8T-SRAMを用いた同時2入力可能な2値化ニューラルネットワーク用インメモリアクセラレータ

田形 寛斗[1,a)]　佐藤 高史[1]　粟野 皓光[1]

概要：本研究では，8T-SRAM を用いた 2 値化ニューラルネットワーク用インメモリアクセラレータを提案する．提案回路は一般的には相補的に利用されるビットラインをそれぞれ独立に用いることで，2 つの入力について XNOR を同時に計算可能とし，積和演算の実行速度を最大 2 倍にした．トランジスタレベル・シミュレーションの結果，93.88 %の MNIST 分類精度を達成し，既存研究と比べ消費エネルギーを 29 %削減できることを確認した．

8T-SRAM based Dual input In-Memory Accelerator for Binarized Neural Network

Hiroto Tagata[1,a)]　Takashi Sato[1]　Hiromitsu Awano[1]

Abstract: This paper proposes an in-memory accelerator for binarized neural networks using 8T-SRAM. The proposed circuit uses independent bit lines, which are generally used in complementary manner, to simultaneously compute XNOR for two inputs, thus doubling the performance of the sum-of-products operation at the maximum. The transistor-level simulation shows that the proposed circuit achieves 93.88% MNIST classification accuracy while reducing the energy consumption by 29% compared to existing studies.

1. まえがき

深層ニューラルネットワーク (DNNs) の発見により，曖昧さを伴う現実世界の情報をコンピュータ上で取り扱うことが可能となった．DNNs はすでに一部のタスクにおいて人間を凌ぐ性能を発揮しており，画像認識，ロボット制御，自然言語処理などへの応用が探求されている．一方で，DNNs は実行時に莫大な量の積和演算やメモリアクセスを必要とするため，電力やスペースに制限のある組込みシステムへの実装が課題となっている．そこで，近年では DNNs の推論に必要な消費電力や演算の実行時間を削減する手法として，Computing-in-memory (CIM) という手法が活発に研究されている．

CIM はデータを保存するメモリ内部に演算回路を組み込む方式である．CIM は演算時のメモリアクセスを必要としないため，従来のノイマン型アーキテクチャと比べ消費電力や実行速度を大幅に削減できる．CIM 回路の概略図を図 1，2 に示す．多くの CIM では電圧値や電流値を用いたアナログ演算回路により各メモリセルによる演算結果の加算を高速化している [1–5].

本論文では，同時 2 入力に対応する 8T-SRAM を用いたインメモリアクセラレータを提案する．提案回路では SRAM に接続された 2 本のビットラインを互いに独立させ，それ

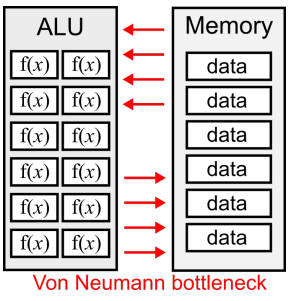

図 1　ノイマン型アーキテクチャ

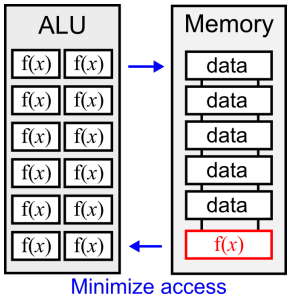

図 2　Computing-in-memory

[1] 京都大学大学院情報学研究科
Graduate School of Informatics, Kyoto University, Yoshida-hon-machi, Sakyo, Kyoto 606–8501, Japan
[a)] htagata@easter.kuee.kyoto-u.ac.jp

ぞれでアナログ演算を行うことで 2 つの入力について同時に積和演算を実行できる．これにより，既存回路と比べ同程度の回路面積と演算時の消費電力を維持しつつ，積和演算のスループットを 2 倍にすることができる．

