# U-Boot

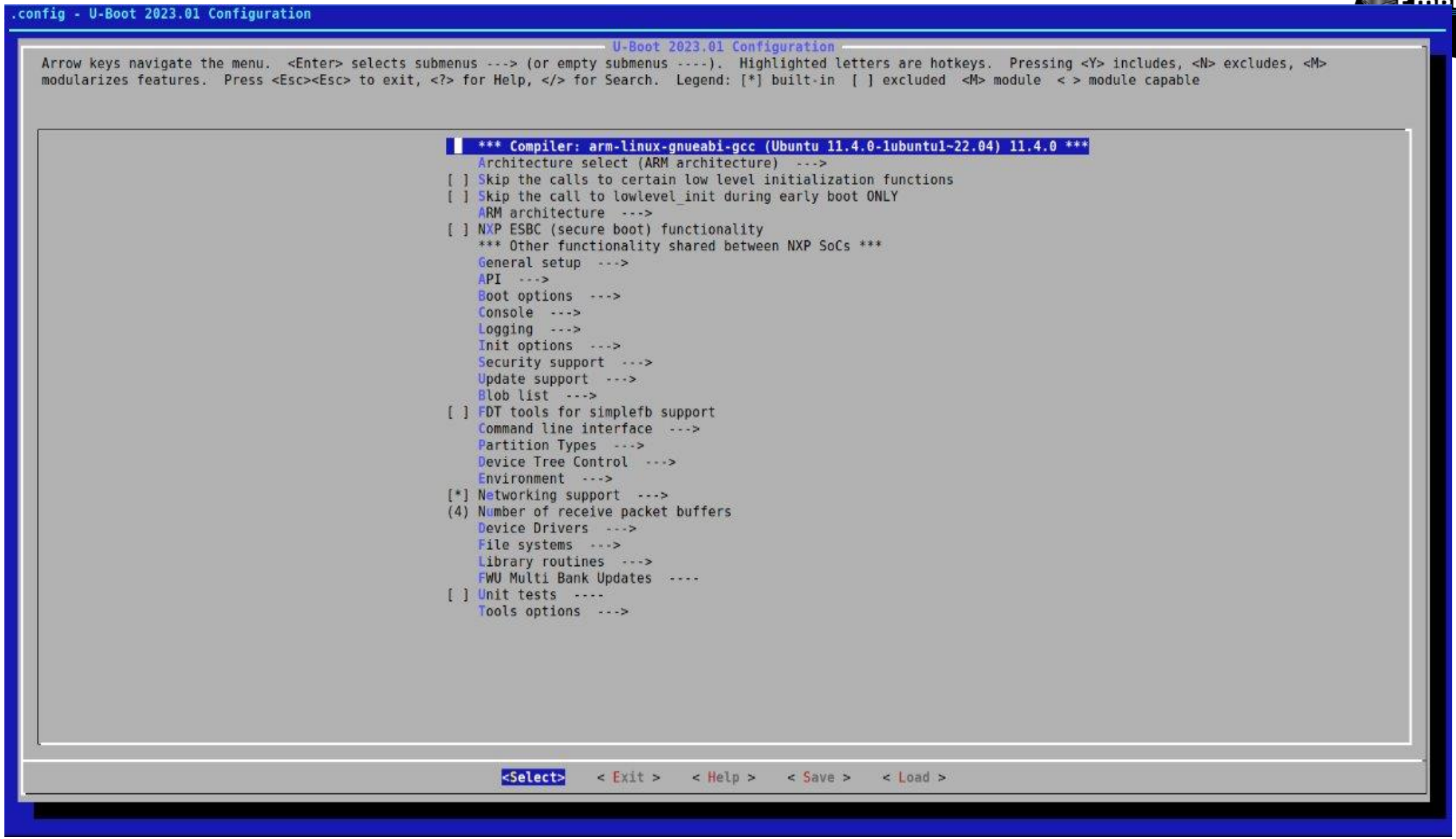
in order to linux image there are three elements are required:

- bootloader
- kernel
- file systems
- any each of them may be stored in adifferent storages for example flash memory, sd card, USB
- file systems bay be stored remotely in NFT or TFTP servers.
- kernel may be remotely in TFTP server

U-Boot in flash, kernel in a remote TFTP server and rootfs in a remote NFS server.
This configuration is common for kernel/application development and debugging from a connected host machine

# U-Boot

- Cloning the source code of u-boot from git-hub:
  - git clone https://github.com/u-boot/u-boot.git
- To set u-boot configuration use this
  - Make menuconfig


.config - U-Boot 2023.01 Configuration
U-Boot 2023.01 Configuration
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module < > module capable
*** Compiler: arm-linux-gnueabi-gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0 ***
Architecture select (ARM architecture) --->
[ ] Skip the calls to certain low level initialization functions
[ ] Skip the call to lowlevel_init during early boot ONLY
ARM architecture --->
[ ] NXP ESBC (secure boot) functionality
*** Other functionality shared between NXP SoCs ***
General setup --->
API --->
Boot options --->
Console --->
Logging --->
Init options --->
Security support --->
Update support --->
Blob list --->
[ ] FDT tools for simplefb support
Command line interface --->
Partition Types --->
Device Tree Control --->
Environment --->
[*] Networking support --->
(4) Number of receive packet buffers
Device Drivers --->
File systems --->
Library routines --->
FWU Multi Bank Updates ----
[ ] Unit tests ----
Tools options --->
<Select>
< Exit >
< Help >
< Save >
< Load >
Embinux

- After make all configuration save it and exit
- Use / to search for any configuration
- check this if there is an error
  https://stackoverflow.com/questions/23050188/cant-make-menuconfig

```
root@mahmoud:/media/mahmoud/1e2c08e4-d43c-465c-bf6e-9a5a656676d5/embedded-linux-qemu-labs/bootloader/u-boot# make menuconfig
scripts/kconfig/mconf  Kconfig


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.
```

# Installing ARM compiler

sudo apt install gcc-arm-linux-gnueabi

## Set env CROSS_COMPILE variable to use this as a default compiler

export CROSS_COMPILE=arm-linux-gnueabi-

## And set the PATH

export PATH=$HOME/mahmoud/usr/bin/arm-linux-gnueabi/bin:$PATH

## Configuring u-boot to the target board

Make [board_name]

make vexpress_ca9x4_defconfig

## Run make command to build the u-boot

## Some errors may be occurs needed to install this

sudo apt install libssl-dev device-tree-compiler swig \python3-distutils python3-dev python3-setuptools

# Installing QEMU User space emulator

sudo apt install qemu-system-arm

## Test u-boot using Qemu emulator

qemu-system-arm -M vexpress-a9 -m 128M -nographic -kernel u-boot

-M: emulated machine
-m: amount of memory in the emulated machine

-kernel: allows to load the binary directly in the emulated machine and run the machine with it. This way, you don't need a first stage bootloader. Of course, you don't have this with real hardware.

## Trying to set any env

```
=> setenv embinux mahmoud
=> save
  save saveenv
=> saveenv
Saving Environment to FAT... Card did not respond to voltage select! : -110
** Bad device specification mmc 0 **
Failed (1)
=> @sS
```

```
root@mahmoud:/media/mahmoud/1e2c08e4-d43c-465c-bf6e-9a5a656676d5/embedded-linux-qemu-labs/bootloader/u-boot# qemu-system-arm -M vexpress-a9 -m 128M -nographic -kernel u-boot


U-Boot 2023.01 (Jul 20 2024 - 11:26:40 +0300)

DRAM:  128 MiB
WARNING: Caches not enabled
Core:  18 devices, 10 uclasses, devicetree: embed
Flash: 64 MiB
MMC:   mmci@5000: 0
Loading Environment from FAT... Card did not respond to voltage select! : -110
** Bad device specification mmc 0 **
In:    serial
Out:   serial
Err:   serial
Net:   eth0: ethernet@3,02000000
Hit any key to stop autoboot:  0
=>
=>
=>
=>
=>
=>
```

We now need to add an SD card image to the QEMU virtual machine, in particular to get a way to store U-Boot's environment.

