CRYPTOCURRENCY AND BLOCKCHAIN TECHNOLOGIES

U23CSV46 | Complete Study Materials

L:3 T:0 P:0 C:3 | Total: 45 Periods

Unit	Topic	Periods
Unit I	Introduction to Blockchain	9
Unit II	Bitcoin and Cryptocurrency	9
Unit III	Bit Coin Consensus	9
Unit IV	Hyper Ledger Fabric & Ethereum	9
Unit V	Block Chain Applications	9

UNIT I
INTRODUCTION TO BLOCKCHAIN

1.1 What is a Blockchain?

A blockchain is a distributed, decentralized digital ledger that records transactions across a network of computers. It is designed so that records cannot be altered retroactively without altering all subsequent blocks — making it tamper-resistant and highly secure.

Think of it as a chain of digital 'blocks', where each block contains a set of transactions, a timestamp, and a cryptographic link to the previous block.

1.2 Public Ledgers

A public ledger in blockchain refers to a record-keeping system that is:

- Open and accessible to all participants in the network
- Transparent — everyone can view transactions
- Immutable — once recorded, data cannot be changed
- Decentralized — no single entity controls it

Blockchain as a Public Ledger means all confirmed transactions are recorded and visible to any participant. Bitcoin uses blockchain as its public ledger to record all BTC transactions globally.

1.3 Block Structure in a Blockchain

Component	Description
Block Header	Contains metadata: previous hash, timestamp, nonce, Merkle root
Block Body	Contains list of validated transactions
Hash	Unique digital fingerprint of the block (SHA-256)
Previous Hash	Links this block to the previous one, forming the 'chain'
Nonce	A number miners vary to find a valid hash (Proof of Work)
Timestamp	Date and time when the block was created

1.4 Transactions in a Blockchain

A transaction is a transfer of value or data between two parties recorded on the blockchain. The lifecycle:

1. User initiates a transaction (e.g., sending Bitcoin)
2. Transaction is broadcast to the peer-to-peer network
3. Network nodes validate the transaction
4. Validated transactions are grouped into a block
5. Block is added to the blockchain after consensus

6. Transaction is confirmed and permanently recorded

1.5 The Chain and the Longest Chain Rule

Each block contains the hash of the previous block, creating a chain. If two valid chains exist (a fork), the network follows the LONGEST chain — the one with the most cumulative Proof of Work. This rule prevents double-spending and ensures consensus.

1.6 Permissioned vs. Permissionless Blockchain

Feature	Permissionless	Permissioned
Access	Anyone can join	Restricted to approved members
Example	Bitcoin, Ethereum	Hyperledger Fabric
Transparency	Fully public	Selective/private
Speed	Slower	Faster
Use Case	Cryptocurrency	Enterprise/Supply Chain

1.7 Cryptographic Hash Functions

A hash function takes an input of any size and produces a fixed-size output (hash/digest). Properties:

- Deterministic: same input always gives same output
- Fast computation
- Pre-image resistance: cannot reverse-engineer input from hash
- Avalanche effect: small input change causes completely different hash
- Collision resistant: two different inputs should not produce the same hash

Bitcoin uses SHA-256 (Secure Hash Algorithm 256-bit). Example:

```
Input: 'Hello' → SHA-256 → 185f8db32921bd46d35cc7e68e...  (64 hex chars)
```

1.8 Hash Pointer and Merkle Tree

Hash Pointer: A data structure that stores the address (pointer) of some information, along with the cryptographic hash of that information. It lets you verify the data hasn't been changed.

Merkle Tree: A binary tree where:

- Leaf nodes contain hashes of individual transactions
- Non-leaf nodes contain hashes of their children
- Root node (Merkle Root) represents all transactions in the block
- Allows efficient and secure verification of large data sets
- Used in Bitcoin to verify transactions without downloading entire blockchain

1.9 Key Definitions — Unit I

Term	Definition
Blockchain	Distributed ledger of chained blocks containing transaction data
Hash	Fixed-length cryptographic output from any input
Merkle Root	Single hash representing all transactions in a block
Nonce	Number used once; varied by miners during Proof of Work
Fork	When blockchain splits into two chains temporarily or permanently
Node	A computer participating in the blockchain network

UNIT II
BITCOIN AND CRYPTOCURRENCY

2.1 What is Cryptocurrency?

Cryptocurrency is a digital or virtual currency secured by cryptography, making it nearly impossible to counterfeit or double-spend. It operates on decentralized networks using blockchain technology.

