

SSRP/LTC1746 Manual V0.1

Check the most recent version:

<http://oscar.dcarr.org/ssrp/hardware/LTC1746/LTC1746.php>

I. Introduction

The Simple Software Radio Peripheral (SSRP)¹ is intended to provide a means to convert analog signals into digital signals and vice-versa in a software radio system. The SSRP itself is comprised of the USB 2.0 interface module manufactured by Elrasoft² and a daughterboard containing an ADC or DAC. The LTC1746 daughterboard is based around the LTC1746 14bit analog to digital converter manufactured by Linear Technology³. It provides a means to sample a single channel analog input at rates up to about 20MSPS. The digital representation of the input signal is transferred to the host computer via the USB interface module and the host machine USB bus. Once the data is received by the host it may be manipulated in a variety of ways. In particular the SSRP is designed to integrate tightly with the GnuRadio⁴ software radio project. The support software contains several modules intended to easily transfer data in and out of the GnuRadio environment using the SSRP. It should be noted that the SSRP is not limited to use in combination with GnuRadio. It is relatively simple to gain access to the raw SSRP data stream for use in other applications. However, in large part the scope of this manual is limited to applications involving GnuRadio as it is difficult to foresee the exact requirements of other custom applications.

II. Getting Started

A. Requirements

i. Hardware

- a) Elrasoft USB 2.0 Module⁵
- b) LTC1746 Daughterboard⁶
- c) DIP-8 Crystal Oscillator (1-20MHz)⁷
- d) Host computer with USB 2.0 (480Mbps) port (a full speed 12Mbps port is not sufficient)
- e) (optional) SMA-M to BNC-F adapter

ii. Software

1 Simple Software Radio Peripheral: <http://oscar.dcarr.org/ssrp/>

2 Elrasoft: <http://www.elrasoft.com>

3 Linear Technology: <http://www.linear-tech.com/>

4 GnuRadio: <http://www.gnu.org/software/gnuradio/>

5 USB 2.0 Module: <http://oscar.dcarr.org/ssrp/hardware/usb/usb.php>

6 LTC1746 Daughterboard: <http://oscar.dcarr.org/ssrp/hardware/LTC1746/LTC1746.php>

7 Oscillator speed is determined by available USB bandwidth. 1MSPS requires 2MB/s bandwidth.

- a) 2.4 or later linux kernel (with high-speed USB support enabled)
- b) libusb
- c) Standard*nix build tools (gcc, automake, etc)

B. Installing fx2_programmer

- i. Download the latest version of fx2_programmer from http://volodya-project.sourceforge.net/fx2_programmer.php
- ii. Untar the downloaded file: “tar -xzvf fx2_programmer[version].tar.gz”
- iii. Change into the new directory: “cd fx2_programmer”
- iv. Build the package: “make”
- v. At this point you'll see a message like “make: *** [test1] Error 255”. Just ignore it. The package does not have a properly set-up Makefile.
- vi. Copy the new program into a 'good spot': “cp fx2_programmer /usr/bin/”
[You might need to be root to do this.]
- vii. Type “fx2_programmer” You should see something like:

fx2_programmer bus device function [parameters]

<i>Function</i>	<i>Parameters</i>	<i>Description</i>
<i>dump_busses</i>		<i>show all available devices</i>
<i>dump</i>	<i>start len</i>	<i>dump RAM contents</i>
<i>bulk_dump</i>	<i>endpoint len chunk</i>	<i>dump data read of bulk endpoint</i>
<i>bulk_bench</i>	<i>endpoint len chunk</i>	<i>benchmark throughput of bulk endpoint</i>
<i>upload</i>	<i>file start len</i>	<i>upload binary file to RAM</i>
<i>set</i>	<i>address byte</i>	<i>changes values of a single byte</i>
<i>program</i>	<i>file.ihx</i>	<i>programs fx2 using Intel hex format file</i>

C. Installing the SSRP firmware, utilities and programming the USB interface board

- i. Download the latest version of the interface board firmware from:
<http://oscar.dcarr.org/ssrp/software/firmware/firmware.php>
- ii. Unpack the file as before (with “tar -xzvf ...”) and change into the new “ssrp-[version]” directory.
- iii. Configure the package with: “./configure --prefix=/usr”
- iv. Build the package with “make”
- v. Install the package with “make install” [Need to be root.]
- vi. Change directory: “cd /usr/share/ssrp/firmware”
- vii. Connect your USB Interface board to the host machine without any daughterboards installed.
- viii. Type “program_and_start.sh bandwidth.ihx”
- ix. Type “checkHeartbeat.sh” and verify the board is operational.

