Enginyeria de Sistemes TIC

Dispositius Programables

Pràctica 9 - Transmissor morse

Francisco del Águila López

Novembre 2011
Escola Politècnica Superior d'Enginyeria de Manresa
Universitat politècnica de Catalunya

1 Objectiu

L'objectiu d'aquesta pràctica és realitzar un transmissor morse. Cada pulsació que es faci d'una lletra o número es transmetrà pel port serie. L'AVR buscarà en una taula com es tradueix el codi ASCII del caràcter rebut en el codi morse. Per una pota de sortida generarà els punts i les ratlles del caràcter rebut.

2 Material necessari

Els elements que intervenen són:

1. Plataforma Arduino
2. polsador
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.

3 Descripció del transmissor morse

Per realitzar aquest transmissor morse caldrà fer servir els següents perifèrics:

1. port serie
2. timer 2: generació del tó de 1kHz
3. timer 1: control de la durada del punt, la ratlla i el silenci

El mètode de generació del senyal morse segueix els següents passos:

1. Es rep un caràcter pel port serie. Aquest caràcter podrà ser una lletra en majúscules o minúscules, l'espai, els números del 0 al 9. Aquest caràcter està codificar en ASCII.
2. A partir d'una taula 2 d'equivalència de ASCII a morse, s'obté un altre byte on els 5 primers bits (major pes) simbolitzen els punts o les ratlles. Un punt queda codificat amb "0" i una ratlla queda codificada amb "1". Els 3 bits de menor pes s'utilitzen per indicar la mida del codi morse, es a dir, per indicar la llargada que tingui el codi. Això permet amb un únic byte codificar la longitud variable del codi morse.
3. Quan s'obté el codi morse que s'ha de transmetre s'inicia un bucle de durada la longitud del codi (segons els 3 bits de menor pes) que anirà generant un punt o una ratlla amb el seu silenci associat en funció dels 5 o menys bits de major pes.
4. El timer2 es configura per generar el to de 1 kHz en una de les potes de sortida OC2A. Aquesta pota s'anirà activant o desactivant en funció de si es vol generar un to o es vol generar un silenci.
5. La durada del to o silenci generat a la pota OC2A ve regulat pel timer0. A la taula 1 s'observa les diferents unitats temporals associades al morse.

Senyals	punt	ratlla	pausa	pausa caràcter	pausa paraula
Unitats de temps	1	3	1	3	7

Taula 1: Unitats temporals de morse

Amb la intenció de fer aquest transmissor el més simple possible i tenint en compte que no haurà de realitzar cap altra feina, no es farà servir cap tipus de interrupció. La justificació d'això és perquè la tasca de generació morse ve activada per la recepció d'un caràcter pel port serie. Un cop s'ha rebut aquest caràcter s'inicia un procés totalment pautat i que no requereix ser interromput per res, per tant totes les accions d'espera es faran per enquesta.

Caràcter	Morse	Codificació binària
A	·-	0b01000010
B	-···	0b10000100
C	-·-·	0b10100100
D	-··	0b10000011
E	·	0b00000001
F	··-·	0b00100100
G	- -·	0b11000011
H	····	0b00000100
I	··	0b00000010
J	·- - -	0b01110100
K	-·-	0b10100011
L	·-··	0b01000100
M	- -	0b11000010
N	-·	0b10000010
O	- - -	0b11100011
P	·- -·	0b01100100
Q	- -·-	0b11010100
R	·-·	0b01000011
S	···	0b00000011
T	-	0b10000001
U	··-	0b00100011
V	···-	0b00010100
W	·- -	0b01100011
X	-··-	0b10010100
Y	-·- -	0b10110100
Z	- -··	0b11000100
1	·- - - -	0b01111101
2	··- - -	0b00111101
3	···- -	0b00011101
4	····-	0b00001101
5	·····	0b00000101
6	-····	0b10000101
7	- -···	0b11000101
8	- - -··	0b11100101
9	- - - -·	0b11110101
0	- - - - -	0b11111101

Taula 2: Equivalència ASCII - morse

4 Estructuració del programa

La temporalització d'aquest programa estarà regulada pel timer 1. El motiu de fer servir el timer 1 és per tenir prou capacitat perquè pugui comptar temps de fins a 1s tenint en compte que el rellotge del sistema és de 16 MHz. Aquest timer té una precisió de 16 bits que serà suficient. Cada vegada que es vulgui esperar una estona es cridarà a una rutina anomenada *espera* que esperarà 1 unitat de temps. Si es vol esperar més unitats de temps es faran crides successives. A priori, les unitats de temps seran de 100 ms, però hauria d'estar en funció d'una variable definida en temps de compilació anomenada *u_temps*. Aquest timer es configurarà en mode de comparació, d'aquesta manera anirà comptant fins arribar al valor marcat pel comparador. Quan arribi al valor de comparació activarà un flag que ens indicarà que ha passat el temps desitjat. Aquest flag s'ha de consultar i esborrar cada vegada que es donar la condició d'igualtat.

