# Virtueel haardvuur

## Waar geen rook is, moet elektronica zijn

**Piero Di Stefano**
(Canada)

Een zacht brandend, knetterend haardvuur vindt iedereen mooi, maar niet iedereen kan zich een open haard in de woonkamer veroorloven. Daarom presenteren we hier een alternatief met embedded elektronica en mp3, maar zonder rook, stank, rondvliegende sintels en (buiten in de kou) houthakken. Deze schakeling produceert niet veel warmte, maar ze is gemakkelijk aan te steken. Dat is een zorg minder bij het voorbereiden van een gezellige avond.

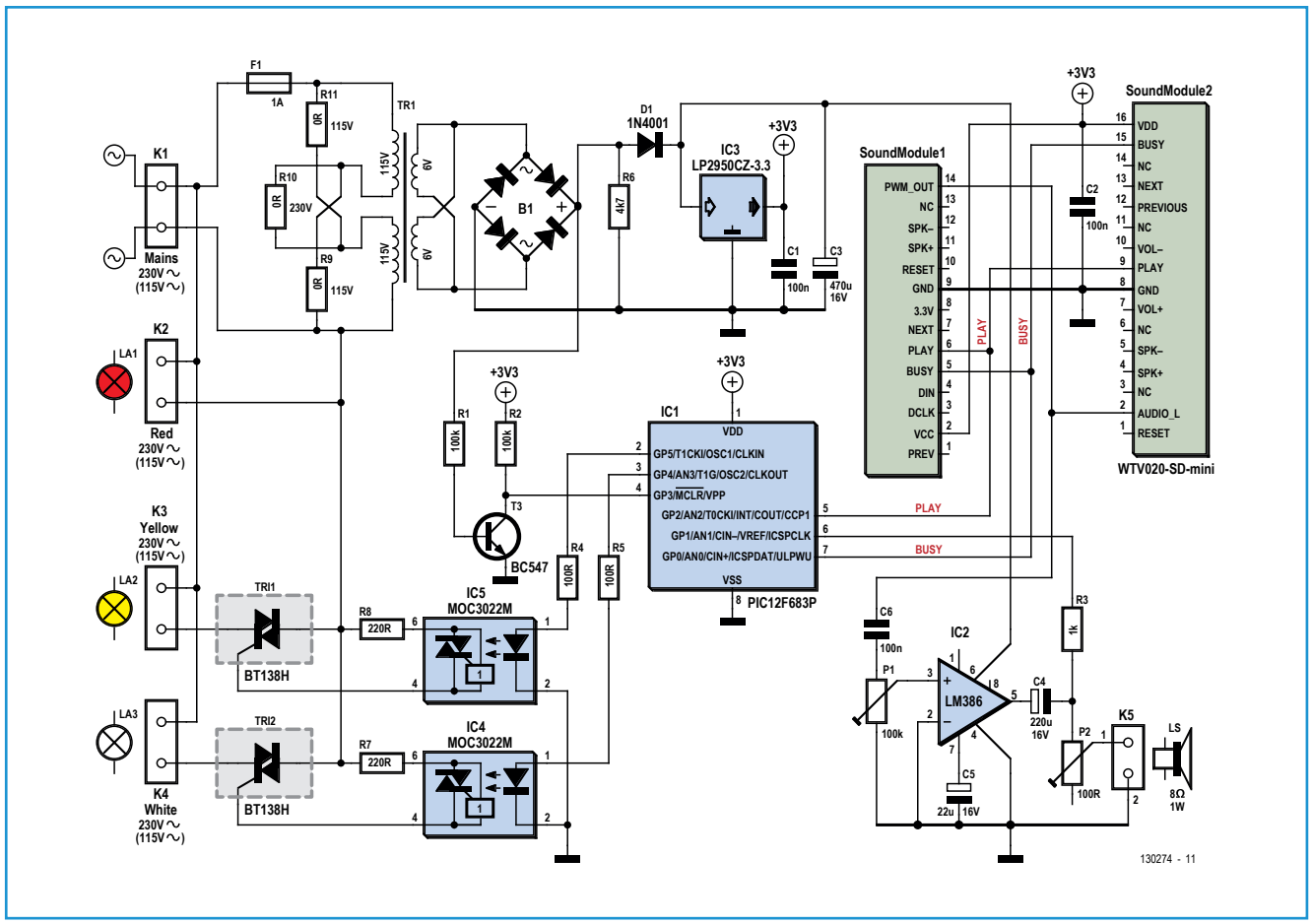
Dit apparaat om een open haard te simuleren komt net op tijd voor de gezellige, lange winteravonden. Het werkt met gewone 230-V-gloeilampen (of andere dimbare lampen). Een rode lamp zorgt voor een diep rode achtergrondkleur (die wordt niet gedimd). Daarnaast zijn er één gele en één witte lamp (of desgewenst een andere kleur) die knipperen in het ritme van het gekraak en gesputter van het virtuele vuur. Het apparaat heeft een kleine (maar krachtige) versterker die het geluid van een echte open haard weergeeft.

