UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science

Oliver Vainikko

Microtransactions for IoT devices

Bachelor's Thesis (9 EAP)

Supervisor(s):
Ulrich Norbisrath

Tartu 2022

Table of Contents

Microtransactions for IoT devices

Abstract:

This study investigates the integration of Internet of Things (IoT) and blockchain technology, specifically focusing on the use of the XRP Ledger and the ESP8266 microcontroller in an IoT payment system within a university-based coffee brewing niche. The primary goals of the research were to understand the benefits and challenges of this integration, design and implement a suitable IoT payment system, and assess the user-friendliness and cost-effectiveness of such a system. Key findings of the study were the validation of lower transaction fees on the XRP Ledger and the successful demonstration of a scalable IoT payment system. Despite the limitations, such as the insufficient number of payments made on the live ledger, the research provides a promising direction towards developing efficient, secure, and cost-effective IoT payment solutions. Potential future research could explore payment channels for offline transactions, the creation of unique tokens for selling coffee, and a detailed analysis of the fee history on the ledger. This research facilitates a promising future for IoT microtransactions.

Keywords: Internet of Things, Blockchain, XRP Ledger, ESP8266 microcontroller, IoT payment system, Microtransactions, Coffee brewing niche.

CERCS: P175 (Informatics, systems theory)

Mikrotransaktsioonid IoT seadmetele

Lühikokkuvõte:

Töös uuritakse asjade interneti (ing *Internet of Things, IoT*) ja plokkahel tehnoloogia kombineerimist, keskendudes *XRP Ledger*'i ja *ESP8266* mikrokontrolleri kasutamisele asjade interneti maksesüsteemis. Uurimistöö peamised eesmärgid olid mõista selle kombinatsiooni eeliseid ja väljakutseid, kujundada ja rakendada asjade interneti maksesüsteem ning hinnata sellise süsteemi kasutajasõbralikkust ja kulutõhusust. Töö peamised leiud olid madalamate tehingutasude valideerimine *XRP Ledgeri*s ja skaleeruva asjade interneti maksesüsteemi demonstreerimine. Vaatamata piirangutele, nagu näiteks reaalajas plokkahelal tehtud maksete kesine arv, annab uurimus paljulubava suuna turvaliste ja kulutõhusate asjade interneti makselahenduste arendamiseks. Võimalikud tulevased uurimused võiksid uurida võrguühenduseta tehingute maksekanaleid (ing *Payment Channel*), unikaalsete tokenite (ing *Unique Token*) loomist kohvi müümiseks ning plokkahela maksetasude ajaloo detailset analüüsi. Uurimustöös täheldati, et asjade interneti mikrotransaktsioonide jaoks paljutõotavat tulevikku.

Võtmesõnad: Asjade Internet, Plokkahel, XRP Ledger, ESP8266 mikrokontroller, IoT maksesüsteem, Mikrotransaktsioonid, Kohvipruulimise Nišš.

CERCS: P175 (Informaatika, süsteemiteooria)

1. Introduction

The fusion of the Internet of Things (IoT) and blockchain technology holds the potential to revolutionise traditional transaction processes. IoT enhances the quality of environmental interactions through precision and efficiency, while blockchain offers secure, precise, and minimal-fee transactions, marking a significant shift in financial operations. The combination of IoT and blockchain paves the way for efficient, cost-effective microtransactions, transforming traditional payment systems.

However, several challenges persist. Blockchain transaction speeds are typically slow, and faster alternatives lack proven reliability. Moreover, consensus protocols face a 'trilemma', struggling to balance scalability, security, and privacy [1]. This introduces an additional layer of complexity when integrating these protocols with IoT systems. Additionally, the lack of standardisation in IoT technologies poses a significant barrier to widespread adoption.

This study aims to investigate the practicality of establishing an IoT-based payment system. Specifically, it explores:

1. The advantages and challenges of integrating the XRP Ledger with the ESP8266 microcontroller in an IoT-based payment system.

2. The user interface and user experience in an IoT-based payment system.

Understanding the potential costs of setting up an IoT payment system, determining the fee structure of the XRP Ledger [2], and assessing the user-friendliness of these systems is critical. This research contends that combining the XRP Ledger and the ESP8266-based microcontroller in an IoT-based payment system, particularly within the university-based coffee brewing niche, could offer lower transaction fees for microtransactions than traditional payment styles.

The study adopts a multifaceted methodology, encompassing a literature review, system architecture design, practical implementation, conclusion and future work. Additionally, the methodology integrates two advanced tools to enhance the research quality. Grammarly [3] is used to fine-tune the linguistic presentation, ensuring clarity and grammatical accuracy. Concurrently, ChatGPT, an advanced language model developed by OpenAI [4], assists in data search and text refinement.

The thesis is structured as follows:

1. Background and Motivation: Provides an overview of the current state of IoT and blockchain technology, emphasising the potential benefits of their integration.

2. Literature Review: Offers a thorough analysis of the existing research on IoT-based payment systems, blockchain technology, the XRP Ledger, the ESP8266 microcontroller, and other relevant topics.
3. System Architecture and Implementation: Details the design and implementation of the proposed IoT-based payment system, focusing on hardware, software, and user interface aspects.
4. Conclusion and Future Work: Summary of the study's findings and provides recommendations for future research.

This study will showcase the feasibility and potential of integrating IoT and blockchain technologies by creating a more efficient and cost-effective transactional system.

1.1 Background and motivation

Micropayments for IoT devices have emerged as a growing field, potentially enabling seamless transactions between devices and their users. Various solutions have been proposed for IoT payments.

The XRP Ledger, a decentralised digital ledger, has the potential to provide the foundation for a convenient and efficient payment system. The XRP Ledger claims to provide a solution for cross-currency payments by enabling transactions through its decentralised exchange, allowing direct currency conversions without intermediaries. Additionally, the platform purports to eliminate fees typically associated with traditional transactions. This motivates the exploration of the XRP Ledger as a potential platform for facilitating IoT payments.

As coffee enthusiasts, the author's supervisor and the author value the taste and quality of freshly brewed espresso. High-quality coffee beans are often expensive, and espresso machines can be complex. With this research, an easy solution is sought that would enable the consumption of good coffee in a research lab setting. A system is envisioned where the espresso machine pays for all its costs, including the coffee beans, electricity, and maintenance, without generating profit. This concept is inspired by existing IoT solutions, such as Christmas lights that can be turned on by users paying the electricity bill directly.

This research aims to develop IoT payment solutions, assess the XRP Ledger's potential, and improve the coffee experience for university professors and students. Research in this domain can create an environment where an ample supply of espressos fuels standardisation in IoT.

1.2 Problem statement

This chapter aims to provide a clear and concise description of the main problems this thesis intends to address. The research focuses on three main issues: the Internet of Things, XRP Ledger, and the coffee preferences of university professors and students. The problems discussed in this chapter are the basis for the research questions and objectives outlined in the following chapters.

The rapid advancements of IoT have led to a fragmented ecosystem due to a lack of unified standards. For instance, multiple solutions have been proposed for IoT payments. However, a suitable solution for IoT microtransactions has yet to be established. XRP Ledger claims to have everything necessary for IoT microtransactions. A thorough investigation of the validity of these claims needs to be conducted to determine whether the XRP Ledger is as effective and beneficial for IoT microtransactions as it claims to be.

High-quality coffee is characterised by beans' freshness, proper brewing techniques, and appropriate water temperature. No satisfactory solution exists on the university campus for easy access to good-quality espresso. Some makeshift solutions involve using a standard coffee machine to produce two espresso shots, but these attempts yield suboptimal results.

Existing solutions using the XRP Ledger for purchasing items from automatic vending machines are not suitable for our use case, as our focus is on a more sophisticated espresso machine with a steeper learning curve for users. In particular, a suitable solution for IoT micropayments (think paying for your coffee is as simple as paying with coins at a vending machine but using some electronic means instead of copper) has not yet been established.

This thesis aims to address the three main problems explained: How can we navigate the fragmented IoT ecosystem to implement a reliable microtransaction solution successfully? To what extent are the XRP Ledger's cross-currency payments and fee elimination claims valid and applicable in an IoT-based payment system? Given the high demand for quality coffee within a university setting, how can an IoT-based payment system be effectively designed and implemented to meet this need?

2. Literature Review

This literature review is aimed to provide a comprehensive overview of the key concepts, technologies, and good practices of building an IoT-based payment solution for coffee.

The literature review is organised into the following sections:

1. Best Practices for IoT Systems: In this section, we discuss the key principles and guidelines for developing, deploying, and maintaining robust, secure, and scalable IoT systems.
2. IoT and Microcontroller-Based Payment Systems: This section categorises different payment types. Additionally, it goes over the importance of researching the area.
3. Blockchain for IoT: This section discusses cryptocurrency's current role in IoT. The importance of cryptocurrencies for micropayments comes into play.
4. The XRP Ledger: This chapter is aimed to give a thorough explanation of what the XRP Ledger is, how it works, and how it can be utilised for IoT.
5. The ESP8266 Microcontroller: This section provides a detailed overview of the ESP8266 microcontroller, its features, and its suitability for IoT applications, specifically in coffee machine payment solutions.
6. IoT and Micropayments: This section delves into the challenges and considerations associated with combining IoT and micropayments, focusing on hardware limitations, programming language performance, and using payment channels to overcome transaction caps.
7. Existing Coffee Machine Payment Solutions in the University: This section outlines the current payment solutions available for accessing coffee on campus, catering to the diverse preferences and requirements of students, staff, and guests.

The insights gained from this literature review will lay the groundwork for designing an effective IoT-based payment solution for buying coffee.

2.1 IoT development and good practice

As the Internet of Things (IoT) continues to expand, good practices must be followed to ensure IoT applications' robustness, security, and scalability. This chapter will discuss essential aspects of IoT development and provide guidelines for implementing good practices in various stages of the development process. The main knowledge of this chapter was acquired from IoTempire [5].

