Raspberry pi board

specifications :

architecture : arm1176
board : bcm2835

Building cross compiler :

1. We use buildroot for building cross compiler.download a buildroot source.from

   http://buildroot.uclibc.org/downloads/

2. After downloading,copy to a seperate directory and uncompress it by appropriate command.

   Ex: tar -xvf <tar file name> (for tar files)

   unzip <zip file name> (for zip files)

3. Go to source folder of buildroot and give command **make distclean**.It removes previous setting if any.

4. Give the make command allong with ARCH variable and default configuration file of architecture for which we built cross compiler.all those files are there in 'configs' folder. In this case following command...

   make ARCH=arm raspberrypi_defconfig

5. Then give following command

   make ARCH=arm menuconfig

   this command gives graphical representation of your build root.

6. Select 'Tool chain' option.you directed to another prompt.in this,

   A. Select kernel headers-> <linux source you wish >

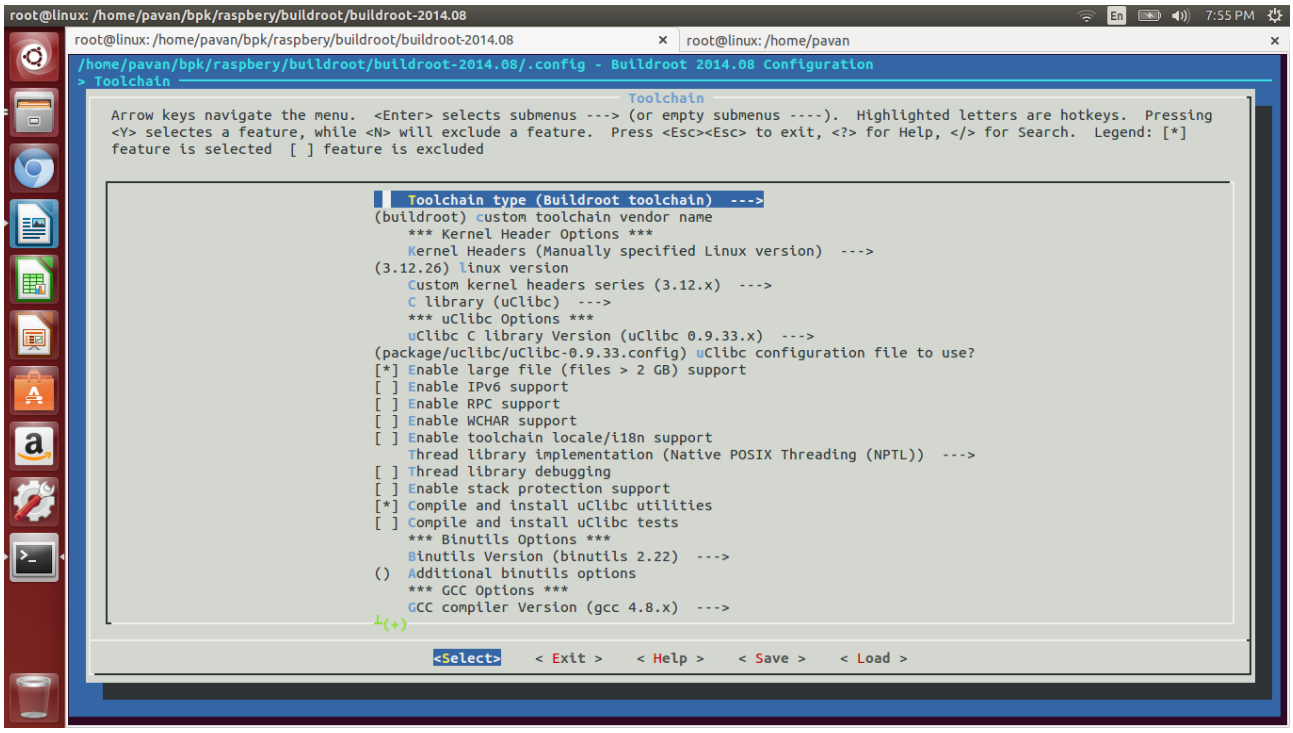
   B. C library-> uClibc

   C. uClibc C library version -> <version you wish>

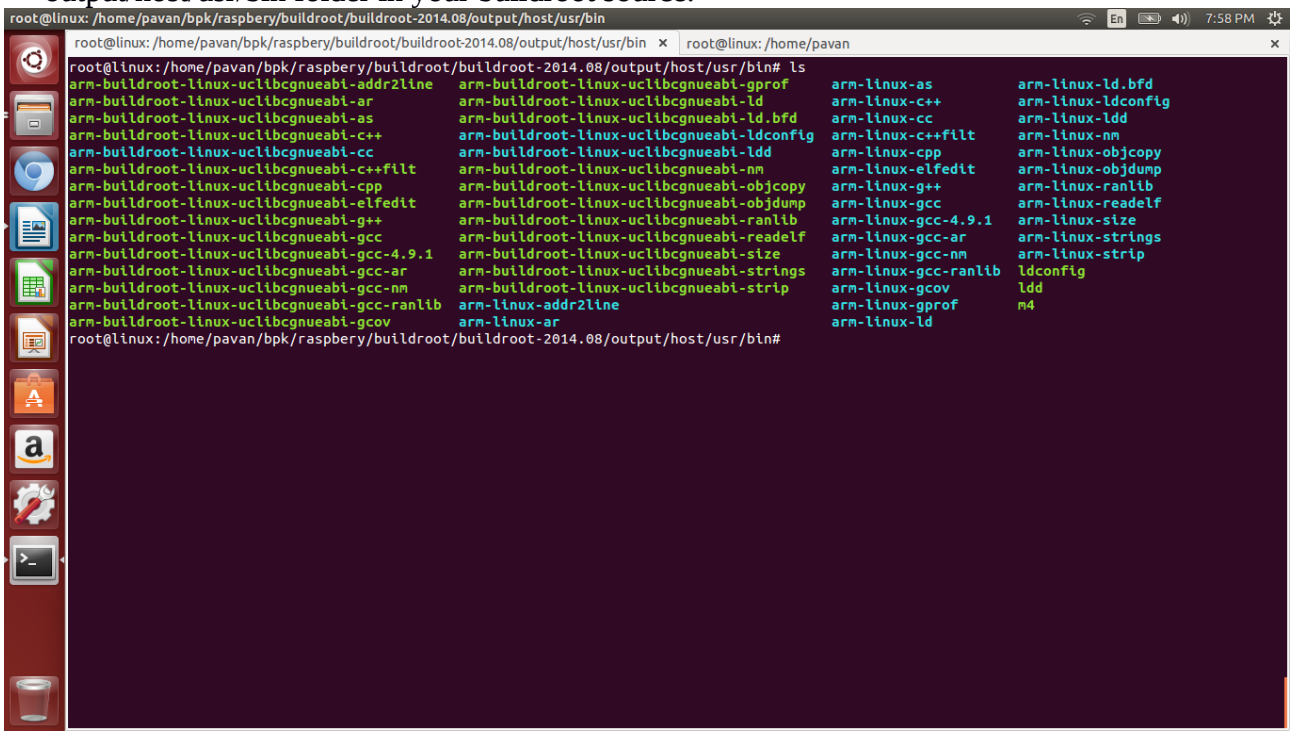
   D. Enable large file,ipv6,rpc,wchar supports.(press space bar until '*' appears against those options). Select 'Binutils' version (your wish),compiler version(your wish).

   E. check all remaining entries.if you find any enable (*) options,disable that.

   press 'Exit' until 'Do you wish to save your new configuration?' appears.then press 'Yes'.



7. Give the command **make**.it compiles your buildroot and obtain your tool chain on output/host/usr/bin folder in your buildroot source.



Building kernel image:

1. Download a kernel source from https://www.kernel.org/

2. copy to a seperate directory and uncompress it and go to that source.

