

MICROCHIP  
PIC64GX 64-BIT  
RISC-V QUAD-  
CORE



# MICROCHIP PIC64GX 64-Bit RISC-V Quad-Core Microprocessor User Guide

[Home](#) » [MICROCHIP](#) » MICROCHIP PIC64GX 64-Bit RISC-V Quad-Core Microprocessor User Guide 

## Contents

- [1 MICROCHIP PIC64GX 64-Bit RISC-V Quad-Core Microprocessor](#)
- [2 Product Information](#)
- [3 Product Usage Instructions](#)
- [4 Introduction](#)
- [5 Boot Process](#)
- [6 Reboot](#)
- [7 Watchdogs](#)
- [8 Lockdown Mode](#)
- [9 Appendix](#)
- [10 Microchip Information](#)
- [11 Worldwide Sales and Service](#)
- [12 Documents / Resources](#)
  - [12.1 References](#)
- [13 Related Posts](#)



**MICROCHIP PIC64GX 64-Bit RISC-V Quad-Core Microprocessor**



## Product Information

### Specifications:

- **Product Name:** Microchip PIC64GX
- **Boot Process:** SMP and AMP workloads supported
- **Special Features:** Watchdog support, Lockdown mode

## Product Usage Instructions

### 1. Boot Process

#### 1. Software Components Involved in Booting

The system boot-up process involves the following software components:

- **Hart Software Services (HSS):** A zero-stage boot loader, system monitor, and provider of runtime services for applications.

#### 2. Boot Flow

The sequence of the system boot flow is as follows:

1. Initialization of Hart Software Services (HSS)
2. Bootloader execution
3. Application startup

### 2. Watchdogs

#### 1. PIC64GX Watchdog

The PIC64GX features a watchdog function to monitor system operation and trigger actions in case of system failures.

### 3. Lockdown Mode

The lockdown mode is designed for customers who require complete control over system actions after boot. It limits the functionalities of the E51 system monitor.

## FAQ

- **Q: What is the purpose of the Hart Software Services (HSS)?**

A: The HSS serves as a zero-stage boot loader, system monitor, and provider of runtime services for applications during the boot process.

- **Q: How does the PIC64GX watchdog function work?**

A: The PIC64GX watchdog monitors system operation and can take predefined actions in case of system failures to ensure system reliability.

## Introduction

This whitepaper explains how the Microchip PIC64GX boots application workloads and describes the system boot process, which operates the same for SMP and AMP workloads. Additionally, it covers how a reboot works for SMP and AMP workloads, watchdogs on the PIC64GX, and a special lockdown mode for systems where customers desire complete control to limit the actions of the E51 system monitor after system boot.

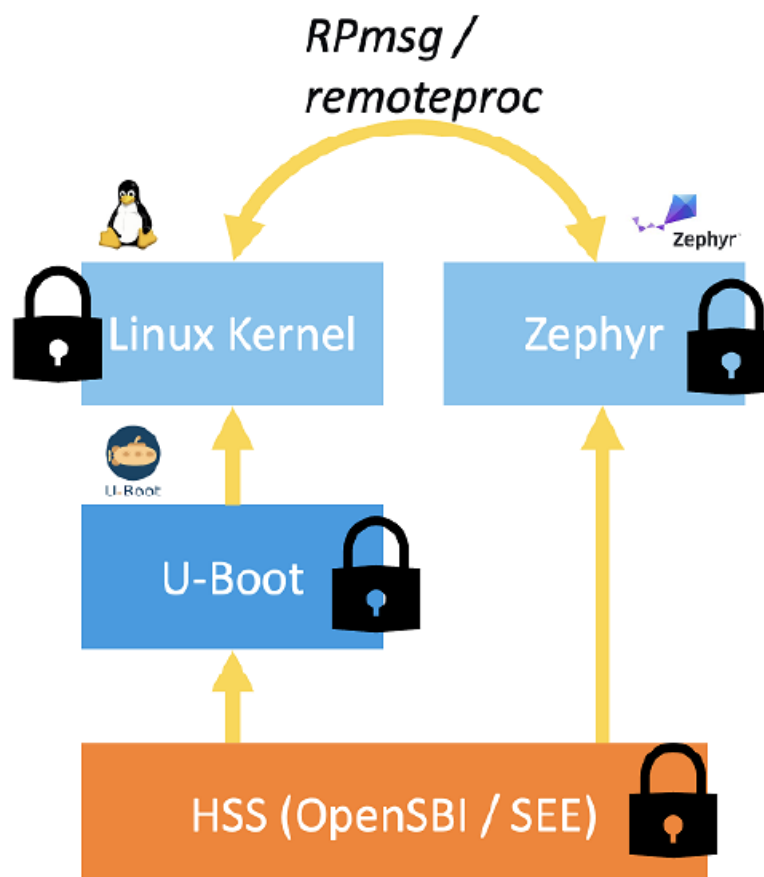
## Boot Process

Let us take a look at the various software components involved in system bootup, followed by a more detailed look at the sequence of the system boot flow itself.

### Software Components Involved in Booting

The following components are involved in the system boot-up process:

**Figure 1.1. Boot-up Components**



- **Hart Software Services (HSS)**

The Hart Software Services (HSS) is a zero-stage boot loader, a system monitor, and a provider of runtime services for applications. The HSS supports early system setup, DDR training, and hardware initialization/configuration. It mostly runs on the E51s, with a small amount of machine-mode level functionality running on each U54s. It boots one or more contexts by loading application “payload” from the boot medium, and provides Platform Runtime Services/Supervisor Execution Environment (SEE) for operating system kernels. It supports secure boot and is an important component in ensuring hardware partitioning/separation for AMP contexts.

- **Das U-Boot (U-Boot)**

Das U-Boot (U-Boot) is an open-source universal scriptable boot loader. It supports a simple CLI that can retrieve the boot image from a variety of sources (including an SD Card and the Network). U-Boot loads Linux. It can provide a UEFI environment if needed. It is generally finished and out of the way once Linux has booted – in other words, it does not stay resident post-boot.

- **Linux Kernel**

The Linux kernel is the world's most popular operating system kernel. Combined with a userland of applications, it forms what is commonly referred to as a Linux operating system. A Linux Operating System provides rich POSIX APIs and developer environment, for example, languages and tools such as Python, Perl, Tcl, Rust, C/C++, and Tcl; libraries such as OpenSSL, OpenCV, OpenMP, OPC/UA, and OpenAMP (RPMmsg and RemoteProc).

Yocto and Buildroot are Linux system builders, that is, they can be used to generate bespoke customized Linux systems. Yocto outputs a Linux distribution with a rich set of applications, tools, and libraries, and optional package management. Buildroot outputs a more minimal root filesystem and can target systems that do not require persistent storage but run entirely from RAM (using Linux's initials support, for example).

- **Zephyr**

Zephyr is a small, open-source Real-Time Operating System (RTOS). It provides a Real-Time Low-Overhead Framework, with RPMmsg-lite communication channels to Linux. It includes a kernel, libraries, device drivers, protocol stacks, filesystems, mechanisms for firmware updates, and so on, and is great for customers wanting a more bare-metal-like experience on PIC64GX.

## **Boot Flow**

PIC64GX includes a RISC-V coreplex with a 64-bit E51 system monitor hart and 4 64-bit U54 application harts. In RISC-V terminology, a hart is a RISC-V execution context that contains a full set of registers and that executes its code independently. You can think of it as a hardware thread or a single CPU. A group of harts within a single core is often called a complex. This topic describes the steps to initialize the PIC64GX coreplex, including the E51 system monitors heart and the U54 application harts.

1. Power on the PIC64GX coreplex.

At power-on, all harts in the RISC-V coreplex are released from reset by the Security Controller.

2. Run the HSS code from the on-chip eNVM flash memory.

Initially, each hart starts running the HSS code from the on-chip eNVM flash memory. This code causes all U54 application harts to spin, waiting for instructions, and lets the E51 monitor hart start running code to initialize and bring up the system.

3. Decompress the HSS code from eNVM to L2-Scratch memory.

Depending on its build-time configuration, the HSS is usually larger than the capacity of the eNVM flash

memory itself and so the first thing the HSS code running on the E51 does is decompress itself from eNVM to L2-Scratch memory, as shown in Figure 1.2 and Figure 1.3.