In the planning and design phase, start with clear specific goals and set some functional requirements. This ensures that the focus stays on the end goal, and if the development process were to halt, a clear vision of the end goal helps to clear the mind. Next, the right hardware and software should be chosen. This includes selecting the appropriate microcontrollers, sensors, and communication protocols while considering power consumption, processing capabilities, and compatibility with existing systems. The overall vision should have scalability and modularity built into the equation.

During development and implementation, open-source protocols can be utilised. This enhances interoperability and reduces the risk of vendor lock-in. Robust security measures should be implemented. Since IoT devices exchange sensitive data, strong authentication, encryption, and access control mechanisms must be in place to prevent unauthorised access. In this process, energy efficiency needs to be kept in mind. Testing and validation should be done throughout the development process to identify and address potential issues early on.

Over-the-air (OTA) updates simplify the maintenance tasks when deploying and maintaining the system. Collecting and analysing data from IoT devices help identify trends, detect anomalies, and optimise system performance. Decommissioning devices should be done securely and responsibly when they end their useful life. Good and detailed documentation about hardware specifications, software architecture, and some user/development guides should be included.

According to ChatGPT[1], good practices presented above can be achieved using tools or platforms such as Arduino, Raspberry Pi, PlatformIO, Node-RED, MQTT, FreeRTOS, Zephyr Project, AWS IoT Core, Microsoft Azure IoT, and Google Cloud IoT. In the implementation part of this thesis, IoTempower is used as it covers almost all of the good practices explained and provides support for or functionality like many of the tools listed.

By adhering to these good practices, secure, scalable, and robust systems can be built. The successful implementation of IoT solutions requires careful planning, ongoing testing and validation, and a commitment to continuous improvement. The next chapter will go over how IoT is used in payments today.

2.2 IoT and Microcontroller-Based Payment Systems

The Internet of Things (IoT) has revolutionised how payments are made by offering various methods, each tailored to accommodate different transaction needs[6][2]. This chapter provides an in-depth analysis of these diverse payment systems, focusing on their unique characteristics and the technology that supports them.

Traditional payment methods, such as cash, credit cards, and debit cards, remain popular among the older generations. These methods require either counting cash or reading card information.

[1] The list of IoT tools and platforms was prompted from ChatGPT (May 7, 2023), i.e., a language model that is trained on a large number of different text sources. ChatGPT has been developed by OpenAI. For more information about ChatGPT and OpenAI, visit: https://openai.com. Input submitted to the model: "List some popular tools, platforms, and technologies used for IoT development and communication."
[2] https://www.mastercard.com/news/media/wddjfrhn/how-iot-will-shape-the-future-of-payments.pdf

Near Field Communication (NFC) technology enables contactless payments for smaller transactions using cards and is fairly popular among people aware that they have NFC on their phones.[7] This convenience eliminates the need for users to enter a Personal Identification Number (PIN) and is gaining popularity among younger people [8].

With the widespread adoption of smartphones,[9] mobile payments have become increasingly popular. Banks have developed apps to facilitate mobile payments through NFC or peer-to-peer (P2P) systems. P2P services, such as Venmo[3], Zelle and Cash App, are designed for social purposes, allowing users to split bills, send emojis, and more.

Digital wallets, including Google Pay[4], Apple Pay[5], and Samsung Pay[6], can also be used for P2P transactions but primarily focus on broader use cases. They securely store users' bank information, allowing for quick contactless payments. Wearable technology, such as smartwatches, further expands the possibilities for mobile payments [10]. To enhance security, mobile payment apps typically require biometric authentication for unlocking [11].

Cryptocurrencies offer another payment option for small and large transactions that can be executed without intermediaries. Three general types of wallets can be used for storing the keys to cryptocurrency accounts: software, hardware, and web wallets [12]. In an IoT setting, when paying for a coffee, a mobile wallet can be used to scan a QR code, initiating a transaction [13]. The authentication process for smartphone cryptocurrency software wallets mirrors traditional digital wallets.

In conclusion, IoT and microcontroller-based payment systems have greatly diversified the options for processing transactions. From traditional methods to mobile payments and cryptocurrencies, each system offers unique advantages and requires specific supporting technology. This work looks at microtransactions for IoT-based cryptocurrency settings, as cryptocurrencies enable the execution of truly small transactions that the fees of other payment types would usually eat up.

2.3 Blockchain for IoT

Blockchain was developed to transact between untrusted parties without a central bank [14]. The very first cryptocurrencies did not focus on scalability and performance. Mercan, Kurt, Akkaya, and Erdin [15] looked at cryptocurrency solutions to enable micropayments in consumer IoT. Early blockchain designs demand heavy resources and is not very scalable, meaning Transactions per second are low, and they need a powerful machine. Newer blockchain designs have a semi-centralised scheme which uses fewer resources and has much better scalability meaning they can be used in microcontrollers. Many of these new designs have not had time to

[3] https://venmo.com/
[4] https://pay.google/about/
[5] https://www.apple.com/apple-pay/
[6] https://www.samsung.com/us/samsung-pay/

prove themselves trustworthy. A semi-centralized structure is less secure and has some privacy concerns. The interest in cryptocurrency from institutions and governments and the wide adoption of IoT will accelerate IoT and cryptocurrency merging efforts.

There are many reasons why blockchain for IoT is needed. Zhou, Song, Liu, and Niu [16] looked at blockchain integration for smart product-service systems (SPSS) and explained why SPSS is useful. SPSS helps to collect a lot of data about product use. SPSS has problems with data loss and leakage, but blockchain can help deal with the issues. Blockchain can secure data efficiently. SPSS is needed to improve the performance and efficiency of seamless information flow between stakeholders. It helps monitor, diagnose and maintain data while knowing it is unchanged.

One contemporary blockchain realisation that shows promise in IoT is XRP Ledger. XRP Ledger is designed to have high throughput, low transaction fees and fast validation times. The next chapter will cover The XRP Ledger in detail.

2.4 The XRP Ledger

The information presented regarding the XRP Ledger has been primarily sourced from the official XRP Ledger website [2]. The purpose of relying on a single source is to evaluate the comprehensiveness and quality of the documentation provided by the developers and to explore the feasibility of learning and understanding the XRP Ledger solely through their official website. This approach is essential in the context of our study, as we aim to assess the practicality of building a system for micropayments using the XRP Ledger. While the implementation and application of the XRP Ledger will be discussed in subsequent chapters, it is essential to acknowledge that the foundation of our understanding is based on the resources available on the XRP Ledger website. By doing so, we ensure transparency and give due credit to the original source of information.

The XRP Ledger (XRPL) is a decentralised, public blockchain [17] designed to facilitate fast, secure [18], and low-cost transactions [19] of various assets, including cryptocurrencies and fiat currencies. The core function of the XRP Ledger is to enable efficient cross-border payments, remittances, and currency exchanges.

Ripple [20] is the company that played a huge part in the development of XRP Ledger's open-source codebase [21]. Ripple maintains a significant role in the XRP Ledger but does not solely control the decentralised network. Ripple collaborates with other organisations like the XRP Ledger Foundation [22] and independent developers to improve the XRP Ledger.

The XRP Ledger utilises a consensus protocol [18], an agreement mechanism that allows designated servers called validators to reach a consensus on the order and outcome of transactions within the network. This consensus process occurs approximately every 3-5 seconds. To achieve consensus, at least 80% of the network's validators must agree on a specific transaction or set of transactions. Validators are chosen based on their trustworthiness and reliability.

The XRP Ledger can tokenise various assets, such as USD, EUR, and other currencies. Tokenisation enables the representation of these assets on the blockchain. XRP, the native cryptocurrency of the XRP Ledger, serves as a bridge currency and is used on the XRP decentralised exchange where the tokens are exchanged.

Changes on the ledger happen through transactions, which modify the ledger by following these steps: create, sign, submit, reach consensus, apply, and validate. Each transaction has a small cost that is *burned*, meaning the XRP is destroyed and removed from circulation. Transaction costs safeguard against denial-of-service attacks. The minimum transaction fee is ten drops (0.00001 XRP). At the time of writing, an offer on the decentralised exchange lists 1 XRP for 0.420 EUR [23]. All accounts on the XRP Ledger must maintain a minimum balance of 10 XRP, which acts as a *reserve* also to prevent denial-of-service.

XRP Ledger has seven transaction types: direct payments, cross-currency payments, checks, escrow, partial payments, and payment channels.

- **Direct** point-to-point payments are the simplest transaction on the XRP Ledger. They involve transferring XRP or other issued currencies between two accounts directly.
- **Cross-currency** payments function similarly to direct payments but involve the exchange of more than one currency. This feature allows users to send a payment in one currency and have the recipient receive it in another, with the XRP Ledger handling the conversion process.
- **Checks** allow a payer to create a check specifying the amount, recipient, and other conditions, such as an expiration date. The recipient can review the check and decide whether to accept or reject the payment.
- **Escrow** payments on the XRP Ledger are conditional transactions. The XRP Ledger executes escrow payments after meeting certain predefined conditions.
- **Partial** payments enable users to send a portion of the total payment amount, with the remaining balance in an escrow waiting for further action.
- **Payment channels** allow parties off-chain payments without fees, with transactions settled on the XRP Ledger only when the channel is closed.

The implementation part of this thesis discusses how these different types of payments could be used to pay for a coffee. However, the implementation will not cover all seven payment types, only direct and Cross-Currency payments.

2.4.1 XRP Ledger Tokens

The XRP Ledger facilitates the representation of assets as digital tokens, providing a framework for managing and exchanging value. The XRP Ledger can be used for device-to-device payments, asset tracking, and access control in the IoT ecosystem by allowing users to create tokens. This chapter will discuss the nature of assets on the XRP Ledger, the characteristics of fungible and non-fungible tokens, and the use and mechanisms for creating and managing trust lines and tokens.

Building on this foundation, the XRP Ledger supports two types of tokens: fungible and non-fungible. Fungible tokens are interchangeable and can be exchanged for one another, while non-fungible tokens are unique and not interchangeable. Any account that meets the reserve requirements (for further explanation, see two paragraphs below) of an account can create both fungible and non-fungible tokens. Fungible tokens are commonly used for representing currencies, commodities, and other assets with equivalent value. Non-fungible tokens (NFTs), on the other hand, represent unique digital or physical assets, such as collectables, artwork, and digital identities. NFTs can have various settings and attributes that the issuer can customise.

