

*****Default*****

**Re-look Paul's MQTT
Ultibo Australian user
ultibo-mqtt
11/07/22**

*****Default*****

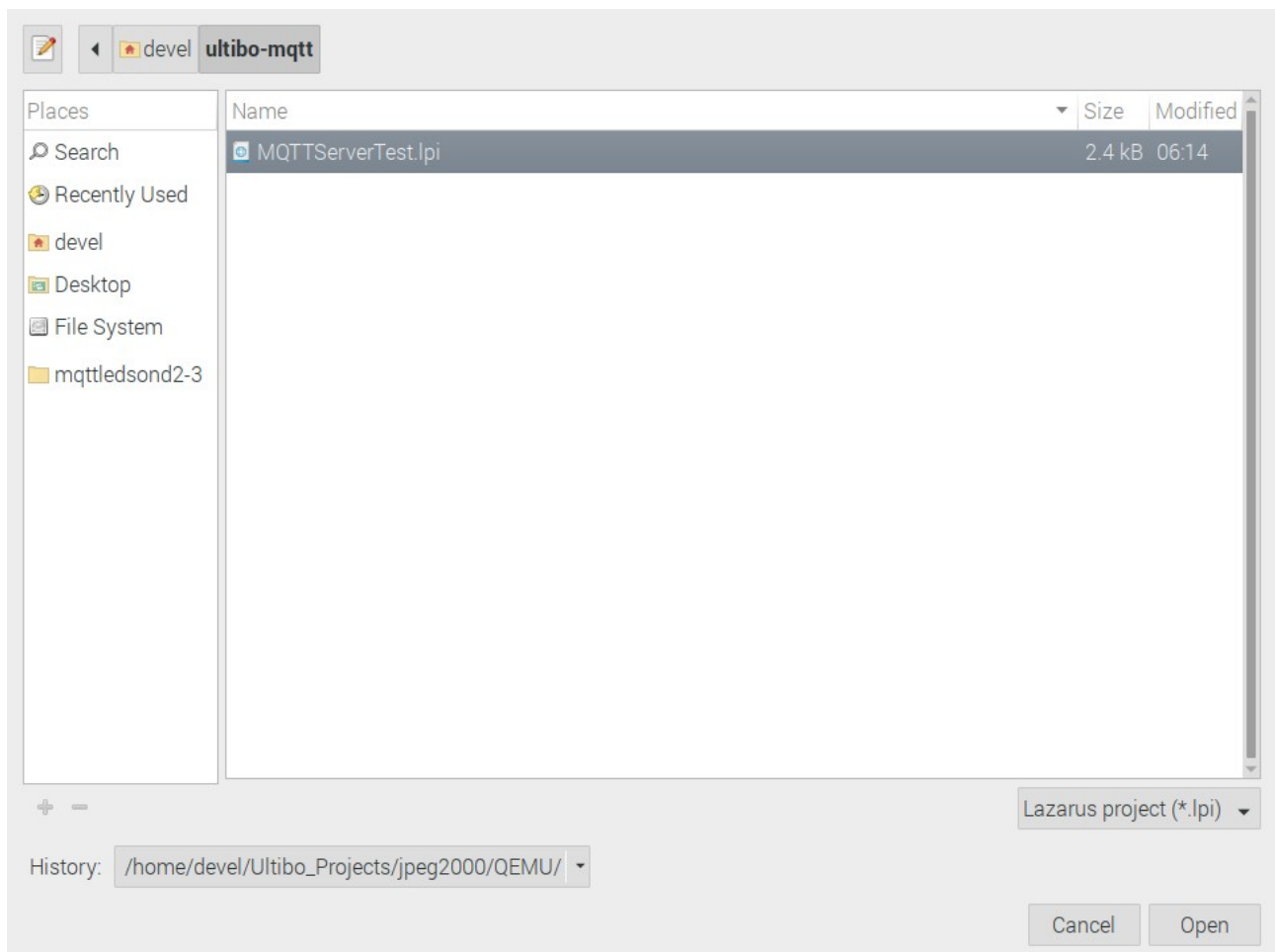
This is a re-look at ultibo-mqtt, earlier work for use with Raspberry Pi Pico W project.

Goal: To see if the code as downloaded from <https://github.com/pjde/ultibo-mqtt.git> , from an Ultibo Australian user, compiles and runs on hardware. This will be used as a learning tool for understanding MQTT in Raspberry Pi Pico W project.

<https://ultibo.org/forum/viewtopic.php?f=10&t=1427&p=9844&hilit=mqtt#p9844>

“git clone <https://github.com/develone/ultibo-mqtt.git>”

With Lazarus IDE (Ultibo-Edition)



First step is to check that code compiles. From the main Tool Bar select Run/Compile. If the code is okay the green bar will appear. At first glance of the file “MQTTServerTest.lpr” after opening “MQTTServerTest.lpi” project file, is that code was intended for a ”RaspberryPi3”. If the code is to be used on the ”RaspberryPi3” Bare Metal you will need the firmware found at “git clone https://github.com/develone/firmwar_for_ultibo.git”

Ultibo Bare Metal only requires a few file.

