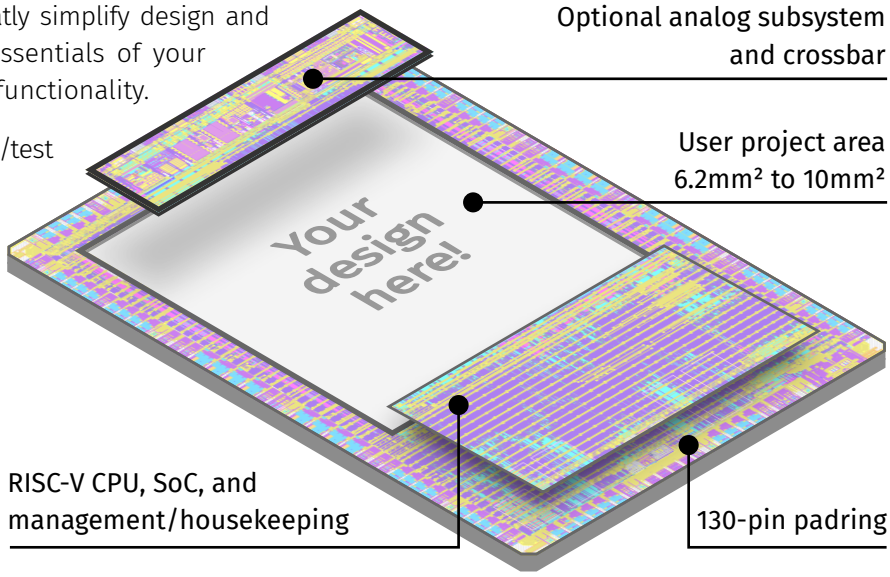
chipIgnite FRIGATE SoC Chip Templates

Efabless chipIgnite and **Frigate** templates greatly simplify design and fabrication of complex ASICs. Focus on the essentials of your design, and simply leverage Frigate's rich SoC functionality.

Frigate's **on-chip SoC** microcontroller/management/test framework & blank silicon design area offer a standard padring, **digital peripherals array** & optional **analog** instrumentation for proprietary, private, commercial, or open-source designs. Incorporate any or all of Frigate's many options...

Available as bare dice, QFN-64, or BGA-130.



RISC-V CPU

- Hazard3 32-bit RISC-V RV32IMAC core, rated to 40 MHz
- On-chip SRAM for data/scratch/program storage
- QSPI flash controller with XIP support for ext. ROM
- 512-byte instruction cache
- JTAG debugging
- DMA controller

Digital Peripherals

- 66 GPIOs:
  - 56 x GPIO (user-hard-set power-on modes, soft override)
  - 8 x soft-config GPIO (power-on system functions)
  - 2 x SIO (Special IO) ports (optional differential pair)
  - Reconfigurable/shared pin functions
  - Individually switchable: user-project-vs.-SoC control
- 6 x 32-bit timer/counter/PWM modules
- 1 x USB CDC Full-speed ("virtual COM port") device
- 4 x UART
- 2 x SPI master controller
- 2 x I²C-compatible master
- 1 x CAN FD controller
- 1 x PSRAM xSPI controller
- 1 x I²S master controller
- 1 x WDT

Clocking

Five clock sources:

- RCO16M: On-chip 16 MHz RC Oscillator
- RCO500K: On-chip 500 kHz RC Oscillator (default at reset)
- HSXO: High Speed Crystal Oscillator (4-16 MHz)
- LSXO: Low power Crystal Oscillator (32.768 kHz)
- XCLK: External clock source (up to 48 MHz)

DLL can be enabled for clock scaling from sources of: HSXO, XCLK, or RCO16M.

Analog Peripherals & Crossbar

- Up to 66 analog IOs (digital GPIO pin-shared, 16 OVT)
- 2 x Additional analog bare pads
- 1 x Audio DAC
- 3 x DACs (8-bit, two voltage RDACs, one current DAC)
- 2 x ADCs (12-bit 1 Msps, SAR)
- 1 x On-chip temperature sensor
- 2 x Instrumentation amplifiers (preamps)
- 4 x High-GBW op-amps
- 4 x 8-bit rheostats
- 2 x Voltage references
- 2 x Current biases
- 1 x High-precision comparator
- 1 x ULP (ultra-low-power) comparator
- 2 x SIO (Special IO) ports with programmable low-power comparator option

User Project Area

- 6.2mm² to 10mm² blank design area
- AHB-Lite interface from on-chip CPU/SoC
- 66 x Direct GPIO connections
- 16 x Direct analog IO connections
- Direct analog connections to several analog peripherals
- 2 x 1.8V digital/core power domains
- 2 x analog power domains (1.8V-5.5V)
- 16 x IRQ to CPU

Housekeeping

Int./external-access SoC control/diagnostics/debug module.

- Front door SPI interface connected to padframe
- Back door APB interface connected to SoC
- Allows pass-through mode to access/program external (firmware) SPI memory

© 2025 Efabless Corporation - efabless.com

efabless.com

Table of Contents

1. Revision History 5

2. Frigate Overview

- 2.1. What is Frigate? 6
- 2.2. What's included? 6
- 2.3. What can I use it for? 6
- 2.4. How to use chipIgnite with Frigate? 7

3. Features of Frigate

- 3.1. Frigate as a framework and a system 8
- 3.2. Frigate Functional Block Diagram 9
- 3.3. Frigate Padframe General Features 10
  - 3.3.1. Frigate User Project Wrapper 10
  - 3.3.2. Housekeeping SPI 10
  - 3.3.3. GPIO (General Purpose Input/Output pins) 11
  - 3.3.4. Clocking and Resetting 12
  - 3.3.5. POR (Power-On Reset) Module 12
  - 3.3.6. Power Domains 12
- 3.4. Frigate Management SoC Features 13
  - 3.4.1. RISC-V CPU Core (Hazard3-based) 14
  - 3.4.2. JTAG Debugging 14
  - 3.4.3. AHB-Lite User Bus 15
  - 3.4.4. UARTs (Universal Asynchronous Receiver-Transmitter) 15
  - 3.4.5. PSRAM Controller 16
  - 3.4.6. IRQs 16
  - 3.4.7. CAN FD Controller 17
  - 3.4.8. TMR32 Counter/Timer/PWM 18
  - 3.4.9. WDT (Watch-Dog Timer) 18
  - 3.4.10. SPI (Serial Peripheral Interface) 19
  - 3.4.11. I²C-compatible Serial Control/Data Interface 20
  - 3.4.12. I²S Serial Audio Interface 21
  - 3.4.13. USB CDC ("Virtual COM Port") Peripheral Device 21
  - 3.4.14. DMA Controller (DMAC) 22
  - 3.4.15. Flash Writer 23

PRELIMINARY

© 2025 Efabless Corporation

- 3.4.16. System Timer 23
- 3.5. Frigate Analog Peripherals 23
  - 3.5.1. Analog Crossbar Fabric 23
  - 3.5.2. Audio DAC 23
  - 3.5.3. ADCs (12-bit 1 Msps, SAR) 24
  - 3.5.4. On-chip Temperature Sensor 24
  - 3.5.5. Instrumentation Amplifiers (Preamps) 24
  - 3.5.6. High-GBW Op-Amps 24
  - 3.5.7. 8-bit Rheostats 25
  - 3.5.8. Voltage References 25
  - 3.5.9. Current Biases 25
  - 3.5.10. 8-bit Voltage-mode DACs (RDACs) 25
  - 3.5.11. 8-bit Current-mode DAC (iDAC) 26
  - 3.5.12. High-precision Comparator 26
  - 3.5.13. ULP (Ultra-Low-Power) Comparator 26

4. User Project Wrapper 27

- 4.1. Overview 27
- 4.2. UPW IOs/ports into the overall chip 27
- 4.3. Firmware requirements 28

5. Frigate Pinouts

- 5.1. Frigate pins and functions 30

6. Clocking, Resetting, and DLL 37

- 6.1. Frigate clock and reset module block diagram 38
- 6.2. Power-on/Reset Behavior 39
- 6.3. Clock Monitoring 39
- 6.4. Delay-Locked Loop (DLL) 39
- 6.5. USB Clock 39

7. Uploading/Flashing Firmware 40

- 7.1. Firmware Transfer by JTAG 40
- 7.2. Firmware Flashing by HKSPI 40
- 7.3. Bootloader Firmware Flashing 41

8. Electrical Characteristics 42

- 8.1. Absolute Maximum Ratings 42

PRELIMINARY

© 2025 Efabless Corporation

- 8.2. Operating Conditions 43

9. Ordering Information 44

10. License 45

11. Legal Disclaimers 46

PRELIMINARY

© 2025 Efabless Corporation

Revision History

Version	Date	Description
0.8	2025-02	Initial pre-release version

PRELIMINARY

Frigate Overview

What is Frigate?

Designing and fabricating a custom ASIC is greatly simplified with Efabless chipIgnite by way of ready-made chip templates (sometimes called “frames” or “harnesses”), and **Frigate** is our most sophisticated, feature-rich, and production-ready template for the 130nm process node.

What’s included?

Frigate, as a chip template, comes equipped with a full, proven, ready-to-run **on-chip SoC** . Boasting a **RISC-V CPU** with a wide range of **digital peripherals** and **analog devices and instrumentation** , plus a plentiful variety of GPIO pins, Frigate can already be used as-is, as a **fully-featured microcontroller** .

Framed by the Frigate System-on-Chip and **standardized 128-pin padring** is the large **blank silicon design area of 6.256mm²** that is ready for your design (or even multiple combined and switchable projects).

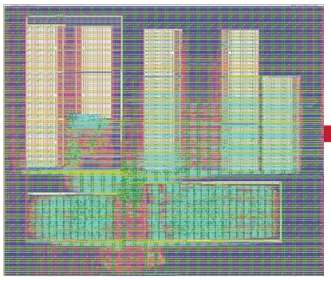
What can I use it for?

Supporting **digital, analog, and mixed-signal** designs and custom layouts that can integrate with (and optionally be managed by) the SoC and analog subsystem, Frigate provides a powerful framework for development, fabrication, and deployment of custom applications and highly-integrated solutions... whether for **proprietary/private/closed** designs, **commercial** projects, or an **open-source** design.

