**System Requirement**

Ubuntu 14.06/ 16.06.

At least 180GB of free space.

## 1. Host Preparing

1.1 Install essential utilities for Yocto project host package:

```
$ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-
multilib build-essential chrpath socat libsdl1.2-dev
```

1.2 Install additional host packages for Ubuntu 14.04/16.04:

```
$ sudo apt-get install libsdl1.2-dev xterm sed cvs subversion coreutils
texi2html docbook-utils python-pysqlite2 help2man make gcc g++ desktop-
file-utils libgl1-mesa-dev libglu1-mesa-dev mercurial autoconf automake
groff curl lzop asciidoc xclip
```

1.3 Install uboot tool packages on Ubuntu 14.04/16.04:

```
$ sudo apt-get install u-boot-tools
```

1.4 Install Repo tool:

```
$ mkdir ~/bin

$ curl http://commondatastorage.googleapis.com/git-repo-downloads/repo >
~/bin/repo

$ chmod a+x ~/bin/repo
```

1.5 Local Git configuration:

This configuration can be skipped if you are already using Git tool actively in your development system.

```
$ git config --global user.name "Your Name"

$ git config --global user.email "Your Email"
```

Verify assigned name and email ID.

```
$ git config –list
```

1.6 Gitlab Configuration:

**Generating SSH Key in Development System.**

* Generate a new SSH keypair:

```
$ ssh-keygen -t rsa -C "your.email@example.com" -b 4096
```

* Save your SSH keypair:

Note: If you do not already have the SSH key pair, use the suggested path by pressing Enter key.

* Enter a password to secure your SSH key pair.

* Copy the public SSH key to clipboard.

```
$ xclip -sel clip < ~/.ssh/id_rsa.pub
```

**Adding SSH Key in GitLab**

* Navigate to [https://gitlab.com/profile](https://gitlab.com/profile) and login with your credentials.

Note: You need to get the access credentials from e-con Systems.

* On the User Settings tab, click SSH Keys.

* Paste the SSH Key which is already copied in clipboard in Key box.

* Enter a title for the Key in Title box.

* Click Add Key to add the key.

* Verify Gitlab configuration.

```
$ ssh -T git@gitlab.com
```

## 2. Downloading Source from GitLab using Repo Tool.

*The source code for the eSOMiMX7 Acacia is maintained in global GitLab (http://www.gitlab.com) with private access. You need to get the access credential from e-con Systems to access the source code. You can write to techsupport@e-consystems.com to get the source code access.*

2.1 Create a new directory for eSOMiMX7 BSP:

In this guide, the BSP directory has name `esomimx7-release-bsp`

```
$ mkdir esomimx7-release-bsp

$ cd esomimx7-release-bsp/
```

2.2 Export the release version of your build. The release version details are in release notes.

```
$ export RELEASE_VERSION="esomimx7-L4.9.11_1.0"
```

2.3 Initialize the source repo then fet it out.

```
$ repo init -u git@gitlab.com:esomimx7-linux-
release/esomimx7_linux_release_manifest.git -b $RELEASE_VERSION

$ repo sync
```

## 3. Build the BSP:

3.1 Setup build parameters:

```
$ DISTRO=<distro name> MACHINE=<machine name> source esomimx7-setup-release.sh -b <build dir>
```

In this guide, the build directory BUILD_DIR is `out_dual-2gb`

```
$ DISTRO=esomimx7-x11 MACHINE=esomimx7d-2gb source esomimx7-setup-
release.sh -b out_dual-2gb
```

3.2 Add following options to configuration file at [BUILD_DIR]/conf/local.conf:

```
EXTRA_IMAGE_FEATURES ?= " \
      ssh-server-openssh \
      tools-debug \
      eclipse-debug \
      debug-tweaks \
      "
IMAGE_INSTALL_append = " \
      gdbserver \
      tcf-agent \
      packagegroup-core-ssh-openssh \
      openssh \
      openssh-sftp \
      openssh-sftp-server \
      "
```

3.3 Build all necessary images for eSOMiMX7:

```
$ bitbake esomimx7-image-gui
```

3.4 Build the tool-chain for development on eSOMiMX7:

```
$ bitbake meta-toolchain
```

## 4. Create the NFS root folder.

4.1 Create folder for NFS:

Here USER is the name of your machine.

```
$ mkdir home/USER/nfs
```

4.2 Copy root file system from build directory to NFS folder:

```
$ sudo cp -a BUILD_DIR/tmp/work/esomimx7d_2gb-poky-linux-
gnueabi/esomimx7-image-gui/1.0-r0/rootfs/.  home/USER/nfs
```