bootcode.bin

fixup.dat

fixup4.dat

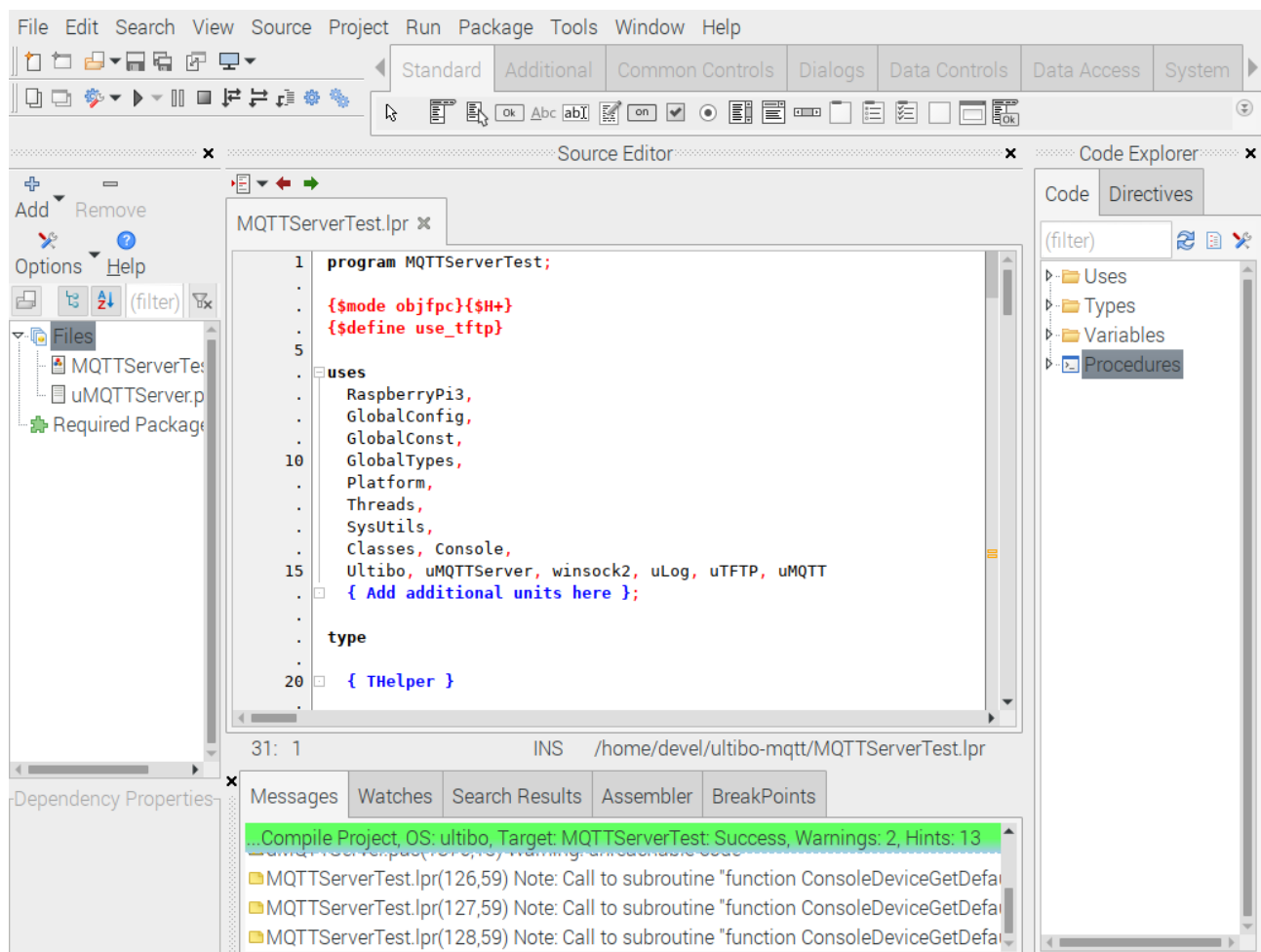
start.elf

start4.elf

config.txt

The files above provide a way for the kernel to boot and have access to the hardware.

Most of code is contained in file kernel7.img.



File Edit View Sort Go Tools

/home/devel/ultibo-mqtt

Name	Size	Modified
backup		11/07/2022 07:45
lib		11/07/2022 06:24
kernel7.img	2.5 MiB	11/07/2022 07:45
MQTTServerTest.elf	3.6 MiB	11/07/2022 07:45
MQTTServerTest.lpi	2.3 KiB	11/07/2022 06:14
MQTTServerTest.lpr	5.2 KiB	11/07/2022 07:45
MQTTServerTest.lps	729 bytes	11/07/2022 07:45
README.md	41 bytes	11/07/2022 06:14
uLog.pas	421 bytes	11/07/2022 06:14
uMQTT.pas	34.7 KiB	11/07/2022 06:14
uMQTTServer.pas	75.8 KiB	11/07/2022 06:14
uTFTP.pas	15.5 KiB	11/07/2022 06:14

"kernel7.img" (2.5 MiB) Raw disk image

Free space: 132.4 GiB (Total: 291.4 GiB)

