ICARC Fox TDOA Tracker

William Robison

April 26, 2026

This is the start of a manual for the ICARC FOX TDOA Tracker. It covers the 102-73174-51, 102-73174-73, 102-73174-61, 102-73174-82, 102-73174-91, 102-73174-97 and 102-73174-99 boards.

It is a work-in-progress right now so suggestion for updates may be sent to **kc0jfq@n952.ooguy.com**.

Full size documents may be found here: http://n952.ooguy.com/HamRDF

LaTeX Source Files:

1. 158 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/Fox_zNEO_TDOA.tex
2. 1591 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Glossary.tex
3. 1603 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Motivation.tex
4. 1615 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Revision_History.tex
5. 1627 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation.tex
6. 163?? //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_1.tex
7. 16414 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_2.tex
8. 16525 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_3.tex
9. 16630 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_4.tex
10. 16739 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_5.tex
11. 16845 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_6.tex
12. 16947 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_7.tex
13. 17048 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_8.tex
14. 17165 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_9.tex
15. 172?? //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_10.tex
16. 17374 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Theory_Operation_11.tex
17. 17477 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Commanding.tex
18. 17578 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_FREQ.tex
19. 17679 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_TONE.tex
20. 17780 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_VOLU.tex
21. 17881 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_SCAN.tex
22. 17982 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_ROTA.tex
23. 18083 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_TRIP.tex
24. 18184 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_WAIT.tex
25. 18285 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_BATR.tex
26. 18386 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_TDOA_Command_FRAM.tex
27. 18599 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_TRACKER_Operating.tex
28. 186100 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Operating.tex
29. 187101 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_STM32_Operating.tex
30. 189103 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Boards.tex
31. 190104 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Board_73174-51.tex
32. 191107 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Board_73174-61.tex
33. 192108 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Board_73174-73.tex
34. 193109 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Board_73174-82.tex
35. 194110 //new/pig2/STMicro/Projects/FOX_TRACKER_2M/tek_manual/FOX_zNEO_Board_73174-97.tex

Contents

List of Figures

List of Tables

Chapter 1

Glossary of Terms

Source File: FOX_zNEO_Glossary.tex

There is an attempt being made to use some terms in this document in a precise manner. Some of the discussions become a bit muddled when terms are used casually.

1.1 FLASH

An in-circuit erasable and programmable memory.

A type of non-volatile memory that exhibits very asymmetric access speed. Write speed is several orders of magnitude slower that read speed.

Update is handled by erasing the entire device and then loading it one (32 byte) record at a time.

1.2 FRAM

Ferro Magnetic **R**andom **A**ccess **M**emory.

A type of non-volatile memory that exhibits symmetric access speed. In other words the write speed is the same as the read speed. The type of memory is byte accessible (for both read and write) as well as non-volatile.

Update may be handled a record at a time or by erasing the entire device.

1.3 ISR

Interrupt Service Routine.

This is a block of code that deals with an even that is not triggered by the normal flow of instructions in the processor.
Examples would be incoming serial traffic or a timer event.

1.4 Processor

This refers to the zNEO system-on-chip. It has an instruction execution engine (i.e. the CPU), program memory, data memory, and a variety of peripherals.

1.5 Program

71 This refers to the code in the zNEO system-on-chip.

1.6 Sequence

We will use the term **SEQUENCE** in this document to describe a set of (FRAM) instructions
82 that are executed as a group.

Typically this *sequence* is stored in external FRAM memory.

1.7 xxx

218 xxx (the $24^{\text{th.}}$ letter of the alphabet)

Chapter 2

Motivation

Source File: FOX_zNEO_Motivation.tex

Why ?

Explore an automated direction finding machine.

2.1 Requirements

These are required for operation and remain unchanged from the 102-73161 series boards.

2.1.1 2M Band

This

2.1.2 Battery Operation

We

2.1.3 Physical Size

Lets keep it in the standard Hammond 1599E enclosure.

2.1.4 Programmable without special tools

Same as the FOX Transmitter.

2.2 Desirements

These are desired features

Chapter 3

Revision History

Source File: FOX_zNEO_Revision_History.tex

List of updates to the hardware and software.

3.1 Software

Updates and changes to the Operating Software

3.1.1 V0.01

Initial Release.

3.2 Hardware

Updates and changes to the Hardware Design.

3.2.1 102-73174-41

Move to the ST Micro STM32U575 device. This change greatly expands the available memory as the STM32U575 provides 1MB or 2MB of program flash and 768KB of data RAM. The STM32U575 also has floating point that is used to in the FFT analysis.

The STM32U575 may be clocked at speed in excess of 100MHz, although the project is targetd to run at 60MHz to reduce power.

Chapter 4

14

Theory of Operation

Source File: FOX_zNEO_Theory_Operation.tex

The tracker is composed of several units and may be configured in multiple ways.

4.1 The main boards

Source File: FOX_zNEO_Theory_Operation_1.tex

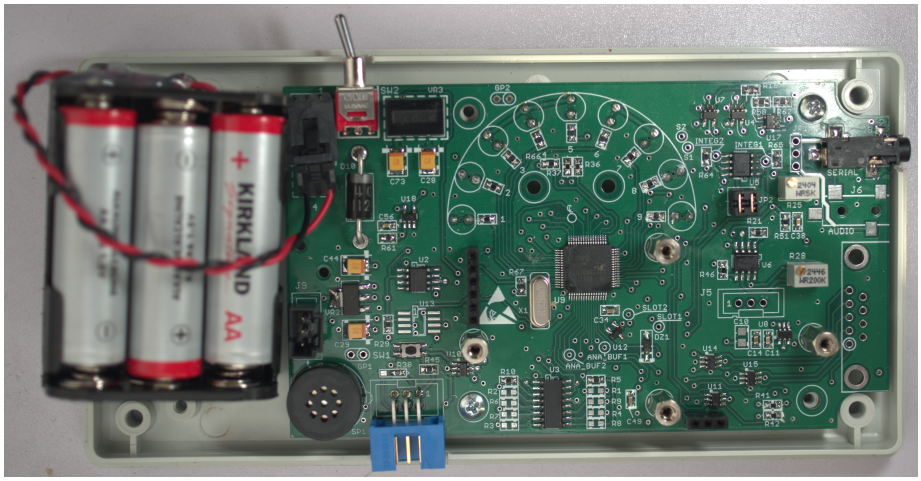


Figure 4.1: 1st. generation TDOA Tracker (102-73174-50)

The processing element is the same zNEO chip that is used on the Fox Transmitter system. Much of the software is reused from that project.

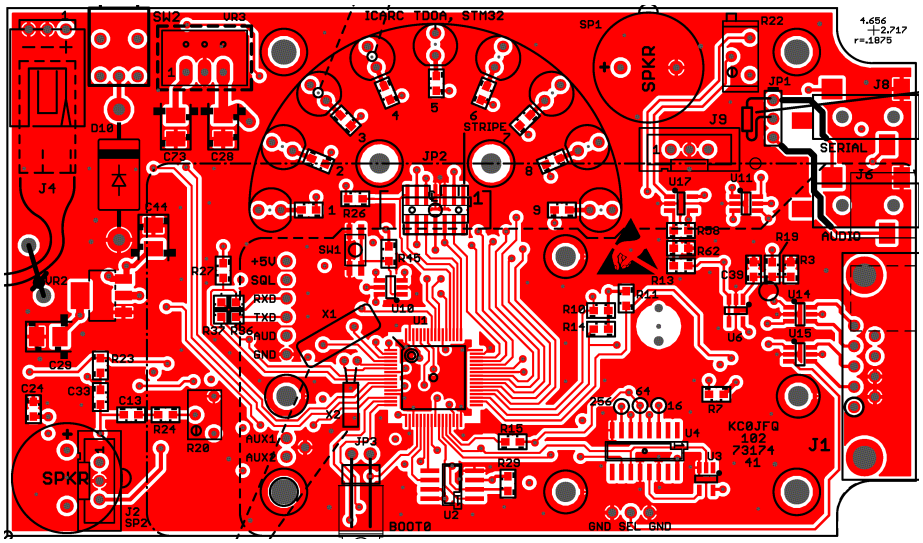


Figure 4.2: 2nd. generation TDOA Tracker (102-73174-41)

Due to lack of resources (Flash, SRAM, FPU, CPU cycles) we replace the zNEO with a STM32U5. This substantially improves conditions and allows much more of the processing to be moved into the digital domain.

Much of the discussion for the zNEO implementation that follows reasonably describes the STM32 implementation. Look into section 4.8 on page 48 for a detailed description of the STM32 board.

4.1.1 Compass Display

Nine LEDs form a meter or compass display. This is directly controlled by the zNEO using the PE and PG pins. Software determines if it acts as a compass display or as a meter display.

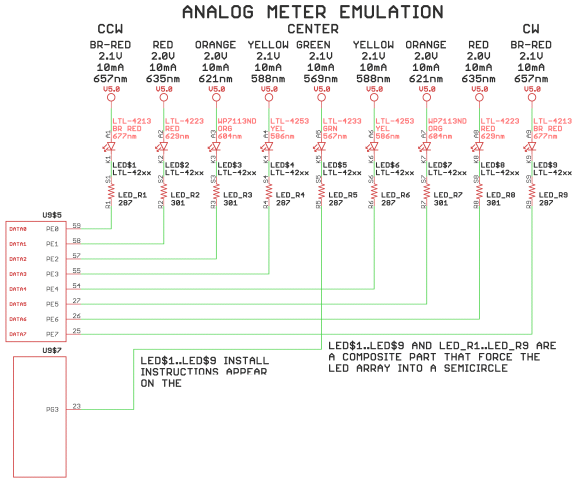


Figure 4.3: Display Logic

The display driver is straight-forward, we simply connect Port E and Port G pins to the LED array.

4.1.2 STM32 Compass Display

The same nine LEDs form the display. This is directly controlled by the STM32U5. Software still determines how it acts.

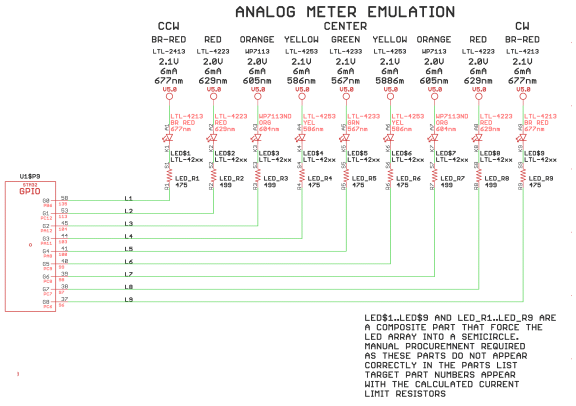


Figure 4.4: STM32 Display Logic

The display driver is similar to that on the zNEO.

4.1.3 Antenna Control Interface

The antennas control interface is simply a set of 4 digital pins that select one of the nine antennas controlled by the antenna controller.

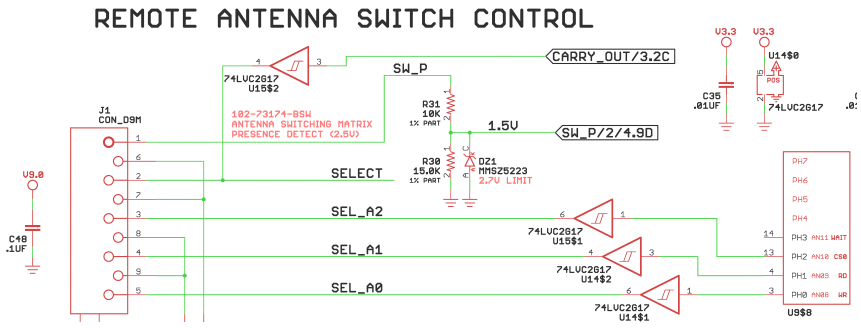


Figure 4.5: Antenna Control Logic

To drive the antenna select, we simply buffer three select lines from the zNEO. The **SELECT** net is driven by a logic gate.

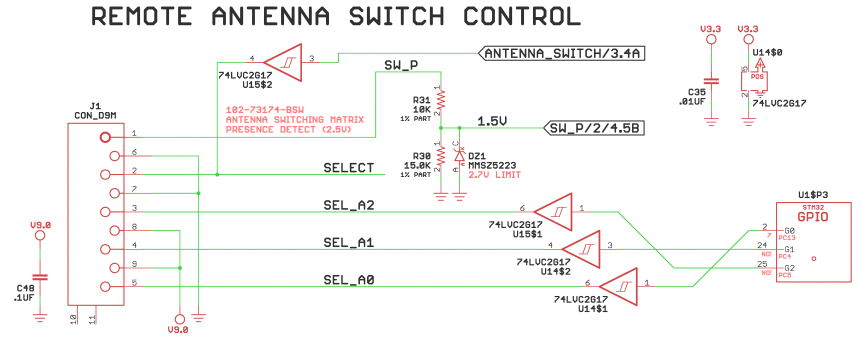


Figure 4.6: STM32 Antenna Control Logic

Virtually identical to the zNEO.

4.1.4 Daughtercard Interface

The daughtercard interface occurs through two vertical connectors.

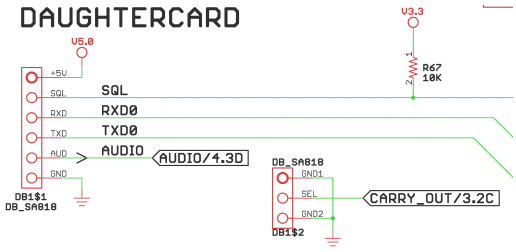


Figure 4.7: Daughtercard Control Logic

The signal interface to the daughtercrd is quite simple.

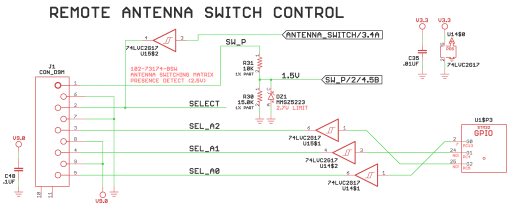


Figure 4.8: STM32 Daughtercard Control Logic

The STM32 implementation functions in the same manner as the zNEO. Interface to the daughtercards is identical (therefor interchangeable).

4.1.4.0.1 SQL from SA818 on daughtercard

The carrier detect signal from the SA818 tranceiver. This line is HI when detecting carrier energy.

4.1.4.0.2 RXD0 from SA818 on daughtercard

The serial data (i.e. status) from the SA818.

This datapath uses the same port to control the SA818 as does the Fox Transmitter to allow use of the same serial handler.

4.1.4.0.3 TXD0 to SA818 on daughtercard

The serial data (i.e. commands) to the SA818.

The datapath matches Fox Transmitter here too.

4.1.4.0.4 AUDIO from SA818 on daughtercard

The audio data from the SA818.

4.1.4.0.5 CARRY_OUT to antenna switch on daughtercard

Control/timing for the antenna switch on the daughtercard.

4.1.4.0.6 Power and Ground for the SA818 on the daughtercard

Note that we call out the SA818 from Nice-RF (and not the DRA818 form Dorji).

The datasheet indicates that the SA818 module will run on 5V.

4.1.5 Host Configuration Interface

This is the same serial interface as seen on the Fox Transmitter to allow configuring the system.

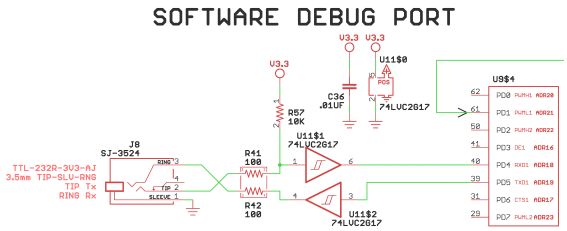


Figure 4.9: Host Configuration Port

As with all these designs, there is a conficuration resistors pair that may be used to swap the TxD and Rxd pins. Here the R41/R42 resistors serve that purpose. Value of these resistors is non critical.

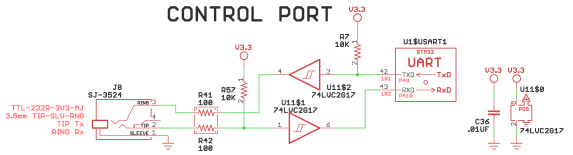


Figure 4.10: STM32 Host Configuration Port

Somehow the reference designators for RxD/TxD swapping remains the same. In either case, the resistor pair is rotated 90°to swap the TxD/RxD pins on J8 (which also keeps its same reference designator).

4.2 Synchronous Detection

Source File: FOX_zNEO_Theory_Operation_2.tex

This section will discuss the detector logic.

Much of the design is based on a description posted by KA7OEI
(https://ka7oei.blogspot.com/2016/11/tdoa-direction-finder-systems-part-1.html).

This function resides on the main board but has been broken out into its own section to make the TOC easier to navigate.

The STM32U5 design, although similar, moves much of the detector away from the analog domain and into the software domain. See section 4.8 on page 48 for the detailed discussion.

4.2.1 The *Simple* TDOA Detector

Time Difference Of Arrival or Time Of Arrival Difference

Simply switching between two antenna elements and using an FM detector can be used to direction find by simply listening to the resulting audio. If we perform the antenna switching at a frequency that is audible, you hear the antenna switching in the audio.

Consider the signal impressed on a pair of antenna elements that sit at different distances from the transmitter. In this discussion we will assume the receiving antenna pair are less than 1/2 wavelength seperation and that there is some high speed means of switching between these two antennas.

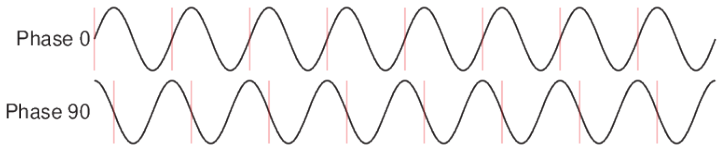


Figure 4.11: Antenna Waveform

We see the RF on the two antennas is displaced by 90°. If we can keep the RF path from these two elements the same to the receiver, we will end up with the following waveform at the receiver when we switch antennas.

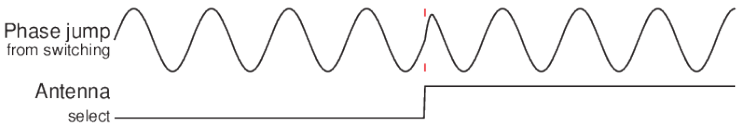


Figure 4.12: Antenna Switching Waveform

You can see that the waveform has been distorted at the switching point.

If we amplify and clip the incoming RF, the change in frequency becomes more apparent.

Figure 4.13: RF Limiter Action

The apparent frequency increases at the middle of the plot as a result of the antenna switch.

With a simple antenna switch (such as the 102-73170-20) we can direction find by ear.

Simply using two antennas fed into a **tee** connector we can achieve reasonable results. If the antennas are equidistant to the receiver, we simply sum the two antenna signals and it acts little different than a single antenna.

Even if one of the antenna elements is 1/4 wavelength further/closer to the transmitter, we are easily able to detect the transmitter.

Figure 4.14: Dual Antenna

The antenna elements need to be seperated by almost exactly 1/2 wavelength or what you hear becomes ambiguous.

Only when the elements are 1/2 wavelength different in distance from the transmitter will the receive signal be nuled out.

Figure 4.15: Dual Antennna, too close

Here, in this example, the elements are too close to see the 180°phase difference. The radio AGC will happily pull the signal out.

There is more useful information in the audio signal from the receive, however, that we can make use of.

By synchronizing the detector logic with the antenna switching logic, we can look a bit deeper and find something useful.

4.2.2 The *Enhanced* TDOA Detector

Given the larger memory footprint with the STM32U5 on the 102-73174-41 board, we can implement the same detection scheme used on the 102-73174-51 board. Here move the analog function into the digital domain.

As we are sampling the antenna switch and the receiver audio, we can narrow the detection window to a little as $\frac{1}{64}$th. of the antenna switching cycle.

Driving the antenna switch at a 1KHz rate, we are sampling audio at 64KHz, so we are free (in the software) to as many samples as needed to detec the antenna switching glitch.

4.2.3 Clock Generator

Master clock generator circuit.

Timing control is the core of what makes our synchronous detector function. We will make use of a simplt decade counter to look at the audio signal only when antenna switching occurs. Two of the ten outputs are selected to sample audio from the receiver.

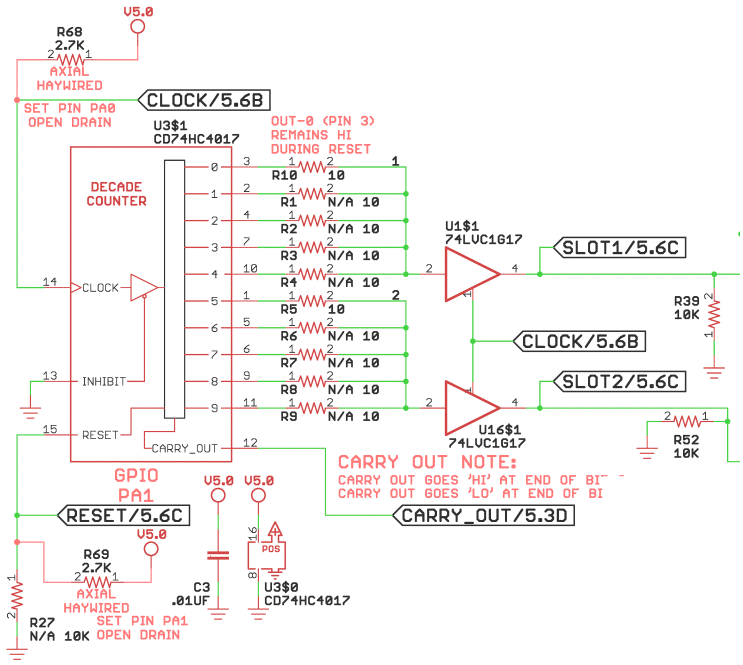


Figure 4.16: B series CMOS Clock Generator

The zNEO uses one of its timers to generate the reference clock supplied on the **CLOCK** net. This is nominally set to 1KHz. The zNEO also controls the **RESET** net.

The 74HC4017 generates a series of pulses on its ten output pins. Two of the ten are selected to produce the **SLOT1** net and the **SLOT2** net.

The **CARRY_OUT** from the 74HC4017 is a aquare ware that drives the antenna switch by selecting between the center element and one of the edge elements, see figure 4.16 on page 16. The **CARRY_OUT** net is supplied to an input pin on the zNEO so that the current state can be monitored.

The two inputs to the 74HC4017 (**CLOCK** and **RESET**) come from the 3.3V domain of the zNEO and need to be shifted to the 5V domain of the 74HC4017. The zNEO has 5V tolerant pins that are programmed as *open drain* outputs. A pair of pull-up resistors to the 5V rail may be haywired in to provide appropriate levels when using a 74HC4017.

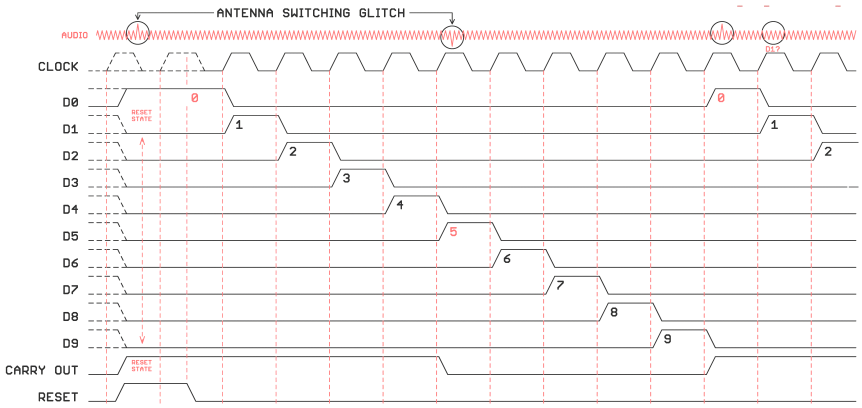


Figure 4.17: Clock Waveform

This is the waveform we expect out of the 74HC4017.

