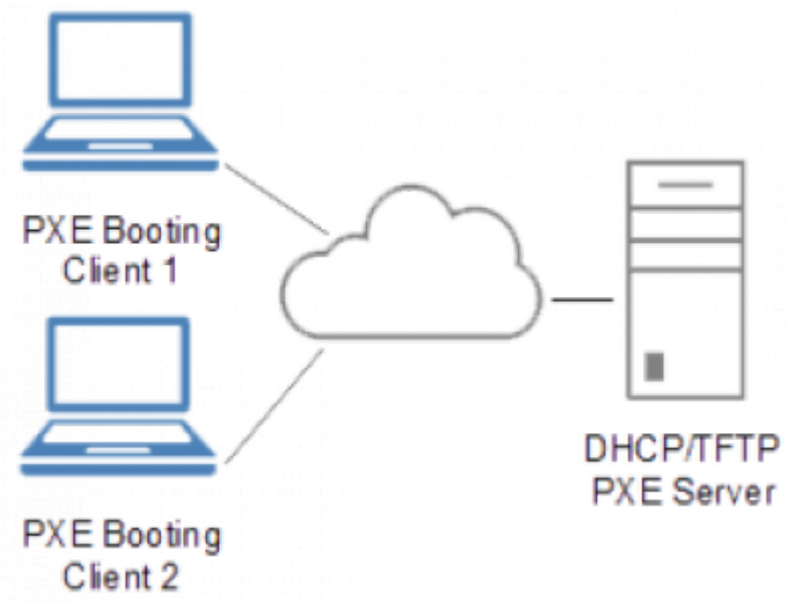
PXE boot on ODROID-XU4/MC1/HC1

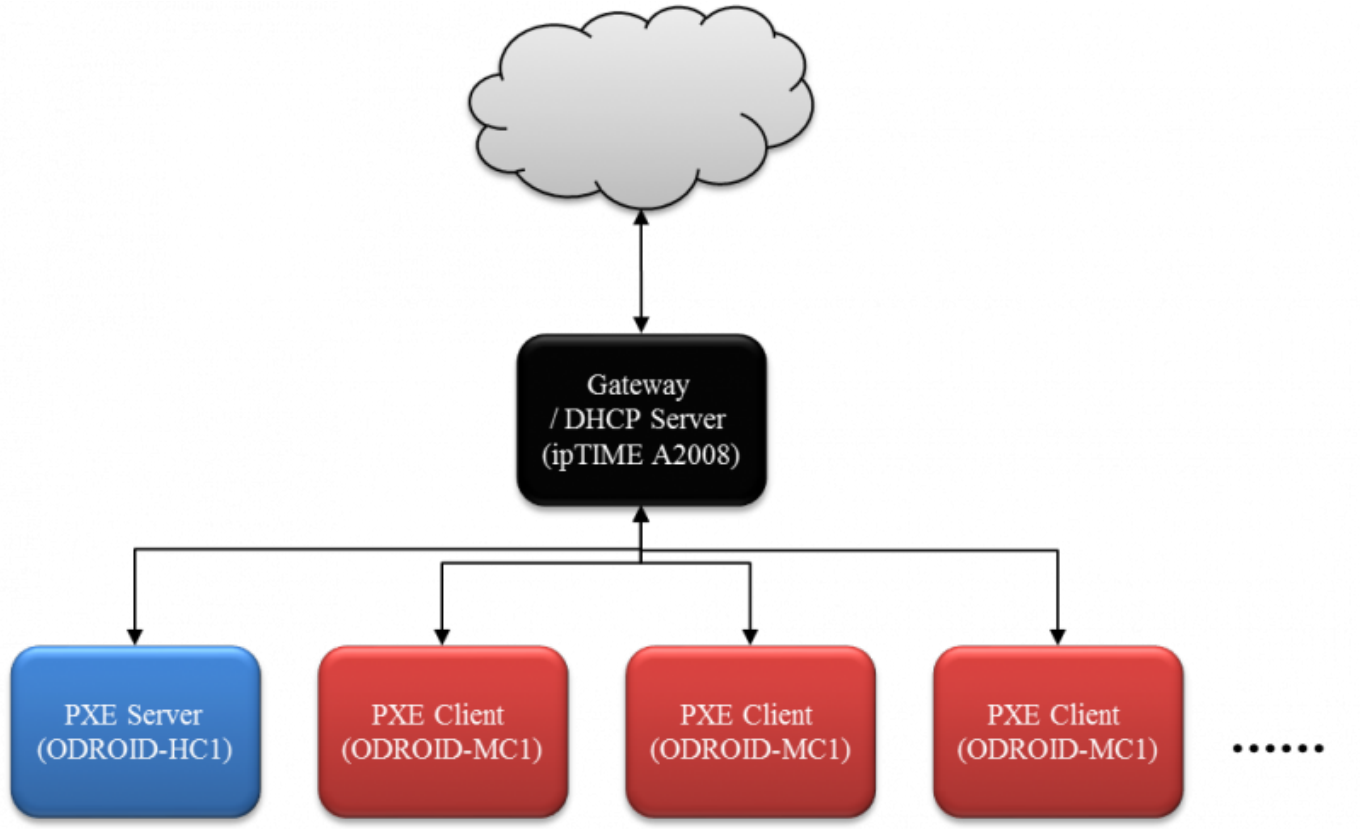
Description about **PXE (Preboot Execution Environment)** : “In computing, the Preboot eXecution Environment (PXE, sometimes pronounced as pixie) specification describes a standardized client-server environment that boots a software assembly, retrieved from a network, on PXE-enabled clients. On the client side it requires only a PXE-capable network interface controller (NIC), and uses a small set of industry-standard network protocols such as DHCP and TFTP. The concept behind the PXE originated in the early days of protocols like BOOTP/DHCP/TFTP, and as of 2015 it forms part of the Unified Extensible Firmware Interface (UEFI) standard. In modern data centers, PXE is the most frequent choice for operating system booting, installation and deployment.” [1]



