Ε.Μ.Π. - ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧ. ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΑΚΑΔ. ΕΤΟΣ 2018-2019

ΑΘΗΝΑ 5 Δεκεμβρίου 2018

7η ΕΡΓΑΣΤΗΡΙΑΚΗΑΣΚΗΣΗ
ΓΙΑ ΤΟ ΜΑΘΗΜΑ "Εργαστήριο Μικροϋπολογιστών"
5η Εργ. Άσκ. στον Μικροελεγκτή AVR
- Σειριακή Επικοινωνία UART και ADC (στο easyAVR6**)**
Εξέταση – Επίδειξη: Τετάρτη 12/12/2018

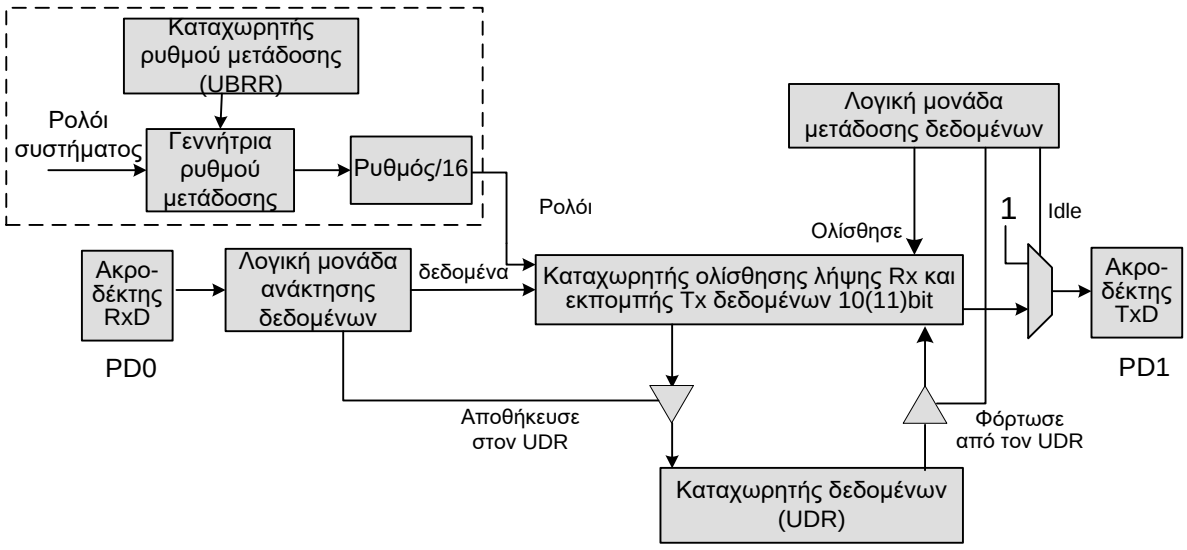
Έκθεση: Κυριακή 16/12/2018

Ασύγχρονη σειριακή επικοινωνία UART

Ο ATmega16 είναι εξοπλισμένος με μια μονάδα USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter) η οποία μπορεί να χρησιμοποιηθεί για Σύγχρονη αλλά και για Ασύγχρονη επικοινωνία. Στο πλαίσιο αυτής της άσκησης θα μελετηθεί ο Ασύγχρονος τρόπος επικοινωνίας (UART). Η μονάδα UART μπορεί να μεταδώσει τους εξής 30 συνδυασμούς πλαισίου:

- 1 bit εκκίνησης
- 5, 6, 7, 8 ή 9 bits
- Ένα η δύο bit λήξης
- Καμία, άρτια ή περιττή ισοτιμία (parity)

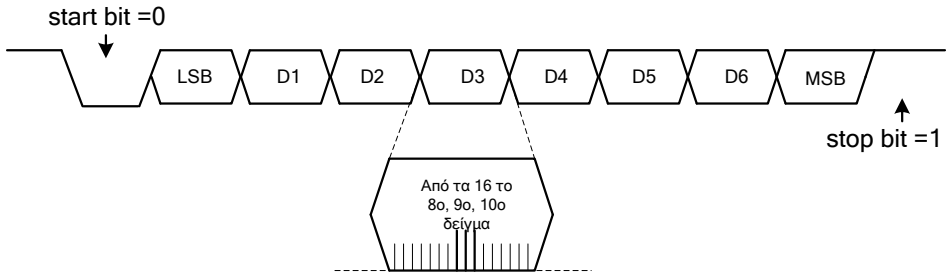
Η βασική δομή φαίνεται στο παρακάτω σχήμα:



Σχήμα 7.1. Κυκλωματικό διάγραμμα Μονάδας UART.

