Nume: Moisa Paula Grupa: 5403EA

Proiect Microcontrolere si Microprocesoare

Enunț: Să se conceapă si să se proiecteze , la nivel de program si de circuit, un sistem cu microcontroler de tip PIC care să genereze o melodie formată din minim 5 note muzicale cu frecvențe date în secvența f1, f2, f3, f4, f5… .

1. Introducere

Sunetul reprezinta vibratia unui corp sonor. Se obtine intotdeauna in urma unui impact intre doua sau mai multe corpuri si se manifesta sub forma de vibratii. Vibratiile ce pot fi percepure de urechea umana sunt cuprinse intre 30 si 20.000 vibratii simple pe secunda. Sunetele muzicale sunt limitate intre 32 si aproximativ 9000 vibratii simple pe secunda. Sunetul cel mai grav are 32 vibratii simple pe secunda. Sunetul cel mai inalt are aproximativ 9000 vibratii simple pe secunda. Sunetele muzicale au 7 denumiri: DO, RE, MI, FA, SOL, LA, SI.

Caracteristicile sunetului

Inaltimea

• caracteristica sunetului muzical de a fi mai grav (jos) sau mai acut (inalt).

• inaltimea sunetului muzical se calculeaza mereu fata de un punct de reper (nota) pe scara notelor muzicale.

Durata

• caracteristica sunetului muzical de a fi mai lung sau mai scurt in timp.

• durata se calculeaza din momentul impactului pana la disparitia ultimei vibratii percepute. .

Intensitatea

• caracteristica sunetului de a fi mai slab sau mai puternic. Se refera la frecventa notei (numarul de vibratii/secunda).

• de exemplu, nota LA de de-asupra notei DO de la mijloc in prezent este considerata ca avand 440 hertz (LA = 440 Hz) iar nota LA cu o octava sub DO este de 880 Hz.

Nume: Moisa Paula Grupa: 5403EA

Octavă este numit intervalul dintre două sunete, dintre care unul are frecvența dublă față de celălalt.

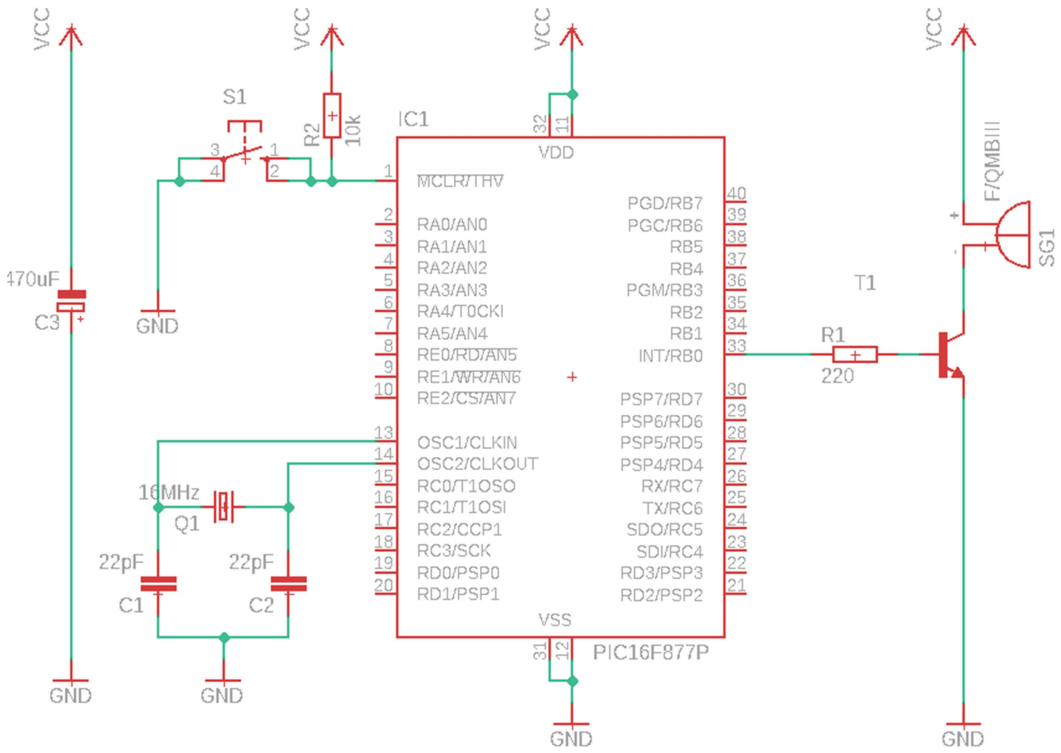
De exemplu în gama Do Major, prima notă DO (C) (numit și nota tonică sau nota rădăcină) are frecvența de 130,8 Hz iar octava (nota numărul opt din gamă) are frecvența de 261,6 Hz.[1]

