Openloop

Openloop Hardware Wallet
Product Specification

Version 1.0.0 | Last updated 2026-08-08
Haudi Crypto, Inc. — Kazunori Asada, President & CEO

1. Product Overview

1.1 Product Name

Openloop - a multi-chain hardware wallet

1.2 Product Concept

A commercial-grade hardware wallet that is easy to use even for beginners while providing advanced security. It is compatible with major existing wallet apps (Sparrow Wallet, MetaMask, etc.) and supports multiple communication methods (QR code, USB, BLE). It also includes FIDO2/CTAP2 passkey (security key) functionality, delivering both crypto asset management and web authentication in a single device.

1.3 Target Market

- **Individual users**: beginners to intermediate users who want secure storage of crypto assets
- **Enterprise users**: in-house asset management and multisig operations
- **Sales channel**: EC sites such as Amazon

2. Supported Assets

2.1 Supported Cryptocurrencies

Asset type	Currency/Token	Chain	Implementation status
Native currency	**Bitcoin (BTC)**	Bitcoin	✓ Implemented

Asset type	Currency/Token	Chain	Implementation status
Native currency	**Ethereum (ETH)**	EVM-compatible chains	✓ Implemented
Native currency	**XRP**	XRP Ledger	✓ Implemented
Native currency	**Solana (SOL)**	Solana	✓ Implemented
Native currency	**TRON (TRX)**	TRON	✓ Implemented
Token	**ERC-20 tokens**	Ethereum/L2	✓ Implemented

2.2 Supported Bitcoin Script Types

Script type	Description	Address format	Status
P2WPKH	Native SegWit (Bech32)	bc1q…	✓ Supported
P2PKH	Legacy	1…	✓ Supported
P2SH	Script Hash	3…	✓ Supported
P2SH-P2WPKH	Nested SegWit	3…	✓ Supported
P2WSH	Native SegWit multisig (BIP48 m/48'/c'/0'/2', Zpub/Vpub)	bc1q…	✓ Supported
P2SH-P2WSH	Nested SegWit multisig (BIP48 m/48'/c'/0'/1', Ypub/Upub)	3…	✓ Supported
P2SH (multisig)	Legacy multisig (BIP45 m/45', xpub/tpub)	3…	✓ Supported

Multisig is supported across the three script types P2WSH / P2SH-P2WSH / P2SH. Cosigner public keys can be exported in Coldcard/Keystone format and imported into a coordinator such as Sparrow.

Bitcoin standards compliance:

- ✓ BIP39 - Mnemonic generation and recovery (12/24 words)
- ✓ BIP32 - HD key derivation
- ✓ BIP44 - Multi-account hierarchy

- ✓ BIP45 - Legacy multisig account structure
- ✓ BIP48 - Multisig account structure (P2WSH / P2SH-P2WSH)
- ✓ BIP49 - Nested SegWit derivation path (m/49'/0'/0'/0/x)
- ✓ BIP84 - Native SegWit derivation path (m/84'/0'/0'/0/x)
- ✓ BIP143 - SegWit signature hash computation
- ✓ BIP174 - PSBT (Partially Signed Bitcoin Transaction)

Receive address features:

The receive screen displays a representative address for each currency. The custom path feature lets you specify any BIP32/BIP44-compliant derivation path.

Currency	Standard derivation path (example)	Network switch	Custom path
Bitcoin	m/84'/0'/0'/0/0	Mainnet / Testnet	✓ secp256k1
Ethereum	m/44'/60'/0'/0/0	-	✓ secp256k1
XRP	m/44'/144'/0'/0/0	-	✓ secp256k1 / Ed25519
Solana	m/44'/501'/0'	-	✓ Ed25519
TRON	m/44'/195'/0'/0/0	-	✓ secp256k1

- On-demand derivation (computed when selected)
- Addresses are derived when selected and are not recomputed within the same session
- When Bitcoin is selected: you can switch networks with the Mainnet/Testnet dropdown (because the address formats differ)

Custom path feature:

On the receive screen, each element of a BIP32 derivation path can be specified individually. There are two entries, one per signing algorithm:

- **Custom Path (secp256k1)**: BIP32 standard key derivation. Generates addresses for BTC/ETH/XRP, etc.

- **Custom Path (Ed25519)**: SLIP-0010-compliant key derivation. Generates addresses for Solana/XRP Ed25519, etc.

Path element	Description	Selectable values
Purpose	BIP standard	44 (Legacy) / 49 (Nested SegWit) / 84 (Native SegWit)
Coin Type	Currency identifier	0 (BTC) / 1 (BTC Test) / 60 (ETH) / 144 (XRP) / 195 (TRX) / 501 (SOL)
Account	Account number	0–9
Change	Receive/change/omitted	(none) / 0 (receive) / 1 (change)
Index	Address index/omitted	(none) / 0–99

Variable path depth: setting Change/Index to `(none)` omits that level (3–5 levels).

- Example: Solana standard `m/44'/501'/0'` → Change=(none), Index=(none) for 3 levels
- Example: BTC standard `m/84'/0'/0'/0/0` → all specified for 5 levels

Hardened derivation rules:

- secp256k1: Purpose/CoinType/Account are hardened, Change/Index are non-hardened (BIP44-compliant)
- Ed25519: all levels hardened (SLIP-0010-compliant)

2.3 Total Built-in Currencies/Tokens and Networks/Chains

Category	Currencies/tokens	Networks	Notes
Bitcoin	1 (BTC)	2	Mainnet + Testnet
Solana	1 (SOL)	3	Mainnet + Devnet + Testnet
TRON	1 (TRX)	3	Mainnet + Shasta + Nile
XRP Ledger	1 (XRP)	2	Mainnet + Testnet
EVM	~2,600 (ERC-20 / 926 unique)	215	129 Mainnet + 86 Testnet
Total	**~2,600**	**225**	

2.4 Built-in EVM Networks - 215 Networks

Openloop supports **all EVM networks**. The list below shows the built-in networks whose network name is displayed on the device screen.

Non-built-in networks: EVM networks not on this list can still be signed for. The signing confirmation screen displays “Chain ID: XXXXX”.

EVM Mainnets (129 networks)

Network name	Chain ID	Native currency
Ethereum	1	ETH
Optimism	10	ETH
Flare Network	14	FLR
Songbird	19	SGB
Elastos	20	ELA
KardiaChain	24	KAI
Cronos	25	CRO
Rootstock RSK	30	RBTC
Telos	40	TLOS
XDC Network	50	XDC
BNB Smart Chain	56	BNB
Syscoin NEVM	57	SYS
OKT Chain	66	OKT
Meter	82	MTR
Viction	88	VIC
Gnosis Chain	100	XDAI
Velas	106	VLX
Fuse	122	FUSE
Huobi ECO Chain	128	HT
Unichain	130	UNI
Polygon	137	POL
Monad	143	MON
Sonic	146	S

Network name	Chain ID	Native currency
Manta Pacific	169	ETH
HashKey Chain	177	HSK
Mint	185	ETH
X Layer	196	OKB
BitTorrent	199	BTT
opBNB	204	BNB
Lens	232	GHO
TAC	239	TAC
Fantom	250	FTM
Fraxtal	252	frxETH
Kroma	255	ETH
Boba Network	288	ETH
Hedera	295	HBAR
zkSync Era	324	ETH
PulseChain	369	PLS
Cronos zkEVM	388	zkCRO
SX Network	416	SX
World Chain	480	ETH
Astar	592	ASTR
Flow EVM	747	FLOW
QL1	766	QOM
Bittensor EVM	964	TAO
Stable	988	FREE
HyperEVM	999	HYPE
Conflux eSpace	1030	CFX
Metis Andromeda	1088	METIS
Polygon zkEVM	1101	ETH
WEMIX	1111	WEMIX
Core	1116	CORE
Lisk	1135	ETH

