بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

میکرو کنترلرهای AVR

# PWM

دانشکده مهندسی برق

دانشگاه صنعتی شاهرود

حسین خسروی

Introduction

➢ Previously, we learnt two features of a timer:

❑ overflow interrupt and

❑ input capture.

➢ Overflow interrupt

❑ triggered when timer reaches its limit;

❑ used to measure interval that is longer than one timer cycle.

❑ for finding the time elapse, creating a time delay.

➢ Input capture

❑ an interrupt triggered when there's a change in pin ICP1.

❑ value of Timer 1 is automatically stored in register ICR1.

❑ for finding period/frequency/pulse width of a signal.

Pulse Width Modulation

Pulse Width Modulation

0% Duty Cycle

25% Duty Cycle

50% Duty Cycle

75% Duty Cycle

100% Duty Cycle
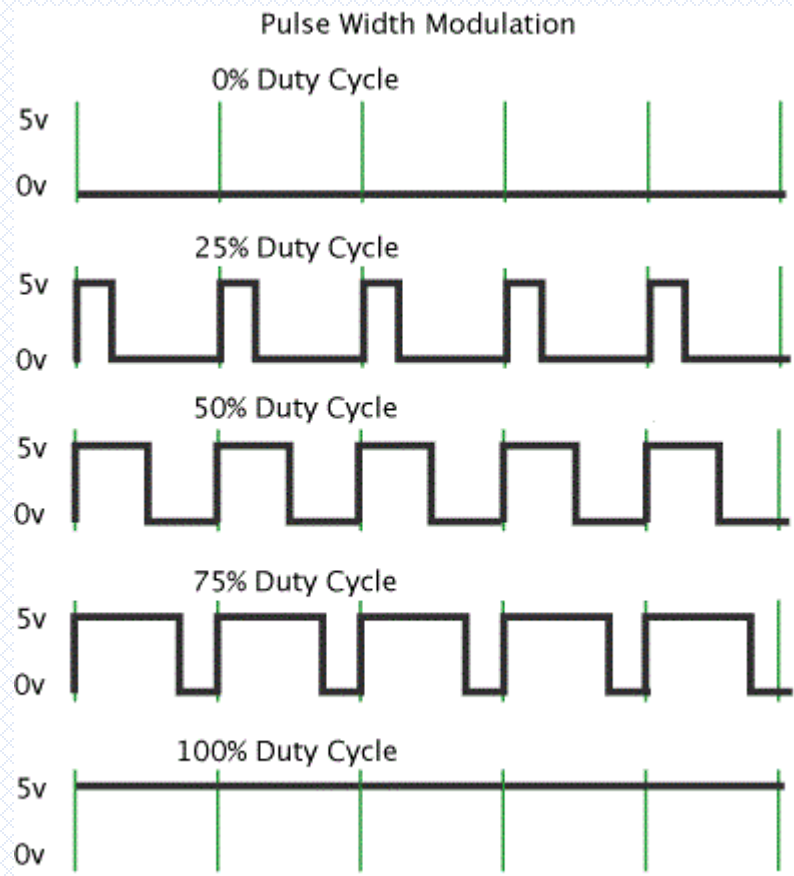
Output Compare

➢ **Output compare allows custom processing to be done when timer reaches a preset target value.**

➢ **Examples of custom processing**
- ❑ **clearing timer,**
- ❑ **changing values of dedicated pins,**
- ❑ **triggering an interrupt.**

➢ **Output compare can be used to**
- ❑ **generate signals of various shapes,**
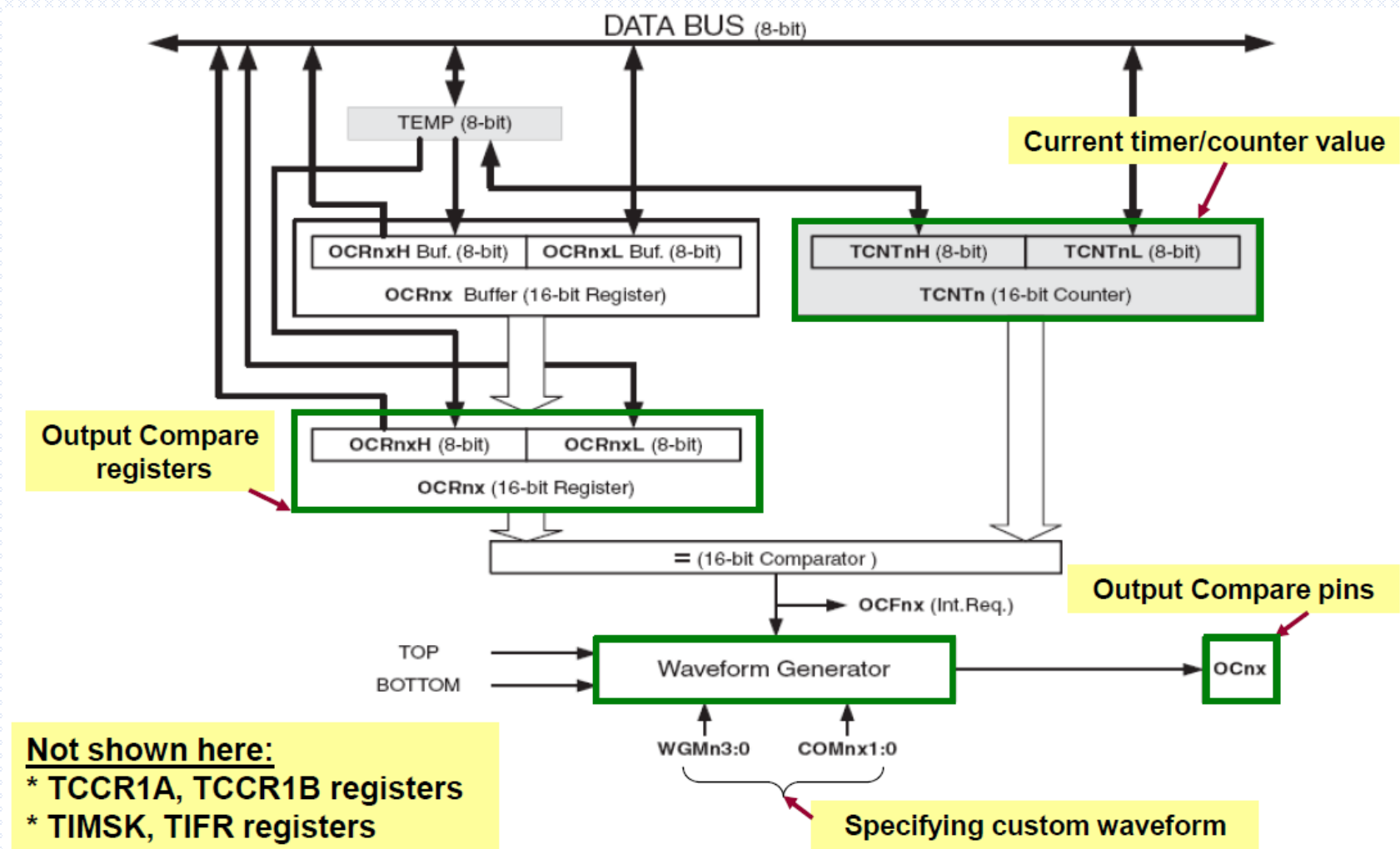- ❑ **perform actions such as ADC at specific time instants.**
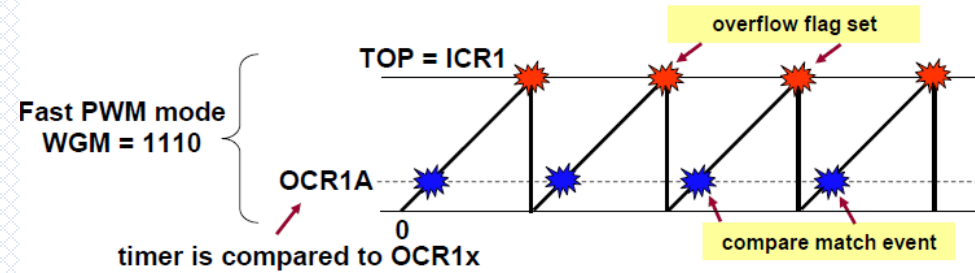
Output Compare: Common elements

