

CS6640 Handout Week3.b

1. egos-2k+ memory layout I

HIGH MEM ADDR

```

+-----+ <- 0x8080_0000 [RAM_END]
| apps free pages | [APPS_PAGES_BASE .. RAM_END]
| for mmu_alloc   | (4MB)
+-----+ <- 0x8040_0000 [APPS_PAGES_BASE]
| app stack       | [APPS_STACK_TOP]
| (grows down)   |
+-----+
| shell work dir  |
+-----+ <- 0x8030_2000 [SHELL_WORK_DIR]
| syscall arg struct
+-----+ <- 0x8030_1000 [SYSCALL_ARG]
| app args
+-----+ <- 0x8030_0000 [APPS_ARG]
| app code + data | [APPS_ENTRY .. APPS_ARG]
| (1MB)
+-----+ <- 0x8020_0000 [APPS_ENTRY]
| egos stack      | [EGOS_STACK_TOP]
| (grows down)   |
+-----+
| grass struct    |
+-----+ <- 0x8010_1000 [GRASS_STRUCT]
| earth struct    |
+-----+ <- 0x8010_0000 [EARTH_STRUCT]
| egos code + data | [RAM_START .. EARTH_STRUCT]
| (1MB)
+-----+ <- 0x8000_0000 [RAM_START]

```

LOW MEM ADDR

2. gdb cheat sheet

Multi-core support

```

(gdb) info threads list all threads known to gdb
(gdb) thread <n> switch the current context to thread <n>

```

Breakpoints & watchpoints

```

(gdb) break main set a breakpoint on a function
(gdb) break basic.c:101 set breakpoint at file and line (or function)
(gdb) info breakpoints show breakpoints
(gdb) delete 1 delete a breakpoint by number
(gdb) watch expression set software watchpoint on variable
(gdb) info watchpoints show current watchpoints

```

Running the program

```

(gdb) c continue the program
(gdb) s a step in C; step into functions
(gdb) si a step in asm; step into functions
(gdb) n a step in C; step over functions
(gdb) ni a step in asm; but step over functions
(gdb) CTRL-C actually SIGINT, stop execution of current program
(gdb) finish finish current function's execution

```

Stack backtrace

```

(gdb) bt print stack backtrace
(gdb) info locals print automatic variables in frame
(gdb) info registers print registers sans floats

```

Browsing Data

```

(gdb) p expr print expression
(gdb) p/x expr print in hex
(gdb) p/t expr print in binary
(gdb) p/i expr print as instructions

```

```

(gdb) x/FMT address low-level examine command
(gdb) x/x 0x80001000 print memory in hex
(gdb) set var = expr assign value

```

```

(gdb) display/FMT expr display expression result at stop
(gdb) display/i $pc print next instruction
(gdb) undisplay delete displays

```

FMT (Format letters) are:

```

o(octal), x(hex), d(decimal), u(unsigned decimal),
t(binary), f(float), a(address), i(instruction), c(char), s(string)
and z(hex, zero padded on the left).

```

Load a program's symbols

```

(gdb) add-symbol-file <elf> load symbol table from <elf>

```

Quit

```

(gdb) quit quit gdb

```

[borrowed and customized from

```

https://gist.github.com/rkubik/b96c23bd8ed58333de37f2b8cd052c30

```