- x. If either of the two previous steps fail or produce obvious errors then you have a problem with your USB setup. Verify the the USB cable is indeed attached at both ends and that you are plugged into a high speed USB 2.0 port. You might try the command “dmesg | grep usb”. It will output any relevant kernel USB messages. Resolving further issues is beyond the scope of this document.
- xi. Assuming that all went well and “checkHeartbeat.sh” produced a stream of numbers as expected, you've successfully programmed the USB interface module. You'll need to program the module every time it is attached to the computer because it has no non-volatile program storage. Just repeat steps “viii” and “ix” in the future and you'll be set. Note: You do have to be in the “/usr/share/ssrp/firmware” directory for this to work.

D. Configuring the LTC1746 daughterboard

- i. In order to support the widest range of sampling rates, the LTC1746 board has two modes: low rate and high rate. The low rate mode can be used for sampling rates from 1-5Msps. The high rate mode supports sampling rates from 5Msps to 25Msps. The mode is selected by three zero ohm resistors on the LTC1746 board, R13, R14 and R15. To select the high speed mode, R13 and R15 should be installed and R14 should be unpopulated. To select low speed mode, R13 and R15 should be removed and R14 installed. Unless otherwise requested, all boards are shipped in the high speed mode.
- ii. The only other configuration option on the LTC1746 is the actual sampling rate selection. A 5V CMOS crystal oscillator in DIP-8 packaging should be used. The frequency of this oscillator directly determines the sampling rate of the board. For example, a 6MHz oscillator will cause the board to sample 6 million times per second (Msps). Unfortunately, choosing the sampling rate is not as easy as selecting a crystal oscillator in the correct range. Each sample causes two bytes to be transferred over the USB bus. Therefore sampling at 6Msps requires that the USB bus transfer 12MB/s. In theory, the USB 2.0 bus can transfer 60MB/s, however most computers can only achieve a fraction of this rate. Fortunately, the SSRP software includes a tool to help you estimate the maximum sampling frequency supported by your system. Note that the bandwidth measuring software will only work if your LTC1746 is in high speed mode. If you're using the low speed mode, you can skip the next step because you're not likely to encounter any bandwidth problems.
- iii. To measure your system's USB bandwidth do the following:
 - a) Insure that the interface board is disconnected from the USB cable. Insert the multi-pin connector on the bottom of the daughterboard into the socket on the top of the interface module. The daughterboard should be installed so that it is vertically above the interface module board. Do not install the

- crystal oscillator or connect power to the LTC1746 board at this time.
- b) Re-connect the USB cable to the host machine and the SSRP. One green LED should light to indicate that the digital section of the board is powered.
 - c) Connect the power leads on the LTC1746 board to a 6-8VDC source.
Before applying power, insure that the polarity is correct. *Incorrect polarity will cause extensive damage.* When power is applied, the second green LED should light to indicate that the analog section of the board is now powered.
 - d) Change directory to the firmware directory: “cd /usr/share/ssrp/firmware”
 - e) Load the bandwidth measuring firmware:
“program_and_start.sh bandwidth.ihx”
 - f) Measure the USB bandwidth: “test_bandwidth” (run as root) You might want to repeat this command a few times to get a good average number.
 - g) The recommended crystal oscillator speed from test_bandwidth is a recommended *maximum*. It is very important to note that you do not have to run the sampling clock this fast if you don't need to. In fact, the faster the sampling clock, the more data your CPU has to process. For this reason it is advisable to choose a sampling rate that gives sufficient bandwidth for your application but does not produce an excessive amount of raw data.
- iv. Crystal oscillators can be obtained from a variety of sources. Digi-Key⁸ and Mouser Electronics⁹ are popular mail-order electronics suppliers. The correct item is usually referred to as a Half-size (or DIP-8) CMOS crystal oscillator. Make sure you get a 5 volt oscillator rather than a 3.3 volt version. Example part numbers from Digi-Key are CTX150-ND (1MHz), CTX157-ND (5MHz), CTX163-ND (10MHz) and CTX166-ND (16MHz).
- v. After the correct mode has been selected and a suitable crystal oscillator selected, the crystal oscillator must be inserted into its socket on the LTC1746. The oscillator socket is the eight pin black square connector located in the top left corner of the board. Most oscillators will have one corner that is more pointed than the others or will have a dot or other marking in one corner. You will notice that the silkscreen around the oscillator socket also has one corner that is more pointed than the others, insert the oscillator into its socket so that the markings on the board and oscillator are aligned.

E. Installing the GnuRadio modules

- i. Download the current version of the GnuRadio 2.x packages from <http://comsec.com/wiki?GnuRadio2.X> You'll need:
 - a) gnuradio-core
 - b) gnuradio-examples

⁸ Digi-Key: <http://www.digi-key.com>

⁹ Mouser: <http://www.mouser.com>

- c) gr-audio-oss
- d) gr-wxgui

ii. Note that the following packages are required for GnuRadio. If you have to install any of these, take a look at the README in gnuradio-core. Some of the packages must be configured with certain parameters.

