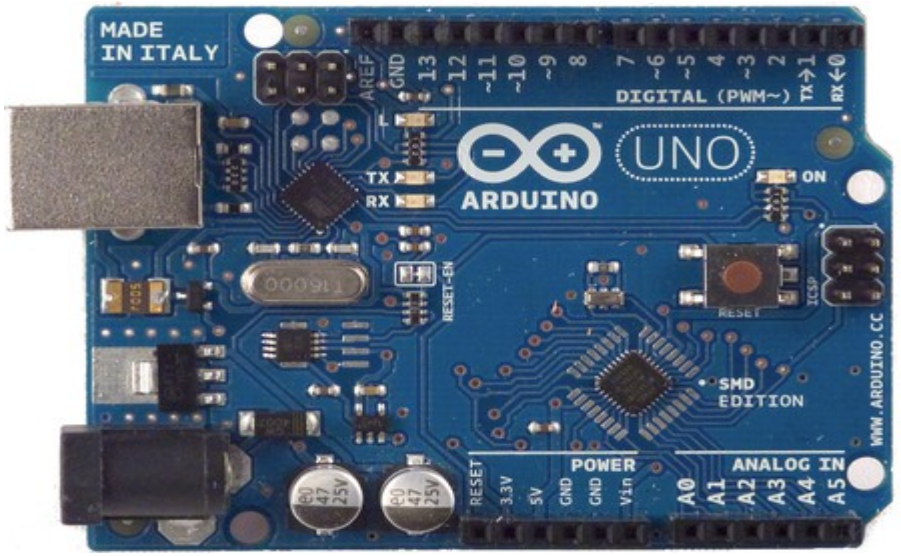
Arduino CAT Controller for HPSDR

John Melton
g0orx/n6lyt

What are Arduinos?

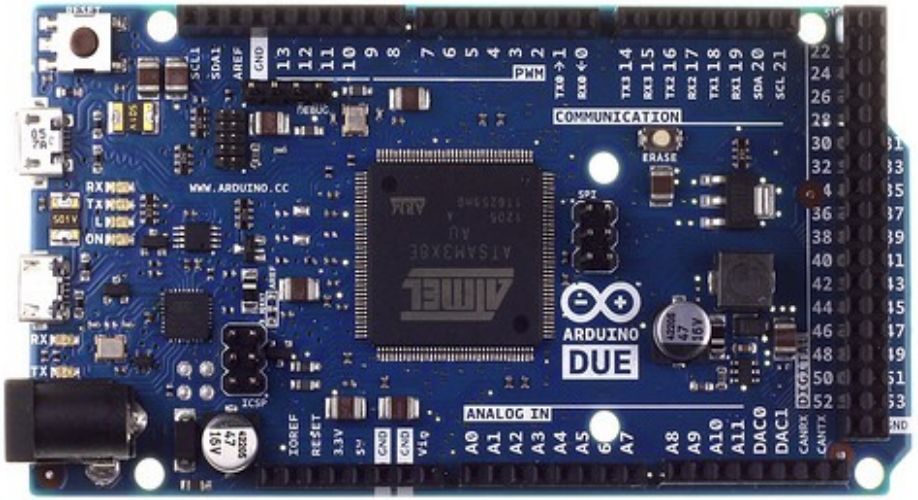
- Open Source Hardware and Software Microcontroller boards.
- Uses various 8-bit Atmel AVR microcontrollers or 32-bit Atmel ARM processors.
- Several boards available:
  - Uno (Pro, Pro Mini, Micro, Nano)
  - Mega
  - Zero
  - Due

Arduino Uno



- ATmega328P
- 8 bit
- 16 Mhz Clock
- 14 Digital Input/Output Pins
- 6 Analog Input Pins
- 32 KB Flash Memory (of which 0.5 KB used by bootloader)
- 2 KB SRAM
- 1 KB EEPROM
- USB connector
- Power Connector

