Building your own system based on the open and free 64-bit “Microwatt” OpenPOWER ISA processor.

H. Peter Hofstee
Paul Mackerras

Contributions from:
Madhavan Srinivasan, Jayakumar Singaram, Pranose Edavoor

Goal for Today's Session

- Provide you with an easy to follow set of instructions on how to build your own MicroWatt based linux system on the Arty A7-100T FPGA

- All instructions (and session recordings) are available at

https://github.com/hofstee-hp/

MicroWatt Summary

Tiny in-order single issue powerpc core

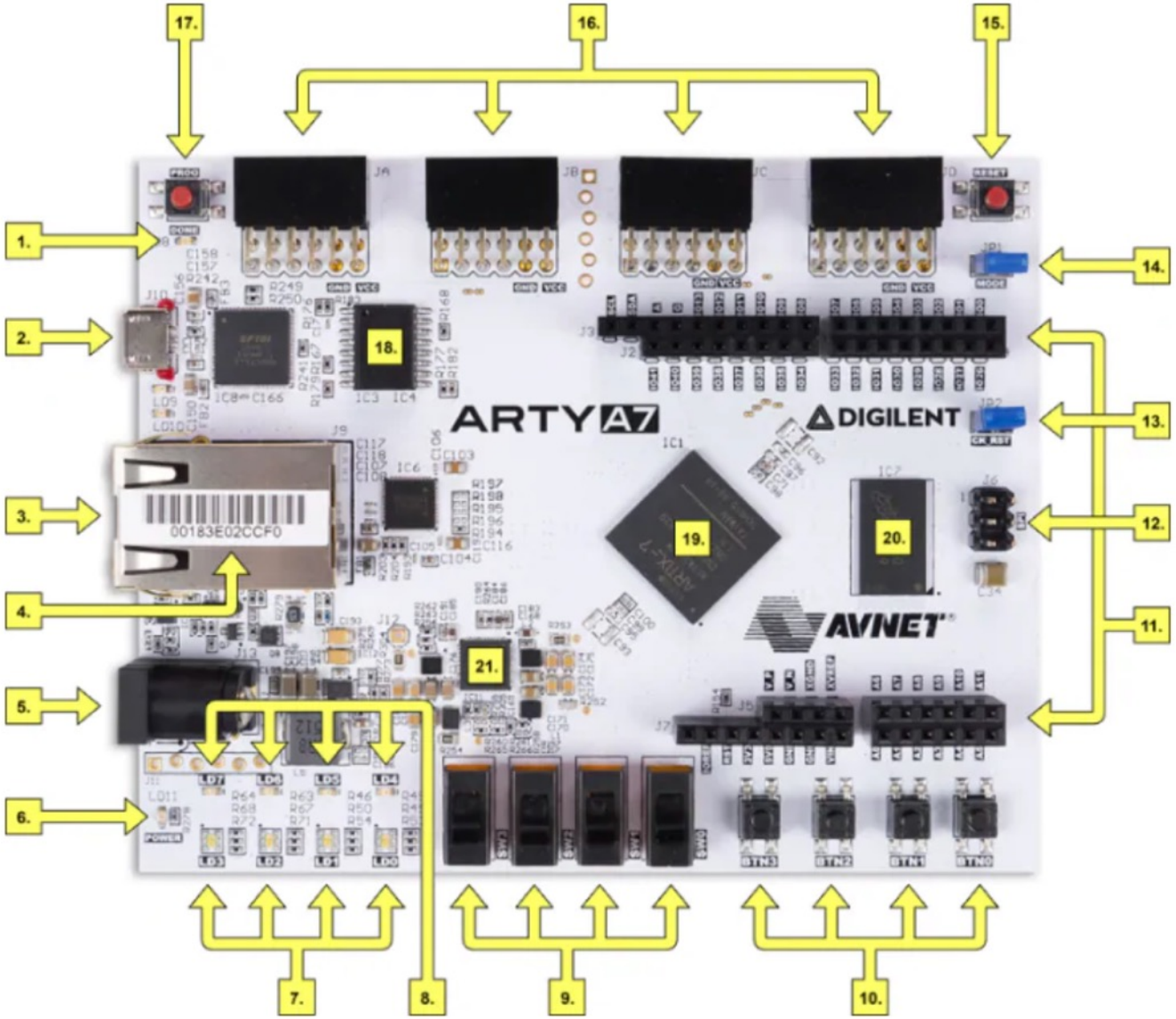
https://github.com/antonblanchard/microwatt
https://youtu.be/uEAoMCE6IKo

Power ISA compliant

https://openpowerfoundation.org/
https://youtu.be/CnMwCrtz6MA

Small enough to fit on a small FPGA and be used as a microcontroller, but capable enough to run linux!

Arty A7-100T



Callout	Description
1	FPGA programming DONE LED
2	Shared USB JTAG / UART port
3	Ethernet connector
4	MAC address sticker
5	Power jack for optional external supply
6	Power good LED
7	User LEDs
8	User RGB LEDs
9	User slide switches
10	User push buttons
11	Arduino/chipKIT shield connectors
12	Arduino/chipKIT shield SPI connector
13	chipKIT processor reset jumper
14	FPGA programming mode
15	chipKIT processor reset
16	Pmod connectors
17	FPGA programming reset button
18	SPI flash memory
19	Artix FPGA
20	Micron DDR3 memory
21	Dialog Semiconductor DA9062 power supply

What this project depends on

- Ubuntu 22.04.4 system (x86-64 connected to network)
  - Recommend at least 8GB RAM and 100GB storage
    - Tested w. 16GB / 1TB storage (but less than 100GB storage used)
  - Instructions assume only a minimal install
  - Only thing needed if you want to simulate MicroWatt

- Vivado 2024.1
  - Installed for CDAC VMs, install instructions included if you have your own Ubuntu 22.04.4 system

- Arty A7-100T

Dependencies – pre-installed at CDAC

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential git ghdl-common ghdl ghdl-llvm gnat
$ sudo apt-get install binutils-powerpc64le* gcc-powerpc64le-* g++-powerpc64le-*
```

The `build-essential` package typically includes:

1. ***GNU Compiler Collection (GCC)***: The essential compiler for C, C++, and other programming languages.
2. ***GNU Make***: A build automation tool that manages the build process of a project.
3. ***libc6-dev***: Development libraries and header files for the C library.
4. ***dpkg-dev***: Tools for building Debian source packages.
5. ***GNU Binutils***: A collection of binary tools, including the linker and assembler.

Source: medium.com

Dependencies (cont.)

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential git ghdl-common ghdl ghdl-llvm gnat
$ sudo apt-get install binutils-powerpc64le* gcc-powerpc64le-* g++-powerpc64le-*
```

GHDL is an open-source simulator for the VHDL language. GHDL allows you to compile and execute your VHDL code directly in your PC.

GHDL fully supports the 1987, 1993, 2002 versions of the IEEE 1076 VHDL standard, and partially the latest 2008 revision (well enough to support fixed_generic_pkg or float_generic_pkg).