本稿の構成は以下の通りである．2 章では今回の回路で用いるニューラルネットワークアルゴリズムである Binarized Neural Networks (BNNs) と既存の SRAM インメモリアクセラレータについて述べる．3 章では提案する 2 入力 8T-SRAM の基本動作について述べ，4 章で提案回路である 8T-SRAM アクセラレータ全体と周辺回路について説明する．5 章にて提案回路の SPICE シミュレーション結果をもとに，MNIST 精度，消費エネルギーの両面で既存回路のと比較評価を行う．最後に，6 章で本論文をまとめる．

2. 2 値化ニューラルネットワークと既存の SRAM インメモリアクセラレータ

2.1 Binarized Neural Networks

BNNs はネットワークのシナプス結合重みと活性化層の出力を共に +1 と −1 の 2 値で量子化したものである [6]. BNNs では積和演算時の各ニューロンでの乗算は 1 ビットの XNOR 演算，加算はビットカウントで実装できるため，演算の計算量や必要なメモリの数を大幅に削減できる．よって，電力や回路面積の制限が大きいエッジデバイスでの DNNs の利用において注目されている．また，BNNs はネットワークの量子化による精度低下が発生するものの，既存研究では実用的な範囲に収まるとされている [7, 8].

BNNs の学習を安定させるために多くの場合でバッチ正規化 (Batch-normalization; BN) が用いられているが，推論時に BN 演算をそのまま実装すると浮動小数点演算が必要となり回路面積が増大する．そこで，Umuroglu らは BN 演算を活性化層の 2 値化関数のしきい値として組み込む (Batch-norm activation as threshold; BAT) ことで，BN 層のハードウェア実装にかかるコストを大幅に削減した [9].

2.2 BNNs 向け SRAM インメモリアクセラレータ

この節では BNNs の推論演算を対象としたインメモリアクセラレータを紹介する．ここでは特に CMOS メモリである SRAM を利用したものに注目し，既存研究の利点と欠点を比較する．

6T-SRAM アクセラレータ：6T-SRAM は 2 つのインバータで構成されるラッチと 2 つのアクセストランジスタで構成された SRAM である (図 3)．6T-SRAM は様々なメモリ回路で実用化されているため大規模に生産できる点や，他の SRAM と比べ回路面積が小さい点で優れている．

先行研究では活性化層を −1/+1 ではなく 1/0 で 2 値化した Modified BNN (MBNN) を用いた 6T-SRAM アクセラレータが提案されている [4]．表 1 に回路動作を示す．この回路では，WL への入力と 6T-SRAM に保存された重みに従ってプリチャージされた BL と BLB のうち一方の電荷を引き抜くことで積和演算を実行する (電流モード)．しかし，MBNN は通常の BNN と比べ積和演算結果の分布範囲が狭いためにアナログ演算による精度低下の影響を受けやすい．また，演算時に BL と BLB の電圧値が減少するため，出力値によっては SRAM に保存されている値が意図せずフリップしてしまうディスターブ不良が発生する可能性があり安定性に劣る．別のアプローチとして，アクセストランジスタをそれぞれ分割した dual-split 6T-SRAM を用いたアクセラレータが提案されている (図 4) [2]．この回路では入力と重みに従い各 SRAM をプルアップ (PU)/プルダウン (PD) ドライバとして機能させ，最終的な PU と PD の割合により出力電圧を生成する (電圧モード)．この回路では通常の BNNs を実行できるものの，電圧モードの演算は電流モードのものと比べ消費電力が大きいという欠点がある．一方で，演算時にアクセストランジスタの一方がオフになっているため，通常の 6T-SRAM と比べディスターブ不良は発生しにくくなっている．

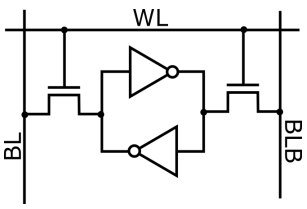

図 3 6T-SRAM

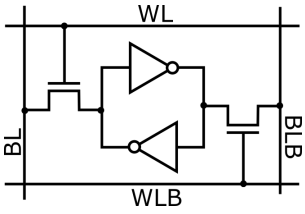

図 4 dual-split 6T-SRAM [2]

