Frosted

FRee Operating System for Tiny Embedded Devices

Why Frosted?

- FreeRTOS is great, but …
  - It’s not a complete OS: scheduler + mutex + semaphore
  - No POSIX API
  - No kernel/userspace separation
  - Not free as in free speech

- (uc)Linux is great, but …
  - Needs “lots” of RAM, flash, …
  - Linux keeps growing bigger… (See kernel tinyfication)
  - Hardware is more expensive
    - RPi, BeagleBone ~ $35
    - STM32 Discovery: ~ $10
    - Even more when scaling

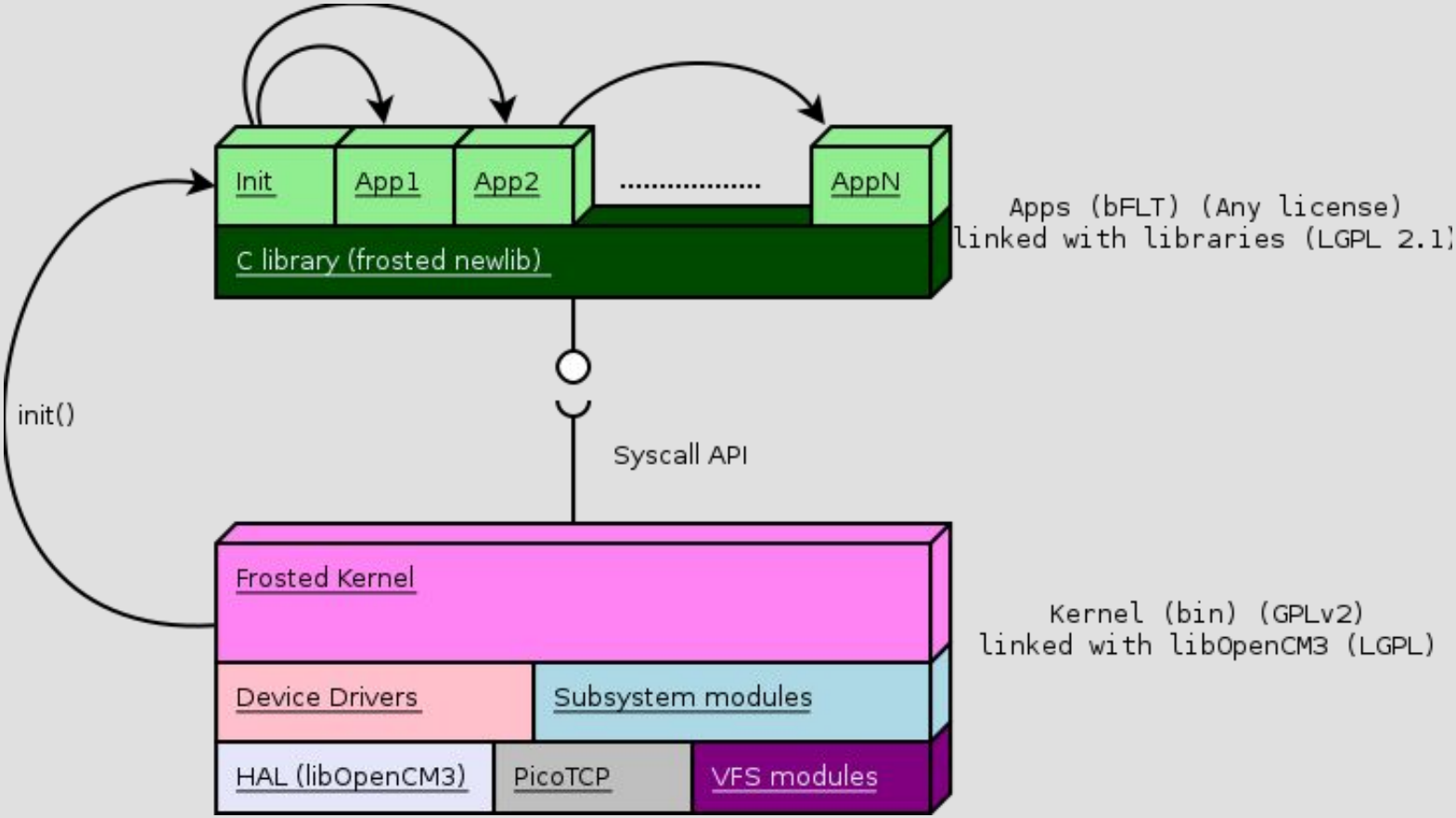
The Frosted Approach

- POSIX for microcontrollers
- Kernel/User space separation
- Complete OS with IoT focus
  - files, subsystems, sockets, …
  - fully featured TCP/IP stack
- Optimized for embedded
  - ARM Cortex-M microcontrollers
- Free Software
  - GPLv2 kernel

