ANALOG DEVICES

AHEAD OF WHAT'S POSSIBLE™

Embedded Linux Workshop

analog.com

Booklet

Pre-requisites

Common Linux commands:

- ***cd***: used to change the current working directory in Linux and other Unix-like operating systems
- ***ls***: lists files and directories within the file system, and shows detailed information about them
- ***cat***: displays the contents of one or multiple text files
- ***echo***: prints out its arguments as standard output; use *>* or *>>* with the *echo* command to print the output to a file instead of displaying it in the terminal
- ***mv***: moves a file from one location to another, rename a file with or without moving it

The # sign indicates that the terminal is ready to accept commands.

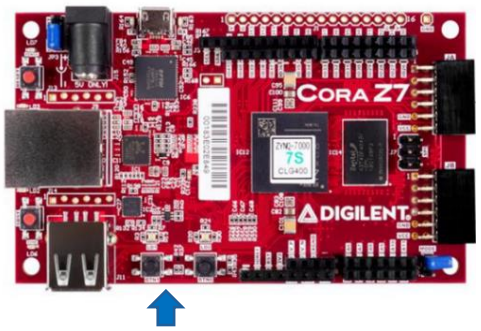
! Make sure NOT to include it in the command you are typing

Workshop Preparation

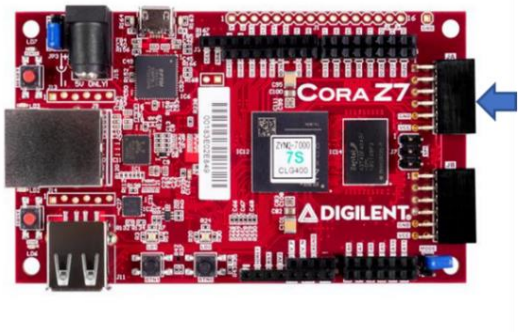
Make sure the Raspberry Pi 5 in front of you is booted and is connected to the monitor, mouse and keyboard. The monitor should show this:

Hardware setup for CoraZ7S:

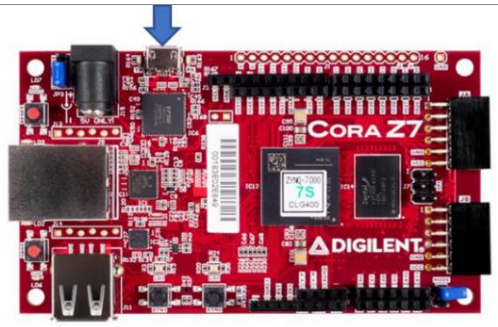
Insert the SD card prepared in the SD port of CoraZ7s.



Connect the ADXL355 to the JA port of the CoraZ7s.



Connect the UART port of the CoraZ7s to a USB port of your Raspberry Pi 5 using the USB cable received.



Connect the Ethernet port of the CoraZ7s to the Ethernet port of your Raspberry Pi 5.

CoraZ7S Environment Exploration:

Steps

1. On your Raspberry Pi 5, search for *Terminal* and open it:

2. Write the following command to discover the available IIO devices:

```
iio_info -s
```

Look for the IP address correspondent to *workshop-cora* machine. You will need it for the next steps. To find it easier you can search by the hostname of the carrier:

```
iio_info -s | grep workshop-cora
```

The output should look like this:

```
0: 169.254.35.68 (xadc) [ip:workshop-cora.local]
```

3. Write the following command to connect via ssh to the CoraZ7s (replace the IP address in the command with the address of your board):

```
ssh -X root@169.254.35.68
```

The password is *analog*.

At this point you should see this in your terminal:

```
analog@workshops:~$ ssh root@169.254.35.68
root@169.254.35.68's password:
```

```
Linux analog 6.1.70-35308-ge2e62cc28c80 #1525 SMP Fri Aug 15 07:04:44 EEST 2025
armhf

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Oct 24 19:11:04 2025 from 169.254.35.68
root@workshop-cora:~#
```

You are now connected to the CoraZ7s. Every command you run from this point forward will be executed on the CoraZ7s, not your workstation.

