SC23
Denver, CO | i am hpc.

An Empirical Comparison of the RISC-V and AArch64 Instruction Sets

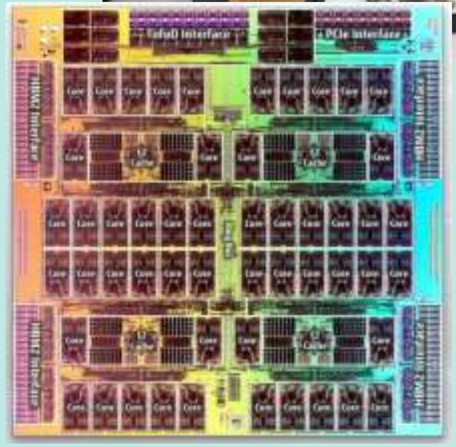
Daniel Weaver & Simon McIntosh-Smith

University of Bristol

University of BRISTOL

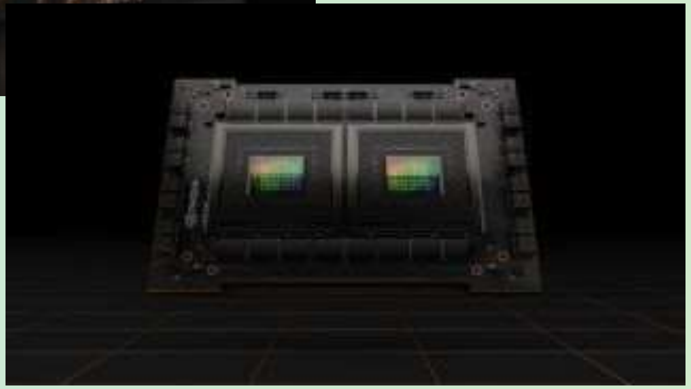
Motivation

CRAY
GW4
arm


TofuD Interface
PCIe Interface
HBM2 Interface
Core
Cache

CAVIUM
THUNDERX2

GW4
UK Research and Innovation
NVIDIA
arm
Hewlett Packard Enterprise

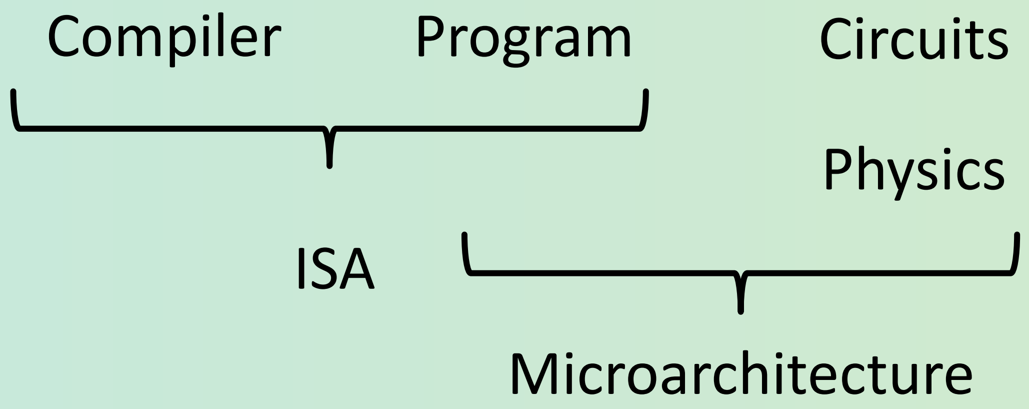


A Fair Comparison

- Power, Performance, Area
- Feature sets
  - Armv8-a+nosimd vs rv64g
- Compilers
- Benchmarks

Problem Definition

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$



The Workloads

- STREAM
  - Sustained memory bandwidth
  - 4 simple kernels
- CloverLeaf_Serial
  - High energy physics simulation on 2D cartesian grid
  - Many kernels each working over entire grid
- miniBUDE
  - Approximation of molecular docking simulation
- Lattice Boltzmann
  - Computational fluid dynamics
  - d2q9-bgk serial code
- minisweep
  - Radiation transportation mini app

