Area-Efficient Transposable 6T SRAM for Fast Online Learning in Neuromorphic Processors

Jongeun Koo, Jinseok Kim, Sungju Ryu, Chulsoo Kim, and Jae-Joon Kim
Pohang University of Science and Technology (POSTECH), Pohang, Republic of Korea
{jongeun.koo, jinseok.kim, sungju.ryu, chulsookim, jaejoon}@postech.ac.kr

***Abstract*—This paper presents a 6T SRAM-based transposable synapse memory aiming to improve online learning performance of neuromorphic processors at the minimum area cost. While a custom 8T SRAM was used in the previous transposable synapse memory, the proposed one uses 6T SRAM, which leads to substantial area savings. Based on the proposed hierarchical word line structure with row transition multiplexer, both row-wise and column-wise accesses are made possible in an integrated SRAM array. A 64K-synapse memory employing the proposed scheme is implemented in a 28nm CMOS technology, which has 17.7% area overhead compared to the non-transposable 6T synapse memory; 50.0% area savings compared to the transposable 8T synapse memory, and 35.5% area savings compared to the previous transposable 6T synapse memory. The estimated performance gain for online spike-timing-dependent plasticity learning using MNIST dataset is 6.7× compared to the non-transposable synapse memory.**

***Keywords*—neuromorphic engineering; spiking neural networks; spike-timing-dependent plasticity; synapse memory;**

I. Introduction

Recently, spiking neural network (SNN) has been drawing attentions for its bio-inspired characteristics and energy efficiency [1], [2]. However, since conventional von Neumann architecture cannot handle the binary spikes and time delay information in SNNs efficiently, interests in dedicated neuromorphic processors have been growing [3], [4], [5]. Especially in [3] and [4], crossbar synapse memory structures were used to provide sufficient connectivity between neurons and high memory bandwidth.

Meanwhile, the importance of online learning is getting increased in neuromorphic processors due to the high communication cost and data security issues in offline learning. Spike-timing-dependent plasticity (STDP) is a well-known SNN unsupervised learning method that updates synaptic weights depending on the spike timing differences between presynaptic neurons (t_{pre}) and postsynaptic neurons (t_{post}) [6]. In the non-transposable crossbar synapse memory based on 6T SRAM, a row-wise weight vector can be concurrently accessed by enabling a single word line (WL), whereas a column-wise weight vector can be accessed element-by-element only by enabling all WLs one-by-one. For post-then-pre ($t_{\text{pre}} - t_{\text{post}} > 0$) STDP iteration, multiple synaptic weights connected to one presynaptic neuron can be updated at the moment of its presynaptic spike (t_{pre}) [3]. Therefore, non-transposable 6T synapse memory provides high throughput for post-then-pre STDP iteration where each presynaptic spike incurs updating row-wise weights. However, the throughput for the pre-then-post STDP iteration, where each postsynaptic spike incurs updating column-wise weights, is severely degraded.

To cope with the limitation, a transposable synapse memory structure based on a custom 8T SRAM bitcell was introduced [3]. However, the transposable 8T SRAM bitcell incurs significant area overhead compared to 6T SRAM bitcell due to the routing of additional WL and bit line (BL) for transposable access. In [7], a novel transposable addressing scheme leveraging 6T SRAM macros was proposed. However, the area overhead is still substantial due to duplicated address decoders and additional routings for control signals.

In this paper, we propose a 6T SRAM-based, area-efficient transposable synapse memory structure that provides the same bandwidth as the previous transposable synapse memories with much smaller area overhead; the proposed memory consumes only 17.7% larger area than the non-transposable 6T synapse memory.

II. Previous Works

Fig. 1 shows an exemplary crossbar synapse memory consisting of 16 synapses, where $w_{i,j}$ represents a single-/multi-bit synaptic weight from i-th presynaptic neuron A_i to j-th postsynaptic neuron D_j. In the synapse memory implemented using 6T SRAM bitcell, a row-wise weight vector, $\{w_{i,0}, w_{i,1}, w_{i,2}, w_{i,3}\}$ where i is the row address, can be concurrently accessed because any element in the vector does not share the same BL with the others. On the other hand, a column-wise weight vector, $\{w_{0,j}, w_{1,j}, w_{2,j}, w_{3,j}\}$ where j is the column address, needs to be accessed row-by-row in sequence because all the elements share the same BL. As a result, transposable access becomes very slow in the conventional 6T SRAM-based synapse memory.

