

# The University of Oklahoma Nielsen Hall SRT Operator's Manual

JOHN TOBIN<sup>12</sup>  
NICKALAS REYNOLDS<sup>23</sup>

February 15, 2017

<sup>1</sup>jjtobin@ou.edu

<sup>2</sup>GitHub: <https://github.com/OUsrc/srtn.git>, Group Email: [nhnradiotelescope@groups.ou.edu](mailto:nhnradiotelescope@groups.ou.edu)

<sup>3</sup>nickreynolds@ou.edu

# Abstract

The University of Oklahoma's Small Radio Telescope (3m) was sponsored by Dr. John Tobin as an outreach and educational tool for the Norman Community. The use of the telescope helps to aid in expanding one's knowledge of radio astronomy, engineering principles, and proper research methods. The kit was implemented off of MIT Haystack's SRT, fabricated through RF Ham Design, constructed by a team of students and professors<sup>1</sup>, and the code was provided by MIT Haystack under the MIT Public License.

Information specifically regarding the MIT Haystack Observatory can be found at <http://www.haystack.mit.edu/edu/undergrad/srt/index.html>

## Additional Information

The website<sup>2</sup> for this file contains:

- A link to the downloadable PDF and L<sup>A</sup>T<sub>E</sub>X code.
- The SRT source code.
- Useful scripts for observing and parsing the data

## Acknowledgements

- Hardware design, development of software, and instructions: ([www.haystack.mit.edu/edu/undergrad/srt/](http://www.haystack.mit.edu/edu/undergrad/srt/))
- Fabrication of the SRT parts: (<https://www.rfhamdesign.com> )
- Parse code and instructions: (<https://github.com/BenningtonCS/Telescope-2014/wiki>)

---

<sup>1</sup>Prof: John Tobin; Grad students: Paul Canton, Hyunseop Choi, Nickalas Reynolds, Rajeeb Sharma, G-PSI; Undergrad Students: Jacob Gill, Lisa Patel, and Brian Stephensen; Staff: Barry Bergeron, Andy Feldt, Debi Schoenberger, John Snellings, Adrienne Wade, and Joel Young

<sup>2</sup><https://github.com/OUsrc/srtn.git>

# Contents

<b>1</b>	<b>Condensed Procedures</b>	<b>3</b>
<b>2</b>	<b>Overview</b>	<b>4</b>
<b>3</b>	<b>Building the Hardware</b>	<b>5</b>
3.1	Rotor . . . . .	5
3.2	LNA . . . . .	5
3.3	Feed . . . . .	5
3.4	Antenna . . . . .	6
3.4.1	Pier . . . . .	6
3.4.2	Dish . . . . .	7
<b>4</b>	<b>SRT Server Computer</b>	<b>8</b>
<b>5</b>	<b>SRT Software</b>	<b>9</b>
5.1	Commands . . . . .	9
5.2	Setup . . . . .	9
5.3	Compiling . . . . .	9
5.4	Files . . . . .	11
5.5	Command Files . . . . .	12
5.6	Example Procedure . . . . .	13
<b>6</b>	<b>Data Reduction</b>	<b>14</b>
6.1	Parsing the Data Files . . . . .	14
6.2	Gnuplot . . . . .	15
6.3	Formulas . . . . .	15
6.4	GitHub . . . . .	15
6.4.1	Quick Command Breakdown . . . . .	15
6.4.2	Proper Usage . . . . .	15
<b>7</b>	<b>Complete Parts List</b>	<b>17</b>
7.1	Rotor . . . . .	17
7.2	Mount & Dish . . . . .	17
7.3	Feed & LNA . . . . .	18

# Chapter 1

## Condensed Procedures

This list of procedures is extremely condensed and it is expected you are fairly experienced with operating the telescope before using these. If you are unsure of something, ask.

### # Parameters

- 1) Edit CALMODE in srt.cat file
- 2) CALMODE 0 - used for yfactor solving. Point to cold sky, cal, place absorber, get tsys and tcal
- 3) CALMODE 2 - only does bandpass correction, okay if no absorber or tree
- 4) CALMODE 3 - if stow looks at absorber like trees.
- 4) Verify target frequency and radial velocity
- 5) If velocity is  $>50\text{km/s}$ , use radvel.py to adjust the frequency

### # Pointing

- 1) Slew to very bright source (sun)
- 2) Use npoints to correct and input offset

### # Simple Cal (with absorber)

- 1) Slew to cold sky
- 2) hit cal on calmode 20
- 3) place absorber
- 4) should have flat bandpass

### # Simple Cal (without absorber)

- 1) Stow
- 2) hit cal on calmode 2
- 4) should have flat bandpass

### # Advanced Cal

- 1) Do above
- 2) Move to blank sky patch and record
- 3) Average  $\sim 5$  power measurements (from terminal). This is  $P_{\text{cold}}$
- 5) Remove absorber and average  $\sim 5$  power measurements. This is  $P_{\text{hot}}$
- 6)  $T_{\text{cold}} \sim 3\text{k}$  .....  $T_{\text{hot}}$  is average ambient temperature or thermometer measurement of absorber
- 7) Run python script tsys.py to find tsys
- 8) Edit srt.cat file with new tsys and reload program