using the dd command, create a 1 GB file filled with zeros, called sd.img:

```
oot@mahmoud:/media/mahmoud/1e2c08e4-d43c-465c-bf6e-9a5a656676d5/e
024+0 records in
024+0 records out
073741824 bytes (1.1 GB, 1.0 GiB) copied, 2.1038 s, 510 MB/s
```

```
dd if=/dev/zero of=sd.img bs=1M count=1024
```

creating the partions we need

```
cfdisk sd.img
```


->One partition, 64MB big, with the FAT16 partition type. Mark this partition as bootable.
->One partition, 8 MB big3 , that will be used for the root filesystem. Due to the geometry of the device, the partition might be larger than 8 MB, but it does not matter. Keep the Linux type for the partition.
-> One partition, that fills the rest of the SD card image, that will be used for the data filesystem. Here also, keep the Linux type for the partition.


**Disk: sd.img**
Size: 1 GiB, 1073741824 bytes, 2097152 sectors
Label: dos, identifier: 0x28c8bf37

| | Device | Boot | Start | End | Sectors | Size | Id | Type |
|---|---|---|---|---|---|---|---|---|
| >> | sd.img1 | * | 2048 | 133119 | 131072 | 64M | 6 | FAT16 |
| | sd.img2 | | 133120 | 149503 | 16384 | 8M | 83 | Linux |
| | sd.img3 | | 149504 | 2097151 | 1947648 | 951M | 83 | Linux |

Partition type: FAT16 (6)
Attributes: 80

[Bootable] [ Delete ] [ Resize ] [ Quit ] [ Type ] [ Help ] [ Write ] [ Dump ]

We will now use the loop driver4 to emulate block devices from this image and its partitions:

sudo losetup -f --show --partscan sd.img

- -f: finds a free loop device
- --show: shows the loop device that it used
- --partscan: scans the loop device for partitions and creates additional /dev/loopp block devices.

```
o.default-release/key4.db" pid=29885 comm="sc
[57712.567862] loop26: detected capacity char
[57712.568009]  loop26: p1 p2 p3
[57712.571749]  loop26: p1 p2 p3
root@mahmoud:/media/mahmoud/1e2c08e4-d43c-465
```

Also run sudo dmesg to confirm that 3 partitions were detected for the loop device selected by losetup

Last but not least, format the first partition as FAT16 with a boot label:

sudo mkfs.vfat -F 16 -n boot /dev/loopp1

Now, you can release the loop device:

sudo losetup -d /dev/loop<x>

# Testing U-Boot's environment

Start QEMU again, but this time with the emulated SD card

qemu-system-arm -M vexpress-a9 -m 128M -nographic \ -kernel u-boot/u-boot \ -sd sd.img

```
WARNING: Image format was not specified for 'sd.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.


U-Boot 2023.01 (Jul 20 2024 - 11:26:40 +0300)

DRAM:  128 MiB
WARNING: Caches not enabled
Core:  18 devices, 10 uclasses, devicetree: embed
Flash: 64 MiB
MMC:   mmci@5000: 0
Loading Environment from FAT... OK
In:    serial
Out:   serial
Err:   serial
Net:   eth0: ethernet@3,02000000
Hit any key to stop autoboot:  0
=>
=>
=>
=>
=> setenv embinux mahmoud
=> saveenv
Saving Environment to FAT... OK
=>
=> printenv embinux
embinux=mahmoud
=>
```

# Now if you try to load the kernel

```
DHCP client bound to address 10.0.2.15 (2 ms)
Using ethernet@3,02000000 device
TFTP from server 10.0.2.2; our IP address is 10.0.2.15
Filename 'boot.scr.uimg'.
Load address: 0x60100000
Loading: *
TFTP error: 'Access violation' (2)
Not retrying...
smc911x: MAC 52:54:00:12:34:56
smc911x: detected LAN9118 controller
smc911x: phy initialized
smc911x: MAC 52:54:00:12:34:56
BOOTP broadcast 1
DHCP client bound to address 10.0.2.15 (1 ms)
Using ethernet@3,02000000 device
TFTP from server 10.0.2.2; our IP address is 10.0.2.15
Filename 'boot.scr.uimg'.
Load address: 0x60100000
Loading: *
TFTP error: 'Access violation' (2)
Not retrying...
smc911x: MAC 52:54:00:12:34:56
cp - memory copy

Usage:
cp [.b, .w, .l, .q] source target count
Wrong Image Format for bootm command
ERROR: can't get kernel image!
=>
```

