Mitigating the Memory Wall with mMPU – memristive Memory Processing Unit

Guest Lecture by Nishil Talati

046278 – Accelerators and Accelerated Systems

26th June 2017
A Bit about Our Group

• ASIC$^2$
  • Leader: Prof. Shahar Kvatinsky
  • Exploring various areas of Electrical and Computer engineering using emerging memory technologies (which we like to call memristors)
  • Memristor – memory + resistor
    • PCRAM – Phase Change RAM
    • RRAM – Resistive RAM
    • STT-MRAM – Spin-Torque Transfer Magnetic RAM

• Nishil Talati
  • B.E. (Hons.): Electrical Engineering, BITS Pilani, India
  • Currently a graduate student at the EE, Technion as a part of ASIC$^2$ lead by
What is the Best Time to be a Computer Architect?
What is the Best Time to be a Computer Architect?
Challenges Faced by Today’s Computers

• Demise of Moore’s and Dennard’s law
• Memory wall problem and von Neumann bottleneck
• Main memory systems incapable of delivering many metrics as required by modern workloads and processing engines
Sources of Inefficiencies in von Neumann Machines

Latency for Data Transfer

von Neumann Machine

Memory (DRAM)

CPU

Performance Gap

Memory

CPEU

(2X: 2 years)

(2X: 10 years)

Time

Performance

Memory Wall Problem
Sources of Inefficiencies in von Neumann Machines

Energy of Data Transfer

<table>
<thead>
<tr>
<th>Operation</th>
<th>Energy/Op (45nm)</th>
<th>Cost (vs. Add)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add operation</td>
<td>0.18 pJ</td>
<td>1X</td>
</tr>
<tr>
<td>Load from on-chip SRAM</td>
<td>11 pJ</td>
<td>61X</td>
</tr>
<tr>
<td>Send to off-chip DRAM</td>
<td>640 pJ</td>
<td>3556X</td>
</tr>
</tbody>
</table>

Challenges Faced by Today’s Main Memory Systems

• Main memory designed using DRAM
• Enhancement in memory capacity, bandwidth, and QoS
• Main memory energy is a key system design concern
• DRAM scaling is ending
Opportunities for Today’s Computer Architects

• Our responsibility is to solve these problems in several ways...
  • Novel computing paradigms
  • CPU-free heterogeneous system design
  • Domain specific accelerator designs
  • Novel methods of data storage (e.g., DNA for archival storage etc.)
  • Processing-In-Memory (PIM) design
Emerging Resistive Nonvolatile “Memory” Technologies

- RRAM
- PCM
- STT MRAM
- CBRAM
Using these Technologies from von Neumann to Processing-in-Memory (mMPU) Architecture

Load-Store Architecture

CPU

Memory

CMD, ADDR

DATA

CMD = LOAD/STORE

mMPU Architecture

CPU

mMPU

Data

CMD, ADDR

Reduced BW

Enabling PIM

CMD = LOAD/STORE + ADD/MUL/…

R. Ben-Hur and S. Kvatinsky, "Memory Processing Unit for In-Memory Processing," NANOARCH, July 2016
Role of mMPU in Designing Future Architectures at ASIC2

• mMPU can be used as...
  • Main memory as DRAM replacement
  • Accelerator like GPUs
  • Compute-enable main memory
Main Memory Design for DRAM Replacement

Nishil Talati\textsuperscript{1}, Heonjae Ha\textsuperscript{2}, Albert Ciprut\textsuperscript{3}, Zhiying Wang\textsuperscript{4},
Eby Friedman\textsuperscript{3}, and Shahar Kvatsinsky\textsuperscript{1}

\textsuperscript{1}Technion, \textsuperscript{2}Stanford, \textsuperscript{3}Rochester, \textsuperscript{4}UCI
Role of mMPU in Designing Future Architectures

• mMPU can be used as...
  • Main memory as DRAM replacement
  • Accelerator like GPUs
  • Compute-enable main memory
Open Question – Future Memory System Design

• Data distribution in a hybrid memory system

mMPU – not Just Memory

Rotem Ben-Hur, Nimrod Wald, Nishil Talati, Ameer Haj Ali, John Ruben, Shahar Kvatinksy, etc.