### # Source

- 1) Reopen srt.n
- 2) Slew to target and record
- 3) Hit the beamsw (for faint signals)
- 4) Wait for at least  $\sim 45$  iterations (counter in bottom spectrum window)
- 5) re-hit beamsw to turn it off/ hit record to stop recording
- 6) This target is finished

### # End of Observation

- 1) Always stow at end of the observation
- 2) Exit the SRTN program, do **NOT** turn off the computer
- 3) Turn off the Controller, Receiver, and Tuner
- 4) Move data files to appropriate directories

# Chapter 2

## Overview

The capabilities of the SRT are as follow:

- Frequency Adjustments from 1420MHz
- Spectral Line Gathering
- Continuum
- Drift Scans
- Point Maps

Aperture	3.0 meters
Frequency Range	1420 – 1470 MHz
Pointing Accuracy	0.2 Degree
Beam Width <sup>1</sup>	4.92 Degrees
wind speed	120 km/h

# Chapter 3

## Building the Hardware

The tools needed were: protective gloves, protective goggles, electric hand drill,  $\frac{1}{4}$  in drill bit, hammer, mallet, riveter, rivets, wire cutter, sharpie, electric cut-off tool (metal blade), soldering iron, mini-pliers, wrench, and a bubble level.

### 3.1 Rotor

\*Working on this section

### 3.2 LNA

The low-noise amplifier is located in an environmental resistant box mounted on the rear of the horn, minimizing the blockage of the aperture. The LNA consists of two ultra-low-noise amplifier modules, a band-pass filter, and a bias-tee to power the amplifiers. Two holes will be made in the aluminum case.

- Drill the two holes with a  $\frac{1}{4}$ " drill bit.
- Remove the nut/washer from one of the SMA-F to SMA-F bulkhead connectors
- Secure the O-ring against the flange and insert the longer side of the connector into the case and secure it with the washer/nut, using a wrench.
- Connect the filter to the "out" port of the ZX60-1614LN-S amplifier and the SMA-M to SMA-M adapter to the "in" port. Insert the 3" SMA cable to the other end of the filter
- Put this total assembly into the box and connect the adapter to the connector
- Attach the other end of the SMA cable to the "in" port of the second amplifier.
- Section two pieces of 12V wire and tin the ends of them and the 12V pins of the amplifiers
- Connect the two 12V pins with one wire and attach the other to the 12V pin of the amplifier not connected to the adapter
- Attach the 3" SMA cable to the "out" port of the second amplifier and to the "RF" port of the bias-tee.
- Connect the other SMA-F to SMA-F connector and insert it through the second hole
- Connect the "RF&DC" port of the bias-tee to the connector
- Tin the 12V pin of the bias-tee and solder the remaining 12V wire to this pin
- Place the rubber gasket lid on the case and screw it in.

### 3.3 Feed

This apparatus consists of a helical antenna inside an aluminum cavity or horn. The helix is a copper tape stretched around the polystyrene foam cylinder, with a folded foil plate to match impedance. The feed is constructed from a 2.5in Styrofoam cylinder. The height of the cylinder is determined by the desired wavelength and is  $\frac{1}{4}$  wavelength.

- Cut the cake pan as indicated on the drawing of the feed.
- Cut the foam rod to 73.7mm in length and drill a  $\frac{1}{4}$  " hole down the long axis of the rod.

- 1mm from the edge of the rod, draw a mark. Make two more marks vertically above this mark each 30mm above the previous mark
- Cut out a strip of the copper tape 4x439mm. Tape one end at the lowest mark and without taping the rest wrap the strip around the foam such that the lower edge of the strip is flush between the previous two marks.
- With a ruler, verify the helix has vertical spacing of 30mm. Once you get the strip in a good position, trace the strips upper edge.
- make a mark 15mm from the lower end of the strip and bend the foil back. Make another mark 13mm from the first mark and then a final mark 26.25mm past that
- Remove the adhesive backing from the bent foil and fold the copper onto itself. Place the bend on the lowest mark on the foam and apply the foil to the foam. Keep the helix aligned with the previously designated points.
- Cut out a strip of foil 26.25x13mm and leave an additional 3-4mm wide strip on one of the long edges. This will be the basis to form the impedance matching section.
- Fold this strip so it is 90 degrees to the rest of the foil. Center it on the helix between the two marks that are 26.25 mm apart and solder the strip to the helix.
- Solder the bent over power of the end of the foil to the SMA connector.
- Place a  $\frac{1}{4}$ " O-ring to the SMA connector and insert into the cake pan center hole.
- Section out a 63mm square piece of PC board and drill a  $\frac{1}{4}$ " hole in the center and a  $\frac{1}{4}$ " indent as indicated.
- Run the bolt through the center of the foam, board, and the cake pan and secure it with hand tightened nut and washer. Make sure the soldered end of the SMA connector is against the bottom of the cake pan and not propped by the PC board.
- Test the apparatus with a network analyzer by attaching the network analyzer to the SMA connect and set to measure S11. Should read roughly -20 dB.
- A shorted  $\frac{1}{4}$  wavelength stub should be added to the feed to act as a DC path for lightning and static protection for the amplifiers
  - Cut out 42mm of long piece coaxial cable
  - Strip the shield and insulation away ~2mm on one end and 4mm on the other.
  - Bend over the 4mm end and solder it to the shield
  - Solder the 2mm side to the center connector of the stub in the SMA connector and solder the shield to one of the ground pins of the SMA connector
- The LNA mount plate will now be made. Drill the holes as indicated by the diagram
- Secure the plate to the back of the pan with four machine screws and nuts through the holes in the corners. Tighten with a wrench
- Thread a nut onto each of the 1" machine screws and place a lock washer on top. Place the LNA onto the machine screws and connect the port of the LNA to the SMA connector on the feed with a SMA-M to SMA-M right angle connector
- Using the nuts, level the LNA case on the pan.
- thread the final nuts onto the machine screws and tighten with a wrench.
- Attach the quadrapod legs to the feed

