Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Semester Thesis

Benchmarking M-series Apple CPUs

Spring Term 2025

Supervised by:
Tommaso Pegolotti
Markus Püschel

Author:
Fabian Sidler

Contents

Abstract

This semester project investigates the performance characteristics of Apple's M-series CPUs, focusing on benchmarking the ARM-A Instruction Set Architecture (ISA) implemented in these processors. The research aims to develop automated benchmarks for measuring performance metrics such as throughput, latency, and µops, with a particular focus on mathematical instructions.

We address unique challenges associated with benchmarking on Apple Silicon Macs. Using kernel functions, this project provides an analysis of instruction performance on the M1 Pro and M3 Max CPUs. The methodology involves creating a benchmark pipeline that extracts documentation, generates tests, and processes results.

Chapter 1

Introduction

With the launch of the M1 CPU, Apple shifted from an x86_64 to the ARM-A Instruction Set Architecture (ISA). With this shift, Apple gained more control on the design and optimization of the CPU for its devices. This mainly showed in the implementation of heterogeneous cores, as well as a focus on enhancing performance and power efficiency. With the release of the Qualcomm Snapdragon X CPUs, the ARM-A ISA is gaining even more traction in the mobile computing world.

Since the x86_64 was the de facto standard ISA for any sort of computer (with exception for tablets or smartphones) for the last 20 years, there is plenty of research covering the x86_64 ISA. Research from project like µops.info[1] or Agner Fog[2] made detailed information about the CPU architecture accessible. The ARM-A ISA gains more and more importance, but still it is not supported by before mentioned projects. Dougal Johnson analyzed the Apple M1 CPU[3], but he did not extend his research to the newer M2 or M3 chips, nor did he release his sources, limiting further study. This work recreates the research from Dougall Johnson and extends it, by studying newer CPUs, as well as a broader set of instructions. The developed benchmarks can be reused to test CPUs implementing the ARM-A ISA, thus not only Apple CPUs with little effort.

At the beginning of this project these three goals were set.

1. Find a setup that allows for consistent benchmarking.
2. Measure performance of instructions.
3. Validate results.

This report documents the effort taken to fulfill the goals, gives background information, and discusses the results produced.

Chapter 2

Related Work

2.1 Benchmarking Microarchitectures

There are several projects with the aim of benchmarking microarchitectures, examples being Agner Fogs table[2] and µops.info[1]. While both of the mentioned examples provide detailed information on instruction throughput, latency and port information, they focus on the x86 ISA and mainly benchmark Intel and AMD CPUs and do not support the ARM-A profile ISA.

2.2 Benchmarking Apple M-series CPUs

Since the release of the apple M-series CPUs, there have been several approaches to benchmark the underlying CPUs and find out more about the CPUs.

The most prominent example, is the work from Dougal Johnson, who released a website[3] with information about the M1 CPUs. On the website, he presents an overview of the core architecture, with detailed information about bandwidths between pipeline stages for both the performance core and the efficiency core. Additionally, there is also a table similar to µops.info or Agner Fog, which display the latency, throughput and port information of the ARM-A ISA instructions measured on the M1 CPU. He obtained this information by running latency and throughput tests for each instruction alongside a custom kernel module[1] (which he strongly discourages to use). He did not extend his research on any CPUs than the M1 CPU. Additionally, he did not release the complete code to the project, thus it is hard to recreate.

In addition to Johnson's work, Daniel Lemire's blog[2] features posts about Apple's M-series processors. While Lemire's blogposts on benchmarks are not as extensive, he has published an article with code[4] that demonstrates how to interact with the kernel and retrieve detailed information about counting cycles and instructions similar to Johnson's work but without requiring a custom kernel module.

[1]https://github.com/dougallj/applecpu/tree/main/timer-hacks
[2]https://lemire.me/blog/

Chapter 3

Background

3.1 ARM-A Instruction Set Architecture (ISA) and Extensions

The ARM-A Instruction Set Architecture (ISA)[5] is the high performance ISA from ARM targeting mobile, PC and enterprise computing[5]. The ARM-A ISA comes in several versions, with version 9 being the most recent version released in 2021[6]. It is only since recently, that the industry switched to the most recent version 9. Up until the Apple M3 CPU, v8 of the ARM-a ISA was implemented as displayed in table 3.1. The ARM-A ISA establishes a set of essential instructions necessary for a CPU to function, called 'base' ISA in this report. Additionally, ARM provides several optional extensions, which add functionality and instructions for specific applications. These extensions are not mandatory and can be incorporated by CPU designers based on the specific requirements of the application. The following sections will cover the most prominent extensions. To determine which instruction set extensions are supported, users can run the `sysctl hw` command in the terminal on a Mac.

3.1.1 NEON extension

The base instructions support by the ARM-A ISA do not have support for floating point and vector (SIMD) instructions. The NEON extension defines a set of SIMD and floating point instructions with a vector length of up to 128-bit. A CPU manufacturer can choose to or not to implement these extensions. In the case of the NEON extension, it is common, that it is supported. All recent Apple mobile and computer CPUs support NEON.

3.1.2 SME/SVE Extension

ARM offers two extensions to the ARM-A ISA designed, to enhance its performance for vector and matrix operations:

- SVE (Scalable Vector Extension): This extension includes a scalable vector processing capability, allowing processors to handle a wide range of vector lengths, which can be adjusted according to the specific needs of different applications. SVE is aimed at improving performance for applications involving large data sets and high-performance computing tasks, such as scientific simulations and machine learning

- SME (Scalable Matrix Extension): SME focuses on accelerating matrix operations, which are crucial for machine learning and digital signal processing tasks. It provides efficient support for matrix manipulations and operations, offering improved performance and flexibility for these types of computations.

Both extensions are mainly targeted at high performance computing applications and thus not supported by consumer device CPUs (and also not by Apple designed CPUs).

3.1.3 ARM Performance Monitoring Units

The Performance Monitoring Unit (PMU) in ARM-A ISA is a hardware feature designed to provide detailed performance metrics and statistics on the processor's operation. It allows recording processor internal events such as cycles, branches, missed branches or retired instructions with the main goal of profiling software running on the processor. Tools like the perf tool[7] use the PMU to get information about the profiled application. Similar to NEON, SVE or SME, the PMU is also an extension to the ARM-A profile architecture and can optionally be implemented.

ARM released additional information as a PDF document describing PMUs in more detail[8]. Otherwise, there is always more information in the reference manual of the specific ISA[9]. In this work, the PMU was used to measure the amount of instructions/cycles passed during the benchmark.

3.2 Apple Developer Manual

Apple released an Apple Silicon CPU optimization guide to give developers insight on the architecture of the CPUs and allow them to optimize their software to run efficiently on Apple Silicon chips[10]. It mainly explains the supported ISA, but also gives insights on the CPU design, but also provides information about execution units (ports), latencies and throughput of instructions, as well as cache bandwidths. In context of this work, mainly the Appendix A is of interest.

3.3 Apple Silicon Desktop and Notebook CPUs

First introduced in 2020 with the M1 chip, Apple started releasing computers with Apple designed CPUs. They feature a unified memory, as well as specialized cores for performance and efficiency according to ARMs big.LITTLE architecture[1]. In table 3.1 the current lineup of apple CPUs is listed.

3.4 Apple Rosetta 2

Apple Rosetta 2 is a dynamic binary translator developed by Apple to allow apps built for x86_64 ISA based Macs to run on the newer Apple Silicon Macs[12]. Rosetta 2 can even translate x86_64 vector instructions, as long as the vector length is also supported on the Apple Silicon processor (Thus mainly the SSE vector extensions[2]). Dougall Johnson published an article going into the detail of how rosetta 2 works on his blog[13],

[1]https://www.arm.com/technologies/big-little
[2]https://www.intel.com/content/www/us/en/support/articles/000005779/processors.html

Processor	Performance Core	Efficiency Core	ISA	Core
M1	Firestorm @3.2 GHz	Icestorm @2.1 GHz	ARMv8.5-A	A14
M2	Avalanche @3.5 GHz	Blizzard @2.4 GHz	ARMv8.6-A	A15
M3	Everest @4.1 GHz	Sawtooth @2.8 GHz	ARMv8.6-A	A16
M4	-@4.4 GHz	-@2.9 GHz	ARMv9.2-A	A17

Table 3.1: Overview of the Apple Silicon M-series CPUs and their cores. The maximum clock frequencies are given next to the core names. The core column refers to the cores shared with the A-series CPUs, which are the processors used in iPhones. Maximum CPU frequencies can easily be determined with tools like cpufetch[11].

3.5 SIMD Everywhere

Usually, the application developers try to make the code as hardware-agnostic as possible and thus also write the application in a hardware-agnostic programming language. Unfortunately, if the code needs to be speed up, the developer often uses vector intrinsic functions and thus looses makes the code incompatible on different architectures.

The goal of SIMDE (Single Instruction, Multiple Data Everywhere)[3] is, to make the vector intrinsic instructions hardware-agnostic, by making x86_64 vector code runnable on ARM systems and vice versa. This is done, by replacing vector intrinsic functions before compiling the code. Additionally, it allows the use of vector lengths unsupported by certain systems, such as emulating 256-bit vector operations on CPUs that natively support only 128-bit vectors. This is achieved by replacing certain vector intrinsic instructions with multiple lower-level assembly instructions.

There is a similar library to achieve a similar result called SSE2NEON[4], which is popular as well, but does not allow emulating longer, unsupported vector lengths.

3.6 Benchmarking on Apple Silicon Macs

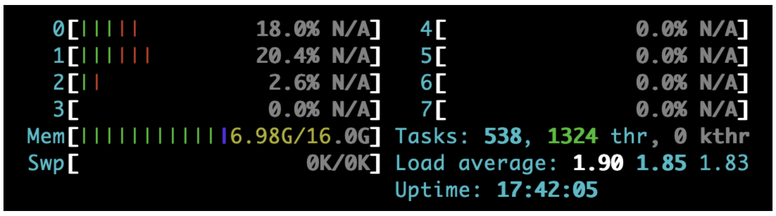
Accurately benchmarking the latency and throughput of instructions requires precise information about the CPU cycle count. While obtaining cycle counts was relatively straightforward on x86 processors (by disabling TurboBoost and using the TSC), the ARM-A architecture presents unique challenges.

The primary obstacle is the lack of a direct equivalent to the TSC on ARM-A CPUs. Moreover, macOS does not provide readily accessible APIs for accessing CPU performance counters. This section details the problems we faced during the development of our benchmarking infrastructure and how we overcame them.

3.6.1 CPU scaling

MacOS is quite secretive about the frequencies the CPU cores run at. We found no tool, which will report the frequencies. For example, the command-line tool `htop` does not report CPU core frequencies even when the feature is enabled, we can only see "N/A" instead of the frequencies (see figure 3.1). Nevertheless, there is a

[3]https://github.com/simd-everywhere/simde
[4]https://developer.arm.com/documentation/102581/0200/Port-with-SSE2Neon-and-SIMDe



```
  0[|||||         18.0% N/A]   4[               0.0% N/A]
  1[||||||        20.4% N/A]   5[               0.0% N/A]
  2[||             2.6% N/A]   6[               0.0% N/A]
  3[               0.0% N/A]   7[               0.0% N/A]
Mem[||||||||||||6.98G/16.0G] Tasks: 538, 1324 thr, 0 kthr
Swp[                 0K/0K] Load average: 1.90 1.85 1.83
                            Uptime: 17:42:05
```

Figure 3.1: Screenshot of the htop command on an M1 Pro macbook using macOS Sonoma.

possibility, to measure CPU frequencies: we can run a loop, where we increment a variable each cycle and measure how much time it takes to reach a specific number. Then we can simply divide the number of cycles/instructions by the time passed and get the average frequency during this time. Daniel Lemire wrote a blog post about this approach and even provides code for it[14]. When running, this code ourselves, we can clearly see, that the CPU frequency is dynamic and also depends, whether the application is scheduled on a performance or efficiency core.

3.6.2 ARM Processor Timers

The closest equivalent to the x86 TSC[5] timer in ARM are the processor timers[6] supported on ARM. There is a simple programmatic interface to access them, and no special OS privileges are required. On the tested Apple Silicon system, this timer runs at a constant frequency of 24 MHz. Due to CPU frequency scaling and since this timer runs at a constant frequency, it cannot be used for the kind of benchmarks we want to do, where we care about how many cycles pass during the execution of the code. Thus, it cannot be used similarly to the TSC timer in x86 systems.

3.6.3 Quality of Service (QoS) Performance Requests

As already mentioned, apple has deployed the ARM big.LITTLE architecture on the M-series CPUs. Thus, the Operating System has to decide whether to schedule the application on a performance or efficiency core. MacOS does not allow the user/programmer of an application to select the CPU affinity of a process, similar to the `taskset` [7] command in most Unix systems. The only interface to control the affinity of a process to a specific type of processor is the Quality of Service (QoS) interface [8].

This interface allows the programmer to specify the importance of a process to the operating system. The OS then prioritizes the high importance processes and runs them on the performance cores of the CPU. The interface differentiates between five QoS levels (User-interactive, User-initiated, Default, Utility, Background, Unspecified), which are translated to an integer between 0 and 33.

While this QoS level does not mandate, that the program will run on a specific (performance or efficiency) core, it is more likely, that the program runs on a performance core with a performance core if it is associated with a high QoS level.

[5]https://wiki.osdev.org/TSC
[6]https://developer.arm.com/documentation/102379/0101/The-processor-timers
[7]https://man7.org/linux/man-pages/man1/taskset.1.html
[8]https://developer.apple.com/library/archive/documentation/Performance/Conceptual/power_efficiency_guidelines_osx/PrioritizeWorkAtTheTaskLevel.html

As long as the processor is not fully utilized, this interface works well to specify whether to run code on a performance or efficiency core.

3.6.4 Benchmarking with Apple Instruments

Since 2019, apple provides the application "Xcode Instruments" for tuning and debugging applications running on Apple devices. This application comes with a graphical user interface, is relatively easy to use and has a lot of features. While most of the features are interesting to develop large applications and debug things like inter-process communication, it also gives access to the performance counter registers in the PMU (section 3.1.3) through the feature "CPU counters".

By using this feature, we can select different events from a list, which we want to track during the execution of a binary file. The available events are the same as the counters in the PMU. Thus, we can measure the number of cycles and instructions during the execution of a benchmark and calculate the throughput and latency as desired. One limitation of Apple Instruments that makes it less suitable for our application is its reliance on a graphical user interface. This makes it challenging to efficiently profile many binaries. For each binary, we would need to manually select it in the GUI, run the analysis, and then extract the output, which would be impractical for our use case, where we aim to benchmark several thousand binaries.

3.6.5 Reverse Engineered Access to PMU

To get accurate metrics for benchmarking instructions, directly accessing the PMU3.1.3 would be optimal. Both Dougall Johnson and Daniel Lemire used the performance counter registers of the M-series processors for benchmarking. Since Xcode Instruments does not offer an API for the performance counter registers, they used a reverse engineered way to access the PMU. Dougall Johnson developed a custom kernel module through which he could access the PMU. He himself discourages using his approach, as it poses a security thread and may mess with the kernel. Daniel Lemire used code developed by github user ibireme[9] to access the PMU. His approach only requires root permissions. This approach was also used for our benchmark setup. To benchmark code, PMU counters are read before and after the execution of the to be tested code snippet. A significant advantage of this approach is its independence from dynamic frequency scaling. Since we measure both cycles and instructions, we can accurately assess performance even if the processor's clock frequency changes during the execution.

As already mentioned, the PMU is an optional extension to the ARM ISA. All tested Apple CPUs do have a PMU present, but not all possible counters are present and depending on the core they differ. To figure out which events are supported, one can run the following command:

```
defaults read /usr/share/kpep/<cpu-type>.list
```

All `<cpu-type>.list` files supported by the kernel can be found in the `/usr/share/kpep/`. Thus, we can even inspect, what kind of events were supported by older generation cores used for the iPhone and even x86 CPUs. As the cores of the computer and smartphones CPUs are very similar, only the smartphone CPUs are listed. In table 3.1 we can see which cores were used in which M-series CPU. In table A.1 in the appendix, a list of all supported events on the M1 Processor can be found.

