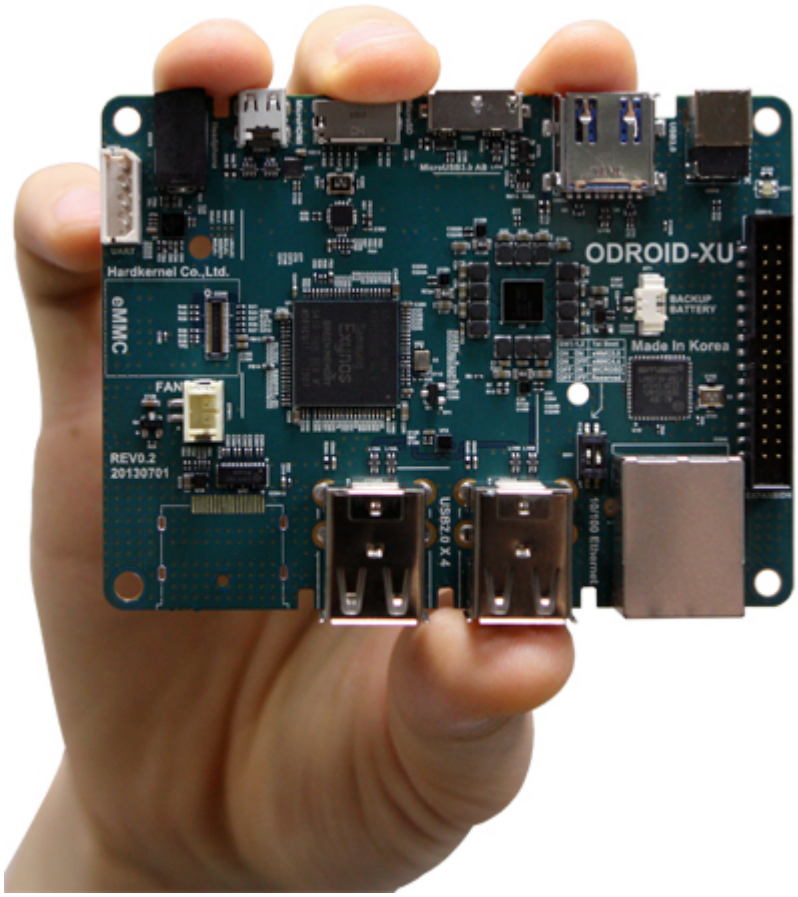
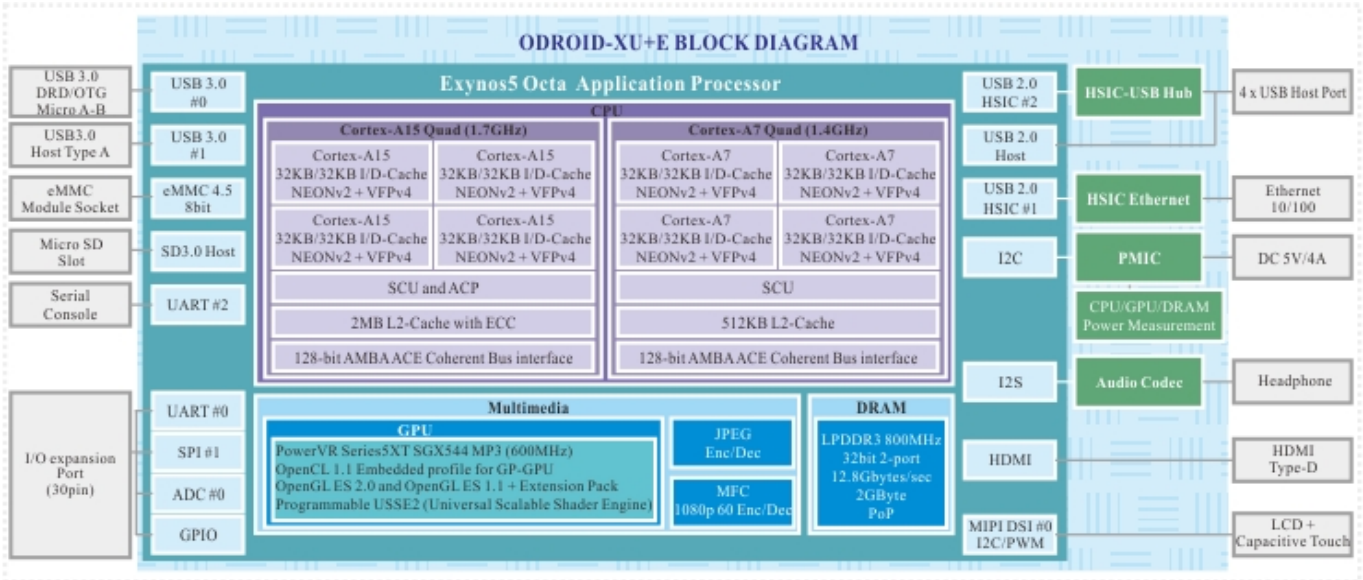
# Introduction

