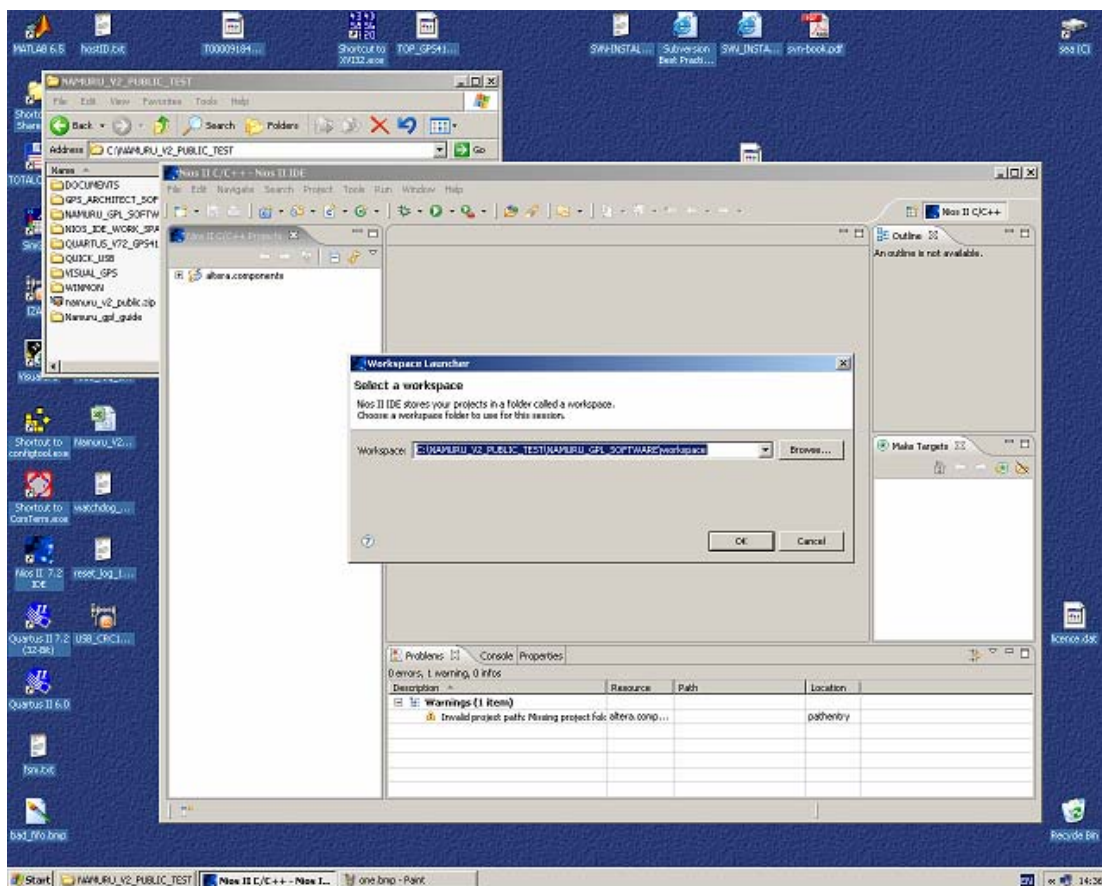


To get started with the Namuru-GPL software, follow these steps.

