TALLINN UNIVERSITY OF TECHNOLOGY

School of Information Technologies

155714IASB

Daaniel Ipsberg

SINGLE BOARD COMPUTERS AND MICROCONTROLLERS AT HOME

Bachelor's thesis

Supervisor: Vladimir Viies

Dotsent

Tallinn 2019

TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

155714IASB

Daaniel Ipsberg

SARD-ARVUTID JA MIKROKONTROLLERID KODUTEHNIKAS

Bakalaurusetöö

Juhendaja: Vladimir Viies

Dotsent

Tallinn 2019

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Daaniel Ipsberg

Date : 01.05.2018

Abstract

The bachelors thesis at hand focuses on finding the differences between single-board computers and microcontrollers and comparing them, at first, between their own kind and then between each other. In the first part both will explored more in depth and then compared against of their own kind to find a superior pick. In second part SBC and microcontrollers will be compared to each other to find differences and to see, which one is superior. Lastly an example project will be done with a superior system to explore its functionality and power.

This thesis is written in English and is 37 pages long, including 4 chapters, 13 figures and 8 tables.

Annotatsioon

Sard-arvutid ja mikrokontrollerid kodu tehnikas

Pole saladus, et tänapäeval, enamus asju on kontrollitud arvutite poolt. Arvutit on täpsemad, kiiremad ja on väiksema võimalusega vigu tekitama, mitte nagu inimesed. Nad ei jää kunagi tööle hiljaks ja vajavad väga vähe energiat, et töötada, mitte nagu inimesed. See ongi põhjus, mis nad on tulnud meie igapäeva ellu. Meelelahutusest ärini, neid võib leida kõikjalt ja kui mingil põhjusel peaksid nad töötamise lõpetama järgneks kaos, kuna inimesed on muutunud väga neist sõltuvaks.

Igapäev võib puutuda kokku mikrokontrollerite ja sard-arvutitega ilma sellest arusaamata. Nad muudavad elu mugavaks ilma ennast näitamata.

Mikrokontroller on arvuti, mis omab kõiki pea kõiki komponente nagu tavaline arvutigi. Tal on protsessor, mälu ja I/O. Kuid mitte nagu meie personaalsed arvutid, mikrokontrollerid on spetsiaalse eesmärgiga arvutid, mis suudavad teha ühte asja aga väga hästi ja kiiresti, mitte nagu meie personaalsed arvutid, mis suudavad teha palju asju aga aeglaselt. Esimene mikrokontroller, Intel 4004 leiutati aastal 1971 [1] ja nad olid esialgselt mõeldud kalkulaatorite jaoks. Selle loojad aga kiiresti avastasid, et neid on võimalik kasutada palju laiemalt, kohtades nagu kellad ja kõiksugu näidukid. Tänapäeval kasutatakse neid igalpool, meditsiinist ja sõjandusest mänguasjade ja kodutehnikani välja.

Aja möödudes inimesed proovisid teha funktsionaalseid arvuteid väiksemaks. Luua midagi, mis töötaks nagu mikrokontroller aga samal ajal saaks kasutada palju laiemal eesmärgil. Loodigi sard-arvutid. Sard-arvutid omavad kõiki komponente, nagu mikrokontroller: protsessorit, mälu ja I/O aga on ka võimalus kasutada operatsiooni süsteemi, mis muudab ta palju sarnasemaks personaal arvutile, kasutajasõbralikumaks ja võimaldab jooksutada mitut protsessi kiirust ohverdades. Mis eristab sard-arvutit personaal arvutitest on see, et ta omab programmeeritavat I/O-d, muutes ta palju funktsionaalsemaks. Ka suurus on palju väiksem tava arvutitest, kuna kõik vajalik on ühel trükkplaadil, kuid peab ohverdama laiendus ava. Esimene sard-arvuti Dyna-Micro loodi

1976. Sard-arvutite lihtne disain võimaldab isegi lastel õppida nendel programmeerimist, samas olles ka väga võimas tööriist masinate automatiseerimises, kui ka hasartmängudes.

The purpose of this bachelors work is to compare Microcontrollers and Single-board computers and see what their everyday use can be and identify areas which one would be more functional than the other.

Selle bakalaurusetöö eesmärgiks kujunesgi sard-arvutite ja mikrokontrollerite võrdlemine ja uurimine, et avastada nende igapäeva kasutus kohad ja tuvastada, kumb sobib paremini teatud valdkondades nagu näiteks kodutehnika.

On soovitud avastada ja tuvastada, kuidas sard-arvutit ja mikrokontrollerid erinevad üksteisest ja kuidas need erinevused mõjutavad nende valikud erinevates kohtades ja projektides. Sellejaoks on võrreldud kõige laialt levinud sard-arvuteid ja mikrokontrollereid. Alguses omasuguste vastu ja siis üksteise vastu. Parameetrid, mida võrreldakse on voolu kasutus, funktsionaalsus, I/O ja teisigi.

Lõpus uuritakse täpsemalt sard-arvutit, et avastada tema funktsionaalsust ja näha kuidas on seda võimalik kasutada kodutehnikas. Uuringu käigus on näha ka selle plusse ja miinuseid.

Lõputöö on kirjutatud Inglise keeles ning sisaldab teksti 37 leheküljel, 4 peatükki, 13 joonist, 8 tabelit.

List of abbreviations and terms

TUT	Tallinn University of Technology
SBC	Single-board computer. Complete computer on a single circuit
MCU/CU	Microcontroller unit. Small computer on a single integrated circuit.
IoT	Internet Of Things. Network of physical devices that are embedded with electronics and connectivity, which enables these objects to connect and exchange data.
RPI	Raspberry PI. A SBC
UART	Universal asynchronous receiver-transmitter. Computer hardware device for serial communication. Used to send data between two devices.
USART	Universal synchronous and asynchronous receiver-transmitter. Same as previous, but also has synchronous capability
BT	Bluetooth. Wireless technology standard for exchanging data over short distances
SPI	Serial Peripheral Interface bus. Synchronous serial communication interface for short distance communicated. Primarily in embedded systems.
I2C	Inter-Integrated Circuit. Widely used for short-distance intra-board communication.

CAN	Controller Area Network. Robust vehicle bus, that allows MCU and devices to communicate with each other without host computer.
JTAG	Joint Test Action Group. Industry standard for verifying designs and testing for printed circuit boards
AVR	Family of MCUs
PIC	Family of MCUs
8051	Old family of MCUs
ARM	Widely used family of MCUs
DSP	Digital signal processor. Specialized microprocessor optimized for digital signal processing.
SAI	Serial audio interface. Peripheral to support wide set of audio protocols.
IrDA	Infrared Data Association. Group that provides specifications for a complete set of protocols for wireless infrared communications
LIN	Local Interconnect Network. Serial network protocol used for communication between components in veichles.
USB	Universal Serial Bus. Industry standard to define cables, connectors and protocols for connection, communication and power supply between personal computers and their peripheral devices.
I/O	Input/Output. Communication ports for information processing system (Computer to the outside world)
GPIO	General-purpose input/output. Generic pin on an integrated circuit, that is controllable by the user.
CPU	Central processing unit. Electronic circuitry that carries out instruction of a program.
GPU	Graphics processing unit. Specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images.
RAM	Random-access memory. Form of computer data storage that stores data currently being used.
EPROM	Erasable programmable read-only memory. Type of memory chip that retains data after power supply is switched off.