ODROID-XU is the world’s first big.LITTLE architecture based bare-board computer.




ODROID-XU+E model has the integrated power analysis tool. This package contains a special ODROID-XU board which has 4 current/voltage sensors to measure the power consumption of the Big A15 cores, Little A7 cores, GPUs and DRAMs individually. The professional developers can monitor

CPU, GPU and DRAM power consumption via included on-board power measurement circuit. By using the integrated power analysis tool, developers will reduce the need for repeated trials when debugging for power consumption and get the opportunity to enhance and optimize the performance of their CPU/GPU compute applications, and therefore keeping power consumption as low as possible.

---

# Specifications

| | |
|---|---|
| Processor | Samsung Exynos5 Octa ARM Cortex™-A15 Quad 1.6Ghz and Cortex™-A7 Quad 1.2GHz CPUs |
| Memory | 2Gbyte LPDDR3 RAM PoP (800Mhz, 1600Mbps/pin, 2 x 32bit Bus) |
| 3D Accelerator | PowerVR SGX544MP3 GPU (OpenGL ES 2.0, OpenGL ES 1.1 and OpenCL 1.1 EP) |
| Video | supports 1080p via HDMI cable(H.264+AAC based MP4 container format) |
| Video Out | micro HDMI connector |
| Audio | On-board Audio codec, Standard 3.5mm headphone jack, HDMI Digital |
| USB3.0 Host | SuperSpeed USB standard A type connector x 1 port |
| USB3.0 OTG | SuperSpeed USB Micro A-B type connector x 1 port |
| USB2.0 Host | High Speed standard A type connector x 4 ports |
| Display | HDMI monitor |
| Storage (Option) | MicroSD Card Slot, eMMC module socket : eMMC 4.5 Flash Storage |
| Fast Ethernet LAN | 10/100Mbps Ethernet with RJ-45 Jack ( Auto-MDIX support) |
| Gigabit Ethernet LAN(Option) | USB3.0 to Gigabit Ethernet adapter (USB module) |
| WiFi (Option) | USB IEEE 802.11b/g/n 1T1R WLAN with Antenna (USB module) |
| HDD/SSD SATA interface (Optional) | SuperSpeed USB (USB 3.0) to Serial ATA3 adapter for 2.5“/3.5” HDD and SSD storage |
| Power (included) | 5V 4A Power |
| Case(included) | Mechanical case & cooler (98 x 74 x 29 mm approx. ) |
| PCB Size | 94 x 70 x 18 mm approx. |

# Expansion Connectors

The Odroid-xu povides one 30-pin dual row expansion header “**CON10**” and one 40pin connector “ **CON15**” with MIPI DSI for LCD. The location and pinout of these connectors is illustrated blew. All signals on expansion headers are 1.8V except PWRON signal.