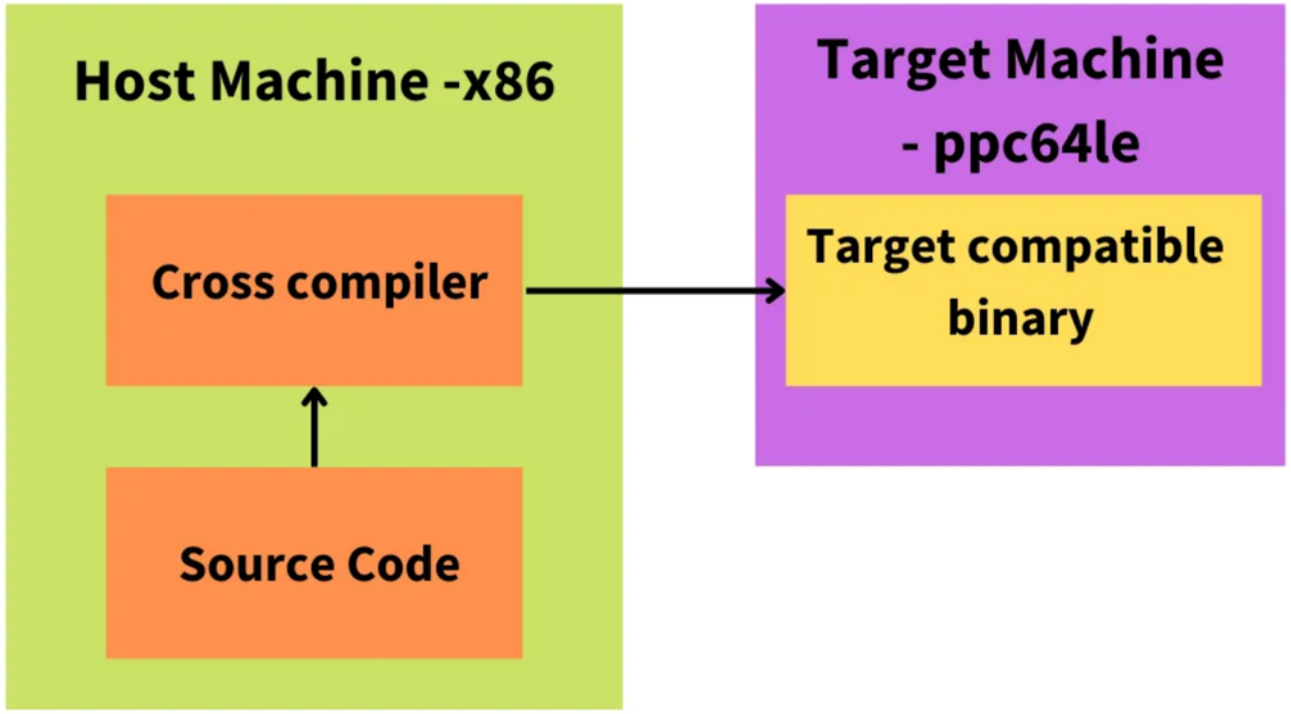
By using a code generator (llvm, GCC or a builtin one), GHDL is much faster than any interpreted simulator. GHDL runs on Linux, Windows and Apple OS X. You can freely download a binary distribution for your OS or try to compile GHDL on your own machine.

Go to GitHub to get releases or to get the sources!

gnat is the GNU Ada compiler

Dependencies (cont.)

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential git ghdl-common ghdl ghdl-llvm gnat
$ sudo apt-get install binutils-powerpc64le* gcc-powerpc64le-* g++-powerpc64le-*
```



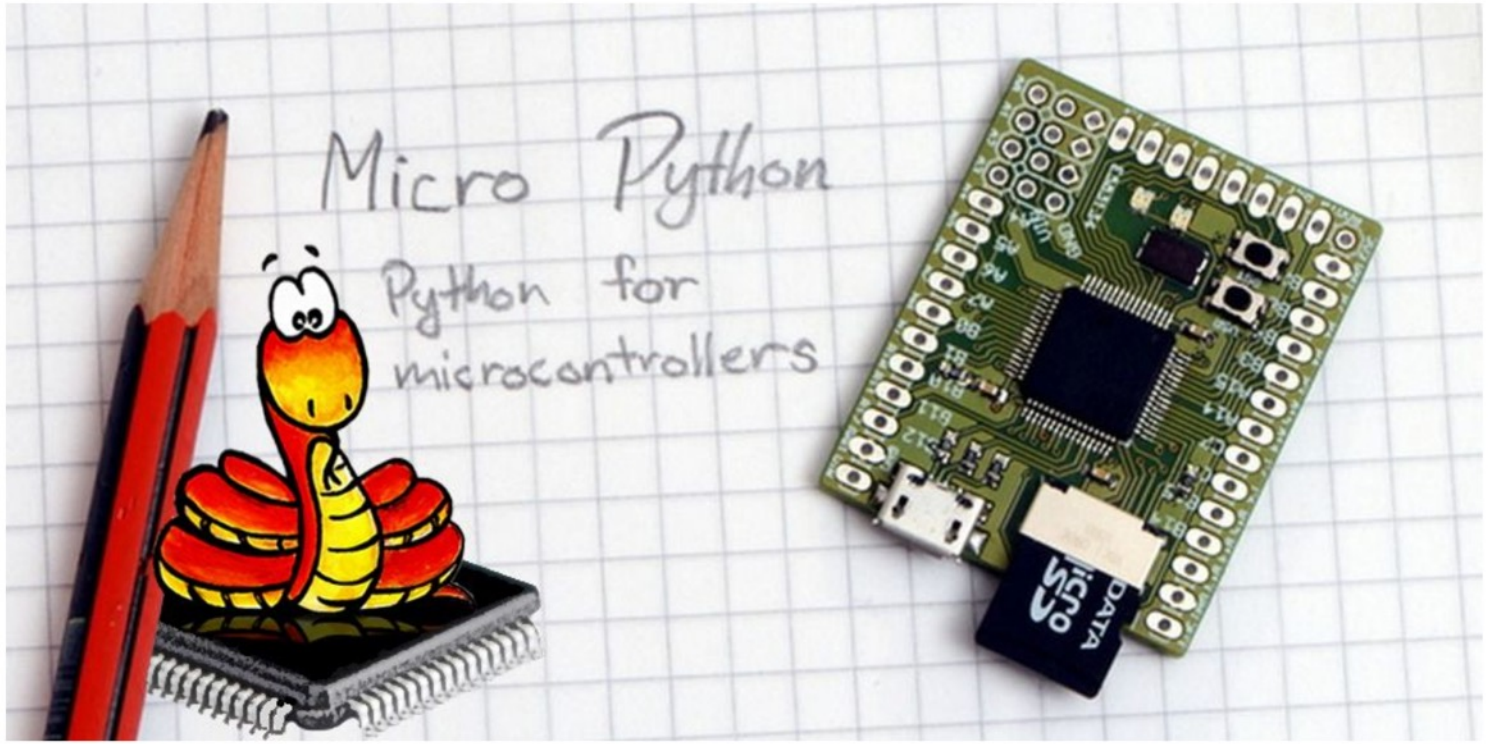
Cross compiler operation

https://medium.com/@mayurwaghmode/cross-compiling-c-c-code-for-ppc64le-on-x86-64-6226ff3a80b7

Building Micropython

