**Layer-2 Networks by Base Blockchain**

Layer-2 networks are protocols built atop base blockchains to increase throughput and reduce fees. Below is a grouping of major Layer-2 networks under their respective base chains, with key details for each L2:

**Bitcoin Layer-2s**

* **Lightning Network (BTC Lightning):** A payment-channel network for instant Bitcoin transfers.
  + **Native Token:** *None* (uses BTC; no separate token).
  + **Gas/Fees:** Paid in BTC (satoshis) via routing fees. No on-chain gas, but off-chain LN transactions charge small BTC fees to nodes.
  + **Wallet Address Format:** Lightning uses invoices (Bolt11 format) usually starting with **lnbc** for Bitcoin mainnet [cryptomus.com](https://cryptomus.com/blog/how-to-create-usdt-trc-20-wallet?srsltid=AfmBOorU6yOlxWZKfzLjjP7bRPCRfav1z4HvzbDtkgBH34-bdbdJr-By#:~:text=How%20To%20Create%20USDT%20TRC,might%20look%20like%20this%3A%20TPAgKfYzRdK83Qocc4gXvEVu4jPKfeuer5) . For example: lnbc1pyd8spp5.... These encode the payment amount, destination node (node pubkey), etc. It’s not a persistent address like Bitcoin’s, but a one-time invoice string.
  + **Token Address Format:** N/A (Lightning doesn’t use token contracts or separate token addresses; it transfers actual BTC off-chain).
* **Liquid Network (Bitcoin Liquid):** A sidechain pegged to Bitcoin, designed by Blockstream for fast settlement and asset issuance.
  + **Native Token:** **L-BTC** (Liquid Bitcoin, a 1:1 BTC peg).
  + **Gas Token:** L-BTC (fees on Liquid are paid in L-BTC).
  + **Wallet Address Format:** Liquid uses Blech32 (confidential addresses) or Bech32. Liquid addresses often begin with **lq** or **EX** (unconfidential), and **VT** (confidential). Example unconfidential address: ex1qqfd....
  + **Token Address Format:** Assets issued on Liquid (Issued Assets) are tracked by asset IDs (a blinding key). There isn’t a human-readable “contract address”; instead, each asset (like L-USDT) has a unique asset ID (a 256-bit hash). These assets can be sent to Liquid addresses (which carry both asset ID and amount in the transaction output).
* **Rootstock (RSK):** A Bitcoin merge-mined sidechain enabling EVM smart contracts.
  + **Native Token:** **RBTC** – pegged 1:1 with BTC (RBTC is used on RSK).
  + **Gas Token:** RBTC (transactions on RSK use RBTC for gas, akin to how ETH is used on Ethereum).
  + **Wallet Address Format:** RSK is EVM-compatible; addresses use Ethereum format (0x + 40 hex). In fact, a user can use the same address on Ethereum and RSK. *Example:* 0x1bf4... on RSK.
  + **Token Address Format:** Same as Ethereum ERC-20. RSK has ERC-20 tokens (e.g. RIF token) with 0x addresses. For instance, RIF token contract on RSK has an address like 0x2acc95758f8b5f583470ba265eb685a8f45fc9d5.
* **Stacks:** (Sometimes considered a Bitcoin layer, though it’s a separate blockchain anchored to Bitcoin blocks.)
  + **Native Token:** **STX** – used on Stacks for transactions and staking (Stacking).
  + **Gas Token:** STX (Stacks has its own gas fees in STX).
  + **Wallet Address Format:** Stacks addresses start with **SP** or **SM** (mainnet) followed by Base58 chars. Example: SP2JNFQ… (they are derived from a hash of an ECDSA key, similarly anchored to BTC hash).
  + **Token Address Format:** Tokens on Stacks (SIP-010 FT standard) are identified by contract principal (which includes the deploying address and contract name). For example, a token contract might be SP3Kk…::token-name. The “address” portion is a Stacks address (SP… string).

