ARM Debugging

At some point, your program will likely fail to compile, and you will need to do some debugging. Here is a simple primer to “gdb”, the debugging tool bundled with your emulator.

1. In the same directory on your emulator as your program, run the following: “gdb <program name>”
2. The result should be similar to the following:

```
root@debian-aarch64:~/workdir# gdb ttt
GNU gdb (Debian 7.12-6) 7.12.0.20161007-git
Copyright (C) 2016 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 "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ttt...done.
(gdb)
```

3. To set a breakpoint, run “b <label name”:

```
(gdb) b main
Breakpoint 1 at 0x968: file ttt.s, line 19.
(gdb)
```

4. To begin running the program, run “r”:

```
Starting program: /root/workdir/ttt

Breakpoint 1, main () at ttt.s:19
19              ldr x0, =welcomePrompt
(gdb)
```

5. To step through the program, run “s”:

```
(gdb) s
20              bl printf
(gdb)
```

6. To view the current register values, run “i r”:

```
(gdb) i r
x0             0xaaaaaaaabb051   187649984540753
x1             0xffffffffc58     281474976709720
x2             0xffffffffc68     281474976709736
x3             0xaaaaaaaaa968    187649984473448
x4             0xffffffffb78     281474976709496
x5             0x0       0
x6             0x0       0
x7             0x10      16
x8             0xffffffffffffffff        -1
x9             0x3fff    16383
x10            0x101010101010101         72340172838076673
x11            0x10      16
x12            0xffffb7fff030    281473768747056
x13            0xffffb7fff028    281473768747048
x14            0x0       0
x15            0x80d     2061
x16            0xffffb7ea7288    281473767338632
x17            0xaaaaaaaabb010   187649984540688
x18            0x40941   264513
x19            0xaaaaaaaaaca0    187649984474272
x20            0x0       0
x21            0xaaaaaaaaa820    187649984473120
x22            0x0       0
x23            0x0       0
x24            0x0       0
x25            0x0       0
x26            0x0       0
x27            0x0       0
x28            0x0       0
x29            0xffffffffb20     281474976709408
x30            0xffffb7ea7364    281473767338852
sp             0xffffffffb20     0xffffffffb20
pc             0xaaaaaaaaa96c    0xaaaaaaaaa96c <main+4>
cpsr           0x60200000        [ EL=0 SS C Z ]
fpsr           0x0       0
fpcr           0x0       0
(gdb)
```

7. To view 16 bytes of memory starting at the stack pointer, run “x/16xb $sp”:

```
(gdb) x/16xb $sp
0xffffffffb20: 0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0xffffffffb28: 0x54    0xa8    0xaa    0xaa    0xaa    0xaa    0x00    0x00
(gdb)
```

8. Detailed format instructions for the memory view command “x/” are here: https://sourceware.org/gdb/onlinedocs/gdb/Memory.html
9. To restart the program from your breakpoint, run “r” again:

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /root/workdir/ttt

Breakpoint 1, main () at ttt.s:19
19              ldr x0, =welcomePrompt
(gdb)
```

10. When you are done debugging, run “quit” to exit:

```
(gdb) quit
A debugging session is active.

        Inferior 1 [process 676] will be killed.

Quit anyway? (y or n) y
root@debian-aarch64:~/workdir#
```