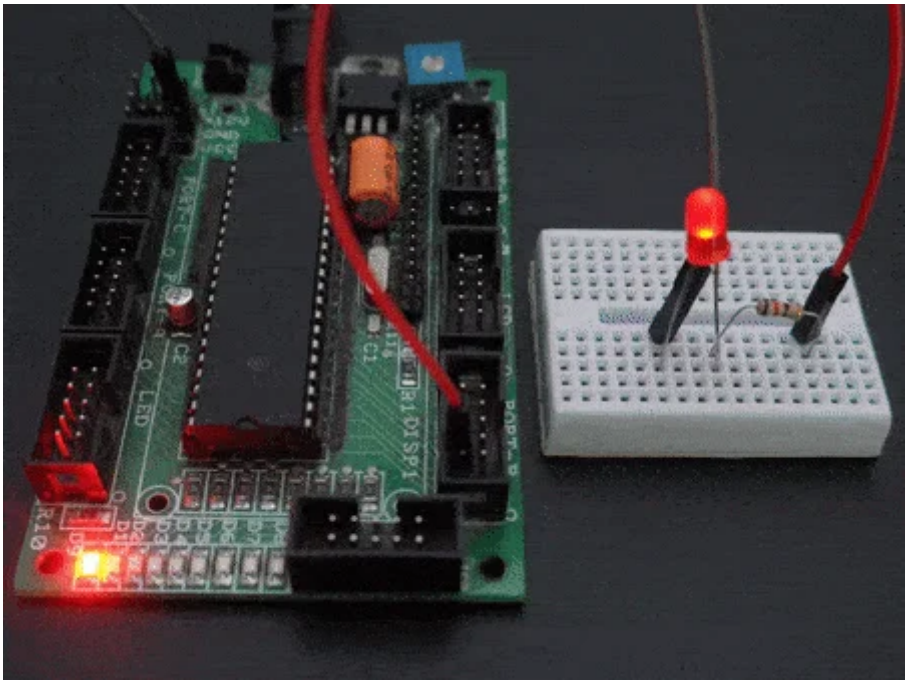
PWM in AVR ATmega16/ATmega32



Introduction to PWM

Pulse Width Modulation (PWM) is a technique by which the width of a pulse is varied while keeping the frequency constant.

Why do we need to do this? Let's take an example of controlling DC motor speed, more the Pulse width more the speed. Also, there are applications like controlling light intensity by PWM.

A period of a pulse consists of an **ON** cycle (5V) and an **OFF** cycle (0V). The fraction for which the signal is ON over a period is known as the **duty cycle**.

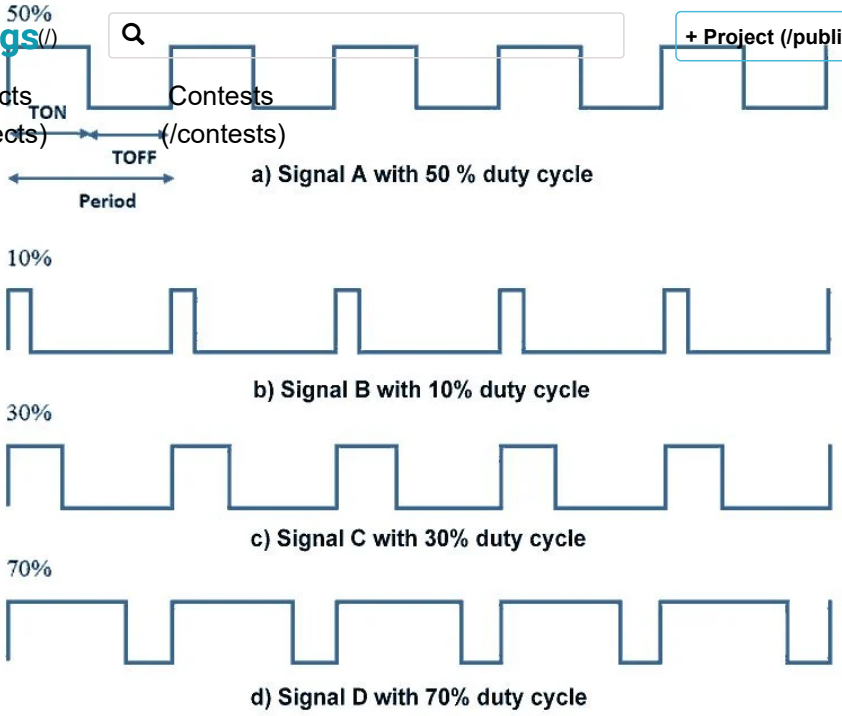
Duty Cycle (In %) = $\frac{T_{ON}}{TotalPeriod} * 100$

