ECE 379K: Machine Learning and Data Analytics for Edge AI
Appendix A2

A2.1. Connecting to Odroid MC1 - Windows

Part 1. Remote Desktop Connection

1) Connect to VPN using Cisco AnyConnect
2) In the search bar on the bottom left search for ***Remote Desktop Connection***
3) After you open ***Remote Desktop Connection***, click on ***Show Options***
4) Fill in the following
    - Computer: **`sld.ece.utexas.edu`**
    - User name: **`austin\<your_eid>`**
5) Click the ***Local Resources*** tab
6) Under ***Local devices and resources*** click ***More***
7) Select the drives you want to share location between the remote connection and your computer
8) Then click ***Connect***
9) Input your UT EID password
10) Accept any certificate warning that may appear

WARNING: You will have to wait a few minutes until your remote desktop is being set up.

Part 2. Connecting to Odroid MC1

1) Inside the remote desktop you can access files and under ***My Computer*** you can find your shared folder from your own computer.
2) Transfer any files necessary for deployment on the remote desktop
3) Open a terminal
4) Use ***scp*** to transfer the files you want to the edge device
5) Next, you can SSH into the edge devices and use them:

```
ssh student@<IP address of your MC1>
```

IMPORTANT: Write the code on your computer, transfer the necessary files to the remote desktop and only then "scp" them to the edge devices for actual deployment. The remote desktop serves only as a gateway to access the devices.

A2.2. Connecting to Odroid MC1 - Mac

Part 1. Microsoft Remote Desktop

1) Search in the Apple Store ***Microsoft Remote Desktop*** and install it
2) Connect to VPN using Cisco AnyConnect
3) Open Microsoft Remote Desktop and add a new pc
4) Fill in PC name: **`sld.ece.utexas.edu`**
5) Click on ***User account*** and ***Add User Account…***
6) Fill in the following:
    - Username: **`<your_eid>`**
    - Password: **`<your_UT_EID_password>`**
    - Domain: **`AUSTIN`**
7) Click ***Add***
8) Go to Folders tab and enable ***Redirect folders***

9) On the bottom left click on + and select the folders you want to share with the remote desktop
10) Click ***Save*** and double click the new connection

WARNING: You will have to wait a few minutes until your remote desktop is being set up.

Part 2. Connecting to Odroid MC1

Identical to **Appendix A2.1, Part 2**

A2.3. Connecting to Odroid MC1 - Linux

Part 1. Remmina

1) After installing Remmina, open it
2) Connect to VPN using Cisco AnyConnect
3) On the upper left corner click +
4) Click ***Protocol*** and select ***Remote Desktop Protocol***
5) Fill in
    - Server: **`sld.ece.utexas.edu`**
    - User name: **`<your_eid>`**
    - User password: **`<your_UT_EID_password>`**
    - Domain: **`AUSTIN`**
6) Turn on **Share folder** and select the folder you want to share with your remote desktop
7) Click ***Save and Connect***

WARNING: You will have to wait a few minutes until your remote desktop is being set up.

Part 2. Connecting to Odroid MC1

Identical to **Appendix A2.1, Part 2**

A2.4. Core Affinity for Multithreaded Applications

Some benchmarks in this assignment may use all cores available on the MC1 by default; when required, you should **use `taskset` to constrain their execution to the big cores**.

Each process and thread in Linux has a *core affinity* – the set of processors on which that task can be scheduled. The CPU affinity can be set in-program using libraries or at runtime by using the command:

```
taskset --all-tasks <hexadecimal core mask> <command>
```

NOTE: Cores in the system are numbered from 0 to 7. Cores 0 through 3 are LITTLE cores and cores 4 through 7 are big cores (see ***Table 1***). For example:

```
taskset --all-tasks 0x3 ./benchmark.out
```

will run a benchmark on cores zero and one, which are the first two cores in the LITTLE cluster. Using a mask of 0xAA will run a benchmark on cores 1, 3, 5, and 7. **Figure 2** shows how two example hexadecimal bitmasks map to the cores in each cluster. Be careful that the cores are in the 7,6,5,4,3,2,1,0 order! If you want to run on big core 6 you will use the mask 0x40=0100 0000

