**Source-level debugging of ROM images using a Majic (GNU)**

Majic along with the GNU debugger allows for debugging binary images from ROM. To debug your ROM code (set breakpoints, single-step, etc) with source-level symbolic debug information perform the following steps:

**Preparing to debug**

The following steps need to be done prior to starting a debug session. Note that you need to make changes to the .gdbinit and startice.cmd file, when you go back to debugging from RAM you need to undo these changes, so it's a good idea to back up these files before starting.

1. First build and program your image into ROM. NET+OS typically first loads a bootloader (rom.bin) and then a compressed image (image.bin).

2. Edit your .gdbinit file and comment out the line which starts with "monitor L……", you can do this by placing a '#' character in the beginning of the line, as shown below:

#monitor L -no b c:\netos63_branch4\netos\netos\src\examples\naftpapp\32b\image.elf

3. Edit the startice.cmd file, you can locate this file by right mouse-clicking on the mdi-server icon on your desktop and going to properties, this file will be located in the 'Start in:' directory. Comment out the line 'fr c ns9xxx.cmd' this is done by placing a '//' at the beginning of the line. You are commenting out the reading of the initialization script which initializes RAM since the ROM image will now initialize RAM. This line is shown below:

// fr c ns9xxx.cmd

**Debug Session**

1. Disable flash, on the ns9750 dev board this is done by putting a jumper on J1, on the connectcore9c this done by ….
2. Power the board ON.
3. Click on the mdi-server icon
4. Go into the 32b directory for your application, the same directory with your .gdbinit file.
5. To debug the bootloader start gdbtk session by typing:

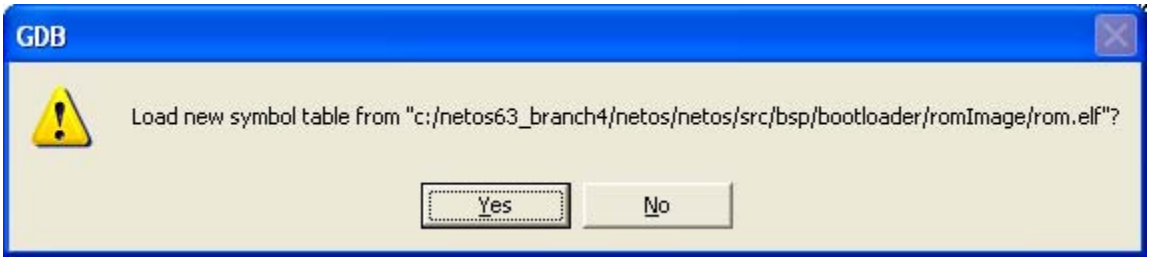
   'gdbtk –se *c:/netos63_gnu/netos/src/bsp/bootloader/romImage*/rom.elf'

   Note that the directory path may be different on your system.

6. Type

'Symbol-file *c:/netos63_gnu/netos/src/bsp/bootloader/romImage*/rom.elf'

7. Click Yes




8. Enable flash by removing the Jumper from J1 (for the ns9750_a)
9. type '$set pc=0x50000000'
10. To start at the very beginning of the ROM image you can type 'si' (step in). Now you can set one breakpoint, ***you can only set one breakpoint at a time and when you hit the breakpoint you must delete it before you set another breakpoint***. To check to see the current breakpoints which are set you can type 'i b' in the gdbtk console window. The rom.elf file is the ROM image portion of the bootloader, if you wish to debug the RAM image portion of the bootloader or the RAM image application you will need to reload symbols. When you set a breakpoint on ROM you will see the following message in the mdi-server window:

'Software breakpoint was promoted to hardware breakpoint'

11. Set a breakpoint on main ('b main') and the type continue ('c')
12. Delete the breakpoint on main
13. Step though main and before you hit blExecuteImage load the symbols for the RAM portion of the booloader by typing:

'Symbol-file *c:/netos63_gnu/netos/src/bsp/bootloader/ramImage*/blram.elf'

14. Click OK
15. Set a breakpoint on NABlMain
16. Hit continue 'c'
17. Delete the breakpoint on NABlMain
18. Step through NABlMain
19. Before hitting blExecuteImage() load the symbols for your application
    'Symbol-file image.elf'
20. Break on applicationStart
21. Continue by typing 'c' and enter
22. Delete the breakpoint on applicationStart
23. You can now debug your application.

**Tips**

There are two common problems when debugging from ROM. The first is that the flash must be disabled and enabled at the correct times. The second is only having one

breakpoint at a time.  Also, if you have optimization turned ON, you can expect to see the program counter ‘jump around’ while single stepping, to disable optimization remove -O2 from Makefile.inc in the root directory of netos.