FLINDERS UNIVERSITY

HONOURS THESIS

8-bit Innovation: The challenges facing community-based and open-source retro-computing projects

---

Author:
Tom AYLEN

Supervisor:
Dr. Paul GARDNER-STEPHEN

Submitted to the
College of Science and Engineering
in partial fulfilment of the requirements for the degree of
Bachelor of Engineering (Software) (Honours)
at Flinders University - Adelaide, Australia

May 27, 2019

Declaration of Authorship

I certify that this work does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

Signed:

Date:

"Everything should be made as simple as possible, but not simpler"

- Common saying

FLINDERS UNIVERSITY

Abstract

College of Science and Engineering

Bachelor of Engineering (Software) (Honours)

8-bit Innovation: The challenges facing community-based and open-source retro-computing projects

by Tom AYLEN

The children who grew up through the home computer revolution of the 1980s are now established in life, and frequently willing to invest time and money in projects and products that have nostalgic value for them. This includes retro-computing projects that re-create the iconic home computers of the 1980s and early 1990s.

There have been a number of community and commercially driven retro-computing projects, such as the Spectrum Next, Spectrum Vega, the C64mini and the C64DTV among many others. The scope and ambition of these projects varies, as has their success. Some, like the C64DTV were commercial successes, while others, such as the Spectrum Vega Plus have suffered long drawn out deaths due to a variety of problems or misadventures.

The purpose of this research is to examine a number of these as case-studies, and to learn from them, so that other such projects can have an increased likelihood of success, and the participants can avoid some of the more unpleasant misadventures that have arisen, especially those involving hostile lawyers. This thesis acts to document the steps and stages of productization and market release for such projects, and then applies these learnings directly to the MEGA65 retro-computing project at Flinders University, to accelerate its approach to market, and to help it avoid the pitfalls of past failures. The learnings presented in this thesis are designed to be of use to any retro-computing project, and it is our hope that it will enable other projects to also have greater success, and less pain in the process.

Acknowledgements

I would like to thank my gorgeous wife, Jade, for her love and support. My Daughter, Eleanor, for going to kindy even when she didn't want to. Dr. Paul Gardner-Stephen for his guidance, encouragement and excellent life tips during my thesis project. And all the giants for allowing me to stand on their shoulders; the view is amazing.

Contents

List of Figures

List of Tables

List of Abbreviations

ASIC	**A**pplication-**S**pecific **I**ntegrated **C**ircuit
BASIC	**B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode
COBOL	**CO**mmon **B**usiness **O**riented **L**anguage
CPLD	**C**omplex **P**rogrammable **L**ogic **D**evice
CP/M	**C**ontrol **P**rogram/**M**onitor
CPU	**C**omputer **P**rocessing **U**nit
FPGA	**F**ield-**P**rogrammable **G**ate **A**rray
IC	**I**ntegrated **C**ircuit
IDII	**I**nteraction **D**esign **I**nstitute **I**vrea
IDE	**I**ntegrated **D**evelopment **E**nvironment
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
LAN	**L**ocal **A**rea **N**etwork
MEGA	**M**useum of **E**lectronic **G**ames and **A**rts
MITS	**M**icro **I**nstrumentation and **T**elemetry **S**ystems
MOS	**M**etal **O**xide **S**emiconductor
NTSC	**N**ational **T**elevision **S**ystem **C**ommittee
ROM	**R**ead **O**nly **M**emory
SIM	**S**ubscriber **I**dentity **M**odule
SLOC	**S**ource **L**ines **O**f **C**ode
PAL	**P**hase **A**lternating **L**ine
PC	**P**ersonal **C**omputer
USB	**U**niversal **S**erial **B**us
USD	**U**nited **S**tates **D**ollar
VHDL	**V**HSIC **H**ardware **D**escription **L**anguage
VHSIC	**V**ery **H**igh **S**peed **I**ntegrated **C**ircuit

Dedicated to my awesome family…

Chapter 1

Introduction

1.1 Background

A retro-computing project is a computing project that is in some way inspired or derived from a home computer the was released between about 1975 and 1990 [2, 3]. This period is referred to as the early home computer era and it was the first time computers where specifically marketed for personal use within the home. This lead to many people having their first interaction with a computer during this period. Many people whom lived through this period have strong memories of the products they used, such as the Commodore 64, the ZX Spectrum and the BBC Micro.

For various reasons, many people have tried, with varying amounts of success, to revive certain computer systems from the period, either by emulating the system in software on a more powerful computer, or by rebuilding the physical circuitry. Some retro-computing projects have been inspired by the home computers of the period being simpler and having less abstraction between the user and the mechanics of the computer. The MEGA65 is a retro-computing project which is currently under development and which aims to innovate the never-released Commodore 65 [4].

1.2 Scope of thesis

This thesis aims to provide a body of knowledge on the process, as well as the challenges and risks associated with retro-computing projects when trying to bring new products to market.

The general product development process has been widely explored from many different angles by various authors over time, e.g., [5, 6, 7, 8, 9, 10, 11, 12, 13]. The general product development concept and process is beyond the scope of this thesis, which instead focuses on the issues particular to retro-computing projects.

1.3 Current situation and how it can be improved

Many retro-computing projects have brought products to the market, with varying amounts of success. The C64 Mini is a Commodore 64-inspired game console which is available for purchase from retail outlets within Australia and other countries. It

received satisfactory reviews and can be considered a successful project by many metrics. Comparatively, the Vega Plus, a hand-held game console inspired by the Spectrum ZX, was an abject failure in regard to project outcomes for stakeholders. The widely varying processes and methods employed by differing projects suggests the outcomes may be more consistent and improved if a more rigorous process was followed.

There is currently no body of knowledge specific to retro-computer projects and the productization process. If a body of knowledge could be formed it should allow future projects to achieve a more consistent outcome. This thesis aims to be the start point for that body of knowledge and it is hoped further research will follow.

Part of this thesis is to highlight the common challenges and risks that retro-computing projects are likely to be exposed to. This list of retro-computing specific risks could then be used by retro-computing projects to help evaluate their risk exposure. With a risk evaluation undertaken and the high risk areas identified, the retro-computing projects should be able to determine and enact strategies to reduce their risk exposure, this should then have the effect of improving the outcomes of the project.

1.4 Research questions

The questions this thesis aims to answer are:

1. What does the retro-computing project productization process look like?
2. What risk and challenges are associated with a retro-computing project?
3. What is the MEGA65 project?
4. How exposed to risks was the MEGA65 project in July 2018?
5. What can be done to reduce the MEGA65 project's risks?
6. Did the MEGA65 project reduce its risk after 10 months?

1.5 Hypothesis

It is hypothesised that the body of knowledge created as part of this thesis, which includes a definition of the process to productization of a retro-computing project as well as its associated risks and challenges, will allow future retro-computing projects to achieve more desirable outcomes more consistently.

1.6 Methodology

The method which was followed to create the body of knowledge and answer the research questions is described here. A diagram is also shown to help illustrate the

methodology process and how the significant segments of the thesis interact, figure 1.1.

