

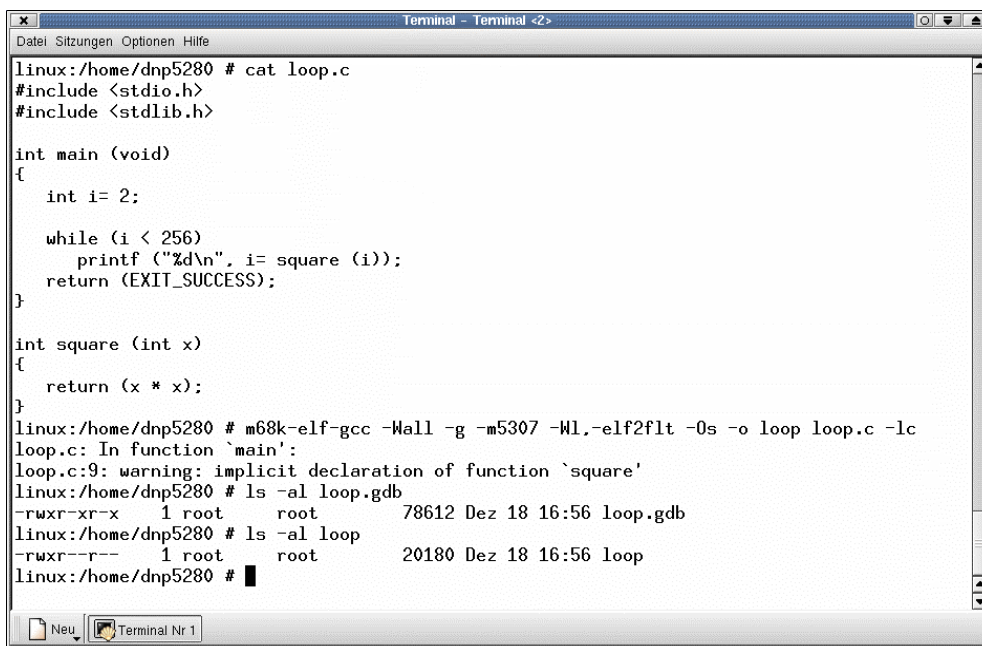
How to use the DDD on the top of the GNU Cross Debugger

The GNU cross tool chain for DNP/5280 Linux C programming offers a pre-build cross version of the GNU Debugger, called `m68k-elf-gdb`. This debugger runs on a PC-based Linux and allows you to debug DNP/5280 uCLinux executables with ELF layout at C source code level over a remote connection to the DNP/5280. The cross debugger needs a Ethernet-based TCP/IP link between the PC and the DNP/5280. In addition the debugger needs also a remote debugging agent, called `gdbserver` for the DNP/5280. This agent is pre-installed within the DNP/5280 Linux.

The GNU debugger offers a simple command line interface and a lot of different commands. With the help of DDD (Data Display Debugger - a graphical front-end for command line debuggers) you get a powerful graphical user interface for the GNU debugger. DDD is a part of many PC Linux distributions. DDD is also available from <http://www.gnu.org/software/ddd/>.

