

AT91 Buildroot development environment setup based on ubuntu-18.04.1-desktop

Table of Contents

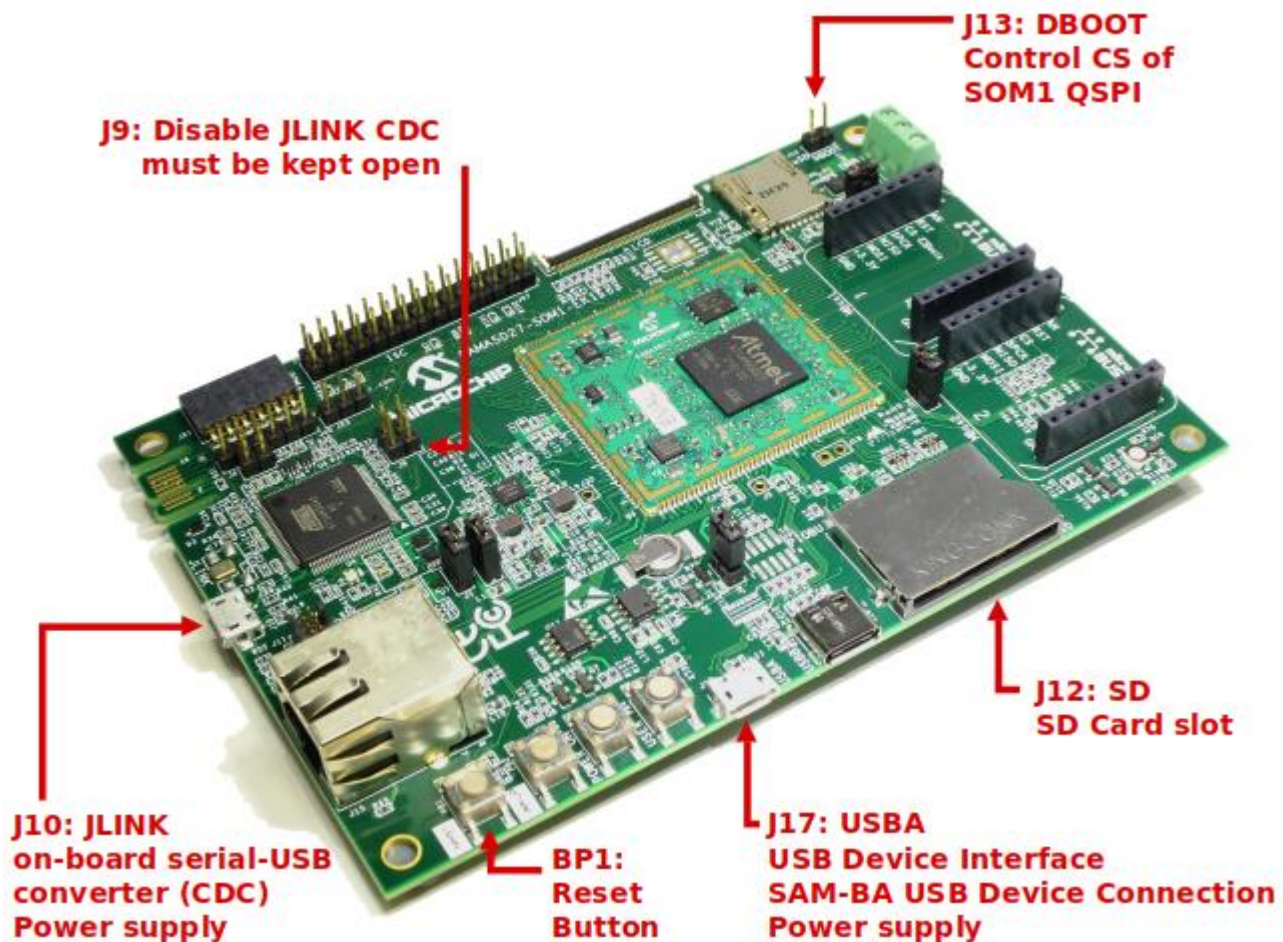
Software requirements.....	2
Hardware requirements	3
1. Install Ubuntu Desktop and applications.....	4
1.1 Install Ubuntu Desktop	4
1.2 Install Ubuntu applications	22
1.3 Reference information of Ubuntu installation	33
2. Setup AT91 development environment and build.....	34
2.1 Linux4sam.....	34
2.2 Buildroot	35
2.3 Quick setup using pre-downloaded Buildroot package.....	36
3. Burn SD Card and run it	38
4. Configuration of target boards in Buildroot	40
5. Cross toolchain in Buildroot	41
6. How to develop at91bootstrap in Buildroot.....	42
7. How to develop u-boot in Buildroot.....	44
8. How to develop kernel in Buildroot.....	45
9. How to update Buildroot project files	46
10. AT91 targets in pre-downloaded Buildroot package.....	48
11. How to make pre-downloaded Buildroot package	49
Appendix.....	51
A. How to check md5sum	51
B. How to turn Windows features on or off.....	52

Software requirements

- ubuntu-18.04.1-desktop-amd64.iso
<https://www.ubuntu.com/download/desktop>
- rufus-3.3
<https://rufus.ie/en IE.html>
- Etcher
<https://www.balena.io/etcher/>
- TeraTerm
<http://ttssh2.osdn.jp/>
- WinMD5
<http://www.winmd5.com/>

Hardware requirements

- Your PC (x64-based processor)
Since Ubuntu 18.04 32-bit processor will not be supported.
- SAMA5D27 SOM1 Evaluation Kit
- SD Card



1.Install Ubuntu Desktop and applications

1.1 Install Ubuntu Desktop

1.1.1 Ubuntu installation image

There are two different instances of Ubuntu system, Ubuntu Desktop and Ubuntu Server.

Some differences between the two instances, the most important difference for us is that:

Ubuntu Desktop has a more friendly user interface based on graphic (Ubuntu Server use text console).

Ubuntu Desktop is a good choice for getting start.

This hands-on is based on version of following pre-downloaded Ubuntu installation image:

[ubuntu-18.04.1-desktop-amd64.iso](#)

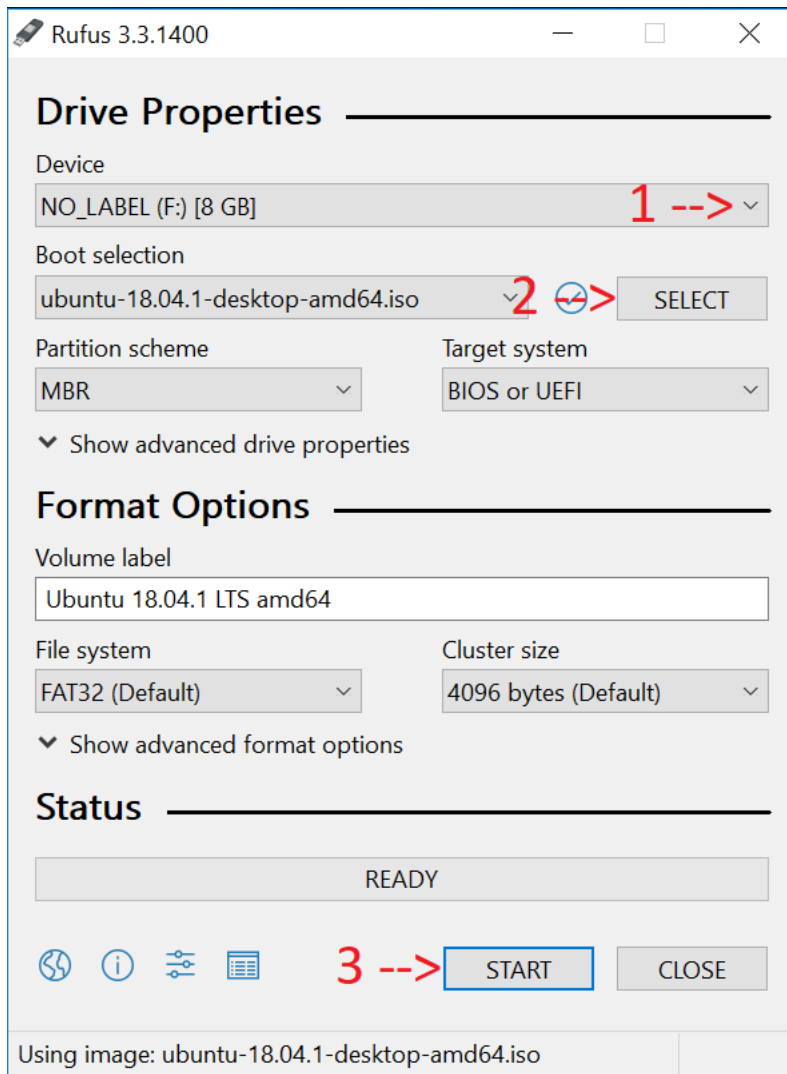
Ubuntu installation image could be downloaded from:

<https://www.ubuntu.com/download>

notice: LTS means Long Term Support

1.1.2 Create Ubuntu installation U-disk

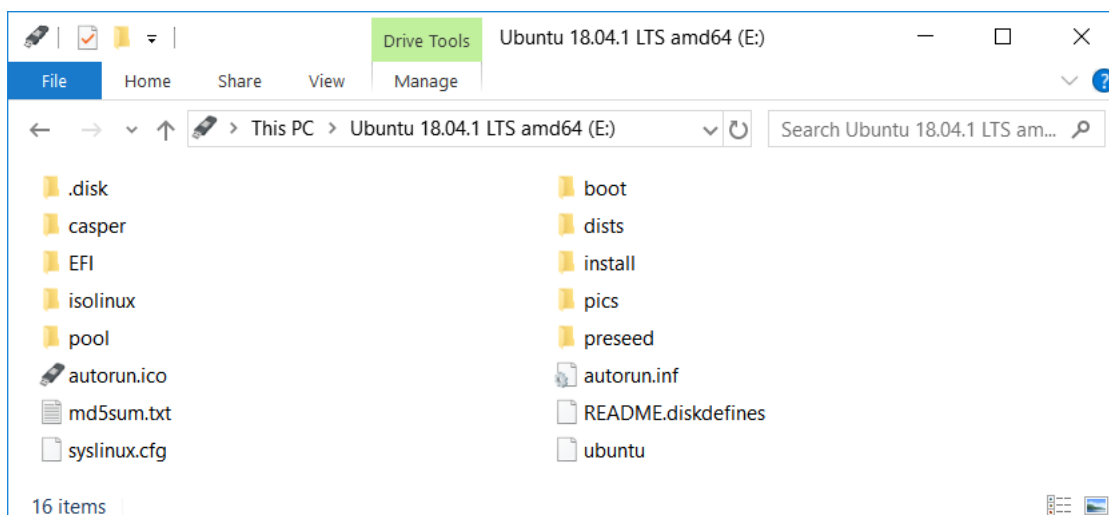
- Insert an empty U-disk into you PC, and open rufus-3.3



Three steps to create an installation U-disk:

1. Drop down and select your empty U-disk device.
2. Select Ubuntu installation image.
3. Click start button.

After successful programming following files could be found in U-disk:



1.1.3 Install Ubuntu Desktop

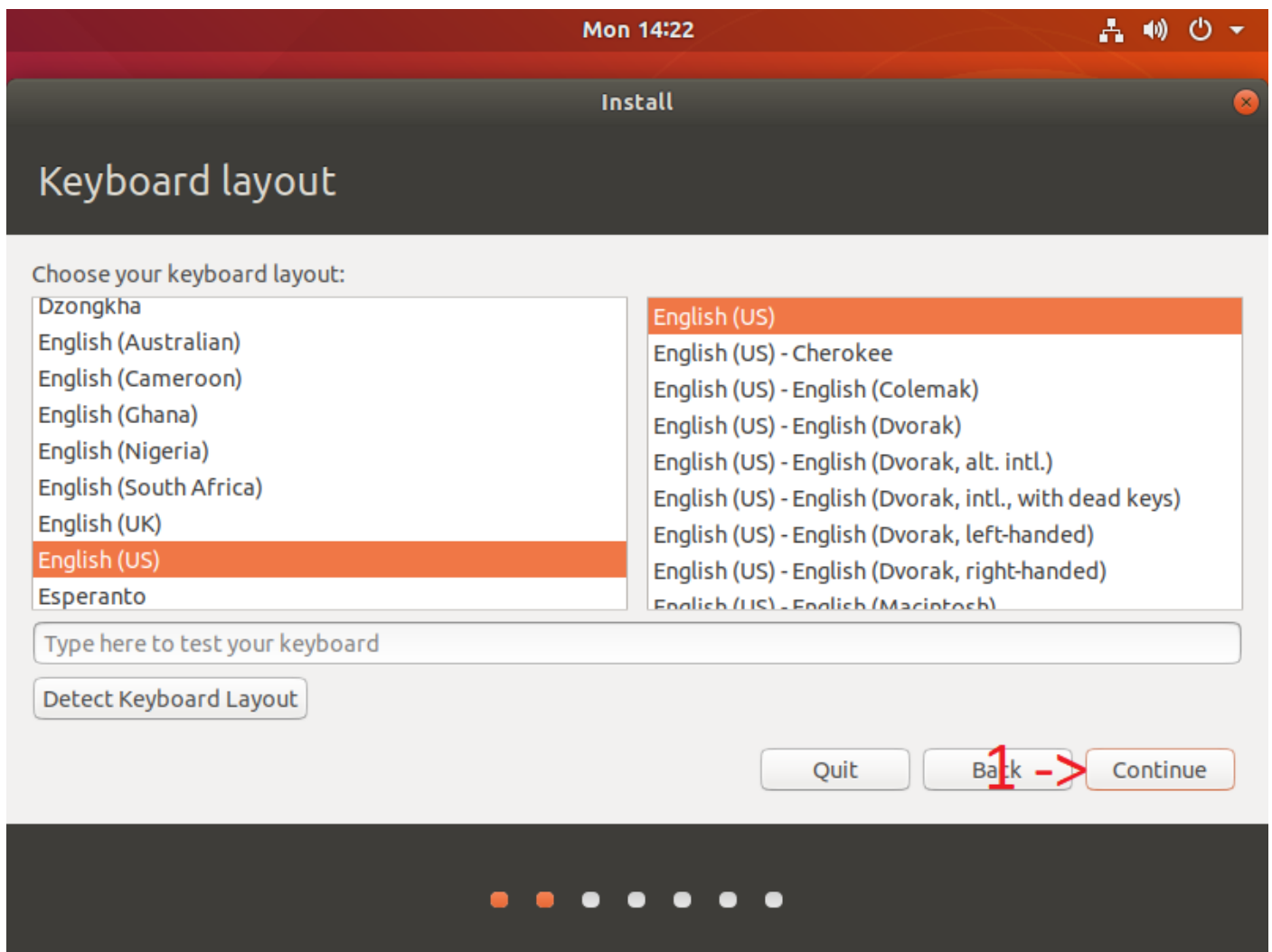
To ensure your PC will boot from U-disk, press F8 before power up it, other ways please check PC BIOS configuration first.

- Insert Ubuntu installation U-disk into your PC.
- Press F8 before power up your PC, select boot from U-disk device.
- After booting from U-disk successful, PC will enter Ubuntu installation interface.
- Select language and install Ubuntu.



