RGB smart contracts computing: AluVM and Schema

Maxim Orlovsky

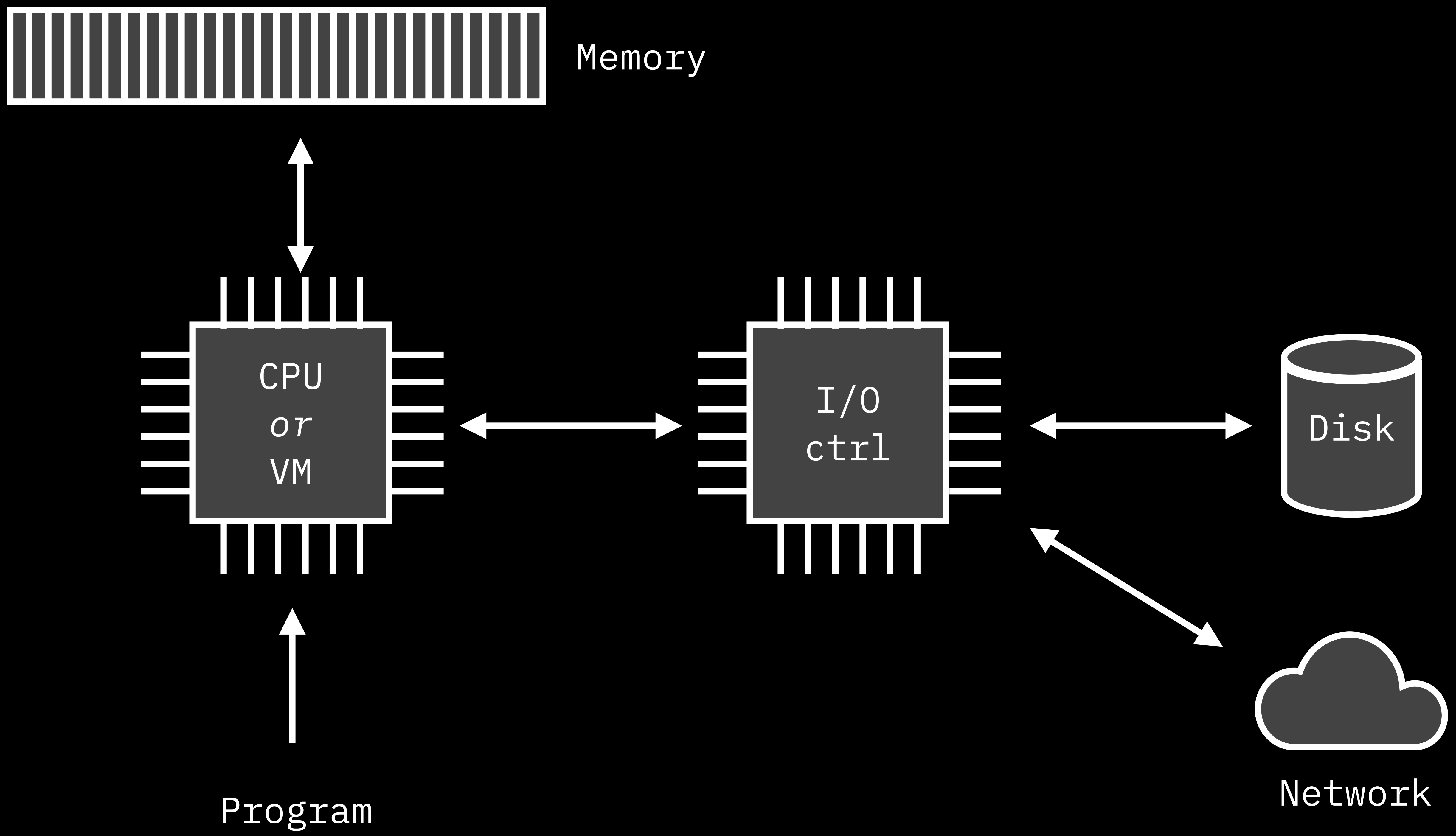
LNP/BP Standards Association &
Pandora Core AG

