School *of* Engineering

BROWN UNIVERSITY

ENGN2911X: Reconfigurable Computing

Final Project (CNN Accelerator)

Maryam Nouh

Hatem Mohamed

Bottleneck Calculations:

Through the log it can be observed that the bottleneck of the design is the part where the convolution is calculated. In the baseline design, the following results were obtained:

```
14  ===>The following messages were generated while  performing high-level synthesis for kernel: cnn Log file: /home/mnouh/project
15  INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_67_1_VITIS_LOOP_68_2_VITIS_LOOP_69_3'.
16  INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 7, loop 'VITIS_LOOP_67_1_VITIS_LOOP_68_2_VITIS_
17  INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_74_4_VITIS_LOOP_76_6_VITIS_LOOP_77_7'.
18  INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 5, loop 'VITIS_LOOP_74_4_VITIS_LOOP_76_6_VITIS_
19  INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_83_8_VITIS_LOOP_84_9_VITIS_LOOP_85_10'.
20  INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 6, loop 'VITIS_LOOP_83_8_VITIS_LOOP_84_9_VITIS_
21  INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_94_14_VITIS_LOOP_95_15_VITIS_LOOP_96_16'.
22  INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 7, Depth = 21, loop 'VITIS_LOOP_94_14_VITIS_LOOP_95_15_VIT
23  INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_104_17_VITIS_LOOP_105_18_VITIS_LOOP_106_19'.
24  INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 10, loop 'VITIS_LOOP_104_17_VITIS_LOOP_105_18_V
25  INFO: [v++ 200-790] **** Loop Constraint Status: All loop constraints were NOT satisfied.
26  INFO: [v++ 200-789] **** Estimated Fmax: 367.92 MHz
```

Figure 1: baseline II

Number of cycles can be calculated as follows:

For computation:

$$20 + \mathbf{7} \times 112 \times 112 \times 64 \times 64 \times 4 \times 4 = 5.75G\ cycles$$

For communication:

$$(7 + 116 \times 116 \times 64) + (5 + 4 \times 4 \times 64 \times 4) + (6 + 112 \times 112 \times 64) + (10 + 112 \times 112 \times 64) = 2.47M\ cycles$$

II represents the number of cycles needed per iteration. We can roughly estimate the theoretical time needed to execute the kernel on FPGA while mainly focusing on the bottleneck.

$$time_{computation} = \frac{\mathbf{7} \times 112 \times 112 \times 64 \times 64 \times 4 \times 4}{200 \times 10^6} = 28.77sec$$

$$time_{communication} = 0.0123sec$$

Because II=7, the result is 28.77 seconds, reordering the loops can get better II, and also changing the port type to be a dual port as follows:

```
    DTYPE local_input[kInImSize][kInImSize][kNum];
#pragma HLS RESOURCE variable=local_input core=RAM_2P_URAM // uram
    DTYPE local_output[kOutImSize][kOutImSize][kNum];
#pragma HLS RESOURCE variable=local_output core=RAM_2P_URAM // uram
    DTYPE local_weight[kKernel][kKernel][kNum][kNum];
```

Figure 2: dual port URAM

Achieving II=1 can result in a timing of 4.11 seconds. This is achieved by reordering the loops to have the output channel as the innermost loop, followed by the output height and width.

```
for (int p = 0; p < kKernel; ++p) {
    for (int q = 0; q < kKernel; ++q) {
        for (int j = 0; j < kNum; ++j) {
            for (int h = 0; h < kOutImSize; ++h) {
                for (int w = 0; w < kOutImSize; ++w) {
                    for (int i = 0; i < kNum; ++i) {
                        local_output[h][w][i] += local_input[h + p][w + q][j] * local_weight[p][q][i][j];
                    }
                }
            }
        }
    }
}
```