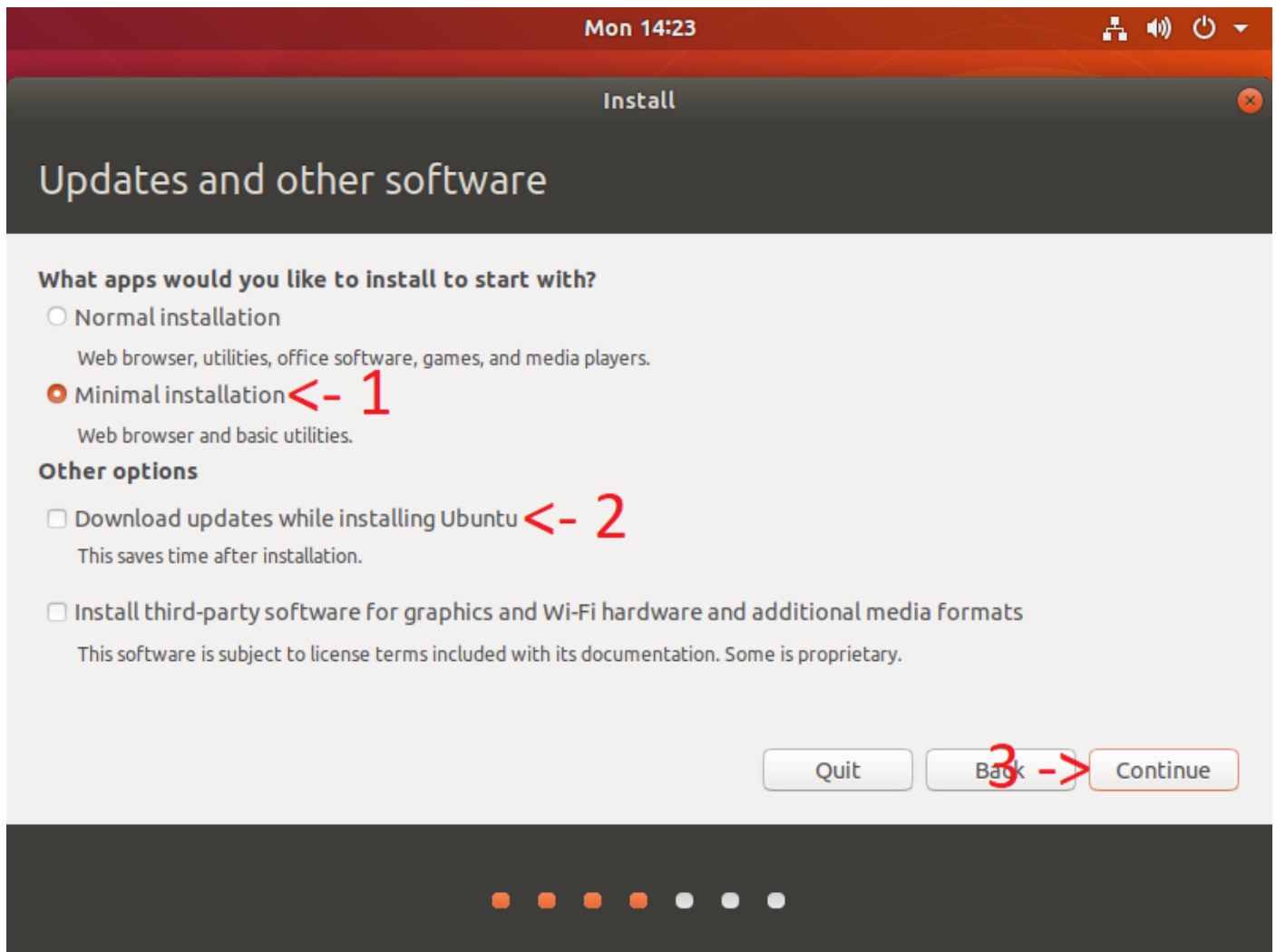
1. Select "English" (or select "Chinese")
2. click "Install Ubuntu"

- Select Keyboard layout



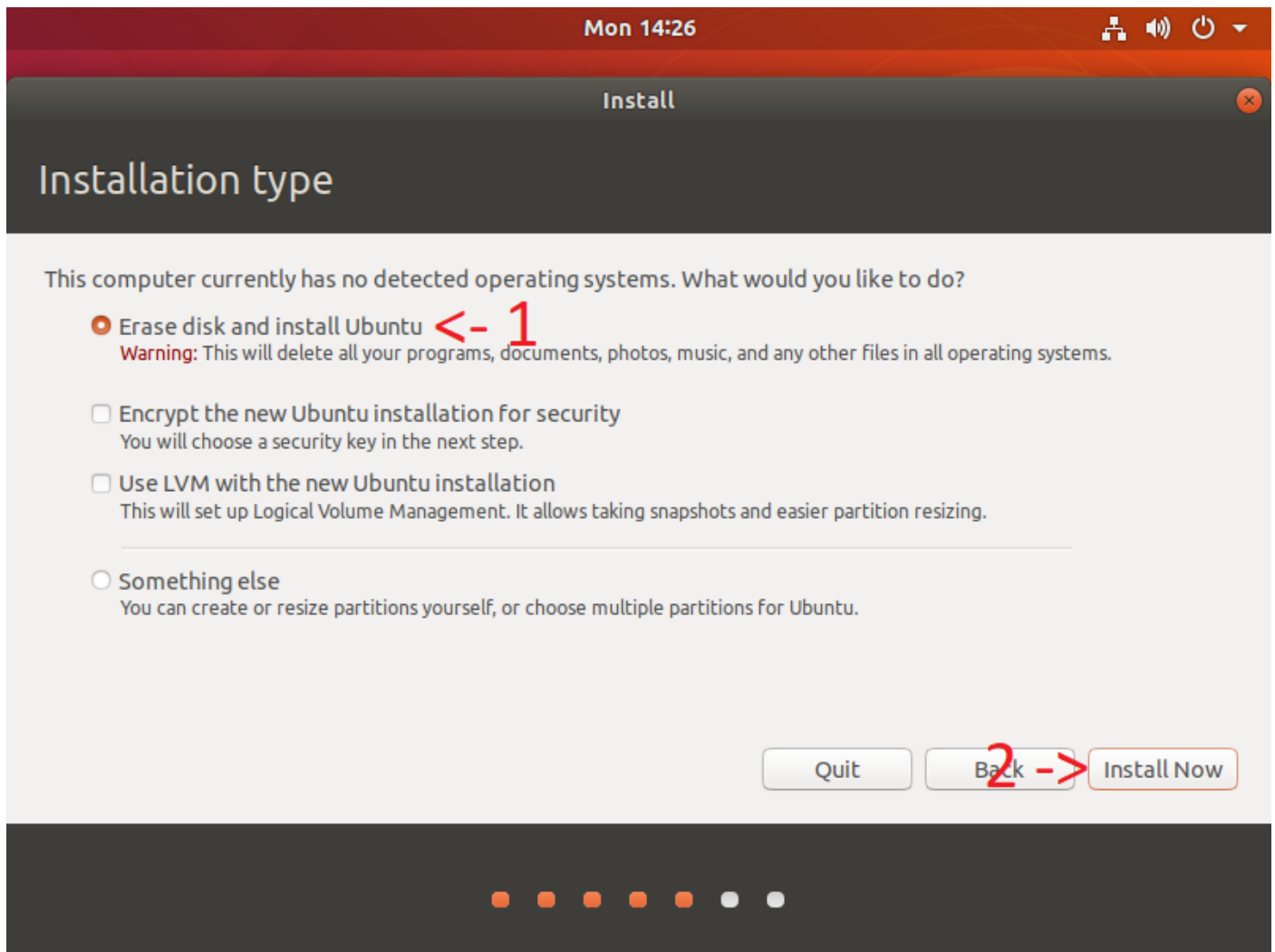
1. Click "Continue"

- "Minimal installation" is a good choice which will speed up installation.
Deselect "Download updates while installing Ubuntu" also will speed up installation.



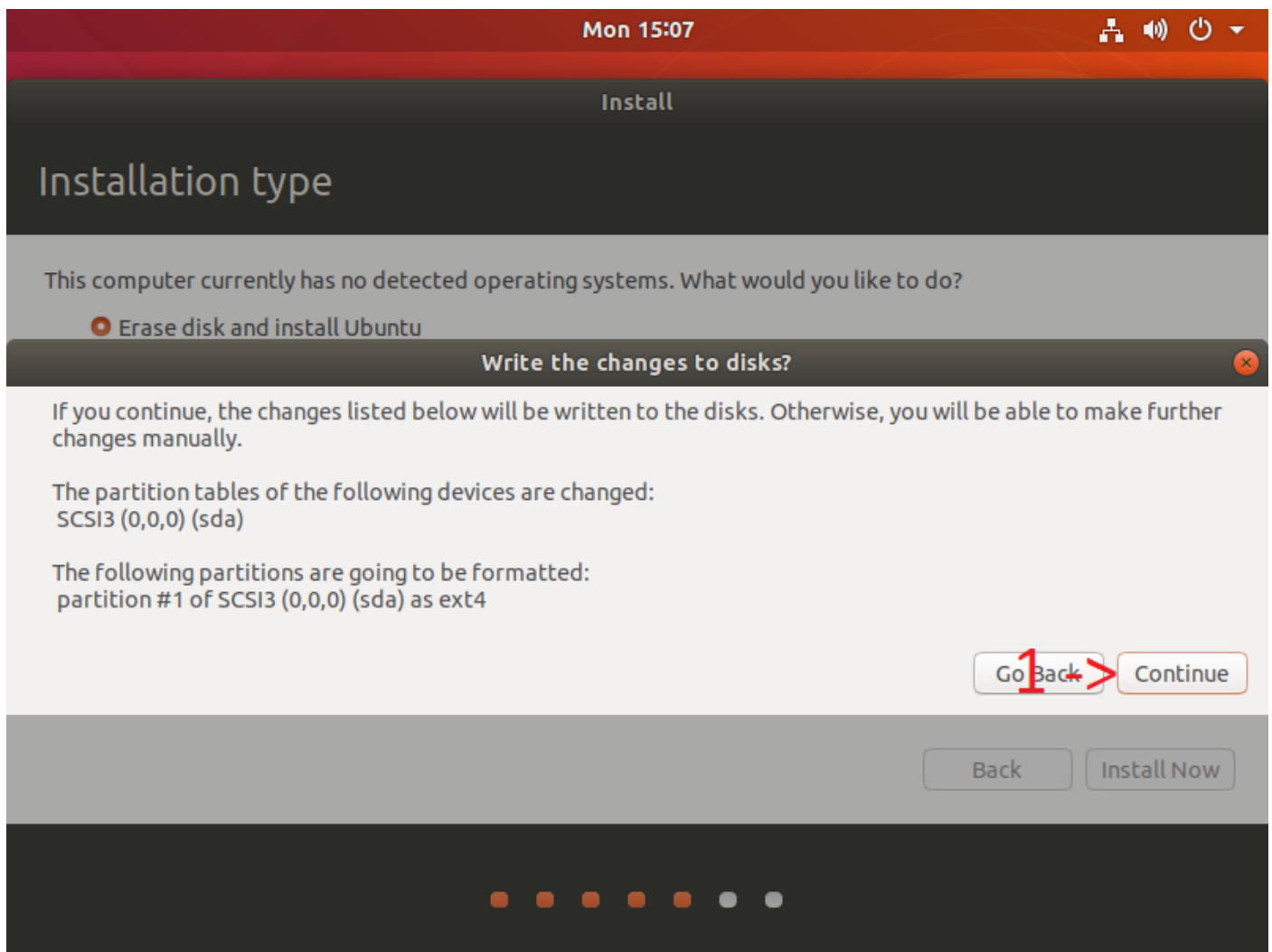
1. Select "Minimal Installation"
2. Unselect "Download updates while installing Ubuntu"
3. Click "Continue"

- Choose installation type.



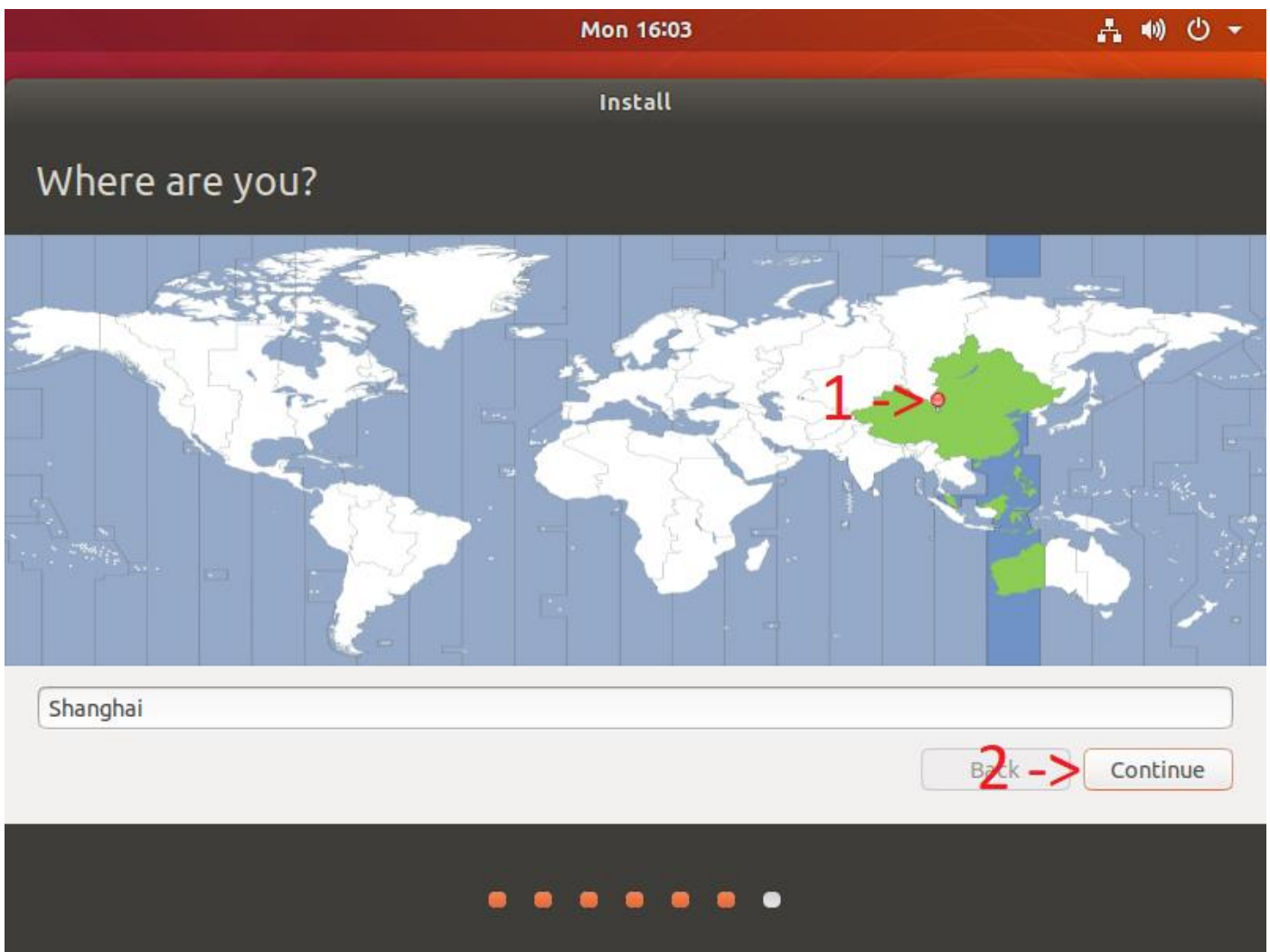
1. Select "Erase disk and install Ubuntu"
2. Click "Install Now"

- Confirm “Write the changes to disks”



1. Click “Continue”

- Select your time zone



1. Click the position of China
2. Click "Continue"

- Fill in account information include user name and password which will be used for login Ubuntu.

The screenshot shows the 'Who are you?' screen in the Ubuntu installer. At the top, the time is 'Tue 00:04' and there are icons for network, volume, and power. The title bar says 'Install'. The main heading is 'Who are you?'. Below this, there are several input fields and checkboxes:

- Your name:** 'user' (marked with a red '1' and an arrow, and a green checkmark).
- Your computer's name:** 'at91' (with a green checkmark). Below it, a note says 'The name it uses when it talks to other computers.'
- Pick a username:** 'user' (with a green checkmark).
- Choose a password:** masked with dots, labeled 'Fair password'.
- Confirm your password:** masked with dots, with a green checkmark.
- Two radio buttons: 'Log in automatically' (unselected) and 'Require my password to log in' (selected).
- At the bottom right, there are 'Back' and 'Continue' buttons. The 'Continue' button is marked with a red '2' and an arrow.

