Group 13 Project README

Welcome to the Group 13 project repository. This project involves working with Xilinx, and here are the details you need to get our project up and running.

Group Members
- Jack Ransome (zID: z5207250)
- Siyu Qiu (zID: z5348946)
- Yuchen Du (zID: z5319020)
- Zhengye Ma (zID: z5158505)

Project Details
- **Xilinx Version**: Xilinx Vivado 2021.1
- **Kria**: Kria KV260 Vision AI Starter Kit
- **I2S mic processor**
- **Arduino 2560 r3**

- **Project Description**:
This system is a hardware step sequencer that allows the user to arrange 4 different sounds recorded by the user on an 8 step timeline, which they may then export as a .wav file for listening. We have implemented this using a kria KV260 board, an arduino 2560 r3, and an I2S microphone, along with a collection of buttons, LEDs, and wires. All of the user input goes through the buttons connected to the arduino, which sends these button events to the kria board over ethernet. The kria board acts as the central hub of the system, deciding what to do based on button events from the arduino, sending new LED states to the arduino, recording sound from the I2s microphone over AXI, and saving all audio files to its onboard storage.

Setting up I2S and KV board
Wiring:
- 3.3V connected 3.3V (port 12)
- GND connected GND (port 10)
- BCLK connected to port 6
- DOUT connected to port 4
- LRCL connected to pin 2
- SEL connected to GND (port 9)

Setting up Arduino board with KV board
- Connect the arduino to the board with an ethernet cable

- Connect the arduino either to a power supply or to a PC using a USB cable

Using Putty for Serial Port Communication

To update files on the board, follow these steps:

1. Connect your board to your computer.
2. Open the Device Manager and navigate to Ports (COM & LPT).
3. Identify the serial port corresponding to your board.
4. Open Putty and select the Serial connection type.
5. In Putty, set the port to the one you identified in the Device Manager.
6. Change the speed to 112500, which is the normal and fastest speed for the board.
7. You will need a usb to transfer files(.dtbo and .bin) which are generated by Vivado from computer to the board.
8. Load the module in putty, after that if you would like to change the VHDL file in the future, you will only need to program the device on the Xilinx.

Now, Putty can be used to communicate with the board and update your files as needed.

Setting Up in WSL after changing code

Follow these steps to set up your environment in WSL:

1. **Clean Your Workspace:** make clean
2. **Start the SDK:** Run the command to activate the Petalinux SDK environment (. /opt/petalinux/2021.1/environment-setup-cortexa72-cortexa53-xilinx-linux depending on your install location)
3. **Build the Sample:** Build the `sample256` using the following command: make sample256
4. **Copy `sample256` to USB:** Once you've built `sample256`, you can copy it to a USB drive using your laptop.

Make sure your laptop recognizes the USB drive, and then simply copy the file to the USB drive as you normally would.

That's it! Your setup and build process are complete.

configuring petalinux to connect to the arduino over ethernet:

You must run the command “sudo ifconfig eth0 192.168.1.178 netmask 255.255.255.0” before attempting to connect to the arduino. This is because they must be on the same subnet to interact.

Testing methodology:

For milestone 4 we split the project into 2 halves, one being the hardware user interface side, and one being the audio manipulation side. These 2 halves interface via a single array in c code, which on the hardware side is manipulated by the buttons and displayed by the LEDs, and on the audio side is used as an input to the generation of a final .wav file. This allowed us to split the testing up and base it around interactions with this array. For the hardware, testing took place by setting up our C code to print the contents of the array, and then using the buttons to manipulate the array in a variety of different ways, along with choosing to display different rows of the array on the LEDs. For the audio side, we made a collection of different configurations of the array that tested all combinations of overlapping sounds and some edge cases such as a completely empty array and an array full of true values. Once these 2 halves were tested and joined together, we undertook further testing by using our system in the way it was intended, by recording lots of new sounds and exporting arrangements of them, which we verified the correctness of by playing out of our laptops.