[9]https://github.com/ibireme/yybench/tree/master

Chapter 4

Benchmark Pipeline

The second goal outlined in the introduction was to measure the performance of the ARM-A ISA, focusing on throughput, latency, and the number of µops. To evaluate the entire ISA, these metrics were measured for each supported instruction. An automated pipeline was created to generate and run these tests. This section documents the pipeline and the steps taken to achieve this goal. Figure 4.1 provides an abstract overview, and the next sections explain each of the three steps in detail.

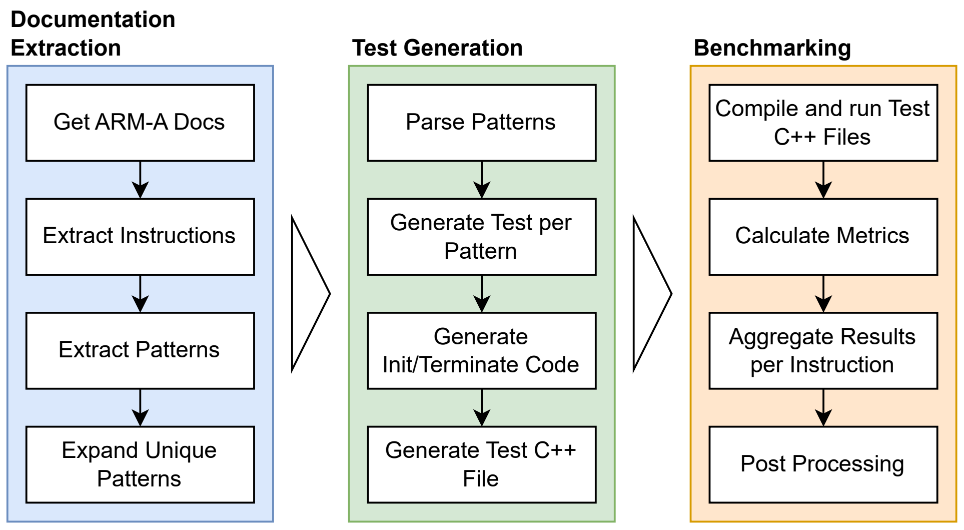


Figure 4.1: Abstract overview of the benchmarking pipeline, split into three main sections.

4.1 Documentation Extraction

To measure all ARM-A ISA instructions, we first need a list of instructions along with details like registers and operands. Since no machine-readable file for the ARM-A ISA was available, the first step in the pipeline was to generate a list of all instructions, including their register and operand patterns. The desired format is for each instruction looks like this example for the ADC[1] instruction: `ADC <Wd>, <Wn>, <Wm>`.

[1]`https://developer.arm.com/documentation/dui0801/g/A64-General-Instructions/ADC`

4.1.1 Extract Instructions

The first step in obtaining the necessary information was to scrape data from ARM's documentation[15], which is available online and can be downloaded[2]. Using the latest version (2024-06 at the time of writing), we extracted all instructions, operand patterns, register details, and metadata like brief descriptions from the HTML and XML files. Some instructions had multiple patterns. The extracted data was then saved in a JSON file for further processing. As an example, the data for the ADC instruction is shown below.

```
"brief": "Add with carry",
"configs":
        "instruction": "ADC",
        "op_string": "<Wd>, <Wn>, <Wm>",
        "registers": [
            {
            "desc": "Is the 32-bit name of the general-
            purpose destination register, encoded in
            the "Rd" field.",
            "name": "Wd",
            "position": 0,
            "text": "<Wd>"
            },
            {
            "desc": "Is the 32-bit name of the first
            general-purpose source register, encoded in
            the "Rn" field.",
            "name": "Wn",
            "position": 1,
            "text": "<Wn>"
            },
            {
            "desc": "Is the 32-bit name of the second
            general-purpose source register, encoded in
            the "Rm" field.",
            "name": "Wm",
            "position": 2,
            "text": "<Wm>"
            }]
```

A total of 720 base instructions and 372 NEON instructions were extracted from the documentation. Not all extracted instructions are easily testable, and some fail later in the pipeline due to unimplemented edge cases. Additionally, a list of all possible operands with brief descriptions was created. The ARM documentation includes the base ISA along with NEON, SVE, and SME extensions. The base ISA including all extensions was scrapped, but only the base and NEON instructions were processed further, as Apple M-series CPUs do not support the other extensions.

4.1.2 Extract Operand Patterns

Many instructions share the same operand patterns. For instance, both the ADD and SUB instructions use a pattern with three operands: one destination register and two source registers, represented in the documentation as `<Wd>, <Wn>, <Wm>`. To simplify the latter test case generation, the test cases can be created once per

[2]https://developer.arm.com/documentation/ddi0487/latest

pattern, and then the pattern is reused with several different instructions.

cFrom the scraped documentation, all unique operand patterns were extracted. The 720 base ISA instructions yielded 156 unique patterns, while the 372 NEON extension instructions resulted in 224 unique patterns.

4.1.3 Expanding Optional Parts and Multiple Options

In the ARM documentation, operand patterns often represent multiple use cases for the same instruction. To test every possible use case, all variations must be extracted. These use cases are encoded in the operand patterns, where operands may have multiple options or support additional optional operands. For example, one of the patterns for the ADD instruction illustrates this:

```
<Wd|WSP>, <Wn|WSP>, <Wm>{, <extend> {#<amount>}}
```

The first three operands, `<Wd|WSP>, <Wn|WSP>, <Wm>`, are required, while the parts in curly brackets represent optional operands. With two sets of nested brackets, this pattern can include three, four, or five operands, depending on whether none, one, or both optional operands are used. Additionally, the first two operands can be either a 32-bit register (Wd/Wn) or the stack pointer (WSP), indicated by the " `|` " character.

The generated tests should cover all possible usages of instructions, in case the metrics depend on the exact form of the instruction. Thus, we want to test for all combinations of optional patterns and operands:

First, all possible options for the optional parts are created, for the pattern from above, we get the following resulting patterns:

```
<Wd|WSP>, <Wn|WSP>, <Wm>
<Wd|WSP>, <Wn|WSP>, <Wm>, <extend>
<Wd|WSP>, <Wn|WSP>, <Wm>, <extend> #<amount>
```

Second, all possible options for the operands are created, thus, the first pattern from above expands into the following:

```
<Wd>,  <Wn>,  <Wm>
<Wd>,  <WSP>, <Wm>
<WSP>, <Wn>,  <Wm>
<WSP>, <WSP>, <Wm>
```

In this pattern, the three optional parts combined with four possible operand variations result in twelve different operand patterns. All other scalar and vector patterns are expanded similarly.

From the 150/224 patterns, extracted from the base/NEON instructions, 270/366 patterns were expanded due to optional parts or multiple operands.

4.2 Test Generation

The next step is, to use the extracted information from section 4.1 and generate valid assembly code to test latency, throughput and number of µops for each instruction. The intuition behind measuring each of these features is as follows:

- **Latency:** Latency is measured when there are dependencies between each line of code, meaning the next line cannot run until the current one finishes. Each line takes a number of cycles equal to the instruction's latency. Latency is calculated by dividing the total number of cycles by the number of instructions executed.

- **Throughput:** Throughput is measured when there are no dependencies between lines of code, allowing each line to execute independently. It is determined by dividing the number of instructions executed by the number of cycles.

- **Number of µops:** A specific counter in the PMU tracks the number of µops executed. The number of µops per instruction is calculated by dividing the total µops by the number of instructions executed. Either the latency or throughput test can be used for this calculation.

The tests are generated per unique pattern and then used for each instruction sharing this pattern. In the next subsections, the steps from instructions and patterns to C++ test files are explained.

4.2.1 Parsing Pattern

The abstract operand patterns from section 4.1 are parsed. To extract the operands from the operand string, the process involves two steps:

Step 1: The operand string is divided into a recursive list of operands and other elements (such as brackets and exclamation marks), structured similarly to an abstract syntax tree. Commas and characters like brackets are used as delimiters. An example of this tree is shown in figure 4.2 for the pattern `<Xt>,[<SP>,#<simm>]!`.

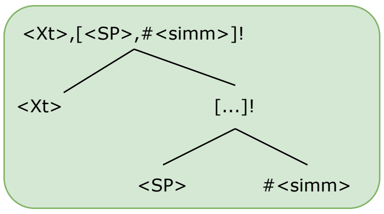


Figure 4.2: Illustrated abstract tree for the `<Xt>,[<SP>,#<simm>]!` pattern.

Step 2: Each operand is individually categorized as an operand object. Each operand object has a type and can have more attributes, like possible representations or values. As an example, the extracted operands objects for the example shown in figure 4.2 are described below:

- `<Xd>` → scalar register, 64-bit

- `[...]!` → memory access with write back

- `<SP>` → stack pointer, 64-bit

- `#<simm>` → signed immediate offset, a multiple of 16 in the range -4096 to 4080

The information about the operand object is then later used, when converting the abstract syntax tree back to executable code. The tree is traversed using a simple depth-first approach.

4.2.2 Register Access Pattern Generation

Depending on the goal of the benchmark (throughput or latency) the access to the registers must be different to get the desired dependencies as described in the beginning of this section. To test for throughput, there should be no read after write or write after write accesses to the same register. On the other hand, to test latency, read after write accesses are required.

This step is visualized with the `<Wd>, <Wn>, <Wm>` pattern, which can be used with the ADD instruction. In this pattern, there is one destination register `<Wd>` where the result is written two, as well as two source registers `<Wn>, <Wm>`. The generated access patterns are shown in table 4.1. To measure throughput, the destination register must change with each instruction to eliminate write-after-write conflicts. For latency measurement, specific read-after-write accesses are created, testing both source registers. Latency tests are performed only when at least one destination operand is in the pattern.

Test	Access Pattern
Throughput	`W0 , W16 , W17`
	`W1 , W16 , W17`
	`W2 , W16 , W17`
	`W3 , W16 , W17`
	`W4 , W16 , W17`
	`W5 , W16 , W17`
	`W6 , W16 , W17`
	`W7 , W16 , W17`
	`W8 , W16 , W17`
	`W9 , W16 , W17`
	`W10 , W16 , W17`
	`W11 , W16 , W17`
	`W12 , W16 , W17`
	`W13 , W16 , W17`
	`W14 , W16 , W17`
	`W15 , W16 , W17`
Latency 1→2	`W0 , W0 , W1`
Latency 1→3	`W0 , W1 , W0`

Table 4.1: Register accesses generated for the `<Wd>, <Wn>, <Wm>` operand pattern. Two latency patterns are generated such that read-after-write accesses are tested for both source registers.

We selected 16 different destination registers for throughput measurements, which allows testing architectures of up to 16 ports if the latency of the instruction is 1.0. For instructions with longer latencies, the maximum supported ports can be calculated by dividing 16 by the latency. During our testing, we did not reach the limit of 16 ports, as the M3 CPU is designed with a maximum of eight ports [10]. For future architectures, this number can easily be adjusted in the code.

4.2.3 Creating Initialization Termination Code

Depending on the operand, the code environment must be initialized such that the code can run. For example, source registers must be filled with values and for stack pointer usage, space must be allocated on the stack and restored after execution. This is handled during the initialization and termination code sequence. Since some instruction latencies (especially for floating-point tasks like square roots or divisions) depend on the values in the registers, instructions are tested with different values.

Passing complex floating-point numbers as arguments to inline assembly is not possible due to the limited number of bits available in the encoded instruction. To address this, multiple variables are defined in the template code. These can be passed as inputs to the inline assembly and can be referenced within it. Similarly, a large memory section is allocated and passed as a pointer to enable memory access operations. An example of initialization/termination code can be seen in section 4.2.5 in the code snippet.

4.2.4 Test Generation

As depicted in figure 4.3, the generated patterns, the initialization code and the assembly instructions are combined in this step, to a single inline assembly code block, which can then be measured to calculate throughput and latency of an instruction.

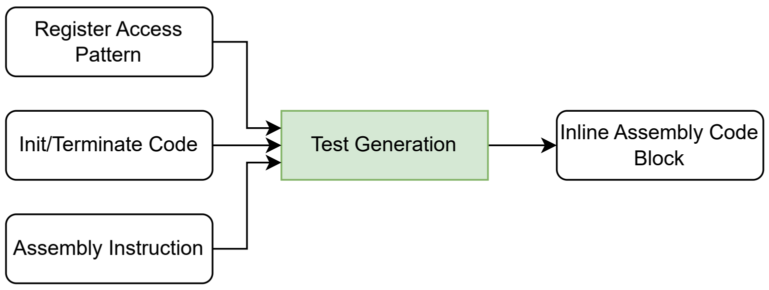


Figure 4.3: Benchmark inline Assembly generation

In Section 4.2.2, we generated multiple test cases for each operand pattern. These test cases were not directly mapped to specific instructions. In this step, each instruction is mapped to all tests it supports via the operand pattern. For example, the table 4.2 illustrates this process for the ADD instruction. Each row in the table corresponds to a separate test case. From the operand pattern test and the instruction, executable C++ tests are generated. This step is explained in the subsections below.

Instruction	Supported Patterns	Tests Generated for Pattern
ADD	`<Wd>, <Wn>, <Wm>`	Throughput
		Latency 1→2
		Latency 1→3
	`<Wd>, <WSP>, <Wm>`	Throughput
		Latency 1→3
	...	...

Table 4.2: Matching of the ADD instruction with two possible patterns and the corresponding tests.

Once the mapping is complete, we generate executable C++ tests from the combination of operand pattern tests and instructions. This process is detailed in the following subsections.

4.2.5 Inline Assembly Code

For each test, an inline assembly code block is generated. The test is unrolled (repeated) until it covers 1000 lines. It is also nested inside a loop implemented in assembly, which iterates over the test for 4000 times. In total, 4 million instructions are executed per test. Before the loop, the initialization code is inserted, during which the registers and the stack pointers are filled with values defined during register access pattern generation. Additionally, variables defined in the template can be used in case more complex values need to be tested. After the loop, the termination code in inserted.

```
asm inline volatile(
    "mov x1, #7 \n\t"                        // init registers
    "sub sp, sp, #128 \n\t"                  // init sp
    "str %[arr_ptr], [sp] \n\t"
    "mov x13, 0 \n\t"
"loop_top_%=: \n\t"
    "isb \n\t"                               // flush pipeline
    "add x0,x1,[sp] \n\t"
    // repeat UNROLL times
    "add x0,x1,[sp] \n\t"
    "isb \n\t"
    "add x13, x13, #1 \n\t"
    "cmp x13, #4000 \n\t"                    // 4000 = #iters
    "ble loop_top_%= \n\t"
    "add sp, sp, #128 \n\t"
    "isb"                                    // flush pipeline
    :                                        // outputs
    : [arr_ptr] "r" (array)                  // inputs
    : "x0", "x1", "x13", "x13", "memory" // clobbers
    );
```

Listing 4.1: Example of inline assembly used to benchmark the ADD instruction.

4.2.6 Structure of the Test Case C++ File

The inline assembly code is then inserted into a template C++ file, which implements the benchmarking infrastructure. The `bench_func` is benchmarked. The python module modifies this function and inserts the inline assembly. The execution of this function is benchmarked `REPEAT` times using the `kperf_get_counters` function. From the results of each round, the minimum of the REPEAT metrics are taken, as they are closest to the lower bound. If the code should be executed on the efficiency core, the `EFFICIENCY_CORE` flag needs to be set. Down below, a shortened version of the template can be found.

```
// constants defined by python module
#define EFFICIENCY_CORE 0
#define UNROLL 1000
#define ITERS 4000
#define REPEAT 5
#define ARRAY_SIZE 10000
```

```
int array[ARRAY_SIZE] = {0};

float fpi = 3.14159265358979323846;
double dpi = 3.14159265358979323846;

inline volatile int bench_func(){
    // code <- inline assembly inserted here
    return 0;
}

int main() {
    #if EFFICIENCY_CORE
    pthread_set_qos_class_self_np(QOS_CLASS_BACKGROUND, 0);
    #else
    pthread_set_qos_class_self_np(
                            QOS_CLASS_USER_INTERACTIVE, 0);
    #endif

    kperf_init();
    performance_counters start, end, best;
    for (size_t i = 0; i < REPEAT; i++) {
        start = kperf_get_counters();
        bench_func();
        end = kperf_get_counters();

        // update counters
    }

    // print output
}
```

Listing 4.2: Simplified benchmark template file, // code gets replaced with actual inline code.