Într-o gamă diatonică, octava cuprinde opt note consecutive.

Frecventele notelor muzicale din octava 5

Note muzicale	Frecvența	Perioada(us)
DO (C5)	523.25 Hz	1911 us
RE (D5)	587.33 Hz	1702 us
MI (E5)	659.25 Hz	1516 us
FA (F5)	698.46 Hz	1431 us
SOL (G5)	783.99 Hz	1275 us
LA (A5)	880.00 Hz	1136 us
SI (B5)	987.77 Hz	1012 us

2. Schema electrică



Nume: Moisa Paula Grupa: 5403EA

In implementarea acestui proiect am folosit microcontrolerul PIC16F877.
Acesta este alimentat la tensiunea de alimentare de 5V. Am folosit un oscilator cu quartz extern de 16MHz, deoarece microcontrolerul nu are un oscilator intern . Chiar daca oscilatorul are 16 MHz, circuitul va folosi doar 4MHz.

Pe pinul RB0 s-a conectat difuzorul(buzzerul) la care se aud cele 7 note muzicale.

C3- are rolul dea mentine constanta tensiunea de alimentare.

C1 si C2- filtreaza zgomotul de la cristalul de cuarț.

R2- tine pinul de reset in 1 logic pentru a nu se reseta (acesta se reseteaza doar la apasarea butonului S1).

Buzzer-ul porneste cand “avem” 5V pe tranzistorul T1.

3. **Pseudocod(descrierea algoritmului**)

```
Loop:
    {
      Buzzer = 1
      Wait_us(955)
      Buzzer = 0
      Wait_us(955)
      .
      .
      .
      Buzzer = 1
      Wait_us(506)
```

```
    Buzzer = 0
    Wait_us(506)
  }
```

4. Calcul timpi pentru delay:

f DO= 1911 us

1911/2=955us (pentru factor de umplere 50%)

Exemplu delay 955us:

```
DELAY_DO_955_us:
    MOVLW D'189'
    MOVWF i
    nop
    nop
loop_do:
        nop
        nop
        DECFSZ i, 1
        goto loop_do
        return
```

i*5 (2 nop+goto(2cm)+decfsz(1cm))-1 + 4(din bucla de delay)+ 5(din bucla de semnal: BSF + CALL+GOTO) = 955

⇨ i=189

5. Cod ASM

#include p16f877.inc

Nume: Moisa Paula Grupa: 5403EA

```
i equ 0x21

main:
    BSF STATUS,RP0
    MOVLW B'11111110'
    MOVWF TRISB; RB0 output
    BCF STATUS,RP0


semnal:
        BSF PORTB, 0
        call DELAY_DO_955_us
        BCF PORTB, 0
        call DELAY_DO_955_us
        BSF PORTB, 0
        call DELAY_RE_851_us
        BCF PORTB, 0
        call DELAY_RE_851_us
        BSF PORTB, 0
        call DELAY_MI_758_us
        BCF PORTB, 0
        call DELAY_MI_758_us
        BSF PORTB, 0
        call DELAY_FA_715_us
        BCF PORTB, 0
```

