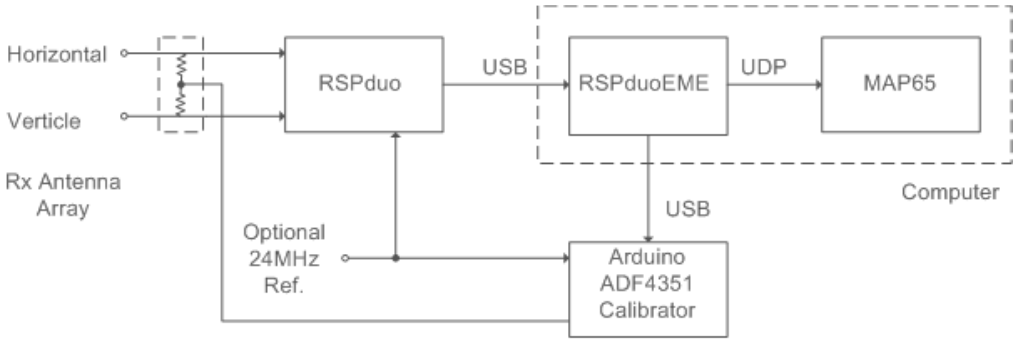
RSPduoEME User Instructions

David Warwick G4EEV

(For Software Version 1.31)
December 2020

1. Introduction

The RSPduoEME application was written to specifically interface the SDRplay RSPduo SDR with the MAP65 wideband JT65 decoder, to allow adaptive polarisation for EME stations. The RSPduo is a 1Khz – 2GHz 14bit dual-tuner SDR receiver with a common mixer frequency and has the option to be locked to an external frequency reference. To allow for adaptive polarisation, both the vertical and horizontal signals from the antenna array are required to be received and processed in phase. This can be achieved with this software, however, the initial start-up phase difference between the two RSPduo channels needs to be calibrated out. From software version 1.31 onwards, the RSPduoEME application includes the option to automatically perform this calibration at start-up with an accompanying Arduio project. The software has been written in C++ using Qt Creator V4.6 and is intended to run on a Windows10 PC.



Adaptive Polarisation System using RSPduoEME and MAP65

2. Installation

RSPduoEME is designed to run on a Windows 10 PC with sufficient processing power (under development this was achieved with an AMD A10-7800 R7 @ 3.5GHz and utilised 16% CPU time). The RSPduo needs to be installed and working with the SDRplay API/Hardware Driver version 3.06 or later. The API/hardware Driver is usually installed with the SDRuno install, however, it is also available separately from the SDRplay website.

The RSPduoEME executable package zip file needs to be downloaded and unpacked into a suitable directory, which will now contain the files required to run the application. (If the project has been downloaded from GitHub, this is the executable package zip file contained in that project.) Additionally, it is necessary to copy the SDRplay x86 sdrplay_api.dll file from where it was installed (typically C:\Program Files\SDRplay\API\x86\) to this directory. The application is run by executing the SDRduoEME.exe file from this directory, it may be convenient to create a short-cut to this file.