➤ Output compare registers
  - ❑ to store the target timer values.

➤ Output compare pins
  - ❑ the values of these dedicated pins can be automatically changed (set, reset, toggled) when there is an output compare match.

➤ Configuration registers
  - ❑ to configure the operations of timer.

➤ Output compare interrupt
  - ❑ code for extra processing when there is an output compare match can be put in ISR.

Hossein Khosravi   Shahrood University of Technology
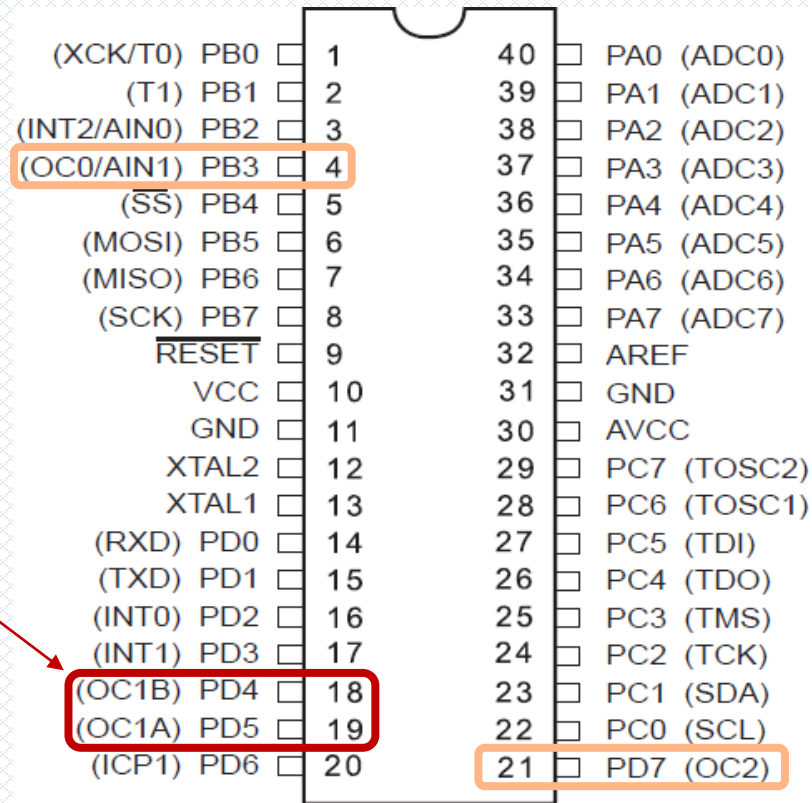
Output Compare Unit in Timer 1

➢ **Timer 1 has two output compare channels: A and B.**

➢ **Timer 1 is continuously compared to OCR1A or OCR1B or a fixed limit.**

➢ **When a match occurs, a flag OCF1x is set (x = A or B)**

➢ **When a match occurs, Timer 1 can**
  ❑ **trigger an output compare interrupt.**
  ❑ **change output compare pins OC1x.**

Output Compare Unit – Block diagram

DATA BUS (8-bit)

TEMP (8-bit)

Current timer/counter value

| OCRnxH Buf. (8-bit) | OCRnxL Buf. (8-bit) |
| TCNTnH (8-bit) | TCNTnL (8-bit) |

OCRnx Buffer (16-bit Register)

TCNTn (16-bit Counter)

Output Compare registers

| OCRnxH (8-bit) | OCRnxL (8-bit) |

OCRnx (16-bit Register)

= (16-bit Comparator)

OCFnx (Int.Req.)

Output Compare pins

TOP
BOTTOM

Waveform Generator

OCnx

WGMn3:0   COMnx1:0

Specifying custom waveform

Not shown here:
* TCCR1A, TCCR1B registers
* TIMSK, TIFR registers

7

Shahrood University of Technology

Output Compare Unit – Relevant pins

```
             (XCK/T0)  PB0 ▯  1        40 ▯ PA0  (ADC0)
                 (T1)  PB1 ▯  2        39 ▯ PA1  (ADC1)
          (INT2/AIN0)  PB2 ▯  3        38 ▯ PA2  (ADC2)
           (OC0/AIN1)  PB3 ▯  4        37 ▯ PA3  (ADC3)
                 (SS)  PB4 ▯  5        36 ▯ PA4  (ADC4)
               (MOSI)  PB5 ▯  6        35 ▯ PA5  (ADC5)
               (MISO)  PB6 ▯  7        34 ▯ PA6  (ADC6)
                (SCK)  PB7 ▯  8        33 ▯ PA7  (ADC7)
                     RESET ▯  9        32 ▯ AREF
                       VCC ▯ 10        31 ▯ GND
                       GND ▯ 11        30 ▯ AVCC
                     XTAL2 ▯ 12        29 ▯ PC7  (TOSC2)
                     XTAL1 ▯ 13        28 ▯ PC6  (TOSC1)
                (RXD)  PD0 ▯ 14        27 ▯ PC5  (TDI)
                (TXD)  PD1 ▯ 15        26 ▯ PC4  (TDO)
               (INT0)  PD2 ▯ 16        25 ▯ PC3  (TMS)
               (INT1)  PD3 ▯ 17        24 ▯ PC2  (TCK)
               (OC1B)  PD4 ▯ 18        23 ▯ PC1  (SDA)
               (OC1A)  PD5 ▯ 19        22 ▯ PC0  (SCL)
               (ICP1)  PD6 ▯ 20        21 ▯ PD7  (OC2)
```
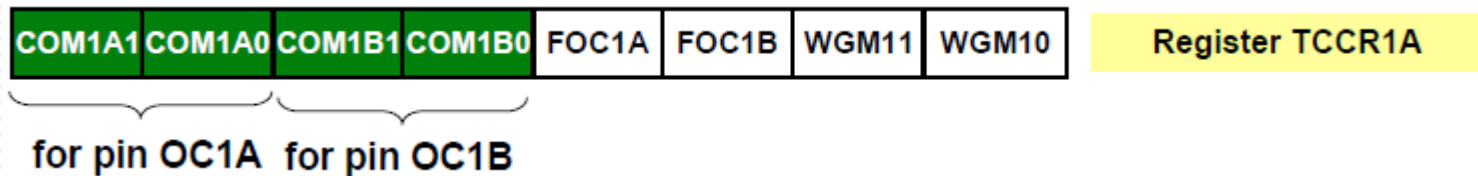
PORTD pins must
be enabled for output

