January 9, 2026

An M. Rodriguez

2026-01-09 9:44

Figure 1: Hydra heads

Cryptocurrencies already enable offline digital cash.

Hydra Heads, a feature of Cardano[1], behave like cash:

- offline
- off-chain
- fee-less
- instant

[1] https://cardano.org/ is a commodity cryptocurrency with token ADA, a top-10 cryptocurrency by market capitalization.

- peer to peer
- private
- invisible to non-participants

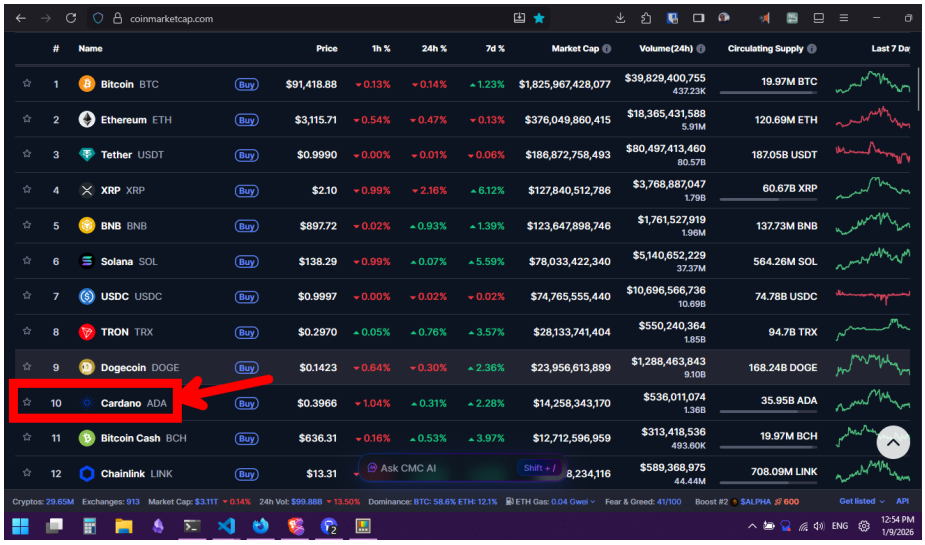

Figure 2: ADA, top ten cryptocurrency

Want to try it? See the instructions at the end of the article.

Coins vs. account balance

There are two broad families of cryptocurrencies:

- **coins**, like Bitcoin, and,
- **bank accounts**, like Ethereum

In both cases, "the bank" is the blockchain (L1).

In **coin-type** systems, like Bitcoin, your wallet holds discrete spendable objects, "coins", or UTxOs,

After a transaction is made (you give or receive coins), you can also "receive change", or new coins, or the "spendable output of the transaction".

Once a coin is spent, it cannot be spent again. This is cryptographically (which is to say "math") ensured.

In **account-type** systems, like Ethereum, you do not hold coins but instead have an "account balance". Before any value can be used, L1 (or a proxy) has to be consulted to avoid double spending. This requires consulting the main ledger (either directly or through a "trusted" proxy)

Also, **offline spending** is not possible because balances are global and mutable (somebody else could have already executed a claim on your values, in the ledger, after you took a snapshot of available balance).

This clearing of accounts is what makes current global financial system such a nightmare.

Because of this, offline, off-chain, peer-to-peer settlement is **only** possible in coin-type cryptocurrencies.

Bitcoin-like, not Ethereum-like

Most prominently **Ethereum** uses an **account-based model**. Also, by extension, all the other 20 standard, and other coins like:

Hydra Head properties

A Hydra Head provides:

- **Fully offline operation** Once opened, a head can operate offline indefinitely. Transactions require only peer connectivity.
- **Off-chain settlement** Transactions are never broadcast globally.
- **Instant finality** Payment and settlement are the same event.
- **Minerless and fee-less** No miners, sequencers, or leaders. Participants validate and finalize their own transactions at zero marginal cost.
- **Double-spend resistance** Enforced by the UTxO model and co-signed state transitions.
- **Privacy by default** No external observers exist.

A working mental model

Cardano L1 acts like an abstract bank where cryptocurrency is held.

Value is locked on L1 to fund a Hydra Head.

Inside the hydra head, that value behaves like cryptographic cash.

This mirrors physical cash: value is withdrawn from a bank and then circulates peer to peer.

Only some participants need to interact with L1. Others can remain fully offline forever.

A “cul-de-sac” monetary system

A Hydra Head is like a monetary “cul-de-sac”.

