Document ID	Revision	Applies To	Status
FPT-2026-0114	v2.3.0	SY32L-S1, Rev. C silicon	Released

Scope and Audience

This document is the firmware-facing register and interrupt reference for the Synapse SY32L-S1 microcontroller family. It is written for embedded engineers who are bringing up board support packages, writing peripheral drivers, or debugging interrupt latency on Rev. C silicon. It does not cover electrical characteristics, package mechanicals, or qualification data – those live in the SY32L-S1 Datasheet (FPT-2026-0110). Register bit layouts, reset values, and access types below are normative for firmware and take precedence over any inline code comments in the reference driver package.

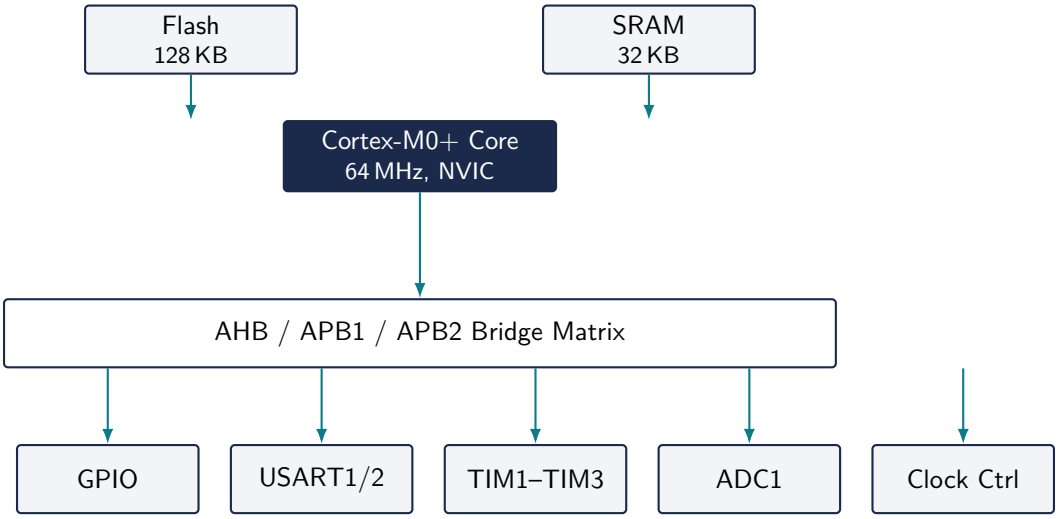
Readers are assumed to be comfortable with ARM Cortex-M0+ programming, memory-mapped I/O, and nested vectored interrupt controllers. Address values are given in hexadecimal without separators (e.g. `0x08000000`). Bit ranges are written `[MSB:LSB]`; a single-bit field is written with one index only.

Register Notation

RW read/write **RO** read-only **RC** read-clears (write ignored) **W1C** write-1-to-clear **Res.** reserved, must be written as reset value. Undefined behaviour results from writing reserved bits to anything other than their reset value.

1 Device Architecture

The SY32L-S1 pairs a Cortex-M0+ core running at up to 64 MHz with a single-cycle AHB matrix and two APB peripheral bridges. Firmware talks to every peripheral in this document through memory-mapped registers on APB1 or APB2; there are no port-mapped I/O instructions on this core.



1.1 Memory Map

Region	Start	End	Notes
Boot ROM (System)	0x1FFF0000	0x1FFF3FFF	Factory bootloader, read-only, entered when `BOOT0`=1
Option Bytes	0x1FFFF800	0x1FFFF80F	Read protection, watchdog-on-reset, BOR level
Main Flash	0x08000000	0x0801FFFF	128 KB application code, erase page = 1 KB
SRAM	0x20000000	0x20007FFF	32 KB, includes stack and .bss
APB1 Peripherals	0x40000000	0x40007FFF	TIM2–TIM3, USART2, I2C1
APB2 Peripherals	0x40010000	0x40017FFF	GPIO, USART1, TIM1, ADC1, EXTI
AHB Peripherals	0x40020000	0x40023FFF	DMA1, RCC (clock control), CRC
Cortex-M0+ Private Bus	0xE000E000	0xE000EFFF	NVIC, SysTick, SCB

2 Peripheral Registers

Base addresses for each peripheral instance are given in Table 1; all offsets below are relative to that base.