Output Compare Unit – Main aspects

➢ **What changes can be made to output compare pins OC1x?**

➢ **What are the available operation modes of timer 1?**

➢ **Steps to produce a custom waveform?**

➢ **How to use output compare interrupt?**

Hossein Khosravi

Changing output compare pins OC1x

➢ When there's a timer event (compare match or timer reaching 0), pins OC1x can be automatically updated: toggle, set to 1, clear to 0, or no change.

➢ The type of update is specified by two bits in TCCR1A register:

❑ COM1x1 and COM1x2 where x = 'A' or 'B'.

COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	Register TCCR1A

for pin OC1A for pin OC1B

➢ The exact change will also depend on the operation mode of Timer 1.

Operations modes of Timer 1

➢ Timer 1 supports 15 operation modes, which can be divided into 5 groups:

❑ Normal

❑ Clear Timer on Compare Match (CTC)

❑ Fast PWM (single-slope operation)

❑ Phase correct PWM (dual-slope operation)

❑ Phase and Frequency Correct PWM (dual-slope operation)

➢ The operation mode is selected by 4 bits:

❑ WGM = {WGM13, WGM12, WGM11, WGM10}

➢ Each groups of operations will be discussed next.

Some Definitions

➢ **BOTTOM**

❑ The counter reaches the *BOTTOM* when it becomes **0x0000**.

➢ **MAX**

❑ The counter reaches its MAXimum when it becomes **0xFFFF** (decimal 65535).

➢ **TOP**

❑ The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be one of the fixed values:

☐ **0x00FF**, **0x01FF**, or **0x03FF**, or to the value stored in the **OCR1A** or **ICR1** Register.

❑ The assignment is dependent of the mode of operation.

Selecting operation mode of Timer 1

COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	Register TCCR1A

ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	Register TCCR1B

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	Reserved	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Normal mode

➢ Timer repeatedly counts from 0 to 0xFFFF.

➢ Overflow flag TOV1 is set after timer reaches 0xFFFF.

➢ No change is allowed on output compare pins OC1x

➢ Discussed before.

overflow flag

FFFF

timer value

0

Hossein Khosravi   Shahrood University of Technology

CTC modes

➢ Timer is reset to 0 when it reaches the value in OCR1A or ICR1.

Hossein Khosravi   Shahrood University of Technology
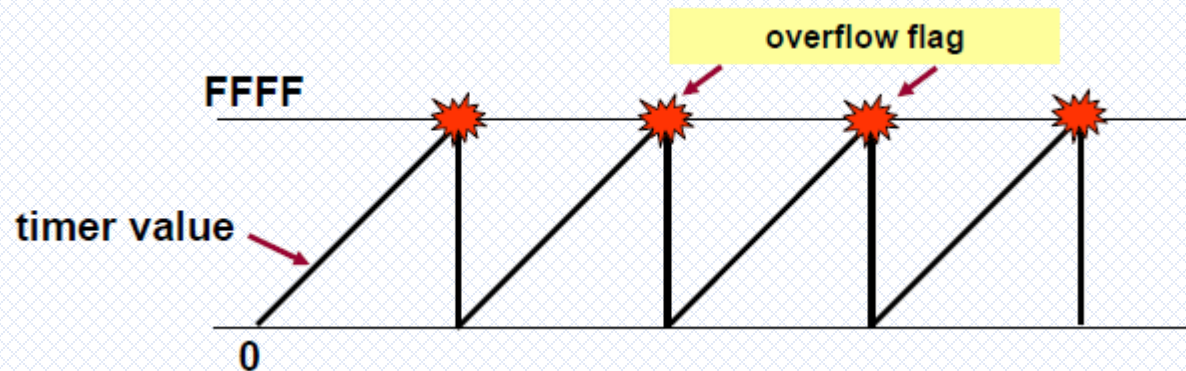
CTC modes

➢ When the OC1A or OC1B is connected to the pin, the function of the COM1x1:0 bits is dependent of the WGM13:0 bits setting.

➢ The following Table shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to a normal or a CTC mode (non-PWM).

Table 44. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on compare match
1	0	Clear OC1A/OC1B on compare match (Set output to low level)
1	1	Set OC1A/OC1B on compare match (Set output to high level)

Hossein Khosravi

Shahrood University of Technology

Fast PWM modes

➢ Timer goes from 0 to TOP, where TOP is equal to

❑ 0xFF (for 8-bit mode,   WGM = 0101)

❑ 0x1FF (for 9-bit mode, WGM = 0110)

❑ 0x3FF (for 10-bit mode WGM = 0111)

❑ value in ICR1 (for   WGM = 1110)

❑ value in OCR1A (for   WGM = 1111)

➢ Compare match occurs when timer = OCR1x register.

Hossein Khosravi   Shahrood University of Technology
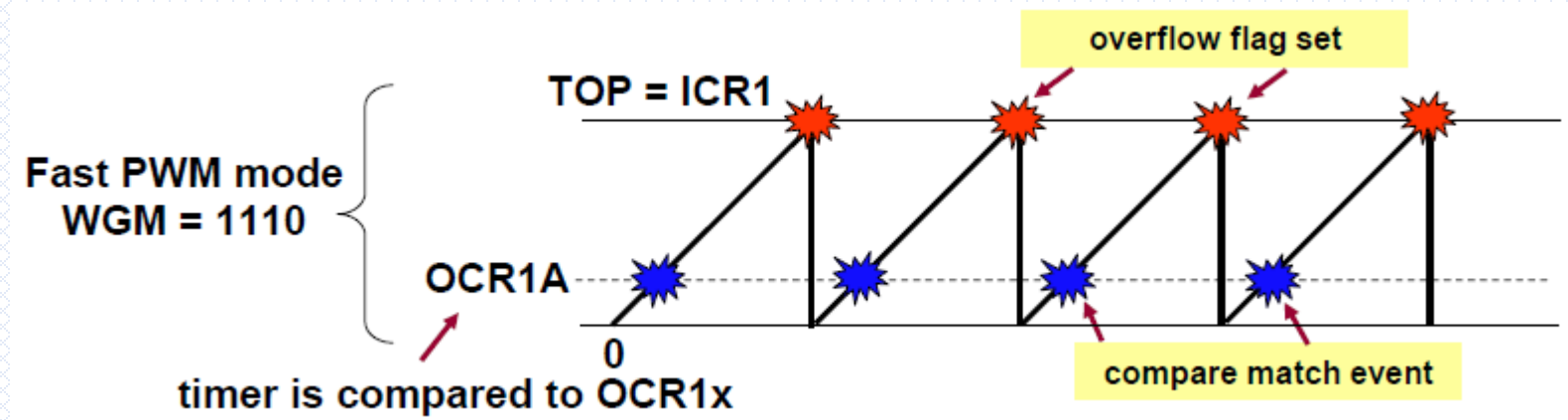
Compare Output Mode in Fast PWM