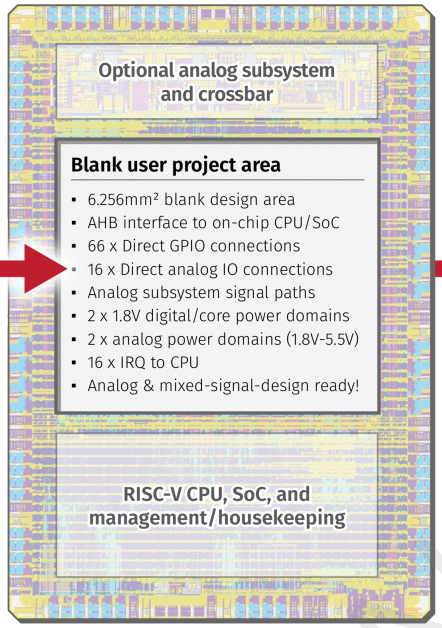
Frigate is suitable for both short-run prototypes and small-to-large-volume production runs. This is even openly offered right now – the Frigate template is available at no cost, allowing any designer to get started immediately and without the usual barriers of non-disclosure agreements or other legal limitations.

PRELIMINARY

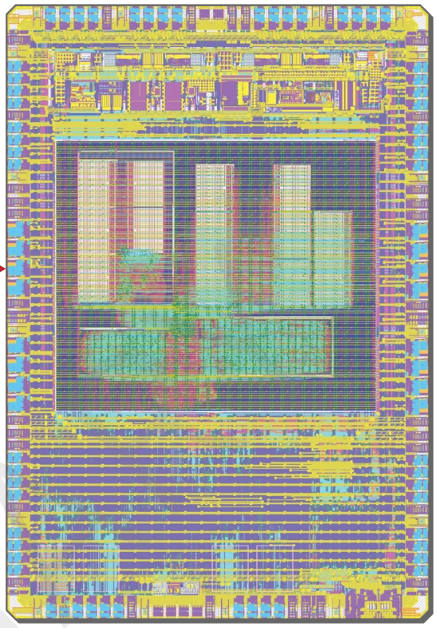
How to use chipIgnite with Frigate?



Your design:
Digital, analog, or mixed-signal. Created using open-source or commercial VLSI design tools



Frigate template:
Standard ready-made chip template with full System-on-Chip and Analog Subsystem, surrounding a large blank design area



Integrated chip:
GDSII design files built by the chipIgnite servers, validated and ready to be submitted for fabrication & (optional) packaging as real chips

With the straightforward and automated chipIgnite submission process, you can simply start with your view of the blank design area and focus on the essentials of your design and how (or whether) you will leverage the additional functionality offered by Frigate.

Submit your design for progress-checking & final tapeout with our cloud secure server vault. Our system auto-integrates your user design area into the template. Our team's high-grade support helps you get to tapeout successfully, and more quickly than with any alternative.

Use our framework for running tests, simulations, and full-chip verification (including simulating the entire Frigate SoC and how it behaves in combination with your design). Observing this behaviour of the SoC as a microcontroller/management/test framework interacting with your design gives you the full scope of how you can use it to your advantage, how you can plan to get the most out of testing the final product, and how you can expect it to function in the field.

PRELIMINARY

Features of Frigate

Frigate as a framework and a system

As a padframe, Frigate offers easy-to-connect standard features such as GPIO pins, power delivery (optionally with multple power domains), clocking, and reset logic (both a dedicated reset pin, and a POR circuit).

As an SoC , Frigate provides a wide range of optional functionality on-chip that is at your disposal and ready for any of bringing up your design, debugging and providing a diagnostic/maintenance interface, controlling different peripherals that your design can leverage, or even as a full microcontroller.

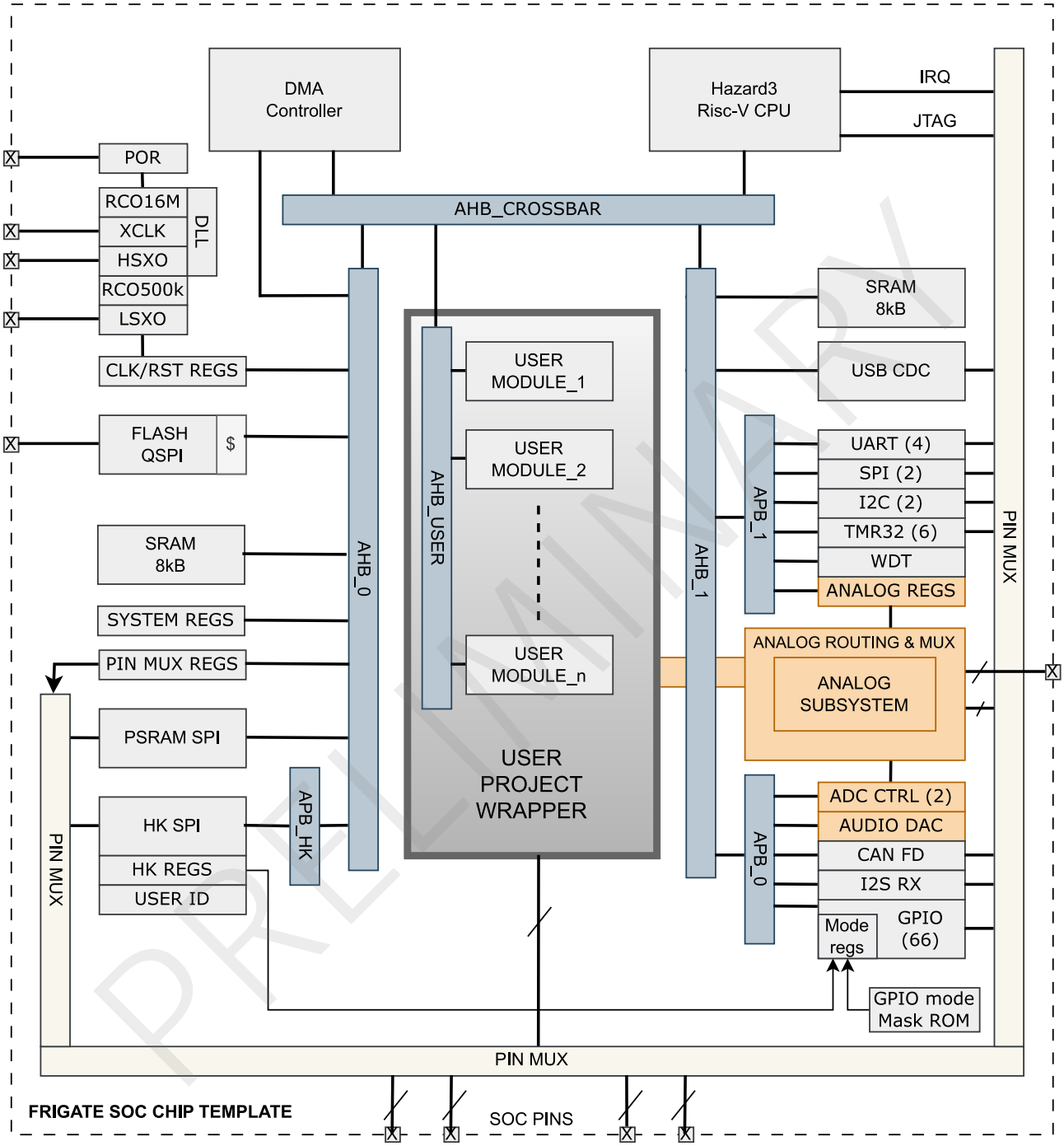
The blank silicon design area that is available to you as the designer can be used as your primary design space and system – simply supported by the Frigate SoC – or the other way around: as your own unique peripheral or accelerator (or set of both and many instances thereof) that supports and extends the function of the SoC.

Viewed as a microcontroller and SoC, Frigate includes a CPU with external firmware interface, on-chip RAM, and multiple industry-standard communications protocols and other interfaces, but also a sophisticated switching matrix of multiple analog devices that can be used discretely or dynamically chained together into other more-complex peripherals.

All Frigate chips are fabricated using a standardized 130-pad padring template (" Panamax "). Frigate will be offered with different packaging options **(TBA)** including BGA and bare dice.

PRELIMINARY

Frigate Functional Block Diagram



PRELIMINARY

Frigate Padframe General Features

These features are universally available to any chip based on the Frigate frame, even without specific use of the **Management SoC** .

Frigate User Project Wrapper

Frigate includes 6.256mm² of "user project wrapper" design area (2720µm x 2300µm). `user_project_wrapper` is what a user submits as a fixed-area GDS macro. It is then automatically integrated (by the Efabless chipIgnite submission process) into the rest of the Frigate chip harness to produce the final GDS that is submitted for fabrication. This design area, in the 130nm node, can accommodate in excess of 2.4 million transistors or 600,000 logic gate primitives.

The designer can choose whether the design they include in the user project wrapper will interface with the Frigate SoC (CPU and peripherals), the padframe GPIOs, the analog instrumentation/subsystem, the built-in clock sources, the PDN (Power Delivery Network) or any combination of these.

The user project wrapper provides **support for digital, analog, and mixed-signal projects** .

For more information, see the section: User Project Wrapper

Housekeeping SPI

The chip's power-on state defaults to assigning 4 GPIOs as a **Housekeeping SPI (HKSPI)** interface for an external SPI controller to assert debugging control over certain base configuration, debugging, and clocking of the chip.

This also includes **SPI pass-through** , able to be driven via HKSPI to take control (for reading/writing) of a firmware SPI ROM connected to the Management SoC.

PRELIMINARY

GPIO (General Purpose Input/Output pins)

Frigate provides **66 GPIO pins** , arranged as:

- 7 × 8-bit GPIO banks (`PA` .. `PG`) that are mask-programmed with a given power-up mode (according to the user's requirements at the time of submission for fabrication)
- 16 of the GPIO pads are Over-Voltage-Tolerant (OVT).
- 8 additional GPIOs that power-up with a system-reserved function for Housekeeping SPI (HKSPI) , debugging, and USB, but which can be reconfigured after power-up
- 2 additional "SIO" (Special IO) ports that offer multiple modes including programmable trip points, a differential pair, or as individual GPIOs.

All of these can be reconfigured at any time via firmware or Housekeeping SPI (HKSPI) for interchangeable use by the user's own digital logic, the Frigate CPU, or in some cases as analog connections.

Each GPIO pin offers the following functionality:

- Input synchronizer
- Input edge detector
- Direction control
- Edge and Level Interrupt generation

Frigate provides ability to configure GPIOs power-up state for each at design time.

