Kernel Build

<https://kernelnewbies.org/KernelBuild>

In the document Cross tool ng.docx I did, I show how to prepare a gcc cross compiler for arm 32 bit HF.

Now using those compilers, let’s compile a Linux kernel.

Host requirements:

sudo apt-get install libncurses5-dev gcc make git exuberant-ctags bc libssl-dev

In this tutorial we will answer on the question, can we compile a kernel with cross compiler that was compiler with different kernel?

Lest compile the latest kernel for ARM

git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git

cd linux-stable

There are possible few options to start with:

1. Use existing defconfig from arch/arm/configs and copy one of them as .config to the top directory
2. Compile a with default kernel config

See here:

<http://www.thegeekstuff.com/2013/06/compile-linux-kernel>

I want to do the following:

1. Compile a kernel for X86
2. Compile a kernel for known arm CPU
3. Clone a known arm CPU , and create new BSP for a new board. Save this new config.

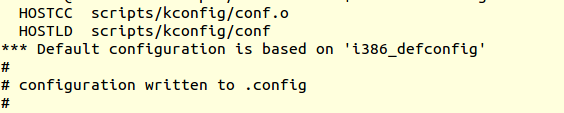
**Default config – 64bit i386**

Run:

make defconfig

Will take default configuration from i386\_defconfig.

There for it will compile Linux for PC.



We can copy from arch/arm/\* *any of them*

For example, the raspberry pi 3 arch/arm/bcm2835\_defconfig

cp arch/arm/configs/bcm2835\_defconfig .config

to compile Linux for i386 , just do:

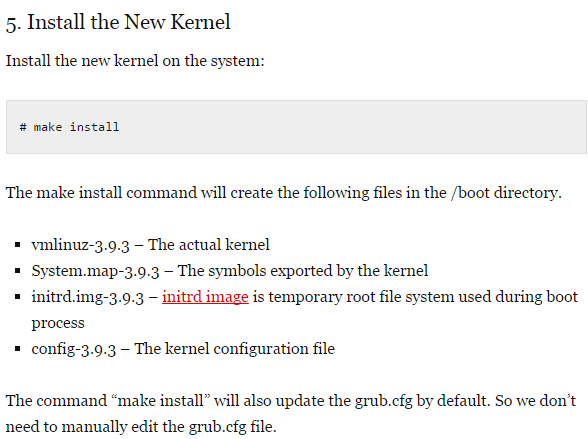
# make

Compile the kernel modules:

# make modules

Install the kernel modules:

# make modules\_install



Taken from [here](http://www.thegeekstuff.com/2013/06/compile-linux-kernel)

**Compile kernel for raspberry pi 3 bcm2835 arm**

We copied the defconfig from arch/arm/bcm2835\_defconig into .config

To the top level directory.

We need to cross compile using ARM tool chain compiler and not using the i386 GCC compiler.

We prepare one with crosstools-ng ( see document)

Add a CC\_LPCATION environment variable:

specify the full path to the compiler include the – at the end

export CC\_LOCATION=/home/adsteam/x-tools/arm-unknown-linux-gnueabihf/bin/arm-unknown-linux-gnueabihf-

To start the compilation we need to tell the Makefile which ARCH and which Compiler we will use

Best is , to create a source script: ( for example) compile:

ARCH=arm CROSS\_COMPILE=$CC\_LOCATION make menuconfig

When it open the menuconfig just exist and save

Now can issue the compilation for the kernel:

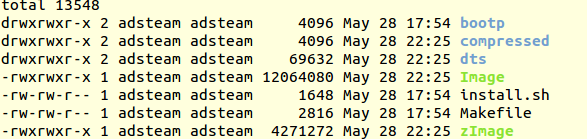
ARCH=arm CROSS\_COMPILE=$CC\_LOCATION make

Or make it also in a source script

\*source script, where you do **source** to execute it

**The kernel ..**

In case of arm , the kernel is reside in arch/arm/boot



Once done , I want to create an img

So I am asking the same question that was asked here:

<https://raspberrypi.stackexchange.com/questions/3196/building-kernel-image-img-including-ramdisk>