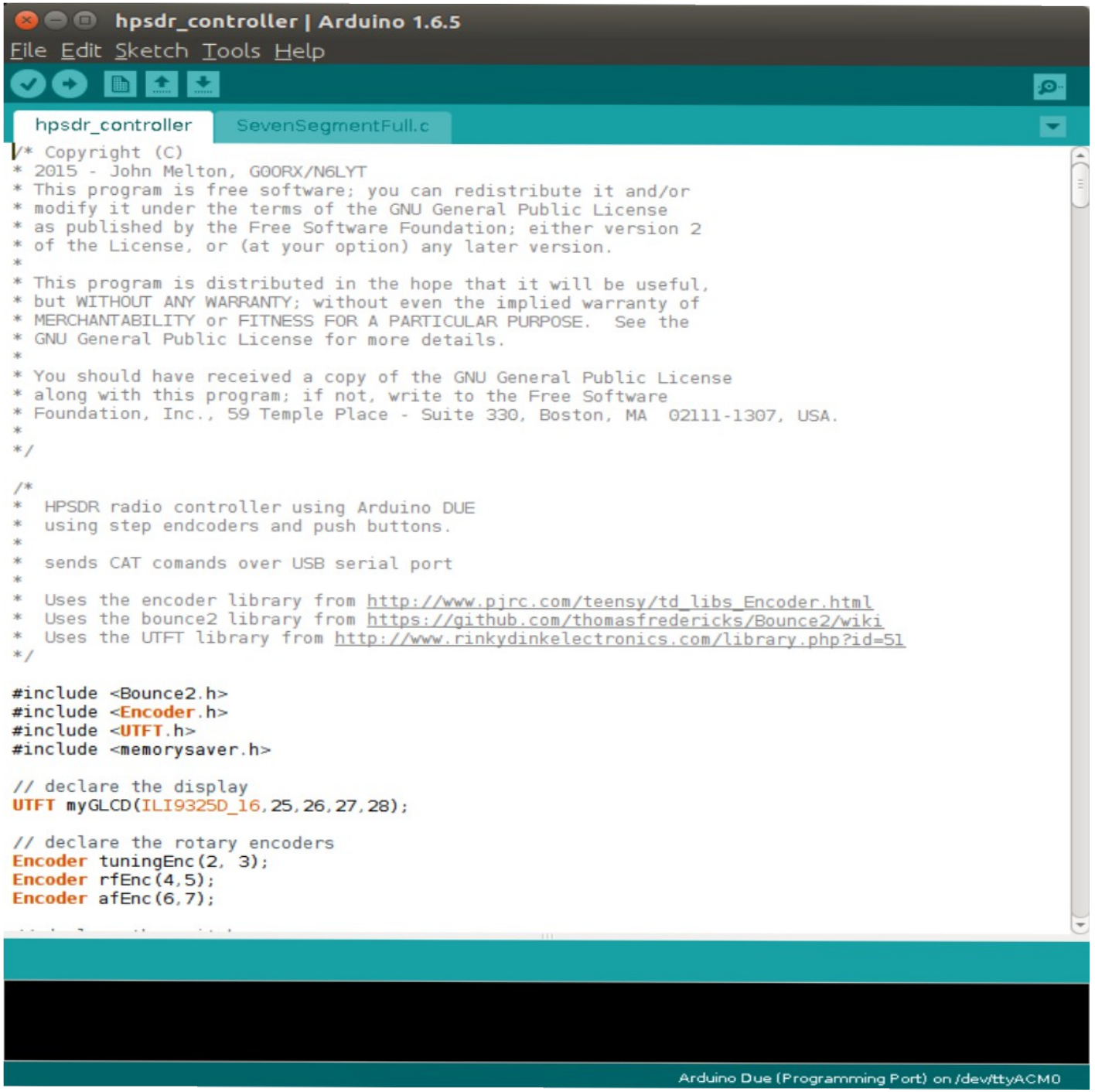
Arduino Due



- ARM CortexM3
- 32 bit
- 84 MHz Clock
- 54 digital input/output pins
- 12 analog inputs
- 2 analog outputs
- 512 KB flash memory
- 96 KB SRAM
- 4 UARTS
- 2 USB ports
- Power Connector

Arduino IDE

- Open-source Arduino Software (IDE)
- Easy to write code and upload it to the board.
- Runs on Windows, Mac OS X, and Linux.
- Written in Java and based on Processing and other open-source software.
- Can be used with any Arduino board.


