Adding auxiliary outputs to the YCCC SO2R mini

Kevin Schmidt, W9CF
6510 South Roosevelt Street
Tempe, Arizona 85283 USA

1 Introduction

The YCCC SO2R Mini https://nn1c.org/so2r/, as its name implies, is a small single-operator two-radio interface box which includes the most important elements needed for interfacing two radios. Comparing to the more full featured YCCC SO2R+ http://www.k1xm.org/SO2R/ the Mini, as delivered, does not include LEDs indicating which of the radios are receiving and which can transmit, manual switches, or auxiliary outputs that can be interfaced to an antenna switch or band decoder.

The SO2R Mini web site gives information on how to add LEDs and switches if desired. This document contains my notes on how I added auxiliary outputs.

2 Current Mini software and hardware

The provided Windows connector software supports the OTRSP, `https://www.k1xm.org/OTRSP/index.html`, (Open Two-Radio Switching Protocol) AUX command to provide two 4-bit auxiliary outputs, i.e. 8 lines total. It sends a command byte and the auxiliary output byte to the Arduino sketch. The provided Arduino sketch ignores these, but has a commented define in the file `keyer_features_and_options.h`

```
// #define FEATURE_SO2R_ANTENNA
// SO2R Box antenna selection (not fully implemented)
```

that turns on detection, and stores the value internally in the code in `k3ng_keyer.ino`

```
if ((incoming_serial_byte & 0xf0) == 0xa0)
{
  so2r_antenna_1 = incoming_serial_byte & 0x0f;
  // TBD:  Provide antenna information outputs
  return;
}
```

and similarly for the other auxiliary output.

The SO2R Mini designers have done the heavy lifting, so it is straightforward to add two auxiliary outputs as defined by the OTRSP.

Looking at the Arduino, 8 additional output lines are not available, so directly connecting Arduino outputs to an auxiliary connector won't work. However, the SO2R Mini designers clearly were thinking about making the circuit expandable and provide various places to mount headers on the board, and in particular an I^2C header for serial communication is available. I decided to use that to control an I/O expander to include auxiliary outputs.

3 My implementation

I mounted a keyed JST connector to my SO2R Mini board, but a plain 4 pin header would also work.

There are many I/O expanders available. I first tested with a Texas instrument PCF8574 since these are readily available. The problem I found is that while they will sink substantial current, they can only provide about 100 microamps of source current at 5V. This was not sufficient for my application, so I decided instead to use a Microchip MCP23017. The datasheet is available at https://ww1.microchip.com/downloads/aemDocuments/documents/APID/ProductDocuments/DataSheets/MCP23017-Data-Sheet-DS20001952.pdf.

This chip actually has 16 I/O pins, but I only used 8. You could expand this to all 16 pins easily, the problem is where to mount a 16 or more pin connector to the SO2R Mini. You can buy a DIP version of the chip for $1.69 at Digikey and mount it on a prototype pc board, but I decided to go with the Adafruit board which has convenient headers and costs $5.95. Its available from Digikey https://www.digikey.com/en/products/detail/adafruit-industries-llc/5346/15913270. You can also buy it directly from Adafruit https://www.adafruit.com/product/5346. The schematic diagram of the board is available at https://learn.adafruit.com/adafruit-mcp23017-i2c-gpio-expander/downloads which shows how to change the I^2C address, etc.

I mounted headers to the Adafruit board for 8 output pins and ground along with 4 pins to interface with the Mini's I^2C port which provides power and ground along with the I^2C clock and data.

