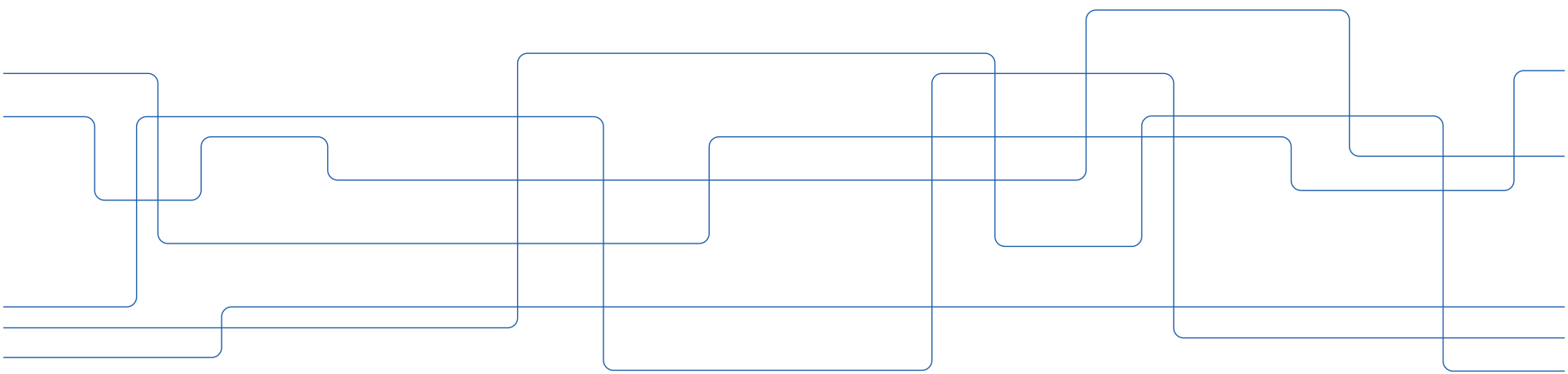
Arm-based Processor Architectures

Dirk Pleiter

PDC and CST | EECS | KTH

January 2023



Overview

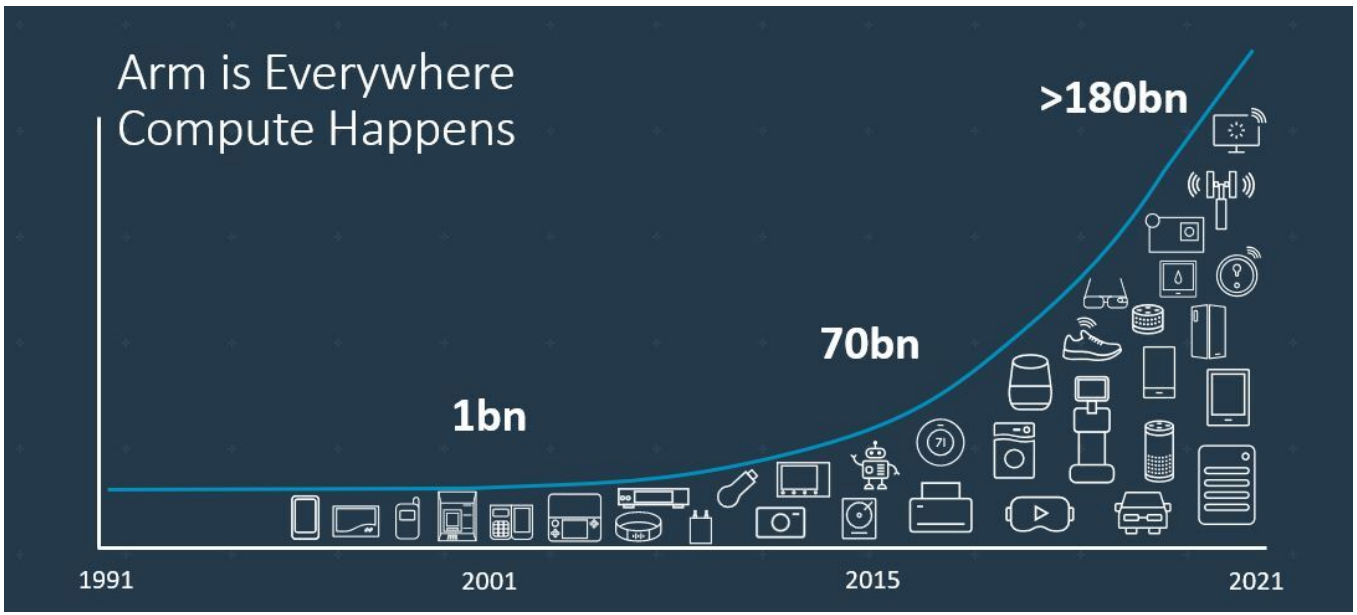
Content

Introduction

- Arm = Semiconductor and software design company
  - Founded in 1990
  - Headquarter in Cambridge (UK)
  - Daughter company of the Softbank Group
- Arm = Architecture
  - Reduced instruction set computer (RISC) instruction set architectures (RISC)
  - The ARMv1 architecture was introduced in 1985
  - The Armv9 architecture has been launched in 2021
- Arm = Designs
  - Example: The Arm Neoverse V1 Core
- Arm $\neq$ Processor solution
  - Arm-based processors are build by Arm licensees

Arm Market Figures (February 2021)

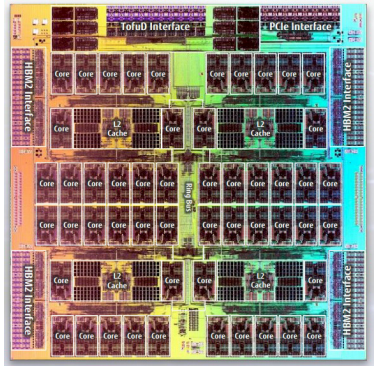
- 6.7 billion Arm-based chips shipped in Q4'2020
- In total $>$ 180 Arm-based chips shipped until end of 2020
- 530 licensees as of early 2021

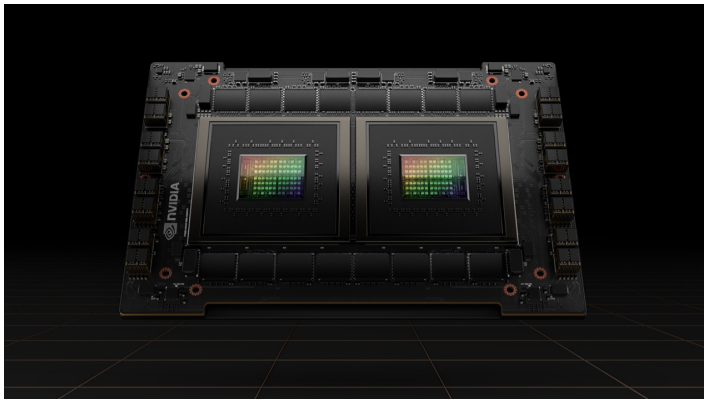


[Arm, 2021]

Arm Server-class CPUs

	launch	N_{cores}
Marvell ThunderX2	2018	32
HiSilicon Kunpeng 920	2019	64
Fujitsu A64FX	2019	48
AWS Graviton 3	2021	64
Ampere Altra Max	2021	128
NVIDIA GRACE	2023	64
SiPearl Rhea1	?	?





Arm Software Ecosystem

Area	Sub area	Status
Applications	Scientific	Many ported
	ISV	Very few available
Development environment	compilers	Several choices
	Numerical libraries	Many ported and optimised
System middleware	OS	Various Linux distributions
	Parallel FS	Very limited choice

Why Arm?

- The Arm business model relies on multiple competing solution providers
  - One single IP provider versus multiple solution providers using this IP
  - The solution providers are forced to look for differentiating features
- Promote a competition for better prices
  - This allows to rebalance different cost components in case of supercomputers
- Promote a competition for better solutions
  - Better solutions for different sub-markets
  - Possible differentiators:
    - Different core count
    - Different width of vector instructions
    - Different memory hierarchy

Arm Myth: Power Efficiency

Green500 list results (in parts estimated):

CPU	Efficiency	Comment
A64FX	$17\,\mathrm{GFlop/s/W}$	Special design, FP opt.
Xeon Cascade Lake	$6\,\mathrm{GFlop/s/W}$	FP optimised
ThunderX2	$\sim 4\,\mathrm{GFlop/s/W}$	
Kunpeng 920	$\sim 3\,\mathrm{GFlop/s/W}$	Not FP optimised