For more information on GPIO modes and configuration, see: https://efabless.com/frigate-docs/gpio

PRELIMINARY

Clocking and Resetting

The Frigate SoC employs a clock and reset manager that provides the following features:

- Five clock sources:
  - RCO16M: On-chip 16 MHz RC Oscillator
  - RCO500K: On-chip 500 kHz RC Oscillator (enabled at reset)
  - HSXO: High Speed Crystal Oscillator (16-25 MHz)
  - LSXO: Low power Crystal Oscillator (32.768 kHz)
  - XCLK: External clock source (up to 48 MHz)
- All-digital Delay Locked Loop (DLL) for frequency multiplication for clocks from RCO16M, XCLK or HSXO.
- Glitch-less clock multiplexing allows energy-efficient smooth transition between different clock sources.
- Clock dividers allowing active clock frequencies from 250 kHz to 48 MHz
- Reset sources are: external reset, POR (Power-On Reset), or Housekeeping reset
- Reset synchronizer (always with the RCO500K)

See: Clocking, Resetting, and DLL

POR (Power-On Reset) Module

The power-on-reset circuit gates an enable signal which holds off chip functions until the primary power supply (`vddio`) and the digital power supply (`vccd0`) have reached sufficient and stable values. It is self-contained and depends on no other circuits.

Power Domains

Frigate provides 4 power domains in the User Project Wrapper : two intended as nominal 1.8V digital supplies; and two intended as analog supplies in the range 1.8V to 5.5V.

These are in addition to other dedicated power domains required by the Frigate SoC.

PRELIMINARY

Frigate Management SoC Features

The Management SoC's RISC-V CPU is built into the die area, adjacent the user project wrapper, and can be interfaced with your design, as well as externally to the chip.

The Frigate SoC includes the following peripherals and capabilities that can be optionally enabled/disabled on subsets of the GPIO pins, to make the SoC useful both as a general-purpose microcontroller and a specialized test/debug interface for your design...

Note

Typical user applications are expected to make use of the Frigate RISC-V CPU and SoC, which depends on off-chip memory for program storage, so an external SPI ROM is required for storing firmware that the CPU will read and execute from power-on, and which can optionally also be used for other data. Alternatively, firmware can be live-loaded and executed using an external controller.

See the section on Uploading/Flashing Firmware for more information.

Note

While it is not mandatory to use the RISC-V CPU, it is required for configuring and activating many of the extended capabilities of the SoC. If your intended project/application will not use the CPU and SoC, please contact Efabless for best practices on disabling it or otherwise minimizing its power consumption.

PRELIMINARY

RISC-V CPU Core (Hazard3-based)

The CPU core is a 32-bit Hazard3 -based RV32IMC RISC-V processor. Intended for use either as a microcontroller, general-purpose CPU, control or debug interface to the user design. It can be considered as a lightweight single-core bare-metal microcontroller, programmable in C or RISC-V assembly, and it comprises:

- **Dedicated firmware ROM QSPI master** for XIP loading of firmware code from an external QSPI memory into a local instruction cache.
- **512-byte instruction cache** arranged as 16 lines of 32 bytes.
- **16 kBytes** on-chip SRAM (extensible in the User Project Wrapper) for stack, scratch space/variables, memory allocation, or high-speed in-memory executable subroutines.
- **Interrupt and exception handling** including 32 system interrupts and 16 user interrupts.
- **Dedicated power domain** .
- **GPIO control** : Ability to reconfigure the 66 GPIOs, including taking over GPIOs as "management mode", either to activate SoC peripherals that have external interfaces, or for firmware to directly use some GPIOs.
- **Dedicated individual GPIO pin** always under CPU control for debugging or external device control.

The CPU core also has **access to a range of other SoC peripherals** as described below.

JTAG Debugging

Hazard3, along with its external debug components, implements version 0.13.2 of the RISC-V debug specification. It supports the following:

- Run/halt/reset control as required
- Abstract GPR access as required
- Program Buffer, 2 words plus impebreak
- Automatic trigger of abstract command (abstractauto) on data0 or Program Buffer access for efficient memory block transfers from the host
- Support for multiple harts (multiple Hazard3 cores) connected to a single Debug Module (DM)
- The hart array mask registers, for applying run/halt/reset controls to multiple cores simultaneously
- System Bus Access, either through a dedicated AHB5 manager interface, or multiplexed with a processor load/store port
- An instruction address trigger unit (hardware breakpoints)

For more details about the JTAG Debugging, see: https://github.com/Wren6991/Hazard3/blob/stable/doc/hazard3.pdf and https://riscv.org/wp-content/uploads/2024/12/riscv-debug-release.pdf

PRELIMINARY

AHB-Lite User Bus

The User Project Wrapper has an incoming **AHB-Lite master bus** interface from the CPU. This includes the `HCLK` and `HRESETn` signals. The AHB-Lite bus enables the user project to optionally implement one or more peripheral devices – ranging from simple responders up to complex high-performance accelerators and device/controller/memory arrays – that can be directly and efficiently be accessed by the CPU, as an extension of its address space.

UARTs (Universal Asynchronous Receiver-Transmitter)

The Frigate SoC includes 4 × UART (Universal Asynchronous Receiver-Transmitter) units, each offering a full-duplex, asynchronous communication channel that communicates with peripheral devices and personal computers through serial protocols such as RS-232, RS-485, and LIN. The UARTs have the following features:

- A configurable frame format:
  - Data bits could vary from 5 to 9 bits
  - Even, odd, sticky, or no-parity bit generation/detection
  - One or Two stop bit generation
- Line-break detection
- Configurable receiver timeout
- Loopback capability for testing/debugging
- Glitch Filter on the RX pin
- Matching received data detection
- 16-byte TX and RX FIFOs with programmable thresholds
- 16-bit prescaler (PR) for programmable baud rate generation
- Ten interrupt sources:
  - RX FIFO is full
  - TX FIFO is empty
  - RX FIFO level is above the set threshold
  - TX FIFO level is below the set threshold
  - Line break detection
  - Receiver data match
  - Frame Error
  - Parity Error
  - Overrun
  - Receiver timeout

For more details about the Frigate UARTs, see: https://efabless.com/frigate-docs/uart

PRELIMINARY

PSRAM Controller

The Frigate SoC includes a PSRAM Controller designed to interface with Quad I/O SPI (QSPI) PSRAM (Pseudostatic RAM). This means the SoC can easily control external PSRAM chip(s) and use them as external RAM through an SPI interface that is equivalent to that used by SPI Flash ROM chips, but typically with faster write access, comparable read rates, and without the endurance constraints of Flash ROM.

The key features of the PSRAM controller in the Frigate SoC are:

- Support for SPI, QSPI, and QPI modes for flexible memory access.
- Support for PSRAM or serial SRAM up to 8MB.
- Memory-mapped into the CPU address space for convenient and transparent random access from SoC CPU code, enabling the CPU addressable RAM to be extended by up to 8MB.
- Programmable configuration for:
  - Read/write commands
  - Number of wait states
  - Quad I/O entry and exit commands

For more details about the PSRAM Controller, see: https://efabless.com/frigate-docs/psram

IRQs

Frigate includes a range of 48 IRQ sources allowing the CPU to respond to a variety of events and external signal triggers.

The types of IRQ sources include:

- GPIO pin sensitivity
- Peripherals that drive/answer external activities (e.g. UART, USB, SPI, I²C, I²S, and CAN Bus events)
- Configurable timers, including WDT
- DMA controller
- ADC and Audio DAC events
- 16 User Project Wrapper -driven IRQs
- An IRQ that can be triggered via Housekeeping

For more details about IRQ configuration and ISRs, see: https://efabless.com/frigate-docs/irq

PRELIMINARY

CAN FD Controller

The Frigate SoC supports CAN Bus using a CTU CAN FD interface, which implements the CAN FD protocol as specified by ISO11898-1. This interface supports high-speed communication for automotive and industrial applications, offering flexibility with both ISO and non-ISO CAN FD protocols. It provides advanced features such as timestamping and time-triggered transmission, useful for time-sensitive applications. The CTU CAN FD also offers various operational modes, including loopback, bus monitoring, and self-test modes, making it suitable for testing and diagnostics.

The interface has the following features:

- Compliance with ISO11898-1:2015
- A RX buffer FIFO with 32 to 4096 words, supporting up to 204 CAN FD frames with 64 bytes of data each
- 2 to 8 TX buffers, with one CAN FD frame per TX buffer
- A 32-bit slave memory interface supporting APB, AHB, and RAM-like interfaces
- Support for ISO and non-ISO CAN FD protocols
- Operational modes including loopback, bus monitoring, ACK forbidden, self-test, and restricted operation modes.
- Interrupt capabilities for efficient event handling, including:
  - Frame received
  - Frame transmitted
  - Error warning limit
  - Data overrun, with a requirement to clear the DOR status to avoid re-triggering
  - Fault confinement state changed
  - Arbitration lost
  - Bus error
  - Overload frame
  - RX buffer full
  - Bit rate shifted
  - RX buffer not empty, which can be re-triggered if the buffer is not read
  - TX buffer hardware command interrupt, triggered when a hardware command changes the TX buffer state

For more details about the CAN FD Controller, see: https://efabless.com/frigate-docs/can

PRELIMINARY

TMR32 Counter/Timer/PWM

The Frigate SoC includes 6 × 32-bit Counter/Timer/PWM (TMR32) units. Each of the 6 units has the following features:

- 32-bit prescaler.
- Up Counting, Down Counting and Up/Down Counting.
- One-shot and Periodic.
- Two independent PWM channels with two compare registers.
- Optional PWM signal inversion.
- Configurable PWM dead time/band to generate PWM signals such as those required by a half-H bridge driver.
- Fault handling.

For details about the TMR32 units, see: https://efabless.com/frigate-docs/tmr32

WDT (Watch-Dog Timer)

The Frigate SoC includes a simple 32-bit Watch-Dog Timer (WDT). It can be used to interrupt the CPU to:

- Indicate/escape a software malfunction.
- Wake up the CPU after a certain sleep time.

The WDT is a simple 32-bit down-counter which is decremented every system clock cycle. When it reaches zero, the counter is loaded with a value stored in a load register and the `TO` (Timeout) flag is raised.

The WDT can be enabled/disabled. When it is disabled, the counter is primed with a load register value.