RGB client-side validation

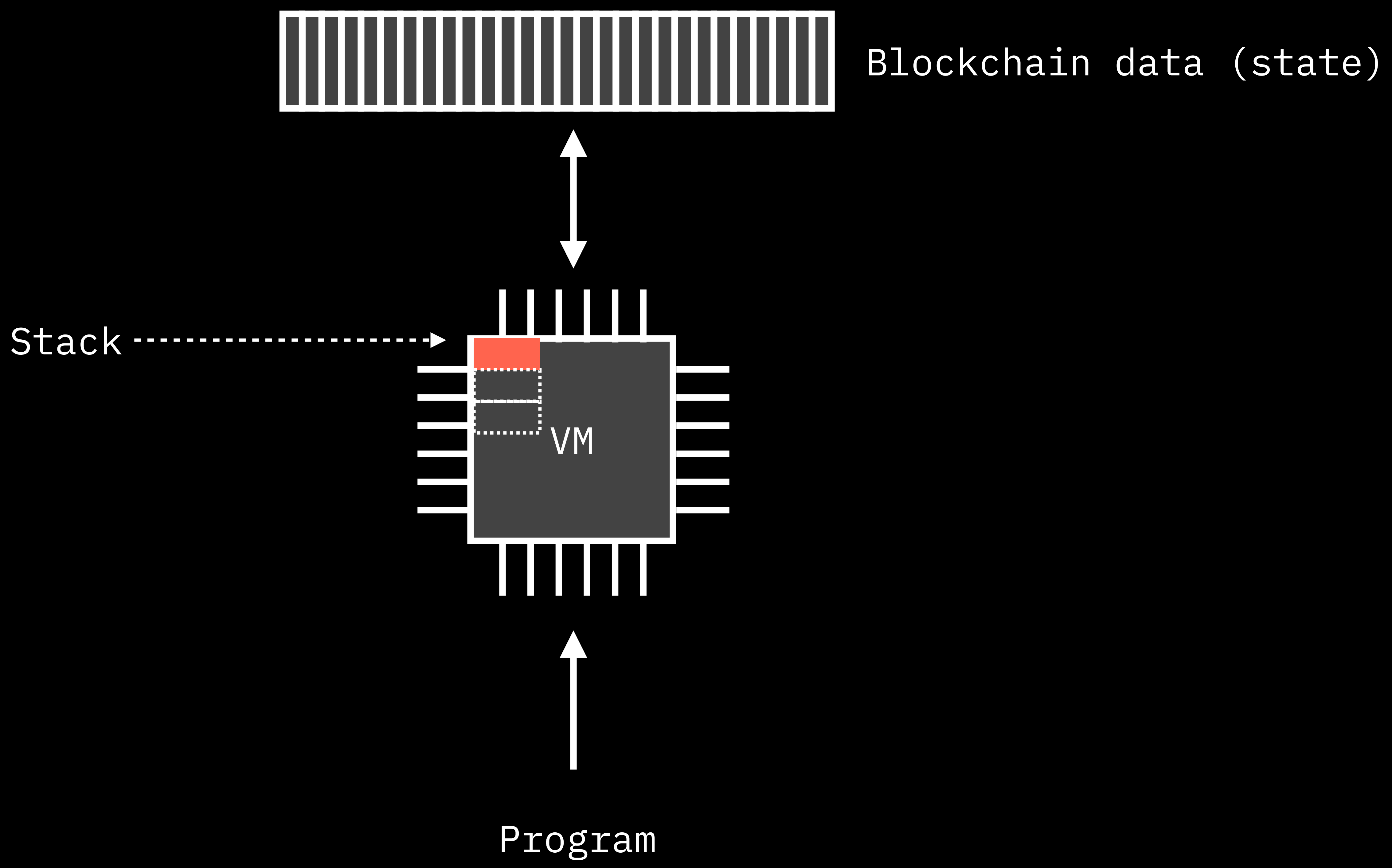
Only restricts bitcoin script-based rules, not extends them.
i.e. you can spend bitcoin output, but get invalid RGB state (loose asset, for instance by trying to inflate its supply)