Tokens on the XRP Ledger move along trust lines, which connect accounts and define the issuer of a token. An account holding a token will have a positive amount of that token, and the issuer will have a negative amount. Trust lines can be used to set limits, disable rippling (see the next chapter), set authorisation (only authorised users can hold the tokens, often used for issuing stablecoins), set fees (fees can be set for trades between accounts and trades between the issuer), and freeze accounts (with options for global freeze, individual freeze, or no freeze).

To prevent spam, each account must have 10 XRP in reserve and an additional 2 XRP for each trust line created. The first two trust lines are considered "free". To use NFTs, each buy offer, sell offer, and each NFT page that can store up to 32 NFTs requires 2 XRP in the account reserve. NFTs can be minted before or during a sale. For example, to sell one token, an account must have at least 14 XRP on its account. The account needs 10 XRP to meet the default requirements, 2 XRP to create a trust line and 2 for a sell offer.

In summary, XRP Ledger's fungible and non-fungible tokens have different applications and unique attributes. Trust lines facilitate token movement and enable various settings. Each account has reserve requirements for accounts, trust lines, and token usage. The next chapter will discuss the decentralised exchange, built on these previous concepts.

2.4.2 Decentralised exchange

Moving to the decentralised exchange, users can trade tokens on the decentralised exchange (DEX). Trades between accounts are tracked as currency pairs. An account can trade tokens for XRP, and tokens can be traded for tokens. Trades can be made through Offers. Trade occurs when a created Offer “consumes” a matching Offer with the same currency pair. The networks will find an Offer with the best exchange rate. Token issuers can give the transactions a fee for trading and issuing. Trading on the DEX will burn the normal transaction cost (minimum of 10 drops). Offers can be partially consumed, meaning an Offer can automatically split into parts if only part of it is consumed.

Trades can *ripple*, meaning that a trade might not take the straightforward path to another account but instead go through another route. Rippling occurs when the transaction cost is lower through that path. By default, each trust line has rippling enabled, allowing tokens to move between accounts that are not directly connected. If two accounts do not have a direct path, tokens can ripple through other accounts, leading to changes in the balances of the accounts. If using XRP as an intermediary currency would be cheaper than trading the two tokens directly, auto-bridging occurs, meaning the token is used to buy XRP, and then XRP is used to buy the desired token.

In conclusion, the XRP Ledger provides a platform for managing digital tokens. Trust lines, reserve requirements, rippling and auto-bridging ensure the movement of tokens, while the whole of decentralised exchange enables the trading. Next, we will discuss integrating IoT devices, such as the ESP8266 microcontroller, into this virtual ecosystem, bridging the gap between the physical and digital worlds.

2.5 The ESP8266 Microcontroller

Bridging the virtual and physical worlds has become increasingly important in the IoT space, with microcontrollers like the ESP8266 and ESP32 becoming the standard for connecting various sensors and actuators [24]–[26]. Affordable and versatile devices, such as the Sonoff Basic [27] and several IKEA and Obi products, have been built using these microcontrollers. The Amazon Dash Button, for example, was based on the ESP8266. These microcontrollers have gained popularity due to their low cost, ranging from 1-2 EUR, in contrast to alternatives like the Raspberry Pi, which has become 5-10 times more expensive over the past few years.

The ESP8266 [28] microcontroller is built for the world of the IoT [29]–[31]. Espressif Systems developed it, and it has a compact design, built-in Wi-Fi capability, and low power consumption. It can be used in a wide range of IoT applications. This chapter provides a detailed overview of the ESP8266's features, capabilities, and use cases.

The ESP8266 is equipped with a single-core Tensilica Xtensa L106 32-bit RISC processor [32], which provides the necessary computing power for basic IoT applications. The microcontroller features a 2.4 GHz Wi-Fi transceiver compatible with the 802.11 b/g/n standards and supports WPA and WPA2 encryption protocols. The ESP8266 has 17 configurable GPIO pins that enable communication with external devices. The microcontroller offers different operating modes, including active mode, modem/light-sleep mode, and deep sleep mode, allowing for options for power management in different scenarios. The ESP8266 has internal SRAM and up to 16 MB of SPI flash memory. The microcontroller supports over-the-air updates.[7]

Arduino IDE [33] or PlatformIO [34] are common options for programming the ESP8266. Both platforms support a range of ESP8266 boards, such as Wemos D1 Mini [35], NodeMCU [36], and Adafruit HUZZAH [37]. In addition to the standard Arduino programming language, the ESP8266 also supports MicroPython, a Python 3 implementation, and Lua, specifically for the NodeMCU board.

The ESP8266's affordability, compact design, and energy efficiency make it an ideal candidate for IoT projects. Furthermore, its support for WPA2 encryption ensures secure communication between devices. Although, in our case, the microcontroller cannot sign XRP Ledger transactions directly, it can utilise encryption keys for secure data transmission [29]–[31].

Some common applications of the ESP8266 microcontroller include environmental monitoring, asset tracking, and real-time data processing [38], [39]. With its GPIO pins and Wi-Fi, the ESP8266 can connect to various sensors and transmit data to other devices. This also has the potential to process microtransactions efficiently. As a result, the ESP8266 is a well-suited microcontroller for IoT applications. The next chapter will go over the limitations of microcontrollers.

2.6 Limitations of Microcontrollers

The drawbacks must be considered when using microcontrollers. One has to take into account the hardware limitations of the microcontrollers. Microcontrollers typically have smaller processors and less memory to use. This puts many constraints on the systems that can be created.

The performance of a programming language does not play a significant role when using a powerful computer, but microcontrollers have limited capabilities. Choosing the right programming language for a microcontroller is as important as selecting the microcontroller itself. Ripple[8] has client libraries for many different programming languages, and the XRP Ledger webpage lists the following: Python, JavaScript/TypeScript, C++, Java, Ruby and HTTP API-s [1]. The ESP32[9] and ESP8266[10] boards have support for Python and C++ only.

Although easy to read and write, Python code can have orders of magnitude slower performance than C. This was proven in a paper by Ionescu and Enescu [40]. They investigated the performance of the two programming languages and compared them to the ESP32 and the

[7] https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf and https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

[8] https://ripple.com/xrp/

[9] https://en.wikipedia.org/wiki/ESP32

[10] https://en.wikipedia.org/wiki/ESP8266

STM32. Three different algorithms were run on both boards twice - once in MicroPython and once in C. The performance level of MicroPython was lower than that of C.

Moreover, developing drivers for MicroPython has proven problematic, whereas importing an Arduino library is relatively easy. As mentioned in the IoTempower documentation explaining the switch from MicroPython to C++ [41], “Earlier versions were based on MicroPython, however, porting some of the C++-based Arduino device driver libraries, managing remote access, updates, dealing with very little memory, and a slightly defunct community, made management very hard leading us to the decision to switch to an admittedly harder to program environment, however, we earned the access to the huge and active Arduino community making problem solving and extensions much easier. We do not regret the switch.”

The capabilities of the cryptocurrency used also have a significant role in the system's performance. The performance of XRPL was tested in a paper by Han, Gramoli and Xu [42], who evaluated blockchains for IoT. The number of successful transactions per second is capped in the case of XRPL. XRPL can handle multiple clients, but as the number of clients increases, the number of successful transactions drops significantly. They also found that as the number of transactions starts reaching the maximum of 1500 transactions per second, the number of successful transactions decreases sharply and plummets to zero.

Payment channels can be used to overcome the ledger's transaction cap. Many small claims can be signed off-chain with the payment channels and later validated with a single transaction. The number of claims is not capped. XRP ledger has 2 options for signing signatures: the default secp256k1 ECDSA signatures and the Ed25519 signatures. The Ed25519 signatures are more efficient and secure [43]. A seminar paper by Bernstein et al. [44] demonstrates that 109,000 Ed25519 signatures can be created per second and 71,000 Ed25519 signatures can be verified per second while keeping the maximum verification latency below 4 milliseconds. Many cryptocurrencies already have solutions for off-chain payment channels like Bitcoin Lightning[11], Ethereum Raiden[12] and Stellar[13].

To reach the transaction cap, the system would have to be big. As IoT systems grow, it becomes increasingly important to follow good practices in developing and deploying these systems. That was discussed in chapter 2.1. In the current implementation using ESP8266, payment channels are not used, but they are worth considering for future work to overcome the transaction cap of the ledger. The next chapter will act as a smooth transition to the implementation part.

[11] https://github.com/lightning/bolts
[12] https://raiden.network/
[13] https://www.stellar.org/

2.7 Existing Coffee Machine Payment Solutions in the University

The university offers three distinct payment solutions for accessing coffee on campus. These systems cater to varying preferences and requirements among students, staff, and guests. The following sections outline the features of each solution in detail.

The university provides designated kitchen areas with the necessary tools for individuals who prefer to bring their coffee grounds and prepare pour-over coffee. These kitchens feature a water kettle, sink, dishwasher, and various mugs. This option allows for a customised coffee experience, with users having control over the choice of coffee and brewing method.

Located on the student side of the building, an automatic coffee machine offers a convenient option for those seeking a quick and standard cup of coffee. This machine dispenses coffee in paper cups and requires users to pay X euros per cup. The selection of coffee types available is standard, making this solution suitable for guests unfamiliar with the building and students and staff who have money and time to spare and prefer a standard coffee.

UT Institute of Computer Science (ICS) also provides exclusive access to free coffee for staff and PhD students. These individuals can access three coffee machines on the third floor, which are off-limits to average students. These machines are known for their quality coffee output, with one even producing semi-decent espressos. It is worth noting that this is the biased opinion of the author and the supervisor. For a touch of humour, we refer to the old joke: "A programmer is a machine that turns coffee into code." Perhaps let us add: "A good programmer is a machine that turns good coffee into good code."

The university has implemented three coffee payment solutions to accommodate its community's diverse needs and preferences. These options range from self-service pour-over coffee to exclusive access for ICS staff members at Delta, ensuring everyone can find a solution that best suits their requirements.

