Εργαστηριακές Ασκήσεις Κεφ. 18

Μπαντής Αστέριος (Αυτ. XXXXXXX)

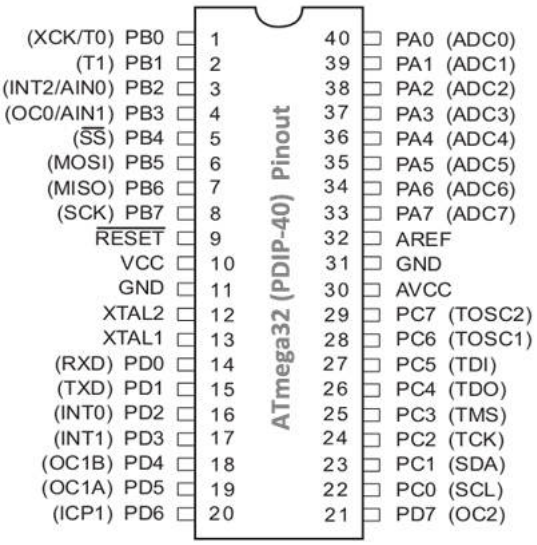
2021-11-28
Σύνδεσμος: censored

Μικροελεγκτές

Περιεχόμενα

1 Εισαγωγή

Στο κεφάλαιο 18 του βιβλίου μαθαίνουμε τον τρόπο χειρισμού του χρονιστή των 16 bits (χρονιστής T1) του ATMega32A. Ο χρονιστής αυτός είναι αρκετά όμοιος με τους χρονιστές των 8 bits, με ειδοποιούς διαφορές την ύπαρξη 2 καταχωρητών ελέγχου (TCCR1A, TCCR1B) αντί ενός (TCCR0), την ύπαρξη 2 καταχωρητών σύγκρισης (OCR1A, OCR1B) αντί ενός (OCR0), και τη δυνατοτητα μέτρησης μέχρι $2^{16} - 1$ παλμών ρολογίου (256 φορές μεγαλύτερη ποσότητα σε σχέση με τον T0/T2). Σε αυτό το κεφάλαιο μαθαίνουμε, επίσης, τον τρόπο με τον οποίο ο μικροελεγκτής χειρίζεται τους αριθμούς των 16 bits ενώ διαθέτει κυρίως καταχωρητές των 8 bits.



Εικόνα 1: Το διάγραμμα pinout του ATMega32A

2 Άσκηση 18.1

Δοκιμάστε τα προγράμματα των παραδειγμάτων αυτού του κεφαλαίου.

2.1 Παράδειγμα 1

2.1.1 Πρόγραμμα

```
.include "m32def.inc"
.def work=R16
.equ xronos=15624

.org 0
```

```
    RJMP reset

.org OC1Aaddr
    RJMP T1compA

reset:
    LDI work,high(RAMEND)
    OUT SPH,work
    LDI work,low(RAMEND)
    OUT SPL,work

    SBI DDRC,0
    CBI PORTC,0

    LDI work,high(xronos)
    OUT OCR1AH,work
    LDI work,low(xronos)
    OUT OCR1AL,work

    CLR work
    OUT TCCR1A,work
    LDI work,(1<<WGM12)|(1<<CS12)|(1<<CS10)
    OUT TCCR1B,work

    LDI work,(1<<OCIE1A)
    OUT TIMSK,work
    OUT TIFR,work

    SEI
loop:
    NOP
    RJMP loop

T1compA:
    SBIC PINC,0
    CBI PORTC,0
    SBIS PINC,0
    SBI PORTC,0
    RETI
```

2.2 Παράδειγμα 2

2.2.1 Πρόγραμμα

```
.include "m32def.inc"
def work=R16
def work2=R17
.equ period=16000

.org 0
    RJMP reset

.org 0x00E
    RJMP TlcompA

reset:
    LDI work,low(RAMEND)
    OUT SPL,work
    LDI work,high(RAMEND)
    OUT SPH,work

    LDI work,high(period)
    OUT OCR1AH,work
    LDI work,low(period)
    OUT OCR1AL,work

    LDI work,(1≪OCF1A)
    OUT TIFR,work
    OUT TIMSK,work

    CLR work
    OUT TCCR1A,work

    LDI work, (1<<CS10)
    OUT TCCR1B,work
    SEI

loop:
   RJMP loop

T1compA:
    PUSH work
```

```
    IN work,SREG
    PUSH work

    IN work2,OCR1AL
    SUBI work2,low(-period)

    IN work,OCR1AH
    SBCI work,high(-period)
    OUT OCR1AH,work
    OUT OCR1AL,work2


    POP work
    OUT SREG,work
    POP work
    RETI
```