Figure 3: loop reordering

```
===>The following messages were generated while  performing high-level synthesis for kernel: cnn Log file: /home/hemohame/Project/_x.hw.xilinx_u250_gen3x16_xdma_4
INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_77_1_VITIS_LOOP_78_2_VITIS_LOOP_81_3'.
INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 6, loop 'VITIS_LOOP_77_1_VITIS_LOOP_78_2_VITIS_LOOP_81_3'
INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_87_4_VITIS_LOOP_89_6_VITIS_LOOP_92_7'.
INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 5, loop 'VITIS_LOOP_87_4_VITIS_LOOP_89_6_VITIS_LOOP_92_7'
INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_98_8_VITIS_LOOP_99_9_VITIS_LOOP_102_10'.
INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 4, loop 'VITIS_LOOP_98_8_VITIS_LOOP_99_9_VITIS_LOOP_102_10'
INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_117_11_VITIS_LOOP_119_13_VITIS_LOOP_120_14_VITIS_LOOP_121_15'.
INFO: [v++ 200-1470] Pipelining result : Target II = NA, Final II = 1, Depth = 22, loop 'VITIS_LOOP_117_11_VITIS_LOOP_119_13_VITIS_LOOP_120_14_VITIS_LOOP_121_15'
INFO: [v++ 204-61] Pipelining loop 'VITIS_LOOP_133_17_VITIS_LOOP_134_18'.
INFO: [v++ 200-1470] Pipelining result : Target II = 1, Final II = 4, Depth = 7, loop 'VITIS_LOOP_133_17_VITIS_LOOP_134_18'
INFO: [v++ 200-790] **** Loop Constraint Status: All loop constraints were NOT satisfied.
INFO: [v++ 200-789] **** Estimated Fmax: 411.00 MHz
```

Figure 4: Optimized II

Next step is motivated by array partitioning and loop unrolling of type cyclic. This can lead to a timing as small as 0.064 seconds, which is the previous timing divided by the unrolling factor. This is done by unrolling completely the innermost loop and partitioning by a factor of 64 the input on the input channels, the output on output channels, and the kernel on the input channels.

```
    #pragma HLS array_partition variable=local_output cyclic factor=64 dim=3
    #pragma HLS array_partition variable=local_weight cyclic factor=64 dim=3
    #pragma HLS array_partition variable=local_input cyclic factor=64 dim=3

for (int p = 0; p < kKernel; ++p) {
    for (int q = 0; q < kKernel; ++q) {
        for (int j = 0; j < kNum; ++j) {
            for (int h = 0; h < kOutImSize; ++h) {
                for (int w = 0; w < kOutImSize; ++w) {
                    for (int i = 0; i < kNum; ++i) {
                        #pragma HLS unroll factor=64
                        local_output[h][w][i] += local_input[h + p][w + q][j] * local_weight[p][q][i][j];
                    }
                }
            }
        }
    }
}
```

Figure 5: code snippet that shows the unrolling and partitioning

Timing should be calculated as follows:

$$time = \frac{\mathbf{1} \times 112 \times 112 \times 64 \times 64 \times 4 \times 4}{200 \times 10^6 \times 64(unrolling\ factor)} = 0.064\ sec$$

And the number of cycles is

$$\mathbf{1} \times 112 \times 112 \times 64 \times 64 \times 4 \times 4 + 22 = 822M\ cycles$$

Baseline CNN results:

```
mnouh@arclab0:~/project/v0$ ./hello_world cnn.xclbin
Open the device 0
Load the xclbin cnn.xclbin
Running FPGA MM...
Done.
Execution time = 28.874
FPGA CNN Time: 28.874 sec, FPGA GOPS: 0.0569429
Running SW CNN...
Running OpenMP with 64 threads...
CPU CNN Time: 0.293721 sec,CPU CNN GOPS: 5.59771
Done
FPGA CNN PASS
```

Figure 6: baseline result

Figure 6 shows that the baseline timing is 28.874 seconds which is close to the theoretical one. The difference should be taken by the other loops of reading and writing.