- **Bitcoin script** controls who owns (who can change the state)
- **RGB schema rules** controls _how_ state _may_ be changed to continue being valid state
- **VM** performs dynamic checks on state changes during state transitions

CPUs and VMs for centralized computing



VMs for “blockchain” setup

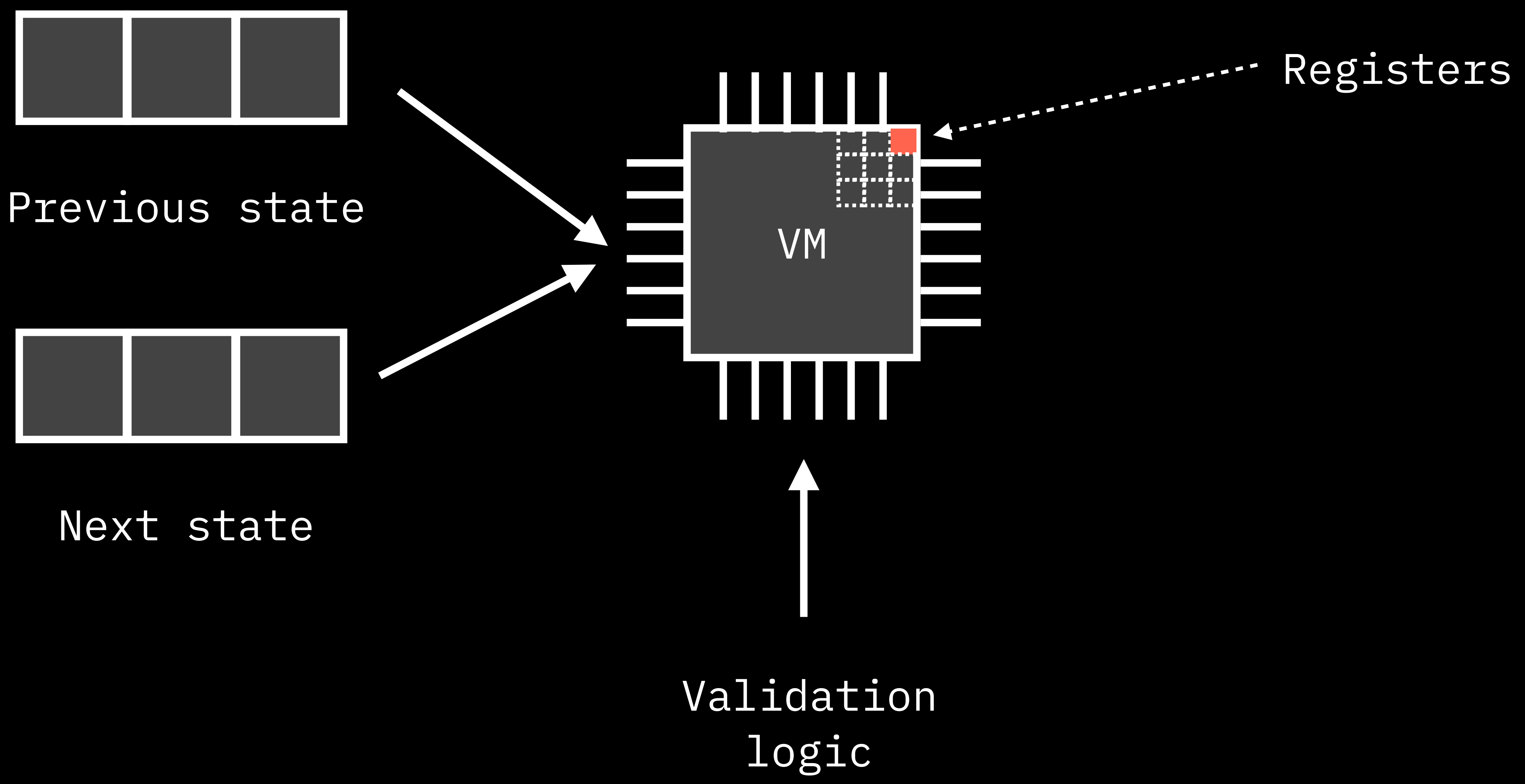


New paradigm (“client-side-validation”) required certain advancements in computing