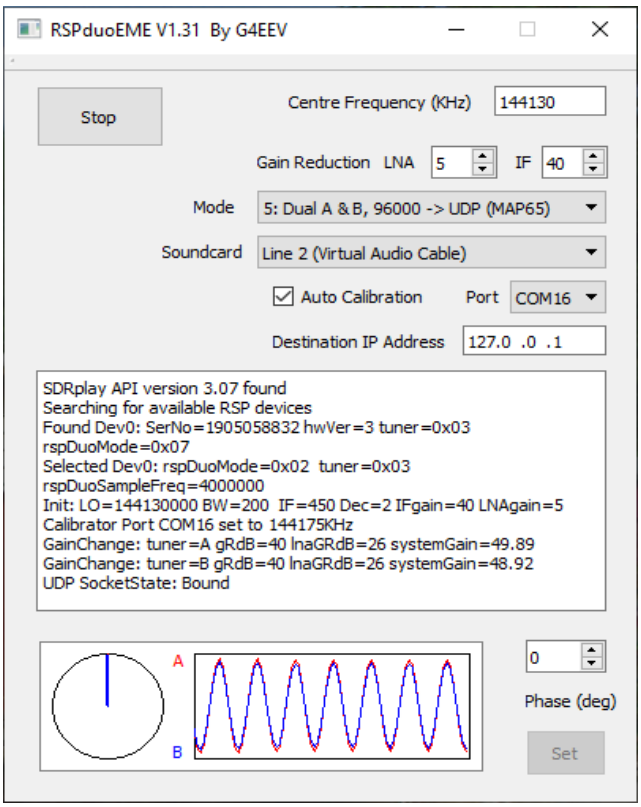


Figure 1, RSPduoEME Main Window

3. Operation

When the application starts it shows a dialog window (figure 1) into which various parameters can be entered prior to starting the RSPduo, these are described in table 1 below. When configured, the 'Start' button should be used to initialise the RSPduo. The application will look for a RSPduo device connected to the host computer (although multiple RSP devices can be connected to the same computer, this feature has not been tested). Once found the RSPduo will be configured for dual receive mode at the required centre frequency and will commence outputting data to either the Soundcard or as UDP IP packets in Linrad TIMF2 format, compatible with MAP65 network input. During operation the Status display will show relevant information and warnings.

Function	Description
Centre Frequency (KHz)	Enter the required centre frequency in Khz
Gain Reduction LNA	This is the RSPduo LNA gain reduction value (0 – 9) , each step is 6dB (0 = full LNA gain). This value can also be adjusted while the RSPduo is running.
Gain Reduction IF	This is the RSPduo IF gain reduction value in dB (20 – 59). This value can also be adjusted while the RSPduo is running. The overall system gain is written to the status window. The RSPduo is operating in manual gain control, should an overload be detected this is displayed on the main window display under the start/stop button, this may happen during TX periods.

Mode	There are nine operating modes available from the drop-down window: 1: Single channel A, output to the Soundcard at 96K samples complex (2 channel audio / 16bit). 2: Single channel A, output to the Soundcard at 192K samples complex (2 channel audio / 16bit). 3: Single channel (A) output as network UDP (Linrad TIMF2 format) compatible for input to MAP65. 4: Single channel (A) duplicated as two channel network UDP (Linrad TIMF2 format) compatible for input to MAP65. This is intended for testing and should produce a 45 degree polarisation on MAP65. 5: Dual channels (A&B) output as network UDP (Linrad TIMF2 format) compatible for input to MAP65. This is intended as the normal mode for adaptive polarisation input to MAP65. The TIMF2 data is sent on UDP port 50004. 6: Dual channels (A&B) output as network UDP (Linrad TIMF2 format) compatible for input to MAP65. Additionally, channel A is output to the soundcard at 96K samples complex (2 channel audio / 16bit). 7: Dual channels (A&B) output as network UDP (Linrad TIMF2 format) compatible for input to MAP65. Additionally, channel B is output to the soundcard at 96K samples complex (2 channel audio / 16bit). Note: Modes 6&7 are reliant upon sufficient PC processing resources and may not work on all systems. 8: Dual channels (A&B) output as network UDP (Linrad RAW16 format) at 96K samples, intended as input to Linrad. This mode is experimental and requires Linrad to be used on a separate PC due to clash of Linrad server port 49812. Network performance has not be verified. The RAW16 data is sent on UDP port 50000. 9: Dual channels (A&B) output to the Soundcard at 96K samples complex (4 channel audio / 16bit). This is intended as input to Linrad in a similar way as using a Delta44 soundcard. This mode is dependant upon using a virtual audio cable configured for 4 channel audio at 96000 samples/sec and appropriate Linrad audio settings. (Linrad offers several audio drivers using Portaudio, some may not work as well as others, I have found that MME works with a latency of ten.)

Soundcard	Used to select the audio output device (or virtual audio cable) for connection to other programs. The card is only selected or changed when Start is pressed (except for modes 6 and 7).
Auto Calibration	Used to enable the Automatic Calibration process in conjunction with an Arduino / ADF4351 Calibrator. When set, the application sends a command via the selected Com Port to start the calibrator, when a stable calibration signal is detected, the application will lock to the required phase and will remove the calibrator signal.
Port	Selects the Arduino Com Port for the Calibrator. The Calibrator will be enabled at start-up and will be deactivated when the phase is Set. To prevent the calibrator signal select 'None'.
Destination IP Address	This is the IP address where UDP packets will be sent. If running on the same PC you can use the loopback address of 127.0.0.1 The network bandwidth required is at least 15Mbit/s, so a Fast Ethernet (100BaseT) network is required as minimum.
Phase	Set the required phase relationship between stream A and B, this will normally be 0. (This can be used to offset the phase relationship for special applications.)
Set	When a calibration signal is used to calibrate the system, pressing set will lock the required phase.