Παρατηρώντας το σχήμα φαίνεται ότι υπάρχει διαίρεση του ρυθμού μετάδοσης με το 16. Ο πομπός χρησιμοποιεί την διαιρεμένη συχνότητα για την αποστολή ενός bit ενώ ο δέκτης χρησιμοποιεί το 16πλάσιο της συχνότητας αυτής ώστε να δειγματοληπτεί το σήμα. Από τα δείγματα που λήφθηκαν χρησιμοποιούνται το 8^o, το 9^o και το 10^o και με τον κανόνα της πλειοψηφίας αποφασίζεται αν λήφθηκε λογικό 1, λογικό 0 ή αν πρόκειται για θόρυβο στην

περίπτωση που αναμένεται startbit (που πρέπει οπωσδήποτε να είναι 0). Επιπλέον, όταν αναγνωριστεί το start bit πραγματοποιείται και ένας ¨συγχρονισμός¨ του δέκτη με τον πομπό, αφού αναγνωρίζεται η μετάβαση από λογικό 1 σε λογικό 0 και στη συνέχεια το επόμενο bit θα καταφθάσει σε συγκεκριμένο χρόνο (1/BAUD). Προφανώς ο ρυθμός μετάδοσης, τα bit που αναμένονται, ο αριθμός των stop bit και το είδος του parity bit πρέπει να είναι όλα γνωστά στον δέκτη για να λειτουργήσει σωστά η επικοινωνία.



Σχήμα 7.2 Δειγματοληψία δεδομένου

Περισσότερες πληροφορίες σχετικά με την λειτουργία των καταχωρητών αλλά και γενικότερα της UART μπορείτε να βρείτε στις σελίδες 83-91 του βιβλίου Συστήματα Μικροϋπολογιστών ii ή και στο ATmega16.

Καταχωρητές

Καταχωρητής δεδομένων (UDR)

Από το σχήμα φαίνεται ότι πομπός και δέκτης μοιράζονται τον καταχωρητή δεδομένων UDR. Στην πραγματικότητα όμως πρόκειται για δύο διαφορετικούς καταχωρητές που έχουν κοινή φυσική διεύθυνση $0C. Ο ένας είναι ο καταχωρητής δεδομένων εκπομπής όπου μεταφέρονται τα δεδομένα που γράφουμε στη διεύθυνση $0C. Ο άλλος είναι ο καταχωρητής δεδομένων λήψης όπου μεταφέρονται τα δεδομένα που διαβάζουμε από τη διεύθυνση $0C.

Καταχωρητής κατάστασης (USCRA)

Bit	7	6	5	4	3	2	1	0	
	RXC	**TXC**	**UDRE**	**FE**	**DOR**	**PE**	**U2X**	**MPCM**	**UCSRA**
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

RXC: Αν είναι λογικό 1 τότε υπάρχουν δεδομένα προς ανάγνωση. Μετά την ανάγνωση τους η τιμή μηδενίζεται.
UDRE: Αν είναι λογικό 1 ο καταχωρητής UDR είναι έτοιμος να δεχθεί νέα δεδομένα.

Καταχωρητής ελέγχου (USCRB)

Bit	7	6	5	4	3	2	1	0	
	RXCIE	**TXCIE**	**UDRIE**	**RXEN**	**TXEN**	**UCSZ2**	**RXB8**	**TXB8**	**UCSRB**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

RXEN:Αν είναι λογικό 1 ενεργοποιείται το τμήμα λήψης της μονάδας UART
TXEN: Αν είναι λογικό 1 ενεργοποιείται το τμήμα εκπομπής της μονάδας UART

Καταχωρητής ελέγχου (USCRC)

Bit	7	6	5	4	3	2	1	0	
	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	UCSRC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	0	0	0	0	1	1	0	

URSEL: Το bit αυτό επιλέγει την πρόσβαση μεταξύ του καταχωρητή UCSRC ή του UBRRH. Διαβάζεται ως 1 όταν γίνεται ανάγνωση του UCSRC. Το URSEL πρέπει ναι είναι 1 όταν γίνεται εγγραφή στον UCSRC.