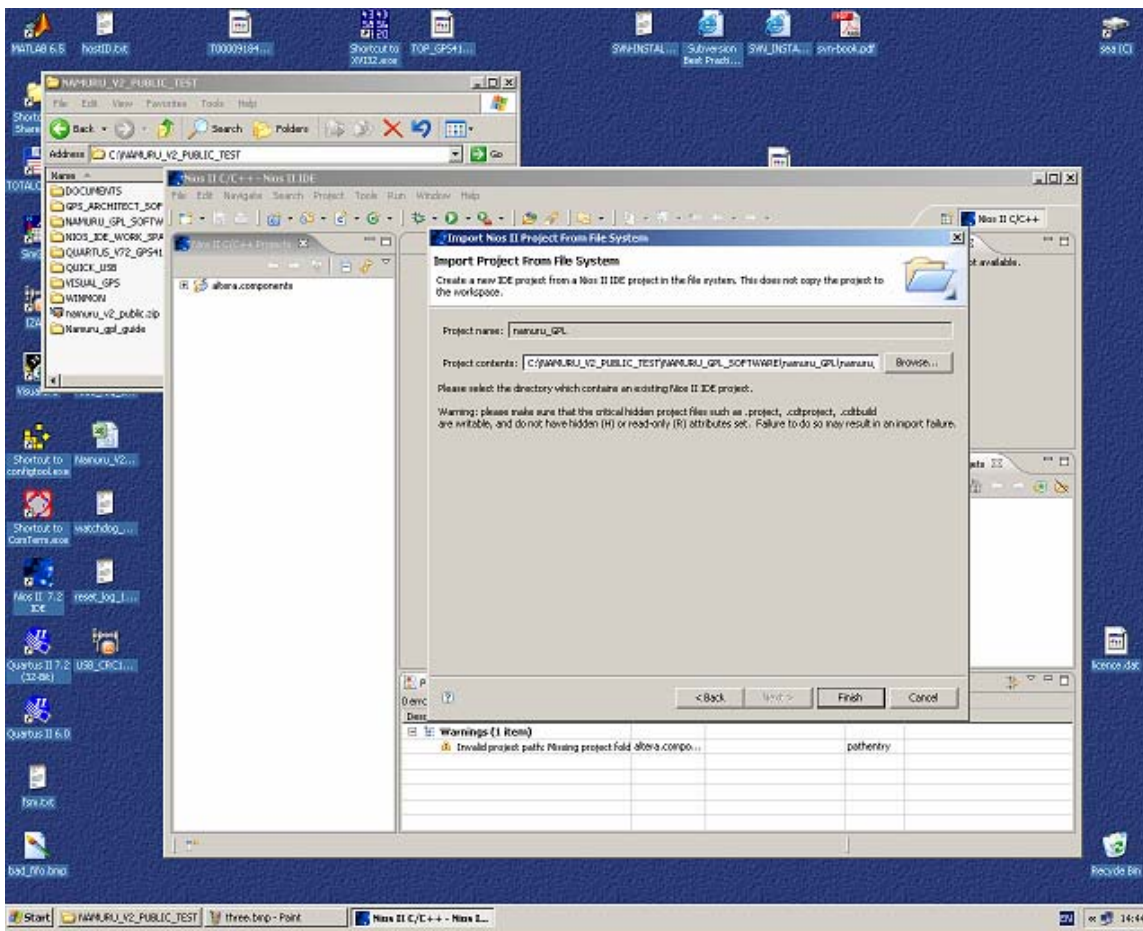
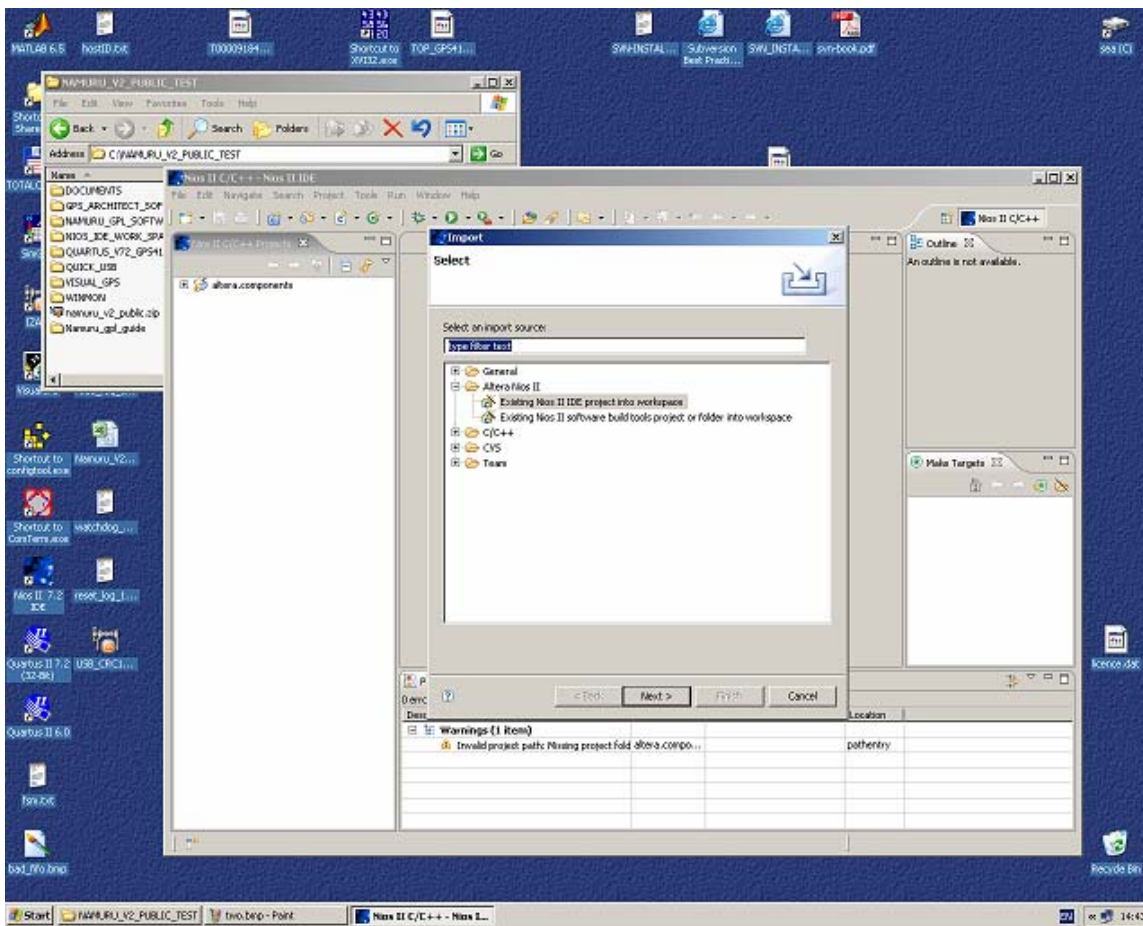
The default directory structure from the top level looks something like:

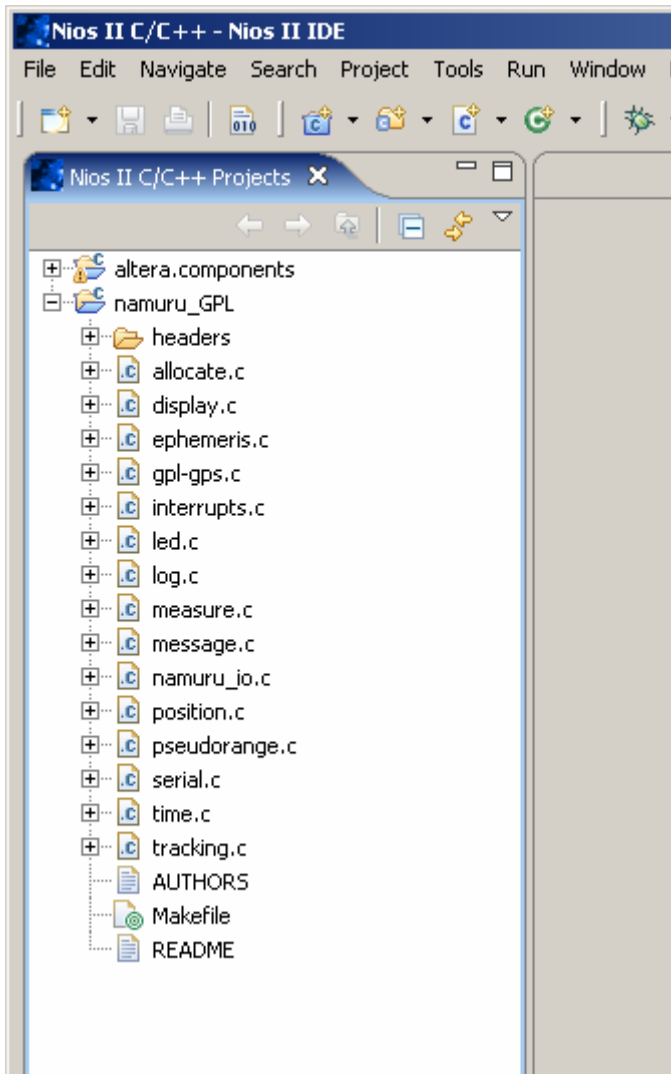
- DOCUMENTS
- GPS_ARCHITECT_SOFTWARE
- NAMURU_GPL_SOFTWARE
- NIOS_IDE_WORK_SPACE
- QUARTUS_V72_GPS410_V2
- QUICK_USB
- VISUAL_GPS
- WINMON

Open the NiosII IDE and choose a workspace. A new workspace could be chosen, or the workspace in the default directory layout could be used.