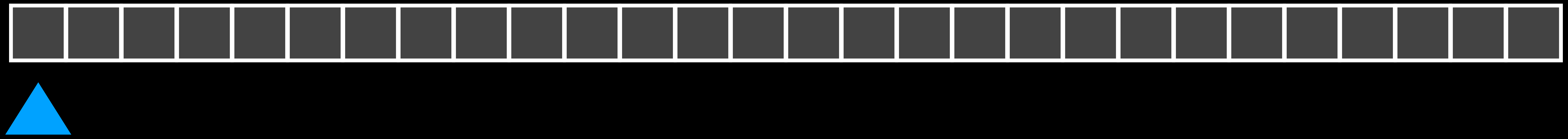
Client-side validation

- Ownership & “double-spent” prevention: re-using **Bitcoin script**
- Internal data consistency & completeness: **Schema**
- Changes in state: **AluVM**

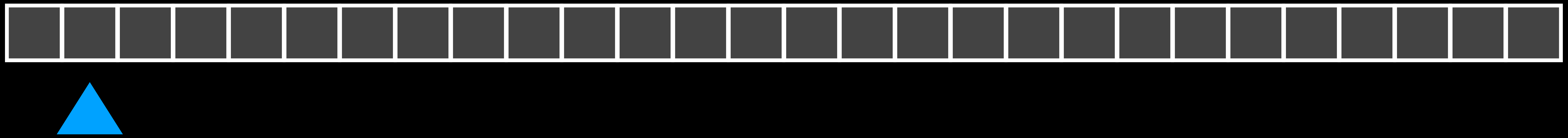
VM in client-side-validation



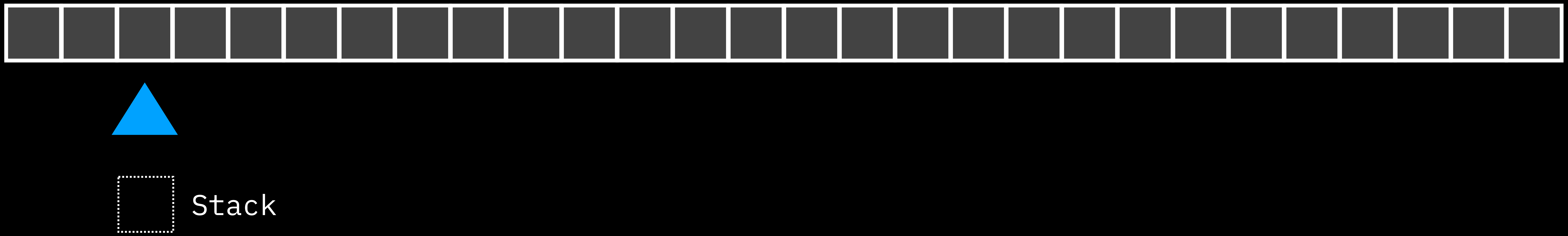
Stack-based VMs (Bitcoin Script, EVM, WASM)