UCSZ1:0: Τα bit UCSZ1:0 σε συνδυασμό με το bit UCSZ2 του καταχωρητή UCSRB καθορίζουν των αριθμό των bit δεδομένων (Character Size) που ανταλλάσσουν πομπός και δέκτης. Συγκεκριμένα:

Table 66. UCSZ Bits Settings

UCSZ2	UCSZ1	UCSZ0	Character Size
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9-bit

Καταχωρητής του ρυθμού μετάδοσης (UBRR)

Bit	15	14	13	12	11	10	9	8	
	URSEL	–	–	–	UBRR[11:8]				UBRRH
	UBRR[7:0]								UBRRL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
	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	
	0	0	0	0	0	0	0	0	

URSEL: Το bit αυτό επιλέγει την πρόσβαση μεταξύ του καταχωρητή UCSRC ή του UBRRH. Διαβάζεται ως 1 όταν γίνεται ανάγνωση του UCSRC. Το URSEL πρέπει ναι είναι 0 όταν γίνεται εγγραφή στον UBRR.

UBRR11:0: Πρόκειται για έναν καταχωρητή 12-bitπου περιέχει το ρυθμό μετάδοσης. Ο καταχωρητής UBRRH περιέχει τα 4 MSB, και ο UBRRL περιλαμβάνει τα 8 LSB. Εγγραφή στον UBRRL ανανεώνει άμεσα τον ρυθμό και θα προκαλέσει αλλοίωση των δεδομένων σε περίπτωση που γίνει κατά τη διάρκεια εκπομπής.

Για τον υπολογισμό του ρυθμού μετάδοσης BAUD από την τιμή του καταχωρητή ισχύει ο τύπος:

$$BAUD = \frac{f_{osc}}{16(UBRR + 1)}$$

όπου f_{OSC} η συχνότητα λειτουργίας του μικροελεγκτή. Λύνοντας ως προς UBRR έχουμε:

$$UBRR = \frac{f_{osc}}{16BAUD} - 1$$

Για να έχουμε BAUD=9600 και αφού ο μικροελεγκτής μας λειτουργεί στα 8MHzπρέπει UBRR=51.0833. Στρογγυλοποιούμε στο 51, εισάγοντας ένα σφάλμα 0,16%.

Για να αρχικοποιήσουμε την USART μπορούμε να χρησιμοποιήσουμε την εξής ρουτίνα:

```
; Routine: usart_init
; Description:
; This routine initializes the
; usart as shown below.
; ------- INITIALIZATIONS -------
;
; Baud rate: 9600 (Fck= 8MHz)
; Asynchronous mode
; Transmitter on
; Reciever on
; Communication parameters: 8 Data ,1 Stop , no Parity
; --------------------------------
; parameters: None.
; return value: None.
; registers affected: r24
; routines called: None


usart_init:
clr r24             ; initialize UCSRA to zero
out UCSRA ,r24
ldi r24 ,(1<<RXEN) | (1<<TXEN)      ; activate transmitter/receiver
out UCSRB ,r24
ldi r24 ,0                          ; baud rate = 9600
out UBRRH ,r24
ldi r24 ,51
out UBRRL ,r24
ldi r24 ,(1 << URSEL) | (3 << UCSZ0)  ; 8-bit character size,
out UCSRC ,r24                         ; 1 stop bit
ret


; Routine: usart_transmit
; Description:
; This routine sends a byte of data
; using usart.
; parameters:
; r24: the byte to be transmitted
; must be stored here.
; return value: None.
; registers affected: r24
; routines called: None.
```

```
usart_transmit:
sbis UCSRA ,UDRE     ; check if usart is ready to transmit
rjmp usart_transmit ; if no check again, else transmit
out UDR ,r24         ; content of r24
ret

; Routine: usart_receive
; Description:
; This routine receives a byte of data
; from usart.
; parameters: None.
; return value: the received byte is
; returned in r24.
; registers affected: r24
; routines called: None.

usart_receive:
sbis UCSRA ,RXC      ; check if usart received byte
rjmp usart_receive  ; if no check again, else read
in r24 ,UDR          ; receive byte and place it in
ret                  ;r24
```

Μετατροπέας αναλογικής σε ψηφιακή μορφή ADC.