Hands-on Activity

Exercise 1 - Turn on a LED and make it blink with a heartbeat rhythm

Steps

1. Move to the LEDs directory using the command below:

```
# cd /sys/class/leds
```

Here you can see 6 directories corresponding to the 3 colors of each RGB LED on the board:

```
# ls
```

Each one of them contains the following attributes:

brightness	stores the status of the LED possible values: *0, 1*
trigger	changes the status of the LED
max_brightness	indicates the max brightness value that can be written to *brightness* attribute
device	a symbolic link pointing to the parent device of the LED

They may contain other folders and files, depending on the settings.

The important features of the *trigger* function are:

none	the default state where no specific trigger is set for the LED
timer	this trigger causes the LED to blink at a specified rate set in *delay_on* and *delay_off* attributes
oneshot	turns the LED on for a short period and then turns it off; the period is set in *oneshot*
heartbeat	causes the LED to blink in a pattern that mimics a heartbeat

cpu	links the LED activity to CPU activity

The *delay_on* and *delay_off* attributes are active when the timer features are enabled, and it will appear in the LED folder

2. Turn on the RED color of the first LED by setting the corresponding *brightness* to 1:

```
# echo 1 >> led0_red/brightness
```

3. Make the red color of the first led blink in heartbeat mode:

```
# echo heartbeat >> led0_red/trigger
```

 **CHALLENGE**: make the first LED light continuously purple and the second LED light green for 1000 ms OFF and for 500 ms ON.

HINT: for the first LED combine two colors to get purple and for the second LED use the *timer* trigger and set *delay_on* and *delay_off.*

Exercise 2 - Run a script

This exercise will demonstrate how to execute a series of commands within a script, rather than running each command individually in the terminal. You will also learn how any Linux script can be turned into a bash command

```
# bash /leds.sh
```

Now make the script available to be run from anywhere in the file system and run it again:

```
# chmod +x /leds.sh
# mv /leds.sh /usr/local/bin
# leds.sh
```

Exercise 3 – Build a kernel object for ADXL355 driver

This exercise will demonstrate how to create a kernel object for an existing driver and load it in such a way that the system recognizes it as part of the kernel.

By running the command *"iio_info"* in the terminal you will see in the output that there is no *adxl355* device found.

Steps

1. Move to the *adxl355* folder:

```
# cd /adxl355
```

2. Create a Makefile:

```
# touch Makefile
```

3. Open the Makefile with a file editor of your choice (nano, Vim):

```
# nano Makefile
```

4. Write to the Makefile the following lines and then save and exit the file with CRTL+S, then CTRL+X (pay attention to the indentation):

```
obj-m := adxl355_spi.o

all:
	make -C /usr/src/linux-headers-6.1.0/ M=$(shell pwd) modules

clean:
	make -C /usr/src/linux-headers-6.1.0/ M=$(shell pwd) modules
```

5. Run *make* to build the kernel object:

```
# make
```

6. Load the new kernel object:

```
# insmod adxl355_spi.ko
```

7. Check again with "*iio_info*" to see readings from the accelerometer.

Exercise 4 – Compute the temperature value from sysfs

The task is allowing you to read the content of attributes files and compute the temperature given by the on-board temperature sensor.

The temperature is computed using the following formula:

$$TEMPERATURE = (RAW + OFFSET) * SCALE$$

1. Move to the iio:device folder:

```
# cd /sys/bus/iio/devices/iio:device1
```

2. Here you can see all the attributes of the adxl355:

```
# ls
```

3. Put temperature values in variables:

```
# SCALE=$(cat in_temp_scale)
# OFFSET=$(cat in_temp_offset)
# RAW=$(cat in_temp_raw)
```

4. Compute the temperature and display it in the terminal (keep in mind that it will be in millidegrees Celsius):

```
# echo "($RAW + $OFFSET) * $SCALE" | bc
```

Exercise 5 – Play a Snake game with the accelerometer

Using the accelerometer module, you will have the chance to play a little oldie but goldie SNAKE game.

Run the game controls in the background:

```
# python3 /game.py &
```

Start the Snake game:

```
# /usr/games/nsnake
```

Move the accelerometer board around and observe the output. Be careful not to disconnect the wires.

Have fun!