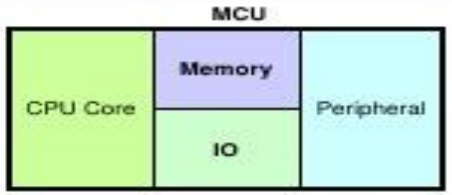
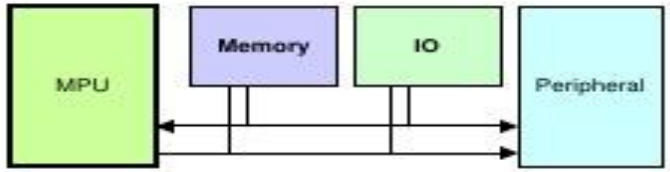
EMBEDDED SYSTEMS TRAINING

Khóa học : “Basic Embedded Linux”

- **Email** : *training.laptrinhnhung@gmail.com*
- **Website** : *http://hethongnhung.com* – *http://laptrinhnhung.com*

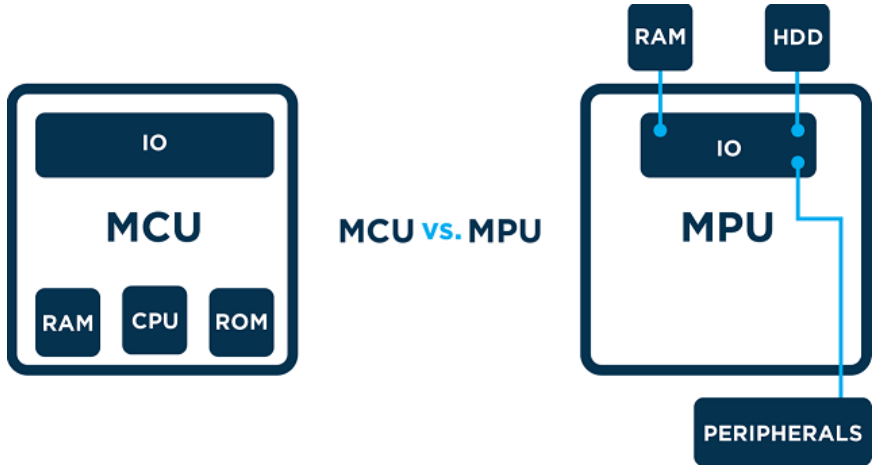
difference between MPU and MCU

MCU and MPU

Classification	MCU	MPU
Chip	Embedded A CPU core, memory, peripherals, IO … into a single chip.	Contain only the main processor (CPU core)
Block Diagram of a system		
General Application Area	- Self contained to complete a task. - Targeted for small, compact, and low cost system	-Need external memory, peripheral to accomplish a task - Targeted for complex, high performance and expandable system
Processor (CPU) Core	4, 8, 16 bit	32 bit or above
Examples	8051,pic16f887a,M16, H8, SH1/2	SH3/4,8085
Application	Washing machine, car side mirror, air con	Handphone, PDA

difference between MPU and MCU

Attributes	Microcontrollers	Microprocessors
Application	Are application specific and are designed to perform certain limited tasks.	Have generic application and are capable of executing big and complicated tasks.
One Solution	Have inbuilt processor, RAM, ROM and I/O Ports. Like a small stand-alone computer in a single Integrate Chip.	Generally don't have inbuilt RAM, ROM and I/O ports. The pins are used to interface with external RAM, ROM and ports.
Performance	Limited performance.	Very high performance.
Speed	Generally operate at speeds from 8 MHz – 200 MHz.	Generally operate at speeds above 1 GHz.
Power Consumption	Are embedded inside other devices, so are designed to consume less power.	Consume relative more power. As performance is the given higher weightage over power.
Cost	Affordable and cheap. Can get started with a minimum external circuit.	Very expensive and requires other peripherals to work along.