---

CON10 - 2×15 pins

| Pin Number | Expansion Net Name | Pin Name of Exynos5410 | Description | Pin Number | Expansion Net Name | Pin Name of Exynos5410 | Description |
|---|---|---|---|---|---|---|---|
| 1 | 5V0 | | Output Power from DC Adaptor | 2 | GND | | Ground |
| 3 | ADC_0.AIN0 | XADC0AIN_0 | | 4 | UART_0.RTSN | UART_0_RTSn | Requests to Send(active low) for UART0 |
| 5 | UART_0.CTSN | UART_0_CTSn | Clears to Send(active low) for UART0 | 6 | UART_0.RXD | UART_0_RXD | Receives Data for UART0 |
| 7 | SPI_1.MOISI | SPI_1_MOSI | In Master mode, this port is used to trasmit data to external slave. | 8 | UART_0.TXD | UART_0_TXD | Transmits Data for UART0 |
| 9 | SPI_1.MISO | SPI_1_MISO | In Master mode, this port is used to receive data from external slave. | 10 | SPI_1.CLK | SPI_1_CLK | Serial Clock |
| 11 | SPI_1.CSN | SPI_1_nSS | Slave Selection Signal | 12 | PWRON | External Power On Signal | This pin has an pull-down (about 800kohm) of PMIC inside. |
| 13 | XE.INT13(GPX1.5) | EXT_INT13 | GPIO and External Interrupt | 14 | XE.INT19(GPX2.3) | EXT_INT19 | GPIO and External Interrupt |
| 15 | XE.INT10(GPX1.2) | EXT_INT10 | | 16 | XE.INT8(GPX1.0) | EXT_INT8 | |
| 17 | XE.INT14(GPX1.6) | EXT_INT14 | | 18 | XE.INT11(GPX1.3) | EXT_INT11 | |
| 19 | XE.INT22(GPX2.6) | EXT_INT22 | | 20 | XE.INT20(GPX2.4) | EXT_INT20 | |
| 21 | XE.INT21(GPX2.5) | EXT_INT11 | | 22 | XE.INT23(GPX2.7) | EXT_INT23 | |
| 23 | XE.INT18(GPX2.2) | EXT_INT18 | | 24 | XE.INT17(GPX2.1) | EXT_INT17 | |
| 25 | XE.INT15(GPX1.7) | EXT_INT15 | | 26 | XE.INT16(GPX2.0) | EXT_INT16 | |
| 27 | XE.INT25(GPX3.1) | EXT_INT25 | | 28 | GND | | Ground |
| 29 | VDD_IO | | LDO3 of PMIC with 1.8V | 30 | GND | | Ground |

CON15 - 1×40 pins

| Pin Number | Expansion Net Name | Pin Name of Exynos5410 | Description |
|---|---|---|---|
| 1 | NC | | |
| 2 | VDD_IO | | |
| 3 | | | |
| 4 | | | |
| 5 | NC | | |
| 6 | I2C_0.SCL | I2C_0_SCL | |
| 7 | I2C_0.SDA | I2C_0_SDA | |