POSIX API

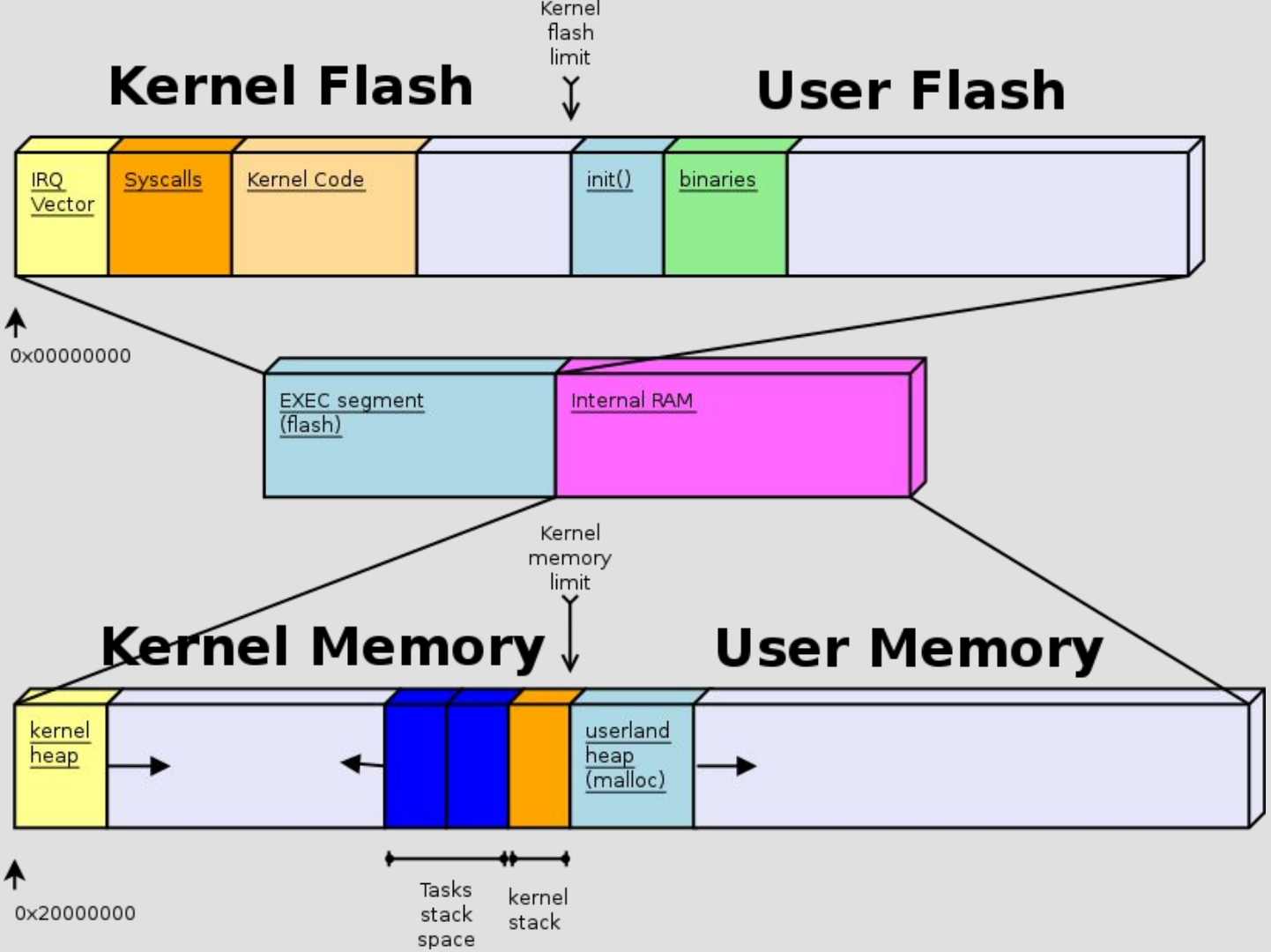
- Port UNIX apps to Frosted
  - A lot of high-quality software can be re-used
  - Use you POSIX-skills
- Berkeley Socket API
  - TCP/IP sockets (using picoTCP)
  - UNIX sockets (work in progress)
- Not a POSIX “compatibility layer”