4.2.7 PMU Library

The `kperf_get_counters` function is implemented in a custom library, which dynamically loads and wraps the kernel libraries required to access the PMU. This custom library is based on the demonstration from GitHub user ibireme about reading out the PMU[3]. The library exposes a simple function with the above-mentioned metrics can be counted.

Porting Test to Different OS's

To execute the benchmark on a different operating system than macOS, the PMU library needs to be adapted to the new kernel, since this library relies on Apple's XNU kernel[4].

[3]https://gist.github.com/ibireme/173517c208c7dc333ba962c1f0d67d12

[4]`https://github.com/apple-oss-distributions/xnu/blob/main/osfmk/kern/kpc.h`

4.3 Benchmarking

The test C++ files with the inline assembly block inserted is compiled using the preinstalled clang compiler (`/usr/bin/clang`) with optimization level 3 (`-O3`). The binary is executed using root access levels (to get access to performance counters) and outputs the information from the performance monitoring unit in a JSON format. The output includes the following information about the `bench_func`:

- executed instructions
- passed cycles
- branches/branch misses
- retired µops categorized in: scalar, simd/fp, load-store

The functions `kperf_get_counters`, along with the initialization/termination code and the call to `bench_func`, introduce a significant overhead that must be subtracted to get accurate results. To address this, each benchmark is initially run without testing any inline assembly instructions, using only the initialization, termination code, and the loop. The results from this 'empty' run are then subtracted from the actual test results to reduce the impact of the benchmarking setup.

4.3.1 Post Processing

After the execution of the benchmarks, the information from the performance counters is aggregated and the throughput of each instruction is calculated by dividing the total executed instructions by the total passed cycles. The throughput is then rounded to the nearest value from a predefined list of possible divisors and their reciprocals. The divisors range from $\frac{1}{2}$ to $\frac{1}{25}$, the reciprocals from 1 to 25. The divisors round the inverse throughput, while the reciprocals handle the throughput rounding. Additional information such as the used port(s) and the number of executed µops per instruction is also determined in this step.

After that, the benchmarks are grouped by instruction, such that we get one row per instruction and show similar information as Agner Fogs table[2].

Chapter 5

Results & Validation

The developed benchmark pipeline created 24481 base and 19269 NEON ISA performance tests for the 720/372 base/NEON instructions. The results produced, as well as the quality of results produced, are discussed in the next sections. The focus in this results section is on the results generated with the M1 Pro macbook, as these results are also used for the validation. In the appendix, the results generated are listed for both the M1 Pro and the M3 Max CPU.

5.1 Results Analysis

The performance test development primarily targeted benchmarking mathematical instructions. Nevertheless, many additional instructions were also successfully benchmarked. Table 5.1 lists the number of instructions tested for both the base and NEON ISAs.

Instructions	base ISA	NEON ISA
Extracted	720	372
Tests generated	720	372
Tests Successful	398	237

Table 5.1: Number of instructions successfully tested compared to all extracted instructions.

For the 375/297 instructions left untested, several reasons were identified:

- Unsupported instructions: Several instructions were tested but were not supported by the implemented ISA. For example, the scalar ABS[1] instruction, listed in ARM documentation, is not implemented on the M1 Pro CPU. Tests with such instructions typically resulted in a compile error similar to this: "error: instruction requires: cssc." In the case of the ABS instruction, the issue is that scalar ABS is only supported in newer ARM-A ISAs, starting from ARM-A v8.7[16].
- Non-mathematical instruction: This work focused on benchmarking mathematical instructions. However, an ISA, particularly the base ISA, includes many instructions necessary for operating system functionality and system operation, such as those for security, peripherals, or accessing special registers.

[1]https://developer.arm.com/documentation/ddi0602/2024-06/Base-Instructions/ADD–immediate—Add–immediate–

As a result, no valid tests were generated for many of these non-mathematical instructions.

- Invalid assembly: While the test generation generally works well, it still produces faulty code in some cases, leading to errors during compilation or execution. A common example is memory access operations, which can result in segmentation faults.
- Unsupported patterns: While the test generation section 4.2 supports many patterns and operands, some remain unimplemented, leading to failures in generation. Examples of unimplemented patterns are ones with operands such as `<tlbi_op>, <option>` and `<dc_op>`. These Operands were not implemented due to limited time and the focus on the mathematical instructions. Consequently, if a pattern for a particular operand is not implemented, it cannot be tested.

5.2 Validation

To ensure the reliability of our results, we ran the benchmarks on an M1 Pro macbook and validated the data against the results from Dougall Johnson's research[3]. Johnson benchmarked both the base and NEON ISAs, providing performance metrics for 213/326 base/NEON instructions. We scraped Johnson's results and used them for validation. The challenge was matching the generated patterns (as described in Section 4.1.2) to Johnson's results, as his benchmarks did not specify the exact pattern used. To simplify the process, validation was done per instruction rather than per pattern. For each instruction, all relevant benchmarks were combined, keeping the minimum and maximum values for throughput, latency, and number of µops. If these values matched Johnson's benchmark results, the validation was considered successful. The results from the validation can be seen in table 5.2.

Instructions	Efficiency Core		Performance Core	
	Base	NEON	Base	NEON
Matched for validation	205	230	205	230
Successfully validated	135	213	135	157

Table 5.2: Results from the validation with results from Johnson, all numbers are count of instructions benchmarked.

Not all the matched instructions could be validated with ones from Johnson. When analyzing the data, we found four main causes:

1. We covered more cases: In some instances, like with the scalar ADD instruction, our test case generator addressed a wider range of scenarios than Johnson's work, resulting in more varied results. This led to a broader set of cases being tested, producing more diverse performance metrics. Two examples of this are the test cases generated for the ADD instruction[2]:
   - Cases where #0 is added to a register in a pattern like the following: `ADD <Xd>,<Xn>,#<imm>`, register renaming is happening, and we get an

[2]`https://developer.arm.com/documentation/ddi0602/2024-06/Base-Instructions/ADD--immediate---Add--immediate--`

  inverse throughput of 0.125 instead of 0.167 coming from the 6 ALU ports.

  - In cases where the stack pointer is a source and destination register (pattern like `ADD <SP>,<SP>,#<imm>`), the inverse throughput can only be 1.0, since the end of the execution needs to be awaited due to a read after write.

2. No Latency test case: As mentioned in 4.2.2, a latency test case is only generated, if a destination operand is present in the pattern. Johnson has invested more time and generated latency tests for specific instructions, such as the LDUR[3] instruction.

3. Potential errors in Johnson's results: For instance, in the case of the STADD (64-bit)[4] instruction, Johnson's table (firestorm/base[5]) lists a latency of 10 cycles. However, on the specific webpage for this instruction[6], two different values are provided, 10 and 11 cycles, depending on the number of iterations/unrolls. Our tests consistently yielded a latency of 11 cycles, suggesting that the value of 10 in the table may be incorrect.

4. Incorrect results: For instance, in the case of the AUTDA[7] instruction, our tests did not produce accurate results. The observed throughput and latency suggest that the instruction may not have been executed and was likely resolved through register renaming. Given that the focus of this project was on benchmarking mathematical operations, this discrepancy was deemed acceptable.

We were unable to automatically assess why certain results didn't match for each row. Typically, if the numbers are in the right ballpark, they are correct. When the results were incorrect, they were significantly so, as seen with the AUTDA instruction. Overall, we believe the results are reliable but should be interpreted with caution. After manual validation, the mathematical operations, in particular, appear to be accurate.

[3] `https://dougallj.github.io/applecpu/measurements/firestorm/LDUR_64.html`

[4] `https://developer.arm.com/documentation/dui0801/l/A64-Data-Transfer-Instructions/STADD--STADDL--STADDL--A64-`

[5] `https://dougallj.github.io/applecpu/firestorm-int.html`

[6] `https://dougallj.github.io/applecpu/measurements/firestorm/STADD_64.html`

[7] https://developer.arm.com/documentation/dui0801/g/A64-General-Instructions/AUTDA–AUTDZA

Chapter 6

Misc Tests

6.1 x86 LLVM to ARM-A assembly

LLVM creates a human-readable language independent assembly like intermediate representation (IR) . The IR is portable by design and even employs architecture independent vector instructions. This raised the question if x86 intrinsic code can be compiled to LLVM IR and then further assembled with the ARM-A architecture as target. To test this out, we took a simple x86 code file `llvm_test.c` with some basic 128-bit vector instructions.

1. It was compiled using rosetta 2, with the -emit-llvm flag to output the IR:

```
$env /usr/bin/arch -x86_64 /usr/bin/clang
    -O2 -emit-llvm -S llvm_test.c
```

2. Then it was further compiled with the native target without rosetta 2, thus for the ARM-A ISA. With the -S flag, the output is assembly, for analysis:

```
$env /usr/bin/arch -arm64-apple-darwin23.5.0
    /usr/bin/clang -target aarch64 -O2 -g llvm_test.ll
    -S -o llvm_test_arm_native.s
```

The resulting assembly code does not have any vector instructions, and only scalar assembly instructions were used. When the second step was done using an x86_64 target, vector instructions were used. Thus, we were not able to use the LLVM IR to effectively replace SIMDe (section 3.5) or similar projects.

6.2 Memory Bandwidth Benchmark

One of the limitations, we come across when optimizing code is the memory bandwidth. Put simply, you simply have to measure how much data the processor can read or write in a given time. To fully utilize maximum memory bandwidth, several threads need to be spawned. Thankfully, there is a tested project, which implements a memory bandwidth test, called TheBandwidthBenchmark[1]. For this test it is important, to close all applications, which could interfere with the test. Even then, the specified maximum of 200 GB/s[2] could not be reached. Measuring on an M1 Pro CPU, we achieved 173 GB/s.

[1]https://github.com/RRZE-HPC/TheBandwidthBenchmark
[2]https://support.apple.com/en-by/111901

6.3 More Interesting Sources

Here are some links to webpages and sources collected, which were looked at during this semester project. They might be of interest to the reader of the report.

- Performance counters in the macOS XNU kernel: `https://github.com/apple-oss-distributions/xnu/blob/main/osfmk/kern/kpc.h`
- Supported CPUs by Apples version of the llvm compiler: `https://github.com/swiftlang/llvm-project/blob/next/llvm/lib/Target/AArch64/AArch64Processors.td`
- Pull request comment in the Asahi Linux project talking about frequency management on Apple M-series CPUs: `https://git.kernel.org/pub/scm/linux/kernel/git/rafael/linux-pm.git/commit/?h=linux-next&id=6286bbb40576ffadfde206c332b61345c19af57f`
- Wiki from Asahi Linux about the M-series CPUs: `https://github.com/AsahiLinux/docs/wiki`
- Article on how macOS handles CPU core scheduling: `https://eclecticlight.co/2022/04/25/how-macos-manages-m1-cpu-cores/`
- Article comparing x86 and ARM timers: `https://cpufun.substack.com/p/fun-with-timers-and-cpuid`

Bibliography

[1] A. Abel and J. Reineke, “uops.info: Characterizing latency, throughput, and port usage of instructions on intel microarchitectures,” in *ASPLOS*, ser. ASPLOS ’19. New York, NY, USA: ACM, 2019, pp. 673–686.

[2] A. Fog, “Instruction tables. lists of instruction latencies, throughputs and micro-operation breakdowns for intel, amd and via cpus,” 2022.

[3] D. Johnson, “Apple microarchitecture research,” https://dougallj.github.io/applecpu/firestorm.html, accessed: 2024-09-23.

[4] D. Lemire, “Counting cycles and instructions on arm-based apple systems,” https://lemire.me/blog/2023/03/21/counting-cycles-and-instructions-on-arm-based-apple-systems/, accessed: 2024-09-23.

[5] ARM, “A-profile architecture,” https://developer.arm.com/Architectures/A-Profile%20Architecture, accessed: 2024-09-23.

[6] A. Frumusanu, “Arm announces armv9 architecture: Sve2, security, and the next decade,” https://www.anandtech.com/show/16584/arm-announces-armv9-architecture, accessed: 2024-09-23.

[7] wiki.perf, “perf: Linux profiling with performance counters,” https://perf.wiki.kernel.org/index.php/Main_Page, accessed: 2024-09-23.

[8] ARM , “Arm coresight performance monitoring unit architecture,” https://developer.arm.com/documentation/ihi0091/latest/, accessed: 2024-09-23.

[9] ——, “Arm architecture reference manual for a-profile architecture,” https://developer.arm.com/documentation/ddi0487/latest/, accessed: 2024-09-23.

[10] Apple, “Apple silicon cpu optimization guide: 3.0,” 2024.

[11] Dr-noob, “cpufetch: Simple yet fancy cpu architecture fetching tool,” https://github.com/Dr-Noob/cpufetch, accessed: 2024-09-24.

[12] Apple, “About the rosetta translation environment,” https://developer.apple.com/documentation/apple-silicon/about-the-rosetta-translation-environment, accessed: 2024-09-23.

[13] D. Johnson, “Why is rosetta 2 fast?” https://dougallj.wordpress.com/, accessed: 2024-09-23.

[14] D. Lemire, “Measuring the system clock frequency using loops (intel and arm),” https://lemire.me/blog/2019/05/19/measuring-the-system-clock-frequency-using-loops-intel-and-arm/, accessed: 2024-09-23.

[15] ARM , “A64 – base instructions (alphabetic order),” https://developer.arm.com/documentation/ddi0602/2024-06/Base-Instructions, accessed: 2024-09-23.

[16] ——, “Arm compiler for embedded reference guide,” https://developer.arm.com/documentation/101754/0622/armclang-Reference/armclang-Command-line-Options/-march, accessed: 2024-09-29.