mount the micro sd

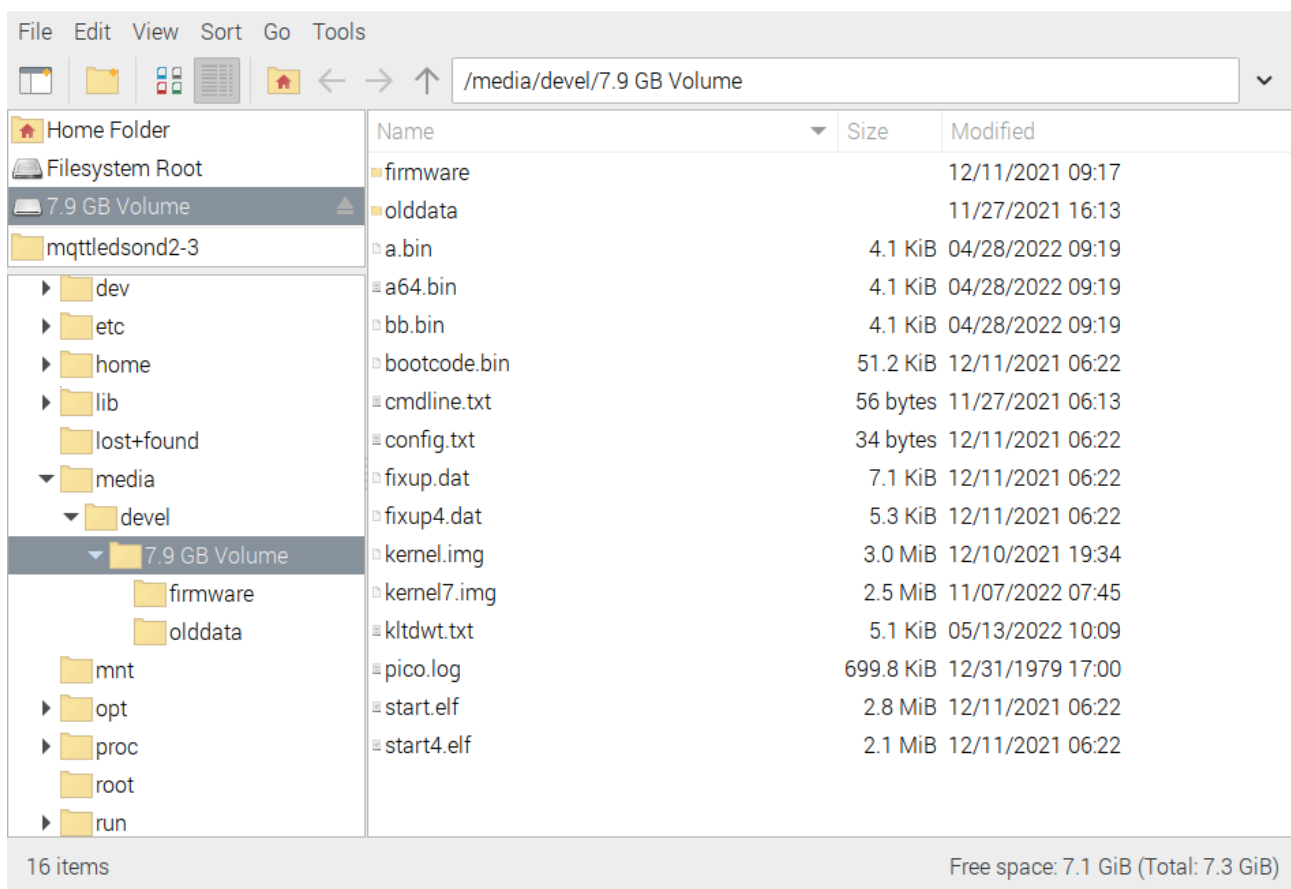
File Edit View Sort Go Tools

/media/devel/7.9 GB Volume

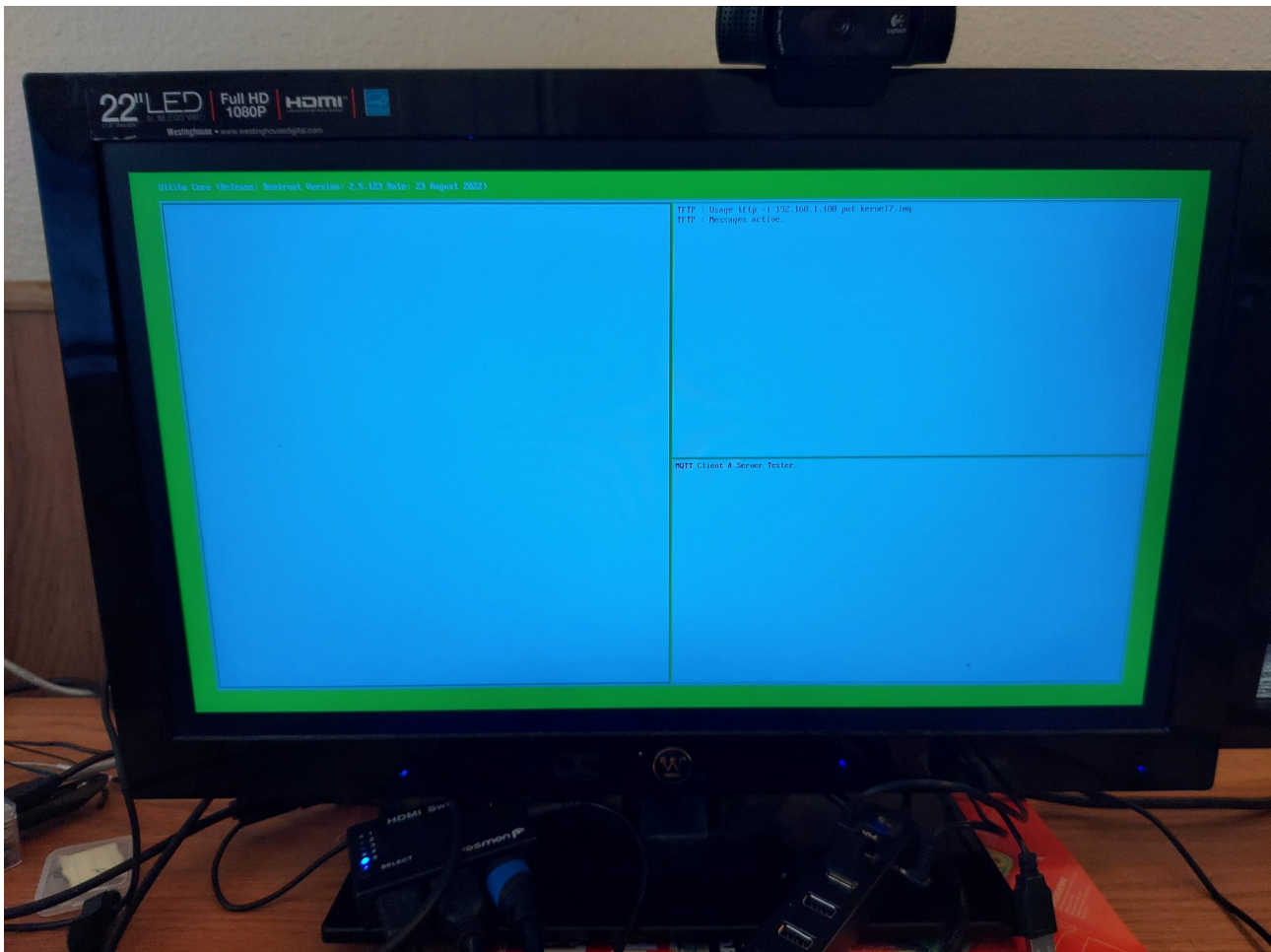
Name	Size	Modified
firmware		12/11/2021 09:17
olddata		11/27/2021 16:13
a.bin	4.1 KiB	04/28/2022 09:19
a64.bin	4.1 KiB	04/28/2022 09:19
bb.bin	4.1 KiB	04/28/2022 09:19
bootcode.bin	51.2 KiB	12/11/2021 06:22
cmdline.txt	56 bytes	11/27/2021 06:13
config.txt	34 bytes	12/11/2021 06:22
fixup.dat	7.1 KiB	12/11/2021 06:22
fixup4.dat	5.3 KiB	12/11/2021 06:22
kernel.img	3.0 MiB	12/10/2021 19:34
kernel7.img	3.1 MiB	05/04/2022 07:56
kltdwt.txt	5.1 KiB	05/13/2022 10:09
pico.log	699.8 KiB	12/31/1979 17:00
start.elf	2.8 MiB	12/11/2021 06:22
start4.elf	2.1 MiB	12/11/2021 06:22

16 items

Free space: 7.1 GiB (Total: 7.3 GiB)



Eject the micro sd and install on Raspberry Pi 2. Put the micro sd in the RPi2 and power up.



```
ping 192.168.1.180
PING 192.168.1.180 (192.168.1.180) 56(84) bytes of data.
64 bytes from 192.168.1.180: icmp_seq=2 ttl=128 time=1.88 ms
64 bytes from 192.168.1.180: icmp_seq=3 ttl=128 time=1.75 ms
c64 bytes from 192.168.1.180: icmp_seq=4 ttl=128 time=2.34 ms
64 bytes from 192.168.1.180: icmp_seq=5 ttl=128 time=1.88 ms
^C
--- 192.168.1.180 ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4052ms
rtt min/avg/max/mdev = 1.746/1.963/2.340/0.224 ms
```

Next we will add telnet to the project

```
{ needed for telnet }
  Shell,
  ShellFilesystem,
  ShellUpdate,
  RemoteShell,
{ needed for telnet }
```

This now provides a telnet feature to our Bare Metal App.

“telnet 192.168.1.180”

```
File Edit Tabs Help
Ultibo Core (Release: Beetroot Version: 2.5.123 Date: 23 August 2022)
(Type HELP for a list of available commands)
Directory of C:\

2022-11-07 15:44:24      2842624 kernel7.img
1980-01-01 00:00:00      716584 pico.log
2021-12-11 16:17:42    <DIR>      firmware
2021-12-11 13:22:30      52456 bootcode.bin
2021-12-11 13:22:30         34 config.txt
2021-11-27 13:13:32         56 cmdline.txt
2021-12-11 13:22:30      7313 fixup.dat
2021-12-11 13:22:30      5442 fixup4.dat
2021-12-11 02:34:42     3176724 kernel.img
2021-12-11 13:22:30     2955936 start.elf
2021-12-11 13:22:30     2231712 start4.elf
2022-04-28 15:19:20        4160 a64.bin
2022-04-28 15:19:22        4160 a.bin
2022-04-28 15:19:22        4160 bb.bin
2022-05-13 16:09:32        5251 klt0wt.txt
2021-11-27 23:13:28    <DIR>      olddata
          14 file(s) 12006612 bytes
          2 dir(s)

C:\>
```

To exit “logout”

Next we will test the tftp support which provides the transfer from the host to the Ultibo Bare Metal and from Ultibo Bare Metal to the host.

First we create an empty file with the command “touch to-bare-metal.txt”.