In [3], a custom transposable 8T SRAM-based synapse memory (Fig. 1(c)) has been proposed, which allows concurrent access to column-wise weight vector through additional

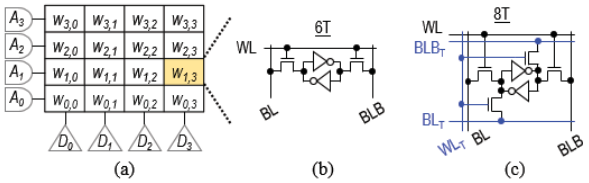


Fig. 1. (a) A simple crossbar synapse memory, (b) a 6T SRAM cell, (c) a transposable 8T SRAM cell [3].

* 978-1-5386-9395-7/19/$31.00 ©2019 IEEE

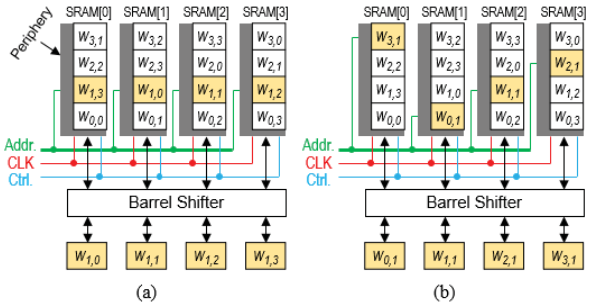


Fig. 2. Transposable addressing scheme using 6T SRAM macros [7]. (a) a row-wise access and (b) a column-wise access.

WL (WL_T) and BLs (BL_T and BLB_T). However, the additional routing of WL_T, BL_T, and BLB_T in the 8T SRAM bitcell incurs significant area overhead up to 250%.

To reduce the area overhead, a transposable memory addressing scheme based on 6T SRAM macros was proposed in [7] (Fig. 2). In the scheme, conventional 6T SRAM cell based arrays are still used to construct synapse memory, but the memory needs to be split into multiple blocks and synaptic weights are remapped so that the elements in the same column-wise weight vector are placed in the different blocks. As a result, all the elements in a column-wise weight vector as well as the row-wise weight vector can be read out simultaneously. Note that the elements in a row-wise weight vector can be read out simultaneously in both of conventional scheme and the scheme in Fig. 2. The transposable synapse memory in [7] reduced area overhead in [3] by using 6T SRAM instead of 8T. However, it still incurs considerable area overhead by occupying 83% larger area due to additional routing of clock, control, address, and data signals as well as the dedicated peripheral circuitry such as address decoder for each SRAM block as shown in Fig. 2.

III. Proposed Transposable Memory Structure

In this section, we propose an integrated 6T SRAM structure for transposable access. The proposed design is based on the transposable memory addressing scheme [7] but eliminates the need of duplicated address decoders and other control overhead by using the hierarchical WL structure. The key idea is to insert 2-to-1 multiplexers (MUXs) periodically on the word line path so that the row-wise access and the column-wise access can read out different set of SRAM cells depending on the *select* signal.

Fig. 3 shows the proposed transposable synapse memory structure, where transposable addressing scheme in [7] is implemented using a custom address decoder and row transition MUXs. The address decoder enables the WL corresponding to the input address for row-wise accesses (*Rb/C*=0) and the WL corresponding to 2's complement of the input address for column-wise access (*Rb/C*=1). The row transition MUXs are placed between two adjacent column arrays to connect the WLs in the same row for row-wise access and connect the WLs in the different rows for column-wise access. For example, when we access $row[1]$ weight vector $\{w_{1,0}, w_{1,1}, w_{1,2}, w_{1,3}\}$, the address decoder enables the WL for $w_{1,3}$. At the same time, all row transition MUXs connect the WLs on the same row to enable WLs for $w_{1,0}$, $w_{1,1}$, and $w_{1,2}$ concurrently (red arrows in Fig. 3). To access $col[1]$ weight vector $\{w_{0,1}, w_{1,1}, w_{2,1}, w_{3,1}\}$, the address decoder enables the WL for $w_{3,1}$ and all row transition MUXs select the WLs from the lower rows to access $w_{0,1}$, $w_{1,1}$, and $w_{2,1}$ concurrently (blue arrows in Fig. 3).