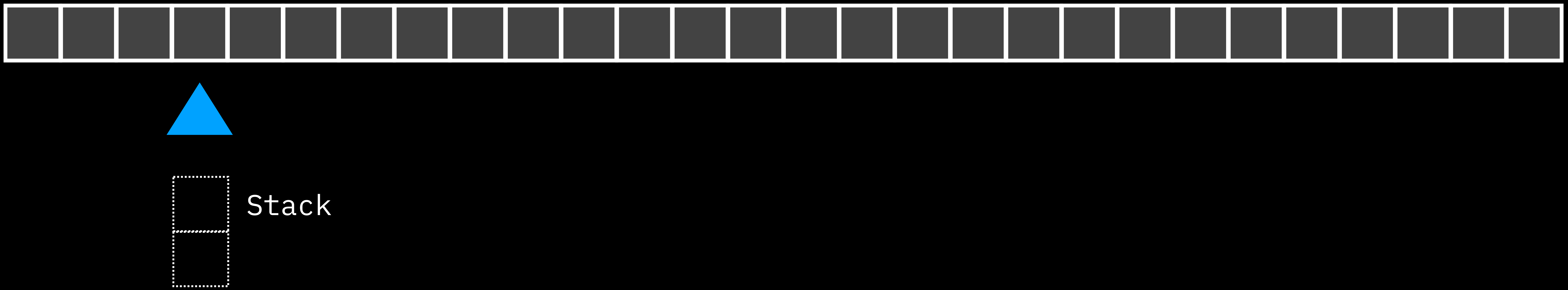
Stack-based VMs (Bitcoin Script, EVM, WASM)



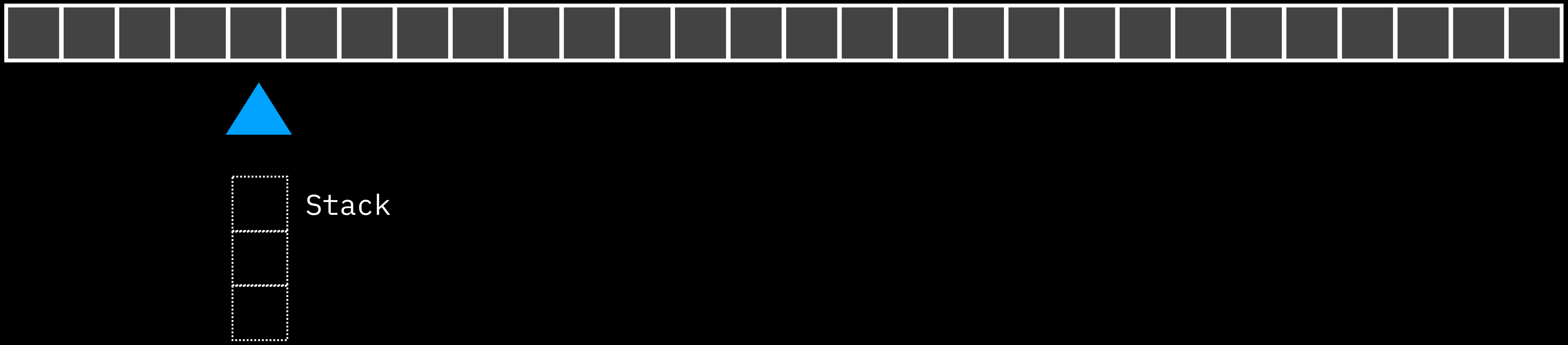
Stack-based VMs (Bitcoin Script, EVM, WASM)



