ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Μικροεπεξεργαστές και Περιφερειακά

Εργαστηριακή Άσκηση 2

ΕΛΕΓΧΟΣ ΑΝΟΙΚΤΟΥ ΒΡΟΧΟΥ ΣΤΡΟΦΩΝ DC ΚΙΝΗΤΗΡΑ

ΝΗΡΑΣ ΔΗΜΗΤΡΗΣ 8057
ΧΑΤΖΗΘΩΜΑ ΑΝΤΡΕΑΣ 8026

Θερινό Εξάμηνο 2016

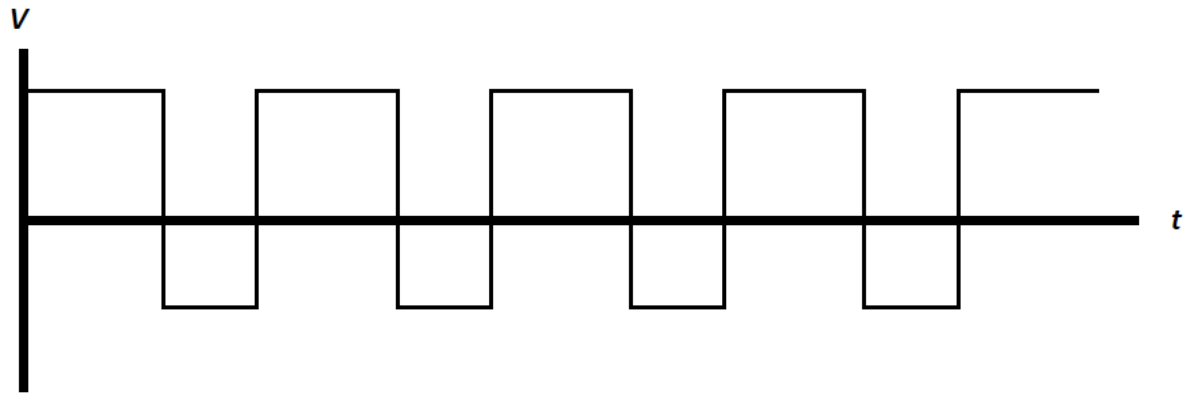
Εισαγωγή

Στην εργασία αυτη καλούμαστε να κάνουμε έλεγχο των στροφών ενός dc κινητήρα χρησιμοποιώντας την PWM έξοδο του AVR ATmega16. Στο $1^{ο}$ τμήμα της εργασίας, ο κινητήρας ελέγχεται από ένα ποτενσιόμετρο που επηρεάζει την τάση τροφοδοσίας του. Στο $2^{ο}$ τμήμα της άσκησης χρησιμοποιείται ο PWM και με το πάτημα 2 συγκεκριμένων SW αυξάνουμε ή/και μειώνουμε το duty cycle. Ακολούθως αυτό γίνεται αυτόματα με τς χρήση ενός timer.

