BitVMX FORCE

A community-supported initiative to drive the establishment of **BitVMX** as the solution of choice in **Disputable Computing on Bitcoin**

September 2025

BitVMX

Building Advanced Bitcoin Protocols with BitVMX

Dive into the BitVMX Platform Architecture

Martin Jonas | BitVMX Principal Engineer
Ariel Futoransky | CSO

Fairgate

What is BitVMX ?

- Disputable Computation on Bitcoin
- Allows to define UTXO spend condition based on the result of program
- Any program compiled into RISC-V
- In particular ZKP Groth16 verifier, but could be other
- If all agree (off-chain) → Happy path
- If not → Dispute (on-chain)

Applications Using BitVMX

- Union Bridge
  - Decentralized bridge
  - 1/n honest assumption
  - Bridge BTC ⟶ RBTC (on rootstock L2 blockchain)

- Cardinal
  - Bitcoin NFT/Ordinal Defi mode
  - 1/n honest assumption
  - Allows to lock an ordinal and trade it on Cardano Blockchain
  - Uses T.O.O.P to reduce the need for operator collaterals

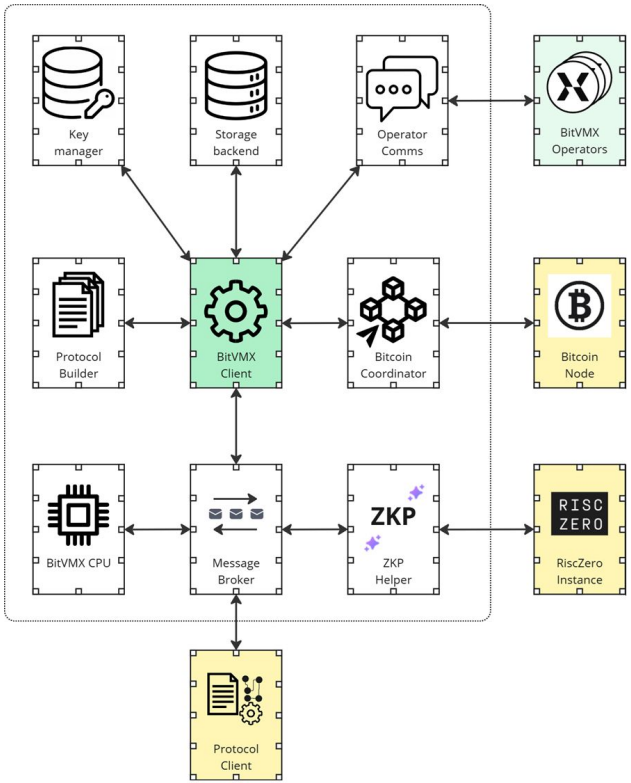
Design Decisions

- Pragmatic
- Protocol Agnostic
- Modular
- Fault Tolerant
- Adversarial Robust
- Scalable
- Standard TXs

Building Blocks

- Verifier execution off-chain / Validation on-chain
- Proof generation
- Graph of transactions agreed on setup
- Persistent Storage
- Tx Signing + WOTS
- Bitcoin monitoring and dispatching
- Connection between subsystems
- Connection between operators
- Orchestration

Architecture Overview



BitVMX-CPU

- Maps every RISC-V instructions into Bitcoin Script
- Off-chain evaluation of the program
- Dispute logic, to find the fault instruction in case of challenge

Protocol Builder

- Design complex protocols expressed in Bitcoin Transactions
- Creates different types of TXs (Segwit, Taproot)
- Stores metadata to know how to sign every input
- CPFP TXs
- Dust and TxFee calculation
- Graph visualization

Storage

- Use RocksDB as Backend (key-value database)
  - Performance
  - Transactions operations
  - Simplicity

- Encryption layer

Key Manager

- Key generation and handling
- Supports signing
  - ECDSA
  - Schnorr
  - Musig2 Protocol for Aggregated Signatures
  - Winternitz OTS
  - RSA

Bitcoin Components

- Indexer
  - Keep track of current blockchain state
- Monitor
  - Track specific TXs (from protocol graphs)
  - Track external TXs (some conditions)
  - Confirm state of dispatched TXs
- Coordinator
  - Dispatch TXs requested by BitVMX
  - Ensures CPFP and RBF are funded and dispatched (increasing fees)

Communication

- Message Broker (using Tarpc)
  - Sends messages between components
  - TLS Encrypted
  - Pinned Certificates for Client and Server
  - Allow list
  - Routing
- Used to connect subsystems
- Communication between BitVMX Operators (setup/accept phases)

