Enginyeria de Sistemes TIC

Dispositius Programables

Pràctica 4 - Generador de Morse

Francisco del Águila López

Octubre 2012

Escola Politècnica Superior d'Enginyeria de Manresa
Universitat politècnica de Catalunya

1 Objectiu

L'objectiu d'aquesta pràctica és dissenyar un transmissor de Morse.

2 Consideracions prèvies

2.1 Components

Els elements que intervenen són:

1. Plataforma Arduino
2. 2 polsadors
3. Conjunt amplificador amb altaveus, per escoltar el senyal generat.
4. Placa *protoboard*
5. Sondes necessàries, per monitoritzar els diferents senyal al oscil·loscopi.
6. Trossos de cable prim per interconnectar la *protoboard* i l'Arduino.

Els polsadors i els altaveus es facilitaran en el moment de la pràctica.

2.2 Morse

Una descripció simple del codi Morse la teniu aprofitant la pràctica que heu fet a l'assignatura de Circuits i Sistemes Lineals. Essencialment podem recordar que el codi Morse internacional defineix cinc elements:

- El punt, que té una durada d'una unitat de temps. Això es pot simbolitzar amb 1.
- La ratlla, amb una durada de tres unitats de temps. Simbòlicament, 111, és a dir, llargada triple que el punt.
- La pausa entre elements d'un caràcter (lletra o nombre), amb una unitat de durada. El símbol pot ser 0.
- La pausa entre caràcters, que dura tres unitats de temps, i que té per representació 000.
- La pausa entre paraules, de set unitats de durada, 0000000.

2.3 Estructuració del programa

El disseny amb els dos polsadors es basa en que cada polsador s'encarrega de generar un *punt* o bé una *ratlla*. D'aquesta manera la seqüència de pulsacions en cadascun dels polsadors determinarà la seqüència de *punts* i *ratlles*.

Una possible estructuració del programa consistiria en una fase inicial d'inicialització dels registres o variables que ens calguin i posteriorment un bucle principal encarregat de detectar quin dels polsadors s'ha polsat. En funció del polsador s'hauria de fer una crida a la subrutina del *punt* o bé de la *ratlla*.

Si considerem com unitat temporal una durada de 150 ms, la subrutina del *punt* ha de consistir en la generació durant una unitat temporal d'un to de 1KHz i un silenci també d'una unitat temporal. Pel cas de la subrutina de la *ratlla*, s'hauria de generar un to igualment de 1KHz, però de durada 3 unitats temporals, seguit d'un silenci de 1 unitat temporal. Les durades dels silencis entre lletres i entre paraules (3 unitats i 7 unitats) ho deixarem a càrrec de l'operador que actua sobre els polsadors.

Un tros de codi *prac4_ex.S* que us pot ser útil per generar aquest programa seria:

```
.set DDRB_o , 0x4
.equ PORTB_o , 0x5
PORTD_o = 0x0b
DDRD_o = 0x0a
OCR0A_o = 0x27
TCCR0B_o = 0x25
TCCR0A_o = 0x24

.global main

waitabit:       ldi r19,41
wait3:  ldi r18,0xFF
wait2:  ldi r17,0xFF
wait1:  subi r17,0x01
        brne wait1
```

```
        subi  r18 ,0x01
        brne  wait2
        subi  r19 ,0x01
        brne  wait3
        ret

main:   ldi  r16 ,0b01000010
        out  TCCR0A_o, r16
        ldi  r16 ,0b00000011
        out  TCCR0B_o, r16
        ldi  r16 ,124
        out  OCR0A_o, r16
        ldi  r16 ,0b01000000
        out  DDRD_o, r16

loop:   call  waitabit
        ldi  r16 ,0b00000000
        out  DDRD_o, r16
        call  waitabit
        ldi  r16 ,0b01000000
        out  DDRD_o, r16
        rjmp  loop
```

Aquest codi utilitza el perifèric de *timer* 0 dels AVR. La informació detallada d'aquest perifèric es troba a l'apartat 15 pàg. 96 de [ATmega328p]. Aquest mòdul es pot configurar de moltes maneres. Té dos modes de generació PWM, un mode de comptador simple i un mode comparador.

Teniu en compte que us pot ajudar molt el fet de veure el resum de registres de entrada / sotitida que intervenen a la configuració de qualsevol perifèric. Aquest resum es troba al final del capítol on es descriu cada perifèric.

Aquest perifèric, com pràcticament la totalitat de la resta, també es pot fer servir mitjançant les interrupcions. De moment no farem cap utilització de les interrupcions, per tant, podeu excloure les explicacions relacionades amb les interrupcions, així com els registres del *timer* 0 associats a les interrupcions TIMSK0, TIFR0.

3 Estudi previ

1. Genera una subrutina **to1** que generi un to de 1KHz durant 150ms. Sense utilitzar el perifèric *timer* 0.
2. Genera una subrutina **sl1** que generi un silenci de 150ms. Sense utilitzar el perifèric *timer* 0.

