为果壳和SiFive等RV硬件移植定制Linux操作系统的经验与思考

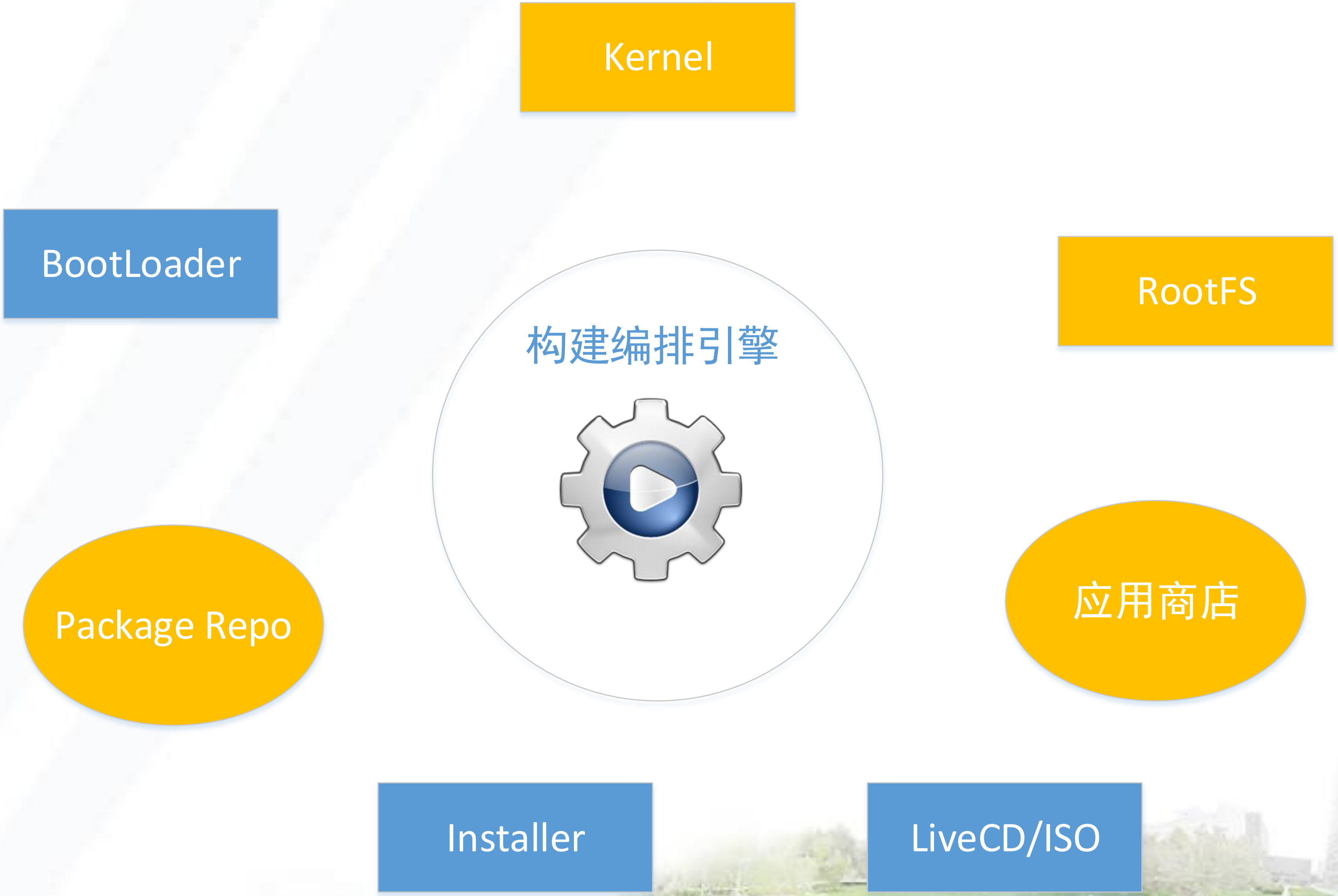
报告人：周鹏 zhoupeng@iscas.ac.cn

2020年12月26日

提纲

- 操作系统发行版构建的典型方法
- openEuler面向RISC-V操作系统构建
- 为SiFive定制操作系统经验与计算能力验证
- 操作系统构建的思考与难点分析

构建一个操作系统发行版涉及的主要版块



构建编排引擎

Kernel

BootLoader

RootFS

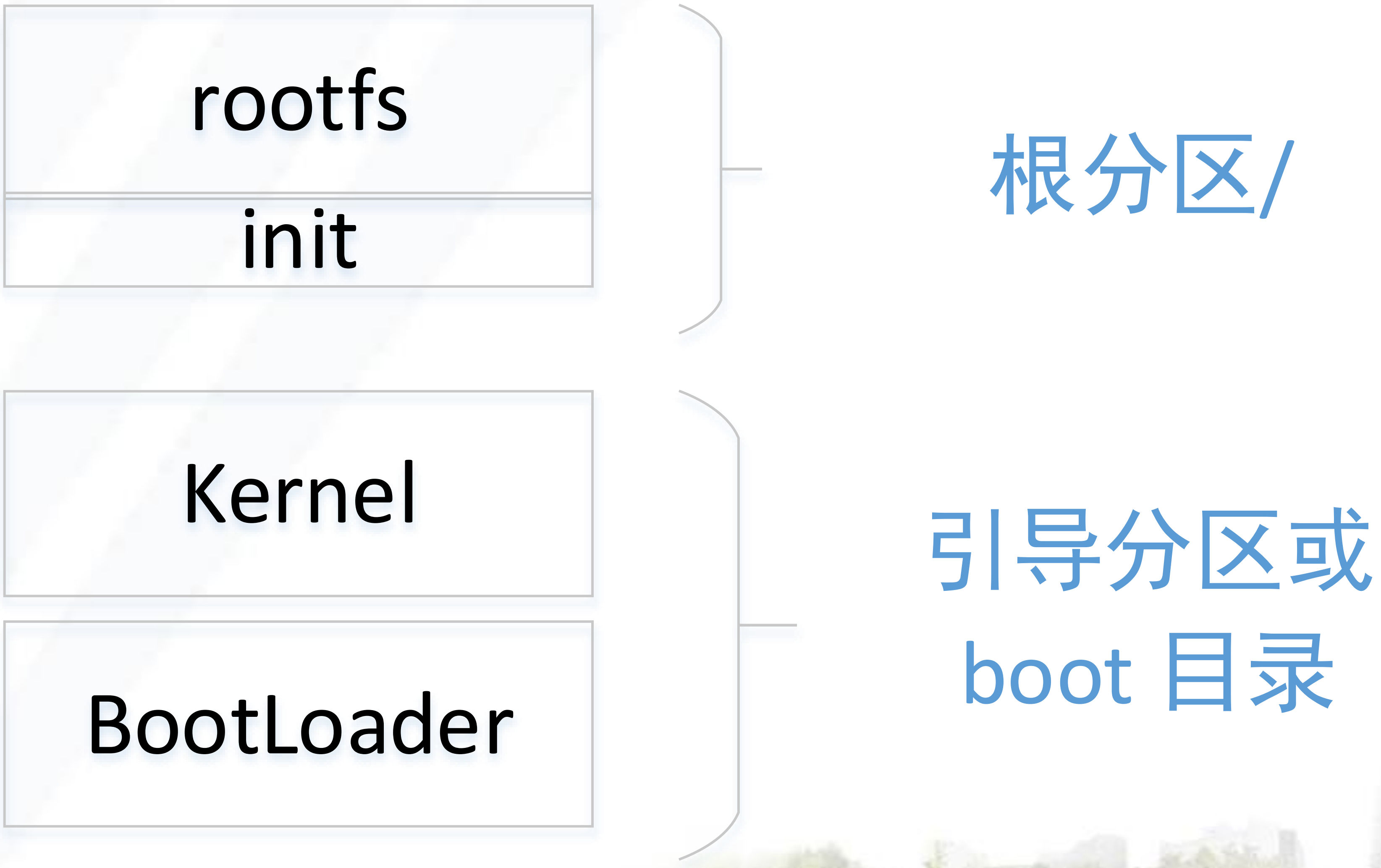
Package Repo

应用商店

Installer

LiveCD/ISO

操作系统组件间关系



rootfs

init

根分区/

Kernel

BootLoader

引导分区或 boot 目录

参考工具

- Rootfs
  - debootstrap、dpkg
  - dnf、yum
- Installers
  - Ubiquity
  - Anaconda
- BootLoader
  - Grub2
  - BBL(Berkeley Boot Loader)
  - U-boot
  - OpenSBI