First, an in-depth case-study was conducted into several retro-computing projects. Because there is a lack of peer-reviewed sources for this information, the majority of this information was sourced from websites and other publications. From these case-studies, a process was distilled and recorded. The challenges that beset each project were recorded and then categorised into a list of risks.

As a way to assess the utility of the list of identified risks, a retro-computing project is evaluated twice over a 10 month period, against the identified risks. This project, called MEGA65, is a retro-computing project which is currently in development, making it an ideal candidate to evaluate.

After the first evaluation, the MEGA65 project is provided with advice and strategies on how to reduce their risk exposure in the high risk areas. After ten months, the MEGA65 project is evaluated again and the results compared to determine if the MEGA65 project's risk expose has changed.

To evaluate the MEGA65 project effectively, it first needed to be understood, this process involved researching the state of the MEGA65 project in July 2018, as well as researching the intended products to be released by the MEGA65 project. The products intended to be released are captured with a use-case study which looks at ways the users will interact with the products. To understand the state of the MEGA65 and its development, most of the information was conveyed via conversations with the MEGA65 team members. This is due to the nature of the development process not allowing time for documentation to be created with all of the relevant details.

1.7 Contributions

The contributions this thesis makes to retro-computing projects is categorised as follows:

1. Creates a body of knowledge exploring the retro-computing project productization process.
2. Creates a body of knowledge discussing the risks and challenges associated with retro-computing projects.
3. A risk evaluation of a specific retro-computing project, the MEGA65.
4. Practical, actionable recommendations to reduce the MEGA65 project's risk exposure.
5. Evidence of the benefit of those recommendations, gathered through a follow-up risk evaluation of the MEGA65 several months after providing the recommendations.

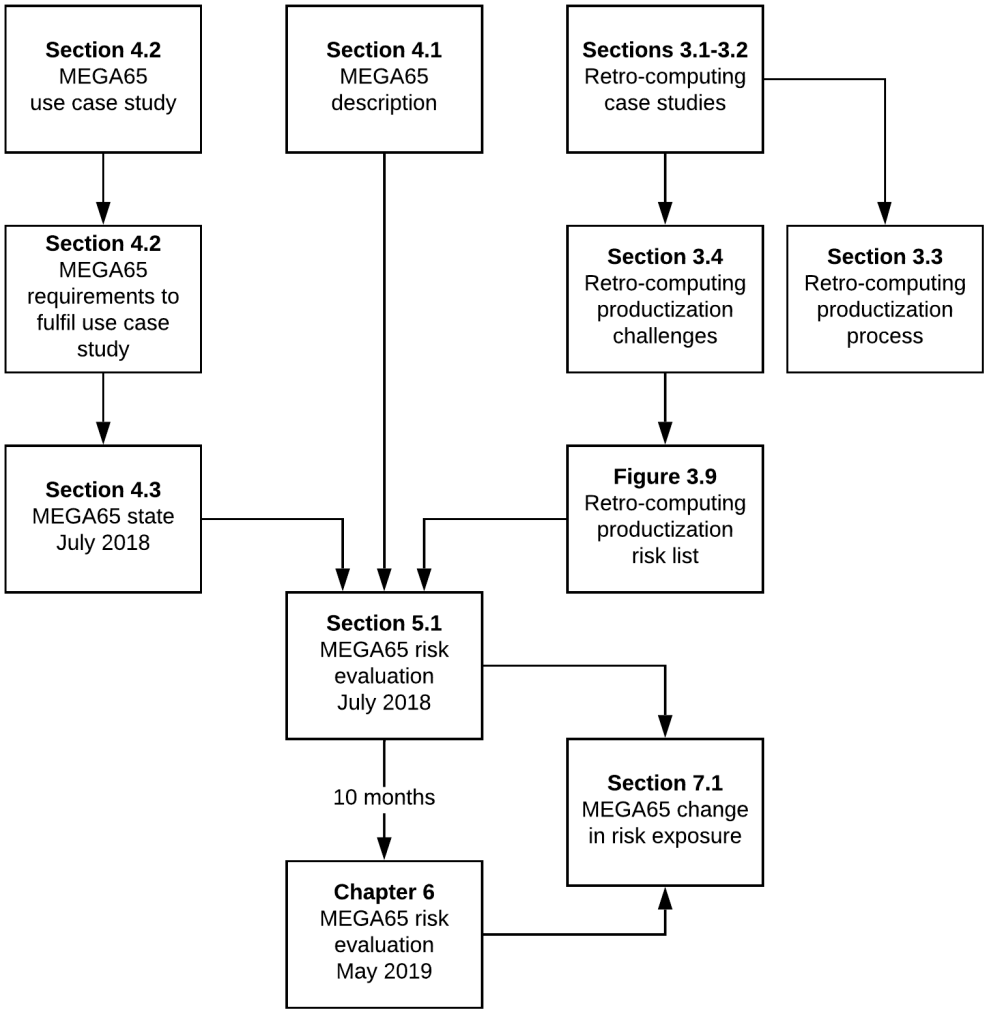

FIGURE 1.1: Methodology flow diagram showing sections of this thesis and how they interact

1.8 Structure of thesis

Following this introductory chapter, there is a literature review in Chapter 2, which focuses on the early home computing era, the growing complexity of computer hardware and software and its relationship to insecurity in modern computers.

Chapter 3 consists of case-studies into several retro-computer projects with a focus on the process they followed and the challenges they faced in releasing a new product. Also, there is a discussion and synthesis of the productization process illuminated from the case-studies. The challenges extracted from the case-studies are then categorised into risks and these risks discussed and rated in terms of potential damage to retro-computing projects.

Chapter 4 is dedicated to explaining the MEGA65 project, its intended products and their state of development as of July 2018. This explanation takes the form of a description of the various aspects of the MEGA65 project, including the project's team characteristics, funding model and the products they are creating. Following the description of the MEGA65, there is a use case study conducted into the MEGA65 project's intended products, the MEGAphone and the desktop form-factor of the MEGA65, also called the MEGA65.

A risk evaluation of the MEGA65 project in July 2018, provides the bulk of Chapter 5. Following the evaluation, the identified high-risk areas are discussed and advice is provided to the MEGA65 project on how to reduce their risk exposure.

Chapter 6 contains the second risk evaluation, which was undertaken 10 months after the first. This risk evaluation only looks at the high-risk areas identified in the first evaluation.

A comparison of the two risk evaluations is found in chapter 7, *Results and discussions*. The research questions and hypothesis are re-considered, and a discussion of how this thesis provided answers to them is given.

Finally, Chapter 8 provides a conclusion and directions for future research.

Chapter 2

Literature Review

2.1 Early home computer era

The mid-1970s to late-1980s was an interesting period in the history of computers. It marked the first time computers were designed and marketed for personal use in the home; the early home computer era. The term *home computer* is ambiguous and not defined by any standard. In this thesis it is taken to be synonymous with a microcomputer or a personal computer (PC) from the era. A PC is classified in the *Proceedings of the IEEE* from 1984 by Gupta, A. and Toong, H. as a computer that fulfils all of the following characteristics [14]:

1. The computer cost less than $5000 USD at the time of sale.
2. The computing power is provided by a microprocessor from the era.
3. The computer is sold through mass-marketing channels.
4. The computer can run a variety of programs for different fields such as industry, business, education and at home; it is a general purpose computer, not designed for a single purpose or a single type of user.
5. The computer can handle at least one high-level language such as BASIC, FORTRAN, or COBOL.

