128-Bit, Exascale Memory Reference Models

Steve Wallach
swallach@micron.com
Acknowledgement

• Prescott Winters – Oracle/Chertoff Group. Previously senior exec, DOD

• Dave Clark (MIT) – reviewed presentation, made many suggestions

• Steve Poole – (my doppelgänger) for many suggestions
Presentation

- **Background/Foundational Notions**
  - From the 1970’s to today

- **Proposal (Strawman)**
  - Protection Structures
  - Exascale Programming Model
    - Compute and Big Data
    - Dual Use
  - Referencing Non-Volatile Memory (NVM)

- **References**
What’s Next

• “The end of Moore’s law could be the best thing that has happened in computing since the beginning of Moore’s law. Confronting the end of an epoch should enable a new era of creativity by encouraging computer scientists to invent biologically inspired devices, circuits, and architectures implemented using recently emerging technologies. “

Background

• Since the late 70’s, **mainstream** processors have increased the size of the virtual space by simply adding more bits
  – DEC PDP/11 & VAX: 16 → 32
  – Data General Eclipse/MV: 16 → 32
  – SPARC & HP RISC: 32 → 64
  – Intel x86;  16 → 32 → 64 (48 used)
    • AMD 32 → 64
    • Itanium 64
  – IBM Power: 32 → 64
    • Mainframe: 24 → 32 → 64 (50 years)
  – ARM: 32 → 64
• Memory management and protection are intermingled.
• The need to address more physical memory (Moore’s law)
• Memory still addressed by virtual/physical address paradigm
Background

• Other’s (pioneer’s) did not simply add more bits
  – IBM – FS (Ref: [13]) – 1976
    • Tagged 16 byte pointers (Capabilities)
    • System/38 is the diminutive of FS
  – Data General - FHP (Ref: [3, 4,17,23]) – 1980
  – Intel 432 iMAX OS – (Ref: [16]) – 1980
    • 24 bit passive address
    • 80 bit UID (16 bit checksum)
ARCHITECTURE OBJECTIVES

- Programming generality is the ability to move a program between computer installations; the ability to maintain a program within changing hardware; the ability to use a program in the construction of another - without altering the program description in any way.

Proposal

• Protection Structures
Going Forward

• Time to define a 128 bit space without the need for 128 address arithmetic
  – What is “i” in A[i]?
  – A Virtual Address greater than 64 bits

• Time to correct and incorporate appropriate security and access mechanisms
  – Network Wide Security Model
    • Now each node has its own security model (client/server/network/server)
  – Access to the web is assumed and required
    • Private Cloud
    • DATA is GLOBAL not just LOCAL

History suggests that whenever it becomes clear that more than 64 bits of address space is needed, architects will repeat intensive debates about alternatives to extending the address space, including segmentation, 96-bit address spaces, and software workarounds, until, finally, flat 128-bit address spaces will be adopted as the simplest and best solution.

RISC V ISA SPEC (page 105 – v2.1)
Security & Facts

• Computer Virtual Address’s (VA) span to local disk only
  – Disk Access is now Global (In practice)
  – Remember a VA references DISK explicitly NOT main memory (CS101)
• Network Addressing (IPv4 & IPv6 span the entire network)
  – IPv6 created a 128 bit network address space. Unique names
• MAC, EMAIL and URL’s addresses are unique
• Phone numbers are global; country code, city code, local code
• Two different (web and local) address structures
  – Two different protection and addressing systems
  – Two different authentication systems
• Software needed to bridge these two domains (too much software)
• What if one unified name structure could be developed?
Some Foundational Basis

• One should recognize that concentration on protection and authentication mechanisms provides a narrow view of information security, and that a narrow view is dangerous. The objective of a secure system is to prevent all unauthorized use of information, a negative kind of requirement.

• Every access to every object must be checked for authority. This principle, when systematically applied, is the primary underpinning of the protection system.