**Ethereum Layer-2s**

* **Arbitrum One:** An Optimistic Rollup on Ethereum focusing on general smart contracts (high EVM compatibility).
  + **Native Token:** **ARB** (governance token for Arbitrum DAO). *Not required for usage.*
  + **Gas Token:** ETH (users pay Arbitrum fees in ETH, which is bridged from L1).
  + **Wallet Address Format:** Same as Ethereum (0x… hex) [support.metamask.io](https://support.metamask.io/start/learn/the-ethereum-address-format-and-why-it-matters-when-using-metamask/#:~:text=On%20Ethereum%20and%20other%20networks,They%27re%20also%20not%20case%20sensitive) – Arbitrum is EVM-equivalent.
  + **Token Address Format:** Same as Ethereum (ERC-20). Contracts have 0x addresses. (Arbitrum uses standard Ethereum tooling; e.g., Arbitrum’s wrapped ETH is at a 0x address.)
* **Arbitrum Nova:** A variant of Arbitrum optimized for social and gaming applications (uses AnyTrust technology).
  + **Native Token:** ARB (also governed by ARB token).
  + **Gas Token:** ETH (Nova fees are paid in ETH as well).
  + **Address Formats:** Same as Arbitrum One/Ethereum (0x addresses).
  + **Notes:** Nova sacrifices some trustlessness for ultra-low fees, intended for microtransactions (e.g., Reddit Community Points).
* **Optimism:** An Optimistic Rollup for Ethereum, securing transactions via fraud proofs.
  + **Native Token:** **OP** (Optimism token for governance). Not needed for transactions.
  + **Gas Token:** ETH (Optimism transactions require ETH for gas, like on mainnet).
  + **Wallet Address Format:** Ethereum-format 0x addresses (Optimism is EVM-compatible).
  + **Token Address Format:** Standard 0x addresses for contracts. (Many Ethereum dApps and tokens are deployed on Optimism identically.)
* **Polygon PoS (formerly Matic):** A sidechain often classed as an Ethereum L2 (validators stake on Ethereum).
  + **Native Token:** **MATIC** – for gas and staking (becoming POL in future).
  + **Gas Token:** MATIC (Polygon’s native currency for fees).
  + **Wallet Address Format:** 0x + 40 hex (same as Ethereum) [support.metamask.io](https://support.metamask.io/start/learn/the-ethereum-address-format-and-why-it-matters-when-using-metamask/#:~:text=On%20Ethereum%20and%20other%20networks,They%27re%20also%20not%20case%20sensitive) .
  + **Token Address Format:** ERC-20 style (0x addresses). Polygon supports the full Ethereum token standard set.
* **Polygon zkEVM:** A true Layer-2 zero-knowledge rollup by Polygon, fully EVM-equivalent using zk proofs.
  + **Native Token:** *None* (no separate token; uses ETH).
  + **Gas Token:** ETH (fees on Polygon zkEVM are paid in Ethereum).
  + **Addresses:** Same 0x format as Ethereum (it’s EVM-equivalent).
  + **Token Format:** Same (ERC-20 contracts with 0x addresses).
  + **Note:** Ensures Ethereum-level security via validity proofs.
* **zkSync Era:** A ZK-rollup on Ethereum supporting smart contracts (using Zinc VM and EVM compatibility).
  + **Native Token:** *None (as of 2024).* (zkSync may introduce a token, but currently uses ETH for gas.)
  + **Gas Token:** ETH (all zkSync Era fees are paid in ETH).
  + **Wallet Address Format:** Ethereum-like 0x addresses (zkSync Era uses account abstraction but still presents addresses as 0x hex).
  + **Token Address Format:** Standard 0x addresses for tokens (it supports ERC-20 contracts).
  + **Note:** Earlier zkSync Lite (v1) was payments-only and used an account key instead of Ethereum address, but Era uses normal addresses.
* **StarkNet:** A ZK-rollup using STARK proofs, with its own VM (Cairo).
  + **Native Token:** **STRK (StarkNet token)** – planned for governance and maybe fees (recently introduced, but not required for basic usage yet).
  + **Gas Token:** ETH *currently* (StarkNet transactions have been paid in ETH on Alpha; eventually might use STRK).
  + **Wallet Address Format:** StarkNet has its own address format (felt 251-bit). User addresses are often represented as hex strings (0x… but not 40 hex – StarkNet addresses are 64 hex digits as well, using Ethereum-style representation for convenience). Many wallets abstract this.
