Yocto Project and OpenEmbedded Training

Raspberry Pi 3 Model B+ variant

Practical Guide

https://banvien.com

January 05, 2022

About this document

Updates to this document can be found on http://gitlab.banvien.com/dy.le-son/doc/blob/master/full-bv-labs.pdf.

This document was generated from LaTeX sources found on http://gitlab.banvien.com/dy.le-son/latex_doc.

Copying this document

Training setup

Download files and directories used in practical labs

Install lab data

For the lab in this guide, your instructor has prepared a set of data (bblayers.conf, local.conf). Download its from a terminal:

```
$ cd
$ git clone http://gitlab.banvien.com/dy.le-son/yocto_config.git
```

`git clone` need account to finish, please enter `Username ex: dy.le-son` and `Password ex: xxxxxx`. Lab data are now available in an directory in your home directory. This directory contains files used in the this practical lab. It will also be used as working space, in particular to keep generated files separate when needed.

Update your distribution

To avoid any issue installing packages during the practical labs, you should apply the latest updates to the packages in your distro:

```
$ sudo apt update
$ sudo apt dist-upgrade
```

You are now ready to start the real practical labs!

Install extra packages

Feel free to install other packages you may need for your development environment. In particular, we recommend to install your favorite text editor and configure it to your taste. The favorite text editors of embedded Linux developers are of course *Vim* and *Emacs*, but there are also plenty of other possibilities, such as Visual Studio Code[1], *GEdit*, *Qt Creator*, *CodeBlocks*, *Geany*, etc.

It is worth mentioning that by default, Ubuntu comes with a very limited version of the `vi` editor. So if you would like to use `vi`, we recommend to use the more featureful version by installing the `vim` package.

More guidelines

Can be useful throughout any of the labs

- Read instructions and tips carefully. Lots of people make mistakes or waste time because they missed an explanation or a guideline.
- Always read error messages carefully, in particular the first one which is issued. Some people stumble on very simple errors just because they specified a wrong file path and didn't pay enough attention to the corresponding error message.
- Never stay stuck with a strange problem more than 5 minutes. Show your problem to your colleagues or to the instructor.

[1]This tool from Microsoft is Open Source! To try it on Ubuntu: `sudo snap install code --classic`

- You should only use the `root` user for operations that require super-user privileges, such as: mounting a file system, loading a kernel module, changing file ownership, configuring the network. Most regular tasks (such as downloading, extracting sources, compiling...) can be done as a regular user.
- If you ran commands from a root shell by mistake, your regular user may no longer be able to handle the corresponding generated files. In this case, use the `chown -R` command to give the new files back to your regular user.
  Example: `$ chown -R myuser.myuser linux/`

First Yocto Project build

Your first dive into Yocto Project and its build mechanism

During this lab, you will:

- Set up an OpenEmbedded environment
- Configure the project and choose a target
- Build your first Poky image

Setup

Before starting this lab, make sure your home directory is not encrypted. OpenEmbedded cannot be used on top of an eCryptFS file system due to limitations in file name lengths.

```
$ cd
$ sudo mkdir $HOME/bv-labs
```

Go to the `$HOME/bv-labs/` directory.

```
$ cd $HOME/bv-labs
```

Install the required packages:

```
$ sudo apt install gawk wget git diffstat unzip texinfo gcc build-essential \
    chrpath socat cpio python3 python3-pip python3-pexpect xz-utils debianutils \
    iputils-ping python3-git python3-jinja2 libegl1-mesa libsdl1.2-dev pylint3 \
    xterm python3-subunit mesa-common-dev zstd liblz4-tool
```

Download Yocto

Download the `honister` version of Poky:

```
$ git clone -b honister --single-branch git://git.yoctoproject.org/poky.git
$ cd $HOME/bv-labs/poky
$ git checkout 4ce984316d038eaed2f48533b31d0a1898942f0a
```

At poky directory (`$HOME/bv-labs/poky`), download the `meta-openembedded` and `meta-security`, `meta-raspberrypi` layers:

```
$ git clone -b honister --single-branch git://git.openembedded.org/\
    meta-openembedded.git
$ git clone -b honister --single-branch git://git.yoctoproject.org/\
    meta-security.git
$ git clone -b honister --single-branch git://git.yoctoproject.org/\
    meta-raspberrypi.git
$ cd $HOME/bv-labs/poky/meta-openembedded
$ git checkout a12eff7e3f6aa1d4150b0724d4b7939622aa598f
$ cd $HOME/bv-labs/poky/meta-security
$ git checkout fb77606aef461910db4836bad94d75758cc2688c
$ cd $HOME/bv-labs/poky/meta-raspberrypi
$ git checkout 1584bddcf31f4bf3acc2304aa8dae927e8bec970
```

Set up the build environment

Check you're using Bash. This is the default shell when using Ubuntu.