```
$ cd ~
$ git clone https://github.com/micropython/micropython.git
$ cd ~/micropython
$ cd ports/powerpc
$ make
```

The MicroPython project



This is the MicroPython project, which aims to put an implementation of Python 3.x on microcontrollers and small embedded systems. You can find the official website at micropython.org.

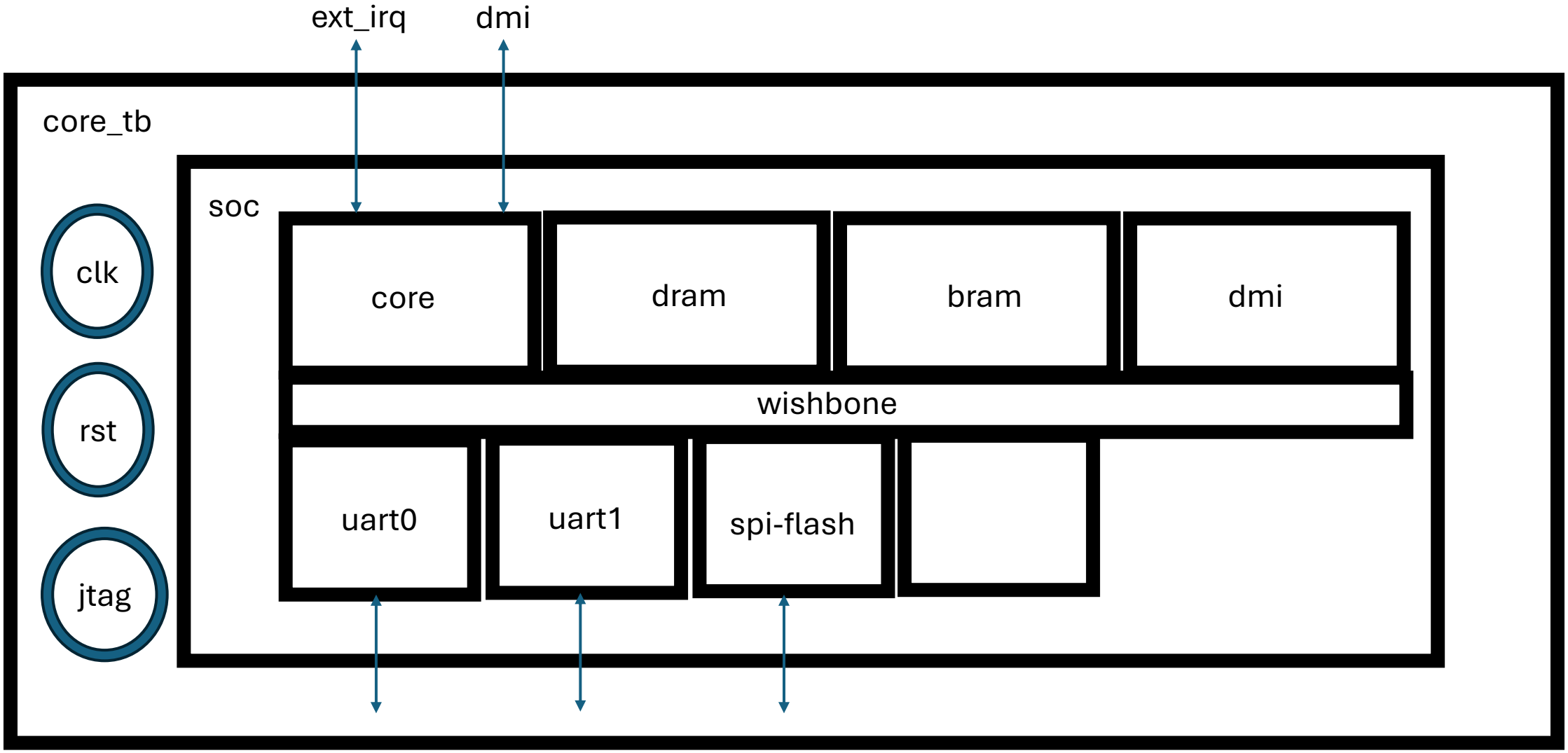
Building MicroWatt

```
$ cd ~
$ git clone https://github.com/antonblanchard/microwatt
$ cd ~/microwatt
$ make
```

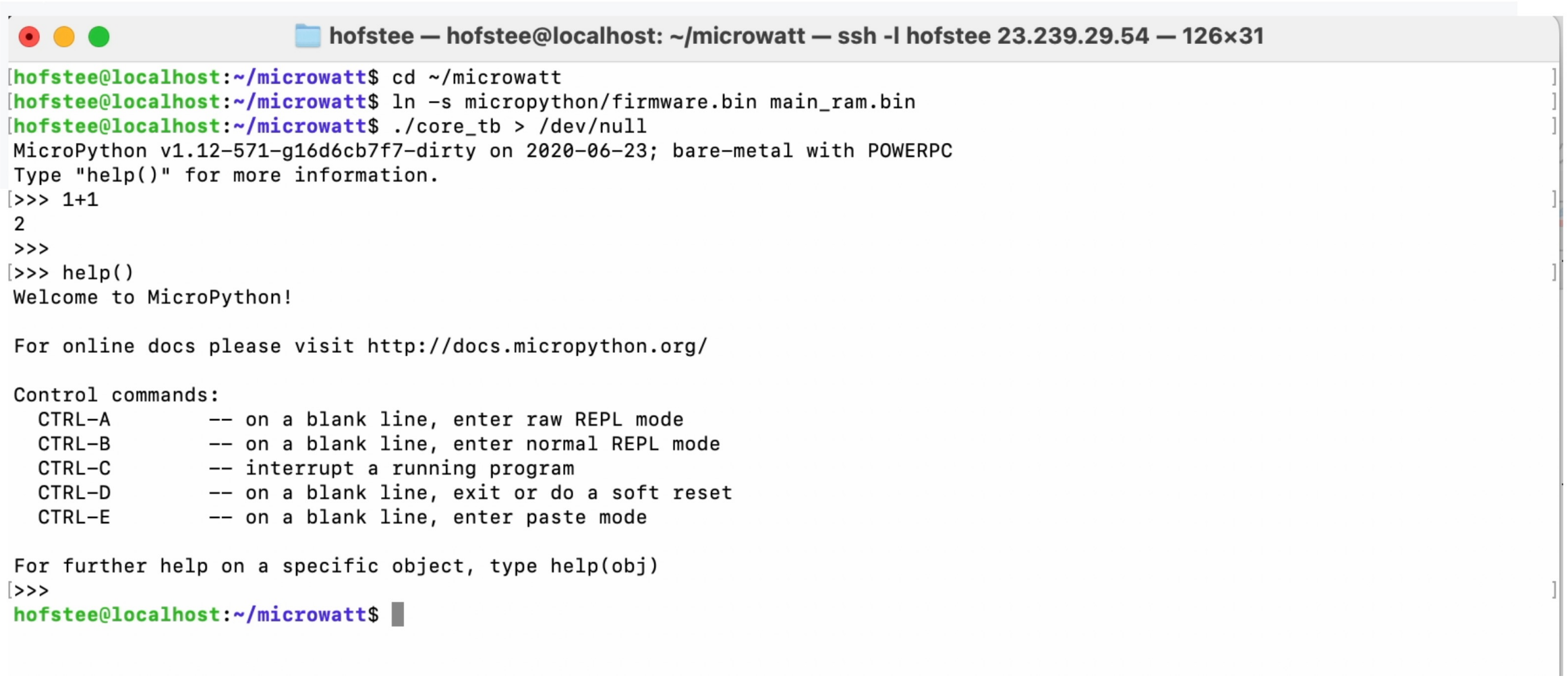
Top-level source is **core_tb.vhd**

UNDER CONSTRUCTION

High-Level SoC Structure



Running GHDL-compiled MicroWatt



Running GHDL-compiled MicroWatt

Building BuildRoot / Linux Kernel

- The default linux kernel for ppc64le has a dependency on library functions that use instructions that operate on the 128b VMX registers, and these instructions are not included in the subset that MicroWatt (currently) implements, hence a special build is required
- Detailed steps on how to build the kernel are included on the github page
- It is not difficult to do, but it takes a long time
- You can instead download a pre-built kernel

Building Buildroot & Linux Kernel

The linux build requires flex and bison - pre-installed if you are using the CDAC environment