ARM cotex-A9 and M4 on UDOO NEO

Cortex®-M processors

MCU + DSP

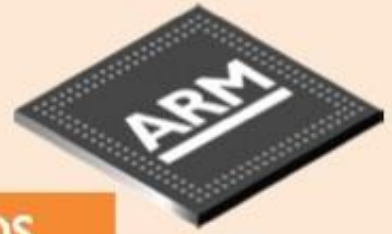
RTOS

Smallest footprint / lowest power

Cortex®-R processors

Highest performance / real-time

Cortex®-A processors

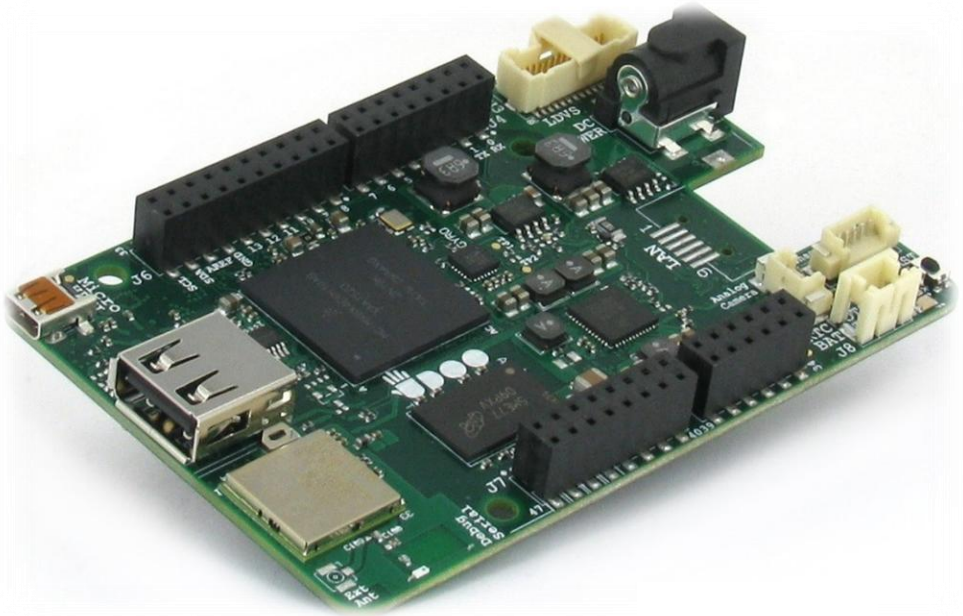


Rich OS

Highest performance

Starting up the board

Competition summary – Standard competition set (cont.)



(*) Display only for graphics-related projects

http://www.udoo.org/docs-neo/Hardware_Reference/Overview.htm

Starting up the board

"UDOO boards have no internal storage or built-in boot code, OS and storage are on microSD so you need to plug a pre-loaded microSD to boot."

The steps to start the board are:

1 - Download UDOO NEO's official Operating system

2 - Prepare the Micro SD Card

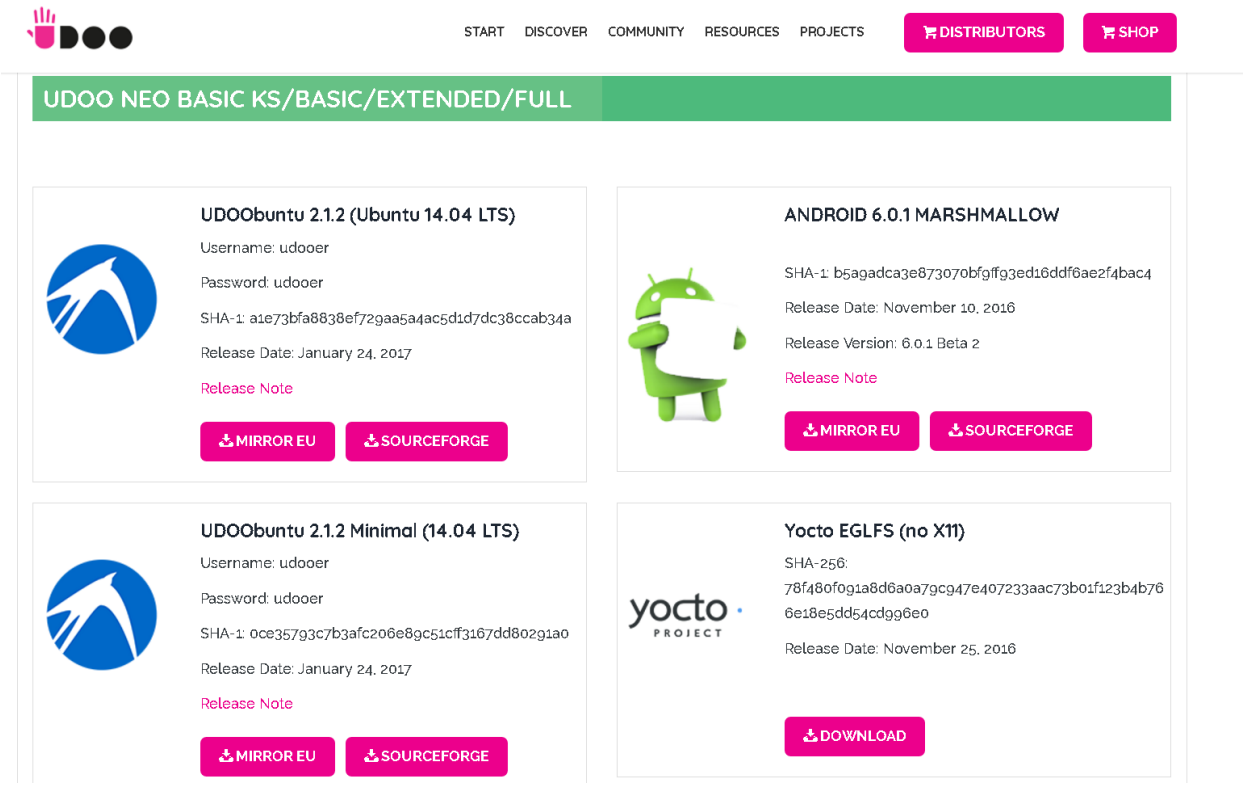
3 - Insert the Micro SD Card

4 - Power up via DC connector or micro USB if no display is connected to board

Starting up the board - UDOO NEO's official Operating system

Download the official image from Site Download section to your PC and unzip it

http://www.udoo.org/downloads/



You can choose from :

