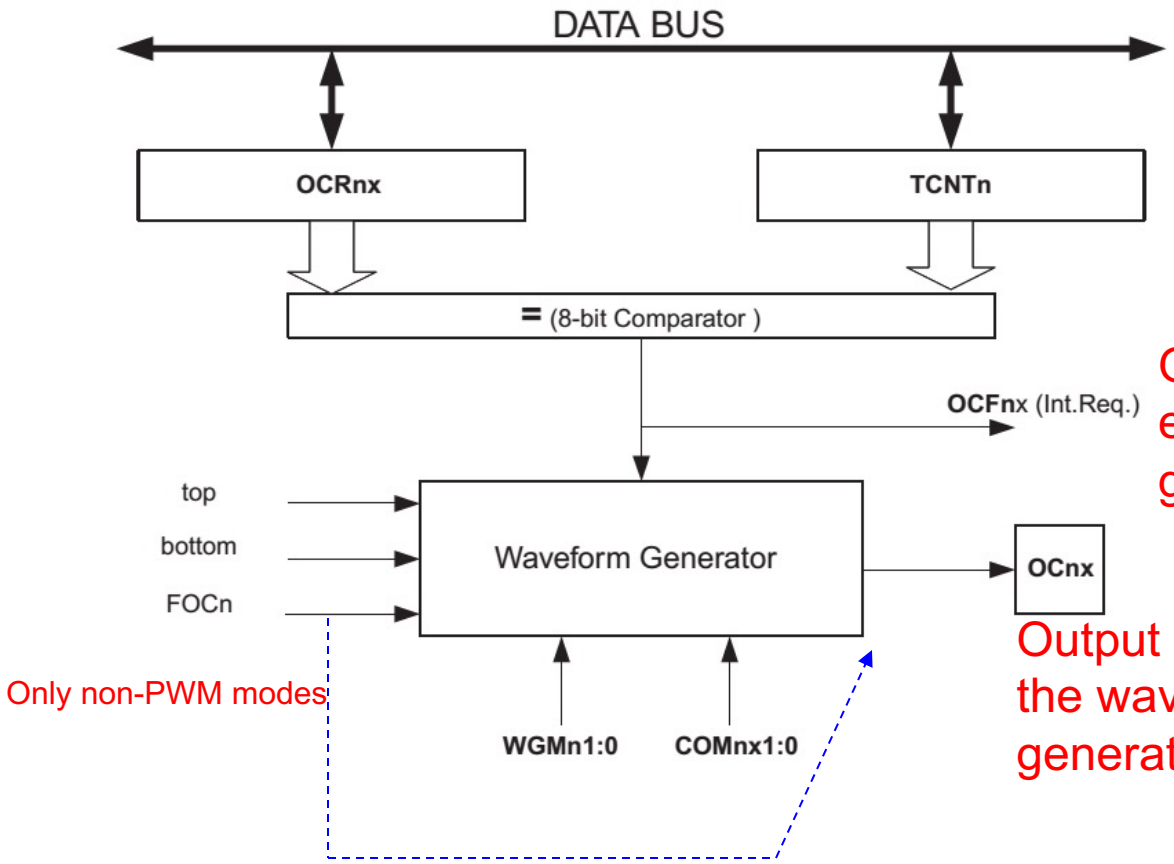


- **count** Increment or decrement TCNT0 by 1.
- **direction** Selects between increment and decrement.
- **clear** Clears TCNT0 (sets all bits to zero).
- **clk$_{T0}$** Timer/Counter clock.
- **top** Signals that TCNT0 has reached maximum value (0xFF).
- **bottom** Signals that TCNT0 has reached minimum value (zero).

- CPU – read/write the TCNTn value (override priority)
- The Timer/Counter Overflow Flag (TOV0) is set according to the mode of operation selected by the WGM02:0 bits. TOV0 can be used for generating a CPU interrupt.

Comparison unit

- Comparison between the count register (**TCNT0)** and the output compare register (**OCR0)** ⇒ used to generate different types of waveforms

