How to boot up a Morello Board with CheriBSD

Carlos Molina-Jiménez[1]

Department of Computer Science and
Technology, University of Cambridge
carlos.molina@cl.cam.ac.uk, www.cl.cam.ac.uk/~cm770/
3 Sep 2024

Resumen This document explains how to boot a brand new Morello Board with the cheriBSD operating system. The first version (version 26 Aug 2022) of this document was originally written for the deployment of the cheribsd-memstick-arm64-aarch64c-22.05p1.img cheriBSD version. I have upgraded my morello boards to the latest cheriBSD version, that is, cheribsd-memstick-arm64-aarch64c-22.12.img and can tell that the procedure remains the same. So the document remains mainly the same, except for small improvements such as the inclusion of figures in Section 6 that explain *minicom* configuration.

Keywords:

1. Introduction

ARM has sent me two Morello Boards to carry out the implementation work involved in the CAMB project. See photo in Fig. 1.

The Morello Boards are brand new, in the sense that they come with no pre–installed software. For instance, they have no operating systems on their internal 250 GB SATA disks. My research work is based on CHERI capabilities, therefore, I need to install CheriBSD operating system on their local disks.

In the following sections I will explain the procedure that I have followed to set up one of the Morello Boards. I followed the same procedure to set up the second Morello Board.

The general setup procedure is explained in ARM [3] and in a video (Morello: Getting Started Guide) [2]. The setup with CheriBSD is explained in [8].

In this document I am sharing my personal experience and discussing some details that [3,2,8] omit.

2. Equipment

I have the Morello Board with all the cables as normally shipped by ARM [3] and in addition, following ARM recommendations, I have:

- Host PC: an x8664 PC running Ubunto Linux 18.04, 16GB of memory, 250GB of free disk space and monitor with HDMI input support for at least 1920x1080 at 60Hz.

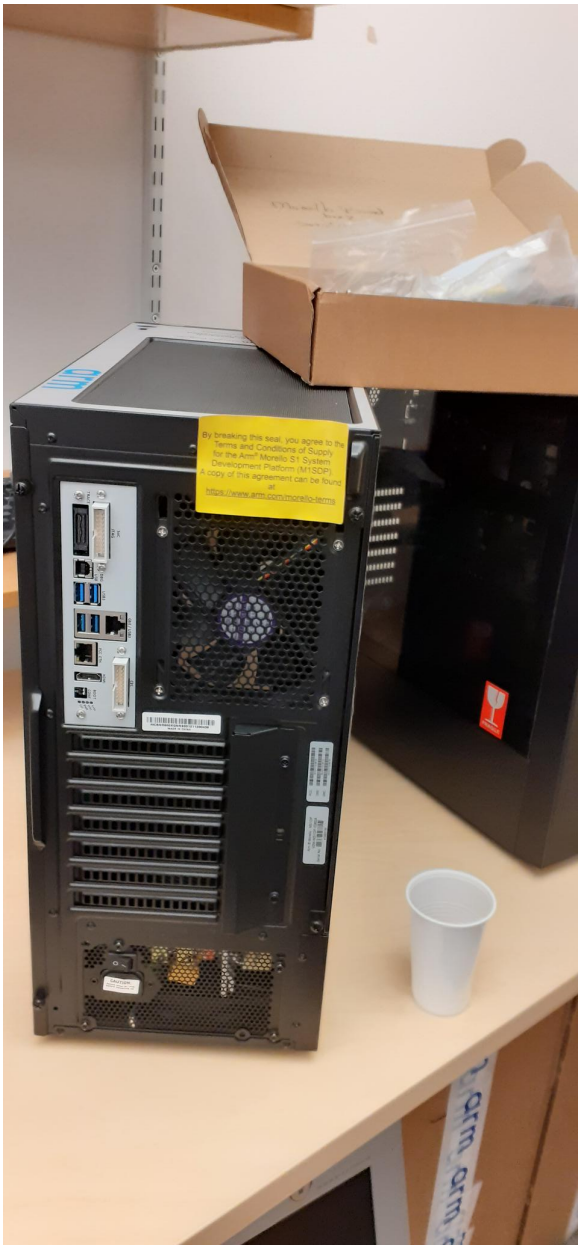

Figura 1. My two brand new Morello Boards.

- SamDisk 64GB 3.0 memory stick. ARM recommends old memory sticks of least 4GB, but new ones worked fine.

3. The main two steps

In general, the procedure involves two related but independent steps.