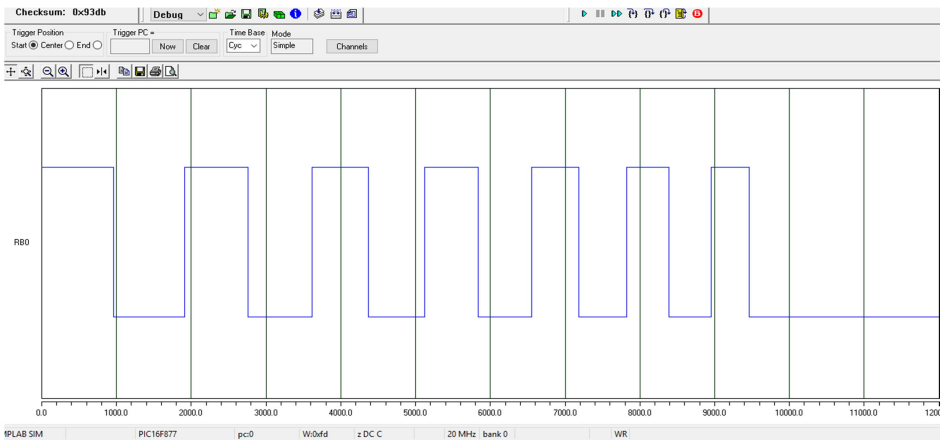
```
        call DELAY_FA_715_us
        BSF PORTB, 0
        call DELAY_SOL_637_us
        BCF PORTB, 0
        call DELAY_SOL_637_us
        BSF PORTB, 0
        call DELAY_LA_568_us
        BCF PORTB, 0
        call DELAY_LA_568_us
        BSF PORTB, 0
        call DELAY_SI_506_us
        BCF PORTB, 0
        call DELAY_SI_506_us
        GOTO end_loop
        nop

DELAY_DO_955_us:
    MOVLW D'189'
    MOVWF i
    nop
    nop
loop_do:
        nop
        nop
        DECFSZ i, 1
```

```
            goto loop_do
            return

DELAY_RE_851_us:
            MOVLW D'211'
            MOVWF i
            nop
loop_re:
            nop
            DECFSZ i, 1
            goto loop_re
            return
DELAY_MI_758_us:
            MOVLW D'150'
            MOVWF i
loop_mi:
            nop
            nop
            DECFSZ i, 1
            goto loop_mi
            return

DELAY_FA_715_us:
            MOVLW D'101'
            MOVWF i
```