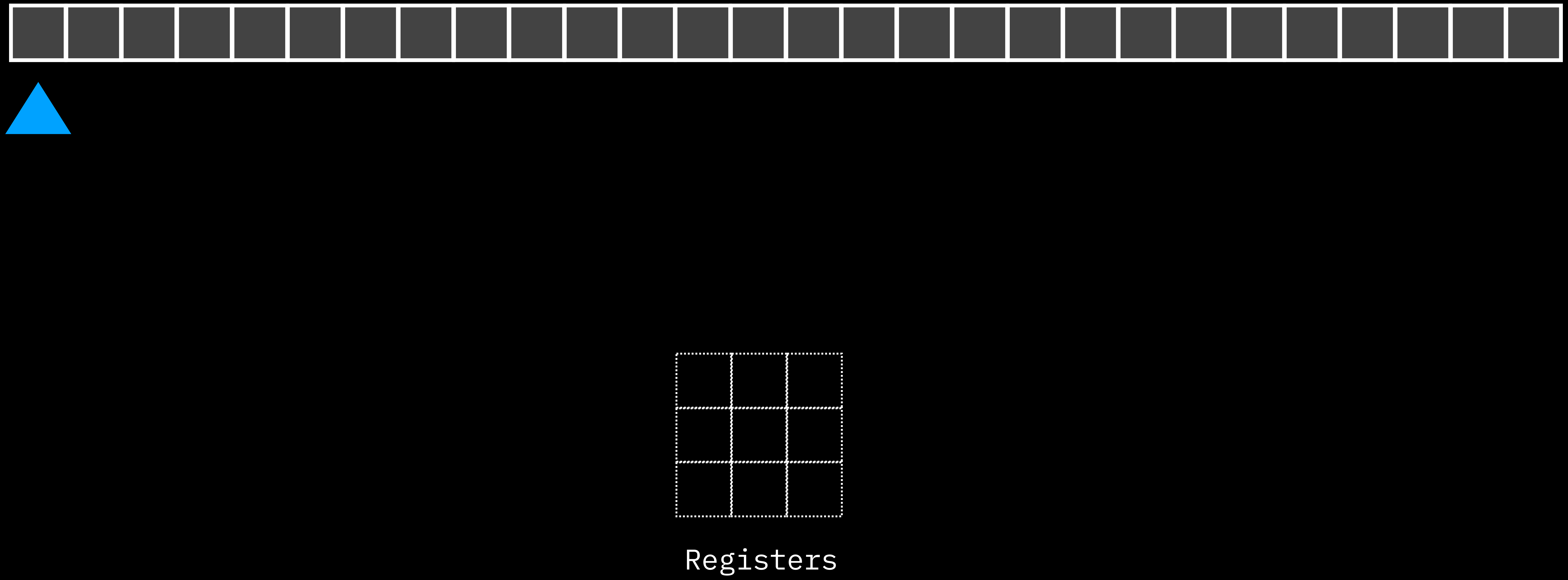
Stack-based VMs (Bitcoin Script, EVM, WASM)



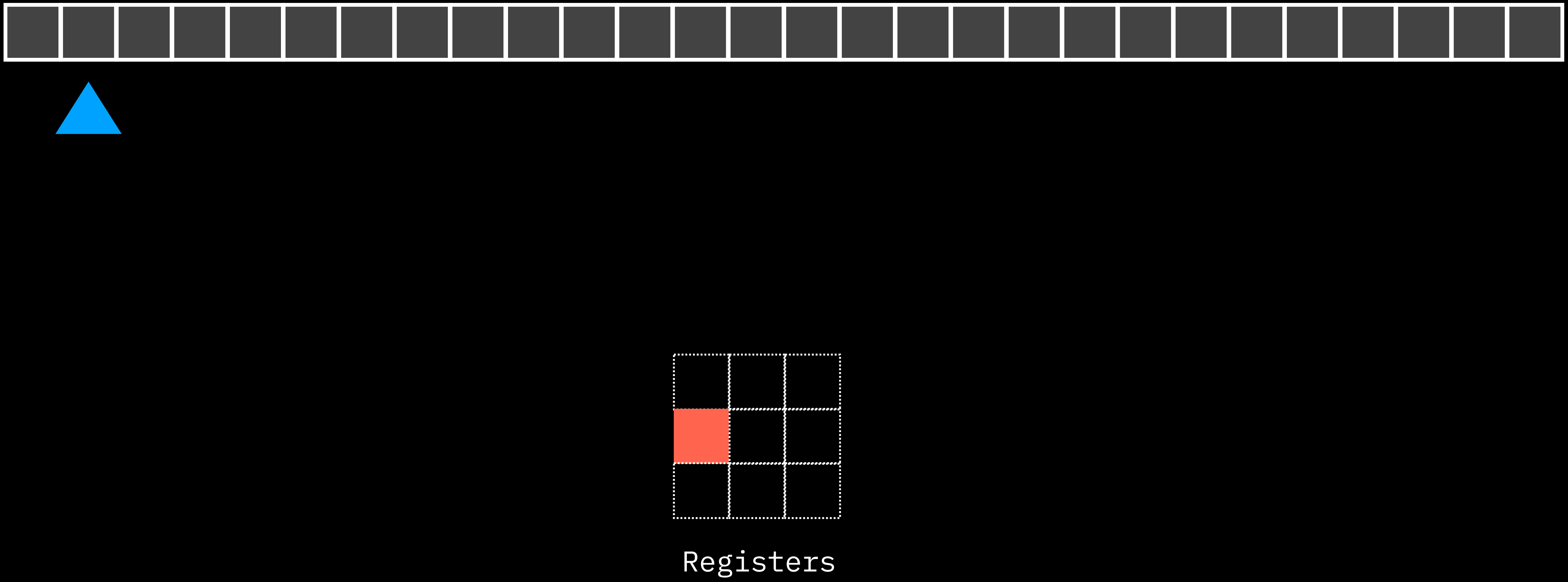
Stack-based VMs (Bitcoin Script, EVM, WASM)