Appendix A

Apple M1 PMU Events

Event	Brief description
ATOMIC_OR_EXCLUSIVE_FAIL	Atomic or exclusive instruction failed (due to contention)
ATOMIC_OR_EXCLUSIVE_SUCC	Atomic or exclusive instruction successfully completed
BRANCH_CALL_INDIR_MISPRED_NONSPEC	Retired indirect call instructions mispredicted
BRANCH_COND_MISPRED_NONSPEC	Retired conditional branch instructions that mispredicted
BRANCH_INDIR_MISPRED_NONSPEC	Retired indirect branch instructions including calls and returns that mispredicted
BRANCH_MISPRED_NONSPEC	Retired branch instructions including calls and returns that mispredicted
BRANCH_RET_INDIR_MISPRED_NONSPEC	Retired return instructions that mispredicted
CORE_ACTIVE_CYCLE	Cycles while the core was active
FETCH_RESTART	Fetch Unit internal restarts for any reason. Does not include branch mispredicts
FIXED_CYCLES	
FIXED_INSTRUCTIONS	INST_ALL
FLUSH_RESTART_OTHER_NONSPEC	Pipeline flush and restarts that were not due to branch mispredictions or memory order violations
INST_ALL	All retired instructions
INST_BARRIER	Retired data barrier instructions
INST_BRANCH	Retired branch instructions including calls and returns
INST_BRANCH_CALL	Retired subroutine call instructions
INST_BRANCH_COND	Retired conditional branch instructions (counts only B.cond)
INST_BRANCH_INDIR	Retired indirect branch instructions including indirect calls

INST_BRANCH_RET	Retired subroutine return instructions
INST_BRANCH_TAKEN	Retired taken branch instructions
INST_INT_ALU	Retired non-branch and non-load/store Integer Unit instructions
INST_INT_LD	Retired load Integer Unit instructions
INST_INT_ST	Retired store Integer Unit instructions
INST_LDST	Retired load and store instructions
INST_SIMD_ALU	Retired non-load/store Advanced SIMD and FP Unit instructions
INST_SIMD_LD	Retired load Advanced SIMD and FP Unit instructions
INST_SIMD_ST	Retired store Advanced SIMD and FP Unit instructions
INTERRUPT_PENDING	Cycles while an interrupt was pending because it was masked
L1D_CACHE_MISS_LD	Loads that missed the L1 Data Cache
L1D_CACHE_MISS_LD_NONSPEC	Retired loads that missed in the L1 Data Cache
L1D_CACHE_MISS_ST	Stores that missed the L1 Data Cache
L1D_CACHE_MISS_ST_NONSPEC	Retired stores that missed in the L1 Data Cache
L1D_CACHE_WRITEBACK	Dirty cache lines written back from the L1D Cache toward the Shared L2 Cache
L1D_TLB_ACCESS	Load and store accesses to the L1 Data TLB
L1D_TLB_FILL	Translations filled into the L1 Data TLB
L1D_TLB_MISS	Load and store accesses that missed the L1 Data TLB
L1D_TLB_MISS_NONSPEC	Retired loads and stores that missed in the L1 Data TLB
L1I_CACHE_MISS_DEMAND	Demand fetch misses that require a new cache line fill of the L1 Instruction Cache
L1I_TLB_FILL	Translations filled into the L1 Instruction TLB
L1I_TLB_MISS_DEMAND	Demand instruction fetches that missed in the L1 Instruction TLB
L2_TLB_MISS_DATA	Loads and stores that missed in the L2 TLB
L2_TLB_MISS_INSTRUCTION	Instruction fetches that missed in the L2 TLB
LDST_X64_UOP	Load and store uops that crossed a 64B boundary
LDST_XPG_UOP	Load and store uops that crossed a 16KiB page boundary
LD_NT_UOP	Load uops that executed with non-temporal hint
LD_UNIT_UOP	Uops that flowed through the Load Unit
MAP_DISPATCH_BUBBLE	Bubble detected in dispatch stage

MAP_INT_UOP	Mapped Integer Unit uops
MAP_LDST_UOP	Mapped Load and Store Unit uops, including GPR to vector register converts
MAP_REWIND	Cycles while the Map Unit was blocked while rewinding due to flush and restart
MAP_SIMD_UOP	Mapped Advanced SIMD and FP Unit uops
MAP_STALL	Cycles while the Map Unit was stalled for any reason
MAP_STALL_DISPATCH	Cycles while the Map Unit was stalled because of Dispatch back pressure
MMU_TABLE_WALK_DATA	Table walk memory requests on behalf of data accesses
MMU_TABLE_WALK_INSTRUCTION	Table walk memory requests on behalf of instruction fetches
MMU_VIRTUAL_MEMORY_FAULT_NONSPEC	Memory accesses that reached retirement that triggered any of the MMU virtual memory faults
RETIRE_UOP	All retired uops
SCHEDULE_UOP	Uops issued by the scheduler to any execution unit
ST_MEMORY_ORDER_VIOLATION_NONSPEC	Retired stores that triggered memory order violations
ST_NT_UOP	Store uops that executed with non-temporal hint
ST_UNIT_UOP	Uops that flowed through the Store Unit

Table A.1: List of all PMU events supported on the M1 CPU extracted from the /usr/share/kpep/a14.list file.

Appendix B

Result Tables

Below are the tables containing the results generated from the benchmarks. Results for both the base and NEON ISAs, as well as for the performance and efficiency cores, are included. Additionally, the results from Johnson[3] used for the validation are included. The rows are color-coded accordingly:

- Green: Validation successful - M1 Pro and Johnson's results match
- Yellow: Validation partially successful - M1 Pro and Johnson's results match at least partially
- Red: Validation unsuccessful - M1 Pro and Johnson's results don't match
- Grey: Matching unsuccessful - could not find the instruction in our or Johnson's results.

B.1 Results of Benchmark base ISA

B.1.1 Performance Core

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
ADC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
ADCS	1/1	0.333/0.333	1/1	1/1	0.333/0.5	1/1	1/1	0.333/0.333	1/1
ADD	1/2	0.125/1	1/1	1/2	0.111/1	1/1	1/2	0.167/0.333	1/1
ADDS	1/2	0.333/0.667	1/1	1/2	0.2/0.5	1/1	1/2	0.333/0.667	1/1
ADR		0.25/0.25	1/1		0.2/0.2	1/1		0.5/0.5	1/1
ADRP								0.5/0.5	1/1
AND	1/2	0.167/0.5	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
ANDS	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
ASR	1/1	0.167/0.25	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
ASRV	1/1	0.167/0.2	1/1	1/1	0.125/0.25	1/1			
AUTDA	0.125/0.143	0.125/0.143	1/1	0.125/0.2	0.125/0.2	1/1	7/7	1/1	1/1
AUTDB	0.125/0.125	0.125/0.125	1/1	0.125/0.2	0.125/0.2	1/1	7/7	1/1	1/1
AUTDZA		0.125/0.125	1/1		0.125/0.125	1/1	7/7	1/1	1/1
AUTDZB		0.125/0.125	1/1		0.125/0.125	1/1	7/7	1/1	1/1
AUTIA	0.125/0.125	0.125/0.143	1/1	0.125/0.2	0.125/0.2	1/1	7/7	1/1	1/1
AUTIA1716		0.125/0.125	1/1		0.125/0.2	1/1			
AUTIASP		0.125/0.125	1/1		0.125/0.2	1/1			
AUTIAZ		0.125/0.125	1/1		0.125/0.2	1/1			
AUTIB	7/7	1/1	1/1				7/7	1/1	1/1
AUTIB1716		7/7	1/1						

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
AUTIBSP		7/7	1/1						
AUTIBZ		7/7	1/1						
AUTIZA							7/7	1/1	1/1
AUTIZB							7/7	1/1	1/1
AXFLAG		0.333/0.333	1/1		0.333/0.333	1/1	1/1	0.333/0.333	1/1
B		1/1	1/1		1/1	1/1		1/1	1/1
B.cc								0.5/1	1/1
BFC		1/1	1/1		1/1	1/1	1/1	1/1	1/1
BFI	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1
BFM	1/1	1/1	1/1	1/1	1/1	1/1			
BFXIL	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1
BIC	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
BICS	1/2	0.333/0.667	1/1	1/2	0.2/0.5	1/1	1/2	0.333/0.667	1/1
BL		1/1	1/1		1/1	1/1		1/1	1/1
BTI		0.125/0.125	1/1		0.2/0.2	1/1			
CAS		3/3	4/4		3/3	4/4		3/3	4/4
CASA		19/20	4/4		4/5	4/4		3/3.053	4/4
CASAB		19/19	4/4		4/5	4/4		3/3	4/4
CASAH		19/19	4/4		4/5	4/4		3/3	4/4
CASAL		21/22	4/5		3/5	4/4		7/7	4/4
CASALB		21/21	4/4		3/5	4/4		7/7	4/4
CASALH		21/21	4/4		3/5	4/4		7/7	4/4
CASB		3/3	4/4		3/3	4/4		3/3	4/4
CASH		3/3	4/4		3/3	4/4		3/3	4/4

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
CASL		7/7	4/4		3/3	4/4		7/7	4/4
CASLB		7/7	4/4		3/3	4/4		7/7	4/4
CASLH		7/7	4/4		3/3	4/4		7/7	4/4
CASP		17/17	6/6		16/17	6/6		14/14	6/6
CASPA		17/17	6/6		15/17	6/6		14/14	6/6
CASPAL		18/19	6/6		16/17	6/6		18/18	6/6
CASPL		18/18	6/6		15/17	6/6		18/18	6/6
CBNZ		0.5/1	1/1		0.333/1	1/1		0.5/1.01	1/1
CBZ		0.5/1	1/1		0.333/1	1/1		0.5/1	1/1
CCMN		0.333/1	1/1		0.25/1	1/1	1/1	0.333/0.333	1/1
CCMP		0.333/1	1/1		0.25/1	1/1	1/1	0.333/0.333	1/1
CFINV		1/1	1/1		1/1	1/1	1/1	0.333/0.333	1/1
CINC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CINV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CLRBHB		0.125/0.125	1/1						
CLREX		5/5	1/1		5/5	1/1		4.976/4.976	1/1
CLS	1/1	0.167/0.2	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
CLZ	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
CMN		0.333/0.667	1/1		0.25/0.5	1/1	1/2	0.333/0.667	1/1
CMP		0.333/0.667	1/1		0.25/0.5	1/1	1/2	0.333/0.667	1/1
CNEG	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CRC32B	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CB	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CH	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
CRC32CW	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CX	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32H	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32W	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32X	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CSDB		12/12	3/3		27/27	4/4		27/27	4/4
CSEL	1/1	0.167/0.333	1/1	1/1	0.125/0.25	1/1	1/1	0.333/0.333	1/1
CSET		0.333/0.333	1/1		0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSETM		0.333/0.333	1/1		0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSINC	1/1	0.167/0.333	1/1	1/1	0.125/0.25	1/1	1/1	0.333/0.333	1/1
CSINV	1/1	0.167/0.333	1/1	1/1	0.125/0.25	1/1	1/1	0.333/0.333	1/1
CSNEG	1/1	0.167/0.333	1/1	1/1	0.125/0.25	1/1	1/1	0.333/0.333	1/1
DGH		0.125/0.125	1/1		0.2/0.2	1/1			
DMB		3/3	1/1		1/1	1/1		2.904/2.913	1/1
DSB		17/27	1/4		17/27	1/4		17/17	1/1
EON	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
EOR	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
ESB		28/28	4/4		29/29	4/4			
EXTR	1/2	1/1	2/2	1/2	1/1	2/2	1/2	1/1	2/2
HINT		0.125/7	1/1		0.111/0.2	1/1			
ISB		28/28	4/4		31/32	5/5		28/28	4/4
LDADD		11/11	3/3		10/11	3/3			
LDADDA		20/20	3/3		16/18	3/3			
LDADDAB		20/20	3/3		16/18	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDADDAH		20/20	3/3		16/18	3/3			
LDADDAL		22/22	3/3		11/11	3/3			
LDADDALB		22/22	3/3		11/11	3/3			
LDADDALH		22/22	3/3		11/11	3/3			
LDADDB		11/11	3/3		10/11	3/3			
LDADDH		11/11	3/3		10/11	3/3			
LDADDL		18/22	3/3		10/11	3/3			
LDADDLB		20/22	3/3		10/11	3/3			
LDADDLH		20/22	3/3		10/11	3/3			
LDAPR		2/3	1/1		0.5/1	1/1			
LDAPRB		2/3	1/1		0.5/1	1/1			
LDAPRH		3/3	1/1		0.5/1	1/1			
LDAPUR		3/3	1/1		0.5/1	1/1			
LDAPURB		2/3	1/1		0.5/1	1/1			
LDAPURH		2/3	1/1		0.5/1	1/1			
LDAPURSB		2/3	1/1		0.5/1	1/1			
LDAPURSH		3/3	1/1		0.5/1	1/1			
LDAPURSW		2/3	1/1		0.5/1	1/1			
LDAR		2/3	1/1		0.5/1	1/1			
LDARB		3/3	1/1		0.5/1	1/1			
LDARH		3/3	1/1		0.5/1	1/1			
LDAXP		7/7	2/2		7/7	2/2			
LDAXR		7/7	1/1		7/7	1/1			
LDAXRB		7/7	1/1		7/7	1/1			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDAXRH		7/7	1/1		7/7	1/1			
LDCLR		11/11	3/3		10/11	3/3			
LDCLRA		20/20	3/3		16/18	3/3			
LDCLRAB		20/20	3/3		16/18	3/3			
LDCLRAH		20/20	3/3		16/18	3/3			
LDCLRAL		22/22	3/3		11/11	3/3			
LDCLRALB		22/22	3/3		11/11	3/3			
LDCLRALH		22/22	3/3		11/11	3/3			
LDCLRB		11/11	3/3		10/11	3/3			
LDCLRH		11/11	3/3		10/11	3/3			
LDCLRL		21/22	3/3		10/11	3/3			
LDCLRLB		19/22	3/3		10/11	3/3			
LDCLRLH		18/22	3/3		10/11	3/3			
LDEOR		11/11	3/3		10/11	3/3			
LDEORA		20/20	3/3		16/18	3/3			
LDEORAB		20/20	3/3		16/18	3/3			
LDEORAH		20/20	3/3		16/18	3/3			
LDEORAL		22/22	3/3		11/11	3/3			
LDEORALB		22/22	3/3		11/11	3/3			
LDEORALH		22/22	3/3		11/11	3/3			
LDEORB		11/11	3/3		10/11	3/3			
LDEORH		11/11	3/3		10/11	3/3			
LDEORL		20/22	3/3		10/11	3/3			
LDEORLB		21/22	3/3		10/11	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDEORLH		21/23	3/3		10/11	3/3			
LDLAR		2/3	1/1		0.5/1	1/1			
LDLARB		3/3	1/1		0.5/1	1/1			
LDLARH		2/3	1/1		0.5/1	1/1			
LDNP		0.333/0.5	2/2		0.333/0.5	2/2		0.333/0.333	2/2
LDP		0.333/1	2/3		0.333/1	2/3	4/4	0.333/0.407	2/3
LDPSW		0.333/1	2/3		0.333/1	2/3	4/4	0.333/0.396	2/3
LDR		0.333/1	1/2		0.333/1	1/2	4/4	0.333/0.368	1/2
LDRAA		0.333/1	2/3		0.333/1	2/3			
LDRAB		0.333/1	2/3		0.333/1	2/3			
LDRB		0.333/1	1/2		0.333/1	1/2	4/4	0.333/0.365	1/2
LDRH		0.333/1	1/2		0.333/1	1/2	4/4	0.333/0.368	1/2
LDRSB		0.333/1	1/2		0.333/1	1/2	4/4	0.333/0.365	1/2
LDRSH		0.333/1	1/2		0.333/1	1/2	4/4	0.333/0.368	1/2
LDRSW		0.333/1	1/2		0.333/1	1/2	4/4	0.333/0.367	1/2
LDSET		11/11	3/3		10/11	3/3			
LDSETA		20/20	3/3		16/18	3/3			
LDSETAB		20/20	3/3		16/18	3/3			
LDSETAH		20/20	3/3		16/18	3/3			
LDSETAL		22/22	3/3		11/11	3/3			
LDSETALB		22/22	3/3		11/11	3/3			
LDSETALH		22/22	3/3		11/11	3/3			
LDSETB		11/11	3/3		10/11	3/3			
LDSETH		11/11	3/3		10/11	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDSETL		22/22	3/3		10/11	3/3			
LDSETLB		22/22	3/3		10/11	3/3			
LDSETLH		22/22	3/3		10/11	3/3			
LDSMAX		19/19	2/2		18/19	2/2			
LDSMAXA		19/19	2/2		18/19	2/2			
LDSMAXAB		19/19	2/2		17/19	2/2			
LDSMAXAH		19/19	2/2		18/19	2/2			
LDSMAXAL		21/21	2/2		17/19	2/2			
LDSMAXALB		21/21	2/2		17/19	2/2			
LDSMAXALH		21/21	2/2		18/19	2/2			
LDSMAXB		19/19	2/2		18/19	2/2			
LDSMAXH		19/19	2/2		17/19	2/2			
LDSMAXL		21/21	2/2		18/19	2/2			
LDSMAXLB		21/21	2/2		17/19	2/2			
LDSMAXLH		21/21	2/2		18/19	2/2			
LDSMIN		19/19	2/2		18/19	2/2			
LDSMINA		19/19	2/2		18/19	2/2			
LDSMINAB		19/19	2/2		18/19	2/2			
LDSMINAH		19/19	2/2		18/19	2/2			
LDSMINAL		21/21	2/2		18/19	2/2			
LDSMINALB		21/21	2/2		18/19	2/2			
LDSMINALH		21/21	2/2		18/19	2/2			
LDSMINB		19/19	2/2		17/19	2/2			
LDSMINH		19/19	2/2		18/19	2/2			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDSMINL		21/21	2/2		17/19	2/2			
LDSMINLB		21/21	2/2		18/19	2/2			
LDSMINLH		21/21	2/2		18/19	2/2			
LDTR		0.333/0.333	1/1		0.333/0.5	1/1			
LDTRB		0.333/0.333	1/1		0.333/0.5	1/1			
LDTRH		0.333/0.333	1/1		0.333/0.5	1/1			
LDTRSB		0.333/0.333	1/1		0.333/0.5	1/1			
LDTRSH		0.333/0.333	1/1		0.333/0.5	1/1			
LDTRSW		0.333/0.333	1/1		0.333/0.333	1/1			
LDUMAX		19/19	2/2		18/19	2/2			
LDUMAXA		19/19	2/2		18/19	2/2			
LDUMAXAB		19/19	2/2		18/19	2/2			
LDUMAXAH		19/19	2/2		18/19	2/2			
LDUMAXAL		21/21	2/2		18/19	2/2			
LDUMAXALB		21/21	2/2		18/19	2/2			
LDUMAXALH		21/21	2/2		18/19	2/2			
LDUMAXB		19/19	2/2		18/19	2/2			
LDUMAXH		19/19	2/2		18/19	2/2			
LDUMAXL		21/21	2/2		17/19	2/2			
LDUMAXLB		21/21	2/2		18/19	2/2			
LDUMAXLH		21/21	2/2		18/19	2/2			
LDUMIN		19/19	2/2		18/19	2/2			
LDUMINA		19/19	2/2		18/19	2/2			
LDUMINAB		19/19	2/2		18/19	2/2			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDUMINAH		19/19	2/2		17/19	2/2			
LDUMINAL		21/21	2/2		18/19	2/2			
LDUMINALB		21/21	2/2		18/19	2/2			
LDUMINALH		21/21	2/2		17/19	2/2			
LDUMINB		19/19	2/2		18/19	2/2			
LDUMINH		19/19	2/2		18/19	2/2			
LDUMINL		21/21	2/2		18/19	2/2			
LDUMINLB		21/21	2/2		18/19	2/2			
LDUMINLH		21/21	2/2		18/19	2/2			
LDUR		0.333/0.333	1/1		0.333/0.5	1/1		0.333/0.333	1/1
LDURB		0.333/0.333	1/1		0.333/0.5	1/1		0.333/0.333	1/1
LDURH		0.333/0.333	1/1		0.333/0.5	1/1		0.333/0.333	1/1
LDURSB		0.333/0.333	1/1		0.333/0.5	1/1		0.333/0.333	1/1
LDURSH		0.333/0.333	1/1		0.333/0.5	1/1		0.333/0.333	1/1
LDURSW		0.333/0.333	1/1		0.333/0.5	1/1		0.333/0.333	1/1
LDXP		7/7	2/2		7/7	2/2			
LDXR		7/7	1/1		7/7	1/1			
LDXRB		7/7	1/1		7/7	1/1			
LDXRH		7/7	1/1		7/7	1/1			
LSL	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
LSLV	1/1	0.167/0.2	1/1	1/1	0.125/0.25	1/1			
LSR	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
LSRV	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1			
MADD	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
MNEG	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
MOV	0.125/0.125	0.125/1	1/1	0.111/0.2	0.111/1	1/1		0.125/0.167	1/1
MOVK		0.167/0.167	1/1		0.125/0.25	1/1	1/1	0.167/0.167	1/1
MOVN		0.125/0.125	1/1		0.111/0.2	1/1		0.125/0.125	1/1
MOVZ		0.125/0.125	1/1		0.111/0.2	1/1		0.125/0.125	1/1
MRS								0.5/8.501	1/1
MSR								0.5/12	1/4
MSUB	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
MUL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
MVN	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
NEG	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
NEGS	1/2	0.333/0.667	1/1	1/2	0.2/0.5	1/1	1/2	0.333/0.667	1/1
NGC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
NGCS	1/1	0.333/0.333	1/1	1/1	0.333/0.5	1/1	1/1	0.333/0.333	1/1
NOP		0.125/0.125	1/1		0.2/0.2	1/1		0.125/0.125	1/1
ORN	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
ORR	1/2	0.167/0.333	1/1	1/2	0.125/0.5	1/1	1/2	0.167/0.333	1/1
PACDA	0.125/0.125	0.125/0.125	1/1	0.111/0.2	0.111/0.111	1/1	7/7	1/1	1/1
PACDB	0.125/0.125	0.125/0.125	1/1	0.111/0.2	0.111/0.2	1/1	7/7	1/1	1/1
PACDZA		0.125/0.125	1/1		0.111/0.111	1/1	7/7	1/1	1/1
PACDZB		0.125/0.125	1/1		0.2/0.2	1/1	7/7	1/1	1/1
PACGA	7/7	1/1	1/1	7/7	1/1	1/1	7/7	1/1	1/1
PACIA	0.125/0.25	0.125/0.25	1/1	0.111/0.2	0.111/0.2	1/1	7/7	1/1	1/1
PACIA1716		0.125/0.125	1/1		0.111/0.111	1/1			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
PACIASP		0.125/0.125	1/1		0.111/0.2	1/1			
PACIAZ		0.125/0.125	1/1		0.111/0.2	1/1			
PACIB	7/7	1/1	1/1	7/7	1/1	1/1	7/7	1/1	1/1
PACIB1716		7/7	1/1		6/7	1/1			
PACIBSP		7/7	1/1		7/7	1/1			
PACIBZ		7/7	1/1		7/7	1/1			
PACIZA							7/7	1/1	1/1
PACIZB							7/7	1/1	1/1
PRFM								1.353/1.553	1/1
PSSBB		27/27	4/4		27/27	4/4		27/27	4/4
RBIT	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
REV	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
REV16	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
REV32	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
REV64	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1			
RMIF		0.333/0.333	1/1		0.333/0.333	1/1	1/1	0.333/0.333	1/1
ROR	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
RORV	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1			
SB		27/27	4/4		27/27	4/4		27/27	4/4
SBC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
SBCS	1/1	0.333/0.333	1/1	1/1	0.333/0.5	1/1	1/1	0.333/0.333	1/1
SBFIZ	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
SBFM	1/1	0.167/0.25	1/1	1/1	0.125/0.25	1/1			
SBFX	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SDIV	7/9	2/2	1/1	7/9	2/2	1/1	7/9	2/2	1/1
SETF16		1/1	1/1		1/1	1/1	1/1	0.333/0.333	1/1
SETF8		1/1	1/1		1/1	1/1	1/1	0.333/0.333	1/1
SEV		26/26	1/1		10/10	1/1			
SEVL		1/1	1/1		1/1	1/1			
SMADDL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
SMNEGL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SMSUBL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
SMULH	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SMULL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SSBB		27/27	4/4		27/27	4/4		27/27	4/4
STADD		11/11	3/3		10/11	3/3		3/3	3/3
STADDB		11/11	3/3		10/11	3/3		3/3	3/3
STADDH		11/11	3/3		10/11	3/3		3/3	3/3
STADDL		20/22	3/3		10/11	3/3		7/7	3/3
STADDLB		22/22	3/3		10/11	3/3		7/7	3/3
STADDLH		20/22	3/3		10/11	3/3		7/7	3/3
STCLR		11/11	3/3		10/11	3/3		3/3	3/3
STCLRB		11/11	3/3		10/11	3/3		3/3	3/3
STCLRH		11/11	3/3		10/11	3/3		3/3	3/3
STCLRL		21/22	3/3		10/11	3/3		7/7	3/3
STCLRLB		22/22	3/3		10/11	3/3		7/7	3/3
STCLRLH		21/22	3/3		10/11	3/3		7/7	3/3
STEOR		11/11	3/3		10/11	3/3		3/3	3/3

