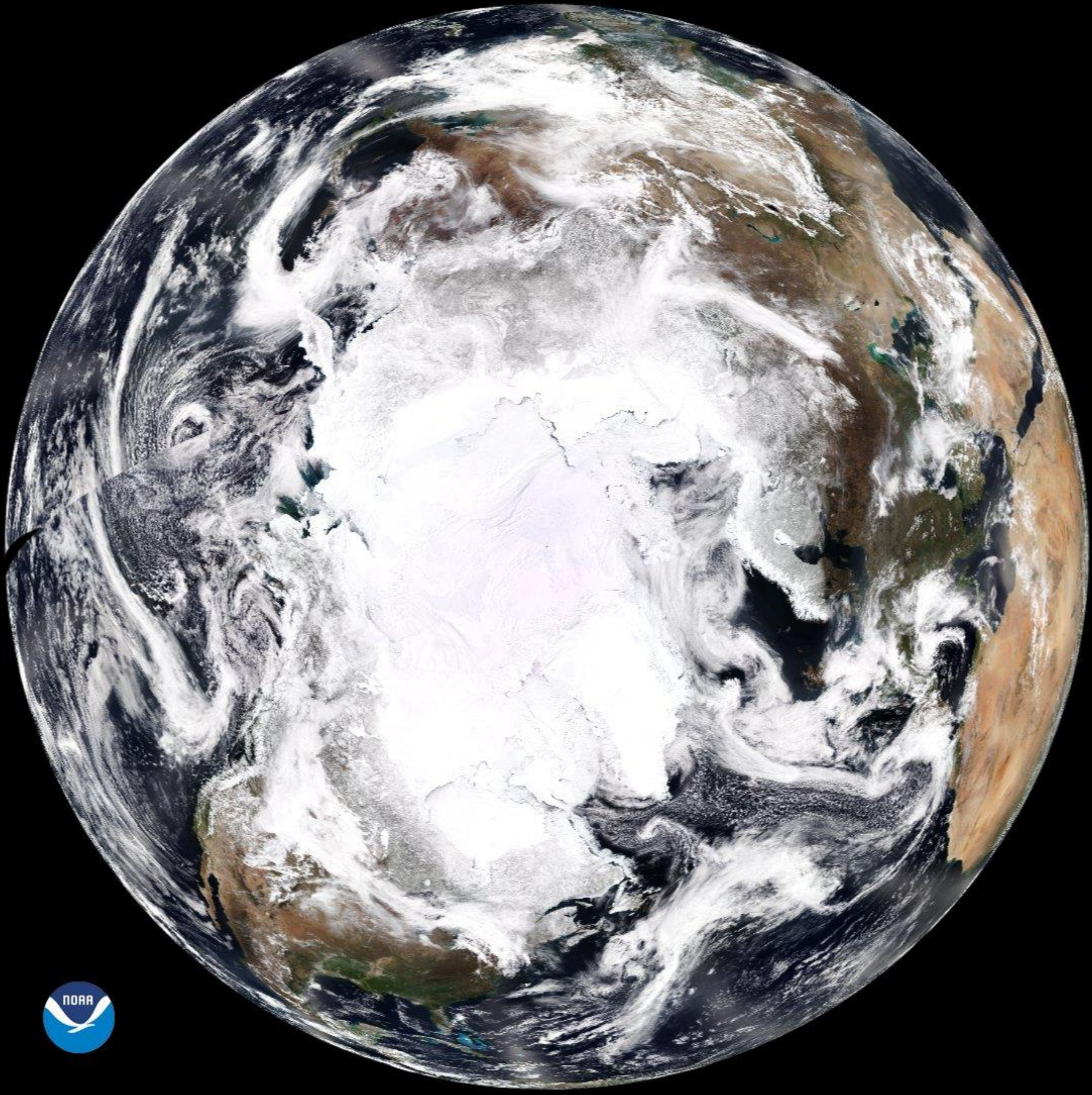
Introduction to Setting Up Ethereum on a Small Raspberry Pi Network

William Favre Slater, III, M.S. MBA, PMP, CISSP, CISA
Sr. Cybersecurity Consultant / Project Manager / Program Manager
Haymarket Pub and Brewery
Chicago, Illinois
United States of America
April 21, 2018

CHICAGO BLOCKCHAIN PROJECT

Slater Technologies



Presents

Introduction to Setting Up Ethereum on a Small Raspberry Pi Network

April 21, 2018
Chicago, Illinois
Haymarket Pub & Brewery

Agenda

- Introduction
- Some Terms
- What is Blockchain?
- Types of Blockchains
- What is Raspberry Pi
  - Raspberry Pi Architecture and Components
- Displaying Raspberry Pi on your Laptop
- Raspberry Pi 101 - Starting up Raspberry Pi
- Ethereum 101
- Installing Ethereum on Raspberry Pi
- Operating Ethereum on Raspberry Pi
  Proving Ethereum works on Raspberry Pi
- Conclusion
- Questions
- References

Slater Technologies

Introduction

- This presentation is about how to build a private blockchain using four Raspberry Pi. This will be followed by a hands-on demonstration of using four Raspberry Pi computers with Ethereum to build a small, but working Blockchain network.
- We are excited that you are here!

Slater Technologies

SOME TERMS

Some Important Terms

Term	Explanation
AES SHA-256	The 256-bit encryption algorithm that is AES standard used for Bitcoin keys.
Bitcoin Network	The Internet-connected network comprised of the software and data that supports Bitcoin transactioms
Blockchain	The Bitcoin ledger of past transactions.
Difficulty	The measure of how difficult it is to find a new block compared to the easiest it can ever be
Exchange	A place that sells can buys Bitcoins, like a stock exchange.
Hash	It is a standard cryptographic algorithm function for the generation and verification of currency
Mining	Bitcoin mining serves 2 purposes, it creates the general ledger of Bitcoin transactions and it provides security.
Private Key	The secret cryptographic key that is used to protect your Bitcoin account
Proof of Work	An economic time-stamped measure to deter service abuses on a network by requiring some work from the service requester, usually meaning processing time by a computer.
Public Key	The public (shared) cryptographic key that is used to protect your Bitcoin account
Transaction	Use of the Bitcoin to purchase good or services, or the purchase of sale of a Bitcoin, or fractional part of Bitcoin
Wallet	A service that will safely store your Bitcoin account for you.

Slater Technologies

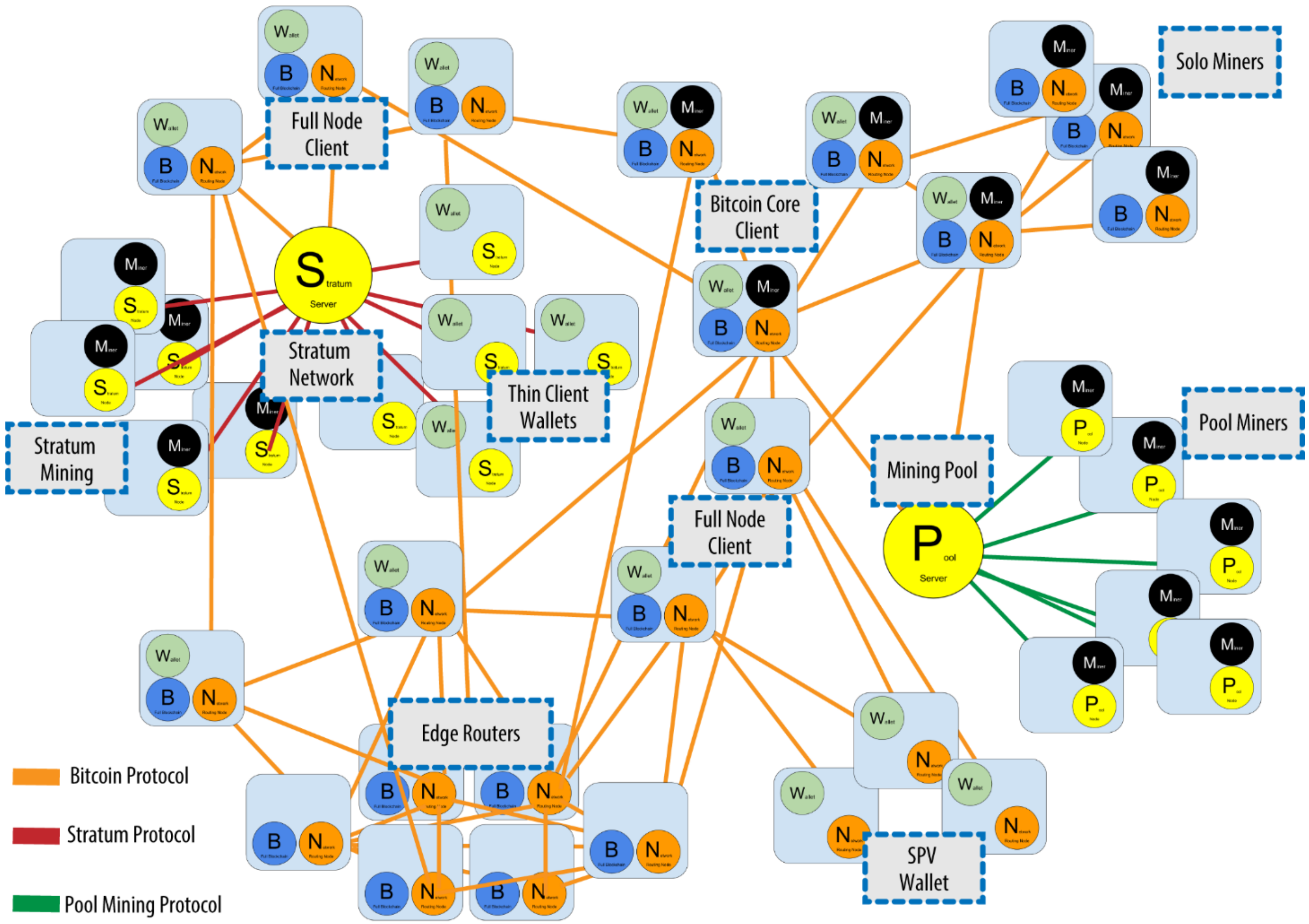
- *Candidate block*: An incomplete block, created as a temporary construct by a miner to store transactions from the transaction pool. It becomes a complete block after the header is completed by solving the PoW problem.
- *PoW* : The problem of discovering a new hash that can be used in the block header of the candidate block. This is a computationally intensive process that involves evaluating a hash taken from the most recent block and appending a nonce to it against the target value of the network. This problem can only be solved using brute force; that is, multiple trials of using the hash (from the most recent block header) and nonce being adjusted each time are necessary to solve the PoW problem.
- *Nonce*: A 32-bit value that is concatenated to the hash from the most recent block header. This value is continuously updated and adjusted for each trial, until a new hash below the target value is discovered.
- *Hash function*: A function used to compute a hash. In the Bitcoin protocol, this function is the SHA-256.
- *Hash value*: The resulting hash output from a hash function.
- *Target value*: A 265-bit number that all Bitcoin clients share. It is determined by the difficulty, which is discussed shortly.
- *Coinbase transaction*: The first transaction that is packaged into a block. This is a reward for the miner to mine the PoW solution for the candidate block.
- *Block header*: The header of a block, which contains many features such as a timestamp, PoW, and more. We describe the block header in more detail in Chapter 3.

Source: **Blockchain Basics: A Non-technical Introduction in 25 Steps by Daniel Drescher**

WHAT IS BLOCKCHAIN?

Slater Technologies

A Logical Diagram of a Blockchain Network



This Photo by Unknown Author is licensed under CC BY-SA

Slater Technologies

What Is Blockchain?

- Distributed Ledger
- Decentralized
- Popularized by Satoshi Nakamoto
- Uses Cryptography and Hashing
- Append-only Transactions
- The Code already exists in Github
- Immutable
- First discussed in 1991
- Ethereum announced in 2015

Slater Technologies

What Is Blockchain?

- Blockchain Consensus Protocol guide. A blockchain is a decentralized peer-to-peer system with no central authority figure. While this creates a system that is devoid of corruption from a single source, it still creates a major problem.
  - How are any decisions made?
  - How does anything get done?
  - Think of a normal centralized organization.
- All the decisions are taken by the leader or a board of decision makers. This isn't possible in a blockchain because a blockchain has no "leader". For the blockchain to make decisions, they need to come to a consensus using "consensus mechanisms".

What is Blockchain?