- **1. Step:** Write your C program and translate the C source code with the GNU cross C compiler to a executable and a symbol file. Use the following command line with the `-g` parameter. This sample command line builds a executable, called `loop` from a source code file with the name `loop.c` and a file `loop.gdb` with symbol information.

```
m68k-elf-gcc -Wall -g -m5307 -Wl,-elf2flt -Os -o loop loop.c -lc
```



```

Terminal - Terminal <2>
Datei Sitzungen Optionen Hilfe
linux:/home/dnp5280 # cat loop.c
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    int i= 2;

    while (i < 256)
        printf ("%d\n", i= square (i));
    return (EXIT_SUCCESS);
}

int square (int x)
{
    return (x * x);
}
linux:/home/dnp5280 # m68k-elf-gcc -Wall -g -m5307 -Wl,-elf2flt -Os -o loop loop.c -lc
loop.c: In function `main':
loop.c:9: warning: implicit declaration of function `square'
linux:/home/dnp5280 # ls -al loop.gdb
-rwxr-xr-x  1 root  root   78612 Dez 18 16:56 loop.gdb
linux:/home/dnp5280 # ls -al loop
-rwxr--r--  1 root  root    20180 Dez 18 16:56 loop
linux:/home/dnp5280 # █

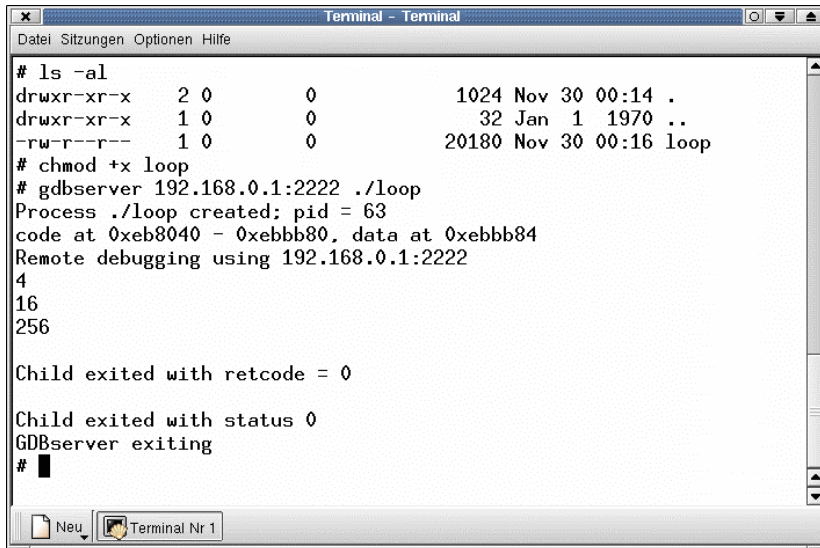
```

- **2. Step:** Transfer the executable from your PC hard disk drive to the DNP/5280 RAM disk or JFFS-based flash disk drive and run the executable on your DNP/5280 with the help of `gdbserver`. Use a TFTP session and a Telnet session for this task. Please enter the following commands within the DNP/5280 Telnet session window:

```
tftp -g -l loop 192.168.0.1
chmod +x loop
```

```
gdbserver 192.168.0.1:2222 ./loop
```