2.3 Παράδειγμα 3

2.3.1 Πρόγραμμα

```
.include "m32def.inc"
.def work=R16
.equ period=16000-1

.org 0
    RJMP reset

.org OC1Aaddr
RJMP T1ctcA

reset:
    LDI work,low(RAMEND)
    OUT SPL,work
    LDI work,high(RAMEND)
    OUT SPH,work

    LDI work,high(period)
    OUT OCR1AH,work
    LDI work,low(period)
    OUT OCR1AL,work
```

```
    LDI work,(1<<OCF1A)
    OUT TIFR,work
    OUT TIMSK,work

    CLR work
    OUT TCCR1A,work
    LDI work,(1<<CS10)|(1<<WGM12)
    OUT TCCR1B,work

    SEI
loop:
    RJMP loop

T1ctcA:
    RETI
```

2.4 Παράδειγμα 4

2.4.1 Πρόγραμμα

```
.include "m32def.inc"

.def work=R16
.def ST2=R17
.def ST3=R18
.equ period=32000-1

.org 0
   RJMP reset

.org OC1Aaddr
    RJMP T1ctcA

reset:
    LDI work,low(RAMEND)
    OUT SPL,work
    LDI work,high(RAMEND)
    OUT SPH,work
    LDI ST2,2
    LDI ST3.3
    LDI work,high(period)
    OUT OCR1AH,work
```

```
    LDI work,low(period)
    OUT OCR1AL,work
    LDI work,(1<<OCF1A)
    OUT TIFR,work
    OUT TIMSK,work

    CLR work
    OUT TCCR1A,work
    LDI work,(1<<CS10)|(1<<WGM12)
    OUT TCCR1B,work
    SEI

loop:
    RJMP loop

T1ctcA:
    PUSH work
    IN work,SREG
    PUSH work
    DEC ST2
    BRNE OXI4ms
    LDI ST2,2
    NOP
OXI4ms:
    DEC ST3
    BRNE telos
    LDI ST3,3
    NOP
telos:
    POP work
    OUT SREG,work
    POP work
    RETI
```

2.5 Επεξήγηση

Στο πρώτο παράδειγμα ο χρονιστής T1 λειτουργεί σε λειτουργεία μηδενισμού, αντιστρέφοντας την κατάσταση ενός (ψευδο)τυχαίου ακροδέκτη κάθε 1 (προσομοιωμένο) δευτερόλεπτο. Το breakpoint τοποθετείται στην εντολή `RETI` στο τέλος του κώδικα. Πράγματι, το πρόγραμμα φτάνει στην εντολή αυτή μετά από 16000000 κύκλους. Ο AVR Simulator έφτασε σε αυτό το σημείο μετά από 63 δευτερόλεπτα. Δηλαδή, για κάθε 63 πραγματικά δευτερόλεπτα, για

τον προσομοιωτή περνά 1 δευτερόλεπτο. Στο παράδειγμα 2 ξανά λειτουργούμε τον χρονιστή T1 σε λειτουργία σύγκρισης, αλλά για χρόνο 1ms. Το πρόγραμμα αυτό λειτουργεί σε 16000 κύκλους. Στο παράδειγμα 3 λειτουργούμε τον χρονιστή σε λειτουργία μηδενισμού και το πρόγραμμα εκτελείται σε 16000±1 κύκλους. Τέλος, στο παράδειγμα 18.4 εκτελούμε 2 διακοπές με τον χρονιστή T1 σε λειτουργία μηδενισμού και παίρνουμε interrupt κάθε 32000 κύκλους. Το interrupt αυτό μειώνει το περιεχόμενο των software timers, οι οποίοι, όταν μηδενιστούν, δίνουν interrupts (στους 64000 και 96000 κύκλους αντίστοιχα).

3 Άσκηση 18.2

Συνδέστε ένα μπουτόν στη γραμμή T0 (PB0) και ένα LED στη γραμμή OC1A (PD5). Για να μην αναγνωρίζονται οι αναπηδήσεις της επαφής ως πολλαπλές πιέσεις του μπουτόν συναρμολογήστε το φίλτρο RC που φαίνεται στο σχήμα 18.2 [του βιβλίου]. Γράψτε ένα πρόγραμμα που να ανάβει επί 1s το LED κάθε 10 πιέσεις του μπουτόν. Χρησιμοποιήστε τον T0 ως μετρητή εξωτερικών παλμών και τον T1 ως χρονομέτρη για το 1s.