Het MP3-geluidsbestand is te downloaden van [1]. Als dit geluid niet bevalt, zijn er op YouTube nog massa's alternatieven te vinden.

### De werking

Het schema is te zien in **figuur 1**. Het audiobestand wordt opgeslagen op een geheugenkaart en weergegeven met een mp3-geluidsmodule, in het schema aangeduid met SoundModule1 of 2. De auteur heeft de Tenda mp3-speler gebruikt, die wordt aangesloten als 'SoundModule1'. In het Elektor-lab is geko-

zen voor een WTV020-SD-mini MP3-speler, deze kan worden aangesloten als 'SoundModule2'. SoundModule1 en SoundModule2 hebben verschillende aansluitingen; daar is op de print ruimte voor gereserveerd.
Na het booten vraagt de microcontroller de MP3-module het eerste bestand af te spelen. De BUSY-lijn (pen 15) wordt dan LAAG tot het afspelen klaar is. Na het afspelen wordt de BUSY-lijn even hoog, waarna het bestand opnieuw wordt afgespeeld. De Microchip 12F683 (IC1) stuurt telkens opnieuw het Play-commando (eigenlijk gewoon een puls) naar de MP3-module. Hij kan aan het niveau op input GP0 zien wanneer het muziekbestand beëindigd is. De 12F683 zit in een simpele DIP-8-behuizing.
Het flikkeren van de lampen regelt de microcontroller via twee triacs. De rode lamp is rechtstreeks verbonden met het lichtnet, de gele en de witte lamp worden bestuurd door de microcontroller via de triacs Tri1 en Tri2 en de optocouplers (nauwkeuriger omschreven: optisch geïsoleerde triac-drivers) IC4 en IC5. Een oude vertrouwde LM386 werkt als audioversterker en stuurt een 8 ohm-luidspreker aan. Het uitgangsvermogen is ongeveer 500 mW. Instelpotmeter P1 dient als volumeregelaar. De andere instelpotmeter, P2, lijkt misschien raar, maar hij heeft wel een functie. Zie het commentaar van de auteur in de mikroC-broncode.
De voeding is opgezet met een simpele 3,3 V-spanningsregelaar, IC3. De ongestabiliseerde voedingsspanning van ongeveer 8 volt wordt alleen gebruikt voor het voeden van de audioversterker. Alle andere chips werken op 3,3 volt.
De microcontroller weet wanneer er een nuldoorgang in de netspanning is dankzij transistor T3. Deze tijdinformatie is nodig om de triacs op het juiste moment open te sturen. Zo kan de helderheid van de lamp worden geregeld met behulp van fase-aansnijding, waarbij IC4 en IC5 er voor zorgen dat de elektronica galvanisch gescheiden is van de netspanning.




Figuur 1.
De virtuele haard is de moderne vervanger van houthakken, vuur maken, rook, vochtige kranten en het frustrerende wachten op een goed vuurtje. Veel warmte geeft deze echter niet.

De componentenwaarden en de aansluiting van de primaire wikkeling van de transformator in het schema zijn voor een netspanning van 230 VAC bij 50 Hz (voor 115 V moeten de draadbruggen R9 en R11 worden geplaatst in plaats van R10 en moet zekering F1 vervangen worden door een type van 2 AT; R7 en R8 worden in dat geval 100 Ω).

