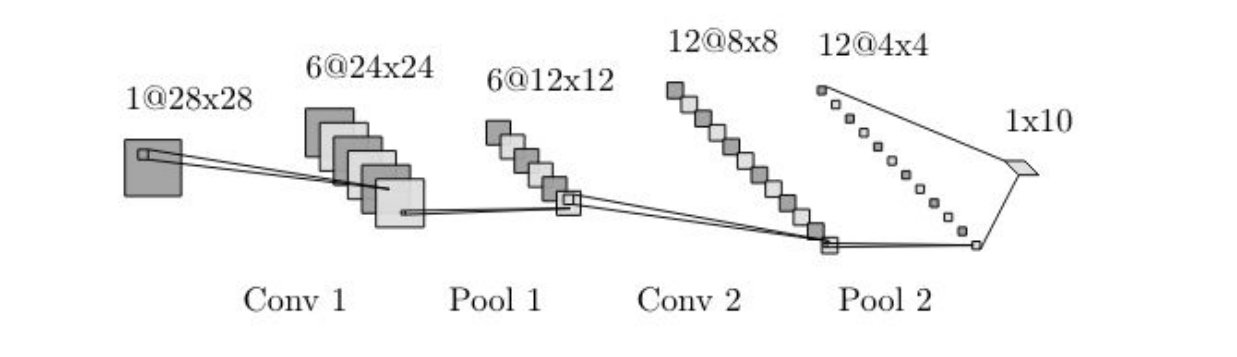
# Hardware Accelerated CNN for Digit Recognition

Liam Bishop, Dan Cooke, Felix Hanau, Ryan Kennedy and Richard Mouradian

# Model




- We used a very compact and simple CNN model featuring 2 convolution layers, 2 pooling layers and 1 Fully Connected layer
- The input to the network was a 28x28 gray scale image
- The output is a probability vector

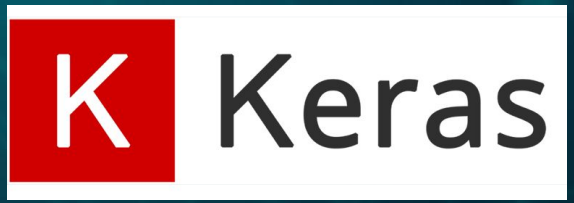
# Model Size and Parameters

The input to the CNN is a grayscale 28x28 image.

| Layer Number | Type | Kernel Size | Output Size | Parameters |
|---|---|---|---|---|
| 1 | Convolution | 6 5x5 (150) | 6x24x24 (3456) | 156 |
| 2 | Avg Pooling | 6 2x2 (24) | 6x12x12 (864) | None |
| 3 | Convolution | 6 groups of 12 5x5 (1800) | 12x8x8 (768) | 1812 |
| 4 | Avg Pooling | 12 2x2 (48) | 12x4x4 (192) | None |
| 5 | Fully Connected | None | 50 | 9610 |