Note that U1 and U16 sit between the 74HC4017 and the **SLOT*** nets. The part number on the drawing is for a non-inverting buffer, which has no effect on the net. The net can be gated with the **CLOCK** net to reduce the sampling time to the integrator. This gating is accomplished by using a tri-state buffer (in the U1 and U16 position) to shrink the sampling period to ½ clock cycle. Use a 74LVC1G126 to select the first half and a 74LVC1G125 to select the second half.

Do take the time to determine the audio response of the radio. When we operate at a 1KHz audio rate, the *glitch* sampling window is only 100uS wide. Thought of in another way, our sampling bandwidth is extremely narrow. You may find it necessary to adjust (i.e. lower) the antenna switching rate in order to see most of the audio glitch at the integrator.

As mentioned a bit earlier, the schematic lists a tri-state buffer between the 74HC4017 (U3 figure 4.16) **SLOT**x select and the input switch (figure 4.20) to the audio integrator. U1 and U16 (also in figure 4.16) when populated as tri-state buffers, will cut the sampling window in half, potentially too short to allow the antenna switching glitch to be observed..

Note that asserting **RESET** to the 74HC4017 will drive the **D0** and **CARRY_OUT** output pins hi. Assuming the radio detects the antenna switch during the **D0** and **D5** interval, the **D0** interval will be selected when the 74HC4017 is held in **RESET**.

Waveform measured on the prototype unit

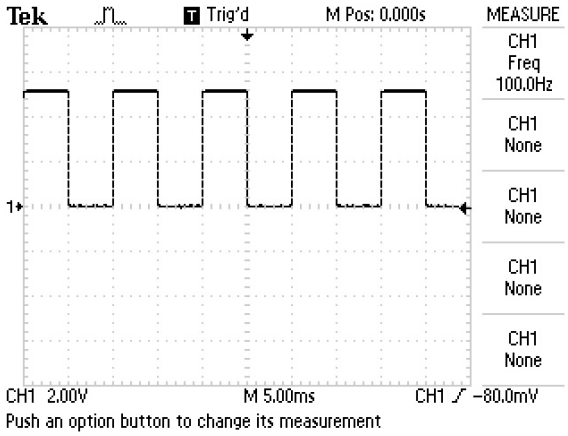


Figure 4.18: 74HC4017 CARRY_OUT

This is the waveform measured on the prototype. This is seen at pin 12 of the 74HC4017.

4.2.4 Audio Chopper

The audio chopper circuit samples audio data for analysis.

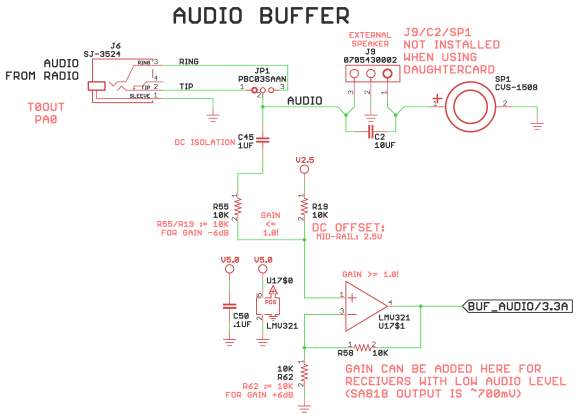


Figure 4.19: Audio input circuit

Audio is taken from the external receiver through a 3.5mm jack (J6) or from the daughtercard through the DB1$1 interconnect. When using the internal receiver, J6 is not populated (mechanical interference issue).

C45 removes any DC offset from the audio. R19 centers the DC voltage to mid-rail. R15, combined with R19 forms a voltage divider resulting in a gain of -6dB.

U17 buffers the audio signal and provides 6dBB of gain so the level seen on the **BUF_AUDIO** net is the same as seend on the **AUDIO** net coming from the receiver.

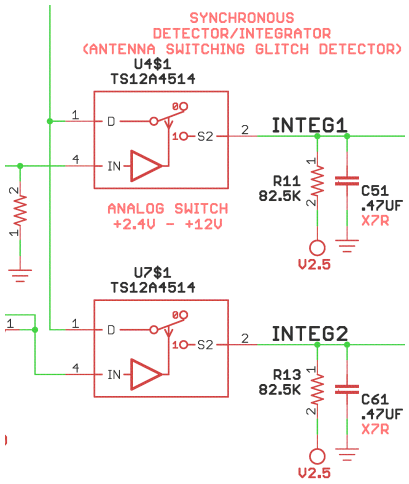


Figure 4.20: Audio chopper circuit

The **SLOT1** and **SLOT2** signals (on pin 4 of the TS12A4514 analog switch) pass audio from the receiver through to a crude integrator. U4 and U7 take the average of the waveform from the radio during **SLOT1** and **SLOT2** and stores the average value on C51 and C61. R11 and R13 allow the integration to decay between sampling periods. See the timing diagram in figure 4.17 on page 17.

The **SLOT1** and **SLOT2** nets are provided to the zNEO so the current system state can be determined.

.

4.2.5 Disintegrator

Sounds catchy?

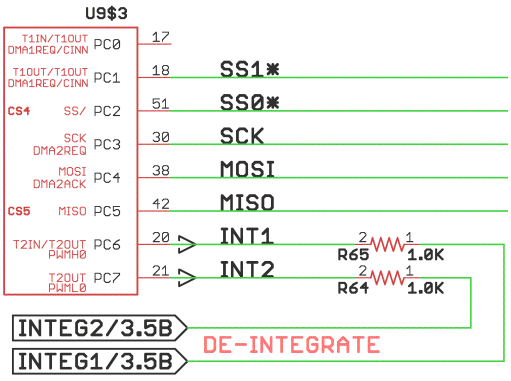


Figure 4.21: De-integrator

Two pins on the zNEO (**PC6** and **PC7**) may be used to clear the integrator nets (**INTEG1** and **INTEG2**) to either 0V or 3.3V.

Clearing to 0V is, perhaps, the simplest case. Here we configure the pin to be open-drain and drive it to logic zero to discharge C51 and C61.

Clearing to 3.3V is a bit more involved in that the output data register is loaded with a '1' and then the data direction register is switched between input and output. The value of R64 and R65 may need to be changed for the clear operation to have the desired effect.

You are, of course, free to treat **INTEG1** and **INTEG2** differently if needed.

4.2.6 Bias Buffer

Mid-rail bias for audio processing.

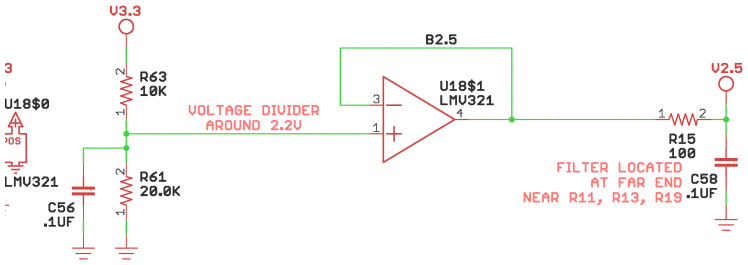


Figure 4.22: Bias Buffer

This voltage is used to de-integrate the **INTEG1** and **INTEG2** nets toward a *no signal* state.

This also provides a DC level for the incoming audio signal in section 4.19 on page 19.

4.2.7 Buffer Amp

Isolate the integrator from downstream stages.

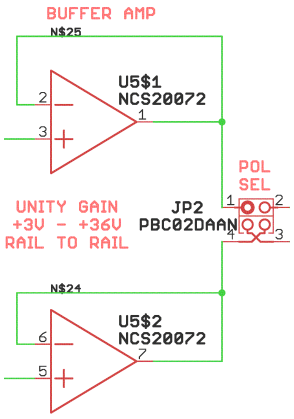


Figure 4.23: OP-Amp Follower

All we do here is present a high impedance to the integrator and a low-impedance to the downstream circuits.

The JP2 jumper array provides a means of swapping the two antenna samples. This provides the capability to always drive the meter in the positive direction.

This will be receiver dependant.

4.2.8 Differential Amplifier

The Differential Amplifier is powered from both a +5V and a -5V rail.

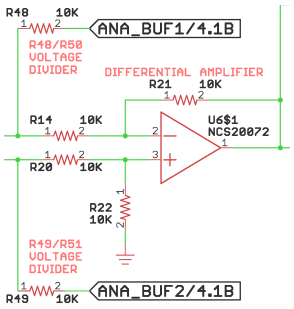


Figure 4.24: Differential amplifier circuit

Now we difference the integrated offsets in the audio signal.

The **ANA_BUF1** and **ANA_BUF2** nets are also sent directly to the zNEO A/D.

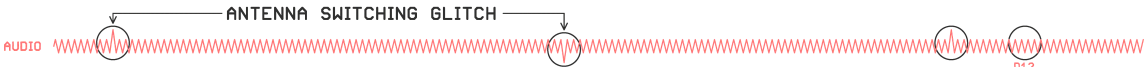


Figure 4.25: Nominal Audio Waveform

With the antenna pair not pointing normal to the transmitter, we expect to get something like the above waveform on the **AUDIO** net. We then attempt to isolate these glitches, caused by our antenna switching, using the U4/U7 switch.
Observe in ficure 4.17 on page 17 that **U3/D0** and **U3/D5** pass the waveform through to the integrator as the antenna element is switched.

.

4.2.9 Offset adjust

U6$1 from the previous stage may well produce a negative voltage, which can be rectified using JP2. U6$1 may also produce a voltage that is out-of-range for a mechanical meter or out-of-range of the zNEO A/D. The gain and offset may be corrected using U6$2.

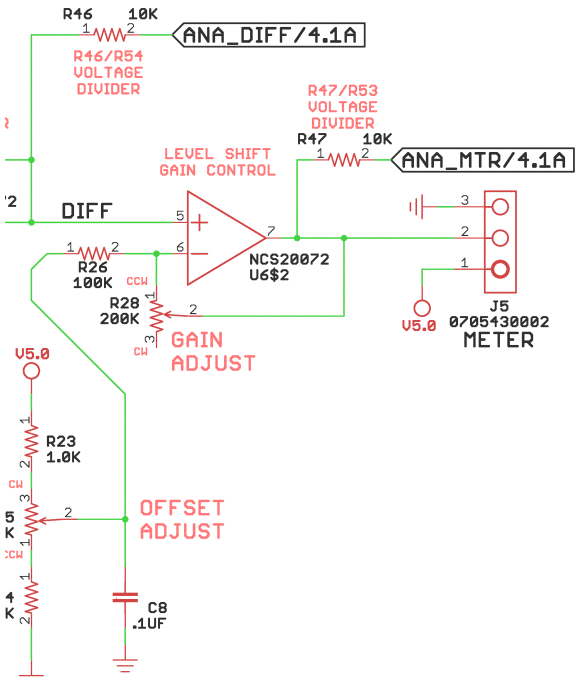


Figure 4.26: Offset adjust

A voltmetxer may be directly attached to J5 to see a real time indication of the difference signal seen by the synchronous detector. This **ANA_MTR** net is also sent to the zNEO A/D section.

4.2.10 zNEO port pin assignments

		BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
102-73174-51	PORT A	CARRY_OUT INTERRUPT	NOT USED OUT	TXD0 UART OUT	RXD0 UART IN	SLOT2 INTERRUPT	SLOT1 INTERRUPT	RESET OUT	CLOCK T0OUT
102-73174-51	PORT B	BMON ANALOG IN	IMON ANALOG IN	ANA_MTR ANALOG IN	ANA_DIFF ANALOG IN	5MON ANALOG IN	ANA_BUF1 ANALOG IN	ANA_BUF2 ANALOG IN	SW_P ANALOG IN
102-73174-51	PORT C	INTEG2 OPEN DRAIN	INTEG1 OPEN DRAIN	MISO ESPI IN	MOSI ESPI OUT	SCLK ESPI OUT	SS0* OUT	SS1* OUT	NOT USED OUT
102-73174-51	PORT D	NOT USED OUT	NOT USED OUT	TXD1 UART OUT	RXD1 UART IN	NOT USED OUT	NOT USED OUT	SQL IN	NOT USED
102-73174-51	PORT E	LED8 OUT	LED7 OUT	LED6 OUT	LED5 OUT	LED4 OUT	LED3 OUT	LED2 OUT	LED1 OUT
102-73174-51	PORT F	NOT USED OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT
102-73174-51	PORT G	NO PIN OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT	LED9 OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT
102-73174-51	PORT H	NO PIN OUT	NO PIN OUT	NO PIN OUT	NO PIN OUT	NOT USED OUT	SEL_A2 OUT	SEL_A1 OUT	SEL_A0 OUT

Figure 4.27: zNEO port pin assignments

4.3 Antenna Rotator

Source File: FOX_zNEO_Theory_Operation_3.tex

This the the antenna switching matrix for a nine element antenna array.

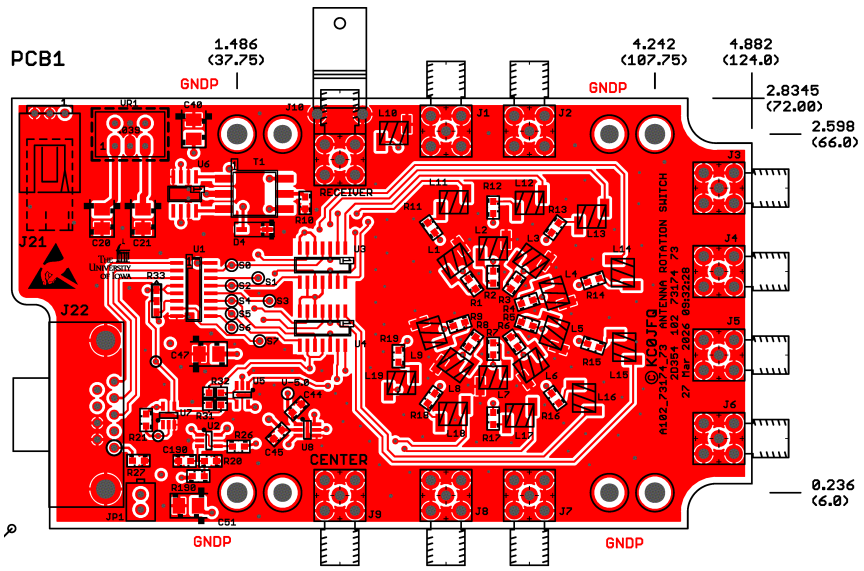


Figure 4.28: 1st. generation TDOA Rotator (102-73174-73)

The system assumes a semicircular array of eight radial antennas and one center antenna. This circuit board is used to electrically rotate an antenna array by switching between a common antenna element and one of eight radial elements.

4.3.1 Diode Switching

The RF switch.

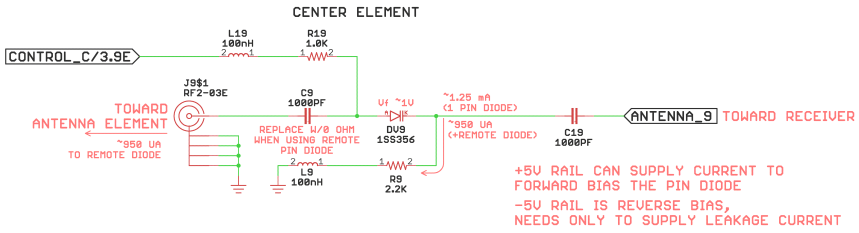


Figure 4.29: Diode Switching

The PIN diode is used as the RF switch.

When forward biased, the diode acts much like a low-value resistor to the incoming RF signal. When reverse biased, the diode acts much like a low-value capacitor to the incoming RF signal.

The capacitors provide DC isolation, allowing only the RF signal to pass.

The capacitor leading to the antenna element may be replaced with a 0Ω resistor when using the 102-73170-32 *PIN Diode Antenna Base*. The 102-73170-32 board is also fitted with the same PIN Diode and ground bias to allow additional isolation.

4.3.2 Diode Bias Calculations

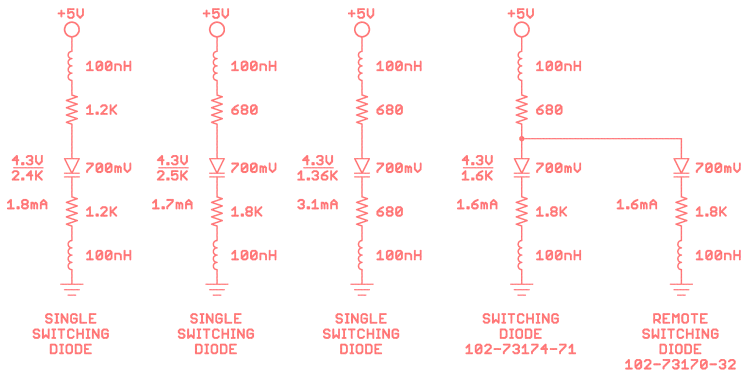


Figure 4.30: Diode Bias Calculations

On the left is the nominal configuration when using only the switching matrix on the 102-73174-73 circuit board. Here a single PIN diode is used as the RF switch. Calculated forward current is, as shown, about 1¾ milliamp.

On the right is shown with the 102-73170-32 board populated with a second PIN diode to further improve isolation inthe system. The additional diode electrically appears as shown on the right side. The overall current through the two diodes rises but this individual currents drops a bit to about 1⅓ milliamp.

Reverse bias, requiring very little current, will be at the negative rail esentially unaffected by the resistors. The negative rail is around -5V.

4.3.3 Diode Bias

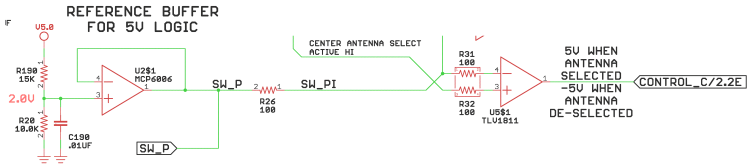


Figure 4.31: Diode Bias

The switching bias is produced using a comparator that is powered from 5V and -5V. The digital control is 5V logic which is routed to the - input of the TLV1811. The + input of the TLV1811 comes from a buffered voltage divider that is set to approximatley 2V.

The TLV1811/TLV1814 are all powered from +5V an -5V to provide a convenient biasing scheme for the PIN diodes.

4.3.4 Antenna Decode

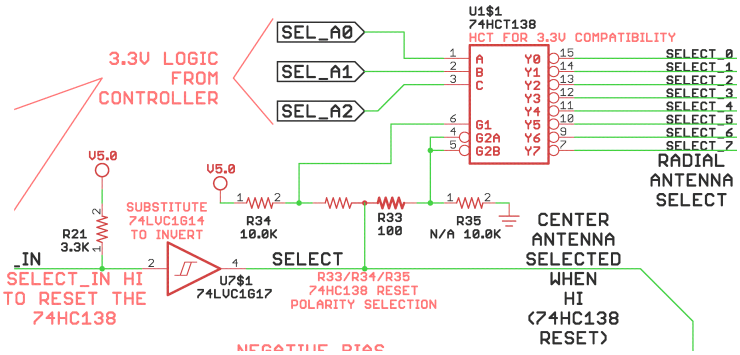


Figure 4.32: Antenna Select

The host controller (see section 4.1 on page 8) supplies the four select signals to the 74HC138.

The -73 revision adds R33, R34, and R35 to allow the antenna select to enable the center element when the **CARRY_OUT** net is held in a reset state.

4.3.5 Antenna Control Timing

Target timing to the antenna switch.

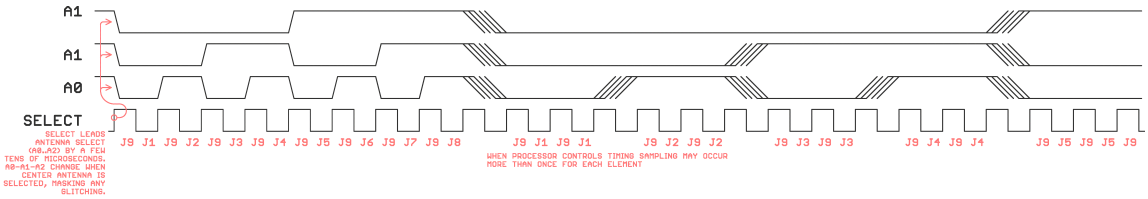


Figure 4.33: Antenna Control Timing

The hardware (i.e. the CD4017 or the 74HC590) generates the **SELECT** net with a 50% duty cycle.

The processor on the main board is then responsible for driving the 3 select lines (A2..A0) to the 3:8 decoder. Although figure 4.33 shows A2..A0 being synchronous with the **SELECT** net, in practice the zNEO/STM32 may sample multiple times before changing the A2..A0 nets to the next antenna.

When the **SELECT** net is **lo** one of the eight radial antennas is selected. When the **SELECT** net is **hi** the center antenna is selected. The **SELECT** net idles **hi** when the 74HC4017 is held in **RESET**. The center element is selected when the 74HC4017 is idle. We are not so fortunate with the 74HC590 with respect to the reset.

The controller electronically rotates the array by switching between the center element and one of the eight radials. The audio signal can then be analyzed as we switch between the center element and the selected radial element. The controller then rotates on to the next antenna element and we repeat the analysis.

.

.

.

4.3.6 Antenna Array

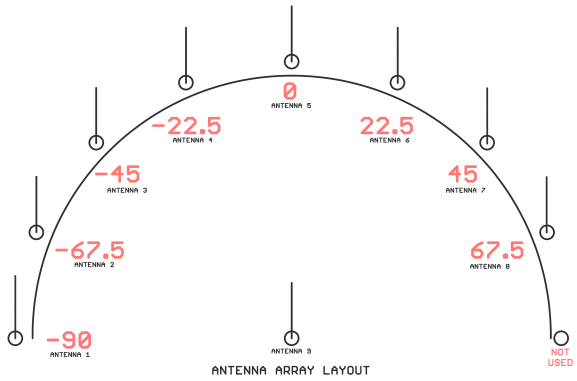


Figure 4.34: Antenna Array

Here is a suggested layout for an eight+one antenna array. This gives us the ability to rotate the array through 360°in 22.5°increments.

The operating software in the 102-73174-51 control board is then free to synthesize almost any pattern.

Once the receiver is characterized, the full 360°pattern can be derived from this antenna layout.

4.4 SA818 Receiver

Source File: FOX_zNEO_Theory_Operation_4.tex

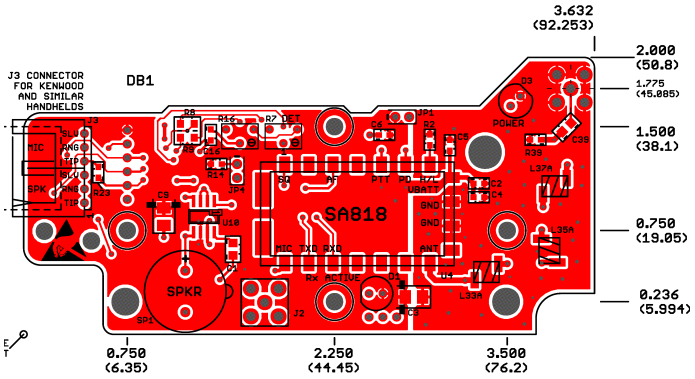


Figure 4.35: 1st. generation TDOA Receiver (102-73174-91)

This receiver board is for use with the antenna rotator switch (102-73174-73).

