**Logic Design Laboratory term project**

로고, 상징, 폰트, 그래픽이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

SUBJECT: Logic Design Laboratory

Professor : Il Yong Jeon

Team Number : 3

NAME(STUDENT NUMBER): TaeHyoung Kim (2023312453), SeoYeung Byun(2024315016), MinHyuk Jeon(2019312305), MinJun Kim(2022316130)

Affiliation : Sungkyunkwan University, School of Mechanical Engineering

Date : 2025.07.12

**I. Introduction**

**1. Objective**

The purpose of this project is to understand the computational principles and processes of 2D Convolution, and to implement them in hardware using Verilog HDL. First, a Single Processing Element (Single PE) is designed, and 3×3 and 2×2 Systolic Array structures are configured based on this to perform efficient matrix multiplication operations. The computational results of each array are sequentially output through a 7-Segment Display, and the results can be confirmed through an actual FPGA board. In addition, the effectiveness of hardware parallel processing is analyzed by comparing the speed difference between the Single PE and Systolic Array structures when performing the same operation. The entire system is configured in a Structural Modeling manner to clearly understand the operating principles and interactions of each module. Through this process, the goal is to deepen the understanding of Verilog-based digital design capabilities and hardware operation optimization.

**2. 2D convolution**

2D convolution is the main operation of DNN. It uses a 4x4 input matrix and a 3x3 filter matrix to generate a 2x2 output matrix. Each output matrix value is calculated as follows.

도표, 라인, 스크린샷, 텍스트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 1. Process of 2D Convolution*

**3. Plan for implementation and simulation**

The outline of the calculator is as follows.

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.First, the controller controls the operation order of each module, and each module must be connected to the top module and work organically. The memory module stores the input and connects these values ​​to the calculation module. The calculation module has a single PE, a 3x3 systolic array, and a 2x2 systolic array, which are used to perform the convolution operation of the input matrix and the filter matrix. The result calculated in this way is confirmed to our eyes through the display module on the 7-segment.

*Fig 2. Structure of the Whole System*

**II. Module Description**

**1. Controller**

**1.1 Controller Theoretical Approach**

텍스트, 스크린샷, 도표, 원이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

*Fig 3. Moore Machine State Diagram*

The controller is implemented using a Moore machine to manage the execution flow of the system. Compared to a Mealy machine, the Moore machine provides more predictable output transitions because outputs depend solely on the current state. The state diagram consists of six states (S0–S5), each corresponding to a specific module operation. **S0**: Initialization, **S1**: Memory operation, **S2**: Single PE operation, **S3**: 3x3 Systolic Array operation, **S4**: 2x2 Systolic Array operation, **S5**: Display operation

**1.2 Controller Implementation Code**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.The controller transitions sequentially from S0 to S5. Each module asserts its **done signal** upon completion, which triggers the transition to the next state in the **Top module** via the **MODE\_NUM** signal. Once S5 is reached, the controller remains in the display state without restarting the sequence.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.*Fig 4. Controller Implementation Code*

Enable signals determine the on/off state of each module. In this design, **enable = 0** turns a module **on**, and **enable = 1** turns it **off** because these signals are directly connected to each module’s reset input. As a result, modules from earlier states stay active while transitioning to subsequent states.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**1.3 Controller testbench and simulation**

The controller was tested by incrementally increasing MODE\_NUM every 100ns to verify state transitions. Simulations confirmed that state transitions occur on the next positive clock edge after MODE\_NUM changes, and previously activated modules remain enabled as intended. Waveform analysis validated the correct timing and activation of enable signals.

스크린샷, 멀티미디어 소프트웨어, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 5. Controller testbench and simulation*

**2. Memory Module**

**2.1 Memory Theoretical Approach**

Memory is a module that simply receives 16 values ​​of the input matrix and 9 values ​​of the filter matrix, temporarily stores them, and then outputs them. Memory operates for at least two positive clock edges. In the first clock, the input values ​​are stored in the storage inside the memory module. In the next clock, the values ​​inside the memory module are output as 25 outputs.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**2.2 Memory Module**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 6. Memory Module Code*