RiscZero Proof Generation Helper

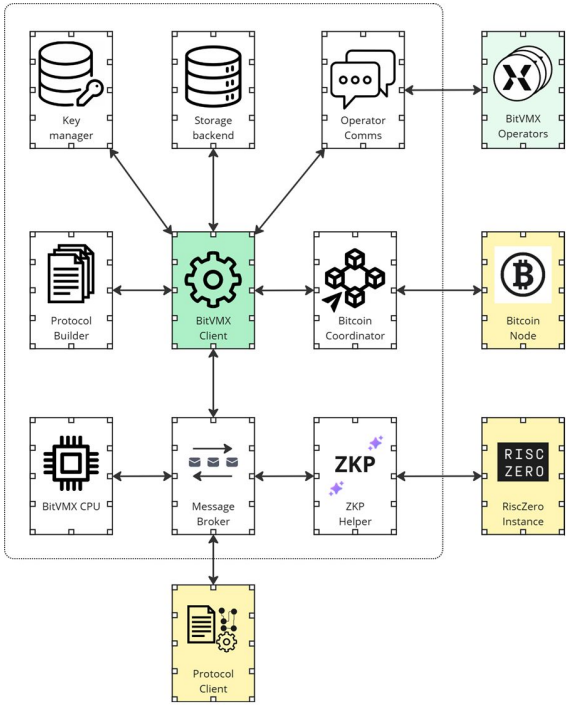
- Helps running the RiscZero proof generator, passing inputs and getting the proof
- Can run locally
- Start AWS instances with specialized hardware

BitVMX Client

- Handles requests made by the Protocol specific Clients (Union, Cardinal)
- Connect with other Operators to setup the protocols
- Tracks and dispatch TXs from the protocols
- Automatically reacts to specific TXs
    - Collaborate with valid flux
    - Challenging malicious activity

BitVMX Support for Garbled Circuit

- Garbled Circuits will be integrated in future versions
- BitVMX modular solution support switching BitVMX-CPU and Dispute Channel with Garbled Circuit Dispute
- Compatible with the rest of the components



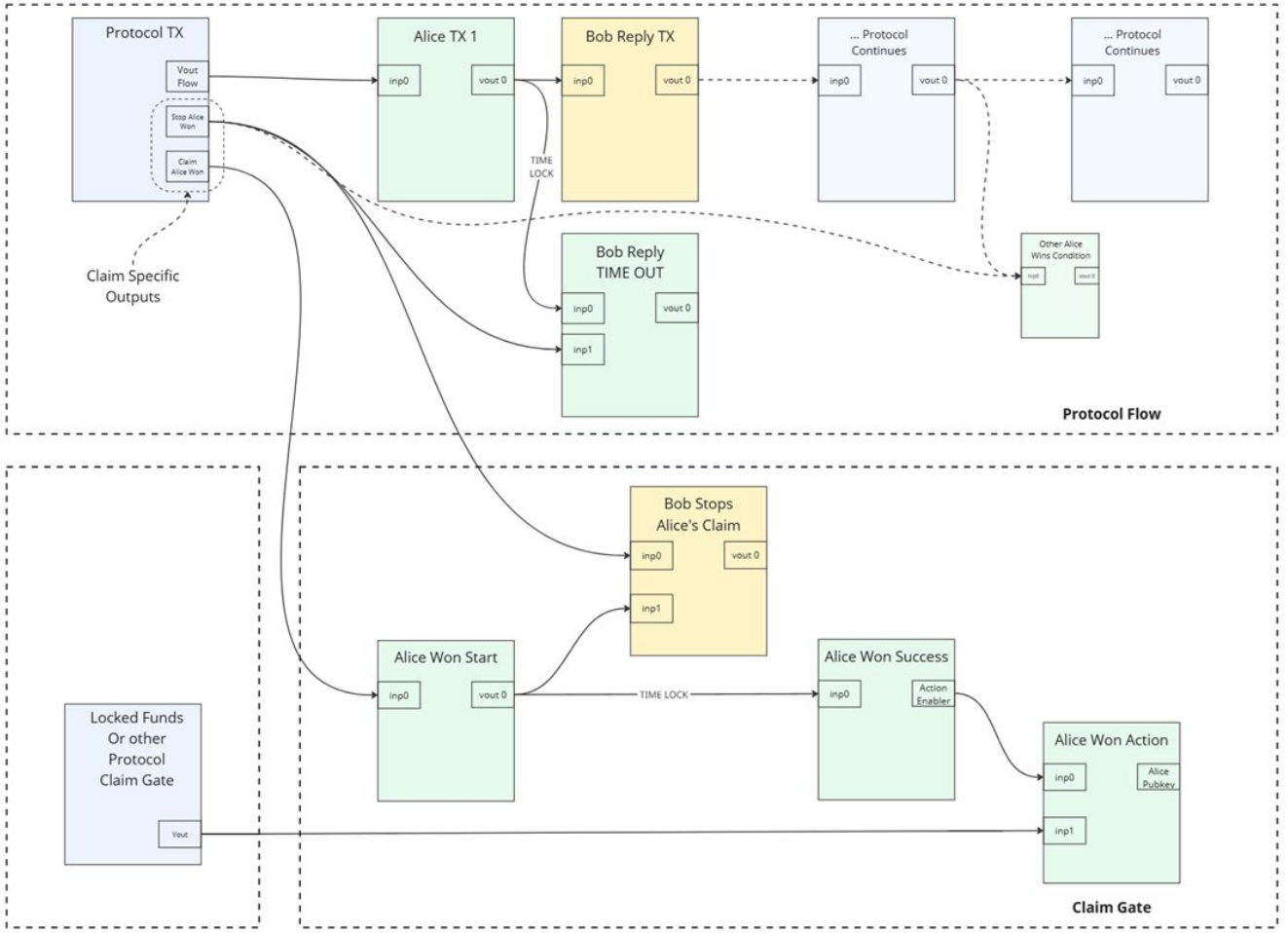
Open Source and BitVMX SDK

- Completing the Open Sourcing of all the repos
- Simplifies connection with BitVMX
- Simple backend layer with API
- Frontend as example