- Build packages
  - dpkg-buildpackage、 debuild
  - rpmbuild、mock、koji
- Make CD/ISO
  - squashfs、mkisofs
  - livemedia-creator

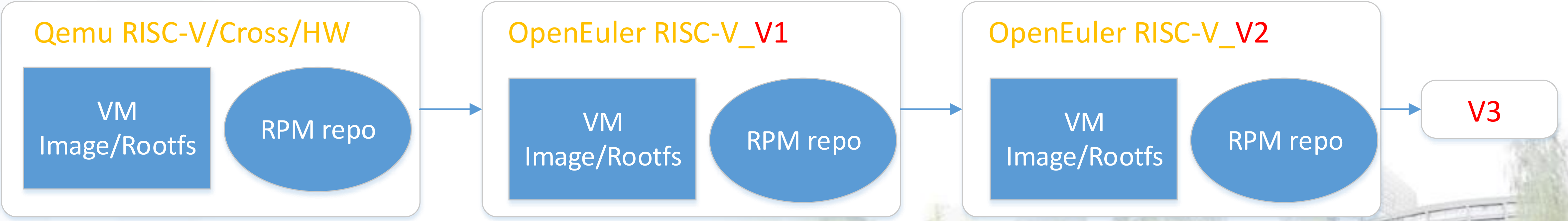
- LFS
  - Linux From Scratch
- Yocto/OpenEmbedded

提纲

- 操作系统发行版构建的典型方法
- openEuler面向RISC-V操作系统构建
- 为SiFive定制操作系统经验与计算能力验证
- 操作系统构建的思考与难点分析

技术路线

- Round1
  - 源码包编译环境：交叉编译，同类系统辅助编译
  - rootfs构建环境：系统构建依赖满足
  - 构建oe-RISCV-repo V1，构建rootfs V1，可 启动运行
- Round2
  - Round1增长到一定程度，辅助使用Round1环境滚动迭代oe-RISCV-repo V2
  - rootfs V2和oe-RISCV-repo V2是由V1和新增混合
- Round3：完全由V2和高于V2的环境构建，依赖丰富自给，满足OBS需要



Qemu RISC-V/Cross/HW

VM Image/Rootfs

RPM repo

OpenEuler RISC-V_V1

VM Image/Rootfs

RPM repo

OpenEuler RISC-V_V2

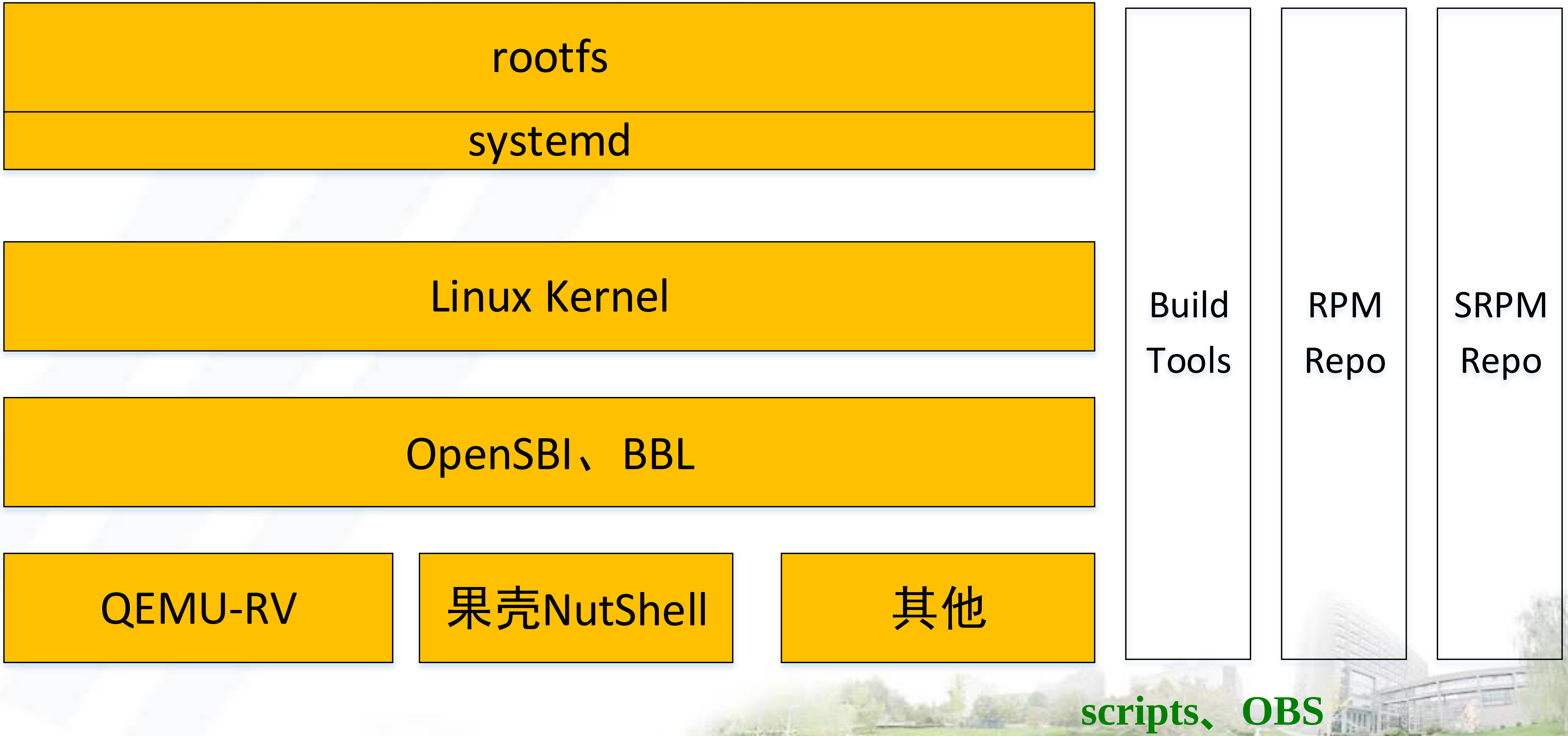
VM Image/Rootfs

RPM repo

V3

自举构建迭代路线图

openEuler RISC-V OS概要结构



rootfs

systemd

Linux Kernel

OpenSBI、BBL

QEMU-RV

果壳NutShell

其他

Build Tools

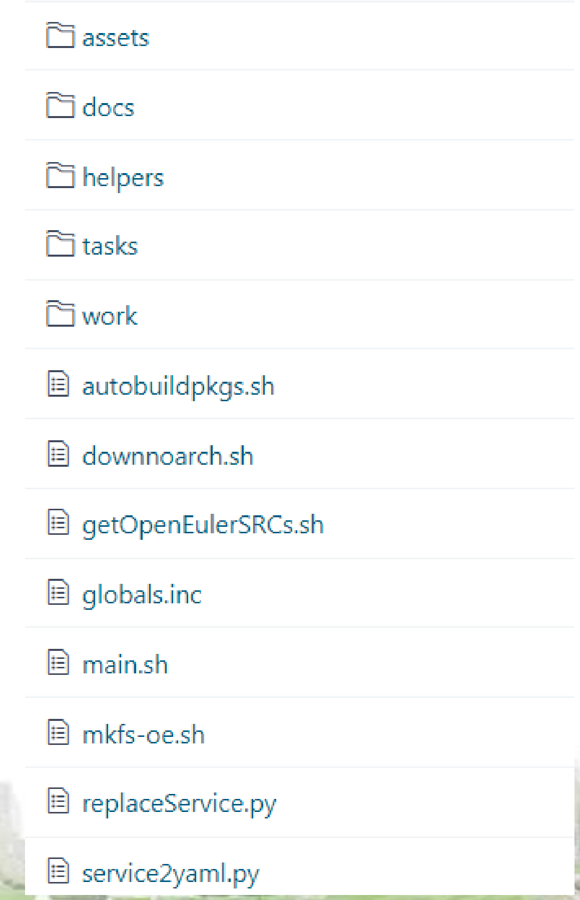
RPM Repo

SRPM Repo

scripts、OBS

openEuler RISC-V OS 构建工具和任务

- 源码包编译构建
  - 批量编译构建编排脚本
    - 指定批量编译任务
    - 以源码包为任务粒度
    - 多个编译实例环境并行编译
    - 分阶段编译错误自动归类
  - 基础指令：rpmbuild + chroot/QEMU
  - OBS
- rootfs/根文件系统构建
  - 构建配置
  - 构建流程编排脚本
  - 基础构建指令：dnf / yum + chroot + rpm repo
- 内核编译构建
- BootLoader编译构建
  - openSBI
  - BBL



```
assets
docs
helpers
tasks
work
autobuildpkgs.sh
downnoarch.sh
getOpenEulerSRCs.sh
globals.inc
main.sh
mkfs-oe.sh
replaceService.py
service2yaml.py
```