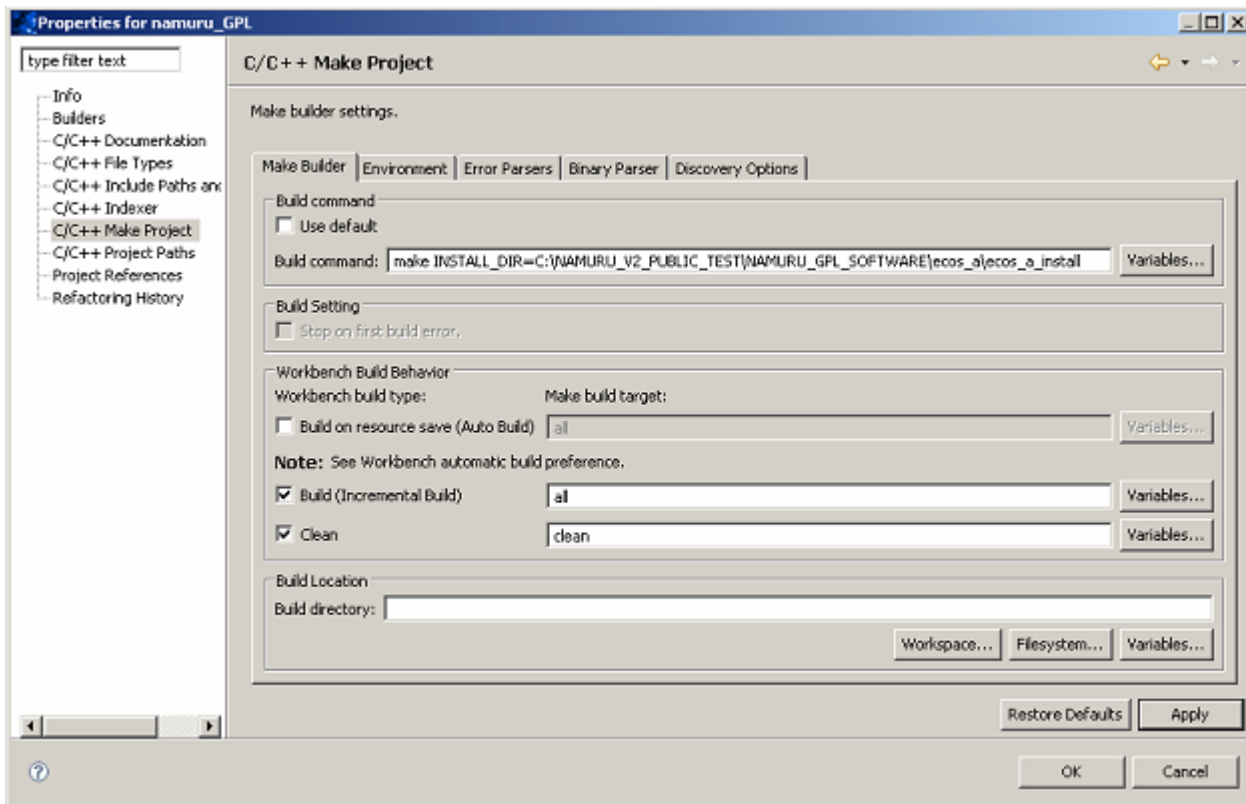
Next, import an existing project into the NiosII IDE. The project is in the NAMURU_GPL_SOFTWARE directory. Two steps are required here.