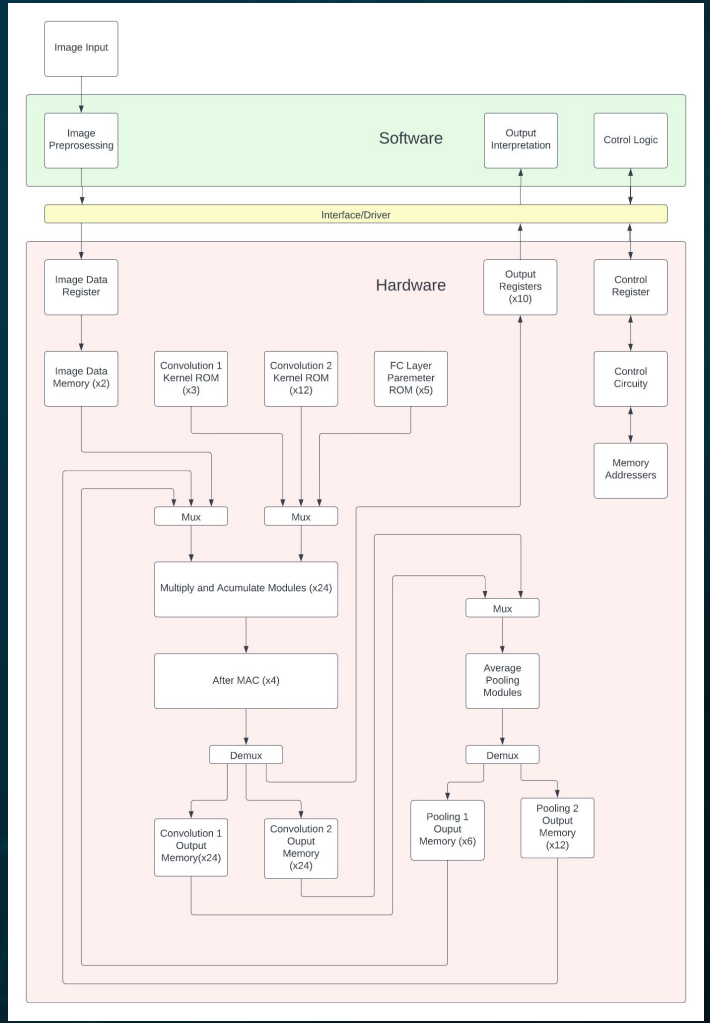
# Software Implementation

- Implementation of CNN written in C to:
  - Verify functionality of configuration
  - Implemented using 16-bit fixed-point approximation of trained weights
  - Debug output of hardware implementation
    - Also used verilator to generate waveforms and compare results of different layers
- Trained model (weight and biases) with Keras
- 98% accuracy when tested with recognizing handwritten digits



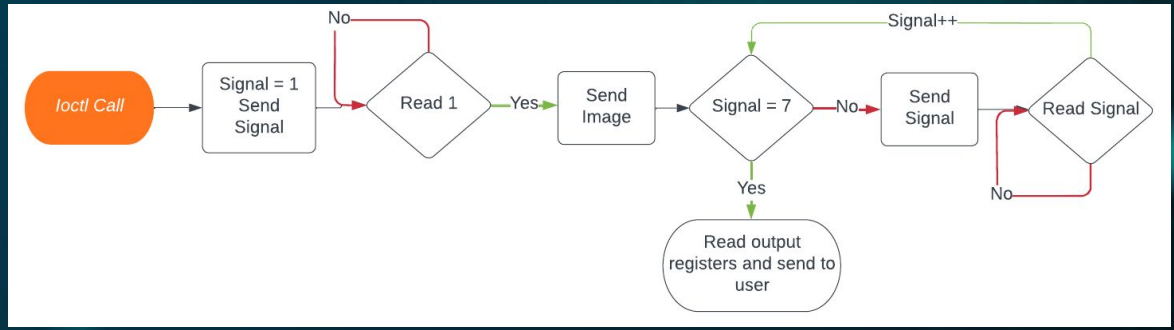

# Block Diagram

- Software:
  - Sends image data for processing
  - Controls hardware by telling it which layer to compute and waits for hardware to return acknowledgement
- Hardware:
  - 4 main Sections
    - MAC and After MAC
    - Average Pooling
    - Memory
    - Control
  - Data stream is looped through modules to perform different layer operations of CNN



# Software Driver Interface

- 14 Registers
  - Control Input
  - Control Output
  - Input address
  - Input data
  - Output registers (10)
- Control states



# MAC and After MAC

- Used to compute Convolution layers and Fully Connected Layer
- MAC:
  - Convolution:
    - 24 multiply and accumulate can be performed simultaneously.
  - Fully Connected:
    - All 10 outputs are computed simultaneously
- After MAC:
  - Processes layers:
    - Convolution:
      - Performs ReLU, adds bias to outputs of MACs, and shift outputs for proper scaling
    - Fully Connected:
      - Shifts outputs from MAC for proper scaling

# Memory

- Stores input image, layer weights, and between layer values
  - Originally Quartus Mega-Wizard Memory blocks but changed to implied memory for ease of use and debugging
  - Redundant memory blocks used to allow for more that 2 accesses at a time
  - Each section of memory controlled by read and write counters

| Layer | Data (Bits) | Weights + Bias (Bits) | Data Mem Blocks | Parameter Mem Blocks | Memory Needed (Bits) |
|---|---|---|---|---|---|
| Input | 28x28x16 | 0 | 2 | N/A | 12544 |
| 1 | 6x24x12x16 | 6x5x5x16 + 6x16 | 24 | 3 | 168384 |
| 2 | 6x12x12x16 | 0 | 12 | N/A | 13824 |
| 3 | 12x8x8x16 | 12x5x5x16 + 12x16 | 24 | 12 | 53568 |
| 4 | 12x4x4x16 | 0 | 12 | N/A | 3072 |
| 5 | 50x16 | 10x192x16 | N/A | 5 | 3242 |
| | | | Total: | 94 | 230634 |

# Pooling

- Used to compute Pooling Layer outputs
  - Average of the 4 inputs

# Benchmarking and Results

- Software implementation runtime: 7.71 ms
- Hardware Implementation runtime (Verilator): 0.205 ms
  - Capable of processing 4878 images per second
- 37 times faster than software
- FPGA resource Utilization
  - Total block memory bits: 371,328 (9%)
  - Total DSP blocks: 24 (28%)
  - Total Registers: 2534