- A Decentralized, Distributed Ledger
- Updated using software, messaging and databases with Append-only transactions
- Records are immutable.
- There are multiple copies
- Updated by miners, and synchronized using Proof of Work, and Consensus
- The foundation technology for Cryptocurrency
- The Future of Trusted Computing Transactions on the Internet and in public and private networks
- First described by Satoshi Nakamoto in his 9-page January 2009 paper: https://bitcoin.org/bitcoin.pdf
- The world's largest Blockchain Database is the Bitcoin Blockchain Database, with 160 GB (it doesn't scale very well)

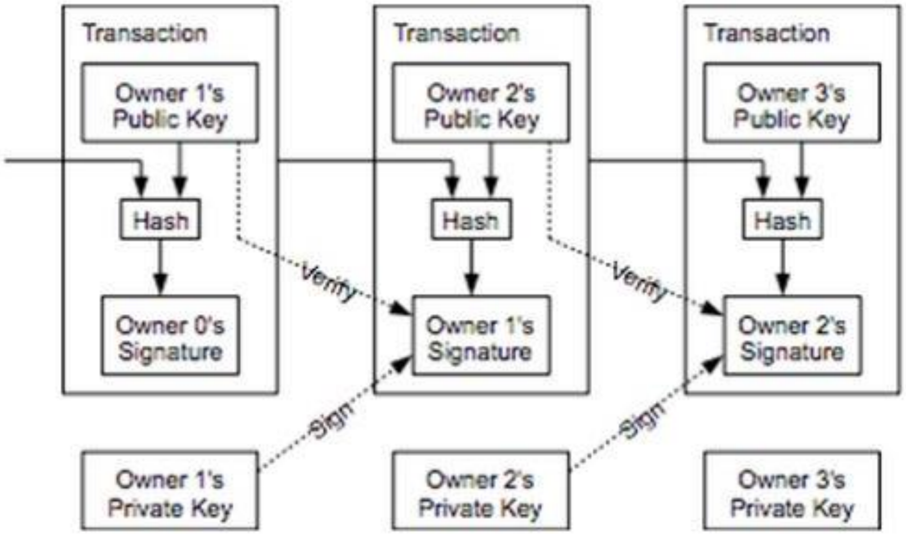


Image: Satoshi Nakamoto

Slater Technologies

The Term Blockchain

- Name for a data structure
- Name for an algorithm
- Name for a suite of Technologies
- An umbrella term for purely distributed peer-to-peer systems with a common application area
- A peer-to-peer-based operating system with its own unique rule set that utilizes hashing to provide unique data transactions with a distributed ledger

Slater Technologies

Blockchain – Simplified View

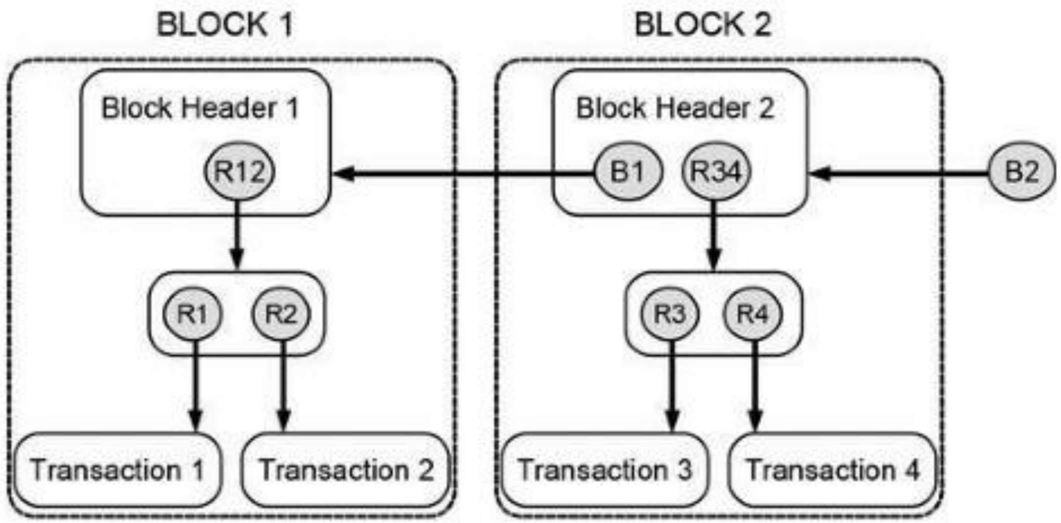


Figure 14-5. A simplified blockchain-data-structure containing four transactions

Source: Drescher, D. (2017). Blockchain Basics. Frankfort am Main, Germany: Apress.

Slater Technologies

Characteristics of the Blockchain

The blockchain is a purely distributed peer-to-peer data store with the following properties:

- Immutable
- Append-only
- Ordered
- Time-stamped
- Open and transparent
- Secure (identification, authentication, and authorization)
- Eventually consistent

Source: Drescher, D. (2017). Blockchain Basics. Frankfort am Main, Germany: Apress.

Slater Technologies

Properties of the Blockchain Non-functional Aspects

When interacting with the blockchain, you will notice how it fulfills its duties. The quality at which the blockchain serves its purpose is described by its nonfunctional aspects:

- Highly available
- Censorship proof
- Reliable
- Open
- Pseudoanonymous
- Secure
- Resilient
- Eventually consistent

Slater Technologies

Why Is Blockchain Important?

- Accessible
- Open source
- Easily provides three challenging elements of the **Parkerian Hexad** model for security:
  - **Authenticity**
  - **Control**
  - **Utility**
- It WORKS!
- Business enabler
- Reduces risk of computer fraud
- It is being widely adopted for trusted computing
- Blockchain developers and architects are in great demand: for every Blockchain professional there are 14 open positions

Donn B. Parker

Slater Technologies

Blockchain Transactions: Satoshi Nakamoto's Vision

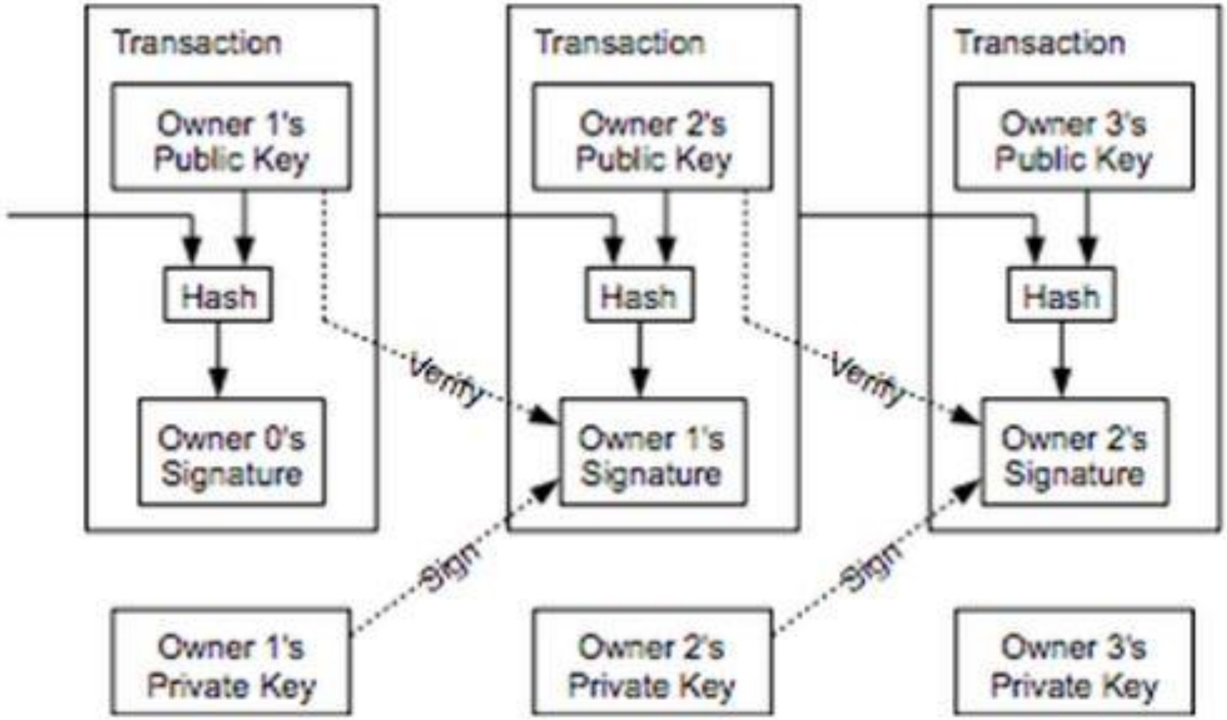


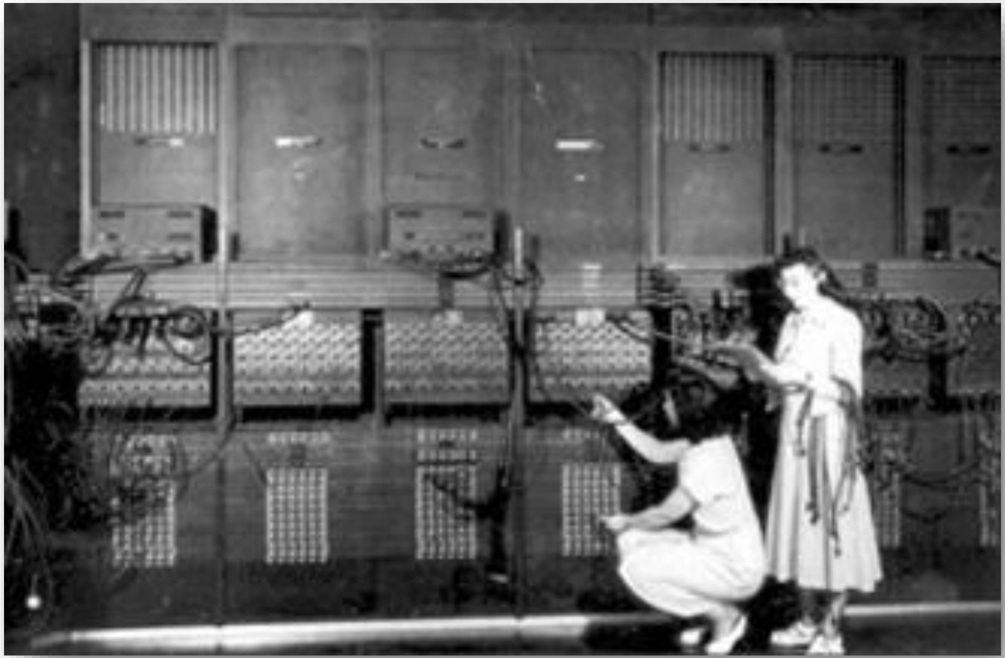
Image: Satoshi Nakamoto

Source: **Bitcoin: A Peer-to-Peer Electronic Cash System**. By Satoshi Nakamoto. Retrieved from https://bitcoin.org/bitcoin.pdf

Slater Technologies

Technologies and Events that Led to the Creation of Bitcoin and Blockchain

- Cryptography
- Transistors
- Digital Computers
- Databases
- Silicon Chips
- Programming
- Applied Cryptography
- Computer Networks
- Transaction Processing
- TCP/ IP and The Internet
- The World Wide Web
- Evolution of Security and Privacy Thought
- The Great 2008 Economic Recession



Slater Technologies

Blockchain Technologies

Technology	Source
The Internet (TCP/IP)	Built into every modern OS
Cryptography	Cryptography software
Bitcoin software	Github
Ethereum	Github
Blockchain Database	MongoDB or BigchainDB

Slater Technologies

TYPES OF BLOCKCHAINS

Slater Technologies

Types of Blockchains

- Bitcoin vs. Ethereum vs, Hyperledger (Linux and IBM)
- Public vs. Private
- Permissioned (private) vs. Permissionless

Bitcoin vs. Ethereum

VS	Bitcoin	Ethereum
Founder	Satoshi Nakamoto	Vitalik Buterin
Release Date	9 Jan 2008	30 July 2015
Release Method	Genesis Block Mined	Presale
Blockchain	Proof of work	Proof of work (Planning for POS)
Useage	Digital Currency	Smart Contracts Digital Currency
Cryptocurrency Used	Bitcoin(Satoshi)	Ether
Algorithm	SHA-256	Ethash
Blocks Time	10 Mintues	12-14 Seconds
Mining	ASIC miners	GPUs
Scalable	Not now	Yes

Slater Technologies

Bitcoin vs. Ethereum vs. Hyperledger

Blockchain characteristics comparison			
Characteristics	**Bitcoin**	**Ethereum**	**Hyperledger**
Permission restrictions	Permissionless	Permissionless	Permissioned
Restricted public access to data	Public	Public or private	Private
Consensus	Proof-of-Work	Proof-of-Work	PBFT
Scalability	High node-scalability, Low performance-scalability	High node-scalability, Low performance-scalability	Low node-scalability, High performance-scalability
Centralized regulation (governance*)	Low, decentralized decision making by community/miners	Medium, core developer group, but EIP process	Low, open-governance model based on Linux model
Anonymity	Pseudonymity, no encryption of transaction data	Pseudonymity, no encryption of transaction data	Pseudonymity, encryption of transaction data
Native currency	Yes, bitcoin, high value	Yes, ether	No
Scripting	Limited possibility, stack-based scripting	High possibility, Turing-complete virtual machine, high-level language support (Solidity)	High possibility, Turing-complete scripting of chaincode, high-level Go-language