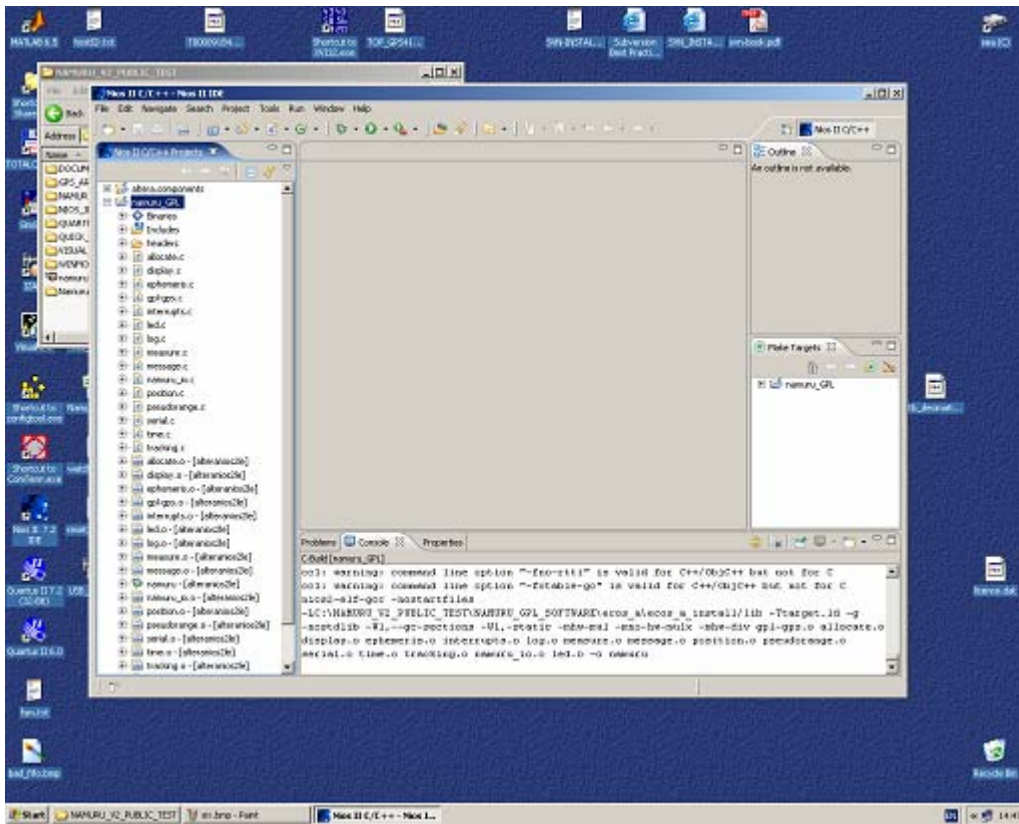


Here we see the source files of the Namuru-GPL software. The header files are in the "headers" directory to keep them organized. Note the Makefile. Unlike a 'normal' NiosII project that uses the Altera built library, the Makefile must be manually managed.

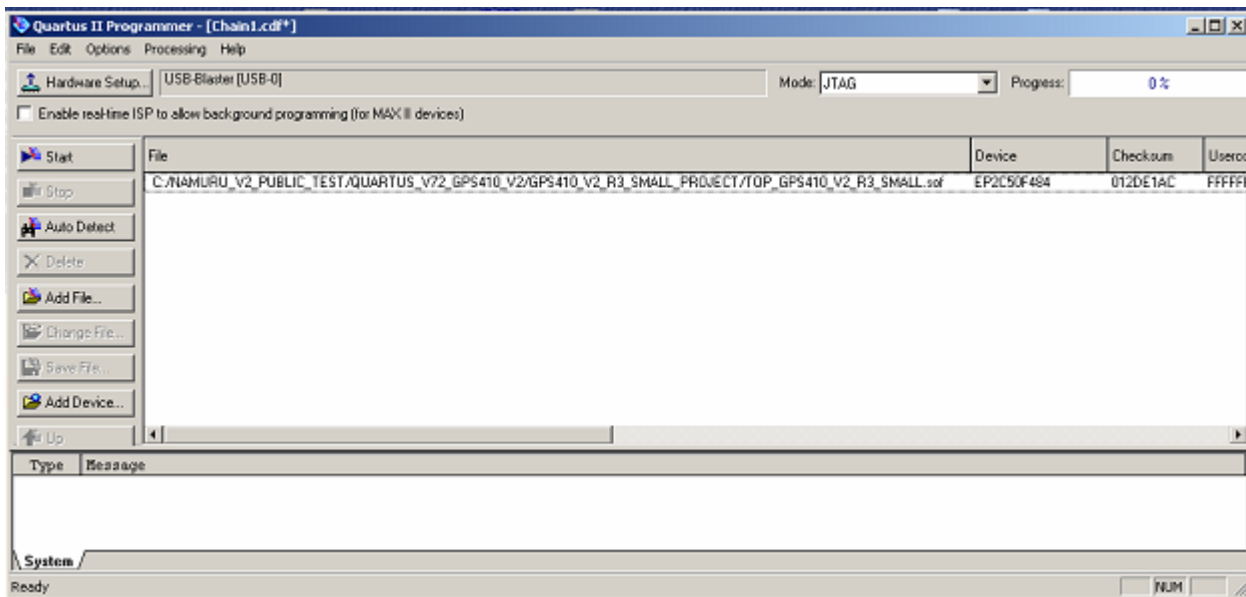
Next, the project properties must be setup so that the eCos library can be linked in. Highlight the namuru_GPL project and select properties. Find the "C/C++ Make Project" property. Edit the "Build command" field shown below to point to the eCos library. The eCos library is built around a NiosII processor design – the eCos library and the Quartus FPGA project must match. In this case we are using the small project and the precompiled eCos library that goes with it.