Since the discovery of semiconductors in the 1940s, there have been continuous efforts and advances in making electronic devices smaller, more powerful and cheaper. 1958 saw the first working Integrated Circuit (IC, also called a microchip or chip), continuing this trend [15]. By 1965, Gordon E. Moore was talking about the observation that ICs are being manufactured with an increasing amount of components, which came to be popularly known as "Moore's Law" [16]. In 1971, Intel developed the first microprocessor; the 4-bit 4004 on a single IC [17], shown in Figure 2.1. A microprocessor is an entire CPU within an IC or a few ICs. The following year, Intel released the more powerful 8-bit 8008 microprocessor. Then in 1974 an even more powerful microprocessor was released by Intel, the 8080. [18].

It seems a tipping point had now been reached, the manufacturing cost of computers was low enough, their size was small enough and their performance was high

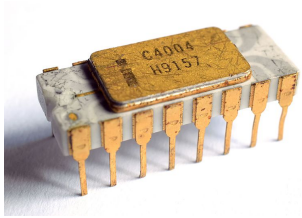

FIGURE 2.1: Intel 4004 microprocessor, an entire CPU on a single chip; the first microprocessor. Released in 1971.
Picture courtesy of Thomas Nguyen

FIGURE 2.2: MITS Altair 8800 home computer, the first market successful home computer. It used an 8-bit 8080 microprocessor and was released in 1974.
Picture courtesy of Michael Holley

enough that some manufacturers decided to start designing and marketing them to personal users. In 1974, MITS released what can be considered the first market-successful home computer [19], the Altair 8800, which used an 8080 microprocessor and is shown in Figure 2.2.

Over the next 15 or so years, a flurry of new companies sprung up offering a multitude of personal computers [20]. In the same period new microprocessors were developed, further increasing their performance and new processes were developed that reduced their cost to manufacture. The most notably of these new microprocessors is the 8-bit 6502 by MOS Technology, which when released in 1975 was the least expensive microprocessor on the market by a sizeable margin [21]. A sense of excitement and seemingly an expectation that computers were going to revolutionise society was abound at this time [22]. Many people were having their first interactions with computers, as the home computer became more widespread.

These early home computers had a relatively simple interface compared to modern computers. This meant there was far less abstraction between the user and the inner workings of the computer, this may have allowed their users to more readily understand the underlying mechanisms. It also meant that users had to learn at least some rudimentary programming skills to use them. Another way users were exposed to programming was from the possibility to acquire programs by typing them into their own computers out of magazines or from computer shows on TV and thus exposing them to source code. Most of the home computers in this period

ran some form of BASIC, which is a family of general-purpose high-level programming languages, all derived from the original BASIC language created at Dartmouth College in 1964 [23].

There were some people that questioned the usefulness of the early home computers, and with some merit, as there was a lack of software and distribution channels at the beginning of the era [24]. Others had unrealistic expectations of what computers would do for them, including doing tasks such as putting the rubbish out and babysitting. Home computers were mainly used in a four areas: business, science and engineering, education and in the home. Business, science and engineering uses included spreadsheets, word processing, basic graphics, databases and communication to connect to host computer or LANs. In the home the most widely reaching use of home computers was to play games [14]. These 8-bit home computers were where many people first met and became interested in the potential of computers and computer programming and they helped kick-off a revolution.

2.2 Commodore 64, 128, 65

2.2.1 Commodore 64

The Commodore 64 (C64) was one of the most successful home computers, selling over 17 million units according to Commodore International [25], the now defunct manufacturers of the Commodore 64. Earning it a Guinness World Record (formally Guinness Book of Records): 'Most computer sales' [26]. It was first sold in January, 1982, for $595 USD at launch [27]. The computational power came from a 8-bit MOS Technology 6510 microprocessor, a modified version of the 6502 mentioned in section 2.1. It had 64 kilobytes of RAM, which was the inspiration of its name. The Commodore 64 was a dominant market presence; in a category of influential home computers, the Commodore 64 may well have been the most influential of all. It ran a version of BASIC and there was a huge software library created for it during its lifetime and there's still new software being developed for the C64 to this day. [28][29][30].

2.2.2 Commodore 128

The Commodore 128 (C128) was Commodore International's evolution of the very successful Commodore 64. Released in January, 1985, It was meant to build on the success of the C64 while still being near 100% compatible with Commodore 64 software. It was powered by a more powerful 8-bit 8502 microprocessor and had 128 kilobytes of RAM. An innovation was the inclusion of a second microprocessor, an 8-bit Zilog Z80. This second CPU allowed the C128 to run the CP/M operating system as well as the Commodore BASIC environment similar to the C64. Running

CP/M allowed the C128 to access the CP/M software library, which was quite extensive. That coupled with the Commodore 64 software library gave the C128 one of the broadest ranges of software, compared to its competitors of the day [31].

2.2.3 Commodore 65

The Commodore 65 (C65) was in development in 1990-1991, when Commodore Business Machines (subsidiary of Commodore Intentional) went bankrupt in 1994. In the fallout of the collapse, a number of C65 prototypes were sold and are now a rare and valuable collectors item [32]. The C65 was planned to be an upgrade of the C64. *Compute! Gazette* reported in an article in 1989 that the C65 had a 16-bit 65816 microprocessor; a 16-bit version of the 6502 [33]. Years later, when the prototypes were sold on the market by liquidators, sources report a modified 65CE02 microprocessor was used instead [34] [35]. The 65CE02 was an 8-bit CPU with a limited ability to use 16-bit instructions. It had 128 kilobytes of RAM, expandable to 1 megabyte. The C65 had a "stunning" 640x400 pixel maximum resolution [33] powered by a VIC-III graphics chip. It had a C64 compatibility mode, meant to allow near 100% compatibility with the C64, but the existing prototypes have significant compatibility problems [34].

2.3 History of the MEGA65 project

Started in December 2013 by Dr. Paul Gardner-Stephen [36], the MEGA65 project is attempting to innovate the Commodore 65, with near 100% compatibility with C64 software. Dr. Gardner-Stephen owned a Commodore 65 prototype between 1994-2010, he also owned a Commodore 128 through the same time and when compared, preferred the C65. Dr. Gardner-Stephen also loved to tinker with these computers during the 1990s and 2000s, devising ways of accelerating the C64 CPU as an example. After deciding to sell the C65 prototype to a collector, Dr. Gardner-Stephen always had the idea of recreating the C65, and implementing some of the tricks and ideas he had worked out or heard of over the years. Then, during Dr. Gardner-Stephen's Ph.D studies, he learnt to program in VHDL and it seemed possible to realise this dream using the technology of FPGAs.

