Analog In-Memory Compute using SST SuperFlash

Mark Reiten
Artificial intelligence based on advanced machine learning models has gained tremendous momentum in many industries. Many machine learning optimized digital processing solutions have been introduced in the last 5 years, but none can match the power and performance advantages of analog memory-based computing devices. SST’s memBrain cost/performance can be 10 times better and the power can be 100x lower than a comparably performing digital solution. This is accomplished by storing multiple levels (up to 256) per cell to represent a “weight” or “synapse” in a neural model. Multiplication is done through cell operation and addition is done by summing the output lines. Vector Matrix Multiplication is accomplished through design techniques so any existing SuperFlash process can support this new optimized compute paradigm.
Agenda

• Neural Systems and Uses
• Deep Neural Networks: *The Problem*
• Analog vs Digital compute
• What is memBrain
• memBrain™ Analog Inference Designware
• Many Model Types can be supported
• Why it works
• Software Flow
Neural Systems and Uses

Training

- Massive input data sets required
- Requires floating point math
- Many training iterations needed
- GPUs or FPGAs or TPUs or... best

Inference

- Only simple math needed
- High parallelism needed for **low latency**
- Memory with embedded processing is best
- ESF1 or ESF3 is best solution!

---

Flash Memory Summit 2019
Santa Clara, CA
Deep Neural Networks: *The Problem*

DNNs require vast numbers of Multiply-Accumulate operations (MACs)

<table>
<thead>
<tr>
<th>Network</th>
<th>Weights</th>
<th>MACs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alexnet</td>
<td>61 M</td>
<td>725 M</td>
</tr>
<tr>
<td>ResNet 50</td>
<td>23 M</td>
<td>3.5 B</td>
</tr>
<tr>
<td>VGG-19</td>
<td>46 M</td>
<td>22 B</td>
</tr>
</tbody>
</table>

Vast numbers of MAC operations favors keeping weights in local storage.

Cannot fit these into a stand alone digital edge processor.
Analog vs Digital Compute

Analog advantages:

Data and compute compression

No Data thrashing!

In neural networks, involving simple operations on large data sets, time needed to read and write computation registers will dwarf actual computation time.

Bus traffic can occupy >90% of the time required to process the calculations.
What is memBrain?

solves the compute problem by storing the weights in eFlash and using analog cell operation to perform the MAC operations inside the storage array.

Each cell stores up to 8 bits in 1.5 transistors. Multiply happens through cell operation.

Output line functions as a Neuron. Summation happens along output.

Compared to 48 SRAM transistors for 8 bits!

Power is ~.3pj per MAC!!
memBrain™ Analog Inference Designware

512x512 tile full frame cycle time 10-30μs
- Depends on D-A and A-D power

Energy is 0.3pJ per MAC with D/A+A/D @ 30us frame cycle time

Area with D to A input and A to D output blocks = .48 mm^2 on 40nm for 512x512 Tile

memBrain uses the SuperFlash cell as a multi-level analog device storing 4-8 bits per cell depending on the application

memBrain is organized as “Tiles” with wide I/O configurations supporting massively parallel multiply/accumulate operations

Performance per silicon area and power are orders of magnitude better than optimized digital solutions
Many Model Types can be supported

A neural network using *In-memory analog computing assembles from memBrain “TILES”*
Why it Works

ESF3 Cell analog operation varies the floating gate potential with repeatable consistent separation between steps across array and temp.

ESF3 Cell data retention is excellent
Noise is low

Noise and temperature can be compensated for
Software Flow

Raw Data
Data can be images, natural language text, audio or virtually anything

Data cleansing and annotation
Humans or machines clean and annotate data to describe desired features

Model Training
Modern solutions use software frameworks and optimized hardware

Model Testing
Check the trained, quantized model on new data to prove accuracy

Load model into memBrain
Generate a bit map which can be programmed into the memBrain device

Python Script kit and examples for weight quantization and noise injection prior to testing

Programming tool for loading Bitmap into memBrain Tile
Thank You