Figure 1.2. HSS Decompresses from eNVM to L2 Scratch

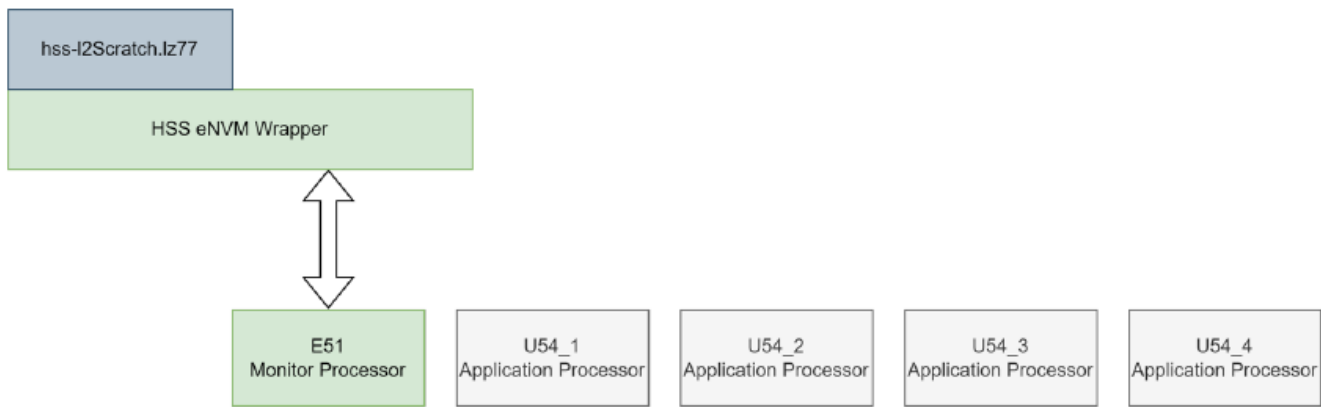
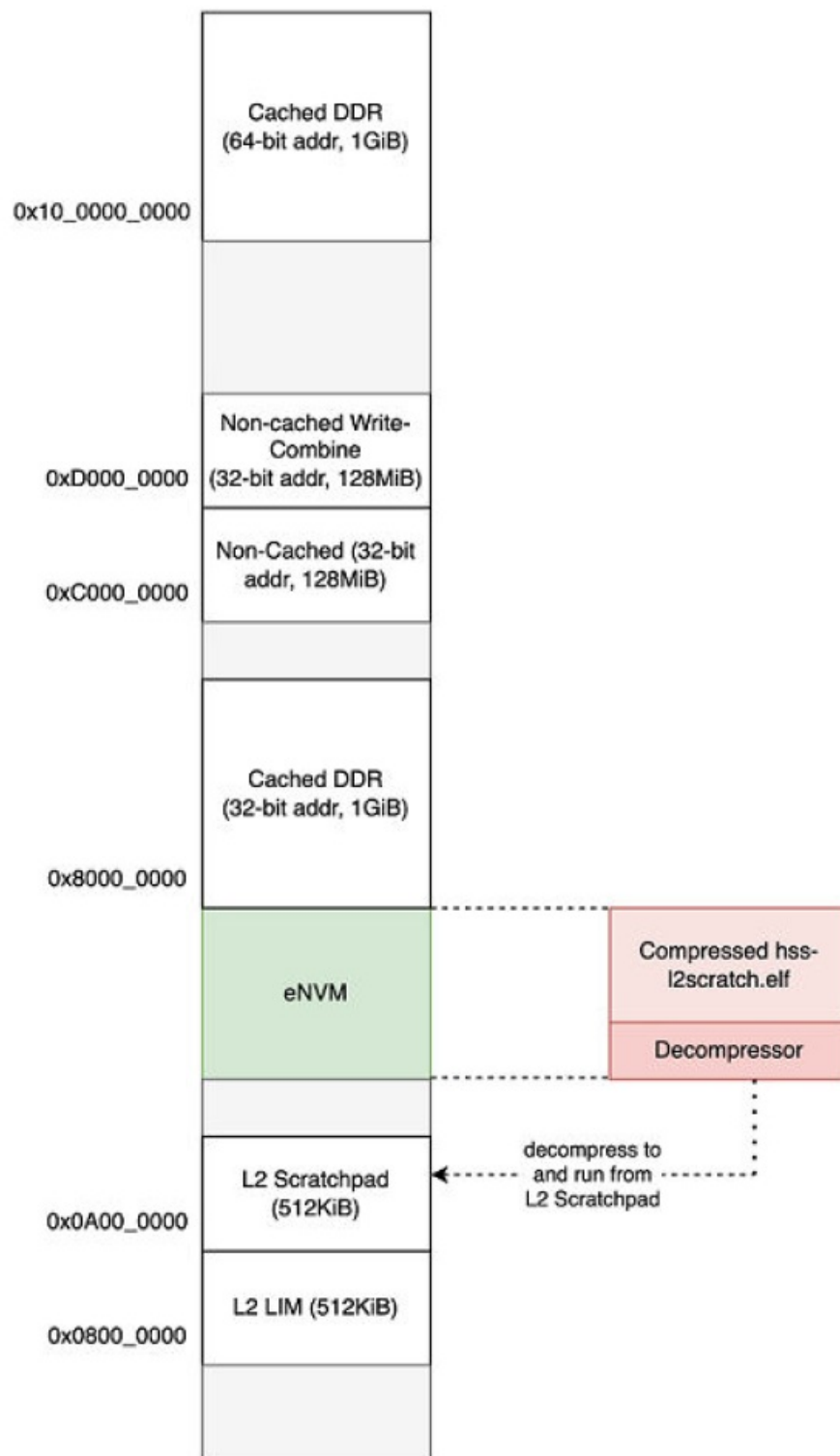
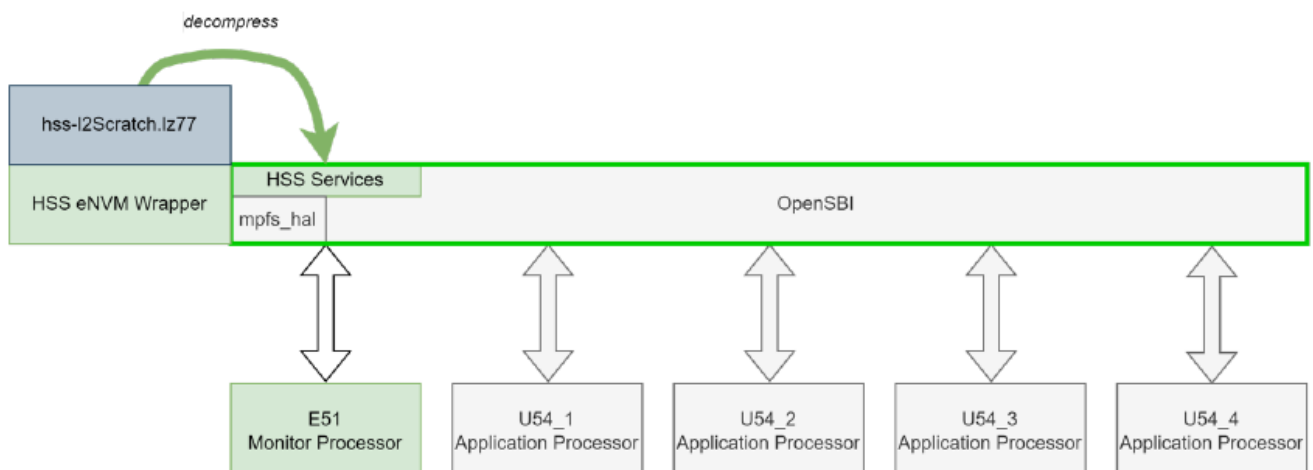


Figure 1.3. HSS Memory Map During Decompression



4. Jump from eNVM to L2-Scratch into an executable as shown in the following figure.

Figure 1.4. HSS Jumps from eNVM into Code Now in L2Scratch Following Decompression



**The executable consists of three components:**

- The hardware abstraction layer (HAL), low-level code, and bare metal drivers
- A local HSS fork of RISC-V OpenSBI (modified slightly from upstream on PIC64GX for AMP purposes)
- The HSS runtime services (state machines run in a super loop)

5. Initialize the hardware and data structures used by OpenSBI.

The HSS service “Startup” is responsible for this initialization.

6. Fetch the application workload (payload.bin) image from external storage. This is shown in Figure 1.5 and Figure 1.6

Important: In case of the PIC64GX Curiosity Kit, this will be from an SD card.