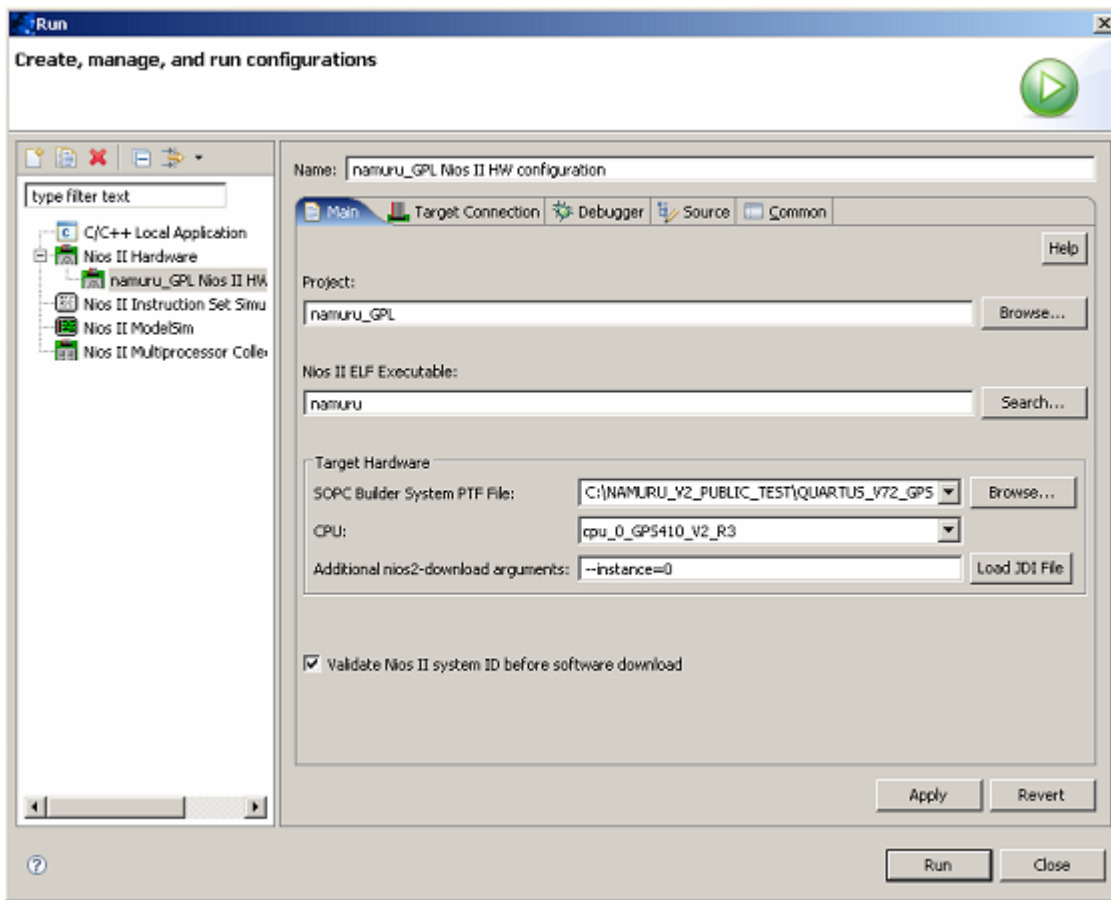
Next, compile the project, and if everything works out OK, you should see the object files, as can be seen (with a magnifying glass) below.



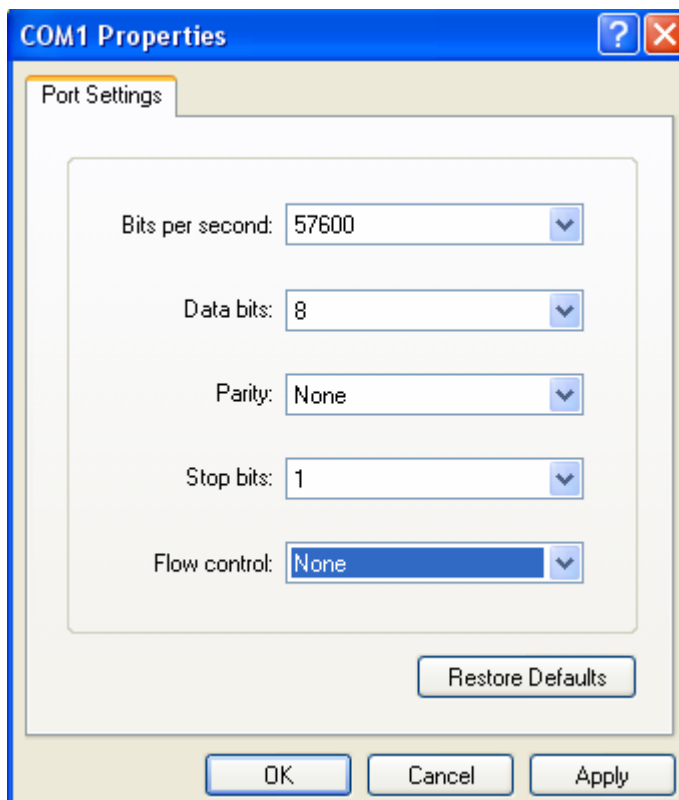
Next, load the FPGA logic image to the Namuru board. In this case the small reference project is used. Use the Quartus programmer, either from Quartus, or from the tools menu in the NiosII IDE.



