IBM POWER9 Processor, NVIDIA V100 GPU and IBM AC922 Node Hardware Architecture

SC18 Tutorial

12.11.2018 | Dirk Pleiter, Forschungszentrum Jülich/JSC & University of Regensburg |

JÜLICH
Forschungszentrum

Goals for Today

- Obtain knowledge about the AC922 heterogeneous server platform including it's
  - IBM POWER9 processor
  - NVIDIA V100 processor
- Obtain skills to analyse application for attainable performance
- Learn and practice measuring performance
- Learn and exercise how to optimise a simple application on IBM POWER9 processors and NVIDIA V100 GPUs
- Learn and practice parallelisation on multiple GPUs
- Study best practices for porting scientific applications

Overview of the Tutorial

Lecture	IBM POWER9 processor, NVIDIA V100 GPU and IBM AC922 node hardware architecture
Lecture + hands-on	Performance counters and tools
Lecture + hands-on	Optimization on POWER9
Lecture + hands-on	V100 GPU programming and optimization Multi-GPU programming
Lecture	Best practices for porting scientific applications


C. Hagleitner


A. Herten


T. Papatheodore


D. Pleiter


A. Ravindar


M. Wagner

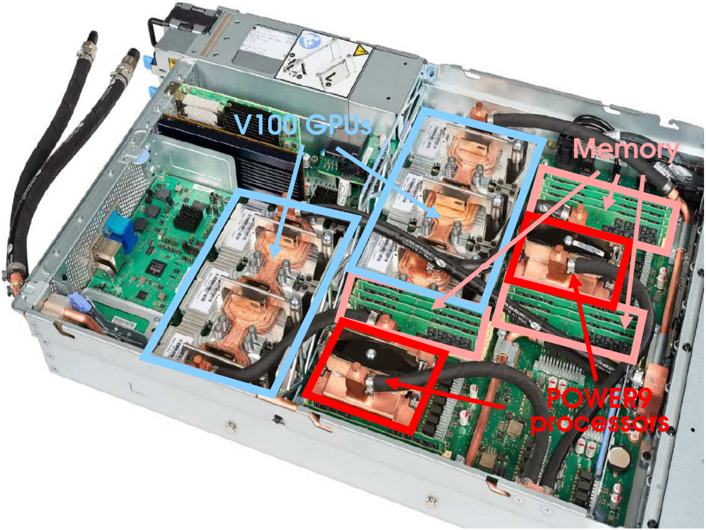
Overview of this Lecture

- Introduction
- IBM POWER9 Processor
- NVIDIA V100 GPU
- IBM AC922 Platform
- Use Case
- Summary

Part I: Introduction

AC922 Server: Overview

[IBM Redbook, 2018]



JÜLICH Forschungszentrum

OpenPOWER for HPC

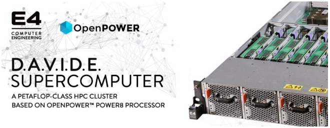
JURON @ JSC
18 Minsky nodes
0.35 PFlop/s

⇓

D.A.V.I.D.E @ CINECA
45 Minsky nodes
0.9 PFlop/s

⇓

SUMMIT @ ORNL
4608 AC922 nodes
188 PFlop/s



Information Exchange Function

- A computation implies that information is transferred from a storage device x to a storage device y.
- **Information Exchange Function**:

$$I^k_{x,y}(W) \quad = \quad \text{data transferred between computer sub-systems for specific computation } k$$

x ... source storage device (e.g. memory)
y ... destination storage device (e.g. register file)
W ... **problem size/work-load**

Bandwidth-Latency Performance Model

Ansatz to predict latency:

$$\Delta t^k_{x,y} \simeq \lambda_{x,y} + I^k_{x,y}/\beta_{x,y}$$

Examples:

- Throughput of arithmetic operations
  - $x, y = \mathrm{R}$
  - $\beta_{\mathrm{R,R}}$ = throughput arithmetic unit
  - $I^k_{\mathrm{R,R}}$ = number of operations

- Load of data
  - $x = \mathrm{M}$, $y = \mathrm{R}$
  - $\beta_{\mathrm{M,R}}$ = memory bandwidth
  - $I^k_{\mathrm{M,R}}$ = amount of data

Balanced Architecture

- Assume perfect overlap of computation and data transport
- Condition for balanced architecture

$$\Delta t_{\mathrm{fp}} \simeq \Delta t_{\mathrm{mem}}$$
$$\Downarrow$$
$$\frac{I_{\mathrm{fp}}}{\beta_{\mathrm{fp}}} \simeq \frac{I_{\mathrm{mem}}}{\beta_{\mathrm{mem}}}$$
$$\Downarrow$$
$$\frac{\beta_{\mathrm{fp}}}{\beta_{\mathrm{mem}}} \simeq \frac{I_{\mathrm{fp}}}{I_{\mathrm{mem}}} = \mathrm{AI}$$