1. Creation of a bootable USB memory stick with a CheriBSD boot image.
2. Upgrade of the internal SDcard of the Morello Board with the latest version of the Morello Board firmware.

The order of the steps is irrelevant, that it, one can first upgrade the SDcard and then create the bootable USB memory stick or the other way around. I will start with the creation of the bootable memory stick because this step does not involve the Morello Board, it only involves the host PC.

4. Creation of a bootable USB memory stick

ARM recommends a USB memory stick of at least 4GB. I used a conventional SamDisk 64GB 3.0 memory stick and experimented with others including Phillips and Verbatim. I remark on this point because it seems that some people have experienced problems to boot Morello Boards with some memory sticks. I used several brands and all of them worked fine for me.

To create my bootable memory stick, I executed the following steps.

1. I downloaded an installable CheriBSD cheribsd-memstick-arm64-aarch64c-22.05p1.img.xz image for the Morello Board from [5] [1].
   - Another alternative is to create an image as explained in [8]. However, to start with a ready to use image is a better approach, the creation of an image might take between 2 and 3 hours of compilation which involves downloads of hundreds of dependencies. This is a time consuming procedure that might crash several times.
2. I uncompressed the downloaded image.

```
$ ls -l
-rw-r--r--@ 1 carlosmolina  staff  426130916 27 Aug 13:26
cheribsd-memstick-arm64-aarch64c-22.05p1.img.xz

bash-3.2$ gunzip cheribsd-memstick-arm64-aarch64c-22.05p1.img.xz

bash-3.2$ ls -l
-rw-r--r--  1 carlosmolina  staff  814955008 27 Aug 13:26
cheribsd-memstick-arm64-aarch64c-22.05p1.img
```

[1] Currently only the latest image (cheribsd-memstick-arm64-aarch64c-22.12.img.xz) is available online [4].

3. I plugged a 64GB USB memory stick to a USB port of my Ubunto PC and run the Linux *lsblk* command to identify the device driver that Ubunto used to attach the USB memory stick.

```
$ lsblk
 ...
sdb      8:16   0 465.8G  0 disk
  sdb1   8:17   0 465.8G  0 part /camb
sdc      8:32   1  58.2G  0 disk
sr0     11:0    1  1024M  0 rom
```

   The *lsblk* output shows that there is a storage device of 58.2G connected to *sdc*. That is the USB memory stick. If not sure, what device is the USB memory stick, you can unplug it and run *lsblk* again, the *sdc* device will not be shown in the output. Plug it and run *lsblk* again and it will be shown in the output again. Make sure you do not specify the wrong device to the *dd* command output ($of = /dev/...$) because *dd* will write on top of the original content.
4. I copied the uncompressed image to the memory stick using the standard *dd* Linux command

```
$ sudo dd if=cheribsd-memstick-arm64-aarch64c-22.05p1.img
          of=/dev/sdc bs=1048576

[sudo] password for cm770:
777+1 records in
777+1 records out
814955008 bytes (815 MB, 777 MiB) copied, 46.1821 s, 17.6 MB/s
```

   In this example, I used a Verbatim USB 3.2 Gen 1, V3 Drive, 64GB. I experimented also with several USBs (SanDisk USB 3.0, 64GB, Philips USB 2.0, 32GB and Kingston USB 3.2, 64GB) —all worked fine for me. The only difference is the time it takes the *dd* command to copy the image to the USB memory stick which is always less than a minute.
5. The USB bootable memory stick is now ready to boot the Morello Board up.

5. Upgrade of internal SDcard of the Morello Board with the latest firmware

Morello boards are shipped with SDcards flashed with an old firmware. ARM strongly recommends to delete it and replace it with the latest version of Morello Board firmware.

1. I downloaded the latest version of the Morello Board from [1].

- Alternatively, as explained in [3] one can compile locally version of the firmware from the sources which are available from Git. However, to start with, it is sensible to download a ready to use firmware.

2. I plugged one end of the HDMI cable into the HDMI port on the Morello Board and the other end into the monitor of the Ubunto PC.
3. I plugged one end of the USB–B cable into the debug (DBG) port on the Morello Board and the other end to a USB port of the Ubunto PC. This cable is used to open two logical connections:
   - ttyUSB0: one to the Motherboard Configuration Controller (MCC).
   - ttyUSB2: one to the Application Processor (AP).

