Just Dab: ECE532 Project

Group 17
Maxim Antipin, Jonathan Chan, Ted Jia

Just Dab: A Dance Game

“Just Dance” is a musical rhythm game released in 2009, a major commercial success with over 40 million units sold.

“Just Dab” is an adaptation of the game, using image recognition techniques to detect and classify dance moves performed by the user.

Just Dab: A Dance Game

The game is played in the following way:

1) The user is instructed to make a particular dance move with on-screen instructions
2) The user performs the requested dance move
3) The game evaluates how well the dance move is performed, and assigns a score
4) The user continues to perform dance moves instructed by the game

The Original Plan

The initial goal was to use PMOD accelerometers to create a game where the user had to perform dance moves using the on screen instructions displayed on the VGA.

The accelerometer data would be fed into a neural network to detect the dance move, and feedback would be given to the user using the VGA and accompanying sound.

Problems Encountered: PMOD Accelerometers

- PMOD accelerometers had very short wires, not feasible for our design
- Alternative was to use longer wires, this resulted in significant signal to noise (SNR) degradation
- Longer wires also have the risk of being pulled out and lose contact with use
- A significant change had to be made, moving from accelerometers to video.

Problems Encountered: PMOD/HDMI Camera

We found out that we could not use PMOD cameras as the necessary hardware was no longer available.

Using HDMI would be a problem as well, since we already had our custom VGA IP working, and the HDMI board did not support VGA output.

We evaluated the possible alternatives and decided on using a Webcam on the host computer, and transmitting the information to the board through the USB connection.

Problems Encountered: Webcam/USB Transfer

Streaming the data from the Webcam data ended up being too fast even when streaming the live feed at the lowest resolution supported by the device.

Transferring data from the Host PC to the FPGA we discovered that the transfer rate going through the USB-UART bridge was very slow and would not work for directly streaming Webcam feed or even large images.

To get around this problem, periodic pictures were taken instead of the live Webcam feed and then scaled down to reduce file size for transferring over USB.

Problems Encountered: PNG Format

- Best output achieved from webcam: loseless but compressed
- Decoding the PNG format was necessary in order to give raw pixel data input to the artificial neural network.
- C PNG decoding library too large to fit in the MB system, instead decode on host.
- Required change in architecture of the system => tradeoff in latency.

Problems Encountered: Artificial Neural Network

Initially wanted complex neural network, millions of weights.

However due to the size of all of these parameters, fitting the neural network into the MicroBlaze was an issue.

Significant compromises had to be made:

- Hidden Layers: 3 to 1
- Number of Hidden Layer Neurons: ~300 to 4
- Input Resolution: 640x480 to 8x6

