Using libxbee, an XBee Open Source library (A test run tutorial).

Test Tutorial Author: Adolph Seema
Date Last Edited: 2013/11/30

Source Code used: latest commit 1208e5ccae attie

1. Introduction

Apart from the official Digi Open source XBee library here:
https://github.com/tomlogic/xbee_ansic_library.

There is another even more useful and well polished library hosted on Google Code (here). It also has a Google Group (here). We are grateful to **Attie Grande** for the great library and continuous support.

I got the source code using git by following the instructions here:
http://attie.co.uk/libxbee/getting_started/get_it#s1
Before you do anything, read the Getting Started Guide on the website thoroughly and browse the documentation/man pages.

The exercises I performed below were compiled and executed on Ubuntu 13.10 Core i7 Quad Core Dell Inspiron for what it is worth. Normal local gcc toolchain for the Ubuntu host OS used. The git source code used, was last committed on: 2013/11/09 as:

2. Key

Green ⇒ Good output,
Red ⇒ Bad output,
Blue ⇒ Actual command ran,
Black ⇒ Comment.
Purple ⇒ Warning.

3. Table of Contents

4. Getting and compiling the source code.

I. Getting source code from GitHub with git.

```
$ cd $HOME/mygits
$ mkdir my_libxbee
$ cd my_libxbee/
$ git clone https://github.com/attie/libxbee3.git

+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
Cloning into 'libxbee3'...
remote: Counting objects: 5514, done.
remote: Compressing objects: 100% (2255/2255), done.
remote: Total 5514 (delta 3544), reused 5138 (delta 3171)
Receiving objects: 100% (5514/5514), 14.40 MiB | 1.70 MiB/s, done.
Resolving deltas: 100% (3544/3544), done.
Checking connectivity... done
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

$ ls -l
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
total 4
drwxr-xr-x 13 aseema aseema 4096 Nov  9 15:33 libxbee3
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

$ cd libxbee3/
$ ls

+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ ls
apps         frame.c    ll.h     mode.c         net_callbacks.h  pkt.c      sample    xbee.h
xsys_darwin.h
conn.c        frame.h    log.c    mode.h         net.h            pkt.h      thread.c  xbee_int.h
xsys.h
conn.h        HISTORY    log.h    modes          net_handlers.c   prepare.c  thread.h  xbeep.cpp
xsys_linux.c
COPYING       html       make     mutex.c        net_handlers.h   prepare.h  tx.c      xbeep.h
xsys_linux.h
COPYING.header  interface  makefile  mutex.h       net_io.c         README     tx.h      xsys.c
COPYING.LESSER  internal.h  man      net.c          net_io.h       rx.c      ver.c    xsys_darwin
error.c         ll.c        manuals  net_callbacks.c  package      rx.h      xbee.c   xsys_darwin.c
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ emacs README
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
… snip …

'xbee1'  => XBee Series 1   (802.15.4 / no meshing)
    http://goo.gl/10a45
    http://sprkfn.com/p8665
    http://doc.libxbee.attie.co.uk/images/xbee1.jpg
'xbee2'  => XBee Series 2.5 (ZNet / DigiMesh) - formerly 'Series 2'
    http://goo.gl/fQbkt
    http://sprkfn.com/p8876
    http://doc.libxbee.attie.co.uk/images/xbee2.jpg
'xbeeZB' => XBee Series 2   (ZigBee)
    http://goo.gl/TJmeQ
    http://sprkfn.com/p10414
    http://doc.libxbee.attie.co.uk/images/xbeeZB.jpg
'xbee3'  => XBee Series 3   (900 MHz)
    http://goo.gl/Wdghyq
'xbee5'  => XBee Series 5   (868 MHz)
    http://goo.gl/66FZI
'xbee6b' => XBee Series 6B  (WiFi)
    http://goo.gl/gV9XwN

… snip ...
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

It is just 95 lines long, please read it.

The README shows the modules above are supported. See config.mk generated below for confirmation.
i.e. see the “commented line #12

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
… snip …
12#MODELIST:=      xbee1 xbee2 xbee3 xbee5 xbee6b xbeeZB net debug
… snip ...
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ make configure
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
… snip …
:~/mygits/my_libxbee/libxbee3$ ls -l c*
-rw-r--r-- 1 aseema aseema  1737 Nov  9 16:01 config.mk
-rw-r--r-- 1 aseema aseema 31277 Nov  9 15:33 conn.c
-rw-r--r-- 1 aseema aseema  2619 Nov  9 15:33 conn.h
… snip ...
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

II. Modifying config.mk

I opened up the config.mk above and uncommented XBEE_LOG_RX and XBEE_LOG_TX in order to get debug info.

See lines 28 and 29. Note that I left HARDWARE flow control enabled.

```
$ emacs config.mk
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
 1### Libxbee configuration options:
 2
 3### system install directories
 4SYS_ROOT?=
 5SYS_LIBDIR:=    $(SYS_ROOT)/usr/lib
 6SYS_INCDIR:=    $(SYS_ROOT)/usr/include
 7SYS_MANDIR:=    $(SYS_ROOT)/usr/share/man
 8SYS_GROUP:=     root
 9SYS_USER:=      root
10
11### using this can create a smaller binary, by removing modes you won't use
12#MODELIST:=      xbee1 xbee2 xbee3 xbee5 xbee6b xbeeZB net debug
13
14### to use the 'install_html' rule, you must specify where to install the files to
15#SYS_HTMLDIR:=   /var/www/libxbee.doc
16
17### setup a cross-compile toolchain (either here, or in the environment)
18#CROSS_COMPILE?=
19#CFLAGS+=
20#CLINKS+=
21
22### un-comment to remove ALL logging (smaller & faster binary)
23#OPTIONS+=       XBEE_DISABLE_LOGGING
```

```

### use for more precise logging options
#OPTIONS+=	XBEE_LOG_NO_COLOR
#OPTIONS+=	XBEE_LOG_LEVEL=100
OPTIONS+=	XBEE_LOG_RX
OPTIONS+=	XBEE_LOG_TX

### un-comment to disable strict objects (xbee/con/pkt pointers are usually checked inside functions)
### this may give increased execution speed, but will be more suseptible to incorrect parameters
#OPTIONS+=	XBEE_DISABLE_STRICT_OBJECTS

### un-comment to remove network server functionality
#OPTIONS+=	XBEE_NO_NET_SERVER
#OPTIONS+=	XBEE_NO_NET_STRICT_VERSIONS

### un-comment to turn off hardware flow control
#OPTIONS+=	XBEE_NO_RTSCTS

… snip ...
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

III. Save config.mk above and build the library.

```
$ make
```

```
+++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
mkdir -p lib
mkdir -p .build
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD -fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD conn.c -c -o .build/conn.o
conn.c: In function ‘xbee_conAddressCmpDefault’:
conn.c:241:2: warning: #warning TODO - handle broadcast endpoint, but probably not here... [-Wcpp]
 #warning TODO - handle broadcast endpoint, but probably not here...
  ^
conn.c: In function ‘xbee_conCallbackProd’:
conn.c:926:2: warning: #warning TODO - there is a gap here, needs a mutex [-Wcpp]
 #warning TODO - there is a gap here, needs a mutex
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD -fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD error.c -c -o .build/error.o
```

```
… snip …

modes/xbee2/mode.c: In function ‘xbee_s2_transmitStatus_rx_func’:
modes/xbee2/mode.c:107:2: warning: #warning TODO - currently missing out on the resolved
network address, retry count, and discovery status [-Wcpp]
 #warning TODO - currently missing out on the resolved network address, retry count, and
discovery status
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee2/sensor.c -c -o
.build/modes_xbee2_sensor.o
ld -r -o .build/__mode_xbee2.o .build/modes_xbee2_at.o .build/modes_xbee2_data.o
.build/modes_xbee2_dataExp.o .build/modes_xbee2_identify.o .build/modes_xbee2_io.o
.build/modes_xbee2_mode.o .build/modes_xbee2_sensor.o
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee3/at.c -c -o
.build/modes_xbee3_at.o
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee3/data.c -c -o
.build/modes_xbee3_data.o
modes/xbee3/data.c: In function ‘xbee_s3_data_tx_func’:
modes/xbee3/data.c:101:2: warning: #warning TODO - currently missing support for NAK and
Trace Route messages [-Wcpp]
 #warning TODO - currently missing support for NAK and Trace Route messages
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee3/dataExp.c -c
-o .build/modes_xbee3_dataExp.o
modes/xbee3/dataExp.c: In function ‘xbee_s3_dataExp_tx_func’:
modes/xbee3/dataExp.c:129:2: warning: #warning TODO - currently missing support for NAK
and Trace Route messages [-Wcpp]
 #warning TODO - currently missing support for NAK and Trace Route messages
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee3/identify.c -c -o
.build/modes_xbee3_identify.o

… snip …

gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee6b/mode.c -c -o
.build/modes_xbee6b_mode.o
```

```
modes/xbee6b/mode.c: In function ‘xbee_s6b_frameError_rx_func’:
modes/xbee6b/mode.c:186:2: warning: #warning TODO - figure out how to use this feedback...
[-Wcpp]
 #warning TODO - figure out how to use this feedback...
  ^
ld -r -o .build/__mode_xbee6b.o .build/modes_xbee6b_at.o .build/modes_xbee6b_data.o
.build/modes_xbee6b_io.o .build/modes_xbee6b_mode.o

… snip …

gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbeeZB/mode.c -c -o
.build/modes_xbeeZB_mode.o
modes/xbeeZB/mode.c: In function ‘xbee_sZB_transmitStatus_rx_func’:
modes/xbeeZB/mode.c:108:2: warning: #warning TODO - currently missing out on the resolved
network address, retry count, and discovery status [-Wcpp]
 #warning TODO - currently missing out on the resolved network address, retry count, and
discovery status
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbeeZB/ota.c -c -o
.build/modes_xbeeZB_ota.o
modes/xbeeZB/ota.c: In function ‘xbee_sZB_ota_rx_func’:
modes/xbeeZB/ota.c:43:2: warning: #warning TODO - check that these are the correct
addresses to be using [-Wcpp]
 #warning TODO - check that these are the correct addresses to be using
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbeeZB/sensor.c -c
-o .build/modes_xbeeZB_sensor.o

… snip …

ln -fs `basename lib/libxbeep.so.3.0.10` lib/libxbeep.so
objcopy --only-keep-debug lib/libxbeep.so.3.0.10 lib/libxbeep.so.3.0.10.dbg
objcopy --add-gnu-debuglink=lib/libxbeep.so.3.0.10.dbg lib/libxbeep.so.3.0.10
objcopy --strip-debug lib/libxbeep.so.3.0.10
touch lib/libxbeep.so.3.0.10.dbg
ar rcs lib/libxbeep.a.3.0.10 lib/libxbeep.o
ln -fs `basename lib/libxbeep.a.3.0.10` lib/libxbeep.a
:~/mygits/my_libxbee/libxbee3$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

IV. Install the library in your Ubuntu wide accessible binary directories:

```
$ sudo make install
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ sudo make install
[sudo] password for aseema:
install -g root -o root -DT -m 755 lib/libxbee.so.3.0.10 /usr/lib/libxbee.so.3.0.10
install -g root -o root -DT -m 755 lib/libxbee.so.3.0.10.dbg /usr/lib/libxbee.so.3.0.10.dbg
ln -fs libxbee.so.3.0.10 /usr/lib/libxbee.so
install -g root -o root -DT -m 755 lib/libxbee.a.3.0.10 /usr/lib/libxbee.a.3.0.10
ln -fs libxbee.a.3.0.10 /usr/lib/libxbee.a
install -g root -o root -DT -m 755 lib/libxbeep.so.3.0.10 /usr/lib/libxbeep.so.3.0.10
install -g root -o root -DT -m 755 lib/libxbeep.so.3.0.10.dbg /usr/lib/libxbeep.so.3.0.10.dbg
ln -fs libxbeep.so.3.0.10 /usr/lib/libxbeep.so
install -g root -o root -DT -m 755 lib/libxbeep.a.3.0.10 /usr/lib/libxbeep.a.3.0.10
ln -fs libxbeep.a.3.0.10 /usr/lib/libxbeep.a
install -g root -o root -DT -m 644 man/man3/libxbee.3 /usr/share/man/man3/libxbee.3.gz
chown -h root:root /usr/share/man/man3/libxbee.3.gz

… snip ...

install -g root -o root -DT -m 644 man/man3/xbee_vsetup.3
/usr/share/man/man3/xbee_vsetup.3.gz
chown -h root:root /usr/share/man/man3/xbee_vsetup.3.gz
install -g root -o root -DT -m 644 xbee.h /usr/include/xbee.h
install -g root -o root -DT -m 644 xbeep.h /usr/include/xbeep.h
:~/mygits/my_libxbee/libxbee3$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

V. Note where the library header include files are installed.

```
$ ls
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ ls
apps        COPYING.LESSER  internal.h  makefile  mutex.h       net_io.c   README    tx.h
xsys.c
config.mk     error.c       lib       man      net.c         net_io.h   rx.c     ver.c     xsys_darwin
config.mk~     frame.c       ll.c      manuals   net_callbacks.c  package    rx.h     xbee.c
xsys_darwin.c
conn.c        frame.h       ll.h      mode.c    net_callbacks.h  pkt.c     sample    xbee.h
```

```
xsys_darwin.h
conn.h         HISTORY        log.c      mode.h    net.h          pkt.h      thread.c  xbee_int.h
xsys.h
COPYING        html           log.h      modes     net_handlers.c   prepare.c  thread.h  xbeep.cpp
xsys_linux.c
COPYING.header  interface      make       mutex.c   net_handlers.h   prepare.h  tx.c      xbeep.h
xsys_linux.h
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

$ ls lib/
++++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ ls lib/
libxbee.a        libxbee.o   libxbeep.a.3.0.10  libxbeep.so       libxbeep.so.3.0.10.dbg
libxbee.so.3.0.10
libxbee.a.3.0.10  libxbeep.a  libxbeep.o        libxbeep.so.3.0.10  libxbee.so
libxbee.so.3.0.10.dbg
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

$ ls apps/
++++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ ls apps/
config  connection_test  makefile  net_term  pseudo_term  README  stress_test
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

$ ls sample/
++++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ ls sample/
buildall  debug  generic  makefile  makefile.cpp  net  xbee1  xbee2  xbee5  xbee6b  xbeeZB
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

VI. Now it is a good time to test the XBee modules I have.

I am lucky to have XBee S1, XBee S2 (XB24-ZB) and XBee 6.
See: “Which modules should I use?

5. Testing XBee1 Examples

Below are the available XBee1 examples that come with the library.