3. System Architecture and Implementation

Drinking coffee is an essential aspect of human culture, providing a source of energy and a means of connecting with others. As people's tastes evolve, they often crave a variety of coffee flavours, roasts, and brewing methods. In pursuit of the perfect cup, many invest in high-quality coffee equipment and premium beans. The aim is to bring premium coffee beans to a university setting. However, this also makes coffee more susceptible to theft, creating a need for a secure and efficient payment system.

A coffee station integrated with a cryptocurrency-based payment system is proposed in response to this challenge. This innovative solution utilises the XRP Ledger, a decentralised digital ledger, to manage transactions through stablecoins. *Stablecoins* are digital tokens pegged to a stable asset, such as a fiat currency, which helps mitigate the risk associated with the volatile nature of cryptocurrencies. In this system, the coffee station accepts payments in the form of stablecoins, ensuring a consistent and secure value for each transaction.

The coffee station is designed to provide a high-quality coffee experience. It uses an electrical burr grinder [45] to process locally roasted beans [46], offering users a fresh and personalised beverage. Upon receiving payment, the system activates the coffee grinder and measures the ground coffee beans. Once the desired amount is reached, the grinder automatically turns off. Additionally, a kettle is activated when the payment is received, further streamlining the coffee-making process.

The following chapters will discuss the intricacies of the system architecture and its implementation, including the integration of the XRP Ledger, IoT components, and user experience. This innovative solution combines cutting-edge technology with the timeless pleasure of enjoying a perfect cup of coffee. An overview of the initial experimentations with the XRP Ledger will be provided first.

3.1 WebSocket transaction monitor on the ESP8266

This chapter will discuss the initial experimentation with the XRP Ledger, focusing on developing a WebSocket transaction monitor using a Wemos D1 Mini [35] ESP8266 microcontroller. A transaction monitor with a little OLED screen running on the ESP8266 can be implemented by following the basic tutorials on the XRP ledgers website and using ChatGPT. The tutorials on the website do not explain how this can be done on an ESP8266, and there is little support for using C++ with the XRP Ledger. Other challenges while creating this kind of device include finding libraries for an OLED display, parsing JSON data, using a WebSocket client and extending the device to a relay for controlling electronic devices. This device does not follow the IoT best practice but is a good starting project. The can be found on GitHub. [47], [48] The XRP ledger website [2] has a detailed tutorial that can be followed while learning about the ledger. The tutorials are in Python, Java, Javascript and many other languages. The tutorials cover the first transactions and other basic XRP ledger things. The tutorials cover almost everything that the XRP ledger has to offer. Following the tutorials, the first system can be built to send transactions. The first transaction tutorial helps create a Testnet account with 1000 XRP on the test account, ready to be used. Following the WebSocket tutorial for subscribing to an account, a separate program can be created to monitor the incoming payments of an account.

The code given in the tutorials on the XRP ledgers website cannot be used on an ESP8266 out of the box. The C++ rippled signing library has no tutorials on how to use them. Although Python code can be run on an ESP8266, MicroPython is stuck on Python 3.4, which is incompatible with the XRPL library. The code for creating a WebSocket client and monitoring incoming payments can be translated into C++ using ChatGPT 3.5. At the time doing this, the model translated the code into C++ easily, and about 6 prompts were needed for the translation process. Some extra knowledge about the libraries needed can be gained by looking at the code samples on the internet on how WebSocket clients run on the Arduino and how JSON files can be parsed. The libraries used were not unified, and each prompt gives a different library for creating a WebSocket client and parsing JSON. The model hallucinates trying to use code from different libraries and the wrong libraries for different parts of the code. With the help of PlatformIO on VS code, the libraries can be sorted out to have code that compiles.

PlatformIO supports many boards and has libraries specific to the ESP8266. PlatformIO has a library manager and builds tools. A library for a 128x64 OLED screen exists, and there is a modification of the library "*ESP8266 and ESP32 OLED driver for SSD1306 displays*" by ThingPulse [49] to also support 64 x 48 screens. The library still thinks the smaller screen is 128x64, which must be counted for in the code. Furthermore, the libraries "*ArduinoJson*" by Benoit Blanchon [50] and "*WebSockets*" by Markus Sattler [51] are used to handle the WebSocket client and the JSON parsing. The code can be compiled and deployed straight onto the ESP8266 via a USB cable.

When the program starts, it connects to WiFi. It then creates a WebSocket connection to the XRP Testnet server. The program subscribes to a specific XRP ledger account, and upon receiving a successful transaction, it changes the amount for the newxrp variable in the code. The code updates the screen if the value of the newxrp variable has changed, and with the use of a state machine, it turns on a relay for 6 seconds. The relay part will be discussed further down.

Once flashed onto the ESP8266, the microcontroller needs only a power supply. To see the serial monitor while the device is running, the ESP8266 should be connected to the computer. The serial monitor can be viewed in the terminal built into VS code. If the code is correct, the device will show the amount of XRP received on the little OLED display shield when a transaction is received.

The device built can only monitor incoming transactions. It is not meant for creating and signing transactions. Transactions can be sent using the code and tutorials on the XRP ledgers website. Using Python code to create a TestNet account, an account address can be entered into the code and flashed onto the microcontroller. Another TestNet account can be used to send small payments to the account in a while loop. The OLED display will show the amount received. Each transaction takes a few seconds to process and validate on the ledger. The device will show the XRP received instantly, and there is no delay except the one that validates the transaction.

Replacing the OLED shield with a relay shield will allow the device to turn on and off electricity going through the wire connected to the relay. The relay shield is built to only receive a signal from the GPIO pin D1. This is important to know since if other shields will be added the pins used may overlap. To turn the relay on for a set time, a state machine has to be added to the code using a variable of the time passed. A *tripler* shield can then connect both the relay and the OLED shield to the board. The device now displays the amount of XRP received and also can use a relay.

In summary, this chapter went over building a WebSocket transaction monitor with the ESP8266 and an OLED screen by following the tutorial on the XRP Ledgers website and using ChatGPT at its early stages to translate the code into C++. The translated code was fixed by choosing the right libraries for the OLED display, WebSocket client, and JSON parsing. Platform IO and VS code helped in the process of managing the libraries and correcting the code. The device's functionality was further extended with a relay to turn on and off electrical devices. Overall this device demonstrates how with little knowledge of IoT, a transaction monitor can be created and what hardware and software challenges must be overcome in the process.

3.2 Hardware components

The IoT system presented in this project consists of sensors, microcontrollers, and a central gateway following the IoTempower default architecture [52] to create a reliable payment system for coffee. By leveraging an Wemos D1 Mini [35] ESP8266 microcontroller for communication, a Sonoff Basic for power control, and a Raspberry Pi as the central gateway, the system enables control over the devices based on what is happening live on the XRP Ledger. This solution could be tailored to various applications, but it is good to start by making coffee.

Here is what the overall component setup looks like.

1. Set up the weight sensor with an HX711.

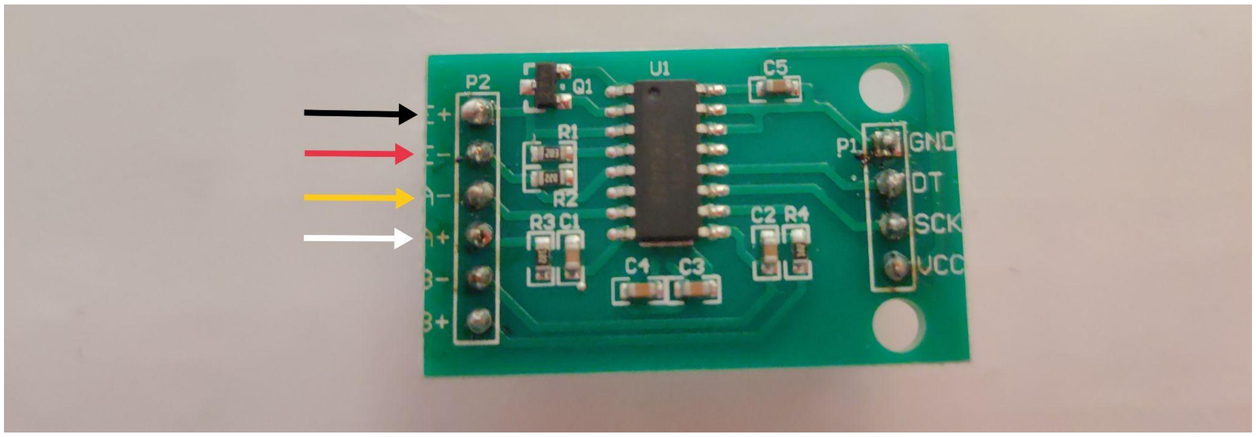

Figure 1. HX711 setup.

Sensors collect environmental data, and the weight sensor uses a metal part connected to four wires: black, white, red, and green. These wires connect to an HX711 microcontroller (Fig 1). The yellow wire goes to A-, the white wire to A+, the red wire to E-, and the black wire to E+. The HX711 then connects to an ESP8266: GND to GND, DT and SCK to GPIO pins and VCC to 3V. (Fig 2)

2. Wire the HX711 to an ESP8266.

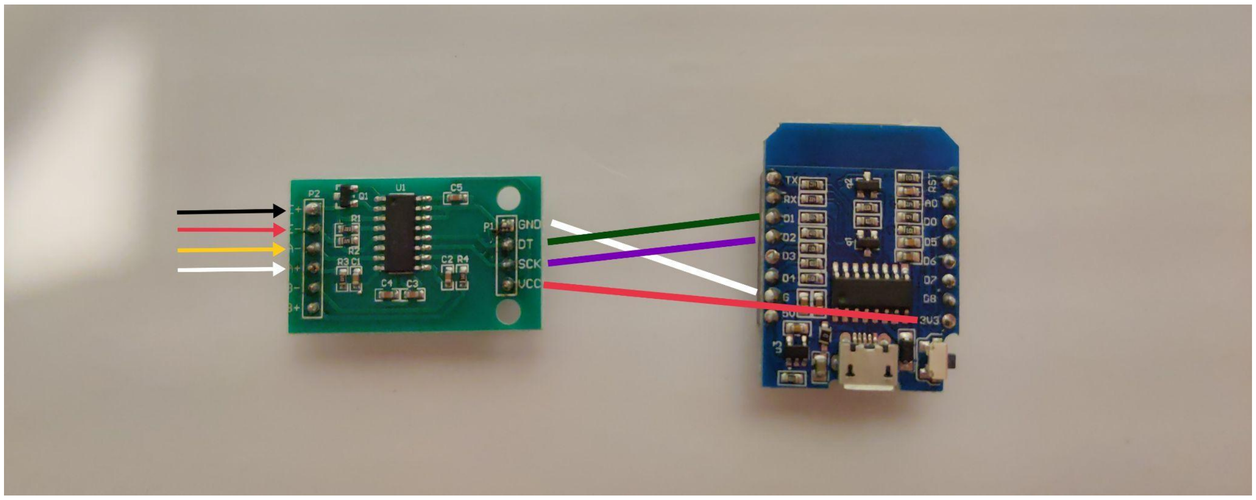

Figure 2. HX711 connection to ESP8266.