3.1 Πρόγραμμα

```
.include "m32def.inc"
.equ Palmoi = 15625-1

.org 0x0000
    RJMP reset

.org 0x000E
    RJMP T1_COMPA

.org 0x0014
    RJMP T0_COMP

reset:
    LDI R16,low(RAMEND)
    OUT SPL,R16
    LDI R16,high(RAMEND)
    OUT SPH,R16

    CBI DDRB,0
    SBI PORTB,0
    SBI DDRD,5
    CBI PORTD,5
```

```
    LDI R16,9
    OUT OCR0, R16

    LDI R16,high(Palmoi)
    OUT OCR1AH,R16
    LDI R16,low(Palmoi)
    OUT OCR1AL,R16

    LDI R16,(1<<OCF0)|(1<<OCF1A)
    OUT TIFR,R16
    OUT TIMSK,R16

    LDI R16,(1<<COM1A0)
    OUT TCCR1A,R16

    LDI R16,(1<<CS02)|(1<<CS01)|(1<<WGM01)
    OUT TCCR0,R16

    SEI

main:
    RJMP main

T0_COMP:
    LDI R16,(1<<COM1A0)|(1<<FOC1A)
    OUT TCCR1A,R16
    LDI R16,(1<<CS12)|(1<<CS10)|(1<<WGM12)
    OUT TCCR1B,R16
    RETI

T1_COMPA:
    CLR R16
    OUT TCCR1B,R16
    OUT TCNT1H,R16
    OUT TCNT1L,R16
    RETI
```

3.2 Επεξήγηση

Η άσκηση 18.2 χρησιμοποιεί και τον χρονιστή T0, και τον χρονιστή T1. Στην αρχή του προγράμματος ενεργοποιούμε το διάνυσμα TIMER1_COMPA (OC1Addr) για να ενεργοποιήσουμε

τη λειτουργία σύγκρισης A του T1 και το διάνυσμα TIMER0_COMP για να ενεργοποιήσουμε τη λειτουργία σύγκρισης του T0. Στο setup αρχικοποιούμε τον stack pointer και ορίζουμε τον ακροδέκτη T0 (PB0) ως είσοδο με pull-up και τον ακροδέκτη OC1A (PD5) ως έξοδο με αρχική κατάσταση "0". Στον OCR0 τοποθετούμε την τιμή των πατημάτων του μπουτόν που θέλουμε να μετρήσουμε, και στον OCR1A τοποθετούμε τους παλμούς ρολογιού που αντιστοιχούν σε 1s (σύμφωνα με τους υπολογισμούς που υπάρχουν στο τέλος της προηγούμενης εργαστηριακής αναφοράς), **γράφοντας *πρώτα* το *high* και *μετά* το *low***. Έπειτα αλλάζουμε κατάσταση στα bits OCF0 και OCF1A του TIFR και τα αντίστοιχα bits του TIMSK και ρυθμίζουμε τον TCCR1A (ανατροπή OC1A σε ισότητα) και TCCR0 (εξωτερικό ρολόι στον ακροδέκτη T0 σε κατερχόμενη παρυφή, CTC mode) σύμφωνα με τα πινακάκια του βιβλίου. Στη συνέχεια ενεργοποιούμε τα interrupts και μπαίνουμε στη main. Σε κάθε 10 πατήματα του μπουτόν ενεργοποιείται η υπορουτίνα T0_COMP, η οποία ξαναρυθμίζει τον TCCR1A (ανατροπή OC1A σε ισότητα, force output compare), ρυθμίζει τον TCCR1B (Prescaler=1024, CTC με κορυφαία τιμή = OCR1A), και επιστρέφει στην κυρίως ρουτίνα. Σε αυτό το σημείο το LED είναι ενεργό. Μετά από 1 δευτερόλεπτο ο T1 δίνει interrupt και μπαίνουμε στην υπορουτίνα T1_COMPA, η οποία απενεργοποιεί τον χρονιστή 1, καθαρίζοντας τους καταχωρητές TCCR1B, TCNT1H, και TCNT1L. Έπειτα επιστρέφει στο κυρίως πρόγραμμα.
Στη δική μας περίπτωση, παρόλο που υπήρχε το φίλτρο RC, συνεχίζαμε να έχουμε πρόβλημα με το debouncing επειδή το μπουτόν μας δεν έκανε πολύ καλή επαφή με το ράστερ, ενώ αρκετές φορές έβγαινε εντελώς απο τη θέση του. Έτσι, στη δική μας περίπτωση το LED άναβε κυρίως σε 7±2 πατήματα του μπουτόν. Το πρόβλημα αυτό μπορεί να λυθεί με τον εξής "μπακάλικο" τρόπο: επειδή κατά μέσο όρο ενεργοποιείται σε 7 πατήματα, μπορούμε να αλλάξουμε την τιμή του OCR0 από 9 σε 12. Έτσι, θα ενεργοποιείται κατά μέσο όρο σε 10 πατήματα. Βέβαια, τέτοιες χαμηλής πιστότητας λύσεις δεν προτείνονται.