## 3.4 Antenna

### 3.4.1 Pier

The dish was "portably" mounted on the roof of Nielsen Hall. There is 1000lbs of ballast holding the pier in place as per wind regulations however, it is not permanently mounted to the building. The mount has a wide base to support a 3in pier pipe in which the SPID rotor mounts on top of. An adapter is needed to convert the pier out from 3in to 2.875in. The adapter is held in place by 12bolts and re-leveled. Secure the PID rotor to the adapter. The assembly of the mount is fairly straightforward.

### 3.4.2 Dish

The dish is constructed of 6mm galvanized aluminum. The ribs are mounted onto the 2 “hubs” (CNC milled centers) with bolts. The hubs are 20cm round and 10mm thick aluminum. Each rib is secured with 2 bolts into the hub. Before you tighten the bolts, make sure the ribs are spaced evenly as they provide the support for the mesh. Fluctuations in the mesh reduce the efficiency of the telescope.

Feed the aluminum wire through the four co-centric holes cut in the ribs to secure them in place. The wire will need to be cut precisely to size and crimped in place with the female-female inserts included.

It is suggested at this point to use large sheets of plotter paper to directly map the size of the mesh sheets needed. This will ensure accurate cutting of the mesh. The mesh should be of an appropriate size, completely overlapping both sides of the rib and with ~4in of spare material on the outer rim to allow fastening. Make sure both the cut mesh and the ribs are labeled to correspond with the appropriate piece. Once all of the mesh needed is cut, it is time to mount them to the dish.

Line the mesh sheets up to the ribs of the telescope. Use the hand drill to score the mesh and drill through the ribs and 2x20mm aluminum strips. You can then use the riveter to secure the mesh onto the ribs. Use about 6 rivets per rib to secure the mesh. Once all of the mesh is secure, remove the excess mesh around the hub using the cut-off tool with the metal attachment.

Use zip-ties to secure the mesh onto the aluminum wires fed through the ribs. Space them as evenly as possible and make sure they are held secure. The mesh should be as smooth as possible and follow the curvature of the ribs. Also use zip-ties to wrap the excess ~4in of mesh around the 3x20mm aluminum strip on the outer rim and secure it in place. Remove all unnecessary mesh excess with the cut-off tool.

Use several people to mount the dish onto the rotor using the provided bracket and bolts. The dish needs to be securely fastened to the rotor and centered. Use the mallet to “ease” it into its final resting position. The dish itself isn’t extremely heavy but acts as a giant sail and can be very difficult to secure in place.

Mount the quadrapod legs into the dish. You will have to play around with the placement of the legs such that the focus of the dish is on the axis of the helix about half way along its length. You can use string to measure the true diameter and then measure the distance from the string to the center of the dish. The focus is given by this formula:

$$f = \frac{D^2}{16 \times d} \quad (3.1)$$

Once the focal length is found and the feed is mounted, use coaxial cable to connect the feed to the receiver. Run the wire down one of the legs and secure it with zip-ties. Secure it along the rib on the back of the telescope and bundle it with the other wires. These wires should be bundled carefully and should feed into protective housing such that environmental effects don’t interfere and such that the telescope has full range along its slew without disrupting the bundle.



## Chapter 4

# SRT Server Computer

First know that the server computer, the SRTN software, and all of the following commands are assuming you are on a Linux OS. The SRTN software and the VNC software are available for other OS architectures but are not covered in the scope of this Manual. You will also have to download and install the VNC software<sup>1</sup>.

Understand that access to the telescope is a privilege that is administered by Dr. Tobin<sup>2</sup> and any abuse of the system will result in that access being revoked.