The ESP8266's GPIO pins are configured via software to send data to the correct pins. It connects wirelessly to a gateway. Since the sensors produce non-human-readable values, the ESP8266 is a translator, sending data over Wi-Fi to the gateway in a human-readable format. A USB cable powers the ESP8266.

3. Set up the Sonoff Basic with a power cable.

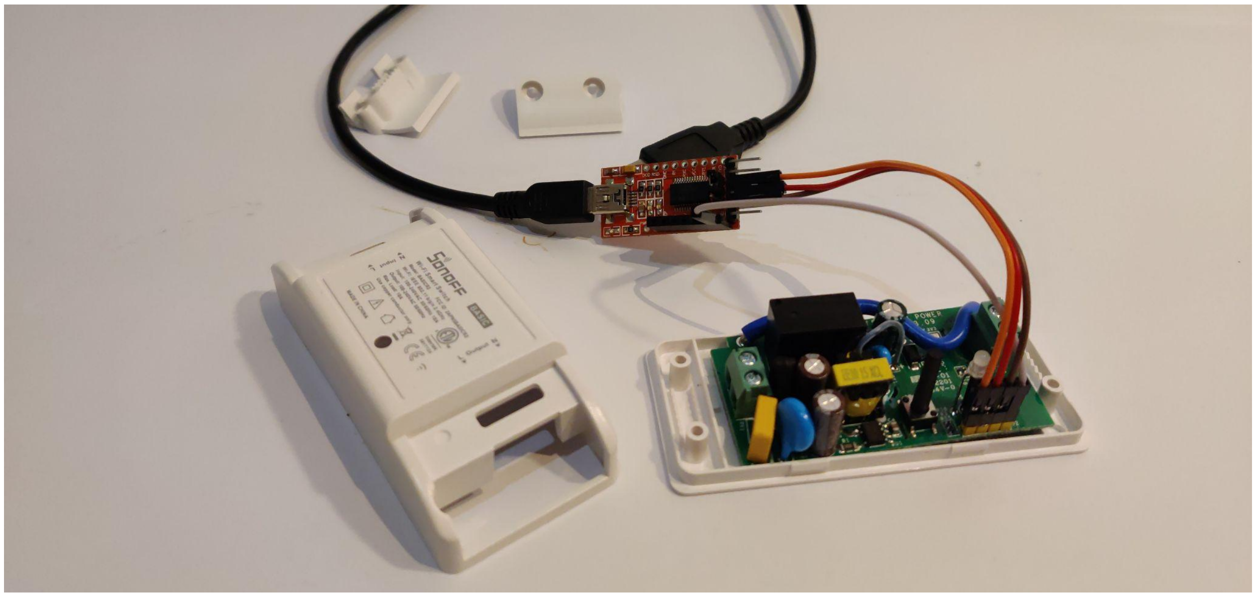
Figure 3. Sonoff Basic Pin Headers.

A Sonoff Basic controls the power supply, functioning as a wireless relay for switching power on and off. The Sonoff basic has an embedded ESP8266 chip that interprets signals to control the relay. This device draws power directly from the cable connected to it. To use it in an IoT setting, it must be flashed with firmware compatible with IoTempower. We need to use a YP-05 or any other USB-to-serial adapter for flashing. The Sonoff Basics pins must be connected: GND to GND, TX to RX, RX to TX, and 3.3V to VCC. A pin header should be soldered onto the Sonoff Basic for this process.

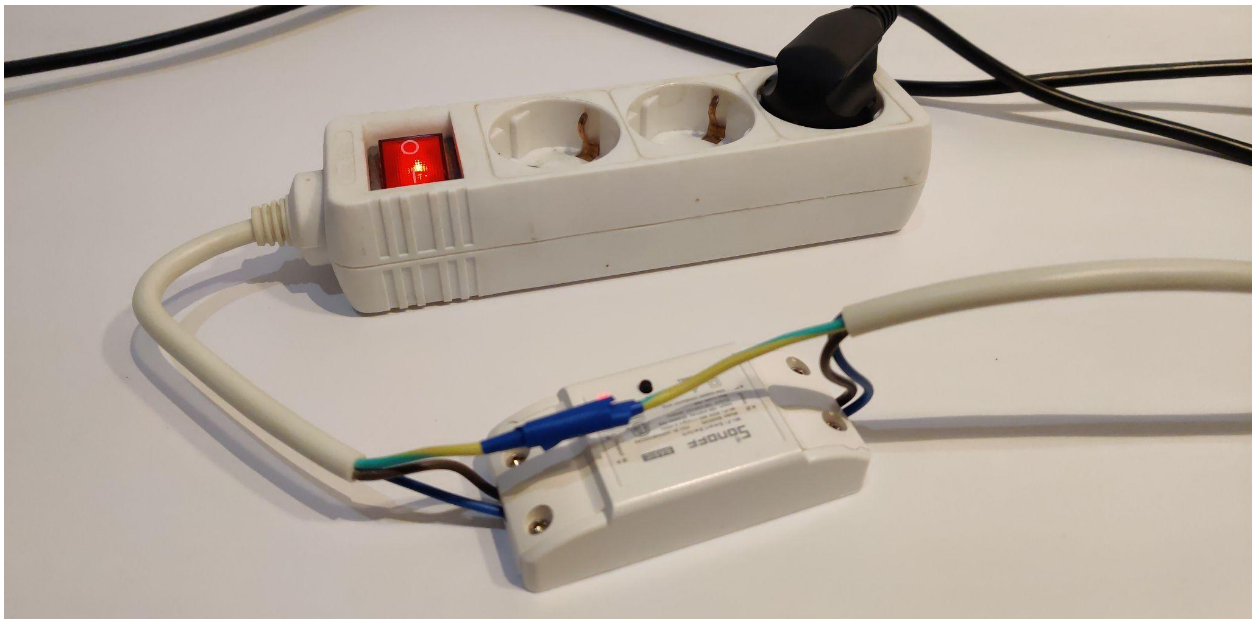
Figure 4. Sonoff Basic.

The Sonoff Basic can now be wired up with a power cable. When connecting the Sonoff to a power cable, there is one important step. The Sonoff doesn't have a dedicated ground wire, so if the ground wire exists, then it needs to be routed outside the case or fit inside.

4. Power the devices.

A Raspberry Pi is a central gateway, providing connectivity between various devices. It is equipped with multiple USB ports for easy flashing and can be configured to connect to a router using Wi-Fi, Ethernet, or USB tethering with a phone. Using a Raspberry Pi as a gateway ensures that all connected devices maintain the same IP address when changing locations. The software for the Raspberry Pi is stored on an SD card.

5. Set up the software.

The next chapter introduces the software, and the software setup will continue at the end of the next chapter.

3.3 Software components

IoTempower is a versatile IoT framework that can operate on any computer. It comes with built-in support for Node-RED and Cloud Commander. Cloud Commander is a web-based file manager that allows users to view and edit files. On the other hand, Node-RED is a visual programming framework that enables the creation of flows. Node-RED also has support for XRP Ledger. IoTempower enables the central control system by running on a Raspberry Pi.

IoTempower follows a folder structure to maintain organisation and clarity. This approach allows for easy navigation and management of devices within the IoT network. IoTempower's built-in MQTT system IoTknit uses this folder structure to set the MQTT topic of a device automatically.

MQTT is used as it is lightweight and suitable for resource-constrained settings. MQTT simplifies IoT integration with its publish/subscribe architecture, as it enables easy debugging and inspection of messages. Various data formats, such as JSON and XML, can be sent. MQTT assists in setting up a system simulation. Using MQTT real hardware components can be replaced with mock-ups until the system's foundation is in place. For a coffee station that accepts cryptocurrency payments, MQTT can be used to mock the payments and expedite the verification process.

In this chapter, we expand on how to use IoTempower and how the system is set up. Setting up IoTempower on a Raspberry Pi involves several steps, which are explained in more detail on the IoTempowers Pi Quickstart guide [53].

1. Flash the IoTempower software onto an SD card.
2. Insert the SD card into the Raspberry Pi.
3. Power the Raspberry Pi using the default power supply.

4. Establish a connection and SSH into the Raspberry Pi.
5. Edit the wifi.txt file to set the desired Wi-Fi name and password.
6. Modify the wifi-in.txt file to connect to the internet and reboot the Raspberry Pi.

Here are the general steps for incorporating a new device with IoTempower: [54]

1. Connect the device to the Raspberry Pi.
2. Identify the device using IoTempower commands.
3. Create a new node within the network.
4. Obtain the example code and modify it according to your needs.
5. Deploy the updated code onto the device.
6. Test the wiring by sending MQTT messages to the topic following the folder structure.
7. Design a flow in Node-RED to manage the device.
8. Utilise the Node-RED UI for further testing and interaction with the device.