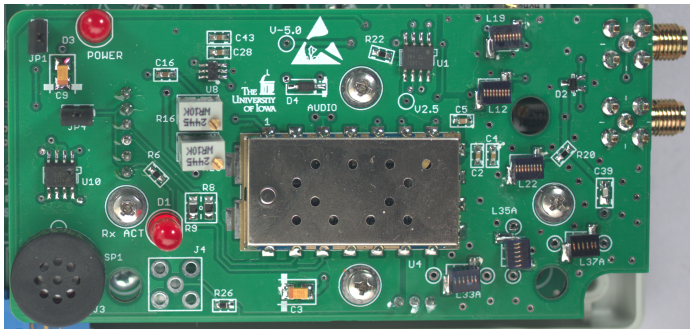


Figure 4.36: 2nd. generation TDOA Receiver (102-73174-97)

This receiver board may be used stand-alone with two antennas or with the antenna switch (102-73174-73).

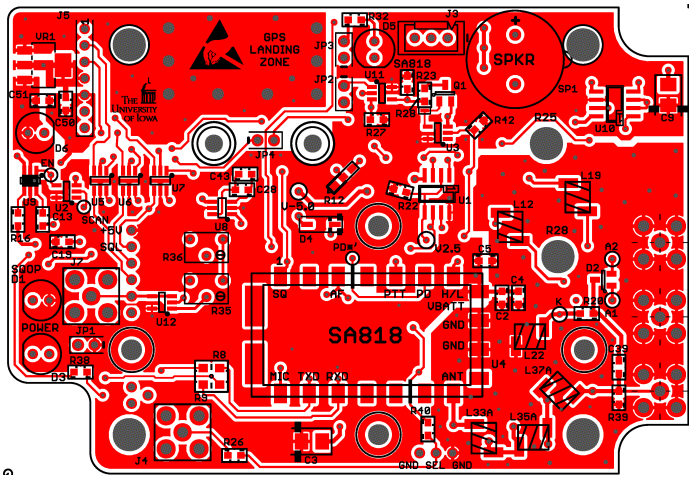


Figure 4.37: GPS TDOA Receiver (102-73174-99)

Add in GPS module for time. This daughtercard covers the entire height of the main board. There are provisions for six spacers to affix the daughter card. Additional access holes are provided to get at the main board mounting screws as well as access to the potentiometers on the 102-73174-51 main board. The level adjust pot is inaccessible on the 102-73174-41 main board.

The lower left corner is notched out to allow access to the zNEO programming header with the daughter card installed.

4.4.1 Receive-only configuration

The receiver section performs the function of a handheld tranceiver. All that is required of the receiver is to listen to the Fox Transmitter and pass the audio back to the synchronous detector.

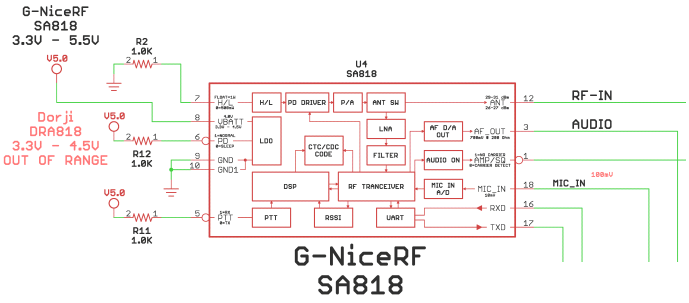


Figure 4.38: SA818 Receiver

The receiver is strapped into a **receive only** mode of operation. There is no mechanism to place the SA818 into a low power mode, it is always listening for radio traffic on the selected frequency. The carrier detect line is passed back to the main board to allow the zNEO to determine when radio traffic in present.

.

4.4.2 Low Pass Filter

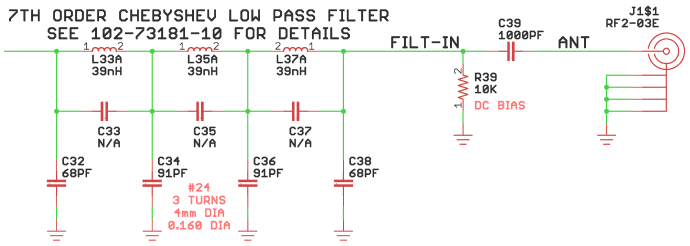


Figure 4.39: Low Pass Filter

The SA818 requires an external low pass filter to deal with out-of-band signals. We copy the filter topology from the Fox Transmitter. The initial values on the schemticallow the filter to operate only as a low-pass.

.

4.4.3 Audio Amplifier

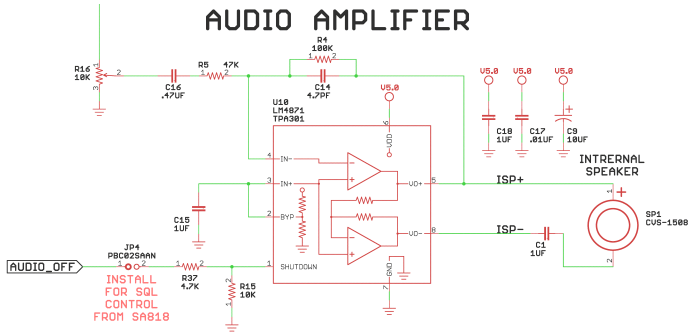


Figure 4.40: Audio Amplifier

We add a minimal audio amplifier to drive a speaker.

Not very loud!

4.4.4 GPS Logic

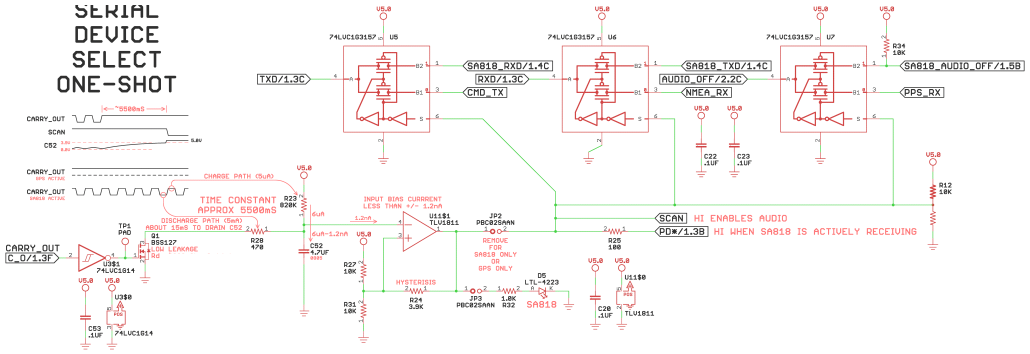


Figure 4.41: Logic Select

The incoming select signal (**CARRY_OUT** from the main board gets renamed **C_O** on this schematic fragment) is buffered by U3/Q1. Q1 must be a low leakage device to avoid draining C52 while also providing adequate current sinking capability to discharge C52. When actively switching antennas, C52 is kept in a discharged state (U3/Q1 driving low) which leaves the output of U11 high. U11 high connects the SA818 signals to the main board.

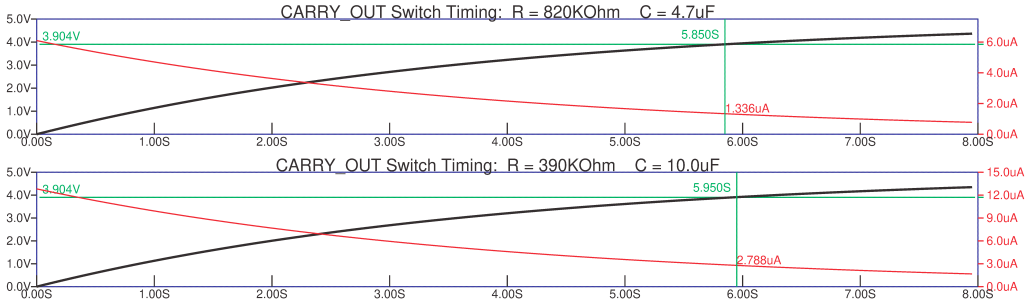


Figure 4.42: C52 charging through R23

When U3 is hi, C52 charges toward the supply rail through R23. Once the voltage on C52 arrives at the comparator switching point the main board signals are switched from the SA818 to the GPS Module. If the 74HC4017 on the main boad is held in RESET, U3 no longer pulls charge from C52.

The plot also shows the calculated charge current into C52. The capacitor is still drawing more than 1 uA of current when the TLV1811 trips which is three orders of magnitude greater than the input leakage current of the TLV1811.

Leakage current on the drain of Q1 may also be an issue that may be addressed by increasing the size of the capacitor and lowering the value of R23. This is intended to increase the charge current being delivered to C52 to swamp out the leakage current through Q1.

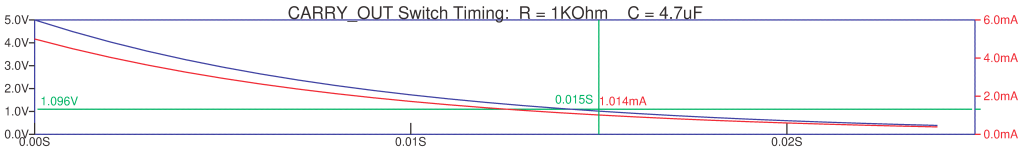


Figure 4.43: C52 discharging through R28

Once we activate the scanner (ie. supply clock to the 74HC4017), U3 will be pulling charge from C52 through R28. Keeping in mind that the **CARRY_OUT** signal from the 74HC4017 is a square wave, we take twice the calculated R/C time to drain C52. Also bear in mind that the current calculared here is the discharge current only when the **CARRY_OUT** signal is low.

At startup, the **CARRY_OUT/C_O** signal is nominally inactive (i.e. high) allowing C52 to charge and eventually cause the output of U11 to go low. With U11 low, the GPS is powered and connected to the main board.

When switching the antenna array the software may, for a short period, disable antenna switching while still having the SA818 remain connected. This allows the antenna switching to be suppressed for a shot period of time.

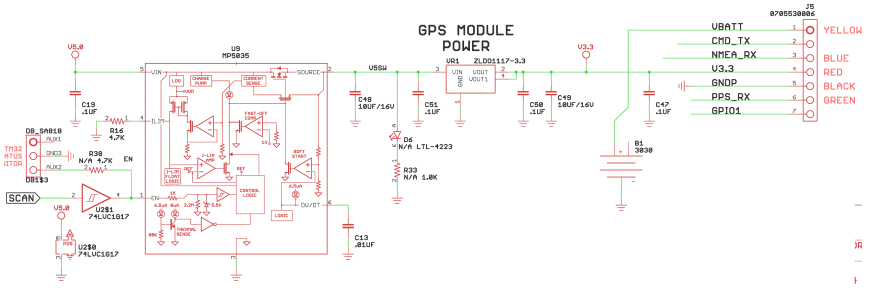


Figure 4.44: GPS Connector and Power

The GPS is only powered when it is connected to the main board. This is intended to keep power drain to a minimum, but it does slow the system down when collecting time from a GPS module.

U2 presents the correct polarity to the power switch, U9. U9 provides a soft-start capability to reduce impulse loads onto the main board power supply. C13 controls the soft-start time. U9 also has a current limit capability which is controlled by R16.

The GPS Module specified in the parts list requires a 3.3V supply which is managed by VR1. The artwork also allows provisioning a battery ackup for the GPS to improve startup time.

Note that 3 signals are routed from the GPS connector back to the main board. TxD, RxD, and PPS. The GPS Module specified in the parts list does not provide a TxD pad, so it would not be wired to the GPS connector.

We can add a GPS module to track time.
Potentially being able to track the schedule of the Fox Transmitters.
The physical connector matches mechanically and uses the same pinout as the Fox Transmitter duaghter cards.

4.4.5 Main Board Interconnecst

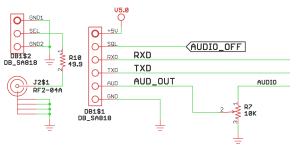


Figure 4.45: Main Board Interconnecst

Power, Ground and signals to and from the main board.

4.4.5.1 AUDIO_OFF

Status signal from the SA818 that indicates when there is no carrier present.

4.4.5.2 AUD_

Demopdulated audio from the SA818.

4.4.5.3 RXD

Serial data from the command interface of the SA818.

4.4.5.4 TXD

Serial data to the command interface of the SA818.

4.4.6 Antenna Switch

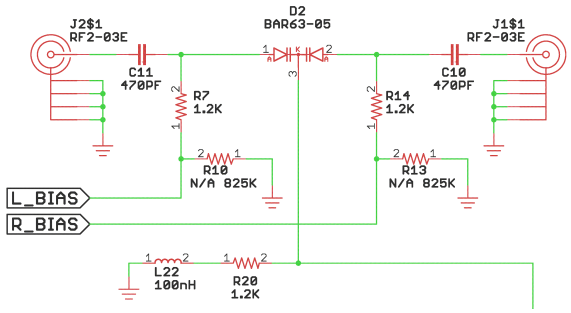


Figure 4.46: Antenna Switch

The antenna switch is borrowed (somewhat) from the 102-73170-20 design with minor changes. Note that R10/R13 are on the *more isolated* side of R7/R14 and are no longer necessary. This location keeps this ground reference a bit further away from the RF path between antennas and radio.

This drawing is from the 102-73174-97 update which reworks the PIN diode drive to match the 102-73174-73 model.

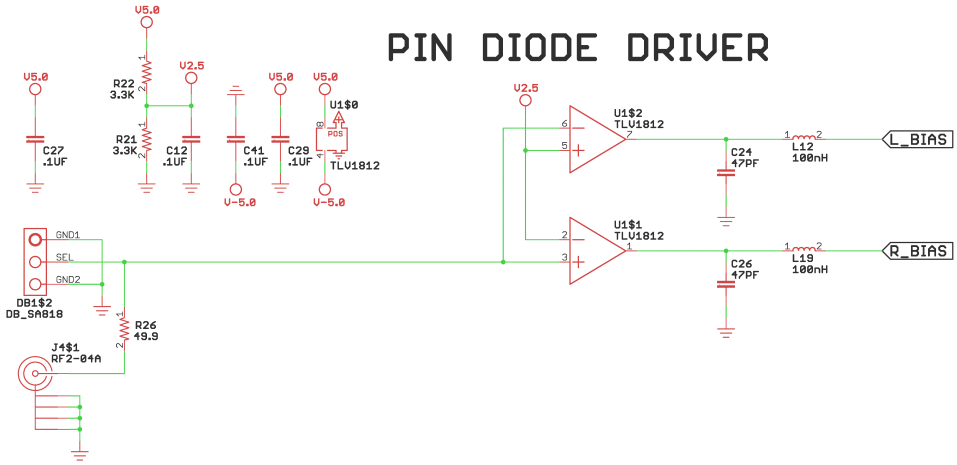


Figure 4.47: Next

The antenna switch driver matches the method used on the 102-73174-73 board. The 102-73174-73 board provides for forward bias from the +5V supply while reverse bias comes from the negative supply. This topology reduces the current demand on the negative supply to that required to reverse bias the PIN diodes (i.e. the -5V rail supplies only leakage currdent to the diodes).

The driver (TLV1812) is a simple analog comparator that is powered from the +5V rail and then -5V rail. We then use a simple resistor divider to generate roughly 2.5V to compare the incoming clock against.

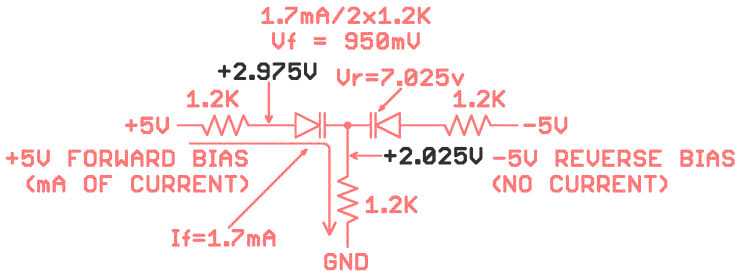


Figure 4.48: Biasing Model

The PIN diode biasing calculation model. (5.0V-0.95V)/2.4KΩ gives roughly 1.7mA

The commmon cathode point is connected (through an RF choke and resistor) to ground. The two anodes are switched between the positive and negative supplies.

Assuming the current limiting resistors are the same value, the conducting diode will be sitting midway between the +5V rail and ground. Given a roughly 950mV drop, the diode anode and cathode of the forward biased diode will be at the indicated voltages.

The other diode in the pair will be reverse biased as the negative supply is gated on as shown. Since no current to speak of flows in the reverse direction, that diode will see a reverse voltage of just over 7 volts.

Although we could have a lower negative voltage (the 102-73174-82 board can produce lower negative rail) the -7V reverse bias puts the PIN diode into a very low capacitance high isolation region of operation.

The BAR63 series datasheet indicates a capacitance of about .2pF with a reverse bias of -5V. The datasheet indicates an RF impedance of near 500KΩ at the same -5V bias. The inseertion loss at -5V is shown as in excess of -20dB.

Driving around 2mA of forward current shows an RF impedance of less than 2Ω.

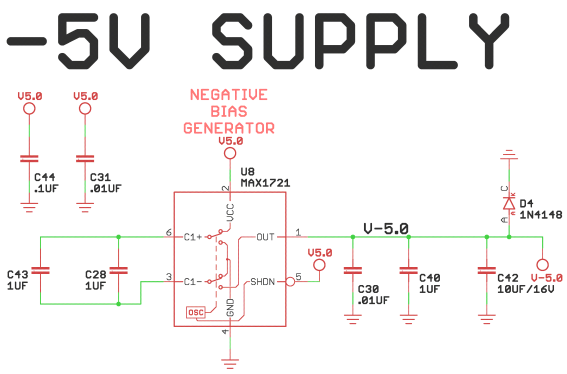


Figure 4.49: Next

The -5V supply, a simple switched capacitor device that can supply over 10mA. This supply rail need only supply leakage current during steady state operation, well within the capability of a switched capacitor topology. The switching load will also be quite low as we are operating at audio frequencies, again well within the capability of this switched capacitor topology.

The 102-73174-97 and 102-73174-82 artwork both include a more substantial negative supply. This is a more expensive approach that isn't necessary as there is no current running through the -5V supply.

4.5 Dual Antenna Switch

Source File: FOX_zNEO_Theory_Operation_5.tex

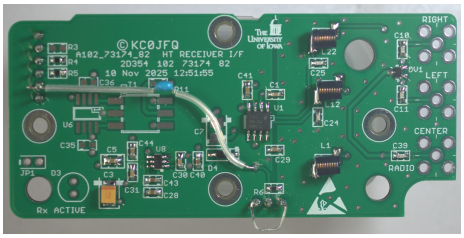


Figure 4.50: Dual Antenna Switch (102-73174-82)

This board provides a basic two antenna switch along the lines of the 102-73170-20 board but clocked by the zNEO on the main board.

There are two antenna element connectors, labeled ***RIGHT*** and ***LEFT***, and one radio connector labeled ***CENTER***.

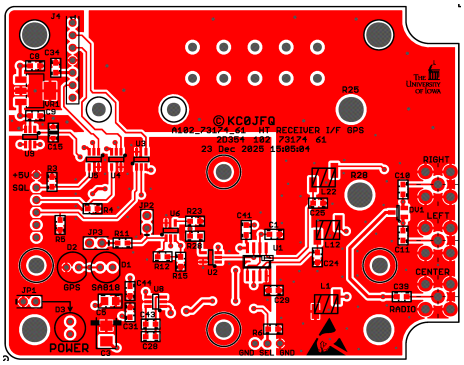


Figure 4.51: Dual Antenna Switch w.GPS (102-73174-61)

The addition of a GPS module is accomplished using the antenna switching logic from the 102-73174-82 board and adding provisions for mounting a GPS module onthe daughtercard.

The GPS steering logic from the 102-73174-99 board is duplicated on this board (see figure 4.4.4 on page 33).

Although we don not have a tranceiver module here that needs access to the szerial connection, the steering logic makes this board act in the same manner (with respect to accessing GPS daya).

PIN diode drive

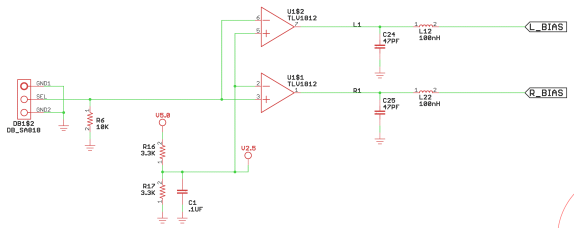


Figure 4.52: PIN Drive Schematic (102-73174-82)

Simple bipolar drive using an analog comparator powered from the +5V rail and the -5V rail.

This configuration has one diode forward biased and the other reverse biased. We receive a 50% duty cycle square wave on the **SEL** net coming in through the **DB1$2** connector.

TLV1812 output

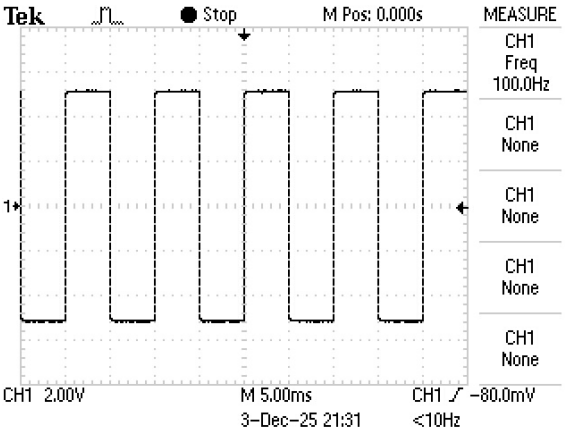


Figure 4.53: TLV1812

This is the waveform measured on the prototype 102-73174-82 daughter card. This is seen at pin 1 and pin 7 of the TLV1812.

PIN diode antenna switch

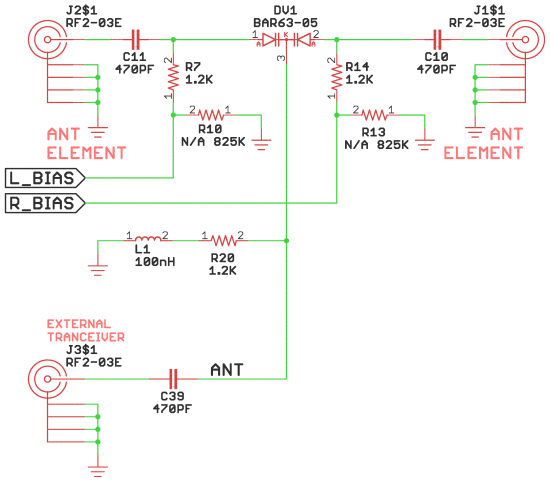


Figure 4.54: PIN Diode Schematic (102-73174-82)

Same PIN diode switch used everywhere. **L_BIAS** and **R_BIAS** are always the opposite polarity.

PIN Diode Anode drive

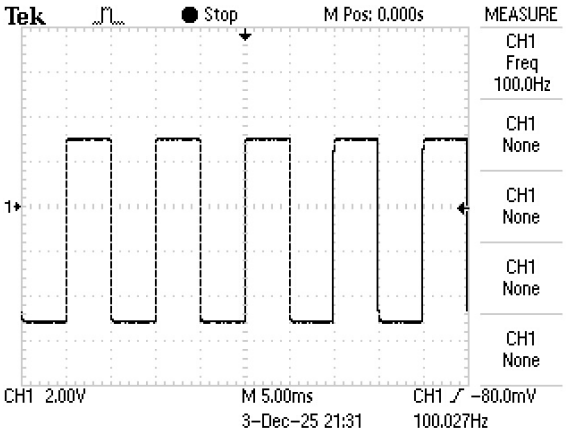


Figure 4.55: PIN Diode Anode

This is the waveform measured on the anodes of DV1 on the 102-73174-82 daughter card.