```
$ sudo apt-get install -y flex
$ sudo apt-get install -y bison
```

Use buildroot to create a userspace.

A small change is required to glibc in order to support the VMX/AltiVec-less Microwatt, as float128 support is mandatory and for this in GCC requires VSX/AltiVec. This change is included in a buildroot fork, along with a defconfig:

Build buildroot

```
$ cd ~
$ git clone -b microwatt https://github.com/shenki/buildroot
$ cd buildroot
$ make ppc64le_microwatt_defconfig
$ make
```

The output is output/images/rootfs.cpio.

Next build the Linux kernel

```
$ git clone https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
$ cd linux
$ make ARCH=powerpc microwatt_defconfig
$ make ARCH=powerpc CROSS_COMPILE=powerpc64le-linux-gnu- CONFIG_INITRAMFS_SOURCE=/buildroot/ou
$ cp arch/powerpc/boot/dtbImage.microwatt.elf ~/arty
```

Installing Vivado 24.01

- This project has been tested with Vivado 24.01
  - Older versions may also work

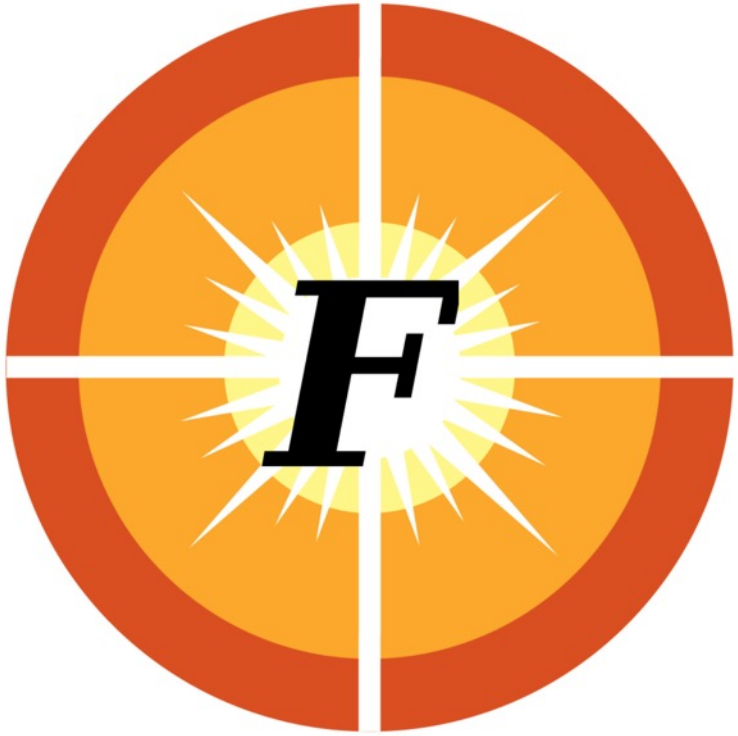
- Vivado is pre-installed in the CDAC build environment
  - You do not have to install Vivado yourself
  - If you prefer to use your own system to run Vivado, instructions on how to install Vivado can be found on the github page

- After Vivado is installed you need to add the Digilent board package for Arty A7-100
  - Also-preinstalled in CDAC environment

FuseSoC

FuseSoC is an award-winning package manager and a set of build tools for HDL (Hardware Description Language) code.

Its main purpose is to increase reuse of IP (Intellectual Property) cores and be an aid for creating, building and simulating SoC solutions.



FuseSoC makes it easier to

- reuse existing cores
- create compile-time or run-time configurations
- run regression tests against multiple simulators
- Port designs to new targets
- let other projects use your code
- set up continuous integration

https://github.com/fusesoc/fusesoc.github.io

Install FuseSoC – preinstalled at CDAC

```
$ cd ~
$ sudo ln -s /usr/bin/python3 /usr/local/bin/python
$ sudo apt-get install -y python3-pip
$ pip3 install --user -U fusesoc
$ export PATH=$PATH:~/.local/bin
```

Building the MicroWatt Bitfile

```
------ Update paths
$ cd <Xilinx install dir>/Xilinx/Vivado/2024.1
$ source settings64.sh
```

Not needed in CDAC env.

Import the required elements and build the bitfile for the MicroWatt on Arty

```
$ fusesoc library add microwatt microwatt
$ fusesoc fetch uart16550
$ fusesoc run --build --target=arty_a7-100 microwatt --no_bram --memory_size=0
$ cp build/microwatt_0/arty_a7-100-vivado/microwatt_0.bit ~/arty
```

Tools Needed to Program and Run Microwatt

```
----- If you are installing on a system that has the board attached
$ sudo apt-get install openocd
$ sudo apt-get install putty
$ sudo apt-get install gtkterm
$ cd <Xilinx install dir>/Xilinx/Vivado/2024.1/data/xicom/cable_drivers/
$ cd lin64/install_scripts/install_drivers
$ ./install_drivers
$ sudo adduser $USER dialout
```

And Finally ... Run Linux on MicroWatt

- Get the linux buildroot elf file if you did not build it yourself, the second "gdown" command allows you to get the bitfile if you did not build it yourself