hpsdr_controller | Arduino 1.6.5
File Edit Sketch Tools Help
hpsdr_controller
SevenSegmentFull.c
/* Copyright (C)
* 2015 - John Melton, G0ORX/N6LYT
* This program is free software; you can redistribute it and/or
* modify it under the terms of the GNU General Public License
* as published by the Free Software Foundation; either version 2
* of the License, or (at your option) any later version.
*
* This program is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
*
* You should have received a copy of the GNU General Public License
* along with this program; if not, write to the Free Software
* Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
*
*/
/*
* HPSDR radio controller using Arduino DUE
* using step endcoders and push buttons.
*
* sends CAT comands over USB serial port
*
* Uses the encoder library from http://www.pjrc.com/teensy/td_libs_Encoder.html
* Uses the bounce2 library from https://github.com/thomasfredericks/Bounce2/wiki
* Uses the UTFT library from http://www.rinkydinkelectronics.com/library.php?id=51
*/
#include <Bounce2.h>
#include <Encoder.h>
#include <UTFT.h>
#include <memorysaver.h>
// declare the display
UTFT myGLCD(ILI9325D_16,25,26,27,28);
// declare the rotary encoders
Encoder tuningEnc(2, 3);
Encoder rfEnc(4,5);
Encoder afEnc(6,7);
Arduino Due (Programming Port) on /dev/ttyACM0

Arduino Application

- A sketch is the name that Arduino uses for a program. It's the unit of code that is uploaded to and run on an Arduino board.
- A function (otherwise known as a procedure or sub-routine) is a named piece of code that can be used from elsewhere in a sketch.
- There are two special functions that are a part of every Arduino sketch: setup() and loop(). The setup() is called once, when the sketch starts. The loop() function is called over and over again and is the heart of most sketches.

Simple Sketch

```
#define buttonPin 3

void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}


void loop()
{
  if (digitalRead(buttonPin) == HIGH)
    Serial.write('H');
  else
    Serial.write('L');
  delay(1000);
}
```

Arduino Libraries

There are a vast number of libraries available that can be downloaded to provide extra functionality for working with different hardware.

- Switch debounce
- Rotary encoder
- TFT display

CAT Commands

PowerSDR has a CAT interface that sends and receives message over a serial port.

For Example:

ZZSA Command

Moves VFO A up one Tune Step

ZZSB Command

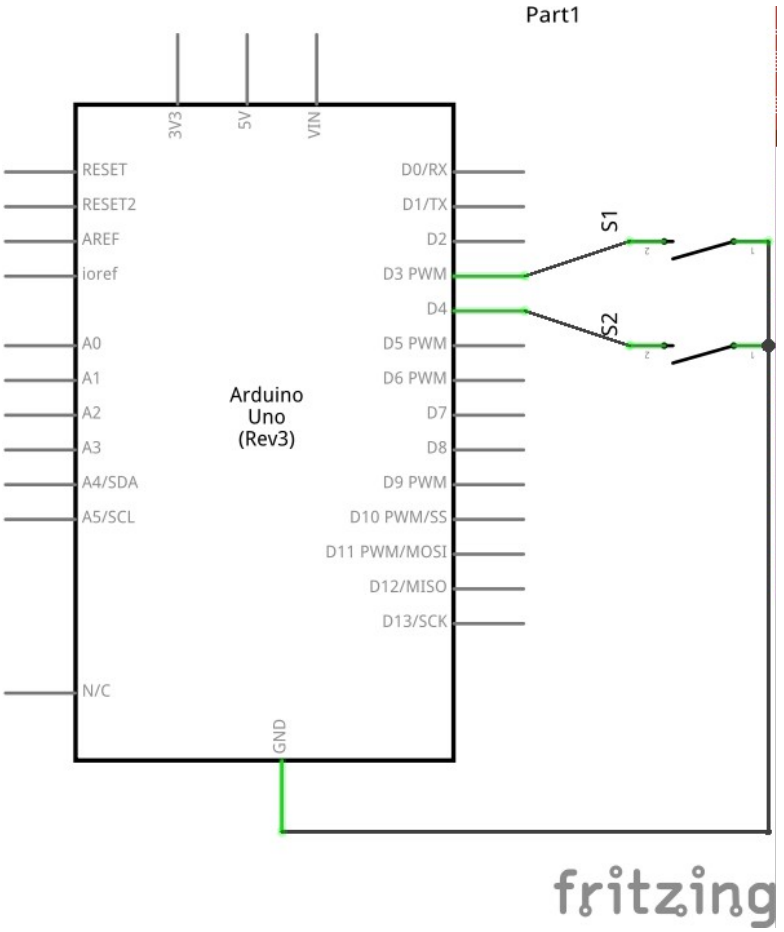
Moves VFO A down one Tune Step

CAT Commands

CAT commands are terminated with a ;

To move VFO A up one tune step we actually send “ZZSA;”