Add some text

```
tftp> put to-bare-metal.txt
Sent 95 bytes in 0.0 seconds
tftp> quit
```

```
File Edit Tabs Help
2021-12-11 13:22:30      52456 bootcode.bin
2021-12-11 13:22:30         34 config.txt
2021-11-27 13:13:32         56 cmdline.txt
2021-12-11 13:22:30      7313 fixup.dat
2021-12-11 13:22:30      5442 fixup4.dat
2021-12-11 02:34:42     3176724 kernel.img
2021-12-11 13:22:30     2955936 start.elf
2021-12-11 13:22:30     2231712 start4.elf
2022-04-28 15:19:20        4160 a64.bin
2022-04-28 15:19:22        4160 a.bin
2022-04-28 15:19:22        4160 bb.bin
2022-05-13 16:09:32        5251 klt0wt.txt
2021-11-27 23:13:28    <DIR>      olddata
          14 file(s) 12006612 bytes
          2 dir(s)

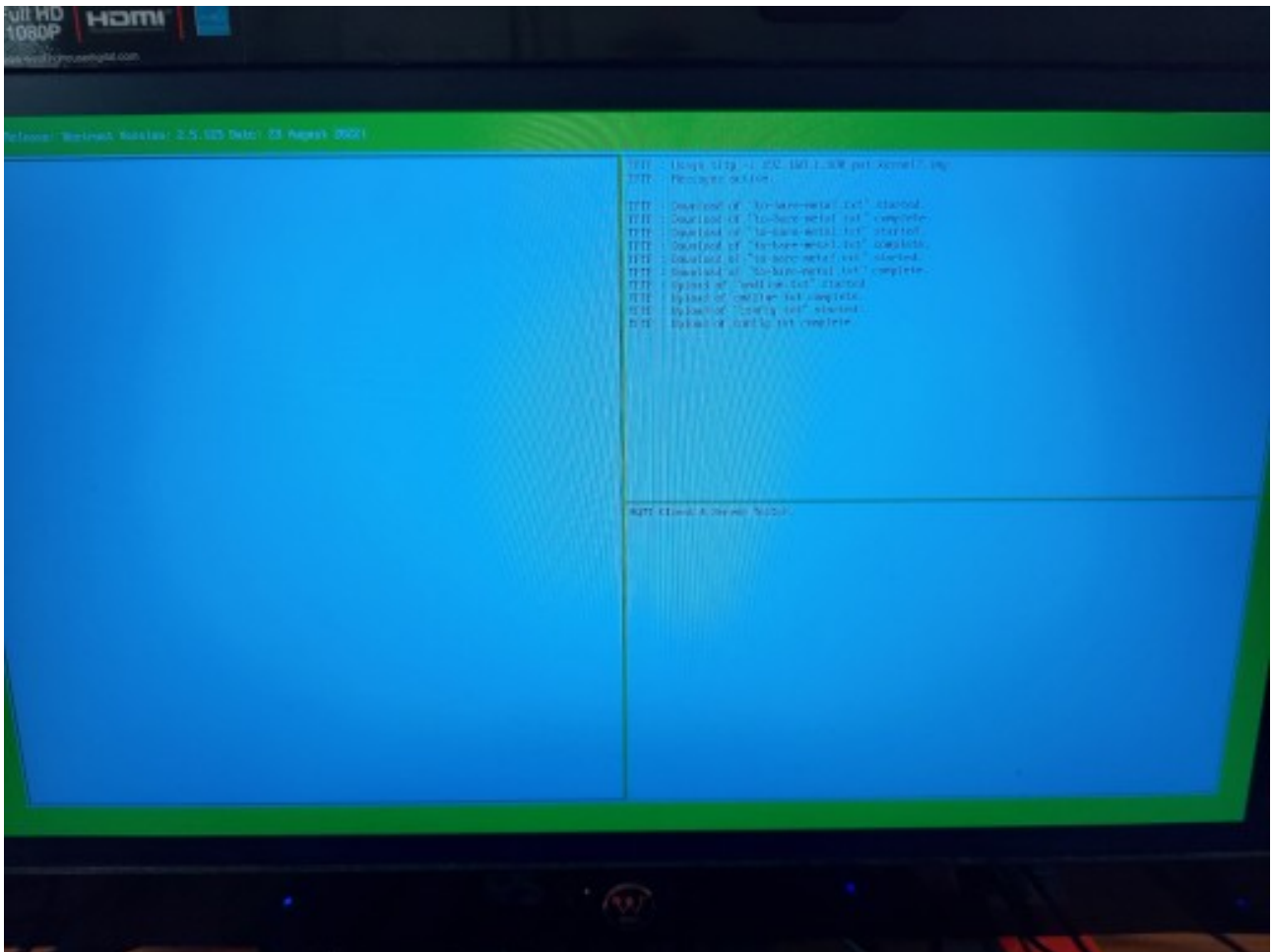
This file will be sent to Ultibo Bare metal system using tftp.
tftp 192.168.1.180
tftp> binary
tftp> put to-bare-metal.txt
Sent 95 bytes in 0.0 seconds
tftp> quit

C:\>
```