Table 45. Compare Output Mode, Fast PWM[1]

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM13:0 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on compare match, set OC1A/OC1B at BOTTOM, (non-inverting mode)
1	1	Set OC1A/OC1B on compare match, clear OC1A/OC1B at BOTTOM, (inverting mode)

We focus on this mode

Hossein Khosravi   Shahrood University of Technology

Fast PWM (COM1x1:0 = 10)

➢ Non-inverting mode

❑ Clear OC1A/OC1B on compare match, set OC1A/OC1B at BOTTOM,
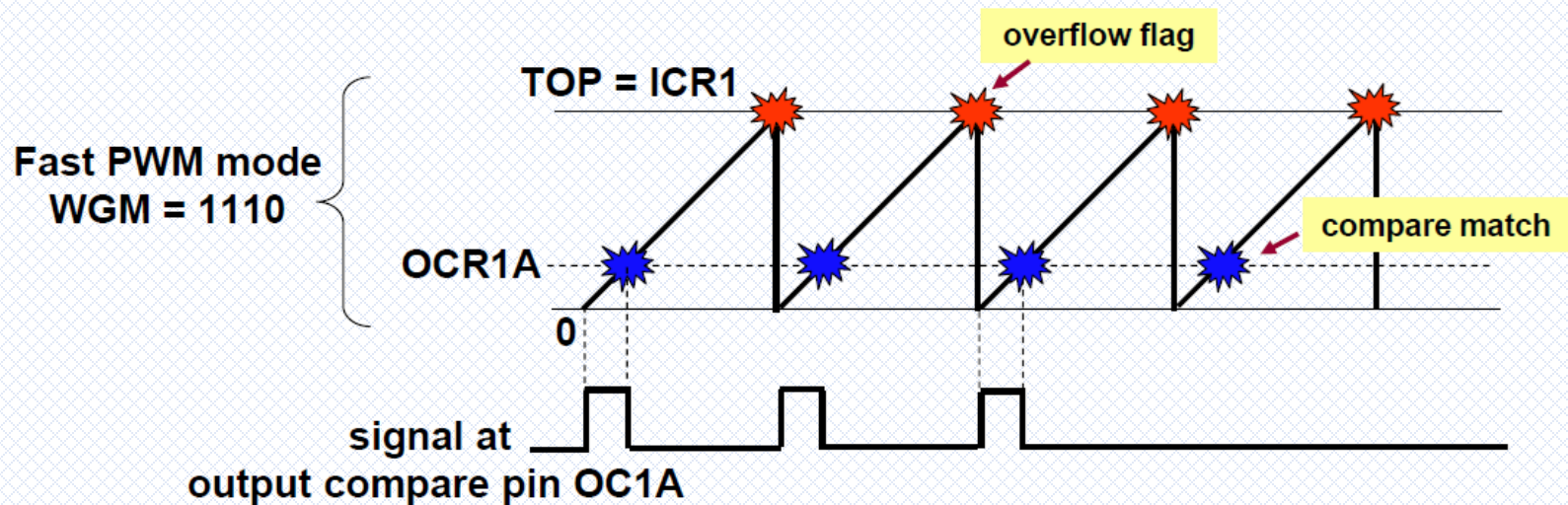
What will happen if we change ICR1 or OCR1A while counting?

Hossein Khosravi

Shahrood University of Technology

Fast PWM Timing Diagram

The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1.
The OC1x Interrupt Flag will be set when a compare match occurs.

Figure 32. Fast PWM Mode, Timing Diagram

Hossein Khosravi

Shahrood University of Technology
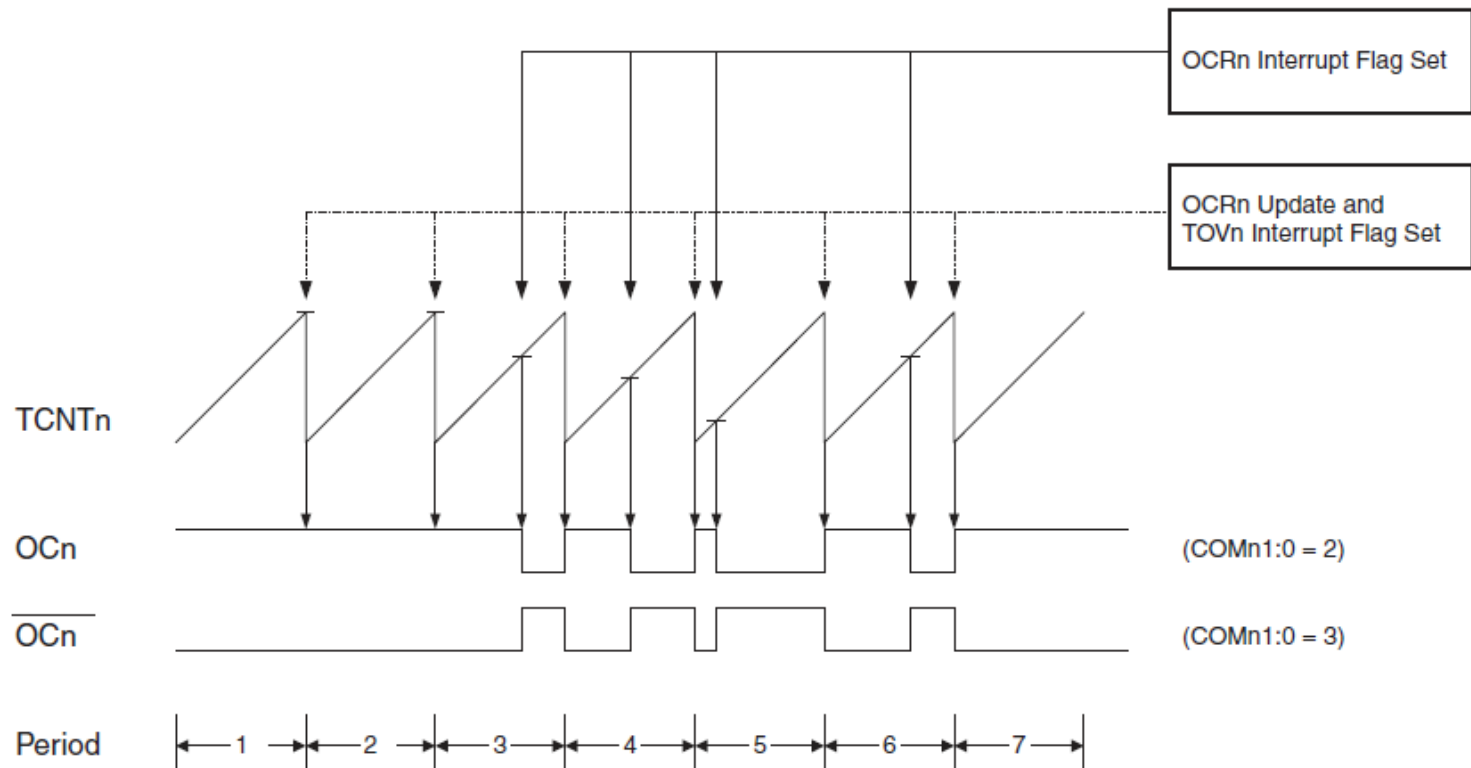
OCR1A buffer register

