NOTE: **uname –a** command, gives info about the BBB card.

```
NOTE: chmod +x /home/root/Test
```

this command adds executable property to e file.

1. Install VMWare Workstation (mine is 12.1.1) with Ubuntu (mine is 12.04-desktop-i386.iso)
2. Install gcc compiler with following command; (mine is 4.9)

```
sudo add-apt-repository ppa:ubuntu-toolchain-r/test
sudo apt-get update
sudo apt-get install gcc-4.9 g++-4.9
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.9 60 --slave
/usr/bin/g++ g++ /usr/bin/g++-4.9
```

NOTE: to compile from console, type this (just for simle .c files)

➔ gcc main.c –o MAIN (then MAIN is the executable file)

3. Install Eclipse with the following command (mine is Eclipse INDIGO Version 3.7.2, Build ID: I20110613-1736)

```
sudo apt-get install eclipse eclipse-cdt g++ gcc
```

4. Be sure that you are able to build a c / c++ project with eclipse (you can control the YourProject->Properties->C/C++ Build->Settings the control the "Command" part of Cross GCC Compiler and Cross Compiler Linker parts.
5. Install Linux Toolchain for BBB with following commands (mine version is 4.6)

```
sudo apt-get install gcc-arm-linux-gnueabi

sudo apt-get install g++-arm-linux-gnueabi
```

6. After installation, control it and open Shell, type "arm-linux-gnueabi-" and press TAB couple of times. You should see your donwloaded packages.
7. To configure your Eclipse project as BBB project, type the following command of that path the YourProject->Properties->C/C++ Build->Settings the control the "Command" part of Cross GCC Compiler and Cross Compiler Linker parts.

"arm-linux-gnueabi-gcc-4.6" OR "arm-linux-gnueabi-g++-4.6" upto your project type and toolchain version.

8. After your build, if there are no errors, that means you created an ecxecutable file for your BBB.
9. To copy the executable file to your BBB, type following command. This command coppy your "B_ToggleLed" named executable file to BBB under /home/root/ path. Note that, to use the FOLLOWING command, you must be in the directory of your executable file.

   csp B_ToggleLed root@192.168.7.2:/home/root/

10. To connect and run the application in the BBB, first, install its driver (it is avaible in the BBB) then connect to BBB with following command.

ssh 192.168.7.2 –l root
password: EMPTY

After that, you should go the director of your executable and type following;