Sent 163 bytes in 0.0 seconds

```
tftp> quit
```

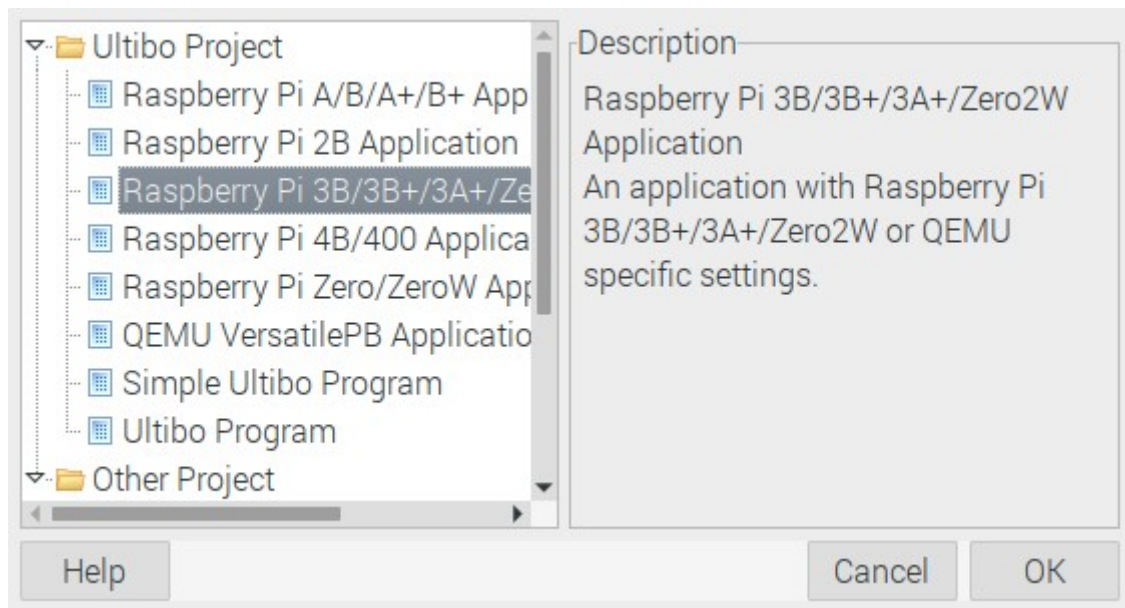
```
gpu_mem=128
```



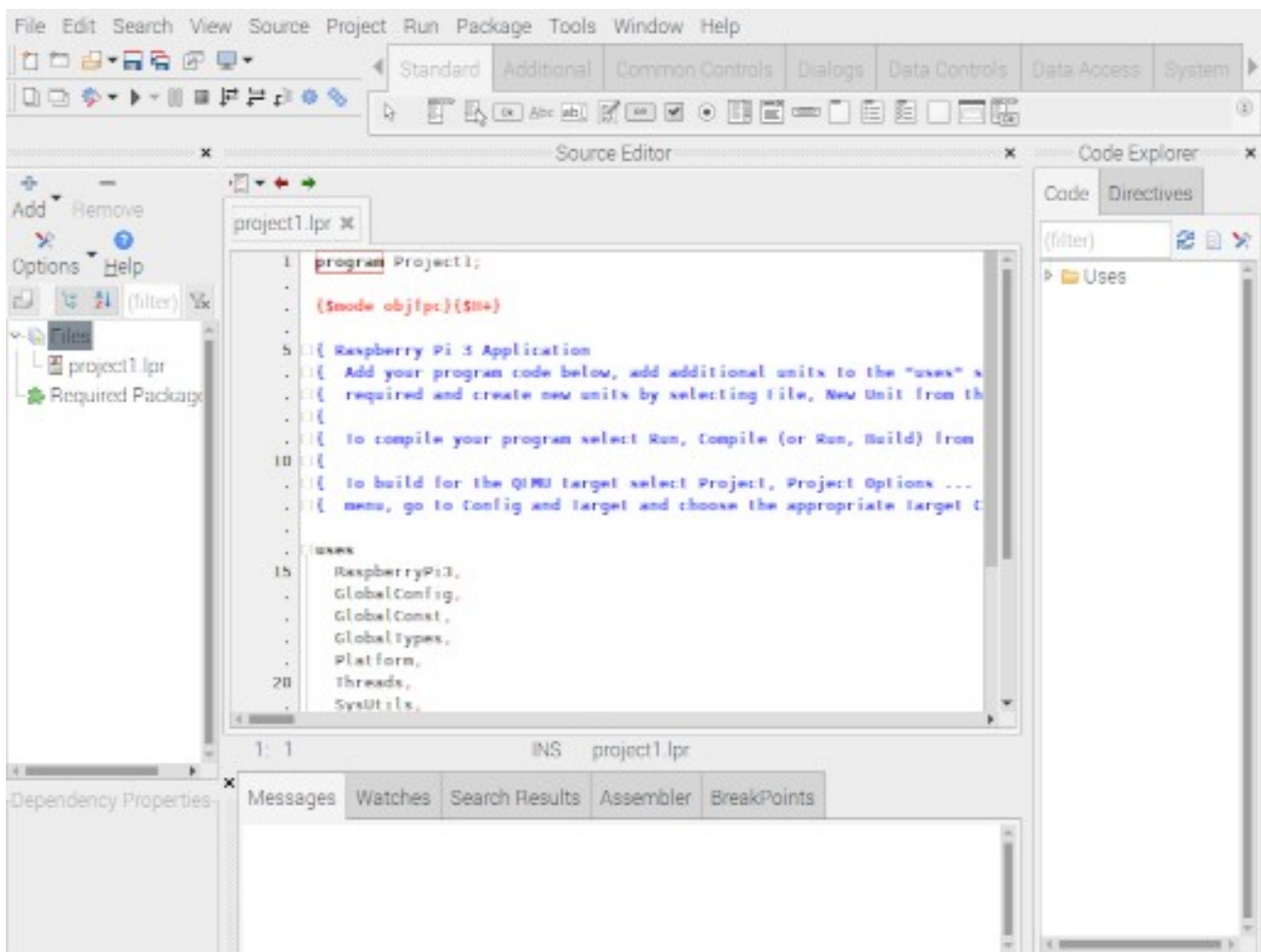
Next step will be to add to Ultibo-Projects

In the folder **“Ultibo_Projects/Pauls-ultibo-mqtt”**

create 3 folders mkdir RPi3, mkdir RPi2, and mkdir QEMU.



This will bring up a new project when you depress “ok”. We will paste Paul’s code in this new project. Then save in the new folder `Ultibo_Projects/Pauls-ultibo-mqtt/`



RPi3

Build modes: Default

Config files

- ☒ Use standard compiler config file (fpc.cfg) (If not checked: -n)
- ☐ Use additional compiler config file (@)
extrafpc.cfg

Target platform

- Target OS (-T): Ultibo
- Target CPU family (-P): arm
- Target processor (-Cp): ARMV7A
- Target controller (-Wp): RPi3B

Target-specific options

- ☐ Win32 gui application (-WG, ignored)

Set compiler options as default: ☐

Buttons: Help, Show Options, Test, Export, Import, Cancel, OK

RPi2

Build modes: Default

Config files

- ☒ Use standard compiler config file (fpc.cfg) (If not checked: -n)
- ☐ Use additional compiler config file (@)
extrafpc.cfg

Target platform

- Target OS (-T): Ultibo
- Target CPU family (-P): arm
- Target processor (-Cp): ARMV7A
- Target controller (-Wp): RPi2B

