The Common RISC Platform

Desktop Edition

Pacific Semiconductor

January 15, 2021

Contents

0.1 Preface

The PC platform used to be quite open. In the early days, companies would cooperate to design and adopt unified standards that allowed for most parts to be cross-compatible with other vendors' machines. This enabled competition not based on vendor lock-in, but on the strengths of the product itself. It would surprise some to learn that once upon a time, you could buy a PC that shipped with a Pentium III, take the CPU out, and then replace it with a VIA C3. Or even earlier than that, go between a P5 Pentium, AMD K6 or Cyrix 6x86. Those same Socket 7 machines were even backward compatible with CPUs for older sockets.

Compare this to the modern PC platforms. Not only are you locked into a specific vendor, in many cases you are restricted from using your system to its fullest capabilities unless you purchase their most expensive motherboards. In Intel's case, for the past decade or so, consumers have been prevented from overclocking their CPUs unless it is socketed into a board with the Z-series chipset, even if the CPU itself has an unlocked clock multiplier. Recently Intel has doubled down on this practice too, disabling the use of XMP DRAM profiles on lower-end chipsets. This makes absolutely no technical sense since the memory controller is integrated into the CPU; if the CPU and DRAM modules support XMP, then it technically would function. This is nothing more than a slimy attempt to segment the market to shift ill-informed consumers towards higher margin products.

This state of affairs sickens me greatly. The success of vendor lock-in in the mobile industry is not something to be lauded or coveted. It should be viewed by all with nothing but the utmost scorn and contempt. Everything that makes (or made) the PC special is under threat from increasingly anti-consumer practices from companies with deeply entrenched monopolies in their respective sectors. Graphics accelerator manufacturers are trying to force the adoption of proprietary APIs and libraries, CPU vendors are locking traditionally common features behind premium paywalls, and major operating system vendors are trying to wrest control of the hardware ecosystem for themselves.

The corporate giants that have been entrusted with the platform have failed the consumer entirely. I started this project as little more than a rage against the machine, a whinge about the state of computing in general, and a wishlist of things I wanted to see emerge in the PC space over the next ten years. It has since morphed into a fully featured, standardised compute platform, and something that both I and my colleagues should be proud of. We invite industry players to join us in more fully defining the Common RISC Platform with us. All contributions are welcome and will be taken into consideration.

0.2 Standard Assumptions and Rules

For the entirety of this document, it is assumed that the reader is familiar with the following standards and concepts:

- The RISC-V Instruction Set Architecture
- ATX 12VO
- PCI Express
- NVM Express
- Coreboot
- ATX and derivative motherboard form factors
- Coreboot

If the reader is only interested in the Common RISC Platform as an academic exercise, then intricate knowledge of these standards is not required; a fundamental understanding of their basic design is sufficient. However, this document assumes that prospective Platform vendors are already familiar with the intricacies of these standards. They will not be explained or elaborated upon other than to prescribe specific operating conditions required by the Platform.

The only exception to this rule is for concepts that require diagrammatic explanation, such as how the SSI board form factors relate to other physical connectors that are part of the Platform specification.

This document describes the 'base' Common RISC Platform, which we have designed to take the place of the modern consumer desktop Personal Computer, from the low-power OEM market up to and including the enthusiast gaming market. We are very interested in developing variants of the Platform for the HEDT and portable device markets. All market stakeholders are invited to contribute proposals for extensions and modifications to adapt the base specification to be better suited for these markets.

0.3 Glossary of Terms

Chapter 1

A Rationale

1.1 Background

In 2006, then-CEO of Apple Steve Jobs got on stage at the Worldwide Developers Conference and announced that the company would be switching over the Macintosh to IA-32 from PowerPC starting from 2007. This was done in response to the enormous performance gains Intel was making at the time; the company had put the failed NetBurst microarchitecture to rest, and had just released chips based on Core, which was a derivative of the older P6 microarchitecture. Core-based chips were significantly more efficient, offered higher performance and were cheaper than the AIM Alliance's PowerPC offerings. IBM and Motorola were simply unable to continue increasing the performance of their PowerPC chips to match.

This Intel transition was the final nail in the coffin for the RISC desktop. The mid-90s and early 2000s saw fierce competition between RISC architectures for the desktop market from Acorn's ARM and clones thereof, to SGI and their MIPS based workstations, to Tadpole and their SPARC-based devices. By the time Apple had announced its transistion, Macs were the last non-x86 client devices left on the market (ignoring the portable device market, which had by this time been completely dominated by ARM).

However, beginning around 2013, performance gains started drying up. Desktop chips were no longer gaining 30% or 40% performance relative to their predecessors. The high-end market stagnated, with performance uplifts getting down to single-digit percentages, all the while prices crept ever upward. This is due to the performance gains of these large CISC chips being almost entirely the result of process node shrinkages rather than any architectural optimisations. This is evidenced even as recently as the Intel *lake series of chips, which were all built on Intel's 14nm process and saw performance gains that almost exactly correlated with the net increase in clock speed, indicating that there was very little architecture-level optimisation made.