Network name	Chain ID	Native currency
Moonbeam	1284	GLMR
Moonriver	1285	MOVR
Glue	1300	GLUE
Sei	1329	SEI
Story	1514	IP
Gravity	1625	G
Soneium	1868	ETH
LightLink	1890	ETH
Swellchain	1923	ETH
Milkomeda C1	2001	milkADA
Ronin	2020	RON
Vanar	2040	VANRY
Kava	2222	KAVA
Goat	2345	BTC
Abstract	2741	ETH
Morph	2818	ETH
Peaq	3338	PEAQ
Botanix	3637	BTC
SX Rollup	4162	SX
Merlin Chain	4200	BTC
IoTeX	4689	IOTX
Mantle	5000	MNT
Somnia	5031	STT
Superseed	5330	ETH
Saga	5464	SAGA
DuckChain	5545	TON
Nibiru	6900	NIBI
ZetaChain	7000	ZETA
Cyber	7560	ETH
Canto	7700	CANTO

Network name	Chain ID	Native currency
Kaia	8217	KAIA
Base	8453	ETH
IOTA EVM	8822	IOTA
Evmos	9001	EVMOS
Plasma	9745	FSN
SmartBCH	10000	BCH
BEVM	11501	BTC
Immutable zkEVM	13371	IMX
0G	16661	A0GI
Map Protocol	22776	MAPO
Oasis Sapphire	23294	ROSE
Mezo	31612	BTC
ApeChain	33139	APE
Mode	34443	ETH
Energi	39797	NRG
EDU Chain	41923	EDU
Arbitrum One	42161	ETH
Arbitrum Nova	42170	ETH
Celo	42220	CELO
Etherlink	42793	XTZ
Hemi	43111	ETH
Avalanche C-Chain	43114	AVAX
Zircuit	48900	ETH
Sophon	50104	SOPH
Superposition	55244	ETH
Ink	57073	ETH
Linea	59144	ETH
Henesys	68414	HNS
Berachain	80094	BERA
Blast	81457	ETH

Network name	Chain ID	Native currency
Zedxion	83872	ZEDX
Plume	98866	ETH
Taiko	167000	ETH
Bitlayer	200901	BTC
Scroll	534352	ETH
Katana	747474	ETH
XRPL EVM	1440000	XRP
Zora	7777777	ETH
Corn	21000000	BTCN
Neon EVM	245022934	NEON
Degen	666666666	DEGEN
Ancient8	888888888	ETH
Aurora	1313161554	ETH
Rari Chain	1380012617	ETH
Harmony	1666600000	ONE
SKALE Europa	2046399126	sFUEL

EVM Testnets (86 networks)

Network name	Chain ID	Native currency
Ethereum Sepolia	11155111	ETH
Ethereum Holesky	17000	ETH
Optimism Sepolia	11155420	ETH
Arbitrum Sepolia	421614	ETH
Base Sepolia	84532	ETH
Polygon Amoy	80002	POL
zkSync Sepolia	300	ETH
Linea Sepolia	59141	ETH
Scroll Sepolia	534351	ETH
Polygon zkEVM Cardona	2442	ETH
BNB Testnet	97	tBNB

Network name	Chain ID	Native currency
Avalanche Fuji	43113	AVAX
Fantom Testnet	4002	FTM
Cronos Testnet	338	TCRO
Flare Coston2	114	C2FLR
Songbird Coston	16	CFLR
Rootstock Testnet	31	tRBTC
Telos Testnet	41	TLOS
XDC Apothem	51	TXDC
OKT Chain Testnet	65	OKT
Meter Testnet	83	MTR
Viction Testnet	89	VIC
Gnosis Chiado	10200	XDAI
Fuse Sparknet	123	SPARK
Huobi ECO Testnet	256	HT
opBNB Testnet	5611	tBNB
Fraxtal Testnet	2522	frxETH
Kroma Sepolia	2358	ETH
Boba Sepolia	28882	ETH
Hedera Testnet	296	HBAR
PulseChain Testnet	943	tPLS
Cronos zkEVM Testnet	282	zkTCRO
Conflux eSpace Testnet	71	CFX
Metis Sepolia	59902	tMETIS
WEMIX Testnet	1112	tWEMIX
Core Testnet	1115	tCORE
Lisk Sepolia	4202	ETH
Moonbase Alpha	1287	DEV
Soneium Minato	1946	ETH
LightLink Pegasus	1891	ETH
Ronin Saigon	2021	RON

Network name	Chain ID	Native currency
Kava Testnet	2221	TKAVA
Abstract Testnet	11124	ETH
Morph Holesky	2810	ETH
Merlin Testnet	686868	BTC
IoTeX Testnet	4690	IOTX
Mantle Sepolia	5003	MNT
ZetaChain Athens	7001	aZETA
Canto Testnet	7701	CANTO
Kaia Kairos	1001	KAIA
IOTA EVM Testnet	1075	IOTA
Evmos Testnet	9000	tEVMOS
SmartBCH Testnet	10001	BCHT
BEVM Testnet	11503	BTC
Immutable zkEVM Testnet	13473	tIMX
Oasis Sapphire Testnet	23295	TEST
ApeChain Curtis	33111	APE
Mode Sepolia	919	ETH
Energi Testnet	49797	tNRG
Celo Alfajores	44787	CELO
Zircuit Testnet	48899	ETH
Berachain Bartio	80084	BERA
Blast Sepolia	168587773	ETH
Taiko Hekla	167009	ETH
Bitlayer Testnet	200810	BTC
Zora Sepolia	999999999	ETH
Neon EVM Devnet	245022926	NEON
Aurora Testnet	1313161555	ETH
Harmony Testnet	1666700000	ONE
Flow EVM Testnet	545	FLOW
X Layer Testnet	195	OKB

Network name	Chain ID	Native currency
BitTorrent Donau	1029	BTT
Sonic Blaze	57054	S
Lens Sepolia	37111	GHO
World Chain Sepolia	4801	ETH
Story Aeneid	1513	IP
Gravity Sepolia	13505	G
Unichain Sepolia	1301	ETH
Manta Pacific Sepolia	3441006	ETH
Mint Sepolia	1687	ETH
HashKey Testnet	133	HSK
Etherlink Testnet	128123	XTZ
Hemi Sepolia	743111	ETH
Ink Sepolia	763373	ETH
Plume Testnet	98864	ETH
XRPL EVM Devnet	1440002	XRP

2.5 Built-in ERC-20 Tokens - ~2,600 Tokens

Openloop supports **all ERC-20 tokens**. The firmware ships with **~2,600 tokens** spanning 129 mainnets (out of all 215 networks) — including the same token deployed across multiple chains, with **926 distinct tokens** — and their token information (symbol, decimals, name) is displayed automatically.

Non-built-in tokens: ERC-20 tokens not on this list can still be signed for. The signing confirmation screen displays “ERC-20 Token”, and the token can be identified by its contract address.

Breakdown by Token Category

Category	Token count (approx.)	Representative examples
Stablecoins	150+	USDT, USDC, DAI, JPYC, USDE, TUSD, FRAX
DeFi tokens	400+	UNI, AAVE, LINK, CRV, SKY, COMP, SNX
Memecoins	100+	

Category	Token count (approx.)	Representative examples
		SHIB, DOGE, PEPE, FLOKI, BONK
L2/infrastructure tokens	200+	ARB, OP, MATIC, IMX, LDO, BLUR
Gaming/NFT	150+	AXS, SAND, MANA, ENJ, GALA, APE
AI tokens	50+	FET, AGIX, RNDR, OCEAN
Others	1,500+	Various project tokens

Major Stablecoins (excerpt)

Token name	Symbol	Decimals	Supported chains
Tether USD	USDT	6	17
USD Coin	USDC	6	29
Dai Stablecoin	DAI	18	9
JPY Coin	JPYC	18	4
Ethena USDe	USDE	18	23
Frax	FRAX	18	24

Major DeFi Tokens (excerpt)

Token name	Symbol	Decimals	Supported chains
Uniswap	UNI	18	12
Chainlink	LINK	18	70
Aave	AAVE	18	11
Sky	SKY	18	1
Lido DAO	LDO	18	4
Arbitrum	ARB	18	4

Notes:

- ERC-20 tokens other than those above can still be sent (displayed as a contract address)
- Token symbols, decimals, and names are verified against major data sources
- The built-in token list is expanded via firmware updates

2.6 Ethereum Extensions

Supported Ethereum standards:

Standard	Description	Status
EIP-55	Address checksum (mixed-case)	✓ Supported
EIP-155	Replay protection (chain ID signing)	✓ Supported
EIP-191	Personal Sign (message signing)	✓ Supported
EIP-712	Typed Structured Data signing	✓ Supported
EIP-1559	Type 2 transaction (Priority Fee)	✓ Supported
EIP-2718	Typed Transaction Envelope	✓ Supported
EIP-2930	Access List (Type 1)	✗ Not supported (rejected at signing)
EIP-4527	QR Code eth-sign-request	✓ Supported
EIP-7702	Set Code for EOAs (smart account delegation)	✓ Supported
ERC-20	Token standard	✓ Supported
ERC-721	NFT standard (clear-signing of transferFrom / safeTransferFrom / approve / setApprovalForAll)	✓ Supported
ERC-1155	Multi-token standard (safeTransferFrom / safeBatchTransferFrom)	✓ Supported
Legacy (Type 0)	Legacy transactions (including pre-EIP-155)	✓ Supported

Smart contract support:

- ✓ **ERC-20 token transfers**: automatic detection of the Transfer function (0xa9059cbb) and amount display
- ✓ **Contract Interaction**: arbitrary smart contract calls
- ✓ **Contract Creation**: contract deployment

EIP-7702 smart account delegation:

- ✓ **Authorization signing**: sign with a specified chain_id, delegate address, and nonce
- ✓ **Built-in delegate registry**: displays the names of the delegate contracts for MetaMask, Coinbase, Uniswap, and OKX
- ✓ **Unverified contract warning**: warns for unregistered delegate targets
- ✓ **Revoke Delegation**: UI for revoking delegation via address(0)
- ✓ **chain_id=0 warning**: warns for a delegation valid on all chains

Multisig support:

- ✓ **EIP-712 Typed Data signing**: Safe (Gnosis Safe) multisig support
- ✓ **EIP-712 v2 streaming protocol**: compatible with Ledger Ethereum App v1.10+ (receives type definitions/values/filters incrementally). Supports the signing path used by MetaMask Mobile and Rabby over BLE/USB
- ✓ **Domain hash / message hash display**: for verification in the host app

Automatic transaction type detection:

Type	Display name	Detection condition
Send	"Send"	value > 0, data = empty
ERC-20 Transfer	"ERC-20 Transfer"	data = 0xa9059cbb···
Contract Interaction	"Contract Interaction"	data ≠ empty, methodSig ≠ transfer
Contract Creation	"Contract Creation"	to = null
Off-chain Signature	"Off-chain Signature"	EIP-712 request
Smart Account Auth	"Smart Account Auth"	EIP-7702 Authorization signing

2.7 XRP Ledger Support

Supported signing algorithms:

Standard	Description	Status
ECDSA secp256k1	secp256k1 signing	✓ Supported
EdDSA Ed25519	Ed25519 signing	✓ Supported

Dual signing algorithms:

XRP Ledger supports two signing algorithms, secp256k1 and Ed25519. Openloop determines the algorithm automatically from the address prefix:

- `r` address → determine key type from account information
- Can be specified explicitly with a custom path

Transaction handling:

Type	Display name	Detection condition
Payment	"XRP Payment"	TransactionType=0x0000
Other	"XRP Transaction"	Anything else

Derivation paths:

- Standard: m/44'/144'/0'/0/0 (secp256k1, BIP44-compliant)
- Ed25519: m/44'/144'/0'/0'/0' (SLIP-0010-compliant, all hardened)
- Address format: Base58Check ("r⋯" prefix)

XRP technical specifications:

Item	Value
Transaction format	XRP Binary Format (direct binary signing)
Hash prefix	0x53545800 ("STX\0")
Signature hash	secp256k1: SHA-512 Half (first 32 bytes) / Ed25519: RFC 8032 (signs the prefixed message directly)
Decimals	6 (1 XRP = 10^6 drops)

2.8 Solana Support

Supported signing algorithms:

Standard	Description	Status
Ed25519	EdDSA signing (RFC 8032)	✓ Supported
Versioned Transaction	v0 transaction (Address Lookup Table support)	✓ Supported
Legacy Transaction	Legacy transaction	✓ Supported

Transaction parsing:

Type	Display name	Detection condition
SOL Transfer	“SOL Transfer”	System Program Transfer instruction
SPL Token Transfer	“SPL Token Transfer”	Token Program Transfer instruction
Other	“Solana Transaction”	Anything else

Derivation paths:

- WalletConnect/Solflare standard: m/44’/501’/0’/0’ (Ed25519, all hardened, 4 levels)
- Receive screen default: m/44’/501’/0’ (3 levels)
- SLIP-0010-compliant; receive addresses support a variable path depth of 2–4 levels

2.9 TRON Support

Supported interfaces: USB, BLE (Air Gap QR not supported)

Supported signing algorithms:

Standard	Description	Status
ECDSA secp256k1	secp256k1 signing	✓ Supported

TRON technical specifications:

Item	Value
Curve	secp256k1 (same as Ethereum)
Address generation	keccak256 → 0x41 prefix → Base58Check(SHA256D) → “T⋯” address
TX signature hash	SHA-256 (different from Ethereum’s keccak-256)
V value	27 + recovery_id (fixed, no EIP-155)
BIP44 path	m/44’/195’/0’/0/0
Decimals	6 (1 TRX = 10^6 SUN)

Transaction parsing:

Type	Display name	Detection condition
TRX Transfer	“TRX Transfer”	TransferContract

Type	Display name	Detection condition
TRC-20 Transfer	“TRC-20 Transfer”	TriggerSmartContract + transfer()
Other	“TRON Transaction”	Anything else

Derivation paths:

- Standard: m/44’/195’/0’/0/0 (secp256k1, BIP44-compliant)
- Address format: Base58Check (“T⋯” prefix)

Ethereum compatibility and differences:

Because TRON is derived from Ethereum, it shares much in common, but there are important differences:

Item	Ethereum	TRON
Curve	secp256k1	secp256k1 (same)
Address hash	keccak256	keccak256 (same)
Address prefix	0x (hex)	0x41 → Base58Check → “T⋯”
TX signature hash	keccak-256	**SHA-256**
V value	EIP-155 (chainId-dependent)	**27 + recovery_id (fixed)**
Coin Type (BIP44)	60	195

2.10 Supported Signing Methods

Signing type	BTC	ETH	XRP	SOL	TRX	Supported interfaces
Transaction signing	✓	✓	✓	✓	✓ *1	USB, BLE, Air Gap *1
Message signing (personal_sign)	✓	✓	-	✓	✓	USB, BLE
Typed data signing (EIP-712)	-	✓	-	-	-	USB, BLE
PSBT signing	✓	-	-	-	-	USB, BLE, Air Gap
	-		-	-	-	USB, BLE

Signing type	BTC	ETH	XRP	SOL	TRX	Supported interfaces
Authorization signing (EIP-7702)		✓				
Batch signing (signAllTransactions)	-	-	-	✓	-	USB, BLE
Sign + broadcast	✓	✓	-	✓	✓	USB, BLE

*1 TRX supports USB/BLE only (Air Gap QR not supported)

Broadcast (transaction submission):

Broadcasting a signed transaction is performed through Openloop Connect:

Currency	Method	Broadcast method	Supported networks
BTC	sendTransfer	Blockstream/ mempool.space API	Mainnet, Testnet
ETH	eth_sendTransaction	Via dApp (WalletConnect)	All EVM chains
SOL	solana_signAndSendTransaction	Solana JSON-RPC sendTransaction	Mainnet, Devnet, Testnet
TRX	tron_signAndSendTransaction	TronGrid API broadcastTransaction	Mainnet, Shasta, Nile

3. FIDO2/CTAP2 Passkey Functionality

Openloop also operates as a FIDO2/CTAP2-compliant security key.

> ⚠ **Certification status**: Openloop has not currently obtained FIDO Alliance **FIDO Certified Authenticator** certification (FIDO2 Certified). While its CTAP2/WebAuthn implementation is spec-compliant, official FIDO Alliance certification and AAGUID registration (FIDO Metadata Service) may be considered for the future.