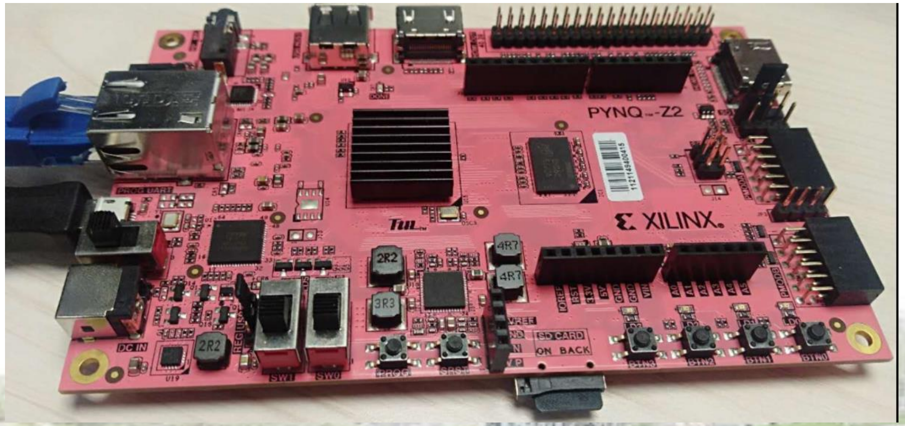
对果壳（NutShell）处理器支持

- 硬件
  - 果壳（NutShell） 处理器Core [1] + Xilinx FPGA
- GCC的RISC-V版本编译工具链准备
- 相对于openEuler for RISC-V需要做修改的包
  - openEuler-Kernel-NutShell：补丁 + kernel config
  - openEuler-systemd-NutShell：定制udev rules适配ttyPS0
  - openEuler-riscv-pk-NutShell： 内核参数、启动参数、DTS
  - openEuler-riscv-glibc-NutShell：编译配置
- openEuler for RISC-V的其他包可以重用

PL (FPGA)

+

PS (SoC)



[1] 包云岗等, https://github.com/OSCPU/NutShell

对果壳（NutShell）处理器支持——部署

- 果壳（NutShell）FPGA硬件环境
  - RV_BOOT.UCAS_COOSCA1.0_V1.BIN
    - BBL + kernel
  - oe-UCAS_COOSCA1.0-rootfs.v1.tar.gz
    - 适配在NutShell硬件环境运行的根文件系统。
  - BOOT.BIN[2]
    - NutShell CPU处理器核

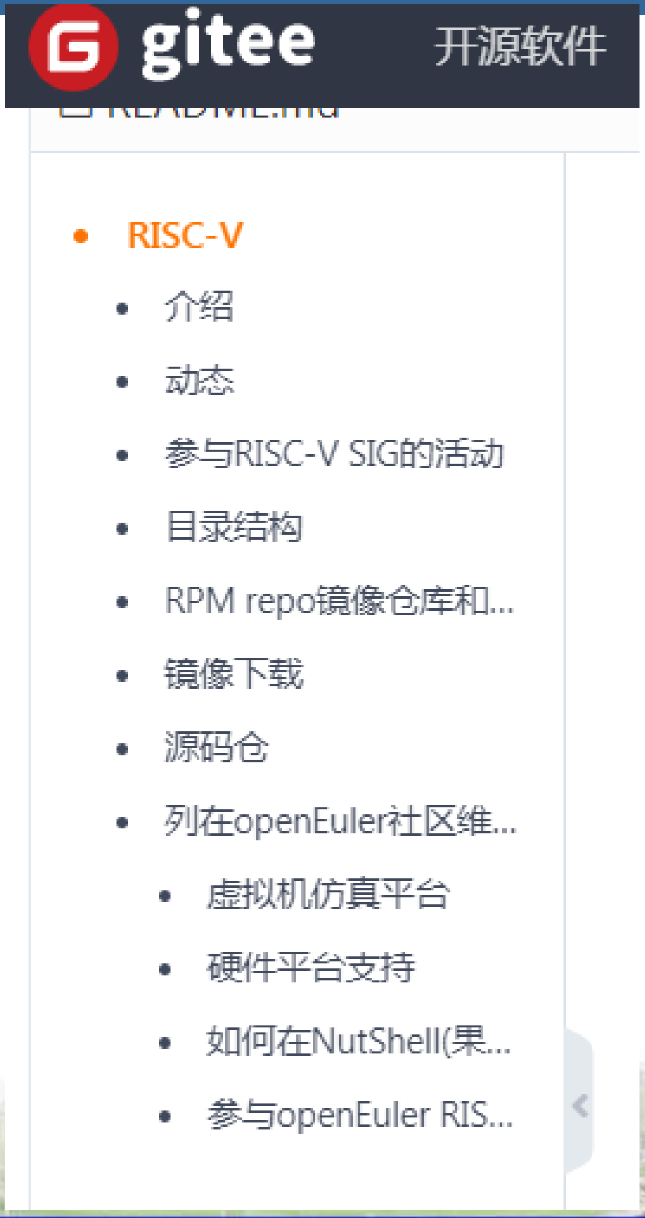
- 在操作系统上安装支持RISC-V 64的QEMU版本

- QEMU RV64仿真环境
  - fw_payload.elf
    - openSBI + kernel
  - oe-rv-rv64g-30G.qcow2
    - 虚拟磁盘镜像文件，内部分区包含根文件系统的内容。
  - oe-rv64g-rootfs.tar
    - 原始根文件系统内容

[2] https://github.com/OSCPU/NutShell/blob/master/fpga/boot/pynq/BOOT.BIN

对果壳（NutShell）处理器支持——部署

- sig-RISC-V, openEuler / RISC-V
  - https://gitee.com/openeuler/RISC-V
- mirror/openeuler-sig-riscv
  - https://isrc.iscas.ac.cn/mirror/openeuler-sig-riscv
- sig-RISC-V
- 文档、镜像、源码、脚本等
- RPM repo
- SRPM repo