Peripheral	Base Address	Instance Notes
RCC (Clock Control)	0x40021000	Single instance, AHB
GPIOA	0x40010800	16 pins, APB2
GPIOB	0x40010C00	16 pins, APB2
USART1	0x40013800	APB2, up to 4 Mbit/s
USART2	0x40004400	APB1, up to 2 Mbit/s
TIM1	0x40012C00	Advanced, APB2
TIM2, TIM3	0x40000000, 0x40000400	General-purpose, APB1
ADC1	0x40012400	12-bit, APB2

Table 1: Peripheral base addresses.

2.1 RCC_CFGR – Clock Control Register (offset 0x04)

Reset value: `0x00000000`. Controls the PLL multiplier, AHB prescaler, and reports which oscillator currently drives the system clock.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLLMUL				AHBPRE			HSEON	HSERDY	HSION	HSIRDY	SWS			SW	

Field	Bits	Access	Reset	Description
PLLMUL	[15:12]	RW	0x0	PLL multiplier, output = HSI/2 × (PLLMUL+2), clamped to 64 MHz
AHBPRE	[11:9]	RW	0x0	AHB prescaler: 000=/1 ... 111=/512
HSEON	[8]	RW	0	Enable external 8 MHz crystal oscillator
HSERDY	[7]	RO	0	Set by hardware once HSE is stable
HSION	[6]	RW	1	Enable internal 8 MHz RC oscillator
HSIRDY	[5]	RO	1	Set by hardware once HSI is stable
SWS	[4:2]	RO	0x0	System clock actually in use: 000=HSI, 001=HSE, 010=PLL
SW	[1:0]	RW	0x0	Requested system clock switch, mirrored into SWS once locked

Firmware must poll `HSERDY` (or `HSIRDY`) before writing `SW`. Switching to an oscillator that has not reported ready leaves `SWS` at its previous value and silently ignores the write – this has been the root cause of two field returns where the bootloader appeared to hang for exactly the HSE startup timeout.

2.2 USART_CR1 – Control Register 1 (offset 0x0C)

Reset value: `0x00000000`. Applies to both USART1 and USART2.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
OVER8	Res.		M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	S

Field	Bits	Access	Reset	Description
OVER8	[15]	RW	0	0: 16× oversampling, 1: 8× oversampling (higher baud rates)
M	[12]	RW	0	Word length: 0 = 8 data bits, 1 = 9 data bits
PCE	[10]	RW	0	Parity control enable
PS	[9]	RW	0	Parity selection: 0 = even, 1 = odd
TXEIE	[7]	RW	0	Interrupt when TDR is empty
TCIE	[6]	RW	0	Interrupt on transmission complete
RXNEIE	[5]	RW	0	Interrupt when RDR has unread data
TE	[3]	RW	0	Transmitter enable
RE	[2]	RW	0	Receiver enable
SBK	[0]	RW	0	Send break; cleared by hardware once the break has been sent

2.3 TIMx_CR1 – Timer Control Register 1 (offset 0x00)

Reset value: `0x00000000`. Applies to TIM1, TIM2, and TIM3.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.						CKD		ARPE	CMS		DIR	OPM	URS	UDIS	CEN

Field	Bits	Access	Reset	Description
CKD	[9:8]	RW	0x0	Clock division for dead-time and filter sampling
ARPE	[7]	RW	0	Auto-reload preload enable; buffers ARR writes until next update
CMS	[6:5]	RW	0x0	Center-aligned mode selection (TIM1 only; 0x0 on TIM2/3)
DIR	[4]	RW	0	Counter direction: 0 = up, 1 = down
OPM	[3]	RW	0	One-pulse mode: counter stops at next update event
URS	[2]	RW	0	Update request source restricted to overflow/underflow
UDIS	[1]	RW	0	Update event disable
CEN	[0]	RW	0	Counter enable

3 Interrupt Vector Table

The vector table below is placed at `0x08000000` at reset (or relocated via `SCB_VTOR` for the bootloader's application slot at `0x08002000`). Priorities are 2-bit (0 = highest) as implemented by this core's reduced NVIC.