Input	Weight	Output	V(BL)	V(BLB)
0	−1	0	0	0
0	+1	0	0	0
+1	−1	−1	0	−ΔV
+1	+1	+1	−ΔV	0

表 1 6T-SRAM の動作

8T-SRAM アクセラレータ：8T-SRAM は 6T-SRAM にトランジスタを追加することで演算性能や安定性を向上させたものであり，キャパシタを利用したものなど様々なものが提案されている．ここでは標準的なトランジスタのみを用いたものを紹介する．図 5 は電流モードで XNOR 演算を実行可能な 8T-SRAM である [3]．回路動作を表 2 に示す．この回路はトランジスタ 2 つ分というわずかなオーバーヘッドで演算性能を向上しているが，6T-SRAM と同様にディスターブ不良が発生する可能性がある．安定性の観点では図 6 に示す回路が提案されている [5]．この SRAM は読み出し時にビットラインと Q, QB が直接接続されないため，電流モードで演算を実行した場合でもディスターブ不良の発生を抑えることができる．しかし，実行可能な演算は 6T-SRAM と同様 MBNN であるという課題が残る．

本研究では，図 5 に示す回路をベースとした新しい 8T-SRAM を提案する．提案回路は SRAM の入力を分割する

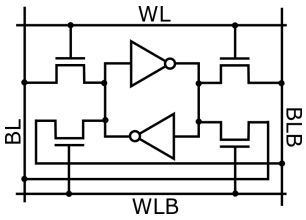

図 5　8T-SRAM [3]

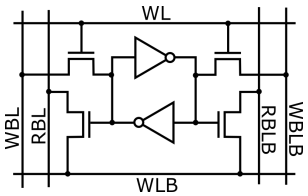

図 6　8T-SRAM [5]

Input	Weight	Output	V(BL)	V(BLB)
−1	−1	+1	−ΔV	0
−1	+1	−1	0	−ΔV
+1	−1	−1	0	−ΔV
+1	+1	+1	−ΔV	0

表 2　8T-SRAM [3] の動作

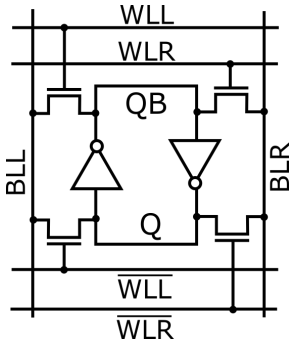

図 7　提案 8T-SRAM メモリセル

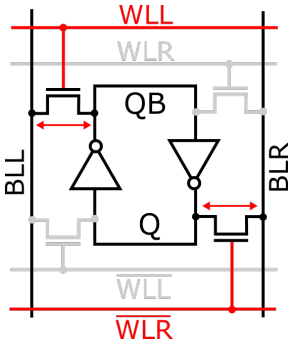

図 8　書き込み動作

Transistors	Width [nm]
n-MOS (latch)	200
p-MOS (latch)	200
n-MOS (access)	150

表 3　SRAM のサイジング

ことにより，1) 演算のスループット及び消費電力の改善，2) 演算時のディスターブ不良の抑制，を実現する．

3.　2 入力 8T-SRAM メモリセルとその動作

提案する 8T-SRAM セルを図 7 に示す．この回路は 2 本のビットライン (BLL，BLR) と，2 組の入力 (WLL，WLLB) と (WLR，WLRB) を持つ．ここで，WLL と WLR はそれぞれ独立した入力である．また，WLLB は WLL の反転入力，WLRB は WLR の反転入力である．提案回路は 2 対のアクセストランジスタを持ち，BLL と BLR はそれぞれ Q と QB の両方に接続されている．今回想定したトランジスタのサイズを表 3 に示す．チャネル長はすべて 60 nm である．書き込み動作を安定させるため，アクセストランジスタの n-MOS はラッチ内の p-MOS より電流を流しやすくした．また，データ読み出し時及び推論持のデータ反転を抑えるため，ラッチ内の n-MOS はアクセストランジスタの n-MOS より電流を流しやすくした．