➢ The procedure for updating ICR1 differs from updating OCR1A when used for defining the TOP value.

❑ The **ICR1** Register is not double buffered
  ☐ It will be changed **immediately** while the counter is running.
  ☐ Counter may miss the compare match at the TOP value

❑ The **OCR1A** Register, is double buffered.
  ☐ This feature allows the OCR1A I/O location to be written anytime.
  ☐ The value written will be put into the OCR1A Buffer Register.
  ☐ The OCR1A will then be updated at the next TOP/BOTTOM.

Phase Correct PWM modes

➢ **Timer counts up and down between 0 and TOP, where TOP is**

❑ **0xFF (for 8-bit mode, WGM = 1000) or**

❑ **0x1FF (for 9-bit mode, WGM = 0010) or**

❑ **0x3FF (for 10-bit mode, WGM = 0011) or**

❑ **value in ICR1 (for WGM = 1010) or**

❑ **value in OCR1A (for WGM = 1011)**

➢ **Compare match occurs when timer = OCR1x register.**

Shahrood University of Technology
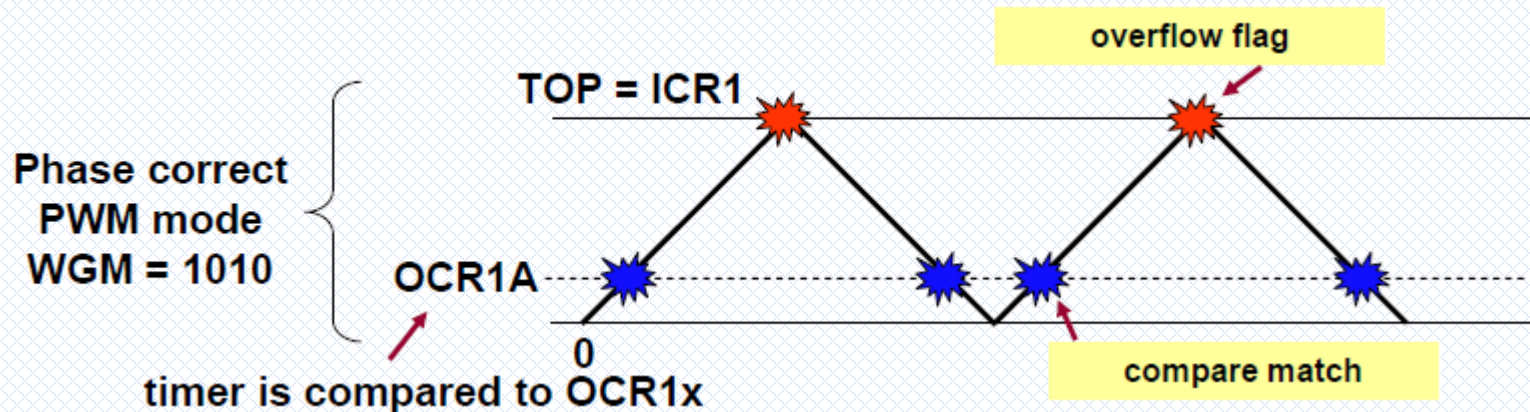
Compare Output Mode, P-Correct and P&F-Correct PWM

➢ **On compare match, change of pins OC1x is allowed.**

Table 46. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM [1]

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 9 or 14: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM13:0 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on compare match when up-counting. Set OC1A/OC1B on compare match when downcounting.
1	1	Set OC1A/OC1B on compare match when up-counting. Clear OC1A/OC1B on compare match when downcounting.

We focus on this mode

Hossein Khosravi   Shahrood University of Technology

Compare Output Mode, P-Correct and P&F-Correct PWM

➤ Clear OC1A/OC1B on compare match when up-counting.

➤ Set OC1A/OC1B on compare match when down-counting.
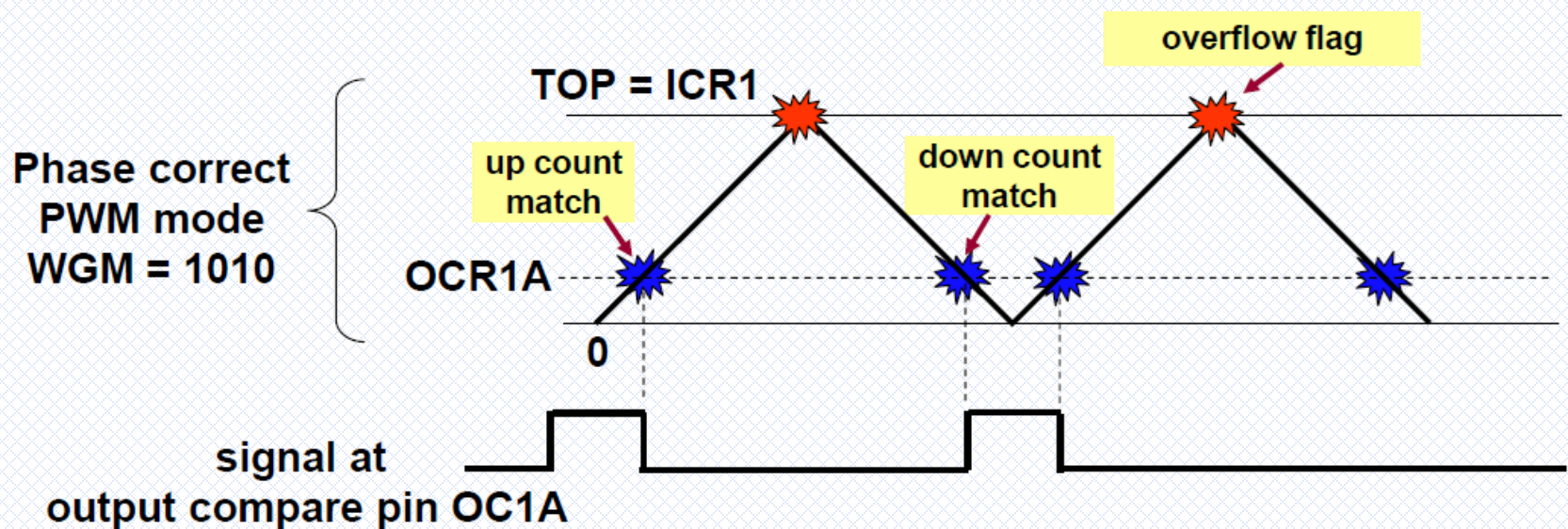
Phase and Frequency Correct PWM modes

➢ **Timer counts up and down between 0 and TOP, where TOP is**

❑ value in ICR1 (for WGM = 1000) or

❑ value in OCR1A (for WGM = 1001)

➢ Compare match occurs when timer = OCR1x register.