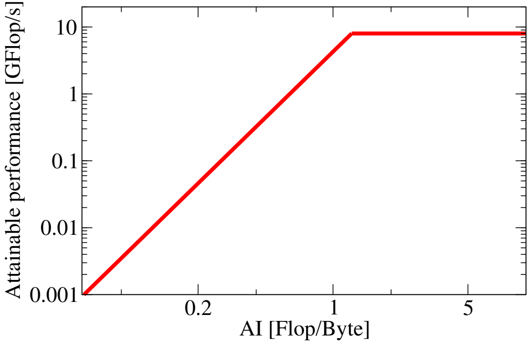
Arithmetic Intensity

- Unbalanced cases
  - $\mathrm{AI} < \beta_{\mathrm{fp}}/\beta_{\mathrm{mem}} \Rightarrow$ **memory bandwidth limited**
  - $\mathrm{AI} > \beta_{\mathrm{fp}}/\beta_{\mathrm{mem}} \Rightarrow$ **compute performance limited**

Roofline Model

Attainable performance

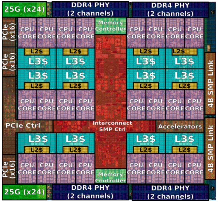
$$b_{\rm fp} < \frac{I_{\rm fp}}{\max(\Delta t_{\rm fp}, \Delta t_{\rm mem})} \simeq \min(\beta_{\rm fp}, \mathrm{AI} \cdot \beta_{\rm mem})$$

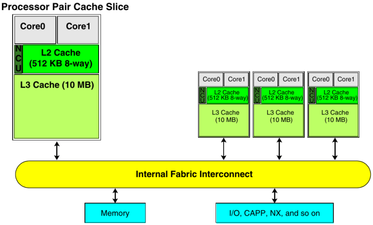


Part II: IBM POWER9 Processor

POWER9 Overview

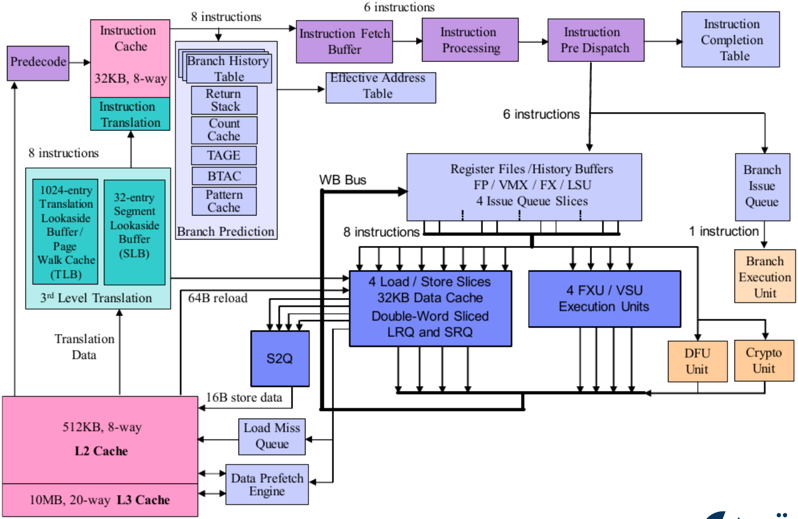
- Introduced in 2017 using 14nm technology
- POWER v3.0B ISA
- Up to 24 cores, frequency up to 4 GHz
  - 4 threads per core in SMT4 configuration
- Core performance figures
  - Double precision floating-point arithmetics:
    $2 \cdot 2$ FMA/cycle = 8 Flop/cycle
  - Load from L1:
    $2 \cdot 16$ Byte/cycle
  - Store to L1:
    $1 \cdot 16$ Byte/cycle





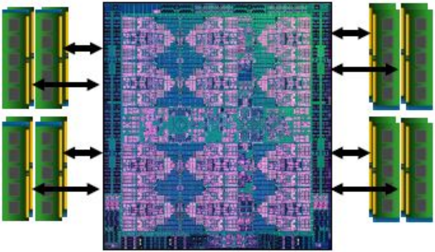
POWER9 Micro-Architecture

[IBM, 2018]



POWER9 Memory Subsystem

[IBM, 2018]



Directly attached memory:

