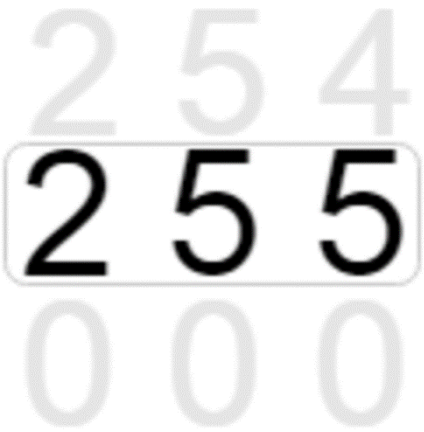
Timers (Χρονιστές) στον ATMega328P

Τσαλμπούρης Γεώργιος ΠΕ84
gtsalmpouris@gmail.com
tsal81@yahoo.com

Μάρτιος 2019

Timer (Ορισμός)

- Οι χρονιστές (timers) είναι ενσωματωμένα κυκλώματα του μΕ, τα οποία μπορούν να χρησιμοποιηθούν για τη δημιουργία κυματομορφών ή την καταμέτρηση χρονικών γεγονότων. Μπορούν επίσης να χρησιμοποιηθούν για την απαρίθμηση παλμών.
- Η υλοποίηση γίνεται μέσω hardware χωρίς να απασχολείται η CPU.



Timers

Ο ATMEGA328 διαθέτει τρεις timers :

- Timer0, Timer2 (8bit- Μέτρηση 0 έως 255)
- Timer1 (16bit- Μέτρηση 0 έως 65535)

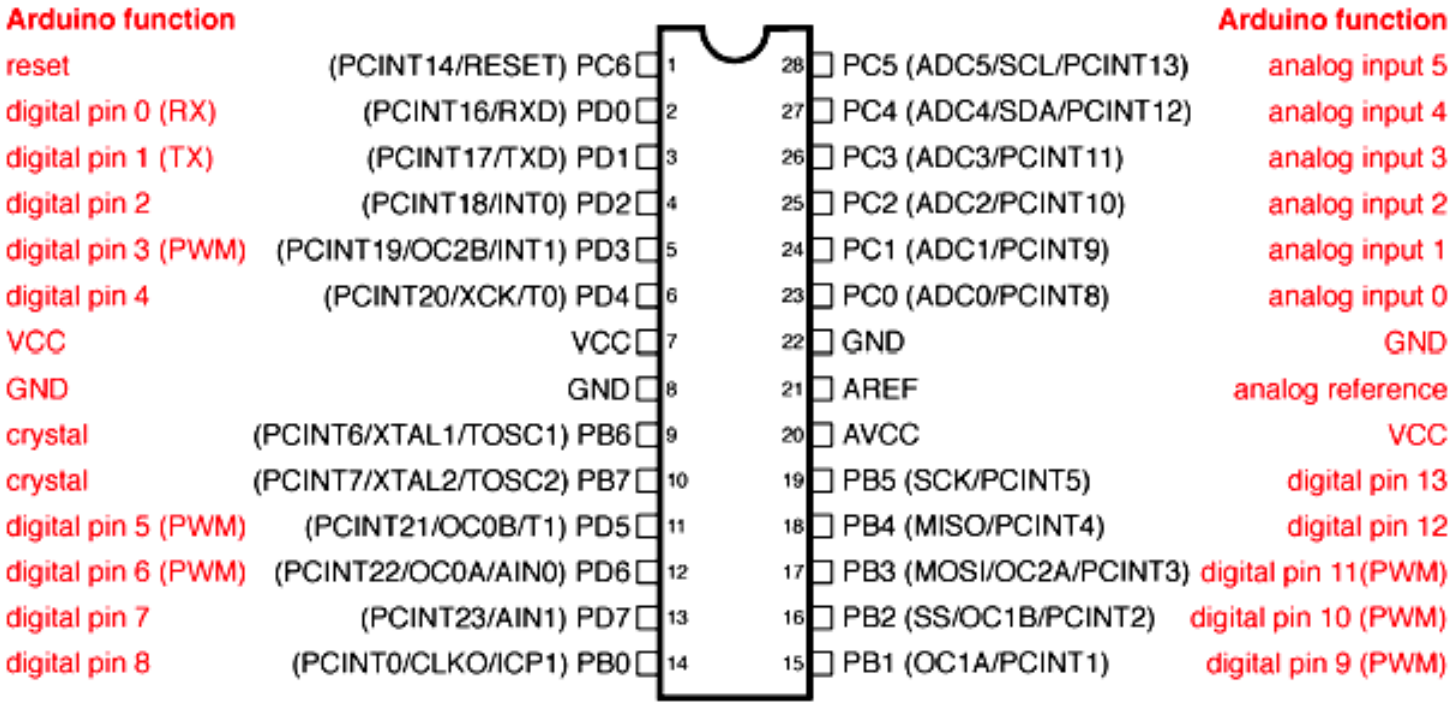
- Μπορούν να λειτουργήσουν ως χρονιστές ή ως απαριθμητές παλμών.
- Ο έλεγχος τους πραγματοποιείται μέσω συγκεκριμένων καταχωρητών