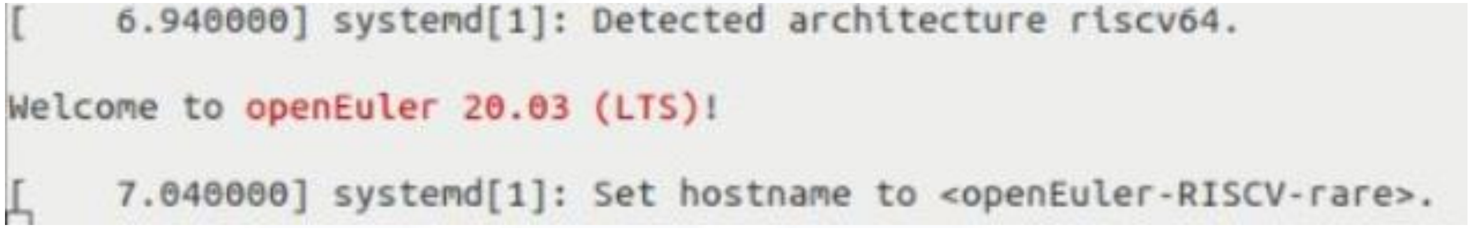


gitee 开源软件

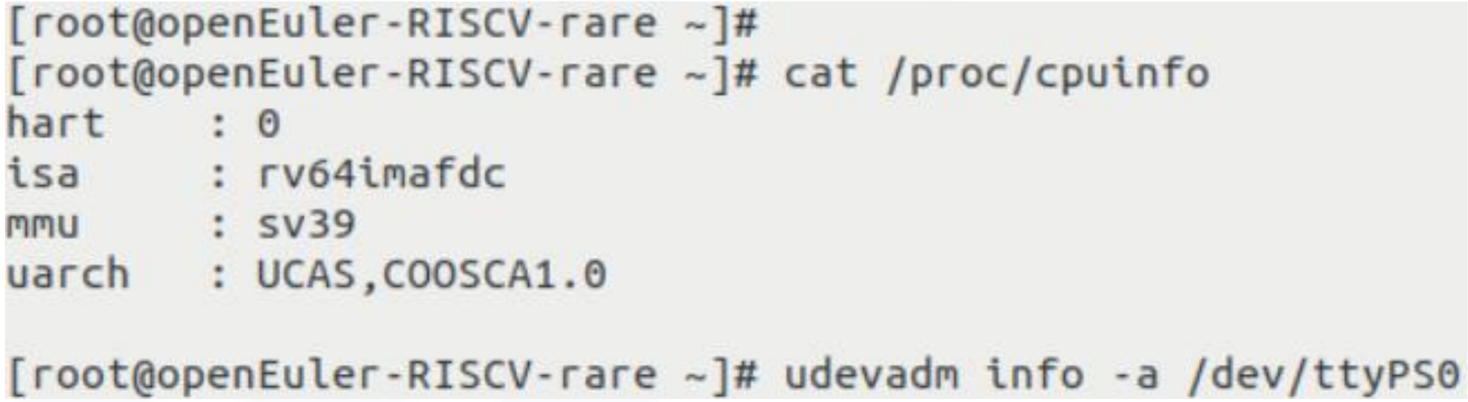
README.md

- RISC-V
  - 介绍
  - 动态
  - 参与RISC-V SIG的活动
  - 目录结构
  - RPM repo镜像仓库和...
  - 镜像下载
  - 源码仓
  - 列在openEuler社区维...
    - 虚拟机仿真平台
    - 硬件平台支持
    - 如何在NutShell(果...
    - 参与openEuler RIS...

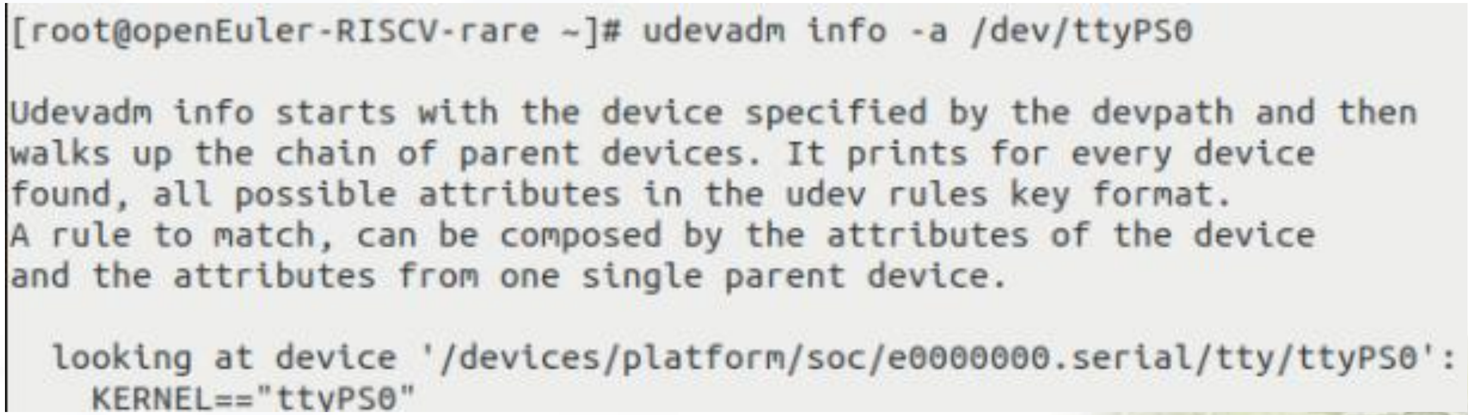
运行



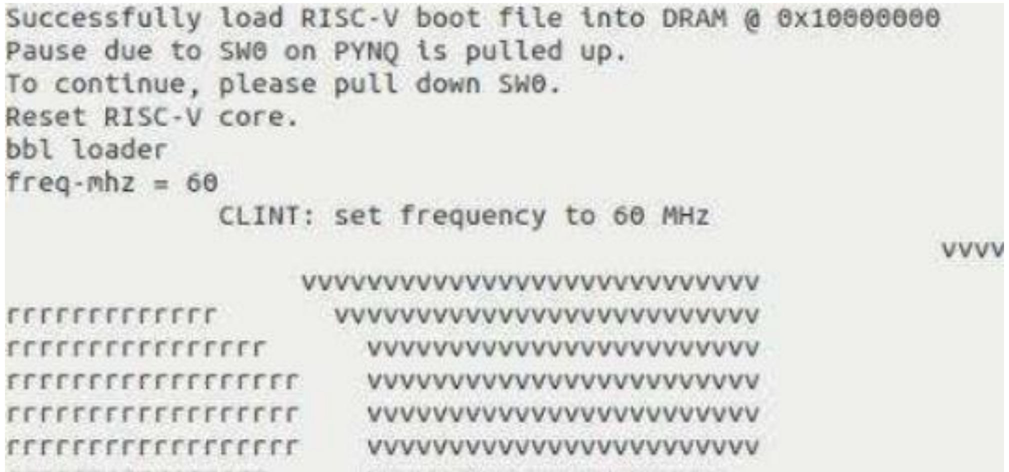
```
[    6.940000] systemd[1]: Detected architecture riscv64.

Welcome to openEuler 20.03 (LTS)!

[    7.040000] systemd[1]: Set hostname to <openEuler-RISCV-rare>.
```



```
[root@openEuler-RISCV-rare ~]#
[root@openEuler-RISCV-rare ~]# cat /proc/cpuinfo
hart     : 0
isa      : rv64imafdc
mmu      : sv39
uarch    : UCAS,COOSCA1.0

[root@openEuler-RISCV-rare ~]# udevadm info -a /dev/ttyPS0
```



```
[root@openEuler-RISCV-rare ~]# udevadm info -a /dev/ttyPS0

Udevadm info starts with the device specified by the devpath and then
walks up the chain of parent devices. It prints for every device
found, all possible attributes in the udev rules key format.
A rule to match, can be composed by the attributes of the device
and the attributes from one single parent device.

  looking at device '/devices/platform/soc/e0000000.serial/tty/ttyPS0':
    KERNEL=="ttyPS0"
```



```
Successfully load RISC-V boot file into DRAM @ 0x10000000
Pause due to SW0 on PYNQ is pulled up.
To continue, please pull down SW0.
Reset RISC-V core.
bbl loader
freq-mhz = 60
             CLINT: set frequency to 60 MHz
                                                         vvvv

                  vvvvvvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrr       vvvvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrr      vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrrrr    vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrrrr    vvvvvvvvvvvvvvvvvvvvvvvv
rrrrrrrrrrrrrrrrrr    vvvvvvvvvvvvvvvvvvvvvvvv
```

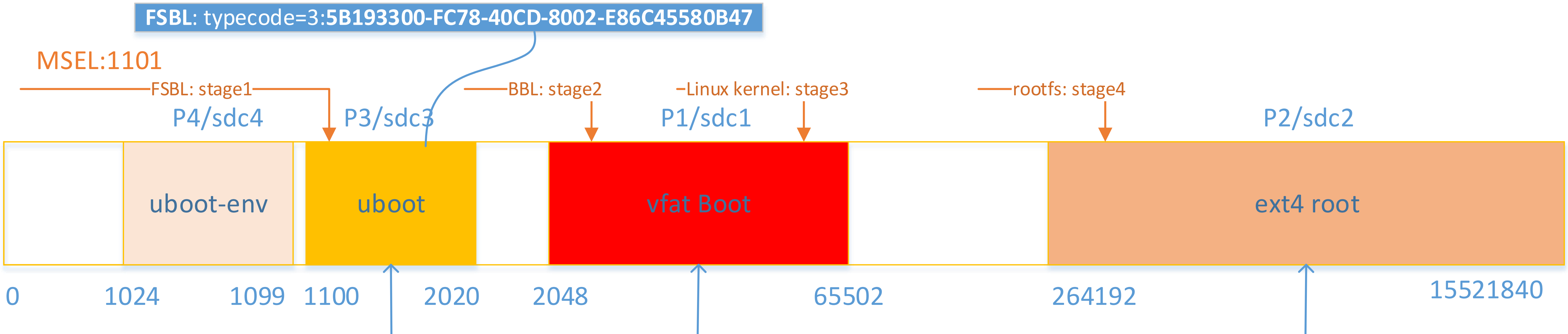
提纲

- 操作系统发行版构建的典型方法
- openEuler面向RISC-V操作系统构建
- 为SiFive定制操作系统经验与计算能力验证
- 操作系统构建的思考与难点分析