Figure 1.5. Fetching payload.bin Workload Image from External Storage

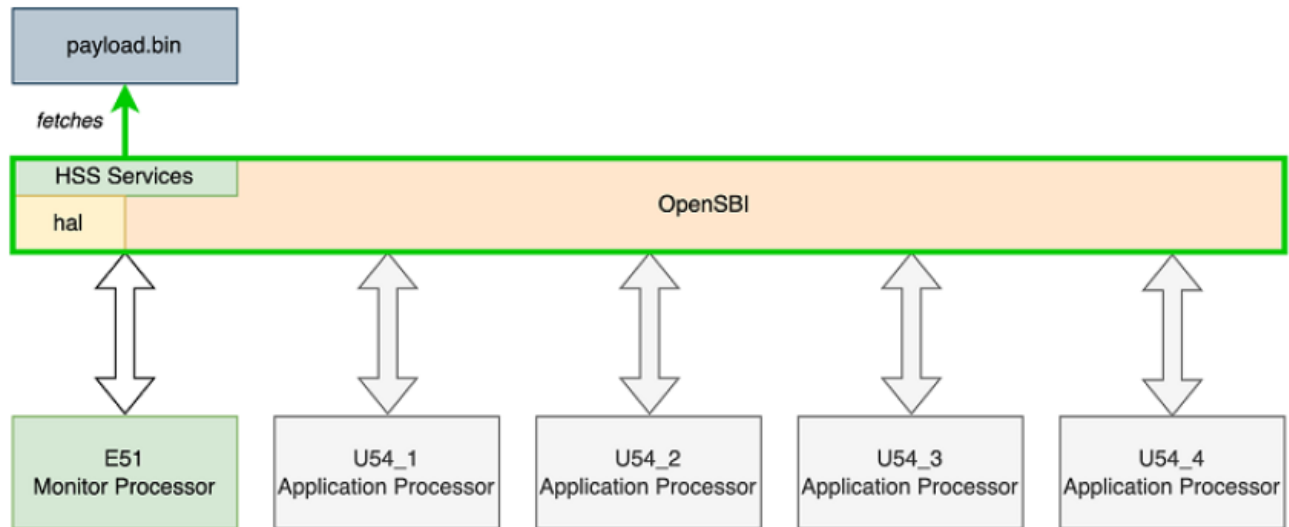
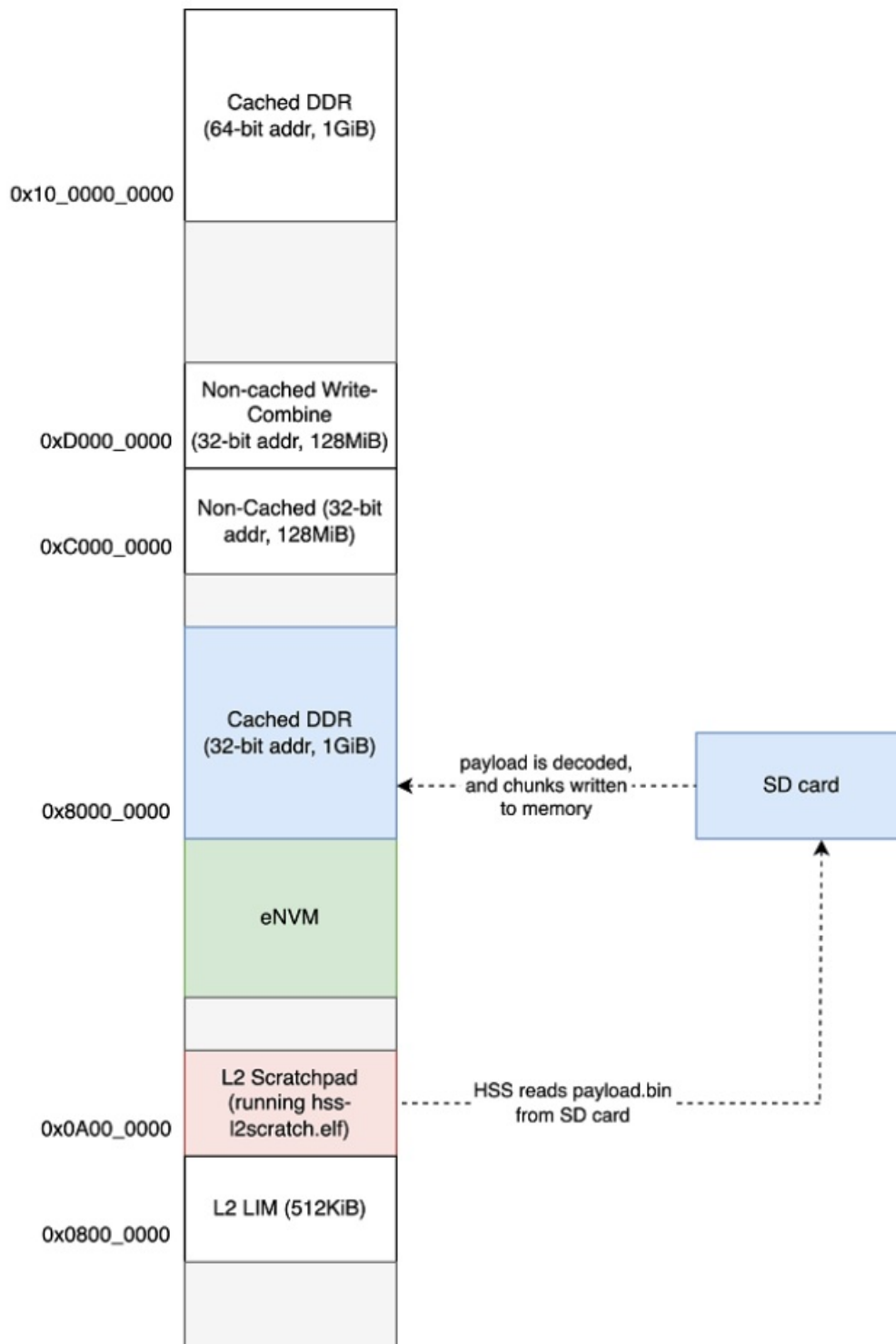


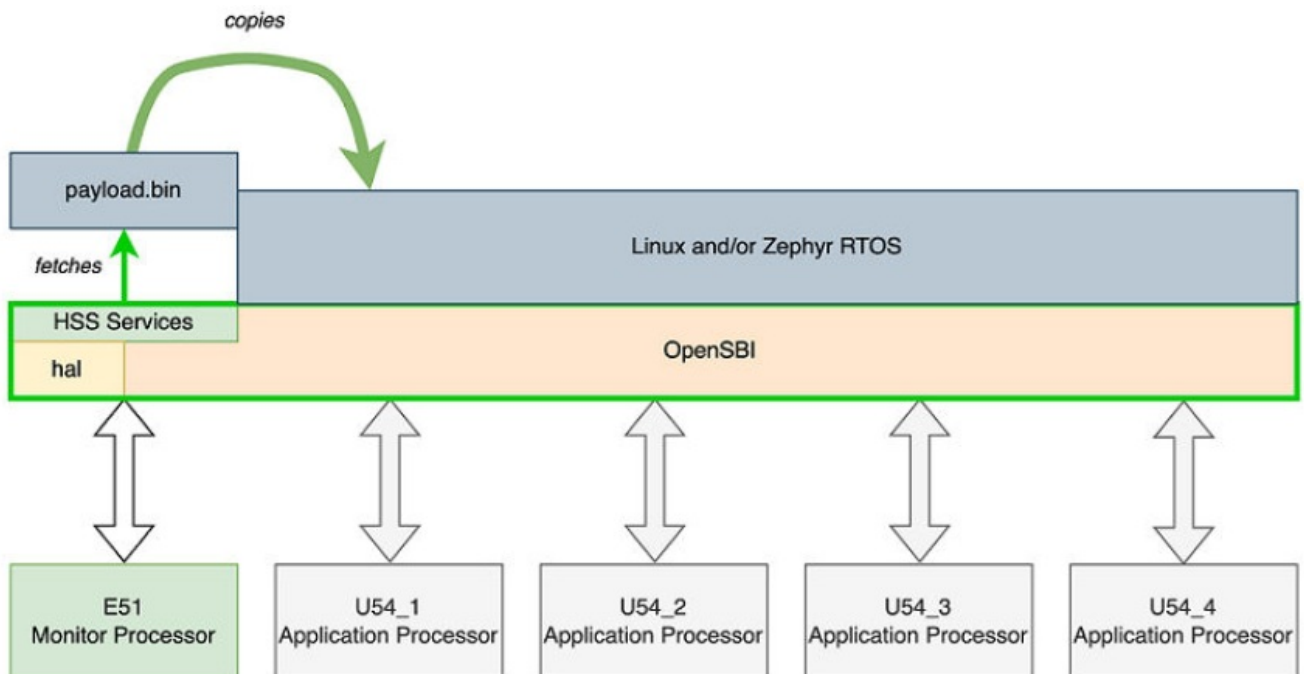
Figure 1.6. HSS Memory Map after Fetching payload.bin



- Copy the various sections from the payload.bin to their execution time destinations. The payload.bin is a formatted image, which consolidates various application images for SMP or AMP workloads. It includes code, data, and descriptor tables which enable the HSS to appropriately place the code and data sections, where they are needed to run the various application workloads.

Figure 1.7. payload.bin is Copied to Destination Addresses





8. Instruct relevant U54s to jump to their execution start addresses. This start address information is contained in the payload.bin.
9. Start the U54 Application harts and any second-stage boot loaders. For example, U-Boot brings up Linux.

## Reboot

Related to the concept of system booting is the need to reboot. When thinking about PIC64GX application workloads, rebooting needs to consider both symmetric multiprocessing (SMP) and asymmetric multiprocessing (AMP) scenarios:

1. In the case of an SMP system, a reboot can safely cold reboot the entire system as there are no additional workloads in another context to consider.
2. In the case of an AMP system, a workload might only be allowed to reboot itself (and not interfere with any other context), or it might be privileged to be able to perform a full system reboot.

## Reboot and AMP

To enable the SMP and AMP reboot scenarios, the HSS supports the concepts of warm and cold reboot privileges, which are assignable to a context. A context with warm reboot privilege is only able to reboot itself, and a context with cold reboot privilege can perform a full system reboot. For example, consider the following set of representative scenarios.

- A single context SMP workload, which is allowed to request a full system reboot
- In this scenario, the context is allowed cold reboot privilege.
- A two-context AMP workload, where context A is allowed to request a full system reboot (affecting all contexts), and Context B is allowed to reboot itself only
- In this scenario, context A is allowed cold reboot privilege, and context B is allowed warm reboot privilege.
- A two-context AMP workload, where contexts A and B are only allowed to reboot themselves (and not affect the other context)
- In this scenario, both contexts are only allowed warm reboot privileges.
- A two-context AMP workload, where contexts A and B are both allowed to request full system reboots
- In this scenario, both contexts are allowed cold reboot privileges.