3. Give the command **make ARCH=arm bcm2835_defconfig**.

In arch/arm/configs folder,default configuration files are there.

4. Export your cross compiler path to this terminal by following command.

PATH=$PATH:<path from home folder to buildroot>/output/host/usr/bin/

5. Give command

make ARCH=arm CROSS_COMPILE=arm-linux- LOADADDR=0x01008000 uImage

if everything goes fine,it gives your kernel image along with u-boot headers uImage in 'arch/arm/boot folder'.

6. Give command **make ARCH=arm CROSS_COMPILE=arm-linux- bcm2835-rpi-b.dtb** for generating dtb file in 'arch/arm/boot/dts' folder.

Building rootfs :

1.Create a directory name rootfs.in which we need to create following folders.
Bin, Sbin, Proc, Sys, Etc, Usr/bin, Usr/sbin, Dev.

populating 'dev':

write the following:

```
mknod console c 5 1

mknod  null  c  1 3

mknod   ttyAMA0 c 204 64
```

Populating Etc:

-> Create a file called inittab and a folder called init.d.

in inittab,write the following:

```
# executing rcS script
null::sysinit:/etc/init.d/rcS

# enabling console respawn
ttyAMA0::respawn:-/bin/sh
```

in init.d folder,create a file rcS.and write following:

```
#!/bin/sh

mount -t proc   null    /proc
mount -t sysfs  null    /sys
```

```
mount -t devtmpfs null  /dev
mount -t mqueue null    /mq
mount -t tmpfs  mdev    /dev
mount -t devpts devpts  /dev/pts

echo "/sbin/mdev" > /proc/sys/kernel/hotplug

/sbin/mdev -s

export PATH=\
/bin:\
/sbin:\
/usr/bin:\
/usr/sbin:\
```

save the file and give executable permissions to that by giving command

chmod +x rcS

populating bin,sbin,usr/bin,usr/sbin:

->These 4 folders are populated by using busy box.download the busy box source from

http://www.busybox.net/downloads/

-> Copy to a seperate directory and uncompress it.export your cross compiler path and go to the busybox folder.

-> Give command **make menuconfig**.

Enable Busybox settings->Build options -> Build busy box as a static binary. Save it.

-> Give command **make CROSS_COMPILE=arm-linux-** and later give command

make CROSS_COMPILE=arm-linux- install

This creates folder _install in your busybox folder.copy contents of this folder into your rootfs folder by below command.

Cp -Rfp _install/* <path to rootfs>/

Firmware:

We have to download some files which are need by raspbery pi for booting. Those are called as firmware files. Download from link https://github.com/raspberrypi/firmware in this link.download zip option is there. After download,copy to seperate directory and uncompres it. Go to source folder create a file called config.txt in boot folder and write the following into that file.

kernel=u-boot.bin

U-boot :

We need bootloader for booting. In this case we use u-boot as a boot loader. We have to download u-boot source from ftp://ftp.denx.de/pub/u-boot/ and copy to a seperate directory and uncompress it. Export your cross compiler and give the command

make ARCH=armrpi_b_defconfig.

Later give **make ARCH=arm CROSS_COMPILE=arm-linux-**. This creates u-boot.bin file in your u-boot folder.