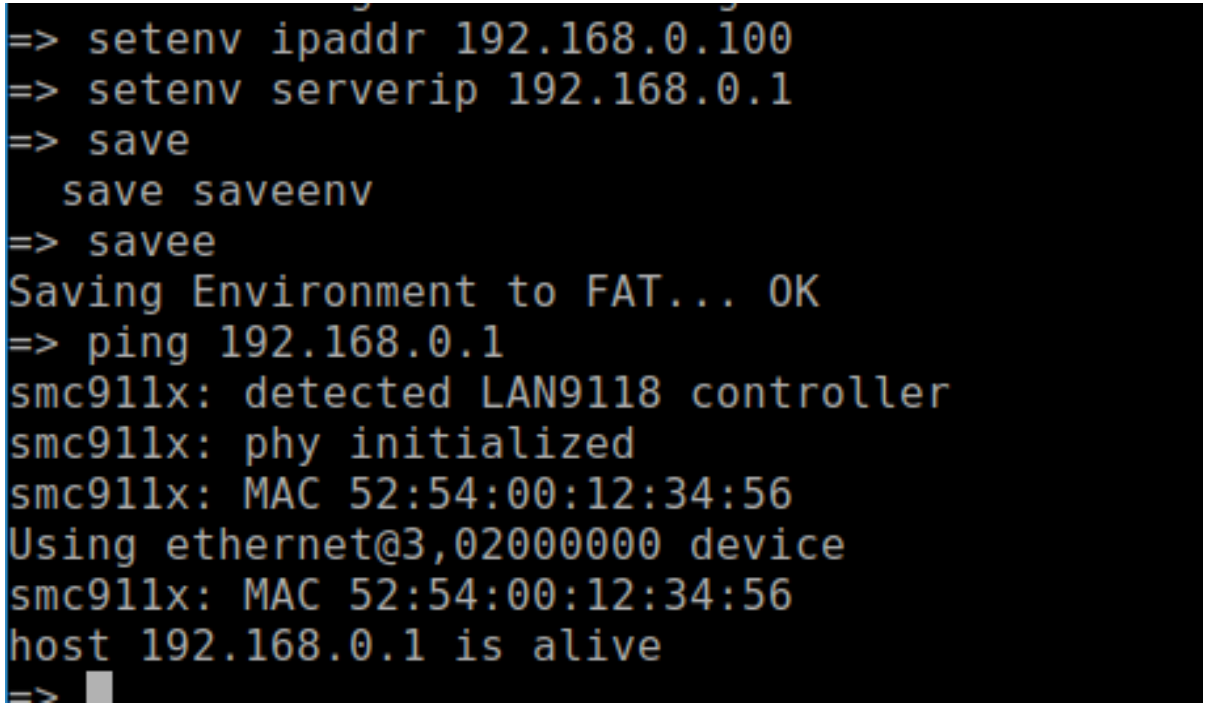
# Setup networking between QEMU and the host

Craeting new bash script file
named qemu-myifup
Contaning to

```
#!/bin/sh
/sbin/ip a add 192.168.0.1/24 dev $1
/sbin/ip link set $1 up
```

you will need root privileges to run QEMU this time, because of the need to bring up the network interface:

```
sudo qemu-system-arm -M vexpress-a9 -m 128M -
nographic \
-kernel u-boot/u-boot \
 -sd sd.img \
-net tap,script=./qemu-myifup -net nic
```