- No central authority (bank/government) controls it
- Transactions are peer-to-peer
- Secured using public-key cryptography
- Bitcoin (BTC) was the first cryptocurrency, created in 2009 by Satoshi Nakamoto

2.2 Creation of Coins — Mining

New Bitcoin coins are created through a process called Mining:

7. Miners collect pending transactions from the mempool
8. They bundle transactions into a candidate block
9. They solve a complex mathematical puzzle (find a valid Nonce)
10. The first miner to solve the puzzle broadcasts the block
11. Other nodes verify and accept the block
12. Miner receives a block reward (newly created BTC) + transaction fees

Bitcoin halving occurs every 210,000 blocks (~4 years), cutting the block reward in half. The reward started at 50 BTC, currently 3.125 BTC (after 2024 halving).

2.3 Payments and Double Spending

Double Spending is attempting to spend the same digital coin twice. Bitcoin solves this through:

- Blockchain's immutable ledger — once recorded, transactions cannot be reversed
- Network consensus — majority of nodes must agree on transaction validity
- Confirmation depth — more confirmations = higher security

A transaction is considered fully secure after 6 confirmations on the Bitcoin network (approximately 1 hour).

2.4 FORTH — Precursor for Bitcoin Scripting

Bitcoin Script is inspired by FORTH, a stack-based programming language from the 1970s.

- FORTH uses Reverse Polish Notation (RPN)
- Stack-based execution: operands pushed, operations pop and push
- Bitcoin Script is intentionally NOT Turing-complete for security
- Scripts define conditions under which Bitcoin can be spent

2.5 Bitcoin Scripts

Bitcoin uses scripts to define spending conditions. Two key script types:

Script Type	Description	Use Case
ScriptPubKey (locking)	Conditions to spend the output	Placed in transaction output
ScriptSig (unlocking)	Proof that conditions are met	Placed in transaction input
P2PKH	Pay to Public Key Hash — most common	Standard Bitcoin transactions
P2SH	Pay to Script Hash — multi-sig	Escrow, multi-signature wallets

2.6 Bitcoin P2P Network

Bitcoin operates on a peer-to-peer (P2P) network where:

- No central server — all nodes are equal
- Full nodes store the entire blockchain (~500 GB)
- SPV (Simplified Payment Verification) nodes store only block headers
- Nodes communicate via gossip protocol — broadcasting transactions/blocks
- New nodes discover peers via DNS seeds or hardcoded bootstrap nodes

2.7 Transaction Structure in Bitcoin Network

Field	Description
Version	Transaction format version
Inputs	References to previous unspent outputs (UTXOs)
Outputs	New UTXOs created (recipient addresses + amounts)
Locktime	Earliest time/block the transaction can be included
txid	Hash of the transaction — unique identifier

UTXO (Unspent Transaction Output) Model: Bitcoin doesn't track balances. Instead it tracks unspent outputs. Your 'balance' is the sum of all UTXOs you control.

2.8 Block Mining, Propagation and Block Relay

Block Mining: Miners repeatedly hash the block header (varying the nonce) until finding a hash below the target difficulty. This is Proof of Work.

Block Propagation: Once mined, a block is broadcast across the network. Propagation must be fast to prevent forks. Compact Block Relay (BIP 152) reduces bandwidth by sending only block headers + short transaction IDs.

Orphan Blocks: When two miners find valid blocks simultaneously, a temporary fork occurs. The network eventually adopts the longer chain, making the other block orphaned.

2.9 Key Definitions — Unit II

Term	Definition
UTXO	Unspent Transaction Output — the basic unit of Bitcoin value
Mempool	Pool of unconfirmed transactions waiting to be included in a block
Block Reward	Newly minted BTC given to the miner who successfully mines a block
Halving	Event every 4 years that cuts Bitcoin block reward in half
Mining Pool	Group of miners combining computational power to mine cooperatively
Orphan Block	Valid block not included in the main chain after a fork resolves

UNIT III
BIT COIN CONSENSUS

3.1 Bitcoin Consensus

Consensus is the mechanism by which all nodes in a distributed network agree on the state of the ledger — which transactions are valid and in what order. Bitcoin uses Nakamoto Consensus:

- Based on Proof of Work (PoW)
- The chain with the most cumulative work is the valid chain
- Byzantine Fault Tolerant up to 50% honest miners
- Probabilistic finality — deeper blocks are exponentially harder to reverse

3.2 Proof of Work (PoW)

PoW requires miners to expend computational effort to add a new block. The puzzle: Find a nonce such that SHA256(SHA256(block_header)) < target.