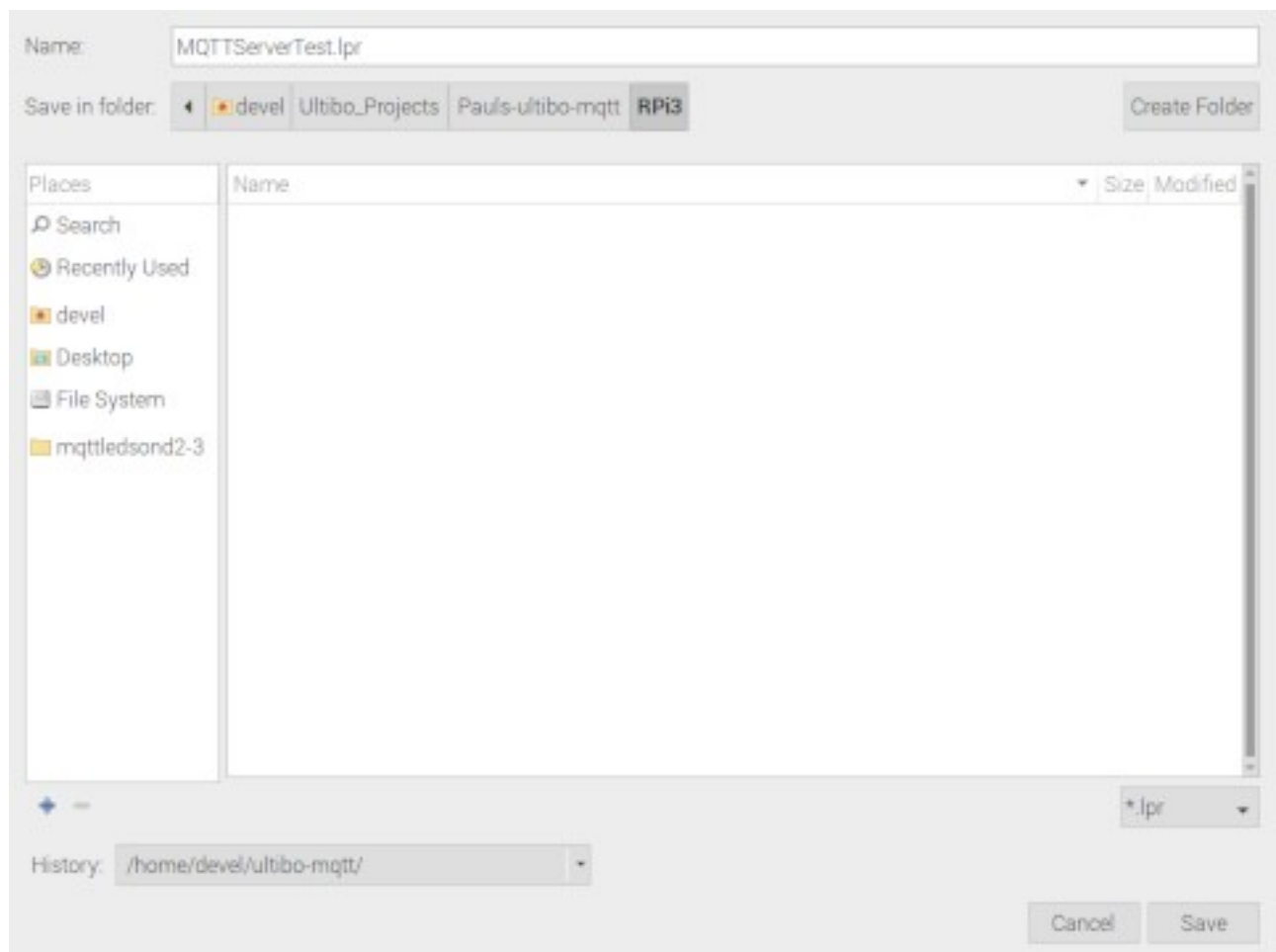
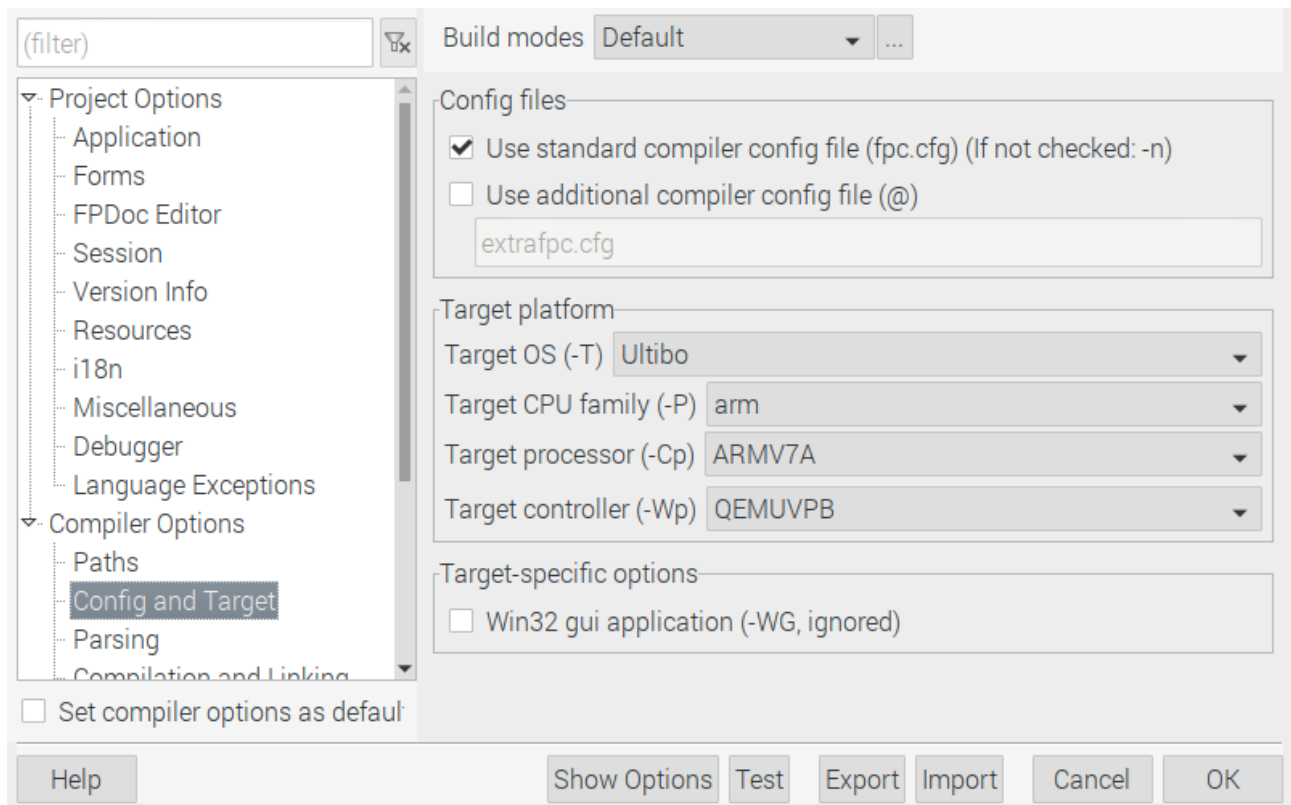
Target-specific options

- ☐ Win32 gui application (-WG, ignored)