4.3 Change the owner of the NFS rootfs to `root`:

```
$ sudo chown -R root home/USER/nfs/rootfs
```

## 5. Setup NFS on the host.

5.1 Install necessary packages for the host:

```
$ sudo apt-get install xinetd tftp tftpd isc-dhcp-server nfs-kernel-
server portmap
```

5.2 Edit the /etc/exports file

```
$ sudo gedit /etc/exports
```

Add the path of NFS rootfs to be exported:

```
# /etc/exports: the access control list for filesystems which may be
exported
#           to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check)
hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/home/USER/nfs/rootfs
*(rw,sync,no_root_squash,no_all_squash,no_subtree_check)
```

Save and close the file.

5.3 Restart NFS service:

```
$ sudo service portmap stop

$ sudo service nfs-kernel-server stop

$ sudo service portmap start

$ sudo service nfs-kernel-server start
```

## 6. Setup TFTP on the host.

6.1 Edit /etc/xinetd.d/tftp:

```
$ sudo gedit /etc/xinetd.d/tftp
```

Add the path of images to the tftp configuration:

```
service tftp
{
protocol = udp
port = 69
socket_type = dgram
wait = yes
user = nobody
server = /usr/sbin/in.tftpd
server_args = -s /home/USER/esomimx7-release-bsp/out_dual-
2gb/tmp/deploy/images/esomimx7d-2gb
disable = no
}
```

6.2 Restart the TFTP service:

```
$ sudo service xinetd restart
```

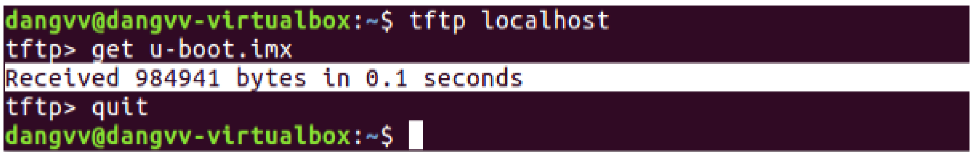
6.3 Verify that TFTP is working:

```
$ tftp localhost

tftp> get u-boot.imx

tftp> quit
```

If TFTP worked well, the number of bytes transferred will be show.



```
dangvv@dangvv-virtualbox:~$ tftp localhost
tftp> get u-boot.imx
Received 984941 bytes in 0.1 seconds
tftp> quit
dangvv@dangvv-virtualbox:~$
```

## 7. Setup the target board

7.1 Power on the board. Hit any key to prevent Uboot load the kernel.

7.2 Set the ip address for board:

```
setenv serverip 192.168.1.100

setenv ipaddr 192.168.1.102

setenv ip_dyn no
```

7.3 Set the path of NFS rootfs folder on the host to the U-boot:

```
setenv nfsroot /home/USER/nfs/rootfs
```

7.4 Set the image to the U-boot:

```
setenv image zImage
setenv fdt_file uImage-imx6q-sabresd.dtb
```

7.5 Set boot arguments to U-boot:

```
setenv netargs 'setenv bootargs console=${console},${baudrate} ${smp}
root=/dev/nfs ip=${ipaddr} nfsroot=${serverip}:${nfsroot},v3,tcp rw'
```