書き込み時の動作：回路の書き込み時の動作を図 8 に示す．WLL と WLRB (または WLR と WLLB) を同時にオンにし，Q と BLR (BLL)，QB と BLL (BLR) を同時に接続することでラッチの値を決定する．

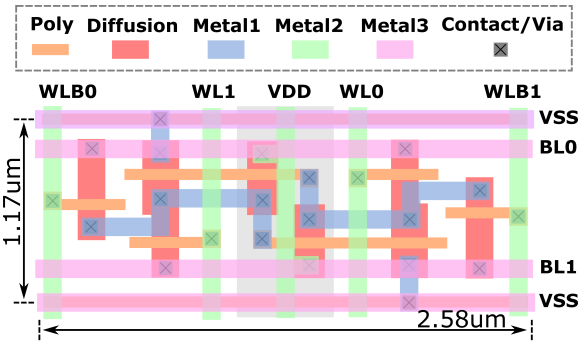

図 9　レイアウト

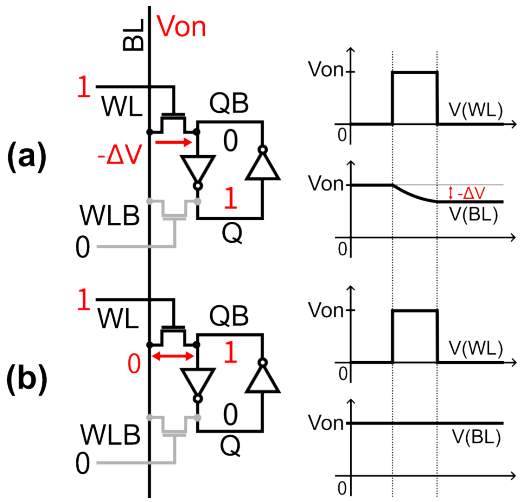

図 10　XNOR 演算動作

XNOR 演算の動作：以下, BLL と BLR, 及び WLL (WLLB) と WLR (WLRB) はそれぞれ独立であるため, 区別せず BL, WL (WLB) と表記する．演算時，入力は WL 電圧を用いて表現され，+1 ならば WL にオン電圧，WLB にオフ電圧を入力し，−1 ならば WL にオフ電圧，WLB にオン電圧を入力する．また，ネットワークの重みは SRAM に保存されている値を用いて表現され，+1 ならば Q をオン電圧，QB をオフ電圧とし，−1 ならば Q をオフ電圧，QB をオン電圧とする．演算時には BL はオン電圧にプリチャージされている．提案 SRAM を用いた XNOR 演算動作を図 10 と表 4 に示す．入力が +1，重みが +1 の場合，WL に接続された NMOS トランジスタがオンとなり BL と QB が接続される．この際，あらかじめオン電圧にプリチャージされた BL から QB へ電流が流れることで BL 電圧値が減少する (図 10(a))．この電圧値の減少分を +1 の出力とみなす．次に，入力が −1，重みが +1 の場合は，WLB に接続されたトランジスタがオンとなり BL と Q が接続される．ここで，Q と BL の電圧値が釣り合っているため電流は流れず，BL 電圧は変化しない．これを −1 の出力とみなす．

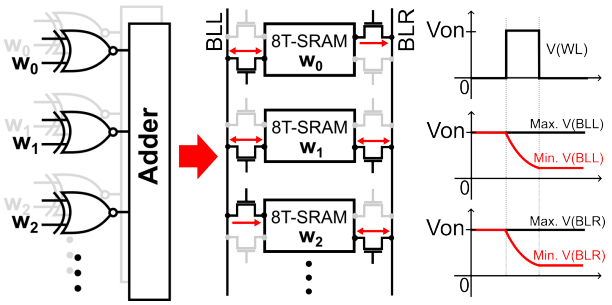

図 11　XNOR 演算結果の加算

Input	Weight	Output	V(BL)
−1	−1	+1	−ΔV
−1	+1	−1	0
+1	−1	−1	0
+1	+1	+1	−ΔV

表 4　提案 8T-SRAM の動作

演算結果の加算：積和演算では，各セルでの XNOR 演算の結果をすべて加算する必要がある．提案回路では 100 個のセルを同一の BL に並列接続し，電流モードによる加算を行う (図 11).