面向SiFive HiFive Unleashed的OS构建技术

- 总体构建步骤分为
  - 1）面向SiFive Unleashed开发版的bootloader构建
    - 即stage1、stage2、stage3三个引导阶段
    - Linux kernel：需要支持RISC-V Arc的补丁、需要支持SiFive Unleashed开发版的驱动补丁，没有后者的kernel只能在Qemu等模拟器上运行
    - 包括：必选u-boot、bbl、Linux kernel、fdt，和可选的initramfs
    - SD-Card GPT分区划分和布局安排、 uEnv.txt启动配置菜单
  - 2）面向RISC-V指令集架构的根分区与根文件系统
    - 即boot stage4
    - 构建Linux 根文件系统，复用ISCAS/RVOS V1.0的修剪版即可
  - 3）写入SD-Card，配置好开发版（插SD卡、电源、MSEL模式开关设置、网线、 Micro USB等），开机、串口通信交互等
  - 4）如果测试网络，准备一个支持DHCP IP分配服务的局域网

SiFive Unleashed开发板bootloader构建技术



```
dd if=u-boot.bin
455k
```

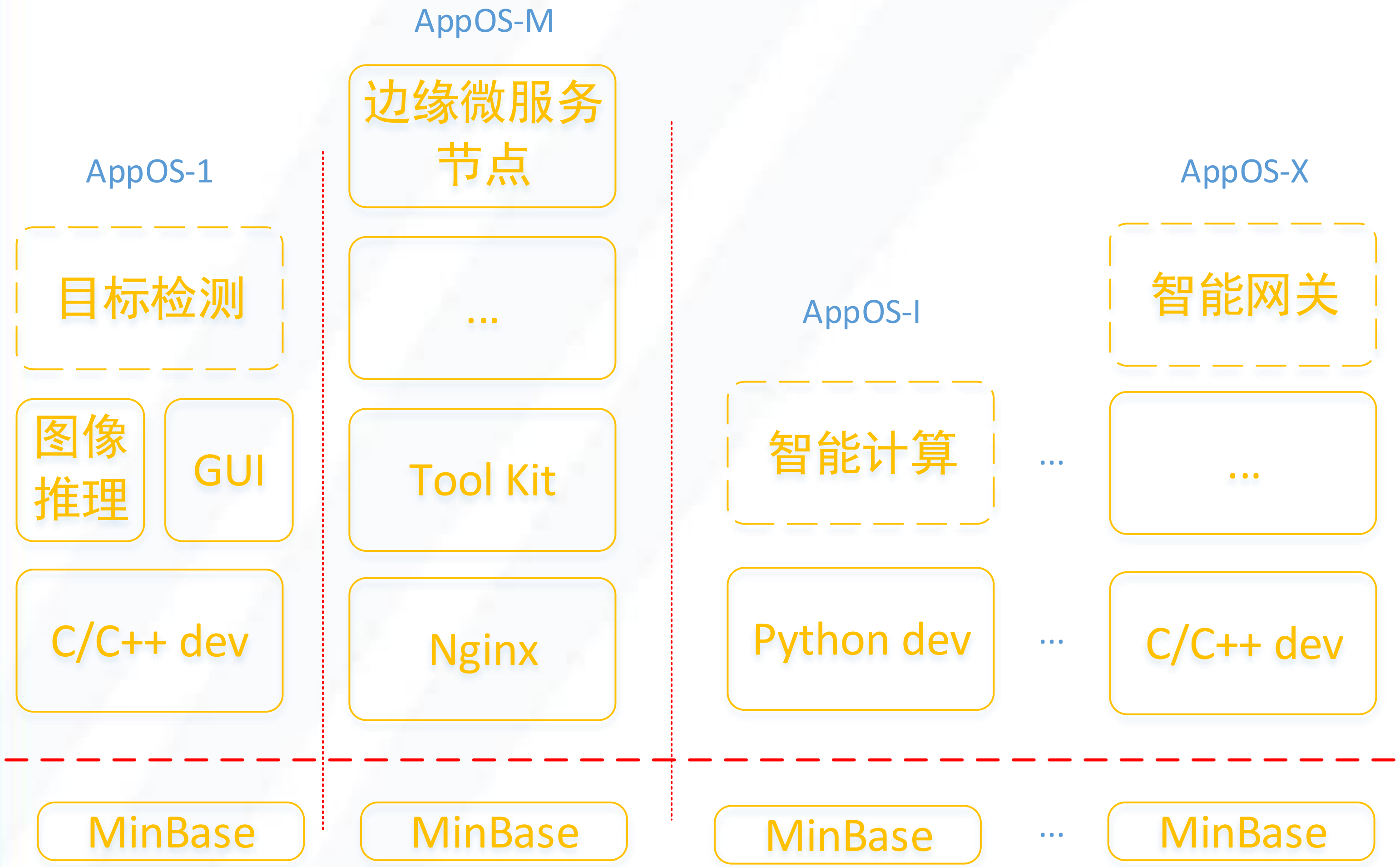
```
dd if=hifive_unleashed-vfat.part
15M/32M
```

```
tar -xf work/stage4-disk-min0-nginx-work-cleandnfcache.tar -C .
```

Bootloader: SD Card Partition Layout for SiFive HiFive Unleashed boot on sd

- 如图所示：
  - 实验用8G的Micro SD Card的分区布局
  - 引导顺序，和相应的文件
  - 注意：一定要设置MSEL为1101模式

高度可定制性



AppOS-1

目标检测

图像推理

GUI

C/C++ dev

MinBase

AppOS-M

边缘微服务节点

…

Tool Kit

Nginx

MinBase

AppOS-I

智能计算

Python dev

MinBase

…

AppOS-X

智能网关

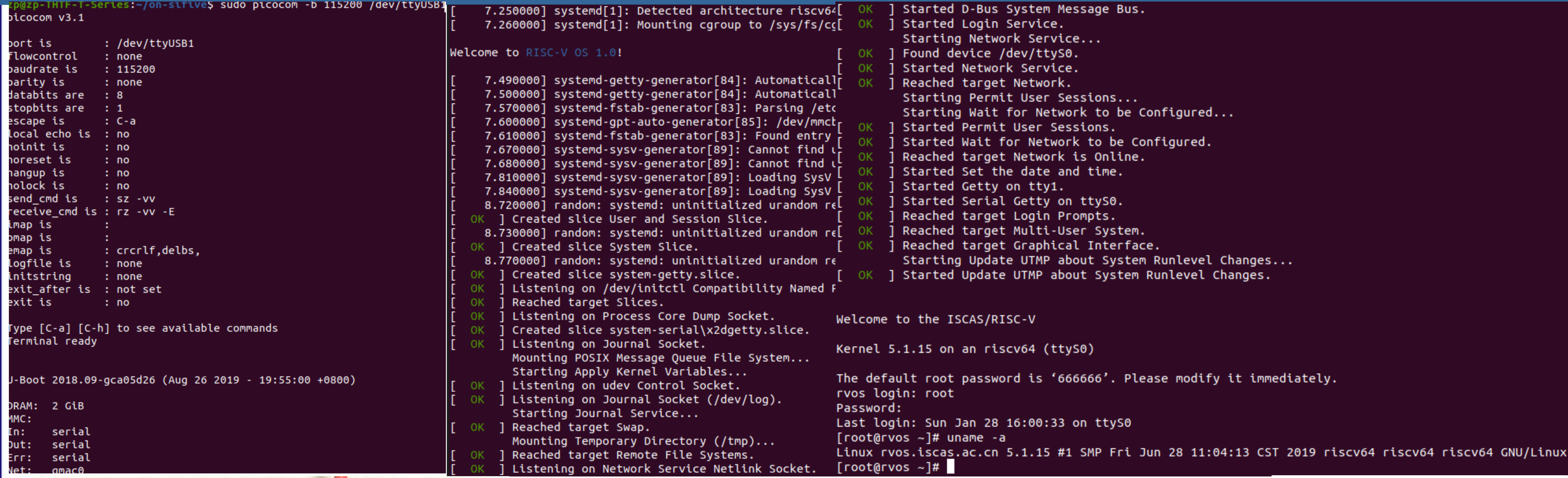
…

C/C++ dev

MinBase

- **模式：MinBase+垂直扩展**
- **MinBase:**
  - 是最小可运行的**RVOS**操作系统
  - 驱动**RISC-V**指令集架构和设备稳定运行
- **垂直领域版:**
  - **MinBase**集成专用领域应用
  - 面向专用领域做优化
  - 互不交叉，垂直发展