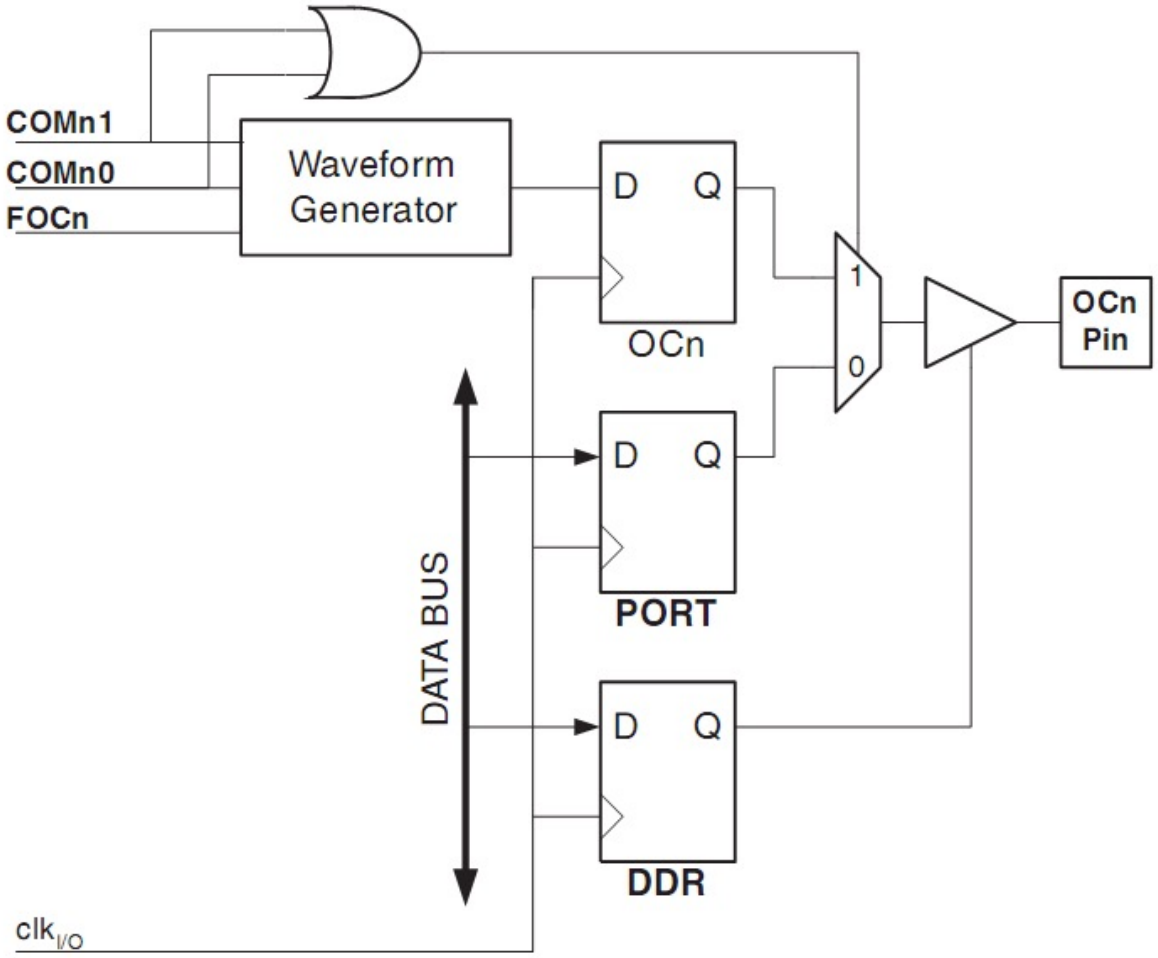


Output compare flag – at equal, interrupt request is generated

Output compare bit – here the waveform will be generated

Comparison unit (cont)

- Generated waveforms are visible through the I/O ports' pins
- The I/O port pin corresponding to OCn should be configured as output



Waveform types (functioning modes)

Table 14-8. Waveform Generation Mode Bit Description

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

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

WGM02:0 bits in combination with
COM1:0 bits are defining the timer behavior

Table 14-2. 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

Table 14-3. Compare Output Mode, Fast PWM Mode[1]

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match, set OC0A at BOTTOM, (non-inverting mode).
1	1	Set OC0A on Compare Match, clear OC0A at BOTTOM, (inverting mode).

Table 14-4. Compare Output Mode, Phase Correct PWM Mode[1]

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match when up-counting. Set OC0A on Compare Match when down-counting.
1	1	Set OC0A on Compare Match when up-counting. Clear OC0A on Compare Match when down-counting.

Waveform types (functioning modes)

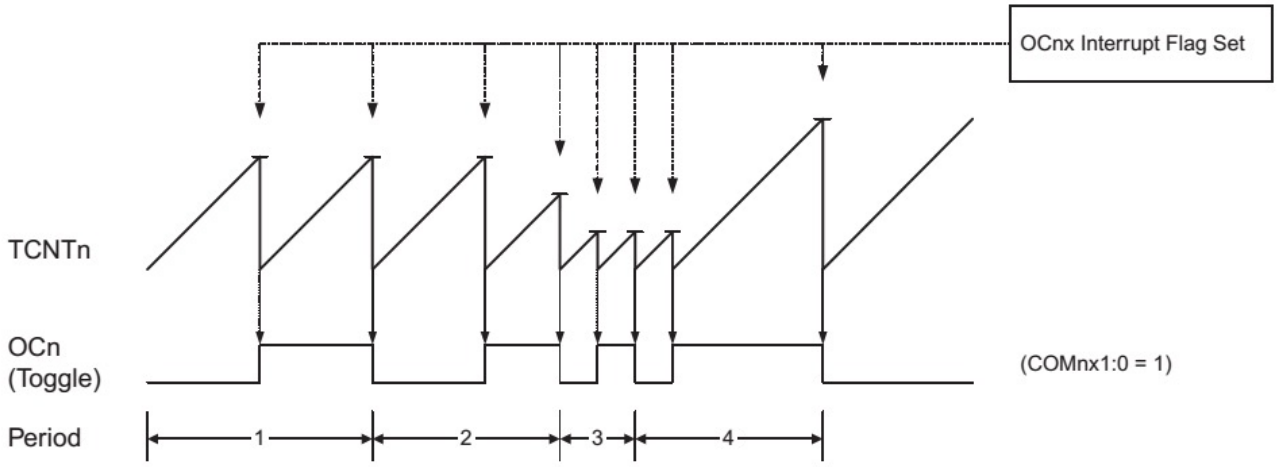
Normal

- Simple counting (incrementing): 0 … 255
- When the counter overruns (0xFF), a timer overflow interrupt is generated and TOV0 flag is set then the counting is restarted from 0x00