   See Section 6.
4. I turned on the Morello Board.
   - The internal SDcard of the Morello Board is taken as a conventional storage divice by the Ubunto PC. Therefore it is accessible through the Ubunto graphical interface. I clicked on its icon to open it. It contains some files and folders.

     ```
     bash-3.2$ ls -la
     total 48
     drwxr-xr-x  10 carlosmolina  staff   320  9 Jun 00:03 ./
     drwxr-xr-x   5 carlosmolina  staff   160 28 Jul 20:32 ../
     -rw-r--r--@  1 carlosmolina  staff  8196 24 Jul 23:06 .DS_Store
     drwxr-xr-x   4 carlosmolina  staff   128  8 Jun 23:39 .Trash-3319/
     drwxr-xr-x   6 carlosmolina  staff   192 17 Jul 02:02 LICENSES/
     -rw-r--r--@  1 carlosmolina  staff   935  1 Jan  2016 LOG.TXT
     drwxr-xr-x   4 carlosmolina  staff   128 17 Jul 02:01 MB/
     drwxr-xr-x   5 carlosmolina  staff   160 14 Feb  2022 SOFTWARE/
     -rw-r--r--@  1 carlosmolina  staff  1203  2 Feb  2022 config.txt
     -rw-r--r--@  1 carlosmolina  staff   215  2 Feb  2022 ee0364b.txt
     ```

     These files and folders are the Morello Board firmware that ARM loaded at manufacturing time. This firmware is very likely to be outdated, therefore, ARM strictly recommend to delete them and flash the SDcard with the latest version.
     In principle, I could have used Ubunto's graphical interface to treat the SDcard as a conventional memory stick: open it, move all the files and folder to the bin and copy the new ones into the SDcard. However, I opted to follow the directions from the ARM documents, therefore, I used Linux online commands, including, *mount* and *umount*.
5. I used the *lsblk* Linux command to identify the device that Ubunto uses to interact with the SDcard.

   ```
   $ lsblk
   NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
   loop0    7:0    0    47M  1 loop /snap/snapd/16292
   ...
   loop5    7:5    0    47M  1 loop /snap/snapd/16010
   ```

```
sda      8:0    0 223.6G  0 disk
  sda1   8:1    0  37.3G  0 part
  sda2   8:2    0  37.3G  0 part /
  sda3   8:3    0 149.1G  0 part /local/scratch
sdb      8:16   0 465.8G  0 disk
  sdb1   8:17   0 465.8G  0 part /camb
sdc      8:32   1   1.9G  0 disk
  sdc1   8:33   1   1.9G  0 part /media/cm770/M1SDP
sr0     11:0    1  1024M  0 rom
```

The SDcard corresponds to the *sdc*1 Ubunto device. I could identify it by its size. The Morello Boards that ARM shipped to me come with SDcards of 2GB. Therefore, */dev/sdc*1 corresponds to the SDcard.

6. I mounted the SDcard into a folder of the Ubunto file system, for example, into */mnt* (any empty folder can be used) and deleted all the files and folders:

```
$ sudo mount /dev/sdc1 /mnt
[sudo] password for cm770:
cm770@pursuit2:~$ cd /mnt

$ ls -l
total 224
-rwxr-xr-x 1 root root  1199 Aug  2 20:49 config.txt
-rwxr-xr-x 1 root root   215 Aug  2 20:49 ee0364b.txt
drwxr-xr-x 2 root root 32768 Aug  2 20:49 LIB
drwxr-xr-x 5 root root 32768 Aug  2 20:49 LICENSES
-rwxr-xr-x 1 root root  1104 Jan  1  2016 LOG.TXT
drwxr-xr-x 3 root root 32768 Aug  2 20:49 MB
drwxr-xr-x 2 root root 32768 Aug  2 20:49 SOFTWARE
```