| Pin Number | Expansion Net Name | Pin Name of Exynos5410 | Description |
|---|---|---|---|
| 8 | MIPI-DSI.D0_N | MIPI0_DN_0 | Specifies the DN signal for MIPI-D-PHY0 Master data-lane 0 |
| 9 | MIPI-DSI.D0_P | MIPI0_DP_0 | Specifies the DP signal for MIPI-D-PHY0 Master data-lane 0 |
| 10 | GND | | Ground |
| 11 | MIPI-DSI.D1_N | MIPI0_DN_1 | Specifies the DN signal for MIPI-D-PHY0 Master data-lane 1 |
| 12 | MIPI-DSI.D1_P | MIPI0_DP_1 | Specifies the DP signal for MIPI-D-PHY0 Master data-lane 1 |
| 13 | GND | | Ground |
| 14 | MIPI-DSI.CLK_N | MIPI0_CLK_TX_N | Specifies the DN signal for MIPI-D-PHY0 Master clock-lane |
| 15 | MIPI-DSI.CLK_P | MIPI0_CLK_TX_P | Specifies the DP signal for MIPI-D-PHY0 Master clock-lane |
| 16 | GND | | Ground |
| 17 | MIPI-DSI.D2_N | MIPI0_DN_2 | Specifies the DN signal for MIPI-D-PHY0 Master data-lane 2 |
| 18 | MIPI-DSI.D2_P | MIPI0_DP_2 | Specifies the DP signal for MIPI-D-PHY0 Master data-lane 2 |
| 19 | GND | | Ground |
| 20 | MIPI-DSI.D3_N | MIPI0_DN_3 | Specifies the DN signal for MIPI-D-PHY0 Master data-lane 3 |
| 21 | MIPI-DSI.D3_P | MIPI0_DP_3 | Specifies the DP signal for MIPI-D-PHY0 Master data-lane 3 |
| 22 | GND | | Ground |
| 23 | XE.INT5 | EXT_INT5 | External Interrupt |
| 24 | NC | | |
| 25 | GND | | Ground |
| 26 | UART_1.RXD | UART_1_RXD | Receive Data for UART1 |
| 27 | UART_1.TXD | UART_1_TXD | Transmits Data for UART1 |
| 28 | GND | | Ground |
| 29 | UART_3.TXD | UART_3_TXD | Transmits Data for UART3 |
| 30 | UART_3.RXD | UART_3_RXD | Receive Data for UART3 |
| 31 | GND | | Ground |
| 32 | GND | | Ground |
| 33 | GND | | Ground |
| 34 | NC | | |
| 35 | PWM.TOUT3 | TOUT_0 | PWMTIMER_TOUT[3] |
| 36 | NC | | |
| 37 | NC | | |
| 38 | P5V0 | | Output Power from DC Adaptor |
| 39 | | | |
| 40 | | | |

# Expansion Boards and Accessories

# Boot Loader

## Boot Media

On ODROID-XU is a DIP Switch (sw1) to select the booting Device.

- both eMMC5.0 and 4.4 booting mode of following table are whatever for eMMC booting.

| SW1-1,2 | 1st Boot media |
|---|---|
| ON ON | eMMC5.0 |
| ON OFF | eMMC4.4 |
| OFF ON | MicroSD card |
| OFF OFF | Reserved |

## Boot Sequence

Upon power on the board will search for the boot media. It will perform the following:

- 1) iROM (Code inside the SoC) will attempt to read the boot media at the first 512 bytes of it. On those first 512 bytes fwbl1 should exist.
- 2) fwbl1 will load bl2 (SPL) that is part of the U-boot.
- 3) bl2 will load U-boot.
- 4) U-Boot will do whats left, such as handle TrustZone, load kernel image if setted.

## fwbl1

This blob is the first thing that CPU will call, we don't have much information on it since its provided by Samsung. Position on SD: 1

## bl2

This is the SPL, part of the U-Boot, upon building U-Boot you'll have the mkspl program what will extract it from the u-boot.bin Position on SD: 31

## u-boot.bin

This is the U-Boot itself built. For more information about U-Boot please check their website: http://www.denx.de/wiki/U-Boot Position on SD: 63

## TrustZone Software

This is the blob done by Samsung/ARM to support Trustzone platform. Position on SD: 719

## Default Env

The default env provided in U-Boot will attempt the following:

- 1) Load boot.ini script from the first FAT type partition from the boot media.
- 2) Load kernel from its on media/ramdisk position (This is the Android Boot)

## Boot media sector map for Odroid-xu

- Odroid-xu has Min 8Gbyte eMMC or MicroSD memory card for system area.
- FAT partition to calculate the remaining blocks to create the partition.