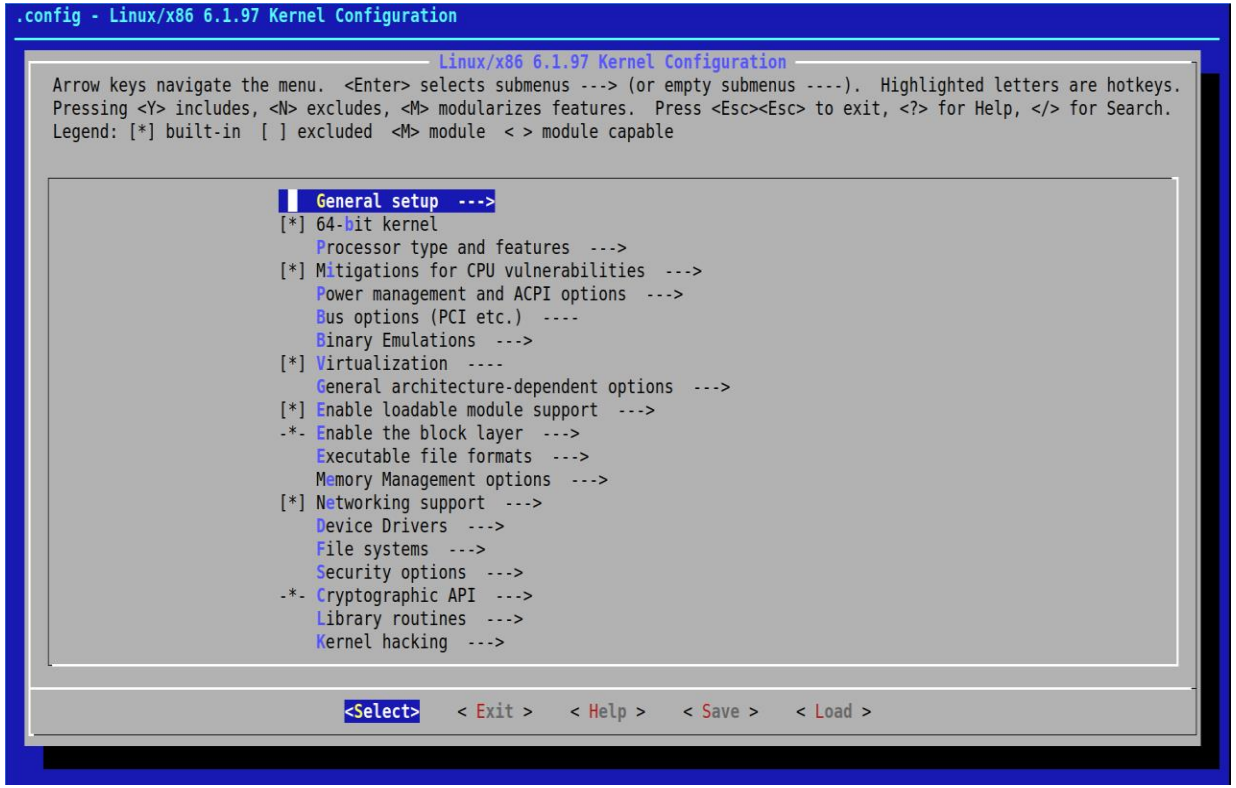
# Kernel - Cross-compiling

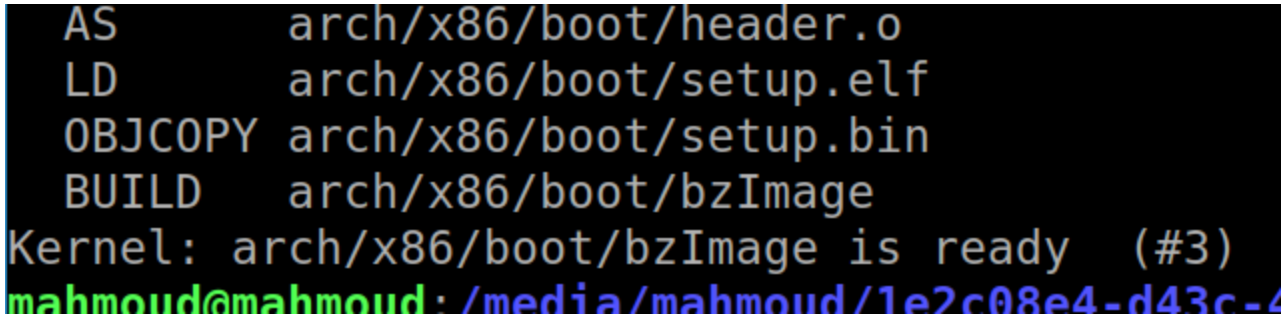
Define the value of the ARCH and CROSS_COMPILE