PIN Diode Cathode drive

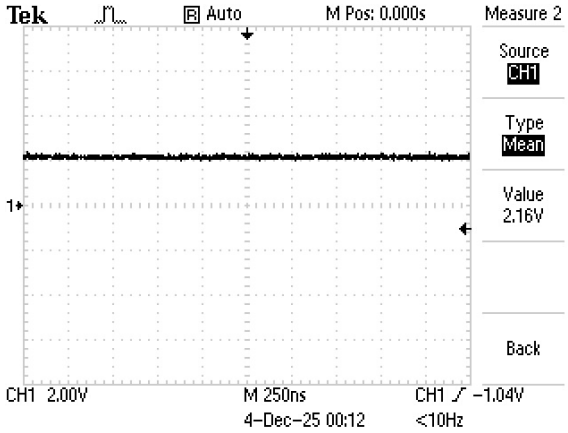


Figure 4.56: PIN Diode Cathode

This is the waveform measured on the cathode of DV1 on the 102-73174-82 daughter card.

The anodes are (continuously) driven, of course, with opposite polarity signals from the TLV1812. The cathode settles at this 2V level due to the current limiting resistors (R7, R14, and R20 in figure 4.54).

Note that the o-scope zero is mid-screen for both captures. We see the voltage drop across R7/R14, the forward biased diode, and R20. The diode drop will be shy of 1000mV when forward biased . We also see the full (-5V) negative bias on the other cathode. The common cathode remains at a steady voltage of just above 2V.

Thus, the negative 5V on one of the anodes and the steady 2V on the common cathode, results in a reverse bias of just over 7V.

PIN diode switching time

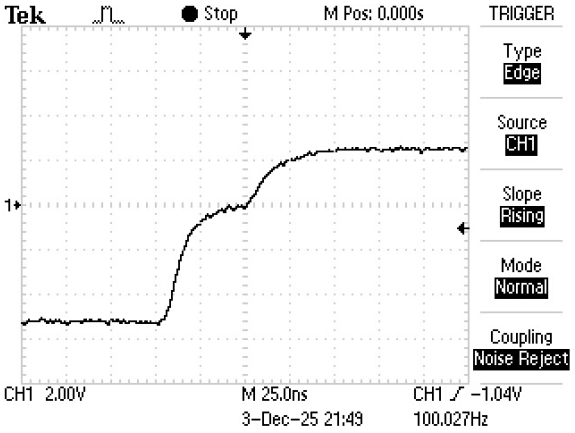


Figure 4.57: PIN Diode switching

This is the waveform measured on the anodes of DV1 on the 102-73174-82 daughter card. The horizontal division is 25nS.

The resistors shown in the schematic do not drive the PIN diode very hard. We could improve the switching waveform a bit by reducing the value of R7/R14 along with R20. The 100nH inductor blocks the RF, so we can reduce the R7/R14 pair to a lower value than R20 and raise the reverse voltage when the PIN diode is switched off. Also bear in mind that the 100nH inductors also affect switching speed.

Increasing the current through the PIN diode will improve turn-on time as well as turn-off time. Increasing current by lowering the resistors also drains charge of the junction when the diode turns off.

Negative Supply

when populatig the 102-73174-82 board, choose only one -5V supply to populate!

Two variants appear on the board, one being able to produce a more negative value than the other. The less expensive path, the MAX1721 (or the MAX87, TPS60400, RS6903, LM828) provides sufficient drive current for proper operation.

Also have a look at the 102-73174-73 schematic for charge pumps that are less expensive than the MAX1721. The listed substitutes are supplied in a SOT32-5 package that is footprint compatible with the MAX1721. The less expensive substitutes lack the enable pin (which isn't used here).

-5V supply

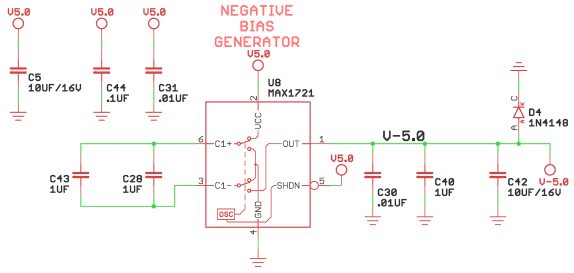
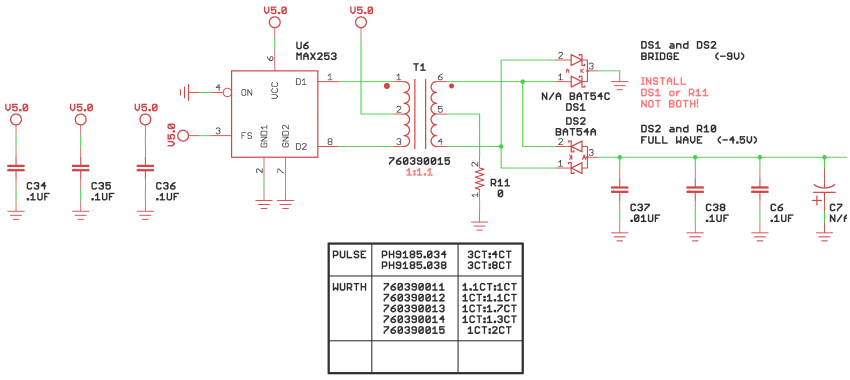


Figure 4.58: Switched Capacitor Convertor (102-73174-82)

Simple switched capacitor design taken from the MAX1721 datasheet (or the MAX87, TPS60400, RS6903, LM828).

The negative supply provides reverse bias for the PIN diode with no current demand. The switched capacitor design easily provides the current spikes that occur during switching of the PIN diode.

-9V supply



PULSE	PH9185.034 PH9185.038	3CT:4CT 3CT:8CT
WURTH	760390011 760390012 760390013 760390014 760390015	1.1CT:1CT 1CT:1.1CT 1CT:1.7CT 1CT:1.3CT 1CT:2CT

Figure 4.59: Transformer Convertor (102-73174-82)

A higher reverse voltage on the PIN diode seems not to be required. This -9V supply can be left unpopulated.

4.6 Antenna Base

Source File: FOX_zNEO_Theory_Operation_6.tex

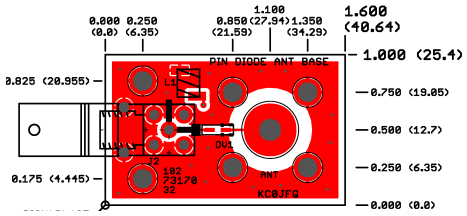


Figure 4.60: Antenna Base (102-73170-32)

The antenna base provides a convenient means of mounting an an ad-hoc antenna element. All the board provides for is a connection point for cables back to the antenna switching box.

The hole patterns will also allow for a right angle BNC. In either case, short spacers are needed to clear the solder joints.

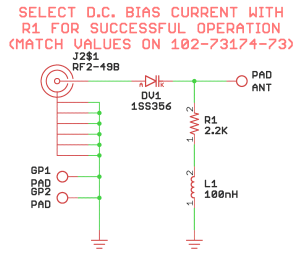


Figure 4.61: Antenna Base Schematic (102-73170-32)

The board also provides for mounting a PIN diode to improve isolation should the need arise. In most cases, the DV1 position is shorted and the resistor and inductor are left off.

The interconnect cable, on the left, carries the DC bias to the PIN diode DV1. To bias the diode on, approximately 5 volts must be supplied. To bias the diode off, a negative voltage is delivered. In the case of the antenna rotator, up to -9V may be provided.

Current Limiting Resistor Model

The value or R1 will need to be recalculated, along with the grounding resistors on the 102-73174-73 boards (corresponding to R9 in figure 4.29).

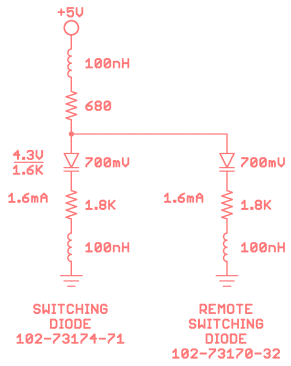


Figure 4.62: Current Limiting Resistor Model

Figure 4.62 roughly represents two PIN diodes that are forward biased for the purposes of calculating the values of the three current limiting resistors. The two bottom resistors will typically be the same value to drive similar currents through the two PIN diodes.

When the PIN diodes are reverse biased, no current flows, to the value of the resistors has little effect.

4.7 Battery Operation

Source File: FOX_zNEO_Theory_Operation_7.tex

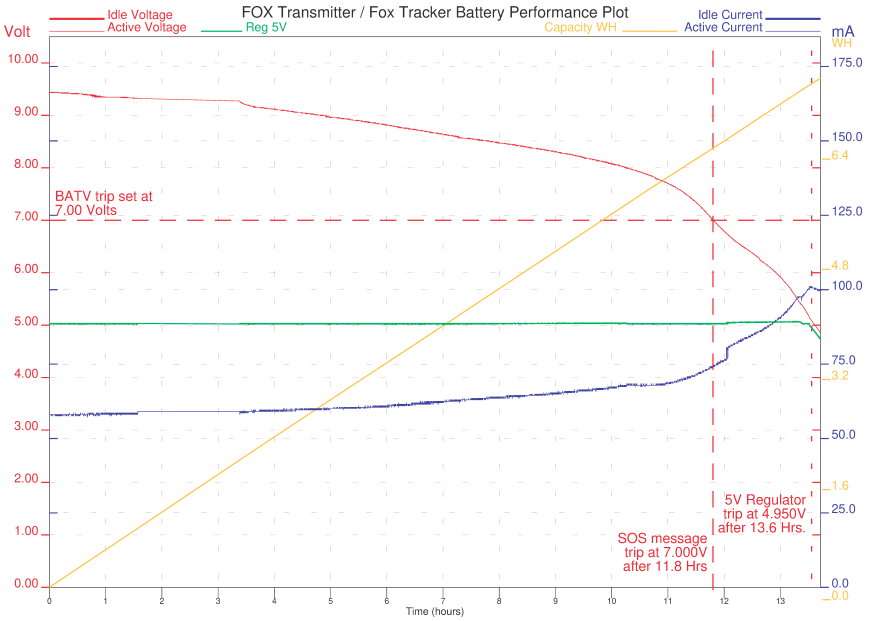


Figure 4.63: Battery Performance

First run of the **BATR SCAN** command to evaluate battery performance. This is similar to the function described in the Fox Transmtter documentation. Here we don't operate in multiple modes as we are simply listening all the time.

This is a test run on an 8-cell AA pack that was partially discharged.

4.8 STM32 update

Source File: FOX_zNEO_Theory_Operation_8.tex

4.8.1 STM32 main board

With the zNEO being retired from production, there comes a need to switch to a more modern SOC device. ST Micro provides a large *universe* of 32 bit ARM based SOC devivces. These devices cover the peripherals provided in the zNEO while considerably expanding the available memory, both Flash and SRAM. The STM32 series of devices also are capabale of running at much higher clock speeds than the zNEO in order to provide processing power when the need arises.

Several members of this series also provide hardware floating point. These additions open the path to providing a more robust detection methodology. What is envisioned here, is moving to a more software intensive detection system.

Consider performing a Fourier Transform on the received waveform to filter out all the unwanted content from the imcoming audio signal. We can also lock the sampling frequency to the antenna switching frequency.

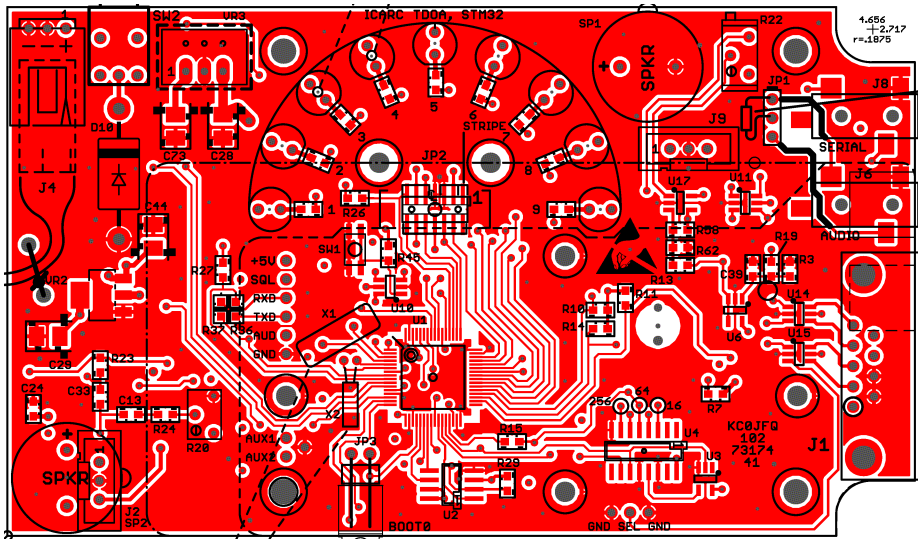


Figure 4.64: 2nd. generation TDOA Tracker (102-73174-41)

The processing element is updated to make use of the STM32U5 series of SOC devices.

4.8.2 STM32 Waveforms

Preliminary testing makes use of a development board from STMicro that allows software development while waiting for thed boards to finish up design and go on to board fabrication.

Refering ahead to the sampling logic in figure 4.80, note that both the audio signal from the receiver as well as the antenna switching signal are digitized. The STM32 is configured to sample these two channels at nearly the same time and at the same sample rate. Also keep in mind that the digitization rate directly determines the antenna switching rate through the the 75HC590 counter.

The size of the waveform buffer is chosen to redeuce the bandwidth of the frequency bins. Most of the spectrum is of little interest to the analysis since we are sampling synchronously. This synchronous sampling tells us exactly which bin will contain energy caused by the regular antenna switching.

We may also choose to operate the antenna switch at multiple frequencies to move the background noise to differing position in the spectrum. Our switching signal, as it is synchronous, always appears in the same bin. The specific bin number being a byproduct of the number of samples in the waveform buffer.

Changing sampling frequency without changing the number of samples leaves the bin we are looking at for switching noise un-shifted.

Start by considering a simulation of what the radio might hear in typical operation. Background noise is variable and the Fox Transmitter may be sending voice traffic, CW traffic, or it may be temporarily silent.

In these example plots, a 1.7KHz sine wave is present to simulate the background signal, synchronized with the audio sampling.

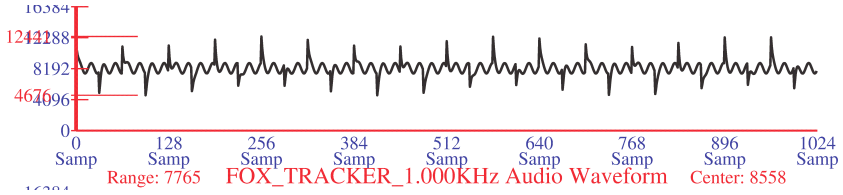


Figure 4.65: Simulated Audio Channel, sample rate 1.000KHz

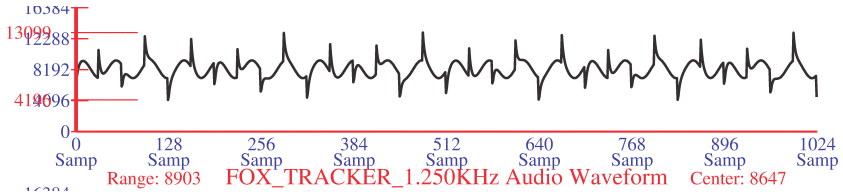


Figure 4.66: Simulated Audio Channel, sample rate 1.250KHz

147

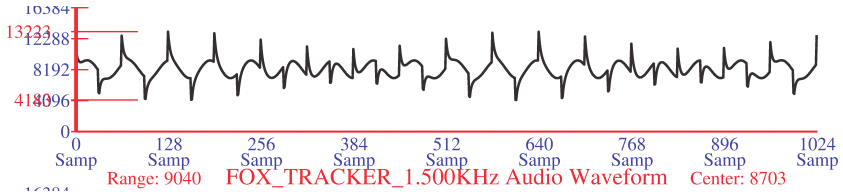


Figure 4.67: Simulated Audio Channel, sample rate 1.500KHz

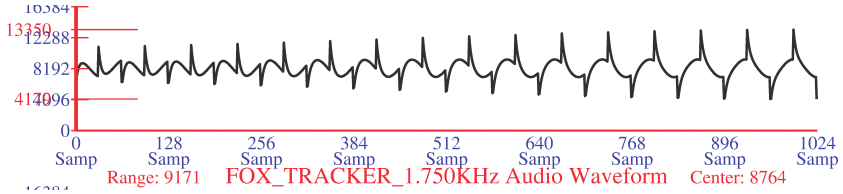


Figure 4.68: Simulated Audio Channel, sample rate 1.750KHz

The antenna switching *interference* appears at the same regularly spaced intervals throughout the audio signal. The audio signal, a 1.7KHz sine wave, appeats to change frequency as the sampling rate changes. These panels taken 2026-APR-23 used a 102-73174-99-STIM board to simulate antenna switching. The the audio input on the 102-73174-41 comes from a signal generator. The signal generator is set to 1.7KHz.

Then observe the (very well correlated) antenna switching signal:

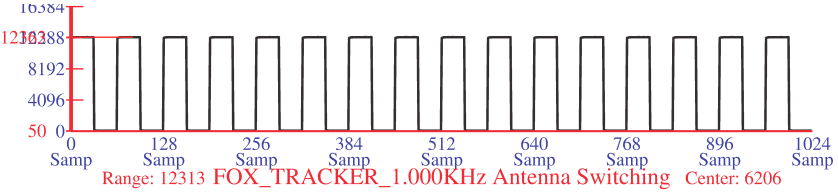


Figure 4.69: Antenna Switching Channel

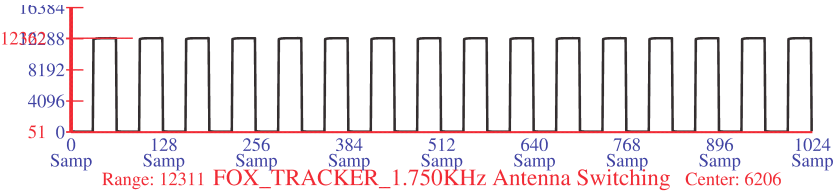


Figure 4.70: Antenna Switching Channel

The antenna switching signal, of course, aligns with the switching *interference*. The antenna switching net is directly used to generate the simulated antenna switching interference.

These two samples from the prototype unit appear to be somewhat phase locked. There is no need to lock the start of sampling to any particular point in the antenna switching cycle. This approximately occurs as the STM32 only enables the sample clock during a data seet collection cycle. The MR net that resets the 74HC590 is also asserted between collection cycles.

The antenna switching signal is somewhat phase-aligned with the data-set. There are several samples discarded at the begining of each data set that is collected.

Setting the number of these discarded samples to a vlaue of 64 serves to better align audio samples with the antenna switch.

Another approach is to set the number of extra samples (at the begining) to 136 and then search, starting at sample 8, for the rising edge of the antenna select signal and then extract the data buffer from that point.

By performing an analysis on the Antenna Switching Signal we see that the primary peak lands in the 32nd. bin as that is the antenna switch frequency:

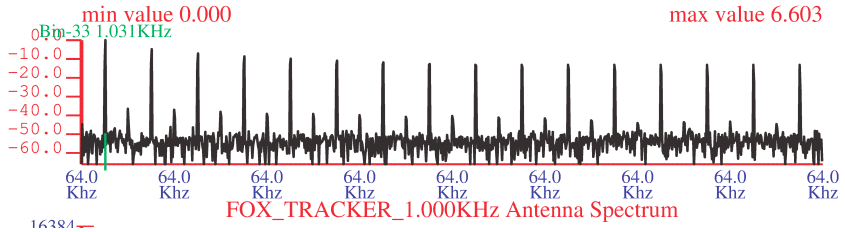


Figure 4.71: Antenna Switching Spectrum

222

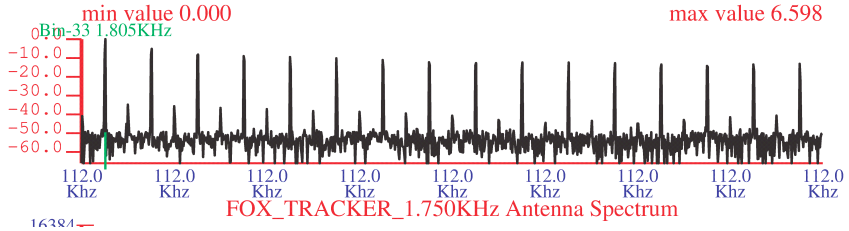


Figure 4.72: Antenna Switching Spectrum

Here we have two data sets taked at the indicated sntenna switching rate. The spectrum is, of course, typical for a square wave signal; the fundamental and odd harmonics right up the band. And, due to the system being synchronous with itself, the spectrums takes at differing sample rates show the same spectral lines.

We also performin this FFT analysis on the Audio Signal:

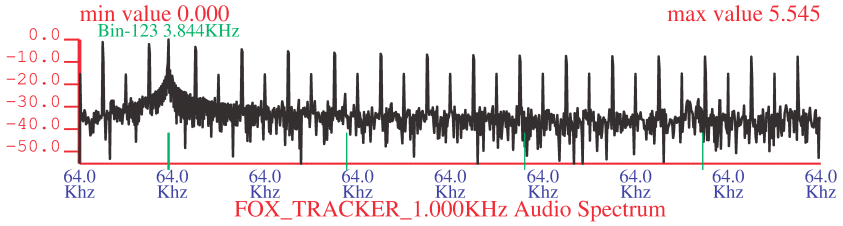


Figure 4.73: Audio Spectrum sampled at 64KHz

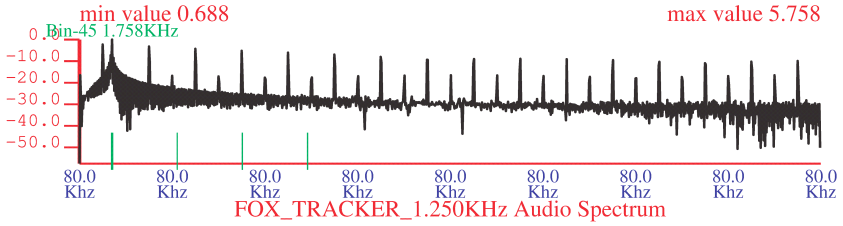


Figure 4.74: Audio Spectrum sampled at 80 KHz

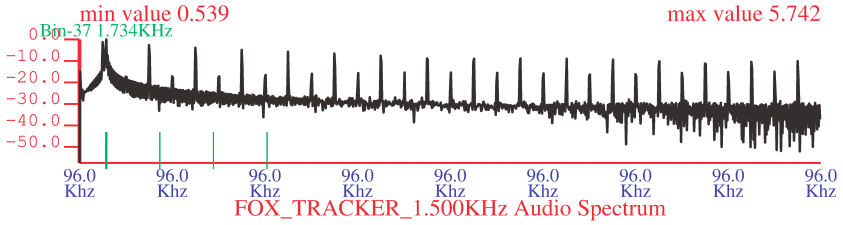


Figure 4.75: Audio Spectrum sampled at 96 KHz

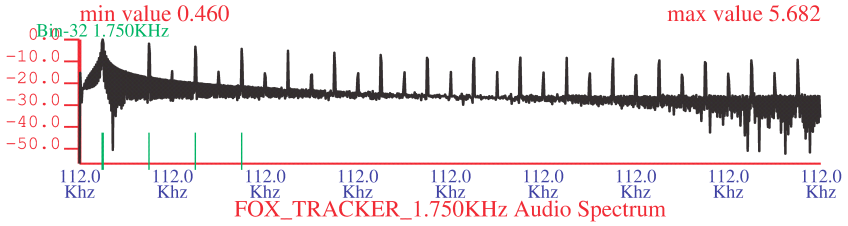


Figure 4.76: Audio Spectrum sampled at 112 KHz

Here are the frequency plots generated from data sets sampled at the indicated frequencies. The signal generator setting of 1.7KHz moves across the plot as the sample rate changes. The antennas switching spikes are fixed as they are synchronous with the sampling clock. In actual use in the field, there will be audio traffic from the fox transmitter that may fall right on top of our antenna switching frequency (as seen in figure 4.76).