Table 1, Input Functions and Descriptions

4. Phase Display and Calibration

The phase control panel appears at the bottom of the display when dual channel modes 5,6,7,8 and 9 are selected. The circular display represents the phase difference between channel A and B, 0 degrees is represented by the 12 O'clock position and 180 degrees by the 6 O'clock position. To the right of the phase display is an oscilloscope display, channel A is indicated in red and channel B in blue. With no strong signals in the 96KHz passband of the receiver just noise can be seen (figure 2 below).

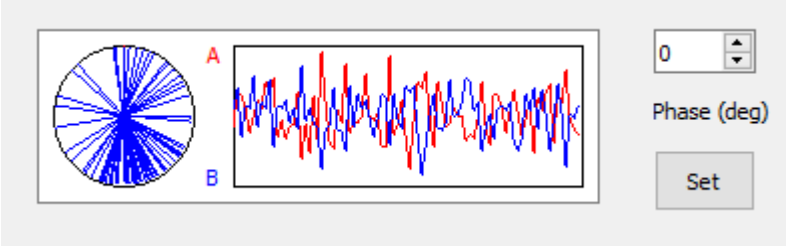


Figure 2
Phase display showing noise and random phase.

To calibrate out the RSPduo start-up phase error, required in dual channel modes, a low level unmodulated test signal is required to be applied to the inputs in phase. The injection point can be anywhere in the antenna and feeder system, however, the phasing prior to that point must be correct. The frequency of the calibration signal has to be within the 96KHz passband of the receiver channel, offsetting from the centre frequency by about 40KHz gives the best display.

The system can be calibrated manually by applying a suitable calibration signal, perhaps from a signal generator or VNA and pressing Set when a stable phase error is observed in the display.

Alternatively, automatic calibration can be performed when using the Arduino / ADF4351 Calibrator project as described below. In this case the application will set the calibration frequency, start the calibrator, wait for a stable display, lock the required phase, and remove the calibrator signal automatically.

Once the phase has been locked the display will freeze showing the locked waveforms and phase angle (see figure 4 below).

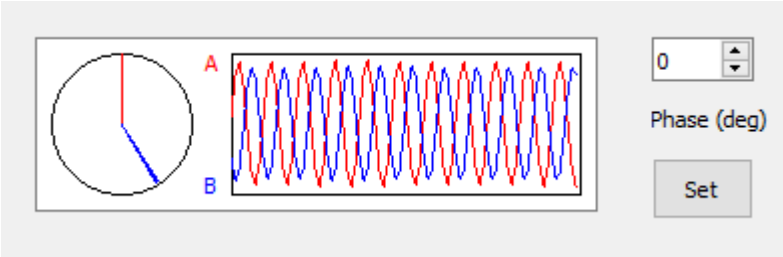


Figure 3
Phase display showing a calibration signal with the two channels out of phase.

.

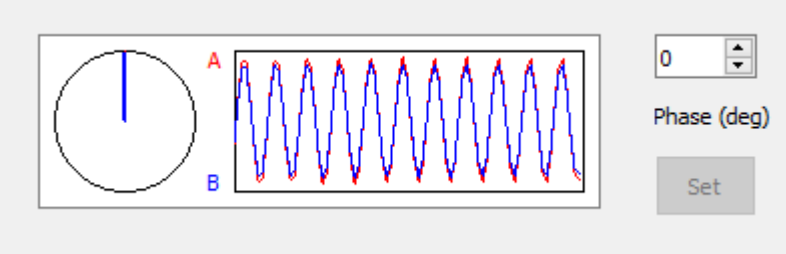


Figure 4
Phase display showing a locked phase, channel A (red) hidden behind channel B (blue).

5. Calibrator

In order to automate the calibration process the following project was adopted and integrated with the RSPduoEME application.

The calibrator is based upon the Analogue Devices ADF 4351 Wideband Synthesizer and the Arduino project by F1CJN (http://f6kbf.free.fr/html/ADF4351%20and%20Arduino_Fr_Gb.htm). This project utilizes an ADF4351 module available on internet auction sites, such as ebay, for around £20. The ADF4351 is capable of operating between 35MHz and 4400MHz in 10KHz steps at a maximum output level of 5dBm. The ADF4351 has a SPI interface, which in this case interfaces with the Arduino.