### Opbouw

Het Elektor-lab heeft voor deze schakeling een ruim opgezette, elektrisch veilige print ontworpen (zie **figuur 2**). Op de print is ook plaats voor de voedingstransformator van 2 x 6 V. Met behulp van de onderdelenlijst en de foto's in dit artikel zou het bouwen van dit project geen problemen moeten opleveren. Er zijn geen SMD-componenten of andere miniatuur-onderdelen aanwezig. Het resultaat is te zien in **figuur 3** en **figuur 4**.

**LET OP:** Verschillende punten in de schakeling zijn verbonden met het lichtnet. Daarom mogen **de print, de bedrading en de aangesloten componenten nooit worden aangeraakt als de schakeling in werking is**. Bovendien MOET de print worden ingebouwd in een goedgekeurde, niet-metalen behuizing. Neem alle voorschriften en voorzorgsmaatregelen voor elektrische veiligheid in acht.

## Onderdelenlijst

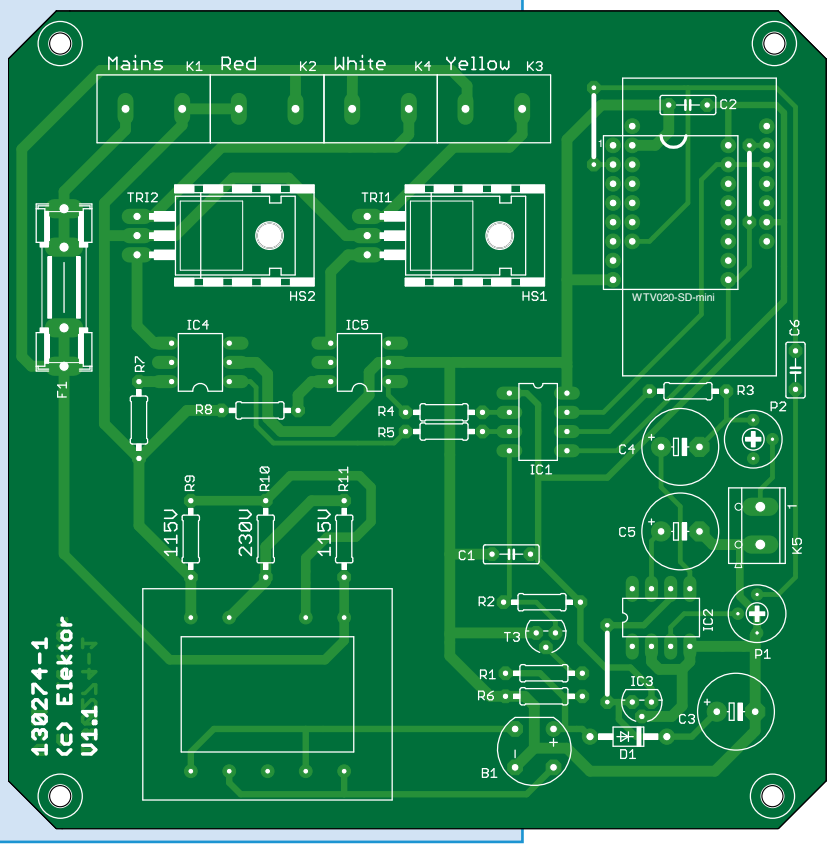
**Weerstanden:**
(0,25 W tenzij anders vermeld)
R1,R2 = 100 k
R3 = 1 k
R4,R5 = 100 Ω
R6 = 4k7
R7,R8 = 220 Ω
R9,R11 = niet aanwezig voor 230 V
R10 = draadbrug voor 230 V
P1 = instelpotmeter 100 k
P2 = instelpotmeter 100 Ω/0,5 W