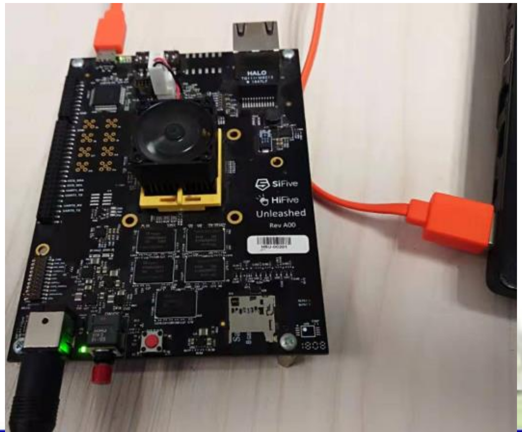
计算能力——可以引导通用Linux操作系统



```
zp@zp-THTF-T-Series:~/on-sifive$ sudo picocom -b 115200 /dev/ttyUSB1
picocom v3.1

port is        : /dev/ttyUSB1
flowcontrol    : none
baudrate is    : 115200
parity is      : none
databits are   : 8
stopbits are   : 1
escape is      : C-a
local echo is  : no
noinit is      : no
noreset is     : no
hangup is      : no
nolock is      : no
send_cmd is    : sz -vv
receive_cmd is : rz -vv -E
imap is        :
omap is        :
emap is        : crcrlf,delbs,
logfile is     : none
initstring     : none
exit_after is  : not set
exit is        : no

Type [C-a] [C-h] to see available commands
Terminal ready


U-Boot 2018.09-gca05d26 (Aug 26 2019 - 19:55:00 +0800)

DRAM:  2 GiB
MMC:
In:    serial
Out:   serial
Err:   serial
Net:   gmac0
```

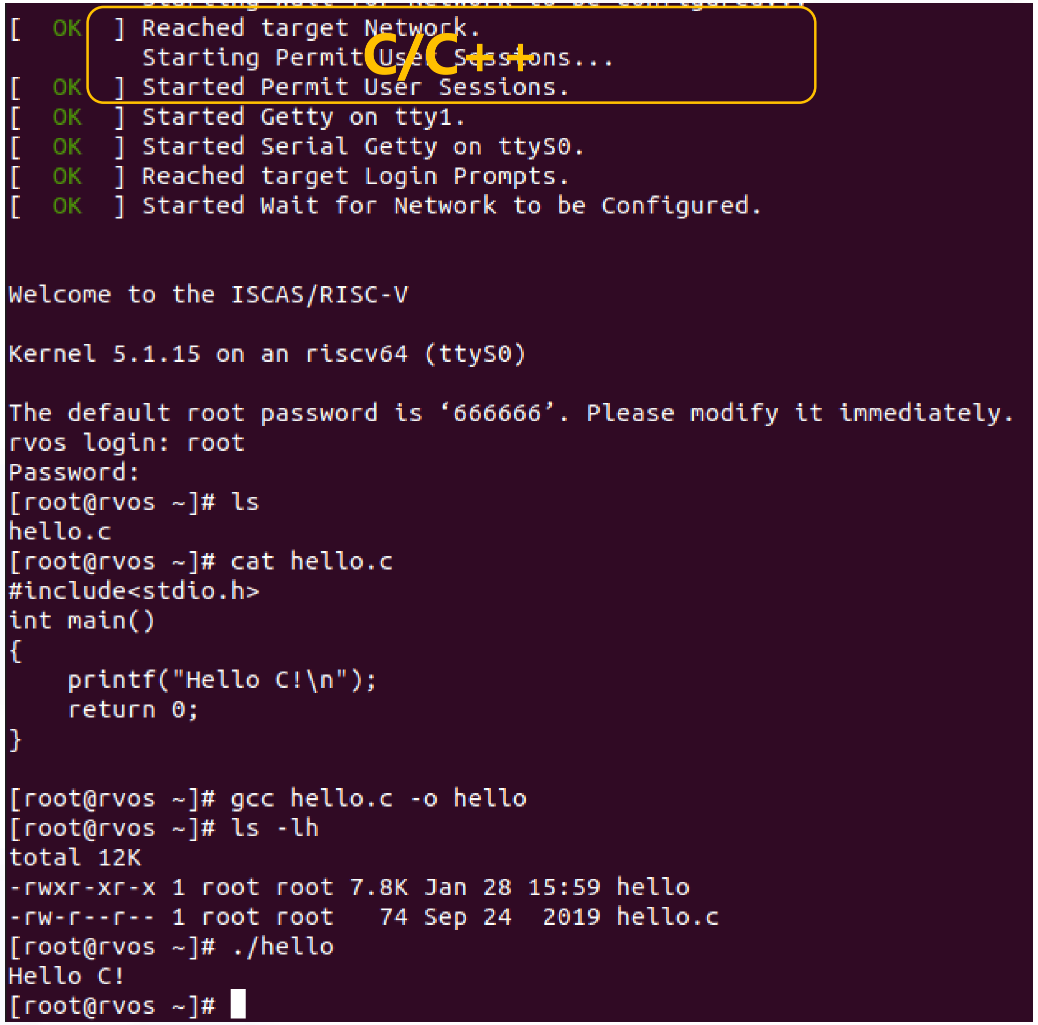
```
[    7.250000] systemd[1]: Detected architecture riscv64
[    7.260000] systemd[1]: Mounting cgroup to /sys/fs/cg

Welcome to RISC-V OS 1.0!

[    7.490000] systemd-getty-generator[84]: Automaticall
[    7.500000] systemd-getty-generator[84]: Automaticall
[    7.570000] systemd-fstab-generator[83]: Parsing /etc
[    7.600000] systemd-gpt-auto-generator[85]: /dev/mmcb
[    7.610000] systemd-fstab-generator[83]: Found entry
[    7.670000] systemd-sysv-generator[89]: Cannot find u
[    7.680000] systemd-sysv-generator[89]: Cannot find u
[    7.810000] systemd-sysv-generator[89]: Loading SysV
[    7.840000] systemd-sysv-generator[89]: Loading SysV
[    8.720000] random: systemd: uninitialized urandom re
[  OK  ] Created slice User and Session Slice.
[    8.730000] random: systemd: uninitialized urandom re
[  OK  ] Created slice System Slice.
[    8.770000] random: systemd: uninitialized urandom re
[  OK  ] Created slice system-getty.slice.
[  OK  ] Listening on /dev/initctl Compatibility Named P
[  OK  ] Reached target Slices.
[  OK  ] Listening on Process Core Dump Socket.
[  OK  ] Created slice system-serial\x2dgetty.slice.
[  OK  ] Listening on Journal Socket.
         Mounting POSIX Message Queue File System...
         Starting Apply Kernel Variables...
[  OK  ] Listening on udev Control Socket.
[  OK  ] Listening on Journal Socket (/dev/log).
         Starting Journal Service...
[  OK  ] Reached target Swap.
         Mounting Temporary Directory (/tmp)...
[  OK  ] Reached target Remote File Systems.
[  OK  ] Listening on Network Service Netlink Socket.
```

```
[  OK  ] Started D-Bus System Message Bus.
[  OK  ] Started Login Service.
         Starting Network Service...
[  OK  ] Found device /dev/ttyS0.
[  OK  ] Started Network Service.
[  OK  ] Reached target Network.
         Starting Permit User Sessions...
         Starting Wait for Network to be Configured...
[  OK  ] Started Permit User Sessions.
[  OK  ] Started Wait for Network to be Configured.
[  OK  ] Reached target Network is Online.
[  OK  ] Started Set the date and time.
[  OK  ] Started Getty on tty1.
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Reached target Login Prompts.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Started Update UTMP about System Runlevel Changes.


Welcome to the ISCAS/RISC-V

Kernel 5.1.15 on an riscv64 (ttyS0)

The default root password is ‘666666’. Please modify it immediately.
rvos login: root
Password:
Last login: Sun Jan 28 16:00:33 on ttyS0
[root@rvos ~]# uname -a
Linux rvos.iscas.ac.cn 5.1.15 #1 SMP Fri Jun 28 11:04:13 CST 2019 riscv64 riscv64 riscv64 GNU/Linux
[root@rvos ~]# 
```