4 Άσκηση 18.3

Να γραφεί πρόγραμμα που να κάνει τρεις χρονομετρήσεις, μία κάθε 2ms, μία κάθε 3ms, και μία κάθε *περίπου* 4ms. Να χρησιμοποιηθούν οι τρεις διακοπές του T1, υπερχείλισης, σύγκρισης A, και σύγκρισης B, όπως φαίνεται στο σχήμα 18.3 και στον πίνακα 18.8 του βιβλίου

4.1 Πρόγραμμα

```
.include "m32def.inc"

.equ compA = 32000
.equ compB = 48000

.org 0x0000
    RJMP reset

.org 0x000E
    RJMP T1_COMPA

.org 0x0010
    RJMP T1_COMPB

.org 0x0012
    RJMP T1_OVF

reset:
    LDI R16,high(RAMEND)
    OUT SPH,R16
    LDI R16,low(RAMEND)
    OUT SPL,R16

    LDI R16,high(compA)
    OUT OCR1AH,R16
    LDI R16,low(compA)
    OUT OCR1AL,R16


    LDI R16,high(compB)
    OUT OCR1BH,R16
    LDI R16,low(compB)
    OUT OCR1BL,R16

    LDI R16,(1<<OCF1B)|(1<<OCF1A)|(1<<TOV1)
    OUT TIFR,R16
    OUT TIMSK,R16

    LDI R16,(1<<CS10)
    OUT TCCR1B,R16
```

```
    SEI

main:
    RJMP main

T1_COMPA:
    PUSH R16
    IN R16,SREG
    PUSH R16
    PUSH R17

    IN R16,OCR1AL
    SUBI R16,low(-compA)
    IN R17, OCR1AH
    SBCI R17,high(-compA)

    OUT OCR1AH,R17
    OUT OCR1AL,R16

    NOP

    POP R17
    POP R16
    OUT SREG, R16
    POP R16

    RETI

T1_COMPB:
    PUSH R16
    IN R16,SREG
    PUSH R16
    PUSH R17

    IN R16, OCR1BL
    SUBI R16,low(-compB)
    IN R17, OCR1BH
    SBCI R17,high(-compB)

    OUT OCR1BH,R17
```

```
    OUT OCR1BL,R16
    NOP
    POP R17
    POP R16
    OUT SREG,R16
    POP R16

    RETI

T1_OVF:
    NOP
    RETI
```

4.2 Επεξήγηση

Στην αρχή του προγράμματος ενεργοποιούμε τα διανύσματα των τριών διακοπών που δίνει ο χρονιστής T1. Αρχικοποιούμε τον stack pointer και τοποθετούμε τις υπολογισμένες τιμές κύκλων στους καταχωρητές OCR1A (2ms) και OCR1B (3ms) αντίστοιχα. Αλλάζουμε κατάσταση στα bits OCF1B και OCF1A του TIFR και ενεργοποιούμε το bit TOV1 του TIFR. Αλλάζουμε κατάσταση στα bits των ίδιων θέσεων και στον TIMSK. Ρυθμίζουμε τον prescaler(=1) στον TCCR1B και ενεργοποιούμε τα interrupts. Στο main εκτελείται οποιοδήποτε πρόγραμμα. Πρώτα δίνει interrupt ο καταχωρητής σύγκρισης A και μπαίνουμε στη ρουτίνα T1_COMPA. Η κατάσταση του R16, SREG, και R17 σώζεται στη στοίβα. Δίνεται στον OCR1A η τιμή του -compA. Επιστρέφουν στους καταχωρητές R17, SREG, και R16 τα αρχικά περιεχόμενά τους και βγαίνουμε από την υπορουτίνα. Έπειτα δίνει interrupt ο καταχωρητής σύγκρισης B και μπαίνουμε στην υπορουτίνα T1_COMPB, όπου εκτελείται πρόγραμμα αντίστοιχο με αυτό της προηγούμενης υπορουτίνας, αλλά για τον OCR1B και για -compB. Τέλος, στα (περίπου) 4ms ο μετρητής κάνει overflow και δίνει interrupt το TOV1, μπαίνοντας στην υπορουτίνα T1_OVF, εκτελώντας κάποιο πρόγραμμα, και επιστρέφοντας στο κυρίως πρόγραμμα.

5 Βιβλιογραφία

[1] Νικολαΐδης Νικόλαος, ΜΙΚΡΟΕΛΕΓΚΤΕΣ: Ασκήσεις, Πειράματα και Εφαρμογές με τον ATmega32, 2018
[2] Νικολαΐδης Νικόλαος, ΕΝΣΩΜΑΤΩΜΕΝΑ ΣΥΣΤΗΜΑΤΑ, 2020