Figure 3: Automatic Teller Machine (ATM)

Figure 4: A cul-de-sac

Value enters and exits from Cardano L1, but value does not need to circulate through L1 to function.

Inside the head, money moves peer to peer **without miners**, **fees**, or global visibility.

There is no routing, no settlement layer beneath it, and no external dependency.

Like cash withdrawn from a bank, value can circulate indefinitely in a closed loop.

Only when participants choose to settle does L1 become relevant again.

Hydra Heads are self-contained monetary systems, anchored to L1 but not mediated by it.

Privacy

Figure 5: Privacy Policy

On Cardano L1, only two events are visible:

- the head is opened
- the head is closed

There is no on-chain record of:

- intermediate payments
- who paid whom

- when payments happened
- internal transaction structure

It's like a "no log policy". Just like cash.

If the head is private, nothing exists externally to reconstruct.

Privacy comes from the absence of observers.

Offline does not weaken security

Offline Hydra does not rely on trust.

- value cannot be double-spent
- invalid histories cannot be finalized on L1
- final settlement is always available if participants choose to exit

Security properties hold without continuous connectivity.

Paper, QR codes, and cash without paper

A Hydra transaction is data.

It can be exchanged via:

- QR codes
- printed strings
- NFC
- Bluetooth
- copy/paste
- any ad-hoc channel

A signed transaction can be printed on paper and handed to someone. When scanned and submitted, settlement is immediate.

This is **cash without paper**.

What if the same paper is printed twice?

This is analogous to photocopying a banknote.

- only the first settlement can succeed
- later attempts are rejected automatically
- no double spending occurs

The risk exists at acceptance time, as with physical cash. The ledger remains consistent.

Hydra prevents fraud from succeeding, not from being attempted.

Figure 6: alt text

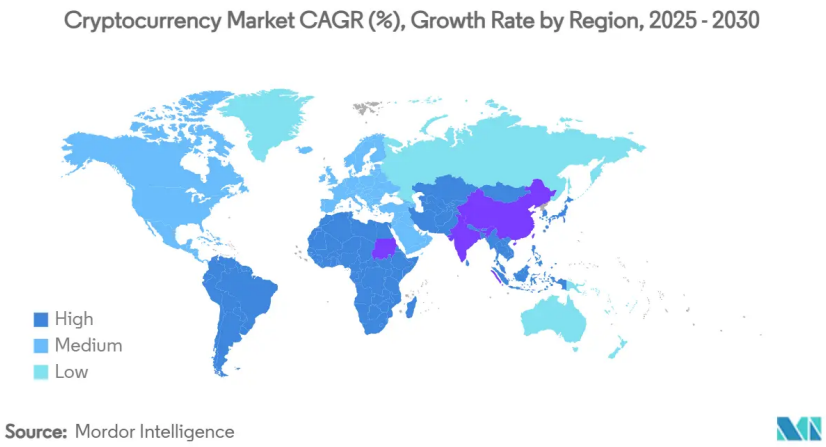

Figure 7: Cryptocurrency market by region, 2025-2030

World cash

Hydra Heads are purely peer to peer and offline-capable.

They work the same across frontiers:

- no jurisdiction
- no clearing system
- no geographic boundary

They are effectively **paperless, world cash**.

Why this matters

Hydra Heads provide a cash-like monetary mode:

- cryptographically secure
- double-spend resistant
- fee-less
- instant
- minerless
- independent of continuous connectivity

This mode exists in parallel to Cardano L1 and to traditional monetary systems, with L1 acting only as a value anchor.

Hydra Heads are not merely a scaling technique, they implement *offline digital cash*.

Figure 8: Crypto bill

Try it for yourself

Two cheat sheets follow:

1. Running a funded offline Hydra Head on Android
2. Building and submitting an offline transaction and observing state changes

Hydra Heads are not merely a scaling technique. They implement a cash-like settlement layer.