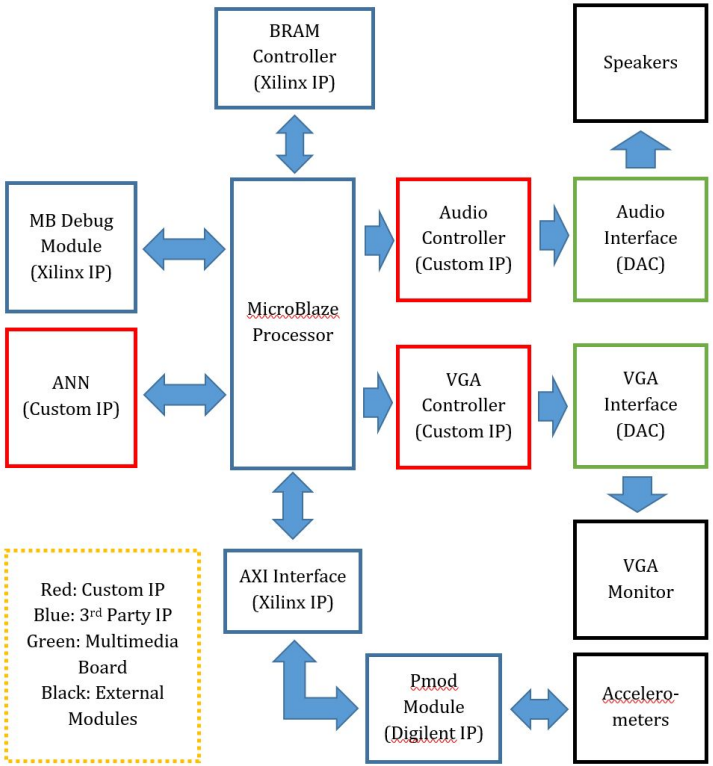
Problems Encountered: VGA Adaptor

1. How to control pixel rate and porches
2. Choosing resolution
3. Storing images and displaying them onto screen

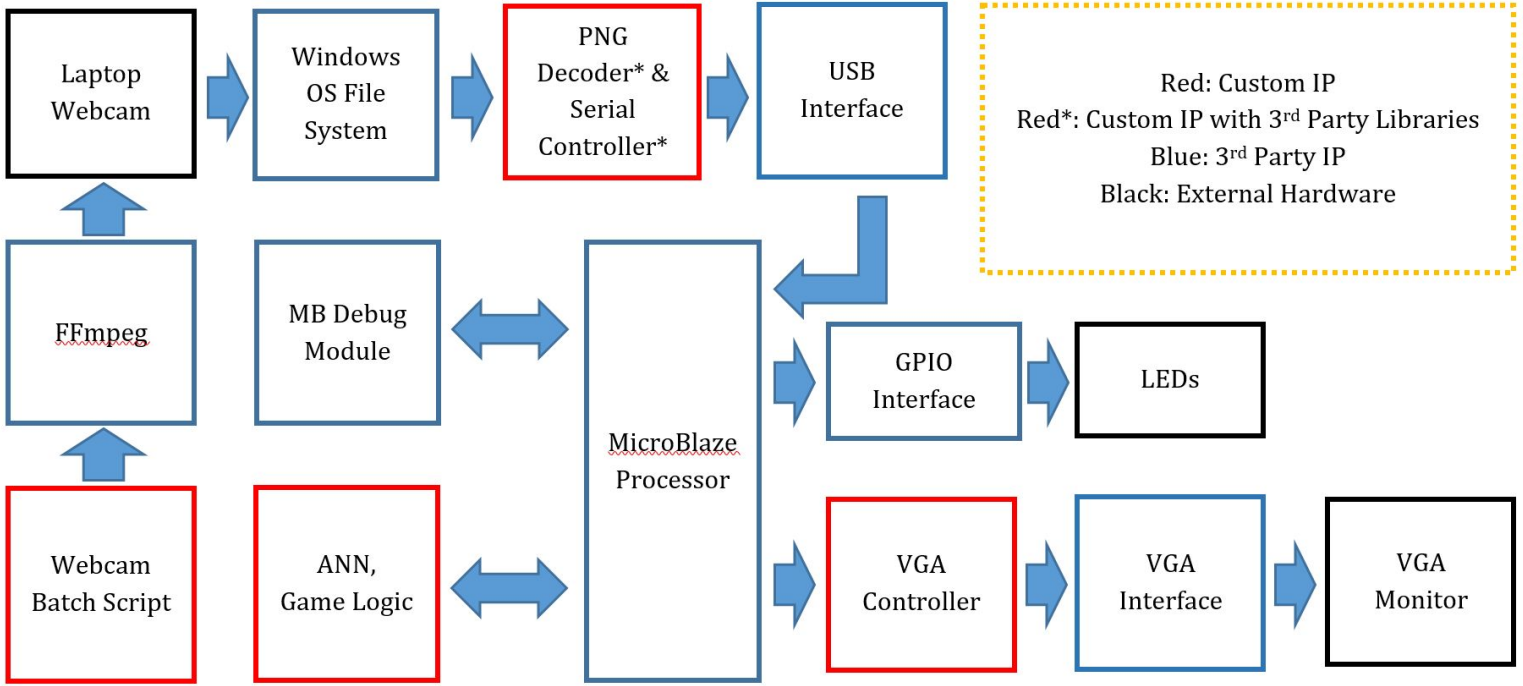
Problems Encountered: FPGA Receive UART Data

1. Needed to find a way to transfer data onto FPGA
2. Controlling the Baud Rate
3. Interpreting raw serial data
4. Found Functions that can be used to receive bytes

Initial Block Diagram



New Block Diagram



IP We Created

- Artificial Neural Network
- Game Logic Controller
- VGA Controller
- Webcam Batch Script
- PNG Controller
- Serial Controller

3rd Party IP

- FFmpeg
- Windows File System
- USB Interface
- VGA Interface
- GPIO Interface
- MicroBlaze Processor
- MB Debug Module

Just Dab: A Dance Game

Design process

- We split the project into 3 definitive sections: input, processing, and output and to work in parallel as much as possible.
- Regular discussions on expected input and outputs for each of our sections to ensure each component had the expected inputs.
- Brought up concerns as they came up and resolved them together

What we learned

- Learned that we should account for resource limitations as early as possible
- Complex designs are a lot easier to do in software than hardware