- UDOObuntu minimal (console mode)
- UDOObuntu full (recommended)
- Android
- Yocto

Starting up the board – prepare the uSD card

Create a bootable uSD card with the downloaded image.

WindowsOS, **MacOS** and **LinuxOS** procedures are described in documentation section on official site.

http://www.udoo.org/docs/Getting_Started/Create_A_Bootable_MicroSD_card_for_UDOO_QUAD-DUAL.html

For WindowsOS the application

Win32DiskImager.exe

can be used to write the image

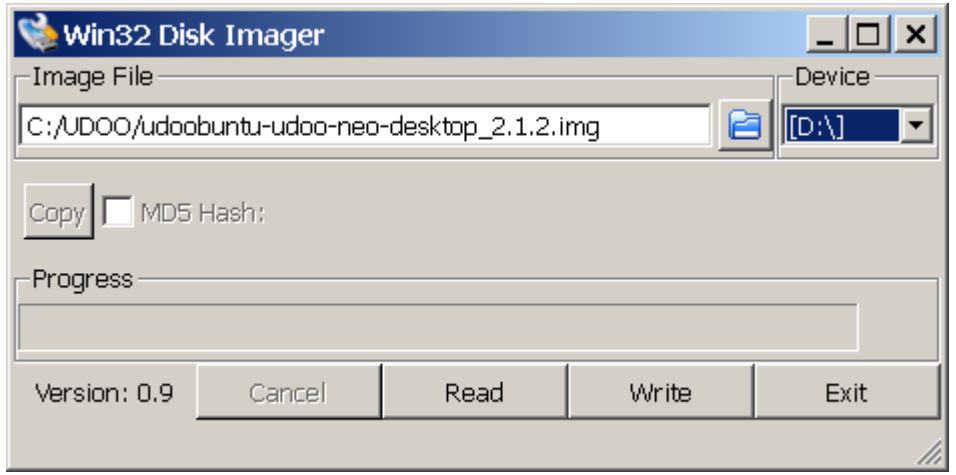
For LinuxOS the command

dd

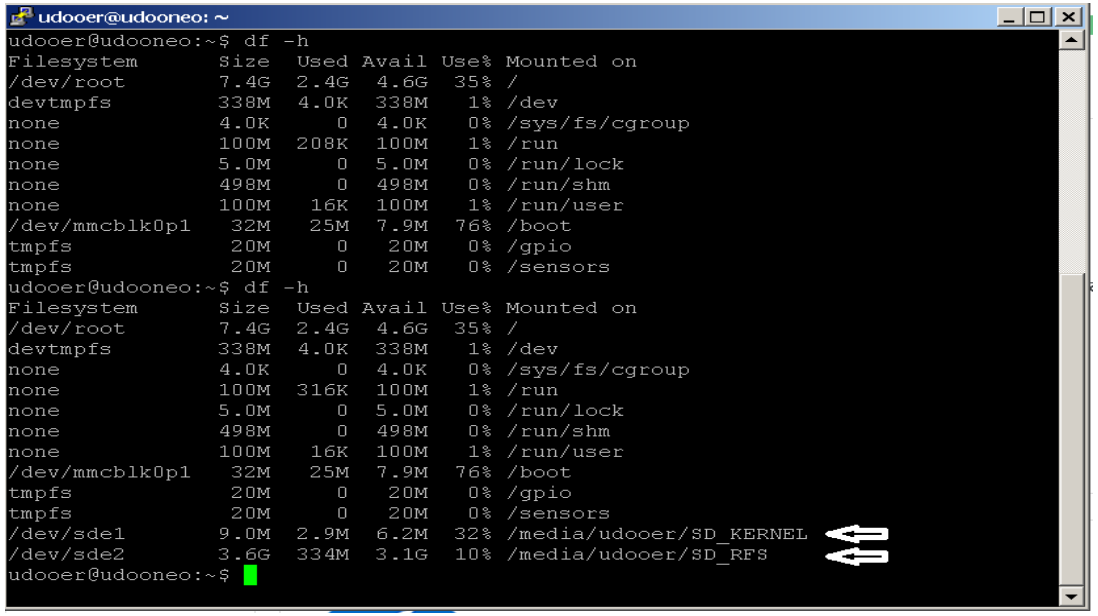
can be used to write the image.