Modes λειτουργίας

1) Κανονική Λειτουργία (Normal Mode)
2) Μηδενισμού Ταύτισης Συγκριτή (CTC-Clear Timer On Compare)
3) PWM ορθής φάσης (Phase Correct PWM)
4) PWM γρήγορης λειτουργίας (Fast PWM)

- Και οι τρεις timers λειτουργούν με πανομοιότυπο τρόπο

ATMega328P and Arduino Uno Pin Mapping



Digital Pins 11,12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

Timer0

Ο έλεγχος γίνεται μέσω τριών καταχωρητών (κατά κύριο λόγο).

- TCNT0: παρέχει πρόσβαση στην τρέχουσα τιμή του μετρητή.
- TCCR0A-B: μέσω αυτών πραγματοποιούνται οι βασικές ρυθμίσεις
- OCR0A ή B : εδώ περιέχεται ο αριθμός με τον οποίο γίνεται συνεχώς η σύγκριση του μετρητή TCNT0 (CTC και PWM modes).

Normal Mode

- Ο μετρητής του Timer0 μετράει από 0-255 (στον καταχωρητή TCNT0) αλλάζοντας τιμή με ρυθμό ανάλογο με την επιλογή ρολογιού σύμφωνα με τον πίνακα

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	

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	

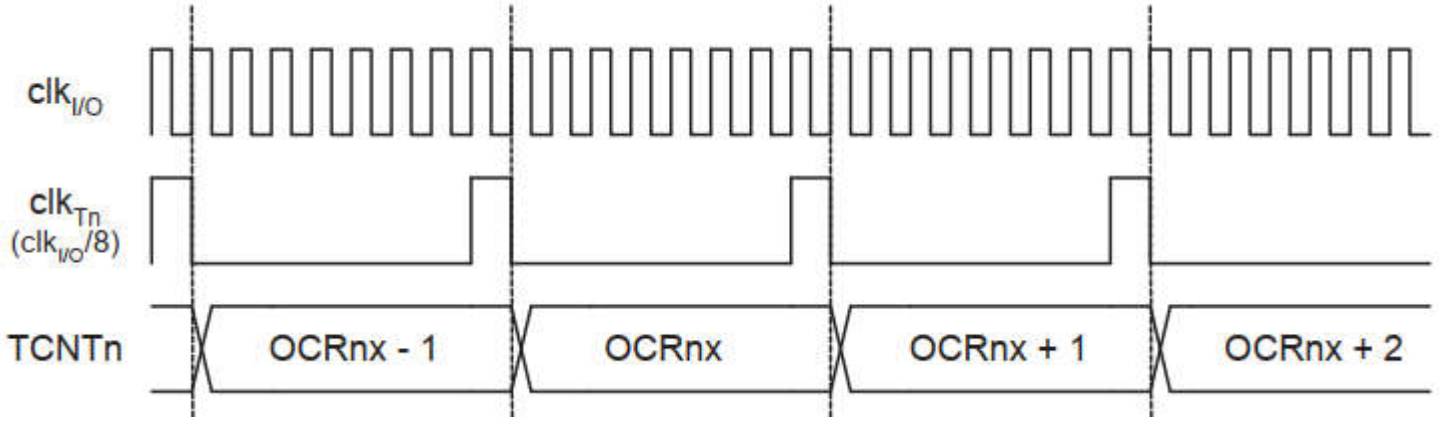
Waveform Generation Mode Bit Description

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

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)

Κανονική Λειτουργία (Normal Mode)



- Πχ αν επιλεγεί ρολόι CLK/8

f=16MHz / 8 =2MHz

T=1/f =0.5μsec Κάθε 0.5μsec ο TCNT0 θα αυξάνει κατά 1

Τολ=256*0.5μ=128μsec απαιτούνται για υπερχείλιση του TCNT0

Βοηθητικός Πίνακας

Pre-scaler	Count	Frequency Hz	Period nS	Period uS	Period mS	Period (Sec)	
1	1	16,000,000	62.5	0.0625	0.0000625	0.0000000625	
8	1	2,000,000	500	0.5	0.0005	0.0000005	
32	1	500,000	2,000	2	0.002	0.000002	**Timer 2**
64	1	250,000	4,000	4	0.004	0.000004	
128	1	125,000	8,000	8	0.008	0.000008	**Timer 2**
256	1	62,500	16,000	16	0.016	0.000016	
1024	1	15,625	64,000	64	0.064	0.000064	

Normal Mode

- Για να μπορέσουμε να αξιοποιήσουμε την υπερχείλιση του TCNT0, πρέπει να ενεργοποιήσουμε το INT υπερχείλισης του TIMER0 μέσω του TIMSK0.