We can also look at the Audio Signal spectrum only around the antenna switching frequency:

Missing Graphic

Figure 4.77: Expanded Audio Spectrum

The *background* signal from the fox transmitter may, as previously mentioned, fall right on top of the antenna switching frequency. In this example, the signal generator has noise added to the simulated switching 1KHz *glitch* to 0.9KHz and at 1.1KHz. We can, potentially, pick out a false indication of the strength of the direction information. Here is where shifting the antenna switching frequency can be of great help.

This assumes, of course, that spectral analysis can be performed fast enough that you can't easily mislead the system by swinging the antenna too fast. The STM32 has single precision floating point, as mentiond at the begining of this section . The device may be clocked at up to 160MHz should the need arise. In practice, clock is configured as part of device startup and is not changed during operation.

Waveform Amplitude	**Power Units**
0.100 V	15.6 - 22.1
0.500 V	21.8 - 25.7
1.000 V	23.3 - 27.7
1.500 V	27.1 - 28.6
2.000 V	27.0 - 29.2

Table 4.1: Initial Data Analysis

This preliminary analysis was performed using the NUCLEO-U575ZI board to evaluate the data analysis plan.

The stimulus signal is from a *FY6900 DDS Arbitrary Waveofrm Generator* that has a very rough simulation of the audio signal coming from a receiver attached to to the antenna switch described in section 4.4.6 on page 36.

The signal level is a summation of 5 FFT bins, 2 on either side of the antenna switching frequency. The FFT result value is expressed as log10 of the raw number from the analysis of a 1024 point dataset.

The stimulus signal has simulated positive and negative pulses that represent the antenna switching signal along with some tones near the antenna switching frequency.

Here the 102-73174-99 daughterboard is configured to provide a test signal along with an external signal generator:

(a) Rising Edge Glitch

(b) Falling Edge Glitch

Figure 4.78: Antenna Switching Glitches

The Green trace is the contents of the 64-channel accumulator. The Yellow trace is the antenna switching signal. The Blue traceis the audio signal.

The 102-73174-99 board is configured as shown int figure 4.79. The antenna switching signal is simply fed back into the audio channel through C25.

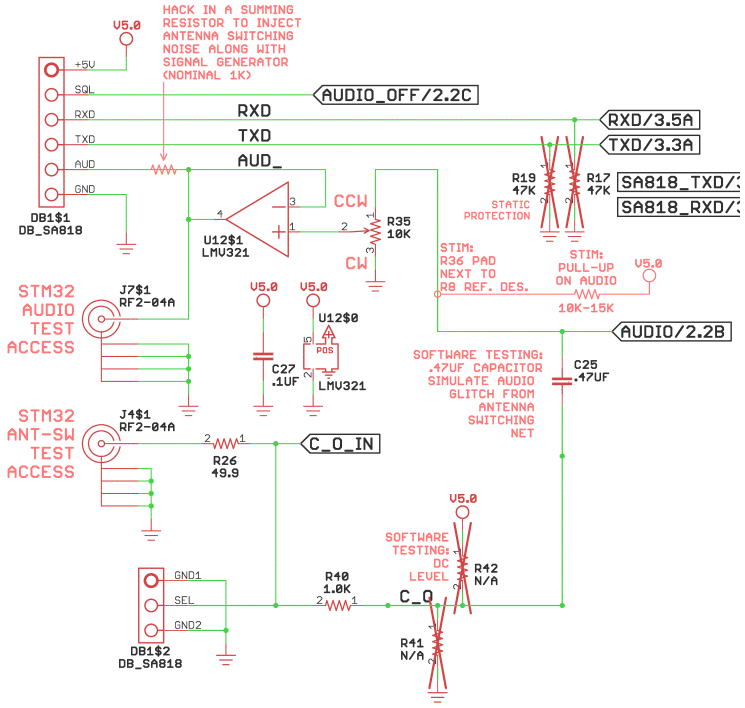


Figure 4.79: Test Simulator

This fragment is taken from the 102-73174-99 schematic showing the addition of C25 to couple the antenna switching signal back into the audio signal. The 102-73174-99 board would be built-up as a test fixture where the SA818 receiver module is not installed. C25 may be selected to control the width of the antenna switching pulse (C25/R35 sets the time constant). Noise may be injected at pin-3 of the SA818 pad array (labeled on the circuit board as "AF").

So, in figure 4.78a, where the antenna switching net rises, we see the rising glitch impressed on to the audio. The audio seen by the STM32 is shown in figure **??**. The analysis being performed is a rough simulation of the synchronous detection of the 102-73174-51 system. Here we are exclusively in the digital domain.

The analysis begins by performing an edge detect on the antenna switching signal. Note that in figure 4.70 theat the rising edge of the antenna select signal appears at the begining of the buffer. Software collects excess samples to allow the edge detect to discard samples at the beginning of the buffer.

Once synchronized, a 64 channel accumulator is used to sum the contents of the audio buffer. Given a 1024 sample buffer, we accumulate 16 samples in each accumulator channel. The glitch stream, being caused by the antenna switching signal, must be synchronous with the 64-channel accumulator. Other noise sources, being uncorrelated, should average out.

We can the go back through the 64-channel accumulator and pick out the high and low channels to determine where the antenna glitch has occurred. The 64-channel accumulator high and low should align with the antenna switching signal, although the rising-edge/accumulator-high falling-edge/accumulator-low may be inverted. This inversion should give us a forward/backward indication of the signal direction.

4.8.3 STM32 Host Control Ports

The STM32 programming interface on these boards is configures to work with the STLINK-V3 programmer. A 14-pin 0.50mm programming connector is present to facilitate fast programming of the STM32U5 device.

The STLINK connector incorporates a serial debugging port that is used duting software development. In the final release product, this port iprovide an alternate control path through which regular commands may be sent.

The software notes from where the command buffer originates and reflects status reports back to that port. This allows a single physical connection to the system to issue comnmands and to load the external memory device.

4.8.4 STM32 Antenna Switching

The antenna switching methodology follows that of the zNEO based prototype, but aims to take advantage of the expanded capability of the STM32.

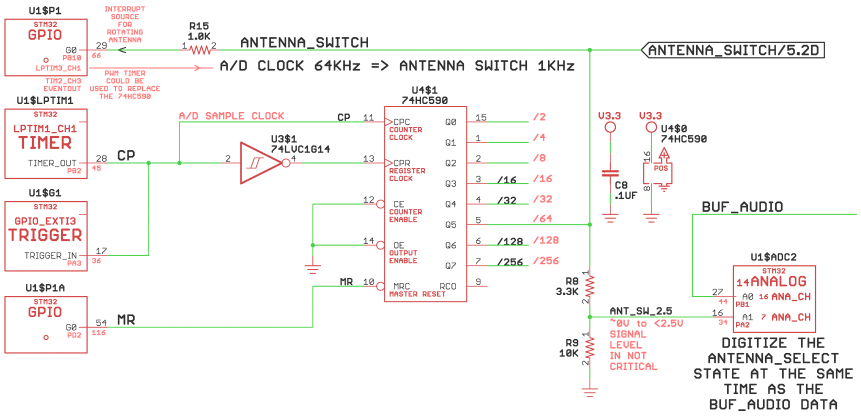


Figure 4.80: STM32 Sampling Logic

U1$LPTIM1 is used to generate the sample clock for the A/D. U4 divided the A/D sample clock by (*nominally*) 64. This gives 64 samples for each antenna transition. Given a dataset size of 1024 samples, the antenna switching energy will fall into bin 8 of the FFT result (out of 512 bins).

We can control which bin the antenna switching energy appears in by increasing the dataset size or by changing the divisor tap on U4. Either way, we keep that antenna switching energy locked to the same FFT bin regardless of the antenna switching frequency we operate at.

By tracking the amplitude of a single FFT bin, we can perform the synchronous detection job. All while using far fewer parts on the Fox Tracker board.

Also note that the schematic indicates that not only the analog audio data (**BUF_AUDIO**) is to be digitized, but also the **ANTENNA_SWITCH** digital control signal out of U4. If we analyze the **ANTENNA_SWITCH** signal, a square wave, we will see the fundamental frequency in the same FFT bin as the audio data we are interested in. **BUF_AUDIO** waveform.

We can also look at the phase data for the *fundamental* bin from both datasets. I think, at this point, that the phase relationship is fixed, as this difference depends only on how lonh the radio takes to process the received FM and convert it into an audio signal (consider how long a SDR radio takes to demodulate the incoming signal).

4.8.4.1 74HC590 Behavior

The 74HC590 provides a convenient binary divider that is used to extract the antenna switching net from the audio sample clock.

The 74HC590 has a reset pin that resets the internal counter, but it does not affect the output register until the **register clock**, pin 13, is clocked (ie. a rising edge).

In active operation, the **counter clock** and the **register clock** are fed the true and inverted sample clock from the STM32. The counter advance one count on the rising edge of the sample clock and the output register clocks on the falling edge of the sample clock.

In order to

4.8.5 STM32 Compass Display

The same group of nine LEDs form a meter or compass display. These LEDs are located in the same position as the zNEO implementation to keep the same enclosure.

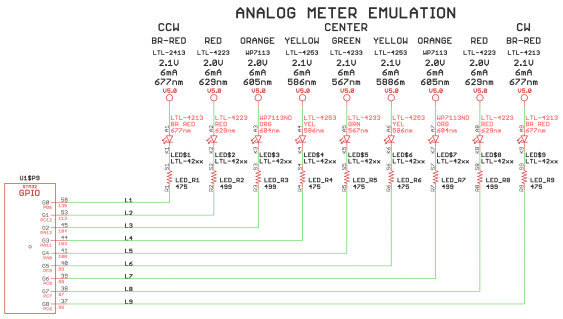


Figure 4.81: STM32 Display Logic

The display driver is straight-forward, although the GPIO bits are scattered across several GPIO ports.

4.8.6 Antenna Control Interface

The antennas control interface is similar. The fast switching comes from a 75HC gate and the 3-of-8 select from GPIO pins.

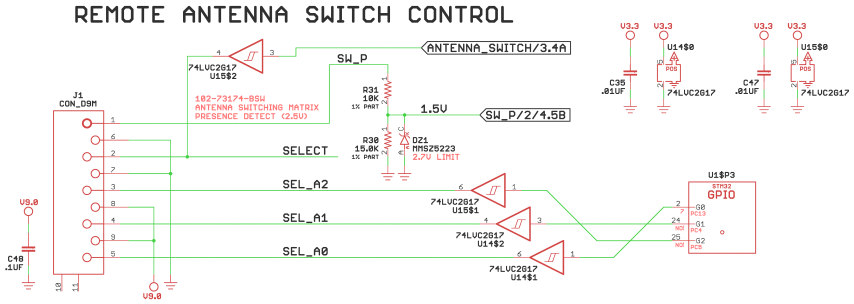


Figure 4.82: Antenna Control Logic

To drive the antenna select, we simply buffer three select lines from the STM32. The **SELECT** net is driven by a logic gate.

4.8.7 Receiver Audio Interface

The receiver audio circuits are similar but improved somewhat.

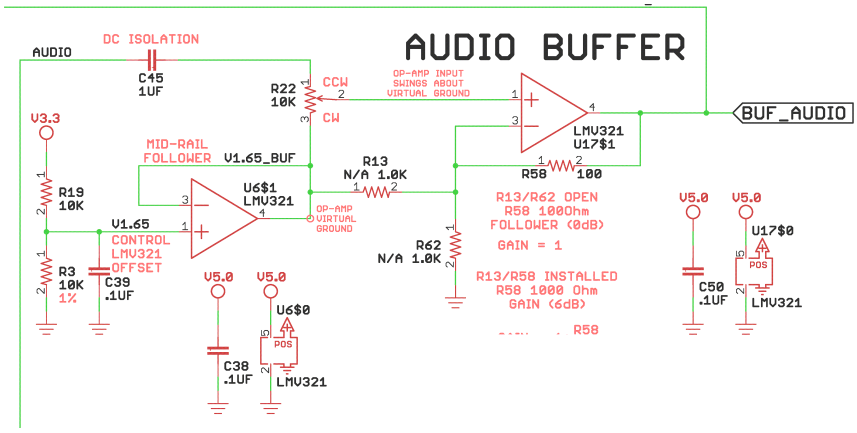


Figure 4.83: Receiver Audio Buffering

Here the receiver audio comes in on the left through C45 into R22. C45 provides DC isolation and R22 is basic level control.

TRhe STM32 ADC uses an internal 2V reference, so we need to center the audio signal around the half-way point (1V). U6 is a follower that buffers the R19/R3 resistor divider. The output of U6 shouldbe close to 1V.

U17 is also configured as a unity gain follower. This follower simply reduces the impedance of the audio signal that is presented to the ADEC in the STM32.

The bottom end of R22 is held at 1V (coming from U6) to set the mid-point voltage level provided to the ADC. R13 may be installed to double the gain through U17.

4.8.8 Daughtercard Interface

The daughtercard interface occurs through two vertical connectors.

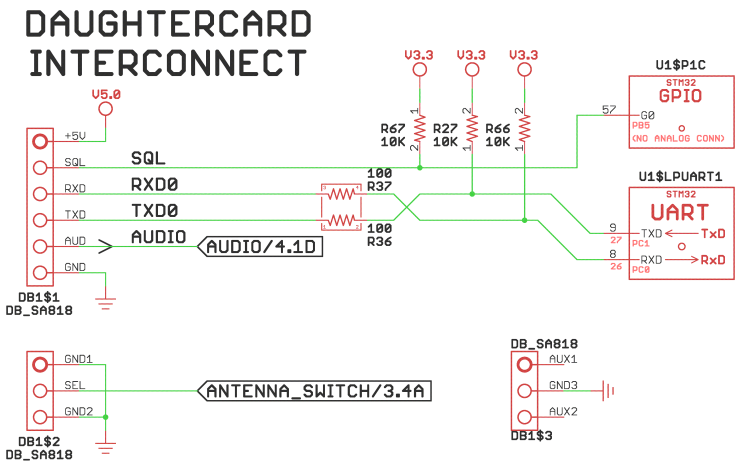


Figure 4.84: Daughtercard Control Logic

The signal interface to the daughtercrd is quite simple.

4.8.8.0.1 SQL from SA818 on daughtercard

The carrier detect signal from the SA818 tranceiver. This line is HI when detecting carrier energy.

4.8.8.0.2 RXD0 from SA818 on daughtercard

The serial data (i.e. status) from the SA818.

This datapath uses the same port to control the SA818 as does the Fox Transmitter to allow use of the same serial handler.

4.8.8.0.3 TXD0 to SA818 on daughtercard

The serial data (i.e. commands) to the SA818.

The datapath matches Fox Transmitter here too.

4.8.8.0.4 AUDIO from SA818 on daughtercard

The audio data from the SA818.

4.8.8.0.5 CARRY_OUT to antenna switch on daughtercard

Control/timing for the antenna switch on the daughtercard.

4.8.8.0.6 Power and Ground for the SA818 on the daughtercard

Note that we call out the SA818 from Nice-RF (and not the DRA818 form Dorji).

The datasheet indicates that the SA818 module will run on 5V.

4.8.9 STM32 Host Configuration Interface

The STM32 has similar UART capability to the zNEO.

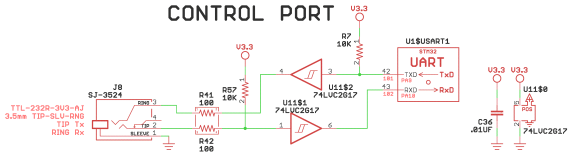


Figure 4.85: Host Configuration Port

The STM32U5 also supports programming through this port using a simple serial cable. The **BOOT0**(JP2) jumper is installed to place the STM32 into a bootstrapping mode.

4.8.10 STM32 JTAG/Flash Programming Interface

Device programming nominally requires something like a *STLINK-V3* from ST Micro. The *STLINK-V3* adds an additional serial channel that is routed to on of the UARTS on the STM32.

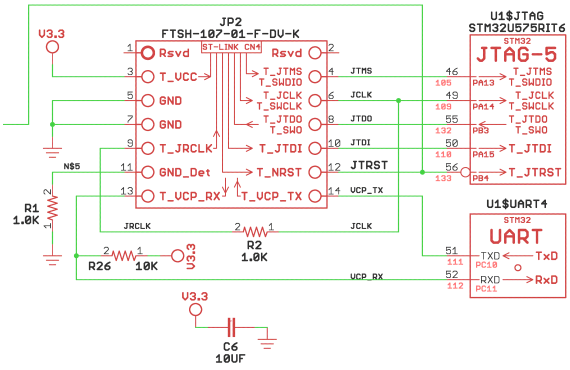


Figure 4.86: Host Configuration Port

The **JP2** connector is the **STLINK** programming connector. Mechanically and electrically compatible with the small 14-pin programming interface on the **STLINK-V3** programmer.

Although this design is pin limited, the full JTAG pinout is provisioned along with a dedicated serial debugging port. The **STLINK-V3** provides a simple UART port that is routed through the programming connector.

4.8.11 STM32 Power-on Reset

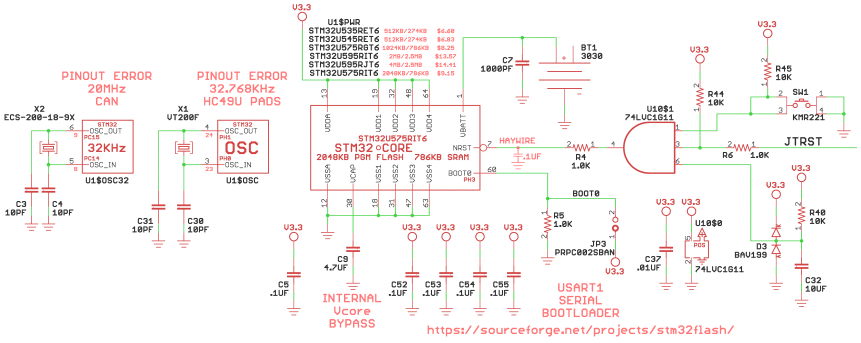


Figure 4.87: STM32 Power and Reset

The POR circuit is recycled from earlier zNEO designs.

Reset signals from a POR (Power On Reset) circuit, a simple button, and the **JTRST** pin on the program/debug connection and sent to the STM32 **NRST** pin. The circuit board is provisioned with parts to isolate the **JTRST** pin if this becomes necessary. Note that the STM32 device should be able to be reset using a JTAG command, thereby making a dedicated reset path unnecessary.

4.8.12 STM32 Pin Allocation

ICARC FOX TDOA RECEIVER PORT ASSIGNMENTS
73174-40

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT A	JTAG TDI 50	JTAG CLK 49	JTAG TMS 46	METER L3 45	METER L4 44	USART1 RXD 43	USART1 TXD 42	METER L5 41	SPI1 MOSI 23	SPI1 MISO 22	SPI1 SCLK 21	DAC OUT 20	GPIO EXTI3 17	ANALOG AUDIO ANT 16	ANALOG SW_P 15	ANALOG SMON 14
PORT B	ENCODER B 36	ENCODER A 35	ENCODER D 34	ENCODER SPARE 33	X	ANT_SW IN 29	DAC QUIET 62	RSVD I2C1 SCL 61	RSVD I2C1 SDA 59	METER L1 58	SQL 57	JTAG TRST 56	JTAG TDO 55	LPTIM OUT 28	ANALOG AUDIO 27	SPI1 SS* 26
PORT C	OSC32 OUT 4	OSC32 IN 3	ASEL A0 2	METER L2 53	UART4 RXD 52	UART4 TXD 51	METER L6 40	METER L7 39	METER L8 38	METER L9 37	ASEL A2 25	ASEL A1 24	ANALOG BMON 11	ANALOG IMON 10	LPUART TXD 9	LPUART RXD 8
PORT D	X	X	X	X	X	X	X	X	X	X	X	X	X	MR OUT 54	X	X
PORT H	X	X	X	X	X	X	X	X	X	X	X	X	BOOT0 60	X	OSC OUT 6	OSC IN 5

Figure 4.88: STM32 Pin Allocation

Pin assignments for the 64LQFP package.

We are left with 3 unused pins on this particular package.

The STM32 has a very comprehensive set of peripheral blocks. UARTS are plentiful, so no port sharing is needed with this device.

Most functions have only one or two pins that can be assigned, so there is a bit of juggling that goes on to route the needed peripherals to package pins.

4.9 STM32U5 Signal Analysis

Source File: FOX_zNEO_Theory_Operation_9.tex

A discussion of the expanded use of CPU cycles to manage the direction finding operations.

First, consider the greatly expanded SRAM footprint of the STM32U575 device. Contained in this device, is a rather large footprint of SRAM, 768KB. This proviudes ample room for collecting data from the outside world and storig it in multiple data formats for later processing.

The A/D hardware itself provides a 14 bit multi-channel convertor. The STM32U5 further adds a general purpose DMA controller to manage data flow from the A/D directly to memory. Sampling rates are controlled by a general purpose timer. A sophisticated interrupt control system is used to manage data flow (i.e. detect when the last sample has been collected).

4.9.1 Sample Clock

The sample clock is generated by LPTIM1 in the STM32U5. This timer is configured as a 16 bit upcounter. The clock for the timer comes from the system clock (60MHz or 80MHz) through a prescaler. The softweare takes a target audio (or antenna switching) frequency and loads the LPTIM1 register to produce the target clock.

The resulting sample clock is routed to the A/D as well as to an external pin. The external pin drives an external counter (see section 4.8.4 on page 57) that, in turn, drive the antenna switching.

The sampling system is internally synchronous. In other words, the audio sampling frequency is locked at 64 time the antenna switching frequency. We are free to choose an audio sampling frequency that avoids environmental noise that is seen by the radio reciver.

4.9.2 DMA Controller

The STM32U5 provides a multi-channel DMA controller that we take advantage of to move data from the A/D subsystem to memory. The DMA device is programmed with a destination address for the incoming data and a byte count in order to effect the transfer.

The DMA device also manages data format translations between the input device and memory. For this application, where memory is freely available, we save each A/D sample in a 32 bit word. The lower 14 bits of each memory word holds the sampled data value and the upper 18 bits are cleared to zero. We are, then, left with integer data that may be treated as unsigned or unsigned data.

The transfer count, loaded by the operating software, counts down with each memory write. When the transfer count reaches zero, the DMA controller sends a completion signal to the system interrupt controller. The processor is free to poll the DMA controller, looking for a transfer complete status, or enable the DMA interrupt to trigger processing when the transfer is complete.

4.9.3 A/D Controller

The A/D device is a multi-channel convertor with far more capability thea is used in this application. Here, we configure the A/D controller to convert two channels (looking for U1$ADC2 in the schematic fragment in section 4.8.4 on page 57).

The two channels are sampled in sequence (the A/D subsystem has a single convertor). The two samples are triggered by the clock signal from LPTIM1. Each clock pulse from LPTIM1 triggers a pair of conversions.

4.9.4 Sampling

We can alter the sampling frequency for each dataset that is collected. This allows for decorrelating non-antenna switching noise .

4.9.5 Direction Ambiguity Removal

When the simple antenna switching is used with an FM receiver, the demoulator in the reciever moves the antenna switching discontinuity into the outgoing audio.

In general, the *glitch* in the audio stream will contain information about which antenna element is closer to the transmitter. Because we digitize the audio, we can look back into the audio and attempt to correlate the audio *glitch* with the antenna switch setting.