This dream was delayed due to technological issues with the FPGA boards of the time not quite being fast enough for Dr. Gardner-Stephen's vision. But by 2013, a sufficiently powerful and affordable board had been released to the market. Dr. Gardner-Stephen chose a Nexys4 development board, used by a lot of teaching institutions and designed with students in mind. This FPGA has many built in peripherals which can be used by the computer as well as its cost to performance ratio makes it ideal. The added benefit of using off-the-shelf FPGA board is that availability should be much greater compared to a PCB created just for MEGA65, which would be limited to small production runs carried out by the MEGA65 team or others using the open-source designs [37].

The MEGA65 project is completely open-source with the hardware VHDL/verilog files describing the hardware and operating system software available on a public git repository [38]. Dr. Gardner-Stephen's stated goals at the inception of the MEGA65 project were as follows [37]:

- *Better graphics than the Apple IIgs, Atari 800 or Plus/4: 1920x1200 @ 60Hz, 256 colour palette from 4,096 colours (later from 24-bit colour palette once I create an HDMI output) via my VIC-IV video controller.*
- *Better sprites than the C64. Plan is for the 8 compatibility sprites, plus perhaps 32 256-colour Enhanced Sprites with hardware scaling and practically unlimited size. Maximum number of displayable sprites will depend on the resolution of the display and the sprites on a given raster line.*
- *Faster CPU than the SuperCPU or any available 65C816 CPU (20MHz), and ideally with enough headroom to beat a 20MHz 65C816 running in 16-bit mode. Currently the 65GS10 runs at 96MHz, but with an effective speed more like 48MHz until I work on some planned IPC improvements, like a 16-bit cache of zero-page to make zero-page indirect instructions take as little as 3 cycles.*
- *More RAM than a fully expanded Apple IIgs or C65 (8.125MB). It will initially have 128KB of chipram like the C65, plus 16MB of slowram, plus "some" ROM.*
- *Comparable or better sound capability than the Apple IIgs. Multiple SIDs plus digital audio channels. Design to be finalised.*

After discussions to make sure their goals were aligned, In April 2015, Dr. Gardner-Stephen and the Museum of Electronic Games and Art (MEGA) announced a partnership to make an open-source Commodore 65-like computer [39], called the MEGA65. When asked about the motivation of the project in the press, Dr. Gardner-Stephen remarked "While It is *rather* pointless, it isn't *completely* pointless. It's like the difference between mostly dead and completely dead in The Princess Bride. The MEGA65 will be fun for those for whom it is fun, which is one purpose. Also, I intend to build and use a set of MEGA65 computers in teaching" [40].

The core of the MEGA65, which provides the computational power, comes from a innovation of a 4502 microprocessor design; the 45GS02. In 2015 there were plans for a laptop as well as a C65-like form-factor for the MEGA65, but during development the plans changed to replace the laptop with a hand-held console/phone form-factor. Several other people besides Dr. Gardner-Stephen have also started working on this project including open-source community members, Flinders University students and staff.

During development Dr. Gardner-Stephen also conceived of another potential use for the MEGA65, as a secure device for communication and general computing. This is thought possible by leveraging a characteristic of the MEGA65: simplicity, both in its hardware and software. This along with its open-source nature, allows

the MEGA65 to be completely verifiable that it is secure. Some extra features have been added to facilitate a secure device, which is talked about in Chapter 4. The MEGA65 is currently in a prototype phase of development.

2.4 The progressing complexity of computer hardware and software

The story of complexity and computers starts very much the same as the story of home computers. The technological advances led to ICs being smaller, faster, cheaper and more complex. Computer manufacturers also started using more ICs in their computers, further increasing the complexity. At the same time, software for computers has also been getting more complex.

2.4.1 History of CPU complexity

An informed discussion on complexity in computer hardware cannot be made without mention of Moore's Law. Gordan Earle Moore, Northern American engineer and co-founder of Intel Corporation, wrote a seminal paper in 1965 titled *Cramming More Components onto Integrated Circuits*. In this paper he observed a trend in electronics: ICs are the future of electronics and they have been getting small, faster and cheaper [16]. He also predicted this trend would continue into 1975, and predicted the advances in ICs would power new technologies such as home computers, automatic controls for cars and 'person portable communication equipment' or mobile phones [16]. In 1975, Moore wrote another paper, *Progress In Digital Integrated Electronics*, revisiting his earlier prediction. In this paper he observed 'Complexity of integrated circuits has approximately doubled every year since their introduction' [41]. He also predicted this would continue into the future and it largely has, as seen in Figure 2.3.

So it can been seen that transistor count has increased almost in line with Moore's prediction, with some modern CPUs having upwards of 19 billion transistors [42]. But is an increase in transistor count related to an increase in complexity? Yes, they are directly proportional; by the very definition of complexity, adding more transistors to an IC would increase its complexity.

For the last few years it has become apparent that the amount of performance increase from each new generation of ICs is not as large as previously. This is due to physical limitations; the silicon wafers are merely nanometres wide already, so they simply cannot be made much smaller [43]. There are also physical limitations in the amount of heat dissipation that can be achieved as well as problems with current leakage. These problems have led to some speculation that Moore's law could be reaching an end [44].

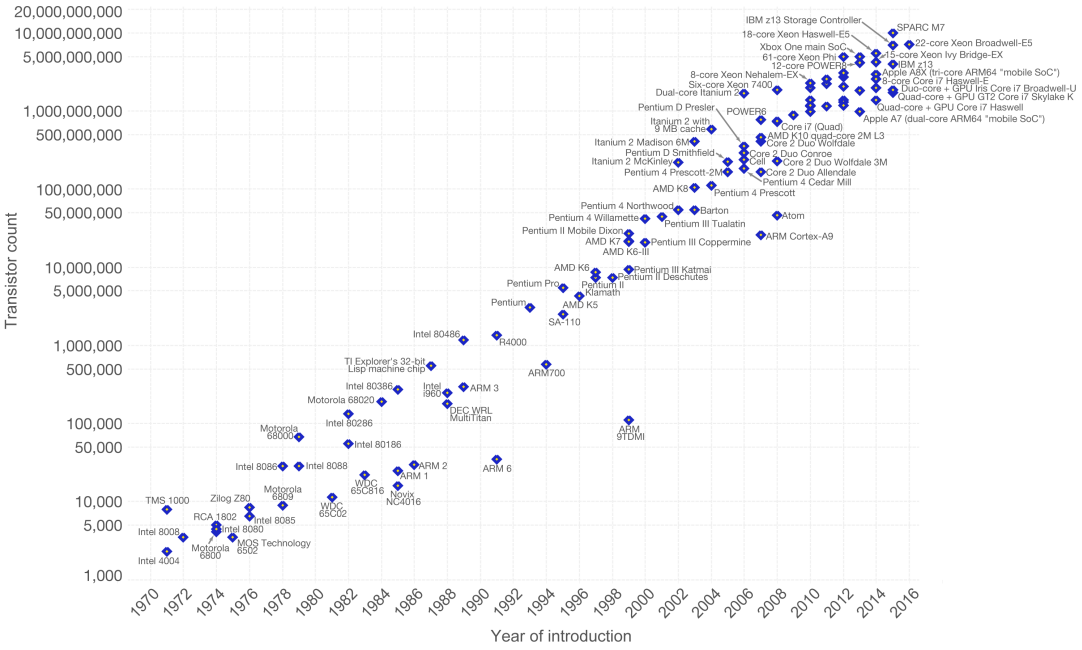

FIGURE 2.3: Plot of transistor count vs released year for microprocessors. *Picture courtesy of Max Roser.*