TIMSK0 – Timer/Counter Interrupt Mask Register

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	

- Πρέπει επίσης να είναι ενεργοποιημένος ο «γενικός διακόπτης» των INT μέσω του SREG

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – I: Global Interrupt Enable

Normal Mode Παράδειγμα 1

- Στα παραδείγματα που παρατίθενται χρησιμοποιείται ο Timer2, λόγω της δέσμευσής του Timer0 από το Arduino IDE.Το μόνο που αλλάζει ελαφρώς είναι η ονομασία των καταχωρητών.

timer2_normal_mode

```
1
2  byte state =HIGH;
3
4  void setup()
5  {
6     pinMode(12, OUTPUT);
7     TCCR2A=0x00;
8     TCCR2B=0x02; //presc8
9     TIMSK2|=0x01;
10 }
11
12 void loop()
13 {
14 }
15
16 ISR(TIMER2_OVF_vect){
17    state = !state;
18    digitalWrite(12, state);
19 }
```

Normal Mode Παράδειγμα 2

- Για δημιουργία χρόνου 1 sec με presc=8

1sec=1000msec=1.000.000μsec

Counter= 1.000.000μsec / 128μsec= 7812,5 υπερχειλίσεις

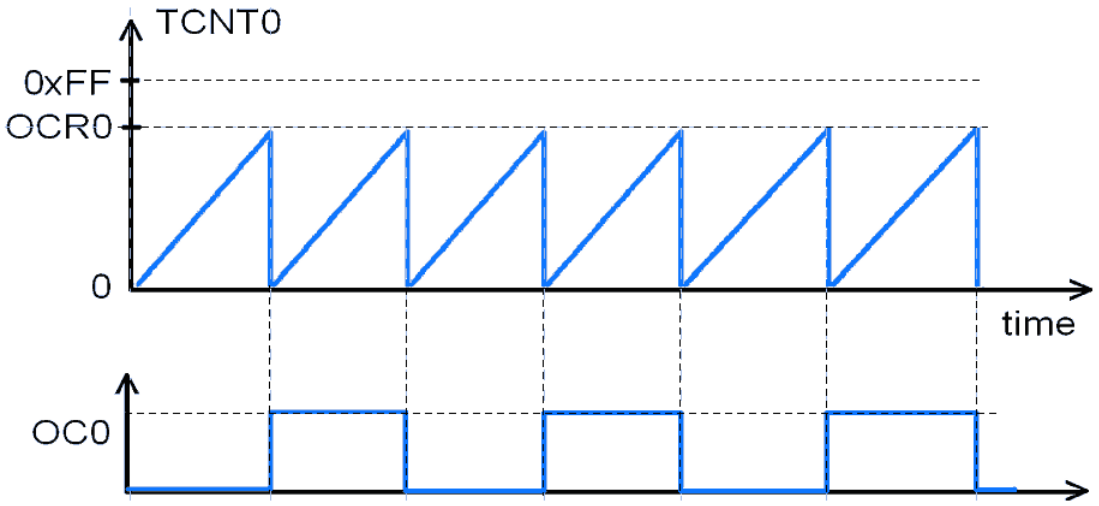
timer2_normal_mode2

```
1
2  byte state =HIGH;
3  int counter=0;
4
5  void setup()
6  {
7     pinMode(13, OUTPUT);
8     TCCR2A=0x00;
9     TCCR2B=0x02; //presc8 128μsec kathe iperxeilisi
10    TIMSK2|=0x01;
11 }
12
13 void loop()
14 {
15 }
```

```
17 ISR(TIMER2_OVF_vect)
18 {
19  counter++;
20   if (counter==7812) //sxedon 1sec
21   {
22    counter=0;
23    state = !state;
24    digitalWrite(13, state);
25   }
26 }
```

CTC Mode (Clear Timer On Compare)

- Σε αυτό το mode ο TCNT0 αυξάνει κατά 1 όπως και πριν και όταν λάβει τιμή ίση με το περιεχόμενο του καταχωρητή OCR0A-B(Output Compare Register), τότε μηδενίζεται.
- Οι OCR0A-B είναι καταχωρητές.
- Οι OC0A-B είναι ακροδέκτες του μΕ