EEPROM	Electrically erasable programmable read-only memory. Same as previous, but have an option to allow individual bytes to be erased and reprogrammed.
DAC	Digital to analog converter. Converts one type of signal into another.
ADC	Analog to digital converter. Converts one type of signal into another.
OS	Operating system. System software to manage computer hardware and software resources.
UI	User interface. Interface that makes it easier for users to use given system.
FPGA	Field-programmable gate array. Integrated circuit designed to be configured by a designer after manufacturing.
PLC	Programmable logic controller. Industrial rugged digital computer adapted to control manufacturing processes.
RISC	Reduced instruction set computer. Instruction set architecture that allows to have fewer cycler per instructions.
CISC	Complex instruction set computer. Instruction set architecture that has instructions that can execute several low-level operations.
KB	Kilobyte. 10^3 Bytes, which is a unit of digital information.
MB	Megabyte. 10^6 bytes
GB	Gigabyte. 10^9 bytes
ISP flash	In-system programming flash. Ability to program embedded devices while installed in a complete system.
MHz	Megahertz. Hertz is a unit of frequency
ATM	Automated teller machine. Cash dispenser machine.
NFC	Near-field communication. A set of communication protocols that enable two electronic devices to establish communication by bringing them together.
Wi-FI	Wireless local area networking. Technology, that allows Wi-Fi enabled devices to connect to the internet wirelessly.
DDR3	Type of RAM

SD card	Secure Digital card. Non-volatile memory card for use in portable devices.
HDMI	High-Definition Multimedia Interface. Proprietary audio/video interface for transmitting uncompressed video data and/or digital audio data.
DX	DirectX. Collection of application programming interfaces for handling tasks related to multimedia.
PCIe	Peripheral Component Interconnect Express. High-speed serial expansion bus standard for attaching hardware to a computer.
ISA	Instruction set architecture. Set of basic instructions that processor (CPU) understands.
SATA	Serial AT Attachment. Computer bus interface to connect mass storage devices.
1-wire	Device communications bus system that provides low-speed data, signaling and power over a single conducting wire.
APU	Accelerated Processing Unit. Microprocessor designed to act as CPU and GPU
LED	Light-emitting diode Emits light when suitable current is applied to the leads.
RFID	Radio-frequency identification. Identifies tags and reads their stored information by using electromagnetic fields

Table of contents

List of figures

List of tables

1 Introduction

It is no secret that now days everything is mostly controlled by computers. Computers are more accurate, faster and less likely to make errors than people. They are never late for work and require little energy to run, unlike humans. That is the reason they have crawled into the everyday lives of people. From entertainment to business, found everywhere and if they were to stop working, chaos would ensure as people have become more and more dependent on them.

Every day microcontrollers and single-board computers are around, without people not even knowing, making life more convenient.

A microcontroller is a computer that has almost all the components like a regular computer. It has a processor, some memory and an I/O. However, unlike our everyday personal computers microcontrollers are special purpose computers, meaning they can do one thing, really well and fast [2], unlike our personal computers that can do a lot of things, but much slower. The first microcontroller, the Intel 4004 was invented in 1971 [1] and they were originally meant to be used for calculators. However the makers quickly saw the potential of the microcontroller. With little programming it could be used in variety of places, from clocks to all types of meters. Now they are used everywhere, from Health care and military to toys and household products [3].

As time goes on people try to make more functional computers smaller and smaller. Something that works as a microcontroller, but at the same time be used in a way wider variety of places. Thus came to be Single-board computers. The SBC has all the components like the microcontroller: a processor, memory and some I/O. However it also has an option to run an operating system on it, giving it an ability to not only have a much user friendlier UI but also the ability to run multiple processes at the cost of speed. The way SBC differ greatly from regular computers is that they have a programmable I/O making it much more functional. It is also much smaller, since all that is needed is already on board, at the cost of expansion slots. The first SBC Dyna-Micro came to life in 1976 [4]. Single-board computers are usually of simple design allowing even kids to study

programming, but in the right hands they can be a very powerful tool for creating machine control automation to slot machines.

The purpose of this bachelors work is to compare Microcontrollers and Single-board computers and see what their everyday use can be and identify areas which one would be more functional than the other.

It is desired to discover how microcontrollers and SBC-s differ from each other and how these differences can affect the choice of one or another in projects. For this most common and widely spread SBC-s and microcontrollers will be chosen and compared. First against others among their kind and then against each other. Parameters compared are power consumption, functionality, I/O and many more.

In the end the SBC will be more looked into to explore their functionality and see why they are superior at home usage, if they are. Pros and cons will also be brought out to see if it was a good pick.

2 Single-board computers and microcontrollers

This chapter will compare the most popular Microcontrollers and Single-board computers. An overview will be given on their history, structure, how they work and their uses in everyday life.

2.1 Microcontrollers

2.1.1 Basic structure of microcontroller

Microcontroller is a small computer on a single integrated circuit, simply put: a tiny computer. Microcontrollers are really small computers. They almost everything that a normal personal computer has, that people use on a daily basis: a CPU, memory, I/O and all the other smaller things. At the same time, it can do so much more than a regular personal computer and so much less.

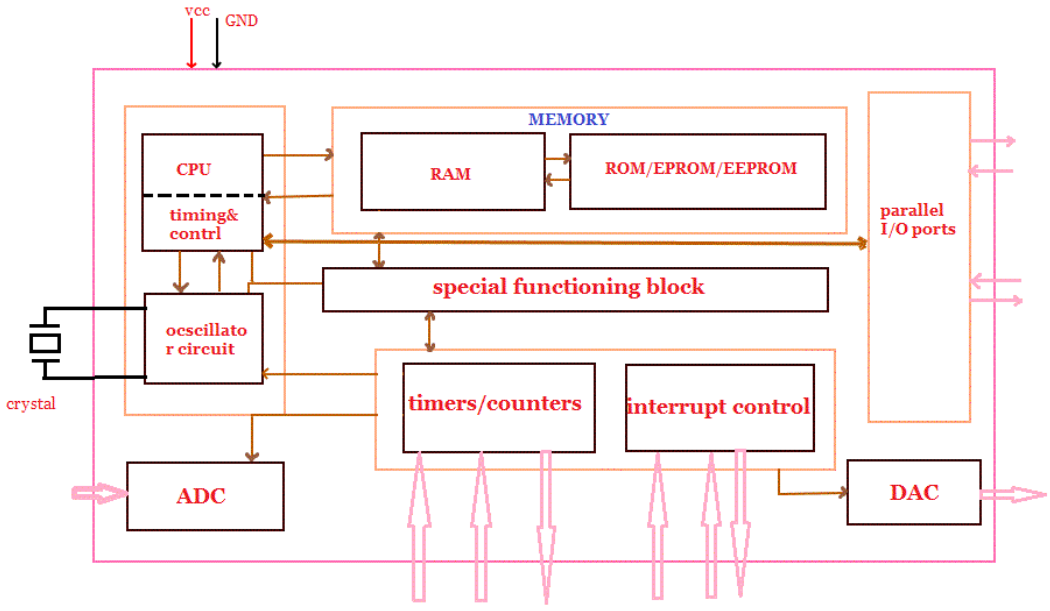

Figure 1 Structure of a microcontroller [6]

The biggest way it differs from regular computers is because Microcontrollers can be programmed and they can do one specific thing, really well and fast. Everyday computers can be programmed to some extend as well, but they are usually running tons of services that the Operating system has to manage. Every service requires to be managed, so it cannot do only one thing, but it has to do many. This is the main way they differ from each other.

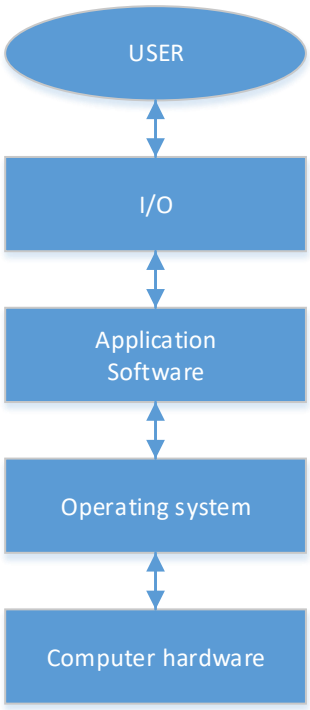

Figure 2 Operating system simplified flow

However, the microcontroller, from the time it starts will only run what is programmed to do, meaning the full extent of its powers can be used to do something really quickly like: measuring temperature with its sensors, adjusting inner climate, pumping water and so on.

The Microcontroller is so powerful because of its programmable I/O and timers. Pins on the controllers can be used for, as the name says, for input and output. It can take in different signals and also put out different signals, like constant or variable signals in time [7]. The way ports react and work is all defined by the user which gives the programmer full control over how the system works. This gives users flexibility but it also has a learning curve, since there is no UI to simplify programming something like this easily. The code running such controllers is usually written in a high-level (HLL) programming language, such as C, python or one of the other hardware programming languages.