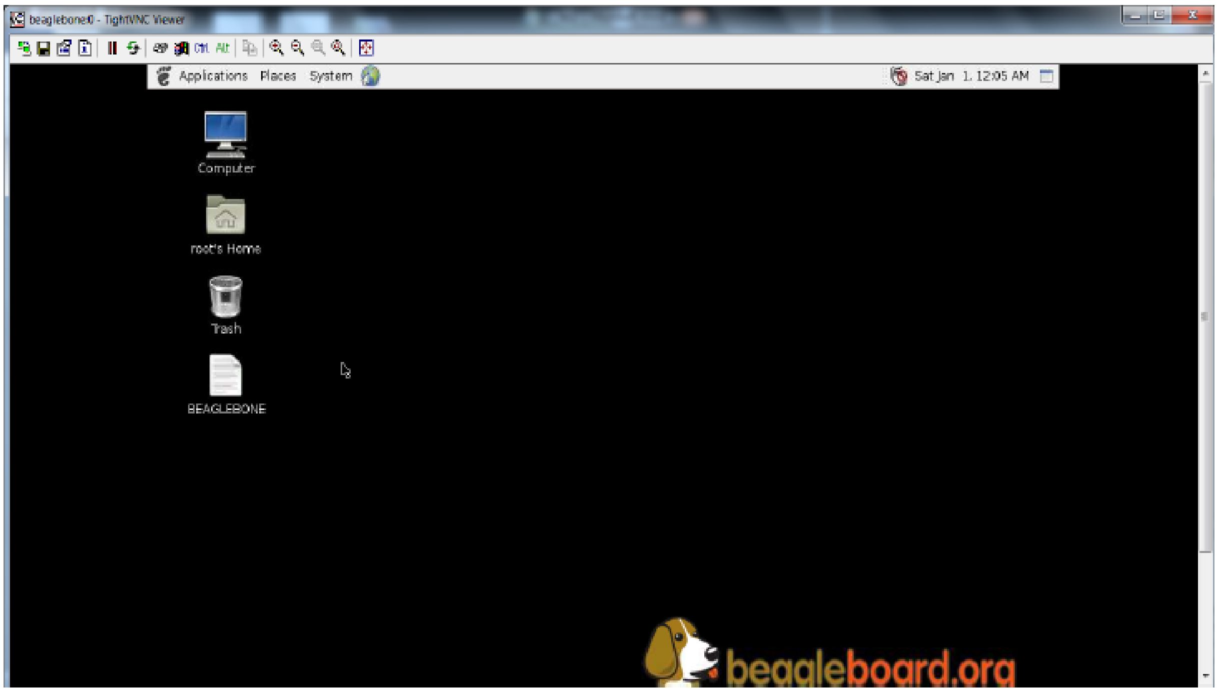
./B_ToggleLed

You should see the app is running.

11. VNC Server for Windows (To see BBB's desktop)
    - Download "tightvnc" server to your BBB with following command (after connection to BBB over ssh)
        - *opkg update*
        - *opkg install x11vnc*
        - *x11vnc -bg -o %HOME/.x11vnc.log.%VNCDISPLAY -auth /var/run/gdm/auth-for-gdm*/database -display :0* -forever

NOTE: just use only last one after installing to connect BBB.

- After that, start TightVNC Connection for BBB's ip (192.168.7.2) the done.



BOOTING BBB

1. you should 4GB-microSD card
2. Google "beagleboard lates-images" and download a img sth like that

- ➔ BBB-eMMC-flasher-distro-YYYY.MM.DD.img.xz.
- ➔ unzip it with 7-zip to .img file

3. Format sd card and wire image to the sd card with (for example) “Win32 Disk Imager”
   NOTE: to use it, you need 7-zip, and run it in admin mode.
4. Unpower BBB, insert SD card to BBB, hold on the (S2) boot button and power-up BBB with 5V-2A (least) from power jack. Do not connect ANYTHING to BBB by the way.
5. After powering, hold the boot button for a while and remove your finger from the button.
6. After (appx.) 30min., proccess was done and whole 4 led must be at HIGH.
7. Directly unpower BBB and remove sd card.
8. Power it again and it should be done.
9. If not, press reset button, reset and boot button together and try again.

NOTE: you can re-format your sd card to original size by using “SDFormatter V4.0” program (for Windows)

PROGRAMMING NOTES:

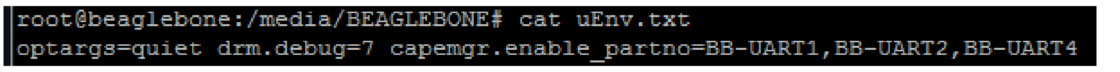
1. To read ADC, P9-34 (GNDA_ADC) must be grounded and max input voltage is 1.8V for these pins.

2. Serial Port;
   - Serial Ports names /dev/ttyOx, where x is 0 to 5 like that;

NAME	RX	TX	CTS	RTS	Device	REMARK
UART0	J1_4	J1_5			/dev/ttyO0	BeagleBone Black only
UART1	P9_26	P9_24	P9_20	P9_19	/dev/ttyO1	
UART2	P9_22	P9_21	P8_37	P8_38	/dev/ttyO2	
UART3		P9_42	P8_36	P8_34	/dev/ttyO3	TX only
UART4	P9_11	P9_13	P8_35	P8_33	/dev/ttyO4	
UART5	P8_38	P8_37	P8_31	P8_32	/dev/ttyO5	

- Just uart 0 is defaultly enabled. To enable others:
- ➔ /media/BEAGLEBONE/uEnc.txt filei we should add that line;
  - ■ capemgr.enable_partno=BB-UART1,BB-UART2,BB-UART4

in the end, file should see sth like that



```
root@beaglebone:/media/BEAGLEBONE# cat uEnv.txt
optargs=quiet drm.debug=7 capemgr.enable_partno=BB-UART1,BB-UART2,BB-UART4
```

Now, you can see ttyO1, ttyO2, ttyO4 in the dev path with ttyO0.