4 Software changes

As noted above, most of the work has already been done by the SO2R Mini designers. Adafruit provides a library to use their board, but I think it is easier to use the Arduino Wire library which is already used in the K3NG sketch for LCD drivers. Looking at the MCP23017 datasheet, along with the example code at https://www.circuitbasics.com/how-to-use-an-mpc23017-port-expander-on-the-arduino/, and noting that the Adafruit board sets the default I^2C address 0x20, made everything easy to program. I have provided a patch file with the changes I made at auxiliary_mcp23017.txt. I program in linux, so the patch command is always available there. It can be applied by downloading the sketch from https://nn1c.org/wp-content/uploads/2020/01/SO2R_Mini_Arduino_Sketch_Configured.zip and in linux

```
unzip SO2R_Mini_Arduino_Sketch_Configured.zip
patch -p 1 < auxiliary_mcp23017.txt
```

If you use Windows, and want to patch the code, you can install git for Windows https://git-scm.com/download/win which includes the patch executable. You can also make the changes manually as described next.

Changing manually:

- In the file `keyer_features_and_options_yccc_so2r_mini.h` uncomment the line

  ```
  // #define FEATURE_SO2R_ANTENNA        // SO2R Box antenna selection (not fully implemented)
  ```

  so it reads

  ```
  #define FEATURE_SO2R_ANTENNA        // SO2R Box antenna selection
  ```

- In the file `k3ng_keyer.ino`
  - At line 2019 change the lines

    ```
    #ifdef FEATURE_SO2R_ANTENNA
      uint8_t so2r_antenna_1 = 0;
      uint8_t so2r_antenna_2 = 0;
    #endif //FEATURE_SO2R_ANTENNA
    ```

    to

    ```
    #ifdef FEATURE_SO2R_ANTENNA
      #include <Wire.h>
      #define SO2R_ANTENNA_ADDR (0x20) // low low low
      #define SO2R_ANTENNA_GPIOA (0x12)
      #define SO2R_ANTENNA_GPIOB (0x13)
      #define SO2R_ANTENNA_IODIRA (0x00)
      #define SO2R_ANTENNA_IODIRB (0x01)
      uint8_t so2r_antenna_1 = 0;
      uint8_t so2r_antenna_2 = 0;
    #endif //FEATURE_SO2R_ANTENNA
    ```

  - At line 2080 change

    ```
    initialize_sd_card();
    ```

    to

    ```
    initialize_sd_card();
    initialize_so2r_antenna();
    ```

  - At line 16994 change

    ```
    #endif //DEBUG_STARTUP
    }


    //------------------------------------------------------------------
    ```

to

```
#endif //DEBUG_STARTUP
}

void initialize_so2r_antenna() {
#ifdef FEATURE_SO2R_ANTENNA
  Wire.begin();
  #if defined(WIRE_HAS_TIMEOUT)
    Wire.setWireTimeout(10000,true); // 10ms  -- timeout should never occur
  #endif
// Set all 16 pins to outputs, default 0, so they will all be low
// We only use register A 8 pins for SO2R mini
  Wire.beginTransmission(SO2R_ANTENNA_ADDR);
  Wire.write(SO2R_ANTENNA_IODIRA);
  Wire.write(0x00);
  Wire.endTransmission();
  Wire.beginTransmission(SO2R_ANTENNA_ADDR);
  Wire.write(SO2R_ANTENNA_IODIRB);
  Wire.write(0x00);
  Wire.endTransmission();
#endif //FEATURE_SO2R_ANTENNA
}

#ifdef FEATURE_SO2R_ANTENNA
  void update_so2r_antenna() {
    Wire.beginTransmission(SO2R_ANTENNA_ADDR);
    Wire.write(SO2R_ANTENNA_GPIOA);
    Wire.write(((so2r_antenna_2 << 4) | so2r_antenna_1) & 0xff);
    Wire.endTransmission();
  }
#endif //FEATURE_SO2R_ANTENNA


//--------------------------------------------------------------------
```

– At line 21370 and again at line 21377, change

```
      // TBD:  Provide antenna information outputs
```

to

```
      update_so2r_antenna();
```

At this point, compile and upload the modified sketch. With the MCP23017 board plugged into the Mini's I^2C header (*caution, the order of the 4 wires is not the same on the Mini as on the Adafruit board – make sure you have them connected correctly!*) the A0

through A3 outputs should reflect AUX1 and A4 through A8 should reflect AUX2. If you use N1MM+, these should follow the Antenna assignments.