CTC Mode Παράδειγμα

- Έστω OCR0A=250 και presc=8

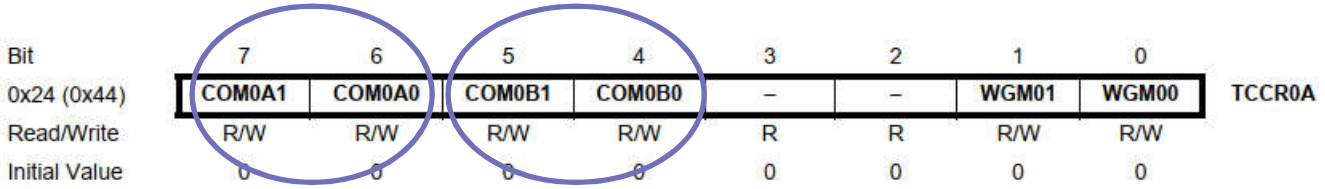
Τολ=250*0.5μ=125μsec απαιτούνται για ταύτιση του TCNT0 με OCR0A

timer2_ctc

```
1
2  byte state =HIGH;
3
4  void setup()
5  {
6     pinMode(13, OUTPUT);
7     TCCR2A=0x02; //ctc
8     OCR2A=250-1; //to metrima ksekinaei apo to miden
9     TCCR2B=0x02; //presc8 125μsec kathe tautisi
10    TIMSK2=0x02; //INT tautisis
11 }
12
13 void loop()
14 {
15 }
16
17 ISR(TIMER2_COMPA_vect)
18 {
19    state = !state;
20    digitalWrite(13, state);
21 }
```

CTC Mode (Hardware Waveform Generation)

- Υπάρχει η δυνατότητα παραγωγής 2 τετρ.κυματομορφών ανά Timer χωρίς χρήση INT μόνο με χρήση του hardware του κάθε Timer.
- Timer0 (OC0A Ard pin 6 , OC0B Ard pin5)
- Timer2 (OC2A Ard pin 11 , OC2B Ard pin3)
- Timer1 (OC1A Ard pin 9 , OC1B Ard pin10)



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	

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

CTC Mode (Hardware Example)

0 1 Toggle OC2A 1 0 CTC

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	

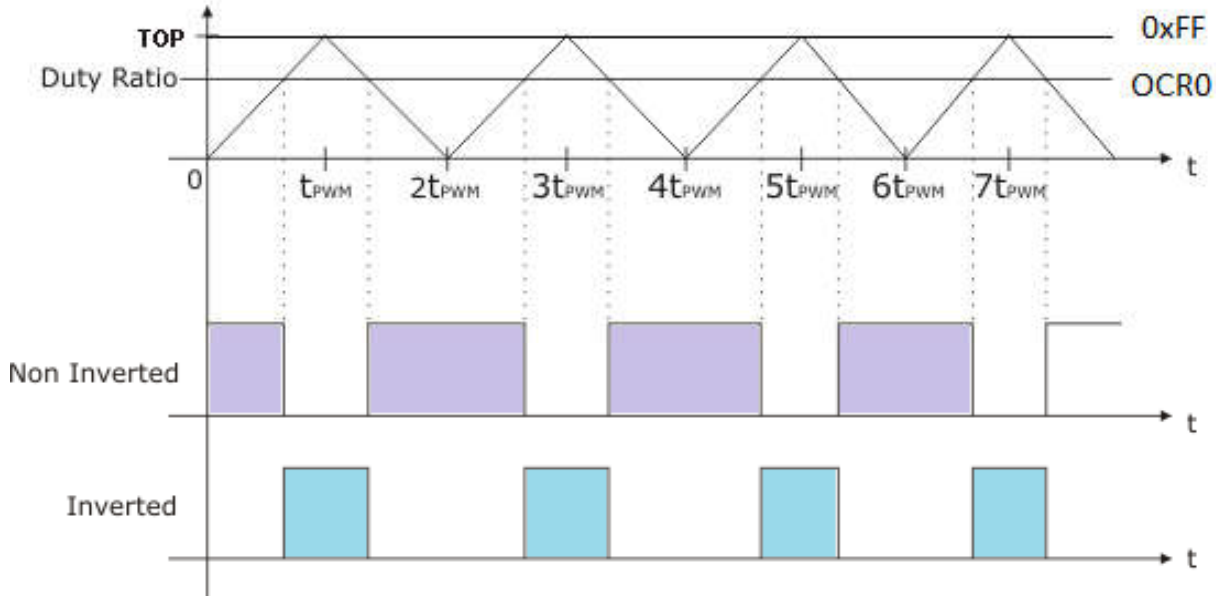
0 1 0 presc8

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	

time2_ctc_wg

```
1  //Automati paragwgi tetr. palmwn me xrisi Hardware
2
3  void setup()
4  {
5     pinMode(11, OUTPUT);
6     TCCR2A=0x42; //ctc toggle OC2A Arduino pin11
7     OCR2A=250;
8     TCCR2B=0x02; //presc8 125μsec kathe tautisi
9  }
10
11 void loop()
12 {
13 }
```