4.　2 入力 8T-SRAM アクセラレータの全体図

提案するアクセラレータの全体図を図 12 に示す．提案アクセラレータは 100 行 100 列の計 10,000 個の 8T-SRAM を持ち，各列の SRAM はそれぞれ並列にビットライン (BLL, BLR) に接続され，各行の SRAM は WLL，WLR，WLLB，WLRB に接続されている．また，周辺回路として 400 個のパルス生成回路，200 個のコンパレータ，400 個のオフセット調節セルがある．

積和演算時における回路全体としての動作を以下に示す．入力信号は 2 値化されているものとし，+1 はオン電圧 (V_{dd})，−1 はオフ電圧 (V_{ss}) で表現するとする．

(1)　回路内の SRAM セルに，使用するネットワークの結合重みに応じた値を書き込む．
(2)　充電回路を用いて回路内すべてのビットラインをオン電圧まで充電する．
(3)　入力 1，2，及びそれらの反転入力を入力信号に従い電圧値をオン電圧かオフ電圧に立ち上げる．
(4)　入力された信号をパルス生成回路へ入力し，幅の短いパルス波に変換する．
(5)　入力 1 は WLL, WLLB に，入力 2 は WLR，WLRB を通して 8T-SRAM へと入力する．この際，SRAM に保存されている値の反転を防ぐため，入力 1 に対して入力 2 を十分に遅延させてから入力する．
(6)　提案 8T-SRAM セルにより入力と SRAM に保存されている値との間で XNOR 演算を行い，電流モードによる演算結果の加算を行う．同時に，オフセット用 SRAM をすべてオンにし，しきい値に応じてビットラインの電荷を引き抜く．
(7)　すべてのセルで演算が終了し，BLL と BLR の電圧値が確定した後にコンパレータの制御信号をオンにする．コンパレータは外部から入力されているしきい値電圧と BL 電圧を比較し，結果が +1 ならオン電圧，−1 ならオフ電圧を出力する．

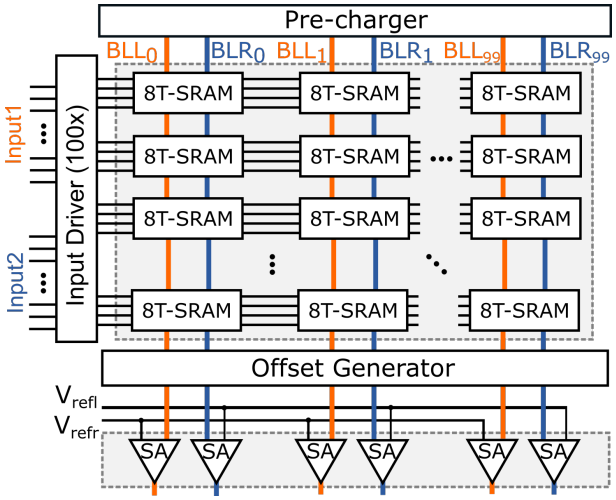

図 12　提案アクセラレータの全体図

4.1　周辺回路

入力パルス生成回路：提案アクセラレータでは，多数の SRAM がビットラインの電荷を同時に引き抜くことで積和演算を実行している．しかし，各 SRAM が引き抜く電荷量の合計がビットラインの充電容量より大きい場合，ビットラインの電荷が完全に放電されてしまい，演算精度の極端な低下や SRAM のディスターブ不良が発生する．そのため，提案アクセラレータでは推論時に入力信号をパルス化し，8T-SRAM のアクセストランジスタがオンである時間を短くすることで各 SRAM がビットラインから引き抜く電荷量を小さくしている．