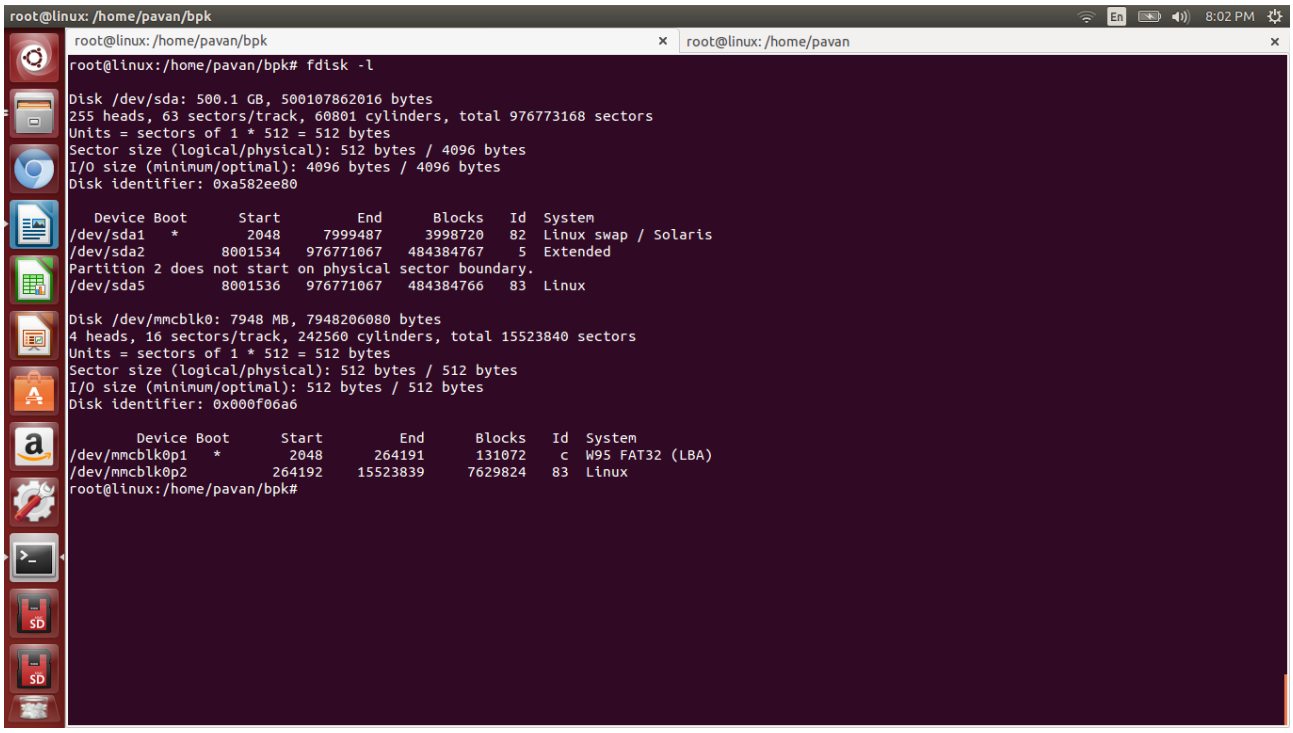
partitioning of SD card:

-> Take sd card and put into slot. Give the command **fdisk -l** shows the card partitions.

For example: in my case,it shows

Device Boot		Start	End	Blocks	Id	System
/dev/mmcblk0p1	*	2048	7999487	3998720	82	Linux

in above example,my partition is /dev/mmcblk0p1.now i need to modifications on that partition by giving **fdisk /dev/mmcblk0p1**.Later give options in following sequence.delete all partitions by giving option d until all are deleted.then by using n option create partitions.in our case,2 partitions are sufficient with whatever size you want.



-> Now,one partition is used for booting files and one partition is for holding rootfile system. So,enable bootable flag for partition which we intended to use for booting by giving option a. And we have to change our partion system id by giving t--> <partition number> L->c. After this,give w for save changes. And unmount the card(not removing,just unmount).

-> Now,we have to make file systems on top of partitions. For this,we have to give following commands:

mkfs.vfat -F 32 <boot partition> -n <name you wish>

mkfs.ext3 <rootfs partition> -L <name you wish>

->After completing above process,we have to mount the 2 partitions.transfer boot files to your boot partition.Those files are

A. uImage(present in 'linuxkernel/arch/arm/boot' folder)

B. Bcm2835-rpi-b.dtb(present in 'linuxkernel/arch/arm/boot/dts' folder)

C. U-boot.bin(present in u-boot source folder)

D. Bootcode.bin,fixup.dat,start.elf,config.txt(present in your <firmware>/boot folder)

-> Copy contents of rootfs folder into rootfs partition of sd card.and unmount and remove sd card from your system.