Unes altres rutines que s'han de definir són *punt* i *ratlla*. Aquestes rutines s'encarregaran de activar la pota de sortida del timer 2, OC2A, per generar el to de 1 kHz de la durada que calgui i de generar un silenci de durada 1 unitat de temps. Aquesta rutina farà ús de *espera*.

La següent rutina a definir és *morse*. A aquesta rutina se li passa com paràmetre un byte pel registre r16 i s'encarregarà de generar la transmissió d'un caràcter morse segons la codificació de la taula 2. Al final de la transmissió, aquesta rutina ha de tenir en compte afegir un silenci de durada 2 unitats de temps amb la intenció de que s'obtingui un silenci total de 3 unitats, que és el que correspon al final de caràcter. Com suggeriments per a la seva implementació són:

1. Separar el byte de paràmetre en els 5 bits de major pes (codi) i els 3 bits de menor pes (llargada).
2. Fer bucle en funció de la llargada que consulti els bits de codi per transmetre punt o ratlla.
3. Testejar el bit de codi fent un desplaçament aprofitant el flag de carry.

Per últim, s'ha de definir una rutina (pot ser la de interrupció de recepció del port serie) que busqui a la taula d'equivalència entre ASCII i morse, el caràcter rebut pel port serie en ASCII i retorni el codi morse corresponent. En el cas que el caràcter rebut sigui un espai, no cal buscar res a la taula i en canvi s'ha de generar un silenci de 4 unitats de manera que juntament amb les 3 unitats corresponents al final de caràcter, s'obtinguin les 7 unitats corresponents al canvi de paraula.

Recordeu de preservar i restaurar, en cas necessari, tots aquells registres que s'utilitzin internament per les diferents rutines que es defineixen.

En cas de que s'utilitzi la rutina de interrupció de recepció del port serie, s'ha de garantir que el sistema no sigui interromput mentre s'executa la generació del senyal morse.

Teniu en compte aprofitar el codi corresponent a les pràctiques prèvies de morse, port serie i xifrador.

5 Guió de treball

1. Defineix la configuració necessària per fer servir el timer 2 per generar un to de 1kHz a la pota OC2A. Dissenya un programa que en funció d'una pota d'entrada activi o no la generació per aquesta pota. Anomena'l *prac9_ 1.S*.
2. Defineix la configuració necessària per fer servir el timer 1 per comptar temps. El mode de funcionament és de comparador. Per calcular el valor de comparació, defineix una formula (que es calcularà en temps de compilació) que calculi els valors de OCR1AH i OCR1AL en funció del paràmetre, *u_ temps*, expressat en ms. Dissenya la rutina *espera*. Comprova el bon funcionament del timer 1 i la rutina *espera* dissenyant un programa que encengui el led durant 1 unitat de temps i l'apagui durant 2 unitats de temps. Anomena'l *prac9_ 2.S*.
3. Dissenya les rutines *punt* i *ratlla* aprofitant els apartats anteriors. Comprova el seu bon funcionament dissenyant un programa que generi el missatge SOS de forma continuada. Anomena'l *prac9_ 3.S*.
4. Dissenya la rutina *morse* aprofitant els apartats anteriors. Comprova el seu bon funcionament dissenyant un programa que generi el codi morse rebut pel port serie. L'enviament es farà amb l'aplicació *moserial*. Anomena'l *prac9_ 4.S*.
5. Dissenya l'aplicació final. El que s'envia pel port serie és directament el codi ASCII, es busca l'equivalència a la taula morse i es genera el missatge. Si s'envia un espai s'ha de generar el silenci entre paraules. Anomena-la *prac9_ final.S*
6. *Opcional:* Comenta possibles canvis de disseny, ampliacions, etc. que consideris interessants.

6 Eines de treball

6.1 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
```

6.2 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
```

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

[USART] Dispositiu USART http://en.wikipedia.org/wiki/Universal_asynchronous_receiver/transmitter

[ASCII] Taula de caràcters ASCII http://en.wikipedia.org/wiki/ASCII