For more details about the WDT, see: https://efabless.com/frigate-docs/wdt

PRELIMINARY

SPI (Serial Peripheral Interface)

The Frigate SoC includes 2 × single-channel SPI master units. The SPI (Serial Peripheral Interface) bus is a high-speed synchronous communication protocol used by the SoC (as master) for control of, and data transfer with, external peripherals.

Each Frigate SPI master controller offers:

- Managing communication with SPI devices such as sensors, displays, and memory modules.
- Flexibility in controlling data transfer modes, clock speeds, and data sizes.
- A programmable prescaler for flexible clock speed control, allowing precise adjustment of the SPI clock to match the requirements of the connected peripherals.
- FIFO buffers for both transmit and receive operations, to ensure smooth operation. These FIFOs prevent data loss by temporarily holding data when the system or bus is busy, which helps to avoid overflow and ensures efficient data transfer.

The following interrupts are supported by the SPI master controllers:

- Transmit FIFO is Empty
- Transmit FIFO is Full
- Receive FIFO is Empty
- Receive FIFO is Full
- Transmit FIFO level is Below Threshold
- Receive FIFO level is Above Threshold

For more details about the SPI master controllers, see: https://efabless.com/frigate-docs/spi

PRELIMINARY

I²C-compatible Serial Control/Data Interface

The Frigate SoC includes 2 × I²C **compatible** (not fully-compliant) master controllers, each enabling Frigate to both transmit and receive data with other I²C devices on the bus.

The Inter-Integrated Circuit (I²C) bus is a two-wire serial communication protocol that facilitates bi-directional data transfer. It uses a Serial Data Line (SDA) for data transmission and a Serial Clock Line (SCL) to synchronize communication between devices. The I²C bus is commonly used to interface with a variety of external I²C devices, such as serial memory (RAMs and ROMs), networking devices, LCD displays, tone generators, and more.

Each Frigate I²C-compatible master controller offers:

- A programmable prescaler for flexible clock speed control.
- FIFO buffers for commands, reads, and writes. The use of FIFOs ensures no data loss during transfer by temporarily holding data when the system or bus is busy, allowing for smoother operation and preventing data overflow.

These controllers also generate several interrupt sources to signal key events during communication. These include:

- An interrupt when the module starts performing an I²C operation.
- An interrupt when the module takes control of the active bus.
- An interrupt when the bus is active.
- An interrupt for a missed slave ACK.
- Interrupts related to the status of the FIFOs:
  - Command FIFO empty
  - Command FIFO full
  - Command FIFO overflow (write 1 to clear)
  - Write FIFO empty
  - Write FIFO full
  - Write FIFO overflow (write 1 to clear)
  - Read FIFO empty
  - Read FIFO full

For more details about the I²C-compatible master controllers, see: https://efabless.com/frigate-docs/i2c

PRELIMINARY

I²S Serial Audio Interface

The Frigate SoC includes an I²S master receiver.

The Inter-IC Sound (I²S or I2S) protocol is a digital audio interface designed for high-quality sound transmission between electronic devices. I²S operates with three main signals: Serial Data (SD), Serial Clock (SCK), and Word Select (WS), ensuring synchronized data transmission. Unlike protocols like SPI, I²S is optimized for audio, providing precise timing and channel separation, making it ideal for hi-fi audio systems, digital microphones, and music streaming applications.

The Frigate I²S master receiver has the following features:

- 32 × 32-bit receive FIFO
- Sample Size selection
- Sample Format Selection
- Left channel, Right channel only or Stereo
- Programmable prescaler
- Automatic Voice Activity Detection (VAD).

For more details about I²S in Frigate, see: https://efabless.com/frigate-docs/i2s

USB CDC ("Virtual COM Port") Peripheral Device

The Frigate SoC includes a USB Communications Device Class (CDC) interface, a standardized USB class used to facilitate communication between a USB device and a host, typically for serial communication. It is commonly used in applications such as modems, serial adapters, and devices that emulate traditional serial ports over USB. The CDC protocol supports the Abstract Control Model (ACM) subclass, which allows devices to manage communication through serial-like control lines and provide necessary notifications, such as data availability or line state changes.

The interface operates at full-speed (12 Mbps), making it suitable for moderate data transfer rates, and it allows the USB device to be recognized as a virtual COM port on the host, enabling users to interact with it using traditional serial software. For proper operation of the USB CDC interface on the SoC, the USB clock (`usb_clk`) should be set to operate at 48 MHz, while the system clock (`clk`) should operate at half of this frequency, 24 MHz. Refer to Clocking, Resetting, and DLL for more details on how to configure these clocks.

For more details about the Frigate USB CDC, see: https://efabless.com/frigate-docs/usb

PRELIMINARY

DMA Controller (DMAC)

The Frigate SoC is equipped with a Direct Memory Access (DMA) controller, designed to offload data transfer tasks from the primary processor, enhancing overall system efficiency. By enabling the DMA controller to handle data transfers between memory and peripherals autonomously, the processor is freed up for other tasks, optimizing bus bandwidth utilization and reducing processing overhead.

Key Features of the DMA Controller:

- **Configurable Source and Destination Addresses** : The DMA controller allows flexible configuration of both source and destination addresses for data transfers.
- **Block Transfer Configuration** : You can specify the block size and the number of blocks to be transferred, offering fine-grained control over the transfer process.
- **Support for Multiple Data Sizes** : The DMA supports data transfers in 8-bit, 16-bit, and 32-bit formats, allowing it to handle a wide range of data types.
- **Configurable Address Increments** : The DMA controller can be configured to increment the source and destination addresses by byte, half-word, or word. Alternatively, no address increment can be applied during transfers.
- **Peripheral Interrupt-Driven Transfers** : The DMA controller can be set to wait for specific peripheral interrupts before initiating data transfers, ensuring synchronization with external devices.

The DMA controller can be triggered by the following interrupt sources:

- **IRQ 0** : USB
- **IRQ 1** : UART0
- **IRQ 2** : UART1
- **IRQ 3** : SPI
- **IRQ 4** : I^2S
- **IRQ 5** : I^2C
- **IRQ 6** : User IRQ0
- **IRQ 7** : User IRQ1

Additionally, the DMA controller can signal an interrupt at the completion of data transfers, allowing for further actions to be taken once a transfer has finished.

For more details about the DMA Controller, see: https://efabless.com/frigate-docs/dma

PRELIMINARY

Flash Writer

A QSPI (Quad SPI) Flash Writer is used to program Flash memory. It can write data to the Flash. A bootloader firmware can be written to interact with the Flash Writer by accessing its registers and driving the Flash SPI interface to program the Flash with the program bytes.

For more details about the Flash Writer, see: https://efabless.com/frigate-docs/flash-writer

System Timer

The system timer serves as the MTIMER (Machine-level Timer Device) on a RISC-V platform, providing machine-level timer functionality. It includes a fixed-frequency, monotonic MTIME register to track elapsed time, along with a MTIMECMP register.

Frigate Analog Peripherals

Analog Crossbar Fabric

TO BE COMPLETED

Audio DAC

The Frigate SoC includes a **delta-sigma audio DAC** (Digital-to-Analog Converter) that supports **16-bit resolution** and a **sampling rate of 48 kHz** . Delta-sigma DACs are known for their high accuracy and noise performance, making them ideal for audio applications. They work by oversampling the audio signal and using a noise-shaping technique to achieve high resolution and low distortion in the analog output.

The Frigate audio DAC driver is a digital amplifier that does not draw power (other than leakage) unless the audio DAC (in the SoC) is driving it. It has no enables or biases needed to operate.

For more information about the Audio DAC, see: https://efabless.com/frigate-docs/audiodac

PRELIMINARY

ADCs (12-bit 1 Msps, SAR)

Frigate includes two SAR ADCs offering 12-bit resolution and sampling at up to 1 Msps, with sample FIFO. The ADCs are controlled through a digital interface on the SoC. Simple conversions can be made by enabling the ADC, applying the Start bit, and reading the conversion from the data register.

For more information on use of the ADC sequencer, FIFO, and status registers, see: https://efabless.com/frigate-docs/adc

On-chip Temperature Sensor

TO BE COMPLETED

Instrumentation Amplifiers (Preamps)

TO BE COMPLETED

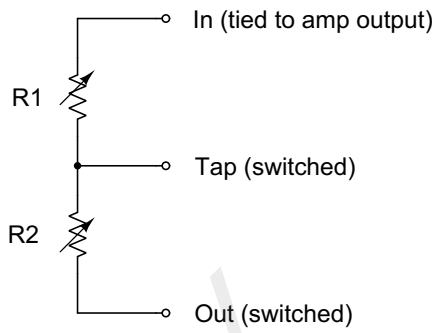
High-GBW Op-Amps

TO BE COMPLETED

PRELIMINARY

8-bit Rheostats

Frigate includes 4 rheostats, each implemented as a tapped 50 kΩ resistor with a variable tap position, forming a two-resistor voltage divider with a controllable ratio. The 8-bit digital input selects from 256 tap positions. At value 0, the tap is connected to Out; at value 255, it is connected to In.



Rtot = 50000
R2 = (Value / 256) * Rtot
R1 = Rtot - R1

Each of the four rheostats' In pin is connected to one of the four amplifier outputs. This position is fixed. Using the analog switch matrix, the rheostat tap can be connected to the amplifier negative input.

The rheostat outputs are connected to the same buses as the instrumentation amplifer inputs; this reduces the wiring in the analog block but limits the use of the op-amps and instrumentation amplifiers at the same time. The limitation can be worked around by using an external resistor for R2.

Voltage References

Frigate includes both a bandgap and CMOS voltage reference.

TO BE COMPLETED

Current Biases

TO BE COMPLETED

8-bit Voltage-mode DACs (RDACs)

Frigate includes two 8-bit RDACs (Resistive DACs). **TO BE COMPLETED**

PRELIMINARY

8-bit Current-mode DAC (iDAC)

The Frigate current-mode DAC takes an 8-bit digital input as its value and generates two outputs: one a source (receiver sinks the current to ground); and one a sink (receiver pulls current into the iDAC from power).

The current outputs (in either direction) follow the equation:

$$Iout = 100nA \times value$$

This gives a minimum current of 0uA and a maximum current of 25.6uA.

For more information, see: https://efabless.com/frigate-docs/idac

High-precision Comparator

TO BE COMPLETED

ULP (Ultra-Low-Power) Comparator

Frigate includes an ultra-low-power (ULP) comparator that is designed to operate in short bursts. It is controlled with digital signals via registers: “Enable” enables comparator operation, and “Clock” samples on a low transition and latches on a high transition.