7. I deleted the old version of the firmware from the SDcard.

```
 $ rm -R * /* delete all files and folders */
```

8. I copied the version of the firmware downloaded from [1] to the SDcard.

```
$ pwd
/camb/board-firmware-morello-mainline

$ ls -l
total 24
-rw-r--r-- 1 cm770 cm770 1199 May 20 09:35 config.txt
-rw-r--r-- 1 cm770 cm770  215 May 20 09:35 ee0364b.txt
drwxr-xr-x 2 cm770 cm770 4096 Jul 30 01:29 LIB
drwxr-xr-x 5 cm770 cm770 4096 Jul 30 01:29 LICENSES
drwxr-xr-x 3 cm770 cm770 4096 Jul 30 01:29 MB
```

```
drwxr-xr-x 2 cm770 cm770 4096 Jul 30 01:29 SOFTWARE
$

$ cp -R board-firmware-morello-mainline/* /mnt/

$ ls -l /mnt
total 192
-rw-r--r-- 1 cm770 cm770  1199 Jul 25 00:55 config.txt
-rw-r--r-- 1 cm770 cm770   215 Jul 25 00:55 ee0364b.txt
drwxr-xr-x 2 cm770 cm770 32768 Jul 25 00:55 LIB
drwxr-xr-x 5 cm770 cm770 32768 Jul 25 00:55 LICENSES
drwxr-xr-x 3 cm770 cm770 32768 Jul 25 00:56 MB
drwxr-xr-x 2 cm770 cm770 32768 Jul 25 00:56 SOFTWARE

$ sync /* advisable to make sure that the copy
       /* command is completed
```

9. I umounted the SDcard from the Ubunto file system

   ```
   $ sudo umount /dev/sdc1
   ```

10. I have the SDcard ready now to be used as firmware to boot up the Morello Board, for example, with the USB bootable memory stick created in Section 4.

6. Booting up with the USB memory stick

The actual booting up process assumes that I have successfully:

- Upgraded the internal SDcard of the Morello board as detailed in Section 5.
- Created a bootable USB memory stick as detailed in Section 4.

1. I turned off the Morello Board.
2. I turned on the Morello Board.
3. I opened two serial connections between the Ubunto PC (my host PC) and the Morello Board. Each from an x-term running a *bash* shell.
   - I used *minicom* to setup (through the debug cable) the two logical serial connections needed:
     - ttyUSB0: one to the Motherboard Configuration Controller (MCC). It connects the the Ubunto PC to the MCC of the morello board.
     - ttyUSB2: one to the Application Processor (AP). It connects the Ubunto PC to the AP of the morello board.

   Minicom configuration is explained [6]. The directions to configure *minicom* within the context of the Morello Boards are provided in page 17 of [3]. Correct configuration of the logical ports is crucial. Fig. 2 to 11 show the correct configuration parameters for the AP (also called the Morello System Console or MSC) logical connection. The parameter for the AP

logical connections are the same. I am showing the configuration of the AP logical connection because this more sensitive that the other. For example, if the *Hardware flow control* is set to *yes*, the connection will fails to accept input from the xterminal to the AP.
To start, I typed:

```
cm770@pursuit2-$ sudo minicom -s
[sudo] password for cm770: cm770password
```

Fig. 2 to Fig. 5 show how to input the configuration parameters and how to save the configuration.
Fig. 2 shows minicom menu.

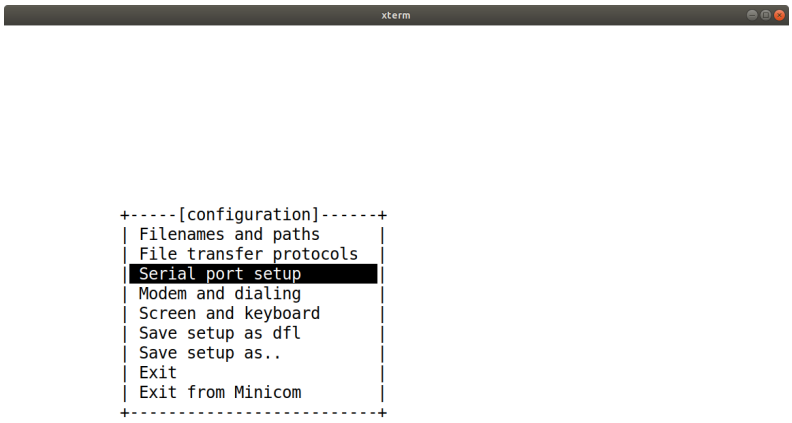

Figura 2. sudo *minicom -s* takes to a setup menu.