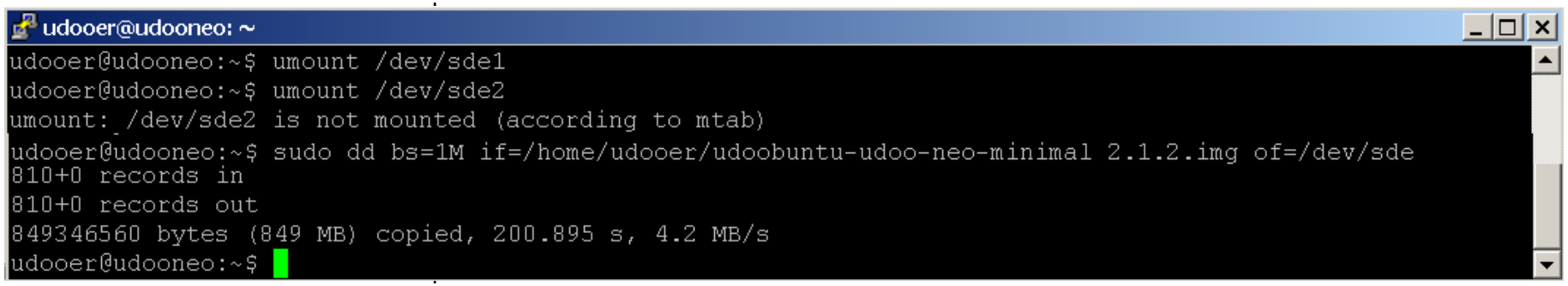
Starting up the board – prepare the uSD card

WindowsOS



LinuxOS





Starting up the board - Insert the Micro SD Card

uSD card should be inserted when board is NOT powered (USB or DC power).

Starting up the board - Power up via Micro USB or DC adapter

There are several ways you may choose to work/connect with your board. They are described in UDOO NEO documentation site on Getting Started section.

You may choose to use it as

- a desktop (monitor/lvds)

http://www.udoo.org/docs-neo/Getting_Started/Use_as_a_Lightweight_Desktop_PC.html

Recommendation for this case is to power up the board by DC adapter.

- an headless device (no display attached to board), case in which you may choose to connect by USB as
  - an SSH connection
  - an serial terminal connection
  - an VNC connection

http://www.udoo.org/docs-neo/Getting_Started/Use_as_a_headless_IoT_Device.html

You may power up the board by **USB only** if the LVDS is not attached to the board.

Headless UDOO – network available or emulated/serial

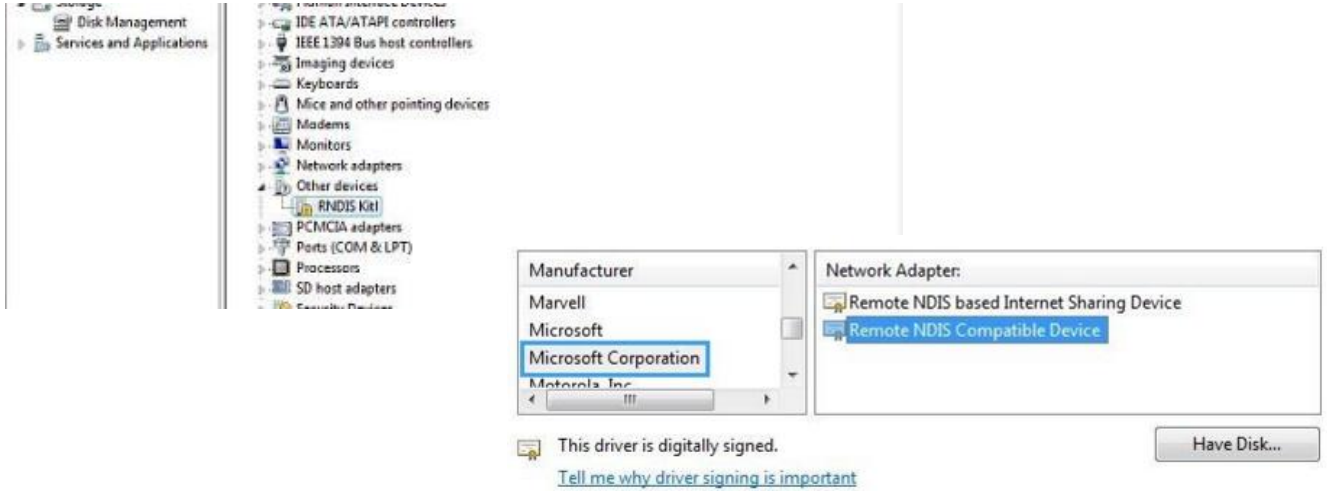
You can access the UDOO board by a remote connection and not using LVDS or monitor.