**make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi- vexpress_defconfig**

Run make menuconfig to configure the kernel as you need

Run **make –j<numberOfcore>**



# Load the kernel using u-boot

After building the kernel copy the zImage and DTB file according to your board to the directory exposed by the TFTP server.

You may find this directories in one of this paths

**/var/lib/tftpboot**
Or
**/src/tftp/**

sudo cp arch/arm/boot/zImage /src/tftp/

sudo cp arch/arm/boot/dts/vexpress-v2p-ca9.dtb /src/tftp/

Run Qemu and set the bootargs environment corresponding to the Linux kernel command line:

=> setenv bootargs console=ttyAMA0
=> saveenv

On the target (in the U-Boot prompt), load zImage from TFTP into RAM and DTB:

tftp 0x61000000 zImage
tftp 0x62000000 vexpress-v2p-ca9.dtb

Boot the kernel with its device tree:

bootz 0x61000000 - 0x62000000

# Why kernel panic?


```
1000a000.uart: ttyAMA1 at MMIO 0x1000a000 (irq = 39, base_baud = 0) is a PL011 rev1
1000b000.uart: ttyAMA2 at MMIO 0x1000b000 (irq = 40, base_baud = 0) is a PL011 rev1
1000c000.uart: ttyAMA3 at MMIO 0x1000c000 (irq = 41, base_baud = 0) is a PL011 rev1
drm-clcd-pl111 1001f000.clcd: assigned reserved memory node vram@4c000000
drm-clcd-pl111 1001f000.clcd: using device-specific reserved memory
drm-clcd-pl111 1001f000.clcd: core tile graphics present
drm-clcd-pl111 1001f000.clcd: this device will be deactivated
drm-clcd-pl111 1001f000.clcd: Versatile Express init failed - -19
drm-clcd-pl111 10020000.clcd: DVI muxed to daughterboard 1 (core tile) CLCD
drm-clcd-pl111 10020000.clcd: initializing Versatile Express PL111
drm-clcd-pl111 10020000.clcd: DVI muxed to daughterboard 1 (core tile) CLCD
drm-clcd-pl111 10020000.clcd: initializing Versatile Express PL111
clk: Disabling unused clocks
ALSA device list:
  #0: ARM AC'97 Interface PL041 rev0 at 0x10004000, irq 37
input: ImExPS/2 Generic Explorer Mouse as /devices/platform/bus@40000000/bus@40000000:motherboard-bus@40000000/bus@40000000:motherboard-bus@40000000:iofpga@7,00000000/10007000.kmi/s
io1/input/input2
drm-clcd-pl111 10020000.clcd: DVI muxed to daughterboard 1 (core tile) CLCD
drm-clcd-pl111 10020000.clcd: initializing Versatile Express PL111
/dev/root: Can't open blockdev
VFS: Cannot open root device "(null)" or unknown-block(0,0): error -6
Please append a correct "root=" boot option; here are the available partitions:
1f00          131072 mtdblock0
 (driver?)
1f01           32768 mtdblock1
 (driver?)
b300         1048576 mmcblk0
 driver: mmcblk
  b301           65536 mmcblk0p1 28c8bf37-01

  b302            8192 mmcblk0p2 28c8bf37-02

  b303          973824 mmcblk0p3 28c8bf37-03

Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(0,0)
CPU: 0 PID: 1 Comm: swapper/0 Not tainted 6.1.97 #1
Hardware name: ARM-Versatile Express
 unwind_backtrace from show_stack+0x10/0x14
 show_stack from dump_stack_lvl+0x40/0x4c
 dump_stack_lvl from panic+0x108/0x334
 panic from mount_block_root+0x190/0x230
 mount_block_root from prepare_namespace+0x150/0x18c
 prepare_namespace from kernel_init+0x18/0x12c
 kernel_init from ret_from_fork+0x14/0x28
Exception stack(0x88825fb0 to 0x88825ff8)
5fa0:                                     00000000 00000000 00000000 00000000
5fc0: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
5fe0: 00000000 00000000 00000000 00000000 00000013 00000000
---[ end Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(0,0) ]---
```