NOTE: Todo that, related device's .dtbo and .dts files must be placed in /lib/firmware path, otherwise, search **Device Tree Overlays (DTOs) title (Exploring BeagleBone** Tools and Techniques for Building with Embedded Linux® Derek Molloy page 222)

3. Reaching USB Flash Memory

You can format the flash in Windows file system FAT32, unit size = 4096 etc. In the BBB, we can reach the files in the flash with:

➔ /media/FLASH_NAME/file1.txt

4. To find out what capemgr yo have (for example like analog input code) you should look at the path:

   ➔ /sys/device

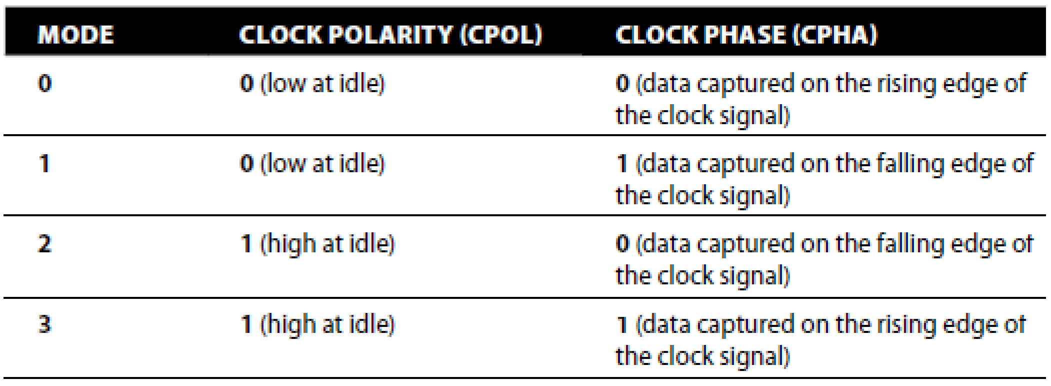
5. To change content of uEnv.txt file, we dont need vnc server

   ➔ create backup file with "cp" command
   ex: **cp** uEnv.txt uEnv_BACKUP.txt

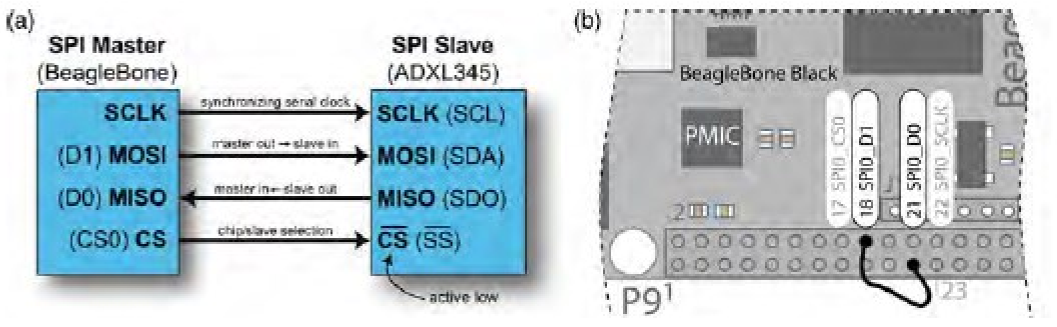
   ➔ re-write uEnv.txt with "echo" command
   ex: **echo** "datadatadata" > uEnv.txt