パルス生成回路の回路図を図 13 に示す．この回路は入力信号の立ち上がり時に，入力信号と入力信号を反転し遅延させた信号を AND 演算し，遅延時間と同じパルス幅を持つパルスを生成する (図 14)．回路内の遅延ゲートはパスゲートスイッチによって実装され，外部からゲート電圧を操作することで遅延量を制御できる．推論時に 8T-SRAM セルへと入力されるすべての信号は，各 WL に接続されたこの回路を通してパルスに変換されてから入力される．

コンパレータ：各ビットラインに接続されるコンパレータの回路図を図 15 に示す．今回用いたコンパレータは入力として，入力信号 V_{BL}，比較信号 V_{Ref}，バッチ正規化用信号 BN，制御信号 EN を持つ．入力信号はビットライン電圧を，比較信号として 2 値化しきい値電圧を入力する．バッチ正規化用信号 (BN 信号) は，2.1 節で述べたバッチ正

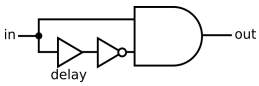

図 13　入力パルス生成回路

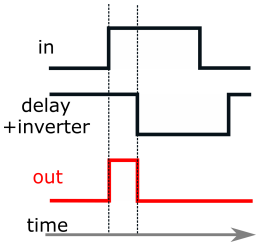

図 14　入力パルス生成回路の動作

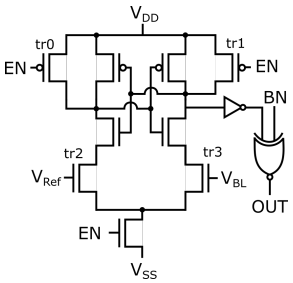

図 15　コンパレータ回路

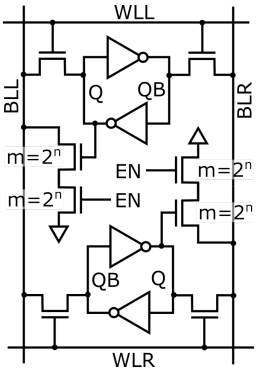

図 16　オフセット用 SRAM セル

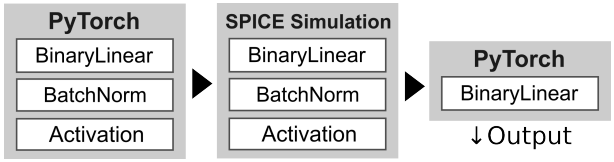

図 17　シミュレーション環境

規化層の学習重みの符号である．BN 信号はコンパレータ外部の SRAM に保存され，入力信号と比較信号の電圧値で決まるコンパレータ内部のラッチの値と共に XNOR ゲートへ入力される．コンパレータは制御信号がオフの時は待機状態であり，オンになった時点での比較結果を出力する．コンパレータの動作を以下に示す．(1) 制御信号 EN をオフにする．tr0 と tr1 によりコンパレータ内部のラッチは共にオン電圧にプルアップされる．(2) 入力信号と比較信号にそれぞれ電圧を印加する．(3) 制御信号 EN をオンにし，tr0 と tr1 をラッチから分離する．入力信号と比較信号の電圧値により，tr2，tr3 が電流を流す強さが決まり，ラッチの値が定まる．ラッチは信号と出力を分離するためのインバータを通して出力され，XNOR ゲートに入力される．(4) ラッチからの出力は BN 信号の値と XNOR 演算され，演算結果が出力される．

オフセット用 SRAM セル：オフセット用 SRAM セルは BAT を利用した際に，コンパレータのしきい値を変更するためのセルである．回路内すべてのコンパレータに外部から比較電圧を与えることが配線上困難であるため，オフセット用 SRAM を用いてビットラインの電圧値を直接制御することでしきい値を調整する．