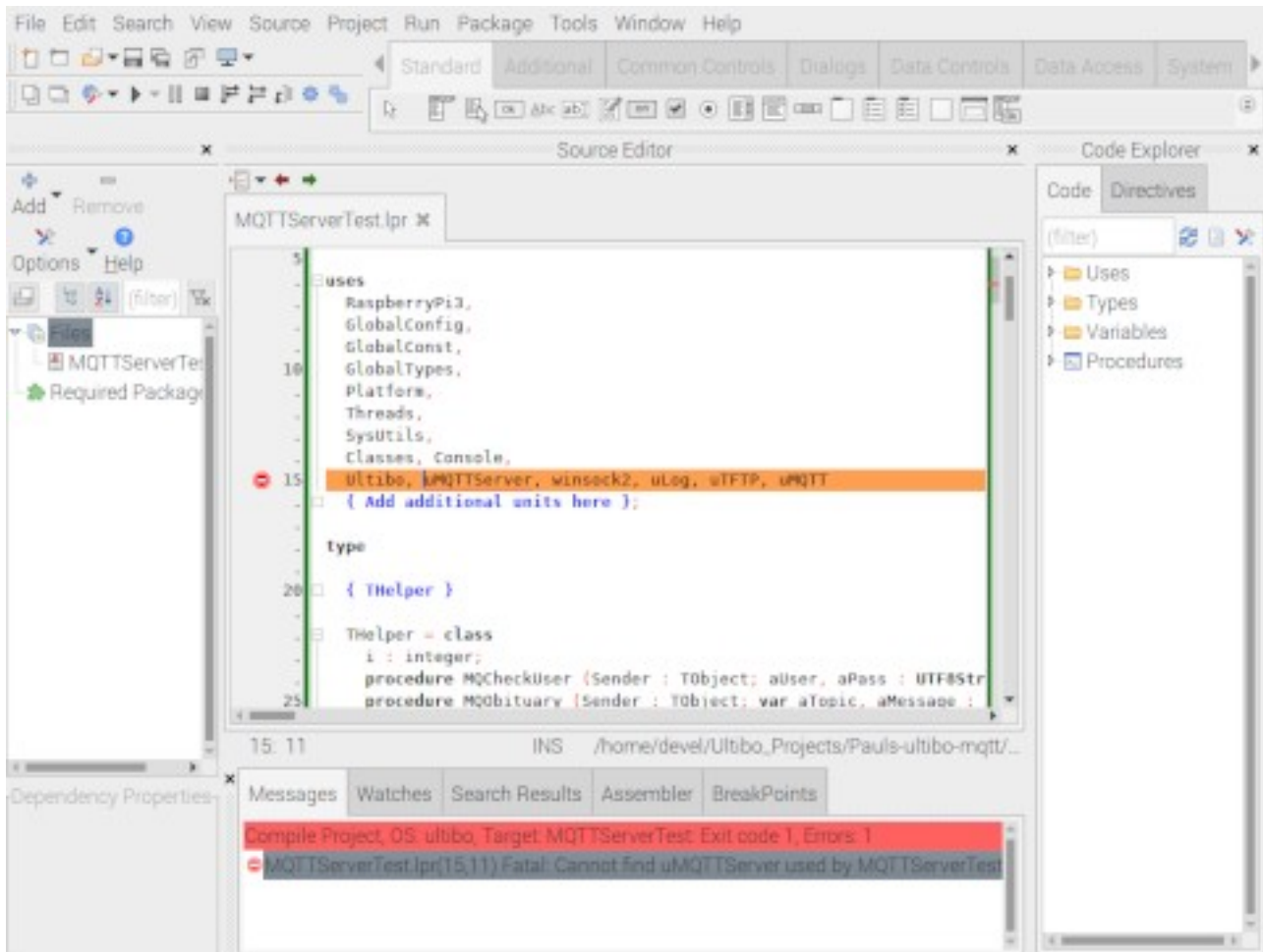
Set compiler options as default: ☐

Buttons: Help, Show Options, Test, Export, Import, Cancel, OK

QEMU



When we try and compile the project we will get an error.



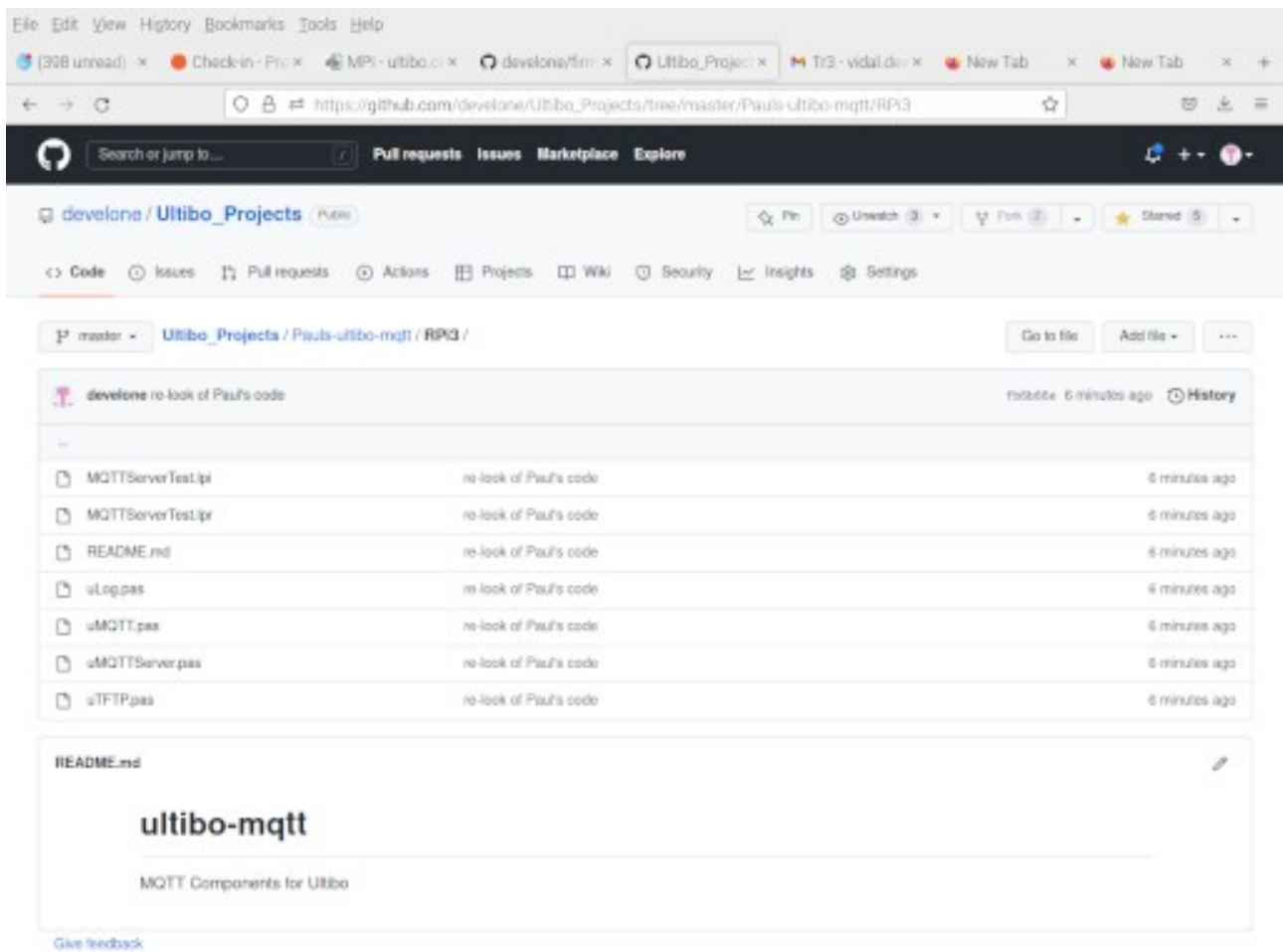
This error occurs, since we do not have the required files.
uLog.pas, uMQTT.pas, uMQTTServer.pas, and uTFTP.pas.
The lower case u is used for Unit.

These steps can only be done if you own the repo and you used
git clone [git@github.com:develone/Ultibo_Projects.git](https://github.com/develone/Ultibo_Projects.git) git clone
instead of https://github.com/develone/Ultibo_Projects.git

git add RPi3/MQTTServerTest.lpi RPi3/MQTTServerTest.lpr RPi3/README.md
RPi3/uLog.pas RPi3/uMQTT.pas RPi3/uMQTTServer.pas RPi3/uTFTP.pas
RPi3/README.md

git commit RPi3/MQTTServerTest.lpi RPi3/MQTTServerTest.lpr RPi3/README.md
RPi3/uLog.pas RPi3/uMQTT.pas RPi3/uMQTTServer.pas RPi3/uTFTP.pas
RPi3/README.md

git push



Next we add telnet and create a RPi2 and QEMUVersatilePB .
 QEMU will require 2 new files
startqemu.sh and **disk.img** which takes the place of the micro sd