Homework: compute the TOV0 frequency for various clock prescaler values

CTC – Clear Timer on Compare Match (variable frequency generator)

- When the counter value (TCNT0) reaches the OCR0 value, the counter is cleared to 0
- Generated waveform frequency can be set by writing OCR0 register
  - ex: COM01:COM00 = 01 – OC0 toggles at equality



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

N = prescale factor

(1, 8, 64, 256, 1024)

Timer generated interrupts

- Events that are generating interrupts:
  - *Overflow*
  - *Compare match*
  - External event (*capture*) – available only for 16 bits timers

	Address		Description
14	$001A	TIMER2 COMPA	Timer/Counter2 Compare Match A
15	$001C	TIMER2 COMPB	Timer/Counter2 Compare Match B
16	$001E	TIMER2 OVF	Timer/Counter2 Overflow
17	$0020	TIMER1 CAPT	Timer/Counter1 Capture Event
18	$0022	TIMER1 COMPA	Timer/Counter1 Compare Match A
19	$0024	TIMER1 COMPB	Timer/Counter1 Compare Match B
20	$0026	TIMER1 COMPC	Timer/Counter1 Compare Match C
21	$0028	TIMER1 OVF	Timer/Counter1 Overflow
22	$002A	TIMER0 COMPA	Timer/Counter0 Compare Match A
23	$002C	TIMER0 COMPB	Timer/Counter0 Compare match B
24	$002E	TIMER0 OVF	Timer/Counter0 Overflow

Timer generated interrupts

- Enable / disable interrupts – TIMSKx register (accessible with I/O instructions)

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Overflow Timer 0

Compare match Timer 0

- Accessing interrupt state –TIFRx register

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	–	–	–	–	–	OCF0B	OCF0A	TOV0	TIFR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Possible usage of timer interrupts
  - Generation of software waveforms
  - Constant timing for different events – ex: SSD cells, LED matrix Rows/Columns shifting etc.
  - Parameters changing for hardware waveforms

Examples

- **Example 1** – specified (constant) frequency waveform generation

- Problem statement: generate a 50 Hz signal
- Mode: CTC (Clear on Compare Match) – allows the setting of the signal period by setting the OCR content
- Frequency computing:

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

- f_{OCn} = 50
- N = 1024 = maximum division allowed by the prescaler
- Fclk_io = 16,000,000 = 16 MHz, MCU frequency
- OCR0 = 16,000,000 / (2*1024*50) – 1 = 154

Examples

- **Example 1** – specified frequency waveform generation

```
.org 0x0000
jmp reset

reset:
  ldi r16, 0b01000010
  out TCCR0A, r16          ; configure Timer0A

  ldi r16, 0b00000101
  out TCCR0B, r16

  ldi r16, 154             ; computed OCR
  out OCR0A, r16

  ldi r16, 0xff
  out DDRB, r16            ; activates timer output OC0A

donothing:
rjmp donothing ; waveform can be monitored using an oscilloscope on OC0A
                                (PB7 for MEGA, PD6 for UNO) pin !!!
```

0	1	0	CTC

Table 14-2. Compare Output Mode, non-PWM Mode

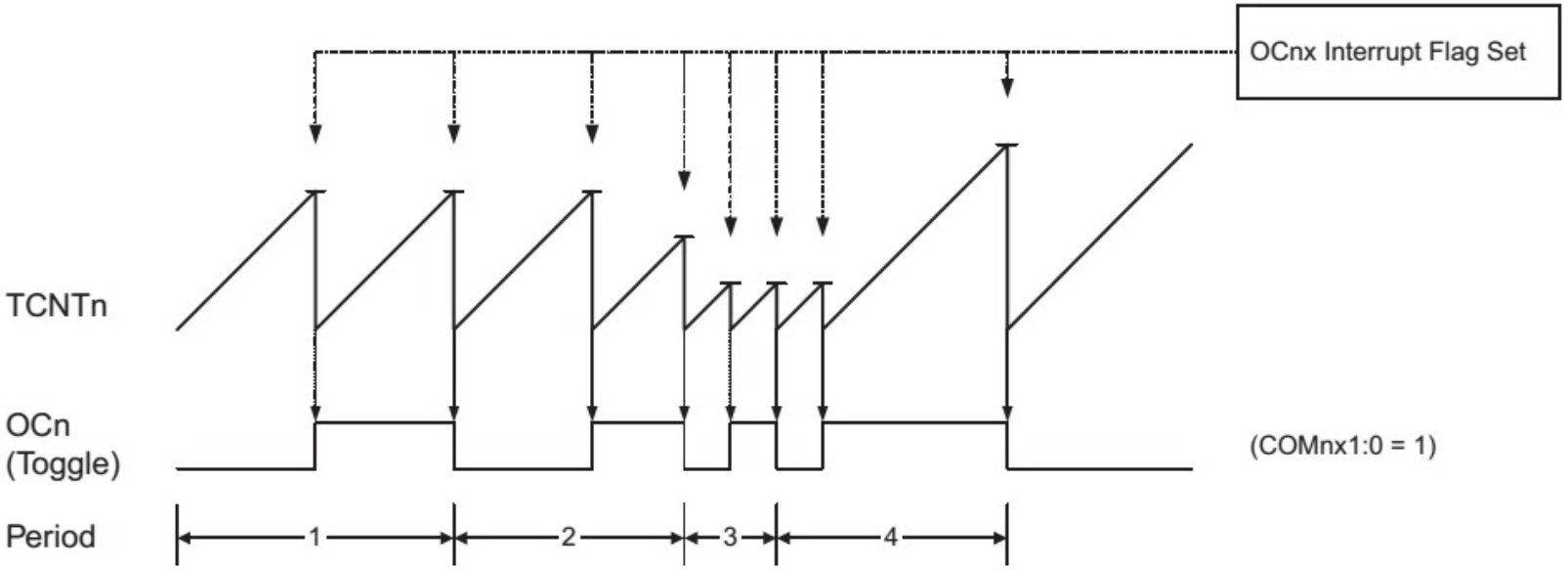
COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter sto
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)
1	1	0	External clock source on T0 pin. Clo
1	1	1	External clock source on T0 pin. Clo