• Validity/Authenticity is a REQUIREMENT (Ref D. Clark, personal communications)

Previous Efforts

- Another solution is to address each segment with a unique integer which is assigned at the time the segment is created, never changed, and not reused even after the segment has disappeared from the system. Call this the unique integer solution. ([3,4,5] & [13] Radin’s H – Handle)


OPAL

- Single Address Space for all applications
- Persistent Pointers
- “A full 64-bit address space will last for 500 years if allocated at the rate of one gigabyte per second. We believe that 64 bits is enough "for all time" on a single computer, enough for a long time on a small network, and not enough for very long at all on the global network.”

Strawman – RV128I

- 128 Bits
- Object ID – Unique Identifier
  - a software (or hardware) structure that is considered to be worthy of a distinct name.
- Indexing is 64 bits – A[i]
  - Program Counter
  - Stack Pointer (Loads and Stores)
- ISA independent
  - Like routing IP packets (Vendor Independent)
- Persistent across time and space
- Protection and memory management are independent
Why?

- We need better security
- We need computer virtual addressing to reflect the contemporary uses
  - Network/Cluster Wide
  - Support of Semantic Memory
- We do **NOT** want 128 bit flat addressing
- We need pointer interoperability between computer systems (maybe)
- We need a simplified sharing mechanism
- We need a authentication, revocation and protection again malware/virus’s
What is a Object?

- A Object is a unique 64 bit number.
- An Object can specify
  - Location and protection mechanisms
  - Language Specific/Architecture attributes
    - E.G., PGAS NODE (e.g., UPC, CHAPEL)
    - Data Encrypted/Two Factor Authentication
    - Blockchain
    - ISA
- The creation of a Object is via a central name server.
  - Just like: IPv6, MAC addresses, ICANN
- Central NAME server Involvement
  - Just like a DNS server
  - Just like Apple’s iCloud
  - Only manages Object’s, NOT DATA/APPLICATIONS
- Should an Object be a IPv6 address
  - Use bit 63 of offset to select; IPv6 or Object UID?
Access Control/Domains

- Defines a sphere of protection and use
- Non-Hierarchical
- Permission bits define what is permitted
  - LOCAL (Rd, Wr, Ex)
  - Global Network (Rd, Wr, Ex)
  - Extended privileged
  - Classical Privileged
  - System Calls
    - Explicit
    - Mediated
  - Shadow Stack
- Domain Crossing and Return
  - Protected Stack (HW maintained)
  - Gate Entry
  - Mediated Number of Gates
  - Stack Switching
Authentication

- The address space is unique over time and space. Any computer supporting this address space is addressable by the name server.
- Accessing an object for the first time requires
  - Permission to access (i.e., download A.OUT or .EXE file, entry within an ACL)
  - Access privileges for the object
    - Local and Network read/write/execute
    - Access only thru a protected sub-object
    - Execution Domain
- Domain of execution
  - Level of user (e.g., gold, platinum, executive platinum)
  - Admin (Level 1, 2, or N)
- In essence we have a global access control list
  - We have that today, but don’t realize it
  - It is distributed (e.g., ADOBE maintains its 2D slice of the matrix)
  - Each Vendor has their own access control list
Example using An Application

- Can Execute in My Domain
  - Can read and write my file system
- Can execute in different domain
  - Determine level of trust
  - Can read my data, but not write
  - Can’t send data back to ADOBE (network permissions)
Memory Management

• Each object can have its OWN memory management structure
  – Page Tables
  – Hashed Indices
  – PGAS like

• There is **NO ACCESS** bits associated with the management of storage (e.g., read, write, execute, etc..)
  – Management is **separate from protection**

• Each object can choose to have object size for constraint access checking (bounds check).
64 BIT LINUX
Machine State Model

Proc_ID (pid_t)

- Process Specific (task_struct)
  - Unique for each process
  - Hashed Proc_ID

File Object

Process Address
Space Object

Process Communications

System Wide Resources

Kernel Maintained
  .Page frame Cache
  .Disk Cache
  .Directory Cache
  .Lower level of Network Stack
64 BIT LINUX

-Memory Management

Proc_ID
32 BITS

VIRTUAL ADDRESS (VA)
64 BITS

BASE

Process Object
Address Space

Hash

Page Table Base

F(VA)