Consider, for a first order attempt, that we simply average the audio signal over the dataset we just sampled. We use the antenna select channel to split the audio into two subsets, one subset when the antenna switch is LO and a second subset when the antenna switch is HI.

The antenna switch serves as a synchronous selector to seperate the positive *glitch* and the negative *glitch* into seperate bins.

4.9.6 102-73174-51 emulation

This scheme may be elaborated upon to somewhat emulate the operation of the 102-73174-51 system. Looking at the schematic fragment in section 4.2.3 on page 16, we can narrow the sampling window to try to catch the glitch directly.

Although this may be effective, timing is very receiver dependant.

4.9.7 A Spectral Analysis Approach

Our primary approach here is to transform the data from the time domain to the frequency domain. A simple FFT analysis, performed on the audio time series, will give an idea of the glitch amplitude. Again, we are synchronous, so given a fixed FFT dataset size, we know which frequency bin to query for *glitch* amplitued.

We do need to take into account that the glitch will show up in the bin associated with the antenna switching frequency, it will also spread across the spectrum as the glitch will not be a sine wave.

4.9.8 Ignoring Background noise

Given the flexibility provided by the STM32U5, we can choose to operate the direction finder at any useful sampling frequency. There is a lower limit below which the audio conditioning circuits in the receiver will affect and possibly interfere with our sampling system.

Likewise, the receiver used in the 2M band is narrow, suitable only for human speech.

The detection system may sweep between these limits. The data collection and analysis currently takes less than 100mS. At this rate we can expect to sample from the reciver at a rate of at least 10 time per socond. With optimizations, that can be increased to 20 to 25 time per second (the STM32U5 is not running anywhere near its peak clock speed).

4.10 STM32U5 System Operation

Source File: FOX_zNEO_Theory_Operation_10.tex

Here we will dig into the operation of the STM32 based tracker.

4.10.1 STM32 Antenna Switching Control

The antenna switching methodology:

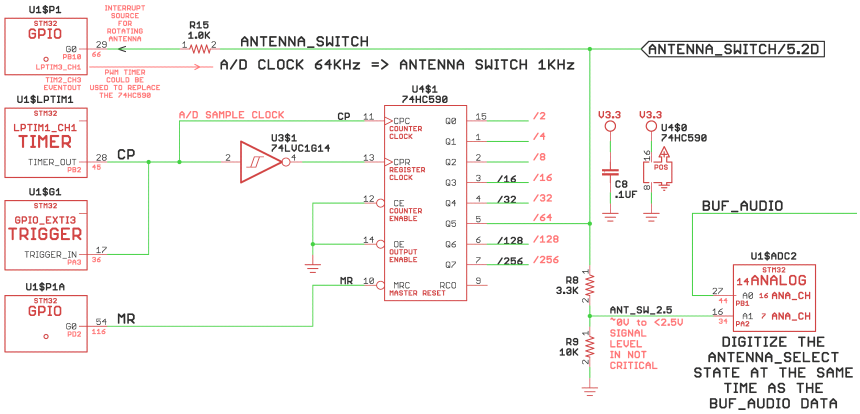


Figure 4.89: STM32 Antenna Switching Control

The schematic fragment from section 4.8.4 is reproduced here.

U1$LPTIM1 is the external presentation of the *master antenna switching clock*. This *master antenna switching clock* is labeled **CP** in the schematic and netlist. The clock is supplied in both true and inverted form to the **74HC590** clock inputs. The **74HC590** clocks the counter on one edge of the **CP** clock and registers the counter on the opposite edge of **CP**.

The **CP** is externally wrapped back into the STM32U5 through the U1$G1 bit. This allows the STM32U5 to use the **CP** clock as an interrupt source or as a trigger for the A/D subsystem. In practice, the **LPTIM1** is delivered to the external pin as well as directly to the A/D sybsystem.

The U1$P1A output pin is used to hold the **74HC590** in reset or allow it to run. The **CP** net runs between the STM32U5 and the **74HC590**. In this design, the **CP** is not used elsewhere on the board so the **MR** net (coming from the U1$P1A output pin) will suppress activity on the **ANTENNA_SWITCH** net.

The **ANTENNA_SWITCH** net coming out of the **74HC590** is wired, in the artwork, to the divide-by-64 output of the **74HC590**. Looking at the traces around U4 (the **74HC590**) you will note that the divide-by-256 and the divide-by-16 outputs from the **74HC590** are labeled in the silkscreen. There are pads/vias on pins 4 and 6 for divide-by-32 and divide-by-128. If there arises a need/desire to change the sampling ratio the trace from the divide-by-64 to the vis leading to R15 can be cut and a haywire installed to connect to an alternate divisor.

Also, take note of the feedback path from the output of the **74HC590** back to U1$P1. This provides visibility into the operation of the **74HC590** for the STM32U5. The STM32U5 can inspect this bit to verify that the LPTIM1 timer is configured and running. It should also be possible to configure the pin to generate an external interrupt to assist in antennas switching when operating the electronic rotator.

The two high speed channels also show up on this schematic fragment. The second channel, from the **74HC590**, simply reports the state of the antennas switching control signal (the **ANTENNA_SWITCH** net). The first channel is audio from the receiver, wherever is happens to be. We will look at audio conditioning in the next subsection.

4.10.2 STM32 Audio Conditioning

The audio conditioning circuits:

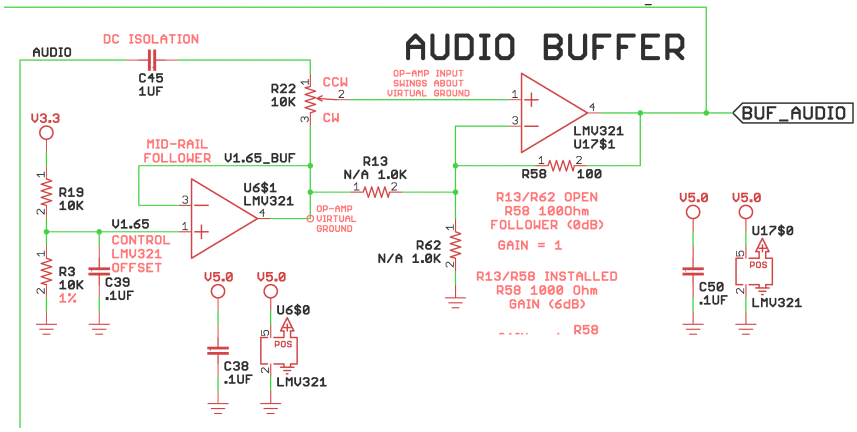


Figure 4.90: STM32

The schematic fragment from section 4.8.7 is reproduced here.

The **BUF_AUDIO** net, the output of the audio conditioning circuit, runs at the top of the schematic fragment over to the A/D input from the previous section. Raw **AUDIO** from the receiver arrives on the left side of this schematic fragment. This raw **AUDIO** is at the DC level presented by the receiver.

We need the audio signal to be centered around a level in the middle of what the STM32U5 A/D hardware expects. C45 serves to provide DC isolation from the receiver and R3/R19/U6 provides the mid-scale bias. R3/R19 generates a voltage midway between the 3.3V rail and ground (the STM32U5 uses the 3.3V rail for its reference). U6, configured as a simple voltage follower, provides a low impedance mid-scale voltage.

By placing the gain adjust pot (R22) between isolated source and the U6 follower, the audio signal is shifted to this mid-scale point and sent to the gain stage.

U17 provides unity gain as shown on the schematic. As shown, U17 is also a simple voltage follower.

If additional gain is needed, we can change R58 to a higher value part and install R13 to provide gain.

If less gain is called for, simply adjust R22.

Both U6 and U17 are powred from the 5V rail. This provides headroom on the high side, but the negative side of everything is ground referenced. Although the LMV321 is advertised as rail-to-rail, we need not increase the gain to the point that the audio signal is *banging* into the limits of the STM32U5 A/D subystem.

4.10.3 STM32 Internal Receiver Interface

The daughtercard interface circuits:

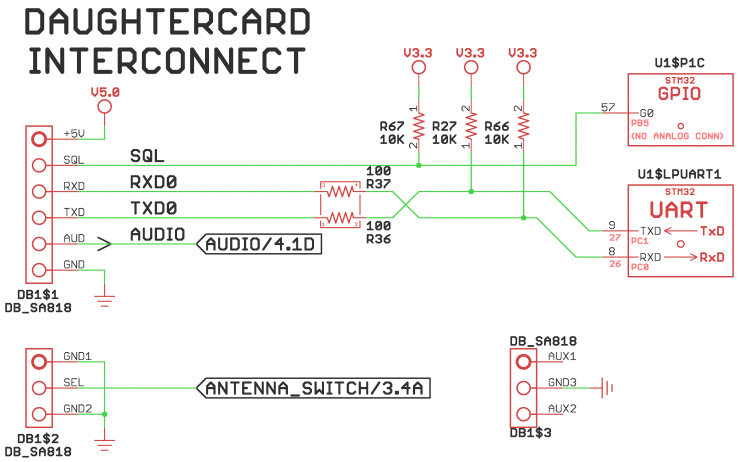


Figure 4.91: Daughtercard Interface Logic

The signal interface to the daughtercrd is simple, just enough to control the SA818 module..

The physical connector is shown as **DB1** on the schematic. The schematic breaks this into three sub-parts, which all appear on this schemnatic fragment. On the *EAGLE* board, it is a seperate part that fixes the connector and mounting holes in place. Each of the various daughterboards use this same connector part to force the connector and mounting holes into correct alignment.

Since we are operating the SA818 as a receiver, there are no controls associated with transmitting. On the SA818 duaghterboard the transmit pin is hardwiared into the receive polarity.

All we are left with are 3 groups of signals to deal with that touch the SA818 and one signal that controls the antenna switch..

Frequency selection on the SA818 is controlled using a simple serial port. One of the manu UART ports on the STM32U5 is used to control the SA818. In our schenatic fragment, R36/R37 provide current limits should we end up with the serial nets miswired. These two resistors are placed on the board to allow switched Rx/Tx nets to be corrected (rotate the resistor pair to swap Tx/Rx).

The audio net from the SA818 runs from the audio pin on the3 SA818, through DB1$1-AUD to the audio buffer. The 3.5mm jack and the SA818 audio nets are not isolated. Only one receiver may be attached at any time.

We also have a squelch control from the SA818. This net indicates when the SA818 detects a carrier. The logic sense is positive, i.e. a '1' indicates that carrier is present.

The pins on DB1$2 provide additional ground and the antenna switching signal. The **ANTENNA_SWITCH** net comes from the **74HC590**.

4.10.4 STM32 Antenna Rotator Interface

The electrically rotated antenna control circuits:

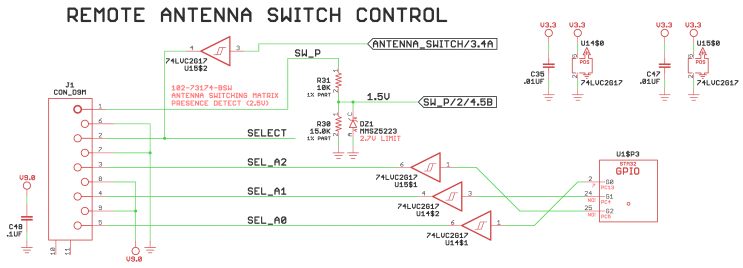


Figure 4.92: Antenna Rotator Interface

This group of logic controls the external antenna rotator

We can, with a little effort, electrically rotate and array of antennas to automate the direction sweep. It is tought that we should also be able to resolve the 180°direction ambiguity by using the antenna switching signal to search the incoming audio for the antenna switching glitch.

We simply generate a 3 bit antenna select to pick one of eight antenna elements at the remote antenna switch. The STM32U5 updates the bit pattern at U1$P3 when the **ANTENNA_SWITCH** net connects the center antnna element to the receiver.

The **ANTENNA_SWITCH** net operates the same for a two-antenna setup and an arrayed-antenna setup. With the arrayed-antenna setup, we switch between a center element and one of eight radial elements.

The **SW_P** net coming back from the antenna rotator board is somewhat undefined currently. The **SW_P** net is routed to the housekeeping A/D channels to allow it to be treated as either an analog channel of a digital channel

The STM32U5 is rather flexible in the functional assignment of the **SW_P** net. We can, if it proves useful, sample this net along with the **AUDIO** and **ANTENNA_SWITCH**. Do keep in mind that there is very little signalk conditioning provided on the **SW_P** net. All that is provided is a zener diode to protect the STM32U5 input.

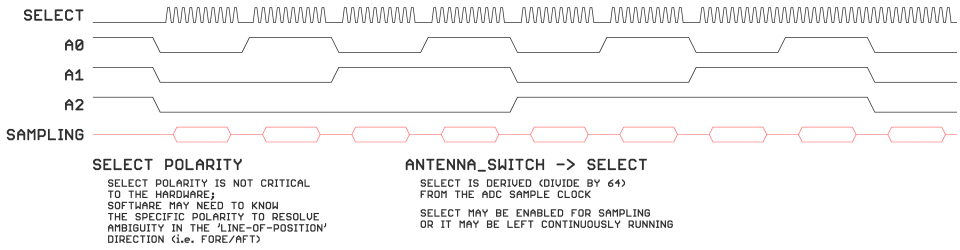


Figure 4.93: Select Timing

4.10.5 STM32 Encoder Interface

The user interface:

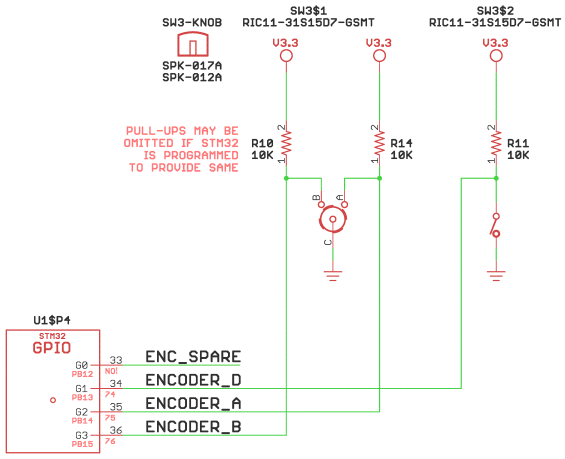


Figure 4.94: Encoder Interface

A simple rotary encoder to operate the tracker.

Most of the operating configuration is held in a small FRAM. The FRAM has a series of configuration commands that are run when the unit is powered on.

We can also use the rotary encoder to select other operating modes. Rotating the encoder selects one of 9 command files that are run when the encoder button is pressed.

The file system in the FRAM comes directly from the earlier *FOX TRANSMITTER*. The **INI=** runs when the unit is powered. Files **S0=** through **S8=** may be selected with the rotary encoder. Pressing the encoder runs the selected file.

The setup file selection is reported on the LEDs. Moving the encoder take the *FOX TRACKER* out of tracking mode and the selected setup file determines what happens next.

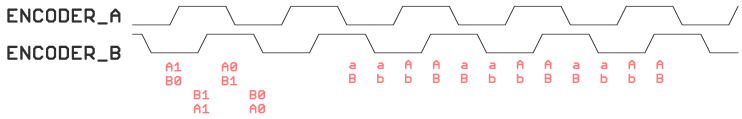


Figure 4.95: Encoder Timing CW

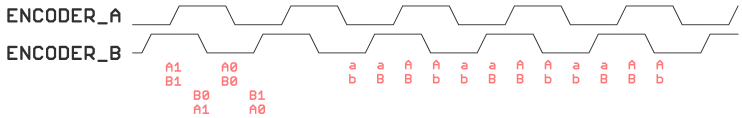


Figure 4.96: Encoder Timing CCW

4.10.6 STM32 5

4.11 DRA818 Antenna Switch

Source File: FOX_zNEO_Theory_Operation_11.tex

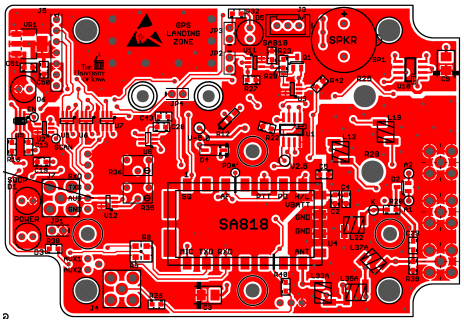


Figure 4.97: SA818 Antenna Switch (102-73174-99)

This board provides a basic two antenna switch along with the SA818 receiver.

There are three antenna connectors to allow the board to function with an external receiver as well as with the internal SA818 receiver.

If the 102-73174-41 board is built without J1 populated, you should be able to mount the SMA connector on the bottom side of the daughtercard to make the cutout in the encolsure a bit more compact.

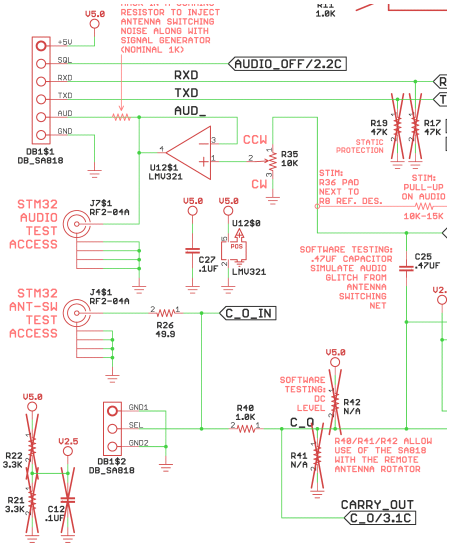


Figure 4.98: SA818 Antenna Switch STIM test

Here we loop the antenna switching signal back into the audio channel to *somewhat* emulate the signature expected from the RF receiver.

Only a few parts are required for the test fixture. We route the antenna switch signal back into the audio net.

The two SMA connectors provide a 'scope connection.

Surface mount resistors and capacitors share common pad dimensions, so we can change R40 into a cap to provide DC isolation. Populate R42 to get a useable DC offset into the op-amp so that it operates between the supply rails.

And finally, C25 swaps for a 10 ohm resistor so that R35 and R42 form a voltage divider to supply a signal to U12 that won't clip.

Chapter 5

Commanding

Source File: FOX_zNEO_Commanding.tex

Control and Configuratiopn commands.

These are take from the Fox Transmitter software.

5.1 Commands

Blah Blah Blah

5.1.1 FREQ

Source File: FOX_zNEO_Command_FREQ.tex

Table 5.1: Receiver frequency control

Command sample	Description
FREQ *freq*	Operating Frequency

Action:

Select the Receiver Operating Frequency.

This new frequency selection will be loaded into the RF subsystem.

Arguments:

Frequency, in MHz.

The action performed depends on the transmit element selected in the **CONF** command.

5.1.1.1 DRA818/SA818

Transceiver Module

The frequency string must be...

5.1.1.2 EXTERN

External Handie Talkie

The frequency string will need to be massaged into the appropriate setup commands for the handie-talkie in use. The V3.70 software does not handle any handie-talkie.

5.1.1.3 Frequency Limiting

The **FREQ** handler does very minimal limits checking to try to keep frequency selection sane. All operating frequencies have been migrated to external tables, so effect any required limits by loading tables with valid entries.

5.1.2 TONE

Source File: FOX_zNEO_Command_TONE.tex

Table 5.2: Antenna Switching frequency control

Command sample	**Description**
TONE *freq*	Switching Frequency

Action:

Select the Antenna Switching Frequency.

The antenna switching subsystemm is set to toggle between adjacent antenna elements at the selected rate.

Arguments:

Frequency, in KHz.

The valid selection range is from 0.1KHz to 2.5KHz. You may also specify a tone frequency of 0KHz to disable the clock to the 74HC4017.

This selection does **not** control the rotation speed of a rotating antenna system. In a dual antenna system this is directly the switching frequency. In a rotating system, the rotation frequency is set by the **????** command.

5.1.3 VOLU

Source File: FOX_zNEO_Command_VOLU.tex

Table 5.3: Antenna Switching frequency

Command sample	**Description**
VOLU *freq*	Volume and Squelch

Action:

Select Volume and Squelch levels.

Set volume level between 1 and 8.
Squelch level are the same and are not required. No change appied to the squelce level if it is not specified.

Arguments:

Volume 1..8

Squelch 1..8

Settings deliverd to the SA818 module.

5.1.4 SCAN

Source File: FOX_zNEO_Command_SCAN.tex

Two antenna TDOA scan. This measures the audio glitch seen during the switch between two antennas. This glitch amplitude and direction (right or left) is displayed on the LED display.

Table 5.4: Antenna Switching scan control

Command sample	**Description**
SCAN *per*	scan period (RTIs)

Action:

Select the Antenna Scanning period and start data collection and analysis.

The antenna system is scanned at the selected rate. The basic 3 antenna scan samples the audio once without switching the antenna and then 3 time while switching.

Arguments:

The scanning period, expressed in RTI units. The RTI is $^1/_{100}$ of a second.

The valid selection range is from 1 to 100 RTI periods. This give the slowest scanning rate of one second. A scanning period in excess of 100 RTIs changed to one second (i.e. 100 RTIs).

5.1.5 ROTA

Source File: FOX_zNEO_Command_ROTA.tex

Nine antenna TDOA rotating scan. This measures the audio glitch seen during the switch between antenna pairs on a nine element array. This glitch amplitude and direction (azimuth relative to the antenna array) is displayed on the LED display.

Table 5.5: Antenna Rotator control

Command sample	Description
ROTA *per*	rotation period ($^1/_{100}$ second RTI periods)

Action:

Select the Antenna Rotation period and start data collection and analysis.

The antenna system is electrically rotated with the selected period. The antenna array is sampled in pairs with the center element being one side of the pair.

Arguments:

The scanning period, expressed in RTI units. The RTI is $^1/_{100}$ of a second.

The valid selection range is from one RTI to 100 RTIs. A scanning period in excess of one second (100 RTIs) is changed to one second (100 RTIs).

5.1.6 TRIP

Source File: FOX_zNEO_Command_TRIP.tex

Table 5.6: Antenna Switching Trip Point Setting

Command sample	**Description**
TRIP *freq*	Txxx

Action:

Set the Antenna Scanning trip points.

The trip points set the levels used to drive the LED display.

Arguments:

Trip points in counts

Field Specifier	**Field Description**
T0=*0*	Select the center LED trip point
T+1=*3*	Select the LED next to center trip point
T+2=*9*	Select the next LED out
T+3=*17*	Select the LED after that
T-1=*4*	Select the LED other side of center
T-2=*8*	Select the next LED out on the other side
T-3=*20*	Select the LED after that

The useful selection range is a 10 bit number. In practice, the range will be small, perhaps less than 100 for the outer ends.

5.1.7 WAIT

Source File: FOX_zNEO_Command_WAIT.tex

Table 5.7: Simple Wait

Command sample	Description
WAIT *n.n*	Wait specified time (in decimal seconds)
WAIT *p/o*	Synchronous Wait (period/offset in seconds)

Action:

Waits for specified time (up to 120 seconds).
Waits for synchronization time (period is 16 bit integer).

Arguments:

Time in seconds (decimal).

Scheduling point in period/offset notation.

Returns:

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

This is a simple delay. Time specified in decimal seconds. Time specification must be between 1/10th. and 60 seconds.

The synchronous wait feature:

This is an immediate synchronization with the schedule specified in the argument. The slash (/) delimiter was just something that is convenient. No whitespace between the period and offset values.

This makes use of the same code fragment developed for the Fox Transmitter. In the Fox Tracker, we do not track time (i.e. we don't have an on-board clock chip to synchronize the system clock), so the synchronous form of the command doesn't provide a means of synchronizing to UT.

Perhaps a GPS module daughtercard???

5.1.8 BATR

Source File: FOX_zNEO_Command_BATR.tex

Table 5.8: Battery Report Text

Command sample	Description
BATR *flag*	Battery Report

Action:

Battery Report (textual).