2.4.2 History of operating system complexity

Software, like hardware, has also been growing increasingly complex, driven by a demand that is growing faster than our capacity to design, test, implement and maintain it [45]. Many people have remarked on the steady increase in software complexity and the adverse effects it has on security, maintenance and design costs. Bruce Shneider has written extensively on computer security topics and successfully argues that 'complexity is the enemy of security' [46] [47]. Edward Ogheneovo wrote an interesting paper linking increased complexity to increased maintenance costs for software [1]. Lawson takes a broader view, talking about the complexity of computer systems as a whole and how software affects that complexity [48], as well as the rise in complexity over time. Gelsinger et al. talk about the increase in software complexity driven by the need to keep up with rapid hardware changes while designing microprocessors at Intel [49]. These papers are discussed more in section 2.5.1 but the salient point linking them is that software complexity is increasing. To further argue this point, a comparison of different versions of popular operating systems is made, as seen in Figure 2.4 and Figure 2.5. These graphs compare the Source Lines Of Code (SLOC) which is the number of lines of code in the source code before compilation. Other metrics were considered, such as the cyclomatic complexity of the source code or the permanent memory space required to install the software. But the availability of the cyclomatic complexity data was lacking and the SLOC was determined to be a more accurate indicator of complexity than the installation size of an operating system. This is partly due to the fact that installation size could be bloated with elements that are not that complex but require a large amount of

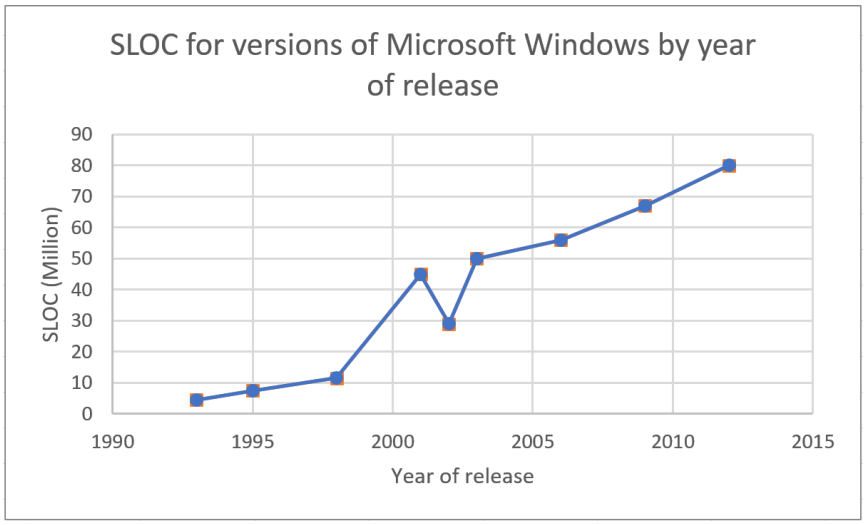

FIGURE 2.4: Source Lines Of Code (SLOC) for different version of Windows operating system vs released year
Data provided by [1]

memory space, such as video, bitmap and audio files. The reason operating systems were considered the best type of software to compare is that they (at least the versions chosen for comparison) are designed for home computers or their modern equivalents and have been is use from the early home computer era until now.

2.5 History of computer insecurity

Security is described in *RFC4949 Internet Security Glossary, Version 2* as a system condition where system resources are free from unauthorised access [50]. In recent years computer security has been a widely discussed topic, mainly due to the damage that malicious programs, or malware, have caused. There are several standards dealing just with this issue, showing its importance in modern business and engineering [51][52][53]. It is now common for companies to lose more from electronic theft than from physical theft [54]. This was not always the case. It seems the inevitable fate of any successful new technology that it will be manipulated by malicious actors.

Computer security had an interesting beginning. The terms we use today had not been coined and were not in common usage, or even were used to mean something slightly different than they do today, such as the computer virus. So to begin, a brief discussion of the origin of some common contemporary phrases.

Bug/Debugging: In 1947 Rear Admiral Grace Murray Hopper noticed the Mark II computer she was using was outputting unexpected results and upon inspection found a moth across a relay, shorting it out [55]. While the term bug was used in electrical engineering fields, this event is credited with popularising the terms use with computer programmers. A computer system 'bug' is an unexpected system behaviour. In this case the relay was shorted out, causing some unexpected behaviour.

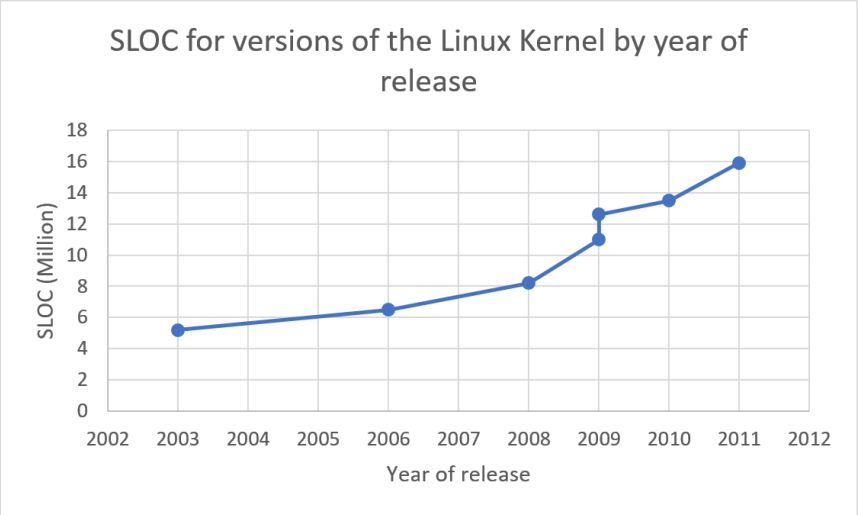

FIGURE 2.5: Source Lines Of Code (SLOC) for different version of the Linux Kernel vs released year
Data provided by [1]

Virus: Fred Cohen first uses the term in his Ph.D. paper, *Computer viruses: Theory and experiments*, published in 1987. [56]. It followed on from Jon Neuman's seminal work on self-reproducing automata, Cohen describes a virus as "a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself" [56]. Cohen also establishes viruses as a morally neutral technology; meaning it could be employed for good or bad deeds. Cohen mentions several beneficial uses for viruses in the paper, as well as security concern relating to viruses [56]. The name virus comes from the fact that the program behaves similar to a biological virus, as it must attach itself to another program, similar to a biological virus attaching itself to another cell. In the time since Cohen's definition, the meaning of a computer virus has changed to mean almost exclusively a form of malware. This change in definition has been encouraged by anti-virus companies, in a bid to help sell their products [57].