| Area Name | Size in bytes | From(sector #) | To(Sector #) | Partition Name |
|---|---|---|---|---|
| FAT32 for Storage | Up to 4GB | 6979920 | remaining blocks | mmcblk0p1 |
| EXT4 for Android cache | 256MB | 6446520 | 6979919 | mmcblk0p4 |
| EXT4 for Android userdata | 2GB | 2240280 | 6446519 | mmcblk0p3 |
| EXT4 for Android system | 1GB | 137160 | 2240179 | mmcblk0p2 |
| Reserved | 58MB | 17647 | 137159 | |
| Kernel | 8MB | 1263 | 17646 | |
| u-boot environment | 16KB | 1231 | 1262 | |
| TrustZone SW | 256KB | 719 | 1230 | |
| u-boot | 328KB | 63 | 718 | |
| bl2 | 16KB | 31 | 62 | |
| fwbl1 | 15KB | 1 | 30 | |

| Area Name | Size in bytes | From(sector #) | To(Sector #) | Partition Name |
|---|---|---|---|---|
| Partition table / MBR | 512 | 0 | 0 | |

* “Sector number - 1” in SD map is used in the eMMC map.

# How to build u-boot

# Download the cross tool chain package

[ http://dn.odroid.com/ODROID-XU/compiler/arm-eabi-4.4.3.tar.gz ]

Note this tool chain only used to build the u-boot.

If you meet any compile error, try this tool chain package. [ http://dn.odroid.in/ODROID-XU/compiler/arm-eabi-4.6.tar.gz ]

# Copy the cross tool package to /opt/toolchains

If the '/opt/toolchains' directory does not exist in host pc, then create the directory.

```
# sudo mkdir /opt/toolchains
# sudo cp arm-eabi-4.4.3.tar.gz /opt/toolchains
# cd /opt/toolchains/
# sudo tar zxvf arm-eabi-4.4.3.tar.gz
```

# Uncompress the cross tool with tar command

```
# cd /opt/toolchains
# sudo tar xvfz arm-eabi-4.4.3.tar.gz
```

# Add Path in your environment file

Modify your ~/.bashrc file to add a new path with editor (gedit or vi)

```
export PATH=${PATH}:/opt/toolchain/arm-eabi-4.4.3/bin
export CROSS_COMPILE=arm-eabi-
```

To apply this change, login again or restart the .bashrc

```
# source ~/.bashrc
```

## Check the tool-chain path to see if it is set up correctly or not.

```
# arm-eabi-gcc -v
Using built-in specs.
Target: arm-eabi
Configured with:
/home/jingyu/projects/gcc/android-toolchainsrc/build/../gcc/gcc-4.4.3/config
ure --prefix=/usr/local --target=arm-eabi
--host=x86_64-linux-gnu --build=x86_64-linux-gnu --with-gnu-as --with-gnu-ld
--enable-languages=c,c++ --with-gmp=/home/jingyu/projects
/gcc/toolchain_build/obj/temp-install
--with-mpfr=/home/jingyu/projects/gcc/toolchain_build/obj/temp-install
--disable-libssp --enable
-threads --disable-nls --disable-libmudflap --disable-libgomp
--disable-libstdc__-v3 --disable-sjlj-exceptions --disable-shared --disa
ble-tls --with-float=soft --with-fpu=vfp --with-arch=armv5te
--enable-target-optspace --with-abi=aapcs --with-gcc-version=4.4.3 --with
-binutils-version=2.19 --with-gmp-version=4.2.4 --with-mpfr-version=2.4.1
--with-gdb-version=7.1.x --with-arch=armv5te --with-multilib
-list=mandroid
--with-sysroot=/usr/local/google/home/android/cupcake_rel_root
--program-transform-name='s&^&arm-eabi-&'
Thread model: single
gcc version 4.4.3 (GCC)
```

## Source code download

You can get the latest source from here.