Page Base

Page Table per Proc_ID

LINUX NAME SPACE

CAT

Page Offset

Physical
Byte Address

TLB ASSOCIATES ON A 96 BIT NAME SPACE
implementation dependent

2017 _april_salishan_workshop
128-BIT VIRTUAL ADDRESSING

-Memory Management

TLB ASSOCIATES ON A 128 BIT NAME SPACE
.implementation dependent
128- BIT VIRTUAL ADDRESSING
- Protection/Data Reference

OBJECT ID

64

BYTE OFFSET

64

Current Domain

Principal

FUNCTION

Access Control Lists

Protection Attributes
- Read/Write/Execute
- System Calls
- External References
- Protected Sub-Objects

Cache Last # of Entries
Validate Reference before Data Reference
Protection - ACL (matrix) – uses OBJECT names -maintained by Name Server-

Note: Domain and Process Are NOT unique
PROTECTION -ACL Entry

• PSO – Protected Sub-Object (Sandbox – [11])
  – Virtual Machine
  – Requires software interpretation
    • Address of Sandbox
  – Mediated access
    • Files embedded in email
  – Part of Object creation
    • The meta data of the object
  – ... the concept of confining a helper application to a restricted environment, within which it has free reign
REFERENCE MONITOR

• It must validate enforcement of the security policy for every reference to information.
• Second, it must be tamper-proof, that is, it cannot be subverted.
• Lastly, it must be verifiable, so we have high assurance it always works correctly.

Exascale Issues

• Subject to cost and power critical apps desire
  – One byte/sec per peak flop (DOE ASC)
  – Really want 4 bytes/flop – no data cache
    • DREAM ON
  – Low latency – HPCG should be the driver
  – Physical Memory - As much as you can
    • $2^{64}$ bit words desirable (global access)

• Memcached Configurations [7]
  – Fronting LARGE Disk+NVM Farms

• Big Data and Compute.
  – Same system will be used for BOTH

• BTW: An ExaByte requires 60 bits of address
The machine that is simplest to program WINS. User cycles are more important than cpu cycles.
Stacked Memory and PIM’s

• One cube, will have:
  – Non-volatile memory
  – Dram
  – External Interfaces
  – Processing

• Impact on programming [26]
  – NUMA reference model

• What type of stacking/tsv’s?
  – Industry or Manufacturer Standard
  – Should every slice be the same form factor (no interposers)
  – DARPA CHIPLET program [22]

Programming a Cube

• The code
  – DO i = 1, N
  – A(i) = B(i) + C(i)
  – ENDDO

  for (int i=0; i<n; i++) {
    a[i] = b[i] + c[i];
  }

/* Executed in memory system */
Load 8, VS;        stride =8
Load N, VL;        setup vector length
Load <A>, GR1;     address of A
Load <B>, GR2;     address of B
Load <C>, GR3;     address of C

/* Executed in Accelerator */
AGAIN:  Load B, VR1
        Load C, VR2
        ADDV, VR1, VR2, VR3
        Store V3, <A>

/* Executed in memory system */
Sub #128, VL
Comp VL < 0
Branch Exit

Add 128x8, <B>
Add 128x8, <A>
Add 128x8, <C>
JMP AGAIN
Programming a CUBE

• We need new memory reference models.
• Same reference model for the last 50 years
• Data Science references objects

• So;
  – DO I = 1, N
    • A(i) = B(I) + C(i)
  – ENDDO
• Is compiled as
  – LD ref:C
  – LD ref:B
  – ADD, B,C
  – ST ref: A
• References are thru descriptors
  – Memory system supplies data
  – Implicit hardware strip mining
• Benefits
  – Low latency memory references
  – Lower power (address data stays in memory)
Non Volatile Memory (NVM)

- CS101 – virtual memory
- A virtual address references archival storage
  - In this case, NVM
- Main memory is a cache/temp for archival/non-volatile storage.
- How to manage integrated dram/non-volatile memory
  - NVM is 4 to 10 x slower than dram.
- Can the virtual to physical translation need to be changed?
Architecture Tradeoffs

- **Block Size**
  - Cache Block or Disk Block or in between

- **Block Replacement**
  - Hardware and/or Software aided
  - Need to have wired down blocks in main memory