```
$ pip3 install gdown
$ cd ~/arty
$ gdown https://drive.google.com/uc?id=1JRmkKseXCFwHaXCmdC5NrvXIwwQXpkey
$ gdown https://drive.google.com/uc?id=1v7KqhiqnXxnyWRlK-5k4L4S6MJzLmkW3
```

This next operation will overwrite the contents of the flash on the Arty board .

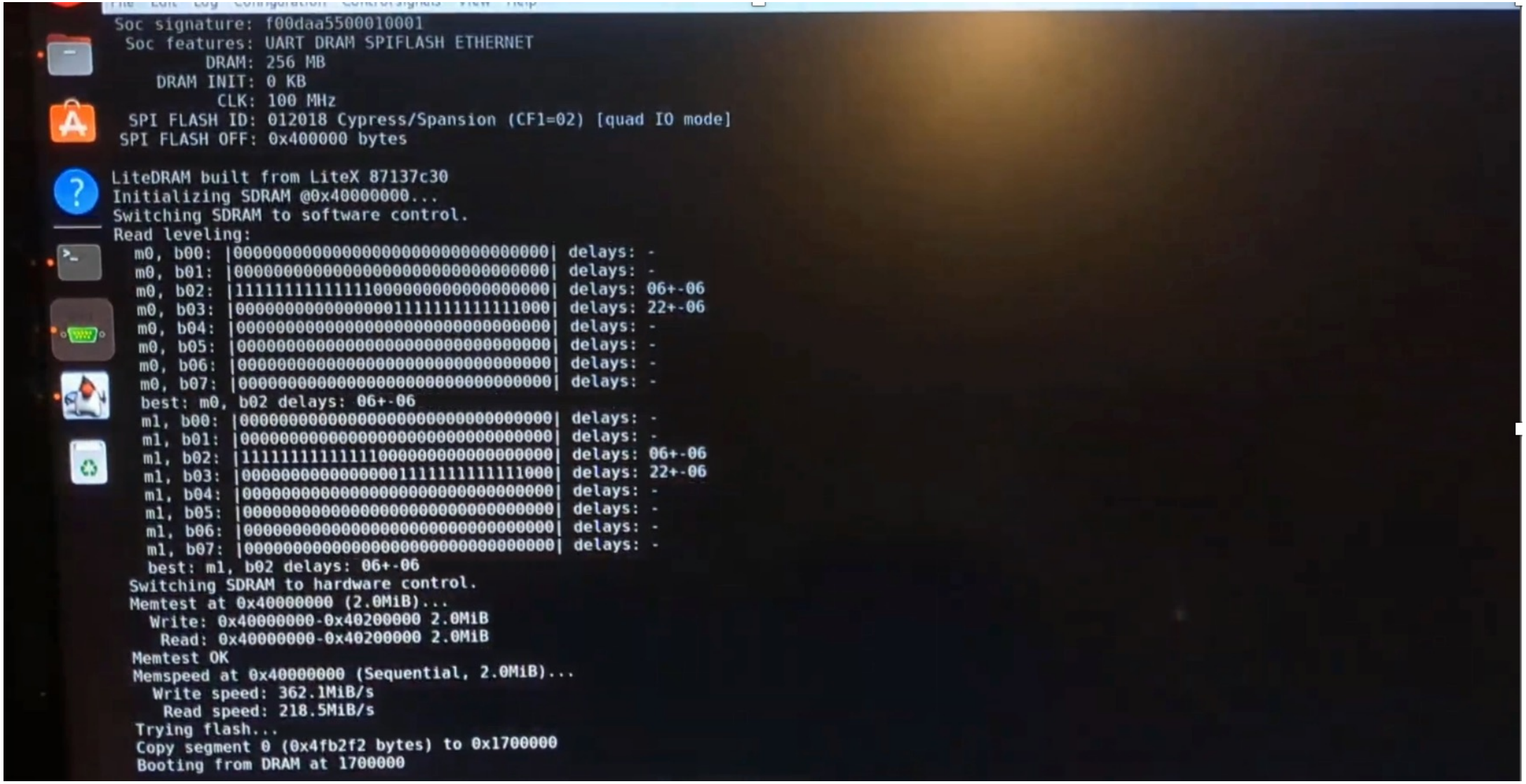
```
$ cd ~/arty
$ ~/microwatt/openocd/flash-arty -f a100 microwatt_0.bit
$ ~/microwatt/openocd/flash-arty -f a100 dtbImage.microwatt.elf -t bin -a 0x400000
```

See it boot!

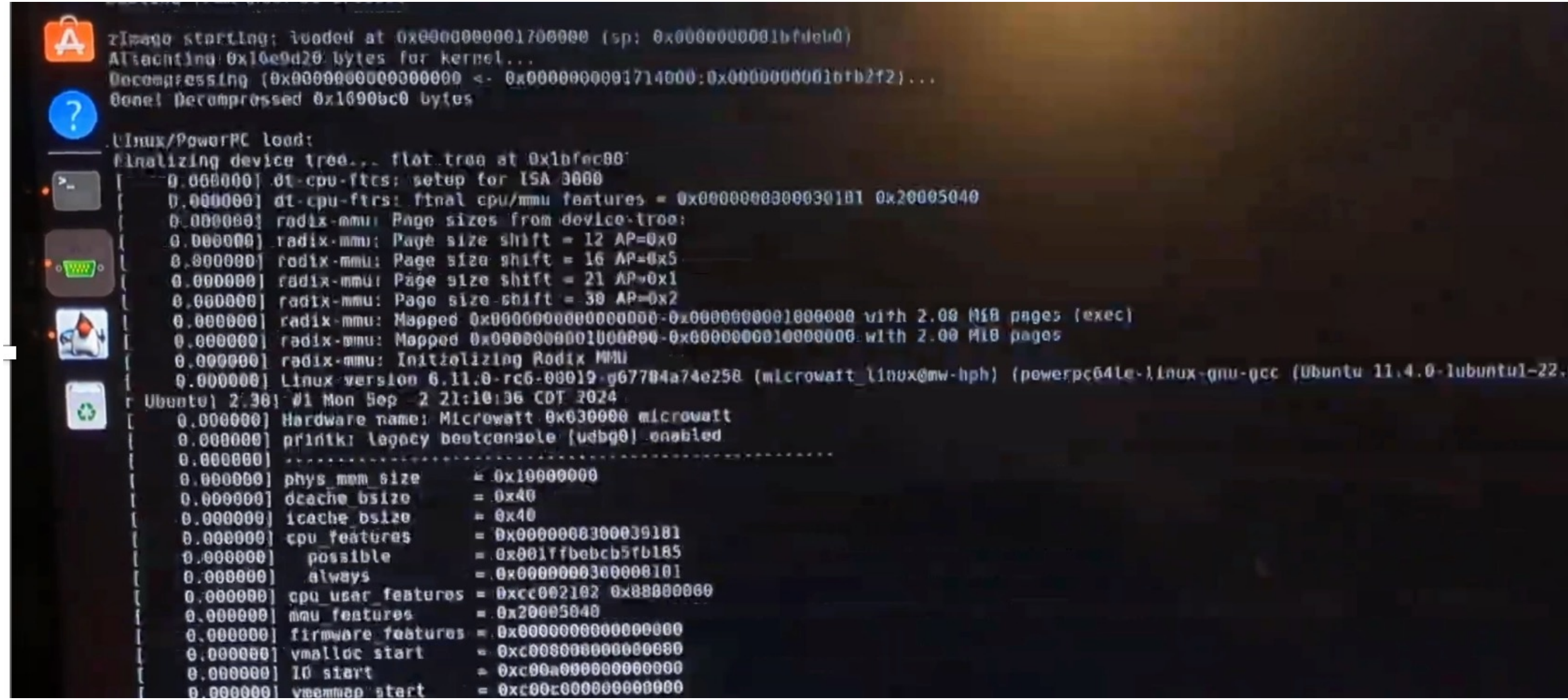
Connect to the second USB TTY device exposed by the FPGA