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
STEORB		11/11	3/3		10/11	3/3		3/3	3/3
STEORH		11/11	3/3		10/11	3/3		3/3	3/3
STEORL		22/22	3/3		10/11	3/3		7/7	3/3
STEORLB		22/22	3/3		10/11	3/3		7/7	3/3
STEORLH		21/22	3/3		10/11	3/3		7/7	3/3
STLLR		1/1	1/1		1/1	1/1			
STLLRB		1/1	1/1		1/1	1/1		1/1	1/1
STLLRH		1/1	1/1		1/1	1/1		1/1	1/1
STLR		1/1	1/1		1/1	1/1			
STLRB		1/1	1/1		1/1	1/1		1/1	1/1
STLRH		1/1	1/1		1/1	1/1		1/1	1/1
STLUR		1/1	1/1		1/1	1/1			
STLURB		1/1	1/1		1/1	1/1			
STLURH		1/1	1/1		1/1	1/1			
STLXP		20/38	1/1		3/43	1/1		38/38	1/1
STLXR		20/38	1/1		3/43	1/1		38/38	1/1
STLXRB		31/38	1/1		3/43	1/1		38/38	1/1
STLXRH		24/38	1/1		3/43	1/1		38/38	1/1
STNP		0.5/0.5	1/1		0.5/0.667	1/1		0.521/0.521	1/1
STP		0.5/1	1/1		0.5/1	1/1		0.5/0.524	1/1
STR		0.5/1	1/1		0.5/1	1/1		0.5/0.51	1/1
STRB		0.5/1	1/1		0.5/1	1/1		0.5/0.509	1/1
STRH		0.5/1	1/1		0.5/1	1/1		0.5/0.51	1/1
STSET		11/11	3/3		10/11	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
STSETB		11/11	3/3		10/11	3/3			
STSETH		11/11	3/3		10/11	3/3			
STSETL		21/22	3/3		10/11	3/3			
STSETLB		22/22	3/3		10/11	3/3			
STSETLH		21/22	3/3		10/11	3/3			
STSMAX		19/19	2/2		18/19	2/2			
STSMAXB		19/19	2/2		18/19	2/2			
STSMAXH		19/19	2/2		18/19	2/2			
STSMAXL		21/21	2/2		18/19	2/2			
STSMAXLB		21/21	2/2		18/19	2/2			
STSMAXLH		21/21	2/2		18/19	2/2			
STSMIN		19/19	2/2		18/19	2/2			
STSMINB		19/19	2/2		18/19	2/2			
STSMINH		19/19	2/2		18/19	2/2			
STSMINL		21/21	2/2		18/19	2/2			
STSMINLB		21/21	2/2		18/19	2/2			
STSMINLH		21/21	2/2		18/19	2/2			
STTR		0.5/0.5	1/1		0.5/0.667	1/1			
STTRB		0.5/0.5	1/1		0.5/0.667	1/1			
STTRH		0.5/0.5	1/1		0.5/0.5	1/1			
STUMAX		19/19	2/2		18/19	2/2			
STUMAXB		19/19	2/2		18/19	2/2			
STUMAXH		19/19	2/2		18/19	2/2			
STUMAXL		21/21	2/2		18/19	2/2			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
STUMAXLB		21/21	2/2		18/19	2/2			
STUMAXLH		21/21	2/2		18/19	2/2			
STUMIN		19/19	2/2		18/19	2/2			
STUMINB		19/19	2/2		18/19	2/2			
STUMINH		19/19	2/2		18/19	2/2			
STUMINL		21/21	2/2		17/19	2/2			
STUMINLB		21/21	2/2		18/19	2/2			
STUMINLH		21/21	2/2		18/19	2/2			
STUR		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
STURB		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
STURH		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
STXP		24/38	1/1		3/43	1/1		38/38	1/1
STXR		5/38	1/1		2/43	1/1		38/38	1/1
STXRB		7/38	1/1		2/43	1/1		38/38	1/1
STXRH		27/38	1/1		2/43	1/1		38/38	1/1
SUB	1/2	0.167/1	1/1	1/2	0.125/1	1/1	1/2	0.167/0.333	1/1
SUBS	1/2	0.333/0.667	1/1	1/2	0.2/0.5	1/1	1/2	0.333/0.667	1/1
SVC					50/50	50/50			
SWP		3/3	2/2		2/3	2/2		3/3	2/2
SWPA		13/13	2/2		3/3	2/2		3/3.047	2/2
SWPAB		13/13	2/2		3/3	2/2		3/3	2/2
SWPAH		13/13	2/2		3/3	2/2		3/3	2/2
SWPAL		18/19	2/3		2/3	2/2		7/7	2/2
SWPALB		18/18	2/2		2/3	2/2		7/7	2/2

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SWPALH		18/18	2/2		2/3	2/2		7/7	2/2
SWPB		3/3	2/2		2/3	2/2		3/3	2/2
SWPH		3/3	2/2		2/3	2/2		3/3	2/2
SWPL		18/18	2/2		2/3	2/2		7/7	2/2
SWPLB		14/18	2/2		2/3	2/2		7/7	2/2
SWPLH		18/18	2/2		2/3	2/2		7/7	2/2
SXTB	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
SXTH	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
SXTW	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
TBNZ		0.5/1	1/1		0.333/1	1/1		0.5/1	1/1
TBZ		0.5/1	1/1		0.333/1	1/1		0.5/1	1/1
TST		0.333/0.667	1/1		0.25/0.5	1/1	1/2	0.333/0.667	1/1
UBFIZ	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
UBFM	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1			
UBFX	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
UDIV	7/9	2/2	1/1	7/9	2/2	1/1	7/9	2/2	1/1
UMADDL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
UMNEGL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
UMSUBL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
UMULH	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
UMULL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
UXTB	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
UXTH	1/1	0.167/0.167	1/1	1/1	0.125/0.25	1/1	1/1	0.167/0.167	1/1
XAFLAG		0.333/0.333	1/1		0.333/0.333	1/1	1/1	0.333/0.333	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
XPACD		1/7	1/1		1/7	1/1	7/7	1/1	1/1
XPACI		1/7	1/1		1/7	1/1	7/7	1/1	1/1
XPACLRI		7/7	1/1		7/7	1/1			
YIELD		0.125/0.125	1/1		0.2/0.2	1/1		0.129/0.129	1/1

B.1.2 Efficiency Core

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
ADC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
ADCS	1/1	0.667/0.667	1/1	1/1	0.5/0.5	1/1	1/1	0.584/0.584	1/1
ADD	1/2	0.25/1	1/1	1/2	0.2/1	1/1	1/2	0.333/0.667	1/1
ADDS	1/2	0.5/0.667	1/1	1/2	0.5/0.5	1/1	1/2	0.5/0.667	1/1
ADR		0.25/0.25	1/1		0.2/0.2	1/1		0.25/0.25	1/1
ADRP								0.25/0.25	1/1
AND	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
ANDS	1/2	0.5/0.667	1/1	1/2	0.5/0.5	1/1	1/2	0.5/0.667	1/1
ASR	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
ASRV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
AUTDA	0.25/0.25	0.25/0.25	1/1	0.2/0.2	0.2/0.2	1/1	6/6	2/2	1/1
AUTDB	0.25/0.25	0.25/0.25	1/1	0.2/0.2	0.2/0.2	1/1	6/6	2/2	1/1
AUTDZA		0.25/0.25	1/1		0.2/0.2	1/1	6/6	2/2	1/1
AUTDZB		0.25/0.25	1/1		0.2/0.2	1/1	6/6	2/2	1/1
AUTIA	0.25/0.25	0.25/0.25	1/1	0.2/0.2	0.2/0.2	1/1	6/6	2/2	1/1
AUTIA1716		0.25/0.25	1/1		0.2/0.2	1/1			
AUTIASP		0.25/0.25	1/1		0.2/0.2	1/1			
AUTIAZ		0.25/0.25	1/1		0.2/0.2	1/1			
AUTIB	6/6	2/2	1/1				6/6	2/2	1/1
AUTIB1716		6/6	1/1						
AUTIBSP		6/6	1/1						
AUTIBZ		6/6	1/1						