**Condensatoren:**
C1,C2,C6 = 100 n
C3 = 470 µ/16 V
C4 = 220 µ/16 V
C5 = 22 µ/16 V

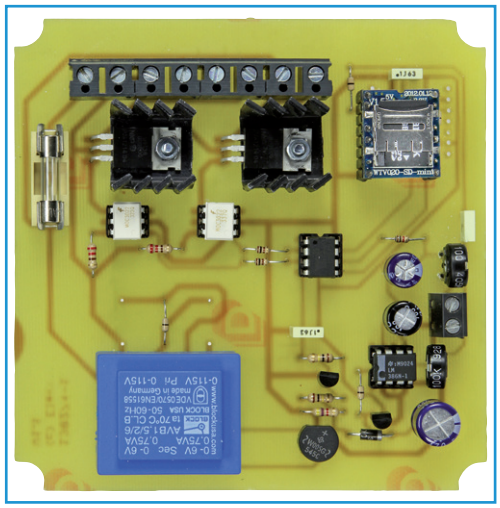
**Halfgeleiders:**
B1 = bruggelijkrichter, bijv. Vishay type 2W005G-E4/51 (Farnell-nr. 1497581)
D1 = 1N4001
IC1 = PIC12F683P-I/P (geprogrammeerd, nr. 130274-41)
IC2 = LM386N-1
IC3 = LP2950CZ-3.3/NOPB
IC4,IC5 = MOC3022M
T3 = BC547
TRI1,TRI2 = BT138-800

**Diversen**
F1 = zekering, 1AT
HS1, HS2 = koellichaam, 21 K/W, Fischer Technik type FK 230 SA L1 (Farnell-nr. 1892318)
K1,K2,K3,K4 = 2-polige printkroonsteen, steek 7,5 mm
K5 = 2-polige printkroonsteen, steek 5 mm
SoundModule1 = Tenda MP3-speler *
SoundModule2 = WTV020-SD-mini MP3-speler *
DIP6-voetje voor IC4 en IC5
DIP8-voetje voor IC1 en IC2
TR1 = voedingstrafo voor printmontage, 2 x 115 V prim./2 x 6 V sec., Block type AVB1.5/2/6 (Farnell-nr. 1131474)
Zekeringhouder voor printmontage, met kap
Behuizing, bijv. Hammond type 1591USBK (Farnell-nr. 1426582)
LS = miniatuur luidspreker 8 Ω/1 W
Print nr. 130274-1, zie [1]

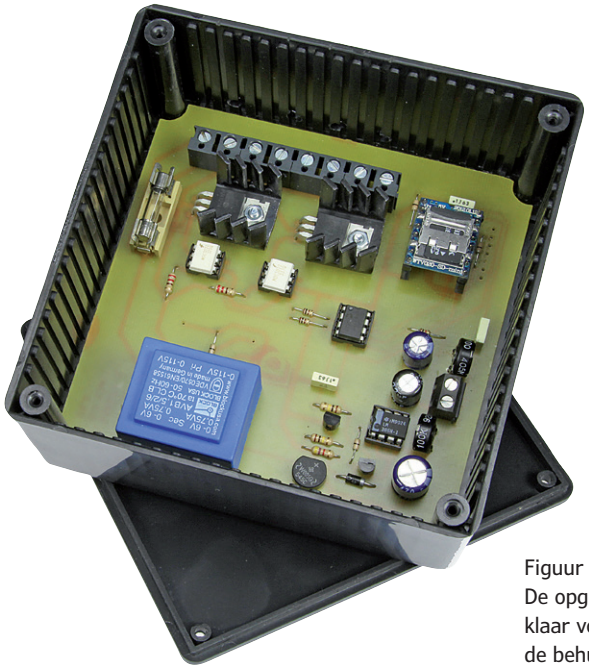
* Gebruik òf SoundModule1 òf SoundModule2




Figuur 2.
In verband met de elektrische veiligheid is het verstandig de virtuele stookplaats op te bouwen op deze print.



Figuur 3. Opgebouwde en geteste print.



Figuur 4.
De opgebouwde print is klaar voor het monteren in de behuizing.

### Gratis software

De firmware is geschreven met de gratis mikroC-versie van mikroElektronika en deze is minder dan 2 kilobyte groot (eigenlijk is de code maar 200 woorden lang). Het mikroC-programma is te zien in **listing 1** aan het einde van dit artikel. De code is gratis te downloaden van [1]. De inhoud van TMR0 moet overeenkomen met de lichtnetfrequentie (50 Hz of 60 Hz), dus bekijk de listing zorgvuldig (aangezien de Elektor-artikelen in veel landen en talen verschijnen, zijn beide mogelijkheden ingebouwd).
Als alle microcontroller-files op hun plaats staan en alles is begrepen, besproken, gedebugd en gecompileerd… moeten we nog letten op de onvermijdelijke instelling van de fuses van de PIC. Voor het gemak zijn deze weergegeven in **figuur 5**. Vervolgens kan de code in de PIC worden gebrand.