Architectural overview

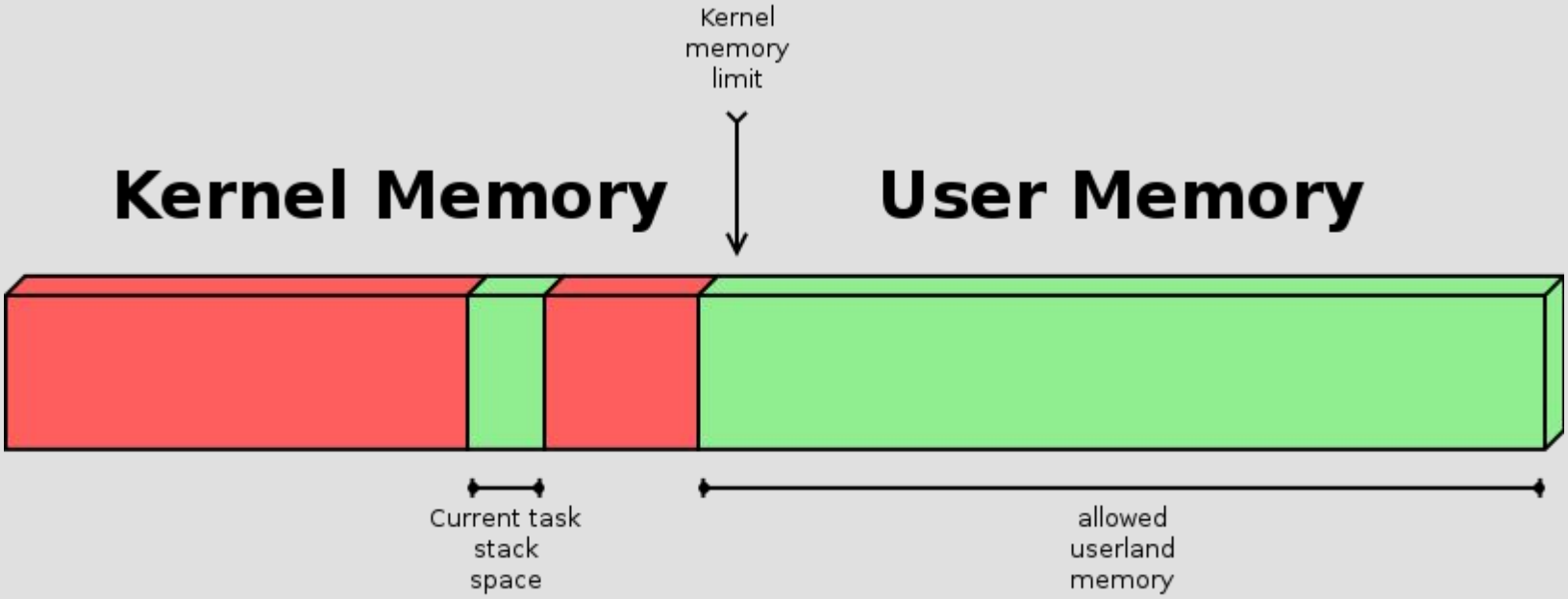


Kernel / Userspace separation

- Compile-time
  - kernel binary
    - frosted kernel
  - apps binaries
    - different application binaries in XIPFS
    - bFLT binary format
    - FDPIC ELF in the future
- CPU support
  - Kernel running in Privileged mode
  - Threads running in User mode
- Memory
  - Separated memory pools
  - MPU


Kernel flash limit
Kernel Flash
User Flash
IRQ Vector
Syscalls
Kernel Code
init()
binaries
0x00000000
EXEC segment (flash)
Internal RAM
Kernel memory limit
Kernel Memory
User Memory
kernel heap
userland heap (malloc)
0x20000000
Tasks stack space
kernel stack

Memory protection



Subsystems

- devfs: Driver subsystems
  - I2C
  - SPI
  - UART
  - RNG
- vfs: Filesystem
  - mount, …
- Sockets
  - TCP/IP through picoTCP
    - First hardware driver yet to be written
  - Unix sockets (not yet)

Userspace interface

- Compiling
  - Standalone executables compiled
  - Compile and link using arm-frosted-eabi toolchain
  - Generated bFLT exectuable
  - Create XIPFS filesystem with many binaries