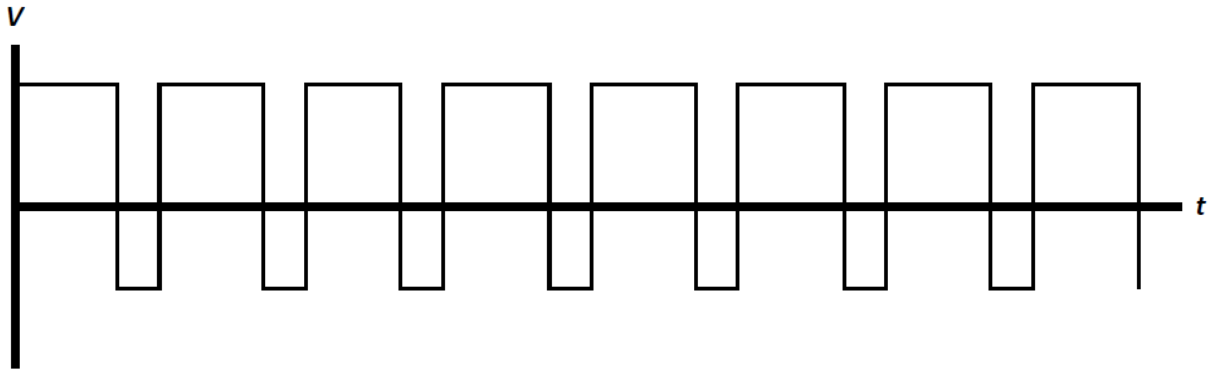
Άσκηση Α2

Αρχικά παρατηρήσαμε στον παλμογράφο την μορφή του σήματος από τον 7414 για 10 διαφορετικές τιμές. Παρακάτω παραθέτουμε ενδεικτικά 3 κυματομορφές με $\frac{Volt}{division} = 1\ V$ και $\frac{time}{division} = 50\ \mu s$.

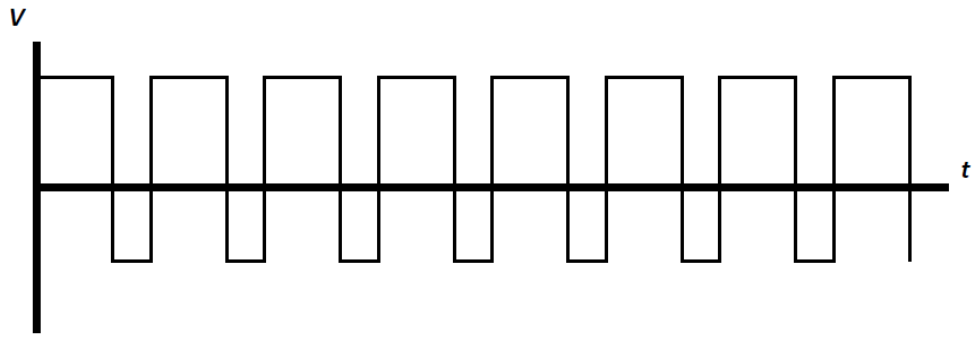
$$V\ =\ 3$$



$$V = 4$$



$$V = 5$$



Ακολούθως πάρθηκαν οι ζητούμενες μετρήσεις και υπολογίστηκε η ταχύτητα του κινητήρα σε rpm όπου $rps = \frac{32\ [παλμοί/κύκλο]}{f}$ και $rpm = rps * 60sec.$

Αριθμός Μέτρησης	Voltage [Volt]	T [sec]	f [Hz]	RPM
1	3.1	0.00024	4166.667	7812.50001
2	3.2	0.000235	4255.319	7978.72341
3	3.3	0.00022	4545.455	8522.72728
4	3.5	0.00019	5263.158	9868.42104
5	3.6	0.000185	5405.405	10135.1351
6	3.8	0.00018	5555.556	10416.6667
7	4.1	0.00017	5882.353	11029.4118
8	4.3	0.000155	6451.613	12096.7742
9	4.6	0.00014	7142.857	13392.8571
10	4.9	0.00011	9090.909	17045.4545

Άσκηση Α3

Για την άσκηση Α3 υλοποιήθηκε ο παρακάτω κώδικας. Επεξηγήσεις για τον τρόπο λειτουργίας του κώδικα βρίσκονται εντός του κώδικα υπό την μορφή σχόλιων.

1	2

.include "m16def.inc"
.cseg
 	
;definitions
.def temp = r16
.def holder1L = r17
.def result = r18
.def counter = r19
.def holder2L = r20
 	
;init interupt vectors addresses
.org 0x0000
rjmp reset
.org 0x000A
rjmp TIM1_CAPT
 	
reset:
;initialize Stack Pointer
ldi temp, high(RAMEND)
out SPH, temp
ldi temp, low(RAMEND)
out SPL, temp
 	
;set LEDs to portB
ldi temp, 0b11111111
out DDRB, temp
 	
;set Switches to portA
ldi temp, 0b01111110
out DDRA, temp
 	
;Set Input ICP1 to portD
ldi temp,0b10111111
out DDRD, temp
 	
;clear counter (counter's start value = 0)
clr temp
out TCNT1H, temp
out TCNT1L, temp
 	