http://www.udoo.org/docs-neo/Basic_Setup/Usb_Direct_Connection.html

The USB connection provides ETH connection over **RNDIS** in Windows and over **usbnet** in Linux.

https://en.wikipedia.org/wiki/Ethernet_over_USB

WindowsOS



LinuxOS

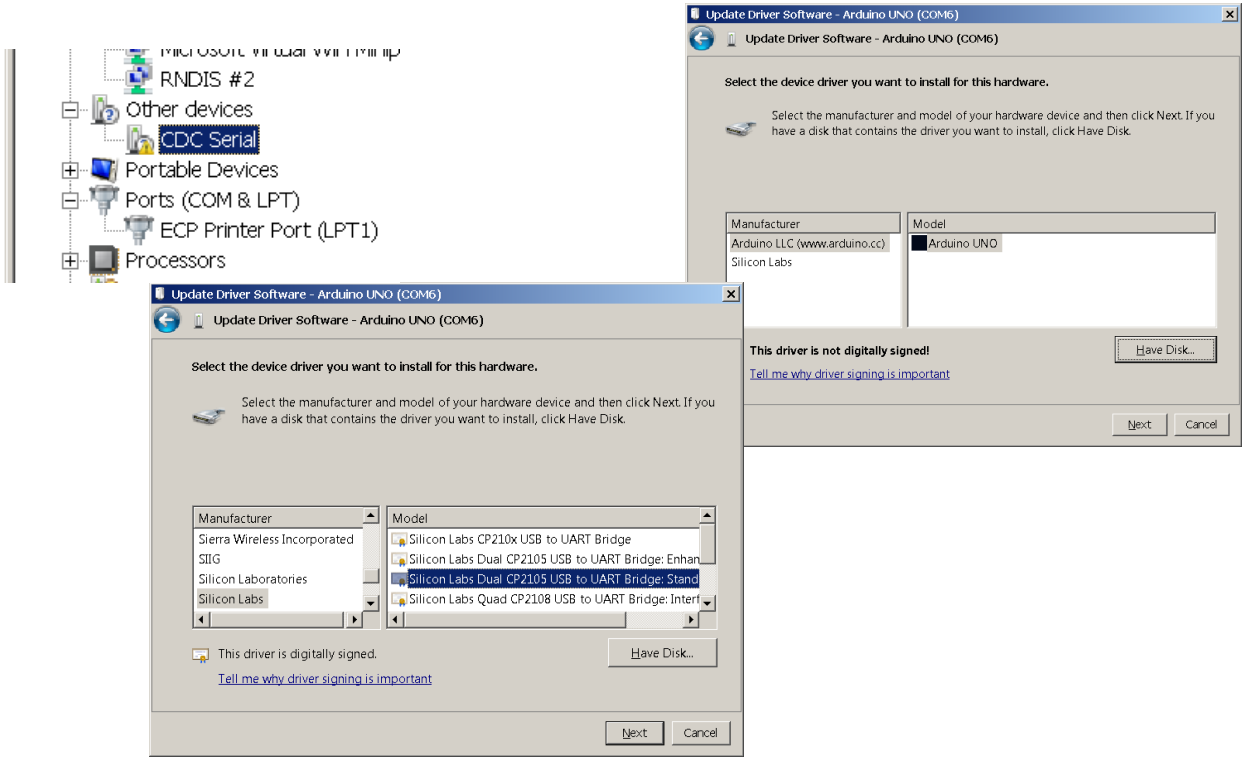
Linux doesn't need a specific driver installation to make USB connection work properly. Everything should work out-of-the-box

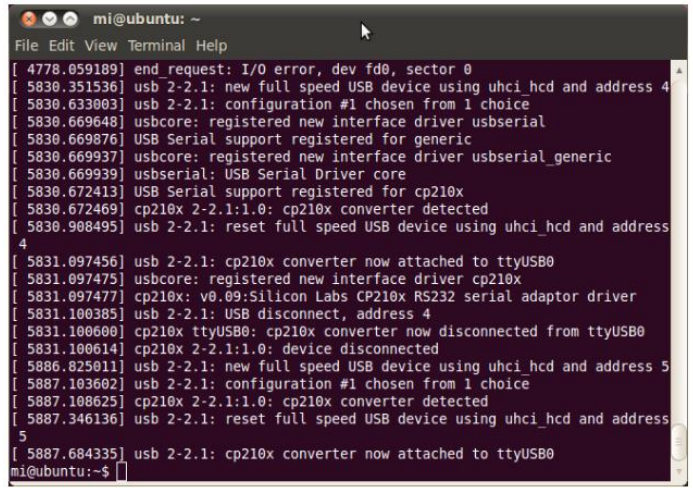
Headless UDOO – network available or emulated/serial

- For virtual serial over USB you will need to install a driver in WindowsOS:

http://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

In LinuxOS you will find it as *cpc210x converter*.