Content

Armv8 Architecture Introduction

- RISC ISA with
  - An uniform register file
  - Load/store architecture
    - Data-processing operations only operate on register contents
    - Load to register, store from register
- 64-bit (AArch64) and 32-bit (AArch32) execution state
- SIMD instructions as part of the base ISA (128-bit in case of AArch64)
- Multiple architecture profiles
  - A: Application profile with support for virtual memory
  - R: Real-time profile
  - M: Micro-controller profile

Arm Architecture Extensions

Extensions of Armv8 (mostly optional):

- **Scalable Vector Extension**: Instructions supporting very wide SIMD instructions
- **Cryptographic Extension**: Instructions for the acceleration of encryption and decryption
- **RAS Extension**: Extensions to improve Reliability, Availability, and Serviceability mainly by adding control and status registers
- **Statistical Profiling Extension (SPE)**: Support for non-invasive performance profiling

Digression: SIMD Parallelism

- Single Instruction Multiple Data (SIMD) instructions exploit data-level parallelism by operating on data items in parallel
  - E.g., SIMD add

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} \leftarrow \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

- Example ISA:
  - Intel Streaming SIMD Extensions (SSE)
  - Intel Advanced Vector Extensions (AVX, AVX2, AVX512)
  - POWER ISA (VMX/AltiVec, VSX)
  - Armv7 NEON, Armv8, Arm SVE

Arm SVE

- SVE = Scalable Vector Extension
  - Extension of Arm-v8 that includes NEON SIMD instructions
- Unique feature of SVE: Vector Length Agnostic (VLA)
  - Vector length is a choice of the CPU architect
  - Supported length: $128, 256, \ldots, 2048\,\mathrm{bit}$ (multiple of 128)
  - Same binary executable on CPUs with different SVE length

Note: VLA needs suitable ISA definition + hardware support

Arm SVE Example: DAXPY

```
void daxpy(double a, double *restrict x,
           double *restrict y, int n)
{
  int i;

  for (i = 0; i < n; i++)
    y[i] += a * x[i];
}
```

gcc-10.2 -march=armv8-a+sve -O3

```
cmp w2, 0
ble .L1
mov x3, 0
mov z2.d, d0
whilelo p0.d, wzr, w2
ptrue   p1.b, all
.p2align 3,,7
.L3:
ld1dz1.d, p0/z, [x1, x3, lsl 3]
ld1dz0.d, p0/z, [x0, x3, lsl 3]
fmadz0.d, p1/m, z2.d, z1.d
st1dz0.d, p0, [x1, x3, lsl 3]
incd x3
whilelo p0.d, w3, w2
b.any   .L3
.L1:
```

Contiguous load double words to vector

Floating-point fused multiply-add

Contiguous store doublewords

Increment loop counter

While incrementing unsigned scalar lower than scalar

Arm Memory Model

- Arm uses a relaxed memory model that makes less consistency guarantees compared to x86
  - Propagation of writes between different threads may happen in different order
- Example
  - Assume the following dual-threaded application

```
// Thread 1        // Thread 2
a = 1              r1 = b
b = 1              r2 = a
```

  - Question: Can `r1 = 1` and `r2 = 0` be observed?
  - Answer:
    - x86: no
    - Arm: yes