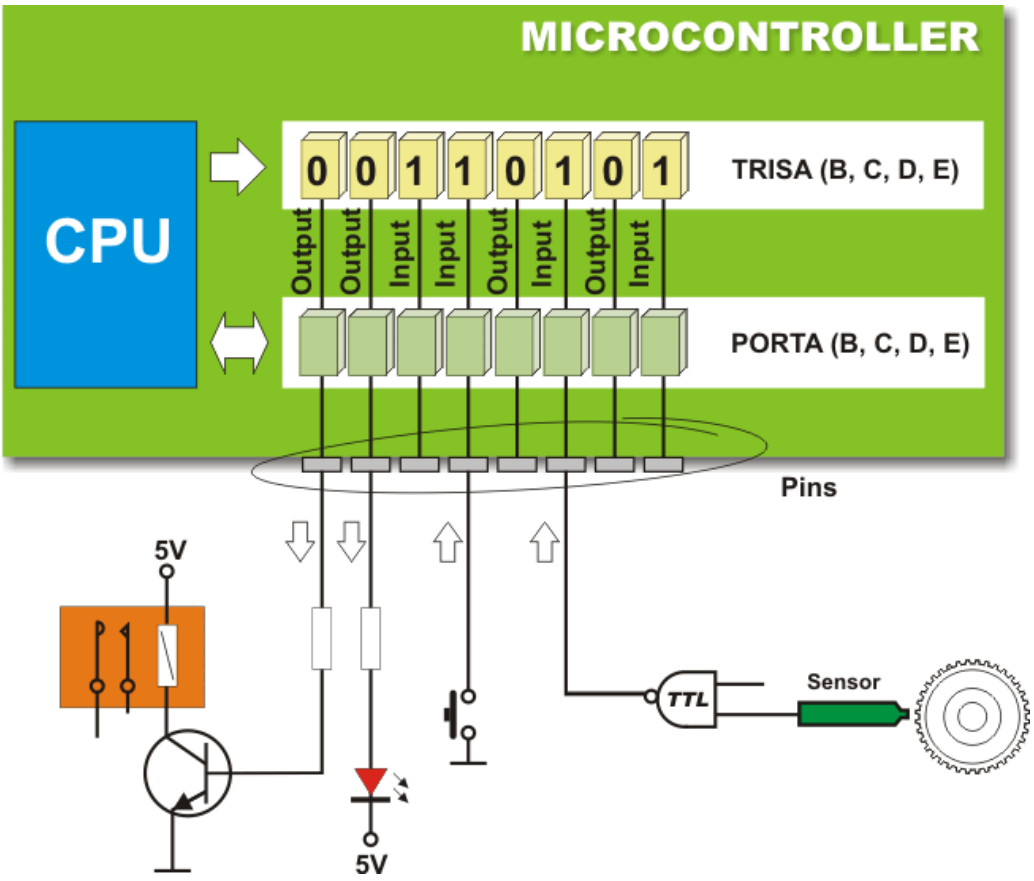

Figure 2 Example of microcontroller I/O [23]

In Figure 2.3 it can be seen how it has two inputs: a button and a sensor.

Because of the way it works it also uses much less power. Microcontrollers have an ability to sleep or low power mode. In such state, Microcontroller waits for outside stimuli on a specific I/O or for an inner timer to reach a certain count. This gives them an ability to be used on the go with a battery. Low power consumption allows microcontrollers to run for days.

2.1.2 Microcontrollers in everyday appliances

Thanks to their small form factor, small size, power and low power consumption microcontrollers can be found everywhere: from every day appliances where we could not even dream they were to multimillion military drones, typically for electrical circuits. A few examples will be brought from robotics and automation of home.

2.1.2.1 Robotics

Microcontrollers are used very widely now days. From small enthusiast robots to huge mega factories. A very widely used microcontrollers in schools for robotics are usually the Arduino microcontrollers. They come in different sizes and prices, but are still much cheaper and easier to handle.

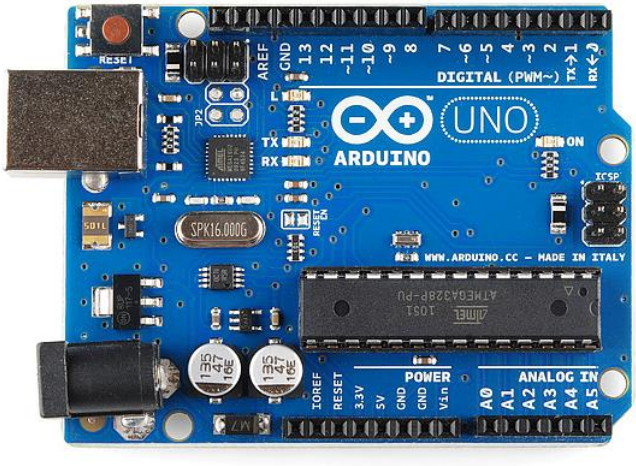

Figure 3 Arduino microcontroller [24]

Arduinos is an open source physical computing platform that can, for example, be programmed to do some line following, maze solving or obstacle avoiding, of course by using the required sensors. They can handle simple workloads and are very flexible. But if it comes to industry robotics, they are way out of their league because they are not very secure, cannot run several processes, and are too slow.

In robotics industry a much higher speed microcontrollers, like FPGAs and PLCs are used. These controllers are much more rugged and usually more expensive as well. Thanks to the large resource of logic gates and ram, they can be used to implement complex digital computation. They are much more secure, faster (can do several processes at a time) and much more flexible and because this they are used widely in safety-critical systems, like defence, medical and other kinds of robotics.

2.1.2.2 Home appliances

Microcontrollers can be found almost every electrical device at home. As homes become smarter and even the smallest things become a part of the internet of things [8], more microcontrollers are implemented to make life easier. Everything is communicating together, to work as efficiently as possible.

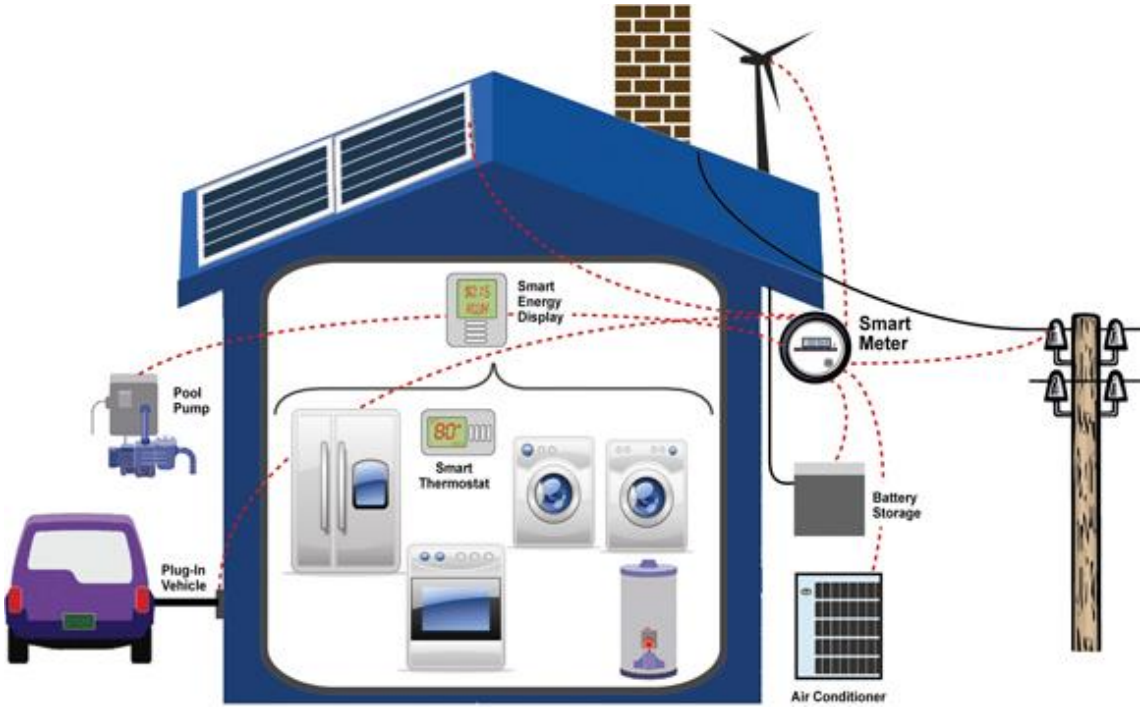

Figure 4 IoT at home [9]

From waking up until going to sleep, everything could be controlled. A coffee machine can start pouring a fresh cup of coffee when the owner wakes up and a machine pulls up the blinds. The house will know when the owner is not at home. It can go into lower power consumption, cooling the place down and give the command for autonomous robots to start cleaning. By the time owner is home, the house is again warm and clean. As time goes on, microcontrollers can be even used to automatically start charging our cars, when the owner comes home, a concept that Tesla is currently working on [10], and it does not stop there.