Arguments:

A flag character **I** for coefficients table dump.

Flag characters of **SCAN** initiate a continuous battery scan.

Returns:

```
sts17,00* Handler_BATR (cmd_battery.c*) T=14.000 V=9.725[031E] I=85.9[00B0] 5=5.032[0204] State-T6
sts17,01* Handler_BATR (cmd_battery.c*) T=15.000 V=9.713[031D] I=89.4[00B7] 5=5.022[0203] State-T6
. . .
STS17,13* Handler_BATR (cmd_battery.c*) T=27.000 V=9.725[031E] I=83.0[00AA] 5=5.022[0203] State-T6 0.01 Sec
RDY00,00* 27.02 (Sp=0xBF98)+1998
```

This command generates a battery state report showing current voltage and current as seen in the Fox Tracker.
This is intended to be used to characterize battery capacity.

You may find it useful to load the current UNIX time into the Fox Tracker prior to a battery performance run. This will result in the current time, rather than a time since reset, to appear in the battery report.

The **SCAN** flag makes the report run continuously until a character is received or the battery runs low. The report is issued every 5 seconds.

You

5.1.9 ESAV

Source File: FOX_zNEO_Command_FRAM.tex

Table 5.9: FRAM control ESAV

Command sample	Description
ESAV	

Action:
Save a command into FRAM-volatile memory.

Arguments:
Command text.

Returns:

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

Overflow:

Writing too many records to the FRAM will not cause overflow. The FRAM handler will not write past the end of the physical FRAM present in the system, rather overflow data is discarded.

The smallest FRAM device the software will deal with is 64Kb which is room for 256 records. This size device is marginally small for managing a large or complicated hunt. Whatever commands overflow the FRAM are, of course, lost.

5.1.10 EDMP text

Table 5.10: FRAM control EDMP

Command sample	Description
EDMP	
EDMP *key*	Entries that have text matching *key*

Action:
Dump FRAM-volatile memory records.

Arguments:
Search Key.

Returns:

sts01,nn* records with active FRAM records.

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

Scanning of the FRAM records stops when an empty record is encountered.

The **EZER** command makes subsequent records *invisible*!

5.1.11 EDID

Table 5.11: FRAM control EDID

Command sample	Description
EDID	

Action:
Dump FRAM/FLASH JEDEC ID.

Arguments:
None.

Returns:
sts01,nn* records for the FRAM device.

sts01,nn* records for the FLASH device.

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

5.1.12 ERAS

Table 5.12: FRAM control ERAS

Command sample	Description
ERAS *n*	remove record (i.e. REM-)
ERAS *start stop*	remove multiple records
ERAS *DEV*	erase device

Action:
Remove the FRAM record by inserting a **MT**** command in place of the existing command.

Remove multiple records by specifying a range of records.

Arguments:
Record Number or keyword

Returns:
sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

5.1.12.1 Numeric Record Number

Specifying the record number (or range of numbers) rewrites the specified record(s) with a **MT**** command.

This leaves the remainder of the FRAM file system visible.

5.1.12.2 DEV Entire Device

Erase the entire FRAM device.

Using *fox_binary* to avoid using the **HERA** command.

There is no need to erase the FRAM when using the **HERA** command as the *fox_binary* loader completely bypasses the (somewhat primitive) file system. The *fox_binary* loader will place a zero record following the last command that is loaded to properly mark the end of active records.

This behavior will be particularly useful if you have audio clips loaded into the FRAM on older boards that do not have a separate FLASH for storing waveform data. The *fox_binary* utility leaves that waveform data undisturbed.

5.1.13 EZER

Table 5.13: FRAM control EZER

Command sample	Description
EZER *n*	erase record (ZERO)
EZER *start stop*	erase multiple records (ZERO)

Action:

Erase a single record from FRAM.

Erase multiple records from FRAM.

Arguments:

Specifying a single record number zeroes the specified record.

Specify a start record and stop record to zero a range of records.

This leaves the remainder of the FRAM file system *invisible*.

Returns:

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

Use the ERAS command to change the record to **REM-**, leaving following records visible.

This command zeros the FRAM record out, making any records that follow *invisible* until an **ESAV** command rewrite the record.

5.1.14 ETAB

Table 5.14: FRAM/FLASH table dump

Command sample	Description
ETAB	FRAM & FLASH device table dump

Action:
Dump FRAM/FLASH JEDEC-ID table.

Arguments:
none.

Returns:

sts01,nn* records with device information.

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

The entire device ID table is dumped.

Each detail line has the following:

Table 5.15: FRAM/FLASH device Table

Column	Contents
Write-Mode	write/erase strategy
JEDEC-ID	device ID bits
Size	device size, in bits
Page	write page size in bytes
Sctr	sector erase size in bytes
Manufact	Device manufacturer
Device	Device part number

5.1.14.1 Column: 2. Write-Mode

A field indicating to the software the write and erase strategy for this device.

The typical FLASH device will be of the **FLASH_PAGE** type, indicating that the device can deal with a multi-byte write.

The typical FRAM device will be of the **FLASH_FRAM** type, indicating the device can deal with any length write and does not require device erase to update records.z

A not-so-useful device, FLASH_AAI, can write 2 bytes at once. Probably too slow to be at all useful for this project.

The FLASH_AAI types will not work with the binary loader!

5.1.14.2 Column: 3. JEDEC-ID

This is the 3-byte sequence used to recognize the device.

Some devices may appear twice, with slightly different patterns to accommodate the chip reading method (1st. byte is 0x7F).

5.1.14.3 Column: 4. Size

This is the device size, in bits.

422 FRAM type devices tend to be more expensive as size increases. We use a reasonable size device, typically 64Kb to 256Kb, to store commands.

Large capacity FLASH type devices tend to be less expensive. These type are used to store waveform data.

5.1.14.4 Column: 5. Page

This is the write page size in bytes.

439 FRAM device will have a page size of 1. This class of device writes a byte-at-a-time at wire speed. This makes for easy erasing of a single command record (32 bytes).

FLASH device need to have a page size of at least 32 bytes. This lower limit is imposed by the expected4d InTel-HEX record size that can be handled by the command parser.

5.1.14.5 Column: 6. Sctr

This is the sector erase size in bytes. A size of zero indicates the device functions and a non-volatile RAM device.

456 Currently, the FLASH device (U12) is managed as a single *lump* of memory. It is erased as a unit.

Sector erase may be fully implements at some point to improve the flexibility of the system, but not now...

5.1.14.6 Column: 7. Manufact

467 Device manufactured by.

This is simply obtained from the device data sheet and entered in the table.

5.1.14.7 Column: 8. Device

478 Device part number. This should match package markings.

This is also obtained from the device data sheet and entered in the table.

5.1.15 HEND

Table 5.16: FLASH control HEND

Command sample	Description
HEND	

Action:
Find start of empty flash device.

Arguments:
None.

Returns:
sts01,nn* response with address of empty space in the flash device.

RDY00,00* response with current stack pointer value and current system time.

The scan occurs with a 4K stride from end of device towards beginning.

This provides a means of finding unused areas of a large flash device.

5.1.16 HERA

Table 5.17: FLASH control HERA

Command sample	Description
HERA BLOCK 0x<*start*>	erase record
HERA ALL	device erase

Action:
Erase region of FLASH device.

Arguments:
BLOCK Erase block: Block Starting Address.

ALL Erase device: No Arguments.

Returns:
sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

This command has **no sanity checking** to prevent an unintended erase operation.

This command is **in**sensitive to the **CONF VOICE** setting (see section **??** on page **??**). You can use the **HERA** command to bulk erase audio data in the FRAM device.

Also keep in mind that chip erase time for many flash devices are quite long, exceeding 100 seconds for some. This command does not wait for the erase operation to finish, it simply returns after sending the **chip erase** or **block erase** command to the device. This makes the flash device look *dead* until the **chip erase** or **block erase** operation has finished. Sending any flash commands will return a **BUSY** message until the device reports it is ready.

For the block erase variant of the command; the address argument is passed, unchanged, along with the block erase command (0xD8) to the FRAM device. The user must consult the datasheet to correctly form the address to erase the desired area of the device (typically 64K)

Also take note that the address argument is in hexadecimal!

5.1.17 HDMP

Table 5.18: FLASH control HDMP

Command sample	**Description**
HDMP *length start*	dump 32 byte records

Action:
Dump an area of the FLASH device.

Arguments:
Length (in 32 byte lines) and start address.

Returns:

:20 .... (InTel HEX dump records)

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

The hex commands (HDMP HERA and :xxxx) are sensitive to the VOICE configuration setting (see the CONF command in section **??** on page **??**). This feature should be unused with the 102-73181 boards with dual serial memory. This feature allows the 102-73161-25 board to store audio clips in the single serial memory device.

5.1.18 H56K/H115

Table 5.19: H115/H56K

Command sample	Description
H56K	Set bit rate to 57,600 b/S
H115	Set bit rate to 115,200 b/S
H56K PROG	Set bit rate to 57,600 b/S, FRAM loader binary mode
H56K WAVE	Set bit rate to 57,600 b/S, FLASH loader binary mode
H115 PROG	Set bit rate to 115,200 b/S FRAM loader binary mode
H115 WAVE	Set bit rate to 115,200 b/S FLASH loader binary mode

Action:

Switch bit rate to 57,600 bits/second (control port defaults to 57,600 b/S).

Switch bit rate to 115,200 bits/second (control port defaults to 57,600 b/S).

Switch bit rate to 57,600 bits/second and operate in binary transfer mode.

Switch bit rate to 115,200 bits/second and operate in binary transfer mode.

Arguments:

None.

Returns:

RDY00,00* response with current stack pointer value and current system time. Operating at the new bit rate.

The clock divisor at this bit rate was off by up to 5%. The data communication between host and target would not have been reliable on versions prior to 3.93 (changed divisor by 1 count).

Older method to speed up the audio loading process (prior to V4).

First, switch the Fox Transmitter over to the higher speed.

V3.68 and prior use H56K to switch to 57,600 b/s.

```
H56K
sLask.jcn
```

V3.69 and later use H11 to switch to 115,200 b/s.

```
H115
sLask.jcn
```

For either case, the STS and RDY response are, of course, garbled (hence the *sLask.jcn* or some other unInTelligible garbage) due to the mis-matched bit rates.

Next switch the monitoring terminal over to match the new bit rate.

```
.../halo_term -b57600 -SFOX2X
.../halo_term -b115200 -SFOX2X
.../halo_term -SFOX115
```

And then we can proceed to download the audio HEX file at the new, higher, rate. We trim the delay between each line sent to the target to around 50 milliseconds (i.e. the **-c50** on the call line).

```
.../fox_simple -b57600 -SFOX2X -c50 -t10 -ffox_73181_rxxa.hex
.../fox_simple -b115200 -SFOX2X -c50 -t10 -ffox_73181_rxxa.hex
.../fox_simple -SFOX115 -c50 -t10 -ffox_73181_rxxa.hex
```

The higher bit rate reduces the transmission time, of course, and reduces the time the target spends sending the shortened RDY message.

Do keep in mind that the target serial channel is buffered on the input side (the entire input line is buffered by the ISR) This may allow some overlap to occur, although the time required to program a 32 byte line in the flash may ultimately limit speed.

H115 WAVE and H56K PROG

The last forms of the command, with either the **WAVE** or the **PROG** modifiers, switches over to the binary loader. The **WAVE** modifier is used to program FLASH memory. The **PROG** modifier is used to program FRAM memory.

When loading FLASH using the binary protocol the target area of the FLASH must be erased before using the **WAVE** modifier.

When loading FRAM using the binary protocol it is not necessary to perform an erase operation (**HERA dev**) to clear FRAM.

The binary loader bypasses the file system, directly writing records to FRAM. One record following the last record sent is cleared to ZERO. A dump of FRAM will show only the content that was just loaded.

The binary protocol is identical for loading either memory device.

The behavior of this form of the command is to switch bit rates to the specified rate and then switch the serial port *Interrupt Service Routine* over to a binary mode in order to perform a fast download to FLASH or FRAM memory.

Currently, downloading InTel HEX files is rather slow, taking 30 to 60 minutes to load the audio file system with waveform data. This hook switches to a binary type of processing where the *ISR* expects fixed length binary packet data to be sent by the host.

Loading time with the binary handler is reduced to a few minutes for a 500KB FLASH load and a few seconds for a 300+ command FRAM load.

You must keep the download within the address limits of the target device. Neither the *fox_binary* utility nor the software in the fox transmitter do any bounds checking when loading using the binary protocol.

Software versions later than **Version 4** are required to deal with this feature.

5.1.19 :hex

Table 5.20: InTel HEX Record Load

Command sample	Description
:hex *record*	InTel HEX record up to 32 bytes long

Action:

Load FLASH device.

Arguments:

InTel HEX record (up to 32 bytes of data)

Data **must** be naturally aligned (FLASH device restriction).

> The flash device data alignment requirements are not enforced by the flash loader, data records **must** be generated that do not violate alignment requirements.

Returns:

sts01,nn* response with status and command execution time.

RDY00,00* response with current stack pointer value and current system time.

Valid Record Types:

TYPE 0: INTEL_RECORD_TYPE_DATA

Memory image data.

TYPE 1: INTEL_RECORD_TYPE_EOF

Indicates this is the last record in a group.

TYPE 3: INTEL_RECORD_TYPE_EXTENDED_SEGMENT

Address bits 19..4

This address data is added to the address in the data record.

TYPE 4: INTEL_RECORD_TYPE_EXTENDED_LINEAR

Address bits 31..16

This address data provides the upper 16 bits of the data record address.

Checksum:

The *InTel HEX* loader validates the checksum at the end of the hex record before it is loaded into FLASH. An invalid checksum will cause the record to be rejected.

Overflow:

Writing too many records to the FLASH **will** cause overflow. The FLASH handler does not check for addresses that extend past the end of the memory device.

Expect problems to occur if the *InTel HEX* Load is too large for the device. At a minimum, expect the first audio file to be corrupted.

The FLASH device requires a separate erase operation to restore the memory array to all **1**s. All we can write to the memory array is **0**s. This behavior indicates the an overflow write will be catastrophic and require a device erase to recover.

The input buffer is examined for the leading colon character before normal command processing. An InTel HEX Record is diverted and processed as FLASH data.

Sample audio hex record:

```
:02 0000 04 0000 FA
:20 0000 00 52494646501000005741564566D74201000000001000100A00F0000A00F0000 4F
:20 0020 00 01000800646174612B10000080808180807F8080808080808080808080808080 E2
```

This example text comes from the audio utility. The embedded spaces are added by the audio utility to make looking at the record a bit easier on the eyes. These *whitespace* characters are ignored, they are **not** required.

This command is sensitive to the **CONF VOICE** setting (see section **??** on page **??**).. See comments in section 5.1.17 on page 92.

When configured to use the FRAM for voice storage, this command will happily write to the command area (i.e. the bottom of FRAM where commands/sequences are stored). Memory allocation is strictly manual.

File load accommodations:

The status report generated when encountering an *InTel HEX Record* is abbreviated in an attempt to minimize traffic on the command channel. All that is
1057 reported is a **RDY00,00*** to indicate we are ready for the next record.

All **sts00,00*** and **STS00,00*** reports are suppressed.

Chapter 6

258

Operating the FOX Trackers

Source File: FOX_zNEO_Operating.tex

6.1 Operating the zNEO TRACKER

Source File: FOX_zNEO_Operating.tex

Not much to elaborae on here as we didn't really get that far along before is became obvious that there is a better approach.

6.2 Operating the STM32 TRACKER

Source File: FOX_STM32_Operating.tex

The STM32U5 based system adds a simple control in the form of a rotary encoder that includes a button feature.

Rotating the encoder knob allows selecting something and the push-button feature is used to activate the selected feature of mode.

The display remains the same, a 9 LED array in a (very) rough approximation of a meter movement.

When tracking, the LED display indicates a direction error, or perhaps, the direction to the transmitter.

If we move encoder, the current tracking operation is stopped and the LED array turns into a 1-of-9 select. This gives us a visible indicator of what selection we are about to activate.

The selected operating *mode* is defined by commands in the FRAM, allowing the feel of the device to be determined by the end user.

The command sequence is named in a manner that should feel similar to that used in the Fox Transmitter. The command decode is shared with the Fox Transmitter with the specific table being almost identical.

6.2.1 SA818 daughterboard

6.2.2 External Tranceiver

6.2.3 Antenna Rotator

6.2.4 STM32 sub 4

107

112

Chapter 7

13

Board Configuration Notes

Source File: FOX_zNEO_Boards.tex

Board Configuration Notes.

7.1 102-73174-51 Notes

Source File: FOX_zNEO_Board_73174-51.tex

Main Board Configuration Notes.

7.1.1 Configuration Notes

7.1.1.1 R25/R28 Meter Trim Procedure

If a mechaincal meter is being used, R25/R28 adjust the center position and the gain seen by the meteer.

7.1.1.2 JP2 Polarity Selection

...

7.1.2 Assembly Notes

Things to keep in mind when assembling the main board.

Do **not** mount the daughter card sockets without using a daughter card with mating pins to achieve good vertical alignment.

7.1.2.1 J7 selection

J7 must be installed as a right angle header when using the 102-73174-97 SA818 receiver board. This connector protrudes outside the housing. Although J7, as a vertical header, will mechanically fit under the 102-73174-97 board (when using 12mm standoffs) it requires removing the 102-73174-97 board to program the zNEO.

The 102-73174-99 board has a cutout to allow the programming header to be installed vertically.

7.1.2.2 Compass Display Assembly

Nine LEDs form a meter or compass display. This is directly controlled by the zNEO using the PE and PG pins. Software determines if it acts as a compass display or as a meter display.

Overlaying the daughtercard in section 4.36 on page 30 on top of the main board you will notice overlap between LED-1, LED-2, LED-8 and LED-9. The other daughtercard from section 4.35 on page 30 is notched to clear all of the LEDs.

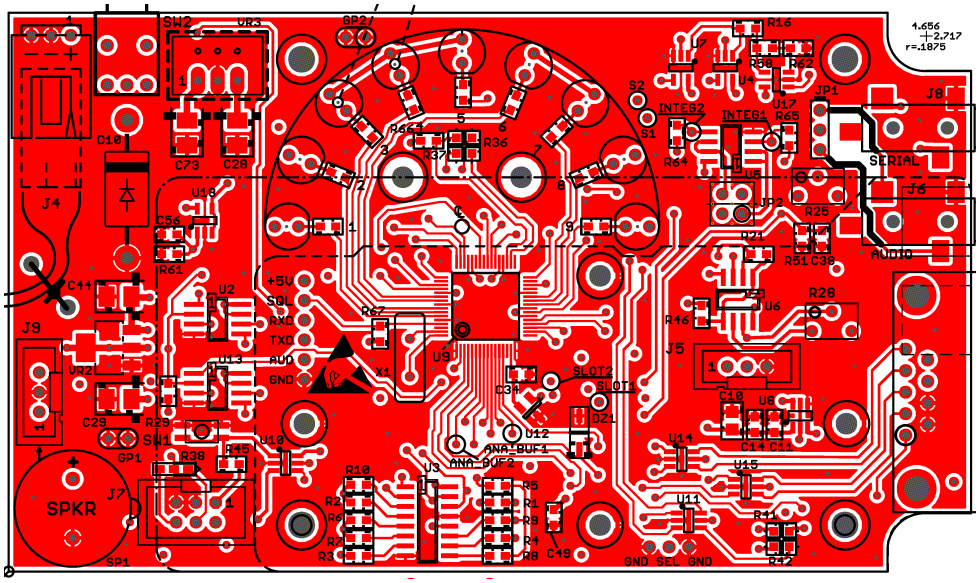


Figure 7.1: Daughter Card Overlap

The LEDs may be installed pointing down, away from the daughtercard, protruding through the bottom of the enclosure.

Also note that the meter LED array is a single library part. This locks the placement of the LEDs and resistors when the library part is built. This also allows the meter array to be placed on the board while keeping all parts correctly oriented.

This method does present a bit of a problem in that the LEDs and resistors are not individually called out in the parts list. Rather the meter array is a single part in the parts list.

...

...

7.1.2.3 Main board J6 connector

The J6 connector is the audio in connector. It should be populated **only** when the board will be used with an external receiver.

...

...

7.1.2.4 Main board J7 connector

The J7 connector is the zNEO programming header. When using the 102-73174-91 daughter card, this connector is obscured or blocked making re-programming the SOC potentially rather difficult.

For the purposes of software debugging, a *tall standing* daughter card may be fabricated with abnormally long interconnect pins and corresponding spacers. All the interconnect signals are low bandwidth so debugging with an extended daughter card should not present any functional issues.

...

7.1.2.5 Main board resistor haywires

Two resistors may be haywired on the backside of the main board near the bottom center as shown in figure 7.1 to accommodate a 74HC4017. When installing a 74HCT4017 they are not necessary as the 74HCT part expects TTL level inputs (much the same as 3.3V CMOS logic levels).

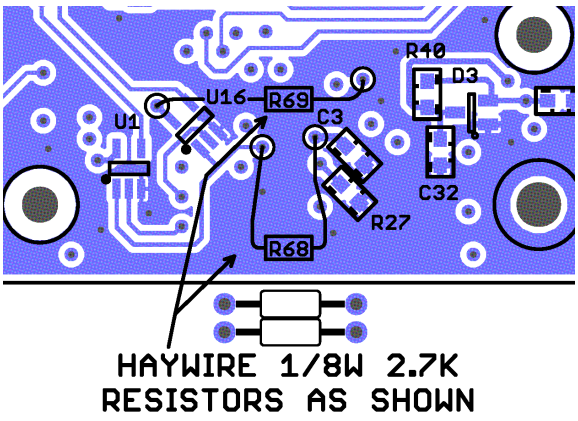


Figure 7.2: Main Board Haywires

These are both axial parts. R68 may be anywhere in the range of 2.7K to 3.9K. The value must be low enough to present a fast rising edge to the 74HC4017 clock input.

R69 may be installed at a much higher value, perhaps 10K, to reduce power a bit as this is, in practice, a static signal.

7.2 102-73174-61 Notes

Source File: FOX_zNEO_Board_73174-61.tex

External Tranceiver Board Configuration Notes.

The design was updated to behave much like the 102-73174-99 board with respect to managing the GPS module. The -61 board will power down the GPS Module when actively scanning the antenna array.

The 102-73174-61 board locates the SMA connectors in the same location used with the 102-73174-82 and 102-73174-99 boards. This allows room for the audio jack to be installed with this board to allow quick daughter card exchange!

7.2.1 Configuration Notes

There is only one configuration jumper on the daughter card.

7.2.1.1 JP1: Power LED

This jumper allows the power indicator to illuminate. Removing the jumper saves less than 5mA on the +5V rail.

7.2.1.2 JP2: GPS Bypass

Remove this jumper to continuously connect the GPS module to the main board.

With the jumper installed...

7.2.1.3 GPS Wiring

The GPS module is installed at the top edge of the circuit board in the area labeled **GPS LANDING ZONE**. There are some large-ish vias provided for lacing cord if desired. Typically a bit of hot-melt glue should be sufficient.

The GPS module must be prepared with wire-wrap wires affixed to the GPS module. The schematic has suggested wire colors for the **YIC31513PGMGG** module.

The **YIC31513PGMGG** module has only 5 connections to be made!

7.2.2 Assembly Notes

As with all daughtercards, mount the daughtercard to the main board to install the 0.025" signal pins.

7.3 102-73174-73 Notes

Source File: FOX_zNEO_Board_73174-73.tex

Antenna Rotator Board Configuration Notes.

7.3.1 Configuration Notes

...

7.3.1.1 JP1: Power

JP1 is installed to power the rotator card from the main board through the 9-pin cable.