The Nielsen Hall SRT is controlled through a network locked, local server. This server does not follow the standard authentication protocol followed by the University nor the department. In order to gain access to the telescope. You will have to populate and send your ssh public keys to Dr. Tobin. Follow these commands to do this:

```
1 # First verify you have ssh keys generated, or generate them
2 ~$ ls ~/.ssh/
3 # There should be a file named id_rsa.pub. If there is not a directory or not this file then use this
   command
4 ~$ ssh-keygen
5 # specify the path to download, typically in /home/username/.ssh/
6 # type in a password to use to unlock the file.
7 ~$ cp ~/.ssh/id_rsa.pub ~/id_rsa_NAME.pub # copy the file to your home directory with your name
```

Now email this file to Dr. Tobin and wait until he confirms that he has added your public key to the server computer. Once you have access you can now login to the telescope computer. Please understand how the SRTN software works **before** controlling the telescope. It is also suggested to have a senior member of the group assist in the first run. To login to the computer use these commands:

```
1 ~$ vncviewer -Shared -ViewOnly -via jjtobin@reber.nhn.ou.edu reber:1 # Viewing the telescope
2 ~$ vncviewer -Shared -via jjtobin@reber.nhn.ou.edu reber:1 # Controlling the telescope
```

Since you are just starting out, use the command with the `-ViewOnly` flag just to see what the server looks like.

You will be prompted for a VNC server password which is `SRT1420MHz`.

To walk you through the commands: VNC is the program used that uses ssh protocol to provide a graphical interface between two computers, `vncviewer` is the command to view an existing VNC server on a host computer, `-Shared` allows multiple users to access the server at once, `-ViewOnly` does not send keyboard or mouse controls to the hosting server, `-via` specifies to route to the domain and specific server, `jjtobin@reber.ou.edu` is the domain under which the server presides, `reber:1` designates the computer and server window. There are probably a few VNC servers operating on the server computer at one time and can be listed with `vncserver -list`. However, unless you are not the one that created those windows, please do not close them as they might be in use.

“With power comes great responsibility”– Uncle Ben

---

<sup>1</sup><https://tigervnc.org>

<sup>2</sup>jjtobin@ou.edu

# Chapter 5

## SRT Software

Do not change the code of the SRT software on the SRT controller computer until you have thoroughly tested the code and passed it by the senior antenna operator.

### 5.1 Commands

For extended help on any commands, within the terminal window, you can type `man [COMMAND]` e.g. `man tar` and it will display the internal manual for the given command, tar in this case.

```
1 cd # changes directory to home directory ~/ or if given input e.g.
2 cd ~/Downloads/ # will move to Downloads directory in home
3 ls # will list all of the files in the current director
4 mv /path/file1 /path/file2 # will move file1 to file2
5 mkdir srt # makes a new directory called srt, flag -p will copy directory structure
6 tar # (un)compression algorithm, x is uncompress, z is gunzip format, v is verbose, f specifies file
```

### 5.2 Setup

First download the source code for the SRT either by downloading directly from the GitHub<sup>1</sup> or by using the command on line 3.

```
1 ~$ mkdir -p ~/srt/scripts/
2 ~$ cd ~/srt/
3 ~/srt$ git clone https://github.com/OUst/srtn.git # can ignore this line if you directly downloaded
4 ~/srt$ cp -r ./srtn/software/srtnver5_reber ./
5 ~/srt$ cd srt_ver5
6 ~/srt/srt_ver5$ ls
```

You should now be in the SRT main code directory and the contents should be on your screen. The code should be able to run immediately or recompiled as necessary.

### 5.3 Compiling

If there are issues with the C compiler not able to locate certain libraries, try searching for solutions to that problem or send me an email<sup>2</sup>. There should be a file called `srtnmake` within the directory. This is the main compiler file that will gather the other C files and compile them in the correct order. When you make changes to the other files within the directory (other than the `srt.cat` file) you will have to recompile the program so it can implement the changes.

```
1 ~/srt/srt_ver5$ ./srtnmake
```

---

<sup>1</sup><https://github.com/OUst/srtn.git>

<sup>2</sup>[nickreynolds@ou.edu](mailto:nickreynolds@ou.edu)