Examples

- **Example 2** – usage of interrupts at compare match

- Problem statement: generate a signal with a 1 sec period / 1Hz (impossible by directly setting the timer presacler: f_{MIN} = 30 Hz with the prescaler at 1024)
- Configuration used in example 1
- Used frequency – 50 Hz, with Toggle on CTC mode ⇒ 2 equalities in 1/50 sec. ⇒ 1 eq. at 1/100 sec. ⇒ one Comp Match Interrupt at every 10 ms
- We will use the interrupt generated by the compare match (TCNT = OCR)
- We will toggle a signal at every 50 such events and to generate a **low** signal of 500 ms long + 50 such events and to generate a **high** signal for a 500 ms period
- The resulted signal will have a 1s period



Examples

- **Example 2** – usage of interrupts at compare match

```
.org 0x0000
jmp reset

.org 0x002A                ; address for Timer0 CompA ISR
jmp timercpm

reset:
ldi r16, low(RAMEND)       ; interrupts are using the stack
out SPL, r16
ldi r16, high(RAMEND)
out SPH, r16

ldi r16, 0b01000010
out TCCR0A, r16            ; same configuration as in example 1
ldi r16, 0b00000101
out TCCR0B, r16
ldi r16, 154
out OCR0A, r16

ldi r16, 0xff              ; Connect LEDs as outputs (1 sec period signal)
out DDRA, r16
```

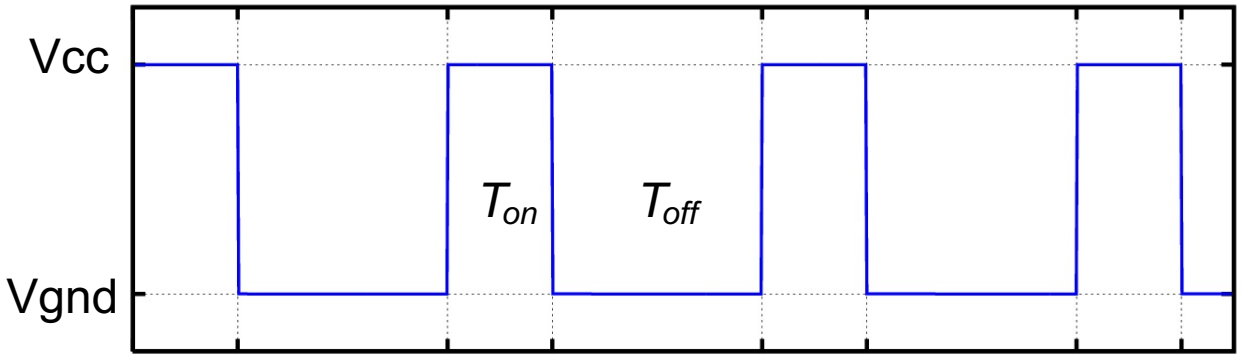
Examples

- **Example 2** – – usage of interrupts at compare match

```
ldi r16, 0b00000010          ; activate Timer0 CompA interrupt
out TIMSK0, r16

ldi r18, 0                   ; event counter
ldi r19, 0                   ; used for output signal toggle
sei                          ; global interrupt system activation

loop:                        ; main program will "stuck" here
rjmp loop

timercpm:                    ; ISR called at every 10 ms (at  Timer0 CompA Match)
  inc r18                    ; increments the event counter
  cpi r18, 50
  brne exit                  ; not reached 50 ⇒ exit
 ; if reached 50 events
  com r19                    ; toggle r19 when reached 50 count (500 ms period)
  out PORTA, r19             ; display content of LEDs
  ldi r18, 0                 ; reset event counter
 exit: ; else (no events < 50) ⇒ do nothing
reti
```

Pulse Width Modulation (PWM)

- ?Some systems are requiring control trough the variation of the input voltage (average input voltage / current / power)
  - Ex: motor speed, LED power
- Digital systems can produce at output only 2 values: 0 (GND) si 1 (Vcc)
- Variable power can be reached through the duty cycle *D (duty cycle)* variation



$$D = \frac{T_{on}}{T_{on} + T_{off}}$$

- Average voltage (and consequently current / power)