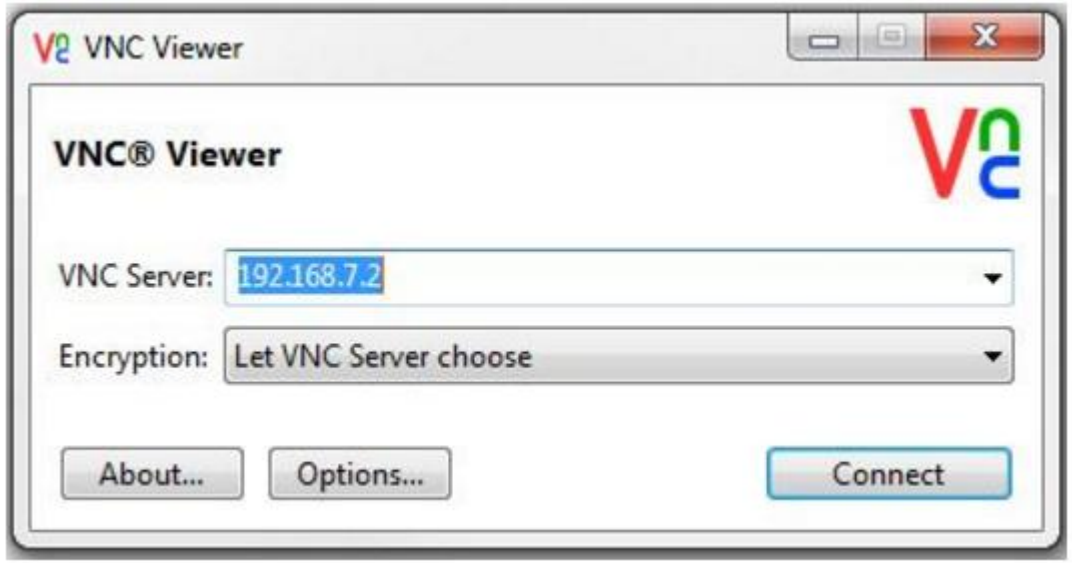


Headless UDOO - network available or emulated/serial

VNC connection

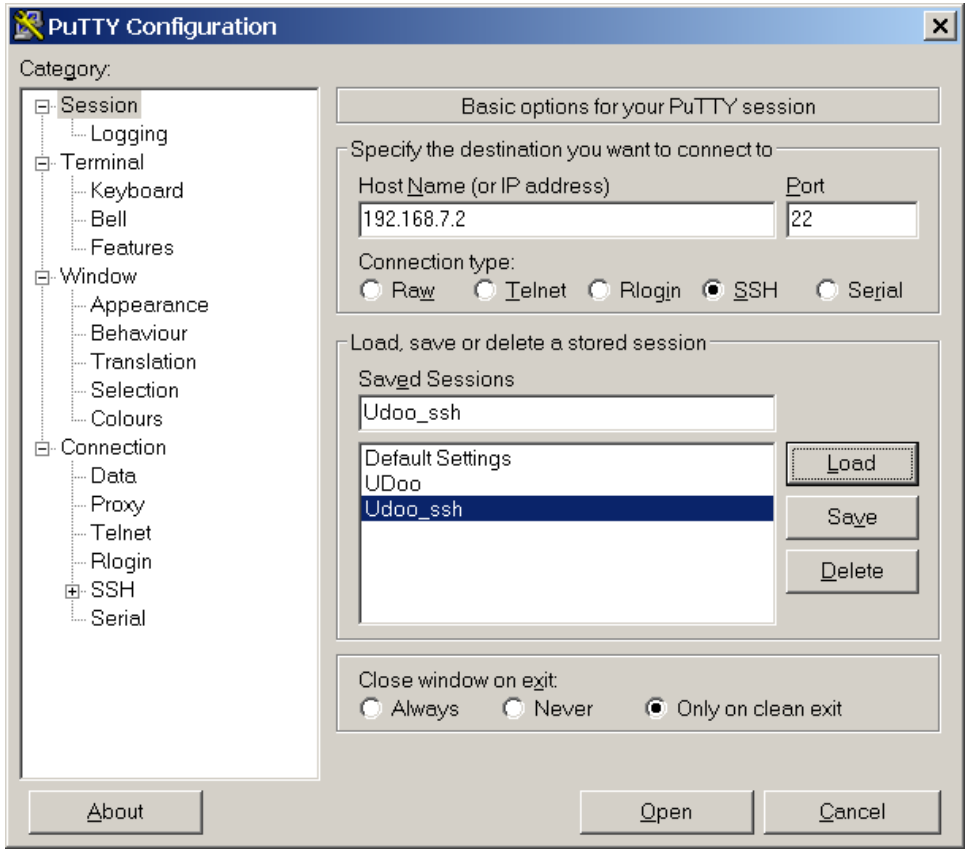
- Use VNC viewer to connect to remote UDOO