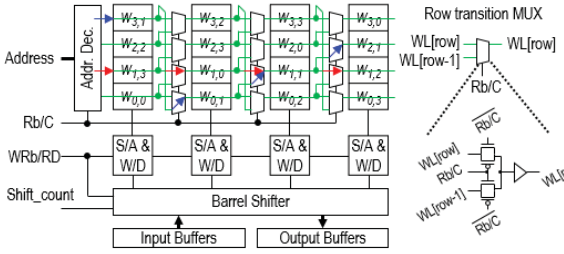


Fig. 3. The overall structure of the proposed transposable synapse memory.

The barrel shifter is needed to rearrange the order of output as the order of column index is different for each row in the proposed memory structure. The barrel shifter is placed between input buffers and write drivers for write operation and between sense amplifiers and output buffers for read operation to shift weights to the right and the left by *Shift_count*.

The row transition MUXs and the barrel shifter increases the area and access time, but the gain from the transposable access is larger than the overhead. The overhead analysis is given in the next section.

Meanwhile, memory array often adopts the column select structure, and the proposed WL row transition scheme needs to be modified when the column select structure is used. An example is shown in Fig. 4. In this case, the output of the address decoder is divided into two parts (MSBs for WLs and LSBs for column selection lines (CSLs)) and each part is sent to WLs and CSLs respectively. It is worthwhile to note that the operating principles of the row transition scheme for WLs and CSLs are different. For CSLs, similar to the case in Fig. 3, the row transition MUXs select the input from the lower rows when column-wise access is selected (Rb/C=1). On the other hand, the row transition MUXs for WLs select the input from the lower rows only when the MSB of the CSL signals in the previous column is "1" for column-wise access.

In this way, the proposed row transition scheme that embeds the transposable addressing scheme in memory array works correctly for any configuration of 6T SRAM-based crossbar

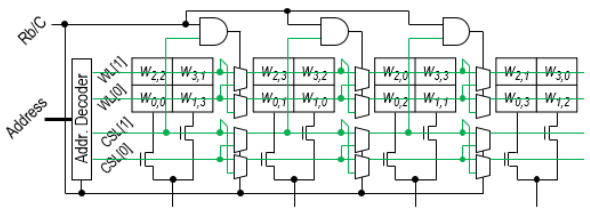


Fig. 4. The proposed row transition scheme for the memory configuration with column multiplexing.

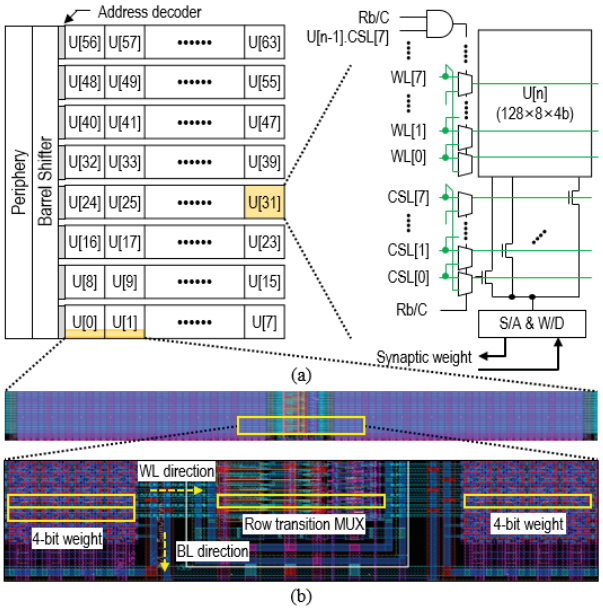


Fig. 5. (a) Configuration of the proposed transposable 64K-synapse memory and (b) the bottom left part of memory array in the layout design.

synapse memory.

IV. Testchip Implementation and Measurement

We implemented a testchip of transposable 64K-synapse (256×256 crossbar) memory with 4-bit weights in a 28nm CMOS technology. We reconfigured the 256×256 crossbar synapse memory with 4-bit weights into four 256×64 weight arrays and stacked them together as a 1024×64 configuration under the 256-bit I/O data width constraint.

A unit array corresponding to 1024×1 weights is implemented as a 128×8×4b cell array with 8-to-1 column multiplexing. 8 successive unit arrays are horizontally connected to configure a sub-array and the 64K-synapse memory consists of 8 sub-arrays (Fig. 5(a)). In all unit arrays except the left most unit array in each sub-array, the proposed row transition MUXs for WLs and CSLs are embedded for transposable access.

