Manual and quick start
2016-2017

Master's thesis
Exploration of the TI AM5728 Audio/Video Subsystem

Dennis Joosens

1 Abstract

This report is a quick start and contains a concise manual to set up the most important executed measurements.

Contents

2 Hardware

A concise overview of the needed hardware.

- 2 x Texas Instrument AM5728 Evaluation Module
- 3.5 mm jack male to 3.5 mm jack male stereo cable
- 3 x Ethernet Cat5e cables
- Gigabit Ethernet switch
- Virtual machine with the latest version of Linux Ubuntu installed on it

3 Setup virtual machine

For this project we have set up a Virtual Machine that runs Linux Ubuntu 16.04 LTS. Make sure that the VM has a static IP configured and that Minicom is installed. You also need SCP but this should be installed by default.

```
# sudo apt-get install minicom
```

To find the name of our serial port:

```
# dmesg | grep tty
```

In our case the name is ttyUSB0. Now we need to change the setup configuration of Minicom.

```
# minicom -s
```

From here go to serial port setup and make sure the **Serial Device name** is **/dev/ttyUSB0**. Eventually save this setup as default.

Now select **exit** and the serial connection to the board starts. If all goes well, we get a terminal login.

3.1 Setup static IP on the EVM board

The communication with the EVM board happens using a serial connection at first. We have setup Minicom so we can now set a static IP on the board and eventually connect the VM and the EVM boards to a network switch.

Note: The IP range can be chosen freely as long as you stay in the same range for all the configurations.

We have chosen the following IP's for the setup:

```
EVM board 1: 192.168.10.10
EVM board 2: 192.168.10.11
         VM: 192.168.10.20
```

We can now configure a static IP by changing the following file:

```
# nano /etc/network/interfaces
```

Add the following to the file:

```
auto lo eth0
iface eth0 inet static
        address 192.168.10.10
        netmask 255.255.255.0
        gateway 192.168.10.1
```

A reboot might be needed.

```
# reboot
```

3.2 Cross compiling

GCC is by default installed on most Linux machines. However to compile code for the ARM architecture we need to download the SDK and install it. The **Processor SDK** is available from the website of Texas Instruments.

```
# opt/ti-processor-sdk-linux-am57xx-evm-03.02.00.05/
linux-devkit/sysroots/x86_64-arago-linux/usr/bin/
arm-linux-gnueabihf-gcc <file.c> -o <file>
```

You can add the full path to the **$PATH** environmental variable.

```
# nano /etc/environment
```

To check if it is successfully added.

```
# echo $PATH
```

3.3 Setup SCP

To transfer a cross compiled file to an EVM board, execute the following command. Note that the IP address can be changed to the IP address of the according EVM board.

```
$ scp <my_file> root@192.168.10.10:/home/root
```

The specified file will be placed at the /home/root directory on the EVM board.

4 Minimal audio latency measurement

For this measurement we need to use the code in the folder /portfolio/code/audio/. The cross compiled code is also available in the folder.

Compile the file.

```
# arm-linux-gnueabihf-gcc latency.edited.c
-o latency.edited -lasound
```

Transfer the file using SCP to an EVM board and execute it.

```
# ./latency.edited
```

By executing the following command with parameters, we run the program for 20 seconds, set the lowest frame size and use polling.

```
# ./latency.edited -m 256 -p -s 20
```

By running the following command, you can see all the command line arguments that can be used.

```
# ./latency.edited -h
```

Note: The same procedure is valid for the latency.original.c file. However for cross compiling you need to add the **-lm** parameter.

4.1 Pink noise

To execute the following command you need the code of the following directory which can be found in the portfolio: pink_noise/. The cross compiled code can also be found here. Make sure that the ALSA library is available on the VM and execute the file in the following folder: alsa-lib-1.1.3/test/.

```
# arm-linux-gnueabihf-gcc pcm_min.c
-o pcm_min -lasound
```

Execute the file on the EVM board.

```
# ./pcm_min
```

To plot 100 samples we need to have gnuplot installed.

```
# apt-get install gnuplot-x11
```

To execute the following command you need the code of the following directory which can be found in the portfolio: pink_noise/.

```
# gcc pcm_min_data_plot.c -o pcm_min_data_plot
```

Execute the file on the VM.

```
# ./pcm_min_data_plot
```

A .PNG file is created in the directory where the file is executed.

5 Network latency measurements

5.1 TCP Round-Trip-Time

Note: To execute the following commands you need the code of the following directory which can be found in the portfolio: network/TCP/RTT/. The cross compiled code can also be found here. If you need to cross compile it again we refer to Section 3.2. No extra arguments are needed.