;clear ICR1
out ICR1H, temp
out ICR1L, temp
 	
pulse:
	;clear ACSR
	out ACSR, temp
	
	;Enable TICIE1 bit of TIMSK (pulse mode interupt)
	ldi temp, 1<<TICIE1
	out TIMSK, temp
	
	;set prescaler CK/64
	ldi temp, 0b00000000
	out TCCR1A, temp
	
	ldi temp, 0b00000011
	out TCCR1B, temp
	
	;set counter (number of pulses)
	ldi counter, 1
	
	;enable interupts (StatusRegister{I} = 1)
	sei
	
	loop:
	;if (SW_A0==pressed) -> display result
	;else -> inf loop until interupt happens
	sbis PINA, 0
	out PORTB, result
	
	rjmp loop
	
	TIM1_CAPT:
	;if counter==0 -> start again
	cpi counter, 0
	breq start_again
	
	;if counter!=0 -> get timer's value
	cpi counter, 1
	breq pulse
	
	reti

3	

;get the current timer value and decrease counter's value
in holder1L, ICR1L
dec counter
 	
reti
 	
start_again:
;calculate pulse' duration
in holder2L, ICR1L
sub holder2L, holder1L
 	
;get result
mov result, holder2L
com result
 	
;clear TCNT1
clr temp
out TCNT1H, temp
out TCNT1L, temp
 	
;clear ICR1
out ICR1H, temp
out ICR1L, temp
 	
;clear ACSR
out ACSR, temp
 	
;set counter's value
ldi counter, 1
 	
reti	

Οι τιμές που καταγράφτηκαν είναι οι παρακάτω:

Αριθμός Μέτρησης	U_{motor} [binary]	U_{motor} [decimal]
1	00000011	3
2	00001100	12
3	00010100	20
4	00011101	29
5	00100110	38
6	00101111	47
7	00111000	56
8	01000000	64
9	01001010	74
10	01010011	83

Άσκηση Β1

Για την άσκηση Β1 υλοποιήθηκε ο παρακάτω κώδικας. Επεξηγήσεις για τον τρόπο λειτουργίας του κώδικα βρίσκονται εντός του κώδικα υπό την μορφή σχόλιων.

1	2

.include "m16def.inc"
.cseg
 	
.def temp = r16
.def state = r17
.def temp2 = r18
.def d1 = r19
.def d2 = r20
 	
.org 0x0000
rjmp reset
 	
reset:
;initialize Stack Pointer
ldi temp, high(RAMEND)
out SPH, temp
ldi temp, low(RAMEND)
out SPL, temp
 	
;set LEDs to portB
ldi temp, 0b11111111
out DDRB, temp
 	
;set Switches to PortA
ldi temp, 0b01111110
out DDRA, temp
 	
;turn off leds
ser temp
out PORTB, temp
 	
;set 7414's Input to portD
ldi temp, 0b00100000
out DDRD, temp
 	
;set PWM (9bit mode)
ldi temp, 0b11000010
out TCCR1A, temp
 	
;set prescaler to 1
ldi temp, 0b00000001
out TCCR1B, temp
 	
;set duty cycle to 20%
ldi temp, 1
out OCR1AH, temp
	ldi temp, 154
	out OCR1AL, temp
	
	clr state
	
	loop:
	sbis PINA, 0 ;if (pinA_0==1) -> skip
	rjmp increament
	
	sbis PINA, 7 ;if (pinA_7==1) -> skip
	rjmp decreament
	
	rjmp loop
	
	increament:
	sbic PINA, 0 ;if (pinA_0==0) -> skip
	rjmp incr_cont
	
	rjmp increament
	
	incr_cont:
	call delay
	
	;get OCR1AH
	in temp2, OCR1AH
	
	;compare state and check if increase is needed or not
	cpi state, 10
	breq no_increament ;if (state==10) -> don't increase state
	inc state ;increase state
	out PORTB, state
	
