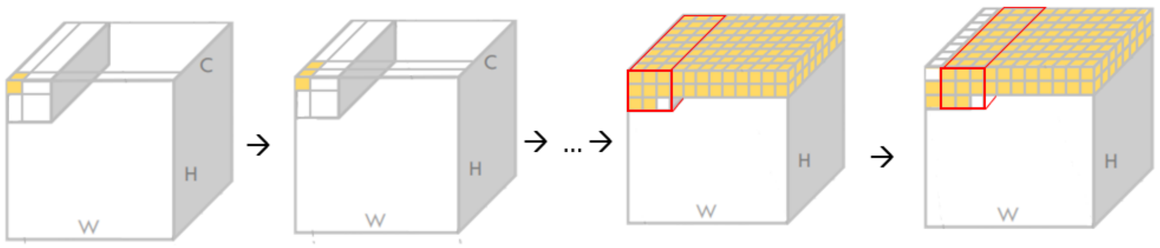
Convolution Neural Network Hardware Design

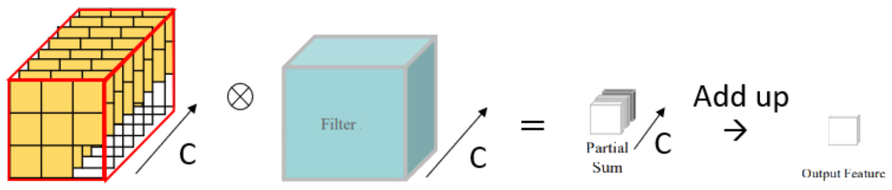
Embedded Deep Neural Network Processing | Group 5
M11202158 陳昱嘉 M11202122 鄭旭恆

Design concept and architecture

Convolution

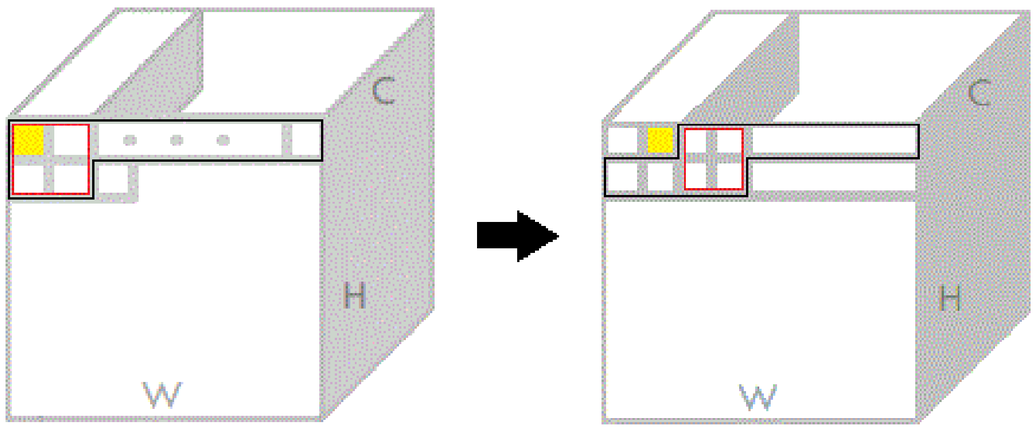
In the convolutional layer, we fetch values channel by channel. The convolution calculation starts after fetching 2 * row + 2 values for each channel. With this, in subsequent readings, registers for different channels would sequentially fetch the essential data for convolution computation. This approach enables us to utilize only one register for accumulating partial sums, and the total clock cycles required to complete the convolution task will be close to the size of the input feature map. Additionally, after the convolution of a channel, the corresponding register can discard the first element, as it is not needed until a filter changes. This ensures that the register size for each channel remains at 2 * row + 3.