- Furthermore, it is possible for the HSS at build time to always allow cold reboot privilege, and to never allow cold reboot privilege.

### Relevant HSS Kconfig Options

Kconfig is a software build configuration system. It is commonly used to select build-time options and to enable or disable features. It originated with the Linux kernel but has now found use in other projects beyond the Linux kernel, including U-Boot, Zephyr, and the PIC64GX HSS.

**The HSS contains two Kconfig options that control the reboot functionality from the HSS perspective:**

- **CONFIG\_ALLOW\_COLD REBOOT**

If this is enabled, it globally allows a context to issue a cold reboot ecall. If disabled, only warm reboots will be permitted. In addition to enabling this option, permission to issue a cold reboot must be granted to a context through the payload generator YAML file or the following Kconfig option.

- **CONFIG\_ALLOW\_COLD REBOOT\_ALWAYS**

- If enabled, this feature globally allows all contexts to issue a cold reboot ECAA, irrespective of the payload.bin flag entitlements.
- Additionally, the payload.bin itself can contain a per-context flag, indicating that a particular context is entitled to issue cold reboots:
  - To allow a context warm reboot another context, we can add the option allow-reboot: warm in the YAML description file used to create the payload.bin
  - To allow a context cold reboot of the entire system, we can add the option allow-reboot: cold. By default, without specifying allow-reboot, a context is only allowed warm reboot itself. Irrespective of the setting of this flag, if CONFIG\_ALLOW\_COLDREBOOT is not enabled in the HSS, the HSS will rework all cold reboot requests to warm (per-context) reboots.

### Reboot in Detail

This section describes how the reboot works in detail – starting with the OpenSBI layer (the lowest M-mode layer) and then discussing how this OpenSBI layer functionality is triggered from an RTOS application or a rich OS like Linux.

### OpenSBI Reboot ecall

- The RISC-V Supervisor Binary Interface (SBI) specification describes a standardized hardware abstraction layer for platform initialization and firmware runtime services. The main purpose of SBI is to enable portability and compatibility across different RISC-V implementations.
- OpenSBI (Open Source Supervisor Binary Interface) is an open-source project that provides a reference implementation of the SBI specification. OpenSBI also provides runtime services, including interrupt handling, timer management, and console I/O, which can be utilized by higher-level software layers.
- OpenSBI is included as part of the HSS and runs at the Machine Mode level. When the operating system or application causes a trap, it will be passed to OpenSBI to handle it. OpenSBI exposes a certain system-call type functionality to the upper layers of the software through a particular trap mechanism called an ecall.
- The System Reset (EID 0x53525354) provides a comprehensive system call function that allows the upper layer software to request system-level reboot or shutdown. Once this ecall is invoked by a U54, it is trapped by the HSS software running in Machine Mode on that U54, and a corresponding reboot request is sent to the E51 to reboot either the context or the entire system, depending on the entitlements of the context.

For more information, see the [RISC-V Supervisor Binary Interface Specification](#) particularly [System Reset Extension \(EID #0x53525354 “SRST”\)](#).

## Linux Reboot

As a specific example of this, in Linux, the shutdown command is used to halt or reboot the system. The command typically has many aliases, namely halt, power off, and reboot. These aliases specify whether to halt the machine at shutdown, to power off the machine at shutdown, or to reboot the machine at shutdown.

- These user-space commands issue a reboot system call to Linux, which is trapped by the kernel and interworked to an SBI ecall.
- There are various levels of reboot – REBOOT\_WARM, REBOOT\_COLD, REBOOT\_HARD – these can be passed as command line arguments to the kernel (for example, reboot=w[arm] for REBOOT\_WARM). For more information on the Linux kernel source code, see [Documentation/admin-guide/kernel-parameters.txt](#).
- Alternatively, if /sys/kernel/reboot is enabled, the handlers underneath can be read to get the current system reboot configuration, and written to alter it. For more information on the Linux kernel source code, see [Documentation/ABI/testing/sysfs-kernel-reboot](#).

## Watchdogs

- A further concept related to system booting and system rebooting is that of system recovery upon the firing of a watchdog timer. Watchdog timers are widely used in embedded systems to automatically recover from transient hardware faults, and to prevent errant or malevolent software from disrupting system operation.
- PIC64GX includes hardware watchdog support to monitor the individual harts when the system is running. The watchdogs ensure that the harts can be restarted if they do not respond due to unrecoverable software errors.
- PIC64GX includes five instances of watchdog timer hardware blocks used to detect system lockups -one for each of the harts. To facilitate mixed Asymmetric Multi-Processing (AMP) workloads, the HSS supports monitoring and reacting to the watchdogs firing.

## PIC64GX Watchdog

- The HSS is responsible for booting the application harts at power-up, and for re-booting them (individually or collectively) at any stage, should it be needed or desired. As a consequence of this, reacting to watchdog events on the PIC64GX is handled by the HSS.
- A 'virtual watchdog' monitor is implemented as an HSS state machine service, and its responsibilities are to monitor the status of each of the U54 individual watchdog hardware monitors. When one of these U54 watchdogs trips, the HSS detects this and will reboot the U54 as appropriate. If the U54 is part of an SMP context, the entire context is considered for reboot, given the context has warm reboot privilege. The entire system will be rebooted if the context has cold reboot privilege.

## Relevant Kconfig Options

- Watchdog support is included by default in released HSS builds. Should you wish to build a custom HSS, this section will describe the configuration mechanism to ensure that Watchdog support is enabled.
- The HSS is configured using the Kconfig configuration system. A toplevel .config file is needed to select what

services get compiled in or out of the HSS build.

- Firstly, the top-level CONFIG\_SERVICE\_WDOG option needs to be enabled (“Virtual Watchdog support” through make config).

**This then exposes the following sub-options which are dependent on Watchdog support:**

- **CONFIG\_SERVICE\_WDOG\_DEBUG**

Enables support for informational/debug messages from the virtual watchdog service.

- **CONFIG\_SERVICE\_WDOG\_DEBUG\_TIMEOUT\_SECS**

Determines the periodicity (in seconds) that Watchdog debug messages will be output by the HSS.

- **CONFIG\_SERVICE\_WDOG\_ENABLE\_E51**

Enables the watchdog for the E51 monitors heart in addition to the U54s, protecting the operation of the HSS itself.

When the E51 watchdog is enabled, the HSS will periodically write to the Watchdog to refresh it and prevent it from firing. If, for some reason, the E51 heart locks up or crashes and the E51 watchdog is enabled, this will always reset the entire system.

### **Watchdog Operation**

The watchdog hardware implements down counters. A refresh-forbidden window can be created by configuring the watchdog Maximum Value up to which Refresh is Permitted (MVRP).

- When the current value of the watchdog timer is greater than the MVRP value, refreshing the watchdog is forbidden. Attempting to refresh the watchdog timer in the forbidden window will assert a timeout interrupt.
- Refreshing the watchdog between the MVRP value and the Trigger Value (TRIG) will successfully refresh the counter and prevent the watchdog from firing.
- Once the watchdog timer value counts below the TRIG value, the watchdog will fire.

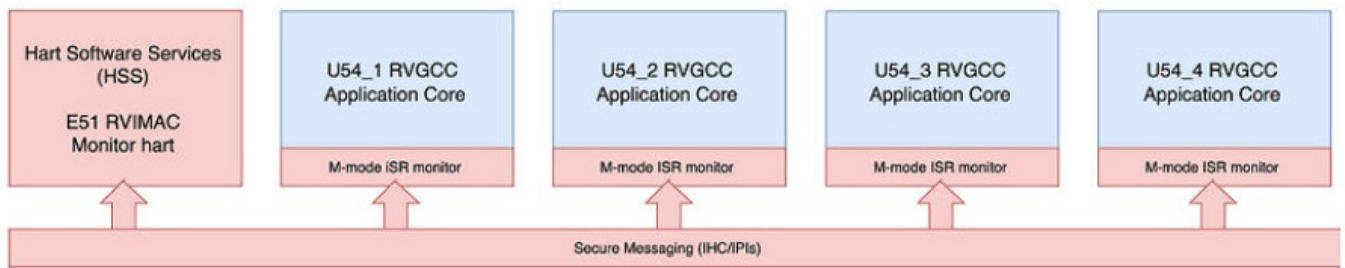
### **Watchdog State Machine**

- The watchdog state machine is very straightforward – starting up by configuring the watchdog for the E51, if enabled, then moving through an idle state into monitoring. Each time around the superloop, this monitoring state is invoked, which checks the status of each of the U54 watchdogs.
- The watchdog state machine interacts with the boot state machine to restart a hart (and any other harts that are in its boot set), if it detects that the hart has not managed to refresh its watchdog in time.

### **Lockdown Mode**

Normally (especially with AMP applications), it is expected that the HSS will stay resident in M-mode, on a U54, to allow per-context reboot (i.e reboot one context only, without full-chip reboot), and to allow the HSS to monitor health (ECCs, Lock Status Bits, Bus Errors, SBI errors, PMP violations, etc).