Content

Kunpeng 920: Introduction

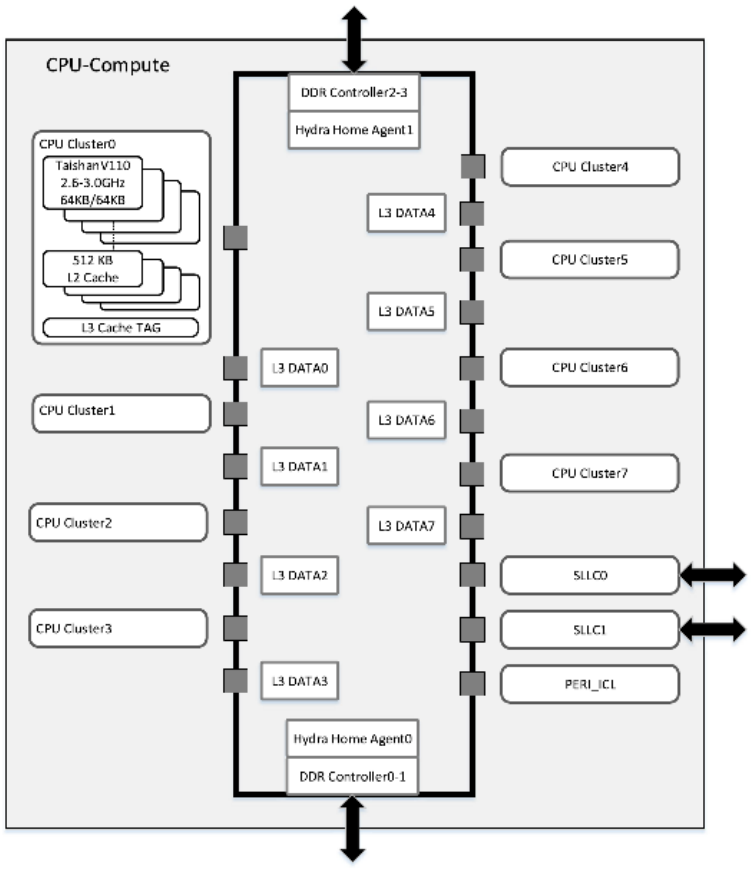
- Up to 64 CPU cores
- Maximum clock frequency of 3 GHz
- LEGO-style architecture based on multiple chiplets
  - The Kunpeng 920 Server processor version comprises
    - 2× CPU-Compute chiplets
    - 1× Compute-IO chiplet
- Process: TSMC 7 nm + 16 nm
- Thermal design power (TDP): 200 W

Kunpeng 920: CPU-Compute Chiplet

- Taishan V110 Core

Instr. decoder	4 instr/cycle
FP64 throughput	2 FMA/cycle
FP32 throughput	4 FMA/cycle
L1-D	64 kiByte
L1-I	64 kiByte
L2	512 kiByte

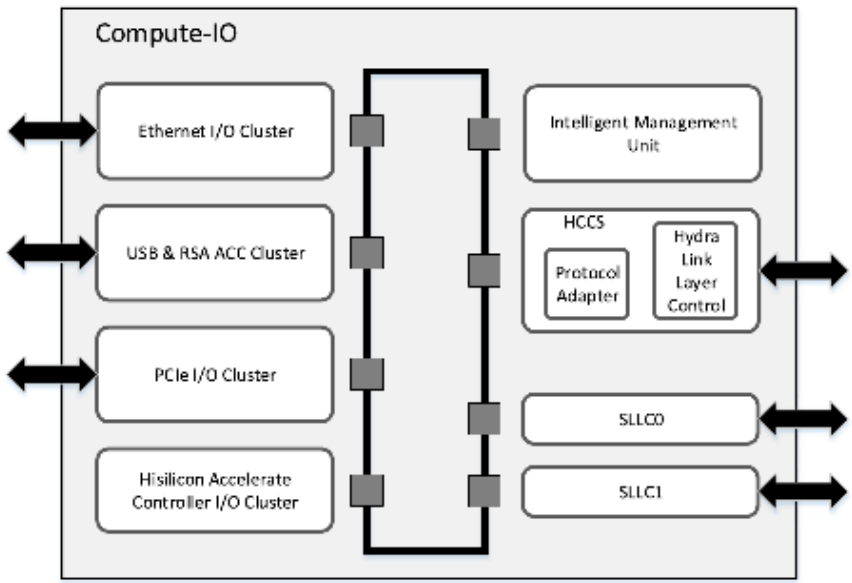
- CPU Cluster
  - 4× cores
- Chiplet
  - 8× CPU clusters
  - 8 × 4 MiByte L3 cache
  - 2× DDR4 memory controller (up to 2993 MT/s)



[Jing Xia et al., 2021]

Kunpeng 920: Compute-IO Chiplet

- Up to 40 lanes PCIe 4.0
- Ethernet
- SAS/SATA 3.0 interfaces



[Jing Xia et al., 2021]

Kunpeng 920: Memory Bandwidth

[Jing Xia et al., 2021]

	Bandwidth	Latency
L1-D	$2 \times 128\,\text{bit/cycle/core} \Rightarrow$ up to $6\,\text{TByte/s/CPU}$	$< 2\,\text{ns}$
L2	$2 \times 128\,\text{bit/cycle/core} \Rightarrow$ up to $6\,\text{TByte/s/CPU}$	$< 4\,\text{ns}$
L3	Up to $3\,\text{TByte/s/CPU}$	$< 15\,\text{ns}$
Memory	Up to $191\,\text{GByte/s/CPU}$	$\sim 100\,\text{ns}$

Finish with a View on Fugaku