Comparison of Ethereum, Hyperledger Fabric and Corda

Characteristic	Ethereum	Hyperledger Fabric	R3 Corda
Description of platform	– Generic blockchain platform	– Modular blockchain platform	– Specialized distributed ledger platform for financial industry
Governance	– Ethereum developers	– Linux Foundation	– R3
Mode of operation	– Permissionless, public or private[4]	– Permissioned, private	– Permissioned, private
Consensus	– Mining based on proof-of-work (PoW) – Ledger level	– Broad understanding of consensus that allows multiple approaches – Transaction level	– Specific understanding of consensus (i.e., notary nodes) – Transaction level
Smart contracts	– Smart contract code (e.g., Solidity)	– Smart contract code (e.g., Go, Java)	– Smart contract code (e.g., Kotlin, Java) – Smart legal contract (legal prose)
Currency	– Ether – Tokens via smart contract	– None – Currency and tokens via chaincode	– None

Ethereum Public Blockchain

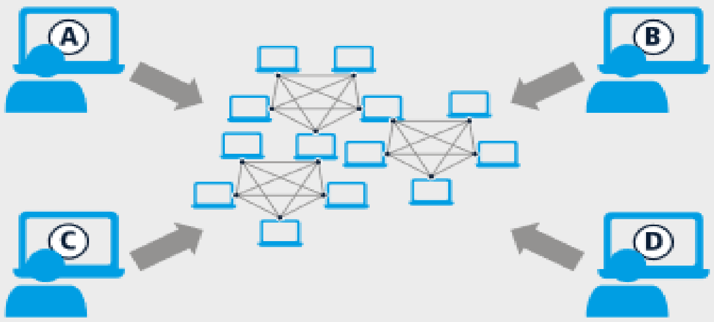
- Ethereum was developed initially for public chain deployment, where trustless transaction requirements outweigh absolute performance. The current public chain consensus algorithms (notably PoW) are overkill for networks with trusted actors and high throughput requirements.
- Public chains by definition have limited (at least initially) privacy and permissioning requirements. Although Ethereum does enable permissioning to be implemented within the smart contract and network layers, it is not readily compatible out of the box with traditional enterprise security and identity architectures or data privacy requirements.
- Naturally, the current Ethereum improvement process (dominated by Ethereum improvement proposals) is largely dominated by public chain matters, and it has been previously challenging for enterprise IT requirements to be clarified and prioritized within it.

Source: **Blockchain Basics: A Non-technical Introduction in 25 Steps**
by Daniel Drescher

Slater Technologies

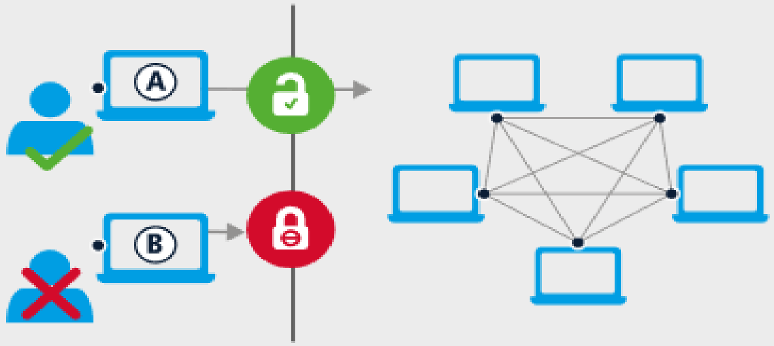
Public vs. Private

PUBLIC VS. PRIVATE BLOCKCHAINS



PUBLIC, PERMISSIONLESS BLOCKCHAINS

- Anyone can join the network and submit transactions
- Anyone can contribute computing power to the network and broadcast network data
- All transactions are broadcast publicly



PRIVATE, PERMISSIONED BLOCKCHAINS

- Only safelisted (checked) participants can join the network
- Only safelisted (checked) participants can contribute computing power to the network and broadcast network data
- Access privileges determine the extent to which each safelisted participant can contribute data to the network and access data from the network

Key differences between public, permissionless blockchains and private, permissioned blockchains; **Source:** Accenture

Slater Technologies

Four Functional Versions of Blockchain Distributed Ledgers

Table 23-2 presents the four versions of the blockchain that arise when combining the extreme cases of reading and writing restrictions.

Table 23-2. Four Versions of the Blockchain as a Result of Combining Reading and Writing Restrictions

	Reading Access and Creation of Transactions	
Writing Access	**Everyone**	**Restricted**
Everyone	Public & Permissionless	Private & Permissionless
Restricted	Public & Permissioned	Private & Permissioned

Source: **Blockchain Basics: A Non-technical Introduction in 25 Steps** by Daniel Drescher

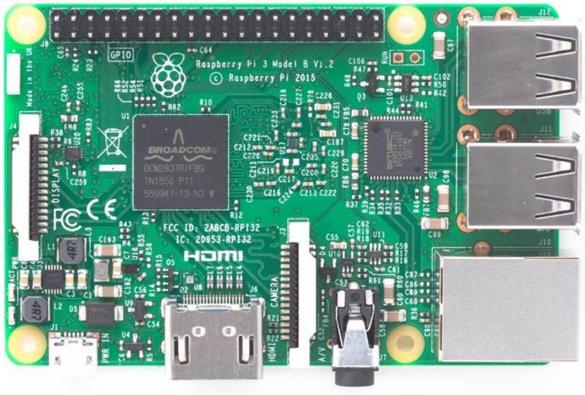
Slater Technologies

WHAT IS RASPBERRY PI?

Slater Technologies

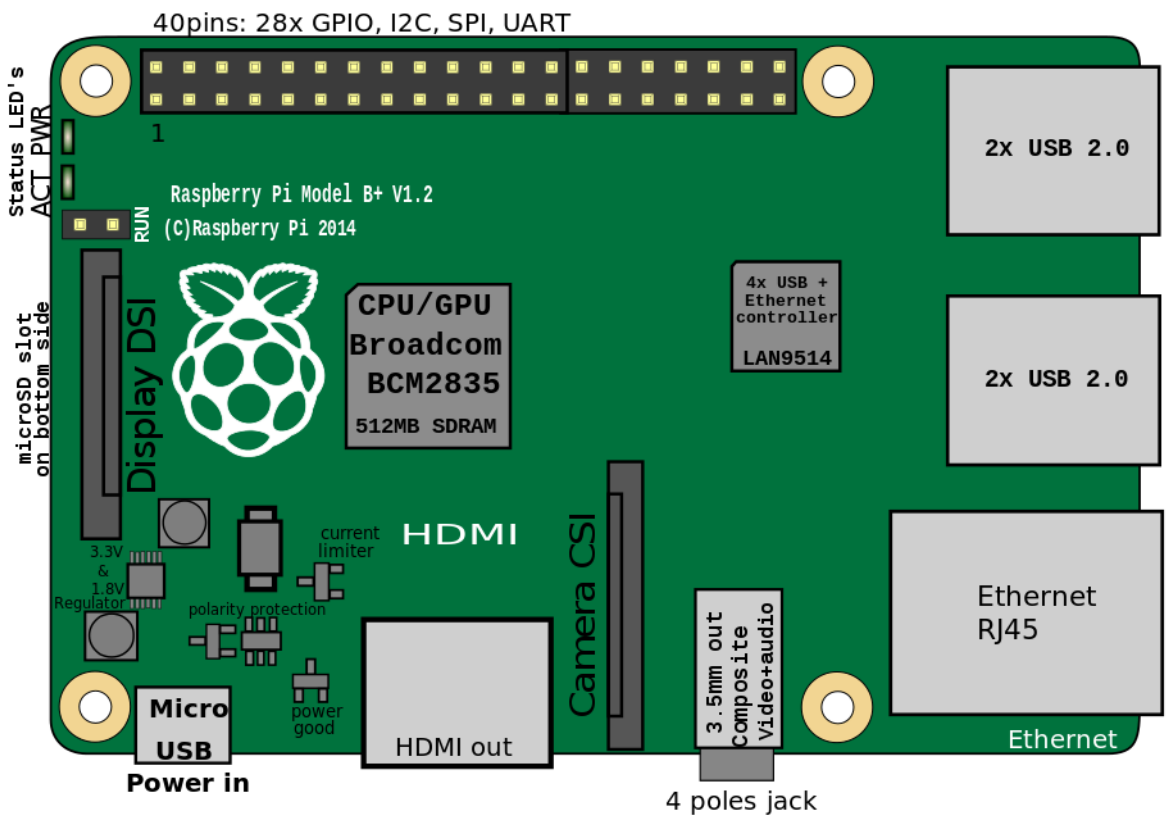
What is Raspberry Pi?

- A powerful, inexpensive ARM computer that runs Raspian Linux
- Invented in 2010 by Eben Upton, Rob Mullins, Jack Lang, and Alan Mycroft at the University of Cambridge in the U.K.
- An Answer to the plight of technical computer illiteracy
- As of 2018, over 7 million units shipped



