

```

/**
 * Simple rotary encoder tuned, I2C SSD1306, AD9850
 * Full 3-digit parsing. 1 MHz to 50 MHz
 * V 1.0.0 ND6T 23 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 <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

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 = 124997110; //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

  display.begin(0x3C); // initialize with the OLED I2C addr

  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

  ////////////Splash//////////
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  display.print("EZoled v 1.0.0");
  display.display();
  delay(2000);
}

```

```

void loop(){
  ind=int((log10(FQ)))-(log10(incr))+2;//Calculate indicator position
  if(FQ<1e7&&FQ>(1e7-13))ind=int((log10(FQ)))-(log10(incr))+1;//Math patch
  if(incr>100)ind-=1;          //Compensate for decimal places
  if(incr>100000)ind-=1;
  if(ind<0)incr=10000000;
  if(olddind!=ind)show();    //Display if cursor is to be moved
  olddind=ind;

  if(FQ!=oldFQ){ //If frequency changed then reprogram
    oldFQ=FQ; //Reset reference
    program(); //Re-program the DDS
  }
}
//*****FUNCTIONS (subroutines)*****
void show() { //OLED display
  int mhz=(FQ/1e6);          //Truncate MHz
  long khz=FQ-(mhz*1000000);
  if(khz==( -1)){khz=999999; //Math patch
  mhz=mhz-1;}
  if(khz==( -2)){khz=999998; //Math patch
  mhz=mhz-1;}
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  display.print(mhz);          //Parse at each 3 digits for easy reading
  display.print(".");
  if((khz/1e3)<10) display.print("0"); //Insert leading zeros if necessary
  if((khz/1e3)<100) display.print("0");
  display.print((khz)/1e3,3); //Parse final 6 digits

  display.setTextSize(2);
  display.setCursor(ind*12,9);
  display.print("-");
  display.display();
}

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
}
}
}
```