Fig. 3 shows minicom menu for port configuration.
Fig. 4 shows correct configuration parameters for the Unbunto–AP connection.
Fig. 5 shows how to save the configuration parameters under the `ttyUSB` name. The name is any string.
Fig. 6 shows that the configuration parameters for the Ubuntu–AP connection have been saved under the mane `ttyUSB2`.
Exit from the menu takes to the screen shown in Fig.7. The logical connection between the Ubunto computer and the AP of the Morello

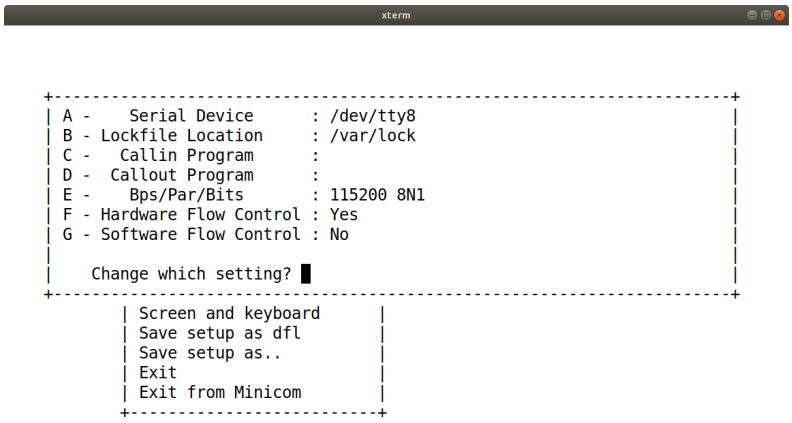

Figura 3. Set up menu for a serial port.

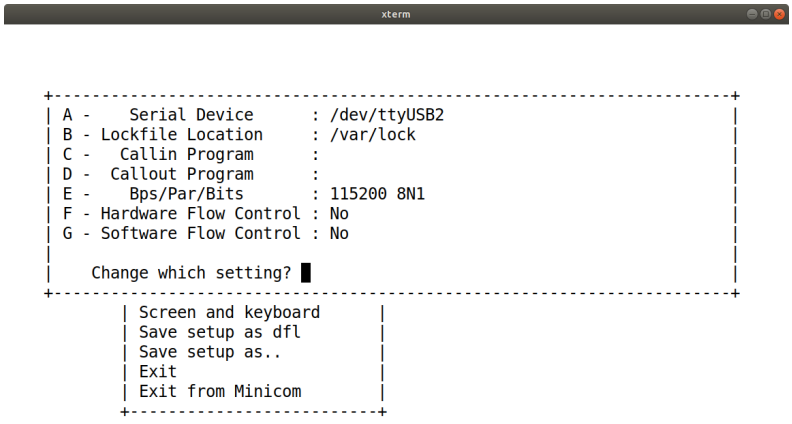

Figura 4. Correct configuration parameters for the Ubunto-AP connection.

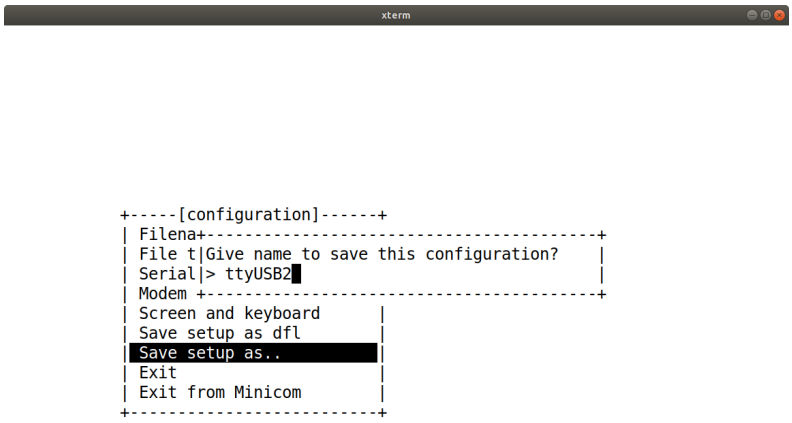

Figura 5. Save configuration parameters under the *ttyUSB2* name, to re-use them.

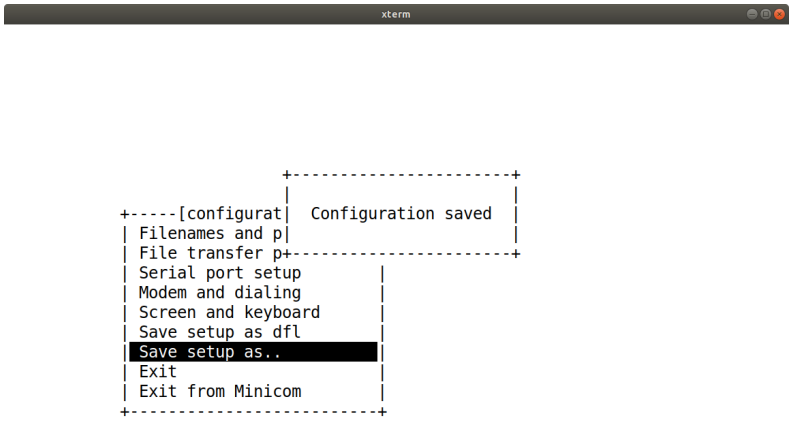

Figura 6. Configuration parameters successfully saved.