To set up the scale, first, an ESP8266 needs to be connected to the gateway. A new node template called "scale" can be created. Navigate into the node folder and edit the setup.cpp. Copy in the following code: "*hx711(weight, D1, D2, 419.0, false);*". This means we must use GPIO pins D1 and D2 for the scale. Edit the node.conf file and set the board as "*Wemos D1 Mini*". Now the node can be deployed, and it will be automatically flashed onto the ESP8266 connected to a USB port. Using Node-RED, an MQTT node can receive the weight values from the topic scale/weight/set.

For the Sonoff Basic, there are a few extra steps. Connect the Sonoff Basic via USB to the computer while pressing the button. Then the Tasmota [55] software needs to be flashed onto the device. Tasmota has a web tool for flashing. Next steps similar to the scale have to be taken. Add the code from the examples folder into the setup.cpp file and set the board as "*sonoff*" in the node.conf. Run a compile command in the Sonoff node folder. Navigate to "/build/.pio/build/sonoff" and download the firmware.bin file. This can be flashed onto the Sonoff device as a firmware update. The Sonoff should now work as any other device. Using Node-RED, an MQTT node can be used to send "coffee-machine/relais1/set", which will toggle the Sonoff on and off.

3.3.1 Assessing XRP Ledger's payment methods

This chapter will assess the viability of XRP Ledger's payment methods for a coffee payment system. The payment methods include Direct payments, cross-currency payments, checks, escrow, payment channels and tokens.

Direct Payments on the XRP ledger involve sending XRP, the ledger's native currency, between accounts. This simple payment method requires two accounts, one for the customer and one for the vendor. A direct payment sends XRP directly from one account to another. Though direct payments can be monitored, their downside lies in the unstable price of XRP, which poses a significant risk when holding large amounts.

Cross-Currency Payments allow the implementation of stablecoins, which are connected to real-world prices. The vendor establishes a trust line with a stablecoin issuer to do this. Customers can then send payments in their desired currency, specifying the currency the vendor should receive. Despite offering a more secure way to store larger sums on the ledger, this method's drawback is the need for a trustworthy stablecoin issuer.

Checks are an inefficient payment method for small transactions like purchasing coffee. Customers need to sign and present the check to the vendor, which can expire if not cashed in time.

Escrows although theoretically possible for coffee payments, escrow is less convenient than direct or other payment types. The customer can cancel the payment if certain conditions are not met, and time-based escrow would release payments from an escrow account. This payment method requires several inconvenient steps to work for coffee transactions.

Payment Channels would be suitable for larger events with multiple transactions. Payment channels allow customers to make various payments and purchases throughout an event by setting up a channel with an expiration date. This method can be used to pay for coffee-related items like milk, sugar, and electricity. Payment channels provide interesting new payment solutions but are more advantageous when many small and precise payment amounts are needed.

Fungible or Non-Fungible Tokens can be created as a unique coffee-selling method. Fungible, interchangeable tokens could represent a specific coffee type or size, allowing customers to purchase coffee using these tokens. Non-fungible tokens (NFTs), on the other hand, could represent limited-edition coffee blends. Customers could even buy the tokens in cash. While this approach introduces an innovative payment solution, it may require additional efforts to set up and maintain a token-based system and ensure customers' understanding and adoption of the new payment method.

In summary, the XRP Ledger offers various payment types. For the coffee purchase scenario, direct payments are the simplest but pose risks due to XRP's price volatility. Cross-currency payments offer a more stable option through stablecoins, but they require trust in the issuer. Checks and escrow are not efficient or convenient for coffee purchases, while payment channels are better suited for large events or microtransactions. Lastly, fungible or non-fungible tokens introduce an innovative approach but require additional setup, maintenance, and customer adoption efforts.

Considering the various factors, direct and cross-currency payments using stablecoins are the most suitable option for coffee purchases.

3.3.2 XRP Ledger integration

This chapter goes over the integration of XRP Ledgers Direct and Cross-Currency payments in an IoT setting using HTTP / WebSockets APIs, client libraries, and Node-RED. Initial experiments focused on WebSockets and client libraries, so moving forward, we will use the IoTempire framework in conjunction with Node-RED.

XRP Ledger supports Websockets and HTTP-based JSON-RPC, which can be used for account management, transactions, and data acquisition. Integrating Websockets into an IoT system involves establishing a persistent WebSocket connection with the XRP Ledger for real-time updates. JSON-RPC communicates in a request-response format, used when real-time updates are unnecessary. Depending on the use case, both WebSocket and JSON-RPC APIs can be used unencrypted (http:// and ws://) or encrypted using TLS (https:// and wss://).

Client libraries provide the users with prebuilt functions for communicating with the ledger. There are client libraries for Python, Javascript/Typescript, C++, Java and Ruby.[14] The libraries are used for connection management, doing transactions and ledger querying. The advantage of using the libraries over a WebSocket client is the ease of use.

Node-RED simplifies the development of IoT applications through a flow-based visual programming tool. It provides a range of pre-built nodes and customisable flows. The *node-red-contrib-xrpl* library [56] provides support for XRP Ledger.

Steps for integrating XRP Ledger with IoTempire and Node-RED:

1. Install IoTempire onto a Raspberry Pi or any suitable computer.
2. Log into Node-RED and ensure it is running on IoTempire.
3. Install the node-red-contrib-xrpl library in your Node-RED environment.
4. Configure your XRP Ledger account, including the private and public keys.
5. Create flows in Node-RED to handle Direct and Cross-Currency payments.
6. Implement the necessary logic for monitoring and validating incoming transactions.
7. Integrate the IoT devices (e.g., grinder, scale, relays) with the Node-RED flows.
8. Test the system for proper functioning.

The system's primary function is to start a coffee brewing process based on received payments. Flows will be discussed below.

[14] https://xrpl.org/client-libraries.html

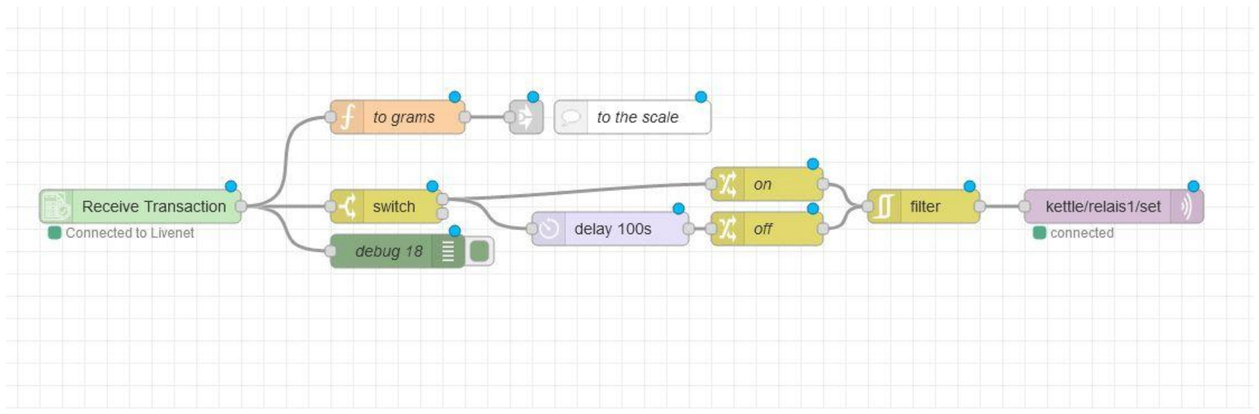

Figure 1. Transaction and Kettle Flow.

The Transaction Flow is responsible for handling incoming payments and initiating the brewing process. Its functionality can be summarised in the following steps:

1. The flow starts by receiving a payment.
2. The "tesSUCCESS" code is extracted from the "msg.payload.meta.TransactionResult" field. This code signifies a successful transaction.
3. If the transaction is successful, the flow triggers an MQTT command to turn on the kettle. The kettle is set to stay on for 100 seconds, providing enough time to boil the water necessary for brewing the coffee.
4. Concurrently, the payment amount received from "msg.payload.meta.delivered_amount.value" is converted into grams using a predefined rate (1 gram of coffee costs 7 euro cents or 0.15 XRP), and this gram-value is sent to the Scale Flow.

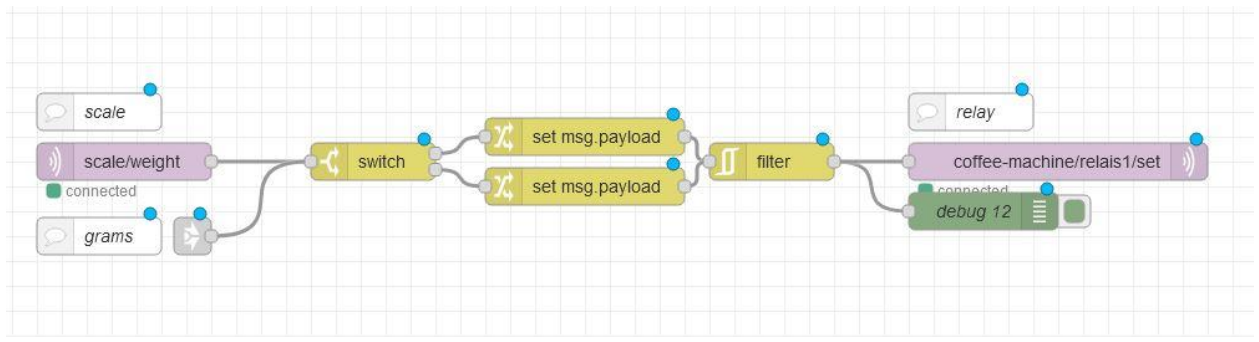

Figure 2. Scale Flow

The Scale Flow is responsible for weighing the coffee grounds and controlling the coffee grinder. The process involves the following steps:

1. The Scale Flow receives the gram-value from the Transaction Flow.
2. It then compares the received gram-value with the actual value measured from the scale.
3. The coffee grinder is activated, and it continues to grind coffee until the specified gram-value is reached. The actual weight is continuously compared with the specified gram-value to ensure precision.
4. Once the required weight is reached, the grinder is turned off via an MQTT command.

In conclusion, the system is designed to automate the coffee brewing process. Upon receiving a successful transaction, the kettle is automatically activated to start boiling the water. Concurrently, the coffee grinder is engaged, grinding the coffee until the purchased amount is reached, after which it turns off automatically. This process provides the precise quantity of coffee paid for by the user.