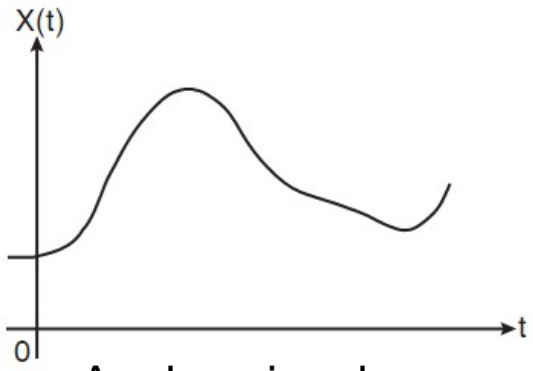
$$V_{AVG} = DV_{CC} + (1 - D)V_{GND}$$

•If V_{GND} = 0:

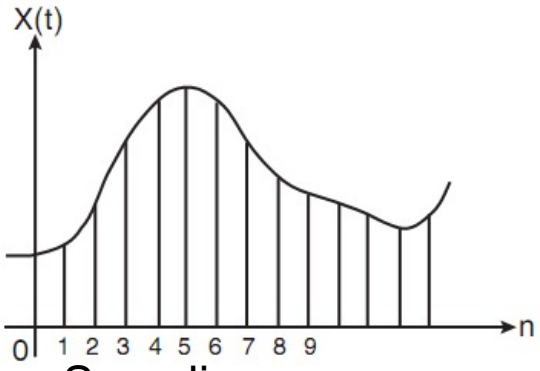
$$V_{AVG} = DV_{CC}$$

Pulse Width Modulation (PWM)

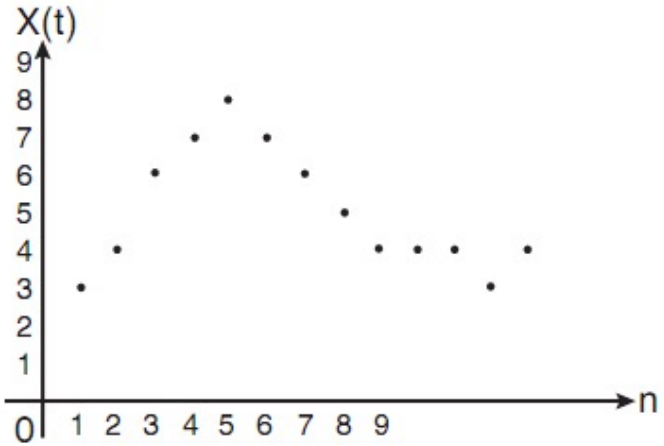
- Low frequency analog signal (ex. sound) can be coded by PWM
- Steps: sampling, digitization, computing D based on the digital value



Analog signal



Sampling



Digitization

Computing the fill factor: D=k*X(n)

Pulse Width Modulation (PWM)

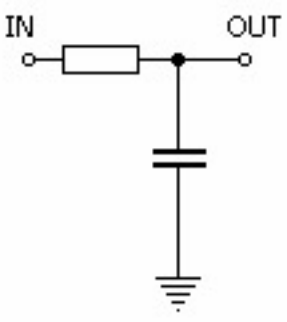
Using the PWM signal

- Can be used as it is, if the application allows it: variable LED power, DC motor speed, etc (the device inertia produces the power averaging effect)

  Ex:

  - speed control of a robot driven by DC motors: speed ~ D
  - brightness control of a LED: brightness ~ D

- *C*an be filtered using a Low-Pass filter to “rebuild” the analog signal:
  - The upper limit of the analog signal frequency (*cutoff frequency*) is set (much smaller than the frequency of the carrying signal)
  - Simple low pass filter: RC