Now, load the software. To do this, create a run configuration by selecting 'run' in the NiosII IDE and create a new NiosII Hardware run configuration as shown below.

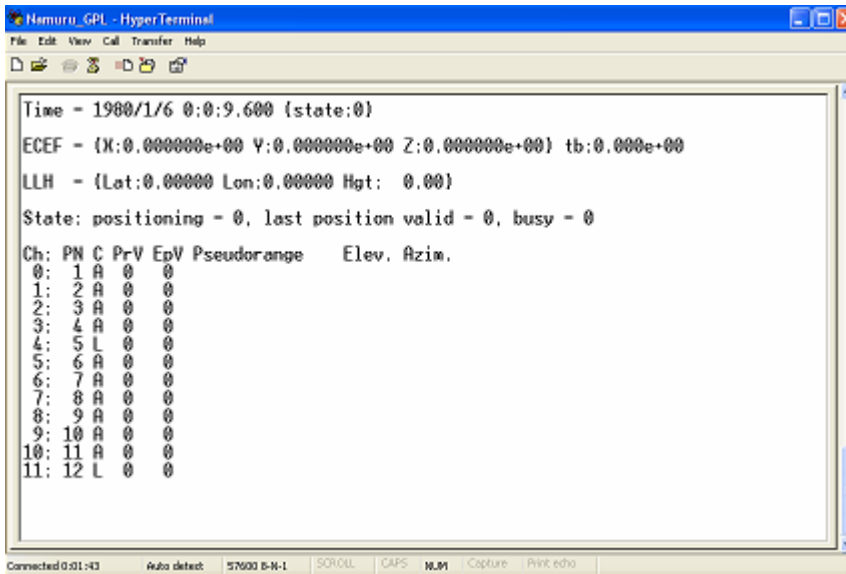


Now, if everything's working fine, you probably want to see what's going on in the receiver. So, connect an antenna, connect a serial cable between the Namuru's J3 and a computer's serial port and fire up Hyperterminal (Windows). Set up Hyperterminal as shown below.



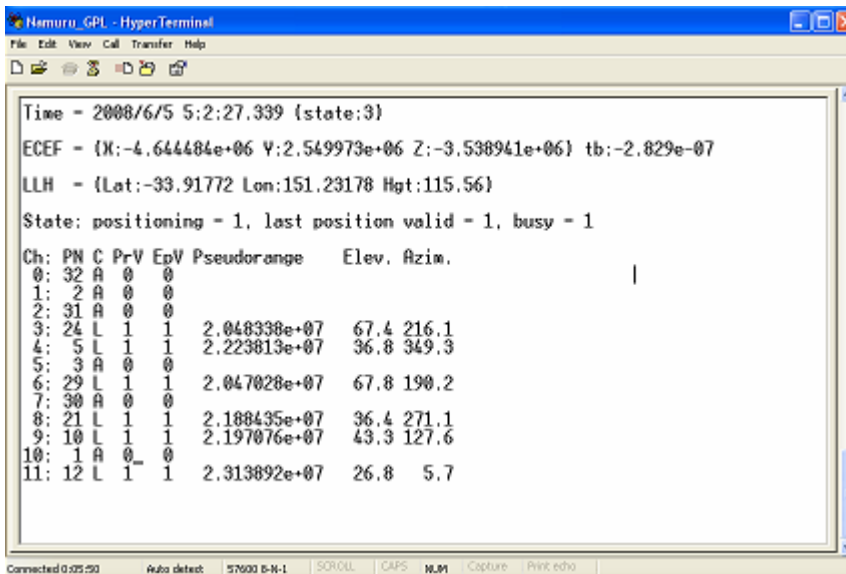
There are a number of screens that can be chosen to view. Several of these are shown in the following screenshots.