```
$ ls -l sample/xbee1/
++++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1$ ls -l
total 68
```

```
drwxr-xr-x 2 aseema aseema 4096 Nov 13 22:42 0.simple_at
drwxr-xr-x 2 aseema aseema 4096 Nov 13 21:14 10.c++
drwxr-xr-x 2 aseema aseema 4096 Nov 11 21:40 11.uart_test
drwxr-xr-x 2 aseema aseema 4096 Nov 11 21:53 12.timestamp
drwxr-xr-x 2 aseema aseema 4096 Nov 14 12:15 13.lxb2lxb_client
drwxr-xr-x 2 aseema aseema 4096 Nov 14 12:16 13.lxb2lxb_server
drwxr-xr-x 2 aseema aseema 4096 Nov 11 21:40 14.request_io
drwxr-xr-x 2 aseema aseema 4096 Nov  9 15:33 15.qt
drwxr-xr-x 2 aseema aseema 4096 Nov 13 22:35 1.simple_at
drwxr-xr-x 2 aseema aseema 4096 Nov 14 10:11 2.remote_at
drwxr-xr-x 2 aseema aseema 4096 Nov 13 23:27 3.simple_data
drwxr-xr-x 2 aseema aseema 4096 Nov 11 21:53 4.simple_io
drwxr-xr-x 2 aseema aseema 4096 Nov 11 21:40 5.get_conTypes
drwxr-xr-x 2 aseema aseema 4096 Nov 11 21:53 6.log_config
drwxr-xr-x 2 aseema aseema 4096 Nov 14 11:53 7.broadcast
drwxr-xr-x 2 aseema aseema 4096 Nov 14 12:02 8.catch-all
drwxr-xr-x 2 aseema aseema 4096 Nov 13 21:24 9.node_detect
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

I. We will start with the first example "0.simple_at":

Before you start make sure you understand, XBee addressing and API basics:

How to setup the HW and to test it, this tutorial is great. Skip Chapter 4 and read only the first 4 pages of Cpater 5 if you are not using parallax MCU. The other Chapters apart from the sample programs they show, are relevant to all XBee modules and network notions.

XBee addressing:
http://www.digi.com/support/kbase/kbaseresultdetl?id=2187

XBee API mode:
http://www.digi.com/support/kbase/kbaseresultdetl?id=2184

For details, check the XBee protocol and meaning everything from official Digi here.
See page 61 for TX (Transmit) Request: 64-bit address. OR begin at page 57 for API Operation in depth.

```
$ cd sample/xbee1/0.simple_at/
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/0.simple_at$ ls -l
total 16
-rwxr-xr-x 1 aseema aseema 12030 Nov 13 22:16 main
```

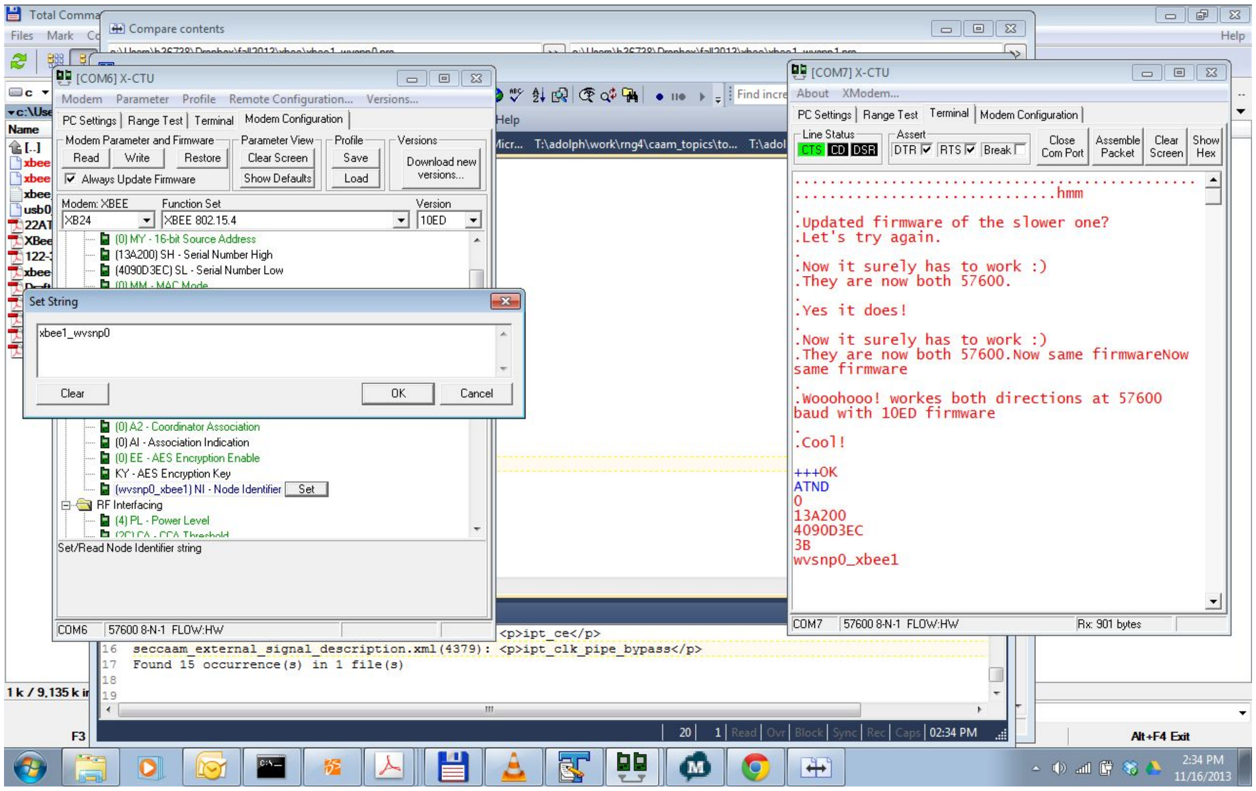
```
-rw-r--r-- 1 aseema aseema  3102 Nov 13 22:31 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

II. Let’s compile the example.

```
$ make; ./main
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/0.simple_at$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee

:~/mygits/my_libxbee/libxbee3/sample/xbee1/0.simple_at$ ./main
tx: 0
Response is 12 bytes long:
  0: 0x78 - x
  1: 0x62 - b
  2: 0x65 - e
  3: 0x65 - e
  4: 0x31 - 1
  5: 0x5F - _
  6: 0x77 - w
  7: 0x76 - v
  8: 0x73 - s
  9: 0x6E - n
 10: 0x70 - p
 11: 0x30 - 0
:~/mygits/my_libxbee/libxbee3/sample/xbee1/0.simple_at$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

This means that the library correctly read my modules’ Node Identifier (NI=xbee1_wvsnp0) which I set as in this image below:



Set String: xbee1_wvsnp0

(wvsnp0_xbee1) NI - Node Identifier

COM6 57600 8-N-1 FLOW:HW

```
.............................................
...............................hmm
.
.Updated firmware of the slower one?
.Let's try again.
.
.Now it surely has to work :)
.They are now both 57600.
.
.Yes it does!
.
.Now it surely has to work :)
.They are now both 57600.Now same firmwareNow
same firmware
.
.Wooohooo! workes both directions at 57600
baud with 10ED firmware
.
.Cool!

+++OK
ATND
0
13A200
4090D3EC
3B
wvsnp0_xbee1
```

COM7 57600 8-N-1 FLOW:HW Rx: 901 bytes

III. How the two module network is configured.

It is probably a good idea at this point to tell you that, I configure two XBee1 modules using X-CTU and plugged them both to my Windoze to configure them fully before i ran these tests on the library.

-----------------Module 1 showed up as COM6 and on Linux it is /dev/ttyUSB0-----------------
I upgraded each's firmware to the latest and each's baud rate to 57600.

---------------USB0---------------

XB24
Firmware=10ED
SH=13A200 SL=4090D3EC
DH=13A200 DL=4090D3DF
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Source Address=FFFF

Detailed Profile saved as: xbee1_wvsnp0.pro

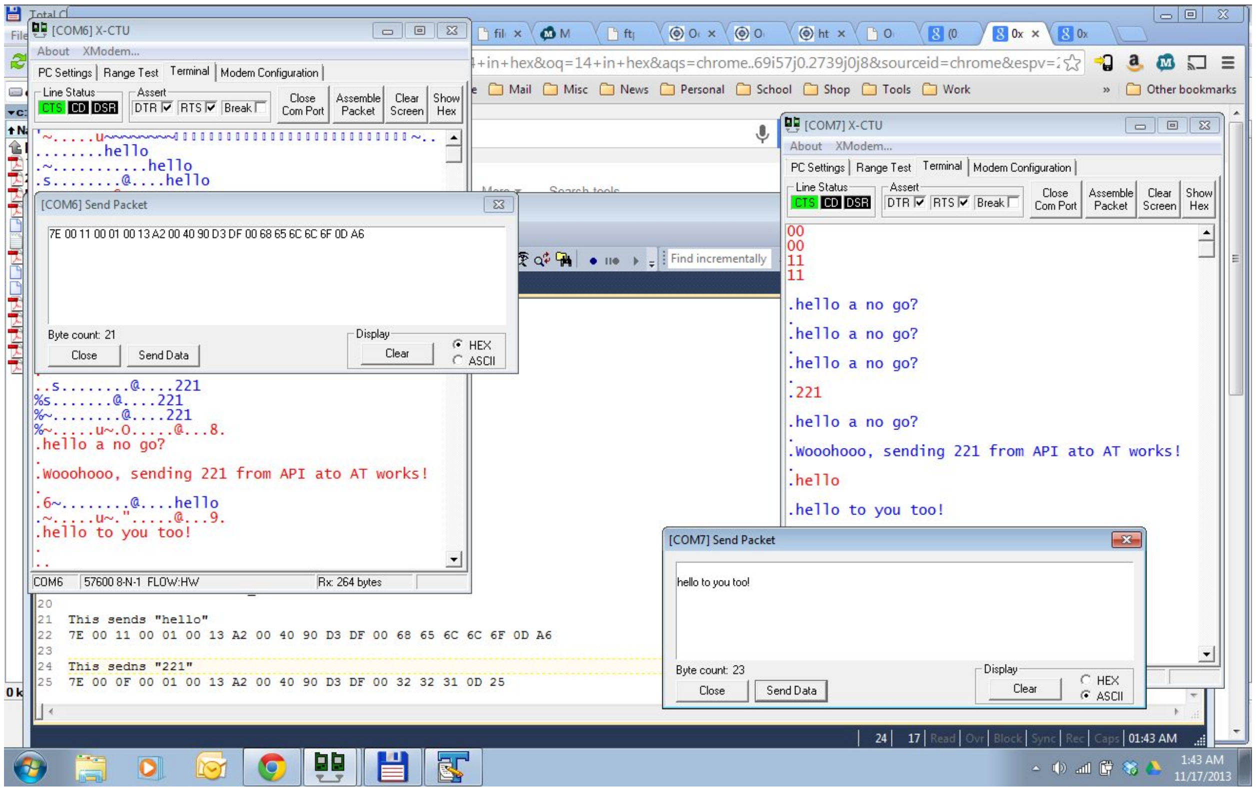
--------------USB1---------------

XB24
Firmware=10ED
SH=13A200 SL=4090D3DF
DH=13A200 DL=4090D3EC
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Source Address=FFFF

Detailed Profile saved as: xbee1_wvsnp1.pro

So, as the profiles show, at this point I disabled 16-bit addressing for each (MY=FFFF) and set USB0 to API mode (AP=1) then connected it to the linux USB as "/dev/ttyUSB0" baud 57600 (BD=6) to match the library examples. Also note that the network is setup as peer to peer duplex between these modules pointing to each other. That is DL of USB0 is set of SL of USB1 module and vice versa.