- **A high-level PXE overview** [1]

This page will give you the guide how to set the PXE environment (server and clients) using ODROID-XU4/HC1/MC1. Although the NFS is used for the root filesystem of PXE client in this page, the ramdisk is also useful if the image size of the root filesystem is compact.
PXE is only available in the u-boot 2017.05 version, not 2012.07 version. All commands in this page are tested on Ubuntu minimal 16.04.3 for ODROID-XU4.

PXE System Diagram



DHCP server setting is unnecessary in this system because the gateway also has DHCP server function. DHCP server have to be installed in the PXE server if there is no external DHCP server unlike this system.

PXE Server Settings

ODROID-HC1 is used for PXE server in this guide. ODROID-XU4 and ODROID-XU4Q are also available with the same guide. But, the host pc is not allowed due to the root filesystem compatibility.

NFS Server Installation

The first step is installing NFS server for the root filesystem of PXE client. In this guide, PXE server is the same machine as PXE client (ODROID-XU4 series). So, PXE client will use the same root filesystem as PXE server via NFS.
Install the nfs server and copy the whole root filesystem to the nfs directory.

```
$ sudo apt update
$ sudo apt install nfs-kernel-server
$ sudo mkdir -p /media/nfs_root
$ sudo cp -ax / /media/nfs_root/
$ sudo nano /etc/exports
```

- **/etc/exports**

```
/media/nfs_root     *(rw,sync,no_root_squash)
```

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

TFTP Server Installation

The kernel and dtb images are transferred via TFTP during the PXE booting process.

```
$ sudo apt install tftpd-hpa
$ cd /var/lib/tftpboot
$ sudo mkdir odroidxu4
$ sudo mkdir pxelinux.cfg
$ sudo nano pxelinux.cfg/default-arm-exynos
```