Phase Correct PWM



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

The N variable represents the prescale factor (1, 8, 64, 256, or 1024)

Timer0 -Timer2 Phase Correct PWM Frequencies Table

Prescaler N	PWM frequency
N=1	31.372,55 Hz
N=8	3921,57 Hz
N=64	490,2 Hz
N=256	122,55 Hz
N=1024	30,63 Hz

- Η ρύθμιση της Fpwm πέραν των 5 συχνοτήτων που παρέχονται , υλοποιείται μέσω του mode 5.
- Στο Mode 1 ο TCNTx counter έχει ως τιμή την TOP=0xFF
- Το duty cycle μπορεί να μεταβληθεί μέσω της τιμής των OCRxA-B.

Phase Correct PWM (Mode 1)

1 0 Non Inv Mode 0 1 Phase Correct

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	

1 0 1 0 presc8

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, Phase Correct PWM Mode

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.

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation
0	0	0	0	Normal
1	0	0	1	PWM, Phase Correct
2	0	1	0	CTC
3	0	1	1	Fast PWM
4	1	0	0	Reserved
5	1	0	1	PWM, Phase Correct
6	1	1	0	Reserved
7	1	1	1	Fast PWM

Phase Correct PWM Παράδειγμα

- Fpwm=3,92 KHZ
- Mode 1

timer2_phase_correct

```
1  //Automati paragwgi PWM me xrisi Hardware
2
3  void setup()
4  {
5     pinMode(11, OUTPUT);
6     TCCR2A=0x81; //Phase Correct Non Inv OC2A Arduino pin11
7     TCCR2B=0x02; //presc8 mode 1 Phase Correct PWM
8     OCR2A=25; //10% duty cycle
9  }
10
11 void loop()
12 {
13 }
```

DC Servo Control PWM με τον Timer1

- Για έλεγχο dc servos προτείνεται η δημιουργία Phase Correct PWM με τον Timer1, γιατί παρέχει δυνατότητες πολύ υψηλότερης ανάλυσης (8,9,10bit) σε σχέση με τους Timers 0 και 2 (8bit).
- Ο Timer1 έχει εύρος 16bit (απαριθμεί από την τιμή 0-65535) μέσω του καταχωρητή TCNT1 .
- Το duty cycle μπορεί να μεταβληθεί μέσω της τιμής των OCR1A-B (με ανάλυση 8,9 ή 10bit).
- Ο Timer1 μπορεί να λειτουργήσει σε 15 modes παρέχοντας πολλές δυνατότητες (εδώ παρατίθεται μόνο το mode10 για έλεγχο dc servo)

Phase Correct PWM με τον Timer1

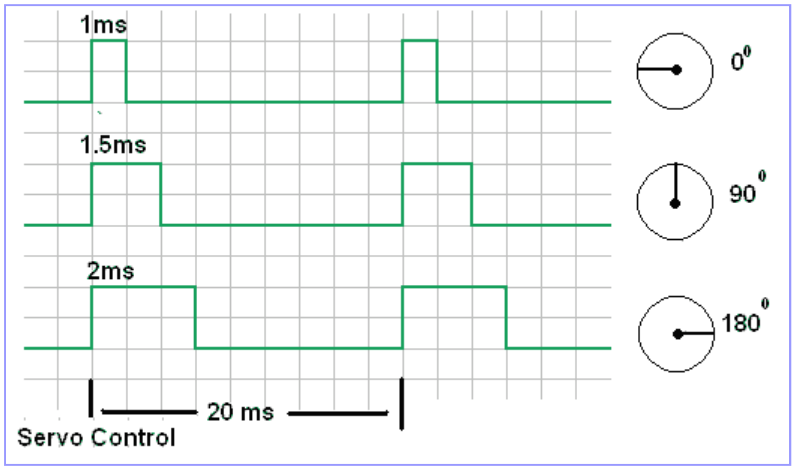
- Οι υπολογισμοί που ακολουθούν αφορούν τη μέγιστη δυνατή ανάλυση (10bit , TOP=1023, fclk=16MHz)

$$fpwm = \frac{fclk}{2 \cdot N \cdot TOP}$$

Prescaler N	PWM frequency
N=1	7820,13 Hz
N=8	977,52 Hz
N=64	122,19 Hz
N=256	30,55 Hz
N=1024	7,64 Hz