PRELIMINARY

User Project Wrapper

Overview

A customer's own chip design is implemented within the Frigate " `user_project_wrapper` " template and submitted through the Efabless chipIgnite flow. It is then automatically integrated (by the Efabless chipIgnite submission process) into the rest of the Frigate chip harness to produce the final GDS that is submitted for fabrication. The user project wrapper provides **support for digital, analog, and mixed-signal projects** including entirely custom designs and designs assembled from existing IP blocks that can be soft (e.g. Verilog) and hard (e.g. premade layouts).

The User Project Wrapper (or "UPW" for short) is a GDS macro (named `user_project_wrapper`) of 2720µm x 2300µm (6.256mm²). It can be manually designed, assembled using ready-made IP blocks (see the Efabless Marketplace IPs), and generated with automated digital PnR and mixed-signal flows (using commercial tools such as Cadence or Synopsys, or open-source tools such as OpenLane).

The available area in sky130 can accommodate in excess of 2.4 million transistors; or 600,000 logic gate primitives.

UPW IOs/ports into the overall chip

The UPW has fixed edge pin placement that a user design will connect to, comprising the following ports:

- AHB-Lite bus including: system clock (`HCLK`); system reset (`HRESETn`); 32-bit address; 32-bit data.
- Up to 64 digital GPIO channels, each configurable as either a direct user IO, a predesignated Frigate SoC function, or (where available) an analog IO path. 56 of these are mask-programmed with the user's required power-on mode but the IO mode of all 64 can be overridden in software.
- Up to 16 analog paths which can pin-shared with the 64 GPIOs, or interconnected (by SoC register control) with analog paths in the Analog Subsystem.
- Multiple Analog Subsystem direct connections, including ADCs, DACs, voltage/current references, and more.

PRELIMINARY

- 2 additional "SIO" channels that may be used for a wide range of different purposes, including as regular GPIOs or as a differential pair.
- 16 IRQ outputs (allowing the user project to trigger interrupt requests for the Frigate CPU).
- PDN (Power Delivery Network) connections for up to 4 power domains (2 digital, 2 analog).

The designer can choose whether the design they include in the UPW will interface with the Frigate SoC (CPU and peripherals), the padframe GPIOs, the analog instrumentation/subsystem, the built-in clock sources, the PDN (Power Delivery Network), or any combination of these.

Firmware requirements

It is generally expected that any chip designed with Frigate will run firmware code on the SoC's CPU, in order to configure clocking and interact with peripherals. This includes interfacing with modules of the analog subsystem and its interconnects, as registers are used to control the switching matrix that enables direct/indirect analog (and digital) connections between the UPW and analog subsystem.

Frigate interfaces with an external SPI memory to load and execute firmware code (as Frigate has no on-chip non-volatile storage), making an external SPI memory an essential part of utilizing the full capability of Frigate.

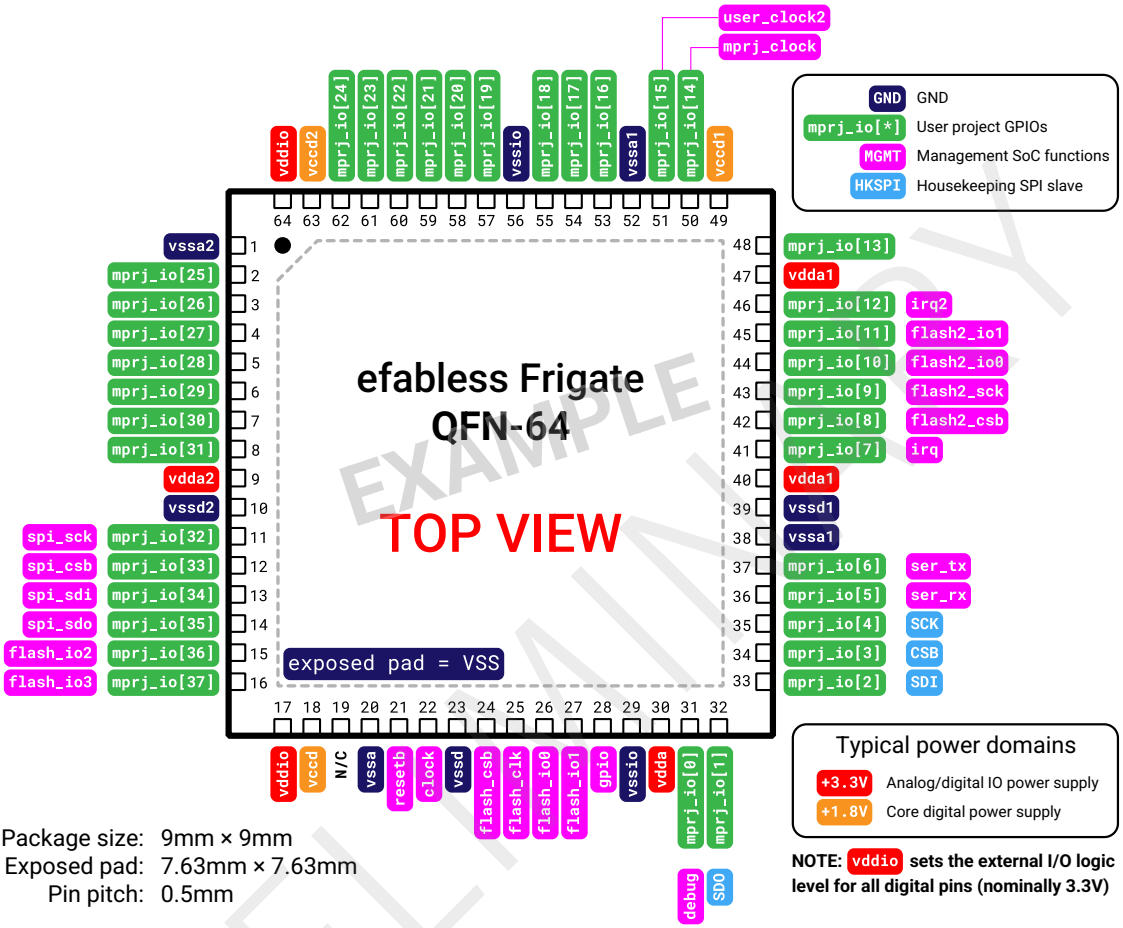
While running firmware on the SoC is not mandatory – i.e. a sophisticated user design can still be made, and a functional chip fabricated – a few specific considerations apply:

- Without firmware, a maximum of 56 GPIOs can be mask-programmed for use by the UPW.
- At power-up the Frigate default system clock `HCLK` source is always driven by a 500 kHz internal RC oscillator; firmware is required to change to a different oscillator or clock source. A design in the UPW would otherwise have to implement its own clock or use an external oscillator connected to a GPIO.
- Most digital and analog peripherals in Frigate cannot be activated or otherwise used without firmware.

For more information, see the section on Uploading/Flashing Firmware .

PRELIMINARY

Frigate Pinouts



Frigate QFN-64 pinout (EXAMPLE ONLY)

All packaged chips fabricated using the Frigate template will use **(TBA)** one of 2 standard packaging options having a standard pinout with some pins dedicated to the SoC (i.e. the CPU), some for power, and most for user-configurable GPIOs that are shared with optional system functions.

Frigate chips can also be ordered as bare dice (unpackaged, bare silicon chips).

Frigate pins and functions

Name	Pad [1]	Power V [2]	Power Area [3]	OVT [4]	GPIO [5]	Dir [6]	Analog [7]	POR [8] Mode Src	F0: General function / Description		Extended SoC Functions [9] F1	Extended SoC Functions [9] F2
vddio_0	1	3.3	S			-		-	Power: `vddio` digital I/O voltage/supply [14]			
gpio0_0	2	-	-		✓	I/O [10]	[11]	S	SoC: (I) `debug`	User [10]: `gpio0[ 0 ]`	N/A	N/A
gpio0_1	3	-	-		✓	I/O [10]	[11]	S	SoC: (O) `SDO` (HKSPI data out)	User [10]: `gpio0[ 1 ]`	N/A	N/A
gpio0_2	4	-	-		✓	I/O [10]	[11]	S	SoC: (I) `SDI` (HKSPI data in)	User [10]: `gpio0[ 2 ]`	N/A	N/A
gpio0_3	5	-	-		✓	I/O [10]	[11]	S	SoC: (I) `CSB` (HKSPI chip select)	User [10]: `gpio0[ 3 ]`	N/A	N/A
vssd1_0	6	GND	U			-		-	Ground: `vssd1` user core digital ground 1 [16]			
vssio_0	7	GND	S			-		-	Ground: `vssio` digital I/O ground [14]			
gpio0_4	8	-	-		✓	I/O [10]	[11]	S	SoC: (I) HKSPI `SCK` (clock)	User [10]: `gpio0[ 4 ]`	N/A	N/A
gpio0_5	9	-	-		✓	I/O [10]	[11]	S	SoC: (I/O) USB CDC D+	User [10]: `gpio0[ 5 ]`	N/A	N/A
gpio0_6	10	-	-		✓	I/O [10]	[11]	S	SoC: (I/O) USB CDC D-	User [10]: `gpio0[ 6 ]`	N/A	N/A
gpio0_7	11	-	-		✓	I/O [10]	[11]	S	SoC: (I) `test` (test signal)	User [10]: `gpio0[ 7 ]`	N/A	N/A
vddio_1	12	3.3	S			-		-	Power: `vddio` digital I/O voltage/supply [14]			
vdda1_0	13	3.3	U			-		-	Power: `vdda1` user analog supply 1 [20]			
vssa1_0	14	GND	U			-		-	Ground: `vssa1` user analog ground 1 [20]			
gpio1_0	15	-	-	✓	✓	I/O	[11]	M	`gpio1[ 0 ]` : Port A - Pin 0		`UART0_TX`	Clock Monitoring
gpio1_1	16	-	-	✓	✓	I/O	[11]	M	`gpio1[ 1 ]` : Port A - Pin 1		`UART0_RX`	-
gpio1_2	17	-	-	✓	✓	I/O	[11]	M	`gpio1[ 2 ]` : Port A - Pin 2		`TMR0_PWM0`	-
gpio1_3	18	-	-	✓	✓	I/O	[11]	M	`gpio1[ 3 ]` : Port A - Pin 3		`TMR0_PWM1`	-
vccd1_0	19	1.8	U			-		-	Power: `vccd1` user core digital supply 1 [16]			
vssio_1	20	GND	S			-		-	Ground: `vssio` digital I/O ground [14]			
gpio1_4	21	-	-	✓	✓	I/O	[11]	M	`gpio1[ 4 ]` : Port A - Pin 4		`TMR0_PWM_Fault`	-
gpio1_5	22	-	-	✓	✓	I/O	[11]	M	`gpio1[ 5 ]` : Port A - Pin 5		`I2S0_WS`	`TMR4_PWM0`
gpio1_6	23	-	-	✓	✓	I/O	[11]	M	`gpio1[ 6 ]` : Port A - Pin 6		`I2S0_SCK`	`TMR4_PWM1`
gpio1_7	24	-	-	✓	✓	I/O	[11]	M	`gpio1[ 7 ]` : Port A - Pin 7		`I2S0_SDI`	`TMR4_PWM_Fault`
vddio_2	25	3.3	S			-		-	Power: `vddio` digital I/O voltage/supply [14]			
vdda1_1	26	3.3	U			-		-	Power: `vdda1` user analog supply 1 [20]			
vssa1_1	27	GND	U			-		-	Ground: `vssa1` user analog ground 1 [20]			
vssd1_1	28	GND	U			-		-	Ground: `vssd1` user core digital ground 1 [16]			