Βασικό περιφερειακό των μικροελεγκτών είναι ο μετατροπέας από Αναλογική σε Ψηφιακή (ADC) μορφή για την εισαγωγή, τον έλεγχο και την επεξεργασία αναλογικών ποσοτήτων από τον εξωτερικό φυσικό κόσμο, όπως θερμοκρασία, πίεση, βάρος. Γενικά, επιτρέπει τη μετατροπή σε ψηφιακή μορφή διαφορών δυναμικού που παράγονται από αισθητήρες. Η λειτουργία ενός ADC περιγράφεται ως εξής:

- Θεωρούμε αναλογική τάση εισόδου Vin και μία τάση αναφοράς V_{REF} βάση της οποίας γίνεται η μετατροπή.
- Επίσης θεωρούμε ότι για να γίνει σωστά η μετατροπή πρέπει $V_{in}<V_{REF}$
- Για ADC ανάλυσης n-bit, χωρίζεται η δυναμική περιοχή $(0 – V_{REF})$ σε 2^n ίσα διαστήματα και αντιστοιχείται το καθένα σε ένα δυαδικό αριθμό των n-bit ανάλογα με την τάξη του. Το κάθε διάστημα Q που αντιπροσωπεύει την ανάλυση του ADC είναι $Q=V_{REF}/2^n$. Αν υποθέσουμε ότι V_{REF}=2,56V και n=10 τότε Q=2,5mV.
- Γενικά η ψηφιακή έξοδος X του ADC είναι $X=[V_{in}\cdot 2^n/V_{REF}]= [V_{in}/Q]$ όπου με το σύμβολο [·] δηλώνεται το στρογγυλευμένο ακέραιο μέρος.

Ο Μικροελεγκτής AVR ATMega16 διαθέτει έναν ADC ανάλυσης 10 bit, που βασίζεται στη μέθοδο των διαδοχικών προσεγγίσεων. Ο ADC είναι συνδεδεμένος με ένα πολυπλέκτη 8 εισόδων που επιτρέπει την απλή μετατροπή των τάσεων από τα pins της θύρας PORTA. Επιτρέπει επίσης και μετατροπή από διαφορικές εισόδους (16 συνδυασμοί των 8 εισόδων), που συνδυάζονται και με ένα στάδιο προγραμματιζόμενης (x1, x10, x200) ενίσχυσης.

Περισσότερες πληροφορίες σχετικά με την λειτουργία των καταχωρητών αλλά και γενικότερα τον ADC μπορείτε να βρείτε στις σελίδες 133-137 του βιβλίου Συστήματα Μικροϋπολογιστών ii ή και στο ATmega16.

Καταχωρητές

Καταχωρητής επιλογής πολυπλέκτη (ADMUX)

Bit	7	6	5	4	3	2	1	0	
	REFS1	**REFS0**	**ADLAR**	**MUX4**	**MUX3**	**MUX2**	**MUX1**	**MUX0**	**ADMUX**
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	

REFS1:0: Χρησιμοποιείται για την επιλογή του V_{REF} σύμφωνα με τον επόμενο πίνακα:

Table 83. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

MUX4:0: Χρησιμοποιείται για την επιλογή των εισόδων που θα οδηγηθούν από τον πολυπλέκτη στον ADC καθώς και για την ρύθμιση του κέρδους. Στον παρακάτω πίνακα φαίνονται μόνο οι επιλογές που έχουν κέρδος 1 και επιτρέπουν την απλή μετατροπή των τάσεων από τα pins της θύρας PORTA.

MUX4..0	Single Ended Input
00000	ADC0
00001	ADC1
00010	ADC2
00011	ADC3
00100	ADC4
00101	ADC5
00110	ADC6
00111	ADC7

Καταχωρητής Ελέγχου και Κατάστασης του ADC (ADCSRA)

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
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	

ADEN: **ADC Enable**
Αν το bit είναι 1 ο ADC τίθεται σε κατάσταση λειτουργίας.