- **/var/lib/tftpboot/pxelinux.cfg/default-arm-exynos**

```
DEFAULT odroidxu4_default

LABEL odroidxu4_default
kernel odroidxu4/zImage
fdt odroidxu4/exynos5422-odroidxu4.dtb
append console=tty1 console=ttySAC2,115200n8 root=/dev/nfs rw
nfsroot=${serverip}:/media/nfs_root/ net.ifnames=0
ip=${ipaddr}:${serverip}:${gatewayip}:${netmask}

PROMPT 1
TIMEOUT 0
```

If the installation and configuration of tftp server are finished for PXE, copy the kernel and dtb images to the tftp directory and restart the tftp server.

```
$ sudo cp /media/boot/zImage /media/boot/exynos5422-odroidxu4.dtb
/var/lib/tftpboot/odroidxu4/
$ sudo service tftpd-hpa restart
```

PXE Client Settings

- **Assumptions** (*Please modify below's variable to suit your environment*)
  - SD card device file: **/dev/sdc**
  - Mounted fat partition path: **/media/boot**
  - PXE Server IP address: **192.168.10.2**

Prepare blank SD card and insert it to host PC. Create FAT partition in order to contain the **boot.ini** file. (Type **n, p, 1, 2048, 264191, w**)

```
$ sudo fdisk /dev/sdc
elcome to fdisk (util-linux 2.27.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

```
Command (m for help): n
Partition type
   p   primary ( primary,  extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-15523839, default 2048): 2048
Last sector, +sectors or +size{K,M,G,T,P} (2048-15523839, default 15523839):
264191

Created a new partition 1 of type 'Linux' and of size 128 MiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
$ sudo mkfs.vfat -n boot /dev/sdc1
```

Eject the SD card and insert it to host PC again. Save the **boot.ini** file to the FAT partition.

```
$ sudo wget -O /media/boot/boot.ini https://git.io/v5ZNg
$ sudo nano /media/boot/boot.ini
```

Modify the **boot.ini** file (at last line). (Comment last line and add the three additional lines.)

- **/media/boot/boot.ini**

```
# Boot the board
# bootz 0x40008000 0x42000000 0x44000000
setenv serverip 192.168.10.2
setenv bootfile default-arm-exynos
run bootcmd_pxe
```

Get and burn the u-boot 2017.05 version to the SD card.