In this first shot, we see the receiver at startup, No satellites are being tracked yet, and there is no position solution. If this situation continues for more than a few minutes, something is wrong! Check the antenna.



```
Time - 1980/1/6 0:0:9.600 {state:0}
ECEF - (X:0.000000e+00 Y:0.000000e+00 Z:0.000000e+00) tb:0.000e+00
LLH - (Lat:0.00000 Lon:0.00000 Hgt: 0.00)
State: positioning = 0, last position valid = 0, busy = 0
Ch: PN C PrV EpV Pseudorange Elev. Azim.
0: 1 A 0 0
1: 2 A 0 0
2: 3 A 0 0
3: 4 A 0 0
4: 5 L 0 0
5: 6 A 0 0
6: 7 A 0 0
7: 8 A 0 0
8: 9 A 0 0
9: 10 A 0 0
10: 11 A 0 0
11: 12 L 0 0
```

In the screenshot below, we see that six satellites are being tracked and a position and time solution is provided.



```
Time - 2008/6/5 5:2:27.339 {state:3}
ECEF - (X:-4.644484e+06 Y:2.549973e+06 Z:-3.538941e+06) tb:-2.829e-07
LLH - (Lat:-33.91772 Lon:151.23178 Hgt:115.56)
State: positioning = 1, last position valid = 1, busy = 1
Ch: PN C PrV EpV Pseudorange Elev. Azim.
0: 32 A 0 0
1: 2 A 0 0
2: 31 A 0 0
3: 24 L 1 1 2.048338e+07 67.4 216.1
4: 5 L 1 1 2.223813e+07 36.8 349.3
5: 3 A 0 0
6: 29 L 1 1 2.047028e+07 67.8 190.2
7: 30 A 0 0
8: 21 L 1 1 2.188435e+07 36.4 271.1
9: 10 L 1 1 2.197076e+07 43.3 127.6
10: 1 A 0 0
11: 12 L 1 1 2.313892e+07 26.8 5.7
```

Shown below is another view on the receiver activities. Here we can see tracking information for each channel.

```

Hamuru_GPL - HyperTerminal
File Edit View Call Transfer Help
[Icons]

Ch: PN Bin   CarNCO  Iprmt  Oprmt  RSSI0  State  Avg
0: 32 15   -75159 -274   -253   400  A(--)  377
1: 2  15   -75159 252   -247   375  A(--)  429
2: 31 17   -85896 28     133    147  A(--)  299
3: 24 15   -79544 392   -2800  3299  L(B-)  3121
4: 5  1     2637  293   -1807  1666  L(B-)  1938
5: 3  15   -75159 -60    -265   295  A(--)  467
6: 29 15   -104405 786  3042  2729  L(B-)  2782
7: 30 5     -29145 -703  3040  3122  L(B-)  3065
8: 21 23   -129701 117  -1903  2028  L(B-)  2345
9: 10 5     -23916 -45   -928   920  L(B-)  1021
10: 1  16   85896  -98   -125   174  A(--)  311
11: 12 2     20816  129  -1159  1336  L(B-)  1479

Connected 0:06:17  Auto detect  57600 B-N-L  SERIAL  CAPS  NUM  Capture  Print echo

```

In this final screenshot, the pseudoranges are shown.

```

Hamuru_GPL - HyperTerminal
File Edit View Call Transfer Help
[Icons]

CH: PN $ bitX50 eb ems Pseudorange Average
0: 0  A
1: 0  A
2: 0  A
3: 24 L 18191150 23 10  2.048145e+07 3238
4: 5  L 18191150 23 4  2.228055e+07 1882
5: 0  A
6: 29 L 18191150 23 10  2.045558e+07 2737
7: 30 L 18191150 23 8  2.120070e+07 2949
8: 21 L 18191150 23 6  2.185539e+07 2470
9: 10 L 18191150 23 5  2.199924e+07 1103
10: 0  A
11: 12 L 18191150 23 1  2.319200e+07 1435
-

Connected 0:07:05  Auto detect  57600 B-N-L  SERIAL  CAPS  NUM  Capture  Print echo

```

The screens are controlled by entering letters as shown bellow:

- t = DISPLAY_TRACKING
- m = DISPLAY_MESSAGES
- s = DISPLAY_STOP
- e= DISPLAY_EPHEMERIS
- r = DISPLAY_PSEUDORANGE
- p = DISPLAY_POSITION
- d = DISPLAY_DEBUG //probably turned off
- l = DISPLAY_LOG
- c = DISPLAY_CLEAR
- h = DISPLAY_PLOT