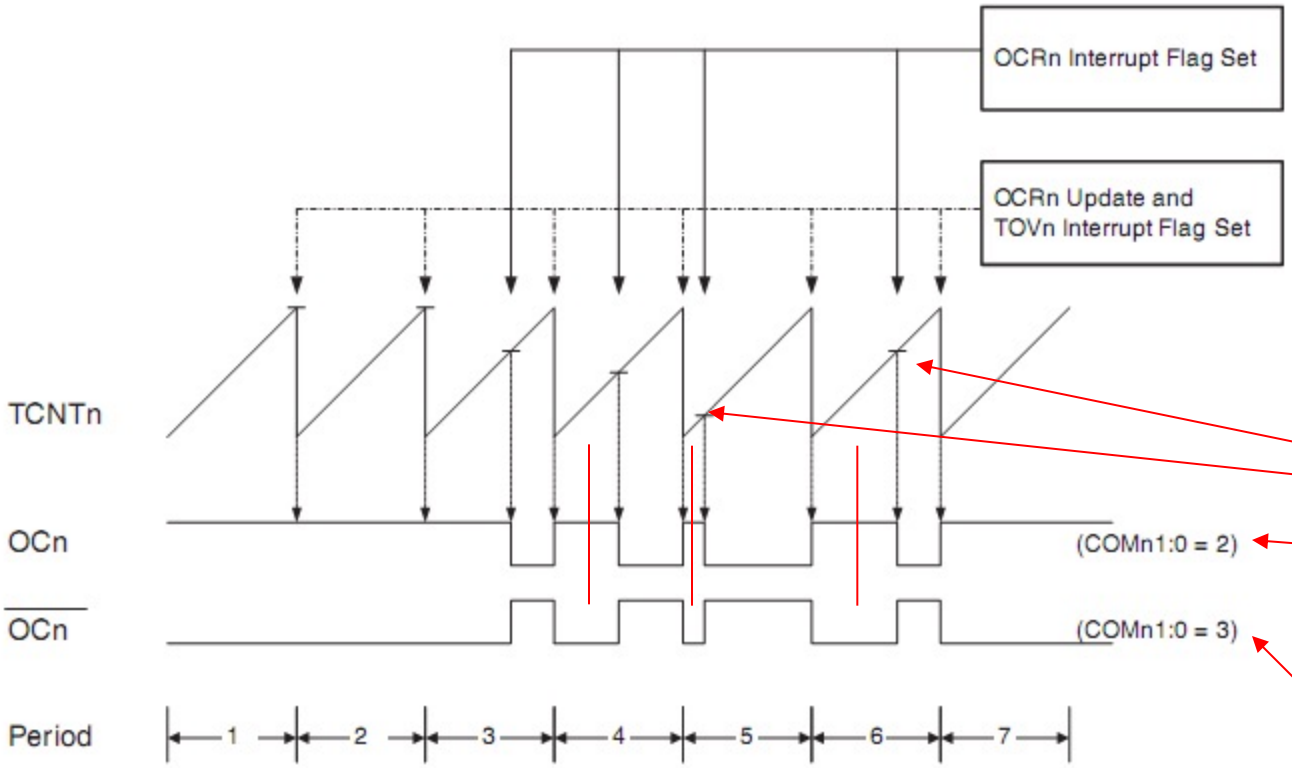


$$f_{cutoff} = \frac{1}{2\pi RC}$$

Waveform types (functioning modes)

Fast Pulse Width Modulation (PWM) mode

- PWM signals generation (“modulare in latimea pulsului”) on OC0
- Fill factor (“factorul de umplere”) is set by writing the value of OCR0 register
- Frequency is fixed, given by the Clock Select (CS) bits (prescaler setting)
- Fill factor = OCR0 / 255 (Ton / T, T= Ton+ Toff)



$$f_{OCnPWM} = \frac{f_{clk\_I/O}}{N \cdot 256}$$

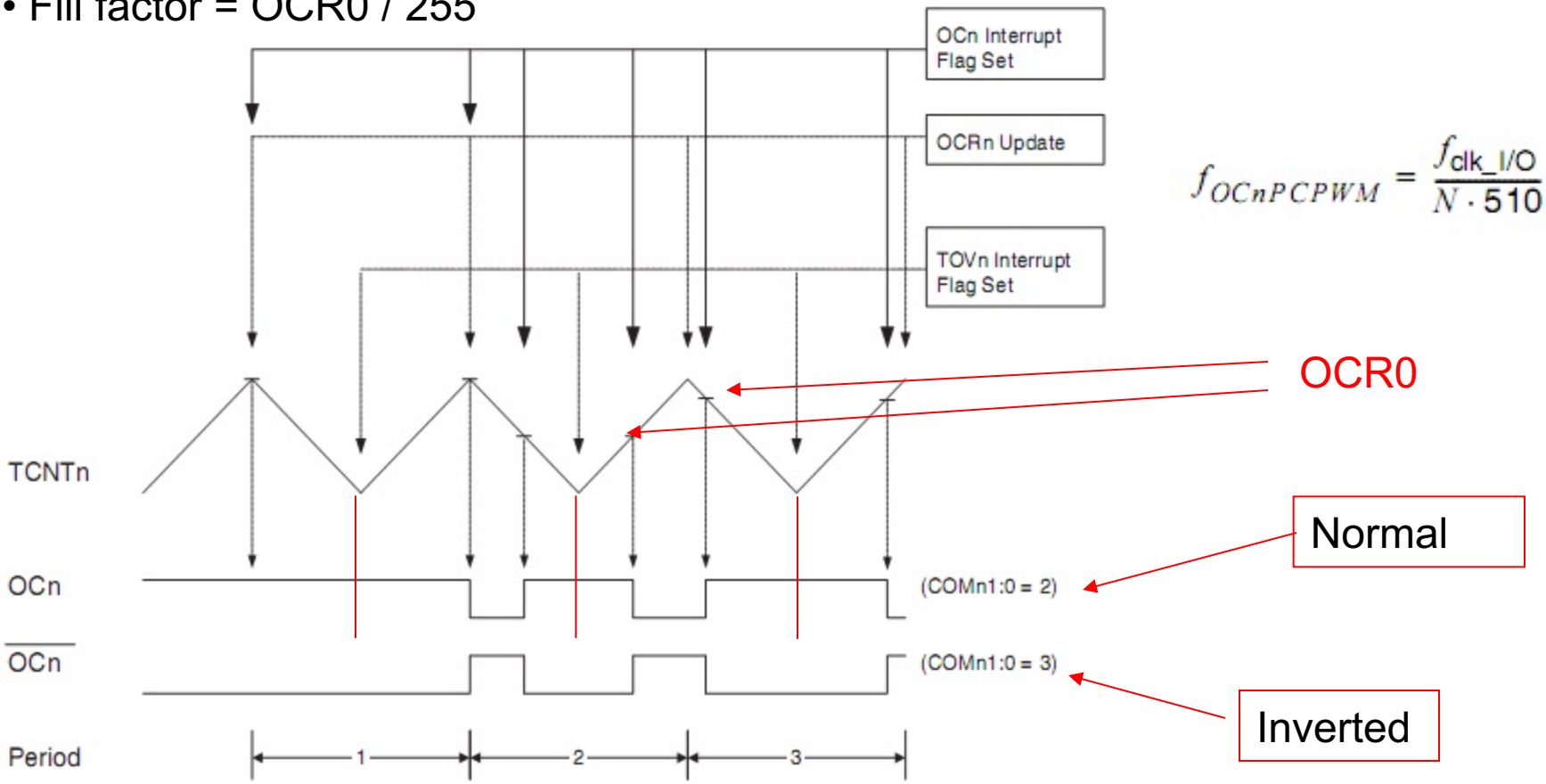
OCR0

‘0’ at equality,
‘1’ at overflow

‘1’ at equality,
‘0’ at overflow

Waveform types (functioning modes)