**Figure 4.1.** HSS Code Running on Each Hart in M-Mode



- In order to provide reboot capabilities on a per-AMP context basis (without requiring the entire system to reboot), the E51 normally has privileged memory access to the entire memory space of the system. However, there may be situations where this is not desirable, and the customer may prefer to restrict what the E51 HSS firmware does once the system has booted successfully. In this case, it is possible to put the HSS into lockdown mode once the U54 Application Harts have been booted.
- This can be enabled using the HSS Kconfig option `CONFIG_SERVICE_LOCKDOWN`.
- The lockdown service is intended to allow restriction of the activities of the HSS after it boots the U54 application Harts.

**Figure 4.2.** HSS Lockdown Mode

```

4.648937] boot_service(u54_2) :: [SetupPMPComplete] -> [ZeroInit]
4.656958] boot_service(u54_3) :: [SetupPMPComplete] -> [ZeroInit]
4.664978] boot_service(u54_4) :: [SetupPMPComplete] -> [ZeroInit]
4.672998] boot_service(u54_1) :: [ZeroInit] -> [Download]
4.680255] boot_service(u54_2) :: [ZeroInit] -> [Download]
4.687511] boot_service(u54_3) :: [ZeroInit] -> [Download]
4.694768] boot_service(u54_4) :: [ZeroInit] -> [Download]
4.702024] boot_service(u54_1)::Processing boot image: "u-boot.bin"
4.709854] boot_service(u54_2) :: [Download] -> [Complete]
4.717110] boot_service(u54_3) :: [Download] -> [Complete]
4.724366] boot_service(u54_4) :: [Download] -> [Complete]
4.791966] boot_service(u54_1) :: [Download] -> [OpenSBIInit]
4.799509] boot_service(u54_1)::Registering domain "u-boot.bin" (hart mask 0x1e)
4.808580] boot_service(u54_1) :: [OpenSBIInit] -> [Wait]
4.817078] boot_service(u54_1) :: [Wait] -> [Complete]
4.872265] u54 State Change: [SBIHartInit] [Booting] [Booting] [Booting]
4.916950] boot_service(u54_1) :: [Complete] -> [Idle]
4.929935] boot_service(u54_2) :: [Complete] -> [Idle]
4.949413] boot_service(u54_3) :: [Complete] -> [Idle]
4.966886] boot_service(u54_4) :: [Complete] -> [Idle]
4.987796] lockdown_service :: [init] -> [active]
4.999731] lockdown:: E51 PMPs should be configured here...

[5.55396] lockdown:: from this point, only servicing watchdogs

```

Once the Lockdown mode starts, it stops all other HSS service state machines from running. It calls two weakly bound functions:

- `e51_pmp_lockdown()`, and
- `e51_lockdown()`

These functions are intended to be overridden by board-specific code. The first is a configurable trigger function to allow a BSP to customize locking the E51 out from the application payloads at this point. The weakly bound default implementation of this function is empty. The second is the functionality that is run from that point forward. The

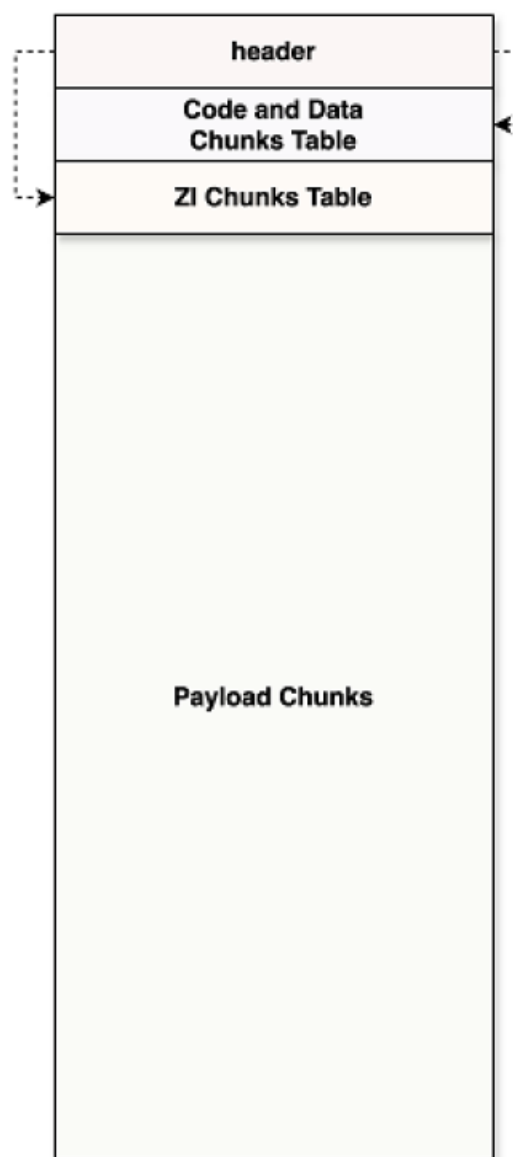
weakly-bound default implementation services the watchdog at this point in the E51, and will reboot if a U54 watchdog fires. For more information, see the HSS source code in the services/lockdown/lockdown\_service.c file.

## Appendix

### HSS payload.bin Format

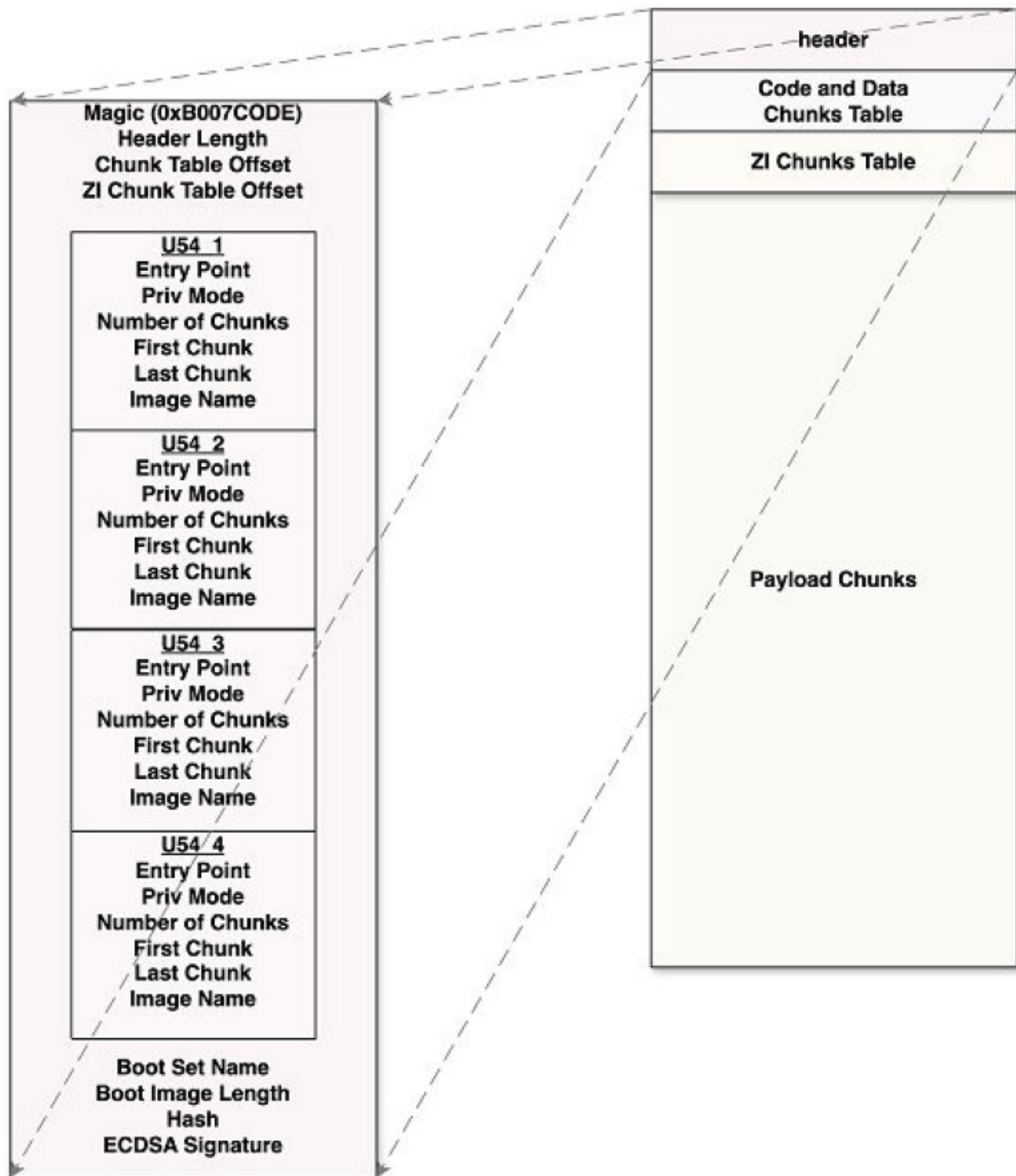
- This section describes the payload.bin file format and the image used by the HSS to boot PIC64GX SMP and AMP applications.
- The payload.bin is a formatted binary (Figure A.10) consisting of a head, various descriptor tables, and various chunks that contain the code and data sections of each part of the application workload. A chunk can be considered as an arbitrary-sized contiguous block of memory.