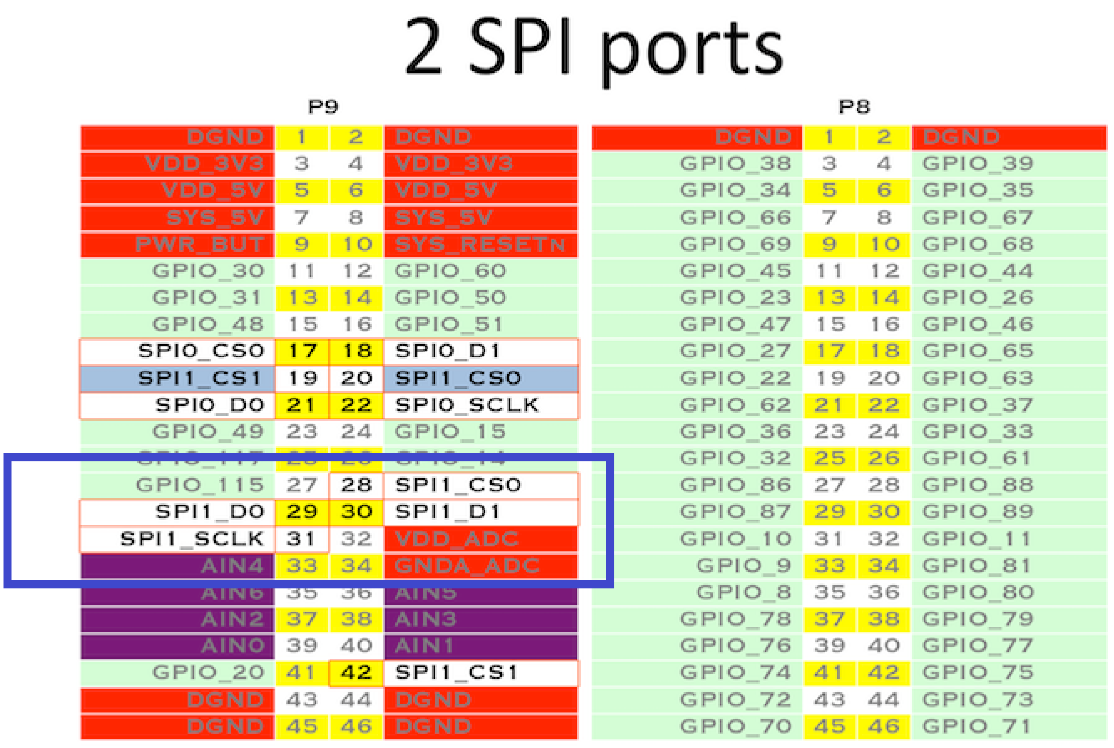
6. SPI;
   Here are some connection figures;

Table 8-4: SPI Communication Modes



MODE	CLOCK POLARITY (CPOL)	CLOCK PHASE (CPHA)
0	0 (low at idle)	0 (data captured on the rising edge of the clock signal)
1	0 (low at idle)	1 (data captured on the falling edge of the clock signal)
2	1 (high at idle)	0 (data captured on the falling edge of the clock signal)
3	1 (high at idle)	1 (data captured on the rising edge of the clock signal)

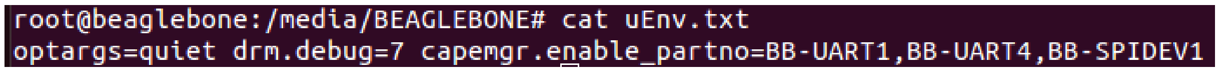




2 SPI ports

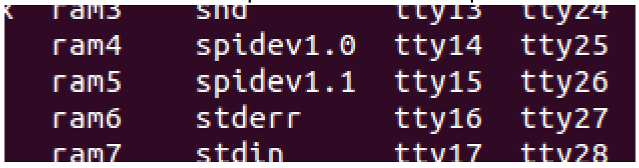
P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	GPIO_38	3	4	GPIO_39
VDD_5V	5	6	VDD_5V	GPIO_34	5	6	GPIO_35
SYS_5V	7	8	SYS_5V	GPIO_66	7	8	GPIO_67
PWR_BUT	9	10	SYS_RESETN	GPIO_69	9	10	GPIO_68
GPIO_30	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
GPIO_31	13	14	GPIO_50	GPIO_23	13	14	GPIO_26
GPIO_48	15	16	GPIO_51	GPIO_47	15	16	GPIO_46
SPI0_CS0	17	18	SPI0_D1	GPIO_27	17	18	GPIO_65
SPI1_CS1	19	20	SPI1_CS0	GPIO_22	19	20	GPIO_63
SPI0_D0	21	22	SPI0_SCLK	GPIO_62	21	22	GPIO_37
GPIO_49	23	24	GPIO_15	GPIO_36	23	24	GPIO_33
GPIO_117	25	26	GPIO_14	GPIO_32	25	26	GPIO_61
GPIO_115	27	28	SPI1_CS0	GPIO_86	27	28	GPIO_88
SPI1_D0	29	30	SPI1_D1	GPIO_87	29	30	GPIO_89
SPI1_SCLK	31	32	VDD_ADC	GPIO_10	31	32	GPIO_11
AIN4	33	34	GNDA_ADC	GPIO_9	33	34	GPIO_81
AIN6	35	36	AIN5	GPIO_8	35	36	GPIO_80
AIN2	37	38	AIN3	GPIO_78	37	38	GPIO_79
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	SPI1_CS1	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	GPIO_70	45	46	GPIO_71