	;get OCR1AL
	in temp, OCR1AL
	subi temp, 26 ;sub with 26
	
	brcc no_carry ;check for carry (OCR1AL overflow)
	dec temp2 ;decrease
	
	no_carry:
	out OCR1AH, temp2
	out OCR1AL, temp

3	4

no_increament:
rjmp loop
 	
decreament:
sbic PINA, 7 ;if (pinA_7=0) -> skip
rjmp decr_cont
 	
rjmp decreament
 	
decr_cont:
call delay
 	
in temp2,OCR1AH ;set OCR1AH
cpi state, 0 ;compare if (state==0)
breq no_increament
 	
dec state
out PORTB, state
 	
in temp, OCR1AL
ldi d1, 26
add temp, d1
brcc no_carry
 	
inc temp2
 	
rjmp no_carry
 	
;delay implement based on micro-1 method
delay:
ldi d1, 0xFF
outer:
dec d1
breq endit
 	
ldi d2, 0xFF
 	
inner:
nop
nop
dec d2
breq outer
 	
rjmp inner
endit:
ret	

Άσκηση Β2

Για την άσκηση Β2 υλοποιήθηκε ο παρακάτω κώδικας. Επεξηγήσεις για τον τρόπο λειτουργίας του κώδικα βρίσκονται εντός του κώδικα υπό την μορφή σχόλιων.

1	2

.include "m16def.inc"
.cseg
 	
.def temp = r16
.def state = r17
.def temp2 = r18
.def d1 = r19
.def d2 = r20
.def counter = r21
.def state2 = r22
.def flag = r23
 	
.org 0x0000
rjmp reset
 	
.org 0x0012
rjmp TIM0_OVFL
 	
reset:
;initialize Stack Pointer
ldi temp, high(RAMEND)
out SPH, temp
ldi temp, low(RAMEND)
out SPL, temp
 	
;**set Ports (I/O)**
 	
;PortD
ldi counter, 16
ldi temp, 0b00100000
out DDRD, temp
 	
;PortA
ldi temp, 0b01111110
out DDRA, temp
 	
;PORTB
ldi temp, 0b11111111
out DDRB, temp
 	
;init TIMSK -> TOIE0==1
ldi temp, 1<<TOIE0
out TIMSK, temp
	;TIMER0
	
	;init TCNT0
	ldi temp, 12
	out TCNT0, temp
	
	;init TCCR0
	ldi temp, 0b00000101
	out TCCR0, temp
	
	;TIMER1
	
	;TCCR1B/A
	ldi temp, 0b11000010
	out TCCR1A, temp
	ldi temp, 0b00000001
	out TCCR1B, temp
	
	;OCR1AH/L
	ldi temp, 1
	out OCR1AH, temp
	ldi temp, 54
	out OCR1AL, temp
	
	;init state
	ldi state, 0
	out PORTB, state
	
	;init flag
	ldi flag, 0xFF
	
	sei
	loop:
	rjmp loop
	
	increament:
	in temp2, OCR1AH
	out PORTB, state
	
	in temp, OCR1AL
	subi temp, 26
	brcc no_carry
	
	dec temp2

3	4

no_carry:
out OCR1AH, temp2
out OCR1AL, temp
 	
ret
 	
decreament:
in temp2, OCR1AH
out PORTB, state
 	
in temp, OCR1AL
ldi d1, 26
add temp, d1
brcc no_carry
 	
inc temp2
 	
no_carry2:
out OCR1AH, temp2
out OCR1AL, temp
 	
ret
 	
TIM0_OVFL:
dec counter
brne restart
 	
ldi counter,16
cpi state, 10
breq inverse
 	
df:
inc state
cpi flag, 0
breq decreas
 	
call increament
 	
rjmp restart
 	
decreas:
call decreament
 	
rjmp restart
 	
inverse:
com flag
	clr state
	
	rjmp df
	
	restart:
	ldi temp, 12
	out TCNT0,temp
	
	reti