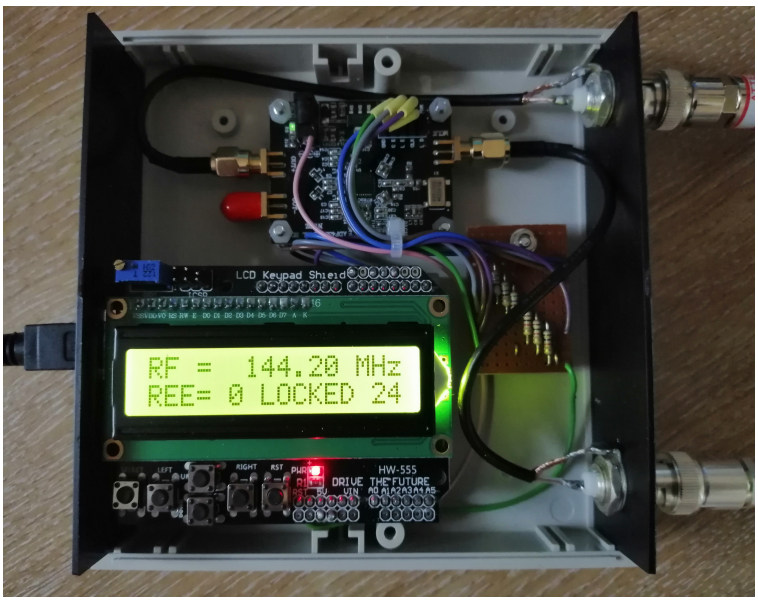
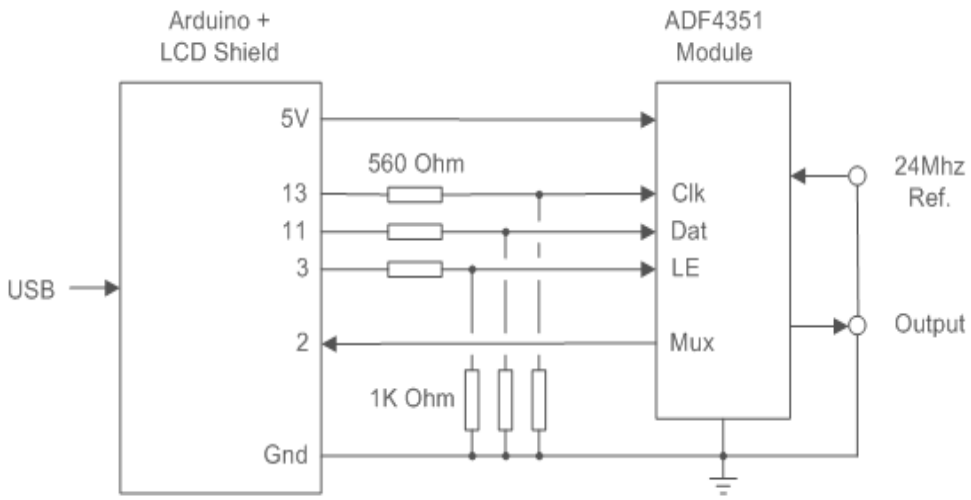


Figure 5

Completed Arduino and ADF4351 project.

The design by F1CJN utilises a LCD Button Shield to display the frequency and the buttons are used to set the required frequency and mode. I have modified the code to allow the frequency to be set via the serial port which is connected to the host computer via USB. Additionally the USB powers the Arduino and the ADF4351 board. A separate 5V power supply may improve spectral purity, however, I have not found this a problem. The RSPduoEME application controls the ADF4351 by sending the required calibration frequency to the Arduino serial port. The calibration is effectively turned off by setting the frequency out of band to 4000MHz. The schematic below shows the interconnections, the 560/1K resistors are used to convert the Arduino 5V to 3.3V logic levels for the ADF4351. Full constructional details of the Arduino project are provided in the F1CJN web article.



Calibrator Schematic

The modified code allows the setting of the external reference to 24MHz, convenient as the RSPduo external reference is also 24MHz. If the external reference is used, the resistor connecting to the XTAL oscillator needs removing on the ADF4351 module. The calibrator reference input mode is set using the push buttons on the shield as described in the F1CJN article. Note, the LCD reference indicator does not work correctly and shows the lock state even when the reference is not applied. This seems to be a ADF4351 hardware issue as the software correctly displays the state of the Mux output. The modified Arduino Sketch is available from the RSPduoEME download zip file.

6. Known Issues

Sometimes when starting the RSPduo the SDRplayAPI software reports the RSPduo can't be found or initiated? Usually restarting RSPduoEME will solve this issue, but sometimes it has been found necessary remove the RSPduo from the USB connection and start again.

When modes 6 or 7 are selected the extra processing may cause occasional brief dropouts of the soundcard output, this usually occurs when the CPU is heavily loaded such as loading a file or program. If this is a problem, the processing priority for the application can be raised to Real Time from the Windows Task Manager, Details menu.

Mode 8 (Linrad RAW16 input) may cause network failure reports due to lost packets. It has not been possible to test this mode on a fast network and PC.