The first command line transfers the executable `loop` from the PC to the DIL/NetPC DNP/5280. This line assumes that your PC is using the IP address 192.168.0.1. The second line makes sure that the executable attribute is set for `hello`. The next command line runs `loop` with the help of `gdbserver`. Within this command line you need the IP address of the PC together with a TCP/IP port number. We use the port number 2222 for this sample.



```

Terminal - Terminal
Datei Sitzungen Optionen Hilfe

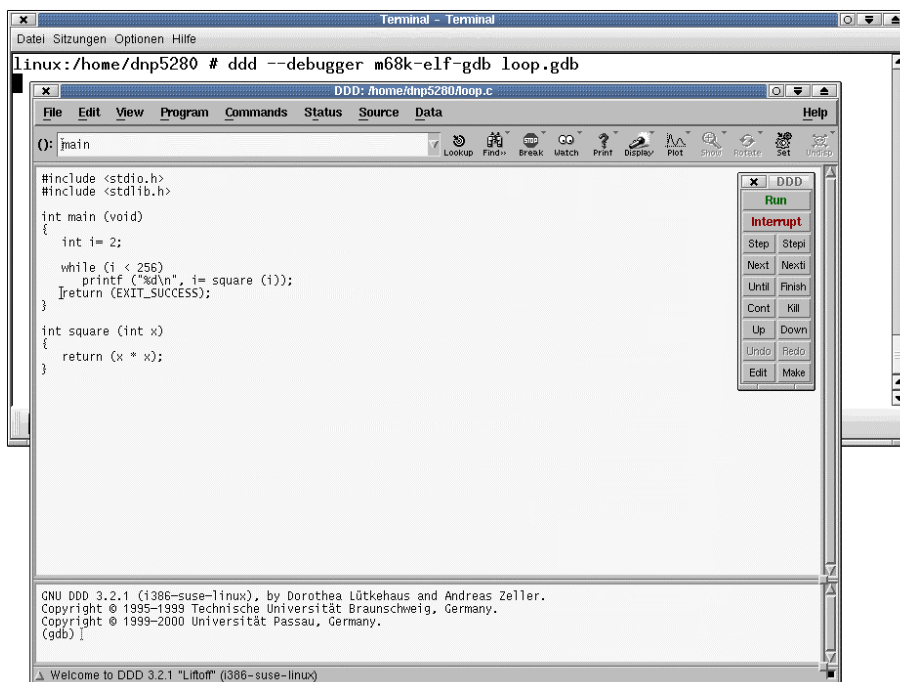
# ls -al
drwxr-xr-x  2 0      0          1024 Nov 30 00:14 .
drwxr-xr-x  1 0      0          32 Jan  1 1970 ..
-rw-r--r--  1 0      0          20180 Nov 30 00:16 loop
# chmod +x loop
# gdbserver 192.168.0.1:2222 ./loop
Process ./loop created; pid = 63
code at 0xeb8040 - 0xebbb80, data at 0xebbb84
Remote debugging using 192.168.0.1:2222
4
16
256

Child exited with retcode = 0

Child exited with status 0
GDBserver exiting
# █
  
```

- **3. Step:** Run the GNU cross Debugger `m68k-elf-gdb` with the help of DDD on your PC. Use the following command line. The parameter `--debugger m68k-elf-gdb` tells DDD the name of the debugger, `loop.gdb` is the file name for the symbol information file.

```
ddd --debugger m68k-elf-gdb loop.gdb
```



```

Terminal - Terminal
Datei Sitzungen Optionen Hilfe

linux:/home/dnp5280 # ddd --debugger m68k-elf-gdb loop.gdb

DDD: /home/dnp5280/loop.c
File Edit View Program Commands Status Source Data Help

(): main
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
  int i = 2;

  while (i < 256)
    printf ("sd\n", i = square (i));
  [return (EXIT_SUCCESS);
}

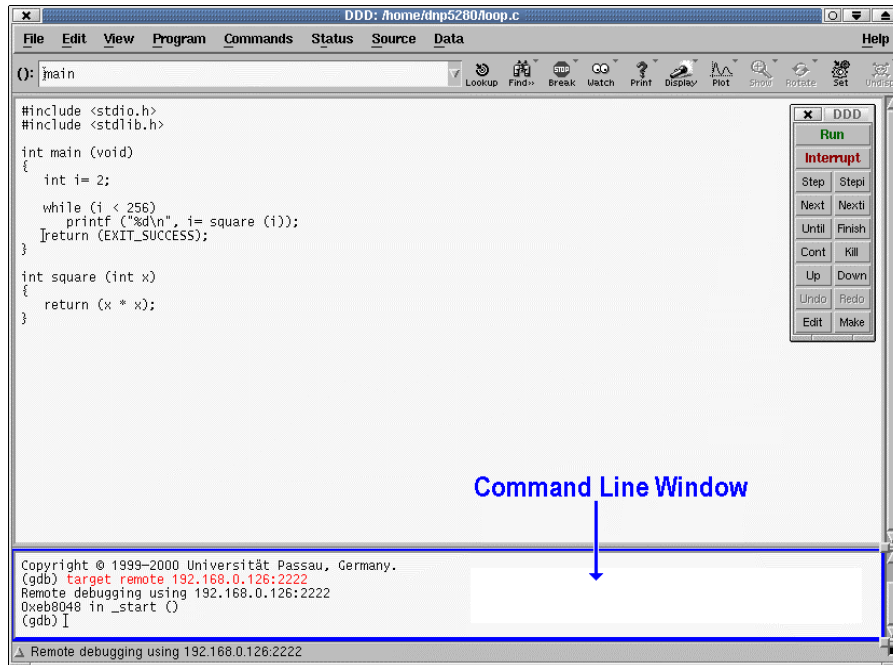
int square (int x)
{
  return (x * x);
}

GNU DDD 3.2.1 (i386-suse-linux), by Dorothea Lütkehaus and Andreas Zeller.
Copyright © 1995-1999 Technische Universität Braunschweig, Germany.
Copyright © 1999-2000 Universität Passau, Germany.
(gdb) █

Welcome to DDD 3.2.1 "Liftoff" (i386-suse-linux)
  