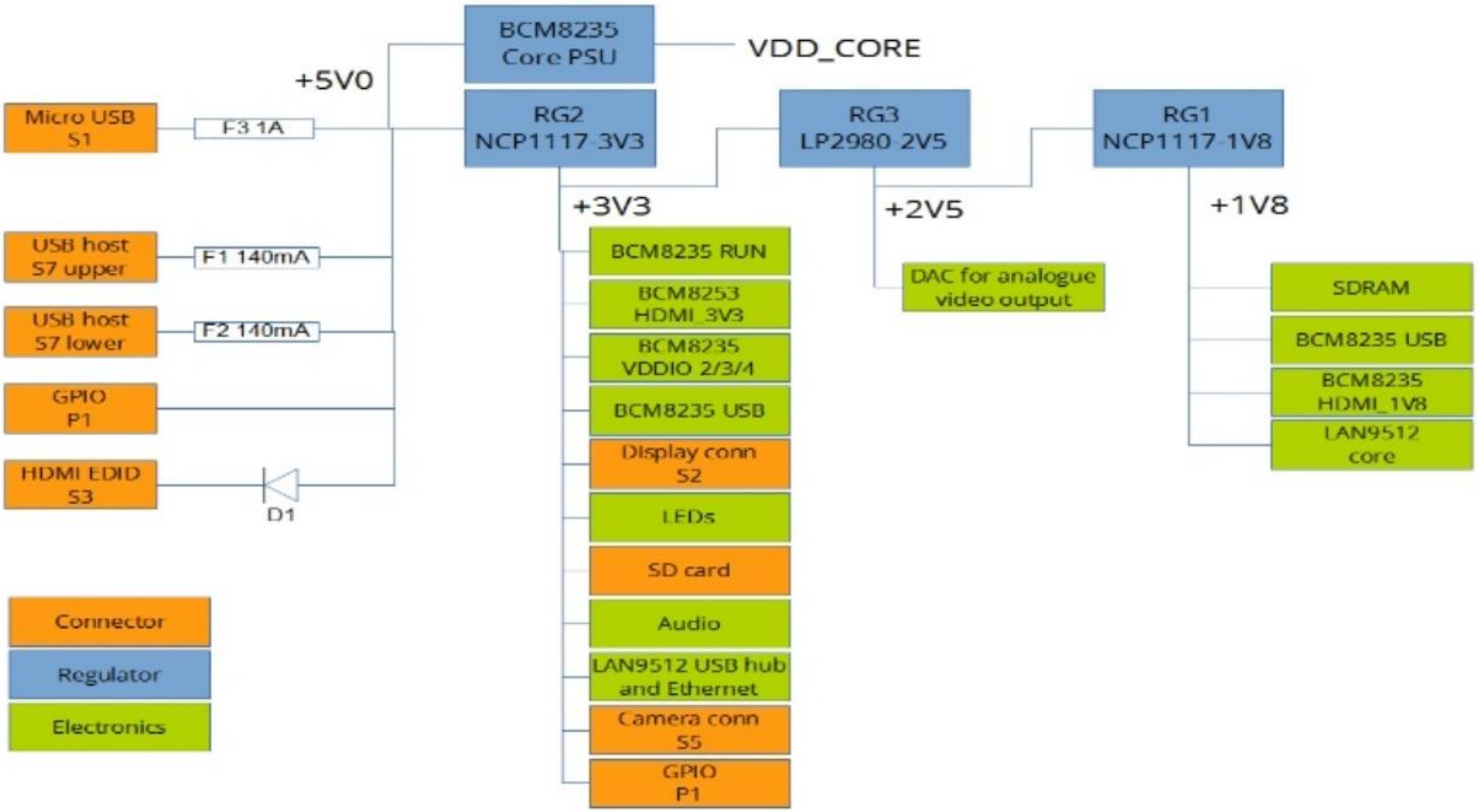
Slater Technologies

Raspberry Pi Architecture



Slater Technologies

Raspberry Pi Architecture

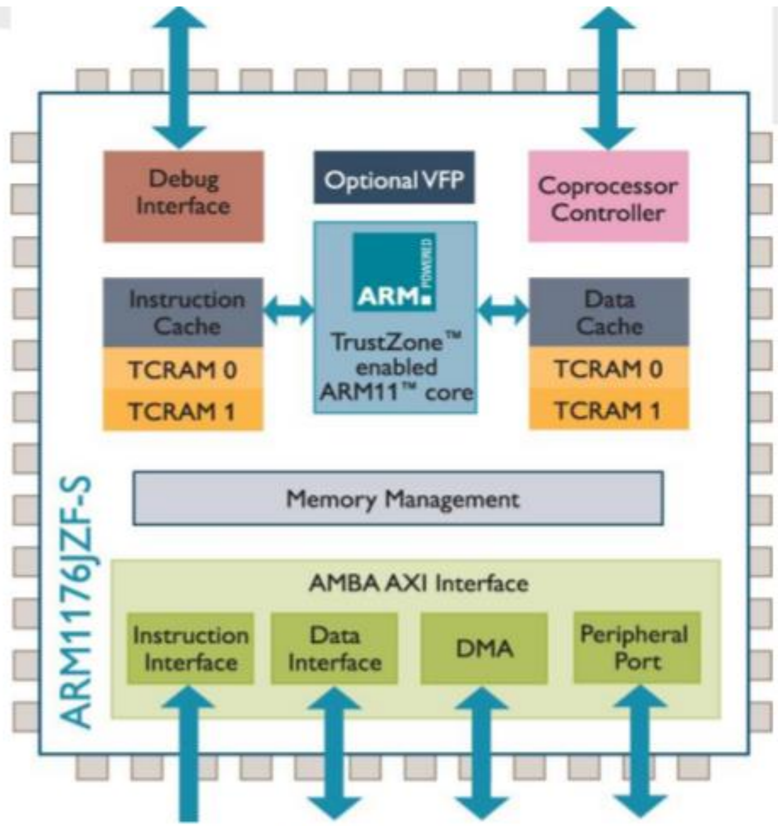


Fazal, R. (2014). Raspberry Pi.

Slater Technologies

Broadcom BCM 2835 Architecture

- ARM11J6JZF-S (ARM11 Family)
- ARMv6 Architecture
- Single Core
- 32-Bit RISC
- 700 MHz Clock Rate
- 8 Pipeline Stages
- Branch Prediction



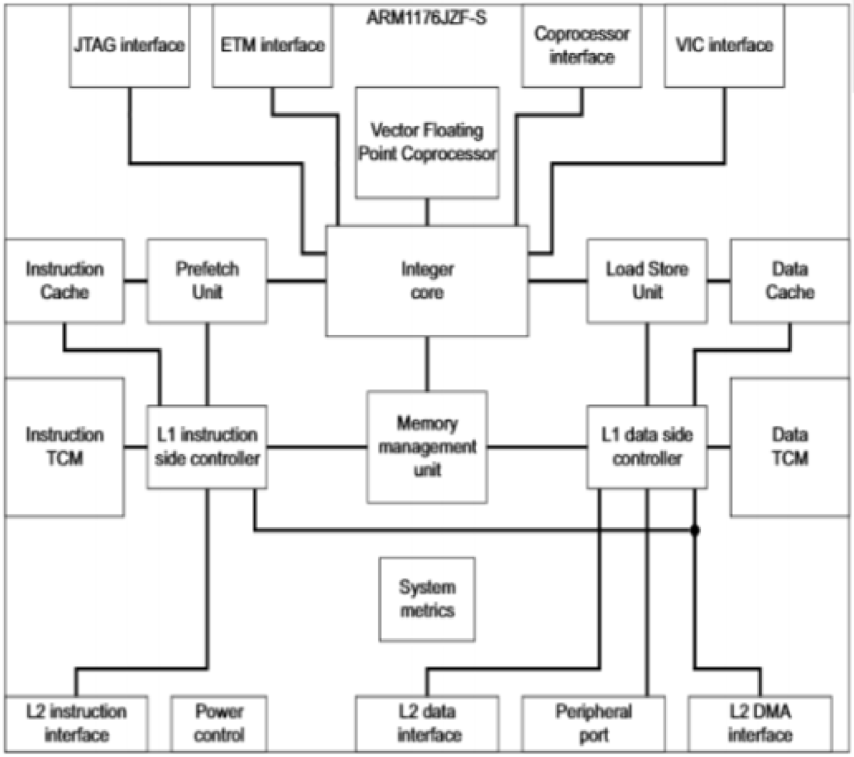
ARM 1176 Processor

Holton, J. and Fratangelo, T. (2016). Raspberry Pi Architecture.

Slater Technologies

Broadcom BCM 2835 Overview with Block Diagram

- Core
- Load Store Unit
- Prefetch Unit
- Memory System
- Level One Mem. System
- Interrupt Handling
- System Control
- AMBA Interface
- Coprocessor Interface
- Debug
- Instruction cycle summary and interlocks
- Vector Floating-Point

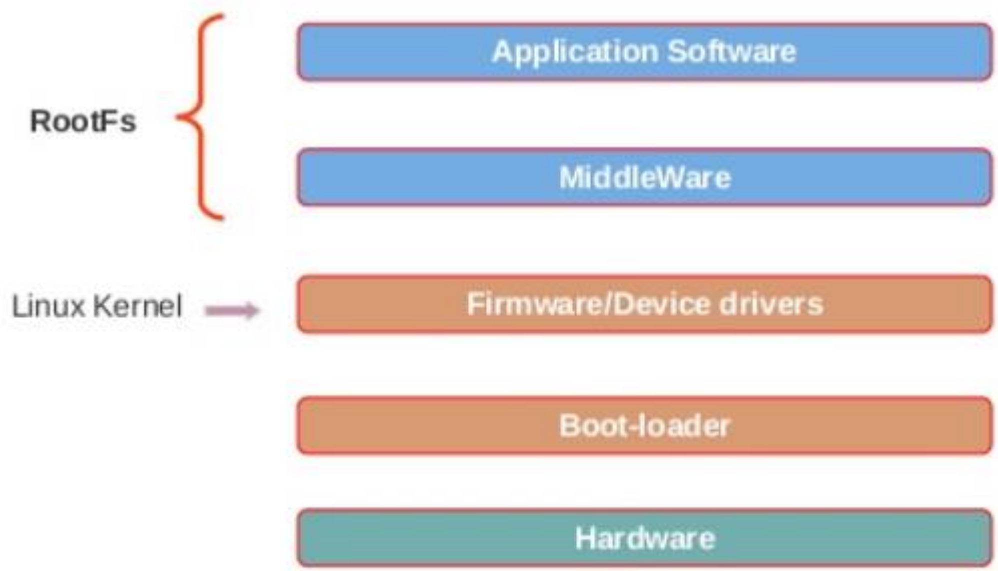


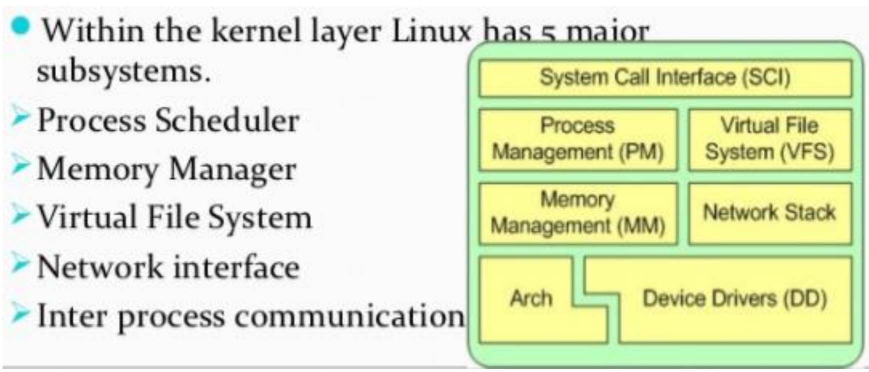
ARM1176JZF-S Technical Reference Manual

Holton, J. and Fratangelo, T. (2016). Raspberry Pi Architecture.

Slater Technologies

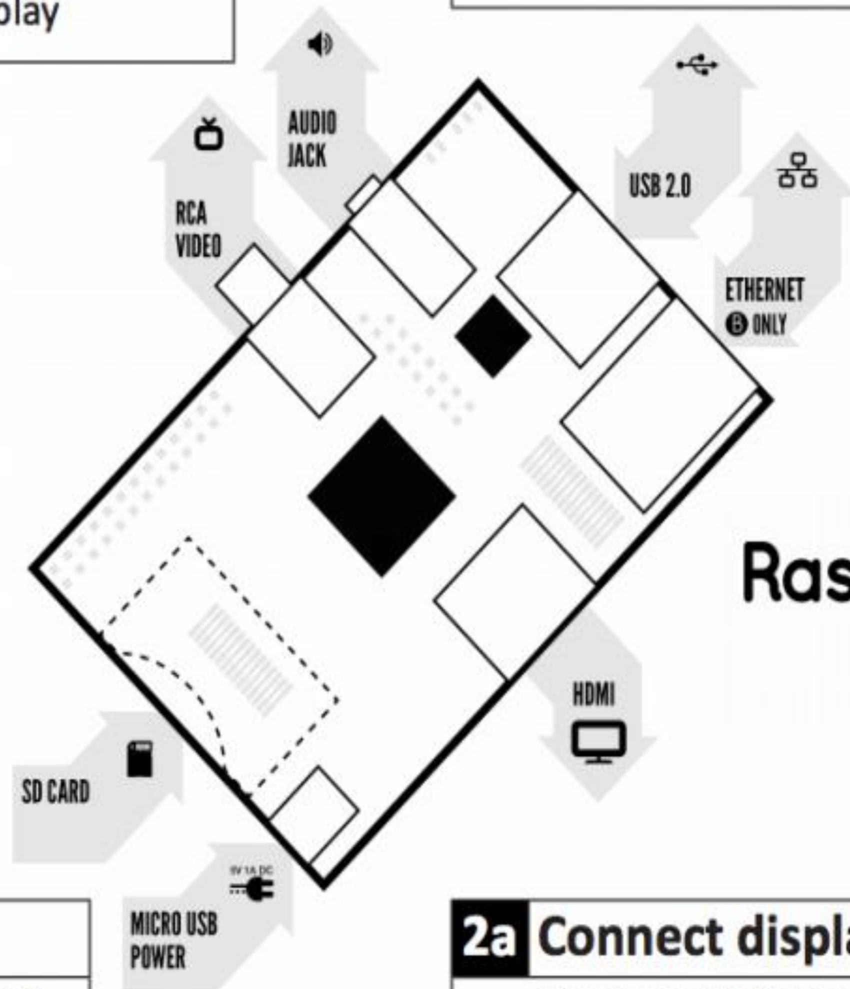
Raspian Linux Architecture





Slater Technologies

Raspberry Pi
Quick start



1 Insert SD card

See page 3 for how to prepare the SD card

2a Connect display

Plug in your digital TV or monitor

2b Connect display

If *not* using HDMI, plug in your analogue TV or display

3 Connect input

Plug in a USB keyboard and mouse

4 Connect network

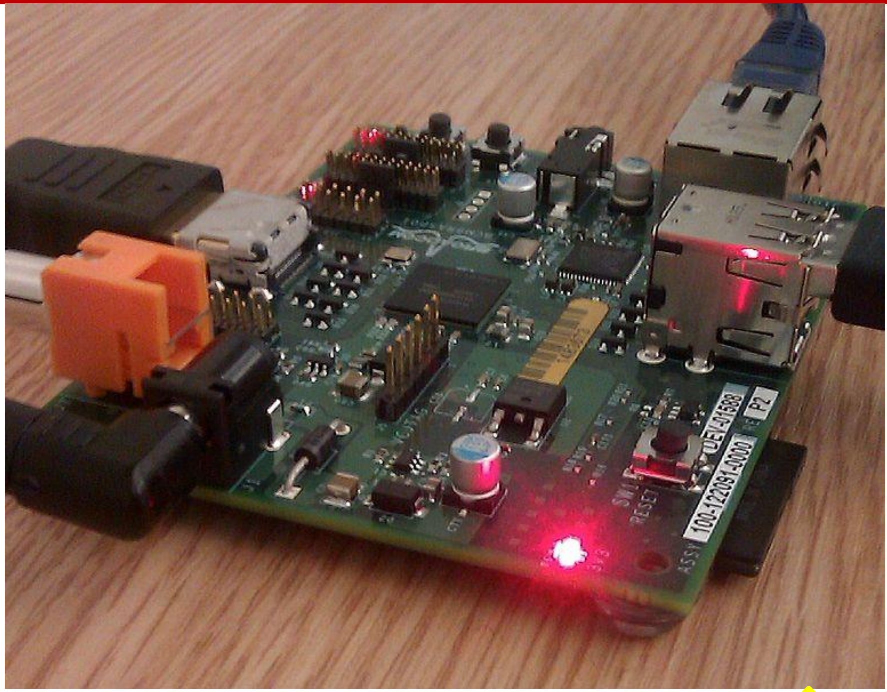
Connect to your wired network [optional]

5 Power up

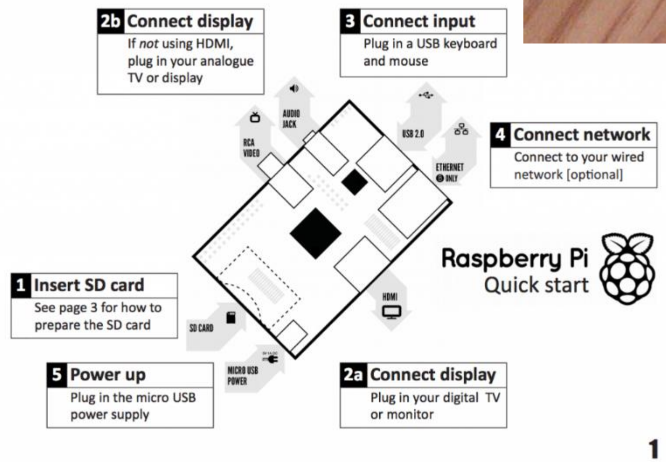
Plug in the micro USB power supply

Raspberry Pi

Buy a Raspberry Pi CanaKit



Connect



Power up and Go!