```
$ git clone --depth 1 https://github.com/hardkernel/u-boot.git -b odroidxu4-
v2017.05
$ cd u-boot/sd_fuse
$ sudo ./sd_fusing.sh /dev/sdc
```

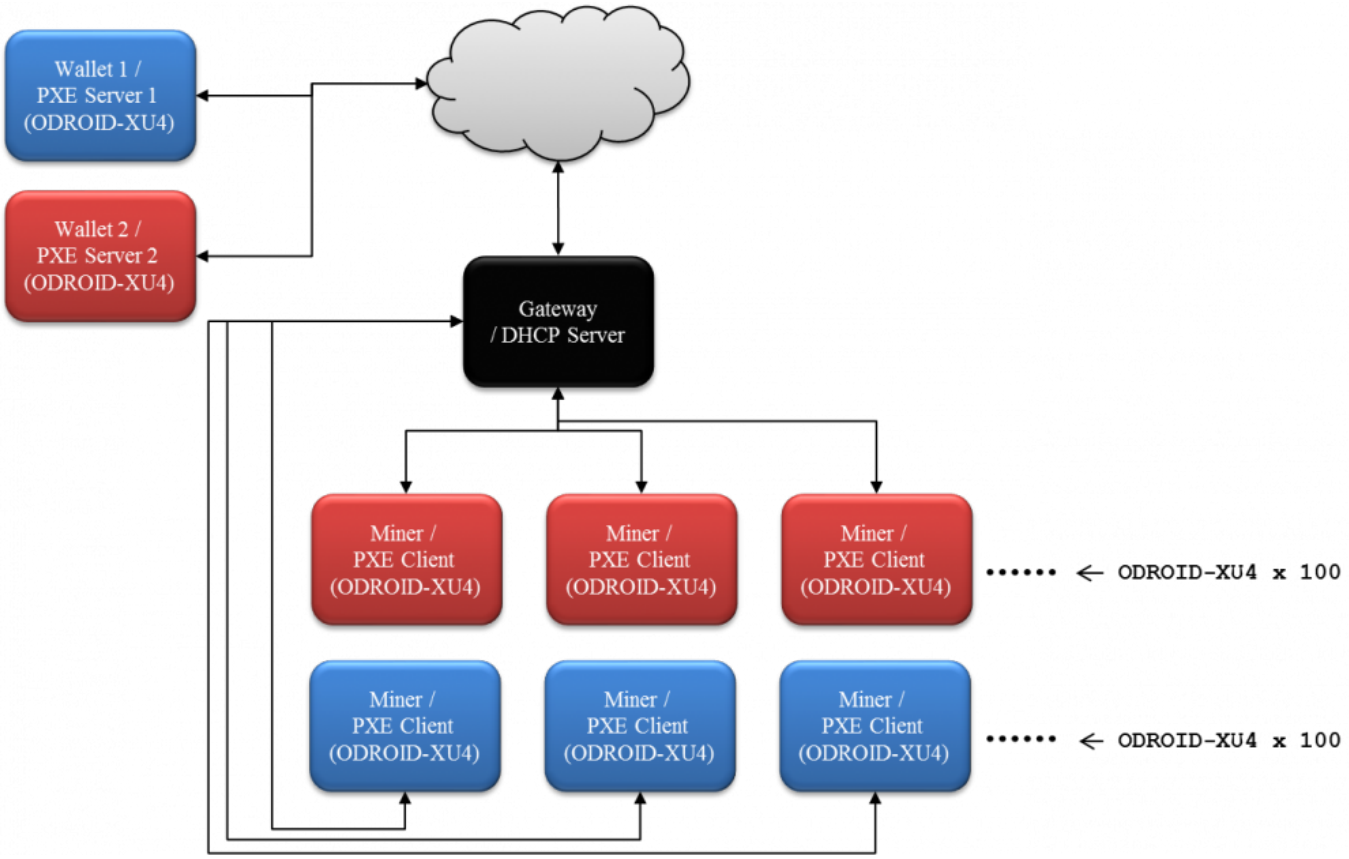
Eject the SD card from host PC and then insert the SD card to the PXE client machine like ODROID-MC1. Power on it. Done!

Example: Large Cryptocurrency Mining System

DO NOT recommend using ODROID-XU4 as cryptocurrency miner. Please refer to this comment.

ODROID-XU4 has both low power consumption and good performance. It is sometimes used for some cryptocurrency(as Verium) mining. About two hundreds ODROID-XU4 were mining Verium. When we make this system, there were two big problems — very long preparing SD card (burning OS image)

time and system upgrade. That was about two hundreds nodes, not just one machine. So, we built the system using PXE as distributed several groups.



Buildroot is used for the root filesystem of miner because miner only executes mining program. Buildroot can build the compact root filesystem image.

- Buildroot for verium miner source repository

The image (rootfs.cpio.gz) size of root filesystem is just 18M. So, the root filesystem in this system utilizes ramdisk.

- default-arm-exynos

```
DEFAULT odroidxu4_default

LABEL odroidxu4_default
kernel odroidxu4/zImage
fdt odroidxu4/exynos5422-odroidxu4.dtb
append console=tty1 console=ttySAC2,115200n8 root=/dev/ram0 rootwait rw
fsck.repair=yes net.ifnames=0
initrd odroidxu4/rootfs.cpio.gz

PROMPT 1
TIMEOUT 0
```

We could solve the problems with PXE! PXE is very convenient solution in such this system.

Video

Reference

[1] Preboot Execution Environment