Booting raspberry pi :

-> Put your sd card into slot of board and connect hardware. Go to the board prompt by using minicom.If everything goes fine,u-boot prompt is appeared on minicom screen.

-> set environment variables as follows.....

setenv bootargs console=ttyAMA0,115200 root=/dev/mmcblk0p2 rootfstype=ext3 rootwait

setenv fdtfile bcm2835-rpi-b.dtb

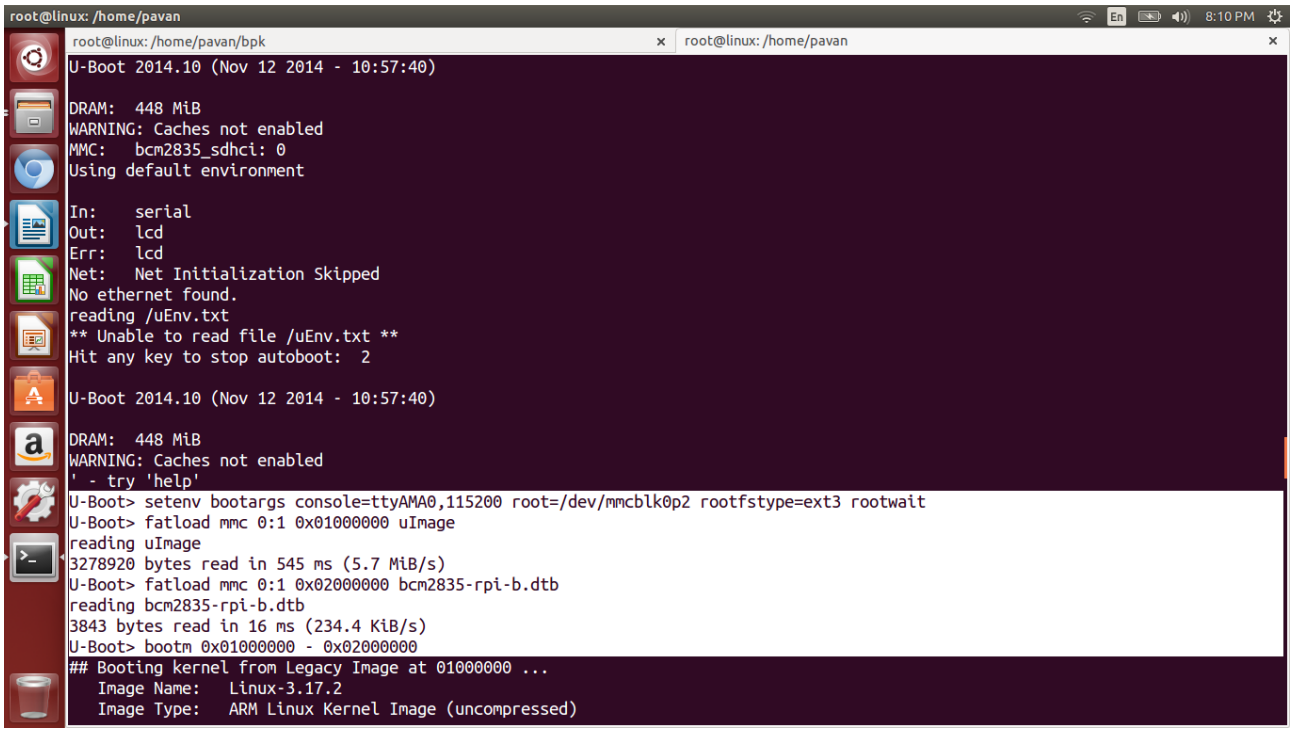
Now,we transfer files to RAM of the board by following commands.....

fatload mmc 0:1 0x01000000 uImage

fatload mmc 0:1 0x02000000 bcm2835-rpi-b.dtb

Now,boot the board by following command.....

bootm 0x01000000 - 0x02000000



after booting, # is appears..