オフセット用 SRAM セルの構成を図 16 に示す．オフセット用 SRAM は 2 対の SRAM で構成され，一方は BLL に，他方は BLR に接続される．ここで，SRAM のラッチとビットラインを直接接続すると，n が大きくなった場合に SRAM 内部のデータが反転する恐れがあるため，通常の 6T-SRAM に 2 つのアクセストランジスタを追加して安定性を高めた．引き抜く電荷量を大きくする際はアクセストランジスタのサイズのみを大きくすればすれば良いため，n が大きくなった場合の面積増加率を低減している．加えて，ビットラインから引き抜く電荷量を通常の 8T-SRAM セルの 2^n 倍となるように設計することで，SRAM セルの個数を削減している．

4.2　キャリブレーション

すべてのコンパレータ間で同一の比較電圧を用いた場合，トランジスタの製造ばらつきにより同じ重みと入力を与えてもビットラインごとに出力値がばらついてしまう．そのため，オフセット用 SRAM セルを用いてコンパレータのしきい値を適切な値に設定する必要がある．キャリブレーションは以下の二部探索アルゴリズムを用いて行う．(1) オフセット用 SRAM セルの値のうち，最もサイズの大きいものに 1，それ以外のものに 0 を書き込む．(2) 積和演算結果が 0 となるような重みと信号を入力し，コンパレータの出力を確認する．この時，出力が +1 であるビットラインのオフセット用 SRAM すべてに −1 を，出力が −1 であるビットラインには +1 を書き込む．(3) 以下，各ビットラインについて出力が反転するまでオフセットの値を単調に増減させていき，出力が反転した際のオフセットの値を最終結果とする．

5.　数値実験

5.1　SPICE・PyTorch 統合シミュレーション

BNNs を用いた線型結合ネットワークを提案回路上で構築し，トランジスタレベルでシミュレーションを行った．ただし，ネットワークの全層に渡って SPICE シミュレーションを行うのは困難であるため，中間層のうち 1 層のみを提案回路のモデルに置換し，その他の部分は Python の PyTorch ライブラリで実装する協調シミュレーションとした (図 17)．提案回路のシミュレーションではすべてのデータが電圧値で扱われるため，ソフトウェアとのデータ受け渡しの際には電圧値 (オン電圧，オフ電圧) とデジタル値 (+1, −1) を相互に変換した．実験では，ネットワークを提案回路に置換したことによる MNIST の分類精度への影響と，画像 1 枚を推論する際に提案回路で消費されるエネルギーを計算した．

5.2　シミュレーション条件

実験には入力層，中間層，出力層としてそれぞれ 1 つの

	提案回路	既存回路	ソフトウェア
MNIST 分類精度 [%]	93.88	94.39	95.38
消費エネルギー [pJ]	40.26	56.55	–

表 5 MNIST 精度と消費エネルギーの比較

線型結合層をもつ 3 層ニューラルネットワークを用いた．各層はそれぞれ 100 個のニューロンを持つ．入力層，中間層の直後にはバッチ正規化を行い，活性化関数には 2 値化関数を用いた．

Python の PyTorch ライブラリと MNIST データセットを用いて，上記のネットワークのシナプス結合重みを学習した．分類精度の極端な悪化を防ぐため，入力画像は 8 ビットのままで入力した．また，第 2 層の線型結合結果に対し平均値 0，標準偏差 5 のガウス分布に従うノイズを加えた．PyTorch 上での最終的な MNIST 精度は 95.38 %であった．

以下に実装したアクセラレータの構成をまとめる．製造プロセスは 65nm で，各トランジスタにはガウス分布に従うしきい値ばらつきを設定した．オン電圧は 1.2 V，オフ電圧は 0.0 V とした．オフセット用 SRAM は $n = 3$ までのものを用いて，−14 から +16 までの間でしきい値を調節できるようにした．以上の条件で MNIST の推論画像 10,000 枚について回路全体のモンテカルロ・シミュレーションを各 1 回ずつ行った．また，既存の 8T-SRAM [3] を用いて同一条件で比較を行った．