1.2 ARM's March Forward

Despite the failure of any RISC-based architecture to penetrate the desktop market, by 2005 ARM chips were being deployed in around 98% of mobile devices. Given the simplicity of the ARM instruction set, fully-featured ARM cores could be implemented with far fewer transistors than a given CISC IA-32 machine. This made it perfect for low-power deployments, such as the microcontrollers found in the PDAs, digital organisers and feature phones of the era.

In a sort of positive-feedback loop, as ARM became entrenched in the portable market, ARM-based designs were increasingly optimised for performance-per-Watt, rather than simply raw performance. So highly optimised has the ARM ecosystem become that not even Intel, with all its financial horsepower, could compete in compute efficiency. However, in recent times the raw performance lead of x86 chips has been eroded. Apple's A-series mobile SoCs are now able to outperform most of the x86-based low-power designs from Intel and AMD, due in part to the embracement of heterogeneous compute principles and the inherent efficiency advantages of the RISC architecture.

Tim Cook has announced that the Macintosh will be transistioning away from IA-32 and to ARM, starting with the MacBooks. As with the PPC transition, this is due to the performance, efficiency and value stagnation of x86-based offerings. The difference,

however, is that this time Apple is setting the trend, not following it. Apple is the first chip vendor to put forward an ARM-based SoC that can truly compete with x86 offerings. They will not be the last.

1.3 Learning From History

As part of the competition between RISC and CISC based machines that defined the industry in the 90s, many standards were developed to compete with the PC platform. The Advanced Computing Environment was a consortium of system vendors that developed the Advanced RISC Computing (ARC) platform to compete with Wintel – the de-facto PC platform of Intel IA-32 microprocessors running various incarnations of Microsoft Windows. Many vendors signed on to develop machines for ARC, but no machine was ever produced that was fully ARC compliant. There are many reasons for ARC's failure. However, as with the fall of the Western Roman Empire or the outbreak of World War I, there is always a catalysing incident.

The Advanced Computing Environment consortium had settled upon the MIPS architecture as the platform standard. This was done as MIPS Computer Systems was at the time a vendor-neutral fabless design company. However, in 1992, SGI purchased MIPS Computer Systems, which caused a major conflict of interest for the other ARC vendors. Could the newly-incorporated MIPS Technologies be trusted to remain neutral, being a wholly owned subsidiary of SGI? Would SGI receive preferential licensing fee arrangements or preferential access to MIPS's engineering resources? History tells us that the answer was yes, however the ACE consortium barely lasted long enough to find out for themselves. Most vendors quickly abandoned ship and returned to designing Wintel machines.

Aside: ARC's enduring legacy was in its influence over the design of many low-level components of Windows NT, which was originally intended for ARC. The NT bootloader and kernel before NT6.0 used the hardware naming conventions and mapping scheme provided by the ARC system firmware, even on IBM PC clones. The first stage of the NT BIOS bootloader was nothing more than an ARC firmware emulator implemented in x86 assembly and C.

1.4 Fair, Open, and Free

We are of the belief that competition and innovation can only best occur when there is a fair, free and open platform on which companies can compete. Anything less stifles consumer choice and encourages vendors to develop closed systems that reinforce lock-in. This only leads to a fragmented market in which consumers have no power to influence vendors. A locked in consumer is a voiceless consumer. One need only look at the anti-competitive and anti-consumer behaviour practiced by many major chip vendors. A chip vendor should not try to sell an ecosystem, and a platform vendor should not try to lock consumers in to using their chips, and their chips alone.

Thus, we present the Common RISC Platform. The CRP aims to be a feature-complete platform based on existing open standards found in the current PC platform, unified and simplified to provide a platform that is as simple or as flexible as the task requires. The CRP utilises standards such as PCI Express, NVM Express, and ATX12VO to provide a

simple, yet highly flexible and customisable platform. The CRP mandates certain high and low level commonalities between all implementations to ensure full compatibility, however implementations are also highly customisable in terms of feature implementation to ensure vendors can cost-effectively implement competitive CRP systems.

We provide to you the Common RISC Platform entirely free of any charge or licensing agreement. We do this in the hopes that the industry recognise the advantages of a universal, vendor-agnostic platform, and that making its adoption as painless as possible encourages vendors to compete not on the strengths of their Bernaysian marketing teams, but on the strengths of their engineering teams.

Chapter 2

System Control Unit

In the Common RISC Platform, the System Control Unit replaces the traditional CPU. All CRP-compliant SCUs **must** implement:

- At least one RISC-V logic core
- A DDR5 memory controller
- A PCI Express Root Complex
- A USB4 Host Controller

This chapter describes how these components should be implemented.

2.1 Nomenclature

The System Control Unit can be thought of in the same way a modern 'CPU' would be. The change in nomenclature reflects the changing role of the CPU in a modern desktop platform. The modern CPU package typically integrates the various I/O controllers traditionally found on the motherboard's northbridge, as well as increasing numbers of fixed-function units to accelerate specific tasks like cryptography. We initially considered prescribing the return of the Northbridge to handle I/O and memory access, however the vast increase in memory latency and the resultant performance hit would be unacceptable, even when traded against the decrease in system complexity.