ILLINOIS INSTITUTE OF TECHNOLOGY

When You Install and Start Up Raspbian Linux

2.6 Raspbian's Desktop Environment

If you have not changed the setting in raspi-config, the Raspberry Pi will boot into Raspbian's command line.

To start the desktop environment:

1. Type *pi* as the username, then press **Enter**.
2. Type your password, then press **Enter**.[1]
3. Type the following command and press **Enter**:

```
startx
```

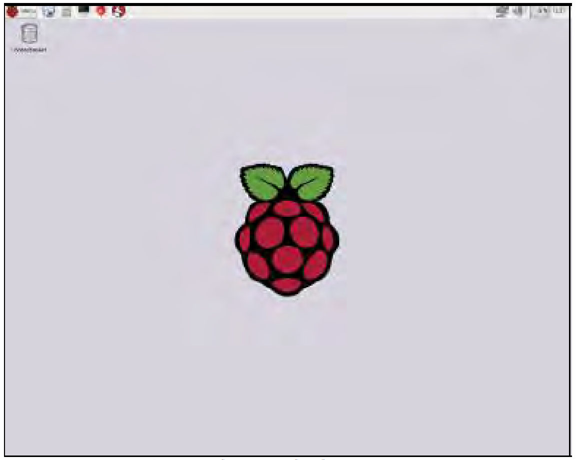


Figure 2. Raspbian's desktop environment

ILLINOIS INSTITUTE OF TECHNOLOGY

DISPLAYING RASPBERRY PI ON YOUR LAPTOP

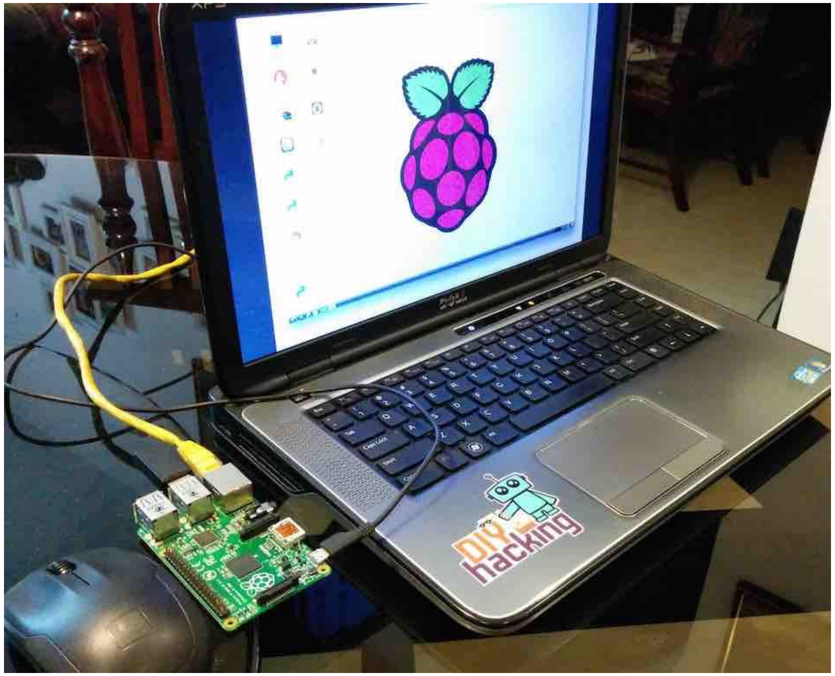
Slater Technologies

Using Your Windows Laptop Display with Raspberry Pi

Follow the excellent tutorial and it will show you how to connect your Raspberry Pi to your Windows Laptop.

https://maker.pro/raspberry-pi/tutorial/how-to-connect-a-raspberry-pi-to-a-laptop-display

Note to Mac users (and I know there are multitudes out there) I don't do Mac, only Windows and Linux – Sorry.



Patel, S. (2015). Connecting Your Raspberry Pi to a Laptop Display

Slater Technologies

RASPBERRY PI 101 - STARTING UP RASPBERRY PI

Slater Technologies

Raspberry Pi 101

- Buy a Raspberry Pi Kit – Preferably the Cana Kit
- Go to this link and download the Getting Started with Raspberry Pi PDF file: https://goo.gl/bTb4mr
- It will easily step you through the process of starting up your new Raspberry Pi.

Slater Technologies

ETHEREUM 101

Slater Technologies

Ethereum 101

- Invented and released in 2015 by Vitalik Buterin, Ethereum is based on the Bitcoin Blockchain, expect that it allows for programmable Smart Contracts
- Facilitates the trade of the Ether Cryptocurrency
- It uses the Ethereum Virtual machine as an execution environment
- Many feel the features of Ethereum environment will offer Blockchain application designers and developers to more fully realize the potential for Blockchain to change the world.
- Available for FREE on Github: https://github.com/ethereum
- Maintained by the Ethereum Foundation (www.ethereum.org)

Slater Technologies

Ethereum 101

```
contract mortal {
    /* Define variable owner of the type address */
    address owner;

    /* This function is executed at initialization and sets the owner
    function mortal() { owner = msg.sender; }

    /* Function to recover the funds on the contract */
    function kill() { if (msg.sender == owner) selfdestruct(owner); }
}

contract greeter is mortal {
    /* Define variable greeting of the type string */
    string greeting;

    /* This runs when the contract is executed */
    function greeter(string _greeting) public {
        greeting = _greeting;
    }

    /* Main function */
    function greet() constant returns (string) {
        return greeting;
    }
}
```

An example Ethereum smart contract. Source: ethereum.org.

Slater Technologies

INSTALLING ETHEREUM ON RASPBERRY PI

Installing Geth

Cloning into go-ethereum:

$ mkdir src

$ cd src

$ git clone -b release/1.7 https://github.com/ethereum/go-ethereum.git

```
pi@raspberrypi:~ $ mkdir src
pi@raspberrypi:~ $ cd src
pi@raspberrypi:~/src $ git clone -b release/1.7 https://github.com/ethereum/go-ethereum.git
Cloning into 'go-ethereum'...
remote: Counting objects: 69714, done.
remote: Total 69714 (delta 0), reused 0 (delta 0), pack-reused 69713
Receiving objects: 100% (69714/69714), 95.44 MiB | 248.00 KiB/s, done.
Resolving deltas: 100% (46419/46419), done.
```

Slater Technologies

Installing Geth

Installing geth

```
pi@chainpi: ~/src/go-ethereum
pi@chainpi: ~/src/go-ethereum        pi@chainpi: ~
pi@chainpi:~/src/go-ethereum $ make
build/env.sh go run build/ci.go install ./cmd/geth
>>> /usr/lib/go-1.7/bin/go install -ldflags -X main.gitCommit=4bb3c89d44e372e6a9ab85a8be0c9345265c
763a -v ./cmd/geth
github.com/ethereum/go-ethereum/common/hexutil
github.com/ethereum/go-ethereum/crypto/sha3
github.com/ethereum/go-ethereum/common/math
github.com/ethereum/go-ethereum/rlp
github.com/ethereum/go-ethereum/crypto/secp256k1
github.com/ethereum/go-ethereum/vendor/github.com/go-stack/stack
github.com/ethereum/go-ethereum/common
github.com/ethereum/go-ethereum/log
github.com/ethereum/go-ethereum/vendor/github.com/rcrowley/go-metrics
github.com/ethereum/go-ethereum/params
github.com/ethereum/go-ethereum/vendor/gopkg.in/karalabe/cookiejar.v2/collections/prque
github.com/ethereum/go-ethereum/vendor/github.com/aristanetworks/goarista/monotime
github.com/ethereum/go-ethereum/crypto/randentropy
github.com/ethereum/go-ethereum/vendor/github.com/pborman/uuid
github.com/ethereum/go-ethereum/common/mclock
github.com/ethereum/go-ethereum/event
github.com/ethereum/go-ethereum/vendor/github.com/rjeczalik/notify
github.com/ethereum/go-ethereum/vendor/golang.org/x/crypto/pbkdf2
github.com/ethereum/go-ethereum/vendor/golang.org/x/crypto/scrypt
github.com/ethereum/go-ethereum/vendor/gopkg.in/fatih/set.v0
github.com/ethereum/go-ethereum/cmd/internal/browser
github.com/ethereum/go-ethereum/vendor/github.com/syndtr/goleveldb/leveldb/util
github.com/ethereum/go-ethereum/vendor/github.com/syndtr/goleveldb/leveldb/cache
github.com/ethereum/go-ethereum/vendor/github.com/syndtr/goleveldb/leveldb/comparer
github.com/ethereum/go-ethereum/vendor/github.com/syndtr/goleveldb/leveldb/storage
github.com/ethereum/go-ethereum/vendor/github.com/syndtr/goleveldb/leveldb/filter
github.com/ethereum/go-ethereum/vendor/github.com/rcrowley/go-metrics/exp
github.com/ethereum/go-ethereum/vendor/github.com/syndtr/goleveldb/leveldb/opt
github.com/ethereum/go-ethereum/vendor/github.com/golang/snappy
github.com/ethereum/go-ethereum/metrics
```

Assuming we have already installed Raspbian, if we start by updating the installed packaged software to the latest versions.

Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started.

Slater Technologies

Start the Node

```
pi@chainpi:~ $
pi@chainpi:~ $ geth --syncmode light --cache 64 --maxpeers 12
INFO [01-30|17:38:56] Starting peer-to-peer node               instance=Geth/v1.7.3-stable-4bb3c89d/linux
-arm/go1.7.4
INFO [01-30|17:38:56] Allocated cache and file handles         database=/home/pi/.ethereum/geth/lightchai
ndata cache=64 handles=1024
INFO [01-30|17:38:56] Writing default main-net genesis block
INFO [01-30|17:39:02] Initialised chain configuration          config="{ChainID: 1 Homestead: 1150000 DAO
: 1920000 DAOSupport: true EIP150: 2463000 EIP155: 2675000 EIP158: 2675000 Byzantium: 4370000 Engine: eth
ash}"
INFO [01-30|17:39:02] Disk storage enabled for ethash caches   dir=/home/pi/.ethereum/geth/ethash count=3
INFO [01-30|17:39:02] Disk storage enabled for ethash DAGs     dir=/home/pi/.ethash               count=2
INFO [01-30|17:39:02] Added trusted checkpoint                 chain name="ETH mainnet"
INFO [01-30|17:39:02] Loaded most recent local header          number=0 hash=d4e567…cb8fa3 td=17179869184
INFO [01-30|17:39:02] Starting P2P networking
INFO [01-30|17:39:04] UDP listener up                          self=enode://b7a599e0eee28d102bed0e874e9b0
d76fe89b0b6fb06354f47339620c6010df4a8f4d5ec6092ef914e220d7a2e567530708be138faf6c2168fc86abdb818e52e@[::]:
30303
INFO [01-30|17:39:04] RLPx listener up                         self=enode://b7a599e0eee28d102bed0e874e9b0
d76fe89b0b6fb06354f47339620c6010df4a8f4d5ec6092ef914e220d7a2e567530708be138faf6c2168fc86abdb818e52e@[::]:
30303
WARN [01-30|17:39:04] Light client mode is an experimental feature
INFO [01-30|17:39:04] IPC endpoint opened: /home/pi/.ethereum/geth.ipc
```

```
$ geth --syncmode light --cache 64 --maxpeers 12
```

Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started.