- Kernel
  - mounts XIPFS filesystem
  - execve() syscall
  - Loads bFLT format executable
    - Execute .text in-place
    - Allocate .data and .bss sections
  - POSIX system calls through SVC interrupt
    - Interface implemented in frosted-newlib

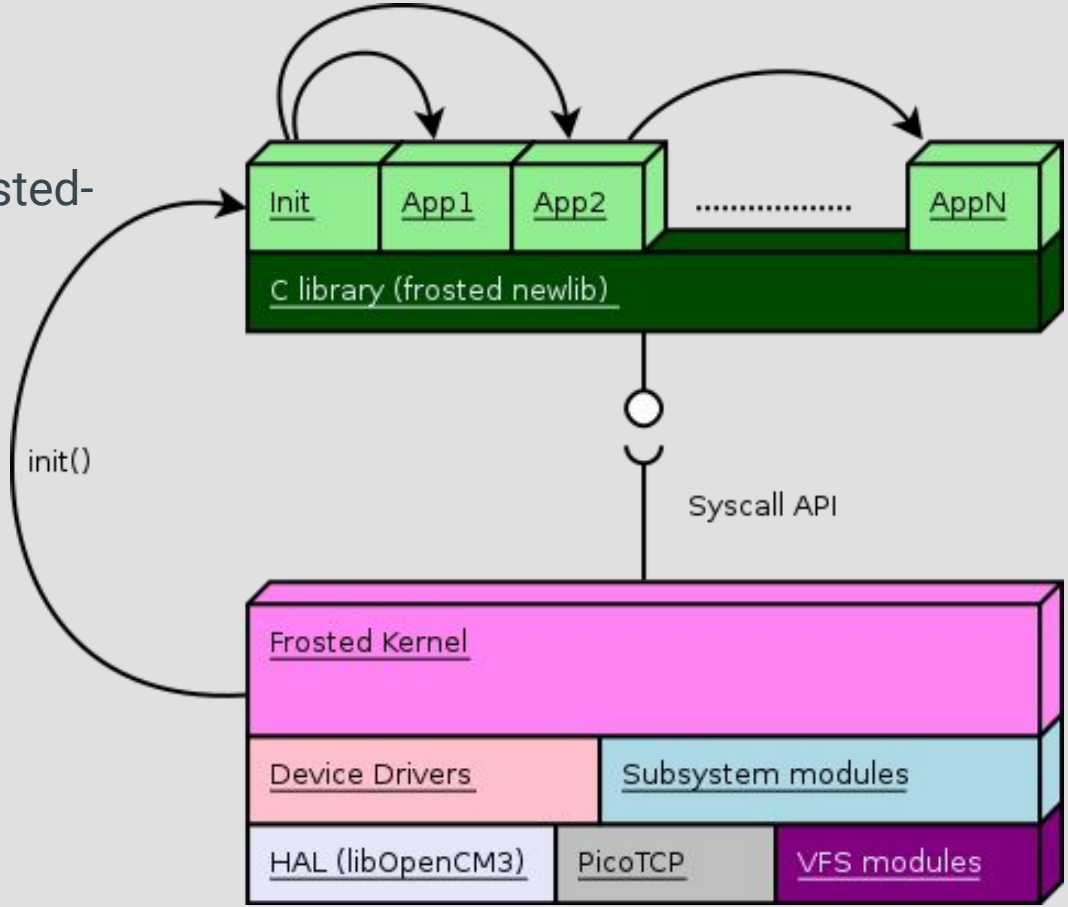
Target platforms

ARM Cortex-M microcontrollers (for now)

- Many manufacturers / very popular
- Specifics
  - Context switch
  - MPU
  - SysTick