- **Phase Correct Pulse Width Modulation (PWM) mode**
  - PWM signals generation with phase correction
  - The pulse is symmetric relative to the period's mid point (TCNT = BOTTOM)
  - Fill factor is set by writing OCR0 register
  - Upward/downwards counting, output changes at successive equalities (compare matches)
  - Fill factor = OCR0 / 255

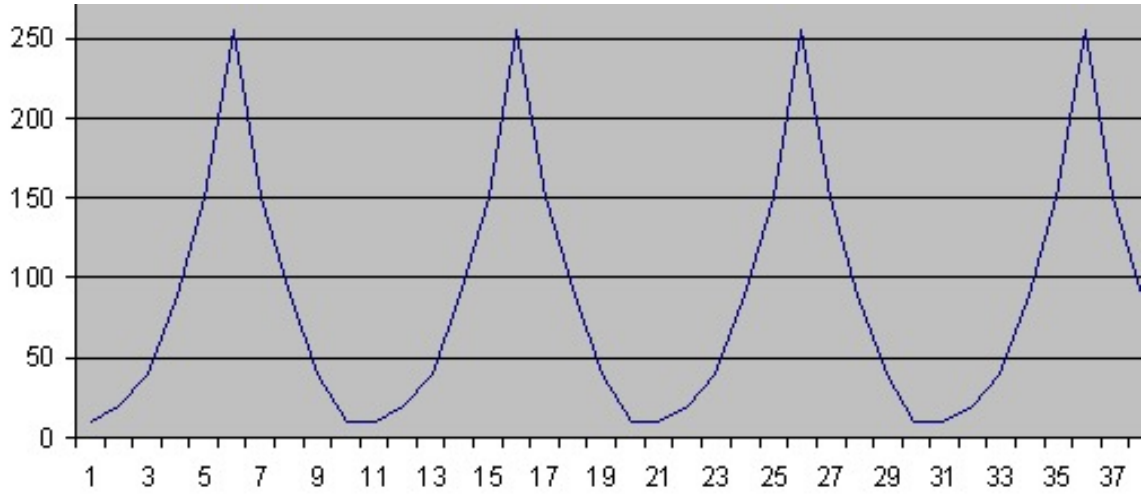


$$f_{OCnPCPWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

Examples

- **Exemple 3** – PWM usage

- Problem statement: generate an analog signal
- Function will be defined by discrete values stored in a LUT (can be the result of an ADC process)
- PWM phase correct mode
- OCR0 will define the pulse width
- We will change OCR0 at the end of each counting period
- Values of a period: 10, 20, 40, 90, 150, 255, 150, 90, 40, 20, 10



Examples

- **Example 3** – PWM usage

```
.org 0x0000
jmp reset
.org 0x002E          ; Timer 0 OVF ISR
jmp timerovf

reset:
ldi r16, low(RAMEND)
out SPL, r16
ldi r16, high(RAMEND)
out SPH, r16

ldi r16, 0b10000001
out TCCR0A, r16
ldi r16, 0b00000001
out TCCR0B, r16

ldi r16, 0xff
out DDRB, r16        ; activates OC0A output

ldi r16, 0b00000001 ; activatesTimer0 Ovf interrupt
out TIMSK0, r16
```

1	0	0	1	PWM, Phase Correct

Table 14-4. Compare Output Mode, Phase Correct PWM Mode[1]

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match when up-counting. Set OC0A on Compare Match when down-counting.
1	1	Set OC0A on Compare Match when up-counting. Clear OC0A on Compare Match when down-counting.

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stop
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)
1	1	0	External clock source on T0 pin. Cloc
1	1	1	External clock source on T0 pin. Cloc