```
$ gtkterm -p /dev/ttyUSB1
```

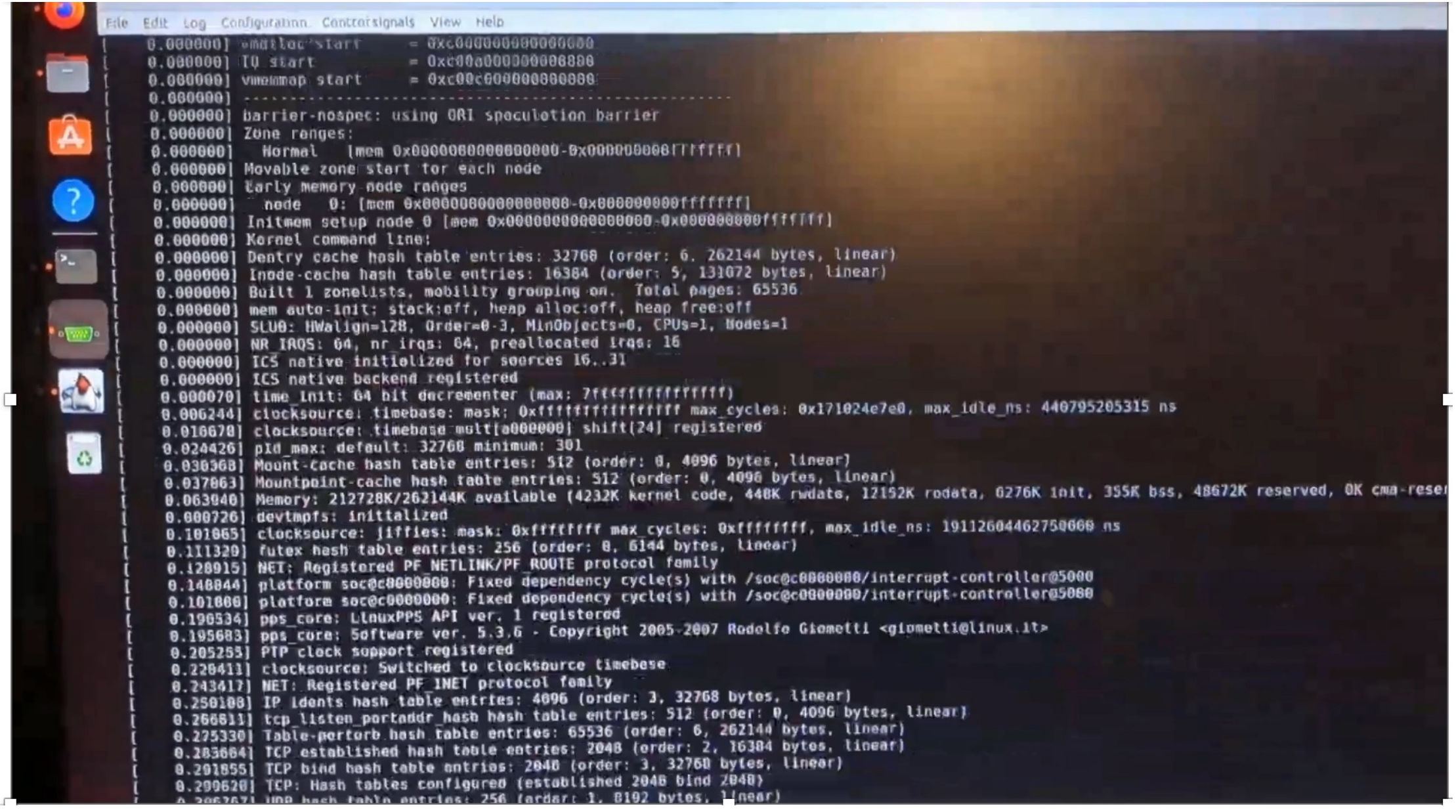
Boot Linux


Soc signature: f00daa5500010001
Soc features: UART DRAM SPIFLASH ETHERNET
DRAM: 256 MB
DRAM INIT: 0 KB
CLK: 100 MHz
SPI FLASH ID: 012018 Cypress/Spansion (CF1=02) [quad IO mode]
SPI FLASH OFF: 0x400000 bytes
LiteDRAM built from LiteX 87137c30
Initializing SDRAM @0x40000000...
Switching SDRAM to software control.
Read leveling:
m0, b00: |00000000000000000000000000000000| delays: -
m0, b01: |00000000000000000000000000000000| delays: -
m0, b02: |11111111111111100000000000000000| delays: 06+-06
m0, b03: |00000000000000011111111111111000| delays: 22+-06
m0, b04: |00000000000000000000000000000000| delays: -
m0, b05: |00000000000000000000000000000000| delays: -
m0, b06: |00000000000000000000000000000000| delays: -
m0, b07: |00000000000000000000000000000000| delays: -
best: m0, b02 delays: 06+-06
m1, b00: |00000000000000000000000000000000| delays: -
m1, b01: |00000000000000000000000000000000| delays: -
m1, b02: |11111111111111100000000000000000| delays: 06+-06
m1, b03: |00000000000000011111111111111000| delays: 22+-06
m1, b04: |00000000000000000000000000000000| delays: -
m1, b05: |00000000000000000000000000000000| delays: -
m1, b06: |00000000000000000000000000000000| delays: -
m1, b07: |00000000000000000000000000000000| delays: -
best: m1, b02 delays: 06+-06
Switching SDRAM to hardware control.
Memtest at 0x40000000 (2.0MiB)...
Write: 0x40000000-0x40200000 2.0MiB
Read: 0x40000000-0x40200000 2.0MiB
Memtest OK
Memspeed at 0x40000000 (Sequential, 2.0MiB)...
Write speed: 362.1MiB/s
Read speed: 218.5MiB/s
Trying flash...
Copy segment 0 (0x4fb2f2 bytes) to 0x1700000
Booting from DRAM at 1700000

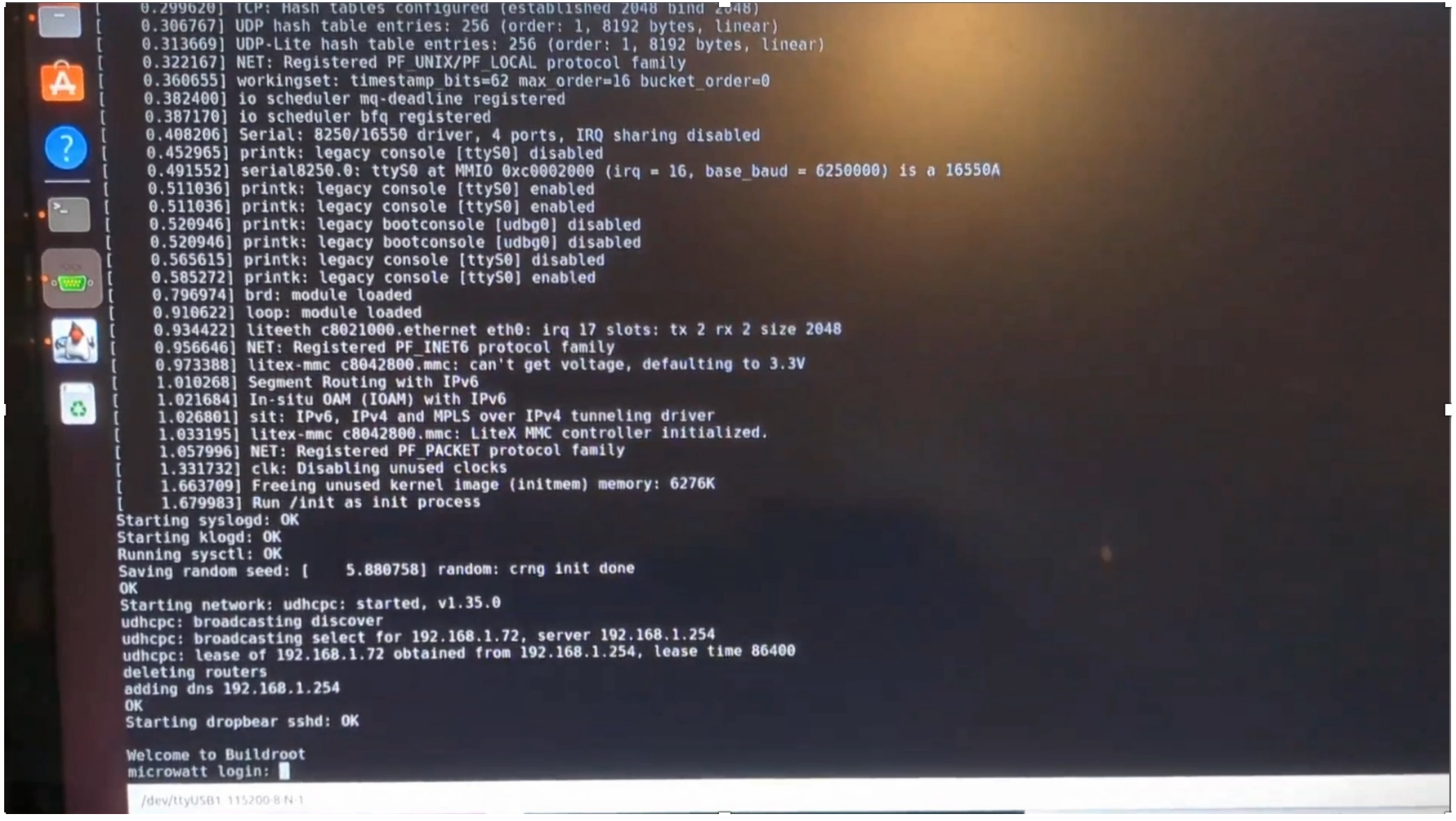
Boot Linux (cont.)



Boot Linux (cont.)

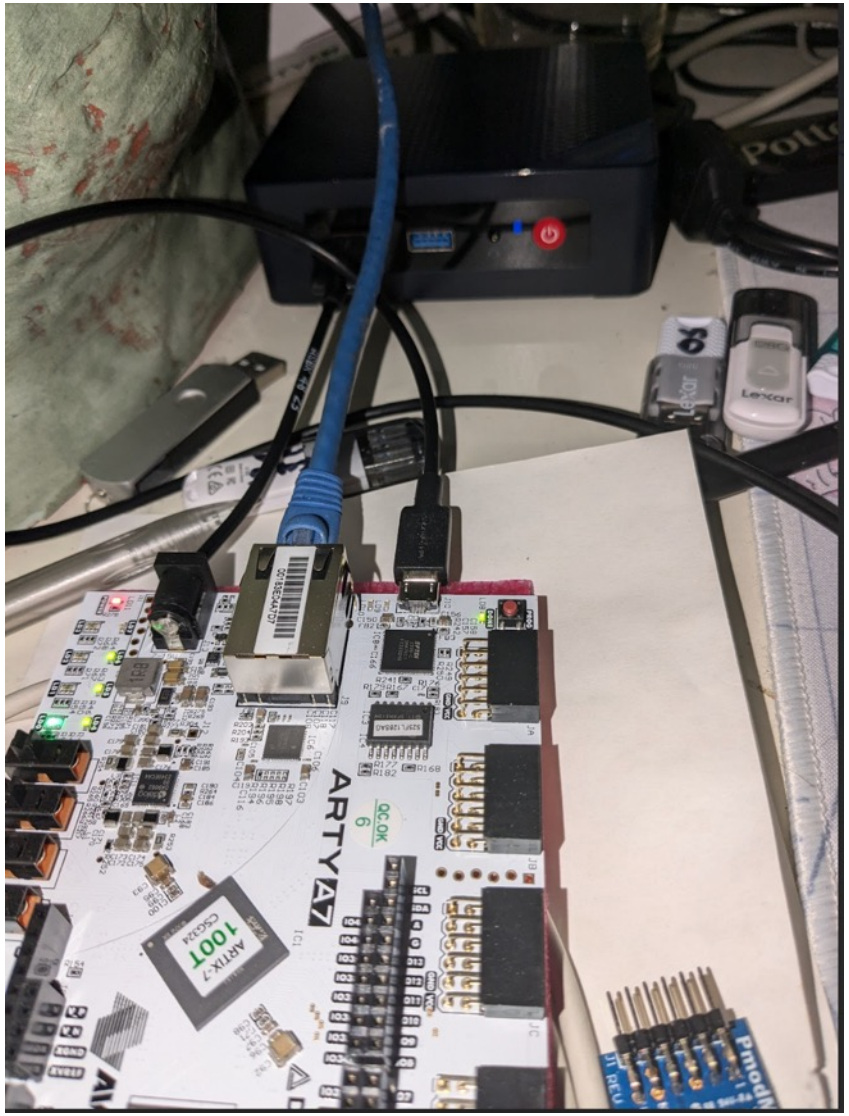


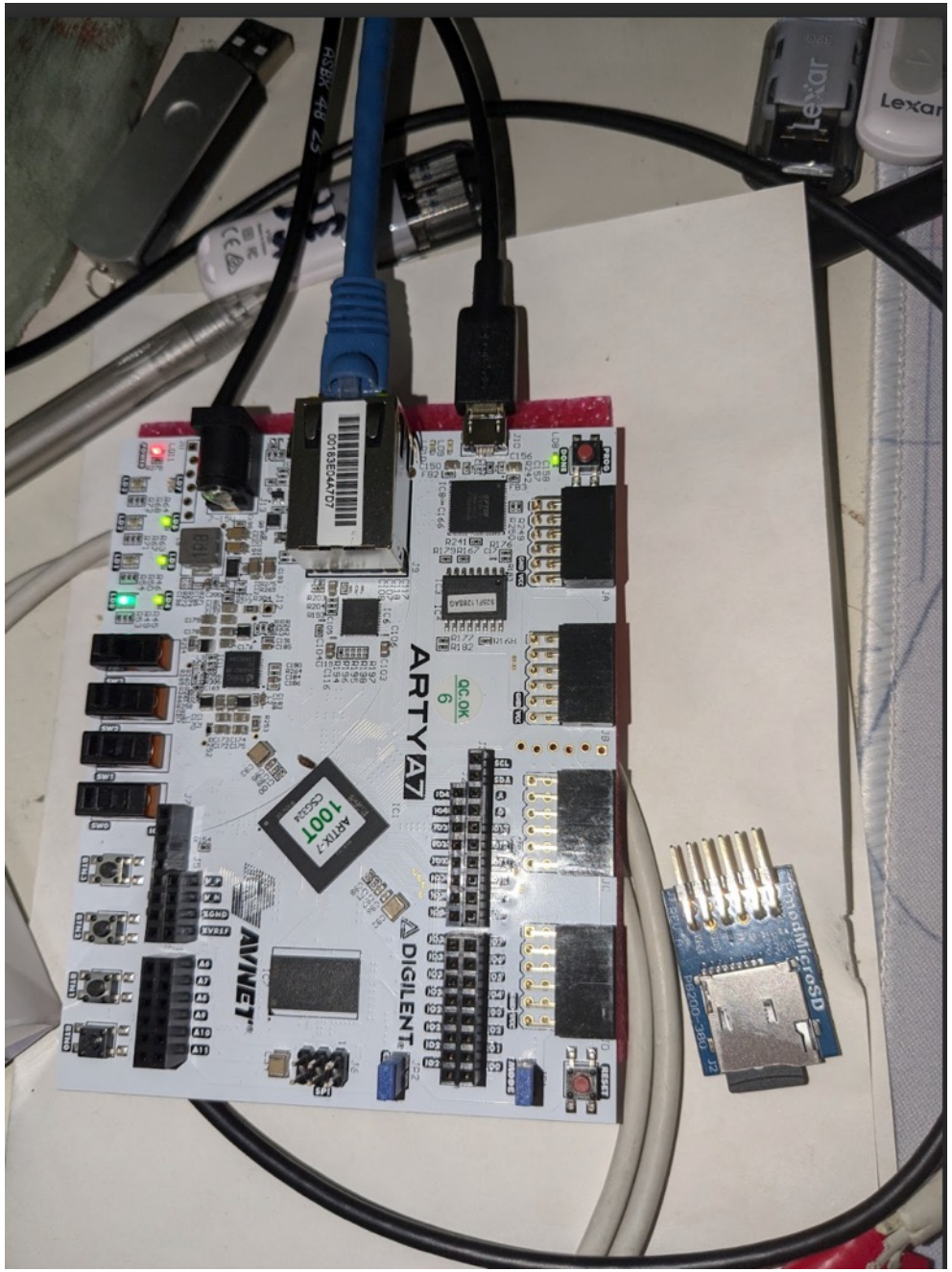
Boot Linux


[0.299620] TCP: Hash tables configured (established 2048 bind 2048)
[0.306767] UDP hash table entries: 256 (order: 1, 8192 bytes, linear)
[0.313669] UDP-Lite hash table entries: 256 (order: 1, 8192 bytes, linear)
[0.322167] NET: Registered PF_UNIX/PF_LOCAL protocol family