Board is ready. It will be activated later when the Morello Board is rebooted through the Ubunto–MCC connection.

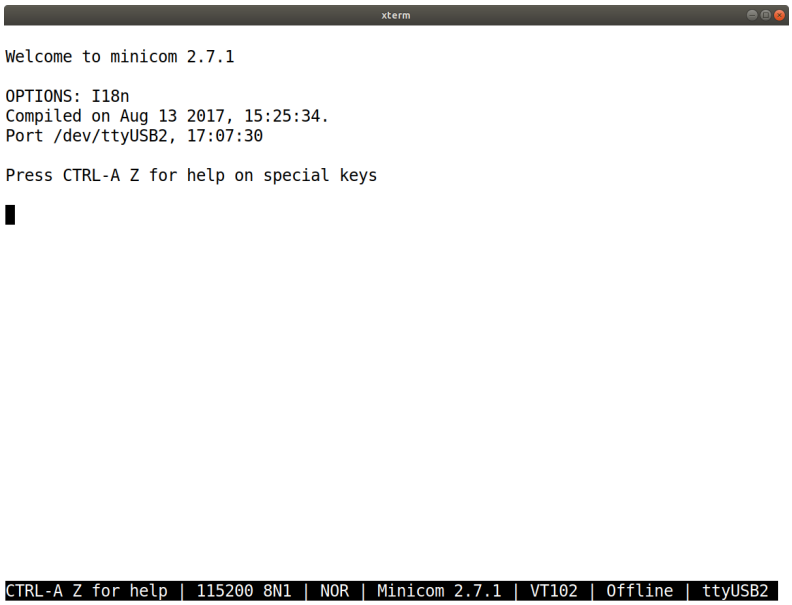

Figura 7. Communication line to Morello System Console is ready.

Notice that, in response to the save instruction, of Fig. 5, minicom creates the *minirc.ttyUSB2* file shown in Fig. 8.

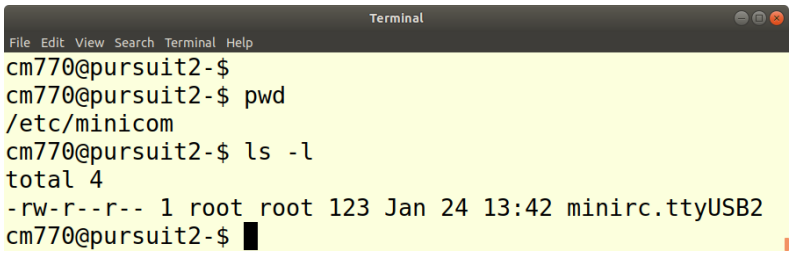

Figura 8. Minicom creates the /etc/minirc.ttyUSB2 file.

To exit minicom at the end of the session one can press `Ctrl+A`, release the keywords and then press Z. This takes to the screen shown in Fig. 9. Press X to `eXit and reset`.
In the screen of Fig. 10 press `yes`.

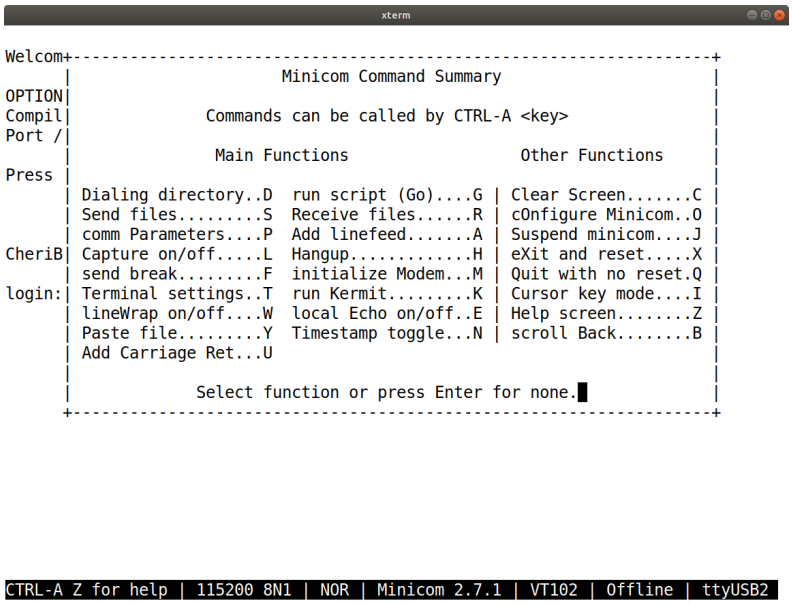

Figura 9. To exit from minicom select *eXit and reset*.