Aspect	Detail
Algorithm	SHA-256 double hash
Difficulty	Adjusted every 2016 blocks (~2 weeks) to maintain 10 min block time
Energy	Highly energy-intensive (intentional — makes attacks expensive)
Security	Changing a historical block requires redoing all subsequent PoW
51% Attack	Attacker with >50% hashrate can rewrite recent history

3.3 Hashcash PoW

Hashcash was created by Adam Back in 1997 as an anti-spam mechanism. Bitcoin's PoW is directly inspired by Hashcash:

- Hashcash requires finding a hash with a certain number of leading zero bits
- Bitcoin extends this concept with adjustable difficulty target
- The work is easy to verify but hard to perform — asymmetric computation

3.4 Attacks on PoW

Attack	Description	Defence
51% Attack	Attacker controls >50% hashrate; can double-spend	High honest miner participation
Selfish Mining	Miner withholds valid blocks to gain advantage	Propagation improvements
Eclipse Attack	Isolate a node from honest network	Multiple diverse connections
Sybil Attack	Create many fake nodes to control network	PoW makes this expensive

3.5 Monopoly Problem

The monopoly problem arises when a single entity or mining pool gains disproportionate control (approaching 51%) of the network's hashrate. This threatens decentralization. Solutions:

- Mining pool size limits (social consensus)
- Alternative consensus mechanisms (PoS, DPoS)
- Geographic and hardware diversity in mining

3.6 Proof of Stake (PoS)

In PoS, validators are chosen to create new blocks based on the amount of cryptocurrency they 'stake' (lock as collateral) rather than computational work.

Feature	Proof of Work	Proof of Stake
Selection Method	Computational puzzle	Coin stake amount
Energy Use	Very high	Very low
Security	Hashrate-based	Economic stake-based
Examples	Bitcoin	Ethereum (post-Merge), Cardano
Attack Cost	51% hashrate	51% of staked coins

3.7 Proof of Burn

In Proof of Burn, participants 'burn' (permanently destroy) coins by sending them to an unspendable address, proving commitment to the network. Burned coins act as virtual mining rigs.

- No physical hardware needed
- Coins burned cannot be recovered
- Used in some altcoins and cross-chain bootstrapping

3.8 Proof of Elapsed Time (PoET)

PoET is a consensus mechanism by Intel using Trusted Execution Environments (TEE). Each participant:

13. Requests a random wait time from the TEE
14. Sleeps for that duration
15. The first node to wake up gets to create the next block

Used in Hyperledger Sawtooth. Energy-efficient and fair but requires trusted hardware (Intel SGX).

3.9 Bitcoin Miner and Mining Difficulty

Mining difficulty is automatically adjusted every 2016 blocks:

- If blocks are mined faster than 10 minutes → difficulty increases
- If blocks are mined slower than 10 minutes → difficulty decreases
- Target: one block every ~10 minutes on average

Mining Hardware Evolution: CPU → GPU → FPGA → ASIC (Application-Specific Integrated Circuit). Modern Bitcoin mining is dominated by ASICs which are thousands of times more efficient than CPUs.

3.10 Mining Pools

Mining pools allow multiple miners to combine hashrate and share rewards proportionally. Types:

- Pay Per Share (PPS): Fixed payout per valid share
- Proportional: Rewards split based on shares submitted per round
- PPLNS (Pay Per Last N Shares): Discourages pool hopping

The Permissioned model in mining pools means pool operators set rules for participation, work submission, and reward distribution — contrasting with the permissionless nature of Bitcoin itself.