7.6 Finally, boot the board using NFS and TFTP:

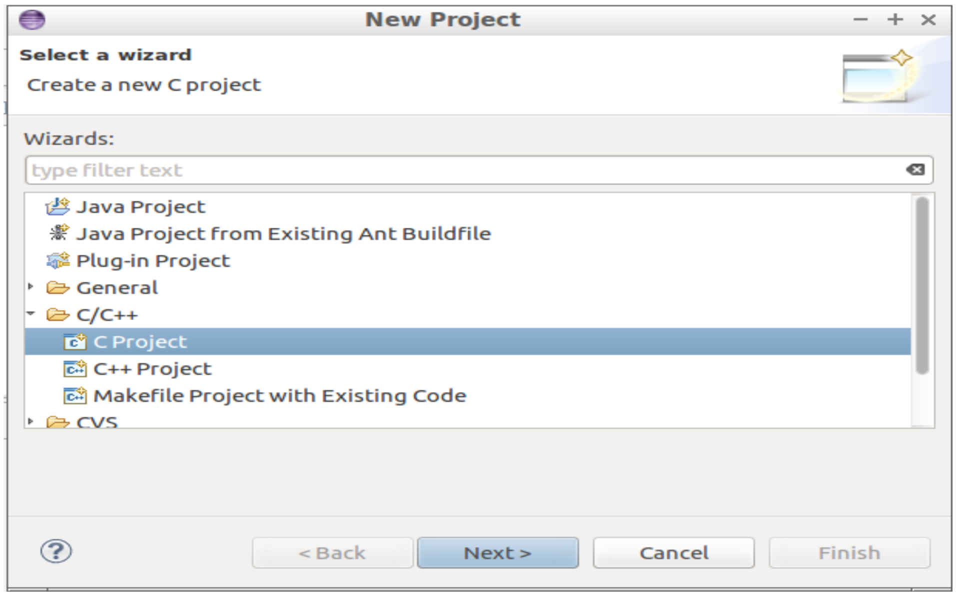
```
run netboot
```

## 13. “Hello word” Project.

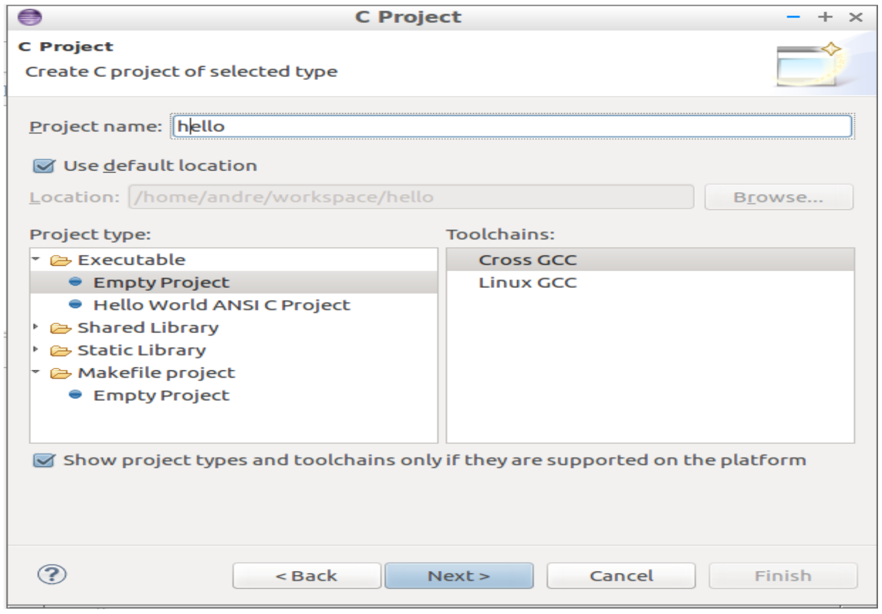
13.1 Start Eclipse:

13.2 Create new project:

Create a new project by going to File  New  Project in the menu. Choose a “C Project” as shown in figure and then click “Next”.



Enter a project name (“hello” in this example). Select project type as an “Empty project” and set toolchain to be “Cross GCC”.