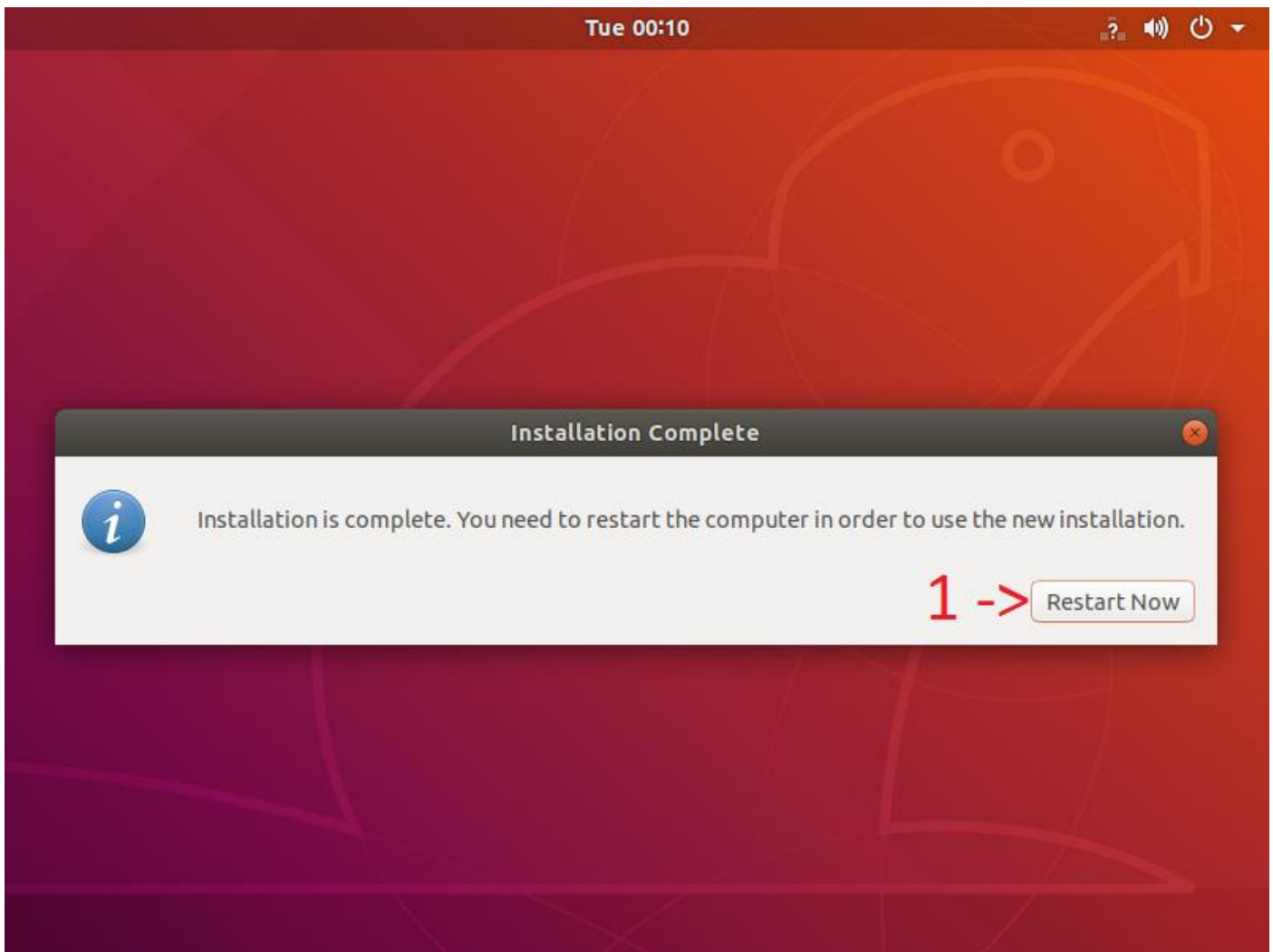
At the very bottom, there is a progress bar with seven orange squares.

1. Fill in your name, user name and password
Username: user
Password : 123456
2. Click "Continue"

- Installing system, it will take a while

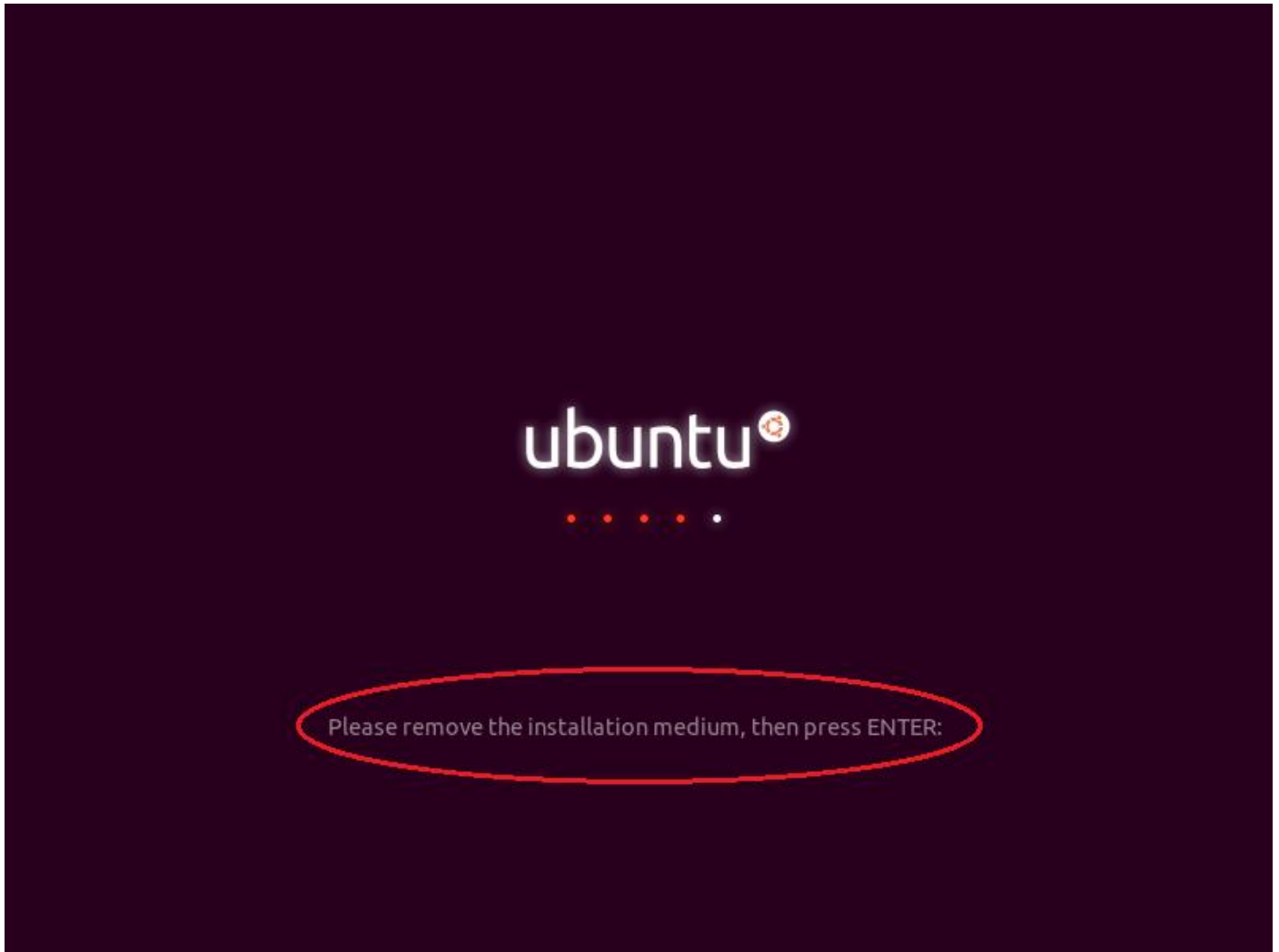


- Installation is complete, please remove U-disk and restart your PC

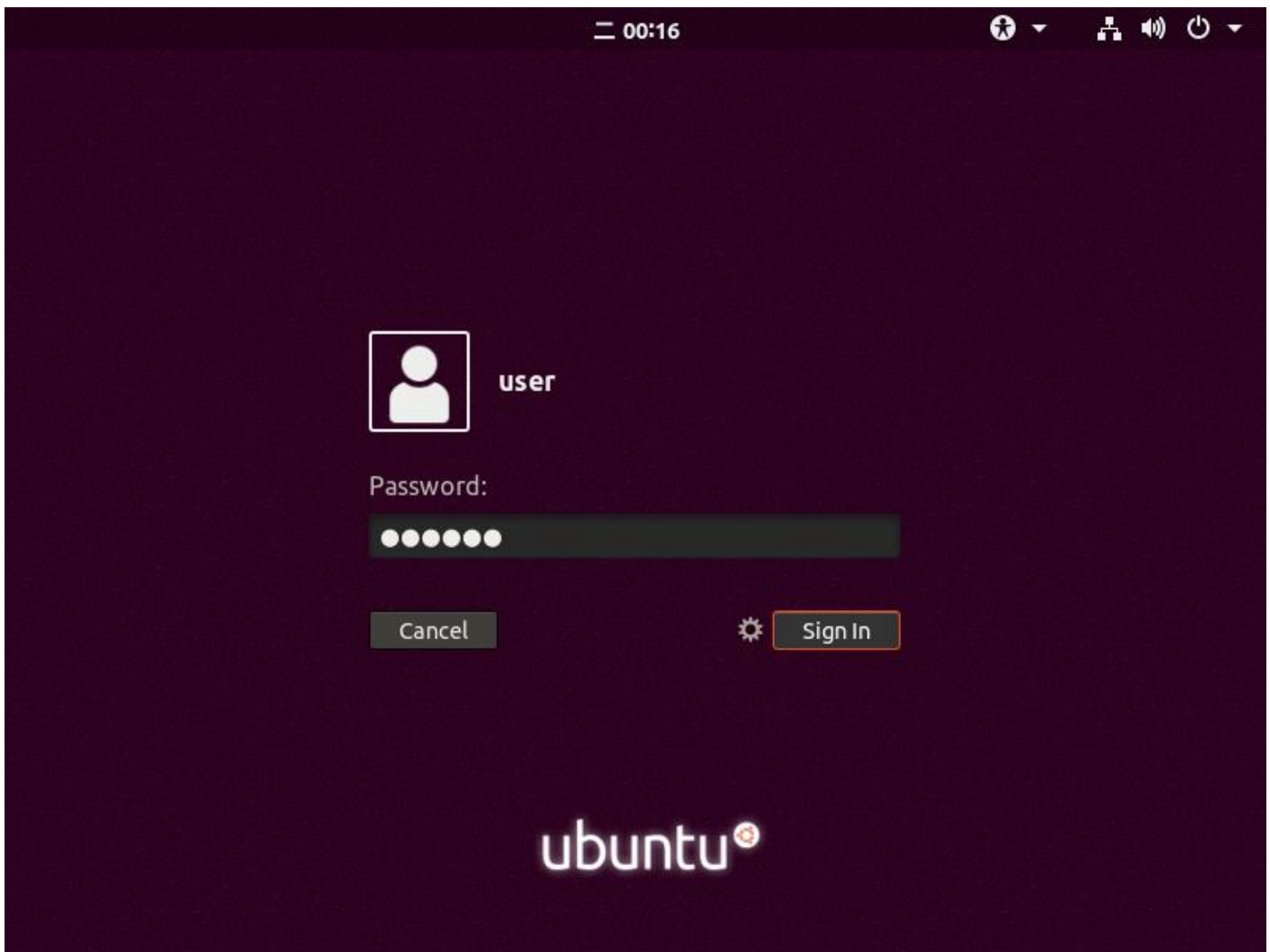


1. Remove U-disk, then click "Restart Now"

- If U-disk wasn't removed before restarting PC, Ubuntu system will be halted in following UI
Please remove U-disk, then press ENTER key.



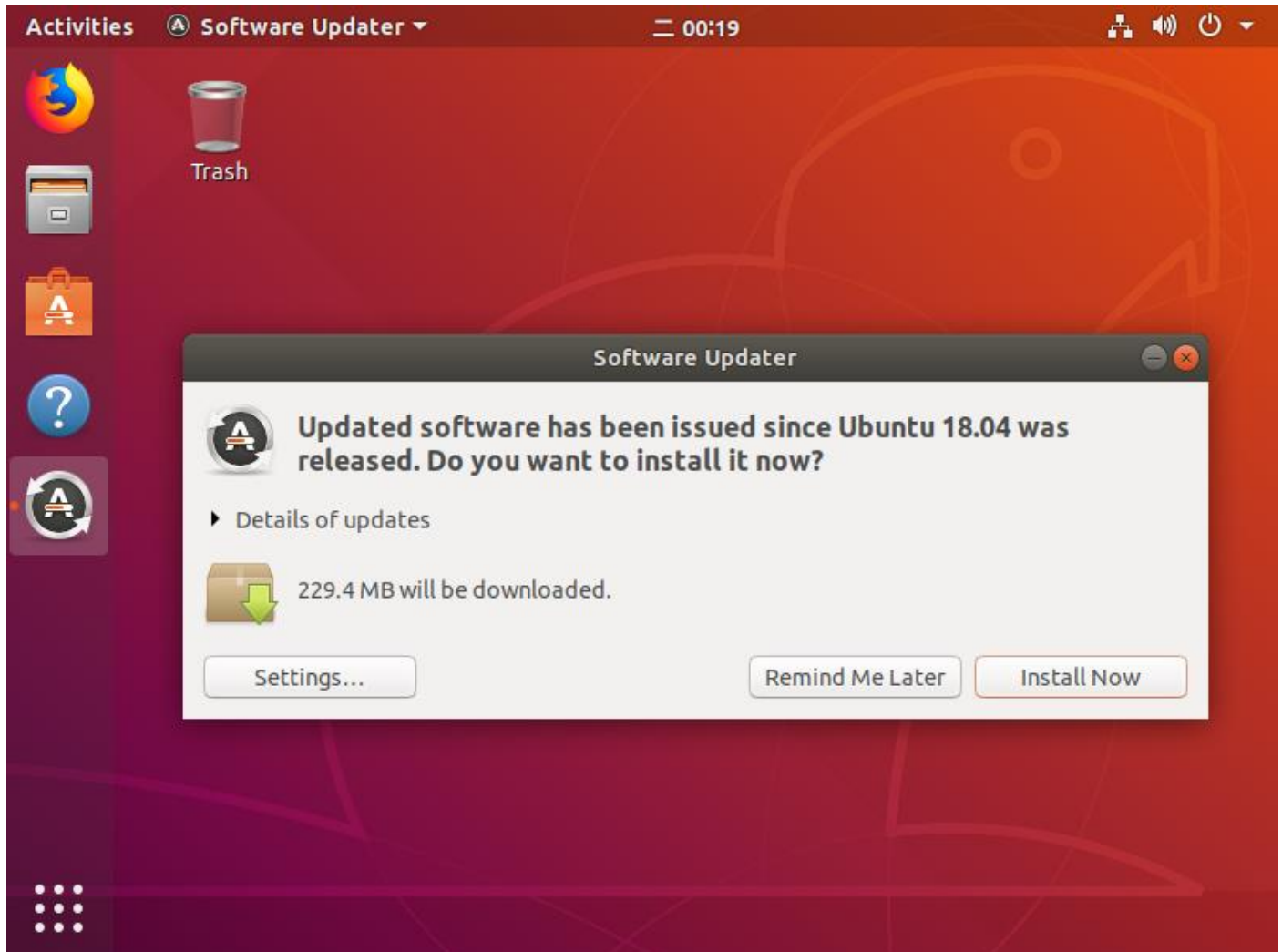
- After restarting successful, will enter Ubuntu login UI



1.1.4 Upgrade Ubuntu Desktop (optional)

Notice: upgrade Ubuntu Desktop may take a long time, you better do it in idle time.

Now Ubuntu Desktop has been installed successfully, "Software Updater" window will pop up for updating after first login:



You also could do it with following commands manually:

```
user@at91:~$ sudo apt update
[...]
```

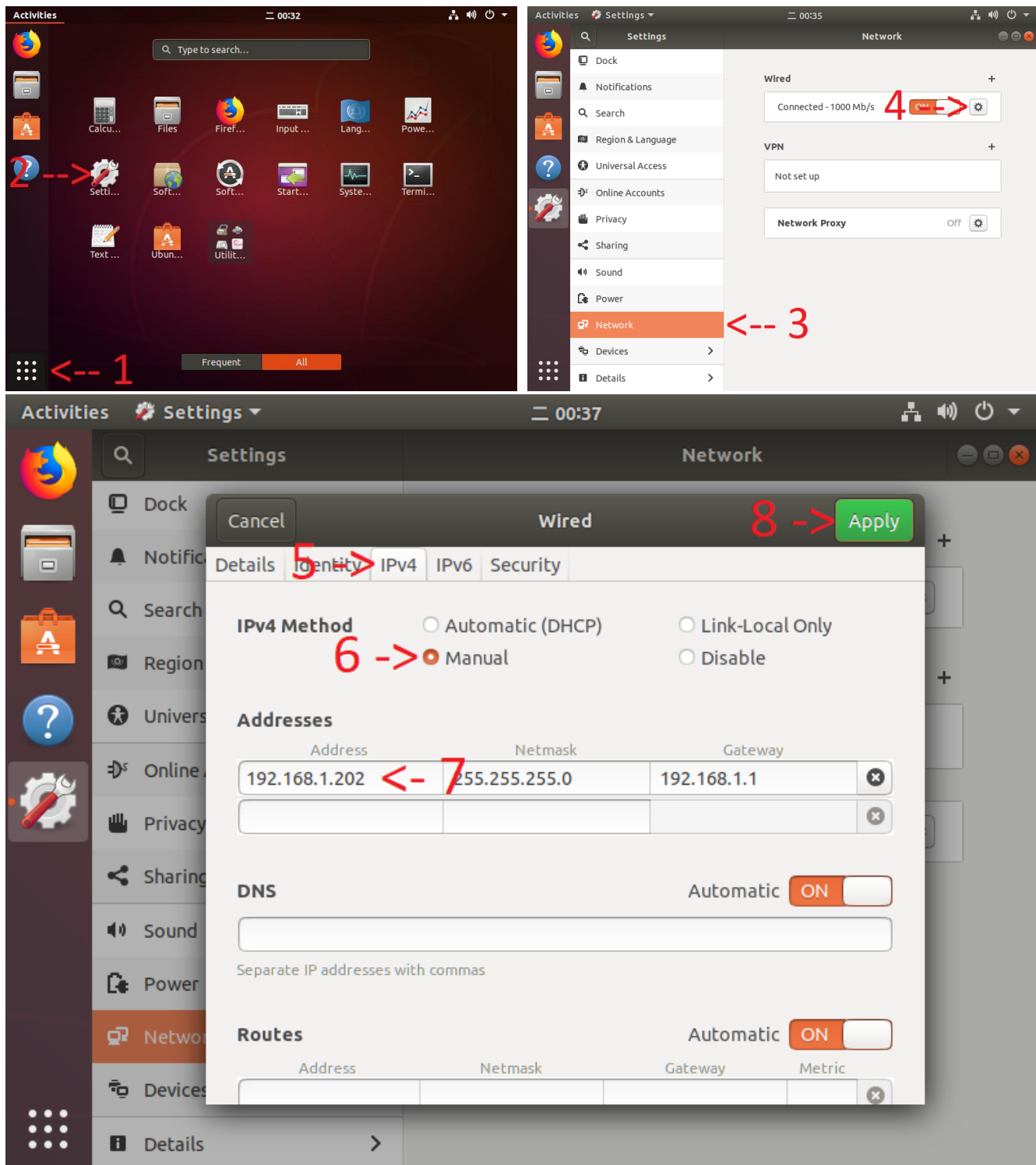
Reading package lists... Done
Building dependency tree
Reading state information... Done
6 packages can be upgraded. Run 'apt list --upgradable' to see them.

```
user@at91:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  evince evince-common fonts-noto-color-emoji libevdocument3-4 libevview3-3 xdg-utils
6 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[...]
```

done

1.1.5 Network configuration

DHCP will be used with default setting, you also could configure static IP in desktop UI with following steps:



1. Click "Show Applications"
2. Select "Settings"
3. Select "Network"
4. Click configure button
5. Select "IPv4" label
6. Select "Manual"
7. Add IP address, netmask and gateway information
8. Click "Apply"

- Check network configuration:

```
user@at91:~$ ifconfig
```

```
enol: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.202 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::b18a:8ff9:d367:9305 prefixlen 64 scopeid 0x20<link>
    ether 30:b4:9e:a9:8e:26 txqueuelen 1000 (Ethernet)
    RX packets 5023 bytes 368478 (368.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 12815 bytes 16049049 (16.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Check network connection:

```
user@at91:~$ ping -c 4 192.168.1.10
```

```
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.886 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.829 ms
64 bytes from 192.168.1.10: icmp_seq=3 ttl=64 time=0.779 ms
64 bytes from 192.168.1.10: icmp_seq=4 ttl=64 time=0.809 ms
```

```
--- 192.168.1.10 ping statistics ---
```

```
4 packets transmitted, 4 received, 0% packet loss, time 3078ms
rtt min/avg/max/mdev = 0.779/0.825/0.886/0.052 ms
```

1.1.6 Enable root account

`sudo` command is a smart choice which can ensure the safety of the system, but sometimes we still need root login

```
user@at91:~$ sudo passwd root
```

```
Enter new UNIX password: 123456
```

```
Retype new UNIX password: 123456
```

```
passwd: password updated successfully
```

1.1.7 Check and disable firewall

Firewall need to be disabled, because lots of network service will be used when developing(tftp, nfs, samba...).

With default setting, firewall should be disabled, but we'd better confirm it.

1. How to check firewall status

```
user@at91:~$ sudo ufw status
```

```
Status: inactive
```

2. How to disable fireware

```
user@at91:~$ sudo ufw disable
```

```
Firewall stopped and disabled on system startup
```

1.2 Install Ubuntu applications

1.2.1 Install commands and tools

With “Minimal installation” option selected, many commands and tools are missed, have to install manually:

```
user@at91:~$ sudo apt -y install vim net-tools git make libncurses5-dev g++ bison flex
u-boot-tools libc6-i386 lib32stdc++6 lib32z1 mtd-utils libgconf2-4
Setting up libgconf2-4:amd64 (3.2.6-4ubuntu1) ...
Setting up make (4.1-9.1ubuntu1) ...
Setting up flex (2.6.4-6) ...
Setting up vim (2:8.0.1453-1ubuntu1) ...
[...]
Processing triggers for install-info (6.5.0.dfsg.1-2) ...
Setting up u-boot-tools (2016.03+dfsg1-6ubuntu2) ...
Setting up libc6-i386 (2.27-3ubuntu1) ...
Setting up mtd-utils (1:2.0.1-1ubuntu3) ...
Setting up libfl-dev:amd64 (2.6.4-6) ...
Setting up libncurses5-dev:amd64 (6.1-1ubuntu1.18.04) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Setting up g++ (4:7.3.0-3ubuntu2.1) ...
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mode
Setting up lib32z1 (1:1.2.11.dfsg-0ubuntu2) ...
Setting up bison (2:3.0.4.dfsg-1build1) ...
update-alternatives: using /usr/bin/bison.yacc to provide /usr/bin/yacc (yacc) in auto mode
Setting up net-tools (1.60+git20161116.90da8a0-1ubuntu1) ...
Setting up git (1:2.17.1-1ubuntu0.4) ...
Setting up lib32gcc1 (1:8.2.0-1ubuntu2~18.04) ...
Setting up lib32stdc++6 (8.2.0-1ubuntu2~18.04) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
```

Please check output log to confirm successful installation, other ways update your Ubuntu first (Section 1.1.4).

1.2.2 Install SSH service

SSH (Secure Shell) usually used for remote network management.

If you wonder that why SSH is so popular in local embedded system development?

1. One reason is we can share one Linux server via SSH remote access for multiple users.
2. The root cause is most of us are more familiar to using Windows than Linux.

(access to Ubuntu terminal via SSH for those things must be done in Linux, all other stuffs will be done in Windows)

- Install SSH server

```
user@at91:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
The following NEW packages will be installed:
  openssh-server
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 332 kB of archives.
After this operation, 898 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 openssh-server amd64
1:7.6p1-4ubuntu0.1 [332 kB]
Fetched 332 kB in 12s (27.0 kB/s)
Preconfiguring packages ...
Selecting previously unselected package openssh-server.
(Reading database ... 162973 files and directories currently installed.)
Preparing to unpack .../openssh-server_1%3a7.6p1-4ubuntu0.1_amd64.deb ...
Unpacking openssh-server (1:7.6p1-4ubuntu0.1) ...
Processing triggers for ufw (0.35-5) ...
Processing triggers for ureadahead (0.100.0-20) ...
ureadahead will be reprofiled on next reboot
Setting up openssh-server (1:7.6p1-4ubuntu0.1) ...
Processing triggers for systemd (237-3ubuntu10.4) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

- Configure SSH server

enable root login:

```
user@at91:~$ sudo vim /etc/ssh/sshd_config
```

modify

```
#PermitRootLogin prohibit-password
```

to

```
PermitRootLogin yes
```

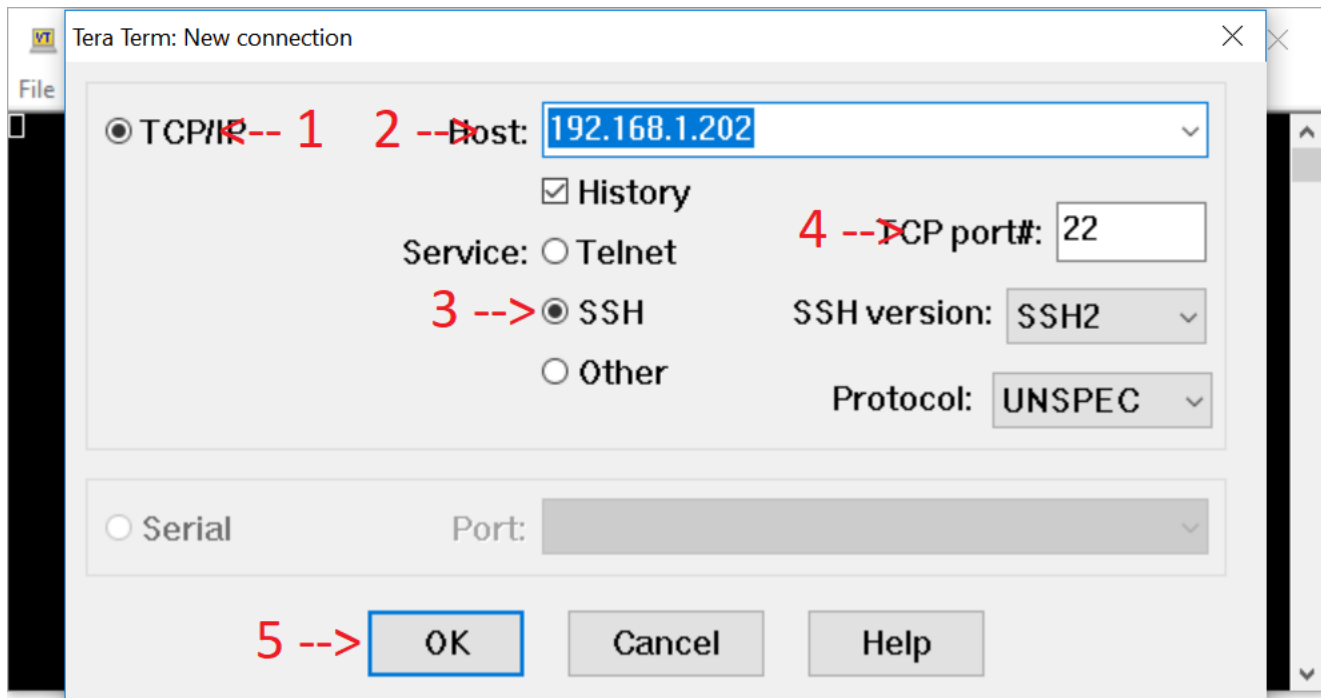
- Restart SSH service

```
user@at91:~$ sudo service ssh restart
```

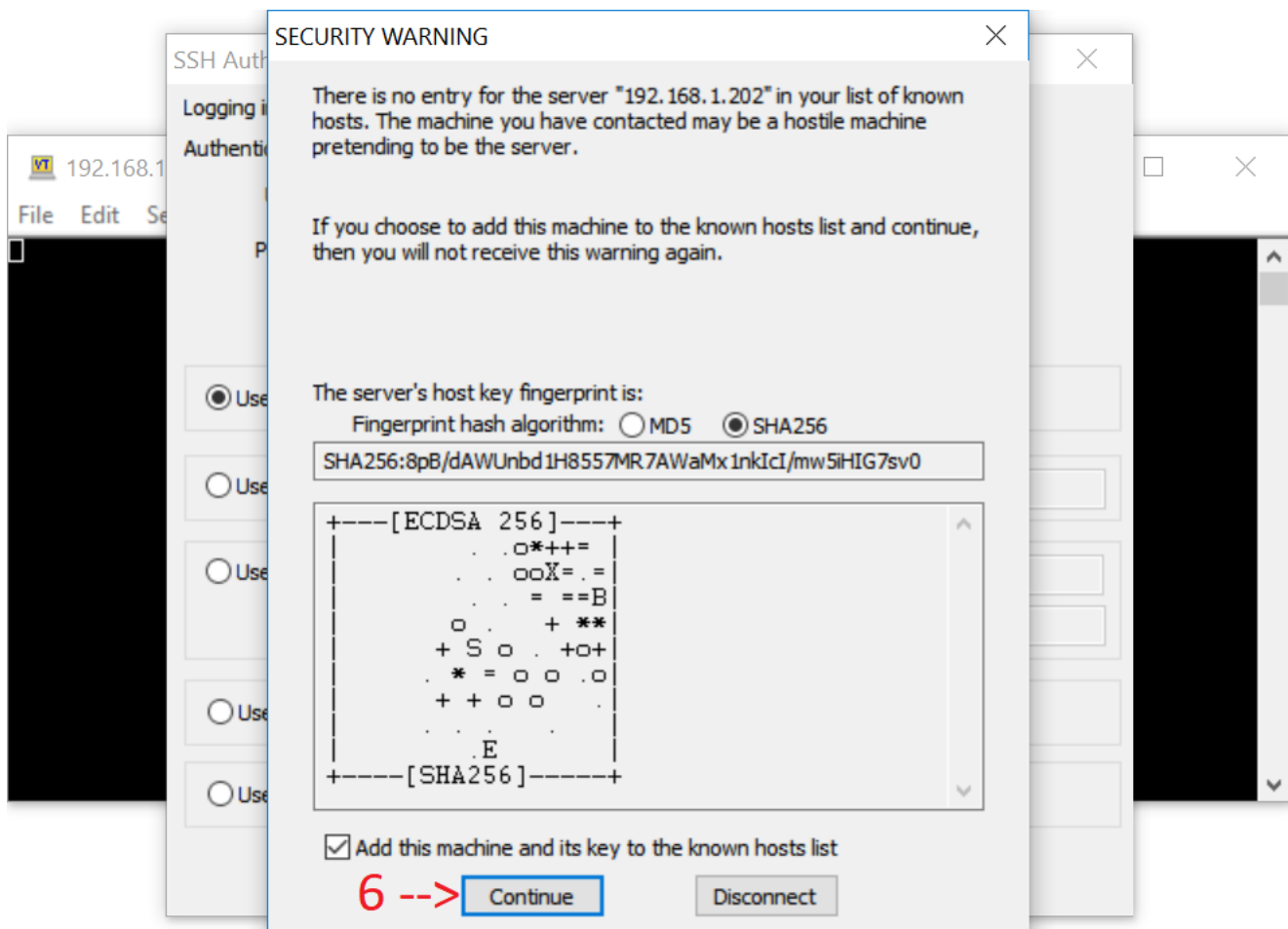
Normally service should work after restart, but if it still doesn't work, please reboot Ubuntu:

```
user@at91:~$ sudo reboot
```

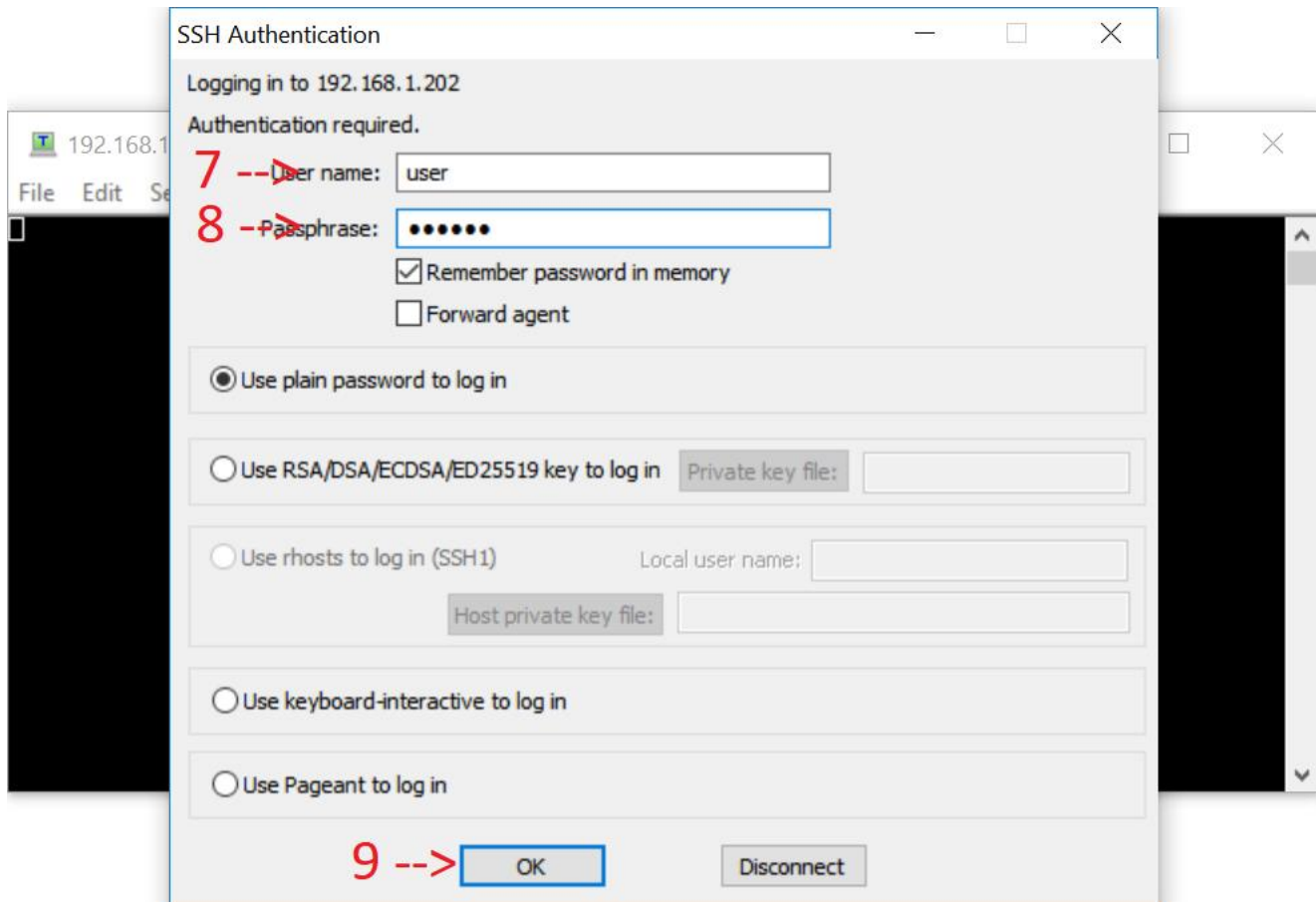
- Connect to Ubuntu shell from Windows via SSH