Max pooling & flatten

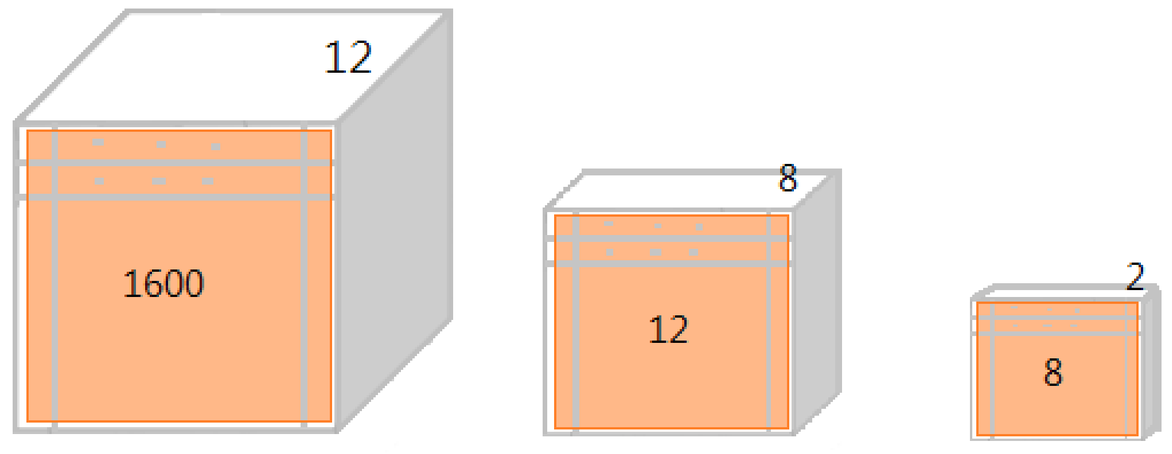
We used a **queue** to implement max pooling. When the queue reads (w+2) sizes and can start computing the max pooling result from the head and the tail of the queue until the end, this design not only saves register size but also accomplishes the mission as soon as possible.



After max pooling2, we formally write the results into RAM like form of the flatten phase, so we don't need to do flatten stage individually.

Fully Connected

We implement a fully connected like a convolutional phase, with a key distinction being that the sliding window in the fully connected layer is equivalent to the flattened size. Moreover, in order to optimize register reuse and minimize the hardware area, we've integrated three layers of fully connected structures into a single unit. This design not only enhances efficiency but also results in a more expedited and efficient performance.



Static Timing Analysis

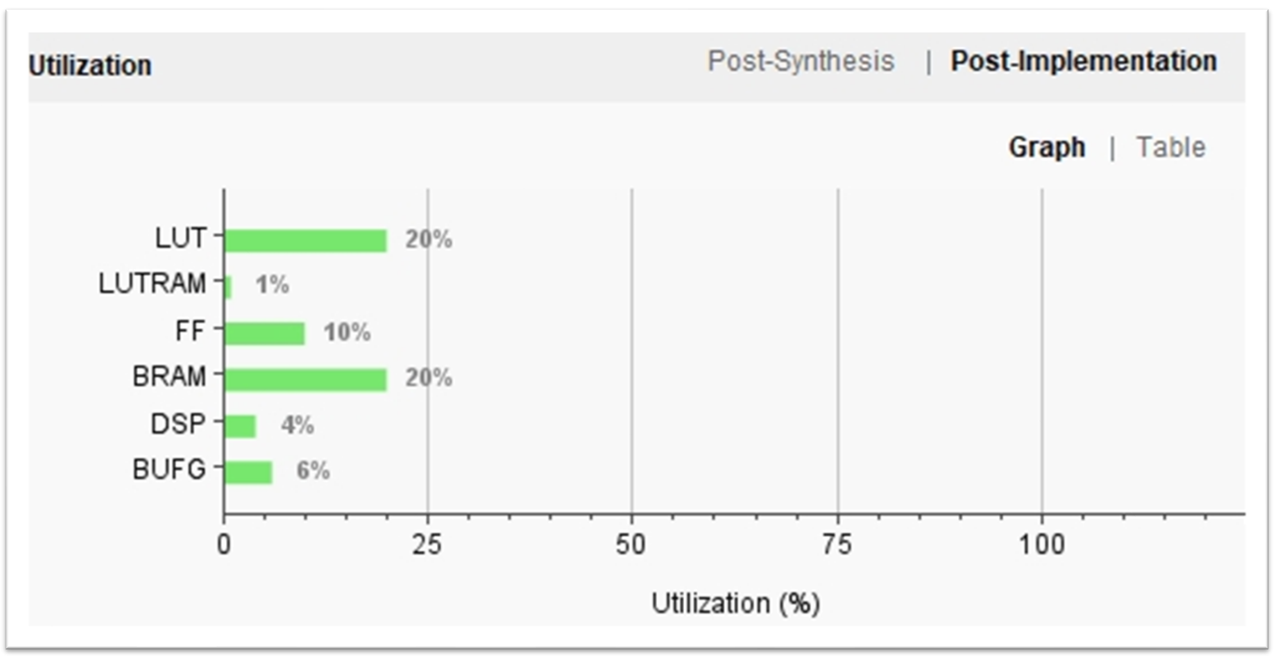
Total clock cycles =214592 clocks (max frequency: 41.68MHZ)

```
create_clock -period 23.9 -name sys_clk -waveform {0.000 11.950} [get_ports sys_clk]
```

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement
Path 1	0.307	7	3	2	NN_bd_i/FC_0/inst/q30__3/CLK	NN_bd_i/FC_0/inst/q3_reg[62]/D	11.545	9.113	2.432	12.0
Path 2	0.481	25	7	647	NN_bd_i/blk_...m/CLKBWRCLK	NN_bd_i/FC_0/...sum_reg[61]/D	11.379	6.137	5.242	12.0
Path 3	0.502	25	7	647	NN_bd_i/blk_...m/CLKBWRCLK	NN_bd_i/FC_0/...sum_reg[63]/D	11.358	6.116	5.242	12.0
Path 4	0.507	7	3	3	NN_bd_i/FC_0/inst/q30__0/CLK	NN_bd_i/FC_0/inst/q3_reg[60]/D	11.378	9.095	2.283	12.0
Path 5	0.576	25	7	647	NN_bd_i/blk_...m/CLKBWRCLK	NN_bd_i/FC_0/...sum_reg[62]/D	11.284	6.042	5.242	12.0
Path 6	0.592	25	7	647	NN_bd_i/blk_...m/CLKBWRCLK	NN_bd_i/FC_0/...sum_reg[60]/D	11.268	6.026	5.242	12.0
Path 7	0.594	24	7	647	NN_bd_i/blk_...m/CLKBWRCLK	NN_bd_i/FC_0/...sum_reg[57]/D	11.265	6.023	5.242	12.0
Path 8	0.615	24	7	647	NN_bd_i/blk_...m/CLKBWRCLK	NN_bd_i/FC_0/...sum_reg[59]/D	11.244	6.002	5.242	12.0
Path 9	0.646	7	3	3	NN_bd_i/FC_0/inst/q30__0/CLK	NN_bd_i/FC_0/inst/q3_reg[58]/D	11.247	9.113	2.134	12.0

Hardware Utilization Rate

Name	Slice LUTs (53200)	Slice Registers (106400)	F7 Muxes (26600)	F8 Muxes (13300)	Slice (13300)	LUT as Logic (53200)	LUT as Memory (17400)	Block RAM Tile (140)	DSPs (220)	Bonded IOPADs (130)	BUFGCTRL (32)
Lab_final_bd_wrapper	10750	10961	130	8	4090	10566	184	28	9	130	2
Lab_final_bd_i (Lab_final_bd)	10750	10961	130	8	4090	10566	184	28	9	0	2
axi_dma_0 (Lab_final_bd_axi	1178	1764	0	0	547	1090	88	2	0	0	0
axi_mem_intercon (Lab_final	484	579	0	0	214	471	13	0	0	0	0
myip_v1_0_0 (Lab_final_bd_	109	278	0	0	86	109	0	0	0	0	0
NN (NN_imp_13PLRVM)	8449	7652	130	8	3024	8428	21	26	9	0	1
processing_system7_0 (Lab_	0	0	0	0	0	0	0	0	0	0	1
ps7_0_axi_periph (Lab_final_	514	655	0	0	239	453	61	0	0	0	0
rst_ps7_0_50M (Lab_final_b	17	33	0	0	9	16	1	0	0	0	0



Conclusion

We make effort to combine some hardware architectures in one module and decrease model calculations within the fewest clock cycles possible. Reusing register in one module could reduce power consumption and decrease hardware utilization. However, this design needs more MUX to identify state in one module, so it causes more critical path delay and only operating in 41.6MHZ.