USB1 is still connected to X-CTU on a windows machine in AT-Mode at this point. The image below shows how I tested both on Windows to make sure they can send to each other even though the other one was in API mode 1 and the other one under AT mode.



```
'~.....u~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~..
........hello
.~...........hello
.s........@....hello
```

[COM6] Send Packet

```
7E 00 11 00 01 00 13 A2 00 40 90 D3 DF 00 68 65 6C 6C 6F 0D A6
```

Byte count: 21

```
..s........@....221
%s.......@....221
%~........@....221
%~.....u~.O.....@...8.
.hello a no go?
.
.Wooohooo, sending 221 from API ato AT works!
.
.6~........@....hello
.~.....u~.".....@...9.
.hello to you too!
.
..
```

COM6 57600 8-N-1 FLOW:HW Rx: 264 bytes

```

This sends "hello"
7E 00 11 00 01 00 13 A2 00 40 90 D3 DF 00 68 65 6C 6C 6F 0D A6

This sedns "221"
7E 00 0F 00 01 00 13 A2 00 40 90 D3 DF 00 32 32 31 0D 25
```

```
00
00
11
11

.hello a no go?
.
.hello a no go?
.
.hello a no go?
.
.221

.hello a no go?
.
.Wooohooo, sending 221 from API ato AT works!
.
.hello

.hello to you too!
```

[COM7] Send Packet

```
hello to you too!
```

Byte count: 23

Also note the example on the bottom left showing how to send “hello” and “221” in API mode frame. You read the above linked documents right?

Ok let’s move on then.

Building all samples at once works as well, so lets go to the top of all examples and run the buildall script.

```
$ cd ../../
$ ls -l
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample$ ls -l
total 44
-rwxr-xr-x  1 aseema aseema  176 Nov  9 15:33 buildall
drwxr-xr-x  3 aseema aseema 4096 Nov  9 15:33 debug
drwxr-xr-x  5 aseema aseema 4096 Nov  9 15:33 generic
-rw-r--r--  1 aseema aseema  212 Nov  9 15:33 makefile
-rw-r--r--  1 aseema aseema  242 Nov  9 15:33 makefile.cpp
drwxr-xr-x  5 aseema aseema 4096 Nov  9 15:33 net
drwxr-xr-x 19 aseema aseema 4096 Nov  9 15:33 xbee1
drwxr-xr-x 11 aseema aseema 4096 Nov  9 15:33 xbee2
drwxr-xr-x  3 aseema aseema 4096 Nov  9 15:33 xbee5
drwxr-xr-x  5 aseema aseema 4096 Nov  9 15:33 xbee6b
drwxr-xr-x  5 aseema aseema 4096 Nov  9 15:33 xbeeZB
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ ./buildall
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample$ ./buildall
make: Entering directory
`/home/aseema/mygits/my_libxbee/libxbee3/sample/xbee6b/1.simple_at'
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee

… snip …

make: Leaving directory `/home/aseema/mygits/my_libxbee/libxbee3/sample/net/3.simple_at'
make: Entering directory
`/home/aseema/mygits/my_libxbee/libxbee3/sample/xbee5/1.simple_at'
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
make: Leaving directory `/home/aseema/mygits/my_libxbee/libxbee3/sample/xbee5/1.simple_at'
:~/mygits/my_libxbee/libxbee3/sample$
```

```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

So it looks like all the samples built successfully!

IV. Running XBee1 “1.simple_at” example.

This example proved that the callback functions are firing as expected. Note that this time the NI I set earlier is now printed from the callback function.

```
$ cd ../1.simple_at/; ls -l; make; ./main
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/1.simple_at$ ls -l
total 24
-rwxr-xr-x 1 aseema aseema 16394 Nov 13 22:35 main
-rw-r--r-- 1 aseema aseema  3213 Nov 17 03:05 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/1.simple_at$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
rx: [xbee1_wvsnp0]
tx: 0
:~/mygits/my_libxbee/libxbee3/sample/xbee1/1.simple_at$ ./main
tx: 0
rx: [xbee1_wvsnp0]
:~/mygits/my_libxbee/libxbee3/sample/xbee1/1.simple_at$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Attie sya: “This is only working because you have installed libxbee... otherwise you'd need the LD_LIBRARY_PATH environment variable set (as above).”

V. Running XBee1 “2.remote_at” example:

This example proves that the callback functions are firing as expected and that 64-bit addressing & communication with a remote node works. For this test the remote node is connected COM7 of a Windoze machine in AT mode. It is configured as:

--------------USB1---------------

XB24
Firmware=10ED
SH=13A200 SL=4090D3DF
DH=13A200 DL=4090D3EC

FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Source Address=FFFF

Detailed Profile saved as: xbee1_wvsnp1.pro

```
$ cd ../2.remote_at/; ls -l; make; ./main
```

```
++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/2.remote_at$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
tx: 0
rx: [xbee1_wvsnp1]
:~/mygits/my_libxbee/libxbee3/sample/xbee1/2.remote_at$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Note that it was able to query the NI of the remote module which I previously named: "xbee1_wvsnp1". when I configured the 2 modules prior to this test.

VI. Running XBee1 "3.simple_data" example:

This example proves that you can receive real text (i.e. data) from a remote module. The program runs in a loop forever waiting for input from a remote module at the address set in this programs main.c i.e. for this test the address was hard coded to:

```
<code>

//---This is the 64-bit address of the module that this module
//is hard coded to send to.
//DH
address.addr64[0] = 0x00;
address.addr64[1] = 0x13;
address.addr64[2] = 0xA2;
address.addr64[3] = 0x00;
//DL
address.addr64[4] = 0x40;
address.addr64[5] = 0x90;
address.addr64[6] = 0xD3;
address.addr64[7] = 0xDF;

</code>
```

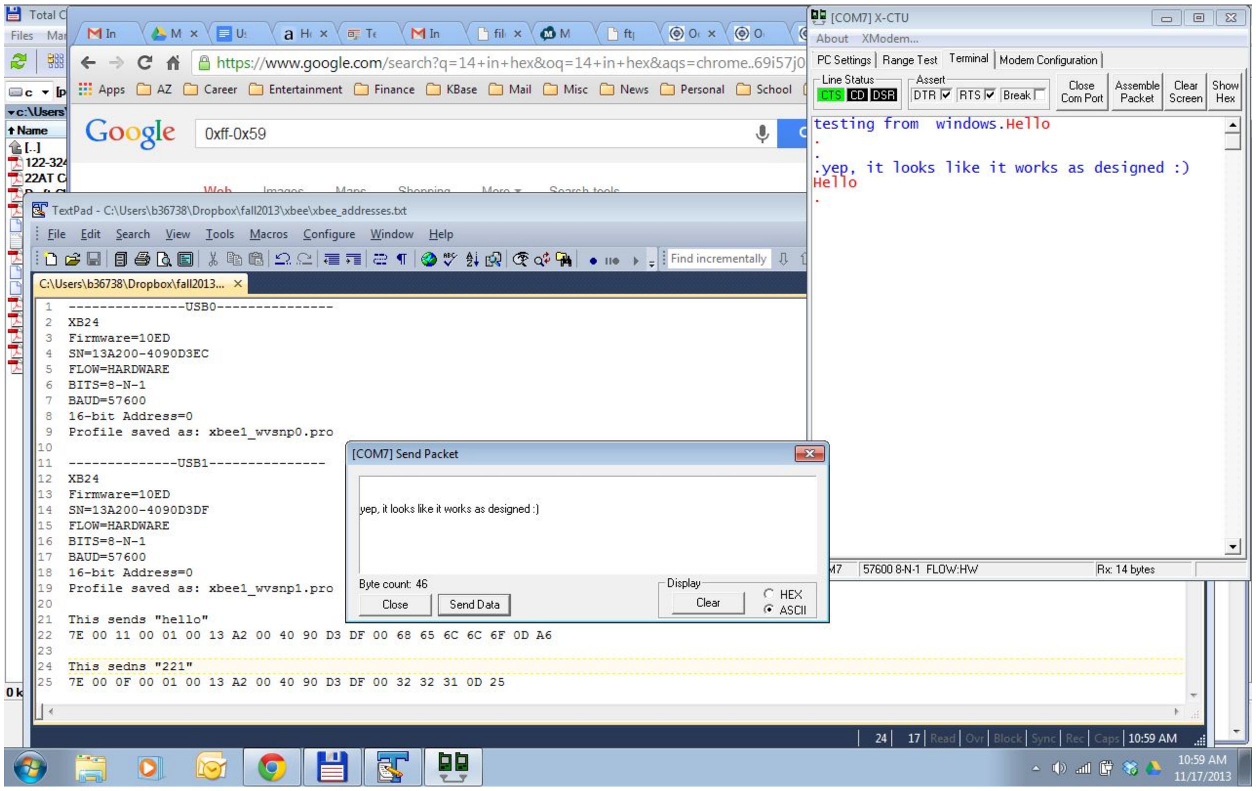
Again, for this test, the remote node is connected to COM7 of a Windoze machine in AT mode. It is configured as:

```
--------------USB1---------------

XB24
Firmware=10ED
SH=13A200 SL=4090D3DF
DH=13A200 DL=4090D3EC
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Source Address=FFFF (i.e. 16-bit addressing disabled).
```

Detailed Profile saved as: xbee1_wvsnp1.pro

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/2.remote_at$ cd ../3.simple_data/; ls -l
total 24
-rwxr-xr-x 1 aseema aseema 16894 Nov 13 23:27 main
-rw-r--r-- 1 aseema aseema  3218 Nov 13 23:26 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/3.simple_data$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
rx: [testing from  windows.]
tx: 0
rx: [

yep, it looks like it works as designed :)]
tx: 0
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Note above that I was able to send text/data from a remote module which I previously named: "xbee1_wvsnp1". when I configured the 2 modules prior to this test. The screenshot below shows the action of duplex communication from the Windows modules point of view of this data exchange. Note that the Linux module sent back a "Hello" every time the Windows module sent it text. You can do something more useful than "Hello" of course :) This is where your own project and incoming data parsing begins.



```
---------------USB0---------------
XB24
Firmware=10ED
SN=13A200-4090D3EC
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Address=0
Profile saved as: xbee1_wvsnp0.pro

--------------USB1---------------
XB24
Firmware=10ED
SN=13A200-4090D3DF
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Address=0
Profile saved as: xbee1_wvsnp1.pro

This sends "hello"
7E 00 11 00 01 00 13 A2 00 40 90 D3 DF 00 68 65 6C 6C 6F 0D A6

This sedns "221"
7E 00 0F 00 01 00 13 A2 00 40 90 D3 DF 00 32 32 31 0D 25
```

```
testing from  windows.Hello
.
.
.yep, it looks like it works as designed :)
Hello
.
```

[COM7] Send Packet

yep, it looks like it works as designed :)

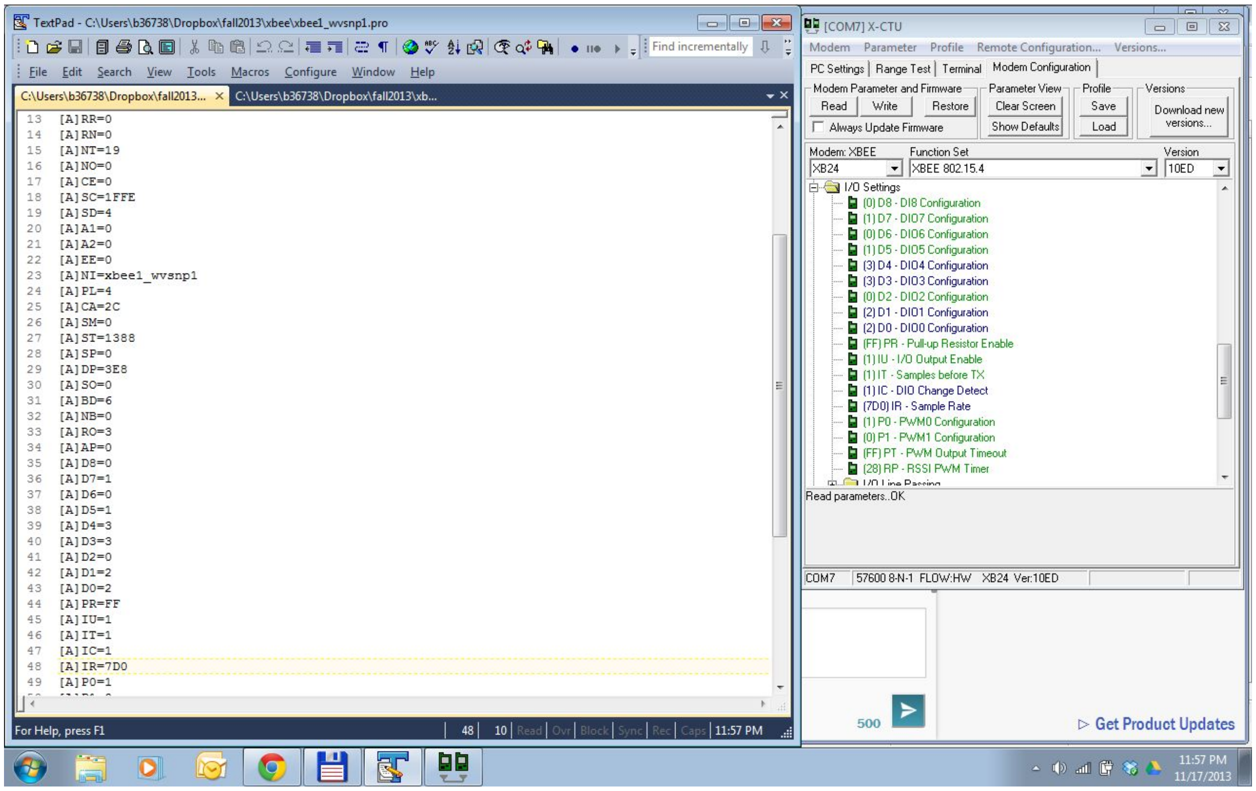
Byte count: 46

VII. Running XBee1 “4.simple_io” example:

This example proves that you can receive real IO data (i.e. pin status) from a remote module’s digital/analog inputs or outputs. To do so, I configured the remote module on Windows via X-CTU Modem Configuration tab, to the following:

o As usual I changed the hard coded the address of the remote IO module xbee1_wvsnp1’s one.
o I/O Setting D0 to ADC (2) (to read analog input)
o I/O Setting D1 to ADC (2) (to read analog input)
o I/O Setting D3 to DI (3) (to read digital input), ***pin #17 of the XBee1 module.***
o I/O Setting D4 to DI (3) (to read digital input)
o IR Setting to $7D0 (2000 decimal to sample once every two seconds)

Then write the parameters to the module. See screenshot of the saved profile and the “read back” parameters showing. Note this example only cares about D3 Digital Input.


TextPad - C:\Users\b36738\Dropbox\fall2013\xbee\xbee1_wvsnp1.pro
File Edit Search View Tools Macros Configure Window Help
C:\Users\b36738\Dropbox\fall2013... C:\Users\b36738\Dropbox\fall2013\xb...
For Help, press F1 48 10 Read Ovr Block Sync Rec Caps 11:57 PM
[COM7] X-CTU
Modem Parameter Profile Remote Configuration... Versions...
PC Settings Range Test Terminal Modem Configuration
Modem Parameter and Firmware Read Write Restore Always Update Firmware
Parameter View Clear Screen Show Defaults
Profile Save Load
Versions Download new versions...
500
Get Product Updates
11:57 PM 11/17/2013