E.g. Consider a pulse with a period of 10ms which remains ON (high) for 2ms.The duty cycle of this pulse will be

D = 2ms / 10ms = 20%

Through the PWM technique, we can control the power delivered to the load by using the ON-OFF signal.

Pulse Width Modulated signals with different duty cycle are shown below



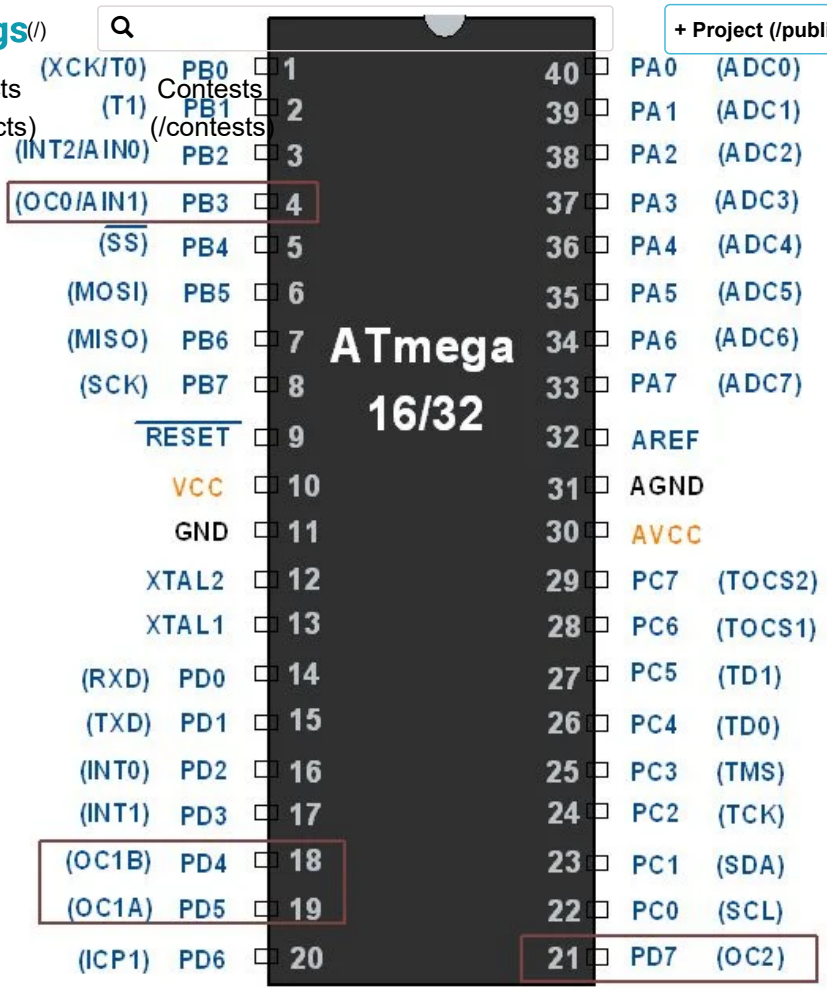
PWM Duty Cycle Waveforms

AVR ATmega PWM

ATmega has an inbuilt PWM unit. As we know, ATmega has 3 Timers T0, T1, and T2 which can be used for PWM generation. Mainly there are two modes in PWM.