Blue labeled part is for SPI1 for example. To enable SPI, just like uart, we add sth in uEnv.txt file like below.



```
root@beaglebone:/media/BEAGLEBONE# cat uEnv.txt
optargs=quiet drm.debug=7 capemgr.enable_partno=BB-UART1,BB-UART4,BB-SPIDEV1
```

after that we can see spi device under /dev path like below



```
k  ram3     snd         tty13   tty24
   ram4     spidev1.0   tty14   tty25
   ram5     spidev1.1   tty15   tty26
   ram6     stderr      tty16   tty27
   ram7     stdin       tty17   tty28
```

- ➔ then "/dev/spidev1.0" it is SPIDEV0.

BE CARREFUL: spi1's a pin is mutual with HDMI. So, before use it, mabe (not tried) you should disable the HDMI just like below.

- ➔ cape_disable=bone_capemgr.disable_partno=BB-BONELT-HDMI,BB-BONELT-HDMIN

GENERAL NOTE: to see device properties, open /lib/firmware path and cat .dts files of the devices!!!!

<u>Configure Static Local IP</u>

1. Connect Ethernet cable to BBB.
2. type following command
   - ➔ /etc/network# **ifconfig**

then you should see output like that:

```
eth0      Link encap:Ethernet  HWaddr C8:A0:30:AB:32:3A
 inet addr:192.168.1.100  Bcast:192.168.1.255  Mask:255.255.255.0  //important
 inet6 addr: fe80::caa0:30ff:feab:323a/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
 RX packets:1855 errors:0 dropped:0 overruns:0 frame:0
 TX packets:424 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:232957 (227.4 KiB)  TX bytes:55698 (54.3 KiB)
 Interrupt:56

lo        Link encap:Local Loopback
 inet addr:127.0.0.1  Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING  MTU:65536  Metric:1
 RX packets:234 errors:0 dropped:0 overruns:0 frame:0
 TX packets:234 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:19082 (18.6 KiB)  TX bytes:19082 (18.6 KiB)

usb0      Link encap:Ethernet  HWaddr A2:CB:A9:A8:0B:F4
 inet addr:192.168.7.2  Bcast:192.168.7.3  Mask:255.255.255.252
 UP BROADCAST MULTICAST  MTU:1500  Metric:1
 RX packets:0 errors:0 dropped:0 overruns:0 frame:0
 TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

➔ then type following command;

```
/var/lib/connman# ls -al
```

output should be;

```
root@beaglebone:/var/lib/connman# ls -al
 total 16
 drwxr-xr-x  3 root root 4096 Jan  1  2000 .
 drwxr-xr-x 18 root root 4096 Jan  1  2000 ..
 drwx------  2 root root 4096 Jan  1  2000 ethernet_c8a030ab323a_cable //NOTE THAT
 -rw-------  1 root root   68 Jan  1  2000 settings
 root@beaglebone:/var/lib/connman# cd ethernet_c8a030ab323a_cable/
 root@beaglebone:/var/lib/connman/ethernet_c8a030ab323a_cable# ls
 data  settings  settings.ODU2WW
 root@beaglebone:/var/lib/connman/ethernet_c8a030ab323a_cable# more settings
 [ethernet_c8a030ab323a_cable]
 Name=Wired
 AutoConnect=true
 Modified=2000-01-01T01:18:21.869401Z
 IPv4.method=dhcp
 IPv4.DHCP.LastAddress=192.168.1.100
 IPv6.method=auto
 IPv6.privacy=disabled
```

➔ goto /usr/lib/conman/test
➔ type;