1. Select "TCP/IP"
2. Input Ubuntu IP address
3. Select "SSH"
4. Use default SSH port 22
5. Click "OK"



6. Click "Continue"

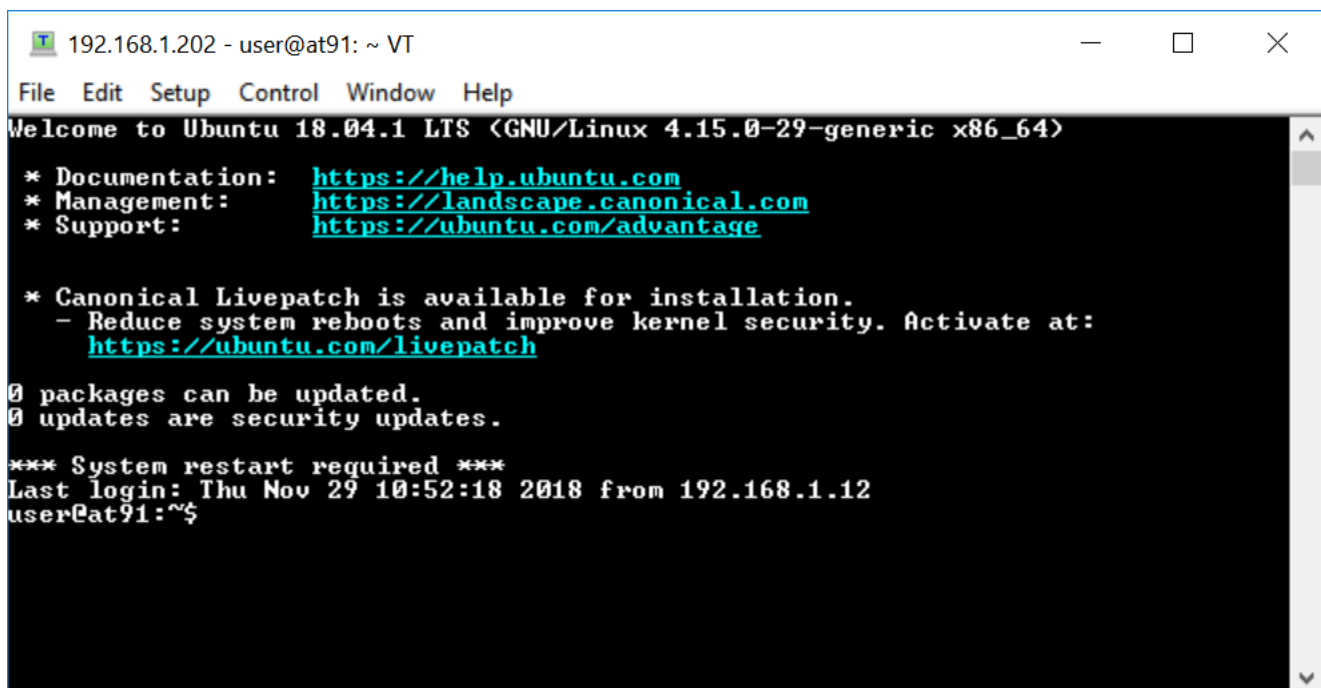


7. Input user name: user

8. Input password: 123456

9. Click "OK"

Then you log in Ubuntu shell:



1.2.3 Install TFTP service

TFTP will be used to transfer files between host PC and target board via network.

- Install TFTP server

```
user@at91:~$ sudo apt install tftp-hpa tftpd-hpa
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  pxelinux
The following NEW packages will be installed:
  tftp-hpa tftpd-hpa
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 57.5 kB of archives.
After this operation, 172 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 tftp-hpa amd64
5.2+20150808-1ubuntu3 [18.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/main amd64 tftpd-hpa amd64
5.2+20150808-1ubuntu3 [39.1 kB]
Fetched 57.5 kB in 2s (29.8 kB/s)
Preconfiguring packages ...
Selecting previously unselected package tftp-hpa.
(Reading database ... 162972 files and directories currently installed.)
Preparing to unpack .../tftp-hpa_5.2+20150808-1ubuntu3_amd64.deb ...
Unpacking tftp-hpa (5.2+20150808-1ubuntu3) ...
Selecting previously unselected package tftpd-hpa.
Preparing to unpack .../tftpd-hpa_5.2+20150808-1ubuntu3_amd64.deb ...
Unpacking tftpd-hpa (5.2+20150808-1ubuntu3) ...
Processing triggers for ureadahead (0.100.0-20) ...
Setting up tftpd-hpa (5.2+20150808-1ubuntu3) ...
tftpd user (tftp) already exists, doing nothing.

tftpd-hpa directory (/srv/tftp) already exists, doing nothing.
Processing triggers for systemd (237-3ubuntu10.4) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Setting up tftp-hpa (5.2+20150808-1ubuntu3) ...
```

- Configure TFTP server

```
user@at91:~$ mkdir -p /home/user/srv/tftp
user@at91:~$ sudo vim /etc/default/tftpd-hpa
# Add the following contents
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/home/user/srv/tftp"
TFTP_ADDRESS=":69"
TFTP_OPTIONS="-l -c -s"
```

- Restart TFTP service

```
user@at91:~$ sudo service tftpd-hpa restart
```

Normally service should work after restart, but if it still doesn't work, please reboot Ubuntu:

```
user@at91:~$ sudo reboot
```

- Verify TFTP service (optional)

Create file on Ubuntu for TFTP testing:

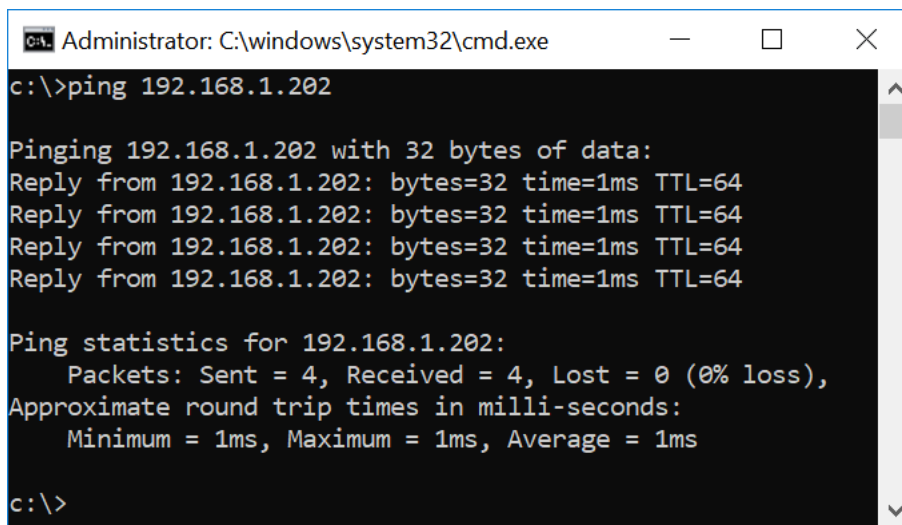
```
user@at91:~$ echo "Hello, TFTP World!" > /home/user/srv/tftp/tftp_test
```

Notice: please enable "TFTP Client" feature on Windows, follow Appendix F "How to turn Windows features on or off"

Download file from Ubuntu on Windows PC

Following command will be executed in Windows cmd:

Check network connection:



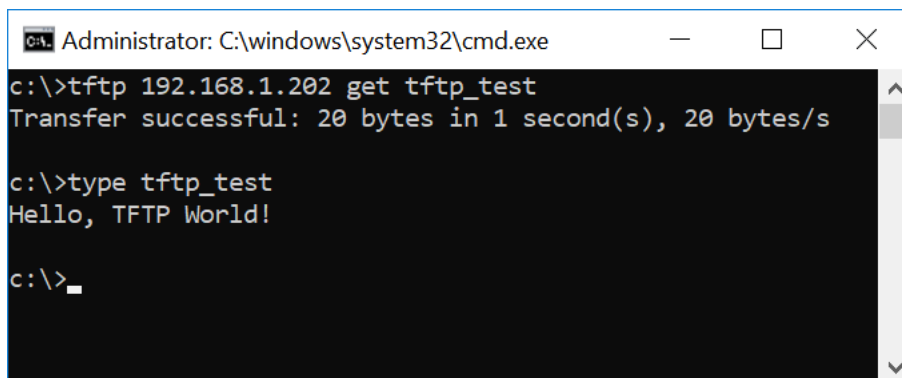
```
Administrator: C:\windows\system32\cmd.exe
c:\>ping 192.168.1.202

Pinging 192.168.1.202 with 32 bytes of data:
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64

Ping statistics for 192.168.1.202:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 1ms, Average = 1ms

c:\>
```

TFTP download:



```
Administrator: C:\windows\system32\cmd.exe
c:\>tftp 192.168.1.202 get tftp_test
Transfer successful: 20 bytes in 1 second(s), 20 bytes/s

c:\>type tftp_test
Hello, TFTP World!

c:\>_
```

With the output of "Hello, TFTP World!", the TFTP service works well.

1.2.4 Install NFS service

NFS are usually used to share files from host PC to target board.

- Install NFS server

```
user@at91:~$ sudo apt install nfs-kernel-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  nfs-kernel-server
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 94.0 kB of archives.
After this operation, 344 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 nfs-kernel-server amd64
1:1.3.4-2.1ubuntu5 [94.0 kB]
Fetched 94.0 kB in 11s (8,614 B/s)
Selecting previously unselected package nfs-kernel-server.
(Reading database ... 162958 files and directories currently installed.)
Preparing to unpack .../nfs-kernel-server_1%3a1.3.4-2.1ubuntu5_amd64.deb ...
Unpacking nfs-kernel-server (1:1.3.4-2.1ubuntu5) ...
Processing triggers for ureadahead (0.100.0-20) ...
Setting up nfs-kernel-server (1:1.3.4-2.1ubuntu5) ...
Processing triggers for systemd (237-3ubuntu10.4) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

- Configure NFS server

```
user@at91:~$ mkdir -p /home/user/srv/nfs
user@at91:~$ sudo vim /etc/exports
# Add the following contents
/home/user/srv/nfs *(rw,sync,no_subtree_check,no_root_squash)
```

- Restart NFS service

```
user@at91:~$ sudo service nfs-kernel-server restart
```

Normally service should work after restart, but if it still doesn't work, please reboot Ubuntu:

```
user@at91:~$ sudo reboot
```

- Verify NFS service (**optional**)

Create file on Ubuntu for NFS testing:

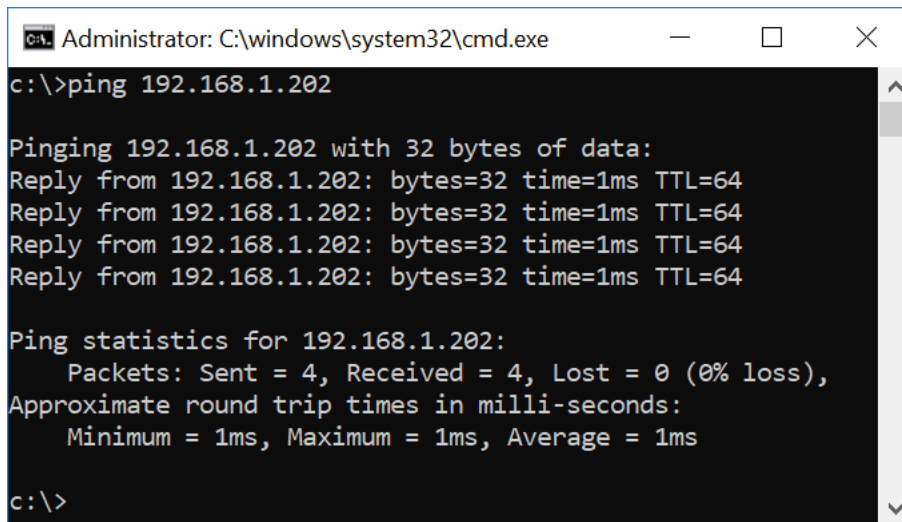
```
user@at91:~$ echo "Hello, NFS World!" > /home/user/srv/nfs/nfs_test
```

Notice: please enable “Services for NFS” feature on Windows, follow Appendix F “How to turn Windows features on or off”

Mount NFS folder from Ubuntu on Windows PC

Following command will be executed in Windows cmd:

Check network connection:



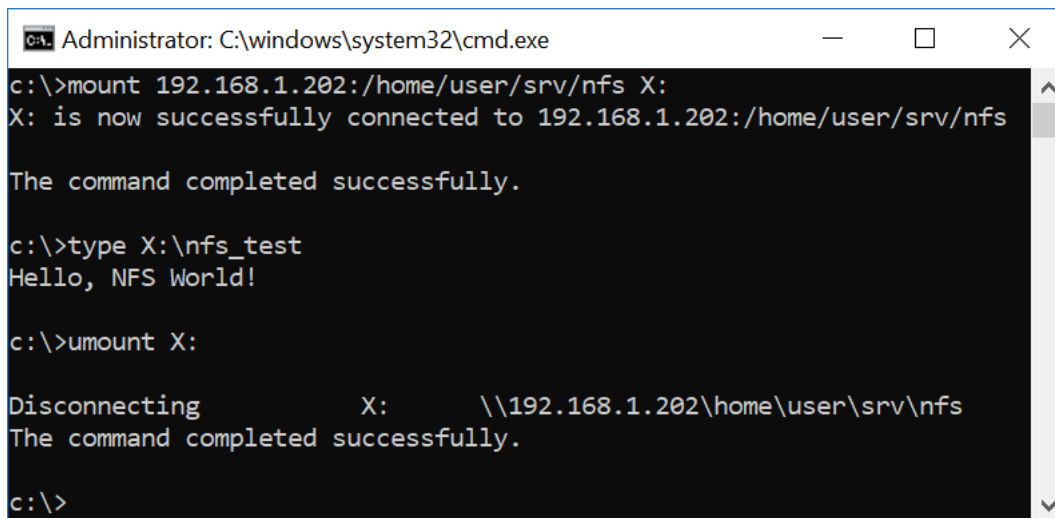
```
Administrator: C:\windows\system32\cmd.exe
c:\>ping 192.168.1.202

Pinging 192.168.1.202 with 32 bytes of data:
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64
Reply from 192.168.1.202: bytes=32 time=1ms TTL=64

Ping statistics for 192.168.1.202:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms

c:\>
```

Mount NFS folder:



```
Administrator: C:\windows\system32\cmd.exe
c:\>mount 192.168.1.202:/home/user/srv/nfs X:
X: is now successfully connected to 192.168.1.202:/home/user/srv/nfs

The command completed successfully.

c:\>type X:\nfs_test
Hello, NFS World!

c:\>umount X:

Disconnecting      X:      \\192.168.1.202\home\user\srv\nfs
The command completed successfully.

c:\>
```

With the output of “Hello, NFS World!”, the NFS service works well.

Use “umount X:” command to disconnect with remote NFS server.

1.2.5 Install SAMBA service

Samba are usually used to share files between Linux server and Windows PC.

- Install SAMBA server

```
user@at91:~$ sudo apt install samba
```

```
Reading package lists... Done
```

```
Building dependency tree
```

```
Reading state information... Done
```

```
Suggested packages:
```

```
bind9 bind9utils ctdb ldb-tools ntp | chrony smbldap-tools winbind
```

```
The following NEW packages will be installed:
```

```
samba
```

```
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
```

```
Need to get 856 kB of archives.
```

```
After this operation, 11.2 MB of additional disk space will be used.
```

```
Get:1 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 samba amd64  
2:4.7.6+dfsg~ubuntu-0ubuntu2.2 [856 kB]
```

```
Fetch 856 kB in 27s (31.2 kB/s)
```

```
Selecting previously unselected package samba.
```

```
(Reading database ... 162795 files and directories currently installed.)
```

```
Preparing to unpack .../samba_2%3a4.7.6+dfsg~ubuntu-0ubuntu2.2_amd64.deb ...
```

```
Unpacking samba (2:4.7.6+dfsg~ubuntu-0ubuntu2.2) ...
```

```
Processing triggers for ufw (0.35-5) ...
```

```
Processing triggers for ureadahead (0.100.0-20) ...
```

```
Processing triggers for libc-bin (2.27-3ubuntu1) ...
```

```
Processing triggers for systemd (237-3ubuntu10.4) ...
```

```
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

```
Setting up samba (2:4.7.6+dfsg~ubuntu-0ubuntu2.2) ...
```

```
Samba is not being run as an AD Domain Controller.
```

```
Please ignore the following error about deb-systemd-helper not finding samba-ad-dc.service.
```

```
Failed to preset unit: Unit file /etc/systemd/system/samba-ad-dc.service is masked.
```

```
/usr/bin/deb-systemd-helper: error: systemctl preset failed on samba-ad-dc.service: No such  
file or directory
```

```
Processing triggers for libc-bin (2.27-3ubuntu1) ...
```