Depending on the complexity, different kind of controllers can be picked. Easier objectives can be fulfilled by Arduino controllers, but as more sensors come to use, home automation purpose controllers come into play. These are controllers that can wirelessly communicate with each other, like the Zigbee devices.

2.1.3 Comparing different Microcontrollers

In the following section ten commonly used microcontrollers will be compared. The compared specs are as follow:

- Data bus : What is the size of instructions and data used
- Clock speed : At what frequency can the microcontroller work at
- Memory and size: Type of memory used and size.
- I/O : What kind of I/O it has, what can it do
- Serial communication : What kind of communication is available
- Power consumption : How much power consumes, power states
- Cost
- Additional

The compared microcontrollers are picked mostly by popularity of the series, however some series have a lot of different choices based on needs so usually a more capable and/or the cheapest is picked, to give an overview. The microcontrollers picked are mostly RISC type controllers as these types of controllers are software-centric in design. One 8051 CISC microcontroller is picked as-well to bring side by side. It is an older microcontroller now mostly obsolete, but still widely used for educational purposes. The four most popular families of microcontrollers will be compared: AVR, 8051, ARM and PIC.

Table 1 Microcontroller comparisons

Microcontroller	Bus width	Clock speed	Memory & size	I/O
AVR ATmega328	8-bit RISC	20MHz	32KB ISP flash 1KB EEPROM	23 pins
AVR ATmega2560	8-bit RISC	16MHz	256KB ISP flash 4KB EEPROM	86 pins
AVR ATmega128	8-bit RISC	16MHz	128KB ISP flash 4KB EEPROM	53 pins
8051 P89C664	8 bit CISC*	20/40MHz	64KB flash	32 pins
ARM LPC2294HBD	32 bit/16 bit RISC	60MHz	256KB flash	112 pins
ARM LPC2101FBD	32-bit/16 bit RISC	70MHz	8KB flash	32 pins
PIC16F887	8-bit RISC	20MHz	8KB flash 256 byte EEPROM	16 pins
PIC24F04KA201	16-bit RISC	32MHz	4KB	20 pins

Microcontroller	Serial communication	Power consumption(A)	Cost	Additional
AVR ATmega328	UART, 2-SPI, I2C	At 1MHz 1.8V AM 0.2mA PDM 0.1uA PSM 0.75uA	1.55€ per unit	2 8-bit timers 16-bit timer
AVR ATmega2560	4-USART SPI, I2C	At 1MHz 1.8V AM 500uA PDM 0.1uA	9.86$ per unit	4 16-bit timers
AVR ATmega128	USART, SPI, I2C	Not specified	9.44$ Per unit	2 16-bit timer
8051 P89C664	UART	Not specified	Obsolete	No on chip peripherals like ADC, I2C
ARM LPC2294HBD	CAN, I2C, SPI, UART	At 60Mhz 1.8v AM 45mA PDM 10uA	16,31€ Per unit	2 32-bit timer Dual power supply
ARM LPC2101FBD	I2C, JTAG, SPI, SSP, UART	At 70Mhz 1.8v AM 41mA PDM 2.5uA DPDM 0.7uA	3.56€ per unit	32 bit timer Deep power-down mode.
PIC16F887	SPI,I2C, USART	Operating 2.0v 11 uA @ 32kHz 220uA@ 4MHz Standby 50 nA	1.83€ per unit	In-circuit debugger
PIC24F04KA201	SPI,I2C,UART	Run mode 8uA Idle 2uA	1.30€ per unit	3 16-bit timers

Table 2 Microcontroller comparisons

The main differences between AVR, 8051, ARM and PIC microcontrollers are seen in Table 2.2

Table 3 Microcontroller comparison summary

Microcontroller	AVR	8051	ARM	PIC
Bus width	8/32-bit	8-bit	16/32-bit also some in 64bit	8/16/32 bit
Communication Protocols	UART, USART, SPI, I2C, (special purpose AVR support CAN, USB, Ethernet)	UART, USART,SPI,I2C	UART, USART, LIN, I2C, SPI, CAN, USB, Ethernet, I2S, DSP, SAI (serial audio interface), IrDA	PIC, UART, USART, LIN, CAN, Ethernet, SPI, I2S
Community support	Very good (Arduino community)	Good	Good	Very good
Instruction set architecture	RISC	CISC	RISC	RISC

The ARM has become one of the most famous microcontrollers in production, because it's the fastest of the others. It is widely used in everyday devices like phones. However hobbyist usually rather use AVR or PIC because of the good community support. They have active forums where users constantly chare projects and problems, making it much easier to get into them. 8051 are still good for learning as well, but they have an old design which only supports 8-bits, making them much slower compared to others and because of the smaller community behind it, its popularity is smaller.

2.2 Single-board computers

2.2.1 Basic structure of single-board computers

Single-board computer is a complete computer that has been built on a single circuit board. They are a mix of microcontrollers and personal computers. SBCs like microcontrollers have programmable I/O. They have a wide variant of variants and are usually backed by a huge community. The biggest difference between SBC and personal computers is that they usually do not have expansion slots and they own a programmable I/O. Unlike microcontrollers, SBCs usually have expandable storage to store the operating system needed to run the system. It is possible to install operating systems that run the bare minimum, but they will never be as fast as microcontrollers for special purpose projects and that is why they are rarely used for this.

Figure 5 Single-board computer Raspberry PI [25]

Single board computers are usually much cheaper than personal computers, but they are rarely as powerful and require more expertise to operate.

2.2.2 Single-board computers in everyday appliances

Even though Single-board computers are not as fast as microcontrollers or as powerful as personal computers they are still widely used all around. They are very famous among hobbyists because they are usually very flexible and have a chance to run multiple processes at once, giving programmers the ability to run scripts and multiple programs.

2.2.2.1 Single-board computers outside homes

Single-board computers are found almost everywhere, where there is a screen to interact with. One of the biggest areas, where single-board computers are used is for gaming, like slot machines and video pokers.

They also can be found in more professional places, like:

- ATM machines
- Industrial computers
- Medical equipment
- Automation equipment
- Cash registers and kiosks

Because they have an ability to also run a screen they are much easier to use for such applications, than microcontrollers. Making them perfect for such applications because they are inexpensive and much smaller compared to personal computers, at the same time have much more functionality than the microcontrollers for purposes like these [11].

Because the added flexibility, they are also much more complicated and prone to outside stimuli, like operating system failures and hackers.

2.2.2.2 Single-board computers for hobbyists and home appliances

Where the Single-board computers shine, as mentioned earlier is its flexibility. The users can create everything they can imagine with these. For example a mirror, that can output all the information you desired or portable lockbox that can be opened with a NFC card. The lockbox project will also be go more into depth later to bring out the strengths of single-board computers.

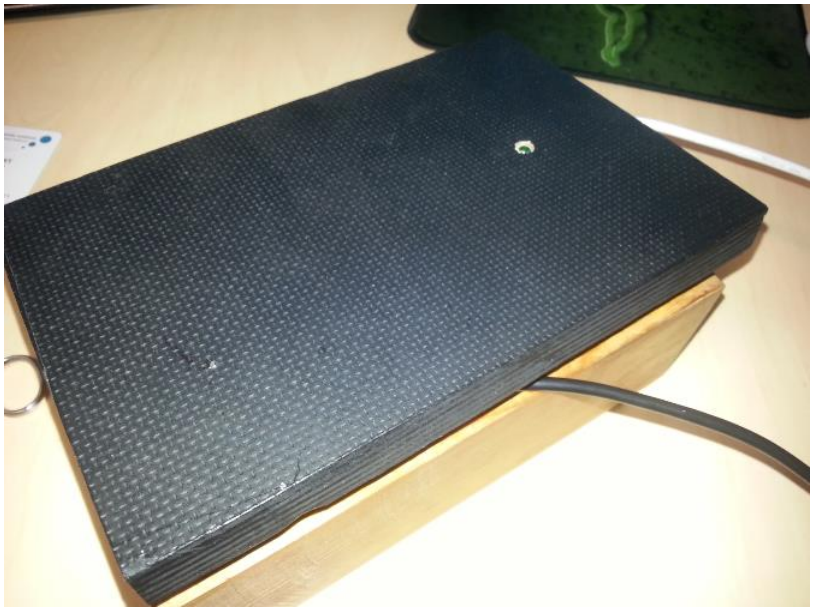

Figure 6 Raspberry PI Lockbox

Figure 7 Smart Mirror [27]

Figure 8 smart coffee machine [26]

As for home appliances, everything from a coffee machine to refrigerator are also usually built on single board computers. They give us a way to interface with everything around us in a simple way, because most of the single board computers are also connected to Wi-Fi they can also interact with everything else in the house, making it easier for the home owner and removing tediousness for those who look for it. For example, the owner wakes up, the blinds start opening and hot coffee made from fresh beans. Meanwhile the owner can go wash and prepare while hearing news and seeing his day plan laid out on the mirror. When he exits his home the car will drive in front of the house, warmed and ready to go. As the owner gets further from home the house will enter secure mode, locking all the doors and adjusting climate controls to save power while no one is home. When the owner starts returning to home the house starts to warm up. The car can enter the garage by itself and start charging automatically so it would be ready for the next day. This is the power single-board computers have and what kind of future they can lay out.