```
[A]RR=0
[A]RN=0
[A]NT=19
[A]NO=0
[A]CE=0
[A]SC=1FFE
[A]SD=4
[A]A1=0
[A]A2=0
[A]EE=0
[A]NI=xbee1_wvsnp1
[A]PL=4
[A]CA=2C
[A]SM=0
[A]ST=1388
[A]SP=0
[A]DP=3E8
[A]SO=0
[A]BD=6
[A]NB=0
[A]RO=3
[A]AP=0
[A]D8=0
[A]D7=1
[A]D6=0
[A]D5=1
[A]D4=3
[A]D3=3
[A]D2=0
[A]D1=2
[A]D0=2
[A]PR=FF
[A]IU=1
[A]IT=1
[A]IC=1
[A]IR=7D0
[A]P0=1
```

```
Modem: XBEE    Function Set     Version
XB24           XBEE 802.15.4    10ED
I/O Settings
  (0) D8 - DI8 Configuration
  (1) D7 - DIO7 Configuration
  (0) D6 - DIO6 Configuration
  (1) D5 - DIO5 Configuration
  (3) D4 - DIO4 Configuration
  (3) D3 - DIO3 Configuration
  (0) D2 - DIO2 Configuration
  (2) D1 - DIO1 Configuration
  (2) D0 - DIO0 Configuration
  (FF) PR - Pull-up Resistor Enable
  (1) IU - I/O Output Enable
  (1) IT - Samples before TX
  (1) IC - DIO Change Detect
  (7D0) IR - Sample Rate
  (1) P0 - PWM0 Configuration
  (0) P1 - PWM1 Configuration
  (FF) PT - PWM Output Timeout
  (28) RP - RSSI PWM Timer
Read parameters..OK
COM7   57600 8-N-1  FLOW:HW   XB24  Ver:10ED
```

Then run the simple_io example.

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/3.simple_data$ cd ../4.simple_io/; ls -l
total 24
-rwxr-xr-x 1 aseema aseema 16562 Nov 11 21:53 main
-rw-r--r-- 1 aseema aseema  2332 Nov  9 15:33 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/4.simple_io$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main

:~/mygits/my_libxbee/libxbee3/sample/xbee1/4.simple_io$ ./main
D3: 1
D3: 1
D3: 1
D3: 0
D3: 0
D3: 1
D3: 0
D3: 1
D3: 1
```

```
D3: 0
D3: 0
D3: 0
D3: 1
D3: 0
D3: 1
^C
:~/mygits/my_libxbee/libxbee3/sample/xbee1/4.simple_io$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Note that since XBee has internal pull-up resistors on it spins the default value of the DIO3, (D3 value) is 1 (HIGH). To see it change value and see 0 (LOW) value, I took a wire and connected between GND and pin #17 of the xbee1_wvsnp1 module. This pull it to GND and hence an input value of as shown in the output above.

VIII. Running XBee1 “5.get_conTypes” example:

This example helps you figure out the type of connections your attached module supports. Attie has more detail here:

1. Creating connections
2. Using connections

Run the example:

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/4.simple_io$ cd ../5.get_conTypes/; ls -l;
total 16
-rwxr-xr-x 1 aseema aseema 10740 Nov 11 21:40 main
-rw-r--r-- 1 aseema aseema  1399 Nov  9 15:33 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/5.get_conTypes$ ./main
connection type 0 - Modem Status
connection type 1 - Transmit Status
connection type 2 - Local AT
connection type 3 - Remote AT
connection type 4 - 16-bit Data
connection type 5 - 64-bit Data
connection type 6 - 16-bit I/O
connection type 7 - 64-bit I/O
:~/mygits/my_libxbee/libxbee3/sample/xbee1/5.get_conTypes$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Above are what my xbee1_wvsnp0 supports. this is the coordinator module attached to my Ubuntu 13.10 Linux machine.

IX. Running XBee1 “6.log_config” example:

This example helps you figure out what log level is set to. This is helpful for debugging if you want the internals of the library print out details of what it is doing or where it gets stuck.

Run the example:

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/5.get_conTypes$ cd ../6.log_config/; ls -l;
total 16
-rwxr-xr-x 1 aseema aseema 11012 Nov 11 21:53 main
-rw-r--r-- 1 aseema aseema  1728 Nov  9 15:33 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/6.log_config$ ./main
libxbee log level is currently: 0

Don't forget you can set the log level via the environment, for example:
        XBEE_LOG_LEVEL=100 ./main

setting libxbee log level to: 100
DEV: 50#[main.c:52] main() 0xcd41f0: Test Message 2...
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Obviously from my example above it is not set to print out details of what it is doing. Now see what happens when I set it to 100 on the fly :)

```
$ XBEE_LOG_LEVEL=100 ./main
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/6.log_config$ XBEE_LOG_LEVEL=100 ./main
libxbee log level is currently: 100

Don't forget you can set the log level via the environment, for example:
        XBEE_LOG_LEVEL=100 ./main

 15#[thread.c:68] threadFunc() 0xcaa1f0: starting thread 0xcade60, function xbee_rxHandler()...
 15#[thread.c:68] threadFunc() 0xcaa1f0: starting thread 0xcadca0, function xbee_rx()...
DEV: 50#[main.c:44] main() 0xcaa1f0: Test Message 1...
setting libxbee log level to: 100
 15#[thread.c:68] threadFunc() 0xcaa1f0: starting thread 0xcae020, function xbee_tx()...
```

```
DEV: 50#[main.c:52] main() 0xcaa1f0: Test Message 2...
:~/mygits/my_libxbee/libxbee3/sample/xbee1/6.log_config$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

So this would be a good way to analyse what the library is doing as it moves data around.

X. Running XBee1 “7.broadcast” example:

This will broadcast numbers 0 to 1000 to any module within xbee1_wvsnp0’s network.
I changed the example a bit so it accepts “/dev/ttyUSB0” and to print out what it is broadcasting.
Leave the address given in example unchanged. It is the default broadcast address.

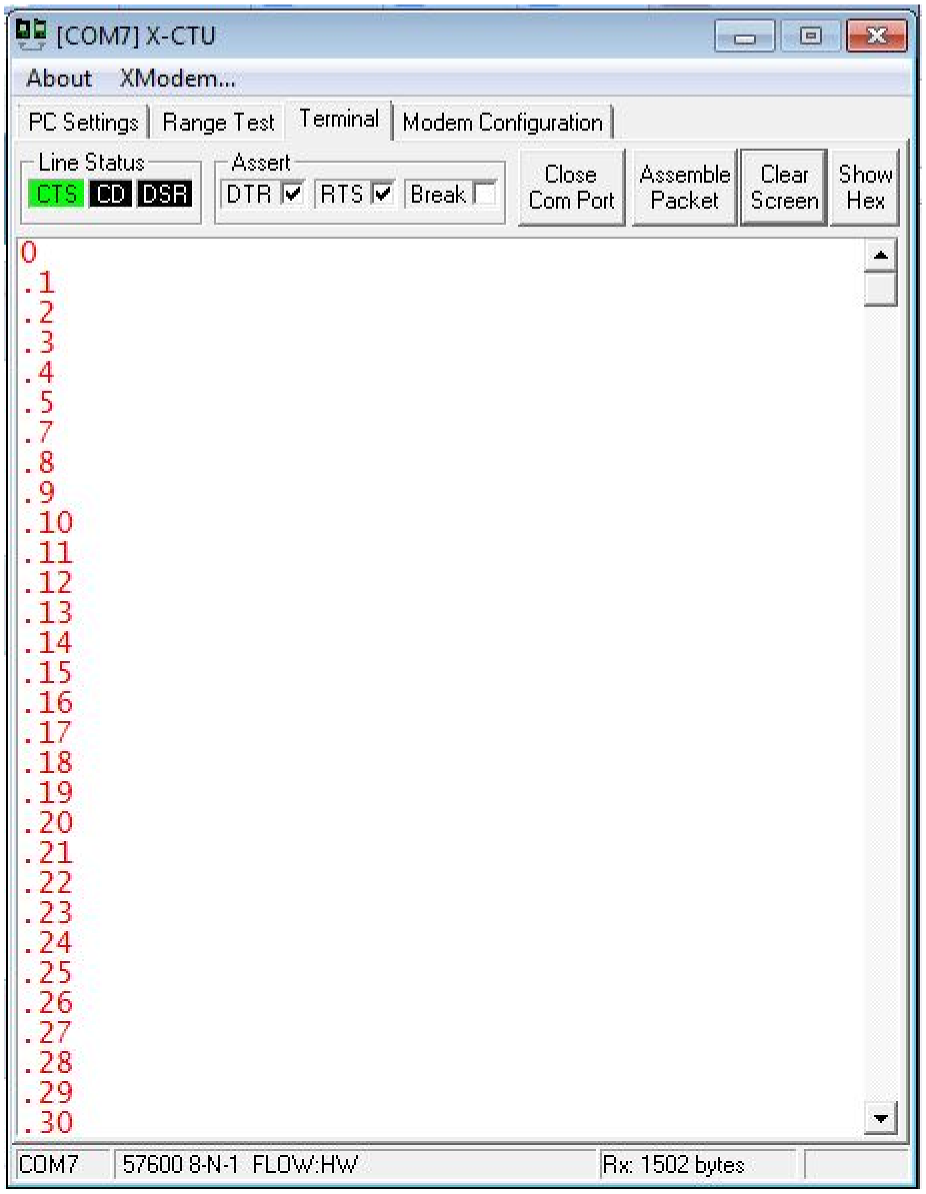
Run the example:

```
$ cd ../7.broadcast/; ls -l; make; ./main
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/6.log_config$ cd ../7.broadcast/; ls -l;
total 20
-rwxr-xr-x 1 aseema aseema 16314 Nov 14 11:53 main
-rw-r--r-- 1 aseema aseema  2478 Nov 18 01:03 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/7.broadcast$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
Missing tty port: USAGE: main '/dev/ttyUSB#'
make: *** [all] Error 255
:~/mygits/my_libxbee/libxbee3/sample/xbee1/7.broadcast$ ./main "/dev/ttyUSB0"
Successfully opened[/dev/ttyUSB0], ret[0]
Broadcasting[0].
Broadcasting[1].
Broadcasting[2].
Broadcasting[3].
Broadcasting[4].
Broadcasting[5].
Broadcasting[6].
Broadcasting[7].
Broadcasting[8].
Broadcasting[9].
Broadcasting[10].
Broadcasting[11].
Broadcasting[12].
...
```

+++

See the xbee1_wvsnp1 module on the windows machine receiving the broadcast message.



```
0
.1
.2
.3
.4
.5
.7
.8
.9
.10
.11
.12
.13
.14
.15
.16
.17
.18
.19
.20
.21
.22
.23
.24
.25
.26
.27
.28
.29
.30
```