### De schakeling in actie, ook op YouTube

De virtuele haard van de auteur is in actie te zien op YouTube [2]. Het Elektor-lab heeft er ook één gebouwd en het resultaat van deze inspanningen is te zien op de foto's in dit artikel. Daarvan is ook een video gemaakt, waarin de werking wordt uitgelegd [3].
Hoewel de antieke stookplaatsen in verschillende kamers van het Elektor House (gebouwd in het midden van de 17e eeuw) enkele tientallen jaren geleden professioneel zijn gerestaureerd, zijn de bijbehorende schoorstenen dichtgemaakt. Dus krijgen we hier niet meer de kans om te genieten van een echt vuur met hout, rook, sterke drank en dat soort dingen. Toch kunnen we nu een mooie kerstsfeer creëren, want we hebben een paar PIC12F683's 'gebrand' in plaats van houtblokken.

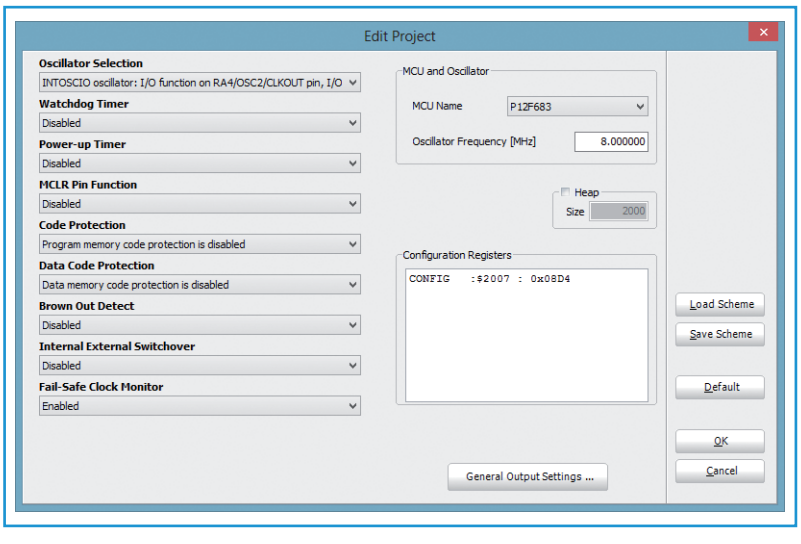
(130274)

### Weblinks


[1] MP3-bestand, software en print-layout: www.elektor-magazine.nl/130274, bestand: 130274-11.zip

[2] Virtuele haard van de auteur op YouTube: http://youtu.be/xK7Jj2aXOXI

[3] Virtuele haard van Elektor op YouTube: http://youtu.be/L_4yy8giTHk


Figuur 5.
Dit zijn de instellingen van de fuses voor het programmeren van de PIC.