1. Fast PWM
2. Phase correct PWM

We need to configure the Timer Register for generating PWM. PWM output will be generated on the corresponding Timer's output compare pin (OCx).



AVR ATmega16/32 PWM Pins

Configuring Timer0 for PWM generation

It is simple to configure PWM mode in Timer. We just need to set some bits in the TCCR0 register.

TCCR0: Timer Counter Control Register 0

7	6	5	4	3	2	1	0
FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00

Bit 7- FOC0: Force compare match

Write only bit, which can be used while generating a wave. Writing 1 to this bit will force the wave generator to act as if a compare match has occurred.

Bit 6, 3 - WGM00, WGM01: Waveform Generation Mode

WGM00	WGM01	Timer0 mode selection bit
0	0	Normal
0	1	CTC (Clear timer on Compare Match)
1	0	PWM, Phase correct
1	1	Fast PWM

Bit 5:4 - COM01:00:

1. When WGM00: WGM01= 11 i.e. **Fast PWM.** Compare Output Mode waveform generator on OC0 pin

COM01	COM00	Mode Name	Description
0	0	Disconnected	The normal port operation, OC0 disconnected
0	1	Reserved	Reserved
1	**0**	**Non-inverted**	Clear OC0 on compare match, set OC0 at TOP
1	**1**	**Inverted PWM**	Set OC0 on compare match, clear OC0 at TOP

2. When WGM00: WGM01= 10 i.e. **Phase correct PWM.** Compare Output Mode waveform generator on OC0 pin

COM01	COM00	Description
0	0	The normal port operation, OC0 disconnected
0	1	Reserved
1	**0**	Clear OC0 on compare match when up-counting, set OC0 on compare match when down-counting
1	**1**	Set OC0 on compare match when up-counting, Clear OC0 on compare match when down-counting

Bit 2:0 - CS02:CS00: Clock Source Select

These bits are used to select a clock source. When CS02: CS00 = 000, then timer is stopped. As it gets a value between 001 to 101, it gets a clock source and starts as the timer.

CS02	CS01	CS00	Description
0	**0**	**0**	No clock source (Timer / Counter stopped)
0	**0**	**1**	clk (no pre-scaling)
0	**1**	**0**	clk / 8
0	**1**	**1**	clk / 64
1	**0**	**0**	clk / 256
1	**0**	**1**	clk / 1024
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.

Fast PWM mode

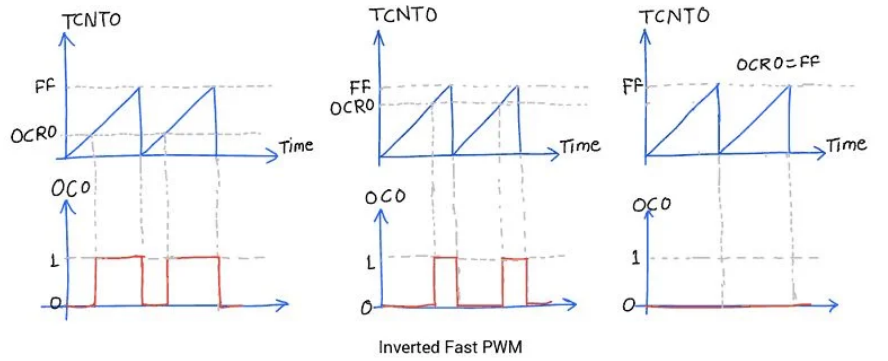
To set Fast PWM mode, we have to set WGM00: 01= 11. To generate a PWM waveform on the OC0 pin, we need to set COM01:00= 10 or 11.

COM01:00= 10 will generate Noninverting PWM output waveform and COM01:00= 11 will generate Inverting PWM output waveform. See fig.