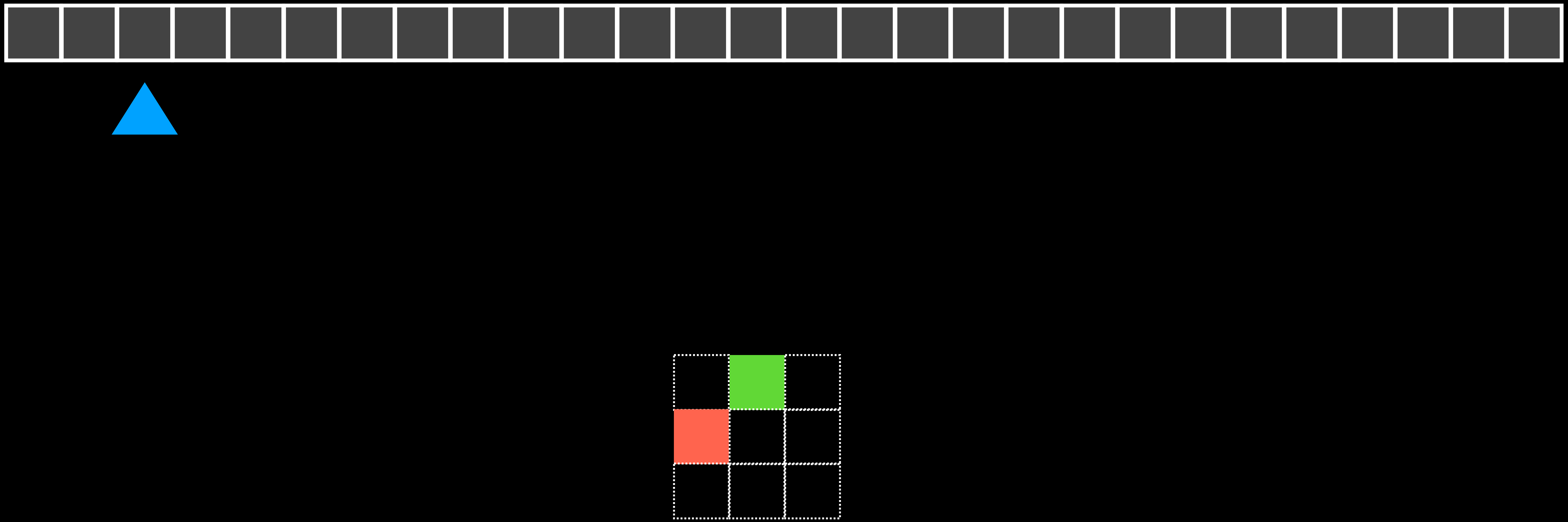
Register-based machines (x86_64, ARM, AluVM)