## android-4.2.2

```
$ repo init -u https://github.com/hardkernel/android.git -b
5410_4.2.2_master
$ repo sync
$ repo start 5410_4.2.2_master --all
$ ./build.sh odroidxu platform
```

# android-4.4.2

```
$ repo init -u https://github.com/hardkernel/android.git -b
odroid_5410_master
$ repo sync
$ repo start odroid_5410_master --all
$ ./build.sh odroidxu platform
```

* Visit this link to install the repo. http://source.android.com/source/downloading.html

* To get the same version source of this Alpha 2.0 in the future, try below command of tag.

```
$ repo forall -c git reset --hard 5410_v2.0
```

### How to configure and compile

# Android-4.2.2

https://github.com/hardkernel/linux/tree/odroidxu-3.4.y-android-jb

Kernel configuration file : odroidxu_android_defconfig

```
$ git clone https://github.com/hardkernel/linux.git -b
odroidxu-3.4.y-android-jb
$ cd linux
$ make odroidxu_android_defconfig
$ make -j8
```

# Android-4.4.2

https://github.com/hardkernel/linux/tree/odroidxu-3.4.y-android

Kernel configuration file : odroidxu_android_defconfig

```
$ git clone https://github.com/hardkernel/linux.git -b
odroidxu-3.4.y-android
$ cd linux
$ make odroidxu_android_defconfig
$ make -j8
```

Write kernel

In the host PC.

```
#fastboot flash kernel arch/arm/boot/zIamge
```

Reboot system

In the host PC.

```
#fastboot reboot
```

- You will have spl/smdk5410-spl.bin and u-boot.bin, if you build it.

### Note

- **You need to sign the smdk5410-spl.bin with Hardkernel's private key to make it bootable.**
- **If you want to contribute your patch, apply it to our u-boot github admin. Or, contact "odroid.uboot@gmail.com"**
- **And we will publically release the signed bl2 image within 48 hours if there is any update.**
- To flash the updated images on your MicroSD card, refer the scriptor file sd_fusing.sh in sd_fuse directory.
- To flash the updated images on your eMMC moduel, refer the scriptor file emmc_fastboot_fusing.sh in sd_fuse directory.
- For eMMC update, you need a fastboot driver and micro-USB cable.

# Kernel

---

## Getting the Right Kernel

## Step-by-Step guide to building a odroid-xu Kernel

## Download the cross tool chain package

[ http://dn.odroid.com/ODROID-XU/compiler/arm-eabi-4.6.tar.gz ]

Note

- This toolchain only used to build the kernel.
- This toolchain is included in the Android source package. ($ANDROID_ROOT/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6)

## Copy the cross tool package to /opt/toolchains

If the '/opt/toolchains' directory does not exist in host pc, then create the directory.

```
# sudo mkdir /opt/toolchains
# sudo cp arm-eabi-4.6.tar.gz /opt/toolchains
# cd /opt/toolchains
# sudo tar zxvf arm-eabi-4.6.tar.gz
```

## Uncompress the cross tool with tar command

```
# cd /opt/toolchains
# sudo tar xvfz arm-eabi-4.6.tar.gz
```

## Add Path in your environment file

Modify your ~/.bashrc file to add a new path with editor (gedit or vi)

```
export ARCH=arm
export PATH=${PATH}:/opt/toolchains/arm-eabi-4.6/bin
export CROSS_COMPILE=arm-eabi-
```

To apply this change, login again or restart the .bashrc

```
# source ~/.bashrc
```