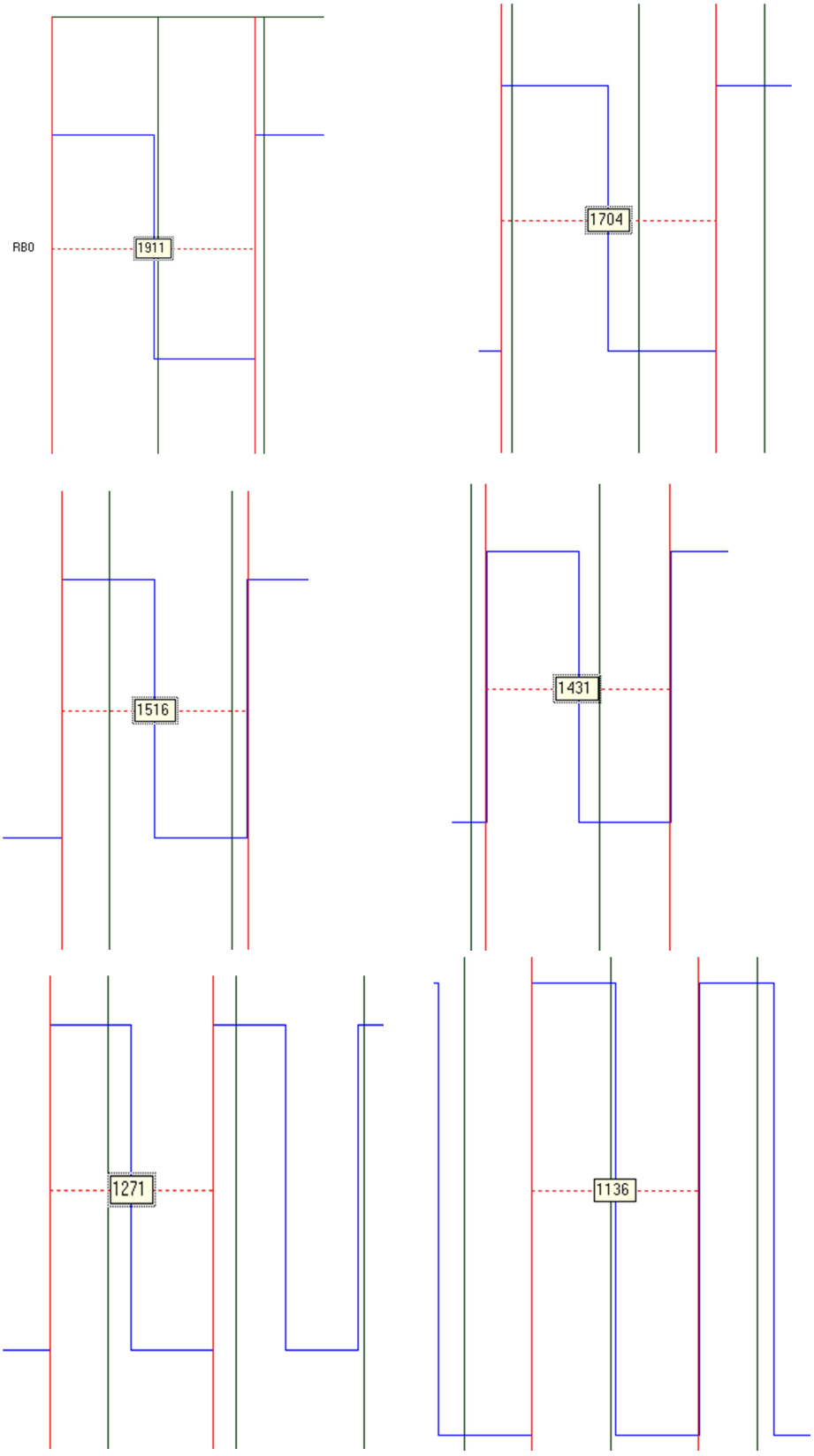
```
loop_fa:
        nop
        nop
        nop
        nop
        DECFSZ i, 1
        goto loop_fa
        return
DELAY_SOL_637_us:
        MOVLW D'157'
        MOVWF i
        nop
loop_sol:
        nop
        DECFSZ i, 1
        goto loop_sol
        return
DELAY_LA_568_us:
        MOVLW D'140'
        MOVWF i

loop_la:
        nop
        DECFSZ i, 1
        goto loop_la
```

```
        return

DELAY_SI_506_us:
        MOVLW D'166'
        MOVWF i
loop_si:
        DECFSZ i, 1
        goto loop_si
        return
end_loop:
        NOP
        end
```