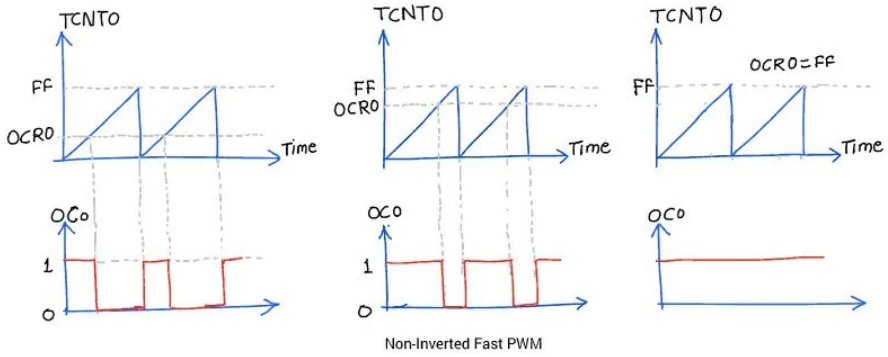
```
void PWM_init()
{
        /*set fast PWM mode with non-inverted output*/
        TCCR0 = (1<<WGM00) | (1<<WGM01) | (1<<COM01) | (1<<CS00);
        DDRB|=(1<<PB3);   /*set OC0 pin as output*/
}
```

Setting Duty cycle: we have to load value in the OCR0 register to set the duty cycle.

255 value for 100% duty cycle and 0 for 0% duty cycle. Accordingly, if we load value 127 in OCR0, the Duty cycle will be 50%.



Inverted Fast PWM

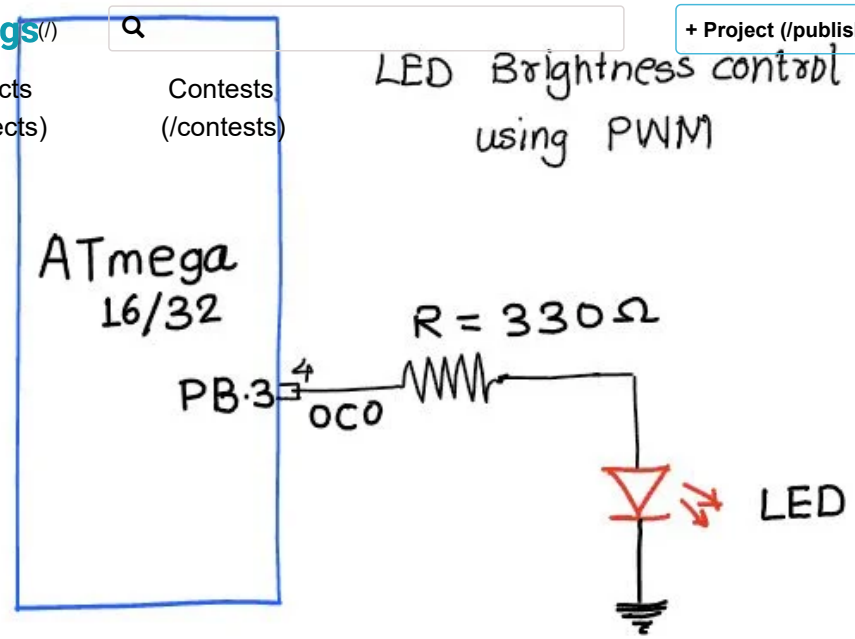


Non-Inverted Fast PWM

The advantage of using PWM mode in AVR is that it is an inbuilt hardware unit for waveform generation and once we set the PWM mode and duty cycle, this unit starts generating PWM and the controller can do other work.

Example

Control LED brightness using Fast PWM.