[0.360655] workingset: timestamp_bits=62 max_order=16 bucket_order=0
[0.382400] io scheduler mq-deadline registered
[0.387170] io scheduler bfq registered
[0.408206] Serial: 8250/16550 driver, 4 ports, IRQ sharing disabled
[0.452965] printk: legacy console [ttyS0] disabled
[0.491552] serial8250.0: ttyS0 at MMIO 0xc0002000 (irq = 16, base_baud = 6250000) is a 16550A
[0.511036] printk: legacy console [ttyS0] enabled
[0.511036] printk: legacy console [ttyS0] enabled
[0.520946] printk: legacy bootconsole [udbg0] disabled
[0.520946] printk: legacy bootconsole [udbg0] disabled
[0.565615] printk: legacy console [ttyS0] disabled
[0.585272] printk: legacy console [ttyS0] enabled
[0.796974] brd: module loaded
[0.910622] loop: module loaded
[0.934422] liteeth c8021000.ethernet eth0: irq 17 slots: tx 2 rx 2 size 2048
[0.956646] NET: Registered PF_INET6 protocol family
[0.973388] litex-mmc c8042800.mmc: can't get voltage, defaulting to 3.3V
[1.010268] Segment Routing with IPv6
[1.021684] In-situ OAM (IOAM) with IPv6
[1.026801] sit: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[1.033195] litex-mmc c8042800.mmc: LiteX MMC controller initialized.
[1.057996] NET: Registered PF_PACKET protocol family
[1.331732] clk: Disabling unused clocks
[1.663709] Freeing unused kernel image (initmem) memory: 6276K
[1.679983] Run /init as init process
Starting syslogd: OK
Starting klogd: OK
Running sysctl: OK
Saving random seed: [5.880758] random: crng init done
OK
Starting network: udhcpc: started, v1.35.0
udhcpc: broadcasting discover
udhcpc: broadcasting select for 192.168.1.72, server 192.168.1.254
udhcpc: lease of 192.168.1.72 obtained from 192.168.1.254, lease time 86400
deleting routers
adding dns 192.168.1.254
OK
Starting dropbear sshd: OK
Welcome to Buildroot
microwatt login:
/dev/ttyUSB1 115200 8 N 1

Boot Linux

Pictures





No power cable … board is powered through USB

Using Arty as an Edge Device

If you want to use Arty in an edge type environment, you will likely want to enable remote access over the Ethernet. To do this you'll need to connect your Arty to an Ethernet router.