https://www.realvnc.com/download/viewer/



SSH connection

Use putty.exe to connect over SSH to remote UDOO



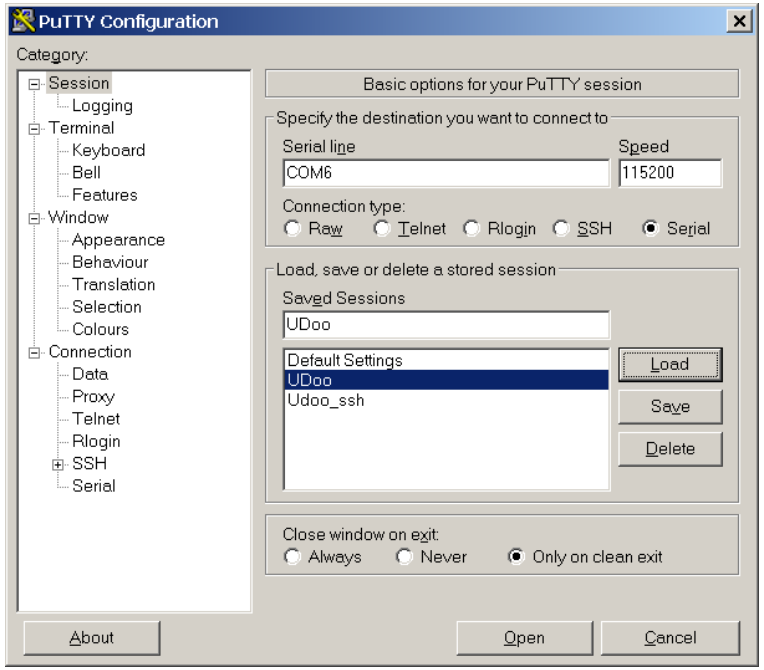
Headless UDOO – network available or emulated/serial

Serial connection

This is provided as a virtual serial over the USB connection.