## Check the tool-chain path to see if it is set up correctly or not.

```
# arm-eabi-gcc -v
Using built-in specs.
COLLECT_GCC=arm-eabi-gcc
COLLECT_LTO_WRAPPER=/opt/toolchain/arm-eabi-4.6/bin/../libexec/gcc/arm-eabi/
4.6.x-google/lto-wrapper
Target: arm-eabi
Configured with: /tmp/android-15472/src/build/../gcc/gcc-4.6/configure
--prefix=/usr/local --target=arm-eabi --host=x86_64-linux-gnu
--build=x86_64-linux-gnu --with-gnu-as --with-gnu-ld
--enable-languages=c,c++ --with-gmp=/tmp/android-15472/obj/temp-install
--with-
```

```
mpfr=/tmp/android-15472/obj/temp-install
--with-mpc=/tmp/android-15472/obj/temp-install --without-ppl --without-cloog
--disable-libs
sp --enable-threads --disable-nls --disable-libmudflap --disable-libgomp
--disable-libstdc__-v3 --disable-sjlj-exceptions --disable-
shared --disable-tls --disable-libitm --with-float=soft --with-fpu=vfp
--with-arch=armv5te --enable-target-optspace --with-abi=aapcs
 --with-gcc-version=4.6 --with-binutils-version=2.21
--with-gmp-version=4.2.4 --with-mpfr-version=2.4.1 --with-gdb-version=7.3.x
--w
ith-arch=armv5te --with-sysroot=/tmp/android-15472/install/sysroot
--with-prefix=/tmp/android-15472/install --with-gold-version=2.21
 --enable-gold --disable-gold --disable-multilib
--program-transform-name='s&^&arm-eabi-&'
Thread model: single
gcc version 4.6.x-google 20120106 (prerelease) (GCC)
```

# Source code download

## Android-4.2.2

https://github.com/hardkernel/linux/tree/odroidxu-3.4.y-android-jb

Kernel configuration file : odroidxu_android_defconfig

```
$ git clone https://github.com/hardkernel/linux.git -b
odroidxu-3.4.y-android-jb
$ cd linux
$ make odroidxu_android_defconfig
$ make -j8
```

## Android-4.4.2

https://github.com/hardkernel/linux/tree/odroidxu-3.4.y-android

Kernel configuration file : odroidxu_android_defconfig

```
$ git clone https://github.com/hardkernel/linux.git -b
odroidxu-3.4.y-android
$ cd linux
$ make odroidxu_android_defconfig
$ make -j8
```

# Android

http://source.android.com/source/initializing.html

Installing required packages (Ubuntu 12.04) Building on Ubuntu 12.04 is currently only experimentally supported and is not guaranteed to work on branches other than master.

```
$ sudo apt-get install git gnupg flex bison gperf build-essential \
  zip curl libc6-dev libncurses5-dev:i386 x11proto-core-dev \
  libx11-dev:i386 libreadline6-dev:i386 libgl1-mesa-glx:i386 \
  libgl1-mesa-dev g++-multilib mingw32 tofrodos \
  python-markdown libxml2-utils xsltproc zlib1g-dev:i386
```

```
$ sudo ln -s /usr/lib/i386-linux-gnu/mesa/libGL.so.1
/usr/lib/i386-linux-gnu/libGL.so
```

Installing the JDK The Oracle/Sun JDK is no longer in Ubuntu's main package repository. In order to download it, you need to add the appropriate repository and indicate to the system which JDK should be used.

If the JDK version is higher than 1.6.0_39, you may meet some compilation errors.

```
sudo add-apt-repository "deb http://ftp.debian.org/debian squeeze main
contrib non-free"
sudo apt-get update
sudo apt-get install sun-java6-jdk
```

```
$ java -version
java version "1.6.0_26"
Java(TM) SE Runtime Environment (build 1.6.0_26-b03)
Java HotSpot(TM) 64-Bit Server VM (build 20.1-b02, mixed mode)
```

Configuring USB Access Under GNU/linux systems (and specifically under Ubuntu systems), regular users can't directly access USB devices by default. The system needs to be configured to allow such access.

The recommended approach is to create a file /etc/udev/rules.d/51-android.rules (as the root user) and to copy the following lines in it. <username> must be replaced by the actual username of the user who is authorized to access the phones over USB.

```
SUBSYSTEM=="usb", ATTRS{idVendor}=="18d1", MODE="0666"
```

# Source code download

You can get the latest source from here.

http://dn.odroid.com/ODROID-XU/Android_bsp/

Build android

```
./build.sh odroidxu platform
```

```
[[[[[[[ OUTPUT FOLDER = /media/codewalker/SSD/adonis/odroidxu-img ]]]]]]]

ok success !!!
```

Write system.img

In the host PC.

```
#fastboot flash system out/target/product/odroidxu/system.img
```

</code> Write userdata.img

In the host PC.

```
#fastboot flash userdata out/target/product/odroidxu/userdata.img
```

Write cache.img

In the host PC.

```
#fastboot flash userdata out/target/product/odroidxu/cache.img
```

Reboot system

In the host PC.

```
#fastboot reboot
```

Note : Ramdisk contents are merged into system.img to minimize the build steps. You can also easily modify the system configuration for example init.*.rd file.

