Enginyeria de Sistemes TIC

Dispositius Programables

Pràctica 10 - Receptor morse

Francisco del Águila López

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

1 Objectiu

L'objectiu d'aquesta pràctica és realitzar un receptor morse. Per una pota digital de l'AVR s'introduirà el senyal morse rebut. Aquest senyal ha d'estar prèviament pre-processat de manera que tan els punt com les ratlles siguin valors de 5 V i els silencis siguin valors de 0 V. Per aquest motiu es farà servir el circuit de detecció d'envolupant de les pràctiques de l'assignatura de Circuits i Sistemes Lineals. Cada vegada que l'AVR detecti una seqüència correcta de punts i ratlles corresponent a una lletra o número, el transmetrà pel port sèrie.

2 Material necessari

Els elements que intervenen són:

1. Plataforma Arduino
2. Circuiteria necessària per fer el detector d'envolupant.
3. Conjunt amplificador amb altaveus, per escoltar el senyal generat.
4. Placa *protoboard*
5. Sondes necessàries, per monitoritzar els diferents senyals a l'oscil·loscopi.
6. Trossos de cable prim per interconnectar la protoboard i l'Arduino.

3 Descripció del receptor morse

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

1. port serie
2. interrupció externa per pin INT0
3. Timer 1: control de la durada del punt, la ratlla i el silenci

En aquesta ocasió es farà servir el mòdul de interrupcions externes de l'AVR (apartat 13 de [ATmega328p]).

Les variables i constants que s'han de fer servir són les següents:

u_temps És la unitat de temps que determina la durada dels to o els silencis en milisegons. La relació entre punts, ratlles i silencis ve determinada per la taula 1.

llindar_L És el valor que tindria el Timer 1 amb el prescaler a 1024 i rellotge intern quan han passat 2 *u_ temps*. Aquesta constant es calcula en temps de compilació. Aquest valor és el que diferenciarà entre un punt i una ratlla (si estem a nivell alt, amb to) o bé diferenciarà entre una separació de símbol o caràcter (si estem a nivell baix, amb silenci).

llindar_H És el valor que tindria el Timer 1 amb prescaler a 1024 i rellotge intern quan han passat 5 *u_ temps*. Aquesta constant es calcula en temps de compilació. Aquest valor és el que diferenciarà entre una separació entre lletres o separació entre paraules. Aquest llindar només s'aplica al silenci.

rAA Registre on es van acumulant els punts o les ratlles del símbol que s'està rebent. Un punt queda codificat amb "0" i una ratlla queda codificada amb "1".

rBB Registre que compte el número de punts i ratlles que té un codi.

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

1. A la pota especial d'entrada a interrupció externa INT0 es rep el senyal morse després de la seva desmodulació amb un detector de envolupant (pràctica de Circuits i Sistemes Lineals).
2. Aquesta pota generalment estarà a nivell baix mentre no es reben codis morse.
3. Quan es rep un punt o una ratlla es crea un flanc de pujada que provoca una interrupció. Això provoca que s'activi i s'inicialitzi el Timer 1 per comptar el temps.
4. Quan s'acaba el punt o la ratlla es crea un flanc de baixada que igualment provoca una interrupció. En aquest moment es consulta el valor del Timer 1 per saber la durada. En funció de *llindar_ L* es decideix si ha sigut un punt o una ratlla. El Timer 1 es torna a inicialitzar a 0 perquè comenci a comptar de nou.
5. Si torna a produir-ser un flanc de pujada i per tant una nova interrupció, es consulta el valor del Timer 1 i comparant amb el *llindar_ L* i *llindar_ H* es decideix si s'ha

produït: un silenci de símbol, un silenci de caràcter o un silenci de paraula. En funció d'aquest fet: es continuarà el procés de rebre el caràcter actual, s'enviarà pel port serie el valor del caràcter rebut després de extreure'l de la taula 2 o bé s'enviarà el caràcter rebut i un espai en blanc per separar les paraules.

6. Si no es produeix el flanc de pujada i es genera la interrupció d'overflow del Timer1, vol dir que el silenci és molt llarg i per tant s'ha parat la recepció morse. En aquest cas, s'actuarà escrivint el caràcter rebut i un espai en blanc. Després es tornarà a l'estat inicial en el que tot està en repòs.