➢ On compare match, changing pins OC1x is allowed in the same way as in Phase Correct PWM modes

➢ The only difference w.r.t. Phase correct mode is in **Update of OCR1x.**

➢ It is recommended to use the P&F-Correct instead of the P-Correct mode when changing the TOP value while the Timer/Counter is running.

➢ When using a static OCR1A value there are practically no differences between the two modes of operation.

Hossein Khosravi

Shahrood University of Technology

Producing a custom waveform

➢ Steps to produce a custom waveform on an output compare pin OC1x

❑ Select the operation mode of Timer 1: CTC, fast PWM, or phase correct PWM, …

❑ Select how output compare pin will be updated on compare match event.

❑ Configure timer 1: clock source, prescaler, …

set registers TCCR1A and TCCR1B

❑ Put correct values in the output compare registers.

set register OCR1A or ICR1

Example 1: Producing a custom waveform

> **Use Timer 1 to create a signal with period = 1000µs, high time = 200µs.**

Hossein Khosravi   Shahrood University of Technology
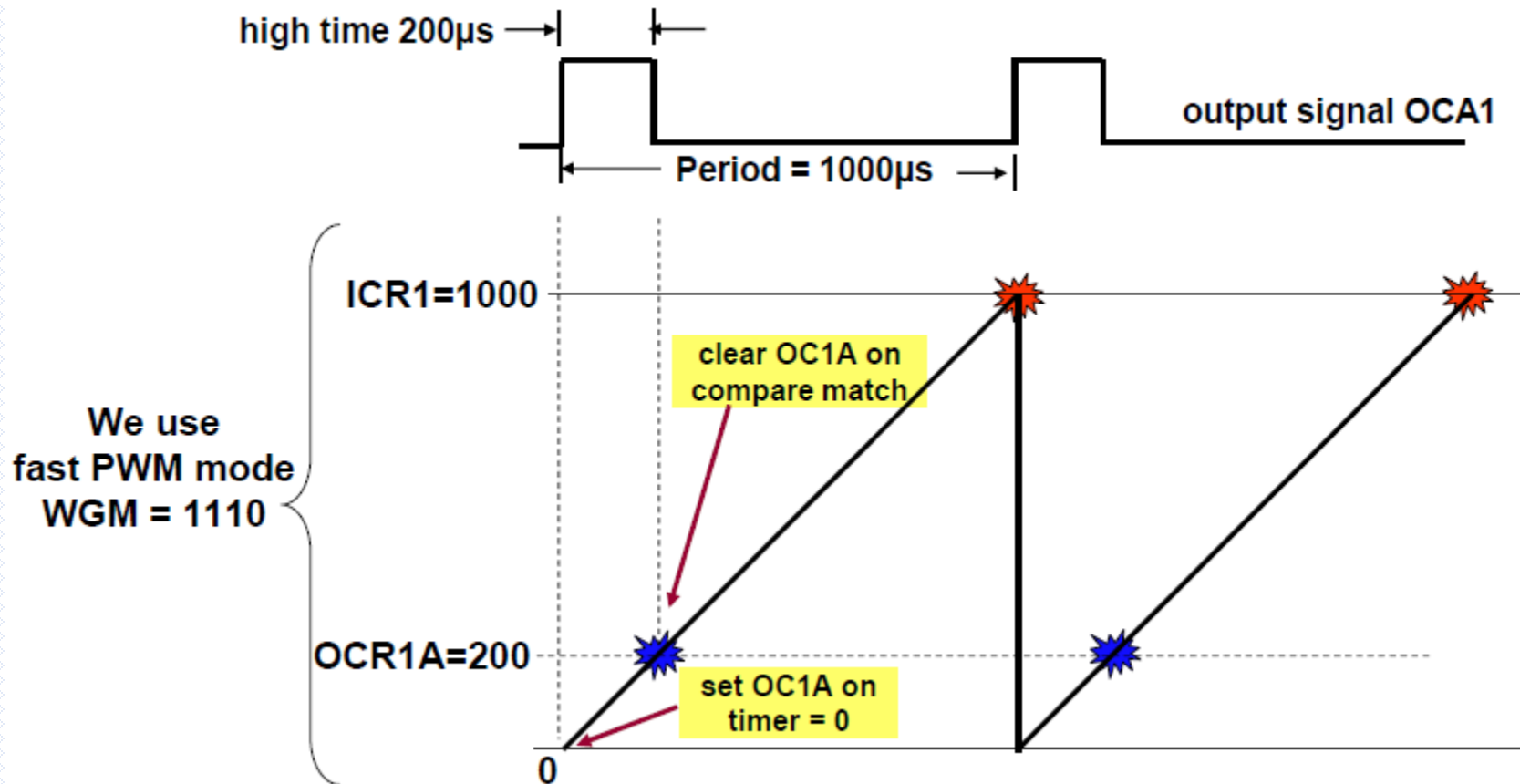
Example 1: Determining registers

COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	Register TCCR1A
1	0	0	0	0	0	1	0	

ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	Register TCCR1B
0	0	0	1	1	0	0	1	

- ICR1 = 1000 → period of output signal
- OCR1A = 200 → pulse width of output signal
- WGM3:0 = 1110 → Fast PWM mode where TOP = ICR1.
- CS12:0 = 001 → Internal clock, no prescaler
- COM1A1:0 = 10 → set OC1A when timer = 0

  clear OC1A when compare match

Example 1: Program make_pwm.c

```c
#include <avr\io.h>
int main(void) {
  DDRD=0b00100000; // set port D for output (D.5 is OC1A)
  // Set register TCCR1A
  // WGM11:WGM10 = 10: with WGM13-WGM12 to select timer mode 1110
  // Fast PWM, timer 1 runs from 0 to ICR1
  // COM1A1:COM1A0 = 10: clear OC1A when compare match, set OC1A
  when 0
  // compare match occurs when timer = OCR1A
  TCCR1A = 0b10000010;
  // Set register TCCR1B
  // WGM13:WGM12 = 11
  // CS12:CS0 = 001: internal clock 1MHz, no prescaler
  TCCR1B = 0b00011001;
  ICR1 = 1000; // period of output signal
  OCR1A = 200; // pulse width of output signal
  while(1){;}
}
```

Example 1: Testing

➢ Download program make_pwm.hex to ATmega32.