It is available only after UDOO boot process

Use putty.exe to connect using serial to UDOO

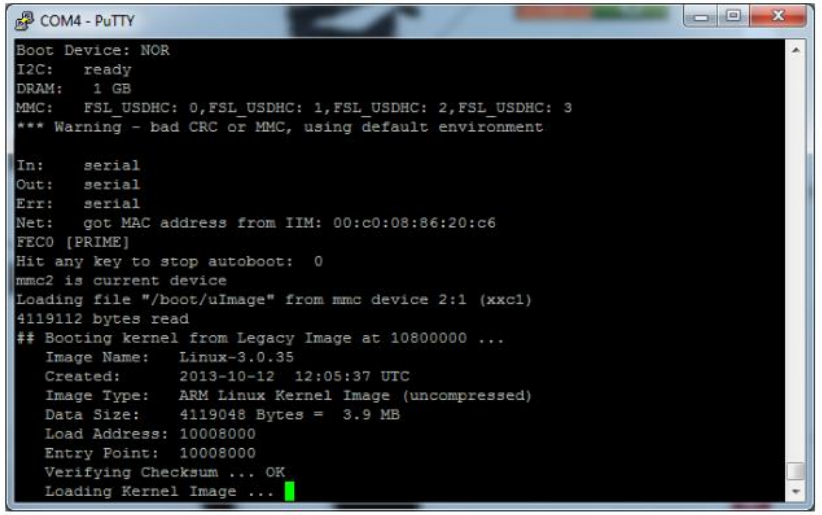


Debug serial connection

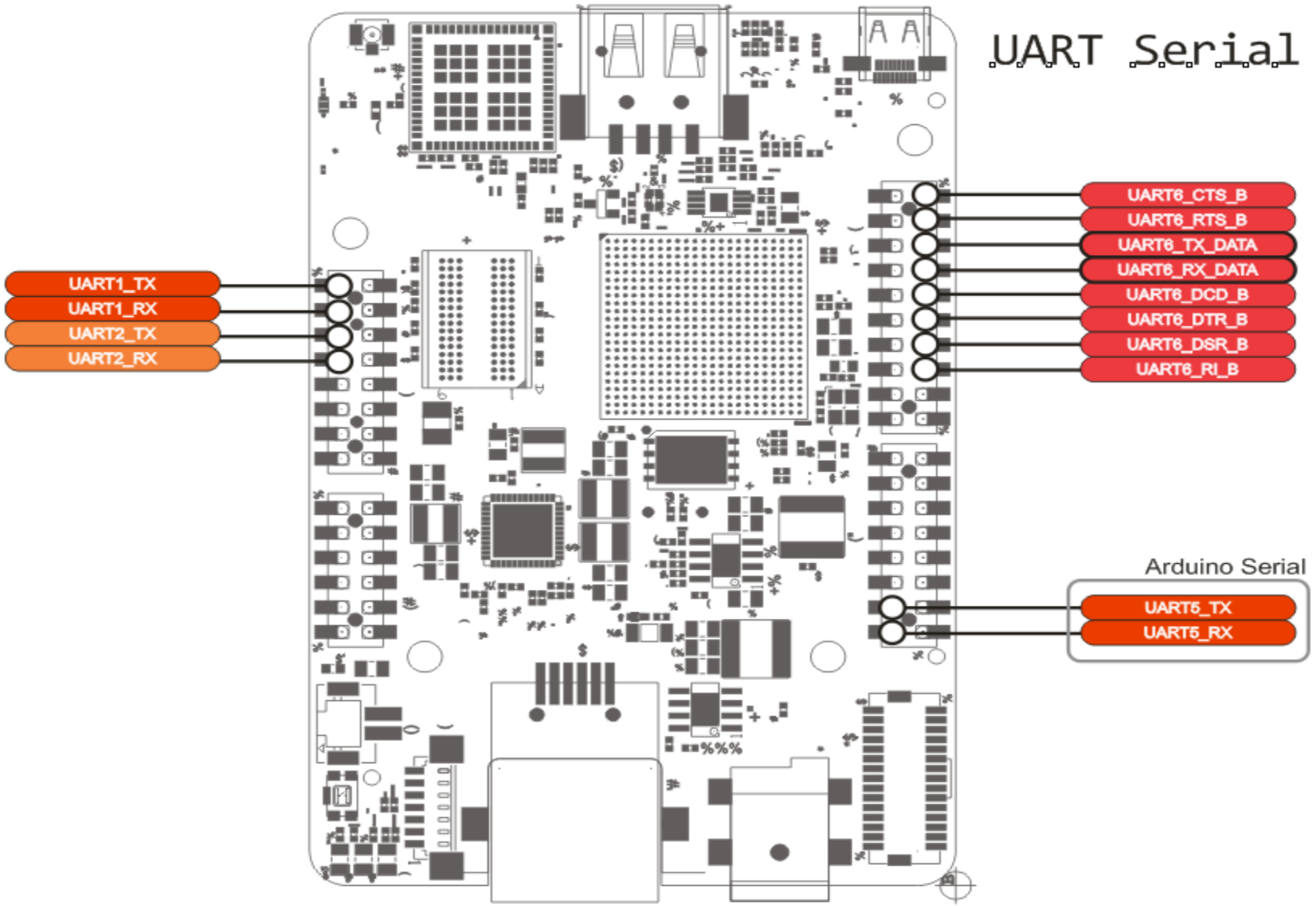
This is available immediately the UDOO board is powered up.

It provide view of the booting process.

Use putty.exe to connect using serial to UDOO



Headless UDOO – Reference UART1 as debug console



Boot process

When the board is powered on, the CPU executes code in its internal ROM, loading the first sectors of the SD card. In this way the U-Boot boot-loader is loaded and executed.

The boot-loader

This thin layer of software takes care of initialize some registers, devices (like the PMIC) and RAM time settings. It is composed of two stages, the first is called SPL (secondary program loader) which initializes several things:

- arch_cpu_init() initializes some registers, the watchdog, the DMA, etc;
- ccgr_init initializes CCGR registers in the CCM (Clock Controller Module);
- board_early_init_f initializes the M4 core and the pads of the UART1;
- timer_init initializes CPU timers and clock sources;
- preloader_console_init initializes serial port communications and prints the message "*U-Boot SPL 2015.04-00267-gd781468 (Dec 16 2015 - 14:44:56)*";
- spl_dram_init sets board-specific DRAM configuration (UDOO Neo Basic has 512MB of RAM and different timings);
- memset zeros BSS memory;
- board_init_r continues the boot, loading the second stage of the boot-loader.

In the second stage, more devices and registers are initialized. I2C buses, LVDS, Ethernet, Wireless and motions sensors pads are initialized. The PFUZE3000 power regulator is setup and MMC is initialized so files can be read from it.

Boot process

Linux kernel boot

The last step is to load the Linux kernel zImage and the device tree file, both from the /boot partition:

```
reading /zImage
4376112 bytes read in 232 ms (18 MiB/s)
Booting from mmc ...
reading dts-overlay/imx6sx-udoo-neo-full-hdmi-m4.dtb
45210 bytes read in 35 ms (1.2 MiB/s)
Kernel image @ 0x80800000 [ 0x000000 - 0x42c630 ]
## Flattened Device Tree blob at 83000000
   Booting using the fdt blob at 0x83000000
   Using Device Tree in place at 83000000, end 8300e099
Switched to ldo_bypass mode!

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 3.14.56-udooneo-01989-.....
```

Questions?