3.4 User interface and experience

This chapter discusses the user interface and experience for purchasing coffee using the system created. Although setting up an account takes about 5 minutes, buying coffee does not take more than 15 seconds.

First, an XRP Ledger account needs to be set up using the XUMM app:

1. Download the XUMM app on the customer's smartphone.
2. Create an XRPL account within the XUMM app.
3. Write down or remember the 48-digit recovery phrase for account access and security.

Customers can purchase XRP using real money through a bank transfer to fund their accounts. Theoretically, for a single coffee purchase, an account needs about 15 XRP, which is about 6 euros. This includes 10 XRP to meet the minimum account reserve requirement and 2 XRP for holding a stablecoin (optional but recommended). The account also needs to have funds for the coffee, so additional 3 XRP is needed. The reserve can be later cashed out.

In practice, the cheapest option (at the time of writing) in the XUMM app is BTCDirect, which charges a 1% fee and has a minimum purchase amount of 30 euros (equivalent to 69.42 XRP). Purchasing XRP requires entering bank details and providing proof of identity to comply with anti-money laundering laws. To cash out all of the funds on an account, the customer can use BTCDirect, and the minimum cash-out option is 30 euros.

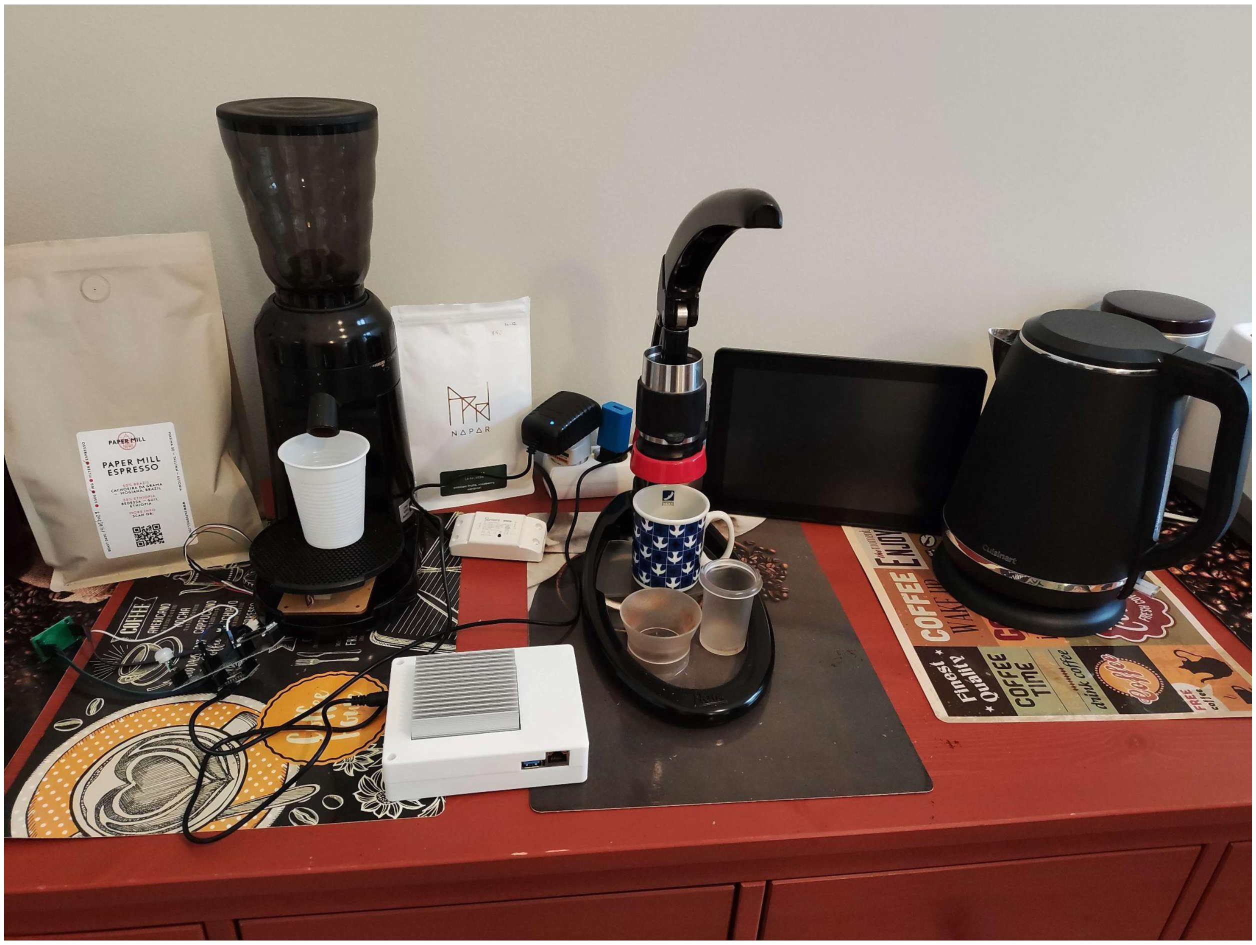

Figure 1. Exresso Machina

Paying for a coffee:

1. After funding their account, customers can scan the QR code to pay for their coffee.
2. The coffee grinder activates, allowing the customer to grind the payed amount of grams before turning it off automatically.
3. Simultaneously, a kettle is turned on, allowing the user to make either a pour-over coffee or an espresso.

A detailed guide with images is on the webpage [57], showcasing each process step and providing visual aid.

Following the outlined steps and incorporating a visually appealing and easy-to-navigate webpage, customers can enjoy a seamless experience while purchasing coffee using the XRP Ledger-integrated IoT system.

4. Conclusion and Future Work

In this thesis, we embarked on a journey that started with an exploration of the best practices for IoT development, delved into the interrelation of IoT and Microcontroller-Based Payment Systems, and studied the implications of blockchain in this domain. We established an understanding of how transactions are made on the XRP Ledger and examined the ESP8266 microcontroller with a specific focus on the limitations of microcontrollers for micropayments. Subsequently, we evaluated existing solutions for buying coffee at the university, leading to the proposal and implementation of an innovative coffee station payment system.

Summary of Main Findings

Our exploration provided key insights. We discovered that the XRP Ledger does indeed have lower fees on the decentralised exchange, suggesting that the system has significant potential for wider application. It was also found that purchasing XRP incurs a small fee, but the trades have lower fees, providing a cost-effective solution for micropayments. Getting funds on the account raised some problems, as the minimum amount of XRP that could be bought on the XUMM app was 30 euros.

Implications of the Study

This study extends beyond the scope of the XRP Ledger, shedding light on the broader potential of blockchain technology in IoT-based payment systems. Our findings confirmed that the XRP Ledger possesses all the necessary features to build an effective IoT payment system and can be implemented successfully while adhering to the best practices of IoT development.

Answer to the Problem Statement

Addressing the problem statement we initially proposed, our research successfully demonstrated a scalable IoT payment system, thereby providing a potential solution to navigate the fragmented IoT ecosystem. The low fees on the XRP Ledger's decentralised exchange, as low as 0.2 % with current stablecoin issuers, validated the claims of fee elimination and provided evidence for its applicability in an IoT-based payment system. Furthermore, we illustrated that cryptocurrency payment could be used to pay for quality coffee within a university setting.

Study Limitations

Despite the promising results, our study was not exempt from limitations. The number of payments made on the live ledger was insufficient to draw definitive conclusions, and the experimental setup of the coffee station, which remained behind closed doors, prevented real customers from interacting with and utilising the system.

Future Work

Potential avenues for future research could include exploring payment channels for offline payments, creating tokens for selling coffee (acting as a unique stablecoin and potentially minimising XRP buying fees), and a detailed analysis of the history of fees on the ledger itself. Several system architecture and implementation elements could be enhanced for future iterations. Notably, the user interface for selecting the number of coffee beans to be ground could be made more intuitive and user-friendly. The practical implications of our research are numerous and warrant further exploration. For example, the coffee station could be strategically placed in areas frequented by professors and PhD students, potentially maximising its use and effectiveness.

Future Predictions

Based on our findings, we predict a bright future for IoT micropayments. With the support of platforms like the XRP Ledger, these systems can flourish, potentially revolutionising how we approach everyday transactions.

References

[1] E. R. Brito, ‘The Units of Permissionless Consensus: Towards Mobile and Edge Computing’. Fall 2022.

[2] ‘Home | XRPL.org’. https://xrpl.org/ (accessed May 07, 2023).

[3] ‘My Grammarly - Grammarly’. https://app.grammarly.com/ (accessed May 10, 2023).

[4] ‘OpenAI’. https://openai.com/ (accessed May 10, 2023).

[5] ‘iotempire’. https://github.com/iotempire (accessed May 08, 2023).

[6] S. Haga and K. Omote, ‘IoT-Based Autonomous Pay-As-You-Go Payment System with the Contract Wallet’, *Secur. Commun. Netw.*, vol. 2021, p. e8937448, Sep. 2021, doi: 10.1155/2021/8937448.

[7] M. Borowski-Beszta and M. Jakubowska, ‘Mobile payments using NFC technology in the light of empirical research’, vol. 17, pp. 5–16, Jun. 2019, doi: 10.19197/tbr.v17i3.311.

[8] ‘Are Americans Embracing Mobile Payments?’, Oct. 03, 2019. https://pew.org/2pfYXCi (accessed May 07, 2023).

[9] 1615 L. St NW, S. 800 Washington, and D. 20036 U.-419-4300 | M.-857-8562 | F.-419-4372 | M. Inquiries, ‘Mobile Fact Sheet’, *Pew Research Center: Internet, Science & Tech*. https://www.pewresearch.org/internet/fact-sheet/mobile/ (accessed May 07, 2023).

[10] ‘Wearable Payments Devices Market Size Report, 2021-2028’. https://www.grandviewresearch.com/industry-analysis/wearable-payments-devices-market (accessed May 07, 2023).

