

# Lab 3

---

## Objective

To implement a full embedded Linux boot process using QEMU with:

- U-Boot (bootloader)
- Linux kernel compiled for ARM
- Initramfs with custom `/init` script
- Root filesystem (`rootfs.ext4`)

Expected Boot Flow:

**U-Boot → Kernel → Initramfs → switch\_root → Rootfs**

---

## Environment Setup

Installed required tools:

```
➔ ~/I/a/Lab3 on main • sudo apt-get install build-essential git qemu-system-arm gcc-arm-linux-gnueabi \
libncurses-dev flex bison libssl-dev device-tree-compiler swig python3-dev python3-setuptools
[sudo] password for anas:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
bison is already the newest version (2:3.8.2+dfsg-1build1).
build-essential is already the newest version (12.9ubuntu3).
flex is already the newest version (2.6.4-8build2).
gcc-arm-linux-gnueabi is already the newest version (4:11.2.0-1ubuntu1).
git is already the newest version (1:2.34.1-1ubuntu1.12).
libncurses-dev is already the newest version (6.3-2ubuntu0.1).
libncurses-dev set to manually installed.
python3-dev is already the newest version (3.10.6-1-22.04.1).
python3-setuptools is already the newest version (59.6.0-1.2ubuntu0.22.04.2).
qemu-system-arm is already the newest version (1:6.2+dfsg-2ubuntu6.26).
The following additional packages will be installed:
  g++-11-arm-linux-gnueabi libssl3 libssl3:i386 libstdc++-11-dev-armhf-cross swig4.0
Suggested packages:
  gcc-11-doc libssl-doc swig-doc swig-examples swig4.0-examples swig4.0-doc
The following NEW packages will be installed:
  device-tree-compiler g++-11-arm-linux-gnueabi g++-arm-linux-gnueabi libstdc++-11-dev-armhf-cross swig swig4.0
The following packages will be upgraded:
  libssl-dev libssl3 libssl3:i386
3 upgraded, 6 newly installed, 0 to remove and 335 not upgraded.
Need to get 19.9 MB of archives.
After this operation, 53.2 MB of additional disk space will be used.
```

## Building U-Boot

```
git clone https://github.com/u-boot/u-boot.git
cd u-boot
git checkout v2022.01

export ARCH=arm
export CROSS_COMPILE=arm-linux-gnueabi-
make vexpress_ca9x4_defconfig
make
```

```

-> ~/I/a/Lab3 on main ◦ git clone https://gitlab.denx.de/u-boot/u-boot.git
Cloning into 'u-boot'...
warning: redirecting to https://source.denx.de/u-boot/u-boot.git/
remote: Enumerating objects: 1061724, done.
remote: Counting objects: 100% (6486/6486), done.
remote: Compressing objects: 100% (2379/2379), done.
remote: Total 1061724 (delta 4255), reused 6077 (delta 4031), pack-reused 1055238 (from 1)
Receiving objects: 100% (1061724/1061724), 234.17 MiB | 862.00 KiB/s, done.
Resolving deltas: 100% (883497/883497), done.
Updating files: 100% (36187/36187), done.
-> ~/I/a/Lab3 on main ◦ cd u-boot
-> ~/I/a/L/u-boot on master ◦ git checkout v2023.01
Updating files: 100% (30881/30881), done.
Note: switching to 'v2023.01'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

-> ~/I/a/Lab3 on main ◦ export ARCH=arm
-> ~/I/a/Lab3 on main ◦ export CROSS_COMPILE=arm-linux-gnueabi-
-> ~/I/a/Lab3 on main ◦
AR      lib/built-in.o
AR      fs/ubifs/built-in.o
AR      fs/built-in.o
LD      u-boot
OBJCOPY u-boot.srec
OBJCOPY u-boot-nodtb.bin
SYM      u-boot.sym
COPY     u-boot.bin
===== WARNING =====
CONFIG_OF_EMBED is enabled. This option should only
be used for debugging purposes. Please use
CONFIG_OF_SEPARATE for boards in mainline.
See doc/develop/devicetree/control.rst for more info.
===== WARNING =====
This board does not use CONFIG_TIMER (Driver Model
for Timer drivers). Please update the board to use
CONFIG_TIMER before the v2023.01 release. Failure to
update by the deadline may result in board removal.
See doc/develop/driver-model/migration.rst for more info.
===== WARNING =====
This board does not use CONFIG_DM_SERIAL (Driver Model
for Serial drivers). Please update the board to use
CONFIG_DM_SERIAL before the v2023.04 release. Failure to
update by the deadline may result in board removal.
See doc/develop/driver-model/migration.rst for more info.
=====
CFGCHK  u-boot.cfg
OFCHK   .config
-> ~/I/a/L/u-boot on 62e2ad1ceaf