- Configure SAMBA server

```
user@at91:~$ sudo smbpasswd -a user
```

```
New SMB password:123456
```

```
Retype new SMB password:123456
```

```
user@at91:~$ sudo vim /etc/samba/smb.conf
```

```
# Add the following contents at the end of file
```

```
[user]
```

```
comment = user folder
```

```
path = /home/user
```

```
browseable = yes
```

```
read only = no
```

```
guest ok = no
```

- Restart SAMBA service

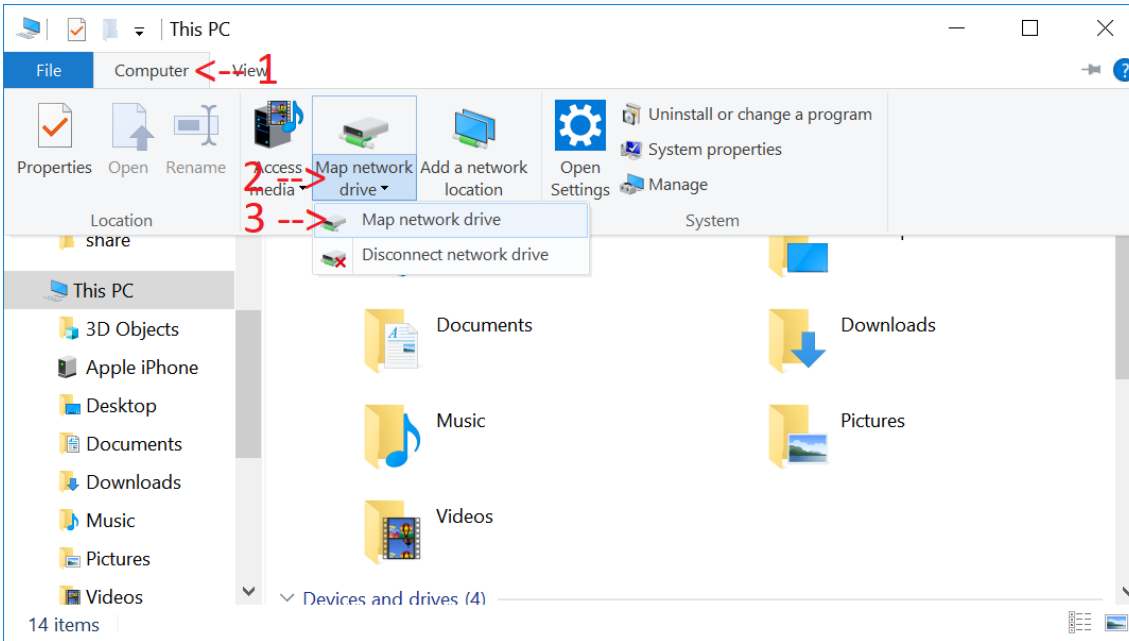
```
user@at91:~$ sudo service smbd restart
```

Normally service should work after restart, but if it still doesn't work, please reboot Ubuntu:

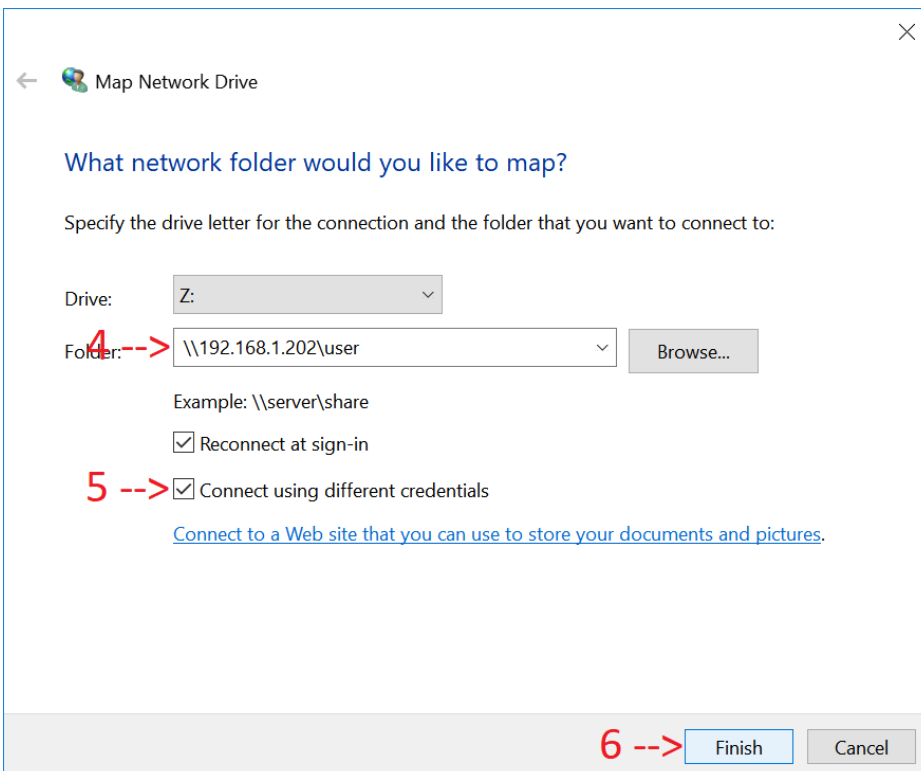
```
user@at91:~$ sudo reboot
```

- Connect to Ubuntu SAMBA share folder

Open "This PC" on Windows:

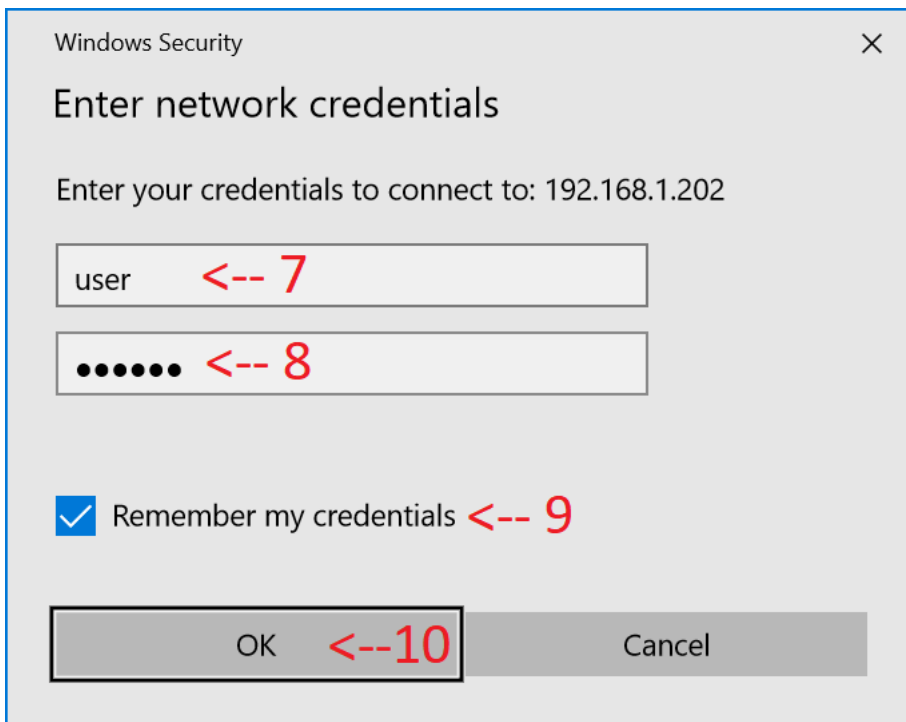


1. Click "Computer"
2. Click "Map network drive"
3. Click "Map network drive"



4. Input "\\192.168.1.202\\user"
5. Select "Connect using different credentials"

6. Click "Finish"



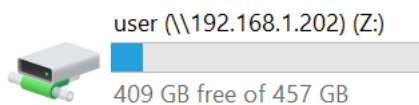
7. Input user name: user

8. Input password: 123456

9. Click "Remember my credentials"

10. Click "OK"

After successful connection, following network drive will be found in "This PC", then remote folder on Ubuntu could be accessed via this network drive.



1.3 Reference information of Ubuntu installation

- Account:

Name:	Password:
user	123456
root	123456

- Network:

Mode:	IP address:
dhcp	use ifconfig check
static	192.168.1.202

- TFTP service

User name	tftp
Directory	/home/user/srv/tftp

- NFS service

Directory	/home/user/srv/nfs
-----------	--------------------

- SAMBA service

User name	user
Password	123456
Directory	/home/user

2. Setup AT91 development environment and build

2.1 Linux4sam



<http://www.at91.com/linux4sam/bin/view/Linux4SAM>

Linux4sam website is the main starting point for Linux OS on SAM products. Its aim is to centralize information about Linux kernel and open source projects on Microchip AT91 Smart ARM-based Microprocessors (aka SAM).

The goal of Linux4sam is to be an interface with open source projects that include AT91 support. We do not want to duplicate information but to link as much as possible to good resources available on the Web.

Keep an eye on this website as it has been designed for instant update. We will try to make it live with the open source community and update AT91 information in those pages. To keep yourself informed, add WebRss or WebAtom feeds in your usual news reader.

Questions, feedback, patches and enhancement are the way open source communities live. Go to LinksToCommunities page for a natural way to interact with material presented on this website

All necessary resources for AT91 embedded Linux development could be found in Linux4sam:

- Software tools
- AT91Bootstrap
- U-Boot / barebox
- Linux Kernel
- DT-Overlay

And some open source solutions for AT91 embedded Linux development are maintained in Linux4sam:

- Yocto Project
- Buildroot
- OpenWrt

In the past, before open source solutions appeared, we always configure and build everything by ourselves, have to configure and build bootstrap, u-boot, kernel and rootfs step by step.

Then we have open source solutions, for example with Buildroot, all necessary resources have been integrated into it, and could be configured and built within two commands:

```
$ make atmel_sama5d27_som1_ek_mmc_dev_defconfig  
$ make
```

In the next chapter, we'll cover Buildroot in detail.

Please check FAQ page of Linux4sam for support:

<http://www.at91.com/linux4sam/bin/view/Linux4SAM/WebFaq>

2.2 Buildroot



<https://buildroot.org/>

Buildroot is a tool that simplifies and automates the process of building a complete Linux system for an embedded system, using cross-compilation.

In order to achieve this, Buildroot is able to generate a cross-compilation toolchain, a root filesystem, a Linux kernel image and a bootloader for your target. Buildroot can be used for any combination of these options, independently (you can for example use an existing cross-compilation toolchain, and build only your root filesystem with Buildroot).

Buildroot is useful mainly for people working with embedded systems. Embedded systems often use processors that are not the regular x86 processors everyone is used to having in his PC. They can be PowerPC processors, MIPS processors, ARM processors, etc.

Buildroot supports numerous processors and their variants; it also comes with default configurations for several boards available off-the-shelf. Besides this, a number of third-party projects are based on, or develop their BSP or SDK on top of Buildroot.

As we talked before in last chapter, In Buildroot all necessary resources could be configured and built within two commands, but Buildroot is not as smart as you think.

In Buildroot all resource was treated as a package, for processing a package, you must tell Buildroot following information:

- How to download the source of this package
- How to configure this package
- How to build this package
- How to install this package

After all needed information was given, then Buildroot could process this package step by step.

A guide could be found for AT91 Buildroot setup in Linux4sam:

<http://www.at91.com/linux4sam/bin/view/Linux4SAM/BuildRoot>

Follow this guide AT91 Buildroot development environment will be setup, but you will find that a lot of time will be spent in the step of package source downloading, maybe you have to wait a few hours, because the source of package will be downloaded from all over the world.

For speedup the step of package source downloading, we made a pre-download package for AT91 Buildroot, with this pre-download package the step of package source downloading could be ignored directly.

Please check the Buildroot support page for support:

<https://buildroot.org/support.html>

2.3 Quick setup using pre-downloaded Buildroot package

- Copy following two files to your Ubuntu home folder (/home/user) via U-disk or SAMBA

```
user@at91:~$ ls
buildroot-linux4sam_6.0.tgz  dl-linux4sam_6.0.tar
```

- Decompress pre-downloaded Buildroot project, two new folders will be got

```
user@at91:~$ tar -xzf buildroot-linux4sam_6.0.tgz
user@at91:~$ ls
buildroot-at91  buildroot-external-microchip  buildroot-linux4sam_6.0.tgz
dl-linux4sam_6.0.tar
```

- Check the tag name of latest release (now it is linux4sam_6.0)

```
user@at91:~$ git -C buildroot-at91/ tag
linux4sam_5.8
linux4sam_5.8-rc1
linux4sam_6.0
linux4sam_6.0-rc1
linux4sam_6.0-rc2
linux4sam_6.0-rc3
```

- Checkout the latest release tag for Buildroot main folder and external folder

```
user@at91:~$ git -C buildroot-at91/ checkout linux4sam_6.0
Note: checking out 'linux4sam_6.0'.
```

```
[...]
```

```
HEAD is now at 5e2felac44 board: atmel: genimage: remove display variant dtbs
```

```
user@at91:~$ git -C buildroot-external-microchip/ checkout linux4sam_6.0
```

```
Note: checking out 'linux4sam_6.0'.
```

```
[...]
```

```
HEAD is now at dc3e575 configs: remove smartrefrigerator from sama5d3_xplained demo
```

- Enter Buildroot main folder

```
user@at91:~$ cd buildroot-at91/
```

- Decompress pre-downloaded Buildroot packages

```
user@at91:~/buildroot-at91$ tar -xf ../dl-linux4sam_6.0.tar
user@at91:~/buildroot-at91$ ls -l dl
[...]
```

- Configure Buildroot project

```
user@at91:~/buildroot-at91$ BR2_EXTERNAL=../buildroot-external-microchip/ make
atmel_sama5d27_som1_ek_mmc_dev_defconfig
```

- Build Buildroot project (this takes 45 minutes with Intel Core I7-3770)

```
user@at91:~/buildroot-at91$ make
```

- After successful of building, check the output files

```
user@at91:~/buildroot-at91$ ls -l output/images/
```

```
total 194132
```

```
-rwxr-xr-x 1 user user    14169 Nov 27 18:24 at91bootstrap.bin
-rw-r--r-- 1 user user   44915 Nov 27 18:27 at91-sama5d27_som1_ek.dtb
-rwxr-xr-x 1 user user    14169 Nov 27 18:24 boot.bin
-rw-r--r-- 1 user user 16777216 Nov 27 18:29 boot.vfat
-rw-r--r-- 1 user user 62914560 Nov 27 18:29 rootfs.ext2
lrwxrwxrwx 1 user user      11 Nov 27 18:29 rootfs.ext4 -> rootfs.ext2
-rw-r--r-- 1 user user 54722560 Nov 27 18:29 rootfs.tar
-rwxr-xr-x 1 user user    14169 Nov 27 18:24 sama5d27_som1_ek-sdcardboot-uboot-3.8.11.bin
-rw-r--r-- 1 user user 80740352 Nov 27 18:29 sdcard.img
-rw-r--r-- 1 user user   442991 Nov 27 18:24 u-boot.bin
-rw-r--r-- 1 user user 3929096 Nov 27 18:27 zImage
```