You will probably get a lot of warnings, this is fine and normal errors can look like this:

```
1 main.c: In function 'main':
2 main.c:62:12: warning: variable 'secstart' set but not used [-Wunused-but-set-variable]
3 main.c:60:12: warning: variable 'ii' set but not used [-Wunused-but-set-variable]
4 main.c:59:14: warning: variable 'color' set but not used [-Wunused-but-set-variable]
5 main.c: In function 'gauss':
6 main.c:555:16: warning: variable 'j' set but not used [-Wunused-but-set-variable]
7 vspectra_four.c: In function 'vspectra':
8 vspectra_four.c:33:28: warning: variable 'min' set but not used [-Wunused-but-set-variable]
9 vspectra_four.c:33:12: warning: variable 'avsig' set but not used [-Wunused-but-set-variable]
10 vspectra_four.c:31:43: warning: variable 'r' set but not used [-Wunused-but-set-variable]
11 disp.c: In function 'clearpaint':
12 disp.c:440:18: warning: variable 'update_rect' set but not used [-Wunused-but-set-variable]
13 outfile.c: In function 'outfile':
14 outfile.c:16:16: warning: variable 'n' set but not used [-Wunused-but-set-variable]
15 sport.c: In function 'rot2':
16 sport.c:402:25: warning: variable 'i' set but not used [-Wunused-but-set-variable]
17 sport.c:402:17: warning: variable 'status' set but not used [-Wunused-but-set-variable]
18 sport.c:408:15: warning: ignoring return value of 'system', declared with attribute warn_unused_result
   [-Wunused-result]
19 cal.c: In function 'cal':
20 cal.c:18:24: warning: variable 'ixe' set but not used [-Wunused-but-set-variable]
21 srthelp.c: In function 'display_help':
22 srthelp.c:39:14: warning: variable 'color' set but not used [-Wunused-but-set-variable]
23 srthelp.c: In function 'load_help':
24 srthelp.c:254:10: warning: ignoring return value of 'fgets', declared with attribute
   warn_unused_result [-Wunused-result]
25 srthelp.c:258:14: warning: ignoring return value of 'fgets', declared with attribute
   warn_unused_result [-Wunused-result]
26 velspec.c: In function 'velspec':
27 velspec.c:19:14: warning: variable 'color' set but not used [-Wunused-but-set-variable]
28 velspec.c: In function 'vplot':
29 velspec.c:162:15: warning: variable 'jmax' set but not used [-Wunused-but-set-variable]
30 librtlsdr.c: In function 'rtlsdr_open':
31 librtlsdr.c:1267:13: warning: variable 'rt' set but not used [-Wunused-but-set-variable]
32 tuner_r820t.c: In function 'R828_RfGainMode':
33 tuner_r820t.c:2859:11: warning: variable 'LnaGain' set but not used [-Wunused-but-set-variable]
34 tuner_r820t.c:2858:11: warning: variable 'MixerGain' set but not used [-Wunused-but-set-variable]
```

Now, within the same directory should be a file called `srtm`. This is the file that will run the GUI window for the telescope. To run it, simply type

```
1 ~/srt/srt_ver5$ ./srtm
```

and a GUI window should pop up that looks like this:

If you do not have both the receiver dongle and the telescope controller plugged into the computer, the program will fail to initialize. In order to get around this, for testing purposes, go into the `srt.cat` file and edit the lines from

```
1 *SIMULATE ANTENNA —> SIMULATE ANTENNA
2 *SIMULATE RECEIVER —> SIMULATE RECEIVER
```

This will allow you to open the program and test certain commands before committing commands to the telescope untested.

## 5.4 Files

Help on the various C code files in the source directory.

- 60-mcc.rules: udev rules that allows PCI, HID devices, and libusb to work with non-root users
- acml.h: allows calling ACML routines via their C or FORTRAN interfaces
- amdfft.c: fast-fourier transform adaptation software
- calc.c: calibration code
- cat.c: parses the srt.cat file
- cmd.txt: the default file for the commands you want srtn to use
- cmdfl.c: the code to interpret and execute cmd.txt
- d1cons.h: declares values used in code
- d1glob.h: globally declared variables for the code
- d1proto.h: also declares variables for code
- d1typ.h: declares a new struct
- disp.c: code for the GUI
- doindent2: code for parsing whitespace
- fftw2.c: fast-fourier transform
- fftw3.c: fast-fourier transform
- four.c: Handling the different coordinates to angles conversion
- geom.c: for parsing the spherical geometries
- hproto.h: help menu functions
- librtlsdr.c: self consistent library for the dongle
- main.c: the main C file for calling and compile the code
- map.c: Deals with the GUI map controls and drawing.
- outfile.c: writes the data to a file
- PCI-DAS4020-12.1.21.tgz: library for PCI boards
- PCI-DAS4020.h: functions for the PCI boards
- plot.c: plotting the spectra in the GUI
- README: basic readme file
- rtk-sdr.rules: udev rules to allow external rtl-sdr devices to work with non-root users
- rtl-sdr\_\_export.h: packing the librtlsdr
- rtlsdr-i2c.h: declaring variables from librtlsdr
- sport.c: controls the antenna movement
- srt.cat: the main config file
- srt.hlp: the help file in the gui
- srthelp.c: displays the Help menu in the GUI and compiles the srt.hlp file
- srtn: the program
- srtnmake: used to compile the program
- time.c: important for keeping track of timing
- tuner\_r820t.c: the driver for the tuner
- tuner\_r820t.h: declare variables for the tuner
- velspec.c: plots the spectrum in the gui
- vspectra.c: inputs the data from the dongle
- vspectra\_fftw.c: handles the data from the dongle
- vspectra\_four.c: handles the data from the dongle
- vspectra\_pci.c: also inputs data compatible with PCI boards
- vspectra\_pci\_fftw.c: handles the data from the dongle, compatible with PCI boards

## 5.5 Command Files

Adapted from Bennington<sup>3</sup>

It is suggested to use input command files as this will add speed the entry of SRT commands as well as reduce command entry mistakes. By default the srtn software provides the syntax for setting up a cmd file if you mouse over the Rcmdfl button in the srtn software. The command file is ASCII formatted and can: accept instruction lines (active lines), blank lines (ignored), comment lines (begins with \*, ignored), and comment sections (begins with /, ignored).