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

The Compilers

GCC 9.2   GCC 12.2

-march=armv8-a+nosimd -mtune=cortex-a55 –static

-march=rv64g -mtune=sifive-7-series –static

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

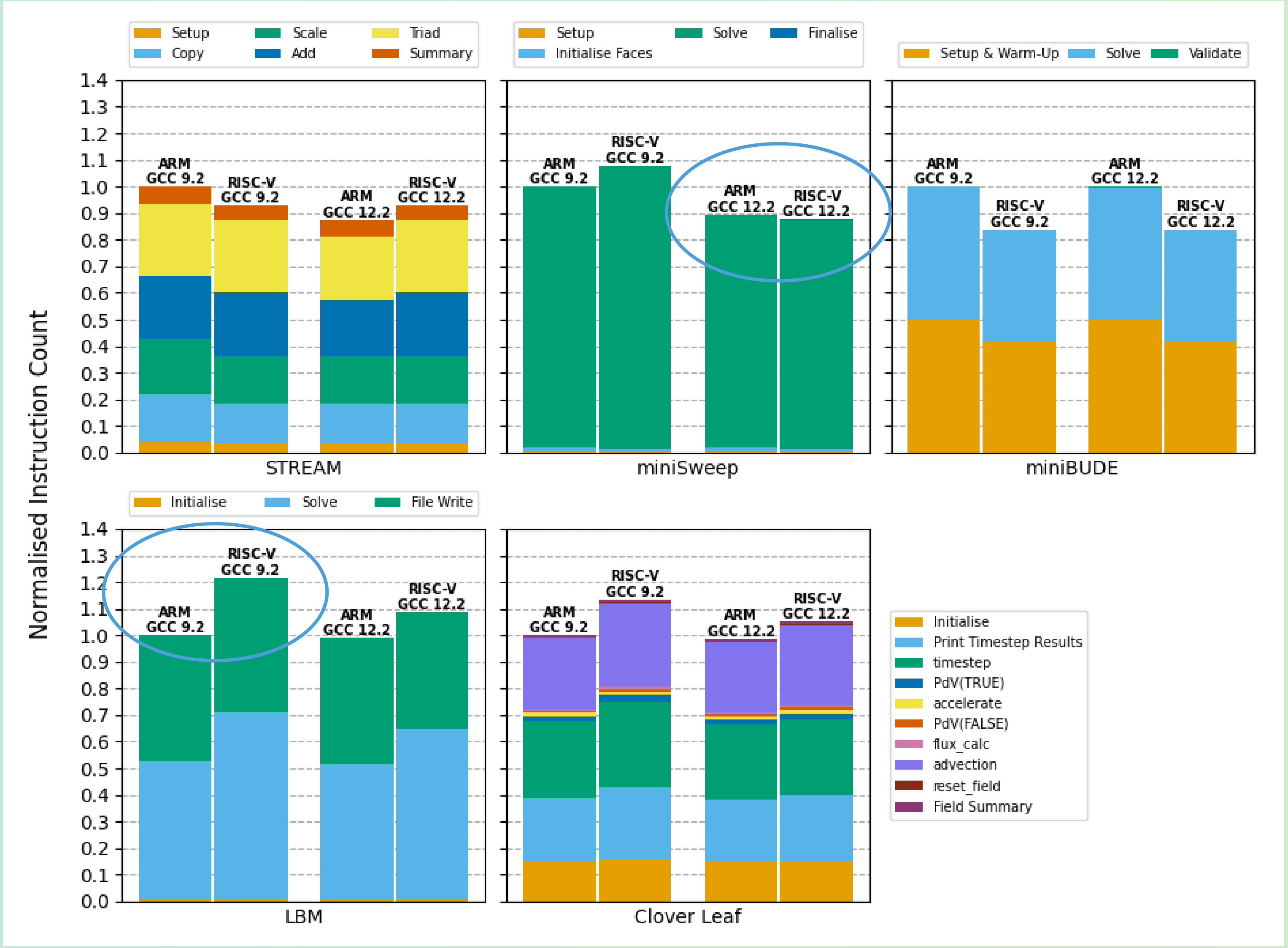
SimEng

- CPU microarchitecture simulator
- Open source
- Fast
- Easily modifiable
- Accurate
- Supported ARMv8
- RISC-V support added

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

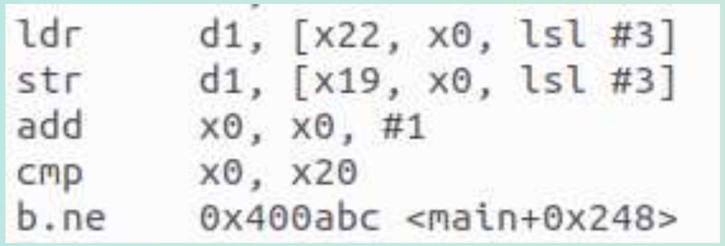
Path Length

- Benchmark source code modified to add instrumentation – NOPs around kernels
- Modified SimEng emulation core