2.2.3 Comparing different Single-board computers

In this section most popular and widely used Single-board computers will be compared according to EEtimes [13], which is a popular online electronics industry magazine. Since they are a little different than microcontrollers mostly the same things will be compared. As the article is from 2015, the tech may be a little out dated on some level.

- Data bus : What is the size of instructions and data used
- CPU/Clock speed : At what frequency can the processor work at
- GPU/Clock speed At what frequency does can the microcontroller work at
- Memory and size: Type of memory used and size.
- I/O : What kind of I/O it has, what can it do
- Serial communication : What kind of communication is available
- Power consumption : How much power consumes, power states
- Cost
- Additional

Table 4 Comparing Single-board computers

Single-board computer	Bus width	CPU & clock speed	GPU & clock speed	Memory	I/O
Raspberry Pi 3B+	64 bit	ARM 1.4 GHz	Broadcom Videocore-IV 400Mhz	1GB DDR2	40 pins
CHIP	32 bit	ARM 1 GHz	Optional VGA	512MB DDR3	Up to 45 pins
Orange PI Lite	64 bit	ARM 1.3GHz	Mali400MP2 GPU 600MHz	512MB DDR3	40 pins
HummingBoard-Gate	64 bit	ARM 1GHz	Vivante GC880/GC2000 500MHz	Up to 2GB DDR3	36 pins
BeagleBone Green	32 bit	ARM 1GHz	Not specified	512MB DDR3	2 x 46 pin
Galileo Gen 2	32 bit	Intel 400MHz	Not specified	256 MB DDR3	20 pins
Gizmo 2	32 bit	ARM 1GHz	300MHz	1GB DDR3	Via peripheral
Fox SBC	32 bit	1GHz	Not Specified	1GB DDR2	8 pins

Table 5 Comparing Single-board computers

Single-board computer	Communication	Power consumption (A)	Cost	Additional
Raspberry Pi 3B+	Wi-Fi, BT4.2, Ethernet, UART, i2c, spi, 1-wire,csi,dsi	Idle 0.4A FL 0.69A	44,18€	85 mm × 56mm Power over Ethernet, huge community, SD card support.win10 support
CHIP	Wi-Fi, BT4.0, 1-wire, I2C, USB	Idle 0.25A FL 0.5A	9€	60 mm × 40 mm HDMI optional
Orange PI Lite	Wi-Fi, Ethernet, UART, i2c, spi, 1-wire, USB	1-2A	11€	69mm × 48mm Wi-Fi with antenna, SD card support
HummingBoard-Gate	Wi-Fi, BT, Ethernet, USB, RS485	FL 1.5A	188€	102mm x 69mm Less memory is cheaper
BeagleBone Green	I2C, UART, Ethernet, USB	Idle 0.3A FL 0.45A	36.77€	126mm x 76mm, SD card support, HDMI via Cape only, supports windows
Galileo Gen 2	SPI,UART, Wi-Fi,USB	FL 0.4A	37€	124mm x 72mm PoE with module, good for alarms
Gizmo 2	USB, 1Gbit Ethernet, SPI,	FL 2A	166€	101.6mm x 101.6mm HDMI video/audio, DX11.1 support, windows support. Have to use expansion ports
Fox SBC	RS-232, USB, Wi-FI, PCIe, mSATA, Ethernet		496 €	108 mm x 96 mm Can work in harsh conditions, supports PCI and ISA expansion

Table 6 Single-board computers comparison summary

SBC	Raspberry Pi 3B+	Fox SBC	CHIP
Bus width	64-bit	32-bit	32-bit
Communication Protocols	UART, Bluetooth 4.2, Ethernet, CSI, DSI, I2C, SPI, 1-wire	SATA, Ethernet, PCIe, Serial I/O (RS232/422/485), SPI,	Wi-Fi, Bluetooth 4.0
Community support	Very good Has a lot of projects online	Very small community, mostly meant for industrial. Hard to find anything online.	Good, developers are adding more, but long way from Raspberry. Poor documentation for now.
Instruction set architecture	RISC	RISC	RISC
Purpose	A low cost computer that enables people of all ages to explore computing and learn how to program in any language	A rugged SBC that can operate in full industrial temperature range. Excellent for industrial and medical applications that need low-power solution.	A very cheap and very small SBC with limited interface but easily upgraded to support more, such as HDMI. Small form factor is excellent for small scale projects.

In table 6 the most famous, cheapest and the most expensive SBC-s are compared, to get a basic overview. From all the SBCs compared the Raspberry Pi is a clear winner when it comes to flexibility, community support and price. But picking the right one all comes down to the requirements. Raspberry Pi could never survive the conditions that the Fox SBC has been designed for and it could not fit into very small projects, like the CHIP can. As it comes down to the users requirements there is no clear winner, but for household appliances and hobbyists, the Raspberry Pi is the best pick out of all the others.

2.3 Single board computers compared to microcontrollers

In the following Table 7 SBC-s will be compared to MCUs, to see which one is more flexible and functional. Thins compared will be

- Architecture: What kind of architecture are they built on
- Size: Size of the system
- Cost efficiency/power: How much current draws
- Ease of use: How easy it is for beginners or non-specialists to use.
- Adaptability: How functional and flexible it is.
- Price: Cost of the system

Table 7 Main differences between SBC and Microcontrollers

	Single-Board computer	Microcontroller
Architecture	CPU (APU), GPU, external memory, secondary storage	Single chip that contains CPU, memory and flash
Size	Bigger, since is put on a board and requires external add-ons to work	Single chip, small
Cost efficiency/power	Depending on the SBC chosen can use everywhere from 0.25-2A	Very power efficient. Usually works within uA
Ease of use	Can be used like a regular computer, Linux a bit more complicated than windows. Has a huge flexibility of things it can do.	Requires programming language knowledge to use and program.
Adaptability	Can run many different processes and scripts, but much slower. Has a lot of expansion boards and circuits can be built around. Has an option to have a screen and audio	Can do one thing really well (Single purpose), Any circuit required can be built around it, can run a primitive screen.
Price	Depending on what to pick. Raspberry pi 3B+ is 44€ and more extreme editions can go up to couple of hundred.	Usually very cheap, couple of euros

If it is flexibility, the Single-Board computers are an obvious choice, but if it comes to doing one thing, really well Microcontrollers are a better choice. Picking the right one also comes down to how capable the engineer is as microcontrollers are usually harder to work with, since they also need a specialised circuit around it.

3 Project “Lockbox”

The purpose of the following project is to choose between microcontroller and single-board computer and decide which one is more powerful. When taking comparison tables above into account it turns out that single-board computers are much more flexible and powerful overall. Because of that the following project will be programmed and built around a single-board computer. This will also allow to explore all the different things that can be done with a SBC and would be impossible on the microcontroller.

To show the power of Single-board computers a project called “Lockbox” was made. The purpose of the project was to build an easy to carry safe that could be opened with a NFC tag. The box had to recognize when a tag was introduced it to first time and be exclusive to that one tag. Since the Raspberry Pi, from earlier comparisons turned out to be the best out of others for this kind of projects, it was picked.

3.1 Hardware

The hardware used is a little dated. Instead of the latest Raspberry Pi 3B+ Raspberry Pi 2B is used. The differences between the two are small.

Table 8 Raspberry Pi 2B compared to Raspberry Pi 3B+

	Raspberry Pi 2B	Raspberry Pi 3B+
CPU	900 MHz 32-bit	1.4 GHz 64-bit
GPU	250 MHz	500 MHz
Memory	1GB	1GB

Communication	Needs dongle for Wi-Fi and Bluetooth	Wi-Fi and Bluetooth 4.2

As the processing power needed is more than enough on the Raspberry Pi 2B, it is perfect for the project.