```

## Building Linux Kernel

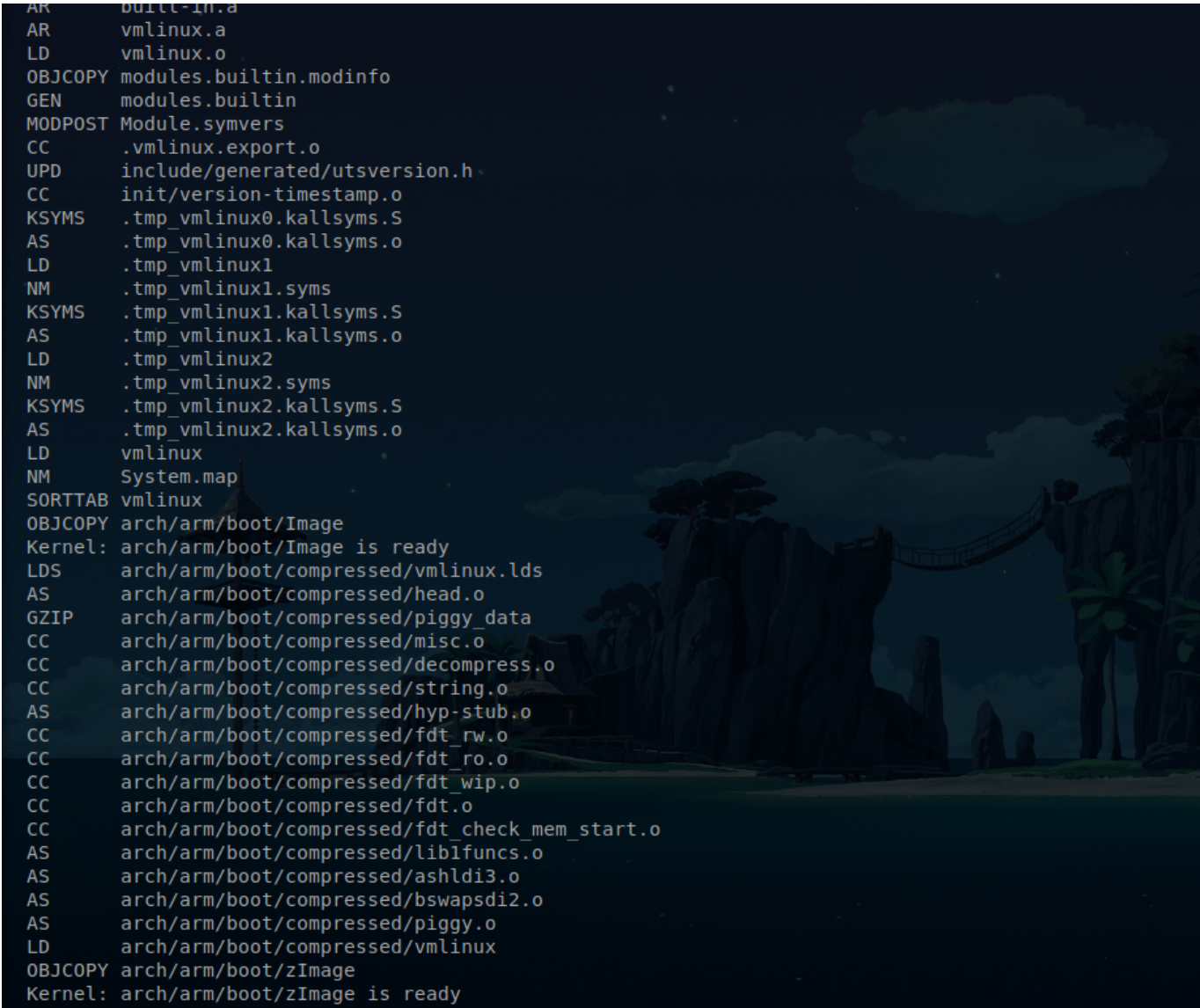
```

git clone --depth=1
https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git
cd linux

export ARCH=arm
export CROSS_COMPILE=arm-linux-gnueabi-
make vexpress_defconfig
make zImage