Once you have the chip working then the hard work of figuring out how to complete the physical part of the project is left.

5 My current implementation

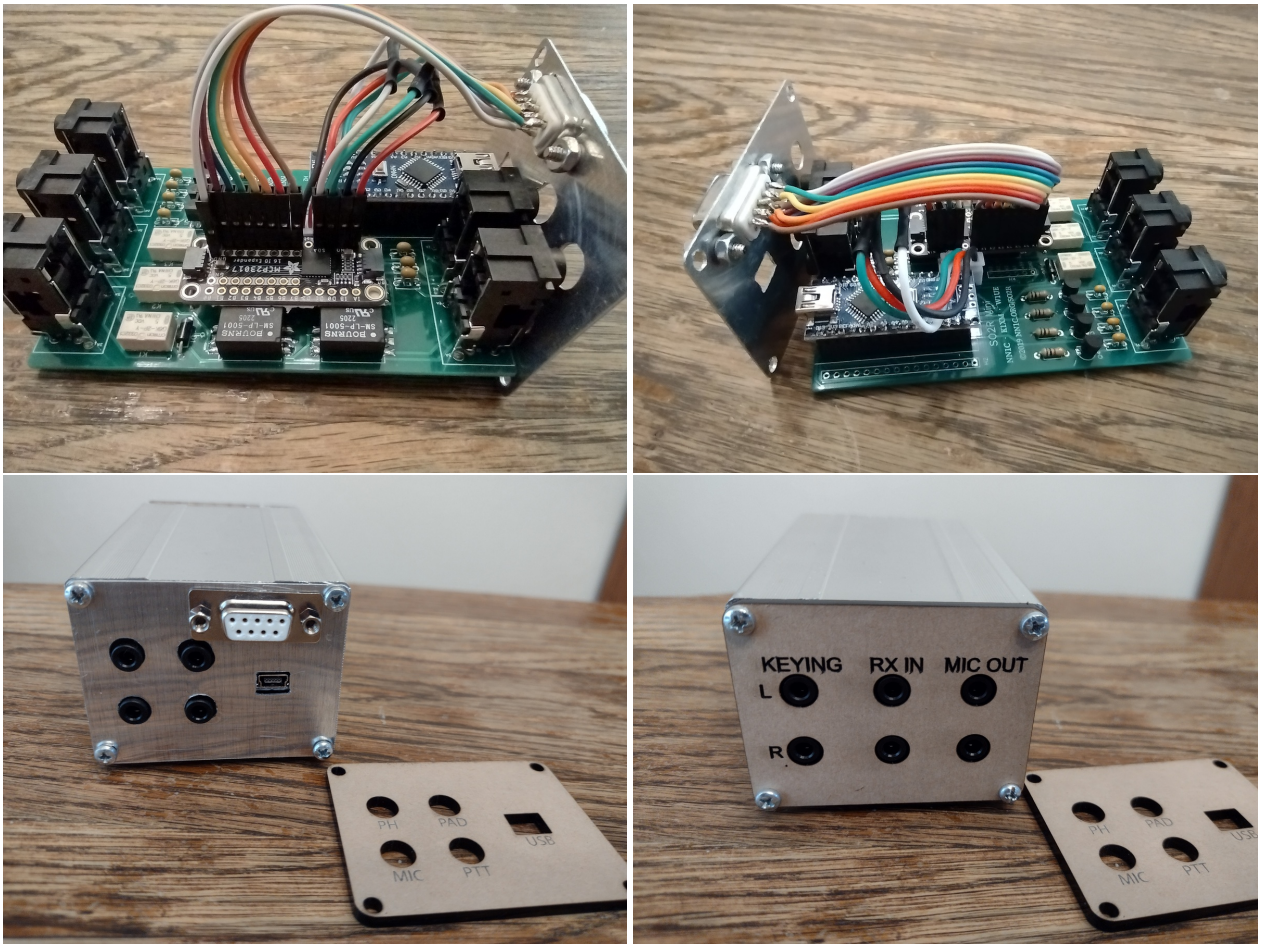

Figure 1: Photographs of the SO2R Mini board with the Adafruit MCP23017 board attached with double sided tape and connected to the Mini's I^2C bus and a DB-9 connector.

I used double sided tape to secure the Adafruit board to the audio transformers in the Mini. I don't think that will induce any audio interference, and I don't hear any so far. It's easy to move if necessary.

After initial testing, I used a scribe to transfer the hole pattern of the Mini end plate to a scrap of thin aluminum sheet, and then squeezed in a DB-9 connector carrying the 8 output lines and a common ground. My plan is to mount the connector on the original Mini end plate after I make sure this works the way I want it to. Everything I have done so far allows me to revert to the original Mini.

The MCP23017 mounting and the scrap aluminum end plate with DB-9 are shown in Fig. 1.