2 Button CAT Controller



2 Button frequency up/down

```
#include <Bounce2.h>

#define stepUpPin 3
#define stepDownPin 4
Bounce stepUpSwitch = Bounce();
Bounce stepDownSwitch = Bounce();

#define PRESSED 0

void setup()
{
  Serial.begin(9600);

  pinMode(stepUpPin, INPUT);
  stepUpSwitch.attach( stepUpPin ); // attach to switch
  stepUpSwitch.interval(20); // 20ms settle time
  digitalWrite(stepUpPin,HIGH); // enable internal pull up

  pinMode(stepDownPin, INPUT);
  stepDownSwitch.attach( stepDownPin ); // attach to switch
  stepDownSwitch.interval(20); // 20ms settle time
  digitalWrite(stepDownPin,HIGH); // enable internal pull up
}

void loop()
{
  if(stepUpSwitch.update()) {
    if(stepUpSwitch.read()==PRESSED) {
      Serial.print("ZZSA;"); // Step the frequency UP
    }
  }
  if(stepDownSwitch.update()) {
    if(stepDownSwitch.read()==PRESSED) {
      Serial.print("ZZSB;"); // Step the frequency DOWN
    }
  }
}
```

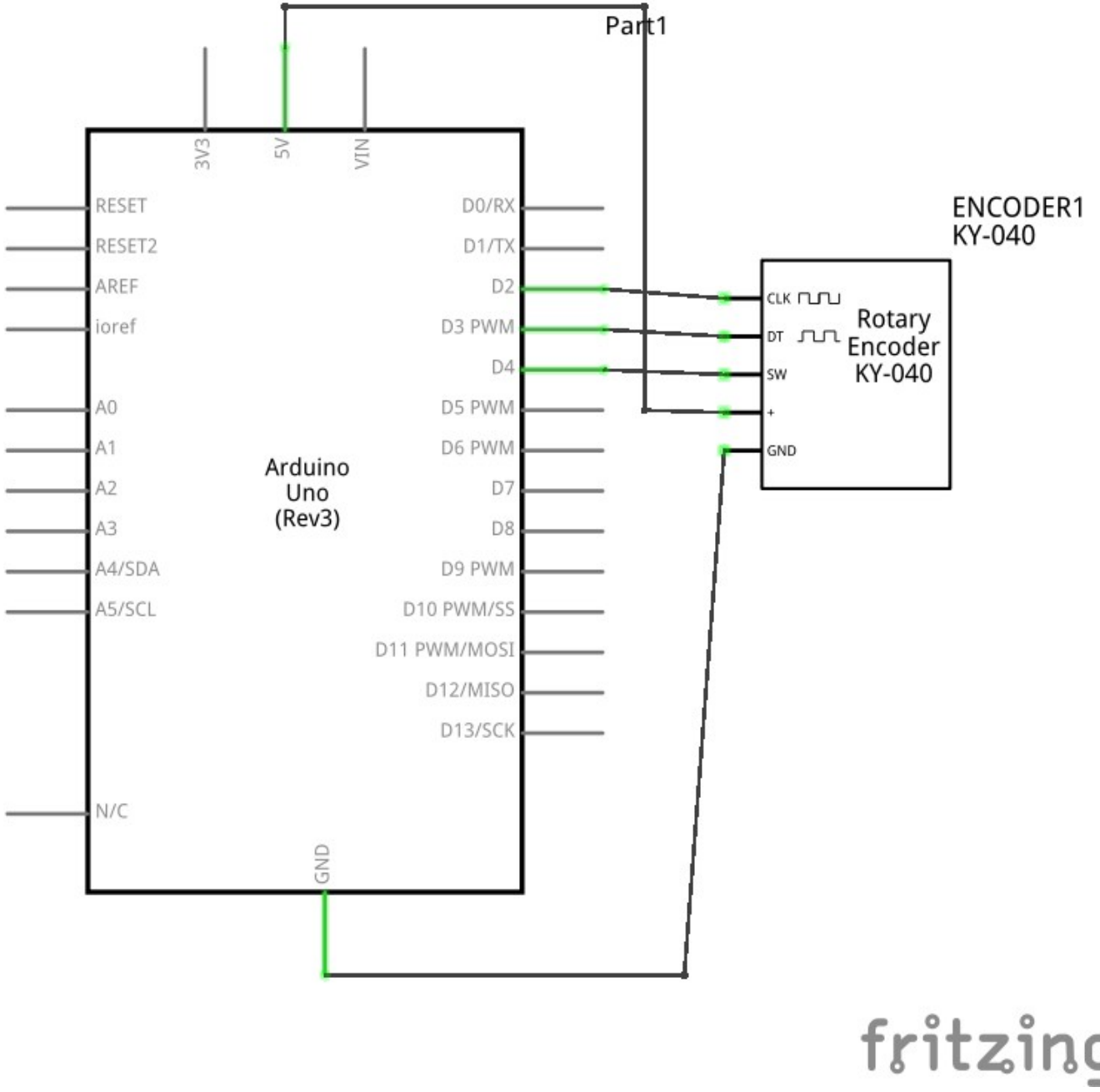
Rotary Encoders

- KY-040 rotary encoder
- 24 steps per revolution
- Push switch

Easy to connect to Arduino

- CLOCK, DATA to 2 analog pins
- GND to GND pin
- +VE to +5v
- SW to analog pin

Encoder CAT Controller



Rotary Encoder frequency up/down