```

```
make modules
make dtbs
```



```
AR      builtin.a
AR      vmlinux.a
LD      vmlinux.o
OBJCOPY modules.builtin.modinfo
GEN      modules.builtin
MODPOST Module.symvers
CC      .vmlinux.export.o
UPD      include/generated/utsversion.h
CC      init/version-timestamp.o
KSYMS    .tmp_vmlinux0.kallsyms.S
AS      .tmp_vmlinux0.kallsyms.o
LD      .tmp_vmlinux1
NM      .tmp_vmlinux1.syms
KSYMS    .tmp_vmlinux1.kallsyms.S
AS      .tmp_vmlinux1.kallsyms.o
LD      .tmp_vmlinux2
NM      .tmp_vmlinux2.syms
KSYMS    .tmp_vmlinux2.kallsyms.S
AS      .tmp_vmlinux2.kallsyms.o
LD      vmlinux
NM      System.map
SORTTAB vmlinux
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
LDS      arch/arm/boot/compressed/vmlinux.lds
AS      arch/arm/boot/compressed/head.o
GZIP     arch/arm/boot/compressed/piggy_data
CC      arch/arm/boot/compressed/misc.o
CC      arch/arm/boot/compressed/decompress.o
CC      arch/arm/boot/compressed/string.o
AS      arch/arm/boot/compressed/hyp-stub.o
CC      arch/arm/boot/compressed/fdt_rw.o
CC      arch/arm/boot/compressed/fdt_ro.o
CC      arch/arm/boot/compressed/fdt_wip.o
CC      arch/arm/boot/compressed/fdt.o
CC      arch/arm/boot/compressed/fdt_check_mem_start.o
AS      arch/arm/boot/compressed/lib1funcs.o
AS      arch/arm/boot/compressed/ashldi3.o
AS      arch/arm/boot/compressed/bswapsdi2.o
AS      arch/arm/boot/compressed/piggy.o
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
```

## Creating Initramfs with BusyBox

```
wget https://busybox.net/downloads/busybox-1.36.0.tar.bz2
tar -xvf busybox-1.36.0.tar.bz2
cd busybox-1.36.0

export ARCH=arm
export CROSS_COMPILE=arm-linux-gnueabi-
make defconfig
make
make install
```



```
./_install//usr/sbin/loadfont -> ../../bin/busybox
./_install//usr/sbin/lpd -> ../../bin/busybox
./_install//usr/sbin/mim -> ../../bin/busybox
./_install//usr/sbin/nanddump -> ../../bin/busybox
./_install//usr/sbin/nandwrite -> ../../bin/busybox
./_install//usr/sbin/nbd-client -> ../../bin/busybox
./_install//usr/sbin/nologin -> ../../bin/busybox
./_install//usr/sbin/ntpd -> ../../bin/busybox
./_install//usr/sbin/partprobe -> ../../bin/busybox
./_install//usr/sbin/popmaildir -> ../../bin/busybox
./_install//usr/sbin/powertop -> ../../bin/busybox
./_install//usr/sbin/rdate -> ../../bin/busybox
./_install//usr/sbin/rdev -> ../../bin/busybox
./_install//usr/sbin/readahead -> ../../bin/busybox
./_install//usr/sbin/readprofile -> ../../bin/busybox
./_install//usr/sbin/remove-shell -> ../../bin/busybox
./_install//usr/sbin/rtcwake -> ../../bin/busybox
./_install//usr/sbin/seedrng -> ../../bin/busybox
./_install//usr/sbin/sendmail -> ../../bin/busybox
./_install//usr/sbin/setfont -> ../../bin/busybox
./_install//usr/sbin/setlogcons -> ../../bin/busybox
./_install//usr/sbin/svlogd -> ../../bin/busybox
./_install//usr/sbin/telnetd -> ../../bin/busybox
./_install//usr/sbin/tftpd -> ../../bin/busybox
./_install//usr/sbin/ubiattach -> ../../bin/busybox
./_install//usr/sbin/ubidetach -> ../../bin/busybox
./_install//usr/sbin/ubimkvol -> ../../bin/busybox
./_install//usr/sbin/ubirename -> ../../bin/busybox
./_install//usr/sbin/ubirmvol -> ../../bin/busybox
./_install//usr/sbin/ubirsvol -> ../../bin/busybox
./_install//usr/sbin/ubiupdatevol -> ../../bin/busybox
./_install//usr/sbin/udhcpd -> ../../bin/busybox
```

