Electronic Design Automation Project Report

Diogo Miguel Freitas de Lemos de Sousa Fernandes nº90056 MEEC
João Barreiros C. Rodrigues nº99968 MEEC

Project Architecture

Hardware Design with HLS

Certain objectives were defined at the project ideation to ensure a viable and Hardware-efficient design:

- Avoid using full-divisor blocks, opting instead to multiplicate by the reciprocal
- Avoid border conditions, opting to pad the input data beforehand

We harnessed the simplicity that **ap_int** and **ap_fixed** types introduce, making the C-code variables more similiar to VHDL signals.

Baseline ("naive") Implementation

The Baseline implementation consists of a simple algorithm:

"For each pixel sum (access) all in-memory pixel values contained in the kernel centered in that pixel"

Since only one kernel is computed at a time, multiple accesses to the same pixel are needed, to compute distinct kernels.

"Ideal" Implementation (Not followed)

The naive implementation, is understandbly inneficient, requiring multiple accesses to the same data. For each data access all arithmetic operations that require this data must be concretized in that same cycle to achieve a throughput of 1. For the blur filter this means that for each accessed pixel its values must be added to all corresponding kernels that use it.

This implementation requires an array of accumulators, ideally registers, where to store the kernel values in parallel computation. The number of kernels being computed simultaneously is equal to:

$$N_{Accumulators} = ((2 \times Radius_{Kernel}) + 1) \times UsableWidth_{DataBlock}$$

For the designed Data Block and a Kernel Radius of 8:

$$N_{Accumulators} = ((2 \times 8) + 1) \times (128 - 16) = 1904$$

Each Accumulator must be of 17-bit size in order to avoid overflows, since for a kernel radius of 8, 288 accumulations are made, which at a maximum are of value 255 (considering channel separation). therefore:

$$Max_{AccumValue} = 288 \times 255 = 73440_{10} = 10001111011100000_2$$

The total number of flip-flops needed is then:

$$N_{FF} = C \times Accumulator_{BitWidth} \times N_{Accumulators} = 3 \times 17 \times = 1904 = 97104$$

Around **91.6%** of the PYNQ-Z2's Zynq-7020 flip-flop resources.

Due to the large number of registers, the HLS tool Vitis mapped the accumulator array to a **BRAM**, adding additional latency to this design, reason why this implementation was discarded as **unviable** .

Aditional notes on possible optimizations

Data Block Division and Packing

Each data block is of size 128×128 pixels, where the middle 112×112 pixels are the relevant data. The surrounding 8 pixel-frame works as padding and ensures that no border conditions exist, allowing the averaging kernel value to **always be "divided" by a constant**.

The 128×128 size was selected based on following table:

To simplify the designed IP and assembled system and to ensure the place and route and area constrains were met, 32-bit RGB packing was adopted.

Compared to the adopted packing, single channel packing can have a more efficient word usage, with 4 pixel values per 32-bit word, althougth it may occupy additional area.

For comparision for the 128×128 block the RGB packing utilizes 16384 32-bit words, at a total cost of a single 64KiB BRAM, while the single channel packing utilizes 4096 32-bit words per channel with a reduced cost of 3 independent 4KiB BRAMs (12KiB total). The latter implementation would also require additional AXI BRAM Controllers (MMIOs), which could constrain the design.

The single-channel packing also has the advantage of allowing each channel to be computed in parallel, in addition to the parallellization implemented in the IP by the HLS tool, althougth this requires 3 independent IP instances.

Processing System Software

Project Profiling and Results

Blur Filter Profiling

Timing Report

Power Report

Conclusions