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
AUTIZA							6/6	2/2	1/1
AUTIZB							6/6	2/2	1/1
AXFLAG		0.5/0.5	1/1		0.333/0.333	1/1	1/1	0.556/0.556	1/1
B		1/1	1/1		1/1	1/1		1.049/1.049	1/1
B.cc								0.584/1.154	1/1
BFC		1/1	1/1		1/1	1/1	1/1	1/1	1/1
BFI	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1
BFM	1/1	1/1	1/1	1/1	1/1	1/1			
BFXIL	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1
BIC	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
BICS	1/2	0.5/0.667	1/1	1/2	0.5/0.5	1/1	1/2	0.5/0.667	1/1
BL		1/1	1/1		1/1	1/1		1.049/1.049	1/1
BTI		0.25/0.25	1/1		0.2/0.2	1/1			
CAS		11/11	4/5		3/3	4/4		7/7	4/4
CASA		22/23	4/5		14/14	4/4		7/7	4/4
CASAB		22/23	4/4		14/14	4/4		7/7	4/4
CASAH		22/22	4/4		14/14	4/4		7/7	4/4
CASAL		24/25	4/5		3/3	4/4		9/9	4/4
CASALB		24/24	4/4		3/3	4/4		9/9	4/4
CASALH		24/25	4/4		3/3	4/4		9/9	4/4
CASB		11/11	4/5		3/3	4/4		7/7	4/4
CASH		11/11	4/4		3/3	4/4		7/7	4/4
CASL		12/12	4/5		3/3	4/4		9/9	4/4
CASLB		12/12	4/4		3/3	4/4		9/9	4/4

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
CASLH		12/12	4/4		3/3	4/4		9/9	4/4
CASP		17/18	7/8		13/13	7/7		17/17	6/6
CASPA		15/16	6/6		12/12	6/6		15/15	6/6
CASPAL		30/32	6/7		13/13	7/7		19/19	6/6
CASPL		19/20	7/7		13/13	7/7		19/19	6/6
CBNZ		0.5/1	1/1		0.5/1	1/1		0.584/1.062	1/1
CBZ		0.5/1	1/1		0.5/1	1/1		0.584/1.062	1/1
CCMN		0.333/1	1/1		0.333/1	1/1	1/1	0.556/0.556	1/1
CCMP		0.333/1	1/1		0.333/1	1/1	1/1	0.556/0.556	1/1
CFINV		1/1	1/1		1/1	1/1	1/1	0.556/0.556	1/1
CINC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CINV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CLRBHB		0.25/0.25	1/1						
CLREX		5/5	1/1		5/5	1/1		4.996/4.996	1/1
CLS	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CLZ	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CMN		0.333/0.667	1/1		0.333/0.5	1/1	1/2	0.363/0.667	1/1
CMP		0.333/0.667	1/1		0.333/0.5	1/1	1/2	0.362/0.667	1/1
CNEG	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CRC32B	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CB	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CH	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CW	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32CX	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
CRC32H	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32W	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CRC32X	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
CSDB		0.25/0.25	1/1		23/23	4/4		0.254/0.254	1/1
CSEL	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSET		0.333/0.333	1/1		0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSETM		0.333/0.333	1/1		0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSINC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSINV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
CSNEG	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
DGH		0.25/0.25	1/1		0.2/0.2	1/1			
DMB		4/4	1/1		1/1	1/1		4/4	1/1
DSB		16/22	1/4		16/23	1/4		16/16	1/1
EON	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
EOR	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
ESB		25/25	4/4		25/25	4/4			
EXTR	1/2	1/1	2/2	1/2	1/1	2/2	1/2	1/1	2/2
HINT		0.25/6	1/1		0.2/0.2	1/1			
ISB		25/25	4/4		25/25	4/4		25/25	4/4
LDADD		13/14	3/3		9/9	3/3			
LDADDA		23/23	3/3		15/15	3/3			
LDADDAB		23/23	3/3		15/15	3/3			
LDADDAH		23/23	3/3		15/15	3/3			
LDADDAL		25/26	3/3		10/10	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDADDALB		25/25	3/3		10/10	3/3			
LDADDALH		25/26	3/3		10/10	3/3			
LDADDB		13/13	3/3		9/9	3/3			
LDADDH		13/13	3/3		9/9	3/3			
LDADDL		11/12	3/3		9/9	3/3			
LDADDLB		11/12	3/3		9/9	3/3			
LDADDLH		11/12	3/3		9/9	3/3			
LDAPR		2/2	1/1		0.5/0.5	1/1			
LDAPRB		2/2	1/1		0.5/0.5	1/1			
LDAPRH		2/2	1/1		0.5/0.5	1/1			
LDAPUR		2/2	1/1		0.5/0.5	1/1			
LDAPURB		2/2	1/1		0.5/0.5	1/1			
LDAPURH		2/2	1/1		0.5/0.5	1/1			
LDAPURSB		2/2	1/1		0.5/0.5	1/1			
LDAPURSH		2/3	1/1		0.5/0.5	1/1			
LDAPURSW		2/2	1/1		0.5/0.5	1/1			
LDAR		2/3	1/1		0.5/0.5	1/1			
LDARB		2/2	1/1		0.5/0.5	1/1			
LDARH		2/2	1/1		0.5/0.5	1/1			
LDAXP		13/13	2/2		13/13	2/2			
LDAXR		13/13	1/1		13/13	1/1			
LDAXRB		13/13	1/1		13/13	1/1			
LDAXRH		13/13	1/1		13/13	1/1			
LDCLR		13/13	3/3		8/9	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDCLRA		23/24	3/3		15/15	3/3			
LDCLRAB		23/23	3/3		15/15	3/3			
LDCLRAH		23/23	3/3		15/15	3/3			
LDCLRAL		25/25	3/3		10/10	3/3			
LDCLRALB		25/26	3/3		10/10	3/3			
LDCLRALH		25/25	3/3		10/10	3/3			
LDCLRB		13/13	3/3		9/9	3/3			
LDCLRH		13/13	3/3		8/9	3/3			
LDCLRL		11/12	3/3		9/9	3/3			
LDCLRLB		11/12	3/3		9/9	3/3			
LDCLRLH		11/11	3/3		9/9	3/3			
LDEOR		13/13	3/3		8/9	3/3			
LDEORA		23/23	3/3		15/15	3/3			
LDEORAB		23/23	3/3		15/15	3/3			
LDEORAH		23/24	3/3		15/15	3/3			
LDEORAL		25/25	3/3		10/10	3/3			
LDEORALB		25/26	3/4		10/10	3/3			
LDEORALH		25/26	3/3		10/10	3/3			
LDEORB		13/13	3/3		9/9	3/3			
LDEORH		13/13	3/3		8/9	3/3			
LDEORL		11/12	3/3		9/9	3/3			
LDEORLB		11/12	3/3		9/9	3/3			
LDEORLH		11/12	3/3		9/9	3/3			
LDLAR		2/3	1/1		0.5/0.5	1/1			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDLARB		2/3	1/1		0.5/0.5	1/1			
LDLARH		2/3	1/1		0.5/0.5	1/1			
LDNP		0.5/0.5	2/2		0.5/0.5	2/2		0.5/0.5	2/2
LDP		0.5/1	2/3		0.5/1	2/3	4/4	0.5/0.763	2/3
LDPSW		0.5/1	2/3		0.5/1	2/3	4/4	0.5/0.75	2/3
LDR		0.5/1	1/2		0.5/1	1/2	4/4	0.5/0.54	1/2
LDRAA		0.5/1	2/3		0.5/1	2/3			
LDRAB		0.5/1	2/3		0.5/1	2/3			
LDRB		0.5/1	1/2		0.5/1	1/2	4/4	0.5/0.54	1/2
LDRH		0.5/1	1/2		0.5/1	1/2	4/4	0.5/0.54	1/2
LDRSB		0.5/1	1/2		0.5/1	1/2	4/4	0.5/0.54	1/2
LDRSH		0.5/1	1/2		0.5/1	1/2	4/4	0.5/0.541	1/2
LDRSW		0.5/1	1/2		0.5/1	1/2	4/4	0.5/0.54	1/2
LDSET		13/14	3/3		9/9	3/3			
LDSETA		23/25	3/4		15/15	3/3			
LDSETAB		23/24	3/4		15/15	3/3			
LDSETAH		23/24	3/4		15/15	3/3			
LDSETAL		25/26	3/4		10/10	3/3			
LDSETALB		25/27	3/4		10/10	3/3			
LDSETALH		25/27	3/4		10/10	3/3			
LDSETB		13/13	3/3		9/9	3/3			
LDSETH		13/13	3/3		9/9	3/3			
LDSETL		11/12	3/3		9/9	3/3			
LDSETLB		11/12	3/3		9/9	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDSETLH		11/12	3/3		9/9	3/3			
LDSMAX		22/23	2/3		14/14	2/2			
LDSMAXA		22/23	2/3		14/14	2/2			
LDSMAXAB		22/23	2/2		14/14	2/2			
LDSMAXAH		22/22	2/2		14/14	2/2			
LDSMAXAL		24/26	2/3		14/14	2/2			
LDSMAXALB		24/24	2/2		14/14	2/2			
LDSMAXALH		24/25	2/2		14/14	2/2			
LDSMAXB		22/23	2/3		14/14	2/2			
LDSMAXH		22/23	2/2		14/14	2/2			
LDSMAXL		24/25	2/2		14/14	2/2			
LDSMAXLB		24/25	2/2		14/14	2/2			
LDSMAXLH		24/24	2/2		14/14	2/2			
LDSMIN		22/23	2/2		14/14	2/2			
LDSMINA		22/23	2/2		14/14	2/2			
LDSMINAB		22/23	2/2		14/14	2/2			
LDSMINAH		22/22	2/2		14/14	2/2			
LDSMINAL		24/25	2/3		14/14	2/2			
LDSMINALB		24/24	2/2		14/14	2/2			
LDSMINALH		24/24	2/2		14/14	2/2			
LDSMINB		22/22	2/2		14/14	2/2			
LDSMINH		22/23	2/3		14/14	2/2			
LDSMINL		24/24	2/2		14/14	2/2			
LDSMINLB		24/25	2/2		14/14	2/2			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDSMINLH		24/25	2/2		14/14	2/2			
LDTR		0.5/0.5	1/1		0.5/0.5	1/1			
LDTRB		0.5/0.5	1/1		0.5/0.5	1/1			
LDTRH		0.5/0.5	1/1		0.5/0.5	1/1			
LDTRSB		0.5/0.5	1/1		0.5/0.5	1/1			
LDTRSH		0.5/0.5	1/1		0.5/0.5	1/1			
LDTRSW		0.5/0.5	1/1		0.5/0.5	1/1			
LDUMAX		22/24	2/3		14/14	2/2			
LDUMAXA		22/23	2/2		14/14	2/2			
LDUMAXAB		22/22	2/2		14/14	2/2			
LDUMAXAH		22/22	2/2		14/14	2/2			
LDUMAXAL		24/26	2/3		14/14	2/2			
LDUMAXALB		24/25	2/2		14/14	2/2			
LDUMAXALH		24/25	2/2		14/14	2/2			
LDUMAXB		22/22	2/2		14/14	2/2			
LDUMAXH		22/23	2/2		14/14	2/2			
LDUMAXL		24/25	2/2		14/14	2/2			
LDUMAXLB		24/25	2/2		14/14	2/2			
LDUMAXLH		24/25	2/2		14/14	2/2			
LDUMIN		22/23	2/2		14/14	2/2			
LDUMINA		22/22	2/2		14/14	2/2			
LDUMINAB		22/24	2/3		14/14	2/2			
LDUMINAH		22/22	2/2		14/14	2/2			
LDUMINAL		24/26	2/3		14/14	2/2			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LDUMINALB		24/26	2/3		14/14	2/2			
LDUMINALH		24/24	2/2		14/14	2/2			
LDUMINB		22/22	2/2		14/14	2/2			
LDUMINH		22/24	2/3		14/14	2/2			
LDUMINL		24/25	2/2		14/14	2/2			
LDUMINLB		24/25	2/3		14/14	2/2			
LDUMINLH		24/25	2/2		14/14	2/2			
LDUR		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
LDURB		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
LDURH		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
LDURSB		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
LDURSH		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
LDURSW		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
LDXP		13/13	2/2		13/13	2/2			
LDXR		13/13	1/1		13/13	1/1			
LDXRB		13/13	1/1		13/13	1/1			
LDXRH		13/13	1/1		13/13	1/1			
LSL	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
LSLV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
LSR	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
LSRV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
MADD	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
MNEG	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
MOV	0.25/0.25	0.25/1	1/1	0.2/0.2	0.2/1	1/1		0.25/0.333	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
MOVK		0.333/0.333	1/1		0.25/0.25	1/1	1/1	0.333/0.333	1/1
MOVN		0.25/0.25	1/1		0.2/0.2	1/1		0.25/0.25	1/1
MOVZ		0.25/0.25	1/1		0.2/0.2	1/1		0.25/0.25	1/1
MRS								0.333/16	1/1
MSR								0.366/27	1/4
MSUB	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
MUL	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
MVN	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
NEG	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
NEGS	1/2	0.5/0.667	1/1	1/2	0.5/0.5	1/1	1/2	0.5/0.667	1/1
NGC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
NGCS	1/1	0.667/0.667	1/1	1/1	0.5/0.5	1/1	1/1	0.584/0.584	1/1
NOP		0.25/0.25	1/1		0.2/0.2	1/1		0.25/0.25	1/1
ORN	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
ORR	1/2	0.333/0.667	1/1	1/2	0.25/0.5	1/1	1/2	0.333/0.667	1/1
PACDA	0.25/0.25	0.25/0.25	1/1	0.2/0.2	0.2/0.2	1/1	6/6	2/2	1/1
PACDB	0.25/0.25	0.25/0.25	1/1	0.2/0.2	0.2/0.2	1/1	6/6	2/2	1/1
PACDZA		0.25/0.25	1/1		0.2/0.2	1/1	6/6	2/2	1/1
PACDZB		0.25/0.25	1/1		0.2/0.2	1/1	6/6	2/2	1/1
PACGA	6/6	2/2	1/1	6/6	1/1	1/1	6/6	2/2	1/1
PACIA	0.25/0.25	0.25/0.25	1/1	0.2/0.2	0.2/0.2	1/1	6/6	2/2	1/1
PACIA1716		0.25/0.25	1/1		0.2/0.2	1/1			
PACIASP		0.25/0.25	1/1		0.2/0.2	1/1			
PACIAZ		0.25/0.333	1/1		0.2/0.2	1/1			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
PACIB	6/6	2/2	1/1	6/6	1/1	1/1	6/6	2/2	1/1
PACIB1716		6/6	1/1		6/6	1/1			
PACIBSP		6/6	1/1		6/6	1/1			
PACIBZ		6/6	1/1		6/6	1/1			
PACIZA							6/6	2/2	1/1
PACIZB							6/6	2/2	1/1
PRFM								1.872/2	1/1
PSSBB		22/22	4/4		23/23	4/4		22/22	4/4
RBIT	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
REV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
REV16	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
REV32	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
REV64	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
RMIF		0.5/0.5	1/1		0.333/0.333	1/1	1/1	0.556/0.556	1/1
ROR	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
RORV	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
SB		22/22	4/4		23/23	4/4		22/22	4/4
SBC	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
SBCS	1/1	0.667/0.667	1/1	1/1	0.5/0.5	1/1	1/1	0.584/0.584	1/1
SBFIZ	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
SBFM	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
SBFX	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
SDIV	7/15	7/21	1/1	7/15	7/21	1/1	7/21	7/21	1/1
SETF16		1/1	1/1		1/1	1/1	1/1	0.556/0.556	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SETF8		1/1	1/1		1/1	1/1	1/1	0.556/0.556	1/1
SEV		9/9	1/1		10/10	1/1			
SEVL		1/1	1/1		1/1	1/1			
SMADDL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
SMNEGL	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
SMSUBL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
SMULH	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
SMULL	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
SSBB		22/22	4/4		23/23	4/4		22/22	4/4
STADD		13/13	3/3		9/9	3/3		3/3	3/3
STADDB		13/14	3/3		9/9	3/3		3/3	3/3
STADDH		13/14	3/3		9/9	3/3		3/3	3/3
STADDL		11/12	3/3		9/9	3/3		6/6	3/3
STADDLB		11/12	3/3		9/9	3/3		6/6	3/3
STADDLH		11/12	3/3		9/9	3/3		6/6	3/3
STCLR		13/13	3/3		9/9	3/3		3/3	3/3
STCLRB		13/14	3/3		9/9	3/3		3/3	3/3
STCLRH		13/14	3/3		9/9	3/3		3/3	3/3
STCLRL		11/12	3/3		9/9	3/3		6/6	3/3
STCLRLB		11/12	3/3		9/9	3/3		6/6	3/3
STCLRLH		11/12	3/3		9/9	3/3		6/6	3/3
STEOR		13/14	3/3		9/9	3/3		3/3	3/3
STEORB		13/14	3/3		9/9	3/3		3/3	3/3
STEORH		13/13	3/3		9/9	3/3		3/3	3/3