ADSC: **ADC Start Conversion**
Αν το bit γίνει 1 τότε ξεκινάει μετατροπή. Μόλις η μετατροπή ολοκληρωθεί το bit γίνεται 0.

ADIF: **ADC Interrupt Flag**
Γίνεται 1 όταν ολοκληρωθεί μια μετατροπή και ενημερωθούν οι καταχωρητές δεδομένων.

ADIE: ADC Interrupt Enable
Αν η τιμή του ADIE καθώς και του I-bit στον SREG είναι 1 τότε θα προκληθεί διακοπή ολοκλήρωσης μετατροπής του ADC.

ADPS2:0: ADC Prescaler Select Bits

Τα bit αυτά θέτουν ένα παράγοντα διαίρεσης μεταξύ του κεντρικού ρολογιού και της συχνότητας λειτουργίας του ADC όπως φαίνεται στον επόμενο πίνακα:

Table 85. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Καταχωρητές δεδομένων του ADC – ADCL και ADCH

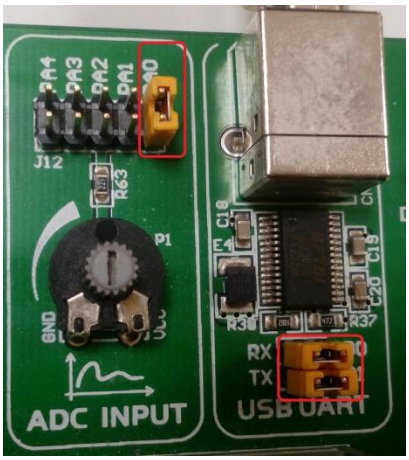
Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	

Τα δεδομένα μόλις ολοκληρωθεί μια μετατροπή αποθηκεύονται σε αυτούς τους δυο καταχωρητές. Πρώτα πρέπει να διαβαστεί ο ADCL και μετά ο ADCH.

Ρουτίνα αρχικοποίησης του ADC:

```
; Routine: usart_init
; Description:
; This routine initializes the
; ADC as shown below.
; ------- INITIALIZATIONS -------
;
; Vref: Vcc (5V for easyAVR6)
; Selected pin is A0
; ADC Interrupts are Enabled
; Prescaler is set as CK/128 = 62.5kHz
; --------------------------------
; parameters: None.
; return value: None.
; registers affected: r24
; routines called: None
ADC_init:
Ldi r24,(1<<REFS0)   ; Vref: Vcc
Out ADMUX,r24        ;MUX4:0= 00000 forA0.
;ADC is Enabled (ADEN=1)
;ADC Interrupts are Enabled (ADIE=1)
;SetPrescaler CK/128 = 62.5Khz (ADPS2:0=111)
ldi r24,(1<<ADEN)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)
out ADCSRA,r24
ret
```

Προκειμένου να λειτουργήσει σωστά η σειριακή επικοινωνία με τον υπολογιστή καθώς και να συνδεθεί το κατάλληλο ποτενσιόμετρο του συστήματος EasyAvr6 με το επιθυμητό pin του ATmega16 πρέπει να τοποθετηθούν τα 3 jumper όπως φαίνεται στην εικόνα 7.3. Το αριστερό jumper συνδέει τον μεσαίο ακροδέκτη του ποτενσιόμετρου με το pinA0 του ATmega16 έτσι ώστε να μετράται η τάση που μπορεί να μεταβληθεί στρέφοντας το ποτενσιόμετρο. Τα 2 δεξιά jumper συνδέουν τα TX, RX του ATmega16 με τα RX, TX (TX->RX) του μετατροπέα USB to UART της εταιρίας FTDI (ο οποίος μετατρέπει τα σήματα της θύρας USB του υπολογιστή σε σήματα UART). Θα χρειαστείτε 2 καλώδια USB αφού το ένα θα παρέχει τροφοδοσία και θα προγραμματίζει τον επεξεργαστή ενώ το άλλο θα αναλαμβάνει την σειριακή επικοινωνία. Για τον χειρισμό της σειριακής μπορείτε να χρησιμοποιήσετε το SerialMonitor (ή και το SerialPlotter για την άσκηση με τον ADC) του προγράμματος Arduino IDE ή όποιο άλλο πρόγραμμα επιθυμείτε.



Σχήμα 7.3.Το ποτενσιόμετρο του συστήματος EasyAvr6.