Part 1:

```
#################################################################
# GLOBAL CONSTANTS (EDIT ONCE IF NEEDED)
#################################################################

PHONE_IP=192.168.1.208
TERMUX_USER=u0_aXXX
TERMUX_PORT=8022

PHONE_HOME=/data/data/com.termux/files/home
HYDRA_BIN=$PHONE_HOME/hydra-node

PC_HOME=$HOME/src/hydra
KEYS_DIR=$PC_HOME/cardano-keys
```

```
OFFLINE_SEED=0000000000000000000000000000000000000000000000000000000000000000

################################################################
# 1) PHONE SETUP (ANDROID)
################################################################

# Install Termux from F-Droid (manual)
# Open Termux

pkg update -y
pkg install -y openssh proot-distro
passwd              # set: aaaa
sshd                # SSH on port 8022

# Check user (note it!)
whoami

################################################################
# 2) CONNECT FROM PC / WSL2
################################################################

ssh -p 8022 ${TERMUX_USER}@${PHONE_IP}

################################################################
# 3) CREATE DEBIAN (GLIBC) ON PHONE
################################################################

proot-distro install debian
proot-distro login debian

uname -m             # must be: aarch64

################################################################
# 4) INSTALL RUNTIME LIBS (PHONE / DEBIAN)
################################################################

apt update
apt install -y \
  ca-certificates curl jq \
  libgmp10 libsodium23 libssl3 zlib1g libnuma1 libsnappy1v5 \
  liblmdb0 libsecp256k1-2 \
  etcd-server

################################################################
# 5) GET HYDRA-NODE ARM64 (ON PC)
```

```
################################################################

mkdir -p $PC_HOME
cd $PC_HOME

docker pull --platform=linux/arm64 blinklabs/hydra-node:main-arm64v8

cid=$(docker create --platform=linux/arm64 blinklabs/hydra-node:main-arm64v8)
docker cp "$cid":/usr/local/bin/hydra-node ./hydra-node
docker rm "$cid"

file hydra-node       # ELF 64-bit ARM aarch64

################################################################
# 6) COPY HYDRA-NODE TO PHONE
################################################################

scp -P 8022 ./hydra-node \
  ${TERMUX_USER}@${PHONE_IP}:${PHONE_HOME}/hydra-node

################################################################
# 7) PHONE: MAKE EXECUTABLE
################################################################

chmod +x $HYDRA_BIN
$HYDRA_BIN --version

################################################################
# 8) PHONE: PROTOCOL PARAMETERS (CONWAY)
################################################################

cd $PHONE_HOME

curl -fsSL https://preprod.koios.rest/api/v1/cli_protocol_params \
  -o protocol-parameters.json

grep -n '"poolVotingThresholds"' protocol-parameters.json

################################################################
# 9) PHONE: SHELLEY GENESIS (PREPROD)
################################################################

# Copy from known config tree or scp from PC if you have it
# Must end up exactly here:
ls -l $PHONE_HOME/shelley-genesis.json
```

```
################################################################
# 10) PHONE: FAKE ADA (INITIAL UTXO)
################################################################

cat > $PHONE_HOME/utxo.json <<'EOF'
{
  "0000000000000000000000000000000000000000000000000000000000000000#0": {
    "address": "addr_test1vpgu5cghta2ydahksr55z88gsz6pjjp43zcatfefw3drw6sut9jxj",
    "value": { "lovelace": 20000000 }
  }
}
EOF

################################################################
# 11) PHONE: HYDRA KEYS (NODE A)
################################################################

mkdir -p /root/hydra-a/keys
$HYDRA_BIN gen-hydra-key --output-file /root/hydra-a/keys/hydra

################################################################
# 12) PHONE: RUN HYDRA NODE A
################################################################

$HYDRA_BIN \
  -n A \
  --hydra-signing-key /root/hydra-a/keys/hydra.sk \
  --offline-head-seed $OFFLINE_SEED \
  --ledger-genesis $PHONE_HOME/shelley-genesis.json \
  --initial-utxo $PHONE_HOME/utxo.json \
  --ledger-protocol-parameters $PHONE_HOME/protocol-parameters.json \
  --listen 0.0.0.0:5001 \
  --advertise ${PHONE_IP}:5001 \
  --api-host 0.0.0.0 \
  --api-port 4001 \
  --persistence-dir /root/hydra-a/state \
  --use-system-etcd

################################################################
# 13) VERIFY PHONE NODE
################################################################

curl http://127.0.0.1:4001/head
curl http://${PHONE_IP}:4001/head

################################################################
```

```
# 14) VERIFY FAKE FUNDS LOADED
################################################################

curl -s http://127.0.0.1:4001/head \
 | jq '.contents.coordinatedHeadState.confirmedSnapshot.initialUTxO'

################################################################
# 15) IF UTXO EMPTY → WIPE STATE & RESTART
################################################################

rm -rf /root/hydra-a/state
mkdir -p /root/hydra-a/state
# rerun step 12

################################################################
# 16) PC SIDE (NODE B - OFFLINE, FOR TX BUILD)
################################################################

# Use Docker cardano-cli only (no hydra-node needed on PC)

cd $KEYS_DIR
mkdir -p pay

docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest address key-gen \
    --verification-key-file /w/pay/b.vkey \
    --signing-key-file /w/pay/b.skey

docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest address build \
    --payment-verification-key-file /w/pay/b.vkey \
    --testnet-magic 1 \
    --out-file /w/pay/b.addr

################################################################
# NEXT STEP (FOR NEXT ASSISTANT)
################################################################
# Build + sign a tx spending:
#   0000...000#0
# Submit via Hydra WebSocket (NewTx)
# Observe /head changing on phone
################################################################
```