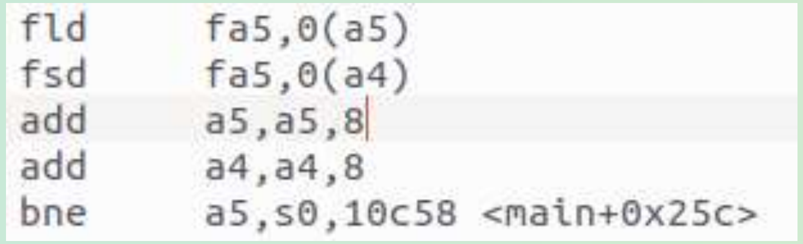


$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

STREAM Analysis



```
ldr     d1, [x22, x0, lsl #3]
str     d1, [x19, x0, lsl #3]
add     x0, x0, #1
cmp     x0, x20
b.ne    0x400abc <main+0x248>
```



```
fld     fa5,0(a5)
fsd     fa5,0(a4)
add     a5,a5,8
add     a4,a4,8
bne     a5,s0,10c58 <main+0x25c>
```

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Ideal CPI

- Ideal processor
    - Execute entire instruction in 1 cycle
    - Execute any number of instructions in 1 cycle
    - Perfect branch prediction
    - MUST obey RAW (true) dependencies between instructions
- How many cycles to complete a program? The longest chain of true dependencies
- Critical path (CP)
- Another modified SimEng emulation core

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Critical Path Length

- Instruction level parallelism (ILP) measure of no. instructions required to be executed per cycle to achieve ideal CPI