Figure A.10. payload.bin Format



The header portion (shown in Figure A.11) contains a magic value used to identify the payload.bin and some housekeeping information, along with details of the image intended to run on each of the U54 application codes. It describes how to boot each individual U54 hart, and the set of bootable images overall. In its housekeeping information, it has pointers to various tables of descriptors to allow the header size to grow.

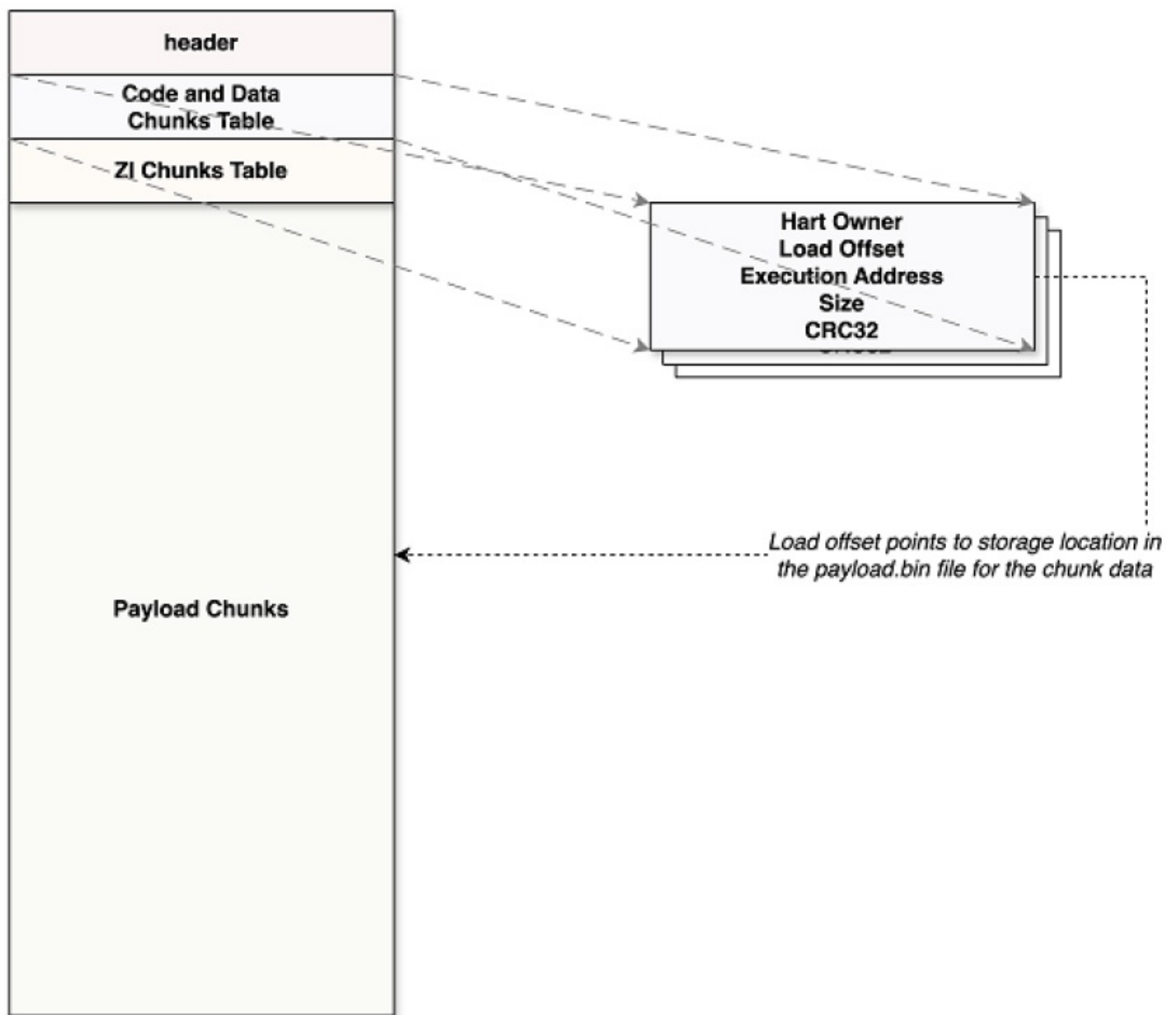
Figure A.11. payload.bin Header



- Code and initialized constant data are considered read-only and stored in a read-only section, which is pointed to by header descriptors.
- Non-zero initialized data variables are read-write data but have their initialization values copied from the read-only chunk at start-up. These are also stored in the read-only section.
- The read-only payload data section is described by a table of code and data chunk descriptors. Each chunk descriptor in this table contains a 'hart owner' (the main hart in the context it is targeted at), a load offset (offset within the payload.bin), and an execution address (destination address in PIC64GX memory), along with a size and checksum. This is shown in Figure A.12.

Figure A.12. Read-Only Chunk Descriptor and Payload Chunk Data

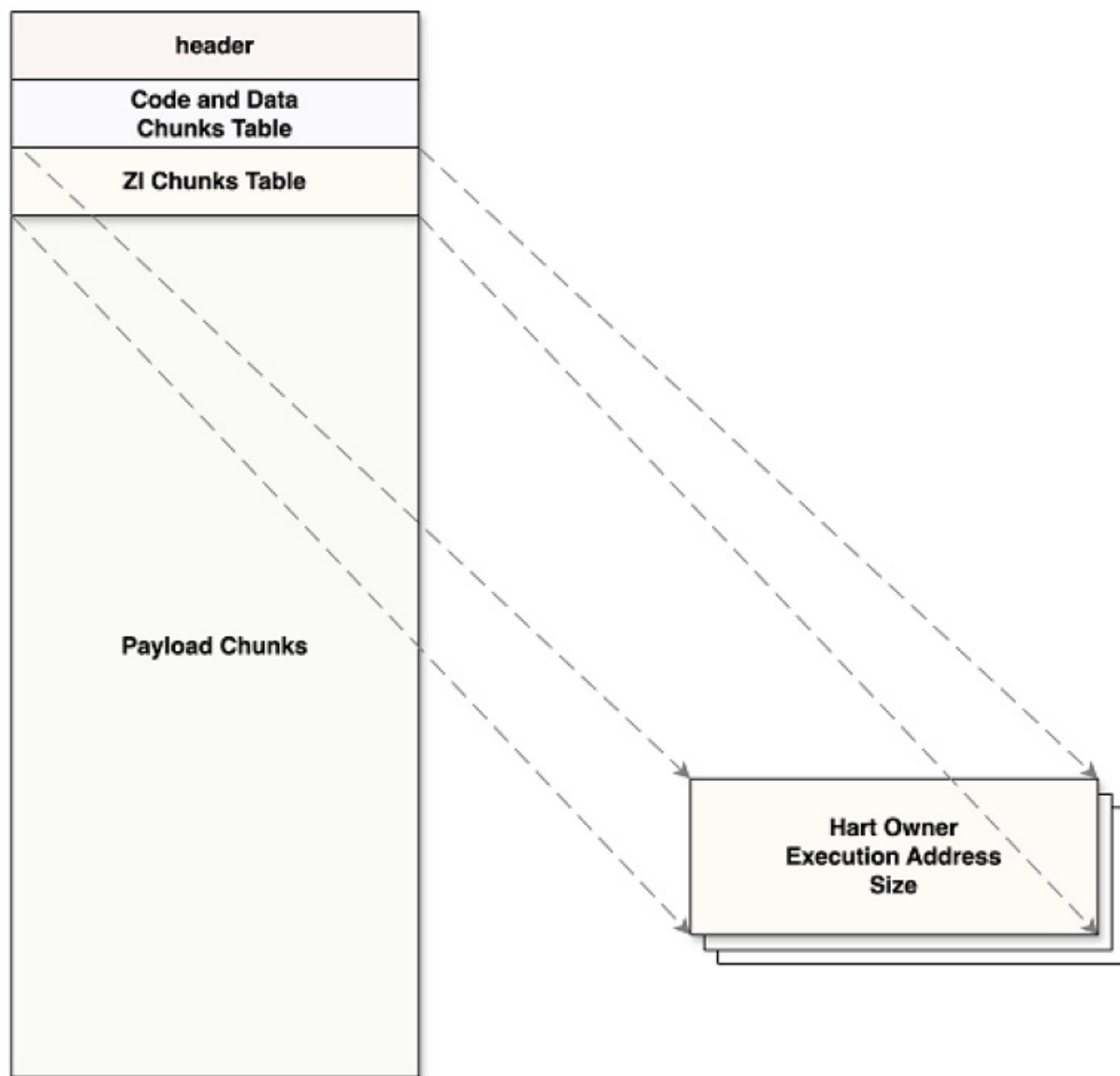




In addition to the aforementioned chunks, there are also chunks of memory corresponding to data variables that are initialized to zero. These are not stored as data in the payload.bin, but instead are a special set of zero-initialized chunk descriptors, which specify an address and length of RAM to set to zero during startup. This is shown in Figure A.13.

