

```

/**
 * Simple rotary encoder tuned, I2C 1602 LCD, AD9850
 * V 1.0.0 ND6T 22 December 2017
 * This source file is under General Public License version 3.0
 *
 * Pin Connections:
 *   Display: SDA = A4, SCL = A5
 *   Encoder: A = D2, B = D3, switch = D4
 *   AD9850:CLK = D9, FU = D10, DAT = D11, RST = D12
 */
#include <Rotary.h>          //Available at:https://github.com/brianlow/Rotary
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7); // 0x27 is the I2C bus address for an
unmodified module
// On 1602;P2=EN (6),P1=R/W (5),P0=RS (4),P4-P7=D4-D7 (11-14);

Rotary r = Rotary(2,3); //Encoder to pins 2,3
unsigned char result; //ISR results
int ind = 4;          //Tuning position indicator
int oldind=4;        //Previous tuning indicator
int x;               //General purpose variable
long incr = 1000;    //Initial tuning increment
float ref = 124997180; // My ref clock
long upperLimit = 50e6; //Upper frequency limit
long lowerLimit = 1e6; //Lower frequency limit
long freq;          //DDS programming variable
long oldFQ;        //Frequency change reference
long FQ = 7.2e6;    //Starting frequency

//Prep for AD9850
#define CLK 9      //clock pin
#define FU 10     //freq update pin
#define DAT 11    //DATA pin
#define RST 12   //Reset pin
#define pulse(pin) {digitalWrite(pin, HIGH); digitalWrite(pin, LOW); }

void setup(){
  PCICR |= (1 << PCIE2); //Interrupt setup
  PCMSK2 |= (1 << PCINT18) | (1 << PCINT19); //Matrix "state machine" decode

  lcd.setBacklightPin(3,POSITIVE);
  lcd.setBacklight(HIGH);
  lcd.begin(16, 2);

  pinMode(CLK, OUTPUT);
  pinMode(FU, OUTPUT);
  pinMode(DAT, OUTPUT);
  pinMode(RST, OUTPUT);
  pulse(RST);
  pulse(CLK);
  pulse(FU); // Enables AD9850 serial mode
  pinMode(4,INPUT_PULLUP); //Tuning increment switch

  lcd.setCursor(0,0); ///////////////Splash//////////////////
  lcd.print("basic");
  lcd.setCursor(0,1);
  lcd.print("version 1.0.0");
  delay(2000);
}

```

```

void loop(){
  ind=int((log10(FQ)))-(log10(incr)); //Calculate indicator position

  if(ind<0)incr=1000000;
  if(oldind!=ind)show();
  oldind=ind;

  if(FQ!=oldFQ){ //If frequency changed then reprogram
    oldFQ=FQ; //Reset reference
    program(); //Re-program the DDS
  }
}
//*****FUNCTIONS (subroutines)*****
void show(){ //Display routine
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print(FQ);

  lcd.setCursor(ind,0); //Indicator position
  lcd.cursor(); //Tuning increment indicator
}

void program() { //Program the AD9850
  freq=FQ*pow(2,32)/ref;
  for(x=0;x<4;x++,freq >=>8)shiftOut(DAT,CLK,LSBFIRST,freq);
  shiftOut(DAT,CLK,LSBFIRST,0);
  pulse(FU);
  show();
}

ISR(PCINT2_vect) { //Interrupt service routine
  result = r.process();
  if(digitalRead(4)==HIGH){ //If tuning knob is not pressed
    if(result == DIR_CW){
      FQ+=incr; //Clockwise. Add the increment
      if(FQ>upperLimit)FQ=upperLimit; //Unless it exceeds upper limit
    }
    if(result == DIR_CCW){
      FQ-=incr; //CounterClockwise subtract it.
      if(FQ<lowerLimit)FQ=lowerLimit; //Unless it is less than lower limit
    }
  }
  else{ //If the tuning knob is pressed then move the cursor
    if(result == DIR_CW){ //Move cursor right
      incr=incr/10;
      if(incr<1)incr=1; //Lower limit
    }
    if(result == DIR_CCW){ //Move cursor left
      incr=incr*10;
      if((log10(incr))>(log10(FQ)))incr=incr/10; //Upper limit
    }
  }
}
}

```