Compared to the baseline non-transposable 6T synapse memory, the proposed design has area overhead in the row transition MUXs and the barrel shifter. In the baseline synapse memory, the area portions of cell array, read/write circuits (sense amplifiers and write drivers), and periphery circuits are 84%, 10%, and 6% respectively. The proposed row transition MUX requires 5× larger width than that of the 6T SRAM bitcell while the cell height is same. Because there are 36 cells on a WL in a unit array including 4 dummy cells for layout regularity, the area overhead due to the row transition MUXs is 13.9% of the cell area in the unit array (Fig. 5(b)). In addition, the barrel shifter occupies almost the same area with the periphery circuits, which means the area overhead is 6.0%. Thus, overall area overhead of the proposed transposable memory is 17.7% (=13.9%×84%+6%). In the 8T synapse memory [3], the cell area increases by 250% and read/write circuit area increases by 200% from our analaysis. Thus, the estimated area overhead is 236% (=250%×84%+200%×10%+6%). Comparison data for memory area is shown in Table I.

TABLE I
Relative Area of Transposable Synapse Memories.

Synapse Memories	Baseline	8T [3]	6T [7]	Proposed
Relative Area	1.00	2.36	1.83	1.18

In the reconfigured transposable synapse memory, it takes 4 cycles to finish the read or write operation for 256 weights in both row-/column-wise directions (64 weights per cycle via 256-bit I/O data path). The number of cycles to read/write 256 weights in the row-wise direction is also 4 in the conventional non-transposable synapse memory. However, the number of cycles for read/write operations for the same number of weights in the column-wise direction is much larger in the non-transposable memory. The number of memory accesses required for 256 weights corresponding to a single column is 256 because every row needs to be accessed one-by-one. Note that the number of cycles to access multiple columns in the same 256×64 weight array is the same as the single column case because the weights that share the same word line can be accessed simultaneously. Hence, the number of access to multiple columns varies depending on the number of the 256×64 weight arrays that include the target columns. If n_{array} arrays have target columns, the required number of memory access is n_{array}×256.

Fig. 6 shows the post-layout simulation results for the delay of WLs in a sub-array. In the baseline synapse memory, the delay from the WL driver to the WL on the far-end cell of a sub-array is 206 ps while the delay in the proposed transposable memory is increased by 331 ps (=537–206 ps) due to the row transition MUXs. In addition, the delay penalty due to the barrel shifter is 450 ps. Thus, overall penalty in memory cycle time is about 780 ps, which is 26% of the cycle time (=3 ns) of the baseline synapse memory. In spite of this increase in delay, overall performance of the proposed memory is much higher than the conventional structure due to the reduced number of cycles for column-wise access. Detailed analysis will be shown in the next section.

Fig. 7 shows the die photograph, specifications, and measurement result of the testchip. 6T SRAM bitcell was manually designed using logic design rules to embed the proposed row transition scheme. If the bitcell from the manufacturer and SRAM design rules are available for the design of the row transition MUXs, the chip area can be reduced down to 1/3.

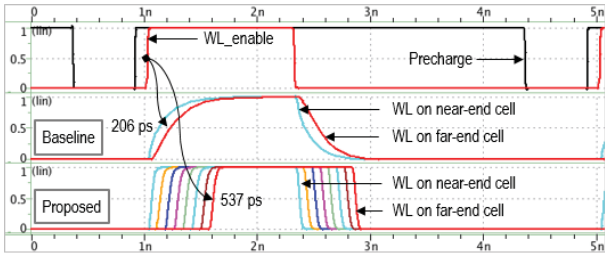
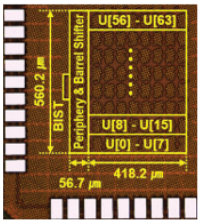
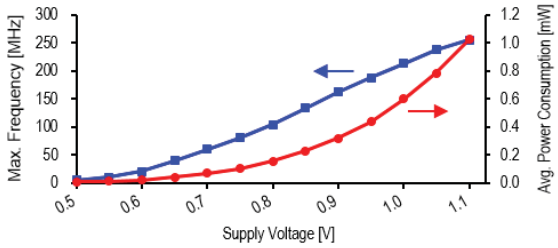


Fig. 6. Post-layout simulation results of the WL signals in a sub-array.



Technology	28nm CMOS
No. of synapses	64K (256×256 crossbar)
Area	0.266 mm²
Supply voltage	0.5V - 1.1V
Max. frequency	6MHz - 255MHz
Average power consumption	0.009mW - 1.023mW

(a)



(b)

Fig. 7. (a) Testchip die photograph, specifications, and (b) measurement results of operating frequency and power consumption.