Technion
Role of mMPU in Designing Future Architectures

• mMPU can be used as...
  • Main memory as DRAM replacement
  • Accelerator like GPUs
  • Compute-enable main memory
Sources of Inefficiencies in von Neumann Machines

Performance Gap

CPU

Memory

I-Cache Access

Register File Access

Control

Add

Energy

Memory Wall Problem

Time

Performance

(2X: 2 years)

(2X: 10 years)
Recent Approaches to Reduce Data Movement

Prior Art

90’s

- Configuration PIM machine
- Active Pages
- SA connected to SIMD pipeline

Recent

- Micron Automata Memory Processor

Architecture of Previous Solutions

- CPU
- Memory (DRAM)
- CMOS Processing Units (PUs)

Data transfer is still required to/from DRAM and PUs

mMPU: memristive Memory Processing Unit
an Ultimate Solution to von Neumann Bottleneck

Moving from conventional DRAM to memristive memory

mMPU: performing computation *USING* the memristive memory cells
Memristor – Memory Resistor
Resistor with Varying Resistance

Current
Voltage

Decrease resistance

Current

ASIC$^2$
**Doing Magical Tricks using Memristors**

Memristor

\[ R_{ON} = \text{Logic '1'} \]
\[ R_{OFF} = \text{Logic '0'} \]

<table>
<thead>
<tr>
<th>IN₁</th>
<th>IN₂</th>
<th>NOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

It turns out...

**MAGIC – Memristor Aided logic!!**


R_{OFF} >> R_{ON}

Initialize OUT to R_{ON}

Increase resistance
How to Perform Computation using Memory Cells?

Logic Hierarchy in mMPU

Sequence of MAGIC NOR operations

Matrix multiplication
Convolution
MUL
POW
SQRT
DIV
ADD
NOR
AND
SUB
NOT
XOR
OR
COPY
NAND

MAGIC - NOR

Complete logic family
Logic Execution within mMPU

\[ f: (A, B) = C \]

Control Pattern
Parallel Vector Operation within mMPU

\[ f^n : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n \]

Latency of the vector operation is independent of the length of the vector

mMPU Controller Design in Detail
Arithmetic and Logical Operations in mMPU
Logic Synthesis and Optimization within mMPU

- Customized standard cell library (.genlib)
- Logic function
- Performance Optimization
- Netlist (.v)
- Synthesis Tool
- In-memory computation constraints
- Execution sequence
- Real-time Address Mapping
- Addresses constraints
Open Question – How to Perform Real-Time Address Mapping?
Processing Data with mMPU

Operands already present within the memory
Processing Data with mMPU

Step 1: compute \((A+B)'\)
Step 2: compute \((B+C)'\)
Step 3: compute \((A+C)'\)

Intermediate outputs stored in the memory cells itself!
Need Additional Cells for Processing...
Processing Area

Control Pattern

A  B  Functional  C

Additional computing memory space to store intermediate results
Processing Area

A

B

Data

C

Desired destination location

Functional

ASIC²
Processing Area
Static

Control Pattern

A  B  Functional  C

A  B
Under utilization of memory
Dynamic processing area

Processing Area

Dynamic
Processing Area
Dynamic
Processing Area
Dynamic

Costly Memory Management

Dynamic Processing Area
Open Question – How to optimize the allocation of processing area?

Static

OR

Dynamic
Image Convolution Acceleration Design using mMPU
Image Convolution Acceleration Design using mMPU
Open Question – Programming Model for PIM-instructions

Programmer-visible PIM Execution

PIM_malloc(...);
PIM_add(#op1, #op2, #dst);

Programmer-visible PIM Execution

Compiler?

CPU
mul
PIM
img_conv
mul(#op1, #op2, #dst)
img_conv(#op1, #op2, #dst)
Open Question – If Programmer-invisible PIM Execution Then…

- Type of instruction
- Type & length of operands

Instruction Allocator

- CPU
- GPU
- mMPU

Programmer-visible PIM Execution

- img_conv()
- mul(#op1, #op2, #dst)

Compiler

- CPU
- PIM
- Compiler?
Open Question –
Data Locality in Accelerator Mode
Open Question – Data Integrity While Processing-In-Memory

ECC Operation in Memory Systems
STORE: encode(data) -> data+par(data)
LOAD: decode(data+par(data)) -> data
Open Question – Data Coherency in Caches while PIM

ASIC²
Open Question – Redesign of Memory Controller to Enable PIM
Conclusions

• Problems faced by modern computers open opportunities for today’s computer architects

• mMPU – memory is not just memory

• With basic ideas, there’s still much to develop
Those who are interested...

Nishil Talati (nishil.talatiwork@gmail.com)
TCE 408