Part 2:

```
################################################################
# CHEAT SHEET #2 - OFFLINE TX INSIDE HYDRA HEAD
```

```
############################################################

# Assumptions (same as sheet #1)
PHONE_IP=192.168.1.208
PHONE_API=4001

PC_HOME=$HOME/src/hydra
KEYS_DIR=$PC_HOME/cardano-keys

TX_IN=0000000000000000000000000000000000000000000000000000000000000000#0

############################################################
# 1) PC: PREPARE ADDRESSES (A = receiver, B = sender)
############################################################

cd $KEYS_DIR
mkdir -p pay

# Generate keys for A (receiver)
docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest address key-gen \
    --verification-key-file /w/pay/a.vkey \
    --signing-key-file /w/pay/a.skey

docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest address build \
    --payment-verification-key-file /w/pay/a.vkey \
    --testnet-magic 1 \
    --out-file /w/pay/a.addr

# (Sender B keys already exist from sheet #1)
cat pay/a.addr
cat pay/b.addr

############################################################
# 2) PC: BUILD RAW TX (FEE = 0 FIRST)
############################################################

A_ADDR=$(cat pay/a.addr)
B_ADDR=$(cat pay/b.addr)

docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest transaction build-raw \
    --tx-in $TX_IN \
    --tx-out "$A_ADDR+1000000" \
    --tx-out "$B_ADDR+19000000" \
```

```
    --fee 0 \
    --protocol-params-file /w/protocol-parameters.json \
    --out-file /w/tx.draft

#############################################################
# 3) PC: CALCULATE MIN FEE
#############################################################

FEE=$(docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest transaction calculate-min-fee \
    --tx-body-file /w/tx.draft \
    --protocol-params-file /w/protocol-parameters.json \
    --witness-count 1 | awk '{print $1}')

CHANGE=$((19000000 - FEE))

echo "FEE=$FEE"
echo "CHANGE=$CHANGE"

#############################################################
# 4) PC: BUILD FINAL TX
#############################################################

docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest transaction build-raw \
    --tx-in $TX_IN \
    --tx-out "$A_ADDR+1000000" \
    --tx-out "$B_ADDR+$CHANGE" \
    --fee "$FEE" \
    --protocol-params-file /w/protocol-parameters.json \
    --out-file /w/tx.raw

#############################################################
# 5) PC: SIGN TX (B SIGNS)
#############################################################

docker run --rm -v "$PWD":/w ghcr.io/intersectmbo/cardano-node:10.6.1 \
  cli latest transaction sign \
    --tx-body-file /w/tx.raw \
    --signing-key-file /w/pay/b.skey \
    --out-file /w/tx.signed

ls -l tx.signed
jq -r '.cborHex | length' tx.signed   # must be > 0

#############################################################
```

```
# 6) PC: SUBMIT TX TO HYDRA (WEBSOCKET)
################################################################

# Install websocat once if missing
sudo curl -L \
  https://github.com/vi/websocat/releases/download/v1.12.0/websocat.x86_64-unknown-linux-mus
  -o /usr/local/bin/websocat
sudo chmod +x /usr/local/bin/websocat

# Submit NewTx
jq -c '{tag:"NewTx", transaction:.}' tx.signed \
 | websocat ws://${PHONE_IP}:${PHONE_API}/

################################################################
# 7) VERIFY TX EFFECT (PHONE)
################################################################

curl -s http://${PHONE_IP}:${PHONE_API}/head \
 | jq '.contents.coordinatedHeadState.version'

curl -s http://${PHONE_IP}:${PHONE_API}/head \
 | jq '.contents.coordinatedHeadState.confirmedSnapshot.initialUTxO'

################################################################
# EXPECTED RESULT
################################################################
# - version increments (e.g. 0 -> 1)
# - original TX_IN disappears
# - new UTxOs appear for A and B
################################################################
```