PWM LED Interfacing With ATmega16/32

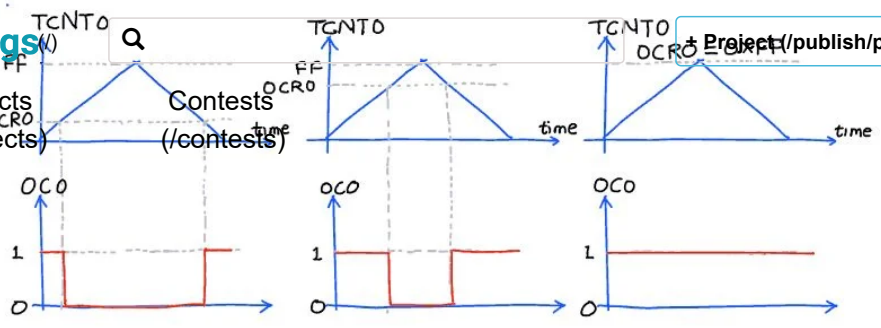
```
/*
  AVR ATmega16 PWM to control LED brightness
  http://www.electronicwings.com
*/

#define F_CPU 8000000UL
#include "avr/io.h"
#include <util/delay.h>


void PWM_init()
{
        /*set fast PWM mode with non-inverted output*/
        TCCR0 = (1<<WGM00) | (1<<WGM01) | (1<<COM01) | (1<<CS00);
        DDRB|=(1<<PB3);   /*set OC0 pin as output*/
}


int main ()
{
        unsigned char duty;
```

Phase correct PWM mode

To set Phase correct PWM, we just have to set the TCCRO register as follows.

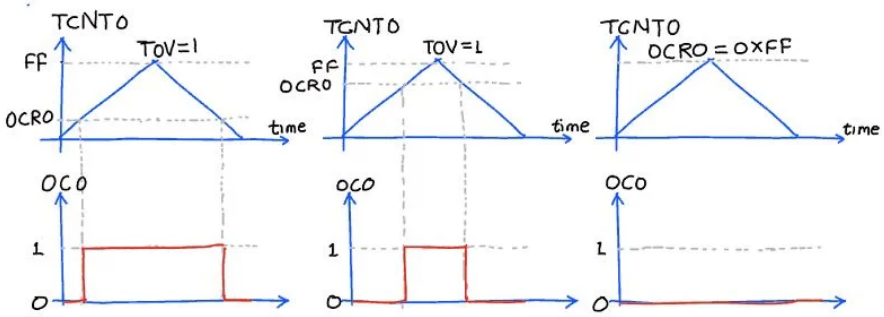
We can set the output waveform as inverted or non-inverted. See fig.

```
TCCR0 = (1<<WGM00) | (1<<COM01) | (1<<CS00);
```



Non-Inverted Phase correct PWM

Non-Inverted Phase Correct PWM



Inverted Phase correct PWM

Inverted Phase Correct PWM

Similarly, we can set PWM output on the other three OCx pins using Timer1 and Timer2.

PWM output is somehow close to the Analog output. We can use it as analog output for generating sine wave, audio signals, etc. it is also referred to as DDS.

Video

MOUSER ELECTRONICS (https://www.mouser.in?utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0)

Powered By

Components Used

ATmega 16 ATmega 16	X 1	(https://www.mouser.in/ProductDetail/Microchip-Technology-Atmel/ATMEGA16L-8PU?qs=%2Fha2pyFaduiGCJtTvs2wv8fVZbVAalLu7Iq%2FglTS0tALAx6fMenLvg%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0) Datasheet (/components/atmega-16/1/datasheet)
Atmega32 Atmega32	X 1	(https://www.mouser.in/ProductDetail/Microchip-Technology-Atmel/ATMEGA32-16PU?qs=aqrrBurbvGdpkmgj7RWmsQ%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0) Datasheet (/components/atmega32/1/datasheet)

Powered By (https://www.mouser.in?utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0)

Components Used

LED 5mm LED 5mm	X 1	(https://www.mouser.in/ProductDetail/Lite-On/LTL-307EE?qs=Yz4wJs0d%252BpgyXm%2FpkMp2pg%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0) Datasheet (/components/led-5mm/1/datasheet)
Breadboard Breadboard	X 1	(https://www.mouser.com/ProductDetail/Bus-Board-Prototype-Systems/BB830?qs=VEfmQw3KOauhPeTwYxNCaA%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0) Datasheet (/components/breadboard/1/datasheet)