So there is no Ramdisk parition in the storage media.

# Linux

Linux ODROID-XU works as follow:

It must have at least two partitions.

First Partition must be a FAT32 partition.

Second Partition can be whatever the Filesystem that your kernel supports (must be built in).

Note: It is possible to use the first partition as ext2, however its strongly not recommended due to Windows Users lost the capability of changing boot.scrs

# Partition Contents

Partition 1:

- Kernel Image (zImage)
- boot.scr
- uInitrd (if applicable)

Partition 2:

- rootfs (a.k.a. File System)

# Currently Supported Linux Distributions

- Ubuntu 13.04 Ubuntu 13.04 Forum Thread
- Debian Wheezy (can be upgraded to Jessie) Debian Wheezy Forum Thread
- Ubuntu 13.07 Server (Linaro's Version) Ubuntu Server Forum Thread

Note: More distributions such as Fedora/OpenSUSE to come

# HDMI Support On Linux

HDMI On Linux works via the Exynos5 Hardware Composer. It is available on our github. Its on tools/hardkernel of branch odroidxu-3.4.y

Installation of it for custom proposes is very simple.

1. sh autogen.sh
2. ./configure
3. make
4. make install

This will install the exynos5-hwcomposer that must he running in order to get X11/Xorg output.

# HDMI Resolution Change

Right now 720P 60Hz and 1080P 60Hz are the supported modes for the HDMI.

It is possible to change it on OFFICIAL images listed above by chaning the boot.scr file on the fat partition.

We current supply inside the FAT partition both boot.scr examples, just rename to what you plan to use.

# Kernel Sources

Kernel sources for ODROID-XU is on our Github. Branch is **odroidxu-3.4.y** defconfig is **odroidxu_ubuntu_defconfig**

# Kernel Rebuild Guide

Please follow the instructions below to rebuild the Linux Kernel for ODROID.

Those instructions cover native build of the Kernel.

1. Install dependencies: **apt-get install build-essential libqt4-dev libncurses5-dev git**
2. Clone Repo: **git clone –depth 0 https://github.com/hardkernel/linux.git -b odroidxu-3.4.y odroidxu-3.4.y**

```
 git clone --depth 0 https://github.com/hardkernel/linux.git -b
odroidxu-3.4.y odroidxu-3.4.y
```

1. Configure Kernel: **make odroidxu_ubuntu_defconfig**
2. Do changes if you need/want: **make menuconfig** or **make xconfig**
3. Build Kernel and Modules: **make -j5 zImage modules**
4. Install zImage: **cp arch/arm/boot/zImage /media/boot**
5. Install Modules: **make modules_install**
6. Copy .config to /boot for initramfs creation: **cp .config /boot/config-3.4.5**
7. Create initramfs: **update-initramfs -c -k 3.4.5**
8. Create uInitrd: **mkimage -A arm -O linux -T ramdisk -C none -a 0 -e 0 -n uInitrd -d /boot/initrd.img-3.4.5 /boot/uInitrd-3.4.5**
9. Install new uInitrd: **cp /boot/uInitrd-3.4.5 /media/boot/uInitrd**
10. reboot: **sync && reboot**

Your new kernel should be installed. Note for your own convenience we provide daily builds of Linux kernel that can be easily installed on Supported distros by using a Simple Script.

```
wget builder.mdrjr.net/tools/kernel-update.sh
sh kernel-update.sh
```

Please make sure that you always download a new script before running it. Small fixes for the rootfs can be added to the script itself.

# References

---