Some Innovations

Claim Gates

- Problems
  - Exponential nature of protocol txs subgraphs
  - Need to use maximum time lock (worst case)
- Allow to connect multiple conditions to a single tx with a single TxId
- Can be executed when any of the conditions is met


Protocol TX
Vout Flow
Stop Alice Won
Claim Alice Won
Claim Specific Outputs
Alice TX 1
inp0
vout 0
Bob Reply TX
inp0
vout 0
TIME LOCK
... Protocol Continues
inp0
vout 0
... Protocol Continues
inp0
vout 0
Bob Reply TIME OUT
inp0
vout 0
inp1
Other Alice Wins Condition
Protocol Flow
Bob Stops Alice's Claim
inp0
vout 0
inp1
Alice Won Start
inp0
vout 0
TIME LOCK
Alice Won Success
inp0
Action Enabler
Alice Won Action
inp0
Alice Pubkey
inp1
Locked Funds Or other Protocol Claim Gate
Vout
Claim Gate

Bitcoin Script Stack Tracker

- Helps creating complex bitcoin scripts
  - Used to implement the optimized versions of Blake3 and SHA256
  - All RISC-V IM32 Opcodes
  - Dispute channel Challenges
- Key features
  - Variable/Struct Tracking and manipulation
  - Debugging (inspired by Johan Halseth presentation on btc++ austin 2024)
  - Optimization

Bitcoin Script Stack Tracker

```
Interactive mode. n: next bp | p: previous bp | Step commands: <- (-1) | -> (+1) | Up (-100) | Down (+100) | PgUp (-100) | PgDown (+100) | +Shift (x10) | t (trim) | q (exit)
Step: 0 BP: start
Last opcode: OP_PUSHNUM_1
======= STACK: ======
id: 1       | size: 1       | name: number(0x1)          |  1
==== ALT-STACK: ====
```

https://github.com/FairgateLabs/rust-bitcoin-script-stack

https://bitvmx.org/knowledge/optimizing-algorithms-for-bitcoin-script-part-2

Minimum Relay Fee Problem

- Witness counts on the minimum relay fee
- Cost of verification scripts and the data encoded using WOTS are really high
- We had to lock on top of the DUST for the whole protocol 1 sat/vb for all the potential witnesses of TXs

Solution: Move Inputs to CPFP

- We were using CPFP on all TXs (impossible to forecast tx fee)
- Keep the graph connections for the flow (no inputs)
- Move the input and execution of big scripts into Speedup output
- Make the CPFP Tx to pay for the fee from the wallet for speedups
- Add some extra time lock TXs for penalising for not dispatching the CPFP
- Protocols more capital efficient

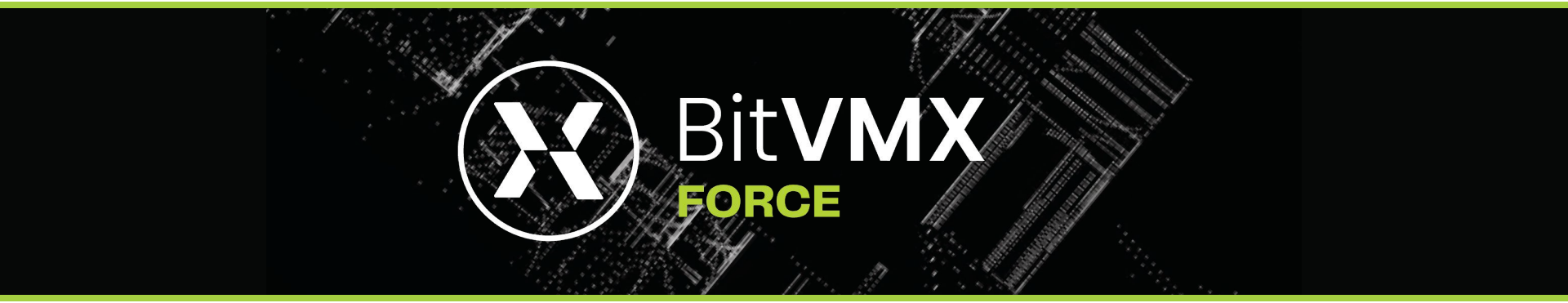
Hackathon

BitVMX
FORCE

bitcoin++

Berlin | October 2-4, 2025

Thank you!



Fairgate
https://www.fairgate.io/

BitVMX
https://bitvmx.org/

https://github.com/FairgateLabs