# Loading the file system using NFS

Create a nfsroot directory in the current lab directory. This nfsroot directory will be used to store the contents of our new root filesystem.

Install nfs server using

Sudo apt install nfs-kernel-server

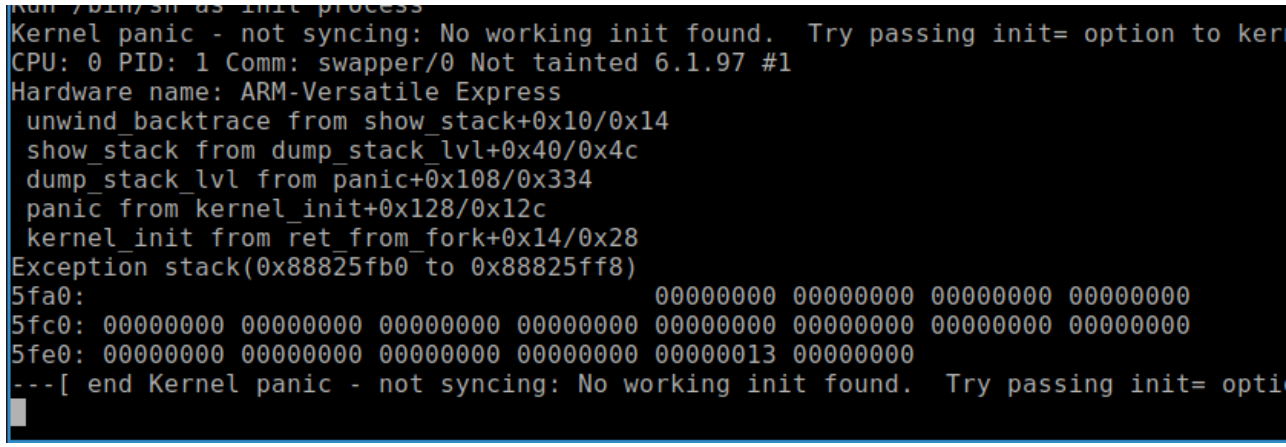
edit the /etc/exports file as root to add the following line:

<nfsroot_path> 192.168.0.100(rw,no_root_squash, no_subtree_check)

Run Qemu again and set this env variable

setenv bootargs ${bootargs} root=/dev/nfs ip=192.168.0.100 nfsroot=192.168.0.1:<nfsroot_path>=3,tcp rw

And boot! Kernel panic again. Why?



# Busy box

Cloneing busybox from git-hub:

git clone https://git.busybox.net/busybox
git checkout 1_36_stable

Configure the busybox:

make menuconfig

Then compile the busybox using

make install

**Note the path of nsfroot!**

Run qemu again.
Not panic but need some file systems we need to create them

```
process '-/bin/sh' (pid 85) exited. Scheduling for restart.
process '-/bin/sh' (pid 87) exited. Scheduling for restart.
can't open /dev/tty3: No such file or directory
can't open /dev/tty2: No such file or directory
process '-/bin/sh' (pid 88) exited. Scheduling for restart.
process '-/bin/sh' (pid 89) exited. Scheduling for restart.
can't open /dev/tty4: No such file or directory
process '-/bin/sh' (pid 90) exited. Scheduling for restart.
can't open /dev/tty3: No such file or directory
can't open /dev/tty2: No such file or directory
```

```
bin  sbin  usr
mahmoud@mahmoud:/media/mahmo
```

Create proc, sys, etc and dev
In etc creat init.d dir and in it create rcs file.In this startup script, mount the /proc and /sys filesystems.