6. Simulare cod ASM:



Nume: Moisa Paula Grupa: 5403EA



7. Cod in C#

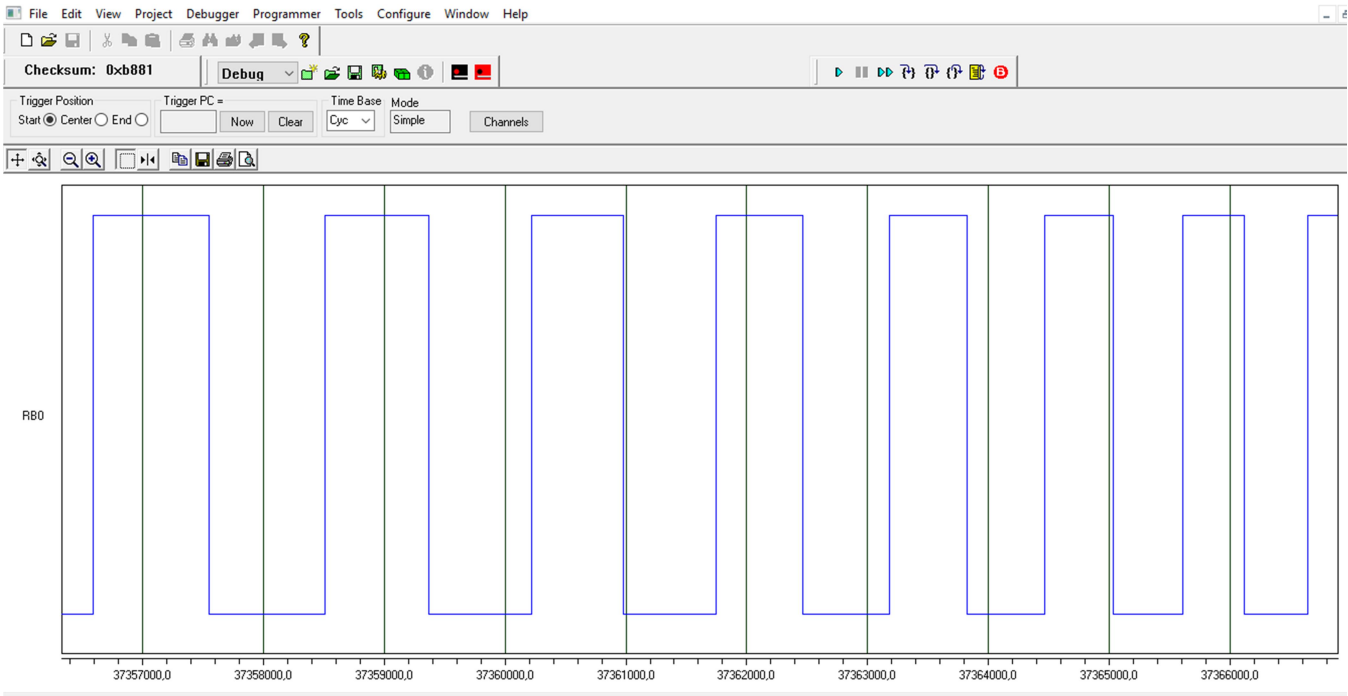
Nume: Moisa Paula Grupa: 5403EA

```
#include <htc.h>

#define _XTAL_FREQ 4000000
#define SET_RB0 PORTB = 1
#define CLEAR_RB0 PORTB = 0

#define DO_DELAY_US 1911
#define RE_DELAY_US 1702
#define MI_DELAY_US 1516
#define FA_DELAY_US 1431
#define SOL_DELAY_US 1275
#define LA_DELAY_US 1136
#define SI_DELAY_US 1012

void main(void)
{
	int n = 5;

	TRISB = 0b11111110; // RB0 output
	PORTB = 0;

	while(n --)
	{
		SET_RB0;
		__delay_us(DO_DELAY_US / 2);
		CLEAR_RB0;
		__delay_us(DO_DELAY_US / 2);
		SET_RB0;
```

```
            __delay_us(RE_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(RE_DELAY_US / 2);
            SET_RB0;
            __delay_us(MI_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(MI_DELAY_US / 2);
            SET_RB0;
            __delay_us(FA_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(FA_DELAY_US / 2);
            SET_RB0;
            __delay_us(SOL_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(SOL_DELAY_US / 2);
            SET_RB0;
            __delay_us(LA_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(LA_DELAY_US / 2);
            SET_RB0;
            __delay_us(SI_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(SI_DELAY_US / 2);
        }
        asm("NOP");
    }
```

8. **Simulare cod C :**

Nume: Moisa Paula Grupa: 5403EA



9. Bibliografie:

- file:///C:/Users/Paula/Downloads/Notiuni%20generale%20de%20teorie%20muzicala158.pdf
- http://pages.mtu.edu/~suits/notefreqs.html?fbclid=IwAR0o5-QrON-M8EHrN97sCkGRhRtck8xhHqq2K1yB5XWXYiTx10eVgq_8yX0
- https://ro.wikipedia.org/wiki/%C3%8En%C4%83l%C8%9Bimea_sunetelor

10. Concluzii

Generarea notelor muzicale a fost facuta prin generarea semnalului PWM cu ajutorul perioadelor notelor muzicale (T=1/frecventa) . Aceste perioade sunt folosite pentru a calcula timpul in care semnalul asociat unei note muzicale va sta in 1 logic , respectiv in 0 logic(50% din perioada unei note in “1” si 50% in “0”). Generarea melodiei s-a facut prin redarea notelor muzicale succesiv, intr-o bucla. Fizic , a fost folosit un buzzer(difuzor) pentru redarea sunetelor care functioneaza cu tensiunea de 5V.