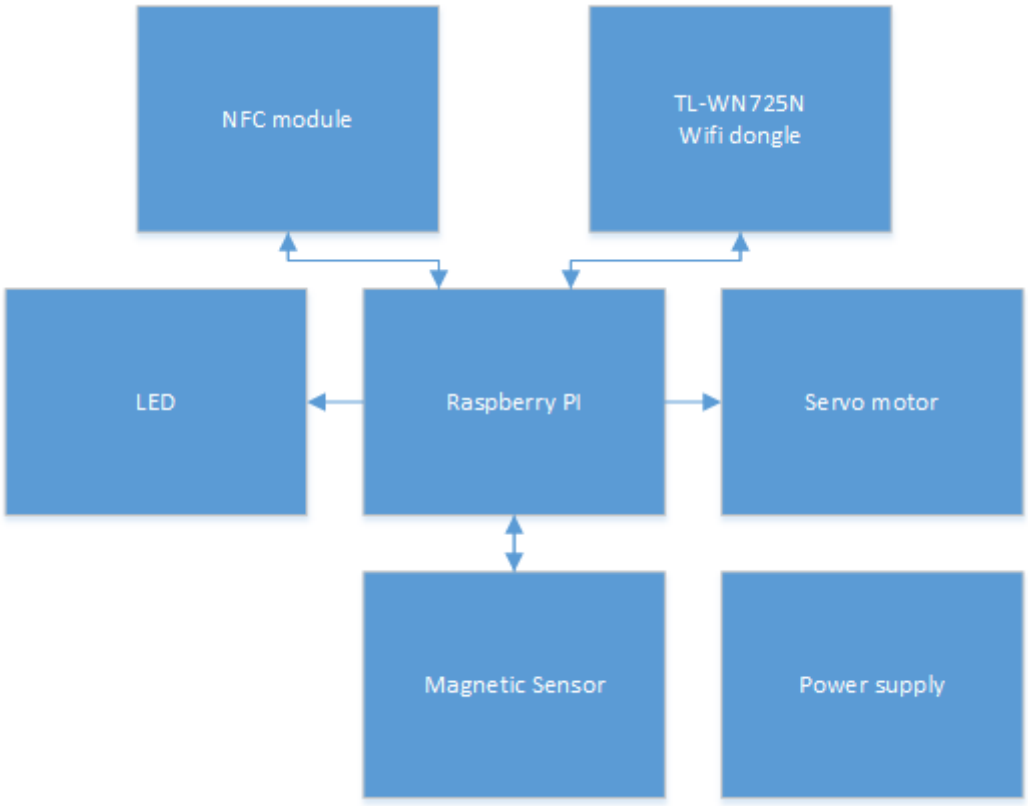

Figure 9 Hardware block diagram

In Figure 10 it is possible to see the block diagram of the project and how everything communicates.

1. NFC module : ITEAD PN532 NFC reader, which reads the UID of a NFC card
2. Raspberry PI : Center of the system, controls modules and runs the algorithm
3. Servo motor : Servo motor that is used as a self made locking mechanism
4. Magnetic Sensor : Magnetic sensor to see if the box is open or closed
5. Power supply : Battery to make the box portable
6. LED : To signal if the box is open
7. TL-WN725N Wi-Fi adapter: Wi-Fi dongle, what is used so the user could access the Admin part of the project and also use a site from where to reset user and password.

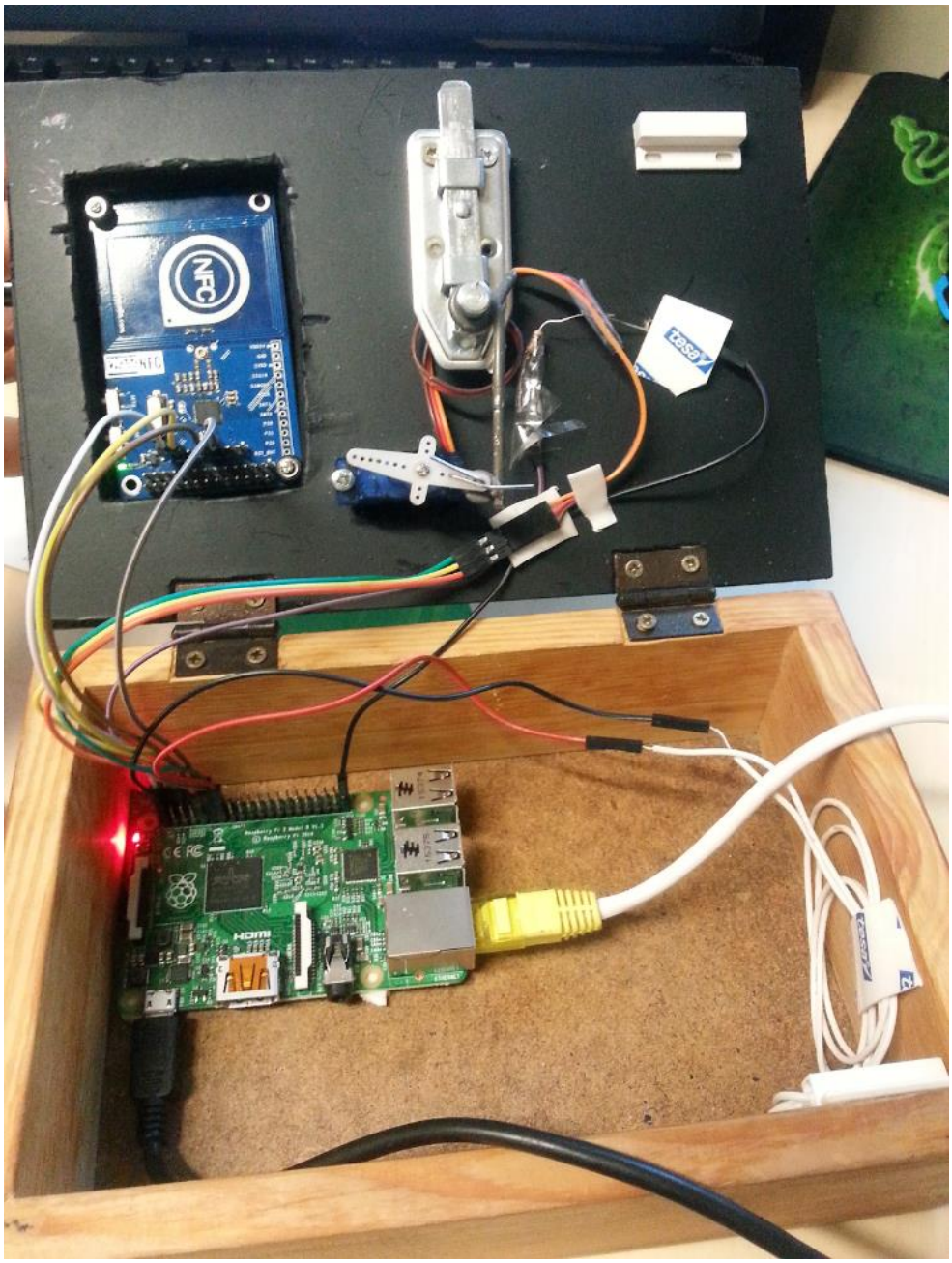

Figure 10 Lockbox open

Since the idea was also to keep it as cheap as possible, a lot of the box is hand made. The box itself is made of wood, but is hard enough to sustain drop damage. The locking mechanism consists of a servo and a slide lock. Much cheaper than an electric lock but not as secure.