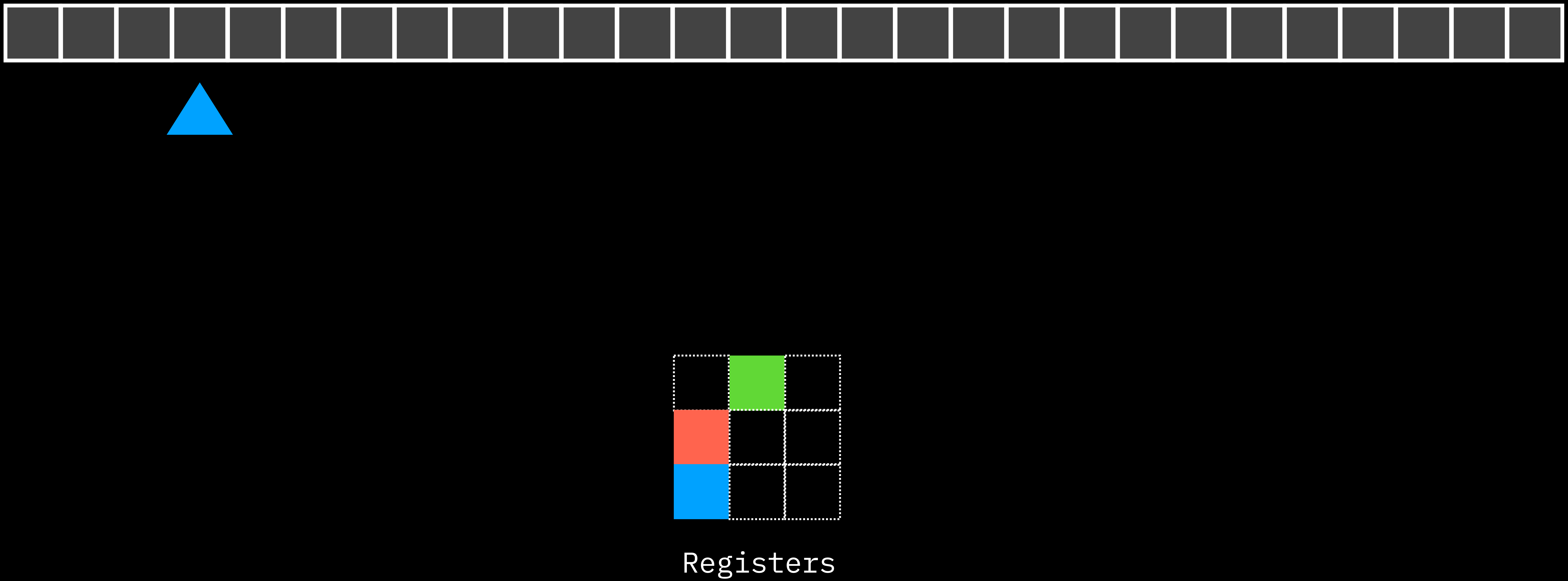
Register-based machines (x86_64, ARM, AluVM)



Register-based machines (x86_64, ARM, AluVM)



Register-based machines (x86_64, ARM, AluVM)



AluVM - from “arithmetic logic unit”

github.com/internet2-org/aluvm-spec

- Purely functional & arithmetical: each operation is an arithmetic function
- No external state; converts set of inputs into false/true validation result
- Extremely robust & deterministic: no exceptions are possible
    - no stack (register-based VM)
    - no random memory access
    - no I/O, memory allocations
- If it compiles, it will always run successfully
- Easy to be implemented in hardware, like in FPGAs

AluVM for RGB

- AluRE: AluVM runtime environment – github.com/internet2-org/alure
  - Pure rust: deterministic
  - Just <10k lines of code: easy to audit
- RGB uses AluRE provided with instructions to access RGB state data

Virtualization selection criteria

	Embedded procedures (*status quo*)	AluVM	WASM	Simplicity
Code audibility	Poor	Good	Average	Excellent (formally provable)
Safety	Endianess is tricky	Good	Average	Excellent
Cryptographic primitives	Grin-based, hard migration	Secp256k1	Not known, but smooth migration	Not ready
Schema validation implementation effort	Medium	Low	High	Extreme high
Can be used by other schema devs	No	Yes	Easily	Not really