```

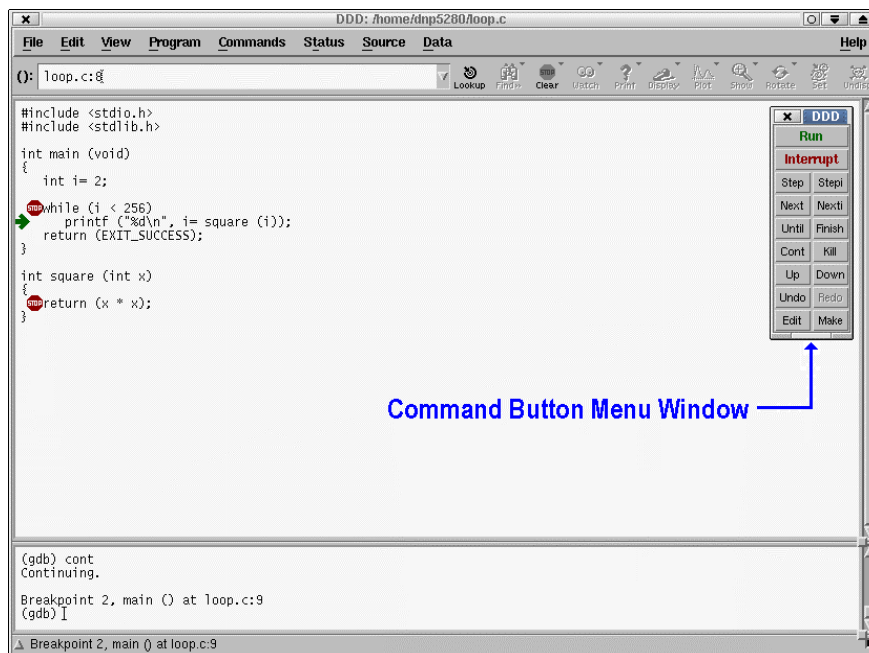
- **4. Step:** Now the debugger waits for your debugging commands. First please enter always the following command line within the DDD command line window:

target remote 192.168.0.126:2222



This debugger command line is setting up the Ethernet-based TCP/IP connection between the PC and the DNP/5280. Please use the same TCP/IP port number (see step 2). The sample command line assumes that the DNP/5280 is using the IP address 192.168.0.126.

- **5. Step:** Then set your breakpoints within the C source code and run your program with your remote debugging session between the PC and the DNP/5280.



DDD allows you to set breakpoints with your mouse. Just put the mouse cursor over the source code line of your choice and press the right hand mouse button. Then use the command button for

Continue

from the command button menu window for running the program. The program runs to the next (or first) breakpoint. You can also use the command button

Step

for single-stepping at C language level thru your program. If the program execution stops, you can enter debugger commands within the DDD command line window. For example

show version

The GNU Debugger shows then some copyright and version information and the current configuration (Build for Host i686-pc-linux-gnu. Build for Target m68k-bdm-elf).

That is all.