3. Genera una subrutina **punt**, aprofitant les anteriors, que generi un senyal de *punt* de Morse. Sense utilitzar el perifèric *timer* 0.
4. Genera una subrutina **ratlla**, aprofitant les anteriors, que generi un senyal de *ratlla* de Morse. Sense utilitzar el perifèric *timer* 0.
5. Crea el programa generador Morse, definint quins seran els pins dels polsadors i quin serà el pin on es generarà el senyal de Morse. Anomena'l *prac4_1.S*. Sense utilitzar el perifèric *timer* 0.
6. Descriu detalladament (en format de comentari de programa) què fa exactament el codi *prac4_ex.S*
7. Modifica si creus convenient el programa del punt 5 aprofitant els recursos que es fan servir en el codi *prac4_ex.S*. Si cal, re-defineix els pins que es fan servir pels polsadors i pel to generat. Anomena'l *prac4_2.S*. En aquest cas es fa servir el *timer* 0.
8. Raona si és possible, com podries modificar el programa anterior per detectar en qualsevol instant que els dos polsadors estan polsats al mateix temps i per tant, es vol encendre el led de la placa Arduino mentre dura la doble pulsació. Anomena'l *prac4_3.S*.

4 Treball pràctic

El treball al laboratori consisteix en la comprovació pràctica de les tasques de l'estudi previ.

4.1 Assemblat del codi font

Per assemblar el nostre codi font s'ha de cridar a la comanda

```
avr-gcc -mmcu=atmega328p -o prova.elf prova.S
```

on el paràmetre **-mmcu** indica el model de microcontrolador que estem fent servir, **-o** indica el fitxer final generat (format *elf*) i l'últim paràmetre és directament el fitxer amb el codi font.

Un paràmetre que pot ser d'utilitat és **-E**, en aquest cas es realitza l'assemblat estrictament. Pot ser d'utilitat per analitzar possibles errors.

```
avr-gcc -mmcu=atmega328p -o prova.asm -E prova.S
```

Per convertir el format *elf* a *ihex*, la comanda és

```
avr-objcopy --output-target=ihex prova.elf prova.hex
```

Una possibilitat molt important és el procés de des-assemblat. En aquest cas, a partir d'un fitxer executable de tipus *elf* s'obté un fitxer en codi font. Òbviament la informació extra que conté el codi font original ha desaparegut. La comanda és

```
avr-objdump -S prova.elf > prova.disam
```

4.2 Transferència dels fitxers executables (Avrdude)

Quan es connecta l'Arduino al ordinador pel port USB, s'ha de comprovar que s'ha detectat el port de comunicació i per tant està reconegut pel sistema. Això es pot comprovar amb l'ordre *dmesg*. A les línies finals hauria d'aparèixer un nou dispositiu amb identificació **ttyACM0.**

Per comprovar que l'Arduino està operatiu i disposat a acceptar ordres, la comanda és

```
avrdude -c arduino -P /dev/ttyACM0 -p m328p
```

on els paràmetres indiquen que el tipus de programador que fa servir l'*avrdude* és el que inclou directament el kit Arduino, que el port de comunicacions és el nou port USB detectat i que el dispositiu amb el que el treballa és un ATmega328p. Aquests paràmetres es poden consultar en el manual *man avrdude*.

Per fer les transferències dels fitxers binaris es fa servir l'opció **-U**(consulteu el manual). Els paràmetres d'aquesta opció són la memòria a la que s'accedeix, si es fa lectura o escriptura, el fitxer que es vol transferir, el format del fitxer. Un exemple per llegit la memòria de programa i crear un fitxer Intel Hex amb el seu contingut seria

```
avrdude -c arduino -P /dev/ttyACM0 -p m328p -U flash:r:prova.hex:i
```

4.3 Tasques

1. Comprova el funcionament de *prac4_ex.S*.
2. Comprova el correcte funcionament de *prac4_1.S*.
3. Comprova el correcte funcionament de *prac4_2.S*.
4. Comprova el correcte funcionament de *prac4_3.S*.

Referències

[Ard] Arduino UNO - `http://arduino.cc/en/Main/ArduinoBoardUno`

[ATmega328p] Atmel. ATmega328P datasheet. `http://www.atmel.com/dyn/resources/prod_documents/doc8271.pdf`

[AtmAss] Atmel Assembler documentation - http://www.atmel.com/atmel/acrobat/doc1022.pdf

[Instr] Joc d'instruccions dels AVR - http://www.atmel.com/atmel/acrobat/doc0856.pdf

[AS] Manual de referència del Assemblador del GNU - http://sourceware.org/binutils/docs/as/

[ihex] Format de fitxer intel HEX - http://en.wikipedia.org/wiki/Intel_HEX

[Hardvard] Arquitectura de tipus Hardvard - http://en.wikipedia.org/wiki/Harvard_architecture

[Endian] Ordre de disposició dels bytes - http://en.wikipedia.org/wiki/Endianness