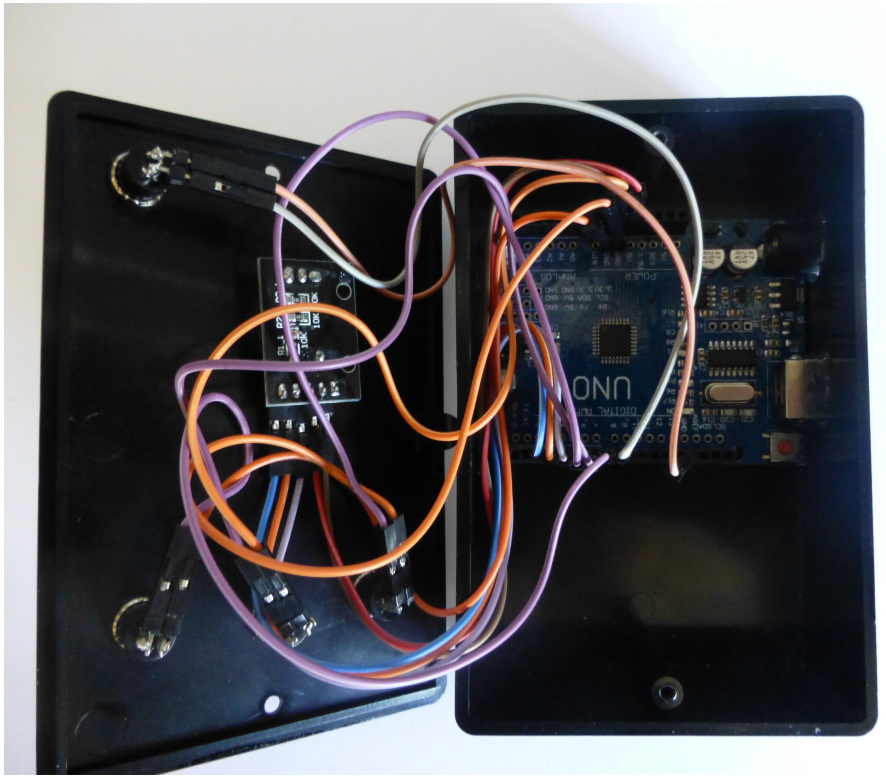
```
#include <Encoder.h>

Encoder tuningEnc(2, 3);

void setup()
{
  Serial.begin(9600);
  tuningEnc.write(0);
}

void loop()
{
  long tunePosition = tuningEnc.read();
  if (tunePosition != 0) {
    if(tunePosition<0) {
      Serial.print("ZZSB;");
    } else {
      Serial.print("ZZSA;");
    }
    tuningEnc.write(0);
  }
}
```

Basic CAT Controller
Arduino Uno

- 1 Rotary Encoder (with switch)
- 4 push to make switches

- Tuning
- Band Up/Down
- Filter Up/Down
- AF Gain
- RF Gain
- PTT

Basic CAT Controller



Basic CAT Controller

```
int function = 0;

#define functionPin 7
Bounce functionSwitch = Bounce();

void setup()
{
  // setup function pin
  pinMode(functionPin, INPUT);
  functionSwitch.attach(functionPin);
  functionSwitch.interval(20);
  digitalWrite(functionPin, HIGH);


}

void loop()
{
  if(functionSwitch.update()) {
    if(functionSwitch.read()==0) {
      function=1;
    } else {
      function=0;
    }
  }

  .
  .
  .
}
```

Basic CAT Controller

```
void loop()
{
  .
  .
  .

  if(bandSwitch.update()) {
    if(bandSwitch.read()==0) {
      if(function) {
        Serial.print("ZZBD;");
      } else {
        Serial.print("ZZBU;");
      }
    }
  }
  .
  .
  .
}
```