Figure A.13. ZI Chunks



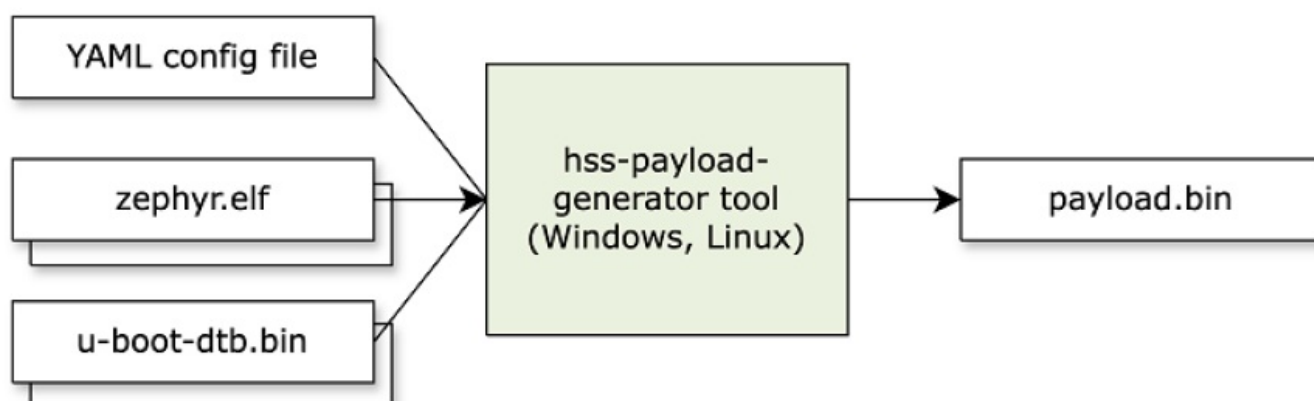


### **hss-payload-generator**

The HSS Payload Generator tool creates a formatted payload image for the Hart Software Service zero-stage bootloader on PIC64GX, given a configuration file and a set of ELF files and/or binaries. The configuration file is used to map the ELF binaries or binary blobs to the individual application harts (U54s).

Figure B.14. hss-payload-generator Flow

**Figure B.14.** hss-payload-generator Flow



The tool does perform basic sanity checks on the structure of the configuration file itself and on the ELF images.

ELF images must be RISC-V executables.

## Example Run

- To run the hss-payload-generator tool with the sample configuration file and ELF files:  
`$ ./hss-payload-generator -c test/config.yaml output.bin`
- To print diagnostics about a pre-existing image, use:  
`$ ./hss-payload-generator -d output.bin`
- To enable secure boot authentication (via image signing), use -p to specify the location of an X.509 Private Key for the Elliptic Curve P-384 (SECP384r1):  
`$ ./hss-payload-generator -c test/config.yaml payload.bin -p /path/to/private.pem`

For more information, see the Secure Boot Authentication documentation.

## Config File Example

- First, we can optionally set a name for our image, otherwise, one will be created dynamically:  
`set-name: 'PIC64-HSS::TestImage'`
- Next, we'll define the entry point addresses for each hart, as follows:  
`hart-entry-points: {u54_1: '0x80200000', u54_2: '0x80200000', u54_3: '0xB0000000', u54_4: '0x80200000'}`

The ELF source images can specify an entry point, but we want to be able to support secondary entry points for harts if needed, for example, if multiple harts are intended to boot the same image, they might have individual entry points. To support this, we specify the actual entry point addresses in the configuration file itself.

We can now define some payloads (source ELF files, or binary blobs) that will be placed at certain regions in memory. The payload section is defined with the keyword `payloads`, and then a number of individual payload descriptors. Each payload has a name (path to its file), an owner-hart, and optionally 1 to 3 secondary harts.

Additionally, a payload has a privilege mode in which it will start execution. Valid privilege modes are `PRV_M`, `PRV_S` and `PRV_U`, where these are defined as:

- `PRV_M` Machine mode
- `PRV_S` Supervisor mode
- `PRV_U` User mode

## In the following example:

- `test/zephyr.elf` is assumed to be a Zephyr application that runs in `U54_3`, and expects to start in the `PRV_M` privilege mode.
- `test/u-boot-dtb.bin` is the Das U-Boot bootloader application, and it runs on `U54_1`, `U54_2` and `U54_4`. It expects to start in the `PRV_S` privilege mode.

## Important:

The output of U-Boot creates an ELF file, but typically it does not prepend the `.elf` extension. In this case, the binary created by `CONFIG_OF_SEPARATE` is used, which appends a device tree blob to the U-Boot binary.

Here is the example Payloads configuration file:

- **test/zephyr.elf:**

```
{exec-addr: '0xB0000000', owner-hart: u54_3, priv-mode: prv_m, skip-opensbi: true}
```

- **test/u-boot-dtb.bin:**

```
{exec-addr: '0x80200000', owner-hart: u54_1, secondary-hart: u54_2, secondary-hart: u54_4,priv-mode: prv_s}
```

**Important:**

Case only matters for the file path names, not the keywords. So, for instance, u54\_1 is considered the same as U54\_1, and exec-addr is considered the same as EXEC-ADDR. If an.elf or .bin extension is present, it needs to be included in the configuration file.

- For a bare metal application that does not want to be concerned with OpenSBI, the option skip-opens, if true, will cause the payload on that hart to be invoked using a simple mret rather than an OpenSBI sbi\_init() call. This means the hart will start running the bare metal code irrespective of any OpenSBI HSM considerations. Note that this also means the hart cannot use ecalls to invoke OpenSBI functionality. The skip-opens option is optional and defaults to false.
- To allow a context warm reboot of another context, we can add the option allow-reboot: warm. To allow a context cold reboot of the entire system, we can add the option allow-reboot: cold. By default, without specifying allow-reboot, a context is only allowed to warm reboot itself.
- It is also possible to associate ancillary data with each payload, for example, a DeviceTree Blob (DTB) file, by specifying the ancillary data filename as follows:  

```
test/u-boot.bin: { exec-addr: '0x80200000', owner-hart: u54_1, secondary-hart: u54_2, secondary-hart: u54_3, secondary-hart: u54_4, priv-mode: prv_s, ancillary-data: test/pic64gx.dtb }
```
- This ancillary data will get included in the payload (placed straight after the main file in the executable space), and its address will be passed to OpenSBI in the next\_arg1 field (passed in the \$a1 register to the image at boot time).
- To prevent the HSS from automatically booting a context (for instance, if we instead want to delegate control of this to a context using remoteProc), use the skip-autoboot flag:  

```
test/zephyr.elf: {exec-addr: '0xB0000000', owner-hart: u54_3, priv-mode: prv_m, skip-opensbi: true, skip-autoboot: true}
```
- Finally, we can optionally override the names of individual payloads, using the payload-name option. For example:  

```
test/u-boot.bin: { exec-addr: '0x80200000', owner-hart: u54_1, secondary-hart: u54_2, secondary-hart: u54_3, secondary-hart: u54_4, priv-mode: prv_s, ancillary-data: test/pic64gx.dtb, payload-name: 'u-boot' }
```

Note that the Yocto and Buildroot Linux builders will build, configure, and run the hss-payload-generator as needed to generate application images. Additionally, the pic64gx-curiosity-kit-amp machine target in Yocto will generate an application image using the hss-payload-generator tool that demonstrates AMP, with Linux running on 3 harts and Zephyr running on 1 hart.

**Revision History**

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

Revision	Date	Description
A	07/2024	Initial Revision

## Microchip Information

### The Microchip Website

Microchip provides online support via our website at [www.microchip.com/](http://www.microchip.com/). This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Datasheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, a listing of seminars and events, listings of Microchip sales offices, distributors, and factory representatives

### Product Change Notification Service

- Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.
- To register, go to [www.microchip.com/pcn](http://www.microchip.com/pcn) and follow the registration instructions.

### Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative, or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: [www.microchip.com/support](http://www.microchip.com/support).

### Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products is strictly prohibited and may violate the Digital Millennium Copyright Act.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

## Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at [www.microchip.com/en-us/support/design-help/client-support-services](http://www.microchip.com/en-us/support/design-help/client-support-services).

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE NUMBER OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify, and hold harmless Microchip from all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motor bench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, TimeCesium, TimeHub, TimePictra, TimeProvider, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, EyeOpen, GridTime, IdealBridge, IGaT, In-Circuit Serial Programming, ICSP, INICnet, Intelligent Paralleling, IntelliMOS, Inter-Chip Connectivity, JitterBlocker, Knob-on-Display, MarginLink, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, mSiC, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, Power MOS IV, Power MOS 7, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAM-ICE, Serial Quad I/O, simple map, SimpliPHY, SmartBuffer, SmartHLS, SMART-I.S., storClad, SQL, SuperSwitcher, SuperSwitcher II, Switchtec, SynchroPHY, Total Endurance, Trusted Time, TSHARC, Turing, USBCheck, VariSense, VectorBlox, VeriPHY, ViewSpan, WiperLock, XpressConnect, and ZENA

are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

- SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.
- The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.
- GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are the property of their respective companies. © 2024, Microchip Technology Incorporated and its subsidiaries. All Rights Reserved.

- **ISBN:** 978-1-6683-4890-1

### Quality Management System


For information regarding Microchip's Quality Management Systems, please visit [www.microchip.com/quality](http://www.microchip.com/quality).