Slater Technologies

Start the Node

If we ran geth without any arguments, it would start up a node and attempt to sync the entire public mainnet blockchain. Which, at >50GB in size and constantly growing, might not be a great idea on an embedded computer. So instead we start the node in light synchronisation mode. This only fetches block headers as they appear and other parts of the blockchain on-demand.

To force the node to exit simply press CTRL-C. To run it as a service at boot time:

```
$ sudo vi /etc/systemd/system/geth@.service
```

Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started.

Slater Technologies

Start the Node

(replace "vi" with your favourite text editor)

And then enter:

```
[Unit]
Description=Ethereum daemon
Requires=network.target

[Service]
Type=simple
User=%I
ExecStart=/usr/local/bin/geth --syncmode light --cache 64 --maxpeers 12
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Save the file. Following which to have the Ethereum node run as the “pi” user:

```
$ sudo systemctl enable geth@pi.service
$ sudo systemctl start geth@pi.service
```

Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started.

Slater Technologies

Start the Node

```
pi@chainpi:~ $
pi@chainpi:~ $ geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.7.3-stable-4bb3c89d/linux-arm/go1.7.4
 modules: admin:1.0 debug:1.0 eth:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> eth.accounts
["0xc0dad8541fd851d5094b4574899ebcf236cd3666"]
>
> 
```

With our Ethereum node running as a service we can now attach to it using:

```
$ geth attach
```

This gives us an interactive JavaScript console. From here we can call functions, such as:

```
> eth.accounts
```

Which will list the current accounts.

Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started.

Slater Technologies

Start the Node

```
pi@chainpi:~ $
pi@chainpi:~ $ geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.7.3-stable-4bb3c89d/linux-arm/go1.7.4
 modules: admin:1.0 debug:1.0 eth:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

>
> admin.peers
[{
    caps: ["eth/62", "eth/63", "les/1", "les/2"],
    id: "a979fb575495b8d6db44f750317d0f4622bf4c2aa3365d6af7c284339968eef29b69ad0dce72a4d8db5ebb4968de0e3b
ec910127f134779fbcb0cb6d3331163c",
    name: "Geth/v1.7.3-stable/linux-amd64/go1.7",
    network: {
      localAddress: "10.0.10.100:60604",
      remoteAddress: "52.16.188.185:30303"
    },
    protocols: {
      les: {
        difficulty: 2.2875535680775693e+21,
        head: "70441005f3c27b9c37c7ebeabd75a62c42543740261674e00b14caf30e0017b6",
        version: 2
      }
    }
}]
>
> █
```

Or to get information about the connected peers:

```
> admin.peers
```

Note that the light client protocol is still in development, somewhat experimental and does rely on full peers/nodes enabling support for it. As such, it may not be entirely practical at the time of writing to transact on the Ethereum mainnet blockchain using this. That said, things are moving fast and this situation could easily change in the not too distant future.

Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started.

Slater Technologies

OPERATING ETHEREUM ON RASPBERRY PI PROVING ETHEREUM WORKS ON RASPBERRY PI

Slater Technologies

Stopping mainnet synchronisation

If you followed along with Part 1 and configured a node to use mainnet and run in light synchronisation mode, this can be stopped and start-up disabled with:

```
$ sudo systemctl stop geth@pi.service

$ sudo systemctl disable geth@pi.service
```

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Create a New Account

Creating a new account

```
pi@chainpi:~ $
pi@chainpi:~ $ geth --datadir .designspark account new
Your new account is locked with a password. Please give a password. Do not forget thi
s password.
Passphrase:
Repeat passphrase:
Address: {1fd4027fe390abaa49e5afde7896ff1e5ecacabf}
pi@chainpi:~ $
pi@chainpi:~ $
```

We need a name for our new blockchain network and for the purposes of this example, we'll use "DesignSpark". By default Ethereum stores data in a sub-directory of your home directory named ".ethereum", i.e. a hidden directory on Linux/BSD. So as to keep the data for our private blockchain separate, we'll use ".designspark".

If we start by creating a new account:

```
$ geth --datadir .designspark account new
```

And take a note of the address of the account, since we'll need this when we initialise the new network if we would like to preallocate any funds to it.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Initialize The Ethereum Blockchain at Block 0

In the beginning, there was block 0

```
pi@chainpi:~ $
pi@chainpi:~ $ geth --datadir .designspark init designspark.json
INFO [03-03|16:59:17] Allocated cache and file handles          database=/home/pi/.des
ignspark/geth/chaindata cache=16 handles=16
INFO [03-03|16:59:17] Writing custom genesis block
INFO [03-03|16:59:17] Successfully wrote genesis state          database=chaindata
                        hash=acf1f3…047b81
INFO [03-03|16:59:17] Allocated cache and file handles          database=/home/pi/.des
ignspark/geth/lightchaindata cache=16 handles=16
INFO [03-03|16:59:17] Writing custom genesis block
INFO [03-03|16:59:17] Successfully wrote genesis state          database=lightchaindat
a                            hash=acf1f3…047b81
pi@chainpi:~ $
pi@chainpi:~ $ 
```

There has to be a first link in a chain and a blockchain is no different, requiring a genesis block to be created that will be used by the initial set of nodes which are to participate in the network. This is configured via a JSON file and the contents of the one we used, for example, are below.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

JSON File to Create Block 0

```
{
    "config": {
        "chainId": 555,
        "homesteadBlock": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
    "difficulty": "20",
    "gasLimit": "2100000",
    "alloc": {
        "1fd4027fe390abaa49e5afde7896ff1e5ecacabf":
        { "balance": "20000000000000000000" }
    }
}
```

The 'chainId' is a numerical value that identifies the network and a list of those currently in use by public networks can be found here. We needed to pick a number for our private DesignSpark network and for some reason 555 seemed like a good choice — you could use a different number.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Other Parameters

- **homesteadBlock.** Homestead is an Ethereum release and for our chain, this is set to 0.
- **eip155Block.** Our chain won't hard-fork for EIP155, so this is set to 0.
- **eip158Block.** Our chain won't hard-fork for EIP158, so this is set to 0.
- **difficulty.** This sets the mining difficulty and in our case, we want this reasonably low.
- **GasLimit.** This is the limit of the Gas cost per block.
- **alloc.** This is where we can pre-allocate funds to accounts.

Ethereum Improvement Proposals (EIPs) describe standards for the Ethereum platform and new ones may be issued to address shortcomings. As a network grows it may be forked at a certain point to allow EIPs to be incorporated. This is not so much a concern for our private network, but for details of where EIP155 was implemented with mainnet and what this does, see Spurious Dragon.

Gas is the unit used as a measure of how much work an action or set of actions takes to perform. Thereby allowing a cost to be attached to executing *smart contracts* — the objects which contain code functions and that live on the blockchain and are able to interact with other contracts, make decisions, store data, and send ether to others. More on this in a future post.

Alloc allows us to preallocate funds to one or more accounts. Here funds have been allocated to the address of the account we created earlier.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Initialize the Network

Having saved our config file to designspark.json we can now initialise the network with:

```
$ geth --datadir .designspark init designspark.json
```

And that's it, we've written out our genesis block and now have the very beginnings of our new network. Provided subsequent nodes are initialised in the same way, they can become members too.

Slater Technologies

Initialize the Network

Starting the first node

```
INFO [03-03|17:00:10] Disk storage enabled for ethash DAGs     dir=/home/pi/.ethash
            count=2
INFO [03-03|17:00:10] Initialising Ethereum protocol           versions="[63 62]" net
work=555
INFO [03-03|17:00:10] Loaded most recent local header          number=0 hash=acf1f3…0
47b81 td=20
INFO [03-03|17:00:10] Loaded most recent local full block      number=0 hash=acf1f3…0
47b81 td=20
INFO [03-03|17:00:10] Loaded most recent local fast block      number=0 hash=acf1f3…0
47b81 td=20
INFO [03-03|17:00:10] Regenerated local transaction journal    transactions=0 account
s=0
INFO [03-03|17:00:10] Starting P2P networking
INFO [03-03|17:00:10] RLPx listener up                         self="enode://01f5ecc7
c232f7571175bffc71c4e1608e1308e2ce7fd6ed3ae17d5e97e2d5253dcaa854286f99991d671788127f7
902fa56d20875eabae49665a515da105047@[::]:30303?discport=0"
INFO [03-03|17:00:10] IPC endpoint opened: /home/pi/.designspark/geth.ipc
INFO [03-03|17:00:10] HTTP endpoint opened: http://127.0.0.1:8080
Welcome to the Geth JavaScript console!

instance: Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7.4
coinbase: 0x1fd4027fe390abaa49e5afde7896ff1e5ecacabf
at block: 0 (Thu, 01 Jan 1970 00:00:00 UTC)
 datadir: /home/pi/.designspark
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1
.0 web3:1.0

> 
```

To start the first node with the JavaScript console we enter:

```
$ geth --identity chainpi --rpc --rpcport 8080 --rpccorsdomain "*" --da
```

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Start-up Parameters

What do all the parameters mean?

- **indentity.** This sets the Ethereum node identity.
- **rpc*.** The various RPC settings configure the available APIs and who has access to them.
- **datadir.** We obviously need to use the same data directory as before.
- **nodiscover.** This means our node is not discoverable.
- **networkid.** This needs to be the same numerical ID configured during initialisation.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Node Account Balance

```
> eth.accounts
["0x1fd4027fe390abaa49e5afde7896ff1e5ecacabf"]
>
> primary = eth.accounts[0]
"0x1fd4027fe390abaa49e5afde7896ff1e5ecacabf"
>
> balance = web3.fromWei(eth.getBalance(primary), "ether");
20
>
>
```

Once we've entered the console we can use eth.accounts to list the available accounts and eth.getBalance to check the balance.

```
> eth.accounts

> primary = eth.accounts[0]

> balance = web3.fromWei(eth.getBalance(primary), “ether”);
```

Note how the figure returned is much smaller than what we preallocated via designspark.json? That's because the balance in *Ether* was returned, whereas during initialisation the allocation was actually specified in *Wei*, a far smaller unit.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Starting Up Additional Nodes

Creating a 2nd node

```
>
> eth.accounts
["0xbdcb5581d531bf819f48bd74a3f667af240a66a7"]
> primary = eth.accounts[0]
"0xbdcb5581d531bf819f48bd74a3f667af240a66a7"
> balance = web3.fromWei(eth.getBalance(primary), "ether");
0
>
> 
```

A blockchain network with only one node wouldn't be much use and so we'll create a second one. This time it's recommended to use a computer with a little more RAM, such as a laptop or desktop running Debian/Ubuntu, as this is likely to be needed should we wish to run a miner at some point.

To recap, the steps involved are:

1. Install geth.
2. Run the command as above to create a new account.
3. Initialise using the same JSON configuration file.
4. Start the node as before, but this time use a different identity!

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Check the Balance on the Newly Added Node

Once we've done this, the node has been started and dropped into the JavaScript console, we can then once again check the new account and its balance with:

```
> eth.accounts

> primary = eth.accounts[0]

> balance = web3.fromWei(eth.getBalance(primary), "ether");
```

This time we should see we have a balance of 0, as we didn't preallocate any funds to the account.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Starting Up the Node

To start the node:

$ geth --identity raspberrypi1 --rpc --rpcport 8080 --rpccorsdomain "*" --datadir .designspark --port 30302 --nodiscover --rpcapi "db,eth,net,web3" --networkid 555 console

To check the balance that we allocated:

>eth.accounts

```
pi@raspberrypi:~ $ geth --identity raspberrypi1 --rpc --rpcport 8080 --rpccorsdomain "*" --datadir .designspark --port 30302 --nodiscover --rpcapi "db,eth,net,web3" --networkid 555 console
INFO [04-19|18:55:44] Starting peer-to-peer node               instance=Geth/raspberrypi1/v1.7.3-stable-4bb3c89d/linux-arm/go1.7.4
INFO [04-19|18:55:44] Allocated cache and file handles         database=/home/pi/.designspark/geth/chaindata cache=128 handles=1024
WARN [04-19|18:55:44] Upgrading database to use lookup entries
INFO [04-19|18:55:44] Database deduplication successful        deduped=0
INFO [04-19|18:55:44] Initialised chain configuration          config="{ChainID: <nil> Homestead: 0 DAO: <nil> DAOSupport: false EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil> Engine: u
nknown}"
INFO [04-19|18:55:44] Disk storage enabled for ethash caches   dir=/home/pi/.designspark/geth/ethash count=3
INFO [04-19|18:55:44] Disk storage enabled for ethash DAGs     dir=/home/pi/.ethash                  count=2
INFO [04-19|18:55:44] Initialising Ethereum protocol           versions="[63 62]" network=555
INFO [04-19|18:55:44] Loaded most recent local header          number=0 hash=42841b…fec43d td=20
INFO [04-19|18:55:44] Loaded most recent local full block      number=0 hash=42841b…fec43d td=20
INFO [04-19|18:55:44] Loaded most recent local fast block      number=0 hash=42841b…fec43d td=20
INFO [04-19|18:55:44] Regenerated local transaction journal    transactions=0 accounts=0
INFO [04-19|18:55:44] Starting P2P networking
INFO [04-19|18:55:44] RLPx listener up                         self="enode://07170f08de0e608cf8c0eaaf5466518264cdaa685834b696f12949811b1398b1bc2f74aef39bb1681d722f8b96b7dbb2951da62201ea860e
65c4e099b069791f8[::]:30302?discport=0"
INFO [04-19|18:55:44] IPC endpoint opened: /home/pi/.designspark/geth.ipc
INFO [04-19|18:55:44] HTTP endpoint opened: http://127.0.0.1:8080
Welcome to the Geth JavaScript console!