UNIT IV
HYPER LEDGER FABRIC & ETHEREUM

4.1 Hyperledger Fabric Overview

Hyperledger Fabric is an enterprise-grade, permissioned blockchain framework hosted by the Linux Foundation. Key characteristics:

- Permissioned: participants have known identities (via MSP — Membership Service Provider)
- Modular architecture: pluggable consensus, identity, and storage
- Supports private channels for confidential transactions
- No cryptocurrency required — runs on standard IT infrastructure
- Smart contracts called Chaincode (written in Go, Node.js, or Java)

4.2 Hyperledger Fabric v1.1 Architecture

Component	Role
Peer Nodes	Maintain ledger and execute chaincode
Orderer Nodes	Order transactions and create blocks (consensus)
CA (Certificate Authority)	Issues digital certificates for identity
MSP	Defines rules for membership and permissions
Channels	Private sub-network for confidential communication
Ledger	State database (LevelDB/CouchDB) + blockchain (append-only log)
Chaincode	Smart contracts — business logic

Transaction Flow in Fabric:

16. Client submits transaction proposal to endorsing peers
17. Endorsing peers simulate and sign the proposal
18. Client collects endorsements and submits to orderer
19. Orderer creates a block and broadcasts to all peers
20. Peers validate and commit the block to the ledger

4.3 Ethereum Overview

Ethereum is a decentralized, open-source blockchain with smart contract functionality, created by Vitalik Buterin in 2015.

- Turing-complete smart contracts (unlike Bitcoin Script)
- Native currency: Ether (ETH)
- Gas mechanism to prevent infinite loops and pay for computation
- Supports DApps (Decentralized Applications)
- Transitioned from PoW to PoS (The Merge, Sept 2022)

4.4 Ethereum Network

Component	Description
Mainnet	The live Ethereum network
Testnets	Goerli, Sepolia — for testing without real ETH
Block time	~12 seconds (post-Merge, PoS)
Block size	Target 15M gas, max 30M gas
Consensus	Proof of Stake (Gasper — Casper FFG + LMD-GHOST)

4.5 EVM — Ethereum Virtual Machine

The EVM is a sandboxed runtime environment for executing smart contracts. It is:

- Stack-based (256-bit word size)
- Deterministic — same input always produces same output
- Isolated — contracts cannot access filesystem, network, etc.
- Every EVM-compatible network runs the same code identically
- Other chains like BNB Chain, Polygon also use EVM

4.6 Transaction Fee, Gas, Ether

Concept	Explanation
Ether (ETH)	Native currency of Ethereum
Wei	Smallest unit: 1 ETH = 10^18 Wei
Gwei	1 Gwei = 10^9 Wei — used to express gas prices
Gas	Unit measuring computational effort of an operation
Gas Limit	Max gas user allows for a transaction
Gas Price	Amount in Gwei user is willing to pay per gas unit
Transaction Fee	Gas Used × Gas Price (in ETH)
EIP-1559	Base fee (burned) + priority tip (to validators)

4.7 Mist Browser

Mist was Ethereum's official desktop browser/wallet for interacting with DApps (now deprecated, replaced by MetaMask and other wallets). It allowed users to:

- Manage Ethereum accounts and ETH balances
- Browse and interact with decentralized applications
- Deploy and execute smart contracts
- View transaction history

4.8 Solidity — Smart Contract Language

Solidity is the primary language for writing Ethereum smart contracts:

- Statically typed, contract-oriented language
- Syntax similar to JavaScript/C++
- Compiled to EVM bytecode
- Key features: inheritance, libraries, user-defined types

```
pragma solidity ^0.8.0; contract SimpleStorage {   uint256 public storedData;   function
set(uint256 x) public { storedData = x; }   function get() public view returns (uint256)
{ return storedData; } }
```

UNIT V
BLOCK CHAIN APPLICATIONS

5.1 Smart Contracts

A smart contract is a self-executing contract with terms directly written in code on the blockchain. Characteristics:

- Automatic execution when conditions are met — no intermediaries
- Immutable once deployed (code cannot be changed)
- Transparent — visible to all network participants
- Trustless — no need to trust the other party, only the code

Smart contracts eliminate the need for traditional legal contracts in many scenarios — escrow, insurance claims, token issuance, voting, etc.