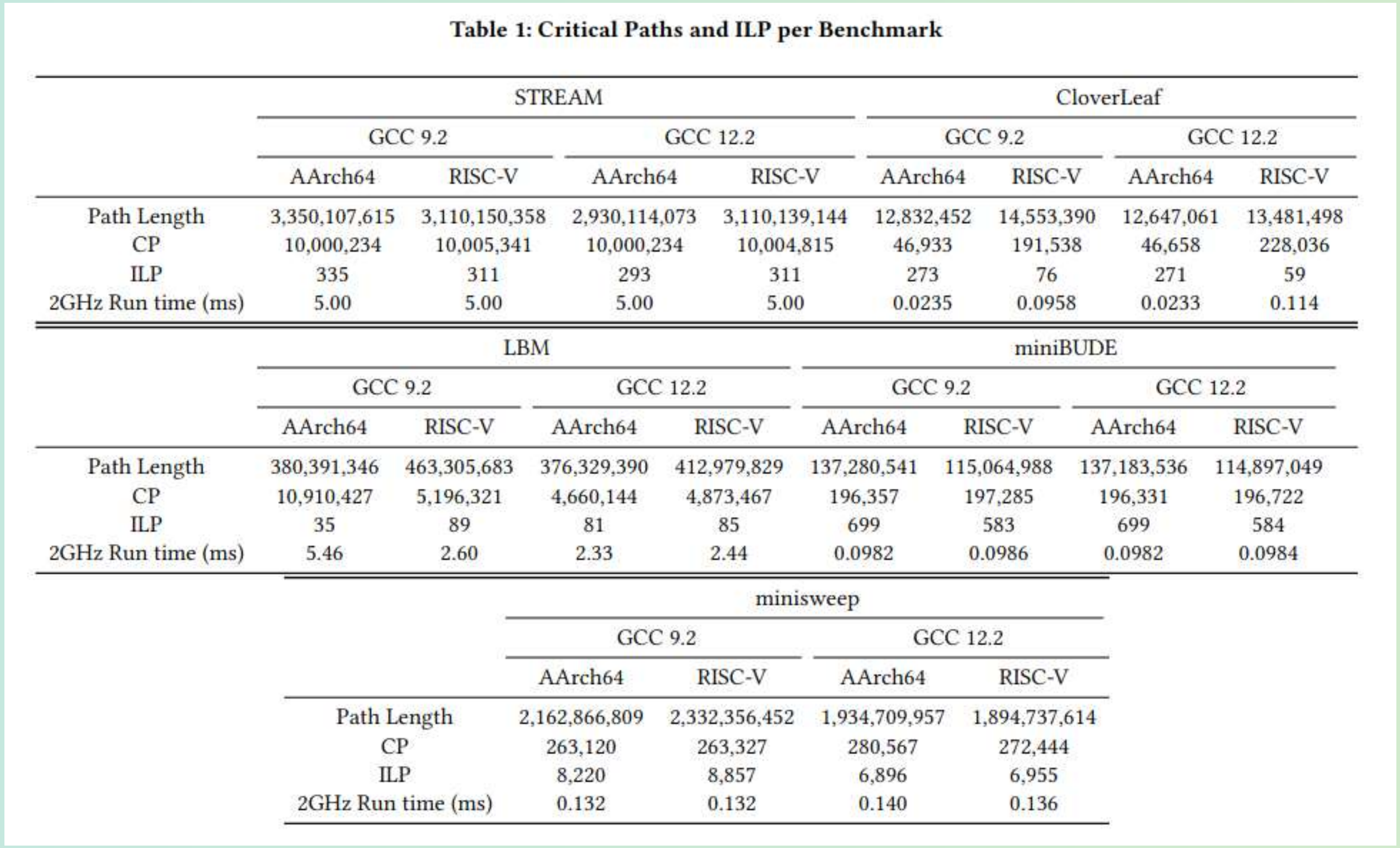


Table 1: Critical Paths and ILP per Benchmark

	STREAM				CloverLeaf			
	GCC 9.2		GCC 12.2		GCC 9.2		GCC 12.2	
	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V
Path Length	3,350,107,615	3,110,150,358	2,930,114,073	3,110,139,144	12,832,452	14,553,390	12,647,061	13,481,498
CP	10,000,234	10,005,341	10,000,234	10,004,815	46,933	191,538	46,658	228,036
ILP	335	311	293	311	273	76	271	59
2GHz Run time (ms)	5.00	5.00	5.00	5.00	0.0235	0.0958	0.0233	0.114

	LBM				miniBUDE			
	GCC 9.2		GCC 12.2		GCC 9.2		GCC 12.2	
	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V
Path Length	380,391,346	463,305,683	376,329,390	412,979,829	137,280,541	115,064,988	137,183,536	114,897,049
CP	10,910,427	5,196,321	4,660,144	4,873,467	196,357	197,285	196,331	196,722
ILP	35	89	81	85	699	583	699	584
2GHz Run time (ms)	5.46	2.60	2.33	2.44	0.0982	0.0986	0.0982	0.0984

	minisweep			
	GCC 9.2		GCC 12.2	
	AArch64	RISC-V	AArch64	RISC-V
Path Length	2,162,866,809	2,332,356,452	1,934,709,957	1,894,737,614
CP	263,120	263,327	280,567	272,444
ILP	8,220	8,857	6,896	6,955
2GHz Run time (ms)	0.132	0.132	0.140	0.136