  + **Token Address Format:** If represented, also a 0x hex for contract addresses (the StarkNet token contract itself has an address on L2). Internally, the addresses are integers <2^251.
  + **Note:** StarkNet uses account abstraction by default. The address one uses is actually a contract account.
* **Loopring:** A Layer-2 zkRollup focused on trading and payments (uses zkSNARKs).
  + **Native Token:** **LRC** (Loopring token, used for protocol staking and can be used to pay fees).
  + **Gas Token:** Primarily ETH (Loopring L2 fees ultimately settle in ETH for L1 costs), but within L2, fees can be paid in LRC or the token being traded [medium.com](https://medium.com/loopring-protocol/guide-how-to-use-loopring-l2-a267d005255b#:~:text=Guide%3A%20How%20to%20use%20Loopring,your%20transaction%20will%20be) . For example, a swap might charge a fee in the token you’re swapping.
  + **Wallet Address Format:** Loopring uses Ethereum-based account model; users sign up with an Ethereum address which is then used to manage their L2 account (with an “account ID”). There isn’t a new address format – one’s Ethereum address essentially maps to a Loopring account.
  + **Token Address Format:** Loopring supports transferring ERC-20 tokens within the rollup. These tokens are identified by the same contract addresses as on L1 Ethereum (Loopring smart contracts map them). No separate address; if anything, an internal token ID.
* **Immutable X:** A Layer-2 (StarkEx-based) specializing in NFTs (gaming assets, etc.), using ZK proofs by StarkWare.
  + **Native Token:** **IMX** – utility and governance token for Immutable X (used for fees, staking, rewards).
  + **Gas Token:** ETH *or* IMX – IMX tokens can be used to pay part of fees on Immutable X; otherwise fees are mostly abstracted (many NFT trades on IMX are “gas-free” for users, costs borne by marketplace and partly paid in IMX).
  + **Wallet Address Format:** Users sign with Ethereum addresses; Immutable X uses an off-chain registration, but your Ethereum address is your identity (with a StarkKey behind the scenes). No separate address string; the system knows your ETH address and maps it to a StarkEx account.
  + **Token Address Format:** Assets on Immutable X are mainly NFTs and some ERC-20s (e.g., GODS token from Gods Unchained) supported via bridging. They are represented by the same addresses as L1 (since ultimately they can be withdrawn to L1 contract addresses). Immutable X itself does not display separate contract addresses on L2.
* **Boba Network:** An Optimistic Rollup fork (based on Optimism) with added features (e.g. hybrid compute).
  + **Native Token:** **BOBA** – governance token (and used for reduced fees on Boba).
  + **Gas Token:** ETH (users can pay fees in ETH, just like Optimism) **or** BOBA – Boba enabled paying gas in BOBA with a 25% discount [boba.network](https://boba.network/tokenomics/#:~:text=Network%20fees%20on%20all%20Boba,can%20propose%20community%20initiatives%2C) .
  + **Wallet Address Format:** Same 0x format (EVM chain).
  + **Token Address Format:** Same as Ethereum (contracts at 0x addresses). (BOBA token itself exists as an ERC-20 on both L1 and L2.)
* **Metis Andromeda:** An Optimistic Rollup (fork of Optimism) aimed at DAO and dApp scalability.
  + **Native Token:** **METIS** – used for staking sequencers and can be used for gas on Metis.
  + **Gas Token:** METIS (Metis chose to use its own token for L2 gas payments, unlike Arbitrum/Optimism).
  + **Wallet Address Format:** 0x addresses (EVM-compatible).
  + **Token Address Format:** 0x addresses for contracts. (METIS token on Metis L2 is native, and other tokens like USDC are bridged as 0x contracts.)

*(Other L2s include* ***Ribbon/Orbit Chains*** *for specific apps,* ***Sorare’s StarkEx****, etc., but the above list covers the most prominent.)*