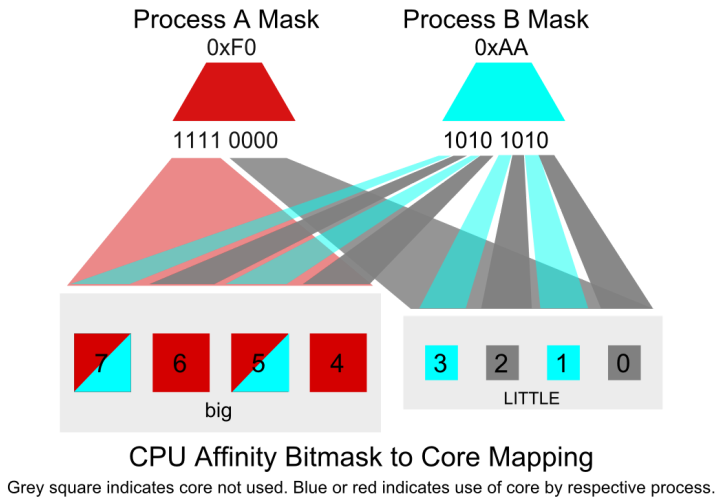

Figure 2. Diagram of two CPU affinity bitmask to core mappings

You can use `htop` to visualize and verify the resource utilization of each core on the MC1 as you run the benchmarks. By default, `htop` shows threads of a program as separate entries.

A2.5. Running the benchmarks

To run the benchmarks on the MC1 check the ***HW2_files*** on the device. The ***blackscholes*** and ***bodytrack*** benchmarks are in the ***HW2_files/parsec_files*** folder.

1) To run ***blackscholes*** go in the ***HW2_files/parsec_files*** folder and run:

```
taskset --all-tasks <affinity_mask> ./blackscholes <number_of_threads> in_10M_blackscholes.txt
                                    <output_name>
```

2) To run ***bodytrack*** go in the ***HW2_files/parsec_files*** folder and run:

```
taskset --all-tasks <affinity_mask> ./bodytrack ./sequenceB_261 4 260 3000 8 3
                                    <number_of_threads> 0
```

3) To run ***TPBench*** go in the ***HW2_files*** folder and run:

```
taskset --all-tasks <affinity_mask> ./TPBench.exe
```

A2.6. Files Transfer Between Local Computer and Edge Devices

To transfer files between your local computer and the MC1, you can use the `scp` command. The general format is:

```
scp <sender address>:<path to send the file> <receiver address>:<path to receive the file>
```

For example, to transfer a file from your local computer to the MC1, you can open a terminal on your computer and type:

```
scp <local path to send the file> student@<Your MC1 IP>:<MC1 path to receive the file>
```

Similarly, if you want to receive a file from the MC1 board, open a terminal on your computer and type:

```
scp student@<Your MC1 IP>:<MC1 path to send the file> <local path to receive the file>
```

A2.7. Checking Connection to Smart Power 2 (SP2)

To gather power readings, you need to access a Smart Power 2 (SP2) device for power monitoring to remotely measure the power of the entire MC1 device. You will access this data from the SP2 over a telnet connection. The MC1 devices are configured to automatically connect to the SP2 wireless link.

Let's suppose you are logged in Odroid MC1 number 01 which has the name **sld-mc1-01**. **Always check the connection status** to the corresponding SP2 (i.e., **sp2-mc1-01**) run:

```
nmcli con
```

The output should be something like:

```
NAME        UUID                  TYPE   DEVICE
Wired connection 1 f059cda3-e7ad-339a-9861-3dc37dc47aeb ethernet eth0
sp2-mc1-01     96c01024-e194-40ab-9e75-579491e05643 wifi   wlan0
```

If instead of `wlan0` you see `--` you will have to manually initiate the connection:

```
sudo nmcli con up sp2-mc1-01
```

Check again by typing

```
nmcli con
```

and also run

```
ip a
```

as a double check to see if you see the right IP address (highlighted below):

```
 wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1280 qdisc mq state UP group default
qlen 1000
  link/ether 70:f1:1c:4c:94:bd brd ff:ff:ff:ff:ff:ff
  inet 192.168.4.100/24 brd 192.168.4.255 scope global dynamic noprefixroute wlan0
```

A2.8. Power Logging over Telnet and Serial on MC1

You can use telnet to connect to the SP2 board over its Wi-Fi connection at the local IP address by executing the following command in terminal:

```
telnet 192.168.4.1
```

To exit `telnet` press **`CTRL+]`** and then type **`close`** to close the connection. The SP2 ***output format*** is Voltage [V], Current [A], Power [W], Energy [W/h]. **For Python development**, you can use the *telnetlib* library instead.