Downloads

Proteus Simulation file ATmega16 PWM	Download (/api/download/platform-attachment/126)
ATmega16_Fast_PWM_Project_File	Download (/api/download/platform-attachment/307)
ATmega16_PhaseCorrect_PWM_Project_File	Download (/api/download/platform-attachment/308)

Comments

Comment

Muhammad
(/users/Muhammad/profile)
2017-12-30 03:24:49

sir please write a c code for speed synchronization of two motors please its really urgent.i need it today

Reply Like

Ramen
(/users/Ramen/profile)
2018-02-27 04:23:58

Sir, the article is extremely helpful for beginners like me. In the codes above, some bits which need not be set, like TCCR0 = (1<<WGM00) | (1<<WGM01) | (1<<COM01) | (1<<CS00); to set in Fast PWM, non-inverting mode, COM00 has to be 0, but we have not mentioned in code. Is it possible that it might remain or become 1, where it becomes Inverted?

Also, kindly tell which is the CS0 pin, and why we had not mentioned CS02, CS01.

Thanks,
Ramen

Reply Like

authorized
(/users/authorized/profile)
2018-02-27 04:59:23

hello ramen,

there is no need to mention COM00 as 0 in code when we mentions other bits that need to set 1 since we are assigning it directly to timer control register.

and CS00, CS01, CS02 are bits in TCCR register (not pins) as mentioned in above doc.

Reply Like

Ramen
(/users/Ramen/profile)
2018-02-27 05:03:57

Thanks a lot!!
Reply Like

sekay003
(/users/sekay003/profile)
2019-03-03 04:18:52

Merci beaucoup pour votre travail
Reply Like

sekay003
(/users/sekay003/profile)
2019-03-05 04:51:46

Can I have the code for atmega 320P please mister
Reply Like

amrmagdi50
(/users/amrmagdi50/profile)
2019-10-01 00:25:49

I need someone to explain this to me for an example if want to create a wave of frequency 1khz with duty 50% using a 4mhz clock how can it be done i do understand the duty part , but how to produce a 1 khz using the 4mhz using pwm mode .can it be done without using the top value function for timer 1 ??
Reply Like

godwithheart
(/users/godwithheart/profile)
2020-05-22 19:06:38

Sir I'm having confusion of this for Atmega 328P as it have way more PWM pins than 16 and 32?
Reply Like

godwithheart
(/users/godwithheart/profile)
2020-05-26 14:58:15

What's use of difference between fast PWM and phase correct
Reply Like

UzefSheikh
(/users/UzefSheikh/profile)
2020-10-10 19:51:43

what is mean by reserved??
Reply Like

poyhabib
(/users/poyhabib/profile)
2022-01-12 17:48:30

Thanx Allot for your help
I need if u plz
Pwm Drawer using GLCD on atmega 32
How to implement and how to output it
Reply Like

Truth19
(/users/Truth19/profile)
2022-03-26 05:37:46

Pls I need help trying to register 1 on arduino to make a 90 degree phase shift between the two wave
Reply Like

Truth19
(/users/Truth19/profile)
2022-03-26 05:42:30

```
void setup(){
DDRB = (1<<PB1)|(1<<PB2);// pin 9 and 10 output
TCCR1A = 0;
TCCR1B = 1;
// Clear OC1A and OC1B on Compare Match / Set OC1A and OC1B at Bottom;
// Wave Form Generator: Fast PWM 14, Top = ICR1
TCCR1A = (1<<COM1A1) + (1<<COM1A0) + (1<<WGM11);
TCCR1B = (1<<WGM13) + (1<<WGM12) + (1<<CS10); // prescaler = 1 (none)
OCR1A = 89;
OCR1B = 89;

}
void loop() {
// do something else
analogWrite(9,OCR1A);

analogWrite(10,OCR1B);
}
```

Reply Like

JayachitraJ
(/users/JayachitraJ/profile)
2023-04-03 19:37:48

How can I set the ocr1a value in fast pwm

Reply Like