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Scaled CPI

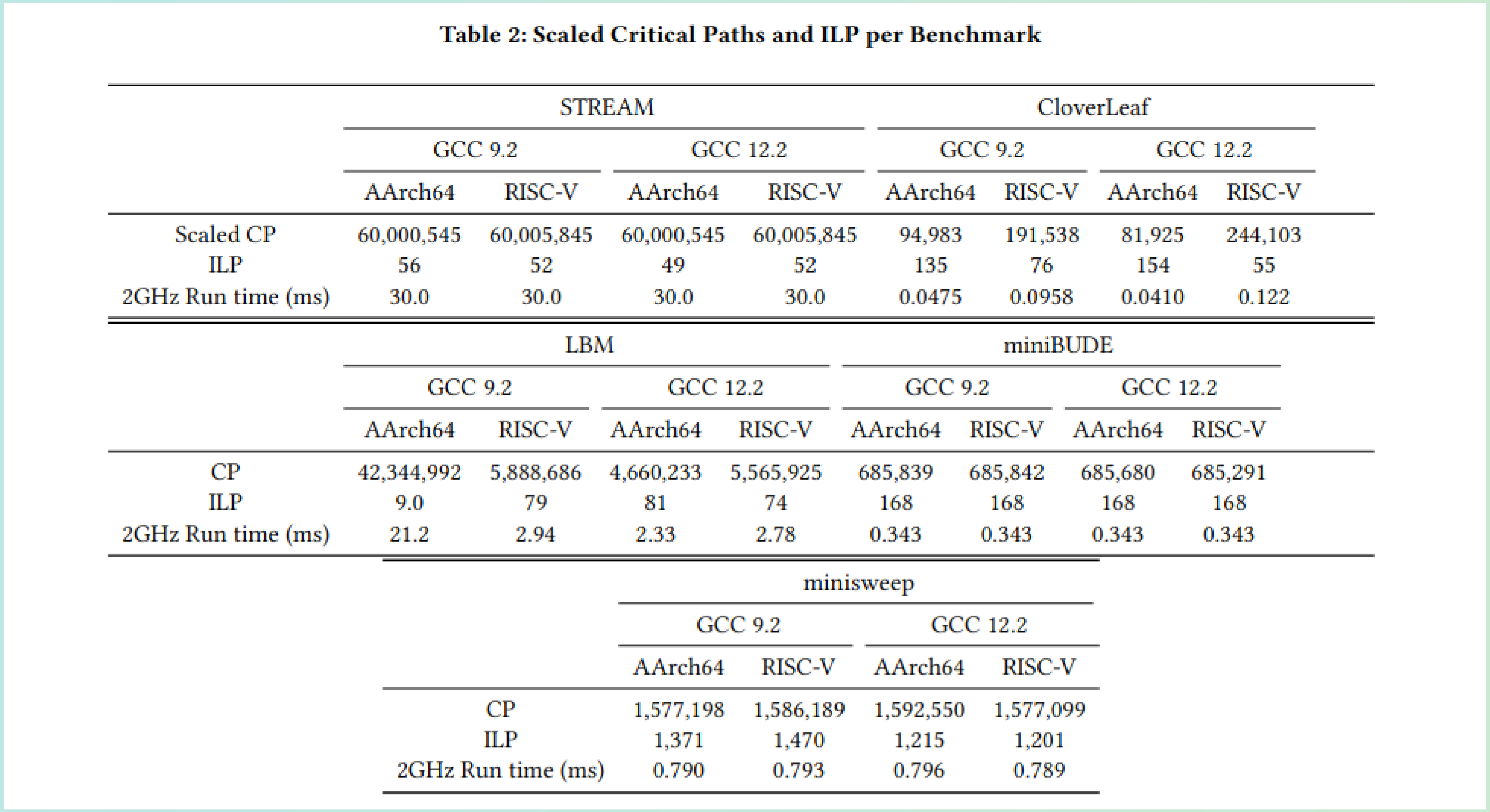


Table 2: Scaled Critical Paths and ILP per Benchmark

	STREAM				CloverLeaf			
	GCC 9.2		GCC 12.2		GCC 9.2		GCC 12.2	
	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V
Scaled CP	60,000,545	60,005,845	60,000,545	60,005,845	94,983	191,538	81,925	244,103
ILP	56	52	49	52	135	76	154	55
2GHz Run time (ms)	30.0	30.0	30.0	30.0	0.0475	0.0958	0.0410	0.122

	LBM				miniBUDE			
	GCC 9.2		GCC 12.2		GCC 9.2		GCC 12.2	
	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V	AArch64	RISC-V
CP	42,344,992	5,888,686	4,660,233	5,565,925	685,839	685,842	685,680	685,291
ILP	9.0	79	81	74	168	168	168	168
2GHz Run time (ms)	21.2	2.94	2.33	2.78	0.343	0.343	0.343	0.343