### Worldwide Sales and Service









AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
<b>Corporate Office</b>  2355 West Chandler Blvd. Chandler, AZ 85224-6199  Tel: <a href="tel:480-792-7200">480-792-7200</a>  Fax: <a href="tel:480-792-7277">480-792-7277</a>  Technical Support: <a href="http://www.microchip.com/support">www.microchip.com/support</a>  Web Address: <a href="http://www.microchip.com">www.microchip.com</a>  <b>Atlanta</b>  Duluth, GA  Tel: <a href="tel:678-957-9614">678-957-9614</a>  Fax: <a href="tel:678-957-1455">678-957-1455</a>  <b>Austin, TX</b>  Tel: <a href="tel:512-257-3370">512-257-3370</a>  <b>Boston</b>  Westborough, MA Tel: <a href="tel:774-760-0087">774-760-0087</a>  Fax: <a href="tel:774-760-0088">774-760-0088</a>  <b>Chicago</b>	<b>Australia – Sydney</b>  Tel: 61-2-9868-6733  <b>China – Beijing</b>  Tel: 86-10-8569-7000  <b>China – Chengdu</b>  Tel: 86-28-8665-5511  <b>China – Chongqing</b>  Tel: 86-23-8980-9588  <b>China – Dongguan</b>  Tel: 86-769-8702-9880  <b>China – Guangzhou</b>  Tel: 86-20-8755-8029	<b>India – Bangalore</b>  Tel: 91-80-3090-4444  <b>India – New Delhi</b>  Tel: 91-11-4160-8631  <b>India – Pune</b>  Tel: 91-20-4121-0141  <b>Japan – Osaka</b>  Tel: 81-6-6152-7160  <b>Japan – Tokyo</b>  Tel: 81-3-6880-3770	<b>Austria – Wels</b>  Tel: 43-7242-2244-39  Fax: 43-7242-2244-393  <b>Denmark – Copenhagen</b>  Tel: 45-4485-5910  Fax: 45-4485-2829  <b>Finland – Espoo</b>  Tel: 358-9-4520-820  <b>France – Paris</b>  Tel: 33-1-69-53-63-20  Fax: 33-1-69-30-90-79  <b>Germany – Garching</b>  Tel: 49-8931-9700  <b>Germany – Haan</b>  Tel: 49-2129-3766400  <b>Germany – Heilbronn</b>  Tel: 49-7131-72400  <b>Germany – Karlsruhe</b>  Tel: 49-721-625370

<p>Itasca, IL</p> <p>Tel: <a href="tel:630-285-0071">630-285-0071</a></p> <p>Fax: <a href="tel:630-285-0075">630-285-0075</a></p> <p><b>Dallas</b></p> <p>Addison, TX</p> <p>Tel: <a href="tel:972-818-7423">972-818-7423</a></p> <p>Fax: <a href="tel:972-818-2924">972-818-2924</a></p> <p><b>Detroit</b></p> <p>Novi, MI</p> <p>Tel: <a href="tel:248-848-4000">248-848-4000</a></p> <p><b>Houston, TX</b></p> <p>Tel: <a href="tel:281-894-5983">281-894-5983</a></p> <p><b>Indianapolis</b></p> <p>Noblesville, IN Tel: <a href="tel:317-773-8323">317-773-8323</a></p> <p>Fax: <a href="tel:317-773-5453">317-773-5453</a></p> <p>Tel: <a href="tel:317-536-2380">317-536-2380</a></p> <p><b>Los Angeles</b></p> <p>Mission Viejo, CA Tel: <a href="tel:949-462-9523">949-462-9523</a></p> <p>Fax: <a href="tel:949-462-9608">949-462-9608</a></p> <p>Tel: <a href="tel:951-273-7800">951-273-7800</a></p> <p><b>Raleigh, NC</b></p> <p>Tel: <a href="tel:919-844-7510">919-844-7510</a></p> <p><b>New York, NY</b></p> <p>Tel: <a href="tel:631-435-6000">631-435-6000</a></p> <p><b>San Jose, CA</b></p> <p>Tel: <a href="tel:408-735-9110">408-735-9110</a></p> <p>Tel: <a href="tel:408-436-4270">408-436-4270</a></p> <p><b>Canada – Toronto</b></p> <p>Tel: <a href="tel:905-695-1980">905-695-1980</a></p> <p>Fax: <a href="tel:905-695-2078">905-695-2078</a></p>	<p><b>China – Hangzhou</b></p> <p>Tel: 86-571-8792-8115</p> <p><b>China – Hong Kong SAR</b></p> <p>Tel: 852-2943-5100</p> <p><b>China – Nanjing</b></p> <p>Tel: 86-25-8473-2460</p> <p><b>China – Qingdao</b></p> <p>Tel: 86-532-8502-7355</p> <p><b>China – Shanghai</b></p> <p>Tel: 86-21-3326-8000</p> <p><b>China – Shenyang</b></p> <p>Tel: 86-24-2334-2829</p> <p><b>China – Shenzhen</b></p> <p>Tel: 86-755-8864-2200</p> <p><b>China – Suzhou</b></p> <p>Tel: 86-186-6233-1526</p> <p><b>China – Wuhan</b></p> <p>Tel: 86-27-5980-5300</p> <p><b>China – Xian</b></p> <p>Tel: 86-29-8833-7252</p> <p><b>China – Xiamen</b></p> <p>Tel: 86-592-2388138</p> <p><b>China – Zhuhai</b></p> <p>Tel: 86-756-3210040</p>	<p><b>Korea – Daegu</b></p> <p>Tel: 82-53-744-4301</p> <p><b>Korea – Seoul</b></p> <p>Tel: 82-2-554-7200</p> <p><b>Malaysia – Kuala Lumpur</b></p> <p>Tel: 60-3-7651-7906</p> <p><b>Malaysia – Penang</b></p> <p>Tel: 60-4-227-8870</p> <p><b>Philippines – Manila</b></p> <p>Tel: 63-2-634-9065</p> <p><b>Singapore</b></p> <p>Tel: 65-6334-8870</p> <p><b>Taiwan – Hsin Chu</b></p> <p>Tel: 886-3-577-8366</p> <p><b>Taiwan – Kaohsiung</b></p> <p>Tel: 886-7-213-7830</p> <p><b>Taiwan – Taipei</b></p> <p>Tel: 886-2-2508-8600</p> <p><b>Thailand – Bangkok</b></p> <p>Tel: 66-2-694-1351</p> <p><b>Vietnam – Ho Chi Minh</b></p> <p>Tel: 84-28-5448-2100</p>	<p><b>Germany – Munich</b></p> <p>Tel: 49-89-627-144-0</p> <p>Fax: 49-89-627-144-44</p> <p><b>Germany – Rosenheim</b></p> <p>Tel: 49-8031-354-560</p> <p><b>Israel – Hod Hasharon</b></p> <p>Tel: 972-9-775-5100</p> <p><b>Italy – Milan</b></p> <p>Tel: 39-0331-742611</p> <p>Fax: 39-0331-466781</p> <p><b>Italy – Padova</b></p> <p>Tel: 39-049-7625286</p> <p><b>Netherlands – Drunen</b></p> <p>Tel: 31-416-690399</p> <p>Fax: 31-416-690340</p> <p><b>Norway – Trondheim</b></p> <p>Tel: 47-72884388</p> <p><b>Poland – Warsaw</b></p> <p>Tel: 48-22-3325737</p> <p><b>Romania – Bucharest</b></p> <p>Tel: 40-21-407-87-50</p> <p><b>Spain – Madrid</b></p> <p>Tel: 34-91-708-08-90</p> <p>Fax: 34-91-708-08-91</p> <p><b>Sweden – Gothenburg</b></p> <p>Tel: 46-31-704-60-40</p> <p><b>Sweden – Stockholm</b></p> <p>Tel: 46-8-5090-4654</p> <p><b>UK – Wokingham</b></p> <p>Tel: 44-118-921-5800</p> <p>Fax: 44-118-921-5820</p>
---	---	---	--

## Documents / Resources

	<p><a href="#">MICROCHIP PIC64GX 64-Bit RISC-V Quad-Core Microprocessor</a> [pdf] User Guide PIC64GX, PIC64GX 64-Bit RISC-V Quad-Core Microprocessor, 64-Bit RISC-V Quad-Core Microprocessor, RISC-V Quad-Core Microprocessor, Quad-Core Microprocessor, Microprocessor</p>
---	---

## References

-  [Empowering Innovation | Microchip Technology](#)
-  [Microchip Lightning Support](#)
-  [riscv-sbi-doc/riscv-sbi.adoc at master · riscv-non-isa/riscv-sbi-doc · GitHub](#)
-  [riscv-sbi-doc/riscv-sbi.adoc at master · riscv-non-isa/riscv-sbi-doc · GitHub](#)
-  [kernel.org/doc/Documentation/ABI/testing/sysfs-kernel-reboot](#)
-  [kernel.org/doc/Documentation/admin-guide/kernel-parameters.txt](#)
-  [Client Support Services | Microchip Technology](#)
-  [Microchip Lightning Support](#)
- [User Manual](#)

[Manuals+](#), [Privacy Policy](#)

This website is an independent publication and is neither affiliated with nor endorsed by any of the trademark owners. The "Bluetooth®" word mark and logos are registered trademarks owned by Bluetooth SIG, Inc. The "Wi-Fi®" word mark and logos are registered trademarks owned by the Wi-Fi Alliance. Any use of these marks on this website does not imply any affiliation with or endorsement.