3.2 Software

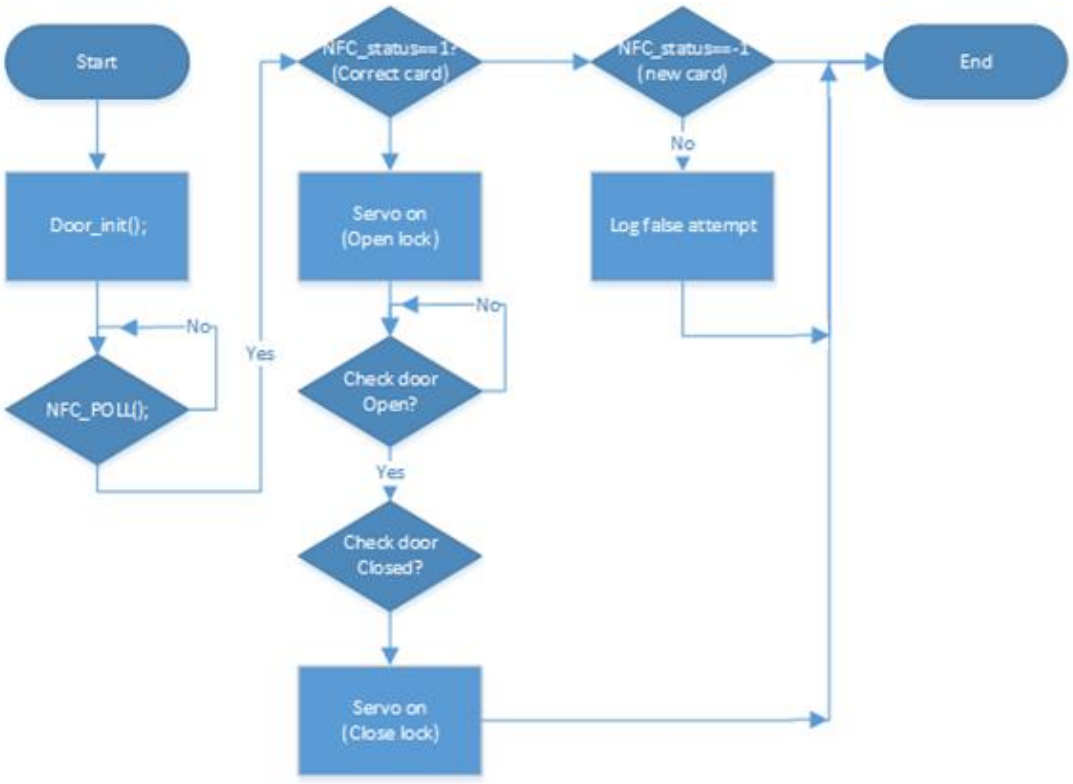

Figure 11 Software logic diagram

Door_init() : Initializes everything needed for the Raspberry PI

NFC_POLL(): Polls the NFC reader, until a response is read

Servo(): Used to control the locking mechanism

Check_door(): Checks the status of the box, is it open or closed.

Raspberry has a huge community in C making it very convinient to use for hardware programming, which made it the best choice to use.

The community also supports a huge varierty of libraries. Most of the code for the project was self written, but to save some time a library to read the NFC cards was used [14]. Another library that make it convinient to use Raspberry PI is the WiringPi library [15]. This library is for controlling the GPIO on the controller.

The most important function is called NFC_POLL(). It is a function that is constantly polling (every 30ms). The reason behind that is to quickly read the card once it is brought to the reader. When a card is read, three of the following things can happen

1. Card has been authenticated, LED will turn on and box opened
2. Memory is empty, so the card is read and made as default
3. Wrong card is read, box stays locked

3.3 Web interface

Thanks to the flexibility of the Raspberry Pi, it is possible to also run a Web interface at the same time, which made it easy to use for the lockbox project as well as a way for users to change their NFC cards via password.

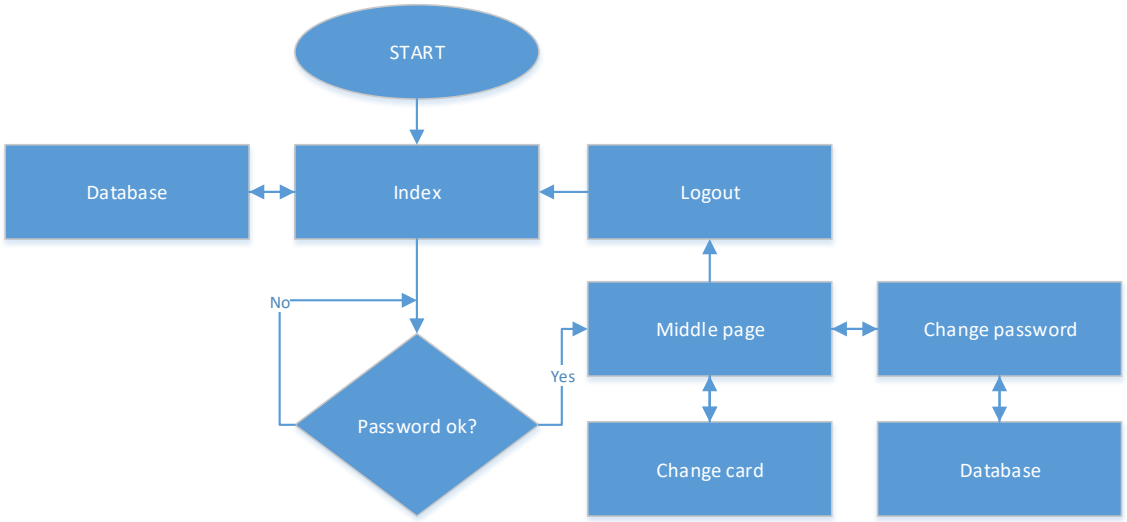

Figure 12 Web interface flow

Index: Logging in being session

Middle page: just a middle landing page

Logout: Ends the session

Change password: User can change the password, updates the database

Change card, User can change the NFC card

Database: Holds the Account name and Password

3.4 Project “Lockbox” conclusion

Thanks to the flexibility of the Raspberry Pi a lot of different things could be brought together to form the project. This really shows the functionality of the Single-board computers as well. A lot of the functions possible to use, were used: a combination of hardware and software working together in unison to achieve one goal, a secure lockbox that could only be opened with a NFC card. With a 10 000mAh battery the Lockbox could be portable up to 17 hours, which is not perfect, but enough to transport something very important from point A to B, or to use as a safe at home that has a backup power source when the battery is cut off.

4 Summary

The purpose of this bachelor's thesis was to give an overview between the differences of microcontrollers and single-board computers. Both the microcontrollers and single-board computer functions were introduced, compared between their own kinds and finally compared between each other to find the need for one or the other. At the end a sample project is made to bring out the strengths of one of them.

At the heart, the purpose is to see which is superior to be used at home appliances. To find that out research is made to find the strengths of both and where are they used most commonly. Since both kind of devices have a wide variety of uses a smaller focus groups are brought out that are more interesting, such as making life more convenient and home automation.

From comparing the both it is found that microcontrollers are usually much cheaper, smaller and faster at single purpose, but single-board computers are more flexible and have an ability to run different processes and scripts simultaneously. This gives users looking to run their security systems and other autonomous technologies a huge edge over microcontrollers.

The thesis consists of a written and a practical part. The practical part is meant to show the power of the single-board computers. A simple project called "Lockbox" is built and programmed to see the flexibility first-hand. It was found out that the process of the project is much easier on the single-board computer at the cost of power efficiency. Because of the size of the microcontrollers, the project would could be done on a much smaller form factor as well.

In Chapter 3, the project can be seen, built around Raspberry PI 2B. An outdated version of the series, but more than enough to build the project on. The project used ITEAD PN532 NFC reader that constantly polls, waiting for a card to be read. When the required outcome was read, the lockbox reacted by turning on a led and opening the box.

Building the prototype was a time consuming process itself and could be made to look much nicer via 3D printing job. Putting aside the aesthetic, functionality the and power of the single-board computers is seen and proven

References

[1] Intel 4004 [WWW] https://www.intel.com/content/www/us/en/history/museum-story-of-intel-4004.html

[2] Special purpose computer [WWW] https://www.elsevier.com/books/special-purpose-computers/alder/978-0-12-049260-2

[3] MCU uses [WWW] https://www.techwalla.com/articles/what-things-use-a-microcontroller

[4] SBC [WWW] https://en.wikipedia.org/wiki/Single-board_computer

[5] Raspberry pi [WWW] https://www.adafruit.com/category/105

[6] MCU structure [WWW] http://www.circuitstoday.com/basics-of-microcontrollers

[7] MCU GPIO [WWW] https://www.mikroe.com/ebooks/architecture-and-programming-of-8051-mcus/input-output-ports

[8] IoT [WWW] https://www.digikey.gr/en/articles/techzone/2011/mar/microcontrollers-and-wireless-connectivity-in-smart-appliances

[9] IoT appliances [WWW]https://www.digikey.gr/en/articles/techzone/2011/mar/microcontrollers-and-wireless-connectivity-in-smart-appliances

[10] Tesla auto charger. [WWW] https://www.youtube.com/watch?v=uMM0lRfX6YI

[11] SBC purpose [WWW]https://www.eetimes.com/document.asp?doc_id=1325095&page_number=2

[12] Top SBC [WWW]https://www.eetimes.com/document.asp?doc_id=1328008&page_number=1