PRELIMINARY

Name	Pad [1]	Power V [2]	Power Area [3]	OVT [4]	GPIO [5]	Dir [6]	Analog [7]	POR [8] Mode Src	F0: General function / Description	Extended SoC Functions [9] F1	Extended SoC Functions [9] F2
gpio2_0	29	-	-		✓	I/O	[11]	M	gpio2[0] : Port B - Pin 0	psram_cen	TMR5_PWM0
gpio2_1	30	-	-		✓	I/O	[11]	M	gpio2[1] : Port B - Pin 1	psram_clk	TMR5_PWM1
gpio2_2	31	-	-		✓	I/O	[11]	M	gpio2[2] : Port B - Pin 2	psram_io3	TMR5_PWM_Fault
gpio2_3	32	-	-		✓	I/O	[11]	M	gpio2[3] : Port B - Pin 3	psram_io2	-
vssio_2	33	GND	S			-		-	Ground: vssio digital I/O ground [14]		
gpio2_4	34	-	-		✓	I/O	[11]	M	gpio2[4] : Port B - Pin 4	psram_io1	-
gpio2_5	35	-	-		✓	I/O	[11]	M	gpio2[5] : Port B - Pin 5	psram_io0	-
gpio2_6	36	-	-		✓	I/O	[11]	M	gpio2[6] : Port B - Pin 6	I2C0_SDA	-
gpio2_7	37	-	-		✓	I/O	[11]	M	gpio2[7] : Port B - Pin 7	I2C0_SCL	-
vccd1_1	38	1.8	U			-		-	Power: vccd1 user core digital supply 1 [16]		
vddio_3	39	3.3	S			-		-	Power: vddio digital I/O voltage/supply [14]		
vssd1_2	40	GND	U			-		-	Ground: vssd1 user core digital ground 1 [16]		
gpio3_0	41	-	-		✓	I/O	[12]	M	gpio3[0] : Port C - Pin 0	SPI0_CLK	-
gpio3_1	42	-	-		✓	I/O	[12]	M	gpio3[1] : Port C - Pin 1	SPI0_SSn	-
gpio3_2	43	-	-		✓	I/O	[12]	M	gpio3[2] : Port C - Pin 2	SPI0_MOSI	-
gpio3_3	44	-	-		✓	I/O	[12]	M	gpio3[3] : Port C - Pin 3	SPI0_MISO	-
vssio_3	45	GND	S			-		-	Ground: vssio digital I/O ground [14]		
vccd1_2	46	1.8	U			-		-	Power: vccd1 user core digital supply 1 [16]		
gpio3_4	47	-	-		✓	I/O	[12]	M	gpio3[4] : Port C - Pin 4	UART1_TX	-
gpio3_5	48	-	-		✓	I/O	[12]	M	gpio3[5] : Port C - Pin 5	UART1_RX	-
gpio3_6	49	-	-		✓	I/O	[12]	M	gpio3[6] : Port C - Pin 6	I2C1_SDA	-
gpio3_7	50	-	-		✓	I/O	[12]	M	gpio3[7] : Port C - Pin 7	I2C1_SCL	-
vddio_4	51	3.3	S			-		-	Power: vddio digital I/O voltage/supply [14]		
vdda0_0	52	3.3	S			-		-	Power: vdda0 SoC analog supply [18]		
analog_0	53	-	-			-	[13]	A	analog_0		
analog_1	54	-	-			-	[13]	A	analog_1		
vssa0_0	55	GND	S			-		-	Ground: vssa0 SoC analog ground [18]		
gpio4_0	56	-	-		✓	I/O	[12]	M	gpio4[0] : Port D - Pin 0	TMR1_PWM0	-
gpio4_1	57	-	-		✓	I/O	[12]	M	gpio4[1] : Port D - Pin 1	TMR1_PWM1	-
gpio4_2	58	-	-		✓	I/O	[12]	M	gpio4[2] : Port D - Pin 2	TMR1_PWM_Fault	-
gpio4_3	59	-	-		✓	I/O	[12]	M	gpio4[3] : Port D - Pin 3	-	-
vssd2_0	60	GND	U			-		-	Ground: vssd2 user core digital ground 2 [17]		
vssio_4	61	GND	S			-		-	Ground: vssio digital I/O ground [14]		
gpio4_4	62	-	-		✓	I/O	[12]	M	gpio4[4] : Port D - Pin 4	-	-
gpio4_5	63	-	-		✓	I/O	[12]	M	gpio4[5] : Port D - Pin 5	-	-
gpio4_6	64	-	-		✓	I/O	[12]	M	gpio4[6] : Port D - Pin 6	UART2_TX	-

PRELIMINARY

Name	Pad [1]	Power V [2]	Power Area [3]	OVT [4]	GPIO [5]	Dir [6]	Analog [7]	POR [8] Mode Src	F0: General function / Description		Extended SoC Functions [9] F1	Extended SoC Functions [9] F2
gpio4_7	65	-	-		✓	I/O	[12]	M	gpio4[7] : Port D - Pin 7		UART2_RX	-
vccd2_0	66	1.8	U			-		-	Power: vccd2 user core digital supply 2 [17]			
vddio_5	67	3.3	S			-		-	Power: vddio digital I/O voltage/supply [14]			
vssd2_1	68	GND	U			-		-	Ground: vssd2 user core digital ground 2 [17]			
gpio5_0	69	-	-		✓	I/O	[11]	M	gpio5[0] : Port E - Pin 0		SPI1_CLK	-
gpio5_1	70	-	-		✓	I/O	[11]	M	gpio5[1] : Port E - Pin 1		SPI1_SSn	-
gpio5_2	71	-	-		✓	I/O	[11]	M	gpio5[2] : Port E - Pin 2		SPI1_MOSI	-
gpio5_3	72	-	-		✓	I/O	[11]	M	gpio5[3] : Port E - Pin 3		SPI1_MISO	-
vssio_5	73	GND	S			-		-	Ground: vssio digital I/O ground [14]			
gpio5_4	74	-	-		✓	I/O	[11]	M	gpio5[4] : Port E - Pin 4		UART3_TX	-
gpio5_5	75	-	-		✓	I/O	[11]	M	gpio5[5] : Port E - Pin 5		UART3_RX	-
gpio5_6	76	-	-		✓	I/O	[11]	M	gpio5[6] : Port E - Pin 6		CAN0_TX	-
gpio5_7	77	-	-		✓	I/O	[11]	M	gpio5[7] : Port E - Pin 7		CAN0_RX	-
vccd2_1	78	1.8	U			-		-	Power: vccd2 user core digital supply 2 [17]			
vssa2_0	79	GND	U			-		-	Ground: vssa2 user analog ground 2 [21]			
vdda2_0	80	3.3	U			-		-	Power: vdda2 user analog supply 2 [21]			
vddio_6	81	3.3	S			-		-	Power: vddio digital I/O voltage/supply [14]			
gpio6_0	82	-	-	✓	✓	I/O	[11]	M	gpio6[0] : Port F - Pin 0		TMR2_PWM0	-
gpio6_1	83	-	-	✓	✓	I/O	[11]	M	gpio6[1] : Port F - Pin 1		TMR2_PWM1	-
gpio6_2	84	-	-	✓	✓	I/O	[11]	M	gpio6[2] : Port F - Pin 2		TMR2_PWM_Fault	-
gpio6_3	85	-	-	✓	✓	I/O	[11]	M	gpio6[3] : Port F - Pin 3		TMR3_PWM0	-
vssio_6	86	GND	S			-		-	Ground: vssio digital I/O ground [14]			
vssd2_2	87	GND	U			-		-	Ground: vssd2 user core digital ground 2 [17]			
gpio6_4	88	-	-	✓	✓	I/O	[11]	M	gpio6[4] : Port F - Pin 4		TMR3_PWM1	-
gpio6_5	89	-	-	✓	✓	I/O	[11]	M	gpio6[5] : Port F - Pin 5		TMR3_PWM_Fault	-
gpio6_6	90	-	-	✓	✓	I/O	[11]	M	gpio6[6] : Port F - Pin 6		-	-
gpio6_7	91	-	-	✓	✓	I/O	[11]	M	gpio6[7] : Port F - Pin 7		-	-
vssa2_1	92	GND	U			-		-	Ground: vssa2 user analog ground 2 [21]			
vdda2_1	93	3.3	U			-		-	Power: vdda2 user analog supply 2 [21]			
vddio_7	94	3.3	S			-		-	Power: vddio digital I/O voltage/supply [14]			
gpio7_0	95	-	-		✓	I/O	[11]	M	debug =0 → gpio7[0] : Port G - Pin 0	debug =1 → JTAG tck	N/A	N/A
gpio7_1	96	-	-		✓	I/O	[11]	M	debug =0 → gpio7[1] : Port G - Pin 1	debug =1 → JTAG trst_n	N/A	N/A
gpio7_2	97	-	-		✓	I/O	[11]	M	debug =0 → gpio7[2] : Port G - Pin 2	debug =1 → JTAG tms	N/A	N/A
gpio7_3	98	-	-		✓	I/O	[11]	M	debug =0 → gpio7[3] : Port G - Pin 3	debug =1 → JTAG tdi	N/A	N/A
vssio_7	99	GND	S			-		-	Ground: vssio digital I/O ground [14]			
vccd2_2	100	1.8	U			-		-	Power: vccd2 user core digital supply 2 [17]			