COM7 | 57600 8-N-1 FLOW:HW | Rx: 1502 bytes

XI. Running XBee1 “8.catch-all” example:

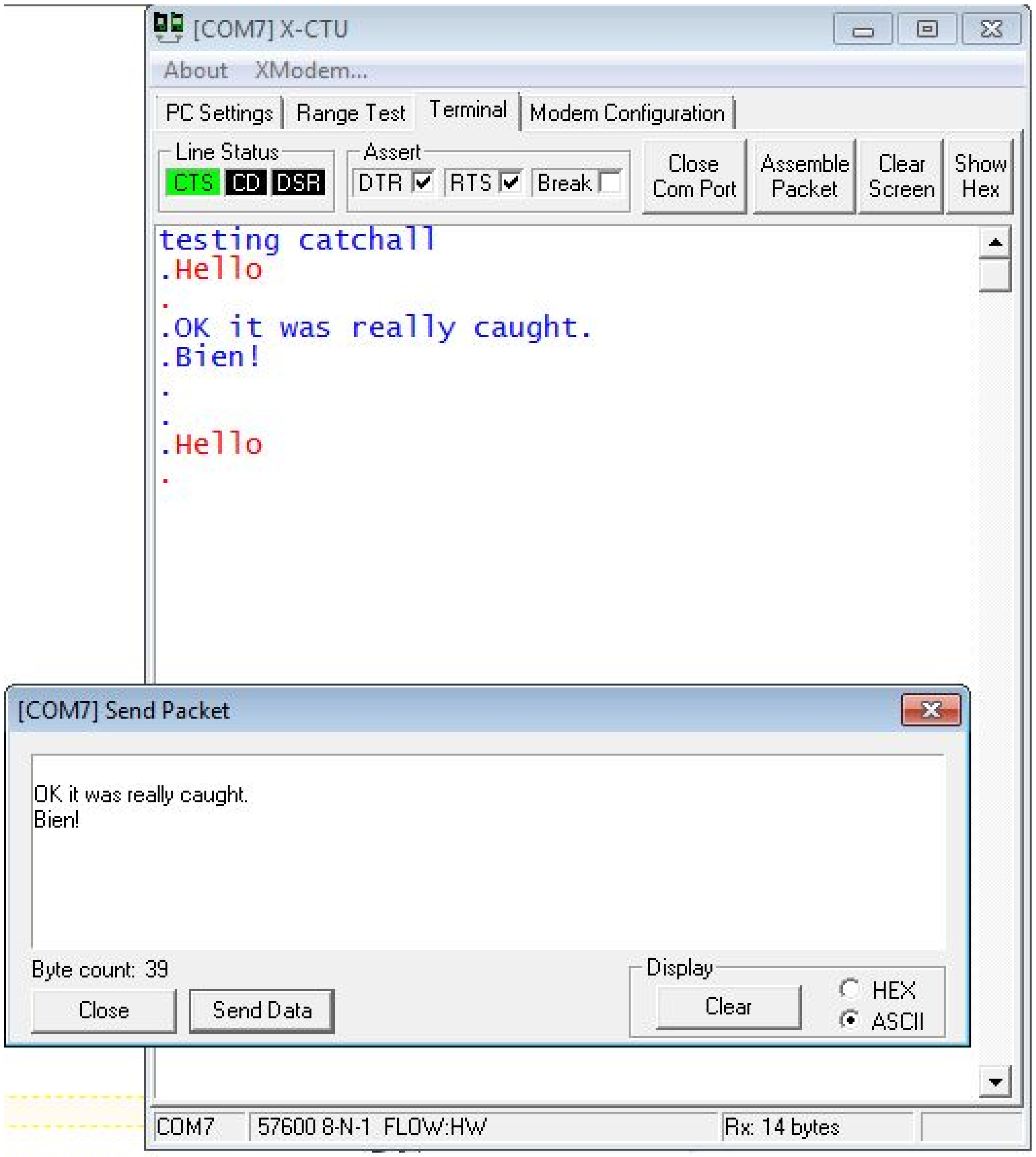
This example is the reverse of a broadcast. So it will accept anything that is sent by other XBee modules in its network. Leave the address given in example unchanged. It is the default catchall address.

Run the example:

```
$ cd ../8.catch-all/; ls -l; make; ./main
```

```
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/7.broadcast$ cd ../8.catch-all/; ls -l;
total 48
-rwxr-xr-x 1 aseema aseema 17302 Nov 14 10:51 main
-rwxr-xr-x 1 aseema aseema 17246 Nov 13 23:10 main1
-rw-r--r-- 1 aseema aseema  4481 Nov 18 01:35 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/8.catch-all$ rm main1
:~/mygits/my_libxbee/libxbee3/sample/xbee1/8.catch-all$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
Ready!... waiting for 30 secs
Got packet from new node!
   64-bit (0x0013A200 0x4090D3DF)
rx: [testing catchall
]
tx: 0
:~/mygits/my_libxbee/libxbee3/sample/xbee1/8.catch-all$ ./main
Ready!... waiting for 30 secs
Got packet from new node!
   64-bit (0x0013A200 0x4090D3DF)
rx: [
OK it was really caught.
Bien!


]
tx: 0
:~/mygits/my_libxbee/libxbee3/sample/xbee1/8.catch-all$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Above was sent from xbee1_wvsnp1 on windows. Note that xbee1_wvsnp0 sends back “Hello” to the correct source of the data using ***specificCB*** function.



```
testing catchall
.Hello
.
.OK it was really caught.
.Bien!
.
.
.Hello
.
```

```
OK it was really caught.
Bien!
```

Byte count: 39

Display: ( ) HEX (•) ASCII

COM7 | 57600 8-N-1 FLOW:HW | Rx: 14 bytes

XII. Running XBee1 “9.node_detect” example:

This is supposed to work just like the ATND command where it lists all the nodes it can find within the network. The sem* seems to be broken for now.
So will wait for Attie to fix it. Surprising that this sem* to work for the “11.uart_test” example.

Run the example:

```
$ cd ../9.node_detect/; ls -l; make; ./main
```

This time I have only one module so I am not expecting it to detect nodes in the network. Will test it again soon, but it works as designed.

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/9.node_detect$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee -lpthread
:~/mygits/my_libxbee/libxbee3/sample/xbee1/9.node_detect$ ls -l
total 24
-rwxr-xr-x 1 aseema aseema 17117 Nov 19 22:25 main
-rw-r--r-- 1 aseema aseema  3213 Nov 13 21:14 main.c
lrwxrwxrwx 1 aseema aseema    22 Nov 19 19:57 makefile -> ../../makefile.pthread
:~/mygits/my_libxbee/libxbee3/sample/xbee1/9.node_detect$ ./main
ND Sent!... waiting for completion
Scan complete!
:~/mygits/my_libxbee/libxbee3/sample/xbee1/9.node_detect$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

The output should pretty much be the same as the "Running XBeeZB "9.node_detect" example" below.

XIII. Running XBee1 "10.c++" example:

This is supposed to showcase C++ interface to the library.
a-- It prints out all modules/modes the library supports.
b-- Then prints the mode of the module it is currently running under.
(see bug comments in the actual modules tested in XBeeZB section).
c-- Then prints available connection types for the above xbee module.
d-- Then prints "Callback!!" to make sure callbacks work.
e-- Then it requests the local name of the running module (NI).
IF you comment out "#define LOCAL_CONNECTION" this will be the name of the remote module. You need to modify the hard coded address in code to the remote modules' SH and SL address.
f-- If you comment out: "#define USE_CALLBACKS" then you are not using callbacks functionality. You won't see "d--" above nor "Testing, 1… 2… 3…" here. You will see only stuff up to "c--" above, then "Packet length: #" and the print out of each character of the packet which is the remote/local name (NI) you just requested.
Packet length # is the number of characters in the name. Each character is printed on a separate line preceded by its index in the "packet".

For real output see: Running XBeeZB "10.c++" example. It should be identical to xbee1 except the slightly different list of connection types, library mode and of course an appropriately different module name (NI).

i.e. Instead of this:

```
… snip …

   Running libxbee in mode 'xbeeZB'
Available connection types:
  Transmit Status  Modem Status  Local AT  Remote AT  Data  Data (explicit)  I/O  Sensor  Identify  OTA Update Status
Callback!!
zigbee_wvsnp1

… snip …
```

You will see this:

```
… snip …

:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$ ./main
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbee1'
Available connection types:
  Modem Status  Transmit Status  Local AT  Remote AT  16-bit Data  64-bit Data  16-bit I/O  64-bit I/O
Callback!!
xbee1_wvsnp1
Testing, 1... 2... 3...
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$

… snip …
```

Again to run it do:

```
$ cd ../10.c++/; ls -l; make; ./main
```

Compare with the Running XBeeZB “10.c++” example output below.

XIV. Running XBee1 “11.uart_test” example:

This example helps you figure out the reliability of your UART. It sends NI query to the remote module and and checks is its request times out. Repeats this 100o times. Then it output the percentage success.

Run the example:

```
$ cd ../11.uart_test/; ls -l; make; ./main
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$ cd ../11.uart_test/; ls -l;
total 28
```

```
-rwxr-xr-x 1 aseema aseema 16991 Nov 11 21:40 main
-rw-r--r-- 1 aseema aseema  2551 Nov  9 15:33 main.c
-rw-r--r-- 1 aseema aseema   222 Nov  9 15:33 makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/11.uart_test$ ./main
rx: [xbee1_wvsnp0]
tx: 0
tx: 0
rx: [xbee1_wvsnp0]
tx: 0
rx: [xbee1_wvsnp0]

… snip ...

tx: 0
rx: [xbee1_wvsnp0]
tx: 0
rx: [xbee1_wvsnp0]
1000 / 0 / 1000 - success/timeout/total - success rate (100.0%) / timeout rate (0.0%)
:~/mygits/my_libxbee/libxbee3/sample/xbee1/11.uart_test$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Apparently mine passes 100 percent!

XV. Running XBee1 “12.timestamp” example:

This example helps you learn how to do time stamping of the the packet.
It sends an NI query to a remote node and then timestamps the incoming RX data.

Run the example:

```
$ cd ../12.timestamp/; ls -l; make; ./main
```

```
+++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/11.uart_test$ cd ../12.timestamp/; ls -l;
total 24
-rwxr-xr-x 1 aseema aseema 16437 Nov 11 21:53 main
-rw-r--r-- 1 aseema aseema  2091 Nov  9 15:33 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/12.timestamp$ ./main
tx: 0
rx: [xbee1_wvsnp0] @ 1384767118.768480872 - Mon Nov 18 02:31:58 2013

:~/mygits/my_libxbee/libxbee3/sample/xbee1/12.timestamp$ make
```

```
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
tx: 0
rx: [xbee1_wvsnp0] @ 1384767160.724053046 - Mon Nov 18 02:32:40 2013

:~/mygits/my_libxbee/libxbee3/sample/xbee1/12.timestamp$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
tx: 0
rx: [xbee1_wvsnp0] @ 1384767193.495388846 - Mon Nov 18 02:33:13 2013

:~/mygits/my_libxbee/libxbee3/sample/xbee1/12.timestamp$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Cool.

XVI. Running XBee1 “13.lxb2lxb_client” and “13.lxb2lxb_server” example:

This example sets up one XBee module as a client and the other as server. In this test, the modules are setup in API mode and both plugged into the Ubuntu Linux laptop’s USB ports as:

```
Client:---------------/dev/ttyUSB0---------------

XB24
Firmware=10ED
SH=13A200 SL=4090D3EC
DH=13A200 DL=4090D3DF
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Source Address=FFFF
NI=xbee1_wvsnp0

Server:--------------/dev/ttyUSB1---------------

XB24
Firmware=10ED
SH=13A200 SL=4090D3DF
DH=13A200 DL=4090D3EC
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600
16-bit Source Address=FFFF
```

NI=xbee1_wvsnp1

Start the server first as below. It will wait until it receives a “Hello?” from the client. Once it receives a “Hello?”, it will respond with a “Hi!” which you will see immediately after invoking the client.

Run each example in a separate terminal:

```
$ cd ../13.lxb2lxb_server/; ls -l; make; ./main
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/13.lxb2lxb_server$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
Missing tty port: USAGE: main '/dev/ttyUSB#'
make: *** [all] Error 255
:~/mygits/my_libxbee/libxbee3/sample/xbee1/13.lxb2lxb_server$ ./main /dev/ttyUSB1
Successfully opened[/dev/ttyUSB1], ret[0]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
^C
:~/mygits/my_libxbee/libxbee3/sample/xbee1/13.lxb2lxb_server$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ cd ../13.lxb2lxb_client/; ls -l; make; ./main
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/13.lxb2lxb_client$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
Missing tty port: USAGE: main '/dev/ttyUSB#'
```

```
make: *** [all] Error 255
:~/mygits/my_libxbee/libxbee3/sample/xbee1/13.lxb2lxb_client$ ./main /dev/ttyUSB0
Successfully opened[/dev/ttyUSB0], ret[0]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
^C
:~/mygits/my_libxbee/libxbee3/sample/xbee1/13.lxb2lxb_client$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Works!