[13] Rpi superiority [WWW] https://raspberrypi.stackexchange.com/questions/8452/what-is-so-special-about-pi?utm_medium=organic&utm_source=google_rich_qa&utm_campaign=google_rich_qa

[14] Raspberry Pi NFC library [WWW] http://nfc-tools.org/index.php?title=Libnfc

[15] Wiring Pi library and guide [WWW] http://wiringpi.com/

[16] NFC [WWW] https://en.wikipedia.org/wiki/Near_field_communication

[17] UARTi [WWW] http://home.roboticlab.eu/et/examples/communication/uart
[18] RFID [WWW] https://en.wikipedia.org/wiki/Radio-frequency_identification
[19] DNS [WWW] https://et.wikipedia.org/wiki/Nimeserver
[20] IP-aadress [WWW] https://et.wikipedia.org/wiki/IP-aadress
[21] Wi-FI [WWW] https://en.wikipedia.org/wiki/Wi-Fi
[22] WLAN [WWW] https://et.wikipedia.org/wiki/Traadita_kohtv%C3%B5rk
[23] MCU I/O [WWW] https://www.mikroe.com/ebooks/pic-microcontrollers-programming-in-c/inputoutput-ports
[24] Arduino [WWW] https://learn.sparkfun.com/tutorials/what-is-an-arduino
[25] Raspberry Pi 2 [WWW] https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/
[26] Smart coffee machine [WWW] https://www.walmart.com/ip/Mr-Coffee-10-Cup-Optimal-Brew-wi-fi-controlled-Smart-Coffee-Maker-enabled-by-WeMo-BVMC-PSTX91WE/39816866
[27] Smart mirror [WWW] https://www.uplabs.com/posts/smart-mirror-ar

Appendix 1 – Code for project “Lockbox”

1.1 Main.c

```
#include <stdlib.h>
#include <stdio.h>
#include <wiringPi.h>
#include "header.h"

main()
{
	int NFC_status=0;
	door_init();
	while(1)
	{
	do
	{
		NFC_status=NFC_POLL();
		if(NFC_status==-1)
		{
			printf("New tag inserted");
			break;
		}
		else if(NFC_status==1)
		{
			printf("/ndoor open/n");
			//servo open
			servo(1);
			break;
		}
		else
			printf("False attempt");
	}
	while(check_door()==1);
	if(NFC_status==1)
	{
		while(check_door()==1);
		while(check_door()==0);
		delay(1000);
		servo(0);
	}
	//servo close
	}
	return 0;
}
```

1.2 Door.c

```
#include <wiringPi.h>
#include <stdio.h>
#include <stdlib.h>

#include "header.h"

#define door_sensor 2

int door_init();
int check_door();

int door_init()
{
	int init_flag = 0;
	if(init_flag == 0)
	{
	wiringPiSetup();
	pinMode(door_sensor, INPUT);
	init_flag=1;
	pinMode(1, PWM_OUTPUT ) ;
	pinMode(green_LED, OUTPUT);
	softPwmCreate(1, 0, 200);
	delay(500);
	}
	return init_flag;
}
int check_door()
{
	return digitalRead(door_sensor);
}

}
```

1.3 Servo.c

```
#include <stdlib.h>
#include <stdio.h>
#include <wiringPi.h>
#include "header.h"
#include <softPwm.h>
#include "servo.h"

#define green_LED 0


void servo(int status)
{

	//pinMode(18, OUTPUT) ;
	//sisend on 1 siis on kõrval (LAHTI)
	if (status==1){
	  softPwmWrite  (1, 10) ;
	  digitalWrite(green_LED, HIGH);
	  delay(500);
```

```
      }
      //sisend on muu siis on all; ()
      else
      {
        softPwmWrite (1, 15);
        digitalWrite(green_LED, LOW);
        delay(500);
      }
}
//5 IS ALL THE WAY RIGHT
//15 IS ALL THE WAY LEFT
```

1.4 NFC.c

```
#include <stdlib.h>
#include <stdio.h>

#include <nfc/nfc.h>
#include "NFC.h"

//Code is deviated from libnfc functions.

FILE *fp;

int NFC_POLL()
{
      int status;
      nfc_device *pnd;
      nfc_target nt;

        // Allocate only a pointer to nfc_context
        nfc_context *context;

        // Initialize libnfc and set the nfc_context
        nfc_init(&context);
        if (context == NULL) {
      printf("Unable to init libnfc (malloc)\n");
      exit(EXIT_FAILURE);
  }

  // Open, using the first available NFC device which can be in order
of selection:
  //   - default device specified using environment variable or
  //   - first specified device in libnfc.conf (/etc/nfc) or
  //    -  first  specified  device  in  device-configuration  directory
(/etc/nfc/devices.d) or
  //   - first auto-detected (if feature is not disabled in libnfc.conf)
device
  pnd = nfc_open(context, NULL);

  if (pnd == NULL) {
    printf("ERROR: %s\n", "Unable to open NFC device.");
    exit(EXIT_FAILURE);
  }
  // Set opened NFC device to initiator mode
  if (nfc_initiator_init(pnd) < 0) {
```

```
    nfc_perror(pnd, "nfc_initiator_init");
    exit(EXIT_FAILURE);
  }

  printf("NFC reader: %s opened\n", nfc_device_get_name(pnd));

  // Poll for a ISO14443A (MIFARE) tag
  const nfc_modulation nmMifare = {
      .nmt = NMT_ISO14443A,
      .nbr = NBR_106,
  };
  if (nfc_initiator_select_passive_target(pnd, nmMifare, NULL, 0, &nt)
> 0) {
   //printf("     UID (NFCID%c): ", (nt.nti.nai.abtUid[0]));
    status = tag_check(nt.nti.nai.abtUid, nt.nti.nai.szUidLen);
  }
  // Close NFC device
  nfc_close(pnd);
  // Release the context
  nfc_exit(context);
  printf("\nStatus: %d\n", status);//Outputs status into the console.
  return status;
}

static void print_hex(const uint8_t *pbtData, const size_t szBytes)
{
      //For printing hex value of the UID into the console (Testing
purposes);
      size_t  szPos;
      fp = fopen("NFC_inf.txt", "r+");
      rewind(fp);
      for (szPos = 0; szPos < szBytes; szPos++)
      {
      printf("%02x ", pbtData[szPos]);
      fprintf(fp, "%02x", pbtData[szPos]); //Prints UID to a file
      }
      fclose(fp);
      printf("\n");
}
static int tag_check(const uint8_t *pbtData, const size_t szBytes)
{
      //Controls the file, if the file has something written into it
checks the UID. If the UID matches with
      //the tag that is used the box will open (Status=1). If the file
is empty, the tag is saved (status=-1).
      //If the tag used is wrong (status=0) is returned and the box
stays locked.
      int tag_flag=0;
      char TmTag[7];
      size_t  szPos;
      fp                                                              =
fopen("/home/pi/coding/libnfc/projects/Varalaegas/code/NFC_inf.txt",
"r+");
      fseek(fp, 0, SEEK_END);
      if (ftell(fp)>0)
      {
      rewind(fp);
      for (szPos = 0; szPos < szBytes; szPos++)
      {
            fscanf(fp,"%02x", &TmTag[szPos]);
            if(TmTag[szPos]!=pbtData[szPos])
```

```
                tag_flag=0;
            else
                tag_flag=1;
        }
        }
        else if(ftell(fp)==0)
        {
        for (szPos = 0; szPos < szBytes; szPos++)
            fprintf(fp, "%02x", pbtData[szPos]);
        tag_flag=-1;
        }
        else
        printf("Tag read failed");

        fclose(fp);
        printf("\n");
        return tag_flag;
}
```

Appendix 2 – Script for project “Lockbox”

```
#!/bin/sh
# kFreeBSD do not accept scripts as interpreters, using #!/bin/sh and sourcing.
if [ true != "$INIT_D_SCRIPT_SOURCED" ] ; then
    set "$0" "$@"; INIT_D_SCRIPT_SOURCED=true . /lib/init/init-d-script
fi
### BEGIN INIT INFO
# Provides:          varalaegas
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Example initscript
# Description:       This file should be used to construct scripts to be
#                    placed in /etc/init.d.  This example start a
#                    single forking daemon capable of writing a pid
#                    file.  To get other behavoirs, implemend
#                    do_start(), do_stop() or other functions to
#                    override the defaults in /lib/init/init-d-script.
### END INIT INFO

# Author: Daaniel Ipsberg

DESC="Runs projects varalaegas on boot"
DAEMON=/usr/sbin/main
```