PRELIMINARY

Name	Pad [1]	Power V [2]	Power Area [3]	OVT [4]	GPIO [5]	Dir [6]	Analog [7]	POR [8] Mode Src	F0: General function / Description	Extended SoC Functions [9] F1	Extended SoC Functions [9] F2
gpio7_4	101	-	-		✓	I/O	[11]	M	`debug` =0 → `gpio7[ 4 ]` : Port G - Pin 4 `debug` =1 → JTAG `tdo`	N/A	N/A
gpio7_5	102	-	-		✓	I/O	[11]	M	`gpio7[ 5 ]` : Port G - Pin 5	N/A	N/A
gpio7_6	103	-	-		✓	I/O	[11]	M	`gpio7[ 6 ]` : Port G - Pin 6	N/A	N/A
gpio7_7	104	-	-		✓	I/O	[11]	M	`gpio7[ 7 ]` : Port G - Pin 7	N/A	N/A
vddio_8	105	3.3	S			-		-	Power: `vddio` digital I/O voltage/supply [14]		
vccd0_0	106	1.8	S			-		-	Power: `vccd0` SoC core digital supply [15]		
select	107	-	-			I		F	Reserved; should be tied low		
resetb	108	-	-			I		F	`resetb` : SoC reset (active-low)		
gpio8_0	109	-	-			I		F	`clock` : SoC external clock input		
gpio8_1	110	-	-			O		F	`flash_csb` : Firmware SPI chip select (active-low)		
gpio8_2	111	-	-			O		F	`flash_clk` : Firmware SPI clock		
gpio8_3	112	-	-			I/O		F	`flash_io0` : Firmware SPI MOSI / QSPI IO0		
vssio_8	113	GND	S			-		-	Ground: `vssio` digital I/O ground [14]		
vssd0_0	114	GND	S			-		-	Ground: `vssd0` SoC core digital ground [15]		
xi0	115	-	-			-		A	LSXO crystal in		
xo0	116	-	-			-		A	LSXO crystal out		
xi1	117	-	-			-		A	HSXO crystal in		
xo1	118	-	-			-		A	HSXO crystal out		
vddio_9	119	3.3	S			-		-	Power: `vddio` digital I/O voltage/supply [14]		
vccd0_1	120	1.8	S			-		-	Power: `vccd0` SoC core digital supply [15]		
gpio8_4	121	-	-			I/O		F	`flash_io1` : Firmware SPI MISO / QSPI IO1		
gpio8_5	122	-	-			I/O		F	`flash_io2` : Firmware QSPI IO2		
gpio8_6	123	-	-			I/O		F	`flash_io3` : Firmware QSPI IO3		
gpio8_7	124	-	-		✓ [22]	I/O		F	`mgmt_gpio` : Dedicated CPU GPIO pin		
vssio_9	125	GND	S			-		-	Ground: `vssio` digital I/O ground [14]		
vssa3_0	126	GND	S			-		-	Ground: `vssa3` user analog ground alt. [19]		
vdda3_0	127	3.3	S			-		-	Power: `vdda3` Soc analog supply alt. [19]		
vssd0_1	128	GND	S			-		-	Ground: `vssd0` SoC core digital ground [15]		
sio0	129	-	-		✓	I/O		U	`sio9_0` : Special IO 0		
sio1	130	-	-		✓	I/O		U	`sio9_1` : Special IO 1		

[1]

Pads on the die are numbered from “1” starting at the top of the left-hand die edge, and run counterclockwise.

[2]

PRELIMINARY

Typical supply voltage levels are: 1.8V for core digital domains; 3.3V for VDDIO and analog domains. For more information, see: Electrical Characteristics .

[3]
U=User design area power domain; S=System core and SoC power domain (mandatory).

[4]
OVT=Over-Voltage Tolerant pin.

[5]
Except where otherwise noted, pins marked “GPIO” can be used as General Purpose I/O pins by either the User Project Wrapper (hardware) or SoC (CPU; software).

[6]
Except where otherwise noted, digital signal pins are all bidirectional-capable (I, O, or I/O) depending on configured mode (as soft-overridden, or mask-programmed where available).

[7]
Most signal pins can optionally be configured for dedicated analog connections, as individually noted.

[8]
At POR (Power-On Reset) each signal pin has a specific source for its mode configuration: S=Specific SoC Function (soft-overridable); M=Mask-programming by user (soft-overridable); F=Fixed (SoC function; not overridable); A=Analog dedicated path only (not overridable); U=User area input (soft-overridable).

[9]
Except where otherwise noted, at reset all digital signal pins will select function “F0” if their “POR Mode” permits (thus typically operating as a simple GPIO that is either user-hardware-controlled or SoC-software-controlled). To use an extended SoC function (where available), the respective pin must be set to a suitable “MGMT” (SoC Management) mode, and the respective function (“F1” or “F2”) must be selected via a firmware write to the respective pin-mux register.

PRELIMINARY

[10] (1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11 , 12 , 13 , 14 , 15 , 16)
`gpio0[ X ]` pins always power up in "MGMT" (SoC Management) mode with their stated SoC function and direction active, ensuring certain core diagnostic/debug functions of the SoC can always be accessed. These pins are still soft-configurable, meaning that firmware running on the SoC (or mode control set via the Housekeeping SPI (HKSPI) interface) can still choose to switch them to a "USER" mode after power-up, thus selecting their direction (I, O, or I/O) and connecting them directly to the User Project Wrapper .

[11] (1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11 , 12 , 13 , 14 , 15 , 16 , 17 , 18 , 19 , 20 , 21 , 22 , 23 , 24 , 25 , 26 , 27 , 28 , 29 , 30 , 31 , 32 , 33 , 34 , 35 , 36 , 37 , 38 , 39 , 40 , 41 , 42 , 43 , 44 , 45 , 46 , 47 , 48 , 49)
Pin can be configured for "ANALOG" mode (thus disabling its digital interface) and optionally switched onto the Amuxbus (Analog Mux Bus).

[12] (1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11 , 12 , 13 , 14 , 15 , 16)
Pin has analog switching capability common to most GPIOs (see [11]) but also has a direct analog path to the User Project Wrapper .

[13] (1 , 2)
Pin is a specialized analog-only pin that can optionally be switched onto the Amuxbus (Analog Mux Bus) or to other signal paths in the Analog Subsystem.

[14] (1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11 , 12 , 13 , 14 , 15 , 16 , 17 , 18 , 19 , 20)
`vddio` and `vssio` are respective power and ground pins that power the supply current and level shifters of all digital I/O pins, and also set the clamping voltage for ESD protection circuits on all signal pins. All `vddio_X` pins must be connected to the same net (i.e. set to the same voltage), which is nominally 3.3V. See Electrical Characteristics for more information.

[15] (1 , 2 , 3 , 4)
`vccd0` and `vssd0` form the digital core power domain of the SoC (i.e. non-user-project), and should nominally be 1.8V.

[16] (1 , 2 , 3 , 4 , 5 , 6)
`vccd1` and `vssd1` form the primary digital core power domain of the User Project Wrapper , and should nominally be 1.8V.

[17] (1 , 2 , 3 , 4 , 5 , 6)
`vccd2` and `vssd2` form the alternate digital core power domain of the User Project Wrapper , and should nominally be 1.8V.

PRELIMINARY

[18] (1 , 2)
`vdda0` and `vssa0` form the primary analog power domain of the SoC, and should nominally be 3.3V.

[19] (1 , 2)
`vdda3` and `vssa3` form the alternate analog power domain of the SoC, and should nominally be 3.3V.

[20] (1 , 2 , 3 , 4)
`vdda1` and `vssa1` form the primary analog power domain of the User Project Wrapper , and should nominally be 3.3V.

[21] (1 , 2 , 3 , 4)
`vdda2` and `vssa2` form the alternate analog power domain of the User Project Wrapper , and should nominally be 3.3V.

[22]
`mgmt_gpio` is a single dedicated GPIO pin belonging to the SoC (under CPU control) which does not connect to the User Project Wrapper and can be configured and used via firmware as a simple GPIO (input, output, or bidirectional pin). At reset, it is configured as an input.

PRELIMINARY

Clocking, Resetting, and DLL

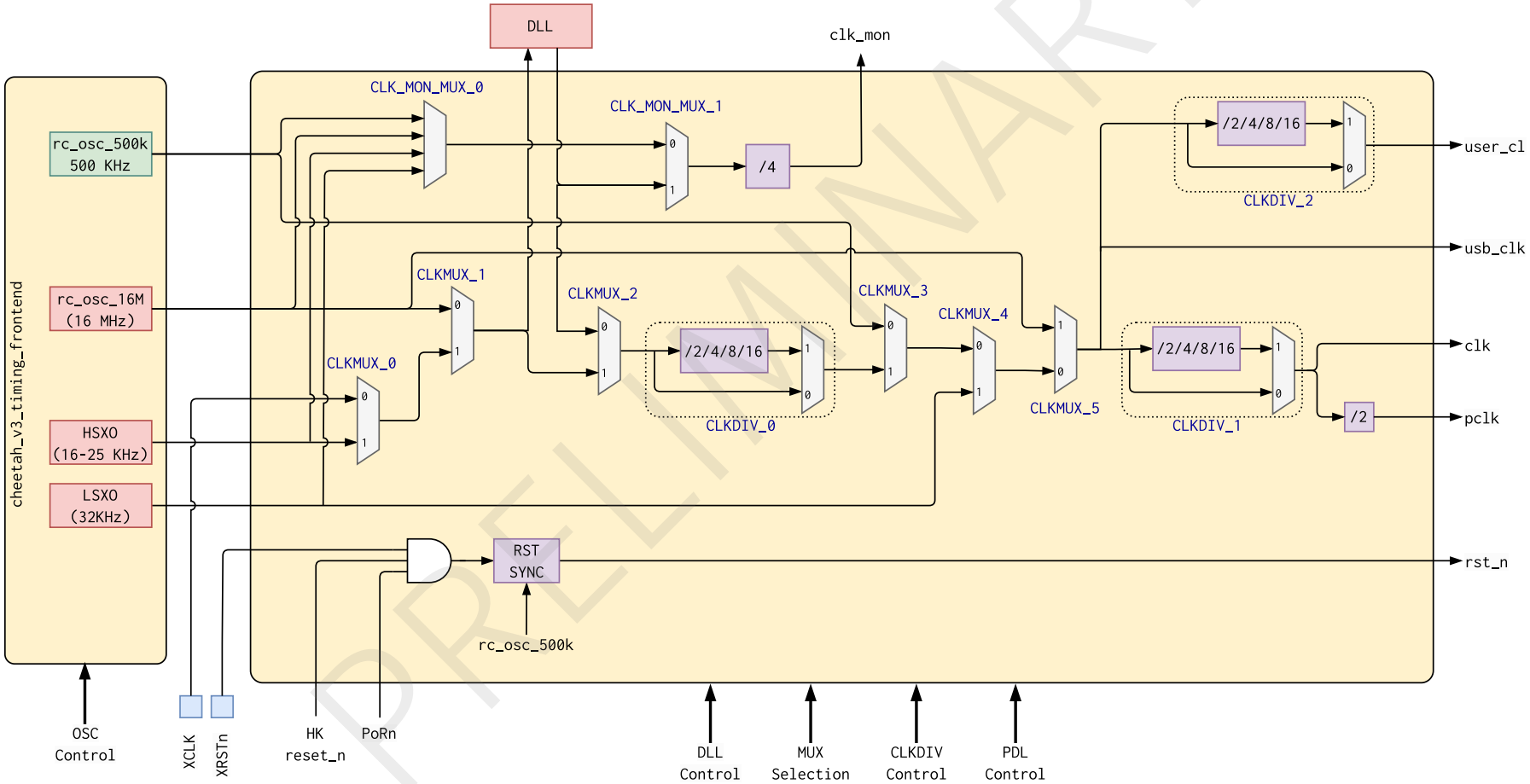
At power-on, the chip's clock source defaults to a ~500 kHz RC oscillator (RCO500K) and passes this clock signal to the SoC core and User Project Wrapper . If an alternate clock source is to be used, and/or optionally the selected clock source is to be multiplied/divided by the on-chip DLL , then explicit activation is required via Housekeeping SPI (HKSPI) or firmware running on the CPU.