Instruction line: Must start with a time mark (either UT or LST) or a colon.

For Example:

```
2005:148:00:00:00 (yyyy:ddd:hh:mm:ss (UT))
23:00:00 06:50:00 ra/dec
LST:06:00:00 (LST:hh:mm:ss)
:120 azel 120 30 (sss azel position)
```

Rules:

```
: cmd /execute the command and proceed to the next line
:120 cmd /execute the command and wait 120 seconds, taking data, before proceeding to the next line in the schedule
:120 /wait 120 seconds, taking data, before proceeding to the next line. This is a convenient way of increasing the
radiometer integration to more than one scan.
```

\*Note: There is NO space allowed between the colon and a time “wait” command

```
LST:06:00:00 /wait until LST 06:00:00, taking data, before proceeding to the next line
2005:148:00:00:00 /wait until UT= yyyy:ddd:hh:mm:ss, before proceeding to the next line
```

Example Set: Command the SRT to take 1420.4 MHz hydrogen spectra in 5 degree spacing along a section of the galactic equator. The user must start data recording, unstow the telescope, calibrate the receiver, set the observing frequency center and frequency scan and then repeat the spectral line observations for ten points along the equator. Note: Allow a space between the colon and the command.

```
: record rotation.rad /(Start data recording of file rotation.rad)
: galactic 206 20 /(unstow and move to calibration position)
: freq 1419 /(Off-hydrogen calibration frequency)
: calibrate
: freq 1420.4 4 /(Set center frequency mode 4)
: galactic 205 0 /(Move to first data point)
: galactic 210 0 /(Next point)
: galactic 215 0
: galactic 220 0
: galactic 225 0
: freq 1419 /(Off-hydrogen calibration frequency)
: galactic 225 20 /(Calibration point)
: calibrate
: freq 1420.4 4 /(Set center frequency mode 4)
: galactic 230 0 /(Move to sixth data point)
: galactic 235 0 /(Next point)
: galactic 240 0
: galactic 245 0
: galactic 250 0
: roff /(End data recording)
```

If this input command file were named galactic.cmd, the user could initiate this observation by clicking in the command text box: galactic.cmd The SRT will read each line in turn and report the current line read as green text in the message board area. If, for example, the start time was 1400 Universal Time on March 15, 2002; and no output file name was entered, the default OUTPUT file would automatically be written and labeled 0214814.rad. Where the file label is: “yydddh.h.rad”

---

<sup>3</sup><https://github.com/BenningtonCS/Telescope-2014/wiki>

## 5.6 Example Procedure

Adapted from Bennington<sup>4</sup>.

### **Calibrate the telescope:**

First thing you will have to do before any observing will be to calibrate the telescope.

For observing Cas A, you're going to need to perform the "advanced" calibration procedure.

Make sure that your center frequency is set to something that is OFF THE H1 LINE, 1415.00 MHz works well. Your center frequency must be off the H1 line because Cas A lies pretty much right in the galactic plane, so any H1 emissions would muddy up the Cas A signal.

Refer to the calibration page for details on how to carry out the advanced calibration.

Once you get your new Tsys and input into the srt.cat file, you're ready to observe!

### **Observing Cas A:**

Once you've closed SRTN and input your new Tsys, fire SRTN back up.

Change your center frequency to 1415.00 MHz or whatever other frequency you are trying out.

Do a regular calibration procedure, aka just the bandpass calibration that comes boxed in SRTN.

Once the telescope is calibrated, slew over to Cas A.

Hit record.

Hit the "beamsw" button on the top bar of buttons in SRTN. This will start the beamswitching procedure, which takes the telescope on and off Cas A by a beamwidth on either side. This technique is used to tease out faint signals.

Wait till about 40-45 beamswitches pass. There is a counter in the average spectrum (bottom spectrum) window for you to keep track of.

Hit "beamsw" again to turn the beamswitching off.

Hit record again to stop recording.

Stow the telescope and close SRTN (unless of course you aren't done looking at stuff!).

This method is of course not the template to use for all sources and the methodologies will depend on current conditions, type of source, strength of source, etc.

It is also suggested to leave enough time to take multiple measurements sets for each source.

---

<sup>4</sup><https://github.com/BenningtonCS/Telescope-2014/wiki>

# Chapter 6

## Data Reduction

### 6.1 Parsing the Data Files

Adapted from Bennington<sup>1</sup>.

Parsing and Plotting the Spectra.