instance: Geth/raspberrypi1/v1.7.3-stable-4bb3c89d/linux-arm/go1.7.4
coinbase: 0x9fc1843c34bcc15e926a2f308748aaaec44a406c
at block: 0 (Thu, 01 Jan 1970 00:00:00 UTC)
 datadir: /home/pi/.designspark
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> INFO [04-19|18:55:46] Mapped network port                     proto=tcp extport=30302 intport=30302 interface="UPNP IGDv2-IP1"
> eth.accounts
["0x9fc1843c34bcc15e926a2f308748aaaec44a406c"]
> admin.peers
[]
> eth.getBalance("9fc1843c34bcc15e926a2f308748aaaec44a406c")
20000000000000000000
```

Slater Technologies

Creating the Second Node

Creating 2nd node:

- **Follow the same steps as mentioned above for node 1.**
- **Create an account**
- **Use the same .json file.**

Slater Technologies

Connect the Peers

Connecting the peers

```
>
> admin.nodeInfo.enode
"enode://5156218119a3697389a34bf0a19ceca49d9f3d06948836b8cc6c206c9f7b7081e64537eeb0f9
c059561736a8e7cb6ebbe438028dd949d0f69f4cab642c11d46c@[::]:30303?discport=0"
>
>
```

Since we don't want our nodes to be discoverable we started them with the --nodiscover option, meaning that we'll need some way of configuring them to peer. We can achieve this by creating a file called static-nodes.json located in the datadir, which in our case is ~/.designspark.

First, though we need to get the *enode URL* for each of our nodes by entering at the JavaScript console on each system:

```
> admin.nodeInfo.enode
```

We then populate the static-nodes.json file with this info as follows:

```
[
"enode://01f5ecc7c232f7571175bffc71c4e1608e1308e2ce7fd6ed3ae17d5e97e2d!
]
```

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Connecting Peers

Note how [::] has been replaced by the node IP address and the ?discport=0 suffix omitted.

```
>
> admin.peers
[{
    caps: ["eth/63"],
    id: "5156218119a3697389a34bf0a19ceca49d9f3d06948836b8cc6c206c9f7b7081e64537eeb0f9
c059561736a8e7cb6ebbe438028dd949d0f69f4cab642c11d46c",
    name: "Geth/snow/v1.8.0-stable-5f540757/linux-amd64/go1.9.4",
    network: {
      localAddress: "10.100.1.196:30303",
      remoteAddress: "10.100.1.229:41152"
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0xacf1f3c3898431e37b0c07c7421c203d9a90475a51b8d1f2c7048de207047b81",
        version: 63
      }
    }
}]
>
```

Once this file has been created on both nodes we can exit geth via CTRL-D and then re-launch the console. Following which if we enter on the first node:

```
> admin.peers
```

We should see the details for the second node.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Peer Verification

```
> admin.peers
```

We should see the details for the second node.