The Frigate SoC employs a clock and reset manager that provides the following features:

- Five clock sources:
  - RCO16M: On-chip 16 MHz RC Oscillator
  - RCO500K: On-chip 500 kHz RC Oscillator (enabled at reset)
  - HSXO: High Speed Crystal Oscillator (16-25 MHz)
  - LSXO: Low power Crystal Oscillator (32.768 kHz)
  - XCLK: External clock source (up to 48 MHz)
- All-digital Delay Locked Loop (DLL) for frequency multiplication for clocks from RCO16M, XCLK or HSXO.
- Glitch-less clock multiplexing allows energy-efficient smooth transition between different clock sources.
- Clock dividers allowing active clock frequencies from 250 kHz to 48 MHz
- Reset sources are: external reset, POR (Power-On Reset), or Housekeeping reset
- Reset synchronizer (always with the RCO500K)

PRELIMINARY

Frigate clock and reset module block diagram



Frigate clock/reset module block diagram

PRELIMINARY

Power-on/Reset Behavior

On resetting the Frigate SoC either by Power-On Reset (PoR), HK reset or the external reset (XRSTn), RCO500K is the only enabled clock source.

Clock Monitoring

The Frigate SoC includes a clock monitor that is redirected to `gpio1[ 0 ]` when the GPIO is configured to use SoC function "F2" by the pin multiplexer. The available clocks for monitoring are the RCO16M, RCO500K, LSXO, HSXO, and DLL output. The selected clock frequency is divided by 4 before redirection.

Delay-Locked Loop (DLL)

The Frigate SoC includes an all-digital DLL (Delay-Locked Loop) that has a controller and a DCO (Digitally-Controlled Oscillator) implemented as a tunable ring oscillator. The controller divides the output of the ring oscillator down (frequency multiply) by the DLL feedback division ratio value, and the result is compared against the input clock to the DLL. The trim of the DLL is adjusted to try to make the divide ring oscillator frequency match the target frequency. The ring oscillator is 13 stages with two trim bits per stage, making the oscillator effectively 13 stages minimum and 39 stages maximum. The maximum rate (measured) is 113.35MHz and the minimum rate (measured) is 50.25MHz at 1.8V supply. In DCO mode, the controller is held in reset and external trim is applied.

USB Clock

The Frigate SoC's USB CDC interface requires a clock of 48MHz, with a 24MHz system clock (as the ratio between the two clocks must be 2:1).

PRELIMINARY

Uploading/Flashing Firmware

The Frigate SoC requires off-chip memory for storage of firmware that is to be run at power-up, necessitating the use of an external quad I/O SPI Flash memory device. To facilitate communication with this external Flash memory, the SoC incorporates a dedicated Flash controller equipped with Execute in Place (XIP) functionality, enabling direct execution of code from the external Flash memory without the need for prior copying to internal memory.

Furthermore, the Flash controller integrates a cache memory to mitigate the inherent latency associated with read operations from the external Flash memory, thereby enhancing overall system performance. Additionally, the Flash controller encompasses a Flash Writer accessible through software, allowing for programmatic modification of the contents of the external Flash memory.

There are several methods available for programming the external Flash memory using the SoC. The following sections will provide a detailed explanation of each of these programming methods, including their specific procedures, potential advantages and disadvantages, and any relevant considerations for implementation.

Firmware Transfer by JTAG

In this approach, the firmware responsible for the programming process is initially loaded into the SoC internal SRAM. This loading is achieved through the JTAG interface, utilizing the GDB (GNU Debugger) and OpenOCD (Open On-Chip Debugger) software. Once the Flash programming firmware resides in the SRAM, GDB instructs the CPU to start running the firmware. Once started, it receives the program binary destined for the Flash memory. This program binary can be transmitted to the firmware through either the UART or USB interface. The firmware then manages the process of programming the received program binary into the Flash memory.

For more information on how to program the Flash memory via JTAG, see: https://efabless.com/frigate-docs/jtag-programming

Firmware Flashing by HKSPI

Another way to program the Flash memory is to utilize the HKSPI pass-through mode. This approach allows the Flash memory to be programmed directly through the same SPI

PRELIMINARY

communication channel that is used for Housekeeping SPI functions. Consequently, additional wiring or connections to the QSPI flash chip are not required, which streamlines the programming process and reduces hardware complexity.

For more information, see: https://efabless.com/frigate-docs/hkspi

Bootloader Firmware Flashing

In this method a bootloader firmware is programmed into the Flash memory only once during the manufacturing or initial setup process. This bootloader plays a critical role in managing Frigate's startup and application execution process.

Upon booting, the bootloader doesn't immediately execute application code. Instead, it enters a brief waiting period, actively monitoring for any I/O (Input/Output) events. This I/O event could be triggered through various means, such as a specific command received via USB or UART interfaces.

If the bootloader detects a predetermined I/O event within the waiting period, it interprets this as a signal to enter Flash programming mode. In this mode, Frigate is ready to receive a new application program, which can be uploaded and written into the Flash memory via USB or UART communication channels. On the other hand, if the bootloader doesn't detect any I/O event within the designated timeframe, it assumes that there's no need for reprogramming. In this scenario, the bootloader proceeds to execute the existing application program that was previously stored in the Flash memory. This ensures that Frigate seamlessly boots into its operational state with the intended application functionality.

To ensure compatibility and smooth interaction between the bootloader and the application code, a special linker script is required during the application build process. This linker script is responsible for organizing the application's code and data in a way that aligns with the bootloader's expectations. This includes placing the application code at a specific memory location where the bootloader can find and execute it. The linker script also handles the placement of interrupt vectors and other critical elements that are essential for the proper functioning of the Frigate system.

For more information, see: https://efabless.com/frigate-docs/bootloader

PRELIMINARY

Electrical Characteristics

Absolute Maximum Ratings

Parameter	Description	Unit	Min	Typ	Max	Conditions
vccd, vccd1, vccd2	Digital supply relative to VSS	V	-0.05	–	1.95	
vdda, vdda1, vdda2, vdda3, vddio	Analog supply relative to VSS	V	-0.05	–	6	vddio must be equal or greater than vdda
gpio_ovt v2, SIO	GPIO voltage	V	-0.05	–	6	
gpio (absolute)	GPIO voltage	V	-0.05	–	vddio+0.5	
	Storage temperature range	C	-25	–	100	
	Maximum junction temperature	C	–	–	125	

PRELIMINARY

Operating Conditions

Name	Unit	Min	Nom	Max	Note
Power supplies					
vccd0	V	1.71	1.8	1.89	SoC digital core
vccd1	V	1.71	1.8	1.89	User project digital
vccd2	V	1.71	1.8	1.89	User project digital
vdda0	V	3	3.3	5.5	SoC analog subsystem
vdda1	V	1.71	3.3	5.5	User project analog
vdda2	V	1.71	3.3	5.5	User project analog
vdda3	V	3	3.3	5.5	SoC analog timing
vddio	V	3	3.3	5.5	GPIO logic level & ESD clamp
IO DC specs					
V_{OH}	V	vddio - 0.6			
V_{OL}	V			0.6	
V_{IH} (CMOS)	V	0.7 * vddio			
V_{IL} (CMOS)	V			0.3 * vddio	
V_{IH} (TTL)	V	2			
V_{IL} (TTL)	V			0.8	
Operating conditions					
Operating temperature range	C	0		70	
Low-power crystal oscillator frequency	kHz		32.768		
High-speed crystal oscillator frequency	MHz	16		25	
External clock input (XCLK) frequency	MHz	0		48	
Core clock frequency	MHz	0		40	
Digital IO input frequency	MHz	0		66	
Digital IO output frequency	MHz	0		33	
Power on reset release (power up)	V	2.5			
Power on reset assertion (power down)	V			1	
Power on rise time delay	msec	50		100	
Reset pulse width	msec	0.1			
Reset input voltage thresholds	V	0.7 * vddio		0.3 * vddio	

PRELIMINARY

Ordering Information

Frigate is part of the chipIgnite solution for Efabless. More information regarding chipIgnite can be found at:

https://efabless.com/chipignite

To get started with creating a project, please see the following library of videos, to get set up, create and develop a project, and submit it to Efabless for prototype fabrication:

https://efabless.com/getting-started

PRELIMINARY

License

Copyright 2025 Efabless Corporation.

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License.

You may obtain a copy of the License at:

https://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

PRELIMINARY

Legal Disclaimers

Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

PRELIMINARY