3.1 Overview

Item	Specification
Protocol	CTAP2 (FIDO_2_0 / FIDO_2_1)
Backward compatibility	U2F (FIDO U2F V2)
Transport	USB HID (CTAPHID, OS native) / BLE (via the Credential Provider in Openloop Connect)
Signing algorithms	ES256 (P-256/NIST) / EdDSA (Ed25519)
Discoverable Credentials	✓ Supported
User Presence (UP)	✓ Confirmation dialog on the device screen
User Verification (UV)	✓ Authentication with the device PIN
Maximum credentials	100
Credential ID length	32 bytes
Credential ID derivation	32 bytes. Cryptographically bound to this device using an SE050 key, and issues different credential IDs for USB and BLE (supporting transport-based classification by RPs)

Behavior by Transport

Transport	Path	Connect involvement
USB CTAPHID	OS standard FIDO authenticator stack (Win: webauthn.dll, Mac/Linux: native) → directly to the device	Not required
BLE (via Connect)	iOS: `ASCredentialProviderViewController` / Android: `CredentialProviderService` → tunnels CTAP2 to the device via the BLE transport inside Connect	**Required** (Openloop Connect mobile app)

> Because the same Openloop device generates a different credential_id per transport, USB and BLE can be **registered as separate credentials** with an RP. For RPs that filter by transport (e.g. Google), this is essential when you want to use both transports.

3.2 Security Design

- Private keys are not stored persistently on the device; they are derived from a hardware-protected seed when used
- DualSecure protection: SE050 (ECDH) + ESP32 dual keys
- SE050 hardware binding: binds each credential to this device with a non-extractable key
- A seed independent from the wallet

3.3 Supported Browsers and Platforms

USB CTAPHID Path

Platform	Chrome	Edge	Firefox	Safari
Windows	✓	✓	✓	-
macOS	✓	-	△	△

Behavior details:

- **Windows**: WebAuthn registration, authentication, and user rejection all work correctly across all browsers (via the OS-standard webauthn.dll)
- **macOS Chrome**: registration and authentication work correctly. User rejection is handled correctly as NotAllowedError
- **macOS Safari/Firefox**: registration and authentication work correctly. If the user does not approve on the device, no error is raised and the browser keeps waiting for a response (a platform-side limitation). You can recover with the browser's "Cancel" button

> Note: the wait-for-response behavior on macOS Safari/Firefox is a limitation of the OS-side AuthenticationServices/authenticator-rs framework, not an issue on the Openloop device side. Cancellation via CTAPHID_CANCEL works correctly on all platforms.

BLE (Openloop Connect) Path

Platform	Browser	Required OS version
iOS	Safari (via the Credential Provider Extension)	iOS 17+
Android	Chrome / Edge (via the Credential Provider Service)	Android 14+

Behavior details:

- **iOS**: Openloop Connect is registered as an `ASCredentialProviderViewController` (passkey provider). The OS picker shows “Openloop Connect” (note: per iOS specification, the extension’s display name is fixed to the parent app name). The actual communication is over BLE.
- **Android**: Openloop Connect is registered as a `CredentialProviderService`. The OS picker shows “Openloop Connect”. Communication is likewise over BLE.
- Required: installation of the Openloop Connect mobile app + BLE pairing with and selection of the Openloop device.

3.4 User Interface

- ON/OFF toggle, credential list (virtual scrolling, up to 100)
- Individual deletion, bulk reset
- Displays the RP name and user name at registration/authentication and lets you approve/reject

3.5 U2F Backward Compatibility

Supports U2F_REGISTER, U2F_AUTHENTICATE, and U2F_VERSION of U2F (Universal 2nd Factor, also known as CTAP1)

3.6 PIV/PKCS#11 Functionality

Openloop includes a PIV (Personal Identity Verification)-compliant APDU interface and can be used, via a PKCS#11 library, for SSH authentication and Firefox client certificate authentication.

Item	Specification
Protocol	PIV (NIST SP 800-73-compatible subset)
Communication	USB HID (APDU framing)
Signing algorithms	ECDSA P-256 / Ed25519 / RSA-2048
Slots	9A (Authentication), 9C (Digital Signature), 9D (Key Management), 9E (Card Authentication)
PIV ON/OFF	✓ Toggleable in the device settings screen
PIV PIN	✓ Optional (6–8 ASCII characters, stored only as non-plaintext)

Dual-mode signing:

Condition	Behavior	Use case
PIN authenticated over USB	Silent signing (no confirmation screen)	Automation / CI/CD
PIN not sent	Confirmation screen shown on the device	Interactive use

PKCS#11 library:

- Supports macOS / Windows / Linux (.dylib / .dll / .so)
- Communicates with the device through the Openloop Connect app
- Usable with PKCS#11-compatible applications such as SSH (`ssh -I`), GPG (`gnupg-pkcs11-scd`), PDF signing (`pyHanko`), Firefox, and `pkcs11-tool`

3.7 Behavior on Factory Reset

Passkey-related data and PIV data are all erased.

---

4. Communication Methods

4.1 QR Code Communication ✓ Implemented

Use case: integration with desktop apps such as Sparrow Wallet (air-gapped operation)

Supported input formats:

Format	Description	Status
BBQr V0/V1	Multi-frame, multi-part (Bitcoin)	✓ Supported
UR crypto-psbt	Bitcoin PSBT (Blockchain Commons)	✓ Supported
UR eth-sign-request	Ethereum signing request (EIP-4527)	✓ Supported
UR sol-sign-request	Solana signing request (Keystone-compatible)	✓ Supported
UR xrp-sign-request	XRP signing request	✓ Supported
Base64 PSBT	Single QR (Bitcoin)	✓ Supported

Format	Description	Status
ZLIB compression	wbits=-10 (BBQr)	✓ Supported

Supported output formats (switchable in settings):

Format	Description	Recommended use
BBQr	ZLIB compression, QR v4-5	Sparrow Wallet (recommended)
UR	Fountain codes	Keystone, AirGap Vault, Solflare
Base64	Single QR, uncompressed	General/debugging

QR code specifications:

- QR version: 4-5 (25×25 to 27×27 modules)
- Error correction: Level L
- Screen fit: Openloop (320×240)

4.2 USB Communication ✓ Implemented

Protocol: USB HID

USB mode selection:

The USB mode can be selected from the settings screen. Mode changes take effect immediately, with no restart required.

Mode	VID	PID	Description
Native	0x303A	0x8341	Default. Espressif VID + Openloop PID (officially allocated)
Compatible	0x2C97	0x1011	For when compatibility with existing wallet apps is required

Common settings:

Item	Value
Manufacturer	“Openloop”
Product	“Openloop Wallet”

Item	Value
Interfaces	HID + FIDO HID + CDC (composite device, up to 4 interfaces)
HID packet size	64 bytes

Three-stage USB configuration switching:

Mode	USB setting	Passkey setting	Interface configuration
CDC only	OFF	-	CDC (2 interfaces)
HID + CDC	ON	OFF	Ledger HID + CDC (3 interfaces)
HID + FIDO HID + CDC	ON	ON	Ledger HID + FIDO HID + CDC (4 interfaces)

USB identifiers:

- ✓ Manufacturer string: “Openloop”
- ✓ Product string: “Openloop Wallet” (shown in the Chrome HID dialog)
- ✓ VID/PID: user-selectable (Native/Compatible)

Supported applications:

- ✓ **Sparrow Wallet**: Bitcoin support
- ✓ **MetaMask**: Ethereum/ERC-20 support
- ✓ **Safe (Gnosis Safe)**: EIP-712 multisig support
- ✓ **Rabby Wallet**: EVM support
- ✓ **Solflare**: Solana support

USB settings:

- USB can be enabled/disabled from the Settings screen
- CDC (serial log) is always enabled

USB OTA update:

- Firmware can be updated via Openloop Connect (USB)
- Transferred data is integrity-checked chunk by chunk, and the whole image is verified with SHA-256 and an RSA-3072 signature (see §6)

4.3 BLE Communication ✓ Implemented

Protocol: Ledger Nano X-compatible GATT

Item	Value
Service UUID	13D63400-2C97-0004-0000-4C6564676572
Notify Characteristic	13D63400-2C97-0004-0001-4C6564676572
Write Characteristic	13D63400-2C97-0004-0002-4C6564676572
Device name	“Openloop”
MTU	247 bytes

Security:

Item	Setting
Pairing method	Secure Connections (LE SC)
Security level	Level 2 (MITM protection)
Authentication method	Numeric Comparison (6-digit code)
Encryption	AES-CCM (128-bit)
Bonding	NVS-persisted (up to 3 devices)

Pairing:

- ✓ Numeric Comparison (6-digit PIN display)
- ✓ Bonding information saved in NVS
- ✓ Unpair function
- ✓ Automatic disconnect after 60 seconds of inactivity

Supported applications:

- ✓ **MetaMask Mobile**: iOS/Android (verified)
- ✓ **Rabby Wallet (mobile)**: iOS/Android

BLE OTA update:

- Firmware can be updated via Openloop Connect (iOS/Android) over BLE
- Uses a dedicated service separate from the wallet service, and the whole image is verified with SHA-256 and an RSA-3072 signature

Supported OTA tools:

- ✓ Openloop Connect (iOS/Android mobile app)

4.4 APDU Protocol (common to USB/BLE) ✓ Implemented

APDU command processing is implemented in the `apdu_handler` component and is shared by both the USB HID and BLE transport layers. Wallet applications have access to the same functionality whether connected over USB or BLE.