➢ Use oscilloscope to measure signal on pin OC1A (D.5).
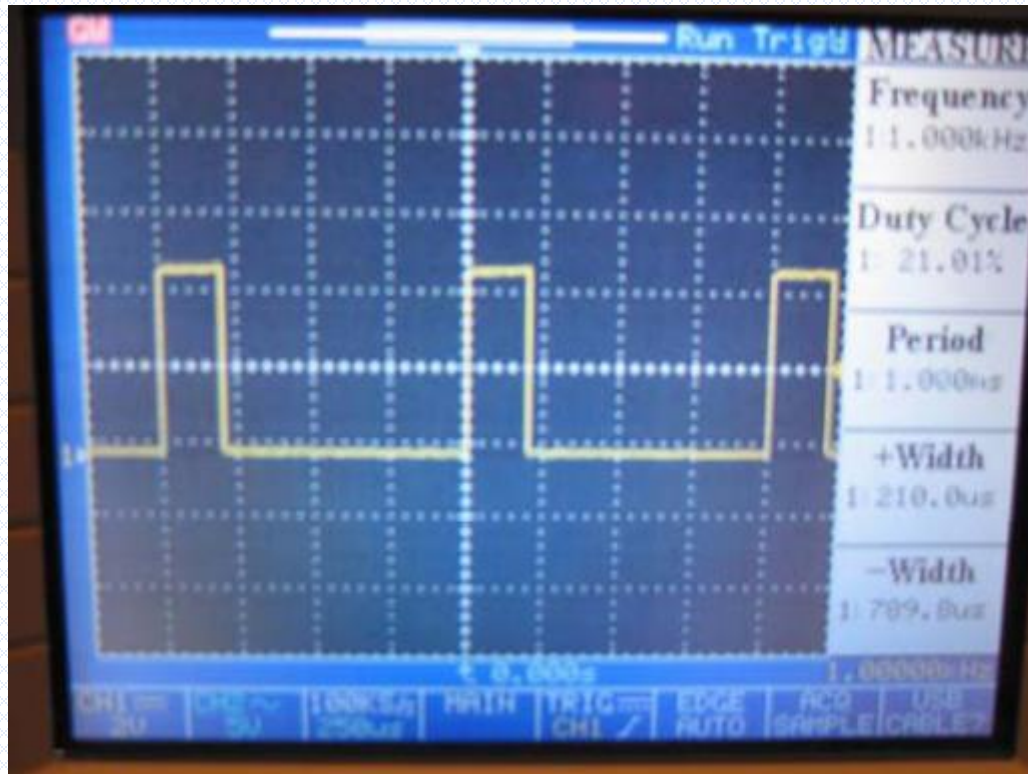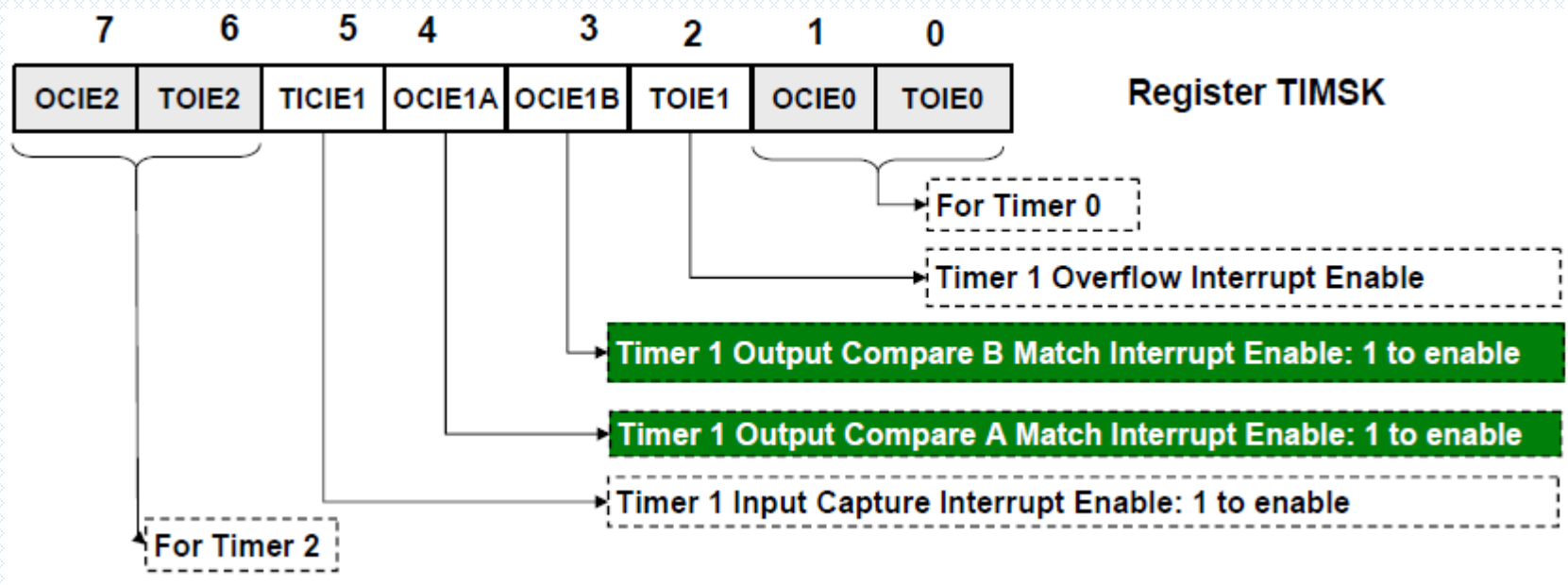
Output Compare Interrupt

➢ We have learnt to produce PWM signals on dedicated output compare pins OC1x.

➢ What if we need to perform custom operations at predefined time instants or produce signals on an arbitrary output pin?

➢ Possible approach:

❑ trigger an output compare interrupt at those time instants.

❑ write ISR that performs the custom operations

Hossein Khosravi   Shahrood University of Technology
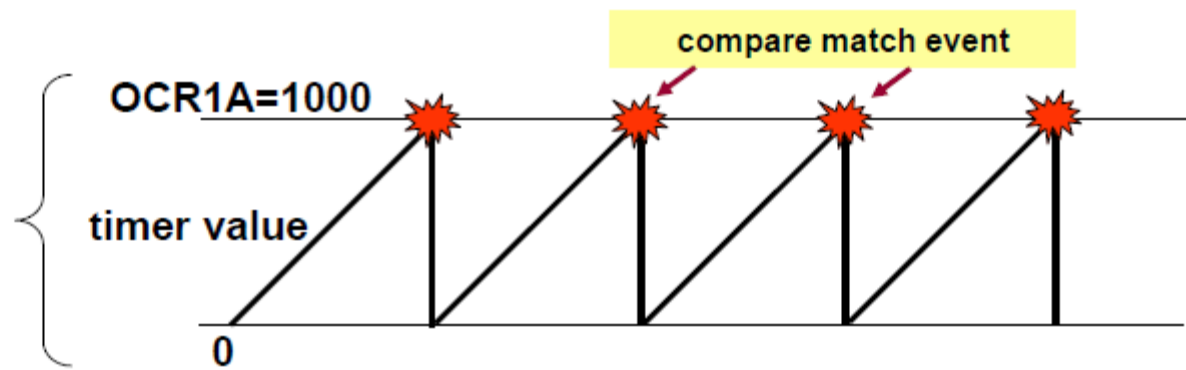
Output Compare Interrupt

➢ Output compare interrupt is enabled by OCIE1A and OCIE1B flag for channel A and B, respectively.

