

Introduction

- GDB is the GNU debugger and is the standard debugger for GNU environments

•

•

Loading the debugger

- To check from the terminal whether GDB is installed type:
 - **gdb**
- If it is installed you will see the license agreement (seen next slide)
- In order to install it type:
 - **sudo apt-get install gdb**

The GDB Licence

```
alex@alex-Ubuntu: ~/Desktop/COS 284 Practical Lecture/Test 2
alex@alex-Ubuntu:~/Desktop/COS 284 Practical Lecture/Test 2$ gdb
GNU gdb (Ubuntu/Linaro 7.4-2012.04-0ubuntu2.1) 7.4-2012.04
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
For bug reporting instructions, please see:
<http://bugs.launchpad.net/gdb-linaro/>.
(gdb) █
```

Starting to debug

- Navigate to the directory containing the executable of interest (using **cd**)
- To load the executable type:
 - **file executableName**
- It is also possible to load the executable at initialization by typing:
 - **gdb executableName**
- Remember that the executable must be compiled to allow debugging i.e.:
 - **-g dwarf2**

Commands - list

- To view the source code of the executable type:
 - **list**
- The **list** command by default displays 10 lines.
When stepping through the program it displays 10 lines with the current line more or less at the centre of the displayed range
- The list command also may be in the form:
 - **list** *startLineNumber, endLineNumber*
- This form displays the specified range of lines
- If only the starting line is selected, list will display 10 lines with the desired line near the centre of the displayed range

Commands - list

```
alex@alex-Ubuntu: ~/Desktop/COS 284 Practical Lecture/Test 2
<http://bugs.launchpad.net/gdb-linaro/>...
Reading symbols from /home/alex/Desktop/COS 284 Practical Lecture/Test 2/test2..
.done.
(gdb) list
1      section .data
2      x db 19
3      y db 4
4      quot dq 0
5      rem dq 0
6      remlabel db "r"
7      newline db " "
8
9      section .text
10
(gdb) list 10, 17
10
11      global _start
12
13      _start:
14      ;Method 1
15
16      mov rdx,0
17      mov rax,0
(gdb)
```

Commands - run

- To run the executable from the beginning type:
 - **run**
- The **run** will attempt to restart the program if it is already running

Commands - break

- To insert a break point into the program type:
 - **break** *lineNumber*
- Program execution will halt whenever a break point is reached
- With no specified line number, i.e.:
 - **break**
- A break point will be inserted at the current line

Commands - continue

- To continue program execution after a break point:
 - **continue**
- This will cause the program to continue running up until the program reaches another break point or until it completes execution

Commands - clear

- To remove a break point into the program type:
 - **break** *lineNumber*
- With no specified line number, i.e.:
 - **clear**
- The break point at the current line will be removed