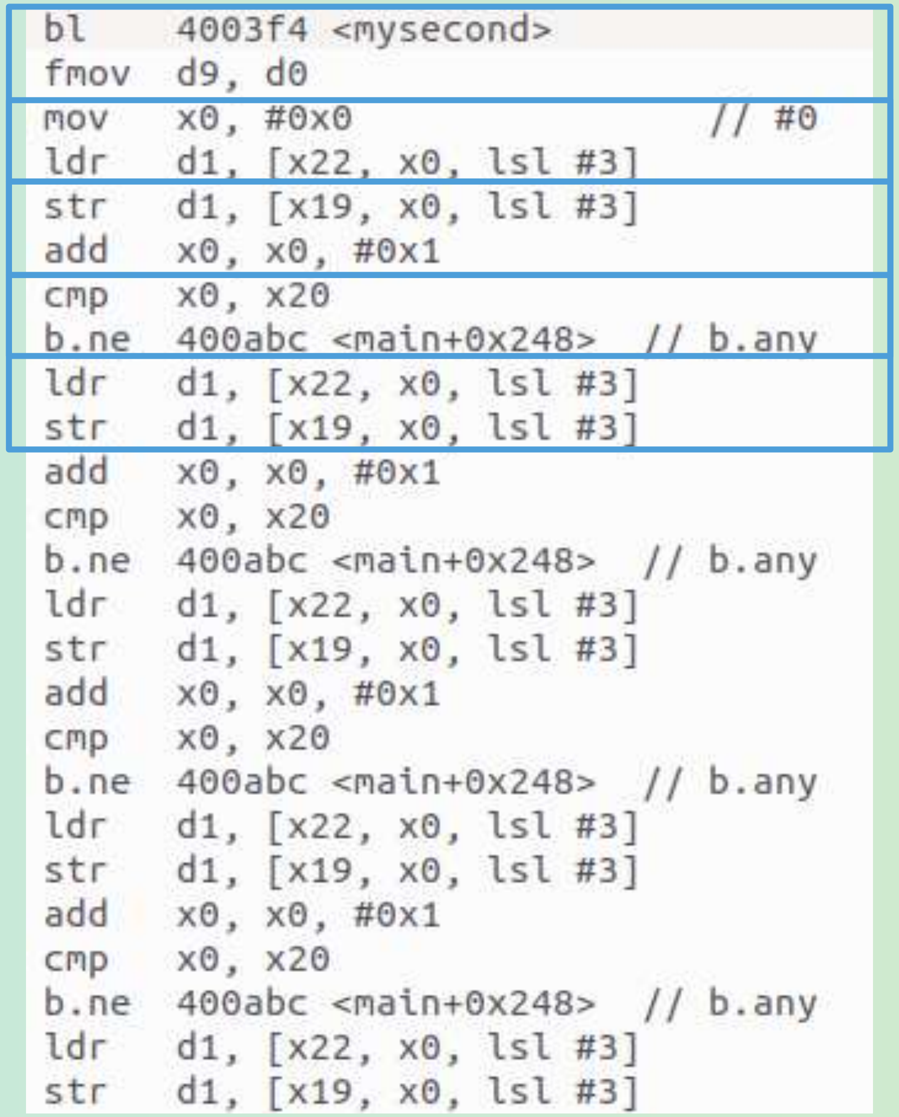
	minisweep			
	GCC 9.2		GCC 12.2	
	AArch64	RISC-V	AArch64	RISC-V
CP	1,577,198	1,586,189	1,592,550	1,577,099
ILP	1,371	1,470	1,215	1,201
2GHz Run time (ms)	0.790	0.793	0.796	0.789

- Scale CP by instruction latencies
- Latencies roughly follow that of TX2
- Do not scale for loads and stores