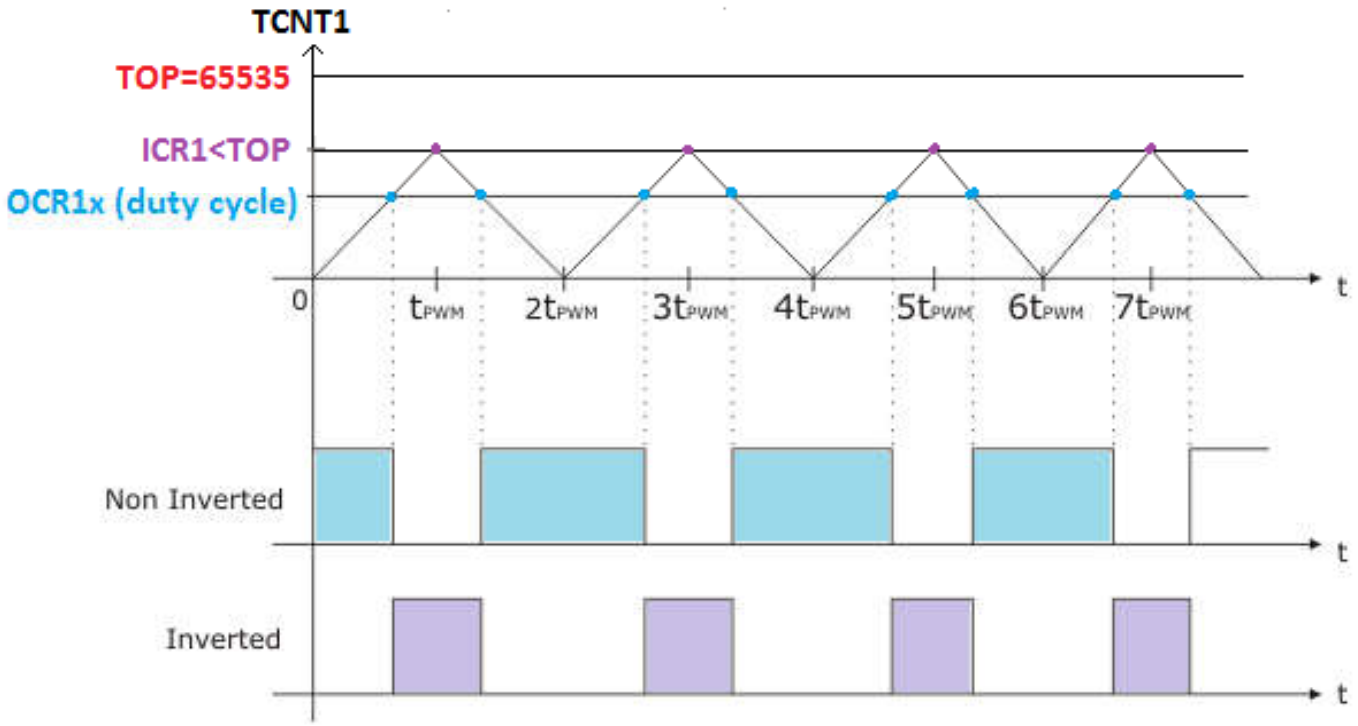
Phase Correct PWM με τον Timer1

- Υπάρχει η δυνατότητα δημιουργίας κυματομορφών PWM διαφορετικής συχνότητας (σε σχέση με αυτές της προηγούμενης διαφάνειας) .
- Τα DC Servo απαιτούν Fpwm=50Hz με διάρκεια παλμού συνήθως 1 έως 2msec.
- Κάποιες φορές υπάρχει ελαφριά διαφοροποίηση (0.5 έως 1.5msec).



Phase Correct PWM (Mode 10)

- TOP Value η τιμή του καταχωρητή ICR1<65535



Phase Correct PWM (Mode 10)

- Το TOP ανάλογα με το mode λειτουργίας μπορεί να παίρνει τιμές διαφορετικού εύρους (8,9,10 ή 16 bit).
- Εύρεση τιμής TOP με βάση την επιθυμητή συχνότητα Fpwm (η τιμή TOP είναι εύρους 16bit (0-65535) για το mode 10.

- Έστω Fpwm=50Hz και presc=8

$$fpwm = \frac{fclk}{2 \cdot N \cdot TOP} \Rightarrow TOP = \frac{fclk}{2 \cdot N \cdot fpwm} = \frac{16 \cdot 10^6}{2 \cdot 8 \cdot 50} = 20000$$

- Άρα για να δημιουργήσουμε Fpwm=50 Hz TOP=20000

Phase Correct PWM (Mode 10)