Export all needed variables and set up the build directory:

```
$ cd $HOME/bv-labs
$ source poky/oe-init-build-env
```

You must specify which machine is your target. By default it is `qemu`. We need to build an image for a `raspberrypi3-B+`. Update the `MACHINE` configuration variable accordingly.

Also, if you need to save disk space on your computer you can add `INHERIT += "rm_work"` in the previous configuration file. This will remove the package work directory once a package is built.

Edit the layer configuration file (`$BUILDDIR/conf/bblayers.conf`) and append the full path to the `meta-raspberrypi`, `meta-poky`, `meta-security`, etc..., directories to the `BBLAYERS` variable.

To convenience for build a first example image, you need to use configuration file has cloned in step above.

```
$ mv $HOME/yocto_config/bblayers.conf $HOME/bv-labs/build/conf
$ mv $HOME/yocto_config/local.conf $HOME/bv-labs/build/conf
```

Build your first image

Now that you're ready to start the compilation, simply run:

```
$ bitbake core-image-weston
```

Once the build finished, you will find the output images under

`$HOME/bv-labs/build/tmp/deploy/images/raspberrypi3-64`.

Set up the SD card

In this first lab we will use an SD card to store the bootloader, kernel and root filesystem files. The SD card image has been generated and is named `core-image-weston-raspberrypi3-64.wic.bz2`.

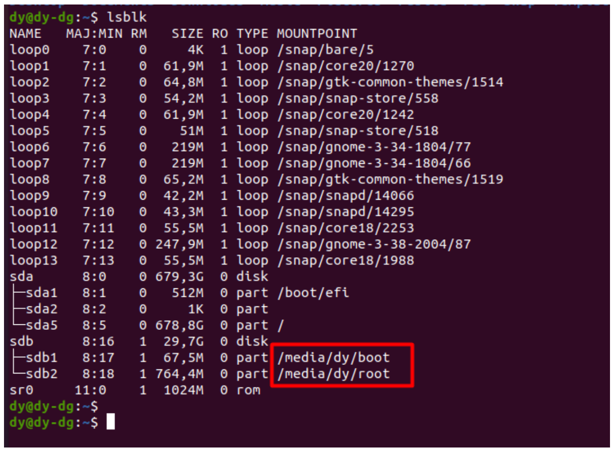
Now uncompress and flash the image with the following command:

```
$ sudo bmaptool copy core-image-weston-raspberrypi3-64.wic.bz2 /dev/sdX
```

`/dev/sdX` you need to check check the name of SD card device by cmd `lsblk`

In case your SD card has mounted, you have to unmount the device:

For example:



```
dy@dy-dg:~$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0    7:0    0     4K  1 loop /snap/bare/5
loop1    7:1    0  61,9M  1 loop /snap/core20/1270
loop2    7:2    0  64,8M  1 loop /snap/gtk-common-themes/1514
loop3    7:3    0  54,2M  1 loop /snap/snap-store/558
loop4    7:4    0  61,9M  1 loop /snap/core20/1242
loop5    7:5    0    51M  1 loop /snap/snap-store/518
loop6    7:6    0   219M  1 loop /snap/gnome-3-34-1804/77
loop7    7:7    0   219M  1 loop /snap/gnome-3-34-1804/66
loop8    7:8    0  65,2M  1 loop /snap/gtk-common-themes/1519
loop9    7:9    0  42,2M  1 loop /snap/snapd/14066
loop10   7:10   0  43,3M  1 loop /snap/snapd/14295
loop11   7:11   0  55,5M  1 loop /snap/core18/2253
loop12   7:12   0 247,9M  1 loop /snap/gnome-3-38-2004/87
loop13   7:13   0  55,5M  1 loop /snap/core18/1988
sda      8:0    0 679,3G  0 disk 
├─sda1   8:1    0   512M  0 part /boot/efi
├─sda2   8:2    0     1K  0 part 
└─sda5   8:5    0 678,8G  0 part /
sdb      8:16   1  29,7G  0 disk 
├─sdb1   8:17   1  67,5M  0 part /media/dy/boot
└─sdb2   8:18   1 764,4M  0 part /media/dy/root
sr0     11:0    1  1024M  0 rom  
dy@dy-dg:~$
dy@dy-dg:~$
```

```
$ sudo umount /media/dy/boot
$ sudo umount /media/dy/root
$ sudo bmaptool copy core-image-weston-raspberrypi3-64.wic.bz2 /dev/sdX
```

Insert the SD card in the dedicated slot on the Raspberrypi3. Plug in the USB cable, HDMI cable and plug in the power. You should see boot flash on the screen.

Congratulations! The board has booted.