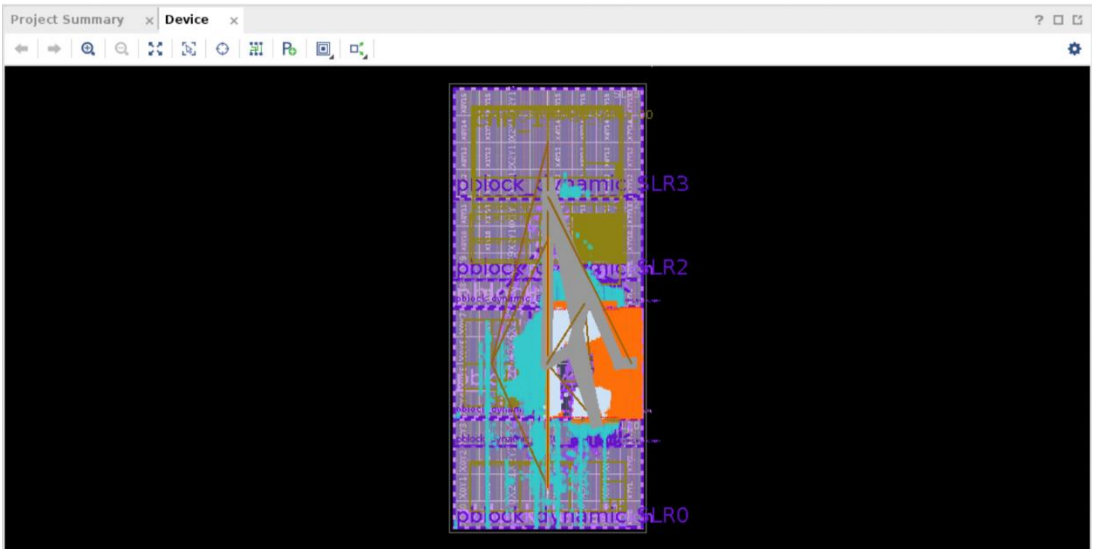

Figure 7: baseline implemented design on the U250 FPGA board

Accelerated CNN results:

```
hemohame@arclab0:~/project2$ ./hello_world cnn.xclbin
Open the device 0
Load the xclbin cnn.xclbin
Running FPGA MM...
Done.
Execution time = 0.0692559
FPGA CNN Time: 0.0692559 sec, FPGA GOPS: 23.7405
Running SW CNN...
Running OpenMP with 64 threads...
CPU CNN Time: 0.32391 sec,CPU CNN GOPS: 5.076
Done
FPGA CNN PASS
```

Figure 8: Optimized kernel result

Figure 8, shows the optimized timing. It is close to the theoretical one however the difference can be dedicated to the read and write.

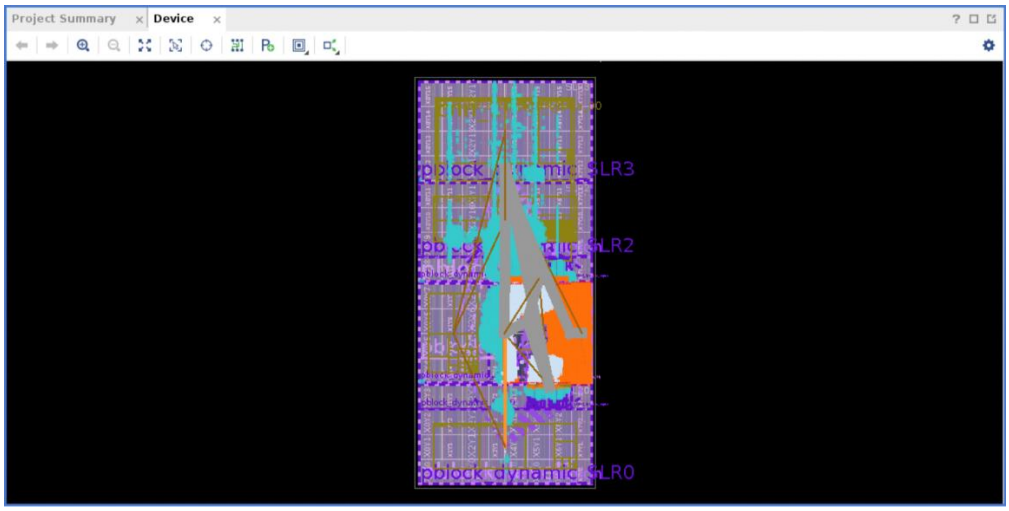

Figure 9: optimized implemented design on the U250 FPGA board

Comparison:

	V0	V1
Time(s)	28.874	0.06925
GOPS	0.0569429	23.7416
LUT	8947	26343
LUTAsMem	695	728
REG	9378	40343
BRAM	79	79
URAM	407	512
DSP	7	322
Freq (MHz)	200	200
WNS(ns)	0.035	0.037

Conclusion:

As could be observed, a speedup of $\frac{28.8}{0.069} = 417$ was obtained.