

Getting to Work with OpenPiton

Jonathan Balkind, Michael McKeown, Yaosheng Fu, Tri Nguyen,
Yanqi Zhou, Alexey Lavrov, Mohammad Shahradsad, Adi Fuchs,
Samuel Payne, Xiaohua Liang, Matthew Matl, David Wentzlaff

Princeton University

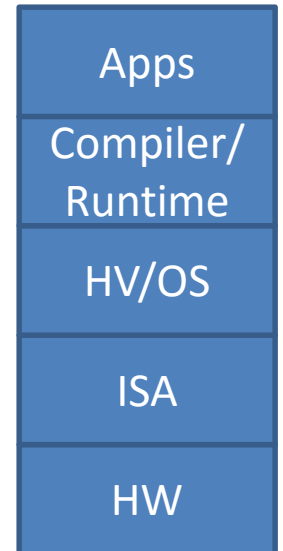
<http://openpiton.org>



Operating System and System Software

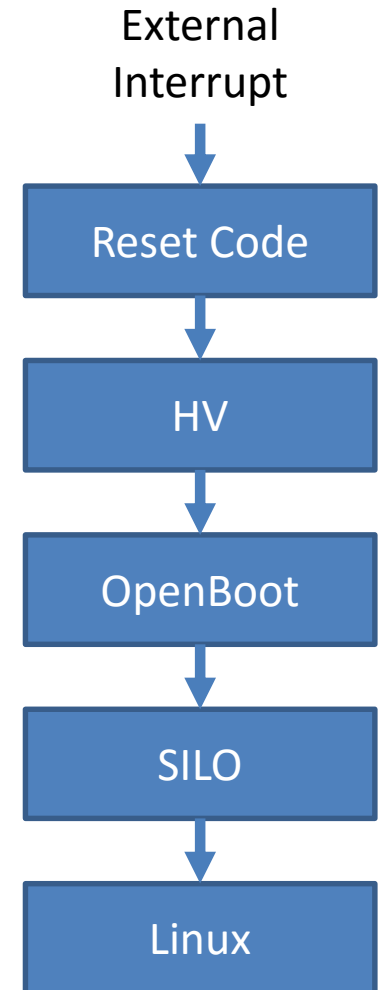
Open source system stack

- Applications run on Linux
 - Linux manages virtualised HW, calls to HV
 - OpenBoot handles OS boot from SD
 - Hypervisor manages HW resources
 - Open source hardware
-
- You can read, modify and recompile all of them!



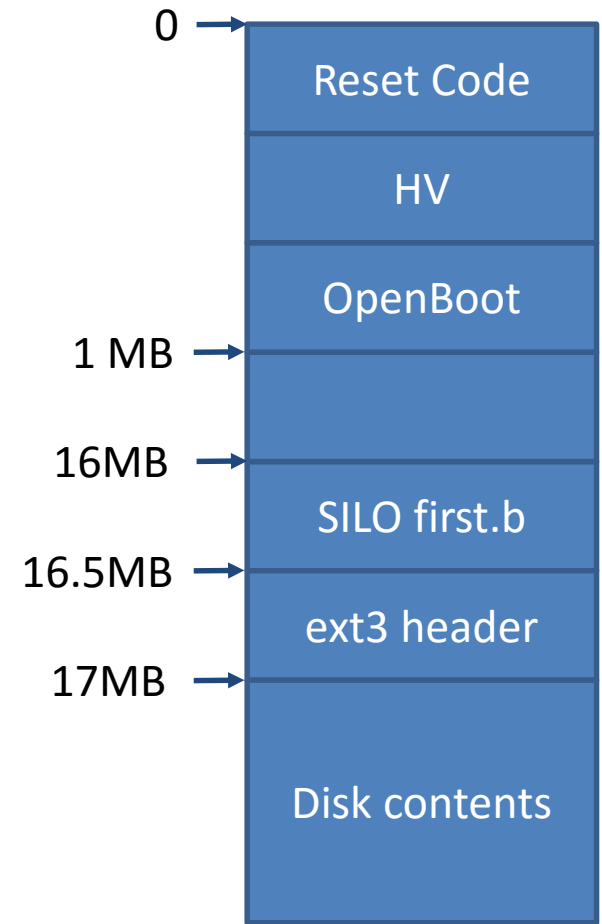
Boot Process

1. Interrupt
 - Core woken from outside
2. Reset Code
 - Clears registers and on-chip memories
3. Hypervisor
 - Sets up trap table, copies self to memory
4. OpenBoot
 - Initial bootloader, reads SILO ELF from SD
 - Drop to OpenBoot shell using Stop-A/Break
5. SILO
 - Loads Linux kernel from SD card
6. Linux



Anatomy of a disk image

- Bottom 1MB:
 - Reset code, HV, OpenBoot
- At next 16MB alignment:
 - Sun disk image
- First sector contains SILO first.b
- Disk image is formatted as ext3
- Debian is vanilla from debootstrap



Mounting a disk image

- `objcopy -I binary -O binary --reverse-bytes=8 mydisk.img mydisk-rev.img`
- `sudo mount -o loop,offset=16M mydisk-rev.img mntdir/`
- `cd mntdir/`
- **Navigate, copy files, etc**

Installing applications

- Works natively, should work modified on Debian x86_64
- **Setup chroot:**
- `mkdir mntdir`
- `sudo mount -o loop,offset=16M mydisk-rev.img mntdir/`
- **Be very careful running these!**
- `sudo mount -o bind /proc mntdir/proc`
- `sudo mount -o bind /dev mntdir/dev`
- `sudo mount -o bind /sys mntdir/sys`
- `sudo cp /etc/resolv.conf mntdir/etc/`
- `cd mntdir/`
- `sudo chroot .`

Installing applications

- **Install apps:**
- `sudo apt-get install <package>`
- **Then when you are done (Be very careful running these!):**
- `exit`
- `sudo umount mntdir/proc`
- `sudo umount mntdir/dev`
- `sudo umount mntdir/sys`
- `sudo umount mntdir/`

- `objcopy -I binary -O binary
--reverse-bytes=8 mydisk-rev.img mydisk.img`

Building the Linux kernel

- Clone our git repository from <https://github.com/PrincetonUniversity/piton-linux>
- Build native or set up a cross-compiler for sparc64
- Set \$ARCH to sparc, change config to cross-compile
- Navigate to the root directory and compile
 - `make oldconfig && make -j32`
- Copy files to disk image
 - `vmlinux, zImage, System.map, .config`

Copying kernel to disk image

- `sudo mount -o loop,offset=16M mydisk-rev.img mntdir/`
- `sudo cp piton-linux/vmlinux mntdir/boot/vmlinux`
- `sudo cp piton-linux/System.map mntdir/boot/System.map-4.1-piton`
- `sudo cp piton-linux/arch/sparc/boot/zImage mntdir/boot/vmlinuz-4.1-piton`
- `sudo cp piton-linux/.config mntdir/boot/config-4.1-piton`

Building the Hypervisor and OpenBoot

- Lightly modified from T1, tested on Solaris 9
- Clone our git repository from <https://github.com/PrincetonUniversity/piton-sw>
- Set up Sun development tools
- `source subos/OpenSPARCT2_SAM.bash`
- `cd t1_fpga/subos/t1_fpga/src/`
- `make`
- **Copy** `xilinx/prom/1c1t_obp_prom.bin` to remote machine
- `dd if=1c1t_obp_prom.bin mydisk-rev.img conv=notrunc`