APDU CLA support levels:

Protocol	Compatibility	Notes
CLA=0xE0 (Legacy)	✓ Partial support	Main commands for Bitcoin/ Ethereum/XRP/Solana/TRON
CLA=0xE1 (Bitcoin v2)	▲ Minimal	GET_EXTENDED_PUBKEY, GET_MASTER_FINGERPRINT only
CLA=0xB0 (Common)	✓ Supported	OPEN_APP, GET_BATTERY_STATUS
CLA=0xF0 (Openloop)	✓ Proprietary	BTC PSBT signing, chain_id-tagged signing, GET_DEVICE_INFO

Emulated app versions:

App	Version	Notes
Dashboard (BOLOS)	2.0.6	For Ledger Live dashboard recognition
Bitcoin	2.0.6	Because the Legacy Protocol (CLA=0xE0) is used
Ethereum	1.17.0	MetaMask/Safe/Rabby-compatible (EIP-7702 support)
XRP	2.5.2	Meets the XRP Toolkit 2.0.0+ requirement
Solana	1.0.0	hw-app-solana-compatible
TRON	1.0.0	TronScan-compatible

- GET_VERSION (INS=0x01): Target ID=0x31100004 (Nano S), SE Version="2.1.0", MCU Version="1.12"

- GET_APP_CONFIGURATION (INS=0x06): returns the version of the currently open app (e.g. ETH=1.17.0 / XRP=2.5.2)

Automatic app switching:

To ensure connectivity with each wallet software, Openloop automatically switches its app mode based on the received APDU command. The user can choose the initial value with “Default currency” on the home screen, and it updates automatically in response to host wallet operations.

Trigger	Behavior
OPEN_APP (CLA=0xB0, INS=0x01 / CLA=0xE0, INS=0xD8)	→ specified currency mode (Bitcoin/Ethereum/XRP/Solana/Tron)
BIP32 path coin_type detection	→ corresponding currency mode (0’=BTC, 1’=BTC testnet, 60’=ETH, 144’=XRP, 501’=SOL, 195’=TRON)
Bitcoin signing INS (0x42, 0x44, 0x48, 0x4A)	→ automatically Bitcoin mode
CLA=0xF0 (Openloop proprietary command)	→ currency mode according to the command
Air Gap QR scan	→ detected currency mode

- Default: Ethereum mode (for MetaMask compatibility)
- Currency state is shared across USB/BLE (unified across transports)
- Because MetaMask and others re-specify the currency mode via OPEN_APP, the device auto-recovers even when the state is inconsistent

Connection handshake / status query commands (read-only):

CLA	INS	Command	Status	Description
0xE0	0x01	GET_VERSION	✓	Get firmware/app version (Ledger-compatible)
0xB0	0x01	OPEN_APP / GET_APP_INFO	✓	Lc=0 queries the current app, Lc>0 switches apps
0xE0	0xD8	OPEN_APP	✓	App switching (Safe/MetaMask-compatible)
0xF0	0xA0	GET_DEVICE_INFO	✓	Get device identity + state + capability information

CLA	INS	Command	Status	Description
0xB0	0xA7	GET_BATTERY_STATUS	✓	Get remaining battery level

Supported APDU commands - CLA=0xE0 (Bitcoin Legacy Protocol):

INS	Command	Status	Description
0x40	GET_WALLET_PUBLIC_KEY	✓	Get public key/ address
0x42	GET_TRUSTED_INPUT	✓	Process previous transaction
0x44	UNTRUSTED_HASH_TX_INPUT_START	✓	Start input hash
0x48	UNTRUSTED_HASH_SIGN	✓	Transaction signing
0x4A	UNTRUSTED_HASH_TX_INPUT_FINALIZE	✓	Finalize output hash

Supported APDU commands - CLA=0xE1 (Bitcoin v2 Protocol):

INS	Command	Status	Description
0x00	GET_EXTENDED_PUBKEY	✓	Get xpub (Sparrow support)
0x01	REGISTER_WALLET	✗	Register wallet policy
0x02	GET_WALLET_ADDRESS	✗	Policy-based address
0x03	SIGN_PSBT	✗	PSBT signing (※ supported via CLA=0xF0 or QR)
0x05	GET_MASTER_FINGERPRINT	✓	4-byte master fingerprint

Supported APDU commands - CLA=0xE0 (Ethereum Protocol):

INS	Command	Status	Description
0x02	GET_PUBLIC_KEY	✓	Get public key/ address
0x04	SIGN	✓	Transaction signing
0x06			Get app information

INS	Command	Status	Description
	GET_APP_CONFIGURATION	✓	
0x08	SIGN_PERSONAL_MESSAGE	✓	Message signing (EIP-191)
0x0C	SIGN_EIP712	✓	EIP-712 signing (64-byte prehash version / Safe support)
0x14	PROVIDE_ERC20_TOKEN_INFO	✓	Set ERC-20 token info (clear-signing)
0x16	PROVIDE_NFT_INFO	✓	Set NFT info (clear-signing)
0x1A	EIP712_STRUCT_DEFINITION	✓	EIP-712 v2 streaming: type definition (Ledger ETH App v1.10+ / MetaMask Mobile, Rabby)
0x1C	EIP712_STRUCT_IMPLEMENTATION	✓	EIP-712 v2 streaming: value (root / array / field)
0x1E	EIP712_FILTERING	▲	ack only (filtering not implemented — the host falls back to displaying raw fields)
0x34	SIGN_AUTH_7702	✓	EIP-7702 Authorization signing

Supported APDU commands - CLA=0xE0 (XRP Protocol):

INS	Command	Status	Description
0x02	GET_ADDRESS	✓	Get XRP address
0x04	SIGN_TX	✓	Transaction signing

Supported APDU commands - CLA=0xE0 (Solana Protocol, Ledger Solana-compatible):

INS	Command	Status	Description
0x05	GET_PUBKEY	✓	Get public key (raw 32-byte Ed25519)
0x06	SIGN_MESSAGE	✓	Transaction signing (P2 chunking support)

INS	Command	Status	Description
0x07	SIGN_OFFCHAIN_MESSAGE	✓	Off-chain message signing

- Ledger hw-app-solana-compatible: P2 chunking protocol (P2_EXTEND=0x01, P2_MORE=0x02)
- Automatic detection and skipping of the pathsCount byte

Supported APDU commands - CLA=0xE0 (TRON Protocol, routed by coin_type=195'):

INS	Command	Status	Description
0x02	GET_ADDRESS	✓	Get TRON address (Base58Check "T…")
0x04	SIGN_TX	✓	Transaction signing (SHA-256 hash)
0x08	SIGN_MESSAGE	✓	Message signing (personal_sign)

- Shares the same INS codes as the Ethereum Protocol (CLA=0xE0). Automatically routed by BIP32 path coin_type=195'
- INS=0x08 is auto-detected: coin_type=195' → TRON mode, otherwise → Ethereum mode

Supported APDU commands - CLA=0xF0 (Openloop proprietary protocol):

INS	Command	Status	Description
0xA0	GET_DEVICE_INFO	✓	Get device identity + state + capability information
0x70	BTC_SIGN_PSBT	✓	Receive and sign PSBT
0x71	BTC_GET_SIGNED_PSBT	✓	Get signed PSBT
0x72	BTC_GET_ACCOUNT_XPUB	✓	Get account xpub (BIP32 derivation)
0x73	BTC_SIGN_MESSAGE	✓	Bitcoin message signing (BIP-137)
0x08	ETH_SIGN_MESSAGE	✓	chain_id-tagged message signing