Τα ζητούμενα της 5ης εργαστηριακής άσκησης του AVR

Στα προγράμματα σας χρησιμοποιείστε χαρακτήρες αλλαγής σειράς '\n' ώστε να είναι τα αποτελέσματα σας ευανάγνωστα.

Ζήτημα 7.1

i) Να γραφεί πρόγραμμα σε assembly για τον ATmega16 στο σύστημα EasyAvr6 το οποίο να στέλνει στην UART ένα string (σύνολο χαρακτήρων που τερματίζεται με τον χαρακτήρα '\0') το οποίο θα έχετε αποθηκεύσει στην RAM στην αρχή του προγράμματος σας.

ii) Να γραφεί σε C πρόγραμμα για τον ATmega16 στο σύστημα EasyAvr6 το οποίο θα διαβάζει έναν αριθμό από 0 έως 8 από την UART και θα ανάβει το αντίστοιχο τάξης LED της PORTC. Θεωρήστε ότι το 0 σβήνει όλα τα LED. Το πρόγραμμα σας πρέπει να απαντάει κάθε φορά που διαβάζει έναν αριθμό με μήνυμα της μορφής "ReadX", όπου X ο αριθμός που στάλθηκε. Σε περίπτωση ανάγνωσης μη έγκυρου αριθμού να εκτυπώνεται μήνυμα "InvalidNumber". *Προσοχή: Από προεπιλογή όταν στέλνετε με το SerialMonitor του Arduino έναν χαρακτήρα αποστέλλεται και χαρακτήρας αλλαγής σειράς. Μπορείτε να το απενεργοποιήσετε από το μενού δεξιά κάτω στο παράθυρο της σειριακής.*

Ζήτημα 7.2

Στο πλαίσιο της άσκησης θα μελετηθούν δύο τρόποι χειρισμού του ADC. Ο πρώτος τρόπος θα είναι με την χρήση της διακοπής ολοκλήρωσης της μετατροπής του ADC. Η διακοπή αυτή μεταφέρει τον έλεγχο στην διεύθυνση **0x1C**, αν είναι ενεργοποιημένη αντίστοιχη διακοπή (από το bit ADIE του ADCSRA) καθώς και οι γενικές

διακοπές. Για να ξεκινήσει μια μετατροπή αρκεί να γραφεί 1 στο bit ADSC του καταχωρητή ADCSRA**.** Οδεύτερος τρόπος είναι το πρόγραμμα να αναμένει να ολοκληρωθεί η μετατροπή. Η αναμονή αυτή γίνεται ελέγχοντας το bit ADSC του ADCSRA το οποίο γίνεται 0 μόλις ολοκληρωθεί η μετατροπή.

i) Να γραφεί πρόγραμμα σε assembly για τον ATmega16 στο σύστημα EasyAvr6 το οποίο θα ξεκινάει μια μετατροπή του ADC και θα αυξάνει έναν μετρητή ο οποίος θα εμφανίζεται στα LED της PORTB κάθε 100ms. Η ανάγνωση των δεδομένων του ADC πρέπει να γίνεται μέσα στην ρουτίνα εξυπηρέτησης της διακοπής ολοκλήρωσης μετατροπής του ADC και τα δεδομένα αυτά πρέπει να μετατρέπονται σε τάση και να εκτυπώνονται στην UART με ακρίβεια ενός δεκαδικού ψηφίου (δεν χρειάζεται στρογγυλοποίηση). Η τάση δίνεται από τον τύπο:

$V_{IN} = \frac{ADC}{1024} V_{REF}$ όπου

V_{IN} η τάση στο pin A0
ADC η τιμή που διαβάζεται από τον ADC (αριθμός 10bit από 0-1023)
V_{REF} η τάση αναφοράς που με την δεδομένη ρουτίνα αρχικοποίησης έχει οριστεί σαν Vcc=5V.

ii) Να γραφεί σε C πρόγραμμα για τον ATmega16 στο σύστημα EasyAvr6 το οποίο θα ξεκινάει μια μετατροπή, θα περιμένει να ολοκληρωθεί η μετατροπή και θα εκτυπώνει την τάση στην UART με ακρίβεια ενός δεκαδικού ψηφίου (δεν χρειάζεται στρογγυλοποίηση).