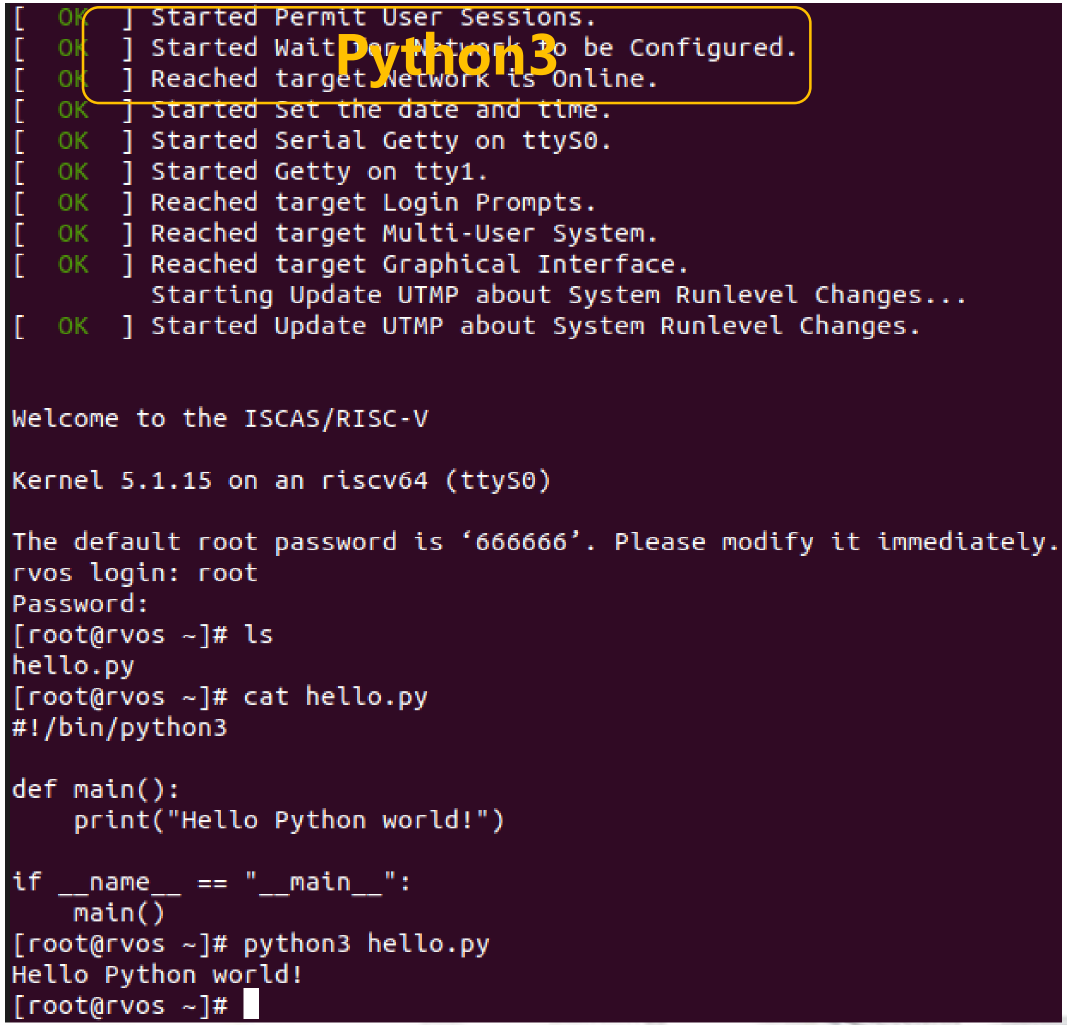
- 第01号LED灯开始闪烁心跳，表示Linux kernel已经引导。

计算能力——C/C++ RVOS-Python



C/C++

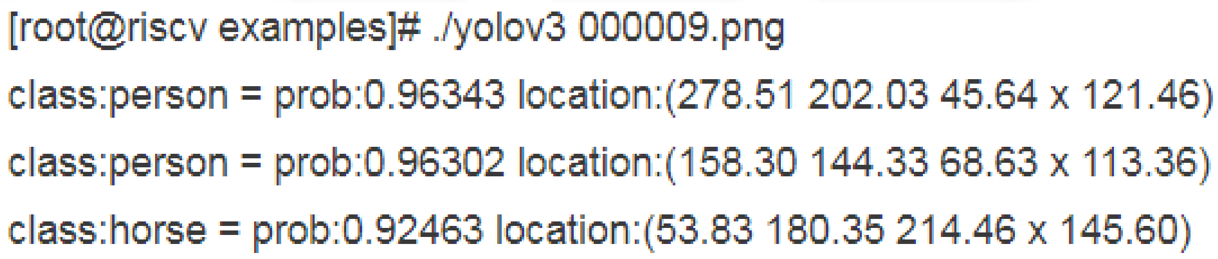
```
         Starting Wait for Network to be Configured...
[  OK  ] Reached target Network.
         Starting Permit User Sessions...
[  OK  ] Started Permit User Sessions.
[  OK  ] Started Getty on tty1.
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Reached target Login Prompts.
[  OK  ] Started Wait for Network to be Configured.


Welcome to the ISCAS/RISC-V

Kernel 5.1.15 on an riscv64 (ttyS0)

The default root password is ‘666666’. Please modify it immediately.
rvos login: root
Password:
[root@rvos ~]# ls
hello.c
[root@rvos ~]# cat hello.c
#include<stdio.h>
int main()
{
    printf("Hello C!\n");
    return 0;
}

[root@rvos ~]# gcc hello.c -o hello
[root@rvos ~]# ls -lh
total 12K
-rwxr-xr-x 1 root root 7.8K Jan 28 15:59 hello
-rw-r--r-- 1 root root   74 Sep 24  2019 hello.c
[root@rvos ~]# ./hello
Hello C!
[root@rvos ~]#
```



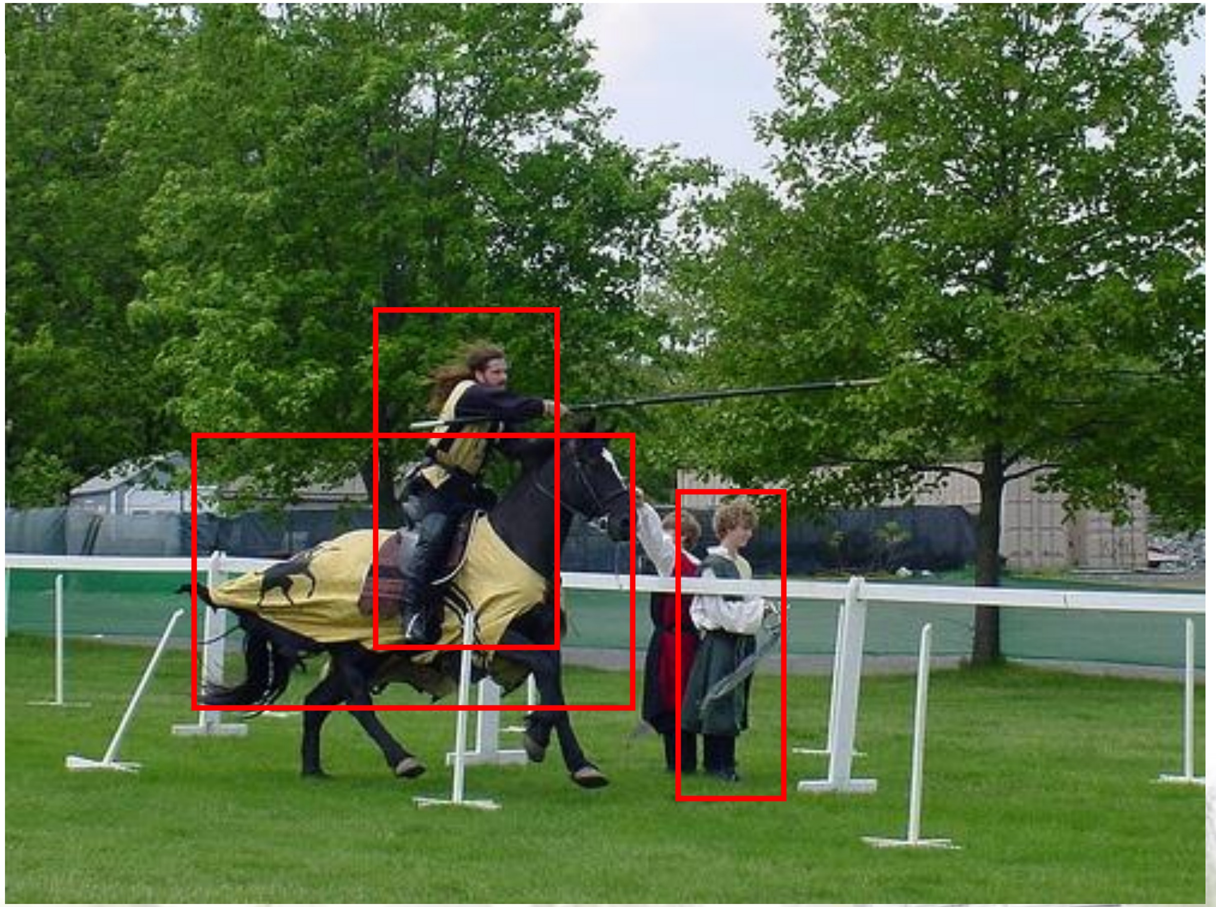
Python3

```
[  OK  ] Started Permit User Sessions.
[  OK  ] Started Wait for Network to be Configured.
[  OK  ] Reached target Network is Online.
[  OK  ] Started Set the date and time.
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Started Getty on tty1.
[  OK  ] Reached target Login Prompts.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Started Update UTMP about System Runlevel Changes.


Welcome to the ISCAS/RISC-V

Kernel 5.1.15 on an riscv64 (ttyS0)

The default root password is ‘666666’. Please modify it immediately.
rvos login: root
Password:
[root@rvos ~]# ls
hello.py
[root@rvos ~]# cat hello.py
#!/bin/python3

def main():
    print("Hello Python world!")

if __name__ == "__main__":
    main()
[root@rvos ~]# python3 hello.py
Hello Python world!
[root@rvos ~]#
```