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
STEORL		11/12	3/3		9/9	3/3		6/6	3/3
STEORLB		11/12	3/3		9/9	3/3		6/6	3/3
STEORLH		11/12	3/3		9/9	3/3		6/6	3/3
STLLR		6/6	1/1		1/1	1/1			
STLLRB		6/6	1/1		1/1	1/1		6/6	1/1
STLLRH		6/6	1/1		1/1	1/1		6/6	1/1
STLR		6/7	1/1		1/1	1/1			
STLRB		6/6	1/1		1/1	1/1		6/6	1/1
STLRH		6/6	1/1		1/1	1/1		6/6	1/1
STLUR		6/6	1/1		1/1	1/1			
STLURB		6/6	1/1		1/1	1/1			
STLURH		6/6	1/1		1/1	1/1			
STLXP		3/3	1/1		3/3	1/1		3/3	1/1
STLXR		3/3	1/1		3/3	1/1		3/3	1/1
STLXRB		3/3	1/1		3/3	1/1		3/3	1/1
STLXRH		3/3	1/1		3/3	1/1		3/3	1/1
STNP		1/1	1/1		1/1	1/1		1/1	1/1
STP		1/1	1/1		1/1	1/1		1/1.014	1/1
STR		1/1	1/1		1/1	1/1		1/1	1/1
STRB		1/1	1/1		1/1	1/1		1/1	1/1
STRH		1/1	1/1		1/1	1/1		1/1	1/1
STSET		13/14	3/3		9/9	3/3			
STSETB		13/14	3/3		9/9	3/3			
STSETH		13/14	3/3		9/9	3/3			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
STSETL		11/12	3/3		9/9	3/3			
STSETLB		11/12	3/3		9/9	3/3			
STSETLH		11/12	3/3		9/9	3/3			
STSMAX		22/23	2/3		14/14	2/2			
STSMAXB		22/23	2/3		14/14	2/2			
STSMAXH		22/22	2/2		14/14	2/2			
STSMAXL		24/24	2/2		14/14	2/2			
STSMAXLB		24/25	2/3		14/14	2/2			
STSMAXLH		24/25	2/3		14/14	2/2			
STSMIN		22/23	2/3		14/14	2/2			
STSMINB		22/23	2/3		14/14	2/2			
STSMINH		22/22	2/2		14/14	2/2			
STSMINL		24/25	2/3		14/14	2/2			
STSMINLB		24/25	2/2		14/14	2/2			
STSMINLH		24/24	2/2		14/14	2/2			
STTR		1/1	1/1		1/1	1/1			
STTRB		1/1	1/1		1/1	1/1			
STTRH		1/1	1/1		1/1	1/1			
STUMAX		22/22	2/2		14/14	2/2			
STUMAXB		22/24	2/3		14/14	2/2			
STUMAXH		22/22	2/2		14/14	2/2			
STUMAXL		24/24	2/2		14/14	2/2			
STUMAXLB		24/24	2/2		14/14	2/2			
STUMAXLH		24/24	2/2		14/14	2/2			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
STUMIN		22/22	2/2		14/14	2/2			
STUMINB		22/22	2/2		14/14	2/2			
STUMINH		22/23	2/2		14/14	2/2			
STUMINL		24/24	2/2		14/14	2/2			
STUMINLB		24/25	2/2		14/14	2/2			
STUMINLH		24/25	2/3		14/14	2/2			
STUR		1/1	1/1		1/1	1/1		1/1	1/1
STURB		1/1	1/1		1/1	1/1		1/1	1/1
STURH		1/1	1/1		1/1	1/1		1/1	1/1
STXP		2/3	1/1		2/3	1/1		2.509/2.509	1/1
STXR		2/3	1/1		2/3	1/1		2.186/2.34	1/1
STXRB		2/3	1/1		2/3	1/1		2.107/2.107	1/1
STXRH		2/3	1/1		2/3	1/1		2.107/2.107	1/1
SUB	1/2	0.333/1	1/1	1/2	0.25/1	1/1	1/2	0.333/0.667	1/1
SUBS	1/2	0.5/0.667	1/1	1/2	0.5/0.5	1/1	1/2	0.5/0.667	1/1
SWP		10/11	2/2		7/7	2/2		3/3	2/2
SWPA		22/23	2/2		14/14	2/2		6/6	2/2
SWPAB		22/23	2/2		14/14	2/2		6/6	2/2
SWPAH		22/22	2/2		14/14	2/2		6/6	2/2
SWPAL		24/25	2/3		9/9	2/2		6/6	2/2
SWPALB		24/24	2/2		9/9	2/2		6/6	2/2
SWPALH		24/24	2/2		9/9	2/2		6/6	2/2
SWPB		10/10	2/2		7/7	2/2		3/3	2/2
SWPH		10/10	2/2		7/7	2/2		3/3	2/2

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SWPL		10/10	2/2		7/7	2/2		6/6	2/2
SWPLB		10/10	2/2		7/7	2/2		6/6	2/2
SWPLH		10/10	2/2		7/7	2/2		6/6	2/2
SXTB	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
SXTH	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
SXTW	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
TBNZ		0.5/1	1/1		0.5/1	1/1		0.584/1.062	1/1
TBZ		0.5/1	1/1		0.5/1	1/1		0.584/1.061	1/1
TST		0.333/0.667	1/1		0.333/0.5	1/1	1/2	0.363/0.667	1/1
UBFIZ	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
UBFM	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1			
UBFX	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
UDIV	7/15	7/21	1/1	7/15	7/21	1/1	7/21	7/21	1/1
UMADDL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
UMNEGL	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
UMSUBL	1/3	1/1	1/1	1/3	1/1	1/1	1/3	1/1	1/1
UMULH	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
UMULL	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
UXTB	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
UXTH	1/1	0.333/0.333	1/1	1/1	0.25/0.25	1/1	1/1	0.333/0.333	1/1
XAFLAG		0.5/0.5	1/1		0.333/0.333	1/1	1/1	0.555/0.555	1/1
XPACD		2/6	1/1		1/6	1/1	6/6	2/2	1/1
XPACI		2/6	1/1		1/6	1/1	6/6	2/2	1/1
XPACLRI		6/6	1/1		6/6	1/1			

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
YIELD		0.25/0.25	1/1		0.2/0.2	1/1		0.254/0.254	1/1

B.2 Results of Benchmark SIMD&FP ISA

B.2.1 Performance Core

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
ABS	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
ADD	2/2	0.25/0.333	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
ADDHN							3/3	0.25/0.25	1/1
ADDHN2							3/3	0.25/0.25	1/1
ADDP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
ADDV							3/3	0.25/0.25	1/1
AESD	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/2
AESE	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/2
AESIMC	2/2	0.25/0.25	1/1	2/2	0.333/0.333	1/1	2/2	0.25/0.25	1/1
AESMC	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
AND	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
BCAX	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
BIC	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1
BIF	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1
BIT	2/2	0.25/0.25	1/1	2/2	0.25/0.5	1/1	2/2	0.25/0.25	1/1
BSL	2/2	0.25/0.25	1/1	2/2	0.5/0.5	1/1	2/2	0.25/0.25	1/1
CLS	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CLZ	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMEQ	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMGE	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
CMGT	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMHI	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMHS	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMLE	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMLT	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CMTST	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
CNT	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
DUP	0.333/0.333	0.333/0.333	2/2	0.333/0.333	0.333/0.5	2/2	2/12	0.25/0.333	1/2
EOR	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
EOR3	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1
EXT	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FABD	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FABS	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FACGE	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FACGT	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FADD	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FADDP	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FCADD	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FCCMP		1/2	1/1		0.5/2	1/1	2/2	1/1	1/1
FCCMPE		1/2	1/1		0.5/2	1/1	2/2	1/1	1/1
FCMEQ	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FCMGE	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FCMGT	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FCMLA	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
FCMLE	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FCMLT	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FCMP		1/1	1/1		0.5/0.5	1/1	2/2	1/1	1/1
FCMPE		1/1	1/1		0.5/0.5	1/1	2/2	1/1	1/1
FCSEL	2/2	0.5/0.667	1/1	2/2	0.333/0.5	1/1	2/2	0.5/0.5	1/1
FCVT	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FCVTAS	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTAU	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTL							3/3	0.25/0.25	1/1
FCVTL2							3/3	0.25/0.25	1/1
FCVTMS	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTMU	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTN							3/3	0.25/0.25	1/1
FCVTN2							3/3	0.25/0.25	1/1
FCVTNS	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTNU	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTPS	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTPU	0.5/3	0.25/0.5	1/2	0.5/3	0.25/0.667	1/2	3/13	0.25/0.5	1/2
FCVTXN	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
FCVTXN2	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
FCVTZS	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/13	0.25/0.5	1/2
FCVTZU	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/13	0.25/0.5	1/2
FDIV	7/10	1/1	1/1	7/10	1/1	1/1	7/10	1/1	1/1
FJCVTZS	1/1	1/1	2/2	1/1	1/1	2/2	13/13	1/1	2/2

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
FMADD	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMAX	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMAXNM	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMAXNMP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMAXNMV	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FMAXP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMAXV	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1
FMIN	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMINNM	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMINNMP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMINNMV	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FMINP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
FMINV	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FMLA	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMLAL	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMLAL2	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMLS	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMLSL	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMLSL2	4/4	0.25/0.333	1/1	4/4	0.25/0.333	1/1	4/4	0.25/0.25	1/1
FMOV	0.333/2	0.25/0.5	1/1	0.333/2	0.25/0.5	1/1	2/12	0.25/0.5	1/2
FMSUB	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMUL	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FMULX	4/4	0.25/0.333	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FNEG	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
FNMADD	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FNMSUB	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FNMUL	4/4	0.25/0.25	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FRECPE	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
FRECPS	4/4	0.25/0.5	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FRECPX	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
FRINT32X	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINT32Z	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINT64X	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINT64Z	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTA	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTI	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTM	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTP	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTX	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRINTZ	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
FRSQRTE	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
FRSQRTS	4/4	0.25/0.5	1/1	4/4	0.25/0.5	1/1	4/4	0.25/0.25	1/1
FSQRT	8/13	2/2	1/1	8/13	2/2	1/1	8/13	2/2	1/1
FSUB	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
INS	2/2	0.25/0.5	1/2	2/2	0.333/0.5	1/2	2/12	0.25/0.376	1/2
LD1		0.333/0.5	1/1		0.333/0.5	1/1		0.333/2	1/4
LD1R		0.333/0.333	2/2		0.333/0.667	2/2		0.333/1	2/2

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LD2								0.5/1	3/4
LD2R								0.5/1	3/3
LD3								1/2	4/6
LD3R								1/1	4/5
LD4								1/2	5/12
LD4R								1/1	5/6
LDNP		0.333/0.333	2/2		0.333/0.5	2/2		0.333/0.667	2/2
LDP		0.333/1	2/2		0.333/1	2/2	9/9	0.333/0.716	2/2
LDR		0.333/0.333	1/1		0.333/0.5	1/1	9/11	0.333/0.378	1/2
LDUR		0.333/0.5	1/1		0.333/0.5	1/1		0.333/0.333	1/1
MLA	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
MLS	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
MOV	2/2	0.125/0.5	1/2	2/2	0.111/0.5	1/2	2/2	0.125/0.25	1/1
MOVI		0.125/0.25	1/1		0.111/0.333	1/1		0.125/0.25	1/1
MUL	3/3	0.25/0.5	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
MVN	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1			
MVNI		0.25/0.25	1/1		0.25/0.333	1/1		0.25/0.25	1/1
NEG	2/2	0.25/0.333	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
NOT	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
ORN	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1
ORR	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
PMUL	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
PMULL							3/3	0.25/0.25	1/2
PMULL2							3/3	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
RADDHN							3/3	0.25/0.25	1/1
RADDHN2							3/3	0.25/0.25	1/1
RAX1	2/2	0.25/0.25	1/1	2/2	0.333/0.333	1/1	2/2	0.25/0.25	1/1
RBIT	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1	2/2	0.25/0.25	1/1
REV16	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
REV32	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
REV64	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
RSHRN							3/3	0.25/0.25	1/1
RSHRN2							3/3	0.25/0.25	1/1
RSUBHN							3/3	0.25/0.25	1/1
RSUBHN2							3/3	0.25/0.25	1/1
SABA	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SABAL							3/3	0.25/0.25	1/1
SABAL2							3/3	0.25/0.25	1/1
SABD	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SABDL							3/3	0.25/0.25	1/1
SABDL2							3/3	0.25/0.25	1/1
SADALP	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SADDL							2/2	0.25/0.25	1/1
SADDL2							2/2	0.25/0.25	1/1
SADDLP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SADDLV							3/3	0.25/0.25	1/1
SADDW							2/2	0.25/0.25	1/1
SADDW2							2/2	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SCVTF	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/13	0.25/0.333	1/2
SDOT	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SHA1C	5/5	4/4	1/1	5/5	4/4	1/1	4/5	4/4	1/1
SHA1H	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA1M	5/5	4/4	1/1	5/5	4/4	1/1	4/5	4/4	1/1
SHA1P	5/5	4/4	1/1	5/5	4/4	1/1	4/5	4/4	1/1
SHA1SU0	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA1SU1	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA256H							4/5	2/2	1/1
SHA256H2							4/5	2/2	1/1
SHA256SU0	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA256SU1	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
SHA512H							2/3	2/2	1/1
SHA512H2							2/3	2/2	1/1
SHA512SU0	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA512SU1	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHADD	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SHL	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SHLL							2/2	0.25/0.25	1/1
SHLL2							2/2	0.25/0.25	1/1
SHRN							3/3	0.25/0.25	1/1
SHRN2							3/3	0.25/0.25	1/1
SHSUB	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SLI	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SMAX	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SMAXP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SMAXV							3/3	0.25/0.25	1/1
SMIN	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SMINP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SMINV							3/3	0.25/0.25	1/1
SMLAL							3/3	0.25/0.25	1/1
SMLAL2							3/3	0.25/0.25	1/1
SMLSL							3/3	0.25/0.25	1/1
SMLSL2							3/3	0.25/0.25	1/1
SMOV							10/10	0.5/0.5	1/1
SMULL							3/3	0.25/0.25	1/1
SMULL2							3/3	0.25/0.25	1/1
SQABS	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQADD	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQDMLAL	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQDMLAL2	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQDMLSL	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQDMLSL2	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQDMULH	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQDMULL	3/3	0.25/0.5	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQDMULL2	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQNEG	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQRDMLAH	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SQRDMLSH	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQRDMULH	3/3	0.25/0.25	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
SQRSHL	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQRSHRN	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQRSHRN2	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1
SQRSHRUN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQRSHRUN2	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
SQSHL	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SQSHLU	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SQSHRN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQSHRN2	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
SQSHRUN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQSHRUN2	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1	3/3	0.25/0.25	1/1
SQSUB	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQXTN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQXTN2	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQXTUN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SQXTUN2	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SRHADD	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SRI	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SRSHL	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SRSHR	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SRSRA	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SSHL	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SSHLL							2/2	0.25/0.25	1/1
SSHLL2							2/2	0.25/0.25	1/1
SSHR	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SSRA	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SSUBL							2/2	0.25/0.25	1/1
SSUBL2							2/2	0.25/0.25	1/1
SSUBW							2/2	0.25/0.25	1/1
SSUBW2							2/2	0.25/0.25	1/1
ST1		0.5/0.5	1/1		0.5/0.5	1/1		0.5/2	1/4
ST2								0.5/1	2/4
ST3								0.5/1.5	2/6
ST4								0.564/2.501	3/12
STNP		0.5/0.5	2/2		0.5/1	2/2		0.519/1.037	2/2
STP		0.5/1	2/2		0.5/1	2/2		0.5/1	2/2
STR		0.5/0.5	1/1					0.5/0.5	1/2
STUR		0.5/0.5	1/1		0.5/1	1/1		0.5/0.5	1/1
SUB	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
SUBHN							3/3	0.25/0.25	1/1
SUBHN2							3/3	0.25/0.25	1/1
SUQADD	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
SXTL							2/2	0.25/0.25	1/1
SXTL2							2/2	0.25/0.25	1/1
TBL							2/4	0.25/0.75	1/3
TBX							2/8	0.25/1	1/4