XVII. Running XBee1 "14.request_io" example:

This example differs from **"4.simple_io"** example because this one polls for pin values on the remote module. Remember in that example I activated 2 pins for Analog IO (A0 and A1) and 2 pins for digital IO (D3 and D4). In **"4.simple_io"** example, xbee1_wvsnp1 was configured to periodically send its IO values to xbee1_wvsnp0. xbee1_wvsnp0 only printed out the remote pin's (D3) value when it receives it (interrupt driven). In this example it is polling the remote module for values of all activated IO pins in the remote module. Since I activated 4 of them it prints only 4 values.

Run the example:

```
$ cd ../14.request_io/; ls -l; make; ./main
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1$ cd 14.request_io/
:~/mygits/my_libxbee/libxbee3/sample/xbee1/14.request_io$ ls -l
total 24
-rwxr-xr-x 1 aseema aseema 16661 Nov 11 21:40 main
```

```
-rw-r--r-- 1 aseema aseema  2844 Nov 18 21:50 main.c
lrwxrwxrwx 1 aseema aseema    14 Nov  9 15:33 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbee1/14.request_io$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
LD_LIBRARY_PATH=../../../lib ./main
D3: 1
D4: 1
A0: 1023
A1: 1023
:~/mygits/my_libxbee/libxbee3/sample/xbee1/14.request_io$
:~/mygits/my_libxbee/libxbee3/sample/xbee1/14.request_io$ ./main
D3: 1
D4: 0
A0: 1023
A1: 1023
:~/mygits/my_libxbee/libxbee3/sample/xbee1/14.request_io$ ./main
D3: 0
D4: 1
A0: 1023
A1: 1023
:~/mygits/my_libxbee/libxbee3/sample/xbee1/14.request_io$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Again, to read zero values on D3 and D4 above, I connected each the respective pin to GND before running the program to get the values above. D4 only for the 2nd run and D3 only for the third run.

XVIII. Running XBee1 “15.qt” example:

This example seems to show a Qt GUI framework using XBee. The README says you can compile it via qmake but it is not saying much about all the Qt dependency setup, etc.

```
$ cd ../15.qt/; ls -l;
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
total 32
-rw-r--r-- 1 aseema aseema  357 Nov  9 15:33 15qt.pro
-rw-r--r-- 1 aseema aseema 1001 Nov  9 15:33 main.cpp
-rw-r--r-- 1 aseema aseema   95 Nov  9 15:33 README
-rw-r--r-- 1 aseema aseema 2789 Nov  9 15:33 window.cpp
-rw-r--r-- 1 aseema aseema 1569 Nov  9 15:33 window.h
-rw-r--r-- 1 aseema aseema 3637 Nov  9 15:33 window.ui
-rw-r--r-- 1 aseema aseema 1116 Nov  9 15:33 xbeeqt.cpp
```

```
-rw-r--r-- 1 aseema aseema 1358 Nov  9 15:33 xbeeqt.h
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ qmake
```

```
---------------------------------------------BAD OUTPUT---------------------------------------------
:~/mygits/my_libxbee/libxbee3/sample/xbee1/15.qt$ qmake
qmake: could not find a Qt installation of ''
:~/mygits/my_libxbee/libxbee3/sample/xbee1/15.qt$
----------------------------------------------------------------------------------------------------------
```

Technically this not a bad output. I just have not bothered to install QT.

So, the libxbee XBee1 tutorial is a success.
Let's all thank Attie Grande for the great work and service!

I will test XBee Zigbee and XBee WiFi next.
----

6. Updating the source code from previous to the latest?

Do this:

```
$ cd ~/mygits/my_libxbee/libxbee3
$ git reset --hard HEAD
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ git reset --hard HEAD
HEAD is now at ec3bb90 implemented file locking on Linux, to prevent multiple applications
accessing the same XBee
:~/mygits/my_libxbee/libxbee3$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ git pull
```

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ git pull
remote: Counting objects: 35, done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 19 (delta 11), reused 16 (delta 8)
Unpacking objects: 100% (19/19), done.
From https://github.com/attie/libxbee3
```

```
   aad1fce..1208e5c  master     -> origin/master
   90163ba..ef2fa04  mingw32    -> origin/mingw32
Updating aad1fce..1208e5c
Fast-forward
 README                              | 6 +++++-
 sample/xbee1/10.c++/main.cpp        | 1 +
 sample/xbee1/14.request_io/main.c   | 1 +
 sample/xbee2/9.node_detect/makefile | 2 +-
 4 files changed, 8 insertions(+), 2 deletions(-)
:~/mygits/my_libxbee/libxbee3$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Note that we updated this now from version: **[aad1fce]** to version **[1208e5c]**, lets rebuild the code. See the files that were updated above.

```
$ make clean
```

```
+++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ make clean
rm -f .build/*.o .build/*.d
:~/mygits/my_libxbee/libxbee3$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
$ make all
```

I am just highlighting the warnings that still exist for future reference.

```
+++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ make all
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD conn.c -c -o .build/conn.o
conn.c: In function ‘xbee_conAddressCmpDefault’:
conn.c:241:2: warning: #warning TODO - handle broadcast endpoint, but probably not here...
[-Wcpp]
 #warning TODO - handle broadcast endpoint, but probably not here...
  ^
conn.c: In function ‘xbee_conCallbackProd’:
… snip …
conn.c:926:2: warning: #warning TODO - there is a gap here, needs a mutex [-Wcpp]
 #warning TODO - there is a gap here, needs a mutex
  ^
… snip …
xsys_linux.c:97:2: warning: implicit declaration of function ‘flock’ [-Wimplicit-function-declaration]
```

```
  if (flock(info->dev.fd, LOCK_EX | LOCK_NB) == -1) {
  ^
… snip …
modes/xbee2/mode.c:107:2: warning: #warning TODO - currently missing out on the resolved
network address, retry count, and discovery status [-Wcpp]
 #warning TODO - currently missing out on the resolved network address, retry count, and
discovery status
  ^
… snip …
modes/xbee3/data.c:101:2: warning: #warning TODO - currently missing support for NAK and
Trace Route messages [-Wcpp]
 #warning TODO - currently missing support for NAK and Trace Route messages
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbee3/dataExp.c -c
-o .build/modes_xbee3_dataExp.o
modes/xbee3/dataExp.c: In function ‘xbee_s3_dataExp_tx_func’:
modes/xbee3/dataExp.c:129:2: warning: #warning TODO - currently missing support for NAK
and Trace Route messages [-Wcpp]
 #warning TODO - currently missing support for NAK and Trace Route messages
  ^
… snip …
modes/xbee6b/mode.c:186:2: warning: #warning TODO - figure out how to use this feedback...
[-Wcpp]
 #warning TODO - figure out how to use this feedback...
  ^
… snip …
modes/xbeeZB/mode.c:108:2: warning: #warning TODO - currently missing out on the resolved
network address, retry count, and discovery status [-Wcpp]
 #warning TODO - currently missing out on the resolved network address, retry count, and
discovery status
  ^
gcc -Wall -c -fPIC -g -DXBEE_LOG_RX -DXBEE_LOG_TX -DLIBXBEE_BUILD
-fvisibility=hidden -Wno-variadic-macros -Wstrict-prototypes -MMD modes/xbeeZB/ota.c -c -o
.build/modes_xbeeZB_ota.o
modes/xbeeZB/ota.c: In function ‘xbee_sZB_ota_rx_func’:
modes/xbeeZB/ota.c:43:2: warning: #warning TODO - check that these are the correct
addresses to be using [-Wcpp]
 #warning TODO - check that these are the correct addresses to be using
  ^
… snip …
ln -fs `basename lib/libxbee.so.3.0.10` lib/libxbee.so
objcopy --only-keep-debug lib/libxbee.so.3.0.10 lib/libxbee.so.3.0.10.dbg
objcopy --add-gnu-debuglink=lib/libxbee.so.3.0.10.dbg lib/libxbee.so.3.0.10
objcopy --strip-debug lib/libxbee.so.3.0.10
touch lib/libxbee.so.3.0.10.dbg
ar rcs lib/libxbee.a.3.0.10 lib/libxbee.o
ln -fs `basename lib/libxbee.a.3.0.10` lib/libxbee.a
g++ -Wall -c -fPIC -g  -DLIBXBEE_BUILD -fvisibility=hidden -Wno-variadic-macros -MMD
xbeep.cpp -c -o .build/xbeep.o
ld -r -o .build/__corep.o .build/xbeep.o
ld -r -o lib/libxbeep.o .build/__corep.o
g++ -shared -Wl,--no-undefined -Wl,-soname,libxbeep.so.3.0.10 lib/libxbeep.o -fPIC -lpthread
-lrt -g -Llib -lxbee -o lib/libxbeep.so.3.0.10
ln -fs `basename lib/libxbeep.so.3.0.10` lib/libxbeep.so
objcopy --only-keep-debug lib/libxbeep.so.3.0.10 lib/libxbeep.so.3.0.10.dbg
objcopy --add-gnu-debuglink=lib/libxbeep.so.3.0.10.dbg lib/libxbeep.so.3.0.10
objcopy --strip-debug lib/libxbeep.so.3.0.10
touch lib/libxbeep.so.3.0.10.dbg
ar rcs lib/libxbeep.a.3.0.10 lib/libxbeep.o
ln -fs `basename lib/libxbeep.a.3.0.10` lib/libxbeep.a
:~/mygits/my_libxbee/libxbee3$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

So far so good. Still 9 warnings as before. Lets re-install it.

```
$ sudo make install
```

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3$ sudo make install
[sudo] password for aseema:
install -g root -o root -DT -m 755 lib/libxbee.so.3.0.10 /usr/lib/libxbee.so.3.0.10
install -g root -o root -DT -m 755 lib/libxbee.so.3.0.10.dbg /usr/lib/libxbee.so.3.0.10.dbg
install -g root -o root -DT -m 755 lib/libxbee.a.3.0.10 /usr/lib/libxbee.a.3.0.10
install -g root -o root -DT -m 755 lib/libxbeep.so.3.0.10 /usr/lib/libxbeep.so.3.0.10
install -g root -o root -DT -m 755 lib/libxbeep.so.3.0.10.dbg /usr/lib/libxbeep.so.3.0.10.dbg
install -g root -o root -DT -m 755 lib/libxbeep.a.3.0.10 /usr/lib/libxbeep.a.3.0.10
:~/mygits/my_libxbee/libxbee3$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

I. Going back to re-build examples.

7. Testing XBeeZB (i.e. S2 Zigbee) Examples

Below are the available XBeeZB examples that come with the library.

```
$ ls -l sample/xbeeZB/
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$ ls -l
drwxr-xr-x 2 aseema aseema 4096 Nov 21 10:42 1.simple_at
drwxr-xr-x 2 aseema aseema 4096 Nov 21 10:42 2.remote_at
drwxr-xr-x 2 aseema aseema 4096 Nov 21 10:42 3.simple_data
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Module 1 showed up as COM4 on Windows7 and on Ubuntu it is /dev/ttyUSB0
Module 2 showed up as COM8 on Windows7 and on Ubuntu it is /dev/ttyUSB1
I upgraded each's firmware to the latest and each's baud rate to 57600.

All xbeeZB examples below were only on Ubuntu 13.10 with two XBee Zigbee modules configured as:

--------------- /dev/ttyUSB0---------------

XB24-ZB --- Coordinator API
PAN-ID=3
Firmware=21A7
SH=13A200 SL=408BAD2D
DH=13A200 DL=408BAE32
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600

Detailed Profile saved as: zigbee_wvsnp0.pro

N.B. If you leave the coordinator's PAN-ID as ZERO(0) the Coordinator will create a random one and your network will likely not work.

-------------- /dev/ttyUSB1---------------

XB24-ZB --- Router API
PAN-ID=3
Firmware=23A7
SH=13A200 SL=408BAE32
DH=13A200 DL=408BAD2D
FLOW=HARDWARE
BITS=8-N-1
BAUD=57600

Detailed Profile saved as: zigbee_wvsnp1.pro

I. Running XBeeZB “1.simple_at” example.

This example proved that the callback functions are firing as expected. Note that the NI I set earlier is printed from the callback function. For simplicity, I compiled code with “/dev/ttyUSB1” and then renamed the binary “main_usb1”. Then recompiled it again “/dev/ttyUSB0” and left as “main”.