The memory module receives input matrix values (in\_X) and filter values (f\_X), storing them in matrix\_in\_X and matrix\_f\_X, respectively. The stored values are then output through out\_in\_X and out\_f\_X. The **save** signal determines whether to store or output data, while **done\_memory** indicates whether the output process is complete. Initially, save is set to 1 and done\_memory to 0. The always block synchronizes operations with clk and rst. When rst is 1, all values are reset to 0. Otherwise, a case statement handles storing values when save is 1 and outputting values when save is 0. After the output is complete, done\_memory is set to 1. In this way, the memory module is implemented in two distinct phases: storing and outputting.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**2.3 Memory Module Testbench and Simulation**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 7. Memory Module Testbench Code*

In Fig 7, the variables to be input to the memory and the variables to be output were specified, and the values ​​of the input matrix and the values ​​of the filter matrix were randomly set and input.

텍스트, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 디자인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.As a result of running the testbench, if we see Fig 8, input value is stored on the first clock and output is generated on the second clock. And at the same time as the output is generated, the done\_memory value becomes 1, indicating that the output is complete.

*Fig 8. Memory Module Simulation Results*

**3. PE Module**

**3.1 PE module Theoretical Approach**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.The PE (Processing Element) module is the fundamental computation block for the systolic array. It accepts two inputs (input and filter), performs multiplication and addition, accumulates the result, and passes data to neighboring PEs (relay function).

**3.2 Detailed Design of PE Module Subcomponents**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 폰트, 번호, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 폰트, 라인, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.*Fig 9. PE Module Subcomponents : (a) Relay Register (b) Multiplier (c) 8bit Adder (d) Full Adder (e) Accumulator*

The module comprises four key components. **(a) Register** : The register stores incoming data synchronized to the clock and propagates it rightward and downward using pass\_right and pass\_down. This enables pipelined dataflow across the systolic array. **(b) Multiplier** : The multiplier computes an 8-bit product of the two inputs. To avoid overflow, the higher bits beyond the 8-bit width are discarded. **(c),(d) Adder** : The adder sums the multiplier output and the accumulator value using a cascade of eight 1-bit full adder modules. **(e) Accumulator** : The accumulator stores the current sum and updates its value on every rising clock edge.

**3.3 Testbench & Simulation of PE module**

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.The testbench periodically provided random inputs to verify multiplication, addition, and relay functions. Simulation results showed that:

**Relay outputs** (pass\_right, pass\_down) updated correctly on the next positive clock edge.

**Result outputs** reflected correct accumulated values in sync with the clock.

This validated that the PE module performs as expected.

스크린샷, 라인, 멀티미디어 소프트웨어, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 10. Testbench & Simulation of PE module*

**4. Single PE Module**

**도표, 스케치, 텍스트, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.4.1 Single PE theoretical design**

In the case of single PE, the previously designed PE module was used for structural modeling. IN and FILTER are input, and the two inputs are multiplied and the product of newly entered inputs is accumulated before initialization. Four output values ​​are produced (c11, c12, c21, c22). In the PE module, **pass-right** and **pass\_down** were excluded because they do not need to be output. Those will be used in the systolic array computation later.

*Fig 11. Outline of Single PE Computation*

**4.2 Single PE module**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**텍스트, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

*Fig 12. Single PE module Code*

The report includes only the core logic of the code specifically the input/output assignments to the matrix, reset behavior, result storage, and the start/end of the case statement (full details are in the code zip). The input to the module includes the input and filter arrays. Since convolution requires the filter to be rotated 180 degrees, the filter is entered in reverse order (f[2][2] to f[0][0]), ensuring proper alignment. The pe module is instantiated, and done\_single is initially set to 0, indicating that the computation hasn't finished. If rst is 1, all values are reset. Otherwise, operations proceed. A 2-bit register **pe\_delay** tracks the processing state. When pe\_delay is 2, it's simply decremented. When pe\_delay is 1, the output is saved, pe\_rst is set to 1, and pe\_delay is again decremented. In the case where pe\_delay is 0, the count increases with each PE operation. Every 9 counts, pe\_delay is set to 2 to process the result. This repeats 4 times (for c11, c12, c21, c22), and when count reaches 36, done\_single is set to 1 to signal completion.

**텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.4.3 Single PE Testbench and simulation**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

*Fig 13. Single PE testbench*

I entered random input values and filter values and made a single pe run.

스크린샷, 텍스트, 라인, 패턴이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.Looking at the simulation results, we can see that c11~c22 values ​​are output in order **every 12 clocks**(c11=12, c12=13, c21=9, c22=10). And when c22 is output, the value of done\_single also becomes 1. Finally, when rst=1, all values ​​are initialized to 0. This shows that the code is working normally.

*Fig 14. Single PE simulation results*

**5. 3x3 Systolic Array Module**

**5.1 3x3 Systolic Array Theoretical Design**

**텍스트, 도표, 스크린샷, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다. 텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

*Fig 15. Structural diagram of 3x3 systolic array*

The 3×3 Systolic Array consists of nine PE (Processing Element) modules, each of which performs partial computations based on the input values. The inputs are **a 4×4 matrix and a 3×3 filter**, and a convolution operation is performed to produce a final **2×2 output matrix**.

As shown in Fig. 15, input values are fed into the array from both the row and column directions. Each PE receives inputs through wires such as row\_pe1 and col\_pe1, performs multiplication and accumulation operations, and passes the input values unchanged to the next PE. The accumulated result within each PE is ultimately output through the resultXY output reg.

**5.2 3x3 Systolic Array Code Design & Simulation**

**텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 문서, 메뉴이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 번호, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

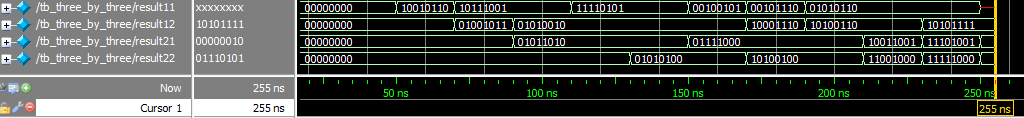
*Fig 16. 3x3 Systolic Array Code (A) Port declaration and Input data mapping (B) Instantiation of PE module (C) Result Assignment*

*Table 1. Summary of major Variable*

**텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**Table 1 provides a brief explanation of key variables used in the code along with their roles. The key variables and their roles are summarized in the table above. In Fig. 16, the input and output ports are declared, including the done\_3\_3 signal that indicates the completion of computation.

The PE modules are arranged in a 3×3 grid following a row-major order and are labeled PE1 through PE9. Among them, **PE1, PE4, PE8, and PE9 generate accumulated results**, which are mapped to **result11, result12, result21, and result22,** respectively. The intermediate values passed between PEs are named row\_pe and col\_pe, with each wire clearly indicating the originating PE for traceability. Every PE receives new input data on each clock cycle, performs multiplication and accumulation, and updates its internal accumulated value. The final result is output through res\_pe once the data flow through the systolic array is complete.

Additionally, the module is designed to reset all internal registers and input buffers to zero when a reset signal is received during operation. To feed data into the PEs, a clock counter cnt is used, pushing input data for a total of **12 cycles** (DATA\_FEED\_CYCLES = 11). The done\_3\_3 signal is asserted on the 13th cycle (cnt == 12) to indicate that computation is complete. Whenever a clock pulse occurs, the output registers are updated to reflect the latest values from the PEs.

*Fig 17. Effort to minimize clock cycles*

In the case of Fig. 17, PE8 and PE9 failed to complete their accumulation operations, resulting in incomplete outputs. In particular, within the 3×3 systolic array structure, input data propagates sequentially, and it became clear through this test that an **additional two zeros** at the end of the input sequence are necessary for the data to reach the third-row PEs. This experiment highlighted the timing requirement for the input to travel through the array and reach the lower-row PEs. Based on this insight, the value of DATA\_FEED\_CYCLES was set to 11, allowing all computations to complete in parallel and ensuring the correctness of the final outputs. All variables were named to clearly reflect their roles, with an emphasis on code readability and maintainability.

*Table 2. (a) Input Feeding Schedule by Clock Cycle (b) Accumulation Results and Computation Timing Summary*

텍스트, 번호, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

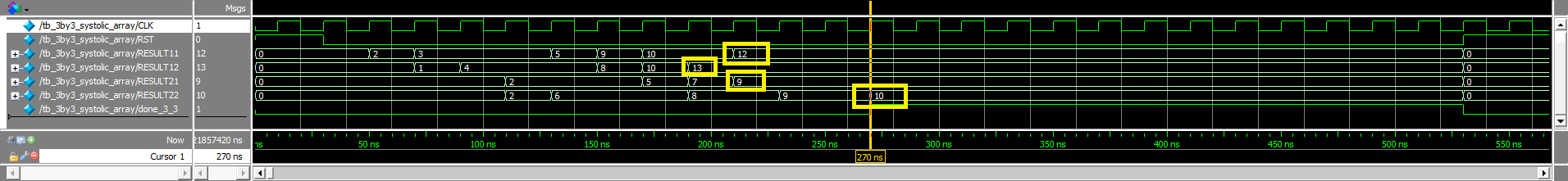
The table above summarizes the accumulation results and computation timing. It indicates the clock cycle at which each PE module receives its inputs, completes the computation, and assigns the result.

**5.3 3x3 Systolic Array Testbench**

**텍스트, 번호, 스크린샷, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

*Fig 18. Input Matrices and result of 3x3 systolic array*

****

When the input values are given as a 4×4 matrix and a 3×3 filter, as shown in the table above, the output results are observed as follows: result11 = 12, result12 = 13, result21 = 9, and result22 = 10. Rst is deasserted at 30 ns, which marks the start of the computation. After 12 clock cycles, the final result is produced, and the done\_2\_2 signal transitions to 1, indicating that the computation has completed. The total computation time is 240 ns. Since these output values match the expected results from a 2D convolution between the input matrix and filter, we can conclude that the model has been implemented correctly.

**6. 2x2 Systolic Array Module**

**텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 번호, 폰트, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.6.1 2x2 Systolic Array Theoretical Design**

*Fig 19. Structural diagram of 2x2 systolic array*

In the case of the 2×2 systolic array, the primary difference from the 3×3 structure lies in the use of **only four PE modules**. While the computation method is fundamentally the same as that of the 3×3 systolic array, the key distinction is that in the 3×3 array, only a subset of PEs produces output values, whereas in the 2×2 design, all PE modules generate res\_pe values by carefully adjusting input timing. This input timing control is achieved using the DUMMY\_ZERO variable, which allows each PE to compute an independent convolution result. As a result, the design maximizes both **parallelism** and **hardware utilization** across the array.

**텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.6.2 2x2 Systolic Array Code design & Simulation**

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 20. 2x2 systolic Array Code*

스크린샷, 멀티미디어 소프트웨어, 그래픽 소프트웨어, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.The systolic\_array\_2by2\_module implements a 2×2 systolic array using four PE modules to perform a convolution between a 4×4 input matrix and a 3×3 filter. The module is designed to feed input values over **15 clock cycles**, with each of the row\_input and col\_input arrays containing 15 values. The DUMMY\_ZERO parameter is used to control **input timing**, ensuring that each PE receives its data at the correct cycle to begin computation and generate a result. This systolic array uses a dataflow structure where the input matrix and filter values flow diagonally through the array. **Row\_input\_1 and row\_input\_2 are inputs injected from the top** in the row direction, while **col\_input\_1 and col\_input\_2 are filter values injected from the left** in the column direction. Zeros are inserted into the inputs to ensure that each PE performs multiplication and accumulation at the correct timing.

*Fig 21. Effort to minimize clock cycles*

In the case of Fig 21, an attempt was made to minimize the number of clock cycles by optimizing the input arrangement and reducing the amount of zero padding. Although one of the final DUMMY\_ZEROs was removed during testing, it was observed that the total number of **required clock cycles remained unchanged**. Therefore, for consistency with the 3×3 structure and ease of code maintenance, two zero paddings were retained at the end of the input sequence.

Each PE module computes and accumulates its result internally and outputs it via its res\_pe port. The final results are assigned to the outputs result11, result12, result21, and result22. Unlike in a 3×3 systolic array where only some PEs produce output, this 2×2 design is structured to produce results in all four PEs, maximizing both **parallelism and hardware utilization.**

*Table 3. Input Feeding Schedule by Clock Cycle and Accumulation Results and Computation Timing Summary*

텍스트, 스크린샷, 번호, 평행이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

**6.3 2x2 Systolic Array Testbench and Simulation**

*Table 4. Input Matrices and Convolution Results*

**텍스트, 번호, 스크린샷, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

When the input values are given as a 4×4 matrix and a 3×3 filter, as shown in the table above, the output results are observed as follows: result11 = 12, result12 = 13, result21 = 9, and result22 = 10, which are **identical to those produced by the 3×3 systolic array.**

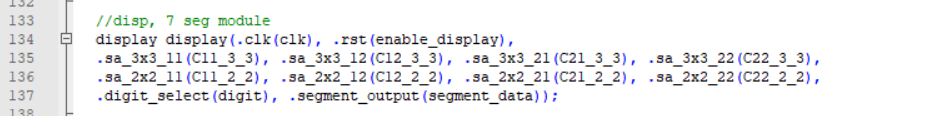
**스크린샷, 텍스트, 멀티미디어 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**The computation begins when the reset signal (rst) is deasserted (set to 0) at 30 ns. After 14 clock cycles, the final output is generated, and the done\_2\_2 signal transitions to 1, indicating that the computation has completed.

*Fig 22. waveform of 2x2 systolic array*

The total computation time is 280 ns. Since the output matches the expected results from a 2D convolution between the input matrix and the filter, it can be concluded that the model functions correctly. When the input values are given as a 4×4 matrix and a 3×3 filter, as shown in the table above, the output results are observed as follows: result11 = 12, result12 = 13, result21 = 9, and result22 = 10, which are identical to those produced by the 3×3 systolic array. The computation begins when the reset signal (rst) is deasserted (set to 0) at 30 ns. After 14 clock cycles, the final output is generated, and the done\_2\_2 signal transitions to 1, indicating that the computation has completed. The total computation time is 280 ns. Since the output matches the expected results from a 2D convolution between the input matrix and the filter, it can be concluded that the model functions correctly.

**7. Display, 7-Segment Module**

**7.1 Display, 7-Segment Module Theoretical Design**

*Figure 23. Display Module Block Inside the Top Module*

The display system consists of three modules: **clock divider**, **seven\_segment**, and **display**. It converts eight 8-bit outputs from the systolic arrays into 3-digit decimal numbers and sequentially presents them on a 7-segment display. The **display** module outputs each value one at a time at fixed intervals, ensuring that results appear in order without overlapping. It uses a **clock divider** and index control logic to manage the overall flow of output, helping to visually distinguish different stages of system operation. The **seven\_segment** module addresses the hardware limitation of the 7-segment display, which can show only one digit at a time, by using a multiplexing approach. It decomposes each value into individual digits and rapidly cycles through them, creating the visual effect of a full 3-digit number being displayed simultaneously. Together, they form the final stage of the Top module, enabling clear visualization of computation results.

텍스트, 스크린샷, 소프트웨어, 컴퓨터 아이콘이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**7.2 Clock Divider Module**

The Clock Divider is a key module responsible for timing control. It converts a high-frequency input clock (clk\_in) into a **slower output clock** (clk\_out). Internally, it uses a 26-bit counter that increments on every rising edge of clk\_in. When the counter reaches a predefined DIV value, clk\_out toggles and the counter resets. The DIV value determines how many input clock cycles are required for one toggle of the output clock, thereby controlling its period.

*Fig 23. Clock Divider Module*

This module is used in both the Display Module and the 7-Segment Module, each with different DIV settings. In the Display Module, the **DIV value is set to 49,999,999** to display each output value at 1-second intervals. In the 7-Segment Module, **the DIV value is set to 99,999** to rapidly cycle through digit positions. This enables smooth and flicker-free display of 3-digit numbers using the persistence of vision effect.

텍스트, 스크린샷, 소프트웨어, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**7.3 Display Module**

The display module is responsible for sequentially presenting eight 8-bit output values—four from the 3×3 Systolic Array and four from the 2×2 Systolic Array—on a 7-segment display. Each value is shown for approximately one second, and to achieve this behavior, the module is composed of a clock divider, index control logic, and a submodule called seven\_segment. The input clock (clk) is first divided into a slower clock (**slow\_clk\_2**) using the clock\_divider. This slower clock determines the interval between output transitions. On the actual FPGA board, the DIV value is set to 49,999,999 to produce a 1-second interval; in simulation, a smaller DIV value is used for easier waveform observation.

*Fig 24. Display module*

The eight input values are stored in an internal array (result\_data), and a 3-bit index register (idx) increments from 0 to 7 on every rising edge of slow\_clk\_2. This index selects one value at a time, which is then passed to the seven\_segment module for display. The seven\_segment module handles digit decomposition and multiplexed output.

**7.4 7-Segment module**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다. 텍스트, 스크린샷, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Figure 25. 7-Segment Module*

The seven\_segment module converts an 8-bit input value into a 3-digit number and displays it smoothly on a 7-segment display. It consists of a clock divider, digit decomposition logic, and segment output logic. The input value is split into **hundreds, tens, and unit digits**, which are stored internally. These digits are displayed one at a time in rapid succession using a slower clock (**slow\_clk\_1**). The digit\_select signal cycles through each digit, and the corresponding value is converted into a 7-segment pattern. Through this process, the display appears to show all three digits simultaneously due to the persistence of vision effect. On the actual FPGA board, the DIV value is set to 99999 to achieve a flicker-free display, while in simulation, it is reduced to 4 for easier waveform observation. This module plays a key role in presenting computational results in a clear and readable format.

**7.5 Display Module Testbench**

텍스트, 스크린샷, 디스플레이, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다. 텍스트, 스크린샷, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Figure 26. Testbench of Display Module*

The display\_tb testbench verifies whether the outputs from the 3×3 and 2×2 Systolic Arrays are correctly displayed through the Display Module. Eight 8-bit input values are provided, and the test confirms that they appear sequentially on the 7-segment display. At the start, all inputs are reset to zero, and a 10 ns clock is generated. Two sets of input values are applied at different times to observe output changes. The clock divider is set to DIV = 12500 to speed up output transitions and make waveform analysis easier.

스크린샷, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.*Figure 27. Display Module Wave Simulaiton*

The simulation results confirmed that both segment\_output and digit\_select signals functioned as expected. All eight input values were displayed in order, and digit\_select rapidly cycled through each digit position, activating the corresponding 7-segment display. As a result, segment\_output changed periodically, successfully creating the visual effect of a 3-digit number being displayed **simultaneously**. By adjusting the clock cycles and parameter values, each output remained visible for a sufficient duration, and the final value was also correctly maintained. These results verify that the module operates as intended and can provide accurate visual output when implemented on an FPGA board.

**7.6 XDC Code (Vivado)**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.This XDC file maps the Top module's ports to the FPGA board’s I/O pins in Vivado. clk and rst are connected to pins R4 and J4, handling the system clock and reset. digit[7:0] selects each digit of the 7-segment display via pins P14 to R17, while segment\_data[7:0] controls the display segments through pins U17 to P19. This mapping ensures that the Top module's output is correctly displayed on the FPGA board.

*Figure 28. XDC Pin Configuration*

**8. Top Module**

**8.1. Top Module Theoretical Approach**

The Top module orchestrates all subsystems: controller, memory, PE, systolic arrays, and display. It monitors **done signals** from each module to increment **MODE\_NUM** and transitions through each computation stage.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

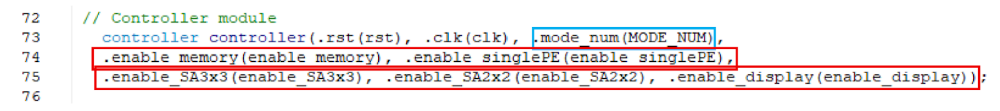
AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**8.2 Top Module Implementation Code**

The Top module initializes 16 input data values and 9 filter data. These propagate through memory and computational modules before being displayed.

*\*

*Fig 29. Define Top module*

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

텍스트, 폰트, 번호, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 폰트, 라인, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 30. Connections with all the Subsystems : (a) Controller Module (b) Memory Module (c) Computational Module (d) Display Module*

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.Each module is enabled via **enable\_xx** signals, and transitions are coordinated using **done\_xx** flags. Once all computations are complete, the display state (S5) remains active to showcase the results.

*Fig 31. Top Module: Mode Number Update Mechanism*

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**8.3 Top module testbench and simulation**

The Top module has a straightforward structure, with rst and clk as inputs and digit and segment\_data as outputs. Accordingly, its testbench was designed with a simple configuration: the clock was set to a 20ns period, and rst was deasserted after 20ns.

*Fig 32. Top Module Testbench*

However, digit and segment\_data alone do not provide insight into the system’s internal operation. They do not convey critical information such as the state transition flow or the computation results from the Single PE, 3x3 Systolic Array, and 2x2 Systolic Array modules. To address this, additional monitoring logic was implemented to print key variables to the **console and waveform** during simulation. Specifically, the enable signals for each module and the computed results (C11, C12, C21, C22) were displayed.

텍스트, 폰트, 번호, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다. 텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.스크린샷, 텍스트, 멀티미디어 소프트웨어, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

*Fig 33. Top Module Simulation (a) Testbench Code to display results in console (b) Results displayed in Console (c) Results displayed in Waveform*

This approach made it possible to verify that all modules were properly connected and activated in sequence, confirming the Top module’s role in coordinating the overall system operation. The **console output Fig. 33-b** clearly shows how state transitions (MOD\_NUM changes) control module enable signals (enable\_memory, enable\_singlePE, etc.) and how each computation result appears as expected. In addition**, the waveform view Fig. 33-c** provides a timeline of these events, showing precise timings of done\_memory, enable\_singlePE, and the sequential appearance of computation results. For example, after **done\_memory** transitions **from 0 (incomplete) to 1 (complete)**, **enable\_singlePE** changes from **1 (off) to 0 (on)** shortly thereafter, activating state S2. Following this transition, the **computation results (C11\_single through C22\_single) are sequentially produced and visualized**.

This approach allowed verification that all modules were properly connected and executed sequentially, confirming the Top module’s role as the orchestrator of the overall system.

**III. Result**

1. **Pre-Demonstration FPGA Test Results**

Prior to the official demonstration, the system was tested on the FPGA board to ensure proper functionality under actual operating conditions. Input matrices and filter kernels were manually applied, and the results of the 2D convolution operation were observed to verify that the module performed as intended.

전자제품, 회로, 전자 공학, 전자 부품이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**Test Case 1**

**사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**텍스트, 폰트, 직사각형, 사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**Input matrix Filter matrix Output matrix Result Test Case 2**

*Fig 34. Test Case 1*

시계, 디지털 시계, 측정기, 텍스트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**Test Case 2**

**사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.사각형, 직사각형, 디자인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**Input matrix Filter matrix Output matrix Result**

*Fig 35. Test Case 2*

1. **Demonstration Result**

**사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.** 사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다. 직사각형, 라인, 번호, 사각형이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다. **시계, 텍스트, 디지털 시계, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**Input matrix Filter matrix Output matrix Result**

*Fig 36. Demonstration Result*

The demonstration confirmed that the system functioned correctly on the FPGA board. All convolution results were accurately computed, and the output values — **82, 95, 92, and 94** — were displayed sequentially on the 7-segment display at 1-second intervals. Throughout the process, no functional errors occurred, and both the processing logic and display module operated as intended.

**3. Performance Comparison (Speedup of 3x3 / 2x2 systolic array over a single PE )**

**스크린샷, 멀티미디어 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**스크린샷, 멀티미디어 소프트웨어, 소프트웨어, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

**멀티미디어 소프트웨어, 소프트웨어, 그래픽 소프트웨어, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

*Fig 37. Output of the Top Module : (A)Single PE (B) 3x3 systolic array (C) 2x2 systolic array*

The images above show the execution speeds of each module when run from the top module. The measurement is based on the moment when the reset signal input goes low, marking the start of computation, and ends when the final accumulated output value is produced.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

**텍스트, 스크린샷, 도표, 라인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.**

*Fig 38. Execution Time of Each Computational Modules and Speed up comparing to Single PE (a) Table (b) Graph*

The execution times were measured **as 960 ns for the Single PE, 300 ns for the 2×2 array, and 260 ns for the 3×3 array.** These results demonstrate that utilizing more PEs enables faster computation. This is because, in parallel processing, performing more operations simultaneously leads to quicker completion. Through these results, it can be confirmed that the systolic array is a viable structure for use as a hardware accelerator, and that the systolic array was successfully implemented in this project.

**4. Contribution**

This project involved the implementation of a 2D convolution calculator, with each team member responsible for designing and testing specific modules. Each member independently handled code development, testbench creation, and report writing for their assigned part. Based on this collaboration, the system was successfully integrated and demonstrated.

텍스트, 폰트, 스크린샷, 영수증이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.