GIT:-

* no zip or any archive
* no generated files

```
git clone https://gitlab.com/gea-training/elinux-bsp/linux-cli-prog/
                                                  # first time
cd linux-cli-prog
git pull
```

Please add "elinuxfaculty" as Reporter under "Members"

CFLAGS

* gcc/g++ options, GNU tools
* multifile prog
* Makefiles
* static libs
* dynamic libs

---

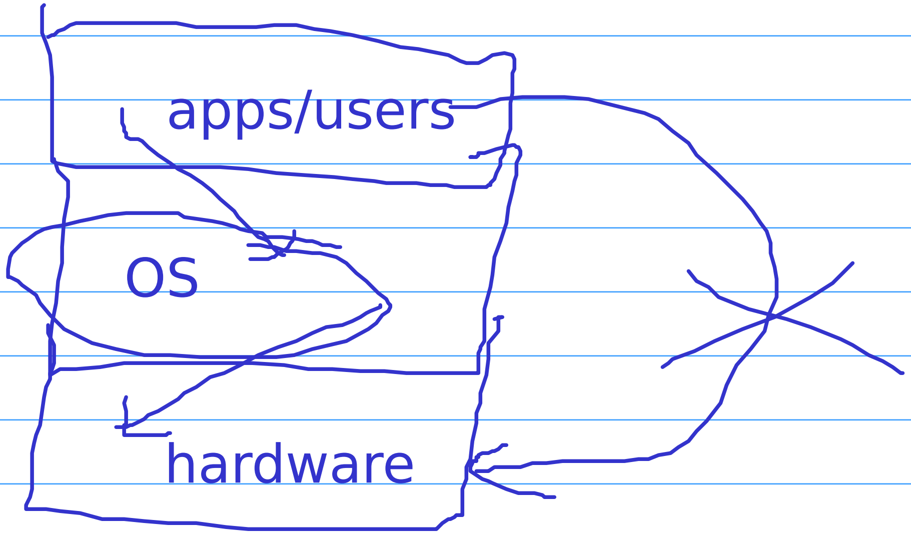
* valgrind
* gdb -- r, s, c, b, bt, q

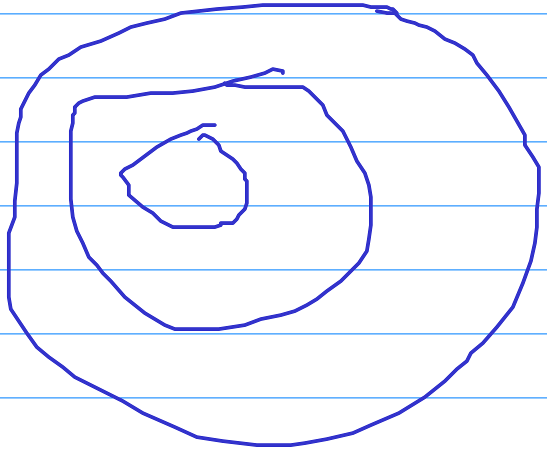
```
export LD_LIBRARY_PATH=~/dlibs
./d.out
```

/etc/ld.so.conf  ==> add custom dir here
ldconfig
--------------------------------------------------------------
https://gitlab.com/gea-training/elinux-bsp/linux-sys-programming

```
git clone xxx
```

OS Cocepts & Linux Programming
System Calls





Basic Comp Arch:-

- * CPU
  - * execution engine (ALU, CU)
  - * CPU regs
  - * timer/clock
  - * cpu cache
- * Memory (RAM)
- * I/O Devices (peripherals) including
  - * storage devices

CPU Regs - special purpose regs (program counter/instruction pointer, flags/program status word, stack pointer, frame/base pointer)
General purpose regs  (including accumulator)

CPU Cache - levels of cache (L1, L2)
- private cache vs common cache
- i-cache, d-cache

I/O -- interrupt driven i/o vs polling techniques, DMA

SMP -- Symmetric Multi Processing (SMP)

----------------------------------------------------------------------

FLAGS reg (program status word)

- status & control bits

mode bit(s) - control bit

* supervisor mode (unlimited/unrestricted/privileged)
* normal mode (limited/restricted/unprivileged)

normal mode -- subset of instructions, subset of memory
-- limited/zero hardware access

supervisor mode -- full hardware access, entire intstruction set
entire memory

What if mode bit is not present -- flat mode

Dual mode operations -- normal / supervisor

enter to superuser mode        -- trap instruction
        e.g. int80h or sysenter in x86
                swi or svc                in ARM