- **Write Thru or Write Back**

- **Ratio of NVM to DRAM**

- ** Concurrent write into dram and processor cache** (upon dram miss)

- **New interface** — supports DRAM row (16 Kbits) read and write.
  - Currently with DDR-X

- **Page Table Entry** — control bits

- **INTEL Cache Allocation Technology [21]**
  - Software control of where data is allocated in the last level cache
  - Another level of domain decomposition

- **Probably different configs for FLOPS/DATA SCIENCE/DATA BASE**
  - Compiler Directives
  - Application Specific

---

The DC P4800X is a server drive riding the NVMe interface, initially available in a 375-GByte version that lists for a retail price of $1,520, about three times the cost of a similar NAND card. It delivers less than 20-microsecond read-and-write latencies and a 30-disk-writes/day endurance with an estimated three-year lifetime.

Page Table Entry - NVM

• Classical –

• PTE/NVM
  – Dram or NVM (present or not)
    • Physical address in both memories.
  – Cache or no Cache
  – Page replacement in DRAM (hardware)
  – Interrupt OS
Summary

• Deal with Virtual Address (and physical address) ranges from 2020 and beyond
• Incorporate Contemporary Protection Mechanisms that function in:
  – Web and Cloud based configurations
  – The objective of a secure system is to prevent all unauthorized use of information, a negative kind of requirement.
• Memory should have semantics not just a virtual/physical address
What if processors like this win?
(A number of vendors like this model)

http://www.wired.com/2014/08/datacenter-of-the-future/

http://www.wired.com/2013/05/google-jason-mars/

Courtesy of Steve Poole
Summary

• Compare to CHERI [12] Capability System


• Should Internet addressing be by a UID and NOT IP address. – Facilitate protection and authentication (Ref: [15])

• Named Data Network Proposal

• What next? – Proposal

• Build a prototype with FPGA’s / Simulate
References

• http://csg.csail.mit.edu/CSGArchives/memos/Memo-32.pdf
References

- [17] [http://people.cs.clemson.edu/~mark/fhp.html](http://people.cs.clemson.edu/~mark/fhp.html)
- [23] [http://people.cs.clemson.edu/~mark/fhp.html](http://people.cs.clemson.edu/~mark/fhp.html)
BACKUP SLIDES

• The following slides discuss
  – Revocations
  – Shadow Stack
  – ETC
Revocation

• Every client has a “KILL SWITCH”
• The central name server is accessed and each object has a kill capability (only initiated by the owner)
• What if the unified name server is NEVER accessed again??
  – Watch Dog Timer?
ACL (Access Control List) - ENTRY

PERMISSIONS (Local and Global)

OBJECT LENGTH - BYTES

PROTECED SUB-OBJECT (PSO) POINTER

64

Principal UID is appended to the PSO pointer, essential which virtual machine to use
Sandbox Particulars

• Sys Calls Mediated to Defined Domain Server
  — (Framework of [11])
• Dispatch Table Within Domain Server traversed to
to decide allow or deny the call.
  — If denied – Kill
• Trusted Apps can bypass the framework
  — Part of ACL entry
• If no sys calls and only reads/writes to permitted
data
  — Timer resolution or other means
Some Object UID Math

- 60 seconds in a minute
- 60 minutes in a hour
- 24 hour in a day
- 365 days in a year
- Yields 31,536 000 seconds in a year
- 30 years of life ➔ 946,080,000 or $2^{30}$
- $2^{34}$ clients (16 billion)
Protection in Unified Name Space

• ACL – access control matrix
• Protection Domains
• Revocation
Shadow Stack

• Independent, somewhat of address proposal
• Solves classical virus/malware attacks
  – Heap Overflow
    • Change return address
  – Writing over Stack
  – Heap Fung Hui Attacks [8]
• Hardware protected SP, AP, FP
  – Akin to domain crossing
  – Return via shadow stack
Shadow Stack - Architecture

Shadow Stack (Hardware Maintained)
Different Domain, Same virtual address
Return via THIS STACK

User Visible Stack
Can not write/read Shadow Stack

Return Block

Stack Pointer

Return Block

Stack Pointer

FP

AP

FP

AP