Worm: Similar to a virus in that it is a self-reproducing program, the difference comes from a worms ability to execute by itself, whereas a virus has to attach to another piece of software. Another way worms are similar to viruses is that now they are almost always malware but they first appeared as beneficial programs [58].

Malware: A portmanteau of malicious and software, which aptly describes malware; it is malicious software designed to cause harm. Examples of malware used today are Trojan Horses, Logic Bombs, Key-loggers, Lures, Worms and Viruses [50].

Ransomware: Also known as a cryptovirus or a cryptoworm. It is another type of malware which encrypts sensitive information and demands payment from the victim to decrypt the data [59]. Payment is normally asked for in a form of cryptocurrency, such as Bitcoin, because it is much harder to trace.

As computer systems have gradually been used to control and store more and more vital information and services, so have the efforts of malicious parties to break into these computer systems for their own benefit.

In 1964, AT& T, a Northern American telecommunications company started monitoring phone calls to catch people illegally using their phone lines to make long distance phone calls for free. This practise involved the use of devices to imitate the tones used by AT& T to control their phone system. These devices could be an electronic tone generator, a version of which is seen in Figure 2.6 or simply a cassette tape recording of keyboard notes [60].

1970 saw the first worms created at Xerox's Palo Alto Research Centre. A series of benevolent worms were created to carry out a variety of useful task, such as display messages or use a computer for calculations when it would otherwise be idle. Due to an error, one of these worm programs misbehaved and caused every computer infected with it to crash. Upon re-starting the computer it would be re-infected and crash immediately [58]. This event proved the potential for worms to cause massive damage to computer systems and afterwards public research was lacking for several years but it almost certainly continued in secret.

In 1986, the Brain virus was found on infected computers in North America. It was designed by two brothers in Pakistan who ran a computer shop, they were worried about pirated copies of their programs and devised the Brain virus as a way to fix this problem. Once infected, a computer would display a message letting the user know they are using pirated software and gave the contact details of the brother's computer shop to get a 'vaccination' [61].

Worms re-emerged into the public spotlight in 1988 with the now infamous Morris Worm, thought to have infected 10% of the 60,000 computers predicted to be connected to the internet at the time. It achieved wide spread news coverage becoming infamous. Released on November 2, 1988, the Morris worm effectively brought the internet to a stand still [62]. There were numerous other worms and viruses created around the same time. These early incidents raised awareness of malware and computer security.

Since the first malware was created, the problem has simply grown. With more and more computer systems controlling important information around the world, the incentive for malicious actors to break into them has grown. As more and more people use computer systems to communicate and organise their daily activities, the incentive for malicious actors to try and deceive people into divulging sensitive information increases. There are now daily reports of malware attacks as well as sophisticated scams run entirely online [63]. Governments of many countries now employ computer security experts to help protect themselves, as well as developing malware to use against their adversaries [64, 65].

A well known example of a government-made malware is the Stuxnet virus [66]. Stuxnet was found on an infected computer in 2010 by Sergey Ulasen. Ulasen ran a small computer security firm and a client was seeking help with a computer stuck

FIGURE 2.6: A Blue Box; electronic tone generating device used to gain access to AT& T's phone network in North America
Picture courtesy of Maksym Kozlenko

in a reboot loop. This triggered a massive investigation by several companies which lead to the conclusion that Stuxnet was a targeted malware designed to hinder Iran's nuclear enrichment capabilities. The target as well as the amount of effort that went into Stuxnet, lead many to believe it must have been a state-sponsored malware attack, with Israel and the USA being the most likely culprits [66, 67]. The Stuxnet virus was also interesting because it was the first case of a malware 'in the wild' being used to cause physical damage, where as the harm caused by other malware was limited to computer systems [66]. Stuxnet virus was designed to attack PLC controllers connected to frequency converter drives in one specific nuclear enrichment plant in Iran. Once these PLCs were identified, Stuxnet would then routinely run the drives at frequencies that would cause them to break.

There are many, many, many more examples of malware and computer security issues, with more happening each day. Its worth noting that there has been a trend of malware being used more frequently to steal money, the first malware seemed to have been created out of curiosity and now more often than not malware is being used to steal money or valuable information which can be sold to other criminals. The WannaCry ransomware being a salient recent example of modern malware being created to steal money [68].

2.5.1 Complexity is the root cause of modern malware susceptibility

There is one common aspect linking each and every case of computer security being breached, they all use some vulnerability within the computer system (the system being inclusive of the users) to gain access they shouldn't have. These vulnerabilities can be in the software, hardware or even in the users choice or password or their habits (use of USB drives, physical security of their computer etc.).

In the examples listed in section 2.5, AT& T had a vulnerability in their phone system in that anyone could control the system through the use of tones played down the line. It can be guessed that the idea that malicious actors would attempt do this this was not considered likely or considered at all when the system was first designed.

The Brain virus exploited vulnerability found in the MS-DOS operating system, which was new at the time [61]. The Morris worm exploited weak passwords as well as vulnerability with in Unix's sendmail, fingerd and rsh/rexec commands [61].

Stuxnet exploited four zero-day vulnerabilities, which is unprecedented and one of the reasons it is believed to be a state-sponsored malware[66]. A zero-day vulnerabilities is a vulnerabilities that is not publicly known; the software manufacture is unaware of it.

WannaCry exploited a flaw in the Windows operating system [68]. So it can be concluded that malware needs a vulnerability to exploit if it is going to gain unauthorised access to a computer system. So what is a vulnerabilities?, what causes them? and how can they be avoided? These are all logical questions that follow.

A vulnerability can be described as a way to let unauthorised actors enter a computer system, which is perhaps not that illuminating in terms of explaining what it is. This is because vulnerabilities can have so many forms, ranging from hardware design flaws such as the type that caused the Meltdown and Spectre exploits to be possible [69], to software flaws such as the type exploited by Stuxnet and WannaCry [66][68]. There could be vulnerabilities in the users selection of password, as exploited in the Morris worm [61]. Users can also provide other avenues for vulnerabilities, their use of external media and role in the physical security of the system can both be exploited. This was probably the case with Stuxnet, as it is believed to have entered the target computer network via a USB storage device, as the computer network in question was isolated from the internet [66]. The types of vulnerabilities are varied in their form, but what causes them? and how can they be avoided?

Vulnerabilities are ways to access a computer network through means that shouldn't be possible in a ideal world. And it follows that these vulnerabilities are only present if they are never found or thought of, and then fixed. So what would make it hard to find the aforementioned vulnerabilities? Well, if the task is simply too large, the design so complex, it's simply too hard to consider every possible avenue of access into the computer system. As talked about earlier in section 2.4, software and hardware have both been continually getting more complex to meet our demand for more functionality, power and convenience. This complexity will almost certainly lead to more vulnerabilities.

Bruce Shneider has extensively talked about what he calls a truism; that "complexity is the worst enemy of security" [70]. Scheider first mentioned this in 1999, in his security blog [47] and has since talked about it in at least two of his books *Secrets and Lies* (2000) and *Practical Cryptography* (2003) as well as in various articles

and interviews. Shneider convincingly argues that good security comes from getting everything right: careful analysis of source code, specifications and systems. A more complex system will require more detailed and expensive analysis, with greater chance of a vulnerability being overlooked, as there is simply more to look at. The more complex a system is: the more lines of codes make up said system, the more interactions with other systems it has and the more configuration options available, mean vulnerabilities are likely to also be more frequent. With a complex system there is more to get wrong and good security require that everything is done correctly.