INS	Command	Status	Description
0x0C	ETH_SIGN_EIP712	✓	chain_id-tagged EIP-712 signing

- Exclusive to Openloop Connect. The chain_id parameter displays the exact network name on the confirmation screen
- BTC PSBT signing: supports direct PSBT signing that the compatible protocol cannot handle
- BTC SIGN_MESSAGE: Bitcoin-specific message signing (BIP-137 format)

GET_DEVICE_INFO response format (16 bytes):

Offset	Content	Size
0-1	Magic “OL” (0x4F 0x4C)	2
2-3	Model ID (BE). Openloop V1 = 0x0001	2
4	HW Revision	1
5-7	Main FW Version (major, minor, patch)	3
8-10	Recovery FW Version (major, minor, patch)	3
11-12	State bitmask (LE)	2
13-14	Features bitmask (LE)	2
15	Reserved	1

State bitmask (16-bit):

bit	Name	Meaning
0	WALLET_PROVISIONED	Wallet has been generated
1-3	LANG	Language index (3 bits, 0=EN, 1=JA)
4	USB_ENABLED	USB communication ON
5	USB_VID_COMPAT	USB VID compatible mode (0x2C97). 0 means standard (0x303A)
6	BLE_ENABLED	BLE communication ON
7	BLE_PAIRED	BLE paired
8	FIDO_ENABLED	FIDO/Passkey functionality ON

bit	Name	Meaning
9	PIV_ENABLED	PIV/PKCS#11 functionality ON
10-15	reserved	For future expansion

The Features bitmask is a bitmask representing hardware capabilities. Because the bit layout may change with firmware version, the public API does not treat it as a fixed value.

---

5. User Interface

5.1 Hardware Specifications

Item	Specification
Device	Openloop
Screen size	2.0-inch IPS LCD
Resolution	320 × 240 pixels
Touch input	Capacitive touch panel
Camera	GC0308 (0.3 MP) for QR scanning
Speaker	AW88298 (I2S audio)

5.2 Screen Layout

Swipe navigation (8 screens):

Screen	Function
Home	Displays receive address
Send	Camera preview, PSBT/transaction scanning, confirmation and signing
Receive	Displays address QR code
Wallet	New wallet, wallet linking, wallet deletion, recovery phrase
Passkey/PIV	Passkey ON/OFF, credential list/deletion/reset, PIV ON/OFF, PIV PIN management
Security	PIN setup, change, deletion
Currency	Displays supported networks and supported tokens, BTC Air Gap QR format switching

Screen	Function
Settings	Audio, language settings, USB settings, BLE settings, factory reset, about this product

- Scrollable navigation bar (86 px wide, 90 px spacing)
- Swipe left/right to switch screens
- “About” (version, copyright, licenses) is a submenu within SETTINGS

Other functional screens:

Screen	Access method
QR scan	Camera preview and signing flow on the SEND screen
Animated QR display	Displays signed data as a QR code after signing
Signed transaction display	Detailed review of the signing result

5.3 Multilingual Support

- ✓ **Japanese**: supported (Noto Sans CJK JP)
- ✓ **English**: supported
- Settings are persisted in NVS

5.4 Audio Feedback

Sound	Timing
Startup sound	On app startup
Click sound	On button tap
Progress sound	On reading a QR frame
Success sound	On successful operation
Error sound	On authentication failure or operation failure
Payment sound	On successful transaction signing

- Volume adjustment: 0-100% (Settings screen)
- Click Sound: ON/OFF toggle

6. Security

6.1 Security Implementation: SE050 DualSecure + eFuse

Secure Element SE050:

Item	Specification
Chip	NXP SE050 (EAL6+ certified)
Connection	I2C (400 kHz)
Signing keys	ECDSA secp256k1 / EdDSA Ed25519

DualSecure encryption:

The entropy (the source data of the mnemonic) is protected with **dual keys** — one on the ESP32 side and one on the SE050 side — and cannot be decrypted unless both are present. The ESP32-side key is burned into eFuse and is non-readable; the SE050-side private key never leaves the chip.

Signing key management:

Key type	Supported currencies
secp256k1	Bitcoin, Ethereum
EdDSA Ed25519	XRP, Solana

Signing keys are managed inside the SE050, and signing always completes inside the SE050 (private keys are never extracted from the secure element). Frequently used keys are kept on the device, and only the keys needed for signing are prepared on demand, speeding up repeated signing.

Encrypted storage:

Item	Specification
Entropy encryption	AES-256-GCM (256-bit key, 16-byte authentication tag for tamper detection)
Key derivation	Key derivation based on the SE050 ECDH shared secret
Wallet storage area	NVS encryption (XTS-AES, key derived at runtime from the eFuse HMAC key and never stored in flash)
Firmware integrity	Secure Boot V2 (RSA-3072 signature verification) boots only signed images

Item	Specification
eFuse	Keys burned into eFuse (irreversible and non-readable)

PIN authentication:

- 4- to 6-digit PIN code
- PIN lock levels (4 stages):
  - On sleep: PIN entry every time it wakes from sleep
  - On long sleep (default): PIN entry after long sleep and on startup
  - On startup only: PIN entry only at power-on
  - OFF: no PIN entry
  - ※ At all levels except OFF, PIN entry is required at startup and when displaying the recovery phrase
- Progressive lockout:
  - 0-2 failures: no lockout
  - 3 failures: 30-second lockout
  - 4 failures: 1-minute lockout
  - 5 failures: 5-minute lockout
  - 6 or more failures: 10-minute lockout
  - 10 or more failures: automatic reset to factory state (all data erased; not a permanent lock — recoverable from the recovery phrase)
- Confirmation entry is required when setting a PIN
- Reset-time-inconsistency attack countermeasure implemented

EULA acceptance on first boot:

- The terms of use (EULA) are displayed on first boot
- Wallet operations are unavailable until the user accepts
- The full text can be viewed on a smartphone via a QR code

Physical confirmation:

- Sign after reviewing the transaction details on the screen
- Explicit approval via the sign/cancel buttons

6.2 Security Highlights

- ✓ **Signing always inside the SE050**: signing keys are managed inside the SE050 and signing completes within the secure element (private keys never leave it)
- ✓ **Entropy protected by DualSecure**: cannot be decrypted unless both the ESP32 eFuse key and the SE050 key are present

- ✓ **Wallet data NVS-encrypted**: stored with XTS-AES (key derived at runtime from eFuse)
- ✓ **Secure Boot V2**: RSA-3072 signature verification boots only signed firmware
- ✓ **Seedless signing**: frequently used keys are kept on the device, signing without reloading the seed

6.3 Comparison with Commercial Wallets

Wallet	Signing method	Security level
Openloop	**Hardware signing inside the SE050**	**High**
Ledger Nano	Signing inside the SE	High
Trezor	Software signing inside the MCU	Medium
Keystone	Signing inside the SE	High
Jade	Software signing inside the MCU	Medium

6.4 Security Best Practices

- **Offline operation**: no internet connection required (the Wi-Fi feature is not used)
- **Firmware signature verification**: RSA-3072 signature + SHA256 verification on OTA update
- **Signing algorithms**:
  - Bitcoin: ECDSA secp256k1 (RFC 6979-compliant deterministic signing)
  - Ethereum: ECDSA secp256k1 (EIP-155 replay protection)
  - XRP: EdDSA Ed25519 / ECDSA secp256k1 (auto-detected from the address)
  - Solana: EdDSA Ed25519 (RFC 8032-compliant)
  - TRON: ECDSA secp256k1 (SHA-256 hash, V=27+recovery_id)
- **Random number generation**: SE050 TRNG (True Random Number Generator)

OTA firmware signature verification:

Item	Specification
Algorithm	RSA-3072 + SHA-256 (RSA-PSS)
Verification timing	On OTA completion (inside esp_ota_end())
On verification failure	OTA error (current FW retained)