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
TRN1	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
TRN2	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UABA	3/3	0.25/0.333	1/1	3/3	0.25/0.5	1/1	3/3	0.25/0.25	1/1
UABAL							3/3	0.25/0.25	1/1
UABAL2							3/3	0.25/0.25	1/1
UABD	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UABDL							3/3	0.25/0.25	1/1
UABDL2							3/3	0.25/0.25	1/1
UADALP	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
UADDL							2/2	0.25/0.25	1/1
UADDL2							2/2	0.25/0.25	1/1
UADDLP	2/2	0.25/0.25	1/1	2/2	0.333/0.333	1/1	2/2	0.25/0.25	1/1
UADDLV							3/3	0.25/0.25	1/1
UADDW							2/2	0.25/0.25	1/1
UADDW2							2/2	0.25/0.25	1/1
UCVTF	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/13	0.25/0.333	1/2
UDOT	3/3	0.25/0.25	1/1	3/3	0.5/0.5	1/1	3/3	0.25/0.25	1/1
UHADD	2/2	0.25/0.25	1/1	2/2	0.333/0.333	1/1	2/2	0.25/0.25	1/1
UHSUB	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UMAX	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UMAXP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UMAXV							3/3	0.25/0.25	1/1
UMIN	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UMINP	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
UMINV							3/3	0.25/0.25	1/1
UMLAL							3/3	0.25/0.25	1/1
UMLAL2							3/3	0.25/0.25	1/1
UMLSL							3/3	0.25/0.25	1/1
UMLSL2							3/3	0.25/0.25	1/1
UMOV							10/10	0.5/0.5	1/1
UMULL							3/3	0.25/0.25	1/1
UMULL2							3/3	0.25/0.25	1/1
UQADD	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UQRSHL	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UQRSHRN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UQRSHRN2	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
UQSHL	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UQSHRN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UQSHRN2	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
UQSUB	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UQXTN	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
UQXTN2	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
URECPE	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
URHADD	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
URSHL	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
URSHR	3/3	0.25/0.25	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
URSQRTE	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
URSRA	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
USHL	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
USHLL							2/2	0.25/0.25	1/1
USHLL2							2/2	0.25/0.25	1/1
USHR	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
USQADD	3/3	0.25/0.5	1/1	3/3	0.25/0.333	1/1	3/3	0.25/0.25	1/1
USRA	3/3	0.25/0.25	1/1	3/3	0.333/0.333	1/1	3/3	0.25/0.25	1/1
USUBL							2/2	0.25/0.25	1/1
USUBL2							2/2	0.25/0.25	1/1
USUBW							2/2	0.25/0.25	1/1
USUBW2							2/2	0.25/0.25	1/1
UXTL							2/2	0.25/0.25	1/1
UXTL2							2/2	0.25/0.25	1/1
UZP1	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
UZP2	2/2	0.25/0.25	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
XAR	2/2	0.25/0.25	1/1	2/2	0.333/0.333	1/1	2/2	0.25/0.25	1/1
XTN							2/2	0.25/0.25	1/1
XTN2							2/2	0.25/0.25	1/1
ZIP1	2/2	0.25/0.5	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1
ZIP2	2/2	0.25/0.333	1/1	2/2	0.25/0.333	1/1	2/2	0.25/0.25	1/1

B.2.2 Efficiency Core

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
ABS	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
ADD	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
ADDHN							3/3	0.5/0.5	1/1
ADDHN2							3/3	0.5/0.5	1/1
ADDP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
ADDV							3/3	0.5/0.5	1/1
AESD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/2
AESE	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/2
AESIMC	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
AESMC	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
AND	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
BCAX	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1
BIC	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
BIF	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1
BIT	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1
BSL	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1
CLS	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CLZ	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMEQ	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMGE	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMGT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMHI	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
CMHS	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMLE	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMLT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CMTST	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
CNT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
DUP	0.5/0.5	0.5/0.5	2/2	0.667/0.667	0.5/0.5	2/2	2/9	0.5/0.5	1/2
EOR	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
EOR3	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1	2/2	0.5/0.5	1/1
EXT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FABD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FABS	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FACGE	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FACGT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FADD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FADDP	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FCADD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FCCMP		1/2	1/1		0.5/2	1/1	2/2	1.048/1.048	1/1
FCCMPE		1/2	1/1		0.5/2	1/1	2/2	1.048/1.048	1/1
FCMEQ	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FCMGE	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FCMGT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FCMLA	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FCMLE	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FCMLT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
FCMP		1/1	1/1		0.5/0.5	1/1	2/2	1/1	1/1
FCMPE		1/1	1/1		0.5/0.5	1/1	2/2	1/1	1/1
FCSEL	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FCVT	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FCVTAS	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTAU	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTL							3/3	0.5/0.5	1/1
FCVTL2							3/3	0.5/0.5	1/1
FCVTMS	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTMU	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTN							3/3	0.5/0.5	1/1
FCVTN2							3/3	0.5/0.5	1/1
FCVTNS	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTNU	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTPS	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTPU	1/3	0.5/1	1/2	0.667/3	0.333/0.667	1/2	3/10	0.5/1	1/2
FCVTXN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FCVTXN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FCVTZS	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/10	0.5/1	1/2
FCVTZU	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/10	0.5/1	1/2
FDIV	7/11	1/2	1/1	7/11	1/2	1/1	7/11	1/2	1/1
FJCVTZS	2/2	2/2	2/2	1/1	1/1	2/2	10/10	2.112/2.112	2/2
FMADD	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMAX	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
FMAXNM	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMAXNMP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMAXNMV	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FMAXP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMAXV	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FMIN	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMINNM	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMINNMP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMINNMV	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FMINP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FMINV	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FMLA	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMLAL	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMLAL2	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMLS	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMLSL	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMLSL2	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMOV	0.5/2	0.5/1	1/1	0.5/2	0.333/0.5	1/1	2/9	0.5/1	1/2
FMSUB	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMUL	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FMULX	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FNEG	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
FNMADD	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FNMSUB	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
FNMUL	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FRECPE	3/4	1/2	1/1	3/4	1/2	1/1	3/4	1/2	1/1
FRECPS	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FRECPX	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
FRINT32X	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINT32Z	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINT64X	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINT64Z	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTA	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTI	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTM	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTP	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTX	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRINTZ	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
FRSQRTE	3/4	1/2	1/1	3/4	1/2	1/1	3/4	1/2	1/1
FRSQRTS	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1	4/4	0.5/0.5	1/1
FSQRT	8/15	2/4	1/1	8/15	2/4	1/1	8/15	2/4	1/1
FSUB	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
INS	2/2	0.5/0.5	1/2	2/2	0.333/0.5	1/2	2/9	0.5/0.75	1/2
LD1		0.5/0.5	1/1		0.5/0.5	1/1		0.5/2	1/4
LD1R		0.5/0.5	2/2		0.667/0.667	2/2		0.5/1	2/2
LD2								1/1.5	3/4
LD2R								1/1	3/3

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
LD3								2/2.072	4/6
LD3R								1.5/2	4/5
LD4								2/4	5/12
LD4R								2/2	5/6
LDNP		1/1	2/2		0.5/0.5	2/2		1/1	2/2
LDP		1/1	2/2		0.5/1	2/2	7/7	1/1.079	2/2
LDR		0.5/0.5	1/1		0.5/0.5	1/1	7/8	0.5/0.551	1/2
LDUR		0.5/0.5	1/1		0.5/0.5	1/1		0.5/0.5	1/1
MLA	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
MLS	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
MOV	2/2	0.25/0.5	1/2	2/2	0.2/0.5	1/2	2/2	0.25/0.5	1/1
MOVI		0.25/0.5	1/1		0.2/0.333	1/1		0.25/0.5	1/1
MUL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
MVN	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1			
MVNI		0.5/0.5	1/1		0.333/0.333	1/1		0.5/0.5	1/1
NEG	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
NOT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
ORN	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
ORR	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
PMUL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
PMULL							3/3	0.5/0.5	1/2
PMULL2							3/3	0.5/0.5	1/1
RADDHN							3/3	0.5/0.5	1/1
RADDHN2							3/3	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
RAX1	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
RBIT	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
REV16	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
REV32	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
REV64	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
RSHRN							3/3	0.5/0.5	1/1
RSHRN2							3/3	0.5/0.5	1/1
RSUBHN							3/3	0.5/0.5	1/1
RSUBHN2							3/3	0.5/0.5	1/1
SABA	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SABAL							3/3	0.5/0.5	1/1
SABAL2							3/3	0.5/0.5	1/1
SABD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SABDL							3/3	0.5/0.5	1/1
SABDL2							3/3	0.5/0.5	1/1
SADALP	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SADDL							2/2	0.5/0.5	1/1
SADDL2							2/2	0.5/0.5	1/1
SADDLP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SADDLV							3/3	0.5/0.5	1/1
SADDW							2/2	0.5/0.5	1/1
SADDW2							2/2	0.5/0.5	1/1
SCVTF	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/10	0.5/0.5	1/2
SDOT	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SHA1C	5/5	4/4	1/1	5/5	4/4	1/1	4/5	4/4	1/1
SHA1H	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA1M	5/5	4/4	1/1	5/5	4/4	1/1	4/5	4/4	1/1
SHA1P	5/5	4/4	1/1	5/5	4/4	1/1	4/5	4/4	1/1
SHA1SU0	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA1SU1	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA256H							4/5	2/2	1/1
SHA256H2							4/5	2/2	1/1
SHA256SU0	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA256SU1	3/3	1/1	1/1	3/3	1/1	1/1	3/3	1/1	1/1
SHA512H							2/3	2/2	1/1
SHA512H2							2/3	2/2	1/1
SHA512SU0	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHA512SU1	2/2	1/1	1/1	2/2	1/1	1/1	2/2	1/1	1/1
SHADD	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SHL	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SHLL							2/2	0.5/0.5	1/1
SHLL2							2/2	0.5/0.5	1/1
SHRN							3/3	0.5/0.5	1/1
SHRN2							3/3	0.5/0.5	1/1
SHSUB	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SLI	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SMAX	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SMAXP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SMAXV							3/3	0.5/0.5	1/1
SMIN	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SMINP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SMINV							3/3	0.5/0.5	1/1
SMLAL							3/3	0.5/0.5	1/1
SMLAL2							3/3	0.5/0.5	1/1
SMLSL							3/3	0.5/0.5	1/1
SMLSL2							3/3	0.5/0.5	1/1
SMOV							7/7	1/1	1/1
SMULL							3/3	0.5/0.5	1/1
SMULL2							3/3	0.5/0.5	1/1
SQABS	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQADD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQDMLAL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQDMLAL2	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQDMLSL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQDMLSL2	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQDMULH	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQDMULL	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQDMULL2	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQNEG	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQRDMLAH	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQRDMLSH	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
SQRDMULH	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SQRSHL	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQRSHRN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQRSHRN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQRSHRUN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQRSHRUN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQSHL	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SQSHLU	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SQSHRN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQSHRN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQSHRUN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQSHRUN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQSUB	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQXTN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQXTN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQXTUN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SQXTUN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SRHADD	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SRI	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SRSHL	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SRSHR	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SRSRA	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SSHL	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SSHLL							2/2	0.5/0.5	1/1
SSHLL2							2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
SSHR	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SSRA	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SSUBL							2/2	0.5/0.5	1/1
SSUBL2							2/2	0.5/0.5	1/1
SSUBW							2/2	0.5/0.5	1/1
SSUBW2							2/2	0.5/0.5	1/1
ST1		1/1	1/1		1/1	1/1		1/4	1/4
ST2								1/2	2/4
ST3								1/3	2/6
ST4								1/4	3/12
STNP		1/1	2/2		1/1	2/2		1.038/2.075	2/2
STP		1/1	2/2		1/1	2/2		1/2	2/2
STR		1/1	1/1					1/1	1/2
STUR		1/1	1/1		1/1	1/1		1/1	1/1
SUB	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
SUBHN							3/3	0.5/0.5	1/1
SUBHN2							3/3	0.5/0.5	1/1
SUQADD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
SXTL							2/2	0.5/0.5	1/1
SXTL2							2/2	0.5/0.5	1/1
TBL							2/4	0.5/1.5	1/3
TBX							2/8	0.5/2	1/4
TRN1	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
TRN2	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
UABA	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
UABAL							3/3	0.5/0.5	1/1
UABAL2							3/3	0.5/0.5	1/1
UABD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UABDL							3/3	0.5/0.5	1/1
UABDL2							3/3	0.5/0.5	1/1
UADALP	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UADDL							2/2	0.5/0.5	1/1
UADDL2							2/2	0.5/0.5	1/1
UADDLP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UADDLV							3/3	0.5/0.5	1/1
UADDW							2/2	0.5/0.5	1/1
UADDW2							2/2	0.5/0.5	1/1
UCVTF	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/10	0.5/0.5	1/2
UDOT	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1	3/3	0.5/0.5	1/1
UHADD	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UHSUB	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UMAX	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UMAXP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UMAXV							3/3	0.5/0.5	1/1
UMIN	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UMINP	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UMINV							3/3	0.5/0.5	1/1
UMLAL							3/3	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
UMLAL2							3/3	0.5/0.5	1/1
UMLSL							3/3	0.5/0.5	1/1
UMLSL2							3/3	0.5/0.5	1/1
UMOV							7/7	1/1	1/1
UMULL							3/3	0.5/0.5	1/1
UMULL2							3/3	0.5/0.5	1/1
UQADD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQRSHL	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQRSHRN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQRSHRN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQSHL	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UQSHRN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQSHRN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQSUB	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQXTN	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
UQXTN2	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
URECPE	3/4	1/2	1/1	3/4	1/2	1/1	3/4	1/2	1/1
URHADD	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
URSHL	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
URSHR	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
URSQRTE	3/4	1/2	1/1	3/4	1/2	1/1	3/4	1/2	1/1
URSRA	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
USHL	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
USHLL							2/2	0.5/0.5	1/1

Instruction	M1 Pro			M3 Max			Johnson		
min/max	latency	throughput	uops	latency	throughput	uops	latency	throughput	uops
USHLL2							2/2	0.5/0.5	1/1
USHR	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
USQADD	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
USRA	3/3	0.5/0.5	1/1	3/3	0.333/0.333	1/1	3/3	0.5/0.5	1/1
USUBL							2/2	0.5/0.5	1/1
USUBL2							2/2	0.5/0.5	1/1
USUBW							2/2	0.5/0.5	1/1
USUBW2							2/2	0.5/0.5	1/1
UXTL							2/2	0.5/0.5	1/1
UXTL2							2/2	0.5/0.5	1/1
UZP1	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
UZP2	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
XAR	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
XTN							2/2	0.5/0.5	1/1
XTN2							2/2	0.5/0.5	1/1
ZIP1	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1
ZIP2	2/2	0.5/0.5	1/1	2/2	0.333/0.333	1/1	2/2	0.5/0.5	1/1