- Copy the data from the server computer to your personal computer to run the reduction sequence.
- The command `scp /file/to/move/ /destination/` will help.
- Make sure you acquire the parsing code<sup>2</sup> in order to parse the data.
- The metadata parser is called `meta_parse.py`. The `meta_parse` script lists columns of data in the order: Date, obsn, az, el, center freq, Tsys, Tant, vlsr, galactic latitude, galactic longitude, source, fstart, fstop, spacing, bw, flw, nfreq, nsam, npoint, integ, sigma, bsw.
- The spectrum parser is called `spectrum_parse.py`. The `spectrum_parse` script lists each frequency in the spectrum in the first column, and then the power values associated with each frequency in each subsequent columns.
- The rotation curve parser is called `rot_curve_spec_parse.py`. The `rot_curve_spec_parse` script usage is the same as the previous two programs. The printing of the data is a little different, however. The first column prints the frequency channels present in the spectrum like the other parser, but then prints the velocities associated with the red/blueshift of the frequencies away from the 1420.406 MHz center frequency in the second column. The next column is the average spectrum for glon 0, then glon 10, then glon 20, and so on until glon 90. It averages all spectra taken at a single point in the galactic plane to produce an average spectrum for that point.
- Another parser is called `hi_spec_parse.py`. It prints the column numbers of the last spectrum taken before the telescope moved to the next point. The velocities associated with frequencies red/blueshifted from the center freq. of 1420.406 MHz are also calculated.
- To run the files, use `~$ python meta_parse.py test_data.rad -o meta_output.txt`. You might have to specify a different python version. These should work with python2.7, it is untested for python3+.
- Once you have your spectra parsed out and written to a new file, you can start to plot them all.
- First, open up the original output file you made. This is sort of tedious but you're going to need to look at the spectra (large blocks of numbers) to see which ones if any pop out at you. Generally if the numbers that make up a certain spectra are larger than the numbers of the other spectra, those are the ones you're looking for.
- Alternatively, look for the spectra that were taken at the Az/El of the source. You'll be able to tell because the spectra will always come back to a certain central Az/El and then go off to one side azimuthally.
- Count how many spectra into the file your spectrum of choice is - this number plus 1 is going to be the column number of your spectrum in the parsed file. Hold on to this number - you will need it when plotting.
- Now that you've picked out the spectra you want to plot (it is a good idea to plot off-source spectra for comparison), you can actually plot the spectra..

---

<sup>1</sup><https://github.com/BenningtonCS/Telescope-2014/wiki>

<sup>2</sup><https://github.com/OUsrc/srtn.git>

## 6.2 Gnuplot

Adapted from Bennington<sup>3</sup>.

Understanding, Interpreting, and changing the units of your plots into something useful.

- So the y-axis in your plot will have units of Kelvins. The numbers in the spectrum correspond to the power level measured at a given frequency, and these numbers are given in Kelvins.
- We want to convert Kelvins to Janskys with  $kT_a = \frac{S_\nu \times A_{dish}}{2}$ ,  $T_a$  is Antenna temperature.  $S_{\nu}$  is spectra flux at a certain frequency.  $A_{dish}$ ,  $k$  is Boltzmann's constant. The value you get at this point will be in J/s/m<sup>2</sup>/Hz, and 1 Jansky =  $1 \times 10^{26}$ , so you'll need to multiply the value you just got by  $1 \times 10^{26}$ , and then you'll have your spectral flux density in Janskys. Now you can compare to established continuous spectra.

## 6.3 Formulas

$$\begin{aligned} \text{Diameter} &= 3 \text{ m} \\ \text{Frequency} &= 1.420401 \text{ GHz} \\ \text{Speed of Light} &= 0.299792458 \text{ giga-m/s} \\ \lambda &= \frac{c}{\nu} \\ \text{Beam Width} &= \frac{70 \times \lambda}{\text{Diameter}} \text{ degrees} \end{aligned} \tag{6.1}$$

## 6.4 GitHub

### 6.4.1 Quick Command Breakdown

This is meant for just a quick overview of GitHub commands. Please understand the commands before using them.

```
1 new repository
2 ->git init :: initialize local repository inside the repository
3 ->git status :: list git files in current directory, tracked and untracked
4 ->git log :: shows a log of all new commits
5
6 new file
7 ->git add $1 :: $1 is name of file, adds the file to the staging environment. Think like indexing
8 ->git commit -m $1 :: $1 is the comment for the commit. Should be related to the commit.
9
10 new branches
11 ->git checkout -b $1 :: $1 is new branch name. Branching allows working on other projects without
    affecting master
12 ->git branch :: lists known branches and * tells which branch you are in
13 ->git checkout branch :: moves you to the branch defined
14
15 online
16 ->git remote add(rm) origin $1 :: $1 is website name. add will add $1 to the origin variable while rm
    will remove the variable
17 ->git push -u origin branch :: pushes the local branch repo to the origin branch repo
18 ->git revert $1 :: $1 is the hash code number. Command is used to revert the branch
19 ->git pull origin branch :: pulls the changes in the branch to your local. Prefer to use fetch and
    merge
20 ->git fetch origin :: remotely stages origin to merge locally
21 ->git merge origin/branch :: remotely merges the branch from origin to your current branch
22 ->git request-pull v1.0 origin master :: initiate a pull request upstream from your master branch to
    their master branch
```

### 6.4.2 Proper Usage

To properly use GitHub in collaboration with other colleagues please follow this guide.