Licensing

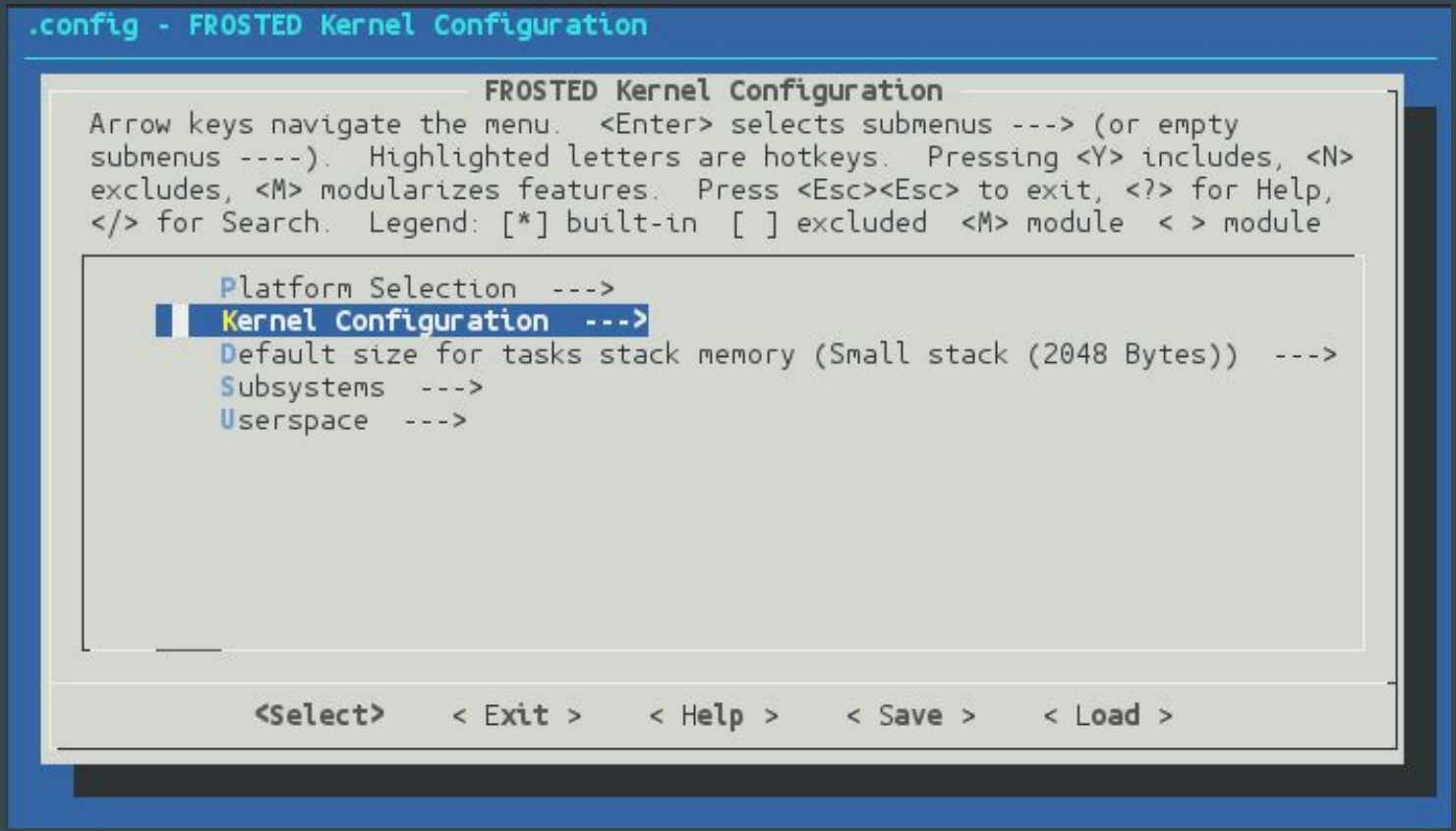
- Applications linked with frosted-newlib
  - LGPL v2.1 + newlib license
  - Applications can be licensed differently
- Kernel + Drivers:
  - GPL v2
- libopencm3
  - LGPLv3 with linking exception



Using Frosted

- Clone git repo
  - clone, submodule init, submodule update
- Install arm-frosted-eabi toolchain
- Configure kernel
  - $> make menuconfig
- Configure userspace
  - $> make menuconfig

Using Frosted



Using Frosted

- Compile
  - $> make
- Run
  - On target
    - STM32F4
    - LPC17xx
    - LM3S
  - Through qemu
    - $> make qemu (waits for gdb)
    - $> make qemu2

Frosted TODOs

- picoTCP integration
  - Integrated, but no driver yet
- Dynamic libraries
  - dlopen()
- Applications
  - Busybox
- Support more boards, drivers, …
- < Your feature here >

Join the team!

github.com/insane-adding-machines/frosted

IRC: #frosted on freenode

maximevince a.k.a. Maxime Vincent

maxime [dot] **vince** [at] **gmail** [dot] **com**

danielinux a.k.a. Daniele Lacamera

root [at] **danielinux** [dot] **net**