5.2 Truffle Framework

Truffle is a development framework for Ethereum smart contracts:

- Smart contract compilation, linking, deployment, and binary management
- Automated contract testing with Mocha and Chai
- Scriptable deployment with migration system
- Network management for deploying to multiple networks
- Interactive console for directly interacting with contracts

Truffle Command	Purpose
truffle init	Initialize a new project
truffle compile	Compile Solidity contracts
truffle migrate	Deploy contracts to network
truffle test	Run test suite
truffle console	Interactive REPL for contract interaction

5.3 DApps — Decentralized Applications

DApps are applications running on a decentralized blockchain network rather than centralized servers.

Feature	Traditional App	DApp
Backend	Centralized server	Blockchain (smart contracts)
Data	Central database	Distributed ledger
Control	Single company	Community/DAO

Downtime	Single point of failure	Extremely resilient
Transparency	Code hidden	Open source on-chain

Categories of DApps: DeFi (Decentralized Finance), NFT marketplaces, DAOs, Gaming (Play-to-Earn), Social media, Identity management.

5.4 NFTs — Non-Fungible Tokens

NFT (Non-Fungible Token) is a unique digital asset verified using blockchain:

- Non-fungible: each token is unique and not interchangeable
- Represents ownership of digital or physical assets
- Standard: ERC-721 (unique tokens), ERC-1155 (semi-fungible)
- Use cases: Digital art, music, gaming items, event tickets, real estate
- Creator receives royalties on secondary sales automatically via smart contracts

5.5 Blockchain in Supply Chain Management

Blockchain transforms supply chain by providing:

- End-to-end traceability of goods from origin to consumer
- Immutable audit trail — prevents fraud and counterfeiting
- Real-time visibility across all supply chain participants
- Automated payments via smart contracts when delivery confirmed
- Example: Walmart uses blockchain to track food safety — tracing produce origin in seconds vs. days

5.6 Blockchain in Logistics

Applications in logistics:

- Bill of Lading digitization — reduces paperwork and delays
- Customs clearance automation via smart contracts
- Carrier performance tracking and reputation systems
- Cargo condition monitoring with IoT + blockchain
- Example: Maersk-IBM TradeLens platform for global shipping

5.7 Blockchain for Smart Cities

Blockchain can power smart city infrastructure:

- Digital identity for citizens — secure, self-sovereign identity
- Transparent voting systems — tamper-proof e-voting
- Energy trading — peer-to-peer renewable energy trading
- Land registry — immutable property records
- Waste management — incentivized recycling using tokens

5.8 Blockchain in Finance and Banking

Financial applications:

- Cross-border payments — faster, cheaper than SWIFT

- Trade finance — automated letter of credit via smart contracts
- KYC (Know Your Customer) — shared identity verification
- Securities settlement — near-instant vs. T+2 days traditional
- CBDC (Central Bank Digital Currency) — government-issued digital money
- Example: Ripple (XRP) for international bank transfers

5.9 Blockchain in Insurance

Insurance use cases:

- Parametric insurance — automatic payout when conditions met (flight delays, crop loss)
- Claims processing automation — reduces fraud and processing time
- Reinsurance — transparent risk-sharing between insurers
- Health data sharing — secure, consent-based patient data

5.10 Case Study: Ethereum-based Supply Chain DApp

A pharmaceutical supply chain DApp on Ethereum:

21. Manufacturer deploys smart contract defining drug batch
22. Each transfer (manufacturer → distributor → pharmacy) triggers contract function
23. NFT represents unique drug batch — linked to quality certificates on IPFS
24. Patient/pharmacist can scan QR to verify authenticity on blockchain
25. Regulators have real-time access to audit trail

Benefits: Counterfeit drug detection, automatic recall management, regulatory compliance, reduced paperwork.

5.11 Summary — All Units Quick Reference

Unit	Core Topics	Key Technologies
I - Blockchain Intro	Public ledgers, block structure, Merkle tree, cryptographic hashing	SHA-256, Hash Pointers
II - Bitcoin & Crypto	Mining, UTXO, transactions, P2P network, scripts	Bitcoin Script, FORTH
III - Consensus	PoW, PoS, PoB, PoET, mining difficulty, 51% attack	Hashcash, Mining Pools
IV - Fabric & Ethereum	Hyperledger architecture, EVM, Solidity, Gas, Ether	Chaincode, Mist, EVM
V - Applications	Smart contracts, DApps, NFT, supply chain, banking	Truffle, ERC-721, IPFS