En funció d'aquesta descripció la manera d'implementar el receptor morse es basarà en una màquina d'estats on els events que poden provocar els canvis d'estat són:

1. Interrupció de flanc de pujada al pin INT0 de l'AVR
2. Interrupció de flanc de baixada al pin INT0 de l'AVR
3. Interrupció d'overflow del Timer 1

No es definiran més interrupcions, ja que el port serie es fa servir per transmetre i en aquest cas no cal fer servir interrupció.

Quan es detecta un punt o una ratlla s'ha de guardar aquest símbol en un registre rAA que actuarà com variable global. Un punt queda codificat amb "0" i una ratlla queda codificada amb "1". La introducció del "0" o "1" es farà fent desplaçament a l'esquerra de rAA. El registre rBB es farà servir per comptar el número de punts o ratlles que té el caràcter que s'està rebent.

Quan es detecta un silenci llarg, indicador de final de caràcter o paraula, s'ha de formar el byte que codifica el caràcter segons la taula 2 en funció de rAA i rBB. El byte final s'ha de trobar a rAA de manera que 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 i els 5 primers bits (major pes) simbolitzen els punts o les ratlles

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

Taula 1: Unitats temporals de morse

4 Implementació

Mireu-vos en detall l'apartat d'accés a registres de 16b (apartat 16.3 de [ATmega328p]). En especial per fer les lectures del valor de Timer 1 per després fer les comparacions.

Per definir la màquina d'estats, considereu tots els casos possibles. En particular, aquells que provoquen situacions d'error (ex. un to de durada molt llarga, superior a $llindar\_H$). En aquest cas, es pot fer alguna indicació del error (missatge per port sèrie, led, ...) i tornar a l'estat inicial esperant la bona recepció del següent codi.

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

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

Teniu en compte aprofitar el codi corresponent a les pràctiques prèvies.

5 Guió de treball

1. Dibuixa el diagrama d'estats que descriu el receptor morse, indicant els events que provoquen els canvis d'estat i les accions que es fan en cada moment.
2. Escull quins seran els registres rAA i rBB.
3. Dissenya la rutina *crea_ codi* de manera que a partir dels registres rAA i rBB retorni en el registre rAA el valor codificat de morse segons la tercera columna de la taula 2.
4. Dissenya la rutina *troba_ codi* que té com a paràmetre el valor codificat de morse segons la tercera columna de la taula 2 i es troba a r16. Ha de retornar al mateix registre r16 el valor de codi ASCII que representa.
5. Defineix la configuració necessària per fer servir la interrupció INT0. Dissenya un programa que en funció del flanc de pujada o baixada per la pota INT0 enviï "H" o "L" pel port sèrie. Anomena'l *prac10_ 1.S*.
6. Defineix la configuració necessària per fer servir el Timer 1 per comptar temps. El mode de funcionament és de comptador simple. Dissenya un programa que aprofiti la interrupció de flanc de pujada i baixada de INT0 per inicialitzar el Timer1 i que indiqui amb "+" o "-" si des de una interrupció a la següent a passat més temps "+" o menys temps "-" que el definit per la constant *u_ temps*. La interrupció d'overflow del Timer 1 s'ha de tenir en compte i enviarà per port serie una "O" en cas que es produeixi. Anomena'l *prac10_ 2.S*.
7. Dissenya un programa que implementi exclusivament la màquina d'estats del descodificador morse. Pel pin INT0 s'introduirà un senyal morse desmodulat. En funció del temps transcorregut entre flancs i si és to o silenci s'ha d'enviar pel port serie els següents caràcters: "." quan s'ha rebut un punt, "-" quan s'ha rebut una ratlla, "s" quan és un silenci entre símbols, "_" quan és un silenci entre caràcters, "P" quan és un silenci entre paraules, "E" quan es dona una situació d'error i per tant un valor no possible. Considereu la possibilitat d'activa el mode de funcionament DEBUGAR en funció de si està activa o no aquesta constant. En cas que s'activi el mode de DEBUGAR s'ha d'afegir com enviament per port sèrie els estats per on va passant el programa. Codifiqueu aquests estats de manera que no interfereixin amb els caràcters anteriors. Anomena'l *prac10_ 3.S*.
8. Dissenya el programa final que descodifiqui el morse. Anomena'l *prac10_ final.S*.
9. *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 llegir 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