Software Specification

1 Introduction

The communication between the host system and the Xillybus embedded Linux system (Xillinux) running on the Zedboard is over Ethernet. Different types of data need to be transmitted:

- The updated weights for the neural network from the host to the Zedboard, as long as the training is running on the host, as well as the output errors
- The 28x28 input images showing digits between 0 and 9

Further, this information needs to be processed and then sent from the ARM processor to the Field Programmable Gate Array (FPGA) over the Xillybus First In, First Out (FIFO) data streams.

A remote connection to the Xillinux system can be established over UART.

2 Top-Level Description

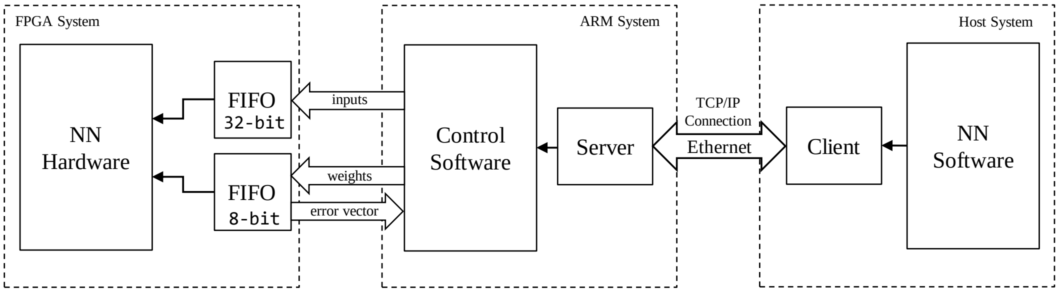

Figure 1: Top level description

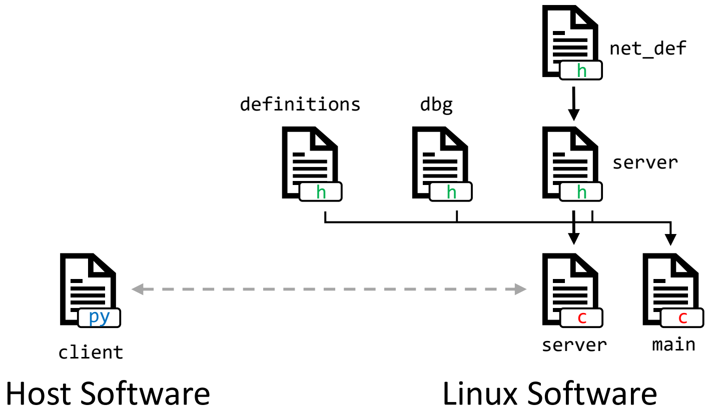

Figure 2: File tree for the software

3 File Tree

- `net_def.h` Contains definitions for networking, e.g. ports used.
- `dbg.h` Contains debugging macros for logging and error handling.
- `definitions.h` Contains information about the neural network, e.g. the number and type of Convolutional Neural Network (CNN) stages, layers in the fully connected network, input size and so on.
- `server.{c,h}` Handles the connection with the host software.
- `main.c` Contains the `main()` function with the main program loop that transmits and manages data to the hardware and from the host system.
- `client.py` Handles the connection with the client software.

4 Software Model

The software running on the host system is written in Python and the embedded software is written in C. The server software should run automatically on start-up.

4.1 Host

The host software written in Python essentially just reads the input images and sends them to the board. As long as training is happening on the host, it also receives error data, from which updated weights can be calculated which are then sent back to the system. Otherwise, it receives the final decision about the digit on the inpuit image.

4.2 Networking

The embedded Linux distribution running on the board should automatically receive an IP address when connected to a network. When in doubt the address can be found out with the `ifconfig` command. The software has a client-server model with the embedded system acting as a server and the host as a client. Once running, the server software is listening for new outside connections.

On Windows host systems, *Network Discovery* needs to be enabled and in some cases a Firewall exception for the used ports needs to be set for a connection to be established.

4.3 FPGA Communication

The Linux system can communicate with the FPGA hardware over data streams which are in fact abstractions of FIFO buffers. For this, three FIFOs are in use:

- `/dev/xillybus_write_32`
  This stream is used to write the input images and updated coefficients to the FPGA logic.

- `/dev/xillybus_write_8`
  This stream is used as a control FIFO to tell the hardware to either read an image or the new coefficients next.

- `/dev/xillybus_read_8`
  This stream is used to read the error vector from the FPGA logic when in training mode OR the decision on the decimal number when in deployment.

5 Hardware Requirements

The hardware should fulfill the following requirements:

- Control the interface of the three FIFO buffers.
- Read the input images from 32-bit write FIFO, 28x28 byte at a time (full image) which is equal to 196x32 bits.
- Read the updated coefficients (if training is on host) and set them in the logic. The hardware needs to know how many bytes are to be read.
- Write the error vector back to the 8-bit FIFO (if training is on host), which should equal 10 bytes.
- If updated coefficients are available in the FIFO buffer, they should immediately be applied to the whole network as soon as the image that is currently processed is done.

The FIFOs have the following signals: `wr_en` or `rd_en`, 32 or 8-bit `data`, `full` and `empty`. It is a plain FIFO and not a first-word fall-trough FIFO, which means that on a clock cycle where `empty` is 0, `rd_en` needs to first be asserted for one clock cycle before the data is read in the following clock cycle.

6 Software Requirements

The host software should fulfill the following requirements:

- Read the error vector and compute new coefficients which are then sent back OR receive the decision from the neural network and generate statistics about the accuracy.
- Do not send so many input images as to overflow the buffers of the control software running on the ARM. The number of error vectors or results received back from the board should serve as a method to find out how many