```
>
> admin.peers
[{
    caps: ["eth/63"],
    id: "01f5ecc7c232f7571175bffc71c4e1608e1308e2ce7fd6ed3ae17d5e97e2d5253dcaa854286f
99991d671788127f7902fa56d20875eabae49665a515da105047",
    name: "Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7.4",
    network: {
      inbound: false,
      localAddress: "10.100.1.229:41152",
      remoteAddress: "10.100.1.196:30303",
      static: true,
      trusted: false
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0xacf1f3c3898431e37b0c07c7421c203d9a90475a51b8d1f2c7048de207047b81",
        version: 63
      }
    }
}]
>
```

Repeating this on the second node we should then see the node info for the first.

So now we have our own private blockchain network complete with two nodes, each configured with an account and one of those with preallocated funds.

Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain.

Slater Technologies

Conclusion

- **Blockchain:**
  - A technical marvel made possible by software, hardware, strong cryptography, and the Internet
  - Has made significant progress in only 100+ months
  - Has significant strengths and a few limitations too
  - Blockchain is starting to be widely used to automate trusted computing transactions and increase efficiencies in many industries
  - Has great potential because of popular support of talented nerds, and now major players in major industries
  - The excitement about the blockchain is based on its ability to serve as a tool for achieving and maintaining integrity in purely distributed peer-to-peer systems that have the potential to change whole industries due to disintermediation.

Source: Drescher, D. (2017). Blockchain Basics. Frankfort am Main, Germany: Apress.

Slater Technologies

Conclusion

- **Ethereum and Raspberry :**
  - Ethereum is FREE
  - Raspberry Pi devices are cheap, plentiful, and easy to purchase at places like www.amazon.com .
  - There is a wealth of free information available on how to use these important technologies.
  - Everyone who is interested in Blockchain and Ethereum should consider learning Blockchain and Raspberry Pi, so they will be familiar with and be able to use it as a spring board to learn Blockchain and Blockchain Development.
- **Chicago Blockchain Community:**
  - Join us, participate and get involved.
  - Chicago Blockchain Project
    - **https://www.meetup.com/chicagoblockchainproject/**
  - Chicago Bitcoin and Open Blockchain Meetup
    - **https://www.meetup.com/Bitcoin-Open-Blockchain-Community-Chicago/**

Source: Drescher, D. (2017). Blockchain Basics. Frankfort am Main, Germany: Apress.

Slater Technologies

Thank You, for Being Here

Special Thanks To:

Joe Hernandez
Co-Founder of the
Chicago Blockchain Project

Hannah Rosenburg
Co-Founder of the
Chicago Bitcoin and Open
Blockchain Meetup

Slater Technologies

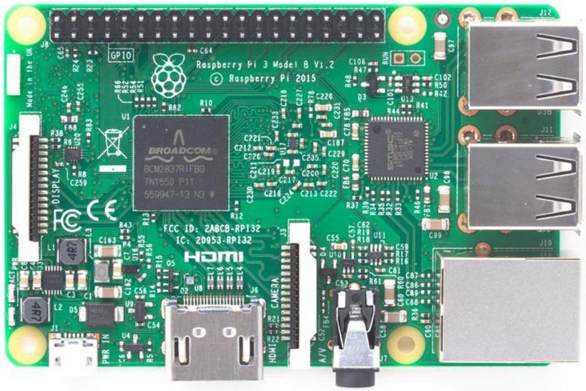
Special Thanks To:

Vitalik Buterin
Inventor of Ethereum

Slater Technologies

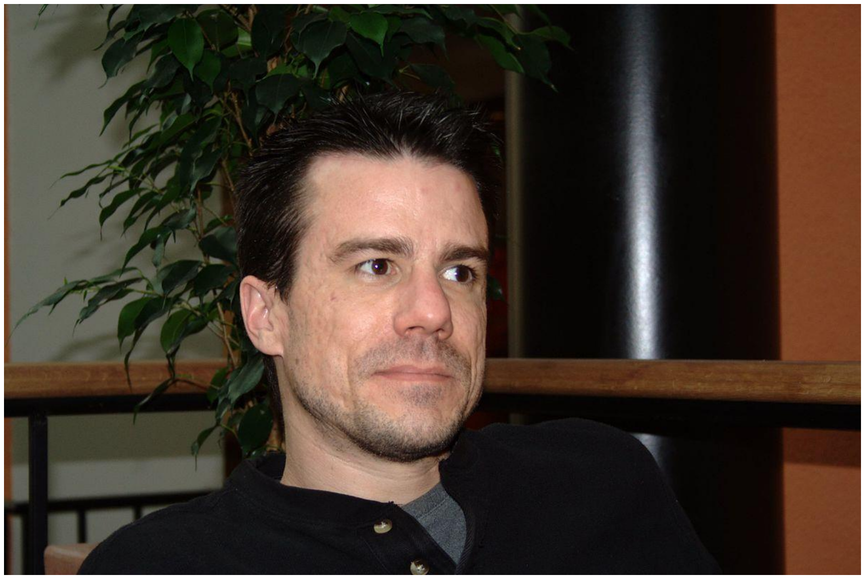
Special Thanks To:

Eben Upton
Inventor of Raspberry Pi



Slater Technologies

Special Thanks To:



The Late Ian Murdock
Creator of the Debian
Linux Distribution
(Rest in Peace)

debian
GNU/Linux

Slater Technologies

Very Special Thanks To:

Vivek Elankumaran
velankumaran@hawk.iit.edu

Thamizharasan ("Tamil") Babu
tbabu1@hawk.iit.edu

ILLINOIS INSTITUTE OF TECHNOLOGY

Slater Technologies

ILLINOIS INSTITUTE OF TECHNOLOGY

SHAMELESS PLUG FOR ILLINOIS TECH

Slater Technologies

Meet the 2018 *Fifty for the Future* Award Recipients From the Illinois Institute of Technology

Jinqing Huang, China

Uday Tak, India

Nikita Kothari, India

ILLINOIS TECHNOLOGY FOUNDATION

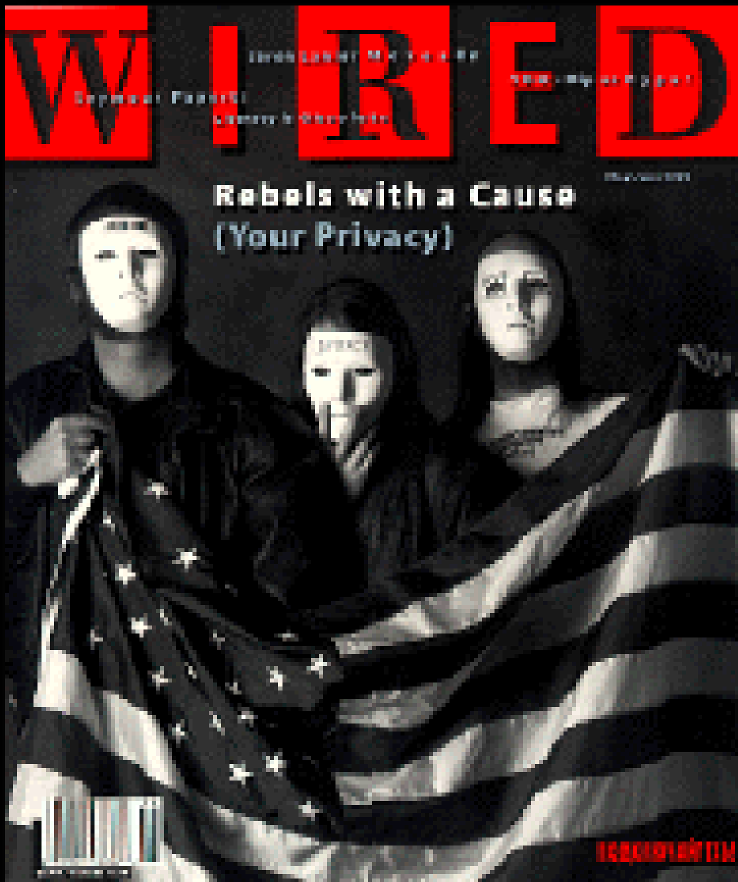
Mohnish Anand, India

Dhivya Venkatachalam, India

ILLINOIS INSTITUTE OF TECHNOLOGY

Moein Mayeh, Iran

Questions?



Wired Magazine, February 1993

General George S. Patton

Slater Technologies

REFERENCES

Slater Technologies

References

- Antonopoulos, A. M. (2018). Mastering Bitcoin: Programming the Open Blockchain, second edition. Sebastopol, CA: O'Reilly Media, Inc.
- Back, A. (2017). Exploring Ethereum with Raspberry Pi - Part 1: Getting Started. Retrieved from https://www.rs-online.com/designspark/exploring-ethereum-with-raspberry-pi-part-1-getting-started on March 1, 2018.
- Back, A. (2017). Exploring Ethereum with Raspberry Pi Part 2: Creating a Private Blockchain. Retrieved from https://www.rs-online.com/designspark/exploring-ethereum-with-raspberry-pi-part-2-creating-a-private-blockchain on March 1, 2018.
- Bahga, A. and Madisetti, V. (2017). Blockchain Applications: A Hands-On Approach. Published by Arshdeep Bahga and Vijay Madisetti. www.blockchain-book.com .
- Bambara, J. J. and Allen P. R. (2018). Blockchain: A Practical Guide to Developing Business, Law, and Technology Solutions. New York, NY: McGraw-Hill Education.
- Bashir, I. (2018). Mastering Blockchain, second edition. Birmingham, UK: Packt Publishing Ltd.
- BBC. (2014). Troubled MtGox Bitcoin boss emerges after shut down Retrieved from http://www.bbc.com/news/technology-26352442 on February 26, 2014.
- Bitcoin. (2014). Bitcoin. Retrieved from https://bitcoin.com/ on April 10, 2014.
- Bitcoin Charts. (2014). Bitcoin Charts. Retrieved from http://bitcoincharts.com/ on March 1, 2014.
- Bitcoin Foundation. (2014). Bitcoin Foundation. Retrieved from https://bitcoinfoundation.org/ on April 10, 2014.

Slater Technologies

References

- Dannen, C. (2017). Introducing Ethereum and Solidity: Foundations of Crytocurrency and Blockchain Programming for Beginners. New York, NY: Apress
- De Filippi, P. and Wright, A. (2018). Blockchain and the Law: the Rule of Code. Cambridge, MA: President and Fellows of Harvard College.
- Dhillon, V., Metcalf, D., and Hooper, M. (2017). Blockchain Enabled Applications: Understand the Blockchain Ecosystem and How to Nake It Work for You. New York, NY: Apress.
- Drescher, D. (2017). Blockchain Basics. Frankfort am Main, Germany: Apress.
- Eddison, L. (2017). Ethereum: A Deep Dive into Ethereum. Published by Leonard Eddison.
- Ethereum Community. (2017). Ethereum Homestead Documentation 0.1, March 1, 2017. Retrieved from https://media.readthedocs.org/pdf/ethereum-homestead/latest/ethereum-homestead.pdf on March 20, 2018
- Etwaru, R. (2017). Blockchain Trust Companies. Indianapolis, IN: Dog Ear Publishing.
- Fazal, R. (2014). Raspberry Pi. Retrieved from https://www.slideshare.net/rhnfzl/raspberrypi-32339482 on April 15, 2018.
- Gerard, D. (2107), Attack of the 50 Foot Blockchain: Bitcoin, Blockchain, Ethereum, and Smart Contracts. Published by David Gerard. www.davidgerard.co.uk/blockchain .
- Casey, M. J. and Vigna, P. (2018). The Truth Machine: The Blockchain Reference and the Future of Everything. New York, NY: St. Martin's Press.
- Champagne, P. (2014). The Book of Satoshi: The Collected Writings of Bitcoin Creator Satoshi Nakamoto. Published by E53 Publishing, LLC.
- Holton, J. and Fratangelo, T. (2016). Raspberry Pi Architecture. Retrieved from https://www.macs.hw.ac.uk/~hwloidl/Courses/F28HS/slides_RPi_arch.pdf on April 15, 2018.

Slater Technologies

References

- Laurence, T. (2017). Blockchain for Dummies. Hoboken, NJ: John Wiley & Sons, Inc.
- Lee, T. B. (2013). 12 questions about Bitcoin you were too embarrassed to ask. Retrieved from http://www.washingtonpost.com/blogs/the-switch/wp/2013/11/19/12-questions-you-were-too-embarrassed-to-ask-about-bitcoin/ on November 19, 2013.
- Nakamoto. S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved from https://bitcoin.org/bitcoin.pdf on November 1, 2013.
- Noyola, E. (2018). Ethereum: Ethereum, Tokens and Smart Contracts. Published by Eugenio Noyola.
- Peterson, O. (2018). An Introduction of Programmable Smart Contracts in Ethereum (Pt 1). Retrieved from https://www.linkedin.com/pulse/introduction-programmable-smart-contracts-ethereum-p1-%CE%BE%CE%BE%CE%BE-oliver/ on February 1, 2018.
- Petrovan, B. (2014) Researchers find Android apps that covertly mine Dogecoin, one of them with more than a million downloads. Retrieved from http://www.androidauthority.com/dogecoin-mining-android-apps-362142/ on March 27, 2014.
- Prusty, N. (2017). Building Blockchain Projects: Building Decentralized Blockchain Applications with Ethereum and Solidity. Birmingham, UK: Pact Publishing.
- SCGNEWS. (2014). The IRS Just Declared War on Bitcoin - Retroactively. Retrieved from http://scgnews.com/the-irs-just-declared-war-on-bitcoin-retroactively on March 27, 2014.
- White, A. (2018). Blockchain: Discover the Technology Behind Smart Contracts, Wallets, Mining, and Cryptocurrency. Published by Andrew K. White.

Slater Technologies

References

- Buterin, V. (2015). The Problem of Censorship. Retrieved from http://blog.ethereum.org/2015/06/06/the-problem-of-censorship/ on April 15, 2018.
- Buterin, V. (2015). Visions - Part 1: The Value of Blockchain Technology
- Buterin, V. (2015). Visions - Part 2: The Problem of Trust
- Buterin, V. (2015). Light Clients – Proof of Stake.
- Buterin, V. (2015). SuperRationality DAOs.
- Ethereum Ecosystem Contributors. (2018). Various articles and resources. Retrieved from http://ecosystem.eth.guide on April 15, 2018.
- White, A. (2018). Blockchain: Discover the Technology Behind Smart Contracts, Wallets, Mining, and Cryptocurrency. Published by Andrew K. White.

Slater Technologies

References: Best Blockchain Texts

- **Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners**
  - Chris Dannen

- **Mastering Blockchain - Second Edition**
  - by Imran Bashir

- **Blockchain Enabled Applications: Understand the Blockchain Ecosystem and How to Make it Work for You**
  - by Vikram Dhillon, David Metcalf, Max Hooper

- **Ethereum, tokens & smart contracts: Notes on getting started**
  - by Eugenio Noyola

- **Distributed Ledger Technology: The Science of the Blockchain**
  - by Roger Wattenhofer

- **The Book of Satoshi: The Collected Writings od Bitcoin Creator Satoshi Nakamoto**
  - By Phil Champagne

Slater Technologies

PRACTICAL EXERCISES

Practical Exercises

1. Create and decode a hash
2. Decode a hash
3. Create a Blockchain record
4. Build a working Ethereum Blockchain Network

Slater Technologies

Practical Exercise 01

- Create a hash

1. Visit this website and type information about yourself or a message, and use the SHA 256 hash algorithm to create a hash http://www.hashemall.com/

2. Save the hash value.

3. Visit this website to decrypt your hash message:
   http://md5decrypt.net/en/Sha256/

Slater Technologies

Practical Exercise 02

- Decode a hash

Hash:
9ec4c12949a4f31474f299058ce2b22a

This hash is found on the emblem of U.S. Cybercommand. It is a message that was hashed

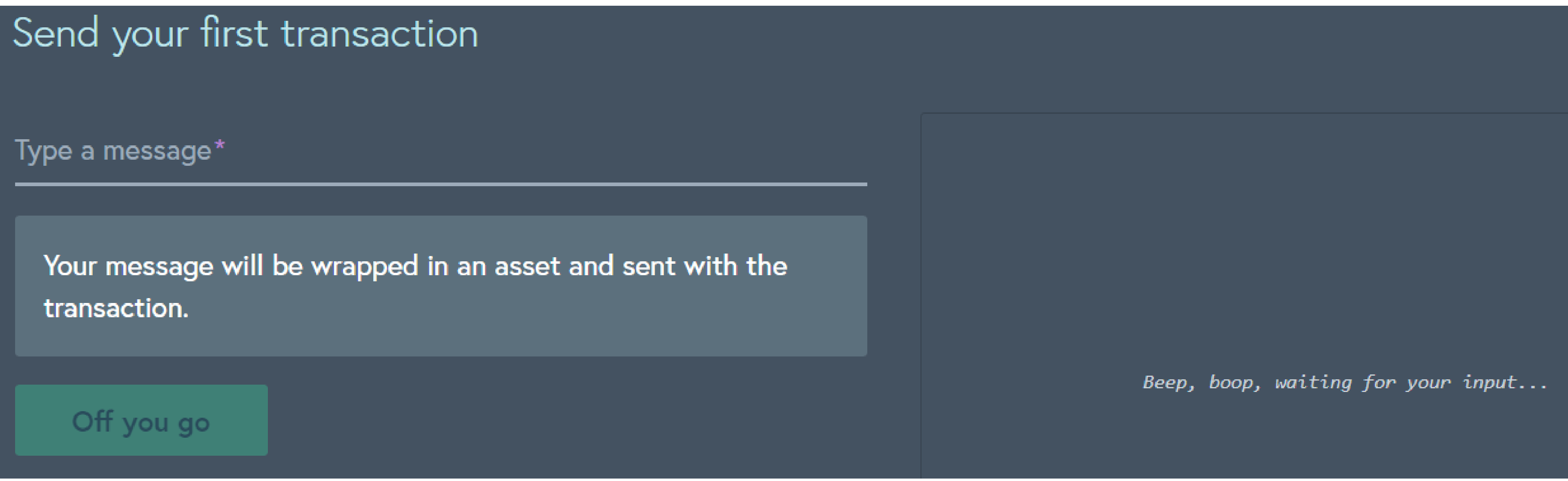
Using a commonly known hashing algorithm. Use this website to see if you can decrypt this Hash and see the message: http://www.hashemall.com/

Slater Technologies

Practical Exercise 03

- Create a Blockchain record

Visit this website and create your first Blockchain record:
https://www.bigchaindb.com/getstarted/
Copy and Save the results to a local text file named:
YYYY_ MMDD_FirstName_LastName_My_First_Blockchain_Transaction_.txt



Slater Technologies

Practical Exercise 04

- Build a Working Prototype Ethereum Blockchain using Raspberry Pi

Slater Technologies

Practical Exercise 04
Part 01 – Getting Started

- Setting up Ethereum on Raspberry Pi – Part 01
- Visit this link and follow the instructions:
  - https://www.rs-online.com/designspark/exploring-ethereum-with-raspberry-pi-part-1-getting-started

Slater Technologies

Practical Exercise 04
Part 02 – Setting up a Private Blockchain

- Setting up Ethereum on Raspberry Pi – Part 02
- Visit this link and follow the instructions:
  - https://www.rs-online.com/designspark/exploring-ethereum-with-raspberry-pi-part-2-creating-a-private-blockchain

Slater Technologies

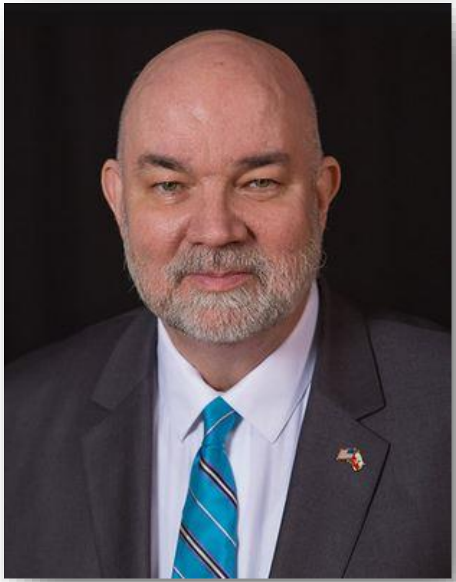
Dedication

- Dedicated with never-ending love, respect, and gratitude to my dear Father-in-law and Mother-in-Law, Wiesiek Roguski (http://billslater.com/wiesiek) and Wiesia Roguska (http://billslater.com/wiesia).

My Lovely Polish Family Who Adopted Me Loved, Nurtured, and Encouraged Me.

Presenter Bio:
William Favre Slater, III

- **Lives in Chicago; Cybersecurity professional by day, Professor at night**
- **Married to my Best Friend and Soul Mate, Ms. Joanna Roguska**
- **Current Position – Project Manager / Sr. IT Consultant at Slater Technologies, Inc.** Working on projects related to
  - Third-Party Cybersecurity Risk Assessments
  - Security reviews and auditing
  - Blockchain consulting
  - ISO 27001 Project Implementations
  - Subject Matter Expert for preparing Risk Management and Security Exams at Western Governor's State University in UT
  - Providing subject matter expert services to Data Center product vendors and other local businesses.
  - Designing and creating a database application that streamlines program management, security management, risk management and reporting activities, for management of teams of IT workers and developers in teleworking environments. It will first be a Windows application and then be ported to the web.
  - Developing and presenting technical training materials for undergraduate and graduate students at the Illinois Institute of Technology in the areas of Blockchain and Blockchain development, Data Center Operations, Data Center Architecture, Cybersecurity Management, and Information Technology hardware and software.
  - Created an eBook with articles about Security, Risk Management, Cyberwarfare, Project Management and Data Center Operations
  - Professor at Illinois Tech for 10 years



William Favre Slater, III
slater@billslater.com

Slater Technologies

William Favre Slater, II

- 312-758-0307
- slater@billslater.com
- williamslater@gmail.com
- http://billslater.com/interview
- 1515 W. Haddon Ave., Unit 309
  Chicago, IL 60642
  United States of America

William Favre Slater, III