- 8 DDR4 channels
- Up to 2.667 GT/s or up to 170 GByte/s/socket

POWER9 Memory Hierarchy

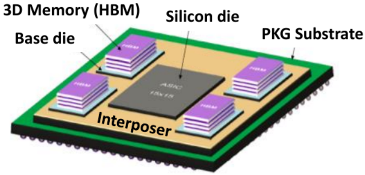
- Cache line size = 128 Byte
- L1 data cache
  - 64 kByte, 8-way set associative
  - Write update policy: store-through
  - Core private
- L2 data cache
  - 512 kByte, 8-way set associative, dual-banked
  - Fully inclusive of the L1 cache
  - Shared by 2 cores
- L3 data cache
  - 10 MByte region per 2-core slice, 20-way set associative
  - Victim for local L2 or other L3 caches, not inclusive of the L2 cache
  - Accessible from other cores

Part III: NVIDIA V100 GPU

Volta Architecture Overview

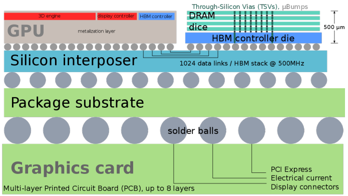
- Introduced in 2017 using 12 nm
- Up to 80 Streaming Multiprocessors (SM)
- SM performance figures
  - 32 FP64 CUDA “cores”
  - 64 Flop/cycle

V100 Memory Subsystem



[H. Jun (SK Hynix), 2013]

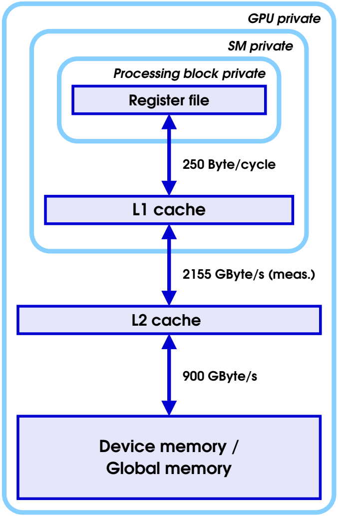
- In-package HBM2 memory
- 4 memory stacks,
  4-8 GByte/stack
- Very wide bus ($2 \cdot 512$ bit) and relative low data rate (1.75 GT/s)
- Aggregate bandwidth:
  900 GByte/s



[Wikipedia]

V100 Memory Hierarchy

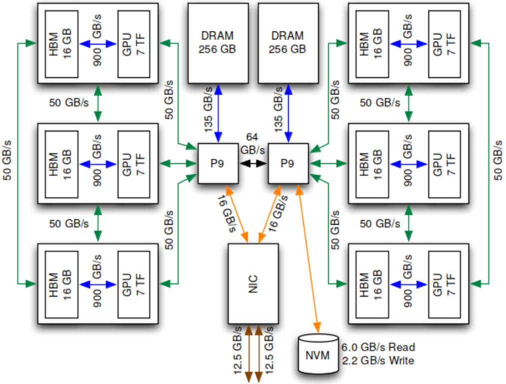
- Register file
  - Size: 256 kiByte/SM or 64 kByte/processing block
- L1 data cache and shared memory
  - Size 128 kiByte/SM or 10 MiByte/GPU
  - L1 hit latency 28 cycle (measured)
- L2 cache
  - 6 MiByte/GPU
  - L2 hit latency 193 cycle (measured)



[Zhe Jia et al., 2018]

Part IV: AC922 Nodes

AC922 Server: Details



TF	42 TF (6x7 TF)
HBM	96 GB (6x16 GB)
DRAM	512 GB (2x16x16 GB)
NET	25 GB/s (2x12.5 GB/s)
MMsg/s	83

- HBM/DRAM Bus (aggregate B/W)
- NVLINK
- X-Bus (SMP)
- PCIe Gen4
- EDR IB

HBM & DRAM speeds are aggregate (Read+Write).
All other speeds (X-Bus, NVLink, PCIe, IB) are bi-directional.