JP1 is left out when the board receives power through the J21 connector. When powered through the J21 connector, there is no ON/OFF control, the board is always on.

7.3.2 Assembly Notes

There are no daughter cards to attach, hence no signal pins that require alignemnt.

7.3.2.1 J10: Receiver RF

J10 may be installed as a BNC or as an SMA connector. Both are rather difficult to remove if you choose incorrectly.

The SAM connector is probably the more robust connector.

7.4 102-73174-82 Notes

Source File: FOX_zNEO_Board_73174-82.tex

External Tranceiver Board Configuration Notes.

The 102-73174-82 board locates the SMA connectors to allow the audio jack to be installed on the main board. The 102-73174-61 and 102-73174-99 boards keep the SMA connectors in this same position to allow quick daughter card exchange!

7.4.1 Configuration Notes

There is only one configuration jumper on the daughter card.

7.4.1.1 JP1: Power LED

This jumper allows the power indicator to illuminate. Removing the jumper saves less than 5mA on the +5V rail.

7.4.2 Assembly Notes

As with all daughtercards, mount the daughtercard to the main board to install the 0.025" signal pins.

7.5 102-73174-97 Notes

Source File: FOX_zNEO_Board_73174-97.tex

Internal Receiver Board Configuration Notes.

7.5.1 Configuration Notes

Jumpers and trims

7.5.1.1 R25: audio level to main board

Start with this turned fully counterclockwise to deliver maximum volume to the mainboard. Note that the speaker volume is independant of the amplitude sent to the main board.

7.5.1.2 R16: amplifier and speaker volume

Initially, turn this pot fully counterclockwise to produce maximum volume.

7.5.1.3 JP1: Power Indicator

This jumper allows the power indicator to illuminate. Removing the jumper saves less than 5mA on the +5V rail.

7.5.1.4 JP4: Amplifier Shutdown

This jumper may be left installed.

7.5.2 Assembly Notes

As with all daughtercards, mount the daughtercard to the main board to install the 0.025" signal pins.

7.5.2.1 R16, R5, R4

If volume is inadequate, and the audio signal is not clipped, replace R4 with a 220K resistor and R5 with a 22K resistor.

Increase R4 to 270K for additional gain. Keep in mind that the amplifier (U10) is limited is its output power.

7.5.2.2 h

J

7.5.2.3 i

J

7.6 102-73174-99 Notes

Source File: FOX_zNEO_Board_73174-99.tex

Internal Receiver Board w/GPS Configuration Notes.

The 102-73174-99 update moved the SMA connectors to the same location used with the 102-73174-61 and 102-73174-82 boards. This allows room for the audio jack to be installed with this board to allow quick daughter card exchange!

7.6.1 Configuration Notes

Jumpers and trims

7.6.1.1 R35: audio level to main board

Start with this turned fully counterclockwise to deliver maximum volume to the mainboard. Note that the speaker volume is independant of the amplitude sent to the main board.

7.6.1.2 R36: amplifier and speaker volume

Initially, turn this pot fully counterclockwise to produce maximum volume.

7.6.1.3 JP1: Power Indicator

This jumper allows the power indicator to illuminate. Removing the jumper saves less than 5mA on the +5V rail.

7.6.1.4 JP4: Amplifier Shutdown

This jumper may be left installed.

7.6.1.5 J6: Selected Antenna

This SMA is provided to allow this one board to function with the on-board receiver or as a simple antenna switch when using an external tranceiver.

Build the board up without any of the SA818 support circuitry. In particular, the Low Pass Filter should be left unpopulated A receiver may then be connected to J6 and operated like the 102-73174-82 board.

7.6.2 Assembly Notes

As with all daughtercards, mount the daughtercard to the main board to install the 0.025" signal pins.

7.6.2.1 R16, R5, R4

If volume is inadequate, and the audio signal is not clipped, replace R4 with a 220K resistor and R5 with a 22K resistor.

Increase R4 to 270K for additional gain. Keep in mind that the amplifier (U10) is limited is its output power.

7.6.2.2 U3/Q1 one shot

Q1 was added and U3 was changed to an inverting push-pull gate to reduce the leakage current affecting C52 (Q1 has very low drain-to-source leakage).

We can revert to using an open collector driver by removing Q1 and populating U3 with a 74LVC1G07 open-drain driver.

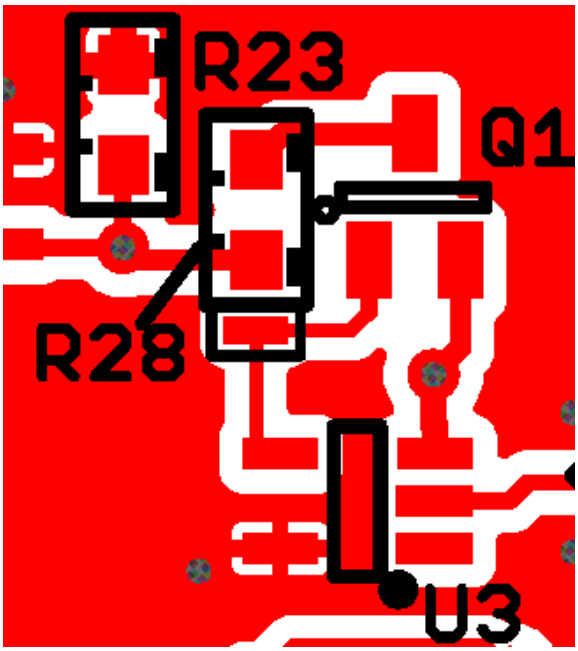


Figure 7.3: Q1 bypass

The exposed pas below R28 may be jumpered to the adjacent pin of R28 to bypass Q1 after Q1 is removed.

7.6.2.3 GPS Wiring

The GPS module is installed at the top edge of the circuit board in the area labeled **GPS LANDING ZONE**. There are some large-ish vias provided for lacing cord if desired. Typically a bit of hot-melt glue should be sufficient.

The GPS module must be prepared with wire-wrap wires affixed to the GPS module. The schematic has suggested wire colors for the **YIC31513PGMGG** module.

The **YIC31513PGMGG** module has only 5 connections to be made!

7.7 102-73170-20 Notes

Source File: FOX_zNEO_Board_73170-20.tex

Simple Antenna Switching Board Notes.

The 102-73170-20 board is a stand-alone antenna switch that may be used to aurally search for hidden transmitters.

7.7.1 Configuration Notes

Jumpers and trims, and other parts

7.7.1.1 R3: Swtching Frequency

R3 is normally not installed, rather we operate the antenna switch at a fixed frequency close to 1KHz. R3 may be installed along with R2 in place of R8. R3 pads allow several different parts to be installed.

7.7.2 Assembly Notes

Almost all surface mount to reduce cost.

7.7.3 Schematics

Timing Generator

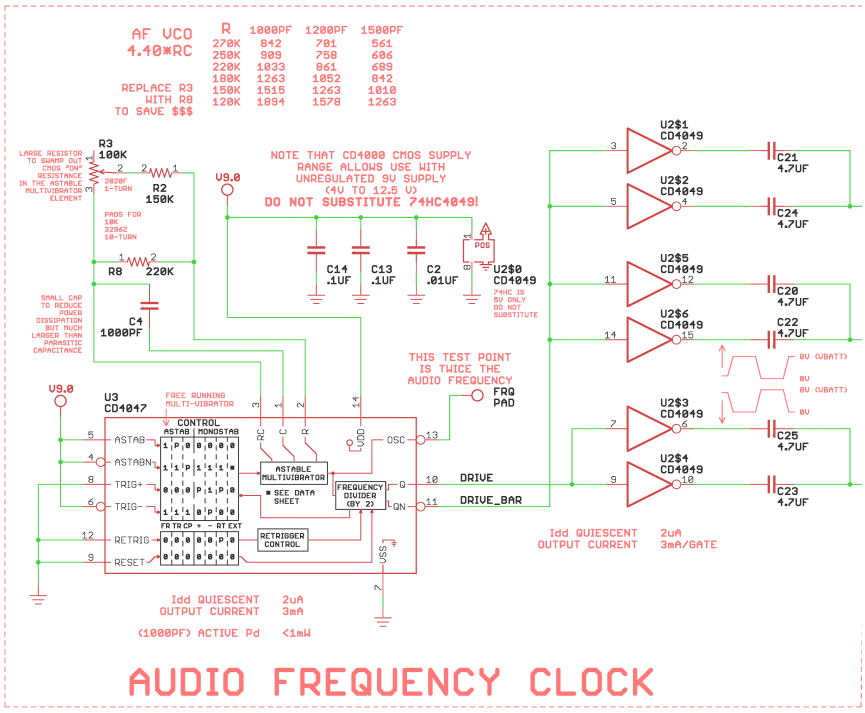


Figure 7.4: Timing Generator

We make use of a CD4047 to produce the master timing signal. All the CD4047 requires to generate a simple swuare wave is an appropriate capacitor and resistor. R8 and C4 produce an antenna switching rate close to 1KHz.

The reference signal on pin 13 runs at twice the switching frequency seen on pins 10 and 11. Also keep in mind that pins 10 and 11 are opposite polarity,] exactly what we need to drive the antenna switch.

R3 and R2 are **not** populated.

U2 provides current drive to the PIN diodes in figure 7.6.

C20 .. C23 provide DC isolation for the PIN diodes.

The 400 series logic is powered directly by a 9V battery. The gates (U2 and U3) **must** be 4000-series CMOS **not** 74HC series to work reliably with a 9V supply.

RF Isolation

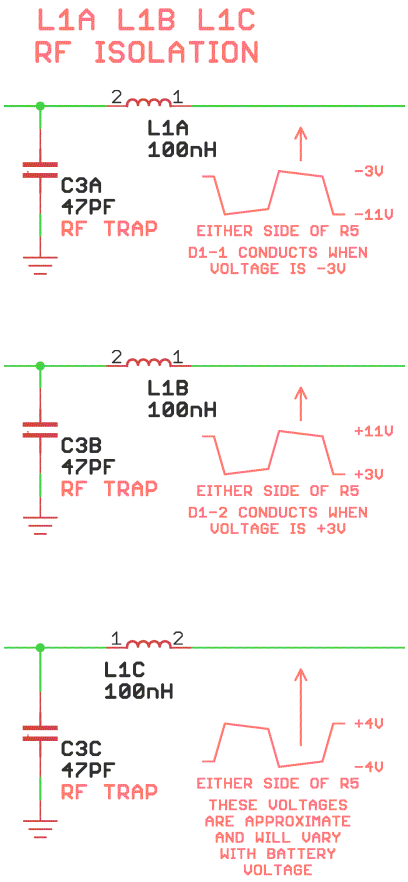


Figure 7.5: RF Isolation

The L1A .. L1C inductors isolate the RF signal that runs between antenna and the radio.

C3A .. C3C are provided to keep RF energy away from the digital logic.

The signal nets on this fragment come directly from the previous schematic.

Operating notes are in red with approximate voltages seen on the prototyp unit.

RF Switching

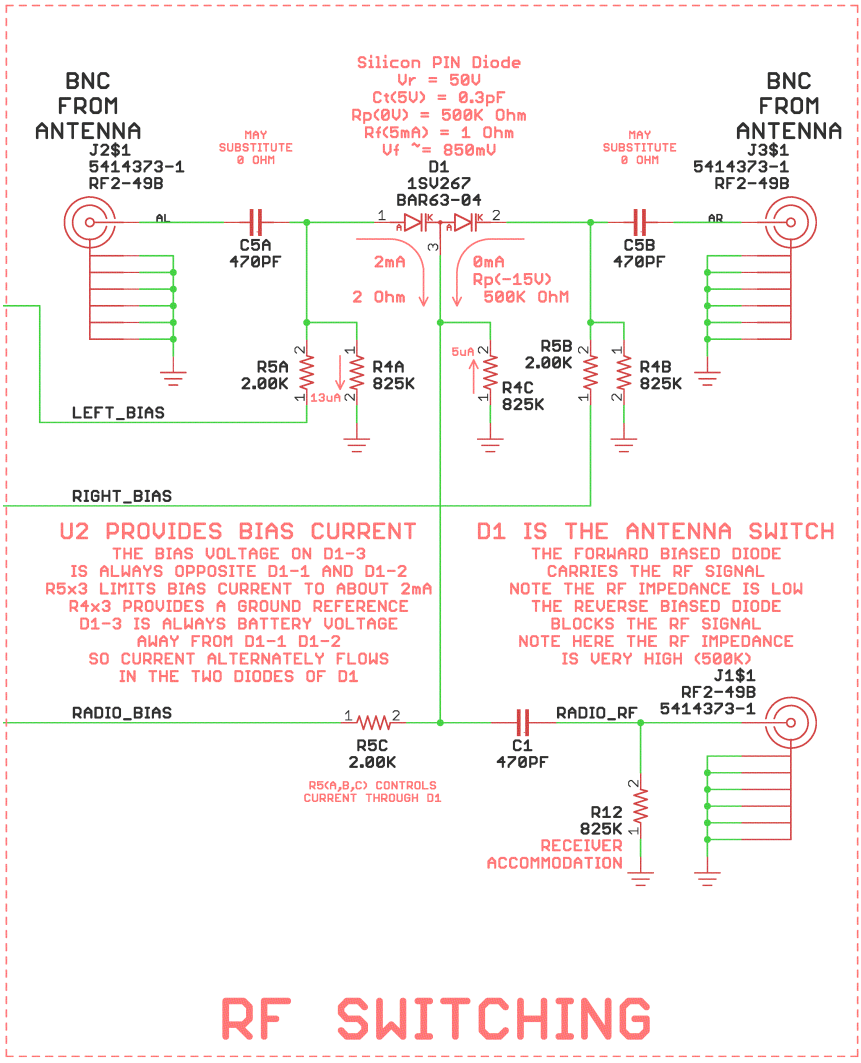


Figure 7.6: RF Switching

Here we have a simple switching network that connects either J2 or J3 to the receiver on J1.

Looking back at figure 7.4, the top gates of U2 are all in-phase (U21, $U22$, U25, $U26$) and the remaining 2 gates (U23, $U24$) are inverted with respect to the first four.

The signal nets on this fragment come directly from the previous schematic.

The **LEFT_BIAS** and **RIGHT_BIAS** nets are the same polarity, but operating with differing DC offsets due to D1. The **RADIO_BIAS** net operates around zero, again due to D1. The R4A .. R4C resistors serve to force the **RADIO_BIAS** net to operate with no DC offset, it switces symetrically about zero.

The **LEFT_BIAS** and **RIGHT_BIAS** nets, then, become biased above zero and below zero to allow current to flow through D1. In operation, one of the diodes of D1 is always forward biased and, therefore, conducting. The voltage drop across the forward biased diode is what would be expected of a silicon diode of about 700mV.

The **LEFT_BIAS**is kept below ground level and **RIGHT_BIAS** ends up above ground.

7.7.4 PIN Diode plots

We can only look at two channels with the available test equipment.

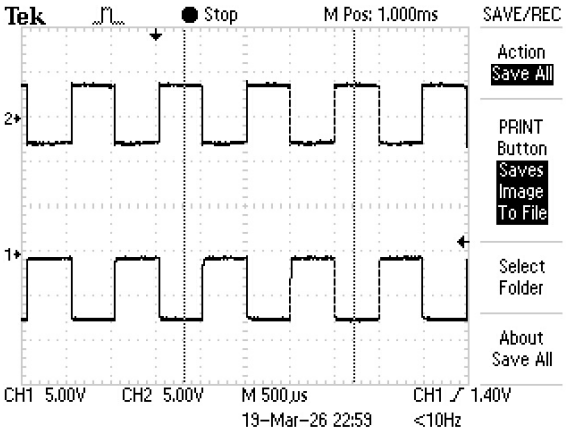


Figure 7.7: PIN Diode plots

Channel 2, on top, is the anode/cathode common pin of D1. As expected it is centered around ground.

Channel 1, on the bottom, is pin 1 of D1. When channel 1 is HI, channel 2 is LO, and the diode at pin 1 of D1 if forward biased keeping the voltage drop below 1 volt.

When we switch, the diode at pin 1 of D1 is reverse biased. With almost no current flowing through this side of D1, the voltage on pin 1 is free to swing the full battery voltage which ends up reverse biasing the junction to par almost 1.5 times battery voltage.

The other diode in D1 is reversed. D1-1 is the anode of the pairt and D1-2 is the cathode. D1-3 being the cathode/anode of the diode pair.

As the D1-2 cathode effectively points the opposite direction, it will look just like the D1-1 anode, but mirrored about ground.

Chapter 8

13

STM32 Issues.tex

Source File: FOX_STM32_Issues.tex

Issues encountered designing and debugging the STM32 design.

8.1 102-73174-41 nRST

Source File: FOX_STM32_Issues_nRST.tex

STM32 Nrst Problem.

There seems to be an issue getting the STM32U575 to deal with a low-going pulse on the nRST pin when supplied by a CMOS gate (rather than using a simple push-button).

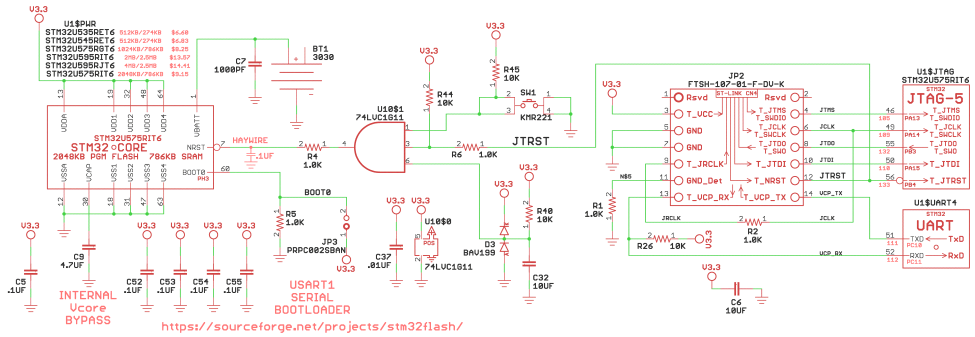


Figure 8.1: STM32 Power/Reset

Here we have the basic power connections for the STM32U575.

8.1.1 Circuit Description

The U1$PWR block (on the left) shows all the power pins as well as the reset pin (**nRST**) and the **BOOT0** pin. The initial design did not have a .1UF cap on the **nRST** pin, but that was easily added to the board.

Not show on the fragment of the schematic is a linear 3.3V low-dropout regulator that supplies the 3.3V rail. This, in turn, comes from a 5V switch-mode regulator being supplied by a six or eight cell alkaline pack. Bulk capacitors, as needed by the regulators, appear on the circuit board.

The reset network is centered around U10. This arrangement simply seperates the various reset sources to isolate noise sources and to keep the manual reseet switch contacts isolated from the POR cap (C32). Were SW1 simply placed into the POR circuit (C32, R40 and D3) it would have to carry the current supplied by C32 were it to be shorted to ground.

The manual reset, then, boils down to a simple switch (SW1) with a pull-up resistor (R45). The contacts of SW1 eed only handle the current through R45 (only 30mA). THis is simply to prolong the life of the switch by not carring the discharge current of the POR circuit should the switch simply short C32 to ground.

The third source comes from the programming header, JP2. This connector mates with a STLINK-V3SET programmer from the chip vendor. The STLINK-V3SET provides for programming STM32U575.

The **JTRST** net is routed to the reset network to allow the STLINK-V3SET to reset the STM32U575. This connection can be broken by removing R6 if necessary. With R6 removed, R44 keeps this input to U10 in an inactive state.

The **JTAG** port of the STM32U575 is represented by the U1$JTAG schematic symbol. It is connected to the programming header as shown. The STLINK-V3SET also provides a serial port the is connected to the STM32U575 UART4 (U1$UART4 schematic symbol)

8.1.2 Problem Description

nRST appears to leave the STM32U5 inoperative when applied at any time other than following application of power. Serial traffic is expectd through UART4 following a reset. When the board is powered on, the STM32U5 software runs as expected; initial traffic through UART4 and then the STM32U5 interacts through UART4, accepting typed in commands and producing expected response traffic.

If the reset switch (SW1) is used to reset the STM32U5, it does not produce the expected serial traffic and no longer communicates through UART4.

A power-cycle of the board produces expected serial traffic and the board interacts (through UART4) as expected.

The board does not reset correctly after being programmed (using a STLINK-V3SET programmer). The program flash is, however, correctly programmed. THe updated flash image works folloing a power-cycle.

This schematic fragment shows power, BOOT0, nRST and programming connections to the STM32U575RIT6 (U1 is 64LQFP package). Bulk capacitor(10UF) appears elsewhere on the schematic near the 3.3V (linear) regulator. Voltage levels appear correct and there appears **not** to be any excessive noise on the 3.3V rail.

The STM32U5 is being clocked internally (HSI at 16MHz). The clock seems to be working correctly as evidenced by the STM32U5 being able to communicate through UART4 (U1$UART4) at the programmed rate of 115,200 b/S.

Looking at logic levels on either side of R4, they appear to be clean and are at expected levels (near ground and 3.3V) when using SW1 to control U10 output. No excessive noise is apparent. Although a .1UF cap on the nRST net was missed in the initial board layout, a cap was added to the board as indicated on the schematic fragment (HAYWIRE note).

I get the impression that I have missed some setup feature in CubeMX (or similar ??) that alters the default behavior of the nRST pin. nRST start (follow application of power) in one mode/state and the software alters this. Popping the reset button (i.e. assert nRST) after the change prevents nRST from behaving correctly.

8.1.3 From: TDK

What gives you that impression?

> Simply a first-order guess to establish a starting point from which to proceed with debugging.
>
> I am looking for a clue as to what parts in the STM32U5 are un-affected by the nRST; What extra actions does POR do that simply asserting nRST doesn't?
>
> The seemingly odd nRST behavior is simply the first apparent symptom, things appear to work as expected at POR (note in the attached schematic that nRST is driven low for tens of milliseconds when power is switched on) but the code doesn't restart as expected following application of nRST (via the reset button) following power-up.
>
> So, in essence, what did I do wrong? (probably in CubeMX) that gets set/configured that is insensitive to a non-POR assertion of a nRST reset?

Stay objective, don't guess.

> Well, I'm looking for a toe-hold on which to start looking. It's rather hard to be objective when there's 3600+ pages of MCU reference and 3800+ pages of HAL reference.
>
> Using CubeMX to configure and generate the HAL layer covers most of the ground, but it's not too difficult to end up unintentionally misconfiguring things.

Is there anything in the option bytes that can change how NRST is handled?

> The code doesn't alter the option bytes as far as I can tell. Or perhaps better stated, the user supplied code is not attempting to alter option bytes (HAL layer???)

8.1.4 From: Andrew Neil

Andrew provides a good link to get started.

> Here I am using gdb/st-util to talk to a STLINK-V3SET. Trying to remember how to use gdb remotely.

8.1.5 More Details

Another datapoint, perhaps:

> The application also builds on a NUCLEO-U575ZI-Q. This is a NUCLEO144 board whereas my design is a 64LQFP. Pin allocations are, of course, much different, but with the NUCLEO144 board the reset works as expected.

8.1.6 Now Operational

Arrgh!!! It now works following some code work.

OK, the HAL setup code calls **Error_Handler();** when it encounters an error. In turn **Error_Handler();** hangs the processor in a tight loop; This is typically before the UART4 (which talks to the STLINK programmer) has been initialized so there is no quick and dirty path for notification.

User code rearranged to perform all setup for the 2 UARTs that talk to the outside world (UART4 and USART1) as early as possible, before user GPIO, DMA, ADC, etc. setup is done. This seems to have addressed whatever the offending issue is.

Some startup text shows up earlier such that any messed up user-land code doesn't hang the machine before something comes out.