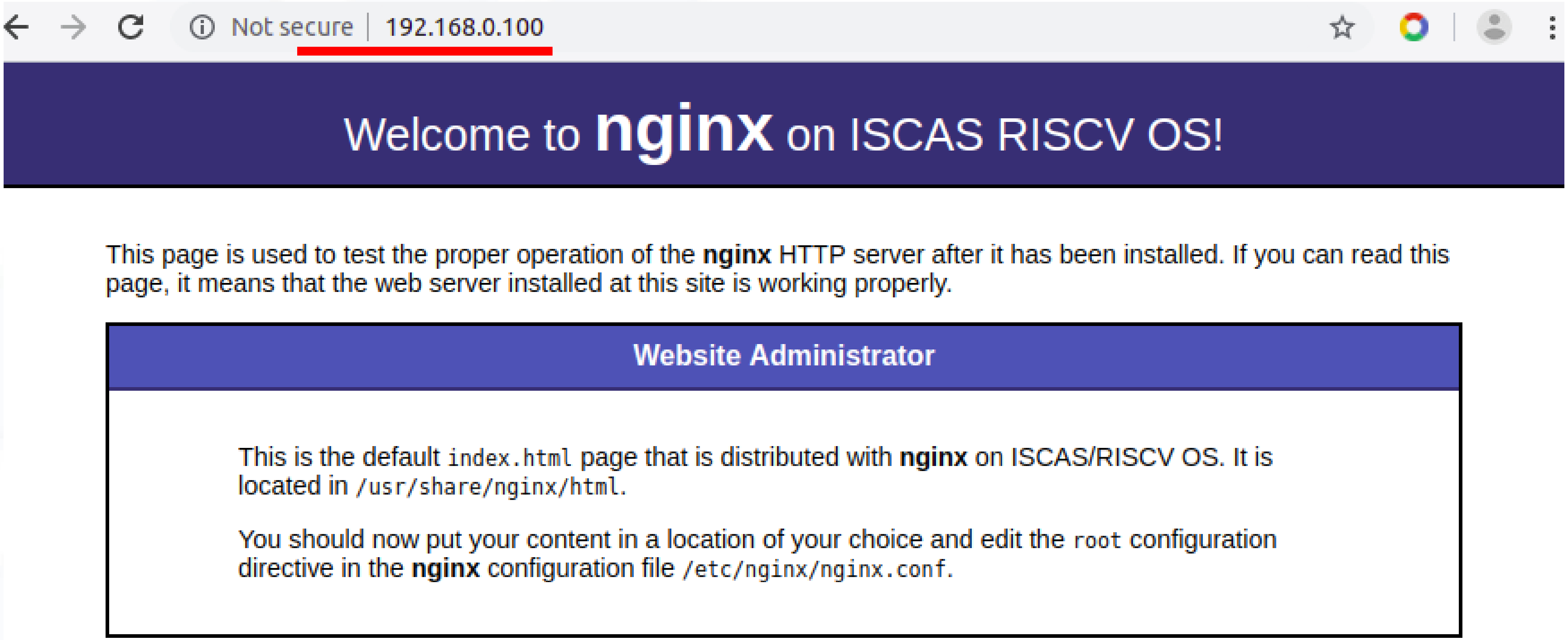
计算能力-AI软件栈



```
[root@riscv examples]# ./yolov3 000009.png
class:person = prob:0.96343 location:(278.51 202.03 45.64 x 121.46)
class:person = prob:0.96302 location:(158.30 144.33 68.63 x 113.36)
class:horse = prob:0.92463 location:(53.83 180.35 214.46 x 145.60)
```



计算能力-Nginx Micro Server



Not secure | 192.168.0.100

Welcome to **nginx** on ISCAS RISCV OS!

This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on ISCAS/RISCV OS. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX

提纲

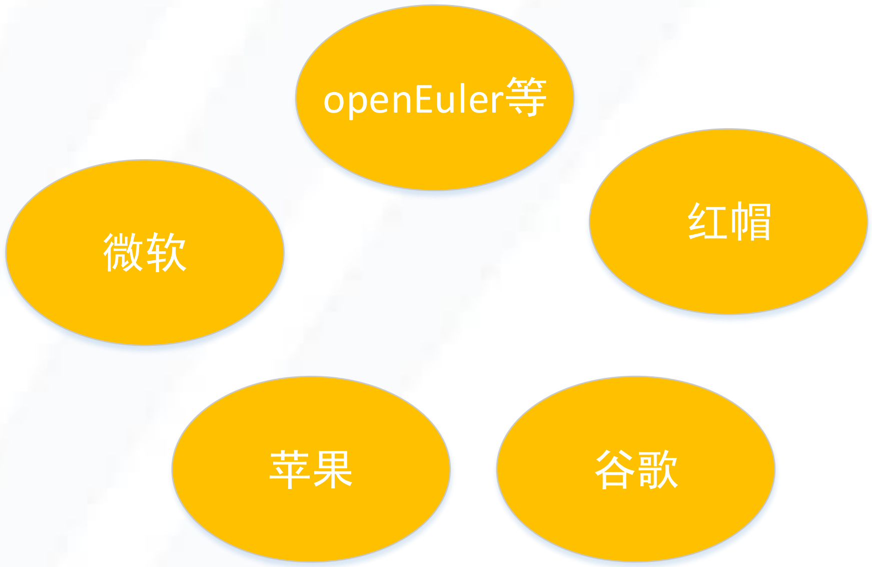
- 操作系统发行版构建的典型方法
- openEuler面向RISC-V操作系统构建
- 为SiFive定制操作系统经验与计算能力验证
- 操作系统构建的思考与难点分析

几个维度

- 历史包袱：即是否依赖已经形成的（培育的）用户行为惯性/习惯
  - 相关概念：生态、先验标准（比如MS office办公软件工具集）
  - 惯性/习惯变成了知识
- 技术生态是否构成闭环
  - 软件包依赖？
  - 成体系、自成体系？
  - 大而全、面面俱到、兼容一切、无限度开放集成、N套运行环境、N套编程语言？
- 软件栈架构是否稳固可控
  - 跟灵活多变、无限可定制的关系？
  - 专用系统，比较专用的系统（办公、娱乐、开发、游戏、计算）？
  - 短线的风筝飞出去就不管了、或者没法管了？

从几个维度分析不同操作系统

- 服务器操作系统
- 桌面操作系统
- 智能手机操作系统
- AIoT/物联网操作系统


openEuler等

红帽

微软

苹果

谷歌

开源社区蕴含着，囊括着最先进的技术？	是
开源技术是最前沿、最富有创新性的？	是
最先进的开源技术的集成是否是最好的产品？	否
最先进的开源技术的集成是否是好的产品？	否
只做产品不关注技术可以吗？	操作系统肯定是否

全球开源技术已经是最先进的技术富矿！
但是富矿不是挖出来就可以直接用的！
最先进的开源技术 + 最先进的产品打磨！

谢谢！