As a first test, of my quick and dirty packaging, I wired a DB-9 male connector to some jumpers and breadboarded 8-LEDs. These follow the antenna outputs in N1MM+ and the outputs using TR log in linux as shown in Fig. 2.

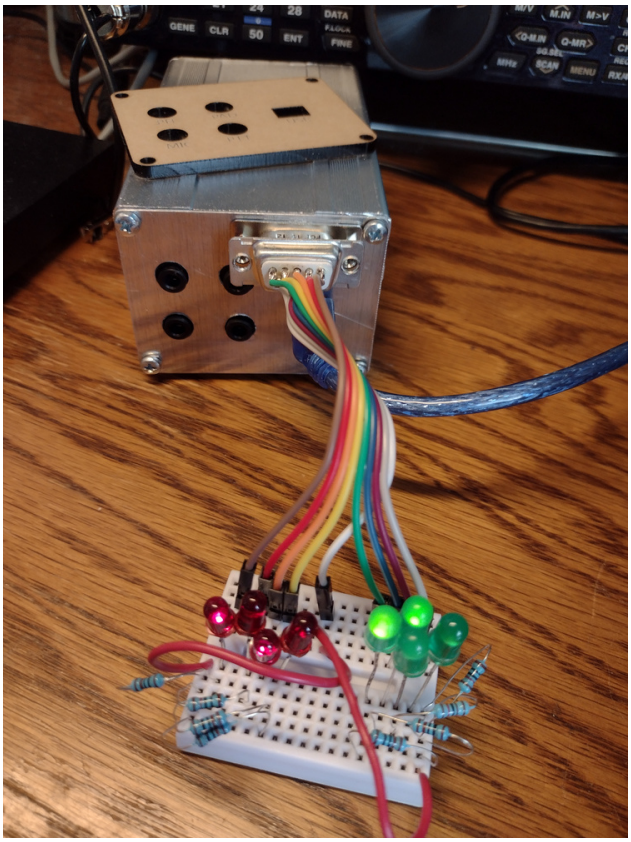

Figure 2: A test showing LEDs following the logging program.

6 Additional thoughts

Looking at the YCCC SO2R+ box schematic, the auxiliary output lines there each have a 220 ohm series current limiting resistor. It would be a good idea to include those here. I plan to mount them on the DB-9 connector. In addition, to keep RF out of the Mini, it would also be a good idea to bypass the DB-9 connections to ground with 0.01 μF capacitors.

The DB-9 I used is quite close to the paddle input, so using this mounting, you may have to choose a slim DB-9 male connector or a right angle paddle connector to keep from having interference. An alternative connector with at least 9 connections (8 signal plus ground) might be a better choice if one can be identified.