```
$ cd ../1.simple_at/; ls -l; make; ./main
```

```
+++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/1.simple_at$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/1.simple_at$ ls -l
total 44
-rwxr-xr-x 1 aseema aseema 16402 Nov 21 11:17 main
-rw-r--r-- 1 aseema aseema  3270 Nov 21 11:17 main.c
-rwxr-xr-x 1 aseema aseema 16402 Nov 21 10:42 main_usb1
lrwxrwxrwx 1 aseema aseema    14 Nov 21 10:38 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/1.simple_at$ ./main
rx: [zigbee_wvsnp0]
tx: 0
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/1.simple_at$ ./main_usb1
tx: 0
rx: [zigbee_wvsnp1]
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/1.simple_at$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

As the output shows, they work! Moving on …

II. Running XBeeZB “2.remote_at” example.

For simplicity, I compiled code with “/dev/ttyUSB1” and then renamed the binary “main_usb1”. Then recompiled it again “/dev/ttyUSB0” and left as “main”.
Note that below I ran the “1.simple_at/main” and “1.simple_at/main_usb1”. first for coordinator and router respectively to verify that they know themselves. The next, I ran the “2.remote_at/main” and “2.remote_at/main_usb1”. This proves that coordinator was able to ask the remote router its name and got a response. And vice versa for the router.

```
+++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ make
```

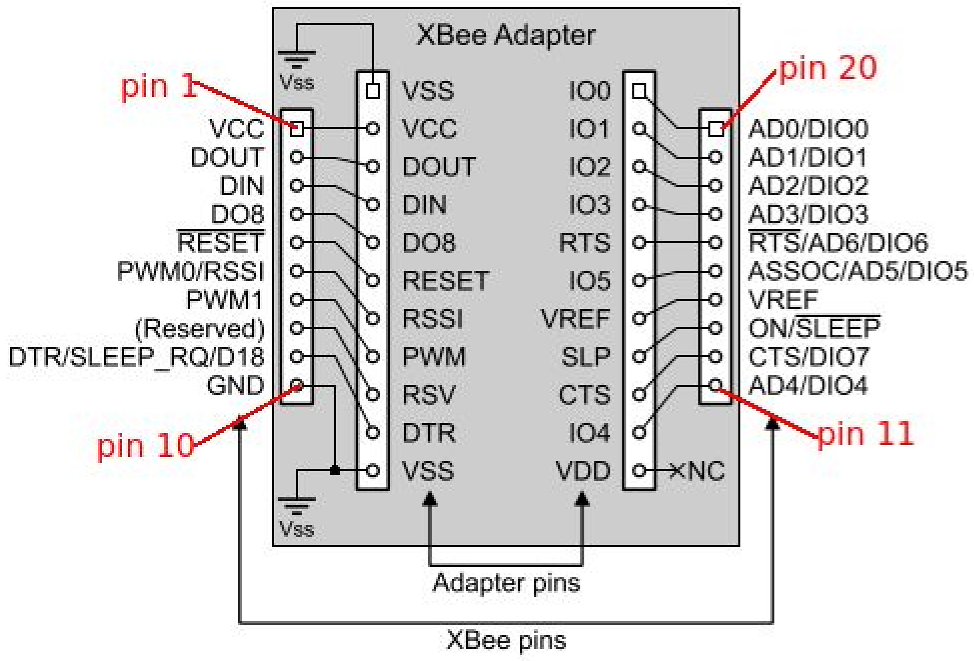
```
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ ls -l
total 48
-rwxr-xr-x 1 aseema aseema 16538 Nov 21 11:54 main
-rw-r--r-- 1 aseema aseema  2741 Nov 21 11:54 main.c
-rwxr-xr-x 1 aseema aseema 16538 Nov 21 11:49 main_usb1
-rw-r--r-- 1 aseema aseema  2751 Nov 21 11:52 main_usb1.c
lrwxrwxrwx 1 aseema aseema    14 Nov 21 10:38 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ ./main
tx: -25
txRet: 4
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ ../1.simple_at/main
tx: 0
rx: [zigbee_wvsnp0]
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ ../1.simple_at/main_usb1
tx: 0
rx: [zigbee_wvsnp1]
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ ./main
rx: [zigbee_wvsnp1]
tx: 0
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$ ./main_usb1
tx: 0
rx: [zigbee_wvsnp0]
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/2.remote_at$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Note that the Coordinator “zigbee_wvsnp0” was able to query the NI of the remote router module “zigbee_wvsnp1” which are configured as above.

III. Running XBeeZB “X.simple_io_configure_and_request” example.

Note that the Coordinator “zigbee_wvsnp0” communicates with the remote router module “zigbee_wvsnp1” which are configured as above.

This example proves that you can configure a remote module on the fly, set, receive real IO data (i.e. pin status) from a remote module’s digital/analog inputs or outputs. This is a more sophisticated example that wraps up many of the examples discussed before into one. Read it carefully to understand it. We encourage you to compile this example statically to learn more about the power of libxbee and Linux beyond examples.



To do so, I configure the remote module (router module “zigbee_wvsnp1”) via Remote AT commands to the following:

o As usual I changed the hard coded the address of the remote IO module
  in main.c to zigbee_wvsnp1’s one.
o I/O Setting D0 to ADC (2) (to read analog input), ***pin #20 of the XBeeZB module.***
o I/O Setting D1 to ADC (2) (to set analog input), ***pin #19 of the XBeeZB module.***
o I/O Setting D3 to DI (3) (to read digital input), ***pin #17 of the XBeeZB module.***
o IR Setting to $7D0 (2000 decimal to sample once every two seconds)
o I/O Setting D4 to DO (4) (to set digital output LOW), ***pin #11 of the XBeeZB module.***
OR
o I/O Setting D4 to DO (5) (to set digital output HIGH), ***pin #11 of the XBeeZB***

o I/O Read & Set RP (28 default) Sets the time length for the RSSI output, ***pin #6 of XBeeZB.***

BELOW ARE NOT designed to be changed for analog in xbeeZB, so we just read them.

o I/O Read P0 (1) Configured PWM0 to RSSI (analog output), ***pin #6 of XBeeZB.***
o I/O Read P1 (0 default) Disabled PWM1 (analog output), ***pin #7 of XBeeZB.***

If the program is ran with:

“$ ./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32 configure”
Then write the parameters to the module and change some of the default parameters above. To

do this, we need to first configure the remote module’s IO parameters/settings.
We first check what each is set to, change them and then read them out to verify.

The static library and main.c are attached here. Note this example only sets D4 Digital output, though it tries to read ALL analog & Digital IO.

Read more about IO setting here:
http://www.digi.com/support/kbase/kbaseresultdetl?id=3221 and
http://www.ladyada.net/make/xbee/ref.html and page 16 of
http://www.makershed.com/v/vspfiles/assets/images/122-32450-xbeetutorial-v1.0.1.pdf
And your module’s product user manual of course.

For this test, the remote address was hard coded to the one configured as above.

To compile this program statically independent of the rest of example, download it from here, then: (If you use ubuntu 13+, my library binary should also work for you without need to compile the library yourself like at the beginning of this tutorial.)

Run it without parameters to see its usage:

```
+++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++++
:~/Desktop/testing_static$ ls -l /usr/lib/libx*
lrwxrwxrwx 1 root root      19 May 28  2013 /usr/lib/libxapian.so.22 -> libxapian.so.22.6.2
-rw-r--r-- 1 root root 2055640 May 28  2013 /usr/lib/libxapian.so.22.6.2
lrwxrwxrwx 1 root root      16 Nov 11 16:58 /usr/lib/libxbee.a -> libxbee.a.3.0.10
-rwxr-xr-x 1 root root  999456 Nov 21 10:14 /usr/lib/libxbee.a.3.0.10
lrwxrwxrwx 1 root root      17 Nov 11 16:58 /usr/lib/libxbeep.a -> libxbeep.a.3.0.10
-rwxr-xr-x 1 root root  494612 Nov 21 10:14 /usr/lib/libxbeep.a.3.0.10
lrwxrwxrwx 1 root root      18 Nov 11 16:58 /usr/lib/libxbeep.so -> libxbeep.so.3.0.10
-rwxr-xr-x 1 root root  140309 Nov 21 10:14 /usr/lib/libxbeep.so.3.0.10
-rwxr-xr-x 1 root root  190386 Nov 21 10:14 /usr/lib/libxbeep.so.3.0.10.dbg
lrwxrwxrwx 1 root root      17 Nov 11 16:58 /usr/lib/libxbee.so -> libxbee.so.3.0.10
-rwxr-xr-x 1 root root  182997 Nov 21 10:14 /usr/lib/libxbee.so.3.0.10
-rwxr-xr-x 1 root root  328457 Nov 21 10:14 /usr/lib/libxbee.so.3.0.10.dbg
lrwxrwxrwx 1 root root      21 Nov 28  2012 /usr/lib/libxklavier.so.16 -> libxklavier.so.16.2.0
-rw-r--r-- 1 root root  107264 Oct  5  2012 /usr/lib/libxklavier.so.16.2.0
:~/Desktop/testing_static$ cp /usr/lib/libxbee.a .
:~/Desktop/testing_static$ ls -l
total 1000
-rwxr-xr-x 1 aseema aseema 999456 Dec  3 02:58 libxbee.a
-rw-r--r-- 1 aseema aseema  19591 Dec  3 02:53 main.c
:~/Desktop/testing_static$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Notice above that I copied it from where the static library was installed. Now that it is definitely located in the same place, I can easily compile it. Again, if you have a libxbee.a binary compiled for your system, that is all you need to use libxbee!!!

```
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++
:~/Desktop/testing_static$ gcc -o my_app main.c -lxbee
:~/Desktop/testing_static$ ls -l
total 1020
-rwxr-xr-x 1 aseema aseema 999456 Dec  3 02:58 libxbee.a
-rw-r--r-- 1 aseema aseema  19591 Dec  3 02:53 main.c
-rwxr-xr-x 1 aseema aseema  17839 Dec  3 03:00 my_app
:~/Desktop/testing_static$ gcc -o my_app main.c -lxbee
:~/Desktop/testing_static$ ./my_app
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Check how to use by not passing parameters:

```
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++
:~/Desktop/testing_static$ ./my_app
Wrong USAGE? Try:
./my_app port baud address64 [configure | on]
e.g.:
./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32 configure
OR
./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32 on
OR
./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32
OR
./my_app /dev/ttyUSB0 9600 0x0013A200408BAE72
OR
./my_app /dev/ttyUSB1 112500 0x0013A20040081826
WHERE:
  port          : /dev/serial_id
  baudrate      : 9600, 19200, 38400, 57600, or 115200
  address64     : 16 digit string => 64-bit address.
  [configure|on]: [Existing AT parameters will be changed as
                  described in the document. | D4 will be
                  turned ON. Otheriwse D4 will be turned OFF.
:~/Desktop/testing_static$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Note that when we run it without “configure” and turning ON D4 we read out default, RP, IR, P0, etc. And of course the request IO at the end of the code reads out the turned on D4 correctly.

```
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/Desktop/testing_static$ ./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32 on
String address parameter given=[0x0013A200408BAE32]
The hex interger conversion=[13a200408bae32]
Depending on your Endianness this will be:
[0][13][a2][0][40][8b][ae][32]
Will wait 0.1 second max for NI response.
rx[zigbee_wvsnp1], length[13].
tx[0].
Will wait 0.1 second max for IR response.
rx[0,0], length[2].
tx[0].
Will wait 0.1 second max for P0 response.
rx[1], length[1].
tx[0].
Will wait 0.1 second max for P1 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for RP response.
rx[28], length[1].
tx[0].
Will wait 0.1 second max for D0 response.
rx[1], length[1].
tx[0].
Will wait 0.1 second max for D1 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for D3 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for D4 HIGH response.
tx[0].
Will wait 0.1 second max for D4 response.
rx[5], length[1].
tx[0].
Will wait 1 second max for IS response.
D4: 1
:~/Desktop/testing_static$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

And if we pass nothing beyond address then D4 will be turned OFF.

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/Desktop/testing_static$ ./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32
String address parameter given=[0x0013A200408BAE32]
The hex interger conversion=[13a200408bae32]
Depending on your Endianness this will be:
[0][13][a2][0][40][8b][ae][32]
Will wait 0.1 second max for NI response.
rx[zigbee_wvsnp1], length[13].
tx[0].
Will wait 0.1 second max for IR response.
rx[0,0], length[2].
tx[0].
Will wait 0.1 second max for P0 response.
rx[1], length[1].
tx[0].
Will wait 0.1 second max for P1 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for RP response.
rx[28], length[1].
tx[0].
Will wait 0.1 second max for D0 response.
rx[1], length[1].
tx[0].
Will wait 0.1 second max for D1 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for D3 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for D4 LOW response.
tx[0].
Will wait 0.1 second max for D4 response.
rx[4], length[1].
tx[0].
Will wait 1 second max for IS response.
D4: 0
:~/Desktop/testing_static$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

As, promised, if you pass "configure", then note we read the default values first then change them, e.g. set the remote module's IR to also send us IO values automatically every 2 seconds

(0x07D0 = 2000ms). We also set Analog RSSI output signal to be maintained for 2 seconds (RP=0x14) instead of the default 4 seconds (RP=0x28), and D4 to be LOW (0).

```
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++
:~/Desktop/testing_static$ ./my_app /dev/ttyUSB0 57600 0x0013A200408BAE32 configure
String address parameter given=[0x0013A200408BAE32]
The hex interger conversion=[13a200408bae32]
Depending on your Endianness this will be:
[0][13][a2][0][40][8b][ae][32]
Will wait 0.1 second max for IR 2000 response.
Packet too short/empty!!!
tx[0].
Will wait 0.1 second max for RP response.
rx[28], length[1].
tx[0].
Will wait 0.1 second max for RP response.
tx[0].
Will wait 0.1 second max for AC response.
tx[0].
Will wait 0.1 second max for NI response.
rx[zigbee_wvsnp1], length[13].
tx[0].
Will wait 0.1 second max for IR response.
rx[7,d0], length[2].
tx[0].
Will wait 0.1 second max for P0 response.
rx[1], length[1].
tx[0].
Will wait 0.1 second max for P1 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for RP response.
rx[14], length[1].
tx[0].
Will wait 0.1 second max for D0 response.
rx[1], length[1].
tx[0].
Will wait 0.1 second max for D1 response.
rx[0], length[1].
tx[0].
Will wait 0.1 second max for D3 response.
rx[0], length[1].
tx[0].
```

```
Will wait 0.1 second max for D4 LOW response.
tx[0].
Will wait 0.1 second max for D4 response.
rx[4], length[1].
tx[0].
Will wait 1 second max for IS response.
D4: 0
:~/Desktop/testing_static$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Now, since we set IR above to make the remote module keep spewing out IO data values. I can go and launch another program that just prints out whatever it gets from from the remote module. That happens to be "Running XBee1 "4.simple_io" example:" we played with before. After changing xbee1 to xbeeZB I can run it as below to capture the data that is being sent every 2 seconds!