7. Hardware Specifications

7.1 Hardware Specifications

Item	Specification
CPU	ESP32-S3 Dual-core Xtensa LX7 (240 MHz)
RAM	512KB SRAM + 8MB PSRAM
Flash	16MB
Display	2.0-inch IPS LCD (320×240) with touch
Camera	GC0308 (0.3 MP) for QR scanning
Speaker	AW88298 I2S audio amplifier (22.05 kHz/16-bit)
USB	USB Type-C
Bluetooth	BLE 5.0 (NimBLE)
Wi-Fi	2.4 GHz 802.11 b/g/n (not used)
Power	USB-powered / built-in Li-Po battery

7.2 Flash Partition Layout

Partition	Type	Offset	Size	Purpose
nvs	data (nvs)	0x9000	32KB	Settings data storage
nvs_secure	data (nvs)	0x11000	32KB	Wallet data (encrypted)
otadata	data (ota)	0x19000	8KB	OTA boot information
phy_init	data (phy)	0x1B000	4KB	Radio calibration
factory	app (factory)	0x20000	4MB	OTA Recovery Mini (recovery FW)
ota_0	app (ota_0)	0x420000	4MB	Main firmware slot 1
ota_1	app (ota_1)	0x820000	4MB	Main firmware slot 2
coredump	data (coredump)	0xC20000	64KB	Crash dump storage

Partition	Type	Offset	Size	Purpose
nvs_data	data (nvs)	0xC30000	256KB	CTAP2 credentials, registry data
jp_font_14	data	0xC70000	768KB	Japanese font (JIS X 0208)

OTA update:

- Three-slot configuration (factory + ota_0 + ota_1)
- Safe updates via A/B partition switching
- Safe update protocol with signature verification

7.3 Recovery Mode

An emergency recovery feature for when the main firmware cannot boot.

Three conditions for entering recovery mode:

Condition	Trigger	Protection level
Power button long-press	Release the power button during the startup sound	User operation
3 consecutive crashes	The main FW crashes 3 times in a row	Application level
APP ROLLBACK	Partition corrupted/invalid	Bootloader level

Power button operation:

1. Turn the device off
2. Hold the power button and turn it on
3. Release the power button when the startup sound (after about 1 second) plays
4. Releasing within 3 seconds enters recovery mode

OTA Recovery Mini features:

Feature	Description
USB OTA	HID + CDC composite USB device (both transports supported simultaneously)
BLE OTA	Via the Openloop-dedicated BLE service
Settings screen	Language switching, USB/BLE enablement, BLE pairing
Boot main FW	“Boot main” button (ota_0 → ota_1 fallback)

Feature	Description
Automatic power-off	Automatic shutdown after 5 minutes of inactivity with no USB/BLE connection
RSA signature verification	RSA-3072 + SHA256 signature verification on main FW update

> Note: the recovery FW only supports updating the main FW. The recovery FW itself cannot be OTA-updated.

Supported OTA tools:

- ✓ Openloop Connect (iOS/Android)

7.4 Crash Dump (Coredump)

Automatically saves debug information when the device crashes.

Item	Specification
Storage	coredump partition (64KB)
Save timing	On panic/abort
Contents	Stack trace, register state
Readout	Used by developers for failure analysis

8. Software Composition

8.1 Technology Stack

Component	Version
ESP-IDF	v5.4
LVGL	9.4.0-dev
ESP-BSP	Openloop custom BSP
mbedTLS	3.6.2
NimBLE	ESP-IDF built-in version
TinyUSB	ESP-IDF built-in version

Component	Version
secp256k1	Bitcoin Core (submodule)
trezor_crypto	trezor-crypto fork

8.2 Key Components

Component	Function
apdu_handler	APDU protocol processing
usb_hid	USB HID transport
usb_cdc	USB CDC transport (OTA/serial log)
ble_comm	BLE GATT transport
wire_protocol	Unified wire protocol
psbt	Bitcoin PSBT parser/serializer
bbqr / bbqr_wrapper	BBQr encode/decode
esp32_bc-ur	UR (Uniform Resources) processing
secure_storage	DualSecure encrypted storage
nvs_wallet	Wallet settings NVS management
se050	SE050 secure element control
network_registry	EVM network management (215 chains)
token_registry	ERC-20 token management (~2,600 tokens)
delegate_registry	EIP-7702 delegate contract management
ethereum	ETH/ERC-20 transaction processing
xrp	XRP transaction processing
solana	Solana transaction processing, AirGap UR
bitcoin_utils	Bitcoin-related utilities
currency_abstraction	Multi-currency abstraction layer (BTC/ETH/XRP/SOL/TRX)
aes_gcm	AES-256-GCM encryption
cbor_rpc	Device control/update protocol processing
operation_lock	Operation mutual exclusion
audio_manager	Audio feedback
language	Multilingual support (EN/JA)

Component	Function
ui_common	Common UI components
ctap2	FIDO2/CTAP2 protocol stack (MakeCredential, GetAssertion, U2F)
piv	PIV/PKCS#11 smart card functionality
ctaphid	CTAPHID packet framing (HID transport layer)
usb_hid_cbor	FIDO HID packet transport + CBOR processing
ota_manager	OTA update management (USB/BLE)
trezor_crypto	Cryptographic library (ed25519, secp256k1, SHA, HMAC, etc.)

8.3 Binary Size

- **Firmware size**: ~2.6MB
- **Available slot size**: 4MB/slot
- **Free space**: ~1.4MB (35%)

9. Compatibility with Existing Wallets

9.1 Supported Wallet Apps

App	Communication	BTC	ETH	XRP	SOL	TRX	Notes
MetaMask Mobile	BLE	-	✓	-	-	-	iOS/Android, recommended
MetaMask (PC)	USB	-	✓	-	-	-	WebHID, EIP-712 support
Sparrow Wallet	QR / USB	✓	-	-	-	-	P2WSH multisig support
Safe (Gnosis Safe)	USB	-	✓	-	-	-	EIP-712 multisig support
Rabby Wallet	USB / BLE	-	✓	-	-	-	

App	Communication	BTC	ETH	XRP	SOL	TRX	Notes
							Multi-chain support
XRP Toolkit	USB	-	-	✓	-	-	XRP only
Solflare	USB / BLE	-	-	-	✓	-	Solana only, Ledger-compatible
TronScan	USB	-	-	-	-	✓	Ledger HID-compatible, Shasta/ Mainnet
WalletConnect dApps	WalletConnect	-	✓	-	✓	✓	Via Openloop Connect, 600+ dApps
FIDO2 websites	USB HID	-	-	-	-	-	Passkey registration/ authentication (Windows/ macOS)
AirGap Vault	QR	✓	✓	✓	✓	-	UR-compatible
Keystone	QR	✓	✓	-	✓	-	UR-compatible
Ledger Live	-	✗	✗	✗	✗	✗	Not supported (device attestation)

9.2 Protocol Compatibility

Protocol	Compatibility	Supported currencies
Ledger Bitcoin App	✓ Compatible	BTC
Ledger Bitcoin v2	▲ Partial	BTC (xpub/fingerprint)
Ledger Ethereum App	✓ Compatible	ETH/EVM/TRX
Ledger XRP App	✓ Compatible	XRP
Ledger Solana App	✓ Compatible	SOL
Ledger TRON App	✓ Compatible	TRX
BIP174 (PSBT)	✓ Compatible	BTC
UR crypto-psbt	✓ Compatible	BTC
UR sol-sign-request/sol-signature	✓ Compatible	SOL
UR xrp-sign-request	✓ Compatible	XRP
BBQr V0/V1	✓ Compatible	BTC
EIP-191 (personal_sign)	✓ Compatible	ETH/TRX
EIP-712 (Typed Data)	✓ Compatible	ETH
CTAP2 (FIDO2)	✓ Compatible	Passkey
U2F (FIDO)	✓ Compatible	Passkey (backward compatibility)
WalletConnect v2	✓ Compatible	ETH/SOL/TRX

9.3 Comparison with Competing Products

Table 1: Basic specifications

Product	Secure element	Screen	Communication
Openloop	✓ **SE050 EAL6+**	**Touch**	**Air Gap, USB, BLE**
Ledger Nano X	✓ EAL5+	Buttons	USB, BLE

