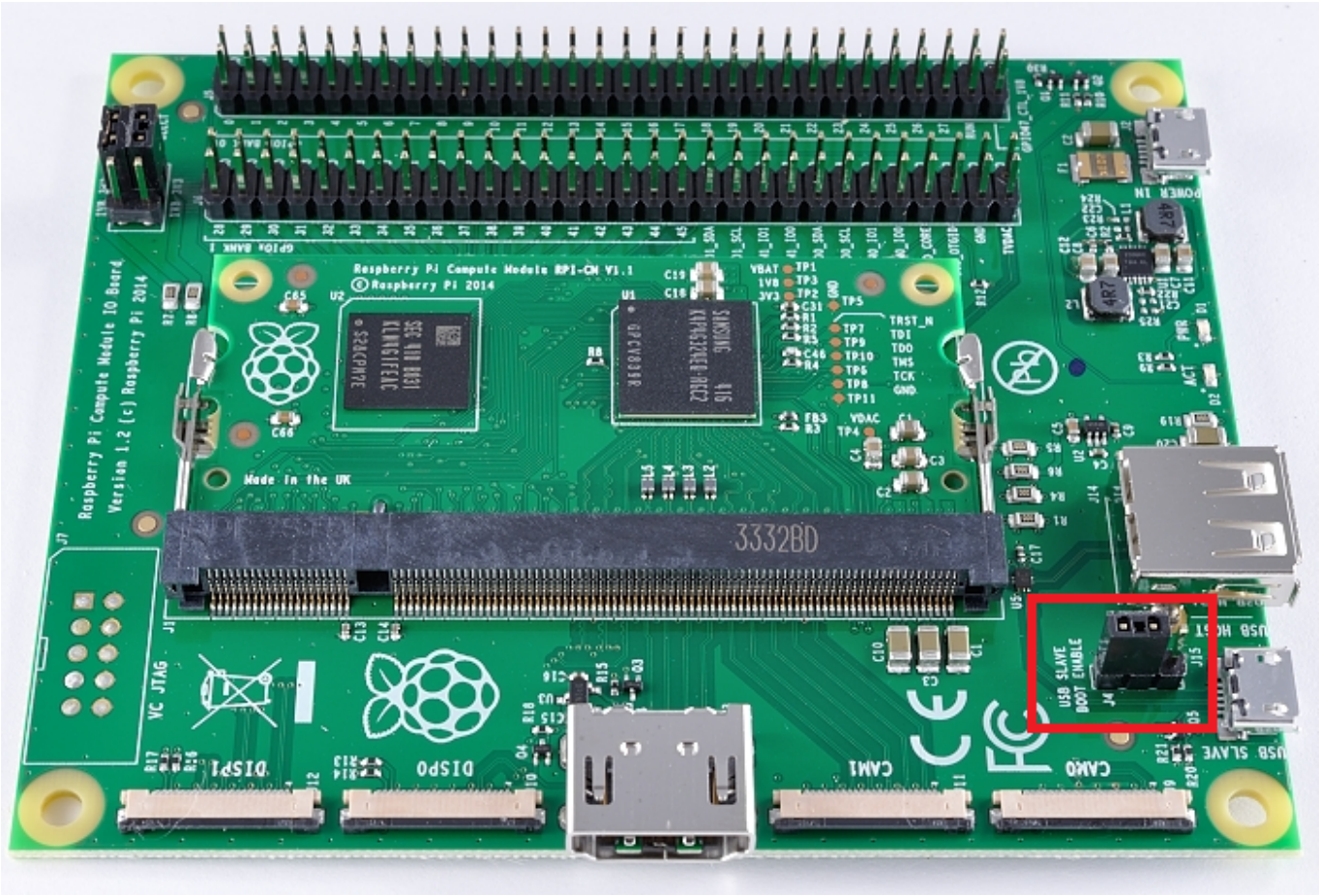
Flashing Modberry with Raspberry Development Kit

Core of Modberry is Raspberry Compute Module which is equipped with an on-board eMMC device connected to the primary SD card interface. This guide explains how to write data to the eMMC storage using a Compute Module IO Board.

As for time being there is no way to flash custom Linux onto Modberry **without** access to Raspberry Development Kit.

You need Raspberry Development Kit, which you can buy from retailer of your choice.
It looks like this:



The red square marks the pins, that you will need later on in this guide.

First steps

- First, you have to pry the top cover
- Secondly, you need to carefully remove the processor board from the slot - it's right behind the ports

Please be mindful, that opening your device voids your warranty.

Steps to flash Modberry after taking out the processor module

You need a host Linux system; a Raspberry Pi will do. A Mac will not, there is a bug in the BCM2835 bootloader which means we return slightly the wrong information, Windows and Linux don't care and carry on regardless (it's completely benign) but MacOS drops the packet

On your Compute Module IO Board:

Make sure that J4 (USB SLAVE BOOT ENABLE)(look at the picture) is set to the 'EN' position.

On your host system:

Git may produce an error if the date is not set correctly, so on a Raspberry Pi enter the following:

```
sudo date MMDDhhmm
```

where MM is month, DD day and hh mm hours and minutes respectively.

Clone the usbboot tool repository and install libusb:

```
git clone --depth=1 https://github.com/raspberrypi/tools
cd tools/usbboot
sudo apt-get install libusb-1.0-0-dev
```

Build the usbboot tool:

```
make
sudo make install
Run the usbboot tool and it will wait for a connection:
sudo rpiboot
```

Now plug the host machine into the Compute Module IO Board USB slave port (J15) and power on the CMIO board. The usbboot tool will discover the Compute Module and send boot code to allow access to the eMMC. Once complete you will see a new device appear; this is commonly /dev/sda but it could be another location such as /dev/sdb, so check in /dev/ before running rpiboot so you can see what changes.

You now need to write a raw OS image (disk image) to the device. Note the following command may take some time to complete, depending on the size of the image:

```
sudo dd if=raw_os_image_of_your_choice.img of=/dev/sda bs=4MiB
```

Once the image has been written, unplug and re-plug the USB; you should see 2 partitions appear (for

Raspian) in /dev. In total you should see something similar to this:

/dev/sda ← Device /dev/sda1 ← First partition (FAT) /dev/sda2 ← Second partition (Linux filesystem)
The /dev/sda1 and /dev/sda2 partitions can now be mounted normally.

Make sure J4 (USB SLAVE BOOT ENABLE) is set to the disabled position and/or nothing is plugged into the USB slave port. Power cycling the IO board should now result in the Compute Module booting from eMMC.