- Ton1=0.5msec και Ton2=1.5msec συνεπώς

$$duty\_cycle1 = \frac{Ton1}{T} \cdot 100\% = \frac{0.5m\sec}{20m\sec} = 2,5\%$$

$$duty\_cycle2 = \frac{Ton2}{T} \cdot 100\% = \frac{1.5m\sec}{20m\sec} = 7,5\%$$

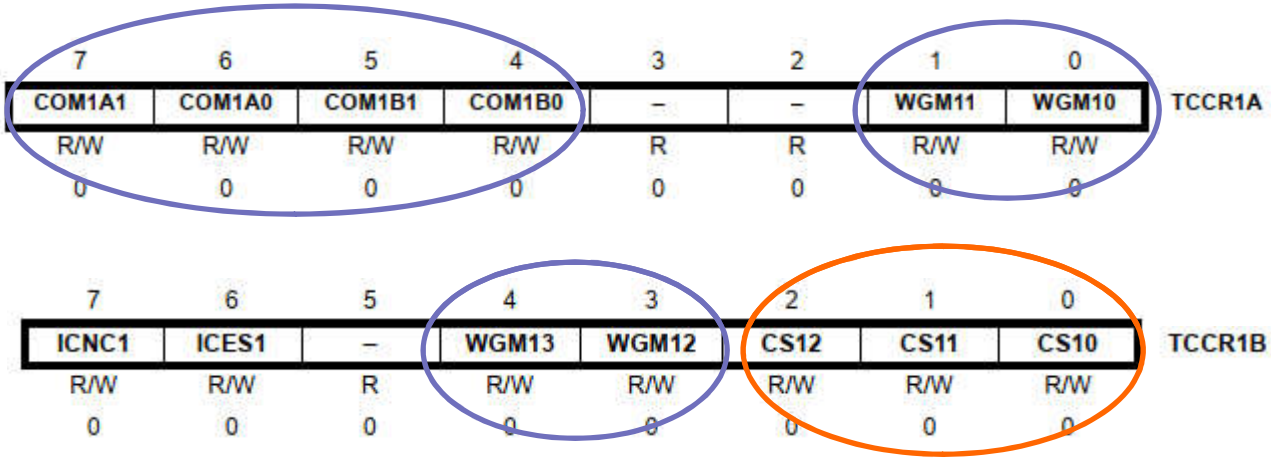
- Με βάση τα παραπάνω, το PWM θέλουμε να έχει κύκλους εργασίας μεταξύ των δύο αυτών τιμών (2.5% -7.5%) ή (5% -10%) ανάλογα τον Servοκινητήρα.

$$OCR1x = 2,5\% \cdot 20000 = 500$$

$$OCR1x = 7,5\% \cdot 20000 = 1500$$

Phase Correct PWM με τον Timer1 (Mode 10)

1 0 1 0 - - 1 0 Non Inv TOP=ICR1



1 0 0 1 0 presc8

Table 16-3. 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 11: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 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.

- OC1A Ard Pin 9
- OC1B Ard Pin10

Modes Λειτουργίας Timer1

Table 16-4. Waveform Generation Mode Bit Description[1]

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	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

Παράδειγμα Δημιουργία PWM για RC Servo x2

- Ο κώδικας δημιουργεί δύο ανεξάρτητες PWM κυματομορφές για έλεγχο θέσης 2 servo (ακροδέκτες 9 και 10 του Arduino).

timer1_2servo_drive

```
1
2  void setup()
3  {
4   pinMode(9,OUTPUT );  //OC1A
5   pinMode(10,OUTPUT ); //OC1B
6   TCCR1A = 0; //prepei na midenistoun oi kataxwrites arxika
7   TCCR1B = 0; //logw Arduino IDE (den iparxei eksigisi)
8
9   ICR1=20000; //TOP value
10  OCR1A=500;  //2,5% duty cycle
11  OCR1B=1500;  //7,5% duty cycle
12  TCCR1A=0xA2; //Energoi OC1A kai OC1B Phase Correct Mode 10
13  TCCR1B=0x12; //Presc 8
14 }
15
16 void loop()
17 {
18 }
```

Παράδειγμα 2 Έλεγχος Θέσης RC Servo μέσω 2 Button (Hard. INT INT0 και INT1)

timer1_servo_drive_up_down

```
1  void setup()
2  {
3    pinMode(2,INPUT); //btn_down (me pulldown antistasi)
4    pinMode(3,INPUT); //btn_up   (me pulldown antistasi)
5    pinMode(9,OUTPUT );  //OC1A
6   attachInterrupt(digitalPinToInterrupt(2),down,CHANGE);
7   attachInterrupt(digitalPinToInterrupt(3),up,CHANGE);
8   TCCR1A = 0; //prepei na midenistoun oi kataxwrites arxika
9   TCCR1B = 0; //logw Arduino IDE einai desmeumena
10
11  ICR1=20000; //TOP value
12  OCR1A=500;  //2,5% duty cycle (gwnia -90)
13  TCCR1A=0x82; //Energos OC1A Phase Correct Mode 10
14  TCCR1B=0x12; //Presc 8
15 }
16
17 void loop()
18 {
19 }
20
```

```
21 void up()
22 {
23   OCR1A+=50;
24   if (OCR1A>2500)
25    {
26      OCR1A=2500;
27    }
28 }
29
30 void down()
31 {
32   OCR1A-=50;
33   if (OCR1A<500)
34    {
35      OCR1A=500;
36    }
37 }
```