Nume: Moisa Paula Grupa: 5403EA

```
#include <htc.h>

#define _XTAL_FREQ 4000000
#define SET_RB0 PORTB = 1
#define CLEAR_RB0 PORTB = 0

#define DO_DELAY_US 1911
#define RE_DELAY_US 1702
#define MI_DELAY_US 1516
#define FA_DELAY_US 1431
#define SOL_DELAY_US 1275
#define LA_DELAY_US 1136
#define SI_DELAY_US 1012

#define NOTE_LENGTH 550

void main(void)
{
	int n = 5;

	TRISB = 0b11111110; // RB0 output
	PORTB = 0;

	while(1)
	{
		for(n = 0; n < NOTE_LENGTH; n ++)
		{
			SET_RB0;
			__delay_us(DO_DELAY_US / 2);
			CLEAR_RB0;
			__delay_us(DO_DELAY_US / 2);
		}
		for(n = 0; n < NOTE_LENGTH; n ++)
		{
			SET_RB0;
			__delay_us(RE_DELAY_US / 2);
			CLEAR_RB0;
			__delay_us(RE_DELAY_US / 2);
		}
		for(n = 0; n < NOTE_LENGTH; n ++)
		{
			SET_RB0;
			__delay_us(MI_DELAY_US / 2);
			CLEAR_RB0;
			__delay_us(MI_DELAY_US / 2);
		}
		for(n = 0; n < NOTE_LENGTH; n ++)
		{
```

```
            SET_RB0;
            __delay_us(FA_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(FA_DELAY_US / 2);
    }
    for(n = 0; n < NOTE_LENGTH; n ++)
    {
            SET_RB0;
            __delay_us(SOL_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(SOL_DELAY_US / 2);
    }
    for(n = 0; n < NOTE_LENGTH; n ++)
    {
            SET_RB0;
            __delay_us(LA_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(LA_DELAY_US / 2);
    }
    for(n = 0; n < 2 * NOTE_LENGTH; n ++)
    {
            SET_RB0;
            __delay_us(SI_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(SI_DELAY_US / 2);
    }
    for(n = 0; n < NOTE_LENGTH; n ++)
    {
            SET_RB0;
            __delay_us(LA_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(LA_DELAY_US / 2);
    }
    for(n = 0; n < NOTE_LENGTH; n ++)
    {
            SET_RB0;
            __delay_us(SOL_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(SOL_DELAY_US / 2);
    }
    for(n = 0; n < NOTE_LENGTH; n ++)
    {
            SET_RB0;
            __delay_us(FA_DELAY_US / 2);
            CLEAR_RB0;
            __delay_us(FA_DELAY_US / 2);
    }
    for(n = 0; n < NOTE_LENGTH; n ++)
```

Nume: Moisa Paula Grupa: 5403EA

```
                {
                        SET_RB0;
                        __delay_us(MI_DELAY_US / 2);
                        CLEAR_RB0;
                        __delay_us(MI_DELAY_US / 2);
                }
                for(n = 0; n < NOTE_LENGTH; n ++)
                {
                        SET_RB0;
                        __delay_us(RE_DELAY_US / 2);
                        CLEAR_RB0;
                        __delay_us(RE_DELAY_US / 2);
                }
                for(n = 0; n < 2 * NOTE_LENGTH; n ++)
                {
                        SET_RB0;
                        __delay_us(DO_DELAY_US / 2);
                        CLEAR_RB0;
                        __delay_us(DO_DELAY_US / 2);
                }
                __delay_ms(2000);
        }
}
```