Edward E. Ogheneovo wrote an illuminating paper titled *On the Relationship between Software Complexity and Maintenance Costs* in 2014. In the paper he claims software is increasing in complexity and an increase in complexity drives up the cost of maintaining the software. He successfully links maintenance costs to the frequency of bugs in the software and shows that the root cause of the bugs is complexity [1].

While working at Intel during the turbulent 1980's to 2010, Patrick Gelsinger, Desmond Kirkpatrick, Avinoam Kolodny, and Gadi Singer witnessed first-hand the increase in complexity, both in hardware being designed by themselves as well as in the CAD software created to facilitate the design process. They provide an excellent recounting of this period at Intel in the paper *Such a CAD!*. They illustrate clearly what is driving this increase in complexity: demand [49]. Customers demand and expect more powerful computer products in line with Moore's Law, this is not to say most customers were aware of Moore's Law, but they were aware (within a few years of the home computer era) that computers continually got more powerful. If Intel didn't meet this demand, others would have and Intel would not have been as successful in business [49].

Going back to 1987 and Fred Cohen's seminal paper *Computer viruses: Theory and experiments*. Cohen comes to the conclusion that the sharing of information needed in a general purpose computer, such as a home computer or PC, is in direct opposition to the goals of viral security. Which is implying that computer systems were already becoming too complex in relation to viral security in 1987.

With all this in mind, it can be said that the root cause of malware susceptibility and computer insecurity is complexity. Which leaves the question, what can be done about it? This question is beyond the scope of this thesis, but a possible answer is the MEGA65 project, an open-source simple computer which has made some consideration for security.

Chapter 3

Case-studies of retro-computing projects

This chapter forms a body of knowledge on retro-computer projects which are attempting to release a new product to market. This is achieved through a series of case-studies into several retro-computer projects. Each case-study extracts the process the project followed in releasing a new product to market. The case-studies also highlight any challenges the projects faced during the productization process.

For each case-study there is a brief description of the product and the team behind it. Any information about the productization process or challenges faced during that process is recorded. From this record, a list of the process steps is extracted, as well as a list of challenges the project faced. The challenges are then categorised into risk types for each case-study. At the end of the chapter the separate case-study processes are combined into a list and then illustrated, see figure 3.8. The separate case-study risks are combined into a risk table and ranked according to the potential damage they could cause.

3.1 Open-source case-studies

Despite not being retro, the Arduino and Raspberry Pi where both studied because of their open-source nature and their market success. The Raspberry Pi is also influenced by retro ideas from the home computer era, such as booting straight into a programming environment as well as the creators being inspired by the BBC Micro, from the home computer era.

3.1.1 Arduino

What is it?
Arduino is a company which designs, produces and sells the Arduino range of single-board microcontrollers as seen in Figure 3.1. These are mostly 8-bit machines with 32-bit machines being introduced later and are known for being inexpensive to purchase. The Arduino company has released all the hardware designs as open-source under a Creative Commons licence. The associated IDE (Integrated Development environment) software is also open-source and Arduino also encourage

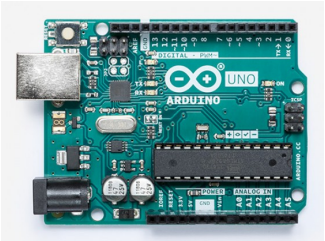

FIGURE 3.1: Arduino Uno Rev 3
Picture from `https://store.arduino.cc/usa/`

and facilitate a community of hobbyist, open-source enthusiasts and professionals [71].

When was it produced?

The first Arduino board was designed and made available to students at the Interaction Design Institute Ivrea (IDII) in Italy in 2005. As word spread, the board quickly became popular outside of the class and Arduino boards remain hugely popular today [72].

Why was it produced?

It was used to help teach the students at IDII interactive design (also known as physical computing). A co-founder of Arduino, Massimo Banzi was using another commercially available microcontroller called the BASIC Stamp to teach his class prior to the Arduino board or its inspiration, Wiring being created [72]. Banzi found the Stamp didn't meet his needs for teaching and it was too expensive for the students. Banzi decided to design his own microcontroller and associated IDE to create an entire platform for users to easily achieve their goals. Banzi drew heavy inspiration from another very similar project called Wiring that was also in development at IDII before and during the creation of the first Arduino board. Wiring in turn built on another related project, Processing, which provided the starting point for the IDE used in Wiring [73][74] [75].

Process

Arduino started with an idea, to create a cheap microcontroller for students and an accompanying platform to make it easy to use. From the idea stage, then a prototype was built. The prototype in this case is taken to be the microcontroller board and accompanying software Massimo Banzi and David Cuartielles developed in 2005 [75]. This prototype had a somewhat convoluted history, being derived from Wiring, another similar project created by a student for their thesis project [73][75]. It's worth noting that the software or Integrated Development Environment (IDE) was (and still is) open-source and several others helped in its development, up to and after the prototype stage. Banzi and Cuartielles invited others to help with the project, an

advisor Tom Igoe, a student David Mellis to help write the software and Gianluca Martino who could help facilitate the production of the board [75]. After discussions with Igoe, the Arduino team decided at some point in 2005, that the target market for their product was much larger than just the students in their respectively schools. The prototype then went into a period of refinement, with the main goals of making it cheap to produce and simple to use. This included fixing some bugs in the hardware design [75] and more extensive work on the IDE to included more user-friendly features. The software and hardware was also modified to allow support for a cheaper chip, which is the microcontroller at the core of the Arduino board.

The Arduino team decided to produce a batch of 200 units with a prearranged agreement with two schools to buy 50 units each. The agreement meant to production run would atleast make back half of its cost, even if the other 100 boards were not sold. This reduced the risk to investors (which were Arduino team members in this case). This small production run was meant as a test to see if there was market interest in the product outside of the schools. The team then placed some paid advertisements marketing the Arduino board, as well as discussing the product with friends and colleges to promote the Arduino via word of mouth [75]. The Arduino boards started to sell, slowly at first but it was obvious at this point that there was a market for the Arduino board.

As the Arduino is a single-board device with no case or human interface device (HID), the hardware process from working prototype to finished design was relatively simple. It consisted of just the electronic board design and parts selection with regard to price, quality and availability. A manufacturer had already been found, Smart Projects, which was owned by Arduino team member Gianluca Martino. The associated software was also being worked on during the period the hardware was being refined for the production run of 200 boards mentioned earlier.

The fact that Arduino creators didn't know if half of the first production run would sell, and if it did, who would be interested in it, leads to the idea that they didn't do a lot of market research before hand. The team thought of the first production run as a test to gauge market interest [75].