-----  
You will probably need to make your busybox binary  
setuid root to ensure all configured applets will  
work properly.  
-----

~> ~/I/a/L/busybox on 1a64f6a20

Wrote a `/init` script inside `initramfs/`:

```
#!/bin/sh
mount -t proc none /proc
mount -t sysfs none /sys
mount -t devtmpfs none /dev

mkdir /newroot
mount -t ext4 /dev/mmcbk0p2 /newroot
exec switch_root /newroot /sbin/init
```

Packed initramfs:

```
cd initramfs
find . -print0 | cpio --null -ov --format=newc | gzip -9 >
../initramfs.cpio.gz
```

```
> ~/I/a/Lab3 on main ◦ ls -l initramfs/
total 32
drwxrwxr-x 2 anas anas 4096 May 10 02:48 bin
drwxrwxr-x 2 anas anas 4096 May 10 02:35 dev
drwxrwxr-x 2 anas anas 4096 May 10 02:35 etc
-rwxrwxr-x 1 anas anas 178 May 10 02:49 init
lrwxrwxrwx 1 anas anas 11 May 10 02:48 linuxrc -> bin/busybox
drwxrwxr-x 2 anas anas 4096 May 10 02:35 proc
drwxrwxr-x 2 anas anas 4096 May 10 02:48 sbin
drwxrwxr-x 2 anas anas 4096 May 10 02:35 sys
drwxrwxr-x 4 anas anas 4096 May 10 02:48 usr
> ~/I/a/Lab3 on main ◦
```

## Root Filesystem (rootfs.ext4)

```
mkdir -p rootfs/{bin,sbin,etc,proc,sys,usr/{bin,sbin},dev,tmp,home}
cp -a busybox-1.36.0/_install/* rootfs/
sudo mknod -m 666 rootfs/dev/console c 5 1
sudo mknod -m 666 rootfs/dev/null c 1 3
```

Added:

### /etc/inittab

```
::sysinit:/etc/init.d/rcS
::respawn:/bin/sh
```

### /etc/init.d/rcS

```
#!/bin/sh
echo "[OK] rcS script started."
mount -t proc none /proc
mount -t sysfs none /sys
```

```

Preview Readme.md  rcS  X  ...

rootfs > etc > init.d > rcS
1  #!/bin/sh
2  echo "[OK] rcS script started."
3  mount -t proc none /proc
4  mount -t sysfs none /sys

~> ~/I/a/Lab3 on main ◦ ls -l mnt/etc/
total 8
drwxr-xr-x 2 root root 4096 May 10 03:06 init.d
-rw-r--r-- 1 root root  44 May 10 03:35 inittab
~> ~/I/a/Lab3 on main ◦

```

## Creating SD Image

```

dd if=/dev/zero of=sd.img bs=1M count=64
fdisk sd.img
# o, n p 1 +16M, n p 2 (default), t 1 c, w

```

```

~> ~/I/a/Lab3 on main ◦ dd if=/dev/zero of=sd.img bs=1M count=64
64+0 records in
64+0 records out
67108864 bytes (67 MB, 64 MiB) copied, 0.074721 s, 898 MB/s