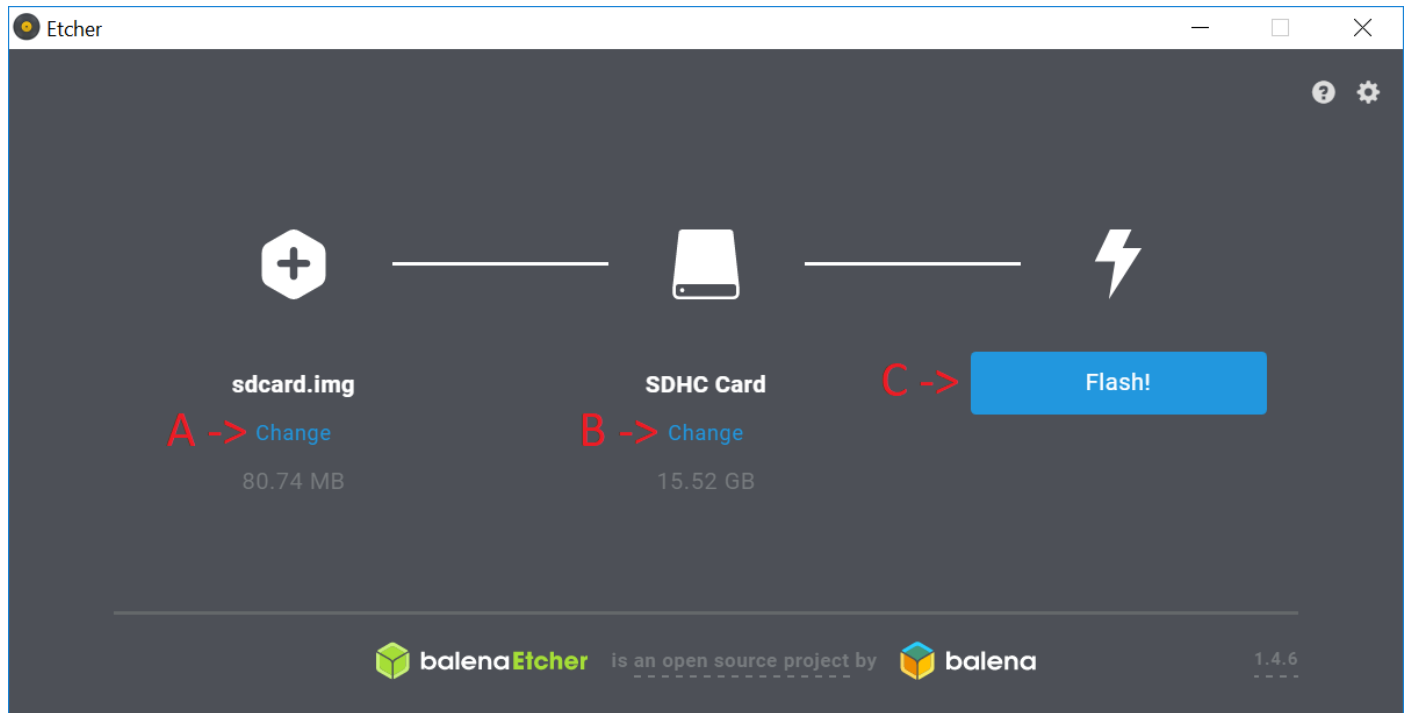
- Use “make help” to get more features about Buildroot

```
user@at91:~/buildroot-at91$ make help
```

3. Burn SD Card and run it

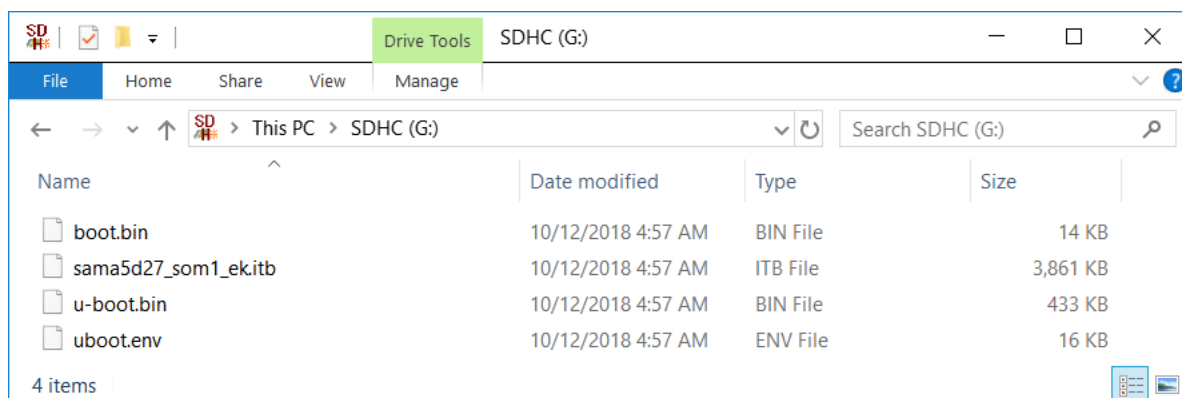
- Burn SD Card in Windows

Insert an empty SD card into your Windows PC and burn it with Etcher.

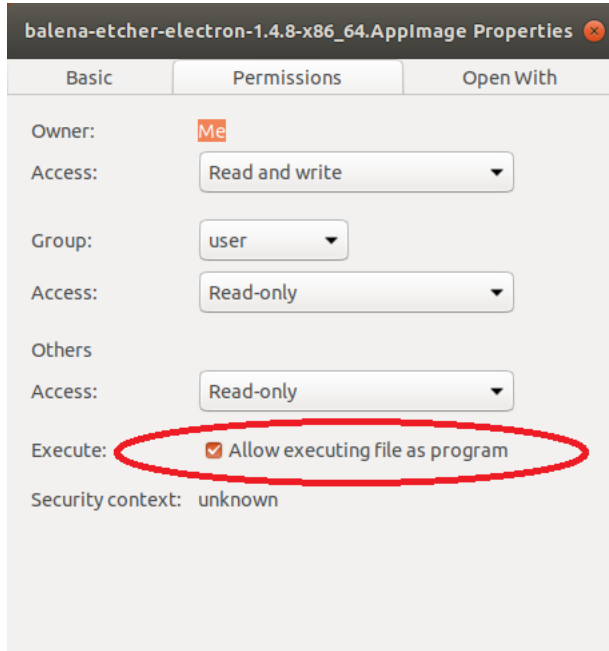


- Click "Change" to select you SD card image files
(Location: buildroot-at91/output/images/sdcard.img)
- Click "Change" to select you SD card device
- Click "Flash!" to burn it.

After "Flash Complete!" following files could be found if you SD card:



- Or burn SD Card in Ubuntu
 1. Copy “balena-etcher-electron-1.4.8-linux-x64.zip” to your Ubuntu
 2. Right click it and click “Extract here”
 3. Enter directory “balena-etcher-electron-1.4.8-linux-x64”
 4. Make sure that the execute permission of “balena-etcher-electron-1.4.8-x86_64.AppImage” has been selected



5. Then double click “balena-etcher-electron-1.4.8-x86_64.AppImage” and burn SD Card follow the steps in Windows
- After successful burning insert this SD card into your target board (J12 SD), then power on target board.

```
COM3 - Tera Term VT
File Edit Setup Control Window Help
RomBOOT
AT91Bootstrap 3.8.11 <Thu Nov 22 16:48:00 AKST 2018>
SD/MMC: Image: Read file u-boot.bin to 0x23f00000
MMC: ADMA supported
SD: Card Capacity: High or Extended
SD: Specification Version 3.0X
SD/MMC: Done to load image
<debug_uart>
U-Boot 2018.07-linux4sam_6.0 <Nov 22 2018 - 16:48:05 -0900>
CPU: SAMA5D27 1G bits DDR2 SDRAM
Crystal frequency: 24 MHz
CPU clock : 492 MHz
Master clock : 164 MHz
DRAM: 128 MiB
MMC: sdio-host@00000000: 0, sdio-host@b0000000: 1
Loading Environment from FAT... *** Warning - bad CRC, using default environment
Failed <-5>
In: serial@f8020000
Out: serial@f8020000
Err: serial@f8020000
Net: eth0: ethernet@f8008000
Hit any key to stop autoboot: 0
```

4. Configuration of target boards in Buildroot

Please check defconfig files in following two folders:

buildroot-at91/configs

buildroot-external-microchip/configs

```
user@at91:~$ ls buildroot-at91/configs/atmel_*
buildroot-at91/configs/atmel_sama5d27_som1_ek_mmc_dev_defconfig
buildroot-at91/configs/atmel_sama5d3_xplained_mmc_defconfig
buildroot-at91/configs/atmel_sama5d2_xplained_mmc_defconfig
buildroot-at91/configs/atmel_sama5d3_xplained_mmc_dev_defconfig
buildroot-at91/configs/atmel_sama5d2_xplained_mmc_dev_defconfig
buildroot-at91/configs/atmel_sama5d4_xplained_defconfig
buildroot-at91/configs/atmel_sama5d3xek_defconfig
buildroot-at91/configs/atmel_sama5d4_xplained_dev_defconfig
buildroot-at91/configs/atmel_sama5d3_xplained_defconfig
buildroot-at91/configs/atmel_sama5d4_xplained_mmc_defconfig
buildroot-at91/configs/atmel_sama5d3_xplained_dev_defconfig
buildroot-at91/configs/atmel_sama5d4_xplained_mmc_dev_defconfig
user@at91:~$ ls buildroot-external-microchip/configs/
at91sam9x5ek_headless_defconfig      sama5d2_ptc_ek_headless_defconfig
sama5d2_xplained_nodered_defconfig   sama5d4_xplained_demo_defconfig
sama5d27_som1_ek_demo_defconfig      sama5d2_ptc_ek_nodered_defconfig
sama5d3_xplained_demo_defconfig      sama5d4_xplained_headless_defconfig
sama5d27_som1_ek_headless_defconfig  sama5d2_xplained_demo_defconfig
sama5d3_xplained_headless_defconfig  sama5d4_xplained_nodered_defconfig
sama5d27_som1_ek_nodered_defconfig   sama5d2_xplained_headless_defconfig
sama5d3_xplained_nodered_defconfig
```


5. Cross toolchain in Buildroot

After building successfully, check following folder:

buildroot-at91/output/host/usr/bin

```
user@at91:~$ ls buildroot-at91/output/host/usr/bin/
[.]
arm-buildroot-linux-uclibcgnueabihf-addr2line
arm-buildroot-linux-uclibcgnueabihf-gcc-6.4.0.br_real
arm-buildroot-linux-uclibcgnueabihf-ldconfig
arm-buildroot-linux-uclibcgnueabihf-ar
arm-buildroot-linux-uclibcgnueabihf-gcc-ar
arm-buildroot-linux-uclibcgnueabihf-ldd
arm-buildroot-linux-uclibcgnueabihf-as
arm-buildroot-linux-uclibcgnueabihf-gcc.br_real
arm-buildroot-linux-uclibcgnueabihf-nm
arm-buildroot-linux-uclibcgnueabihf-cc
arm-buildroot-linux-uclibcgnueabihf-gcc-nm
arm-buildroot-linux-uclibcgnueabihf-objcopy
arm-buildroot-linux-uclibcgnueabihf-cc.br_real
arm-buildroot-linux-uclibcgnueabihf-gcc-ranlib
arm-buildroot-linux-uclibcgnueabihf-objdump
arm-buildroot-linux-uclibcgnueabihf-c++filt
arm-buildroot-linux-uclibcgnueabihf-gcov
arm-buildroot-linux-uclibcgnueabihf-ranlib
arm-buildroot-linux-uclibcgnueabihf-cpp
arm-buildroot-linux-uclibcgnueabihf-gcov-dump
arm-buildroot-linux-uclibcgnueabihf-readelf
arm-buildroot-linux-uclibcgnueabihf-cpp.br_real
arm-buildroot-linux-uclibcgnueabihf-gcov-tool
arm-buildroot-linux-uclibcgnueabihf-size
arm-buildroot-linux-uclibcgnueabihf-elfedit
arm-buildroot-linux-uclibcgnueabihf-gprof
arm-buildroot-linux-uclibcgnueabihf-strings
arm-buildroot-linux-uclibcgnueabihf-gcc
arm-buildroot-linux-uclibcgnueabihf-ld
arm-buildroot-linux-uclibcgnueabihf-strip
arm-buildroot-linux-uclibcgnueabihf-gcc-6.4.0
arm-buildroot-linux-uclibcgnueabihf-ld.bfd
[.]
```

6. How to develop at91bootstrap in Buildroot

- Configuration of at91bootstrap package in Buildroot

Please check following defconfig file:

```
buildroot-at91/configs/atmel_sama5d27_som1_ek_mmc_dev_defconfig
```

```
BR2_TARGET_AT91BOOTSTRAP3=y
```

```
BR2_TARGET_AT91BOOTSTRAP3_CUSTOM_GIT=y
```

```
BR2_TARGET_AT91BOOTSTRAP3_CUSTOM_REPO_URL="https://github.com/linux4sam/at91bootstrap.git"
```

```
BR2_TARGET_AT91BOOTSTRAP3_CUSTOM_REPO_VERSION="v3.8.11"
```

```
BR2_TARGET_AT91BOOTSTRAP3_DEFCONFIG="sama5d27_som1_eksd_uboot"
```

- Source code location(*WARNING):

```
buildroot-at91/output/build/at91bootstrap3-v3.8.11
```

- "menuconfig" was supported by at91bootstrap3

```
user@at91:~/buildroot-at91$ make at91bootstrap3-menuconfig
```

- Use following command to rebuild at91bootstrap codes:

```
user@at91:~/buildroot-at91$ make at91bootstrap3-rebuild
```

- Use following command to update buildroot-at91/output/images/sdcard.img

```
user@at91:~/buildroot-at91$ make
```

- Use following command to get more support

```
user@at91:~/buildroot-at91$ make help
```

***WARRING**

buildroot-at91/output/build is treated as a temporary folder, following commands will delete package's project files in this folder:

```
make <pkg>-dirclean
```

```
or
```

```
make clean
```

```
or
```

```
make distclean
```

If your changes are important, DO NOT modify the files in this folder directly, then use local.mk to override your package's source path:

```
user@at91:~/buildroot-at91$ make at91bootstrap3-dirclean
```

```
user@at91:~/buildroot-at91$ vim local.mk
```

```
# Add content
```

```
AT91BOOTSTRAP3_OVERRIDE_SRCDIR = ../at91bootstrap3-v3.8.11
```

```
user@at91:~/buildroot-at91$ tar -xzf ../dl/at91bootstrap3-v3.8.11.tar.gz -C ../
```

```
user@at91:~/buildroot-at91$ ls ../
```

```
at91bootstrap3-v3.8.11 buildroot-at91 buildroot-external-microchip
```

```
user@at91:~/buildroot-at91$ make at91bootstrap3
```

Please modify package's project file in the following path:

```
/home/user/at91bootstrap3-v3.8.11
```

7. How to develop u-boot in Buildroot

- Configuration of u-boot package in Buildroot

Please check following defconfig file:

```
buildroot-at91/configs/atmel_sama5d27_som1_ek_mmc_dev_defconfig
```

```
BR2_TARGET_UBOOT=y
```

```
BR2_TARGET_UBOOT_BUILD_SYSTEM_KCONFIG=y
```

```
BR2_TARGET_UBOOT_CUSTOM_GIT=y
```

```
BR2_TARGET_UBOOT_CUSTOM_REPO_URL="https://github.com/linux4sam/u-boot-at91.git"
```

```
BR2_TARGET_UBOOT_CUSTOM_REPO_VERSION="linux4sam_6.0"
```

```
BR2_TARGET_UBOOT_BOARD_DEFCONFIG="sama5d27_som1_ek_mmc"
```

```
BR2_TARGET_UBOOT_NEEDS_DTC=y
```

- Source code location(***WARNING**):

```
buildroot-at91/output/build/u-boot-linux4sam_6.0
```

- Use following command to configure u-boot

```
user@at91:~/buildroot-at91$ make uboot-menuconfig
```

- Use following command to rebuild u-boot

```
user@at91:~/buildroot-at91$ make uboot-rebuild
```

- Use following command to update buildroot-at91/output/images/sdcard.img

```
user@at91:~/buildroot-at91$ make
```

- Use following command to get more support

```
user@at91:~/buildroot-at91$ make help
```

8. How to develop kernel in Buildroot

- Configuration of kernel package in Buildroot

Please check following defconfig file:

```
buildroot-at91/configs/atmel_sama5d27_som1_ek_mmc_dev_defconfig
```

```
BR2_LINUX_KERNEL=y
```

```
BR2_LINUX_KERNEL_CUSTOM_GIT=y
```

```
BR2_LINUX_KERNEL_CUSTOM_REPO_URL="https://github.com/linux4sam/linux-at91.git"
```

```
BR2_LINUX_KERNEL_CUSTOM_REPO_VERSION="linux4sam_6.0"
```

```
BR2_LINUX_KERNEL_DEFCONFIG="sama5"
```

```
BR2_LINUX_KERNEL_DTS_SUPPORT=y
```

```
BR2_LINUX_KERNEL_INTREE_DTS_NAME="at91-sama5d27_som1_ek"
```

- Source code location(*WARRING):

```
buildroot-at91/output/build/linux-linux4sam_6.0
```

- Use following command to configure Linux

```
user@at91:~/buildroot-at91$ make linux-menuconfig
```

- Use following command to rebuild kernel

```
user@at91:~/buildroot-at91$ make linux-rebuild
```

- Use following command to update buildroot-at91/output/images/sdcard.img

```
user@at91:~/buildroot-at91$ make
```

- Use following command to get more support

```
user@at91:~/buildroot-at91$ make help
```

9. How to update Buildroot project files

In the following example we will update Buildroot project files from linux4sam_5.8 to linux4sam_6.0

- Enter Buildroot main director

```
user@at91:~$ cd buildroot-at91/
```

- Check current release tag

```
user@at91:~/buildroot-at91$ git tag
```

```
linux4sam_5.8
```

```
linux4sam_5.8-rc1
```

```
linux4sam_6.0-rc1
```