## Listing 1. Virtuele haard mikroC-broncode.

```
/*
 * Project name:
     PIC12F683 Electronic fireplace
 * Piero Di Stefano, MAY 8th 2013.
   The triacs are driven generating a delay that is inversely proportional to the sound level.
  a 60Hz half-wave takes 1/30= 16.66ms to complete, so we have to handle any time between zero and
      8333us.
  Here we go up to 8160, then above such a value, we keep the triac triggered. It's impossible to
      notice it.

  At 50Hz the timespan will be longer, 10ms, so we should set TMR0 prescaler to 128. That means
      that TMR0 interrupt will fire every
  128*255/2000000 = 16320us

  So first thing, in the main() remember to change the following statement:
  OPTION_REG = 0b10000101;  //TMR0 prescaler: 1/64

  with
  OPTION_REG = 0b10000110;  //TMR0 prescaler: 1/128

  In order to limit its range between 0 and 10000us, we must use an init value for TMR0 of:
      255*10000/16320 = 156.25
  Obviously we must use a smaller value, so no less than 156
  Countercheck:

  TMR0 = 156
  Tmr0 interrupt will fire every (156)*128/2000000 =  9984us

  Last important step to be taken:
  inside the TMROIF routine, we must change the following line:

  TMR0 = ADC_reading << 2;

  that prepares TMR0.
  Since we must use a minimum value of 156 for TMR0 to properly trigger the thyristor at 50Hz, we
      must limit our range to 255 -156 = 99.
  I suggest to limit the ADC reading to 127, shifting its value 3 times instead of just 2 and
      adding no more than 29 to it

  TMR0 = (ADC_reading << 3) + 29;

  A smaller value would give us more allowance

  So the problem should be solved.


 * Test configuration:
```

```
     MCU:             PIC12F683
     Dev.Board:       EasyPIC6
     Oscillator:      Internal, 8MHz
     Ext. Modules:    none

     SW:              mikroC PRO for PIC
                      http://www.mikroe.com/eng/products/view/7/mikroc-pro-for-pic/
 * NOTES:
 Tenda mp3 mplayback module:
http://www.google.ca/#hl=en&sclient=psy-ab&q=tenda+electronics+tdb380&oq=tenda+tdb&gs_l=hp
      .3.1.0j0i22i30.848.5801.0.8070.9.9.0.0.0.0.120.811.6j3.9.0...0.0...1c.1.14.psy-ab.Cd-
      q3yzoZcI&pbx=1&bav=on.2,or.r_qf.&bvm=bv.46751780,d.dmQ&fp=ca03a074a5a6f34d&biw=1017&bih=596
*/

     unsigned short dummy;
     unsigned int ADC_reading;

void interrupt()
{
   if(INTCON.T0IF)
   {
     GPIO.F5 = 1;          //past some time between 0 and 8160 us, brings LOW the output
     INTCON.T0IE = 0;      //disables TMR0 IRQ preventin it to fire until the next rising edge
      (GPIF)
     INTCON.T0IF = 0;      //clear bit
   }

   if(INTCON.GPIF)
   {
     dummy = GPIO;
     /*IMPORTANT, this is the core of the program
     First, we multiply ADC_reading by 4 to compensate for the low voltage level from the
      loudspeaker.
     Can't always amplify the mp3 file level, otherwise it will sound distorted. So keep the volume
      down and change the voltage threshold
     Note that the louder the volume, the higher the number we get from the ADC, the higher value
      TMR0 will be initialized at.
     So we get a smaller delay on triggering the triac and the light will appear brighter.
     */
     TMR0 = ADC_reading << 3 + 29;
     //If the sound level is all the way up to the max (at 245/255 to be precise), disable TMR0 and
      set GP5 HIGH permanently,
     //it will keep the triac triggered for the whole cycle
     if (TMR0 > 245)
     {
       INTCON.T0IE = 0;
       GPIO.F5 = 1;
     } else
       {
```

```
         delay_us(160);       //fine adjust zero-crossing synchronism, to compensate the early
      saturation of the bjt and trigger the triac precisely
         INTCON.T0IE = 1;
         GPIO.F5 = 0;         //triac gate LOW  NOTE: GP2 is LOW to begin with, then goes high.
       }

     INTCON.GPIF = 0;
   }

}


void main() {

OSCCON = 0b01110001;      //Enables 8MHz int osc  (we must SUPERSEDE Project prop. dialog box in
      order to get it to work)

OPTION_REG = 0b10000110;  //TMR0 prescaler: 1/128
INTCON = 0b10001000;      //general irq enabled  + GPIF (TMR0 enabled programmatically)
ANSEL = 0x02;             //AN1 enabled
CMCON0 = 0x07;            //disables comparators
IOC.IOC3 = 1;             //enables RBINT on GPIO.F0

trisIO = 0b00001011;      //GP0 digital input (~BUSY) to replay the file, GP1 = analog input from
      audio player, GP3 digital input from ZCD
GPIO = 0b00000100;        //GP2 must should start high.
TMR0 = 156;               //Init value


    // play mp3 file at startup
    GPIO.F2 = 1;
    delay_ms(100);
    GPIO.F2 = 0;
    delay_ms(100);



 while(1)
 {
    ADC_reading = ADC_Read(1);

    //White light bulb: when the sound level is higher, it goes on, and stays that way until next
      reading.
    if (ADC_reading > 1000) GPIO.F4 = 1; else GPIO.F4 = 0;


    if (GPIO.F0)     //F0 = HIGH only when device is in IDLE mode, goes LOW when playing
    {
      delay_ms(500);
```

```
      GPIO.F2 = 1;
      delay_ms(100);
      GPIO.F2 = 0;
      delay_ms(100);
    }

 }//end while

}// end main
```