```

```

LOOP=$(sudo losetup --find --show --partscan sd.img)
sudo mkfs.vfat -n BOOT ${LOOP}p1
sudo mkfs.ext4 -L ROOT ${LOOP}p2

```

## Filled Boot Partition:

```

mkimage -A arm -T ramdisk -C gzip -n "Initramfs" -d initramfs.cpio.gz
uInitrd

sudo mount ${LOOP}p1 mnt
sudo cp linux/arch/arm/boot/zImage mnt/
sudo cp linux/arch/arm/boot/dts/arm/vexpress-v2p-ca9.dtb mnt/vexpress.dtb
sudo cp uInitrd mnt/
sudo umount mnt

```

```

-> ~/I/a/Lab3 on main ◦ ls -lh mnt/
total 6.8M
-rwxr-xr-x 1 root root 1.1M May 10 03:42 uInitrd
-rwxr-xr-x 1 root root 14K May 10 03:42 vexpress.dtb
-rwxr-xr-x 1 root root 5.7M May 10 03:42 zImage
-> ~/I/a/Lab3 on main ◦

```

### Filled Rootfs Partition:

```

sudo mount ${LOOP}p2 mnt
sudo cp -a rootfs/* mnt/
sudo umount mnt
sudo losetup -d "$LOOP"

```

```

-> ~/I/a/Lab3 on main ◦ sudo cp linux/arch/arm/boot/dts/arm/vexpress-v2p-ca9.dtb /tmp/sd_mount/
-> ~/I/a/Lab3 on main ◦ sudo cp busybox/initramfs.cpio.gz /tmp/sd_mount/
-> ~/I/a/Lab3 on main ◦ ls
busybox linux sd.img task.txt u-boot
-> ~/I/a/Lab3 on main ◦ sudo umount /tmp/sd_mount
-> ~/I/a/Lab3 on main ◦ sudo losetup -d /dev/loop25
-> ~/I/a/Lab3 on main ◦

```

### Boot in QEMU with U-Boot

```

qemu-system-arm -M vexpress-a9 \
  -m 512M \
  -kernel u-boot/u-boot \
  -drive file=sd.img,format=raw,if=sd \
  -nographic

```

```
> ~/I/a/Lab3 on main ◦ qemu-system-arm -M vexpress-a9 \
    -m 512M \
    -kernel u-boot/u-boot \
    -drive file=sd.img,format=raw,if=sd \
    -nographic
```

```
Usage:
```

```
cp [.b, .w, .l, .q] source target count
Wrong Image Format for bootm command
ERROR: can't get kernel image!
=> fatload mmc 0:1 0x60000000 zImage
5940760 bytes read in 1058 ms (5.4 MiB/s)
=> fatload mmc 0:1 0x61000000 uInitrd
1080542 bytes read in 179 ms (5.8 MiB/s)
=> fatload mmc 0:1 0x62000000 vexpress.dtb
14329 bytes read in 12 ms (1.1 MiB/s)
=> setenv bootargs "console=ttyAMA0 loglevel=3 init=/init"
=> bootz 0x60000000 0x61000000 0x62000000
Kernel image @ 0x60000000 [ 0x000000 - 0x5aa618 ]
## Loading init Ramdisk from Legacy Image at 61000000 ...
   Image Name:   Initramfs
   Image Type:   ARM Linux RAMDisk Image (gzip compressed)
   Data Size:    1080478 Bytes = 1 MiB
   Load Address: 00000000
   Entry Point:  00000000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 62000000
   Booting using the fdt blob at 0x62000000
```

## Final Result: Booted Root Shell

```
Loading Ramdisk to 7falf000, end 7fb26c9e ... OK
Loading Device Tree to 7fal8000, end 7fale7f8 ... OK
```

```
Starting kernel ...
```

```
GIC CPU mask not found - kernel will fail to boot.
```

```
GIC CPU mask not found - kernel will fail to boot.
```

```
[OK] rcS script started.
```

```
/bin/sh: can't access tty; job control turned off
```

```
~ # ls
```

```
bin      etc      linuxrc  proc     sys      usr
dev      home    lost+found /sbin   tmp
```

```
~ #
```

THANK YOU for this course! We learned alot of cool things.

---