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Windowed CP

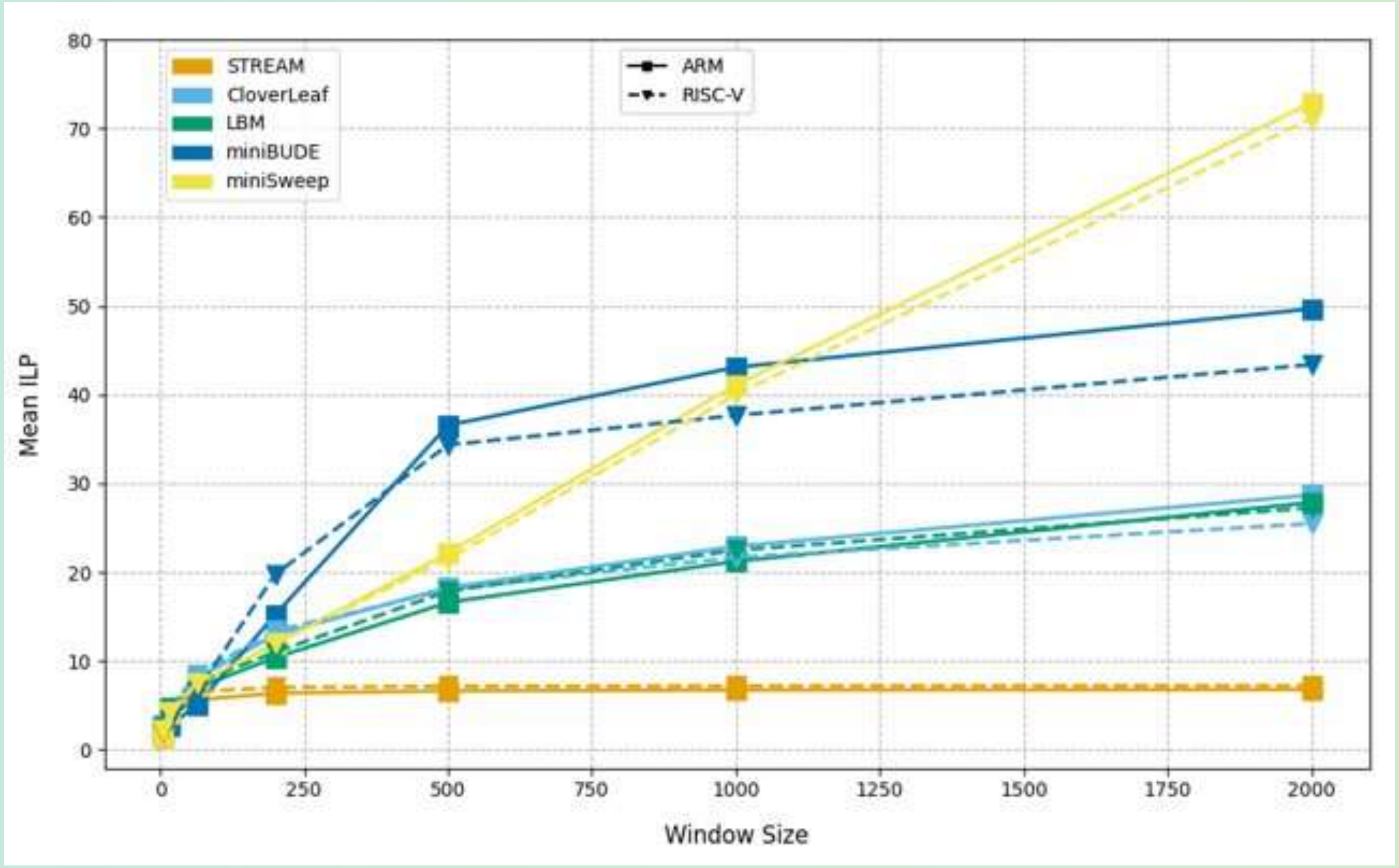
- Slide window down dynamic instruction trace
- Determine CP length per window
- Window size and stride parameterisable
- We choose values ranging from 4 – 2000
- Stride 50% of size
- GCC 12.2 only



```
bl     4003f4 <mysecond>
fmov   d9, d0
mov    x0, #0x0                  // #0
ldr    d1, [x22, x0, lsl #3]
str    d1, [x19, x0, lsl #3]
add    x0, x0, #0x1
cmp    x0, x20
b.ne   400abc <main+0x248>  // b.any
ldr    d1, [x22, x0, lsl #3]
str    d1, [x19, x0, lsl #3]
add    x0, x0, #0x1
cmp    x0, x20
b.ne   400abc <main+0x248>  // b.any
ldr    d1, [x22, x0, lsl #3]
str    d1, [x19, x0, lsl #3]
add    x0, x0, #0x1
cmp    x0, x20
b.ne   400abc <main+0x248>  // b.any
ldr    d1, [x22, x0, lsl #3]
str    d1, [x19, x0, lsl #3]
add    x0, x0, #0x1
cmp    x0, x20
b.ne   400abc <main+0x248>  // b.any
ldr    d1, [x22, x0, lsl #3]
str    d1, [x19, x0, lsl #3]
```

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Windowed CP Results



$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Conclusion and Future Work

- Full OoO superscalar microarchitecture simulation
- Varying available resources
- More benchmarks
- More compilers
- More ISA extensions

$$Program\ Execution\ Time = Path\ Length \times CPI \times Time\ per\ Cycle$$

Important Links

- Daniel Weaver – ra18837@bristol.ac.uk
- Paper - https://dl.acm.org/doi/10.1145/3624062.3624233
- Artifact - https://github.com/UoB-HPC/Arm-RISCV-Empirical-Comparison-Artifact
- SimEng repository - https://github.com/UoB-HPC/SimEng
- SimEng Documentation - https://uob-hpc.github.io/SimEng/