The maximum operating frequency is 255 MHz at 1.1 V while consuming 1.023 mW (Fig. 7(b)).

V. Online Learning Performance Estimation

In a post-then-pre (or pre-then-post) STDP learning iteration of a 64K-synapse SNN, a single presynaptic (or postsynaptic) spike incurs synaptic weight updates on row-wise (or column-wise) 256 weights. Plot (1) in Fig. 8 shows the number of memory accesses to complete a post-then-pre STDP iteration (row-wise access) using the conventional non-transposable synapse memory. The data for the plot (1) is calculated as $n_{\text{spike}}\times4\times2$, where n_{spike} represents the number of presynaptic spikes in the iteration. Each spike needs 4 cycles to access 256×4 bits with the 256-bit I/O path as explained in the section IV. In addition, 2 cycles need to be multiplied to account for read-then-write operation.

Note that the number of accesses for both post-then-pre and pre-then-post iterations for the proposed transposable memory is same as the data in plot (1), which is the number of accesses for post-then-pre iterations for the non-transposable memory.

However, the number of accesses for pre-then-post iterations (column-wise access) for the non-transposable memory is much higher as shown in plot (2) in Fig. 8. The data for plot (2) is calculated as $n_{\text{array}}\times256\times2$. The n_{array} is the number of 256×64 weight arrays in which the indices of the postsynaptic spikes are included. Assuming that the spikes occur at neurons randomly, we can calculate the expected value of n_{array} for a given number of postsynaptic spikes.

Plot (3) in Fig. 8 shows online learning performance gain of the proposed transposable memory against the non-transposable memory when there are n_{spike} presynaptic and n_{spike} postsynaptic spikes in a STDP iteration.

Even with the penalty in the memory cycle time ($1.26\times$), it can be seen that the proposed transposable memory computes the STDP learning faster than the baseline scheme.

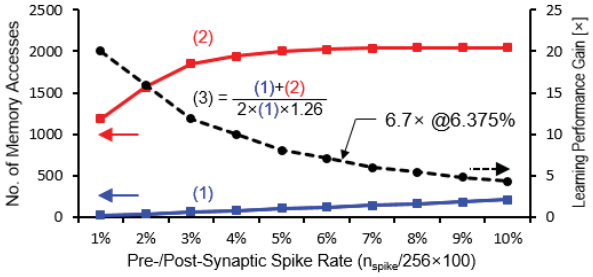


Fig. 8. Learning performance gain in the proposed transposable memory.

In the previous work on STDP learning [6], the reported firing rates are 0–63.75 Hz for MNIST dataset, which means the maximum pre/postsynaptic spike rate in a certain iteration of STDP learnings is 6.375% at 1 ms SNN simulation resolution. In such a condition, the online learning performance gain becomes $6.7\times$ (Fig. 8).

VI. Conclusion

We presented a 6-transistor SRAM-based transposable synapse memory for fast online learning in neuromorphic processors. Based on the hierarchical word line with row transition MUX structure, the proposed design enables the transposable memory addressing in the integrated SRAM array structure. Detailed design methodology for row transition scheme in various SRAM array configurations such as column multiplexed array was also introduced. The proposed memory shows $6.7\times$ higher performance for STPD learning with MNIST dataset than the conventional non-transposable memory. The area overhead of the proposed design over the non-transposable memory was 17.7% only, which is much smaller than the area overhead in the previous works.

Acknowledgment

This research was supported by Samsung Research Funding Center of Samsung Electronics under Project Number SRFC-TC1603-04 and the MSIT (Ministry of Science and ICT), Korea, under the ICT Consilience Creative program (IITP-2018-2011-1-00783) supervised by the IITP (Institute for Information & communications Technology Promotion).

References

[1] Z. Du *et al.*, "Neuromorphic accelerators: a comparison between neuroscience and machine-learning approaches," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2015, pp. 494–507.

[2] E. M. Izhikevich, "Polychronization: computation with spikes," *Neural computation*, vol. 18, no. 2, pp. 245–282, 2006.

[3] J. Seo *et al.*, "A 45nm cmos neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in *IEEE CICC*, 2011, pp. 1–4.

[4] F. Akopyan *et al.*, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, 2015.

[5] M. Davies *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[6] P. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in computational neuroscience*, vol. 9, p. 99, 2015.

[7] J. Kim *et al.*, "Efficient synapse memory structure for reconfigurable digital neuromorphic hardware," *Frontiers in Neuroscience*, vol. 12, p. 829, 2018.