Basic CAT Controller

```
char message[8];
int messageIndex=0;

void loop()
{
  checkSerialData();
  .
  .
  .

  if(filterSwitch.update()) {
    if(filterSwitch.read()==0) {
      Serial.print("ZZFI;");
    }
  }
  .
  .
  .
}
```

```
void checkSerialData() {
  while(Serial.available() > 0) {
    char c=Serial.read();
    if(c==';') {
      if(strncmp(message,"ZZFI",4)==0 && messageIndex==6) {
          int filter=((message[4]-'0')*10)+(message[5]-'0');
          if(function) {
            filter--;
            if(filter<0) {
              filter=11;
            }
          } else {
            filter++;
            if(filter>11) {
              filter=0;
            }
          }
          Serial.print("ZZFI");
          Serial.print(filter/10);
          Serial.print(filter%10);
          Serial.print(";");
      }
      messageIndex=0;
    } else {
      message[messageIndex++]=c;
    }
  }
}
```

Advanced CAT Controller
Arduino Due

- Multiple Rotary Encoders
- Multiple push buttons
- TFT Display

Advanced CAT Controller

- AF Gain / AGC Gain
- RF Gain / Tune Gain
- VFO A / VFO B
- Band Up / Down
- Mode Up / Down
- Filter Up / Down
- NR / NF
- PTT / Tune

Advanced CAT Controller

Frequency
Mode
S Meter
AF Gain
RF Gain
AGC Gain

Advancede CAT Controller

- Display – SainSmart 2.8 inch TFT (240x320)
- Touch screen with serial interface (not currently used)
- RGB
- Integrated Power, Gate and Source Driver With RAM
- UTFT Library
  - API with drawing primitives and fonts

Advancede CAT Controller

- For performance:
  - init Function to draw static information
    Called from setup() function
  - update Function to draw dynamic information
    Called from loop() function

Advanced CAT Controller

```
void initSMeter() {
  myGLCD.setColor(0,255,0); // Green
  myGLCD.drawLine(10,85,64,85);
  myGLCD.drawLine(10,85,10,80); // S0
  myGLCD.drawLine(16,85,16,82); // S1
  myGLCD.drawLine(22,85,22,82); // S2
  myGLCD.drawLine(28,85,28,80); // S3
  myGLCD.drawLine(34,85,34,82); // S4
  myGLCD.drawLine(40,85,40,82); // S5
  myGLCD.drawLine(46,85,46,80); // S6
  myGLCD.drawLine(52,85,52,82); // S7
  myGLCD.drawLine(58,85,58,82); // S8

  myGLCD.setColor(255,0,0); // Red
  myGLCD.drawLine(64,85,124,85);
  myGLCD.drawLine(64,85,64,80); // S9
  myGLCD.drawLine(84,85,84,80); // S9+20
  myGLCD.drawLine(104,85,104,80); // S9+40
  myGLCD.drawLine(124,85,124,80); // S9+60

  myGLCD.setFont(SmallFont);
  myGLCD.setColor(255,255,255); // White
  myGLCD.print("3",26,87);
  myGLCD.print("6",44,87);
  myGLCD.setColor(255,0,0); // Red
  myGLCD.print("9",62,87);
  myGLCD.print("+60",120,87);
}
```

```
void drawSMeter() {
  myGLCD.setColor(0,0,0); // Black
  myGLCD.fillRect(10,75,124,79);
  myGLCD.setColor(255,255,0); // Yellow
  myGLCD.fillRect(10,75,10+127+dbm,79);

  myGLCD.setFont(SmallFont);
  myGLCD.setColor(255,255,255); // WHite
  char s[32];
  sprintf(s, "%d dBm", dbm);
  myGLCD.print(s, 130, 75);
}
```

Advanced CAT Controller

```
unsigned long smetertimer=0;

void loop() {
  .
  .
  .
  sMeter();
  .
  .
  .
}

void sMeter() {
  if(smetertimer==0) {
    smetertimer=millis()+100L;
  } else {
    if(millis()>smetertimer) {
      Serial.print("ZZSM0;");
      smetertimer=millis()+100L;
    }
  }
}
```

Q & A

- Arduino
  - http://www.arduino.cc
- Source Code
  - http://svn.tapr.org/repos_sdr_hpsdr/trunk/N6LYT/Arduino
- PowerSDR CAT reference
  - http://svn.tapr.org/repos_sdr_windows/PowerSDR/branches/doc/CAT/PowerSDR%20CAT%20Command%20Reference%20Guide.pdf