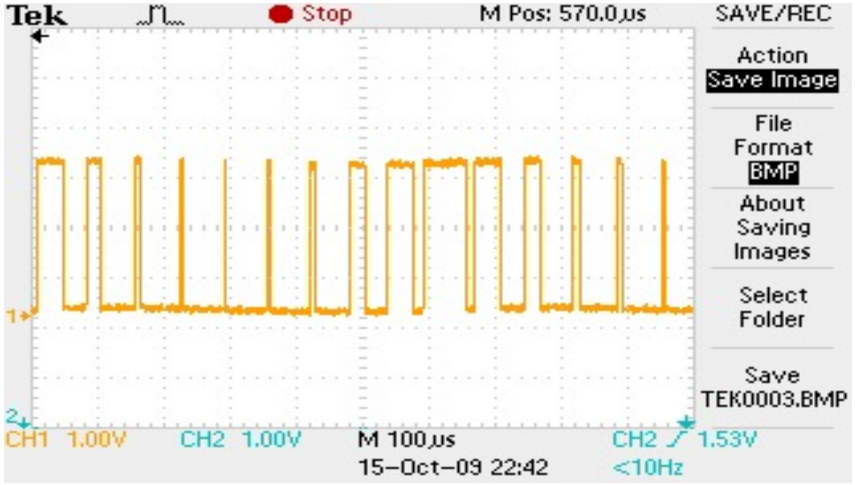
Examples

- **Example 3** – PWM usage

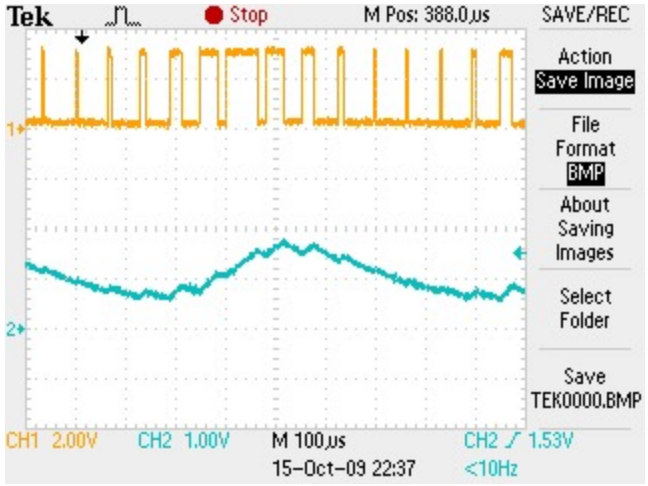
```
ldi r18, 0  ; LUT index
sei         ; global interrupt enable

loop: rjmp loop      ; main program will “stuck” here

timerovf:            ; Timer 0 Ovf ISR – called at the end of a counting period
   ldi r17,0         ; compute address in the LUT
   ldi zh, high(2*translut)
   ldi zl, low(2*translut)
   add zl, r18
   adc zh, r17
   lpm r17, Z
   out OCR0A, r17 ; LUT value set into OCR
   inc r18
   cpi r18, 10       ; maximum address in the LUT
   brne exit         ; if r18 < 10  ⇒ go to exit label (do nothing)
   ldi r18,0         ; else (r18 = 10) ⇒ reset the LUT index
  exit:
reti
translut:  ; LUT – defines the function
.db 10, 20, 40, 90, 150, 255, 150, 90, 40, 20, 10
```

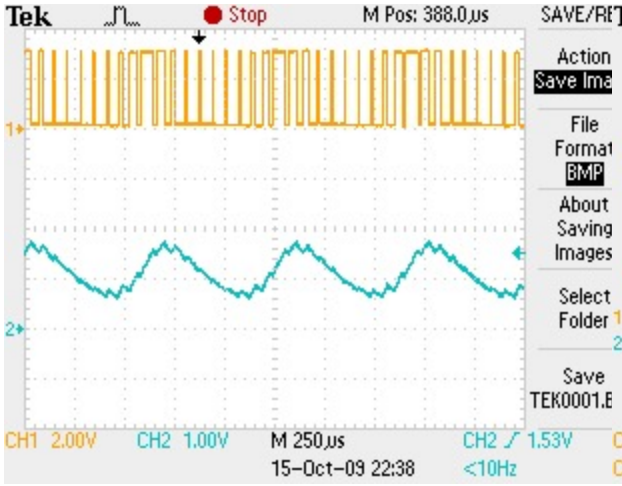
Examples

- **Example 3** – PWM usage



- By adding an RC low pas filter, R = 1K, C = 0.22 uF $\Rightarrow$ an analogue waveform:





Examples

- **Example 4**
- Problem statement: measure the period between two external events
- Configuration – Normal, (minimum counting frequency)
- External event generates an external interrupt (rising edge)

```
.org 0x0000
jmp reset
.org 0x0002          ; ISR for ext. interrupt (nr=0)
jmp int0ISR

reset:
ldi r16, low(RAMEND)
out SPL, r16
ldi r16, high(RAMEND)
out SPH, r16

ldi r16, 0b00000000
out TCCR0A, r16
ldi r16, 0b00000101
out TCCR0A, r16

ldi r16, 0xff
out DDRA, r16      ; activate outputs (port A – LEDs)
```

WGM02	WGM01	WGM00	Timer/Counter Mode of Operation
0	0	0	Normal

Table 14-2. 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

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counte
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)

Examples

- **Example 4** – reading the timer status

```
ldi r16, 0b00000011        ; configure INT0 – rising edge
sts EICRA, r16
ldi r16, 0b00000001        ; activate INT0
out EIMSK, r16

ldi r17, 0                 ; last value of the counter
sei

loop: rjmp loop

int0ISR:
in r16, TCNT0              ; read the counter register value
mov r18, r16
sub r16, r17               ; subtracts from it the previous value
mov r17, r18               ; new state becomes the old state
out PORTA, r16             ; display the difference
reti
```

Further study (for homework, lab & project)

Muhammad Ali **Mazidi**, Sarmad Naimi, Sepehr Naimi.

The AVR Microcontroller and Embedded Systems Using Assembly And C, 1st Edition, Prentice Hall, 2009.

http://www.microdigitaled.com/AVR/AVR_books.htm

Source code for the examples from the book:

http://www.microdigitaled.com/AVR/Code/AVR_codes.htm