5.3 結果と考察

提案回路を用いた場合の MNIST 分類精度は 93.88 %，消費エネルギーは 40.26 pJ であった．また，既存回路，及び PyTorch 上での精度と比較した結果を表 5 に示す．提案回路は既存回路に匹敵する精度を出しつつ，消費エネルギーを 28 %削減できた．提案回路での精度低下の原因としては，以下が考えられる．(1) トランジスタのばらつきにより発生するコンパレータのオフセットを完全に補正できていない．(2) 既存回路は各列それぞれにおいて 2 本のビットラインを用いた相対的な電圧比較であるのに対し，提案回路は外部からすべての列に対して同一の比較電圧を入力する絶対的な電圧比較である．そのため，提案回路では各ビットラインごとのトランジスタばらつきの影響が大きくなり，精度が低下している．これらの問題はオフセット調節セルの個数を増加することにより対処できるが，回路規模，消費電力が増大する点や，ビットライン電圧の低下に注意が必要となる．

6. 結論

本論文では，2 つの積和演算を同時に実行できるインメモリコンピューティング回路を提案し，トランジスタレベルのシミュレーションにより MNIST の分類精度と 1 推論あたりの消費エネルギーの評価を行った．提案回路は 2 つの入力を持つ 8T-SRAM の他，パルス生成回路，ビットライン充電回路，コンパレータ，及びオフセット調節回路で構成され，8T-SRAM をシナプスとして利用することで 2 値化ニューラルネットワークの積和演算を行う．既存回路では相補的に用いていたビットラインを独立に用いることで，面積の増加を抑えつつ積和演算の計算速度を最大で 2 倍にした．提案回路のシミュレーションでは，MNIST の分類精度は 93.88 %と既存回路と比べ遜色のない精度を実現し，消費エネルギーは 29 %削減できた．

今後の展望として，提案回路をハードウェア実装し，計算の精度や消費エネルギー，シミュレーションでは確認できなかった大規模なネットワークを用いた推論について確認したい．

謝辞

本研究は，JSPS 科研費 21H03409 の支援を受けたものである．

参考文献

[1] J. Zhang, Z. Wang, and N. Verma, “A machine-learning classifier implemented in a standard 6T SRAM array,” Symposium on VLSI Circuits, pp.1–2, 2016.

[2] X. Si, W.-S. Khwa, J.-J. Chen, J.-F. Li, X. Sun, R. Liu, S. Yu, H. Yamauchi, Q. Li, and M.-F. Chang, “A dual-split 6T SRAM-based computing-in-memory unit-macro with fully parallel product-sum operation for binarized DNN edge processors,” IEEE Trans. Circuits Syst. I, vol.66, no.11, pp.4172–4185, 2019.

[3] H. Jiang, R. Liu, and S. Yu, “8T XNOR-SRAM based Parallel Compute-in-Memory for Deep Neural Network Accelerator,” Int. Midwest Symp. on Circuits and Syst. (MWCAS), pp.257–260, 2020.

[4] R. Liu, X. Peng, X. Sun, W.-S. Khwa, X. Si, J.-J. Chen, J.-F. Li, M.-F. Chang, and S. Yu, “Parallelizing SRAM arrays with customized bit-cell for binary neural networks,” Design Automat. Conf. (DAC), pp.1–6, 2018.

[5] C. Yu, T. Yoo, K.T.C. Chai, T.T.-H. Kim, and B. Kim, “A 65-nm 8t sram compute-in-memory macro with column adcs for processing neural networks,” IEEE Journal of Solid-State Circuits, pp.••–••, 2022.

[6] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, “Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or −1,” CoRR, p.arXiv:1602.02830, 2016.

[7] M. Courbariaux, Y. Bengio, and J.-P. David, “BinaryConnect: Training Deep Neural Networks with binary weights during propagations,” Int. Conf. on Neural Inf. Process. Syst. (NeurIPS), pp.3123–3131, 2015.

[8] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, “XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks,” Eur. Conf. on Comput. Vision (ECCV), pp.525–542, 2016.

[9] Y. Umuroglu, N.J. Fraser, G. Gambardella, M. Blott, P. Leong, M. Jahre, and K. Vissers, “Finn: A framework for fast, scalable binarized neural network inference,” Proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays, pp.65–74, 2017.