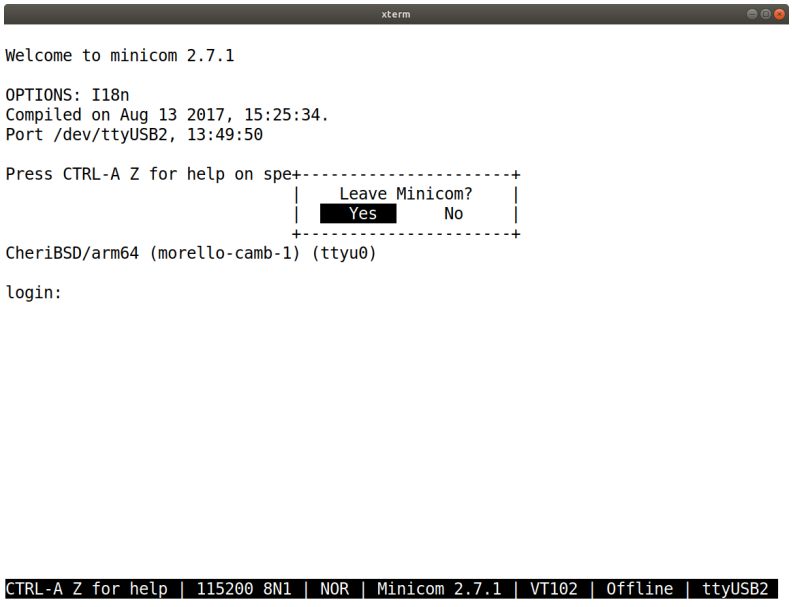

Figura 10. Exit from minicom: press *Yes*.

Since we saved the configuration parameters under the `ttyUSB2` name, one can repetitively open communication session at any time between the Ubunto computer and the AP of the Morello Board in the following manner:

```
cm770@pursuit2-$ sudo minicom ttyUSB2
```

For example, Fig. 11 shows the output of the AP on the screen of the Ubunto computer after the installation of cheriBSD, The information shown on the screen will become clear after the installation of the cheriBSD operating system (completion of this Section).

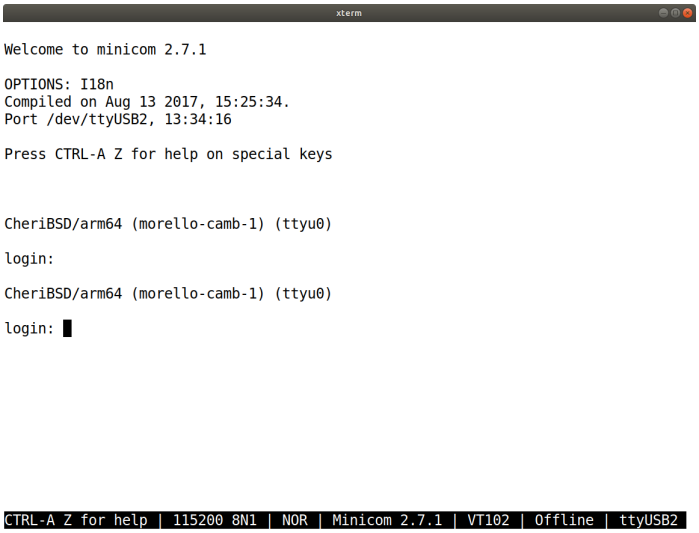

Figura 11. Output of the AP of the Morello Board sent through /dev/ttyUSB2 to Ubuntu.