```
./set-nameservers ethernet_c8a030ab323a_cable 192.168.1.1 8.8.8.8

//note: 8.8.8.8 is DNS and yellow part should be yours cable id.
```

- ➔ type;

```
./set-ipv4-method ethernet_c8a030ab323a_cable manual 192.168.1.80
 255.255.255.0 192.168.1.1

//NOTE: Ethernet name should be yours and yellow ip should be that you want.
```

And the ssh client connection should likely freeze as the IP address of your Beaglebone should have changed. To kill your shh client session type **~.** (i.e. tilda followed by a full stop) in your frozen shell window and this should terminate the ssh client session. Give the Beaglebone a few seconds to reboot and then from your desktop client you should be able to:

```
 type:  ssh 192.168.1.80 -l root
```

OPENCV RELATED (ON LINUX DESKTOP)

1. Firstly, some extra packages should be instaled as follows;

```
➔ sudo apt-get install build-essential
➔ sudo apt-get install cmake git libgtk2.0-dev pkg-config
  libavcodec-dev libavformat-dev libswscale-dev
➔ sudo apt-get install python-dev python-numpy libtbb2
  libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-
  dev libdc1394-22-dev
```

2. Download OpenCV zip file for Unix (mine is 2.4.13)
3. unpack it (mine is /home/dogus/openCV)
4. Insert openCV folder and type follows;
   - ➔ mkdir release
   - ➔ cd release
   - ➔ `cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local ..`

```
(from the page: “cmake -D CMAKE_BUILD_TYPE=RELEASE -D
CMAKE_INSTALL_PREFIX=/usr/local ..” )
```

- ➔ in the release folder; type follows
- sudo make install

(related link: http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html)

NOTE: after installing, you should see the files/folders in the /user/local/ path.

Important that: bin, lib,include folders should generated properly for example include folder should hase opencv and opencv2 folders!!!!!

- ➔ To include these folders to eclipse project, follow these instuctions;
- Project->properties->C/C++ Build->Environment->PATH then edit it and add:
  /usr/local/bin/ and /usr/local/lib/

- Project->properties->C/C++ Build->Settings->CrossGCC Compiler->Includes and add “/usr/local/include” to Include paths
- Under Cross G++ Compiler->Includes and add “/usr/local/include” to Include paths
- Under vCross G++ Linker->Libraries and add “/usr/local/lib” to library search path
- In that tool, again add lib names to Libraries part like “opencv_highgui”, “opencv_imgproc” etc that you need.
- Lastly; do followings;
  - Click on Run As -> Run Configurations
  - On the window on the right hand side you see the Environment tab.
  - Here click on New, you'll see a New Environment Variable pop up.
  - Here, for **Name** enter **LD_LIBRARY_PATH**,
    for **Value** enter**$LD_LIBRARY_PATH:/usr/local/lib**
    click **ok** and **Apply**

  (related link: http://stackoverflow.com/questions/27907343/error-while-loading-shared-libraries-libopencv-core-so-3-0)

  THATS IT!!!!!!!!!!!!

IMPORTANT NOTES: (BBB open cv libs’ path is /usr/lib)

- to compile project (main.cpp file) from Shell on Linux desktop, LD_LIBRARY variable must be set (like in the project scope) like that;
  - ➔ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
- then, to compile the code over desktop again, type following command;
  - ➔ g++ main.cpp –o MAINEXE –lopencv_core –lopencv_highgui –lopencv_imgproc

and so on so fort with library names.

- FOR BBB: LD_PATH variable is already defined so, we can scp to BBB main.cpp and compile it with arm-linux-gnueabi-g++ (sth like that look for your BBB) then type the command like above.
  - ➔ arm-linux-gnueabi-g++ main.cpp –o MAINEXE –lopencv_core –lopencv_highgui –lopencv_imgproc
  - ➔ then you can execute MAINEXE with ./MAINEXE

YOU ARE DONE !!!!!!!!!!!!!!!!!!!!!!

General Note: After write new flash to BBB, you should delete”known_hosts” file under “.ssh “ file.