➢ C names for these interrupts: TIMER1_COMPA_vect and TIMER1_COMPB_vect.

Example 2: Output Compare Interrupt

➢ **Use Timer 1's output compare interrupt to toggle pin B.1 every 1000μs.**

Example 2: Program oc_int.c

```c
#include <avr\io.h>
#include <avr\interrupt.h>
ISR(TIMER1_COMPA_vect){
    PORTB = PORTB ^ 0b00000010; // toggle B.1
}
int main(void) {
  DDRB = 0xFF; // set port B for output
  PORTB = 0xFF; // initial value of port B
  // WGM11:WGM10 = 00: with WGM13-WGM12 to select timer mode 0100
  // CTC, timer 1 runs from 0 to OCR1A
  TCCR1A = 0b00000000;
  // WGM13:WGM12 = 01
  // CS12:CS0 = 001: internal clock 1MHz, no prescaler
  TCCR1B = 0b00001001;
  OCR1A = 1000; // interrupt will be triggered every 1000us
  TIMSK = (1<< OCIE1A); // enable Timer 1 Output Compare A
  interrupt
  sei(); // enable interrupt subsystem
  while(1){;}
}
```

Example application of PWM

➢ PWM signals are commonly used in embedded applications: motor control, sound alarm and radio transmission.

➢ A PWM signal is a periodic, rectangular pulse. The period and the duty cycle can vary.

➢ **Here, we'll generate a PWM signal to control a servo motor.**

high time T_{high}

low time T_{low}

period T
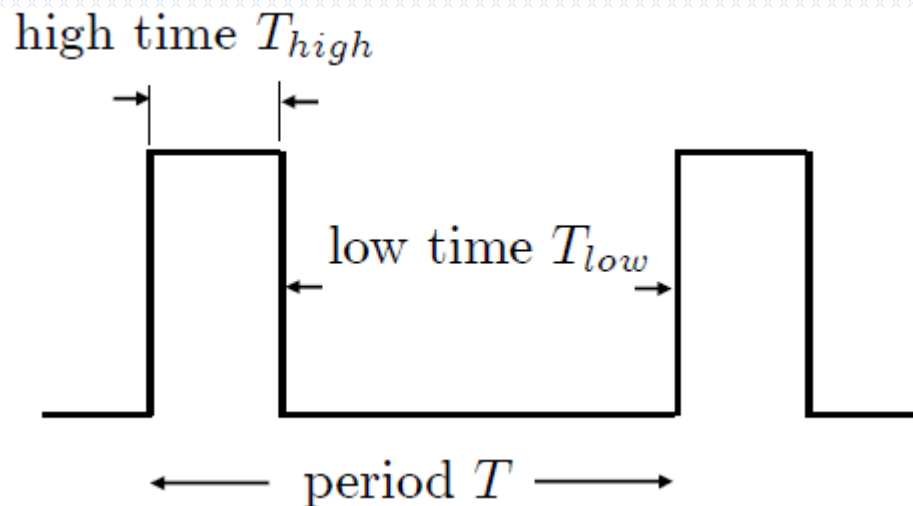
$$\text{mark space ratio} = \frac{T_{high}}{T_{low}}$$

$$\text{duty cycle} = \frac{T_{high}}{T}$$

Hossein Khosravi   Shahrood University of Technology

Controlling servo motor

➢ We use a servo motor Futaba S3003 (price 53,000T @ 99-01-29).

➢ It has three wires

❑ Black: ground

❑ Red: 4.8V to 6V DC supply

❑ White: PWM signal

➢ The frequency of the PWM signal is 50Hz.

➢ This motor have a rotation range of 180º.

➢ To keep the motor at a given angle, we must send a PWM signal of a specific duty cycle.

➢ Range of duty cycle: 1% to 12%.

Hossein Khosravi   Shahrood University of Technology

Controlling servo motor

- ➢ **Write C program that lets the user press switches SW6 and SW7 on STK500 board to rotate the motor left and right, respectively.**

- ➢ **The switches can be connected to pins of port A.**

- ➢ **Depending on which switch is pressed, we increment or decrement the duty cycle.**

- ➢ **We then produce a PWM signal with a period of 20000μs and the given duty cycle.**

Hossein Khosravi

Shahrood University of Technology

Example 3: Controlling servo motor (motor_control.c)

```c
#include <avr\io.h>
int main(void) {
  unsigned int duty_cycle;
  unsigned char button;
  DDRA = 0b00; DDRB = 0xFF; // set port A for input, port B for output
  DDRD = 0b00100000; // set pin D.5 for output (OC1A)
  // WGM11:WGM10 = 10: with WGM13-WGM12 to select timer mode 1110
  // Fast PWM, timer 1 runs from 0 to ICR1
  // COM1A1:COM1A0 = 10: clear OC1A when compare match, set OC1A when 0
  TCCR1A = 0b10000010; // compare match occurs timer = OCR1A
  TCCR1B = 0b00011001; // WGM13:WGM12=11; CS12:CS0=001: internal clock 1MHz, no prescaler
  ICR1 = 20000; // PWM frequency = 50Hz, period = 20000us
  duty_cycle = 6; // initial duty cycle
  OCR1A = duty_cycle * 200; //(20000 * duty_cycle /100) calculate high time
  while (1){
    if (button == PINA) // ignore repeated press
      continue;
    button = PINA; PORTB = button; // store button press, display on port B
    if ((button & 0b11000000) == 0b11000000)
      continue;
    if ((button & 0b10000000) != 0) // Increment duty cycle if switch SW7 is pressed
      duty_cycle = (duty_cycle < 12) ? duty_cycle+1 : duty_cycle;
    if ((button & 0b01000000) != 0) // Increment duty cycle if switch SW6 is pressed
      duty_cycle = (duty_cycle > 1) ? duty_cycle-1 : duty_cycle;
    OCR1A = duty_cycle * 200;// calculate high time
  }
}
```

Controlling servo motor: Testing

ECTE333 Digital Hardware 2

Motor control with PWM signal
[Timer 1 Output Compare]

Lam Phung, 2008.

Hossein Khosravi   Shahrood University of Technology