[11] ‘Juniper Research: Biometrics to Secure Over $3 Trillion in Mobile Payments by 2025; Driven by Shift to App-based mCommerce’, Feb. 02, 2021. https://www.businesswire.com/news/home/20210202005592/en/Juniper-Research-Biometrics-to-Secure-Over-3-Trillion-in-Mobile-Payments-by-2025-Driven-by-Shift-to-App-based-mCommerce (accessed May 07, 2023).

[12] S. Jokić, A. S. Cvetković, S. Adamović, N. Ristić, and P. Spalević, ‘Comparative analysis of cryptocurrency wallets vs traditional wallets’, presented at the Ekonomika, Oct. 2019. doi: 10.5937/ekonomika1903065J.

[13] *Buying Coffee with XRP*, (Oct. 04, 2019). Accessed: May 07, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=7Kv2oKWq5Ug

[14] N. Papadis and L. Tassiulas, ‘Blockchain-Based Payment Channel Networks: Challenges and Recent Advances’, *IEEE Access*, vol. 8, pp. 227596–227609, 2020, doi: 10.1109/ACCESS.2020.3046020.

[15] S. Mercan, A. Kurt, K. Akkaya, and E. Erdin, ‘Cryptocurrency Solutions to Enable Micropayments in Consumer IoT’, *IEEE Consum. Electron. Mag.*, vol. 11, no. 2, pp. 97–103, Mar. 2022, doi: 10.1109/MCE.2021.3060720.

[16] C. Zhou, W. Song, L. Liu, and Z. Niu, ‘Blockchain Technology-Enabled Smart Product-Service System Lifecycle Management: A Conceptual Framework’, in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, Aug. 2020, pp. 1175–1180. doi: 10.1109/CASE48305.2020.9216809.

[17] S. Nakamoto, ‘Bitcoin: A Peer-to-Peer Electronic Cash System’.

[18] B. Chase and E. MacBrough, ‘Analysis of the XRP Ledger Consensus Protocol’. arXiv, Feb. 20, 2018. Accessed: May 07, 2023. [Online]. Available: http://arxiv.org/abs/1802.07242

[19] ‘Transaction Cost | XRPL.org’. https://xrpl.org/transaction-cost.html (accessed May 07, 2023).

[20] ‘Crypto Solutions for Business | Ripple’. https://ripple.com/ (accessed May 07, 2023).
[21] ‘XRP Ledger Foundation (Official)’, *GitHub*. https://github.com/XRPLF (accessed May 07, 2023).
[22] ‘XRP Ledger Foundation’. https://foundation.xrpl.org/ (accessed May 07, 2023).
[23] ‘Unhosted Exchange DEX’. https://unhosted.exchange/?base=XRP&quote=EUR_rhub8VRN55s94qWKDv6jmDy1pUykJzF3wq (accessed May 07, 2023).
[24] S. Swayamsiddha, D. Mukherjee, and S. Ramavath, ‘Home Automation Using ESP8266 of IOT Module’, in *Emerging Technologies in Data Mining and Information Security*, A. E. Hassanien, S. Bhattacharyya, S. Chakrabati, A. Bhattacharya, and S. Dutta, Eds., in Advances in Intelligent Systems and Computing. Singapore: Springer, 2021, pp. 409–417. doi: 10.1007/978-981-15-9927-9_40.
[25] Admin, ‘IoT Smart Agriculture & Automatic Irrigation System with ESP8266’, *How To Electronics*, Aug. 31, 2020. https://how2electronics.com/iot-smart-agriculture-automatic-irrigation-system-with-esp8266/ (accessed May 08, 2023).
[26] O. Taiwo and A. E. Ezugwu, ‘Internet of Things-Based Intelligent Smart Home Control System’, *Secur. Commun. Netw.*, vol. 2021, p. e9928254, Sep. 2021, doi: 10.1155/2021/9928254.
[27] L. Zhang, ‘SONOFF BASICR2-WiFi DIY Wireless Smart Switch’, *SONOFF Official*, Jan. 10, 2021. https://sonoff.tech/product/diy-smart-switches/basicr2/ (accessed May 08, 2023).
[28] ‘ESP8266 Wi-Fi MCU I Espressif Systems’. https://www.espressif.com/en/products/socs/esp8266 (accessed May 07, 2023).
[29] J. Mesquita, D. Guimarães, C. Pereira, F. Santos, and L. Almeida, ‘Assessing the ESP8266 WiFi module for the Internet of Things’, in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2018, pp. 784–791. doi: 10.1109/ETFA.2018.8502562.
[30] P. Macheso, T. Manda, S. Chisale, N. Dzupire, J. Mlatho, and D. Mukanyiligira, ‘Design of ESP8266 Smart Home Using MQTT and Node-RED’, Mar. 2021, pp. 502–505. doi: 10.1109/ICAIS50930.2021.9396027.
[31] Y. S. Parihar, ‘Internet of Things and Nodemcu A review of use of Nodemcu ESP8266 in IoT products’, vol. 6, p. 1085, Jun. 2019.
[32] ‘Tensilica Controllers and Extensible Processors’. https://www.cadence.com/en_US/home/tools/ip/tensilica-ip/tensilica-xtensa-controllers-and-extensible-processors.html (accessed May 07, 2023).
[33] ‘Arduino IDE 2 Tutorials | Arduino Documentation’. https://docs.arduino.cc/software/ide-v2 (accessed May 07, 2023).
[34] PlatformIO, ‘PlatformIO is a professional collaborative platform for embedded development’, *PlatformIO*. https://platformio.org (accessed May 07, 2023).
[35] ‘D1 Boards — WEMOS documentation’. https://www.wemos.cc/en/latest/d1/index.html (accessed May 10, 2023).
[36] ‘NodeMcu -- An open-source firmware based on ESP8266 wifi-soc.’ http://www.nodemcu.com/index_en.html (accessed May 10, 2023).
[37] ‘Adafruit Feather HUZZAH ESP8266’, *Adafruit Learning System*. https://learn.adafruit.com/adafruit-feather-huzzah-esp8266/overview (accessed May 10, 2023).

[38] D. Parida, A. Behera, J. K. Naik, S. Pattanaik, and R. S. Nanda, ‘Real-time Environment Monitoring System using ESP8266 and ThingSpeak on Internet of Things Platform’, in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, May 2019, pp. 225–229. doi: 10.1109/ICCS45141.2019.9065451.

[39] ‘ESP8266 NodeMCU: Getting Started with Firebase (Realtime Database) | Random Nerd Tutorials’, Sep. 18, 2021. https://randomnerdtutorials.com/esp8266-nodemcu-firebase-realtime-database/ (accessed May 07, 2023).

[40] V. M. Ionescu and F. M. Enescu, ‘Investigating the performance of MicroPython and C on ESP32 and STM32 microcontrollers’, in *2020 IEEE 26th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, Oct. 2020, pp. 234–237. doi: 10.1109/SIITME50350.2020.9292199.

[41] ‘IoTempower’. iotempire, Mar. 21, 2023. Accessed: May 08, 2023. [Online]. Available: https://github.com/iotempire/iotempower/blob/7d29c7b315ba194e492e6ed6c2e7589e95639922/doc/hardware.rst

[42] R. Han, V. Gramoli, and X. Xu, ‘Evaluating Blockchains for IoT’, in *2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, Feb. 2018, pp. 1–5. doi: 10.1109/NTMS.2018.8328736.

[43] M. C. Semmouni, A. Nitaj, and M. Belkasmi, ‘Bitcoin security with a twisted Edwards curve’, *J. Discrete Math. Sci. Cryptogr.*, vol. 25, no. 2, pp. 353–371, Feb. 2022, doi: 10.1080/09720529.2019.1681673.

[44] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, ‘High-speed high-security signatures’, *J. Cryptogr. Eng.*, vol. 2, no. 2, pp. 77–89, Sep. 2012, doi: 10.1007/s13389-012-0027-1.

[45] ‘Hario V60 Electric Coffee Grinder’, *Hario UK*. https://www.hario.co.uk/products/hario-v60-electric-coffee-grinder (accessed May 08, 2023).

[46] ‘Paper Mill’. https://papermill.coffee/ (accessed May 08, 2023).

[47] ‘maakler/expresso-machina: The repository for a coffee station’. https://github.com/maakler/expresso-machina/tree/main (accessed May 10, 2023).

[48] O. Vainikko, ‘maakler/xrp-coffee-machine-esp8266’. May 09, 2023. Accessed: May 10, 2023. [Online]. Available: https://github.com/maakler/xrp-coffee-machine-esp8266

[49] ‘ThingPulse OLED SSD1306 (ESP8266/ESP32/Mbed-OS)’. ThingPulse, May 03, 2023. Accessed: May 10, 2023. [Online]. Available: https://github.com/ThingPulse/esp8266-oled-ssd1306

[50] B. Blanchon, ‘bblanchon/ArduinoJson’. May 08, 2023. Accessed: May 10, 2023. [Online]. Available: https://github.com/bblanchon/ArduinoJson

[51] Markus, ‘WebSocket Server and Client for Arduino’. May 06, 2023. Accessed: May 10, 2023. [Online]. Available: https://github.com/Links2004/arduinoWebSockets

[52] ‘iotempower/architecture.rst at master · iotempire/iotempower · GitHub’. https://github.com/iotempire/iotempower/blob/master/doc/architecture.rst (accessed May 08, 2023).

[53] ‘iotempower/quickstart-pi.rst at master · iotempire/iotempower · GitHub’. https://github.com/iotempire/iotempower/blob/master/doc/quickstart-pi.rst (accessed May 08, 2023).

[54] ‘iotempower/first-node.rst at master · iotempire/iotempower · GitHub’.

https://github.com/iotempire/iotempower/blob/master/doc/first-node.rst (accessed May 08, 2023).
[55] ‘Tasmota Documentation - Tasmota’. https://tasmota.github.io/docs/ (accessed May 10, 2023).
[56] ‘node-red-contrib-xrpl’. http://flows.nodered.org/node/node-red-contrib-xrpl (accessed May 08, 2023).
[57] ‘Expresso Machina’. https://maakler.github.io/expresso-machina/ (accessed May 10, 2023).

Appendix

I. License

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks