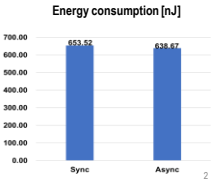
Design of Asynchronous CNN Circuits on Commercial FPGA from Synchronous CNN Circuits

Hayato Kato, The University of Aizu, Japan
Hiroshi Saito, The University of Aizu, Japan

Summary of This Presentation

- Field Programmable Gate Arrays (FPGAs) are used for Convolutional Neural Network (CNN) to accelerate performance
  - To reduce the energy consumption
- We propose a design method to design asynchronous CNN circuits on commercial FPGAs
  - Conversion from Register Transfer Level (RTL) models of synchronous CNN circuits to RTL models of asynchronous CNN circuits
  - Design flow based on a commercial FPGA design environment
- Experiment
  - Compared to the synchronous counterpart, about 2.3% lower energy consumption



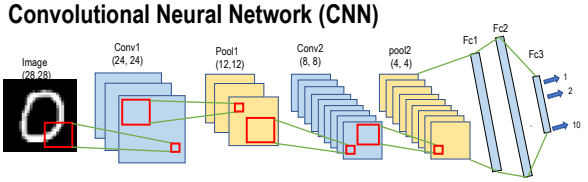
Outline

- Background
- Motivation and purpose
- Proposed method
- Experiment
- Conclusions

Outline

- Background
- Motivation and purpose
- Proposed method
- Experiment
- Conclusions

Convolutional Neural Network (CNN)



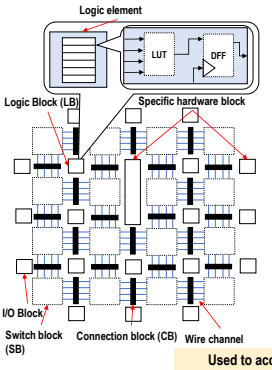
- Realizing high accuracy of image recognition [1]
  - Convolution layer and pooling layer
    - Extracting features from images
  - Full connect layer
    - Classifying images by extracted features from previous layers

Increase of accuracy

Increase of the number of operations and the usage of memory space

Significant problems when real-time execution and low energy consumption are required

Field Programmable Gate Array (FPGA)

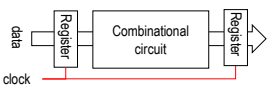


- A reconfigurable device
  - Low design cost and longer life cycle
- Structure
  - Logic Blocks (LBs)
    - Consists of logic elements
      - Look-Up-Table (LUT)
      - D Flip-Flop (FF)
  - I/O pins
    - Used for input and output
  - Wire elements
    - Connects LBs
  - Others
    - Memory blocks
    - Specific hardware blocks

Used to accelerate CNN

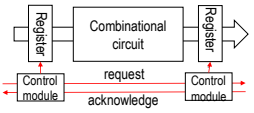
Synchronous Circuits and Asynchronous Circuits

- Synchronous circuits



  - Controlled by a global clock signal
  - Problems by the semiconductor minimization technology
    - High power consumption
    - High electromagnetic radiation
  - Supported by FPGA design tools

- Asynchronous circuits



  - Controlled by local handshake signals
  - No global clock signal
    - Low power consumption
    - Low electromagnetic radiation
  - Commercial FPGA design tools do not intend asynchronous circuits on FPGA

Has a potential to reduce power consumption for CNN

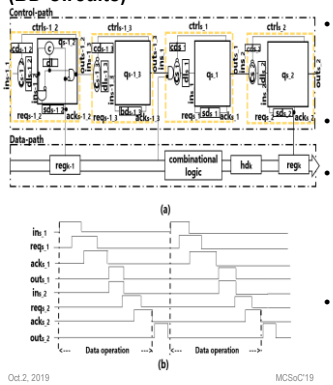
Outline

- Background
- **Motivation and purpose**
- Proposed method
- Experiment
- Conclusions

Motivation and Purpose

- Motivation
  - Increase of the number of operations and the usage of memory space to archive high accuracy by CNN
  - Many CNN models have been implemented on FPGAs [2-6]
    - These researches implemented synchronous CNN circuits on FPGAs
  - Design of asynchronous circuits on FPGAs is difficult although asynchronous circuits are low power consumption
- Purpose
  - We propose a design method to design asynchronous CNN circuits on FPGAs
- Approach
  1. Conversion of Register Transfer Level (RTL) models of synchronous CNN circuits to asynchronous ones
  2. Design of asynchronous CNN circuits using a commercial FPGA design environment

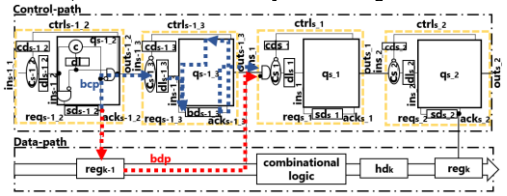
Asynchronous Circuits with Bundled-data Implementation (BD Circuits)



- N+2 signals to represent N-bit data
  - Request signal (req)
  - Acknowledge signal (ack)
- Data-path circuits
  - Same as synchronous circuits
- Control circuit
  - Consists of control modules $ctrls\_t$
    - One pipeline stage is controlled by more than two control modules
- Control timing
  - Guaranteed by delay elements

Timing Constraints

- BD circuits need to satisfy setup, hold, and control initialization constraints described in [13]
  - Delay adjustment for sd_s, hd_k, and cd_s is required when violated
- In addition, BD circuits need to satisfy the following branch constraints



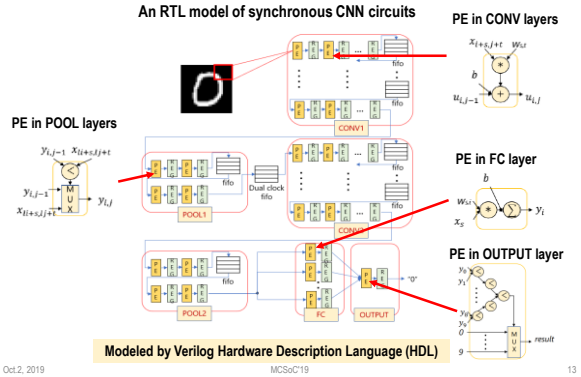
$$t_{minbcp} > t_{maxbdp} + t_{marginbdp}$$

(blue line) (red line)

Outline

- Background
- Motivation and purpose
- Proposed method
- Experiment
- Conclusions

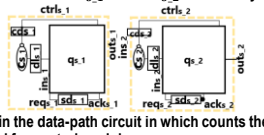
Synchronous CNN Circuit Models



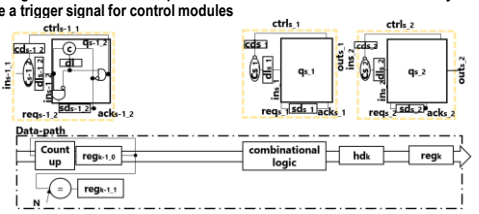
Modeled by Verilog Hardware Description Language (HDL)

1. Conversion from RTL Models of Synchronous CNN Circuits to the RTL Models of Asynchronous Circuits (1/4)

1. Assign two control modules $ctrl_{s\_1}$ and $ctrl_{s\_2}$ for each layer

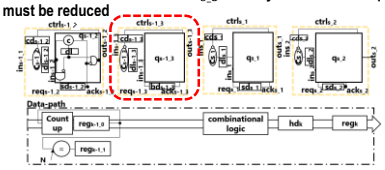


2. Inserting a counter in the data-path circuit in which counts the number of cycles and generate a trigger signal for control modules

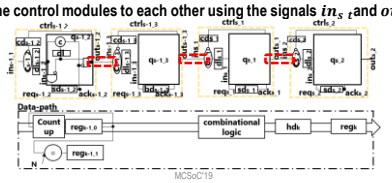


1. Conversion from RTL Models of Synchronous CNN Circuits to the RTL Models of Asynchronous Circuits (2/4)

3. Assign an additional control module $ctrl_{s\_3}$ for a layer where the frequency of the control signal must be reduced

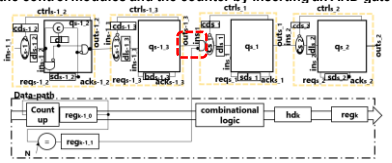


4. Connect the control modules to each other using the signals in_{s_t} and out_{s_t}

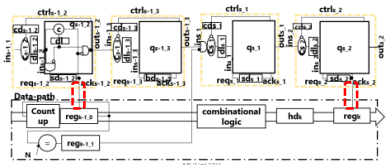


1. Conversion from RTL Models of Synchronous CNN Circuits to the RTL Models of Asynchronous Circuits (3/4)

5. Connect the control modules and the counter by inserting an AND gate



6. Replace the clock signal of registers in each layer *s* to *ack*

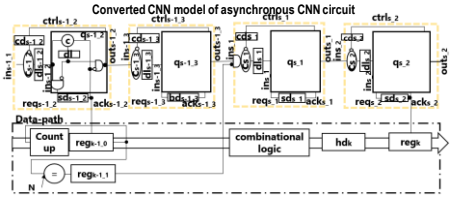


1. Conversion from RTL Models of Synchronous CNN Circuits to the RTL Models of Asynchronous Circuits (4/4)

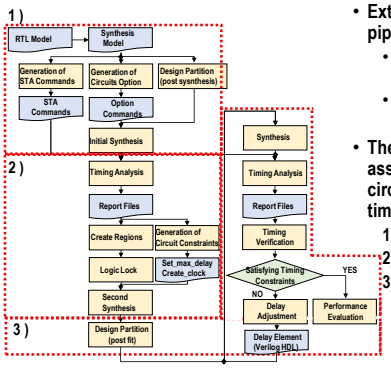
7. Generate ***sd*** to each control module using Eq (6)

$$num_{sd_{s,t}} = (ct * r_{max})/(2 * 2 * t_{cell}) \cdots (6)$$

- $num_{sd_{s,t}}$: the number of cells used in
- ct : target cycle time
- r_{max} : the ratio of the maximum control module delay for ct
- t_{cell} : the delay per cell in delay elements



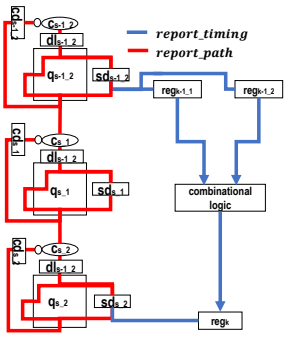
2. Design of Asynchronous CNN Circuits (1/4)



- Extension of [14] to deal with pipelined circuits
  - Target BD circuits are pipelined circuits
  - Placement constraints are used
- The proposed design flow is assumed to synthesize BD circuits more than three times
  1) Until the initial synthesis
  2) Second synthesis
  3) After second synthesis

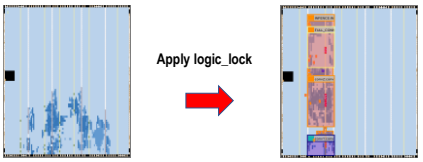
2. Design of Asynchronous CNN Circuits (2/4)

- Preparation for the initial synthesis
  - Path delay analysis commands
    - *report_timing*
    - *report_path*
  - Synthesis options
    - Preventing the use of global clock signals
    - Preventing the hold optimization
  - Design partitions (post synthesis)
    - To keep the design hierarchy
- Initial synthesis
  - To obtain delay and area to generate design constraints



2. Design of Asynchronous CNN Circuits (3/4)

- Preparation of design constraints for the second synthesis to satisfy performance requirement
  - Two sets of timing constraints
    - *create_clock* for register/write signals
    - *set_max_delay* for control/data-paths
  - Placement constraints
    - *logic_lock* for each layer



- Second synthesis with design constraints

2. Design of Asynchronous CNN Circuits (4/4)

- After second synthesis
  - Iteration of synthesis, timing verification, and delay adjustment until all timing constraints required for BD circuits are satisfied
  - Change of design_partition for data-path resources to preserve placement and routing of the previous synthesis

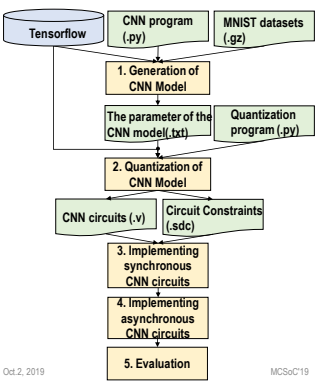
Outline

- Background
- Motivation and purpose
- Proposed method
- Experiment
- Conclusions

Experiment

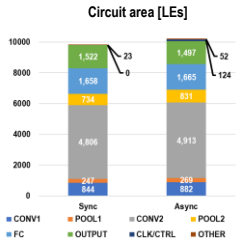
- Purpose of this experiment
  - Designing an asynchronous CNN circuit for an Intel FPGA using the proposed method
  - Evaluating performance by comparison with the synchronous CNN circuit
- Target FPGA
  - Cyclone IV FPGA (EP4CE115F29C7)
- Tools
  - Intel Quartus Prime 17.1 Standard Edition
  - ModelSim Intel FPGA Starter Edition 10.5b
  - Tensorflow [2]
- Evaluation items
  - Circuit area
  - Execution time
  - Dynamic power consumption
  - Energy consumption

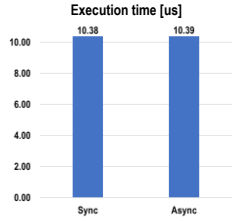
Flow of the Experiment



- Generation of CNN Model
  - Using Tensorflow [7]
  - Dataset : MNIST dataset
  - Accuracy : 81.9%
- Quantization of CNN Model
  - Quantized from 32-bit data to 8-bit
  - Accuracy : 81.0%
- Implementing synchronous CNN circuits
  - Cycle time : 12.5 nsec
- Implementing asynchronous CNN circuits using proposed method
  - Density of each region for placement constraints : 70%
- Evaluation
  - Input "0" image data

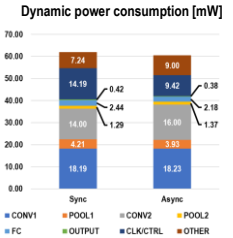
Circuit Area and Execution Time

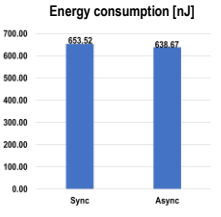




- The synthesis report generated by Quartus Prime
- 4.1% increase for asynchronous circuits
  - Insertion of control modules and delay elements
- An arbitrary test input is given to ModelSim with the designed circuit
- 0.1% increase for asynchronous circuits
  - Handshake overhead from handshake signals

Dynamic Power and Energy Consumption





- Obtained by the PowerPlay Power Analyzer in Quartus Prime
- 2.4% decrease for asynchronous circuits
  - Asynchronous circuits used local handshake signals instead of the global clock
- Execution time * dynamic power consumption
- 2.3% decrease for asynchronous circuits
  - Due to the decrease the dynamic power consumption

Outline

- Background
- Motivation and purpose
- Proposed method
- Experiment
- Conclusions

Conclusions

- We propose a design method for asynchronous CNN circuits on FPGAs
  - A conversion method from RTL models of synchronous CNN circuits to RTL models of asynchronous CNN circuits
  - A design flow using a design support environment provided by an FPGA vendor
- The designed asynchronous CNN circuits was 2.3% lower energy consumption than the synchronous counterpart
- Future work
  - Reduction of the dynamic power consumption
    - Reconsidering the control circuit of the asynchronous CNN circuits

References

[1] Y. LeCun et al., "Gradient-based learning applied to document recognition", Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
[2] J. Qiu et al., "Going Deeper with Embedded FPGA Platform for Convolutional Neural Network", Proc. ISFPGA, pp. 26–35, 2016.
[3] N.Li et al., "A multistage dataflow implementation of a Deep Convolutional Neural Network based on FPGA for high-speed object recognition", Proc. SSIAI, pp. 165–168, 2016.
[4] C. Huang et al., "A layer-based structured design of CNN on FPGA", Proc. ASICON, pp. 1037–1040, 2017.
[5] N. Yildiz et al., "Architecture of a Fully Pipelined Real-Time Cellular Neural Network Emulator", IEEE TCAS I: Regular Papers, vol. 62, no.1, pp. 130–138, 2015.
[6] N. Suda et al., "Throughput-Optimized OpenCL-based FPGA Accelerator for Large-Scale Convolutional Neural Networks", Proc. ISFPGA, pp. 16–25, 2016.
[7] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning", IEEE Micro, Vol. 38, Issue 1, pp. 82–99, 2018.
[8] S. Semba and H. Saito, "Comparison of RTL Conversion and GL Conversion from Synchronous Circuits to Asynchronous Circuits", Proc.ISCAS, pp. 1–4, 2019.
[9] Q. T. Ho et al., "Implementing Asynchronous Circuits on LUT Based FPGAs", Proc. FDL, pp. 33–46, 2002.

References

[10] M. Tranchero and L. M. Reyneri, "Exploiting sychronous placement for asynchronous circuits onto commercial FPGAs", Proc. Field Programmable Logic and Applications, pp. 622–625, 2009.
[11] R. Mocho et al., "Asynchronous Circuit Design on Reconfigurable Devices", Proc. SBCCI, pp. 20–25, 2006.
[12] L. Reyneri, and M. Tranchero, "Implementation of Self-Timed Circuits onto FPGAs Using Commercial Tools", Proc. EUROMICRO, pp. 373–380, 2008.
[13] J. Furushima and H. Saito, "Design of an Asynchronous Processor with Bundled-data Implementation on a Commercial Field Programmable Gate Array", Informatica 40 (2016), pp. 399–408 , 2016.
[14] K. Takizawa and H. Saito, "A Design Support Tool Set for Asynchronous Circuits with Bundled-data Implementation on FPGAs", Proc. FPL, pp.1–4, 2014.
[15] F. U. Rosenberger et al., "Q-modules: internally clocked delay insensitive modules", IEEE TC, vol. 37, no. 9, pp. 1005–1018, 1988.
[16] Y. LeCun, and C. Cortes, "THE MNIST DATABASE of handwritten digits", http://yann.lecun.com/exdb/mnist/
[17] M. Abadi et. al. "TensorFlow: A System for Large-Scale Machine Learning", Proc. OSDI, pp. 265–283, 2016.

Thanks for Listening…