Copy the file to EVM board 1:

```
# scp tcpserver root@192.168.10.10:/home/root
```

Connect to EVM board 1:

```
# ssh root@192.168.10.10
```

Make sure you run the server before running the client.

```
# ./tcpserver
```

Copy the file to EVM board 2:

```
# scp tcpclient root@192.168.10.11:/home/root
```

Now connect to EVM board 2 from the VM:

```
# ssh root@192.168.10.11
```

Execute the following command:

```
# ./tcpclient
```

Note: In the tcpclient.c is a static IP defined (server's IP). If the IP range is changed, you also need to change it here.

5.2 UDP Round-Trip-Time

Note: To execute the following commands, you need the code of the following directory which can be found in the portfolio: network/UDP/RTT/. The cross compiled code can also be found here. If you need to cross compile it again we refer to Section 3.2. No extra arguments are needed.

Copy the file to EVM board 1:

```
# scp udpserver root@192.168.10.10:/home/root
```

Connect to EVM board 1:

```
# ssh root@192.168.10.10
```

Make sure you run the server before running the client.

```
# ./udpserver
```

Copy the file to EVM board 2:

```
# scp udpclient root@192.168.10.11:/home/root
```

Now connect to EVM board 2 from the VM:

```
# ssh root@192.168.10.11
```

Execute the following command:

```
# ./udpclient
```

Note: In the udpclient.c is a static IP defined (server's IP). If the IP range is changed, you also need to change it here.

5.3 RTP Round-Trip-Time

Note: To execute the following commands, you need the code of the following directory which can be found in the portfolio: network/RTP/RTT/. The cross compiled code can also be found here. If you need to cross compile it again we refer to Section 3.2. No extra arguments are needed.

Copy the file to EVM board 1:

```
# scp rtpserver root@192.168.10.10:/home/root
```

Connect to EVM board 1:

```
# ssh root@192.168.10.10
```

Make sure you run the server before running the client.

```
# ./rtpserver
```

Copy the file to EVM board 2:

```
# scp rtpclient root@192.168.10.11:/home/root
```

Now connect to EVM board 2 from the VM:

```
# ssh root@192.168.10.11
```

Execute the following command:

```
# ./rtpclient
```

Note: In the rtpclient.c is a static IP defined (server's IP). If the IP range is changed, you also need to change it here.

5.4 Audio transfer with latency file over UDP

Note: To execute the following commands you need the code of the following directory which can be found in the portfolio: network/UDP_with_latency/. The cross compiled code can also be found here. If you need to cross compile it again we refer to Section 3.2. Note that we need to add the "lasound" parameter.

Copy the file to EVM board 1:

```
# scp latency_udp_receiver root@192.168.10.11:/home/root
```

Connect to EVM board 1:

```
# ssh root@192.168.10.11
```

Make sure you run the server before running the client.

```
# ./latency_udp_receiver -m 256 -p
```

Copy the file to EVM board 2:

```
# scp latency_udp_sender root@192.168.10.10:/home/root
```

Now connect to EVM board 2 from the VM:

```
# ssh root@192.168.10.10
```

Execute the following command.

```
# ./latency_udp_sender -m 256 -p
```

Note: In the latency_udp_sender.c is a static IP defined (server's IP). If the IP range is changed, you also need to change it here. You also need to connect the two boards with an Ethernet switch and add a microphone to the sender and a speaker to receiver board.

5.5 Audio transfer with latency file over RTP

Note: To execute the following commands you need the code of the following directory which can be found in the portfolio: network/RTP_with_latency/. The cross compiled code can also be found here. If you need to cross compile it again we refer to Section 3.2. Note that we need to add the "lasound" parameter.

Copy the file to EVM board 1:

```
# scp latency_rtp_receiver root@192.168.10.11:/home/root
```

Connect to EVM board 1:

```
# ssh root@192.168.10.11
```

Make sure you run the server before running the client.

```
# ./latency_rtp_receiver
```

Copy the file to EVM board 2:

```
# scp latency_rtp_sender root@192.168.10.10:/home/root
```

Now connect to EVM board 2 from the VM:

```
# ssh root@192.168.10.10
```

Execute the following command.

```
# ./latency_rtp_sender
```

Note: In the latency_rtp_sender.c is a static IP defined (server's IP). If the IP range is changed, you also need to change it here. You also need to connect the two boards with an Ethernet switch and add a microphone to the sender and a speaker to receiver board.