# RISC-V Configuration Description
# Version 0.1.0-DRAFT
# 70f7d51e6452bd20bc30289cca8077102b2159a5

Editors:
Abner Chang <abner.chang@hpe.com>, Hewlett Packard Enterprise
Tim Newsome <tim@sifive.com>, SiFive, Inc.

Tue Apr 7 12:55:41 2020 -0700

Contributors to all versions of the spec in alphabetical order (please contact editors to suggest corrections): TODO

# Preface

**Warning! This draft specification will change before being accepted as standard, so implementations made to this draft specification will likely not conform to the future standard.**

# Contents

**Preface** i

**1 Introduction** 1

1.1 Background . . . . . 1

1.2 Goals . . . . . 1

1.3 Use Cases . . . . . 2

1.3.1 External Debuggers . . . . . 2

1.3.2 System Firmware . . . . . 2

**2 Machine-Readable Format** 3

**3 Human-Readable Format** 4

**4 Access Method** 5

**A External Industrial Standards** 6

**Index** 7

**B Change Log** 7

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Background

It is generally useful for software to be able to programmatically determine the capabilities of the hardware it is running on. External debuggers need to know the same information. RISC-V extensions mostly use CSRs for this information, but that is not flexible enough for some kinds of data like hardware breakpoint capabilities, and cache hierarchy.

This document specifies syntax, semantics, discovery method, and access method for a static data structure that can accommodate implementation parameters of RISC-V standards beyond what can be easily put into CSRs. The structure is called a Configuration Structure (CS).

## 1.2 Goals

The task group will deliver:

1. A specification for a machine-readable format for the configuration structure. It's intended to be easily accessible by M-mode software and debuggers.
2. A specification for a human-readable format for the configuration structure. It's intended to be used to talk about the system description in this document as well as other documentation, to help designers write a configuration structure, and to display the configuration structure to people.
3. A software tool that translates back and forth between the machine-readable and human-readable format.
4. A specification for a method to discover and access, from M-mode software, the machine-readable configuration structures.

The configuration structure should be flexible enough that future task groups won't feel the need to create another structure used to describe implementation parameters. Implementation parameters are details that a RISC-V specification explicitly leaves up to an implementation. This includes hart-specific details like the kinds of hardware triggers supported, as well as details that are outside

harts such as the supported abstract debug commands.

## 1.3 Use Cases

### 1.3.1 External Debuggers

When an external debugger connects to a system, it would like to discover as much as possible about that system in as little time as possible. Some of this is merely to show the user (e.g. a manufacturer name), while other features are critical to the user (e.g. XLEN), and other features determine what kind of operations the user can perform (e.g. supported hardware trigger types). Most of these are already discoverable, although many require writing a value and checking the result to see whether support exists.

In the case of hardware trigger types there are so many possible combinations of options that this is prohibitive. In other cases the software to detect all features is quite complex. When some harts are powered down, no discovery is possible on them at all. Supplying a system description can solve all these problems.

### 1.3.2 System Firmware

Typical system firmware is executed when the system is powered on. It initializes hardware, and builds up firmware services or data structures for booting up system to OS. Examples are uboot for embedded systems, and BIOS for PCs.

Throuh a combination of checking CSRs and accessing the system description (Section 4), firmware can programmatically determine the hardware capabilities and configure hardware accordingly. These hardware capabilities can include availability and implemented features of Physical Memory Protection (PMP), Core Local Interrupt Controller (CLIC), Core Local Interruptor (CLINT), memory map, Platform-Level Interrupt Controller (PLIC), Real Time Clock (RTC), reset mechanism, and any future optional core features.

The Configuration Structure is an efficient alternative to testing for specific hardware features (including handling failures) or customizing system firmware for the specific system it will run on.

Often system firmware will take the information it has learned from the system description as well as through other methods, and encode it into a different industry-standard data structure (like Devicetree). This structure is then passed to the subsequent boot process.

---

*The system firmware mentioned refers to startup software which is executed when the system powers on. The system firmware initializes hardware configuration and builds up firmware services or data structures for booting up system to OS. The typical system firmware such as uboot for embedded systems, BIOS for PCs or other firmware frameworks.*

# Chapter 2

# Machine-Readable Format

Ideas:

1. Fully custom, along the lines of https://github.com/riscv/riscv-debug-spec/pull/450/files#diff-9cee87c1f0feb10d9e3e9b2bfad92985R380
2. Semi-custom, using CBOR
3. ...

# Chapter 3

# Human-Readable Format

Ideas:

1. Take the binary format, and expand constants to strings, add whitespace, keeping the mapping as close to 1:1 as possible.
2. XML
3. JSON
4. ...

# Chapter 4

# Access Method

Ideas:

1. A CSR contains the base address of where the structure can be accessed on the system bus
2. A CSR address/data pair can be used to access the structure. (Write index to the address CSR, then read from the data CSR.)
3. ...

# Appendix A

# External Industrial Standards

# Appendix B

# Change Log

| Revision | Date | Author(s) | Description |
|---|---|---|---|
| 70f7d51 | 2020-04-07 | Tim Newsome | Basic scaffolding. |