C Project

C Project
Create C project of selected type

Project name: hello

☑ Use default location

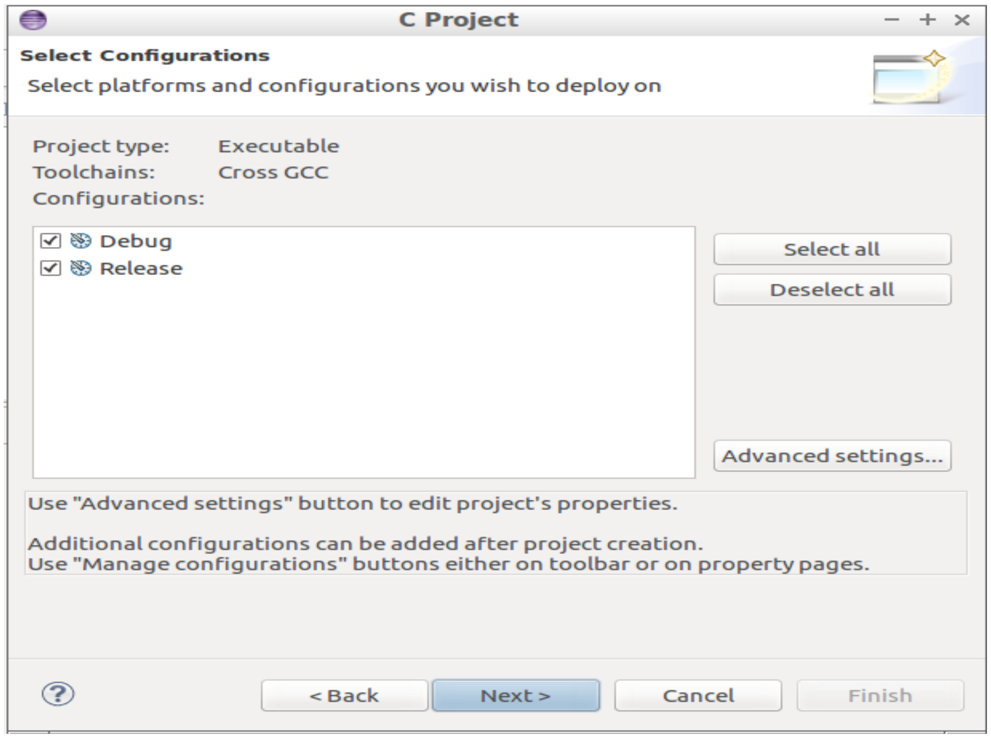
Location: /home/andre/workspace/hello

Project type:
- Executable
  - Empty Project
  - Hello World ANSI C Project
- Shared Library
- Static Library
- Makefile project
  - Empty Project

Toolchains:
- Cross GCC
- Linux GCC

☑ Show project types and toolchains only if they are supported on the platform

Click “Next” and then just use the default settings in the “Select configurations” dialog.


C Project

Select Configurations
Select platforms and configurations you wish to deploy on

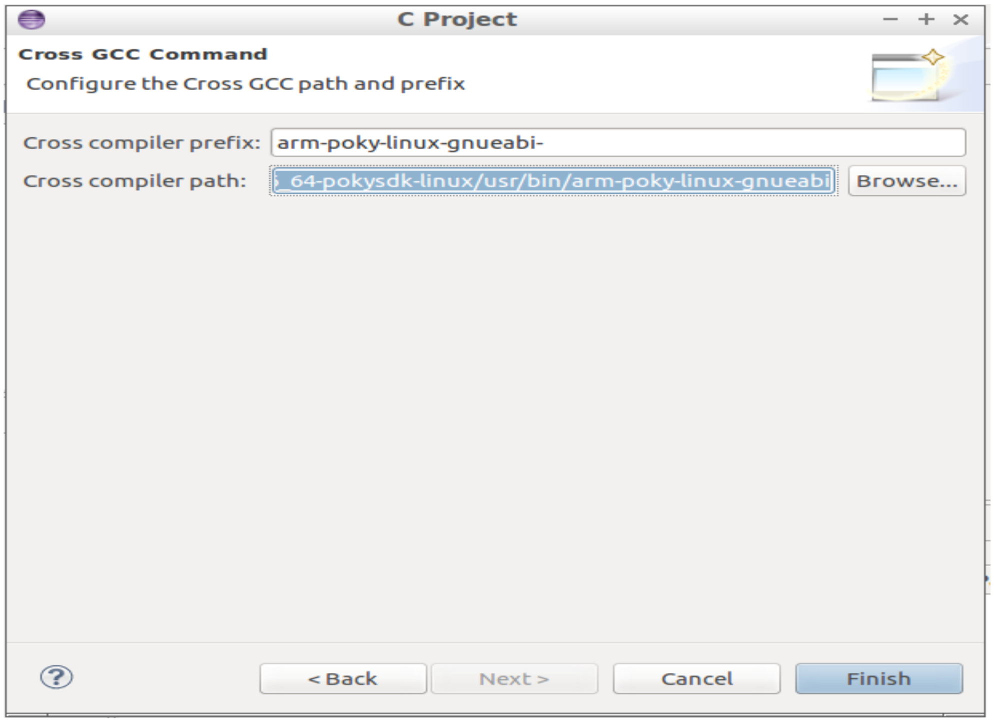
Project type: Executable
Toolchains: Cross GCC
Configurations:

☑ Debug
☑ Release

Use "Advanced settings" button to edit project's properties.

Additional configurations can be added after project creation.
Use "Manage configurations" buttons either on toolbar or on property pages.

Click “Next”. Now it is time to set the path to the cross-compiler. Set the prefix to `arm-poky-linux-gnueabi-`. Set the path to (change if you have installed the toolchain elsewhere) `/opt/fsl-imx-fb/4.1.15-1.2.0/sysroots/x86_64-pokysdk-linux/usr/bin/arm-poky-linux-gnueabi`. When you are done click the “Finish” button



Cross GCC Command

Configure the Cross GCC path and prefix

Cross compiler prefix: arm-poky-linux-gnueabi-

Cross compiler path: _64-pokysdk-linux/usr/bin/arm-poky-linux-gnueabi

## 13.3 Create the application