WIP ArC TWO Start-Up, Usage, and Programming WIP

IMPORTANT NOTICE

Any user who wants to use the ArCTWO from the "Steghorn" computer will need an institute account and must be part of the "dialout" group. If you request a new account for a student, be sure to ask Mateo or Adam to include them in the "dialout" group. Otherwise, the USB drivers won't be available.

Documentation:

ArC TWO User's guide — ArC TWO documentation – Documentation for a general approach. Installation, system specifications, daughterboard descriptions and protocols.

ArC TWO Control Panel — ArC TWO Control Panel 0.1.1-a0 documentation - ArC TWO Control Panel documentation. User guide and built-in modules. Emodules, H5 data storage, Channel Mapping, and UI Signals.

libarc2 Python bindings — pyarc2 0.3.2-a1 documentation – Documentation for the pyarc2 python module. Overview and API reference.

libarc2 - Rust – Documentation for the low-level library libarc2. Opening this is like looking into the abyss, but it has some useful information.

First-time installation for new users:

Log in to the "steghorn" computer and turn on the arctwo with the switch. **NOTE: The ArCTWO system tends to run quite hot. Please do not forget to turn it off after working with it.**

Open the console and create a conda environment with all the dependencies:

```
$ conda env create --name arctwoenv --file /scratch/arctwoenv.yml
```

Activate the conda environment you just created

```
$ conda activate arctwoenv
```

If the previous step gives an error, try:

```
$ conda init –all
```

Close and reopen the console and try activating again.

You need to manually install the ArcTwo modules because the latest dev versions are not available in pip:

```
$ python -m pip install pyarc2 git+https://github.com/arc-instruments/arc2control
```

Everything should be set now. Try to start up the GUI:

```
$ python -m arc2control
```

A window should open with a prompt, choose the default 32x32 setup and press OK. The GUI should open. On the top left, click the "chip" button to manage firmware. Press the green button with arrows to load the list. Select the most recent firmware (top of the list), press the blue button to download it, and close the window. There should be a "0" next to the "Connect ArC2" button, which means it is correctly detected. Press Connect. If it turns green, you are good to proceed. If you have devices connected, try using the “Read All” function. You should see them on the map.

Now that the arctwo is running, we will move on to more elevated programming. There are two libraries designed to interact with the ArcTwo. The low-level Rust-based “libarc2”, and its wrapper “pyarc2”. We will generally work with pyarc2, which arc2control is based on, but libarc2 might be needed for some high-level applications that need access to the backdoor.

It is recommended to use Visual Studio and a Jupyter notebook to work on the arctwo. On the Steghorn computer, there are notebooks available in the “Scratch” folder. DO NOT EDIT THEM. Make a new folder for yourself in the Scratch or your home directory and copy the notebooks, because you will need to change the address of the firmware.

Open visual studio by typing “code” into the console and select as workspace the copied folder you made for yourself. Open the “XXX” notebook file. First, the ArcTwo must connect to the Python kernel. The first module will do so. While connecting, the firmware to be used has to be specified, so you will need to set the correct DIRECTORY to it by editing the [USERNAME]. You only need to run this section once per notebook and session. If successful, it should print a “0”.

NOTE: Before running this section you must set the Python Kernel to use the “arctwoenv” we created before. If it is your first time on vscode, it will ask you to install a Python interpreter before, please do so. The environment we created has all the dependencies you should need, but you can manually install additional python modules to code with them.

“

```
from pyarc2 import Instrument, find_ids, BiasOrder

import numpy as np

ids = find_ids()

print(ids)


if len(ids) == 0:

    print('no arc2')


# fw.bin is the firmware to load on ArC TWO, as of right now, 20241918 is the latest

arc = Instrument(ids[0],
'/home/[USERNAME]/.local/share/arc2control/firmware/efm03_20240918.bin')
```

“

After initializing, be aware that the GUI cannot be opened. If desired, restart the Python kernel and then connect to the GUI through the usual means.

If you reached this point you have successfully completed the setting-up of the ArCTWO. You may now start working on it either with the GUI or with vscode.

Opening the ArCTWO GUI:

Login and open console

```
$ conda activate arctwoenv

$ python -m arc2control
```

Choose the experiment setup and press OK. Connect to ArC Two.

PRO Tip: You can work remotely on the Arctwo with the GUI by using an X11 tunnel. Connect to the computer with an appropriate software (like MobaXterm) and start-up arc2control with the -X argument:

```
$ python -m arc2control -X
```

Measuring with the ArCTWO and Python:

After we have connected to the system (remember the tutorial), we can start measuring the DUTs. To be able to measure with Python, we need to know how the connections of the daughter board being used are physically mapped to the chipset, because we need to work with the indexes of the connectors. The daughterboard interface — ArC TWO documentation

We can easily read the current between any two physical lines with the arc.read_one(LowV_ID1,HighV_ID2,ReadV) command, and display, save, or transform as needed. This example command will send a read voltage of 0.1 V between lines 14 and 13 and measure the current through them. Then, we transform the current into a resistance and print it in kOhm:

"

```
read_V=0.1

current=arc.read_one(14,13,read_V)

resistance=abs(read_V/current)

resistance/1e3
```

"

We can loop the read_one command to read the I through multiple devices, and then display as an array or matrix using a tool such as Numpy. In this example we create a 2x2 matrix and read between lines 13,49; 13,50;14,49; and 14,50. We then display the resistance values.

"

```
resistances=np.zeros((2,2))

for i,idx in enumerate([13,14]):

  print(i,idx)

  for j,idz in enumerate([49,50]):

    current=arc.read_one(idx,idz,read_V)

    resistances[i,j]=abs(read_V/current)
```

resistances

“

Looping the read_one command will induce a Python-sided delay. Each read takes around 5ms, so reading a full 32x32 array (1024 addresses) will take around 5 seconds. The read_all command reads the full default 32x32 array on the hardware side, so it is one order of magnitude faster. This will read every address between “bit” and “word” lines. The internal definition of bit and word is done via a mapping that indicates the physical addresses to be used for each type. The read_all uses the default 32x32 definition. The Channel Mapper — ArC TWO Control Panel 0.1.1-a0 documentation

“

currents=arc.read_all(read_V, BiasOrder.Rows)

resistances=np.abs(read_V/currents)

“

Connect to an institute computer

You will need one to four pieces of software if on Windows: An SSH client (like PuTTY) to connect remotely to the computer, a VNC program (like RealVNC Viewer) to open a remote desktop and navigate, an SCP program (like WinSCP) to transfer files to the computer storage, and an X server for advanced functionalities (like MobaXterm). Note: MobaXterm alone already provides full SSH and SCP functionality albeit it is heavier and more difficult to use than PuTTY and WinSCP. If on MAC, look for alternative software for all provided examples.

Open SSH and connect to the computer with [name].ee.ethz.ch. Login with your institute username and password. You may now send commands to the computer and work virtually on the Linux machine.

If you want to access the graphical desktop, you need to start a VNC server in the Linux machine. In the console:

```
$ vncserver
```

This will start a server on the Linux machine. Ignore the error it will print and take note of the number of the connection "[name].ee.ethz.ch:[number]" in the console. Go to your VNC software and connect to "[name].ee.ethz.ch:[number]" using the institute password.

PRO Tip: before you log in, go to Properties, Expert, and change "ColorLevel" to "full" and "Quality" to "high", otherwise it will look like shit.