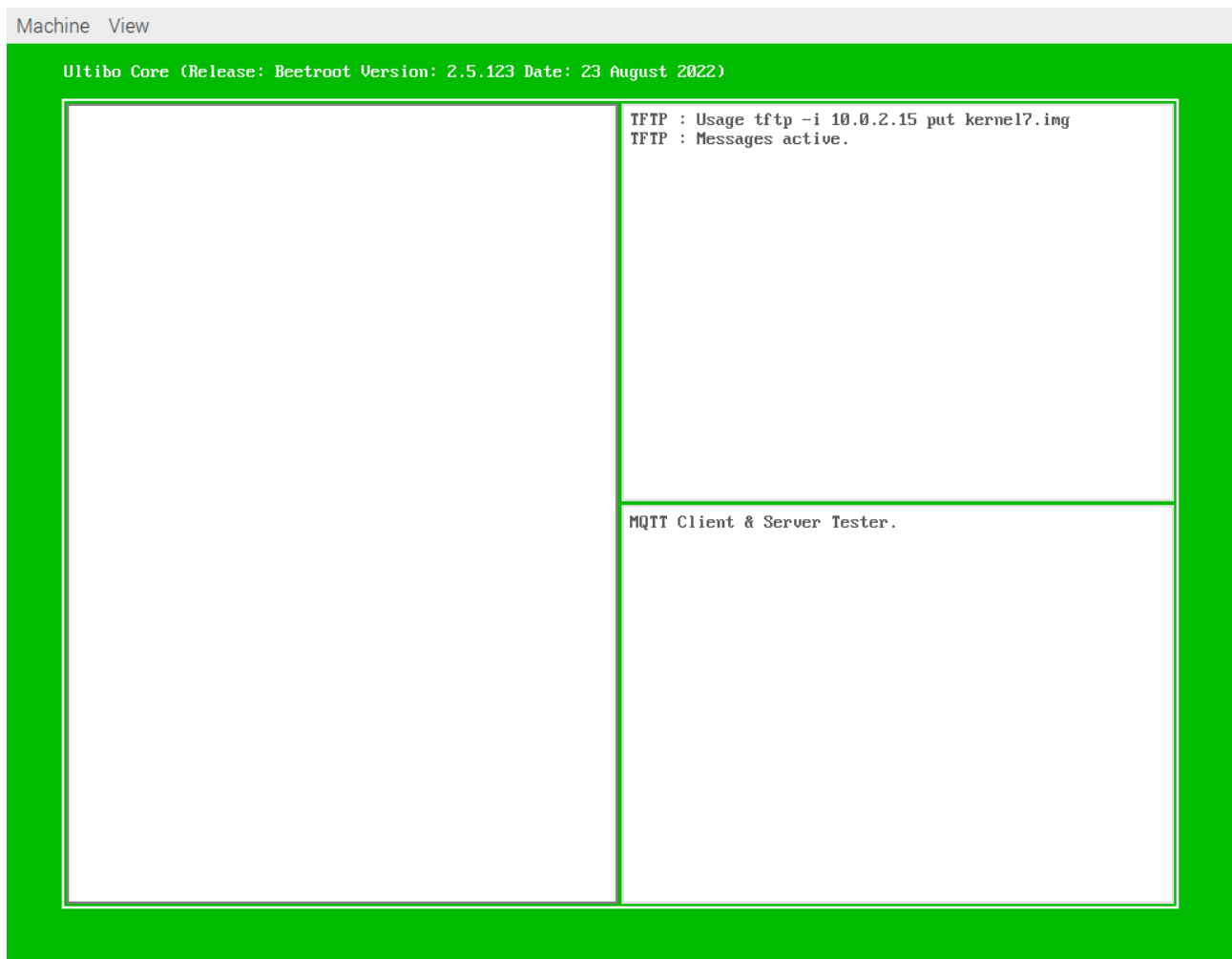
```
#!/bin/bash
qemu-system-arm -machine versatilepb -cpu cortex-a8 -kernel kernel.bin \
-net
user,hostfwd=tcp::5080-:80,hostfwd=tcp::5023-:23,hostfwd=udp::5069-:69,hostfwd=tcp::6050-:5050 -net nic \
-drive file=disk.img,if=sd,format=raw
```

The startqemu.sh maps the ports for telnet 20 to 5023 and tftp 69 to 5069.

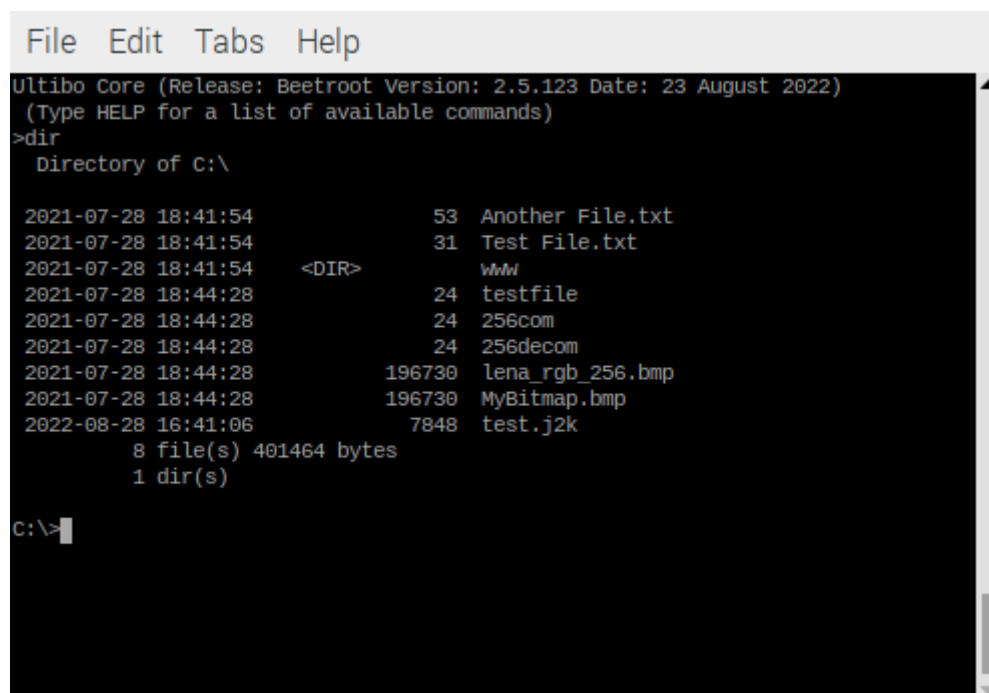
If you start a new shell without . Ultibo_Projects/picoultibo.sh

File Edit Tabs Help

```
devel@pi4-37:~ $ cd Ultibo_Projects/Pauls-ultibo-mqtt/QEMU/
devel@pi4-37:~/Ultibo_Projects/Pauls-ultibo-mqtt/QEMU $ ./startqemu.sh
./startqemu.sh: line 2: qemu-system-arm: command not found
devel@pi4-37:~/Ultibo_Projects/Pauls-ultibo-mqtt/QEMU $ . ~/Ultibo_Projects/pico
ultibo.sh
/home/devel/ultibo/core:/home/devel/qemu-6.2.0-rpios/bin:/home/devel/local/openo
cd/bin:/home/devel/picotool/build/:/home/devel/.pyenv/plugins/pyenv-virtualenv/s
hims:/home/devel/.pyenv/shims:/home/devel/.pyenv/bin:/usr/local/sbin:/usr/local/
bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games
devel@pi4-37:~/Ultibo_Projects/Pauls-ultibo-mqtt/QEMU $ ./startqemu.sh
```



telnet xx.xx.xx.xx 5023



tftp xx.xx.xx.xx 5069


```
tftp> binary
tftp> get xx.txt
Received 53 bytes in 0.1 seconds
tftp> quit
cat xx.txt
This is another test file, just like the first one tftp xx.xx.xx.xx 5069
tftp> binary
tftp> get xx.txt
Received 53 bytes in 0.1 seconds
tftp> quit
cat xx.txt
This is another test file, just like the first one
```