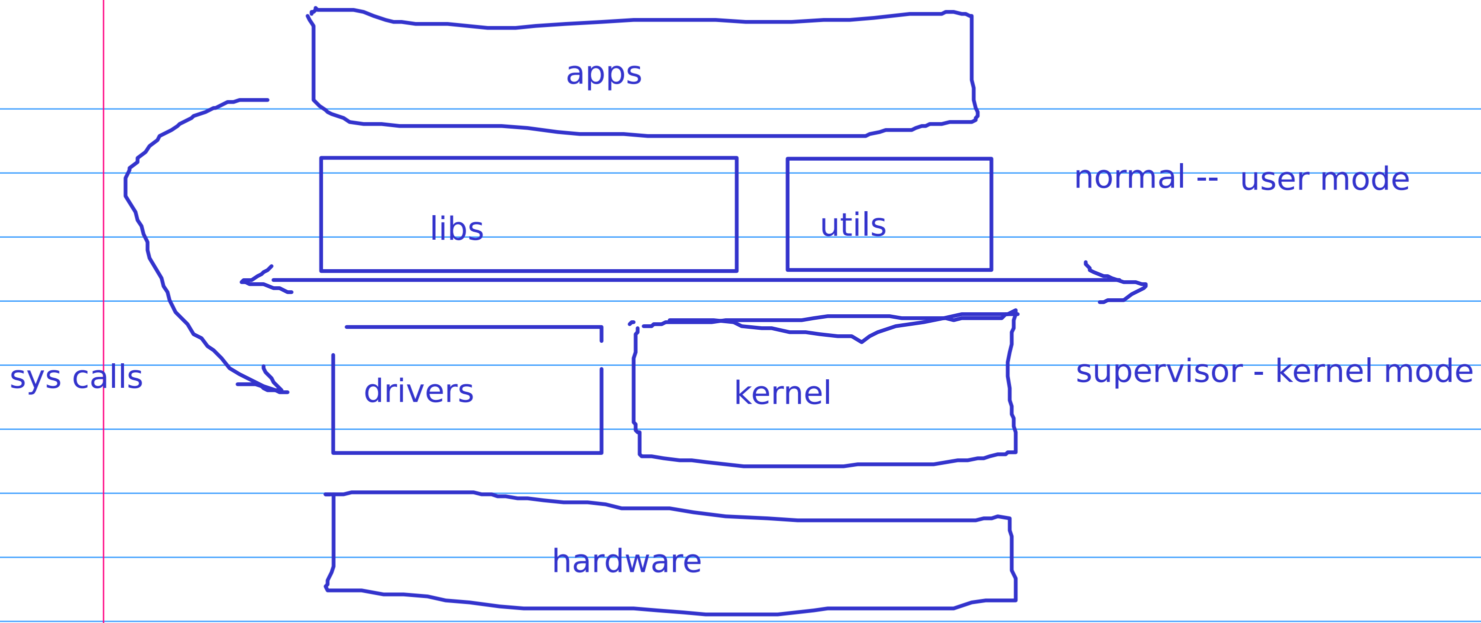
SOCs

Key differences:-              MMUs, MPUs

-------------------

OS Architecture & Components:-
* Kernel    -- core/essential component
* Drivers  -- additional hardware, i/o
* Libraries
* System utilities

Kernel:-
* core/essential component of OS
* it resides in meomory all time
* provides services to apps & libraries (system calls)



Memory space -- user space vs kernel space

user mode exec - userspace only
kernel mode exec - kernel space + user space also (entire)
--------------------
Types of kernel (self-study)
Monolithic Kernel
Micro Kernel

Modular kernel, e.g. Linux

Linux is a modular kernel  -- collection of modules

-- static vs dynamic modules

dynamic modules can be loaded/unloaded at runtime

every driver is a module in Linux kernel

-------------------

interrupt pending bit

Interrupts:-

* Typically caused by i/o devices (becz of previous request)
* ISRs / Interrupt Handlers -- service the interrupts
* IRQ - Interrupt Request, IRQ lines
* service the interrupts utmost priority
* ISRs should be short (time)
* No blocking calls inside interrupt
* Maskable vs Non Maskable
* Disabling interrupts for long duration is not recommended.

System Calls:-

* services provided by kernel (service handlers, way of providing kernel services)
* defined in kernel space, executed/requested from userspace
* mode switching occurs (trap instruction)
* identified by unique number
* also known as software interrupt
* parameter flow, return values

Further (Pre) Reading:-

* https://www.tutorialspoint.com/inter_process_communication/index.htm
* https://www.tutorialspoint.com/operating_system/index.htm (optional)
* https://linuxjourney.com ==> process, kernel
* https://linuxhint.com/category/system-calls/
* http://www.yolinux.com/TUTORIALS/ForkExecProcesses.html

Process Management, Signals, Threads, Scheduling
IPC , Memory, File System

Youtube:- Shell Wave, Neso Academy