External Interrupt ATMEGA328 (INT0-INT1)

- Τα Hardware Interrupt (INT0-INT1) υποστηρίζονται στα pin2 και 3 του Arduino UNO, αντίστοιχα.
- Οι εξωτερικές διακοπές ελέγχονται μέσω των καταχωρητών EIMSK και EICRA.

EICRA – External Interrupt Control Register A

The External Interrupt Control Register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
(0x69)	–	–	–	–	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

External Interrupt ATMEGA328 (INT0-INT1)

EIMSK – External Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	–	–	–	–	–	–	INT1	INT0	EIMSK
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Απαραίτητη προϋπόθεση να είναι καθολικά ενεργοποιημένες οι διακοπές μέσω του SREG

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – I: Global Interrupt Enable

Παράδειγμα 3 - Έλεγχος Θέσης RC Servo μέσω 2 Button (Hard.INT υλοποίηση με καταχωρητές)

timer1_servo_drive_up_down_reg

```
1  void setup()
2  {
3    pinMode(2,INPUT); //btn_down
4    pinMode(3,INPUT); //btn_up
5    pinMode(9,OUTPUT );//OC1A sto pin9
6
7   TCCR1A = 0;  //prepei na midenistoun oi kataxwrites arxika
8   TCCR1B = 0;  //logw Arduino IDE einai desmeumena
9   EIMSK=0x03;   //Hardware INT (INT0 kai INT1) Enable
10  EICRA=0x05;   //ta INT0 kai INT1 energopoiountai me kathe allagi katastasis
11
12  ICR1=20000; //TOP value
13  OCR1A=500;  //2,5% duty cycle (gwnia -90)
14  TCCR1A=0x82; //Energos OC1A Phase Correct Mode 10
15  TCCR1B=0x12; //Presc 8
16 }
17
18 void loop()
19 {
20 }
```

```
22 ISR(INT1_vect) //INT1 ISR
23 {
24   OCR1A+=50;
25   if (OCR1A>2500)
26    {
27      OCR1A=2500;
28    }
29 }
30
31 ISR(INT0_vect) //INT0 ISR
32 {
33   OCR1A-=50;
34   if (OCR1A<500)
35    {
36      OCR1A=500;
37    }
38 }
```

ATMEGA328 Interrupt Vectors

```
 1  Reset
 2  External Interrupt Request 0  (pin D2)          (INT0_vect)
 3  External Interrupt Request 1  (pin D3)          (INT1_vect)
 4  Pin Change Interrupt Request 0 (pins D8 to D13) (PCINT0_vect)
 5  Pin Change Interrupt Request 1 (pins A0 to A5)  (PCINT1_vect)
 6  Pin Change Interrupt Request 2 (pins D0 to D7)  (PCINT2_vect)
 7  Watchdog Time-out Interrupt                     (WDT_vect)
 8  Timer/Counter2 Compare Match A                  (TIMER2_COMPA_vect)
 9  Timer/Counter2 Compare Match B                  (TIMER2_COMPB_vect)
10  Timer/Counter2 Overflow                         (TIMER2_OVF_vect)
11  Timer/Counter1 Capture Event                    (TIMER1_CAPT_vect)
12  Timer/Counter1 Compare Match A                  (TIMER1_COMPA_vect)
13  Timer/Counter1 Compare Match B                  (TIMER1_COMPB_vect)
14  Timer/Counter1 Overflow                         (TIMER1_OVF_vect)
15  Timer/Counter0 Compare Match A                  (TIMER0_COMPA_vect)
16  Timer/Counter0 Compare Match B                  (TIMER0_COMPB_vect)
17  Timer/Counter0 Overflow                         (TIMER0_OVF_vect)
18  SPI Serial Transfer Complete                    (SPI_STC_vect)
19  USART Rx Complete                               (USART_RX_vect)
20  USART, Data Register Empty                      (USART_UDRE_vect)
21  USART, Tx Complete                              (USART_TX_vect)
22  ADC Conversion Complete                         (ADC_vect)
23  EEPROM Ready                                    (EE_READY_vect)
24  Analog Comparator                               (ANALOG_COMP_vect)
25  2-wire Serial Interface  (I2C)                  (TWI_vect)
26  Store Program Memory Ready                      (SPM_READY_vect)
```