- a) pkgconfig 0.15.0 or later¹⁰
- b) FFTW 3.0 or later¹¹
- c) Python 2.3 or later¹²
- d) Boost C++ Libraries¹³
- e) cppunit 1.9.14 or later¹⁴
- f) swig 1.3.22 or later¹⁵

iii. If you'd also like GUI support in GnuRadio, you'll also need:

- a) wxPython 2.5.2.7 or later¹⁶

iv. Change directory to the location where you downloaded gnuradio-core.

v. Untar gnuradio-core: “tar -xzvf gnuradio-core-[version].tar.gz”

vi. Change directory into the new gnuradio-core directory:

“cd gnuradio-core-[version]”

vii. Configure the package: “./configure --prefix=/usr”

viii. Build the package: “make”

ix. Check the build: “make check”

x. Install: “make install”

xi. Change directory to the location where you downloaded gnuradio-examples.

xii. Untar gnuradio-examples: “tar -xzvf gnuradio-examples-[version].tar.gz”

xiii. Change directory into the new gnuradio-examples directory:

“cd gnuradio-examples-[version]”

xiv. Configure the package: “./configure --prefix=/usr”

xv. Build the package: “make”

xvi. Install: “make install”

xvii. Change directory to the location where you downloaded gr-audio-oss.

xviii. Untar gr-audio-oss: “tar -xzvf gr-audio-oss-[version].tar.gz”

xix. Change directory into the new gr-audio-oss directory:

“cd gr-audio-oss-[version]”

10 pkgconfig: <http://www.freedesktop.org/Software/pkgconfig>

11 FFTW: <http://www.fftw.org>

12 Python: <http://www.python.org>

13 Boost C++ Libraries: <http://www.boost.org>

14 cppunit: <http://cppunit.sourceforge.net>

15 swig: <http://www.swig.org>

16 wxPython: <http://www.wxpython.org>

- xx.Configure the package: `“./configure --prefix=/usr”`
- xxi.Build the package: `“make”`
- xxii.Install: `“make install”`

- xxiii.Change directory to the location where you downloaded gr-wxgui.
- xxiv.Untar gr-wxgui: `“tar -xzvf gr-wxgui-[version].tar.gz”`
- xxv.Configure the package: `“./configure --prefix=/usr”`
- xxvi.Build the package: `“make”`
- xxvii.Install: `“make install”`
- xxviii.Download gr-ssrp from
`http://oscar.dcarr.org/ssrp/software/gnuradio/gnuradio.php`
- xxix.Change directory to the location where you downloaded gr-ssrp
- xxx.Untar gr-ssrp: `“tar -xzvf gr-ssrp-[version].tar.gz”`
- xxxi.Configure the package: `“./configure --prefix=/usr”`
- xxxii.Build the package: `“make”`
- xxxiii.Install: `“make install”`

F. Testing the entire system

- i. Plug the assembled SSRP (interface board and LTC1746 board) into the USB bus. One green LED should light.
- ii. Connect the power leads on the LTC1746 board to a 6-8VDC source. Before applying power, insure that the polarity is correct. The second green LED should light.
- iii.Switch to the root user.
- iv.Program the interface board with the appropriate firmware. If you are using low speed mode, `“program_and_start.sh ioA_512.ihx”` or using high speed mode `“program_and_start.sh ioS_512.ihx”` Note that you must be in `/usr/share/ssrp/firmware` for `program_and_start.sh` to find the firmware files.
- v. Verify the programming was successful with the `“checkHeartbeat.sh”` command. Type `“Ctrl-c”` to terminate the command. If no numbers appear, insure that the USB cables are connected correctly and re-program.
- vi.Verify that the SSRP is sending data with the `“monitorEP6_512.sh”` command. A lot of data should appear, timeout errors should not. Type `“Ctrl-c”` to terminate this command. If you receive timeout errors, verify that the crystal oscillator is installed correctly. Then remove the USB cable, reconnect the cable and reprogram the interface board.
- vii.Change directory into the ssrp examples directory: `“cd /usr/share/ssrp/examples”`
- viii.Set the python path:
`“export PYTHONPATH=/usr/lib/python2.3/site-packages”` You have to set

this to help python find the GnuRadio packages. You'll need to set it every time you open a new shell and want to work with GnuRadio. For convenience you might want to add it as a line in your `~/.bash_profile` file. If you do so, the variable will be set automatically for you in the future.

- ix. Run the `ssrp_fft` example: `./ssrp_fft.py`. A window should appear showing a real time FFT of the data from the SSRP. Feel free to attach a signal source (Function generation, radio, etc) to the SSRP at this point. Note that the SSRP input is DC-coupled and the input signal should not exceed 1.6Volt speak-peak.
- x. If you see data, the SSRP is working.