Hence, we believe that 'CPU' is a poor monicker for such a device going forward, as modern 'CPUs' are typically neither central to many compute tasks nor do they solely process information. We did not want to use the term 'SoC' for similar reasons – the 'CPUs' of today are already employing advanced packaging techniques that separate functional blocks into unique ICs that are interconnected. Likewise, the term 'SoP' is equally inaccurate as many devices will indeed remain integrated, monolithic ICs. Additionally, neither term truly reflects devices that more closely resemble traditional CPUs.

We believe that the term System Control Unit represents a more accurate description of the device's job in a modern computer system that embraces the princples of heterogeneous compute – execute the control logic of running software and direct that software's data to the appropriate accelerator for processing. Of course, that is not to say that the SCU cannot be an integrated system, or be designed for high compute performance. In fact, we encourage SCU vendors to continue pushing the boundaries of general purpose compute performance as much as possible. However, we must also recognise that the modern device that shares a name with the CPUs of days past bears little functional resemblance to those older devices, and its name should reflect that.

2.2 Minimum Logic Implementation

Common RISC Platform SCUs shall contain **at least one** RISC-V core that fully implements the **RV64GC** Instruction Set. This is the superset of RV64I, M, A, F, D, Zicsr, Zifencei, and C instruction sets. This is the minimum instruction set necessary to cleanly boot a Linux userspace targeted at RISC-V. Core vendors may choose to implement other RISC-V extensions based on their target market, however subsets are strictly forbidden. All other design decisions are at the sole discretion of the designer.

Devices that employ microcode as part of their design must store the code on package. As the Platform Initialisation system is designed to be interoperable, SCU-specific microcode cannot be stored to and loaded from the ROM on the motherboard as is possible with the current PC platform.

IMP. DEP. #1 Core/SCU designers are free to implement additional RISC-V Instruction Sets as they see fit, however subsets are strictly forbidden.

IMP. DEP#2 Vendors of logic devices that use microcode are encouraged to make that code publicly available for audit. Vendors are also encouraged to store microcode on EEPROM, and provide utilities that allow this to be flashed with updated code.

2.3 Memory Controller

The Common RISC Platform uses JEDEC DDR5 SDRAM for main system memory. In keeping with current desktop platforms, the SCU shall implement a dual-channel DDR5 memory controller for communicating with the system memory. Implementation details are at the discretion of the vendor.

2.4 System Communication

The Common RISC Platform prescribes a univeral SCU socket, CRP-1. All compliant SCUs must be electrically and physically compatible with the CRP-1 socket, as described under **MOTHERBOARD CHAPTER POINTER GOES HERE**.

The SCU shall accept two voltage inputs – V_{CORE} and V_{UNCORE}. SCU-VRM communications are explained in detail in **POWER DELIVERY CHAPTER GOES HERE**. The SCU will, at stock settings, draw no more than 85W from the socket.

The SCU must expose a PCI Express Root Complex. Most off-package communication with other system components must be done through the PCI Express fabric. The Root Complex must expose 48 lanes to the system. 40 of these lanes may be flexibly assigned by motherboard vendors for , and the remaining 8 are reserved for interfacing with the Platform Hub. The CRP-1 pinout allocates specific pins for each category of lane. Other interconnects must communicate with the SCU through the Platform Hub. This is described in depth in **MOTHERBOARD CHAPTER GOES HERE**.

The SCU must also expose a USB4 Host Controller. SCUs that implement a GPU on package must use USB4 DisplayPort Alternate Mode for display output. All boards, regardless of other design decisions, must implement at least one USB Type-C connector on the rear I/O panel connected directly to the SCU Host Controller such that any board can be fully compatible with any SCU, including those with integrated graphics.

Chapter 3

Platform Initialisation

3.1 Specifications

All CRP-compliant motherboards shall use **coreboot** for platform initialisation. Systems must support all coreboot payloads, and no CRP-compliant device shall prevent the execution of any payload under any circumstances. The coreboot image shall be stored on a socketed 64Mb EEPROM chip. Additionally, compliant boards shall not prevent the user from replacing the manufacturer-provided PI firmware with any compatible firmware image from third parties.

Motherboard vendors are encouraged to open source their coreboot images

---

IMP. DEP. #3 A CRP-compliant coreboot may or may not provide a pre-payload user interface that allows for platform settings to be customised, e.g. a coreboot may have facilities that allow the user to overclock their SCU and memory.

Chapter 4

The CRP-1 Socket

The CRP-1 socket defines a standard interconnect for all SCUs. This chapter describes the physical and electrical characteristics of the CRP-1 socket.

4.1 Physical Description

CRP-1 is a Land Grid Array socket with 2000 pins, each with a diameter of 0.75mm and a pitch of 1mm.