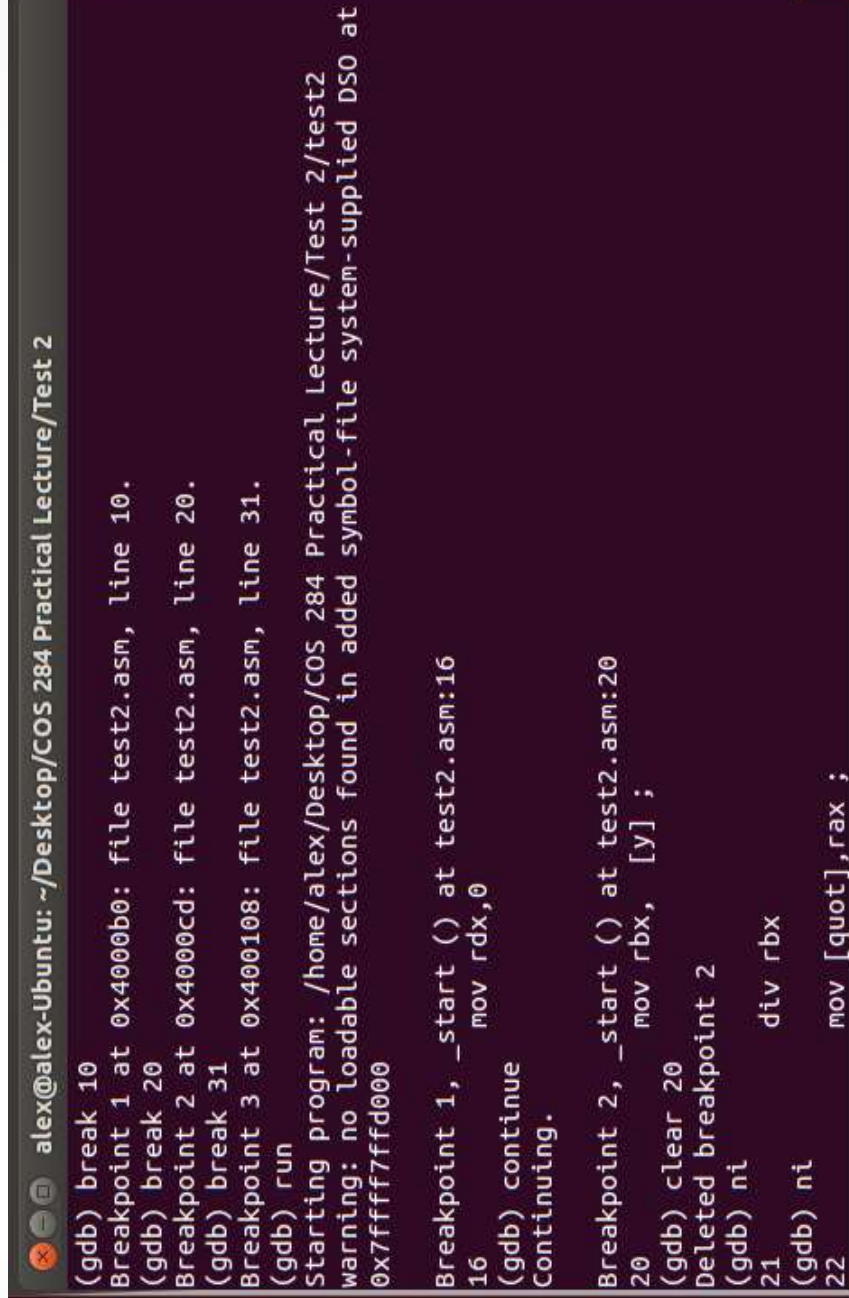
Commands - ni

- To execute only the next instruction:
 - **next instruction, nexti, ni**
- Using the **nexti** command allows the program to be stepped through line by line
- The **nexti** command continues from the current instruction

Commands - next

- To execute only the next instruction:
 - **next, n**
- **next** is very similar to **nexti**
 - Both execute the next instruction
 - **next** executes functions as one statement
 - **nexti** executes calls as one instructions

Commands – run, break, continue, clear



```
alex@alex-Ubuntu: ~/Desktop/COS 284 Practical Lecture/Test 2
(gdb) break 10
Breakpoint 1 at 0x4000b0: file test2.asm, line 10.
(gdb) break 20
Breakpoint 2 at 0x4000cd: file test2.asm, line 20.
(gdb) break 31
Breakpoint 3 at 0x400108: file test2.asm, line 31.
(gdb) run
Starting program: /home/alex/Desktop/COS 284 Practical Lecture/Test 2/test2
warning: no loadable sections found in added symbol-file system-supplied DSO at
0x7ffff7ffd000

Breakpoint 1, _start () at test2.asm:16
16      mov rdx,0
(gdb) continue
Continuing.

Breakpoint 2, _start () at test2.asm:20
20      mov rbx, [y] ;
(gdb) clear 20
Deleted breakpoint 2
(gdb) ni
21      div rbx
(gdb) ni
22      mov [quot],rax ;
```

Commands - **print**

- To display memory address or register:
 - **print/FMT** *variableName* (e.g. **x/d &loopVar**)
 - **print/FMT \$register**(e.g. **print/x \$rax**)
- The **print** command allows for an output formatting letter (the parameter FMT above).
- Output formats
 - o(octal), d(decimal), t(binary), f(float), a(address), i(instruction), c(char), s(string)

Commands - x

- To examine memory address:
 - **x/FMT &variableName** (e.g. **x/12cb &loopVar**)
 - **x/FMT address** (e.g. **x/2xb 0x7fffffffe618**)
- The **print** command allows for a count, output formatting letter and a size letter in that order (the parameter FMT above).
- Repeat count
 - This is an integer that determines how many objects to display
- Output formats
 - o(octal), d(decimal), t(binary), f(float), a(address), i(instruction), c(char), s(string)
- Size letter
 - b(byte), h(halfword), w(word), g(giant, 8 bytes)
- The default count is 1 and default size and format are those previously used
- Protip: use the address **\$rsp** to view the stack

Commands – print,x

```
alex@alex-Ubuntu: ~/Desktop/COS 284 Practical Lecture/Test 2
(gdb) print $rax
$1 = 1043
(gdb) print/x $rax
$2 = 0x413
(gdb) print/o $rbx
$3 = 04
(gdb) print/o x
$4 = 02023
(gdb) print/d x
$5 = 1043
(gdb) x/12xw &x
0x60025c <x>: 0x00000413 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x60026c <rem+6>: 0x20720000 0x0000002c 0x00000002 0x00000000 0x00000000 0x00000000
00
0x60027c: 0x00000000 0x004000b0 0x00000000 0x000001ac
(gdb) x/8cb &quot;
0x60025e <quot>: 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000'
0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000'
(gdb) x/8cg &quot;
0x60025e <quot>: 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000'
0x60026e <remLabel>: 114 'r' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000'
0x60027e: 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000'
0x60028e: 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000' 0 '\000'
(gdb) █
```


Commands - help

- To view all the gdb commands and their manuals type:
 - **help**
- To view the manual for a specific command
 - **help** *commandName*

Commands - kill

- To kill (or end) the program running in gdb type:
 - **kill,k**
- The **kill** command is also useful if you wish to recompile and relink your program

Commands - quit

- To exit gdb type:
 - **quit**