To implement changes to the repository, you will have to make an account and request access online. Once this is finished, you will not have to check the website again. If you do not already have a directory you wish to use to contribute, make a directory now. You can use a pre-existing directory. Once in this directory, run the command `git init` to initialize the current directory.

<sup>3</sup><https://github.com/BenningtonCS/Telescope-2014/wiki>



To pull changes from the website, you first have to set the variable `origin` using the command `git remote add origin $1` and input the name of the repository<sup>4</sup>. Then use `git fetch origin` to remotely pull the most updated, official code from the GitHub page. You then merge the most up to date version into your local master with `git merge origin/master`.

You can then create local branches to work and merge projects using `git checkout -b $1` to create the branch and `git checkout $1` to move to a branch. The command `git branch` will tell you the list of current branches within the initialize directory you are in. Make the changes you wish to make and then issue the index command `git add $1`. This prepares the file for the commit stage. Once you are satisfied with all of the indexed files, commit the changes using `git commit -m "$1"` where `$1` in this case is a short comment on what this change is.

Once committed push the changes to the online repository with `git push -u origin branch`. If an error comes up regarding unable to push, probably one of two things happened: incorrect spelling of command/branch or someone else has since updated the branch. Like I said, you will be unable to push changes to the GitHub master branch so push changes to another branch and initiate a pull request.

---

<sup>4</sup>In this case, the repository is <https://github.com/00srt/srtn.git>

# Chapter 7

## Complete Parts List

### 7.1 Rotor

Power supply unit: PW15015LC

Specifications:

Voltage adj. range: 13.5 - 16.5 Volt / 10 Amp DC

Overload protection

Rimpel noise max 180 mVpp

AC input range: 88 ~ 132VAC/176 ~ 264VAC selected switch

Dimension: 19 x 11 x 5 cm (LxBxH)

Weight: 0.8 kg

Power supply unit: PW32015

Specifications:

Universal AC input / Full range

Built-in active PFC function. PF>0.95

Voltage adj. range: 13.5 - 18 Volt / 20 Amp DC

Protections: Short circuit/Over load/Over voltage/Over temperature

Forced air cooling by built-in DC Fan

Built-in fan speed control

Voltage range: 88 ~ 264VAC 124 ~ 370VDC

Dimensions: 23 x 11.5 x 5.5 cm (LxBxH)

Weight: 1.1 Kg

Core control cable

SPID RAS rotator bracket: FPD-BR02

### 7.2 Mount & Dish

- Mount Plate/bracket
- 3" x 10' Pier
- Ballast: 1000lbs
- 6mm galvanized aluminum wire mesh 4x150ft
- 2 pcs CNC milled centers (Hub)
- 12 Rib's made of 20mm sq aluminium tube
- 4 inner rings: 2mm x 100ft are used to secure mesh
- Outer rim made of 3x20mm aluminum strip
- Mesh secured by using of 2x20mm aluminum strip
- 3-Leg feed support 15 or 20 square aluminum tube
- 3" to 2.875" adapter for rotor

## 7.3 Feed & LNA

- Ultra Low Noise Amplifier
- Coaxial Bias-Tee
- Coaxial Cable 086-6SM+
- Coaxial Cable 086-4SM+
- SMA-F to SMA-F Adapter
- SMA-F to Solder Pin Bulkhead Connector
- SMA-M to SMA-M Right Angle Adapter
- 3M Adhesive Copper Foil Tape
- 22 Gauge Copper Wire
- Semi-Rigid Coaxial Cable
- F-Type Male to BNC Female
- SMA-M to SMA-M Coupler
- DC Barrel Jack connector
- F-type plug connector
- Cap Screw 1/4-20x3.25"
- Lock Washers 6-32
- Machine Screws, 6-32x1/2"
- Machine Screws, 6-32X1"
- Nuts 6-32
- Nuts 1/4-20
- Washers 1/4"
- Styrofoam Rod 2.5" x 24"
- Cake Pan 7"x3"
- O-rings 7/32" inner diameter
- O-rings 1/4" inner diameter
- SMA-M Crimp Connector Straight
- LMR-400
- Type-F Male Crimp Connector
- In-Line Amplifier
- BNC Female Jack to Type-F Male Plug
- BNC-M to BNC-M Patch Cable
- Power Injector
- BNC-F to SMA-M Adapter
- Bandpass Filter
- SMA-F to BNC-F Adapter
- BNC-F to SMB Plug Adapter
- AC-to-DC Power Supply
- Breadboard
- DC Jack
- F-Type Plug