#	Offset	Exception / IRQ	Prio	Description
—	0x00	Initial Stack Pointer	—	Loaded into `MSP` before Reset_Handler runs
1	0x04	Reset	−3	Fixed highest priority, not configurable
2	0x08	NMI	−2	Clock security system failure routes here
3	0x0C	HardFault	−1	Bus/usage faults with no dedicated handler
11	0x2C	SVCall	configurable	Used by the reference RTOS for context switch
14	0x38	PendSV	configurable	RTOS tail-chaining hook
15	0x3C	SysTick	configurable	1 ms tick, drives the software timebase
16	0x40	WWDG	1	Window watchdog early-warning
17	0x44	PVD	2	Programmable voltage detector threshold crossed
18	0x48	RTC	1	Alarm or wakeup from the always-on domain
19	0x4C	FLASH	2	End-of-write / end-of-erase from the flash controller
20	0x50	RCC	2	Oscillator or PLL ready
21	0x54	EXTI0_1	0	External lines 0 and 1
22	0x58	EXTI2_3	0	External lines 2 and 3
23	0x5C	EXTI4_15	0	External lines 4 through 15
25	0x64	DMA1_CH1	1	DMA channel 1 complete/half/error
28	0x70	ADC1	1	End-of-conversion, watchdog, overrun
29	0x74	TIM1_UP	0	TIM1 update event
31	0x7C	TIM2	0	TIM2 global interrupt
32	0x80	TIM3	0	TIM3 global interrupt
34	0x88	I2C1	1	Combined event and error
35	0x8C	SPI1	1	TX/RX buffer and error
36	0x90	USART1	0	TX empty, RX not empty, framing/overrun errors
37	0x94	USART2	0	Same as USART1, second instance

Shared Handlers

`EXTI4_15` covers twelve GPIO lines. Drivers must read `EXTI_PR` inside the handler and clear only the bit they own – clearing the whole register has caused missed edges on adjacent pins in the motor-control reference design.

4 Boot Sequence

1. Power-on reset holds the core until `VDD` crosses the brown-out threshold latched in the option bytes.
2. Boot ROM samples `BOOT0`. High selects the system bootloader at `0x1FFF0000`; low proceeds to the application.
3. Core fetches the initial stack pointer from offset `0x00` of the active vector table, then the Reset_Handler address from offset `0x04`.
4. `SystemInit()` configures HSE, starts the PLL, and only then switches `RCC_CFGR.SW` – see the notice in Section 2.1.
5. `.data` is copied from flash to SRAM and `.bss` is zeroed by the startup file.
6. `main()` is called with interrupts still globally masked; application code unmasks them once peripheral clocks and NVIC priorities are configured.
7. Independent watchdog, if enabled by option bytes, must receive its first refresh within 26 ms of this point.

Cold Boot Timing

Reset to HSI stable	2 µs
HSE crystal startup	1.8 ms
PLL lock	90 µs
Flash wait-state calc	4 µs
`main()` entry	2.1 ms

5 Power Modes

Mode	Typ. Current	Wake Latency	Wake Sources
Run	4.8 mA @ 64 MHz	—	—
Sleep	1.6 mA	1 cycle	Any enabled interrupt
Stop	3.2 µA	4 µs	EXTI lines, RTC alarm, USART wake
Standby	0.4 µA	60 µs	WKUP pins, RTC alarm, IWDG reset only

SRAM contents are lost in Standby. Any driver state that must survive (calibration constants, communication sequence counters) has to be pushed to the 128-byte backup register block before entering this mode – there is no automatic retention.

Entry to Stop or Standby is via `PWR_CR` followed by the Cortex-M0+ `WFI` instruction. Firmware should disable the SysTick interrupt before `WFI` in Stop mode; a pending SysTick otherwise wakes the core every 1 ms and defeats the purpose of the low-power entry.

6 Revision History

Rev.	Date	Change
v1.0.0	2025-02-10	Initial release, Rev. A silicon
v2.0.0	2025-09-04	Added TIM1 advanced-timer fields, Rev. B silicon
v2.2.0	2026-03-18	Corrected `RCC_CFGR.SWS` reset behaviour after HSE fallback
v2.3.0	2026-07-01	Rev. C silicon: added `EXTI4_15` shared-handler notice, updated Stop-mode current