- Add and check remote information

(origin of git could be changed, we add linux4sam remote and force pull from it)

```
user@at91:~/buildroot-at91$ git remote add linux4sam
```

```
https://github.com/linux4sam/buildroot-at91.git
```

```
user@at91:~/buildroot-at91$ git remote -v
```

```
linux4sam      https://github.com/linux4sam/buildroot-at91.git (fetch)
```

```
linux4sam      https://github.com/linux4sam/buildroot-at91.git (push)
```

```
origin https://github.com/linux4sam/buildroot-at91.git (fetch)
```

```
origin https://github.com/linux4sam/buildroot-at91.git (push)
```

- Pull from remote and merge

(**WARNING** "git pull" will fetch from remote and auto marge into local workspace)

```
user@at91:~/buildroot-at91$ git pull linux4sam master:master
```

```
remote: Enumerating objects: 58, done.
```

```
remote: Counting objects: 100% (58/58), done.
```

```
remote: Compressing objects: 100% (5/5), done.
```

```
remote: Total 47 (delta 38), reused 47 (delta 38), pack-reused 0
```

```
Unpacking objects: 100% (47/47), done.
```

```
From https://github.com/linux4sam/buildroot-at91
```

```
* [new branch]      master      -> master
```

```
* [new tag]         linux4sam_6.0 -> linux4sam_6.0
```

```
* [new tag]         linux4sam_6.0-rc3 -> linux4sam_6.0-rc3
```

```
* [new branch]      master      -> linux4sam/master
```

```
* [new tag]         linux4sam_6.0-rc2 -> linux4sam_6.0-rc2
```

```
Updating 7f5e52e..5e2fela
```

```
[...]
```

```
19 files changed, 71 insertions(+), 84 deletions(-)
```

- Check updated release tag

```
user@at91:~/buildroot-at91$ git tag
```

```
linux4sam_5.8
```

```
linux4sam_5.8-rc1
```

```
linux4sam_6.0
```

```
linux4sam_6.0-rc1
linux4sam_6.0-rc2
linux4sam_6.0-rc3
```

- update Buildroot external with the same steps

```
user@at91:~/buildroot-at91$ cd ../buildroot-external-microchip/
user@at91:~/buildroot-external-microchip$ git tag
linux4sam_5.8
linux4sam_5.8-rc2
user@at91:~/buildroot-external-microchip$ git remote add linux4sam
https://github.com/linux4sam/buildroot-external-microchip.git
user@at91:~/buildroot-external-microchip$ git remote -v
linux4sam      https://github.com/linux4sam/buildroot-external-microchip.git (fetch)
linux4sam      https://github.com/linux4sam/buildroot-external-microchip.git (push)
origin https://github.com/linux4sam/buildroot-external-microchip.git (fetch)
origin https://github.com/linux4sam/buildroot-external-microchip.git (push)
user@at91:~/buildroot-external-microchip$ git pull linux4sam master:master
remote: Enumerating objects: 559, done.
remote: Counting objects: 100% (559/559), done.
remote: Compressing objects: 100% (142/142), done.
remote: Total 514 (delta 373), reused 488 (delta 349), pack-reused 0
Receiving objects: 100% (514/514), 56.37 KiB | 227.00 KiB/s, done.
Resolving deltas: 100% (373/373), completed with 29 local objects.
From https://github.com/linux4sam/buildroot-external-microchip
* [new branch]      master      -> master
* [new tag]         linux4sam_6.0 -> linux4sam_6.0
* [new branch]      master      -> linux4sam/master
* [new tag]         linux4sam_6.0-rc1 -> linux4sam_6.0-rc1
* [new tag]         linux4sam_6.0-rc2 -> linux4sam_6.0-rc2
* [new tag]         linux4sam_6.0-rc3 -> linux4sam_6.0-rc3
* [new tag]         linux4sam_6.0-rc4 -> linux4sam_6.0-rc4
* [new tag]         linux4sam_6.0-rc5 -> linux4sam_6.0-rc5
Updating a972c4c..dc3e575
[...]
 63 files changed, 1441 insertions(+), 259 deletions(-)
[...]
   create mode 100644
patches/qt5base/5.10.1/0005-Support-DRM-KMS-planes-in-linuxfb-DRM-backend.patch
user@at91:~/buildroot-external-microchip$ git tag
linux4sam_5.8
linux4sam_5.8-rc2
linux4sam_6.0
linux4sam_6.0-rc1
linux4sam_6.0-rc2
linux4sam_6.0-rc3
linux4sam_6.0-rc4
linux4sam_6.0-rc5
```

10. AT91 targets in pre-downloaded Buildroot package

AT91 targets in following two folders are supported by pre-downloaded Buildroot package:

- **buildroot-at91/configs**

```
atmel_sama5d27_som1_ek_mmc_dev_defconfig
atmel_sama5d2_xplained_mmc_defconfig
atmel_sama5d2_xplained_mmc_dev_defconfig
//atmel_sama5d3xek_defconfig // don't support, target with too old configuration
atmel_sama5d3_xplained_defconfig
atmel_sama5d3_xplained_dev_defconfig
atmel_sama5d3_xplained_mmc_defconfig
atmel_sama5d3_xplained_mmc_dev_defconfig
atmel_sama5d4_xplained_defconfig
atmel_sama5d4_xplained_dev_defconfig
atmel_sama5d4_xplained_mmc_defconfig
atmel_sama5d4_xplained_mmc_dev_defconfig
```

- **buildroot-external-microchip/configs**

```
at91sam9x5ek_headless_defconfig
sama5d27_som1_ek_demo_defconfig
sama5d27_som1_ek_headless_defconfig
sama5d27_som1_ek_nodered_defconfig
sama5d2_ptc_ek_headless_defconfig
sama5d2_ptc_ek_nodered_defconfig
sama5d2_xplained_demo_defconfig
sama5d2_xplained_headless_defconfig
sama5d2_xplained_nodered_defconfig
sama5d3_xplained_demo_defconfig
sama5d3_xplained_headless_defconfig
sama5d3_xplained_nodered_defconfig
sama5d4_xplained_demo_defconfig
sama5d4_xplained_headless_defconfig
sama5d4_xplained_nodered_defconfig
```


11. How to make pre-downloaded Buildroot package

- Clone the latest linux4sam release from github

```
user@at91:~$ git clone https://github.com/linux4sam/buildroot-at91.git
user@at91:~$ git clone https://github.com/linux4sam/buildroot-external-microchip.git
user@at91:~$ ls -l
total 8
drwxrwxr-x 15 user user 4096 Dec  4 17:52 buildroot-at91
drwxrwxr-x  9 user user 4096 Dec  4 17:52 buildroot-external-microchip
```

- Find the tag name of latest release (now it is linux4sam_6.0)

```
user@at91:~$ git -C buildroot-at91/ tag
linux4sam_5.8
linux4sam_5.8-rc1
linux4sam_6.0
linux4sam_6.0-rc1
linux4sam_6.0-rc2
linux4sam_6.0-rc3
```

- Pack buildroot project files, name it with the tag name of latest release

```
user@at91:~$ tar -czf buildroot-linux4sam_6.0.tgz ./*
user@at91:~$ ls -l
total 72460
drwxrwxr-x 15 user user      4096 Dec  4 17:52 buildroot-at91
drwxrwxr-x  9 user user      4096 Dec  4 17:52 buildroot-external-microchip
-rw-rw-r--  1 user user 74190326 Dec  4 17:55 buildroot-linux4sam_6.0.tgz
```

- Checkout the latest release tag for Buildroot main folder and external folder

```
user@at91:~$ git -C buildroot-at91/ checkout linux4sam_6.0
```

Note: checking out 'linux4sam_6.0'.

[...]

HEAD is now at 5e2felac44 board: atmel: genimage: remove display variant dtbs

```
user@at91:~$ git -C buildroot-external-microchip/ checkout linux4sam_6.0
```

Note: checking out 'linux4sam_6.0'.

[...]

HEAD is now at dc3e575 configs: remove smartrefrigerator from sama5d3_xplained demo

- Enter Buildroot main folder

```
user@at91:~$ cd buildroot-at91/
```

- Download dependent packages for every target

(loop execute following commands with every target's configuration)

```
user@at91:~/buildroot-at91$ BR2_EXTERNAL=../buildroot-external-microchip/ make
xxxx_defconfig
user@at91:~/buildroot-at91$ make source
```

- Pack dl folder after all packages has been downloaded successfully

```
user@at91:~/buildroot-at91$ tar -cf ../dl-linux4sam_6.0.tar ./dl
user@at91:~/buildroot-at91$ cd ..
user@at91:~$ ls -l
total 1103792
drwxrwxr-x 17 user user      4096 Dec  4 18:23 buildroot-at91
drwxrwxr-x  9 user user      4096 Dec  4 17:52 buildroot-external-microchip
-rw-rw-r--  1 user user    74190326 Dec  4 17:55 buildroot-linux4sam_6.0.tgz
-rw-rw-r--  1 user user 1056081920 Dec  4 18:23 dl-linux4sam_6.0.tar
```

- Generate md5sum file for output files

```
user@at91:~$ md5sum buildroot-linux4sam_6.0.tgz > buildroot-linux4sam_6.0.tgz.md5
user@at91:~$ cat buildroot-linux4sam_6.0.tgz.md5
8e90bd12a24c8117e9d81579c5c2ce0e  buildroot-linux4sam_6.0.tgz
user@at91:~$ md5sum dl-linux4sam_6.0.tar > dl-linux4sam_6.0.tar.md5
user@at91:~$ cat dl-linux4sam_6.0.tar.md5
1fb65e8de032394573bf047a99b95022  dl-linux4sam_6.0.tar
```

- Check release files

```
user@at91:~$ ls -l
total 1103804
drwxrwxr-x 17 user user      4096 Dec  4 18:23 buildroot-at91
drwxrwxr-x  9 user user      4096 Dec  4 17:52 buildroot-external-microchip
-rw-rw-r--  1 user user    74190326 Dec  4 17:55 buildroot-linux4sam_6.0.tgz
-rw-rw-r--  1 user user       62 Dec  4 18:28 buildroot-linux4sam_6.0.tgz.md5
-rw-rw-r--  1 user user 1056081920 Dec  4 18:23 dl-linux4sam_6.0.tar
-rw-rw-r--  1 user user       55 Dec  4 18:28 dl-linux4sam_6.0.tar.md5
```

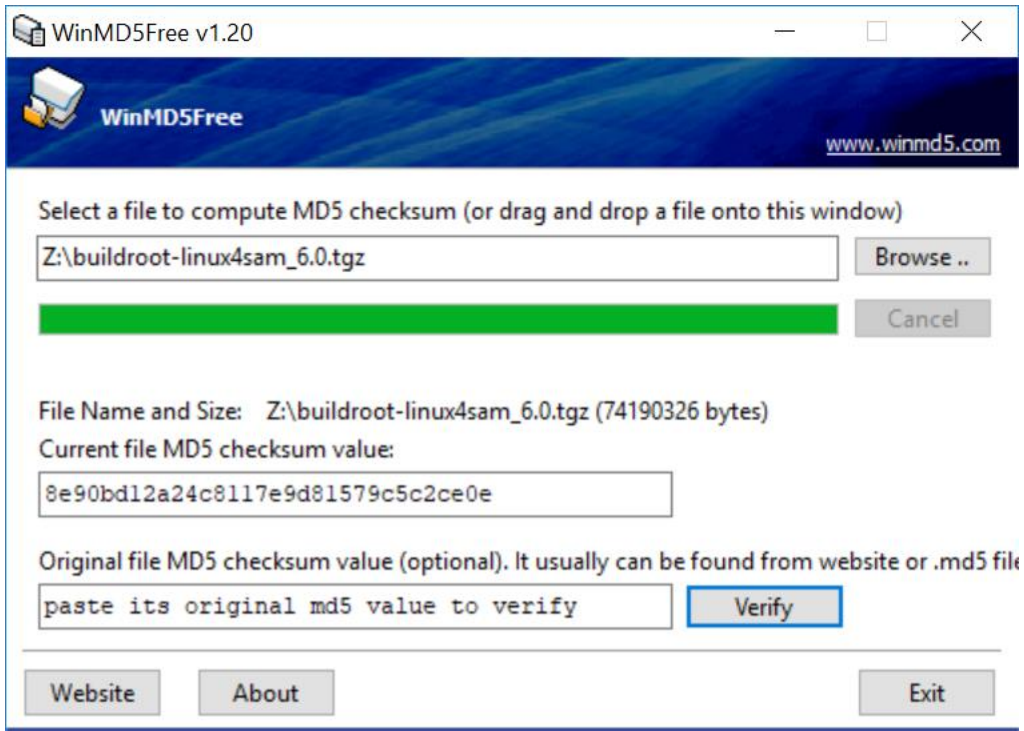
Appendix

A. How to check md5sum

- In Linux use md5sum command, two md5sum value strings should be the same.

```
user@at91:~$ ls
buildroot-linux4sam_6.0.tgz  buildroot-linux4sam_6.0.tgz.md5  dl-linux4sam_6.0.tar
dl-linux4sam_6.0.tar.md5
user@at91:~$ md5sum buildroot-linux4sam_6.0.tgz
8e90bd12a24c8117e9d81579c5c2ce0e  buildroot-linux4sam_6.0.tgz
user@at91:~$ cat buildroot-linux4sam_6.0.tgz.md5
8e90bd12a24c8117e9d81579c5c2ce0e  buildroot-linux4sam_6.0.tgz
```

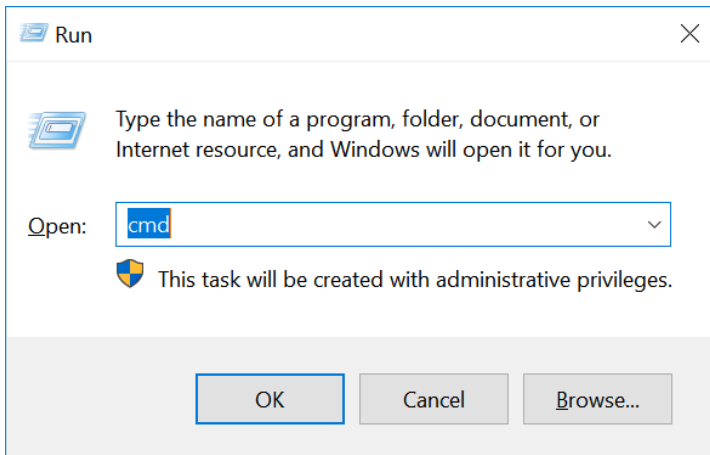
- In Windows use WinMD5 software



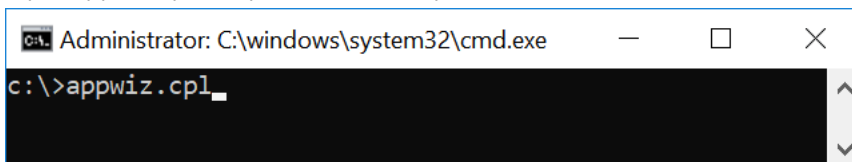
- Click "Browse" to select file or drop file into WinMD5 window directly
Then WinMD5 will automatically calculate the md5sum value
- Open buildroot-linux4sam_6.0.md5 with Notepad
Then copy md5sum value string into WinMD5 "Original file MD5 checksum value"
- Click "Verify" button

B. How to turn Windows features on or off

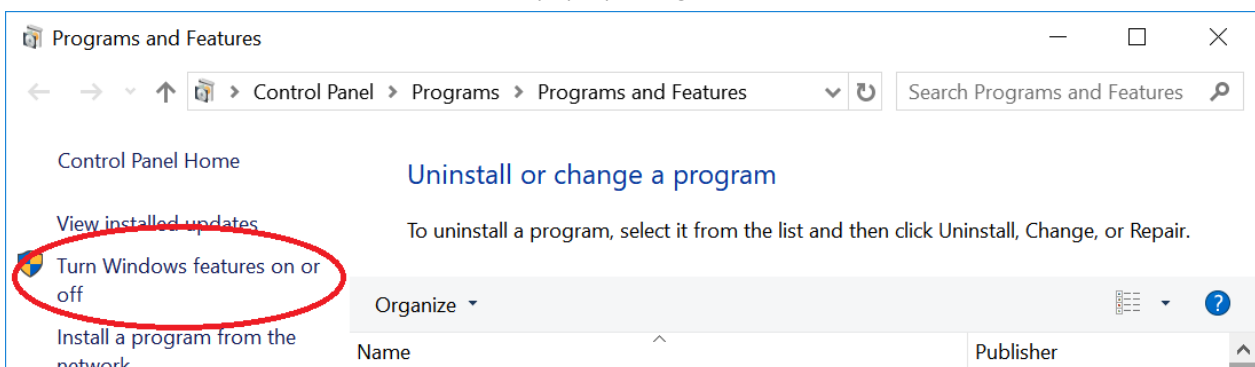
- Press “Winkey + r” to open Run window, type “cmd” and click “OK” button



- Input appwiz.cpl and press “Enter” key to execute it



- Click “Turn Windows features on or off” in the pop-up “Programs and Features” window



- Select your needed features and click “OK” (here we turn on “Services for NFS” and “TFTP Client”)