Product	Secure element	Screen	Communication
Ledger Stax	✓ EAL6+	Touch (E-Ink)	USB, BLE
Trezor Safe 5	✓ EAL6+	Touch	USB
Trezor Safe 7	✓ TROPIC01	Touch	USB, BLE
Keystone 3 Pro	✓ EAL5+ ×3	Touch	Air Gap
Jade Plus	✗ Virtual SE	Buttons	Air Gap, USB, BLE

Table 2: Supported ecosystems

Product	Bitcoin	EVM	XRP	Solana	TRON	Currencies/tokens
Openloop	✓	✓ **(215 chains)**	✓ **Ed25519/ secp256k1**	✓	✓	**~2,600**
Ledger	✓	✓	✓ secp256k1	✓	✓	5,500+
Trezor	✓	✓	✓	✓	✓	9,000+
Keystone	✓	✓	✓	✓	✗	5,500+
Jade	✓	✗	✗	✗	✗	BTC only

Table 3: Feature comparison

Feature	Openloop	Ledger Nano X	Keystone	Jade
Air Gap	✓	✗	✓	✓
USB	✓	✓	✗	✓
Bluetooth	✓	✓	✗	✓
Touchscreen	✓	✗	✓	✗
Chain/token name display	✓	✗	✓	✓
Passkey (FIDO2)	✓	✗	✗	✗

Feature	Openloop	Ledger Nano X	Keystone	Jade
Multi-chain	✓	✓	✓	✗

Openloop highlights:

- **Supports all three communication methods**: Air Gap, USB, and Bluetooth
- **SE + touchscreen + multi-chain**: a rare combination
- **XRP Ed25519 signing support**: auto-detected from the address
- **Full TRON support**: transaction signing + message signing + broadcast
- **Rich transaction confirmation screen**: displays chain/token names and transaction details
- **FIDO2/CTAP2 passkey support**: delivers both a hardware wallet and a security key in a single device

9.4 Native Mode Strategy

Openloop has two operating modes: "**Compatible mode**" and "**Native mode**".

Mode	VID/PID	Use case
Native mode	0x303A / 0x8341	Used with Openloop Connect, the Web SDK, etc. (default)
Compatible mode	0x2C97 / 0x1011	Used with existing wallet apps (Sparrow, Rabby, etc.)

The connection methods that operate in native mode keep expanding, and the reliance on compatible mode is declining.

Connection methods that support native mode:

Connection method	Path	Platform
WalletConnect	Via Openloop Connect	All platforms
LocalWebSocket	Via Openloop Connect	Desktop / Android
Safari Web Extension	Via Openloop Connect	iOS
WebHID	Direct connection from the browser	Desktop (Chrome, etc.)

Advantages of native mode:

- **Freedom to add features**: implement innovative features without worrying about compatibility
- **Optimized UX**: a user experience tailored to the Openloop hardware

- **Fast development**: no need to follow other vendors' specifications; develop on our own roadmap

With WalletConnect-compatible dApps (600+), WebHID-capable web apps, and Web3 dApps via a browser extension, an environment is taking shape in which **native mode alone is fully sufficient as a practical wallet**.

---

10. References

10.1 Bitcoin Technical Standards (BIP)

Standard	Description	URL
BIP-39	Mnemonic code	https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki
BIP-32	HD Wallets	https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki
BIP-44	Multi-account hierarchy	https://github.com/bitcoin/bips/blob/master/bip-0044.mediawiki
BIP-84	Native SegWit derivation path	https://github.com/bitcoin/bips/blob/master/bip-0084.mediawiki
BIP-143	SegWit Sighash	https://github.com/bitcoin/bips/blob/master/bip-0143.mediawiki
BIP-174	PSBT	https://github.com/bitcoin/bips/blob/master/bip-0174.mediawiki

10.2 Ethereum Technical Standards (EIP/ERC)

Standard	Description	URL
EIP-155	Replay protection	https://eips.ethereum.org/EIPS/eip-155
EIP-712	Typed Structured Data	https://eips.ethereum.org/EIPS/eip-712
EIP-1559	Fee Market Change	https://eips.ethereum.org/EIPS/eip-1559
EIP-2718	Typed Transaction	https://eips.ethereum.org/EIPS/eip-2718
EIP-4527	QR Code Data	https://eips.ethereum.org/EIPS/eip-4527
EIP-7702	Set Code for EOAs	https://eips.ethereum.org/EIPS/eip-7702
ERC-20	Token standard	https://eips.ethereum.org/EIPS/eip-20

10.3 XRP Ledger Technical Standards

Standard	Description	URL
XLS-11d	Secret Numbers	https://github.com/XRPLF/XRPL-Standards/tree/master/XLS-0011d-secret-numbers
XLS-12d	Secp256k1 Keys	https://github.com/XRPLF/XRPL-Standards/tree/master/XLS-0012d-secp256k1-keys
XRP Binary Format	Transaction serialization	https://xrpl.org/docs/references/protocol/binary-format
XRP Signing	Signing algorithm	https://xrpl.org/docs/concepts/transactions/finality-of-results/sign-the-transaction
XRP Address	Address format (Base58Check)	https://xrpl.org/docs/concepts/accounts/addresses
Payment Tx	Basic payment transaction	https://xrpl.org/docs/references/protocol/transactions/types/payment

Openloop implementation details:

- Key derivation: BIP-44 (m/44’/144’/0’/0/0)
- Signing: ECDSA secp256k1
- Address: rXXX⋯ format (Base58Check + “r” prefix)
- Transaction: direct binary format signing

10.4 Solana Technical Standards

Standard	Description	URL
Ed25519	EdDSA signing	https://ed25519.cr.yp.to/
Solana Transaction	Transaction format	https://solana.com/docs/core/transactions
sol-sign-request	Keystone-compatible QR signing request	https://github.com/KeystoneHQ/Keystone-developer-hub/blob/main/research/solana-qr-data-protocol.md
Solana JSON-RPC	RPC API	https://solana.com/docs/rpc

Openloop implementation details:

- Key derivation: SLIP-0010 Ed25519-compliant, variable path depth (2–5 levels)
  - WalletConnect/Solflare standard: m/44’/501’/0’/0’ (4 levels, all hardened)
  - Receive screen default: m/44’/501’/0’ (3 levels)
  - Because the derivation path depth varies by app, a variable path depth of 2–5 levels is supported

- Signing: EdDSA Ed25519 (RFC 8032)
- Address: Base58 (Base58 encoding of the 32-byte public key)
- Transaction: both Legacy and Versioned (v0) supported

10.5 TRON Technical Standards

Standard	Description	URL
TIP-1	TRON Address Standard	https://github.com/tronprotocol/tips/blob/master/tip-1.md
TRON Protocol	Protocol Buffers definitions	https://github.com/tronprotocol/protocol
TronGrid API	TRON HTTP/JSON-RPC API	https://www.trongrid.io/

Openloop implementation details:

- Key derivation: BIP-44-compliant (standard path m/44'/195'/0'/0/0)
- Transaction signing: ECDSA secp256k1 (rawData hashed with SHA-256, V=27+recovery_id)
- Message signing: TIP-191 (keccak256, following ETH personal_sign)
- Address: T··· format (keccak256 → 0x41 prefix → Base58Check)
- Transaction: signs the Protocol Buffers rawData hashed with SHA-256

10.6 FIDO2/CTAP2 Technical Standards

Standard	Description	URL
CTAP2	Client to Authenticator Protocol	https://fidoalliance.org/specs/fido-v2.1-ps-20210615/fido-client-to-authenticator-protocol-v2.1-ps-20210615.html
WebAuthn	Web Authentication API	https://www.w3.org/TR/webauthn-2/
U2F	Universal 2nd Factor	https://fidoalliance.org/specs/fido-u2f-v1.2-ps-20170411/
CTAPHID	HID Protocol	https://fidoalliance.org/specs/fido-v2.1-ps-20210615/fido-client-to-authenticator-protocol-v2.1-ps-20210615.html#usb

10.7 QR Code / Air Gap Communication Standards

Standard	Description	URL
UR	Uniform Resources	https://github.com/BlockchainCommons/Research/blob/master/papers/bcr-2020-005-ur.md
BBQr	Bitcoin Efficient QR	https://bbqr.org/
crypto-psbt	PSBT UR Type	https://github.com/BlockchainCommons/Research/blob/master/papers/bcr-2020-006-urtypes.md