JÜLICH Forschungszentrum

Comparison POWER9 vs. V100 on AC922

	POWER9	V100
Number of cores/SMs	≥ 16	80
Double-precision Flop/cycle	≥ 128	5,120
Clock frequency [GHz]	3.8	1.312
Double-precision TFlop/s	≥ 0.5	6.7
Memory bandwidth read+write [GByte/s]	135	900
Balance [Flop/Byte]	≥ 3.7	7.5
Number of CPU or GPU	2	6
Aggregate double-precision TFlop/s	0.97	40.3
Aggregate memory bandwidth [TByte/s]	0.27	5.4
Aggregate memory capacity [GByte]	512	96

Part V: Use Case

The Poisson Equation

- Poisson equation in 2 dimensions:

$$-\frac{\partial^2 v(x,y)}{\partial x^2} - \frac{\partial^2 v(x,y)}{\partial y^2} = f(x,y)$$

- Discretisation of 2nd-order derivative

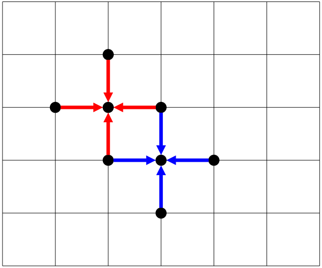
$$-\frac{\partial^2 v(x,y)}{\partial x^2} \leftarrow \frac{2v_{i,j} - v_{i-1,j} - v_{i+1,j}}{h^2}$$

- Discrete Poisson equaiton in 2 dimensions:

$$T\,v = h^2\,f \quad \text{where} \quad T = \begin{pmatrix} 4 & -1 & 0 & \cdots \\ -1 & 4 & -1 & \cdots \\ \vdots & \vdots & \ddots & \end{pmatrix}$$

Discrete Poisson Equation: Data Locality

Graphical representation of $T\,v$:



Observations:

- Matrix T acts as a stencil operator
- Any element of vector v is reused 4 times

Jacobi Algorithm

- Algorithm suitable for diagonally dominant systems
- Matrix decomposition: $T = D + R$

$$D = \begin{pmatrix} t_{0,0} & 0 & 0 & \cdots \\ 0 & t_{1,1} & 0 & \cdots \\ \vdots & \vdots & \ddots & \end{pmatrix}, \quad R = \begin{pmatrix} 0 & t_{0,1} & t_{0,2} & \cdots \\ t_{1,0} & 0 & t_{1,2} & \cdots \\ \vdots & \vdots & \ddots & \end{pmatrix}$$

- Obtain solution through iterative precedure

$$v^{(k+1)} = D^{-1}\left(h^2 f - R\, v^{(k)}\right)$$

Jacobi Algorithm Implementation

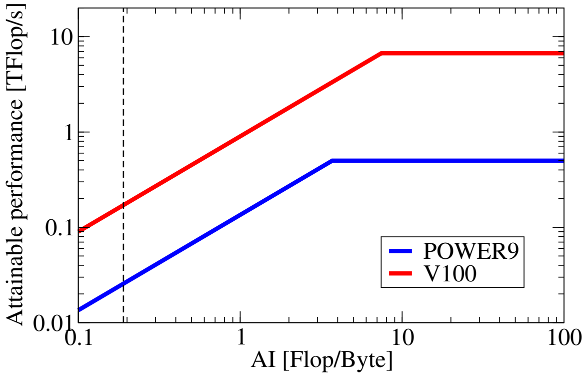
```
for( int iy = 1; iy < ny-1; iy++)
{
  for( int ix = 1; ix < nx-1; ix++ )
  {
    Anew[iy*nx+ix] = -0.25 * (rhs[iy*nx+ix]
      - (   Aref[iy    *nx + ix+1] + Aref[iy    *nx + ix-1] +
            Aref[(iy-1)*nx + ix  ] + Aref[(iy+1)*nx + ix  ]  ));
    error = fmaxr(error, fabsr(Anew[iy*nx+ix]-Aref[iy*nx+ix]));
  }
}
```

Information exchange analysis assuming small cache:

- Load of `Anew`, `rhs`, `Aref`: $I_{\mathrm{ld}} = n_x \cdot n_y \cdot 3 \cdot 8\,\mathrm{Byte}$
- Store `Anew`: $I_{\mathrm{st}} = n_x \cdot n_y \cdot 1 \cdot 8\,\mathrm{Byte}$
- Floating-point arithmetics: $I_{\mathrm{fp}} = n_x \cdot n_y \cdot 6\,\mathrm{Flop}$

Jacobi on POWER9/V100: Roofline Analysis

$$\mathrm{AI} = I_{\mathrm{fp}}/(I_{\mathrm{ld}} + I_{\mathrm{st}}) = 0.19\ \mathrm{Flop/Byte}$$



JÜLICH Forschungszentrum

Part VI: Summary

Summary

- Approach for systematic performance analysis and modelling based on Information Exchange introduced
- Discussed the AC922 node architecture as a building block for supercomputers based on
  - POWER9 processors
  - V100 GPUs
- Introduced use case for today: Solving 2-dimensional Poisson equation using Jacobi solver
  - Memory bandwidth limited application
  - Roofline model defines upper performance limit