RISC-V UEFI PROTOCOL Specification

RISC-V Platform HSC Group

Version 1.0.0, 05/09/2022: Ratified

Table of Contents

Preamble

This document is in the Ratified state

No changes are allowed. Any desired or needed changes can be the subject of a follow-on new extension. Ratified extensions are never revised.

Copyright and license information

Contributors

This RISC-V specification has been contributed to directly or indirectly by:

- Sunil V L <sunilvl@ventanamicro.com>
- Ard Biesheuvel <ardb@kernel.org>
- Heinrich Schuchardt <heinrich.schuchardt@canonical.com>
- Jessica Clarke <jrtc27@jrtc27.com>
- Atish Patra <atishp@atishpatra.org>
- Anup Patel <apatel@ventanamicro.com>
- Abner Chang <abner.chang@hpe.com>

Chapter 1. Introduction

This specification details all new UEFI protocols required only for RISC-V platforms. These protocol specs are maintained by RISC-V community.

Chapter 2. Revision History

Date	Revision	Change
05/09/2022	1.0.0	Ratified version
03/23/2022	1.0-rc3	Frozen version
01/21/2022	1.0-rc2	Few documentation template changes
01/18/2022	1.0-rc1	Renamed EFI_RISCV_BOOT_PROTOCOL to RISCV_EFI_BOOT_PROTOCOL as per Ard's feedback.
01/13/2022	0.3 Draft	Improved text as per feedback from Atish.
01/12/2022	0.2 Draft	1) Added wrapper EFI_PROTOCOL spec 2) Added protocol revision field directly in BOOT_PROTOCOL and remove GetProtocolVersion 3) Changed UINT32 * to UINTN * for BootHartId in BOOT_PROTOCOL
01/10/2022	0.1 Draft	Initial Draft for Platform HSC review

Chapter 3. RISCV_EFI_BOOT_PROTOCOL

Either Device Tree (DT) or Advanced Configuration and Power Interface (ACPI) configuration tables are used to convey the information about hardware to the Operating Systems. Some of the information are known only at boot time and needed very early before the Operating Systems/boot loaders can parse the firmware tables.

One example is the boot hartid on RISC-V systems. On non-UEFI systems, this is typically passed as an argument to the kernel (in a0). However, UEFI systems need to follow UEFI application calling conventions and hence it can not be passed in a0. There is an existing solution which uses the /chosen node in DT based systems to pass this information. However, this solution doesn't work for ACPI based systems. Hence, a UEFI protocol is preferred for both DT and ACPI based systems.

This UEFI protocol for RISC-V systems provides early information to the bootloaders or Operating Systems. Firmwares like EDK2 and u-boot need to implement this protocol on RISC-V UEFI systems.

This protocol is typically called by the bootloaders before invoking **ExitBootServices()**. They then pass the information to the Operating Systems.

The version of RISCV_EFI_BOOT_PROTOCOL specified by this specification is 0x00010000. All future revisions must be backwards compatible. If a new version of the specification breaks backwards compatibility, a new GUID must be defined.

3.1. GUID

```
#define RISCV_EFI_BOOT_PROTOCOL_GUID \
    { 0xccd15fec, 0x6f73, 0x4eec, \
    { 0x83, 0x95, 0x3e, 0x69, 0xe4, 0xb9, 0x40, 0xbf } }
```

3.2. Revision Number

```
#define RISCV_EFI_BOOT_PROTOCOL_REVISION 0x00010000
#define RISCV_EFI_BOOT_PROTOCOL_LATEST_VERSION \
        RISCV_EFI_BOOT_PROTOCOL_REVISION
```

3.3. Protocol Interface Structure

```
typedef struct _RISCV_EFI_BOOT_PROTOCOL {
    UINT64                Revision;
    EFI_GET_BOOT_HARTID   GetBootHartId;
} RISCV_EFI_BOOT_PROTOCOL;
```

3.3.1. RISCV_EFI_BOOT_PROTOCOL.GetBootHartId

This interface provides the hartid of the boot cpu.

Prototype

```
typedef EFI_STATUS
(EFIAPI *EFI_GET_BOOT_HARTID) (
    IN RISCV_EFI_BOOT_PROTOCOL *This,
    OUT UINTN                  *BootHartId
    );
```

Parameters

Table 1. GetBootHartId Parameters

Parameter	Description
This	Pointer to the protocol
BootHartId	Pointer to the variable receiving the hartid of the boot cpu.

Status Codes Returned

Table 2. GetBootHartId Return Value

Return Value	Description
EFI_SUCCESS	The boot hart id could be returned.
EFI_INVALID_PARAMETER	**This** parameter is NULL or does not point to a valid RISCV_EFI_BOOT_PROTOCOL implementation.
EFI_INVALID_PARAMETER	**BootHartId** parameter is NULL.

References

- Discussion on the requirement