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/4.simple_io$ ./main
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
D4: 0
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

You can set more IOs above as inputs and you will read out more stuff with simple IO or this program.

That should be enough material to get you going!!!

IV. Running XBeeZB “9.node_detect” example.

For this one I copied the “sample/**xbee1**/9.node_detect” example to “sample/**xbeeZB**/9.node_detect”. Then just changed the code below to xbeeZB.

```
… snip ...
  if ((ret = xbee_setup(&xbee, "xbeeZB", "/dev/ttyUSB0", 57600)) != XBEE_ENONE) {
      printf("ret: %d (%s)\n", ret, xbee_errorToStr(ret));
      return ret;
  }
… snip …
```

And wallah! It works!

```
++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee -lpthread
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$ ls -l
total 28
-rwxr-xr-x 1 aseema aseema 17117 Nov 21 12:57 main
-rw-r--r-- 1 aseema aseema  3214 Nov 21 12:57 main.c
-rw-r--r-- 1 aseema aseema  3213 Nov 21 11:07 main.c~
lrwxrwxrwx 1 aseema aseema    22 Nov 21 11:07 makefile -> ../../makefile.pthread
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$ ./main
ND Sent!... waiting for completion
Node: igbee_wvsnp1         0x7809  0x0013A200 0x408BAE32
Timeout while waiting for ND command to complete...
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

V. Running XBeeZB “13.lxb2lxb_server” and “13.lxb2lxb_client” example.

Well not so fast! I set the Coordinator as Server and Router as Client from the examples copied from xbee1 “13.lxb2lxb_server” and “13.lxb2lxb_client” examples above and compiled as example above. But trying to get them to talk had a glitch. Well, as the code from “sample/xbeeZB/3.simple_data/main.c” shows, you have to change the mode to Zigbee’s “Data”.

```
… snip …
```

```
if ((ret = xbee_conNew(xbee, &con, "64-bit Data", &address)) != XBEE_ENONE) {
    xbee_log(xbee, -1, "xbee_conNew() returned: %d (%s)", ret, xbee_errorToStr(ret));
    return ret;
}

...snip …
```

to

```
...snip …

if ((ret = xbee_conNew(xbee, &con, "Data", &address)) != XBEE_ENONE) {
    xbee_log(xbee, -1, "xbee_conNew() returned: %d (%s)", ret, xbee_errorToStr(ret));
    return ret;
}

...snip …
```

Run the server first in a separate window, then the client. And it works!

```
+++++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$ cd ../13.lxb2lxb_server/
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ ls -l
total 28
-rwxr-xr-x 1 aseema aseema 16630 Nov 21 13:17 main
-rw-r--r-- 1 aseema aseema  2476 Nov 21 13:14 main.c
-rw-r--r-- 1 aseema aseema  2423 Nov 21 11:08 main.c~
lrwxrwxrwx 1 aseema aseema    14 Nov 21 11:08 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ cd ../13.lxb2lxb_client/
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$ ls -l
total 28
-rwxr-xr-x 1 aseema aseema 16491 Nov 21 13:18 main
-rw-r--r-- 1 aseema aseema  2409 Nov 21 13:16 main.c
-rw-r--r-- 1 aseema aseema  2346 Nov 21 11:08 main.c~
lrwxrwxrwx 1 aseema aseema    14 Nov 21 11:08 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$ cd ..
```

```
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$ 13.lxb2lxb_server/main
DEV: -1#[main.c:61] main() 0x1db11f0: xbee_conNew() returned: -21 (The requested item does
not exist)
```

FIX THE GLITCH! As discussed above. Both in ”sample/xbeeZB/13.lxb2lxb_server/main.c”
and
”sample/xbeeZB/13.lxb2lxb_client/main.c”.

```
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$ cd 13.lxb2lxb_server/
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ ls -l
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ e main.c
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$ ./main
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
rx: [Hi!]
^C
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_server$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

On the client side run:

```
+++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$ cd 13.lxb2lxb_client/
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$ ls -l
total 28
-rwxr-xr-x 1 aseema aseema 16491 Nov 21 13:18 main
-rw-r--r-- 1 aseema aseema  2402 Nov 21 13:39 main.c
-rw-r--r-- 1 aseema aseema  2409 Nov 21 13:16 main.c~
lrwxrwxrwx 1 aseema aseema    14 Nov 21 11:08 makefile -> ../../makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$ make
gcc main.c -g -o main -I ../../.. -L ../../../lib -lxbee
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$ ./main
rx: [Hello?]
rx: [Hello?]
```

```
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
rx: [Hello?]
^C
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/13.lxb2lxb_client$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

So, here is my xbeeZB sample list:

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$ ls -l ../
total 24
drwxr-xr-x 2 aseema aseema 4096 Nov 21 11:08 13.lxb2lxb_client
drwxr-xr-x 2 aseema aseema 4096 Nov 21 11:08 13.lxb2lxb_server
drwxr-xr-x 2 aseema aseema 4096 Nov 21 11:17 1.simple_at
drwxr-xr-x 2 aseema aseema 4096 Nov 21 11:54 2.remote_at
drwxr-xr-x 2 aseema aseema 4096 Nov 21 10:42 3.simple_data
drwxr-xr-x 2 aseema aseema 4096 Nov 21 12:57 9.node_detect
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/9.node_detect$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

VI. Running the XBeeZB “10.c++” example:

This is supposed to showcase C++ interface to the library. As others above, I copied the “samples/xbee1/10.c++” directory into “samples/xbeeZB/10.c++”. the modified the hard coded mode in the code and the ttyUSB# for Coordinator and Router respectively.

a-- The example prints out all modules/modes the library supports.
b-- Then prints the mode of the module it is currently running under.
(see bug comments in the actual modules tested in XBeeZB section).
c-- Then prints available connection types for the above xbee module.
d-- Then prints “Callback!!” to make sure callbacks work.
e-- Then it requests the local name of the running module (NI).

Running as the Coordinator on tttyUSB0, with UN-commented “#define LOCAL_CONNECTION” and UN-commented “#define USE_CALLBACKS”.

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$ cp -R ../../xbee1/10.c++ ../../xbeeZB/
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$ ls -l ../../xbeeZB/
total 40
drwxr-xr-x 2 aseema aseema 4096 Nov 29 12:59 10.c++
drwxr-xr-x 2 aseema aseema 4096 Nov 21 13:40 13.lxb2lxb_client
drwxr-xr-x 2 aseema aseema 4096 Nov 21 13:40 13.lxb2lxb_server
drwxr-xr-x 2 aseema aseema 4096 Nov 21 19:09 14.request_io
drwxr-xr-x 2 aseema aseema 4096 Nov 21 18:58 1.simple_at
drwxr-xr-x 2 aseema aseema 4096 Nov 21 13:15 2.remote_at
drwxr-xr-x 2 aseema aseema 4096 Nov 21 22:51 3.simple_data
drwxr-xr-x 2 aseema aseema 4096 Nov 21 19:15 4.simple_io
drwxr-xr-x 2 aseema aseema 4096 Nov 21 19:14 5.get_conTypes
drwxr-xr-x 2 aseema aseema 4096 Nov 21 12:57 9.node_detect
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$ cd ../../xbeeZB/
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$ ls
10.c++  13.lxb2lxb_client  13.lxb2lxb_server  14.request_io  1.simple_at  2.remote_at
3.simple_data  4.simple_io  5.get_conTypes  9.node_detect
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB$ cd 10.c++/
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ls
main  main.cpp  main.cpp~  makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ make
g++ main.cpp -g -o main -I ../../.. -L ../../../lib -lxbeep
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ./main
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbeeZB'
Available connection types:
  Transmit Status  Modem Status  Local AT  Remote AT  Data  Data (explicit)  I/O  Sensor
Identify  OTA Update Status
Callback!!
zigbee_wvsnp0
Testing, 1... 2... 3…
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Running as the Router on tttyUSB1, with UN-commented “#define LOCAL_CONNECTION” and UN-commented “#define USE_CALLBACKS”.

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ mv main main_usb0
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ls
main.cpp  main.cpp~  main_usb0  makefile
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ make
g++ main.cpp -g -o main -I ../../.. -L ../../../lib -lxbeep
```

```
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ./main
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbeeZB'
Available connection types:
  Transmit Status  Modem Status  Local AT  Remote AT  Data  Data (explicit)  I/O  Sensor
Identify  OTA Update Status
Callback!!
zigbee_wvsnp1
Testing, 1... 2... 3...
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

IF you comment out "#define LOCAL_CONNECTION" this will be the name of the remote module. You need to modify the hard coded address in code to the remote modules' SH and SL address.

Running as the Coordinator (zigbee_wvsnp0) on tttyUSB0, with commented-OUT "#define LOCAL_CONNECTION" and UN-commented "#define USE_CALLBACKS".

```
+++++++++++++++++++++++GOOD OUTPUT++++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ make
g++ main.cpp -g -o main -I ../../.. -L ../../../lib -lxbeep
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ mv main main_usb0
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ./main_usb0
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbeeZB'
Available connection types:
  Transmit Status  Modem Status  Local AT  Remote AT  Data  Data (explicit)  I/O  Sensor
Identify  OTA Update Status
Callback!!
zigbee_wvsnp1
Testing, 1... 2... 3...
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

f-- If you comment out: "#define USE_CALLBACKS" then you are not using callbacks functionality. You won't see "d--" above nor "Testing, 1… 2… 3…" here. You will see only stuff up to "c--" above, then "Packet length: #" and the print out of each character of the packet which is the remote/local name (NI) you just requested.
Packet length # is the number of characters in the name. Each character is printed on a separate line preceded by its index in the "packet".

Running as the Coordinator (zigbee_wvsnp0) on tttyUSB0, with commented-OUT "#define LOCAL_CONNECTION" and commented-OUT "#define USE_CALLBACKS".

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ./main_usb0
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbeeZB'
Available connection types:
  Transmit Status  Modem Status  Local AT  Remote AT  Data  Data (explicit)  I/O  Sensor
Identify  OTA Update Status
Packet length: 13
  0 z
  1 i
  2 g
  3 b
  4 e
  5 e
  6 _
  7 w
  8 v
  9 s
  10 n
  11 p
  12 1
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Running as the Router (zigbee_wvsnp1) on tttyUSB1, with commented-OUT "#define LOCAL_CONNECTION" and commented-OUT "#define USE_CALLBACKS".

```
++++++++++++++++++++++++GOOD OUTPUT+++++++++++++++++++++++++
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ make
g++ main.cpp -g -o main -I ../../.. -L ../../../lib -lxbeep
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$ ./main
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbeeZB'
Available connection types:
  Transmit Status  Modem Status  Local AT  Remote AT  Data  Data (explicit)  I/O  Sensor
Identify  OTA Update Status
Packet length: 13
```

```
  0 z
  1 i
  2 g
  3 b
  4 e
  5 e
  6 _
  7 w
  8 v
  9 s
  10 n
  11 p
  12 0
:~/mygits/my_libxbee/libxbee3/sample/xbeeZB/10.c++$
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

N.B. There is a subtle bug in the example code.
I compiled the original “sample/xbee1/10.c++” code and ran it with xbeeZB modules NOT xbee1 modules.

```
… snip …

   libxbee::XBee xbee("xbee1", "/dev/ttyUSB0", 57600);

… snip …
```

Surprisingly it ran OK and produced the output below.

```
----------------------------------------BAD OUTPUT----------------------------------------
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$ ./main
Available libxbee modes:
  xbee1  xbee2  xbee3  xbee5  xbee6b  xbeeZB  net  debug
Running libxbee in mode 'xbee1'
Available connection types:
  Modem Status  Transmit Status  Local AT  Remote AT  16-bit Data  64-bit Data  16-bit I/O  64-bit I/O
Callback!!
zigbee_wvsnp0
Testing, 1... 2... 3...
:~/mygits/my_libxbee/libxbee3/sample/xbee1/10.c++$
------------------------------------------------------------------------------------------------
```

The “BAD OUTPUT” above was running as the Coordinator (zigbee_wvsnp0) on tttyUSB0, with commented-OUT “#define LOCAL_CONNECTION” and commented-OUT “#define

USE_CALLBACKS”.

Note that example prints out that it is in “xbee1” mode. And it prints out xbee1 connection types. But the actual modules is xbeeZB. Which means the library is not getting this info from the real module itself. It is just using its own code. The only real info from the real module is the NI response!

Next WiFi!

------------------------END OF DOCUMENT FOR NOW-------------------------------------