- I used an x–term to open the connections to match the terminal type specified to FreeBDS later at installation time. I have observed that a terminal type mismatch causes blurred output lines (nearly unreadable) from the configuration of the FreeBSD.

```
Welcome to FreeBSD!

Please choose the appropriate
terminal type for your system.
Common console types are:
   ansi     Standard ANSI terminal
   vt100    VT100 or compatible terminal
   xterm    xterm terminal emulator (or compatible)
 ...
Console type [xterm]:
```

4. I plugged the bootable USB memory stick into one of the four USB ports located on the back of the Morello Board. Any of the four USB ports will work.
5. I have the Ubuntu-MCC connection ready, that is, a) the physical connection through the DBG USB cable from the Morello Board connected to USB port of Ubunto and b) the logical connection through minicom using ttyUSB0.
6. I turned the Morello Board on.
   - Some output were displayed on the MCC console.
7. I typed *reboot* from the MCC console.
8. I followed and responded to the outputs from the AP console, precisely, to the questions posed by the installation process of the FreeBSD.
   - The installation process asks for the network parameters to logically connect the Morello Board to the local network, such as IP host name, address, netmask, gateway and DNS servers. Get them from your network administrator and have them ready.
   - Several time during the installation process, one is presented with menus to select and type configuration parameters located in two adjacent boxes (top and bottom) like the one shown in Fig. 12. Use the keyboard arrows (←, →, ↑, ↓) to move the cursor within a field. To move between boxes, us the tan key.

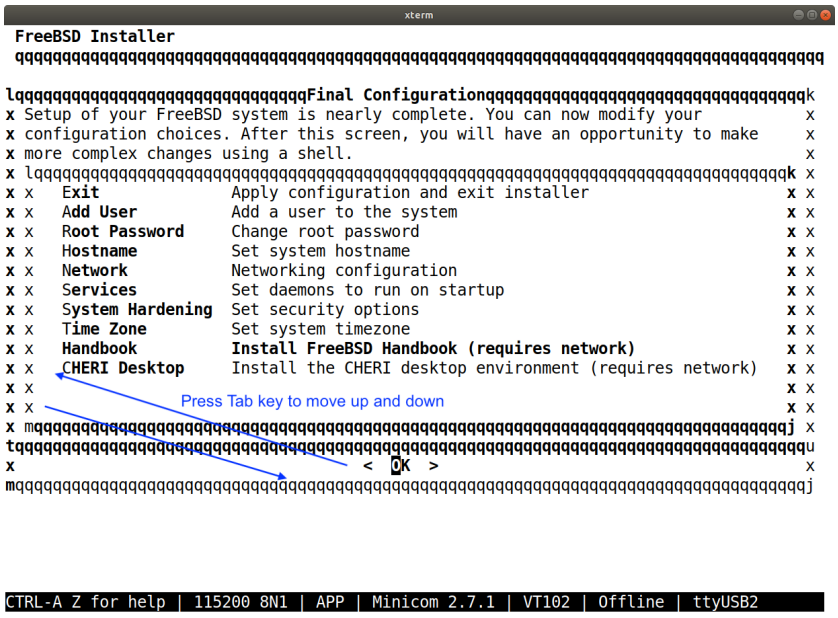

Figura 12. Example of an installation menu.

7. Conclusion and recommendation

The task is challenging if you have not configure a Morello Board before. This is an experimental hardware and therefore, the documentation is not well organised or complete. Fortunately, the ARM community forum is very keen to help [7].

Acknowledgements Carlos Molina has been supported by CAMB UKRI project (G115169)[2].

Referencias

1. ARM: Morello board-firmware. `https://git.morello-project.org/morello/board-firmware` (2022), visited on 13 Aug 2022
2. ARMlimited: Arm morello program. `https://www.arm.com/architecture/cpu/morello` (2022)
3. ARMlimited: Morello development platform and software version 1.0 getting started guide. `https://developer.arm.com/documentation/den0132/latest/` (2022), visited on 6 Jun 2022
4. Cheri team: Downloading image files. `https://ctsrd-cheri.github.io/cheribsd-getting-started/downloading/index.html` (2023), visited on 14 Jan 2023
5. CherriBSD: Cheribsd. `https://www.cheribsd.org` (2022), visited on 13 Aug 2022
6. EMAC: Getting started with minicom. `https://wiki.emacinc.com/wiki/Getting_Started_With_Minicom` (2022), visited on 26 Aug 2022
7. Support Forum: Arm community forum. `https://community.arm.com/support-forums/f/morello-forum` (2022), visited on 8 Oct 2022
8. Watson, R.N.M., Davis, B.: Getting started with cheribsd. `https://ctsrd-cheri.github.io/cheribsd-getting-started/` (2022), visited on 24 Jul 2022