```
In your gtkterm terminal after the system boots you should see

Welcome to Buildroot
microwatt login: ( enter "root" – without quotes )
# passwd root
# udhcpc -i eth0
```

If you look at the output while Linux is booting you'll see the IP address (lease obtained for ...)
Connect to Arty from a network connected device

ssh root@x.y.z.u

Using Arty as an Edge Device


hofstee — hofstee@localhost: ~ — ssh root@192.168.1.72 — 126×31
hofstee@Hs-MacBook-Pro ~ % ssh root@192.168.1.72
@@
@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @
@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ED25519 key sent by the remote host is
SHA256:Vma1aunP4XzmBvXdSlSaVKfvT/67SD1QJm4I96xscoo.
Please contact your system administrator.
Add correct host key in /Users/hofstee/.ssh/known_hosts to get rid of this message.
Offending ED25519 key in /Users/hofstee/.ssh/known_hosts:25
Host key for 192.168.1.72 has changed and you have requested strict checking.
Host key verification failed.
hofstee@Hs-MacBook-Pro ~ % sudo vi ~/.ssh/known_hosts
hofstee@Hs-MacBook-Pro ~ % ssh root@192.168.1.72
The authenticity of host '192.168.1.72 (192.168.1.72)' can't be established.
ED25519 key fingerprint is SHA256:Vma1aunP4XzmBvXdSlSaVKfvT/67SD1QJm4I96xscoo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.72' (ED25519) to the list of known hosts.
root@192.168.1.72's password:
ls
hello
#

Still to come …

- Adding a MicroSD to Arty
  - Support should already be there, will update the github page with detailed instructions once it works
- Example of using MicroWatt as a microcontroller (for something on or attached to the board)
- Example of adding a custom instruction to MicroWatt
- **Instructions on how to prepare the MicroWatt SoC for tapeout to a semiconductor fab (or two)**