Линукс курс 2016/2017

Упознавање са buildroot алатом и U-boot-ом

ЦИЉ

Научити како да користите buildroot и U-boot.

ИСХОД

Након ове вежбе ћете моћи да:

- Генеришете toolchain са buildroot-ом
- Генеришете компелтан rfs и кернел са buildroot-ом
- Запишете кернел и rfs на трајну меморију

ПОСТАВКА

Позиционирајте се у директоријум `~/linux-kernel-labs/src` и затим од полазне гране `master` направите нову грану `dan09` и позиционирајте се у исту:

```
git checkout master
git checkout -b dan09
```

Преузмите изворни код buildroot-а следећим командама на путању `~/linux-kernel-labs/src/`:

`git clone git://git.buildroot.net/buildroot` (клонира гит репозиторијум)

```
cd buildroot
```

`git checkout 7afacef` (преузети одређену верзију са репозиторијума)

Преименујте добијени директоријум са buildroot-ом (нпр. у `buildroot_tc`, за генерисање toolchain-а), па поновите исту процедуру за генерисање комплетног rfs-а са кернелом (преименовати нпр.у `buildroot_rfs`).

Преузмите изворни код U-boot-а са гит репозиторијума `git://git.denx.de/u-boot.git` на путању `~/linux-kernel-labs/src/`.

ГЕНЕРИСАЊЕ TOOLCHAIN-А

Почните од минималистичке конфигурације buildroot-а (`make allnoconfig`). Коришћењем `menuconfig` алата, изаберите следећу конфигурацију:

- Target options
  - target Architecture -> ARM(little endian)
  - Target Binary Format (ELF)
  - Target Architecture Variant (cortex-A7)
  - Target ABI (EABIhf)
  - Floating point strategy (NEON/VFPv4)
  - ARM instruction set (ARM)
- Build options
  - Enable compiler cache
- Toolchain
  - Toolchain type (Buildroot toolchain)
  - Kernel Headers (Linux 4.7.x kernel headers)
  - C library (uClibc/ng)
  - Enable RPC support
  - Enable WCHAR support

  - Enable toolchain locale
  - Thread library debugging
  - Enable stack protection
  - Compile and install uClibc utilities
  - Enable C++ support
  - Enable compiler tls support
  - Build cross gdb for host
  - Purge unwanted locales
  - Enable MMU support
- System configuration
  - Root FS skeleton (custom target skeleton)
  - custom target skeleton path ($(TOPDIR)/skel)
  - Init system (None)
- Target packages
  - **Isključiti** Busybox

У `buildroot_tc` директоријуму направите угњеждене директоријуме `skel/usr` и затим покрените превођење. Резултат превођења ће се налазити на путањи `output/host`. По завршетку превођења генерисани toolchain тестирајте превођењем једноставног Hello world програма и његовим покретањем на циљној платформи. НАПОМЕНА: пошто се toolchain не налази на некој од уобичајених путања, потребно је експортовати `PATH` и `LD_LIBRARY_PATH` варијабле на следећи начин:

```
export PATH=<putanja do toolchain-a>/bin:$PATH

export LD_LIBRARY_PATH=<putanja do toolchain-a>/lib:$LD_LIBRARY_PATH
```

Линукс курс 2016/2017

ГЕНЕРИСАЊЕ КОМПЛЕТНОГ RFS-А СА КЕРНЕЛОМ

У овом кораку треба искористити закрпу која се налази на путањи `~/linux-kernel-labs/bootloader/rpi-2-b/src/0001-RPI2-customization-for-RTRK-EMBEDDED-LINUX-PROGRAMMI.patch`. За примену закрпе искористите следећу команду након позиционирања у buildroot директоријум:

```
git apply <putanja do zakrpe>
```

Након примене закрпе примените подразумевану конфигурацију за RPI командом `make raspberrypi2_defconfig`. Коришћењем `menuconfig` алата прегледајте које су опције укључене у подразумеваној конфигурацији и након тога покрените превођење.

КОНФИГУРАЦИЈА И ПРЕВОЂЕЊЕ U-BOOT-А

Пре постављања конфигурације и покретања превођења кернела је потребно подесити одговарајуће окружење за унакрсно превођење (`ARCH` и `CROSS_COMPILE`). За конфигурацију U-boot-а искористите подразумевану конфигурацију за RPI платформу. Коришћењем `menuconfig` алата проверите да ли је укључена подршка за ext и fat системе датотека и, ако није, укључите је. Такође, додајте суфикс верзије (`LOCALVERSION`) како би се при покретању добила потврда да је покренута нова верзија U-boot-а. Покрените превођење командом `make all`. Резултат превођења `u-boot.bin` пребаците у коренски директоријум tftp послужиоца.

ЗАМЕНА U-BOOT-А НА МЕМОРИЈСКОЈ КАРТИЦИ

Пронађите на којој партицији се налази `u-boot.bin` на меморијској картици. За излиставање свих партиција користите команду `mmc part`, а за преглед садржаја партиције `ext4ls` или `fatls` у зависности од система датотека који се налази на датој партицији.

Када је постојећи `u-boot.bin` пронађен, потребно га је заменити новом верзијом. Учитајте нову верзију `u-boot.bin`-а у меморију на произвољну адресу

(нпр. 0x1000000). Запишите `u-boot.bin` из меморије на SD картицу одговарајућом функцијом (`ext4write` или `fatwrite`). После овог корака рестартујте RPI и уверите се да се покреће нова верзија U-boot-a.

ЗАМЕНА КЕРНЕЛА И RFS-А НА МЕМОРИЈСКОЈ КАРТИЦИ

Након завршетка генерисања комплетног rfs-а са кернелом, генерисане резултате упишите на меморијску картицу и то:

- Запишите `zImage` и `.dtb` датотеке на исту партицију и на исти начин као и `u-boot.bin`
- Препишите комплетну партицију која садржи rfs на картици командом `mmc write` са генерисаном сликом rfs-а `rootfs.ext2` (**претходно се добро упознајте са коришћењем `mmc write` команде, јер погрешно коришћење може да доведе до губитка читаве партиције. Обратите пажњу на величину блока коју уз остале детаље можете сазнати командом `help mmc` имајући у виду да се сви параметри, поред адреса, команди `mmc write` задају у броју блокова у хексадецималном формату**.

Покрените из U-boot-а нови кернел и rfs са меморијске картице командом `run mmc_boot`.

САЧУВАЈТЕ СВЕ ИЗМЕНЕ

Да бисте потврдили и сачували све измене, најбоље је да их додате, а потом и локално комитујете на GIT, док сте позиционирани у директоријум репозиторијума који је мењан, `~/linux-kernel-labs`:

```
git add -A

git commit -as -m "dan09 zavrsen"
```

Да би измене постале видљиве и у репозиторијуму на серверу, потребно би још било урадити нпр. `git push`, али то у овом случају није неопходно нити имамо неопходна права за то.