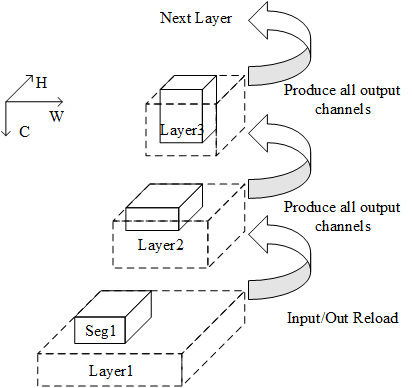
**Accomplishment / Week3 3.06-3.13**

**1. Feature Map Forward**

**1.1 Feature Map Forward Flow**

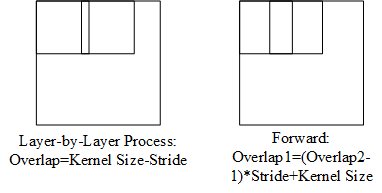


A portion of input channels in Layer2 will be produced first.

All of partial sums in input channels in Layer3 will be produced instantly. Hence the input channels produced before can be abandoned.

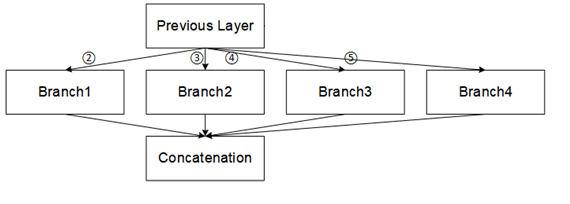
The top layers of Layer3 have the same process flow as Layer3.

**1.2 Overlap**

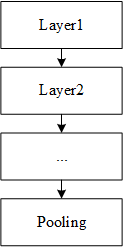


To satisfy the convolution in the edge of the next layer, some output feature should be produced again in next segment.

**1.3 Stop Point**

****

**Stop Point1:** The final feature map will be written back to DRAM when meeting concatenation layer.



**Stop Point2:** The output feature map of pooling layer will be sent back to DRAM if necessary.

**Stop Point3:** Data Movement of Forward Process[n-1] + layer[n] process > Forward Process[n].

**2. ISSCC2019 Session7 Reading**

See the PPT and Word

**3. Verilator**

See the Verilator Instruction

**Next Week’s Work Plan**

1. Reproduce an open-source DLA toy. (RISC-V+DLA) at

***<https://github.com/chenhaoc/cnnhwpe>***

2. Finish the forward process code and paper.

**Accomplishment / Week3 3.13-3.20**

**1. Feature Map Forwarding Code**

**2. Feature Map Traversal over Multiple Layers without Repeated Computation**

The sliding window between two segments will be computed in two successive cycles. For Tanji3, the zero padding is used to compute the partial sum of the sliding window in the edge.

Because of the extra partial sums (eg: Psum1 in Cycle1 and Psum2 Bottom in Cycle2), the size of output may be larger than the size of input. If the GB can not store all input and output, we don’t produce the psums in last cycle (Cycle3). And the output feature map will be sent back to DRAM or processed through layers with extra overlap.(See 3/13 work report).



In order to store the partial sums in successive address in GB, the segment is divided in one dimension instead of two. For Tanji3, it’s divided in height because of the feature map is usually stored as row-major in DRAM.

**3. Comparison between TVM and My Work**

The TVM focuses on tensor computation optimization and considers less on reducing the memory access. My work aims at least memory access between on-chip memory and off-chip DRAM.

The feature map traversal over multiple layers is not taken into consideration in TVM.

I will present the TVM and the comparison with my work in slides in one or two days.

**Next Week’s Work Plan**

1. Finish the related code in section 2

2. Combine the feature map forwarding method for different case (Tough Work)

3. Reproduce an open-source DLA toy. (RISC-V+DLA) at

***<https://github.com/chenhaoc/cnnhwpe>***

**Accomplishment / Week4 3.21-3.28**

1. Finish the code of feature map forwarding without repeated calculation.

2.Compare the TVM and MyWork and share the conclusion with Dr.Zhou.

3.Reproduce the ToyDLA in Vivado and understand the basic computation flow.

4.DC synthesize studying(Timing Constraint).

**Plan**

1.Finish the compiler document.

2.Go to further studying of ToyDLA

**Accomplishment / Week5 3.29-4.04**

1.Finish the compiler document.

2.Design Compiler (Physical Constraint and Physical Synthesis.)

3.IC Compiler(Data Setup and Design Planning)

**Plan**

1. IC Compiler Study

2. Tanji3 Data Path

**Accomplishment / Week6 4.04-4.11**

1. IC Compiler(Placement, CTS, Routing and DRC&LVS check)

**Plan**

Tanji3 Data Path

**Accomplishment / Week6 4.11-4.18**

1. Synthesize a module using TSMC65nm library.

2. Mapping the netlist to ICC and process the back-end flow.

3. Consider a special CNN/FC/GEMM accelerator computing flow (the input and weight only need to be loaded once). I will summary the idea in slides and give the presentation next week.

**Plan**

1. Finish the TSMC65nm back-end flow

2. Figure out the accelerator computing flow in detail (derive more cases with different kernel sizes and strides, zero-skipping etc.)