Steps Identified

1. Construct a working prototype to fulfil an idea or need.
2. Decide final product features and specifications based on prototype feedback.
3. Refine prototype to remove bugs/errors and add any features that are required for the product launch.
4. Add team members as needed to help grow the project. Focus on what skills they can provide when deciding on members.
5. Self fund a small production run of the final product.

6. Place advertisements and use word of mouth to spread awareness of the product.

7. Partner with or form agreements with distributors to increase the availability of the product.

Challenges faced during productization and beyond

The biggest challenge for the Arduino company was simultaneously making a profit for the company while being open-source with their products and allowing anyone to manufacture and sell them. Arduino handled this successfully, with both the designs still being open-source and the company (Arduino LLC) still trading but as it is a private company it doesn't need to publish earnings so they could not be scrutinised. The original decision to make the designs open-source, according to Banzi [75] [72], stemmed from the realisation that IDII was running out of money and would shut within a few years, potential endangering the Arduino project. Hernando Barragán, the creator of Wiring, claims Banzi didn't have a choice and had to make it open-source. This was because Arduino used work from the Wiring and Processing projects and both of them are open-source and require any derivative to also be open-source [73]. The Arduino team partnered with a manufacturer and got the first boards produced and sold for a small profit. Arduino found that as they were the first to market (with good reason) that they still sold a lot due to a lack of competition. But when cheaper copies started being manufactured in Taiwan and China, this initially had the effect of increasing the sales through Arduino, even though there were cheaper versions available. Arduino put this down to the competition being of a lesser quality at that time [76]. Arduino also charge a fee if other producers wanted to use the Arduino name. As the market matured, Arduino expected other producers and manufactures to be able to produce the Arduino boards at a cheaper price and of similar quality, which has happened. The Arduino company now focuses more on selling their expertise as the inventors, consultancy work, enabling the Arduino community and designing and selling Arduino accessories, such as 'shields', plug in devices that add extra functionality to the Arduino range [76].

The other big challenge Arduino faced was from legal challenges around ownership of the Arduino Trademark. In 2009, after the initial success of the Arduino boards, the co-founders decided they needed to create a company to hold all the trademarks and created Arduino LLC in the USA, as well as trademarking the name Arduino in the USA. It was decided to focus on the USA first, but one of the team members that was working on the production of the existing Arduino boards, separately registered the Arduino trademark in Italy without telling the rest of the Arduino team until 2014. In 2014, the company which held the Italian trademark and manufactured Arduino boards, Smart Projects, hired a new CEO. Smart Projects then changed its name to Arduino SRL and stopped paying Arduino

LLC royalties for using the Arduino name [77]. What followed was a drawn-out legal and community battle for a couple of years. In October 2016 the two parties announced an agreement to join forces and cooperate moving forward [78]. The schism seems to have come from two different parts of the initial group wanting to take the company in different directions, the production team wanting to keep production in Italy and the others wanting to expand into USA and China and let many other manufactures in.

A minor challenge the Arduino team faced was from attacks on their integrity from people that didn't like the way Banzi treated Hernando Barragán in regard to using his open-source work to create Arduino. The perceived slight was not from the use of the open-source code, which is by definition free to use, but from the fact that Banzi chose to fork and use the code without asking Barragán to be part of the Arduino team. Barragán was a past student of Banzi and they worked closely together during Barragán's creation of Wiring for his master's thesis project. Barragán also had plans to make the Wiring board compatible with cheaper microcontrollers (the chip at the core of the Arduino and Wiring boards) which is the work Banzi and team did shortly after forking the code from Wiring.

Relevant useful lessons gleamed from the study

1. Open-source hardware can make a profit in the short term by being first to market.
2. Open-source businesses can generate profit in other ways including: consultation or selling expertise of the product, licence fees for use of trademark such as Arduino did with their name as well as designing and producing accessories.
3. It is extremely important to keep control of trademarks and other intellectual property (IP).
4. Open-source projects encourage others to participate simple by being open, this in turn creates a community of people, these open-source communities seem to naturally converge around the original creator(s). The original creator of the Linux kernel, Linus Torvalds and the communities around Arduino and Raspberry Pi are other examples of this. The original creator(s), by being the focal point, should generally be well informed on what the community is doing, this can be leveraged by hosting community-oriented services, such as forums to share ideas and knowledge which in turn should increase the popularity of the product.

Challenges faced

1. Open-source business model

FIGURE 3.2: Raspberry Pi 3 Model B+
Picture from https://www.raspberrypi.org/products/

2. Legal challenges to ownership of Arduino trademark

3. Attack on integrity from use of open-source work

3.1.2 Raspberry Pi

What is it?
A cheap, open-source, single-board computer which runs a version of the Linux operating system shown in Figure 3.2. The first version had a 32-bit processor with later version including a 64-bit processor. [79].

When was it produced?
The Raspberry Pi version 1 Model B was first sold in February, 2012 for $35USD, with Model A coming out soon after for $25USD [80].

Why was it produced?
In 2006, Eben Upton, who grew up using a BBC Micro (from the early home computer era), had a desire to help kids learn computer science and related topics. He compared the BBC Micro on which he learned, which booted up straight into a BASIC command prompt, to modern devices children have access to, such as a tablet running the Android operating system. He lamented that a child with a tablet is so far removed from the inner workings of the device that learning computer programming and related skills is a lot more difficult [81]. To achieve his goal of helping kids learn about computers, Upton decided to make a small, cheap computer which would hopefully encourage kids to play and learn, as he did with the BBC Micro. Upton and several others formed a charity in 2009 with the purpose of advancing this agenda, called The Raspberry Pi Foundation. It wasn't until a few years later that Upton found a suitable chip for the Raspberry Pi. While working for Broadcom on the design team for the BCM2835 chip, Upton realised the BCM2835 would be perfect for the Raspberry Pi computer he had envisaged [81].

Process
The Raspberry Pi first started with a desire to help children engage with computers

and computer topics. Eben Upton, one of the founders of the Raspberry Pi Foundation, decided the best way to achieve this was to give them access to a cheap computer that they could "mess around" with and learn along the way, much as he did with a BBC Micro when he was young [82]. Upton built several prototypes while trying to achieve this goal. The first was built in 2006 using a microcontroller as the core and gave about the same computational power as a BBC Micro [83]. Upton originally thought a highly-integrated remake of the BBC Micro would meet his goal, but after discussion with like-minded people, Upton abandoned this in favour of a more powerful machine [82]. Upton teamed-up some of these like-minded individuals in 2009, and formed the Raspberry Pi Foundation with the stated goal to "put the power of computing and digital making into the hands of people all over the world." [84]. Upton and the rest of the foundation now faced the problem that all the microprocessor available at the time were too expensive to include in a computer which needed to be cheap. It wasn't until a couple of years later that Upton found a suitable chip, the Broadcom BCM2835, a System-on-Chip (SoC), which is a complete microcomputer within a chip. In 2011, the Raspberry Pi Foundation had made a new prototype, with the BCM2835 at its core. This new prototype was shown publicly in May 2011 and received a huge amount of attention [85]. In this somewhat unplanned showing, the Raspberry Pi team said they would have it finished by May 2012 and it would cost £15. This self-imposed deadline put some pressure on the team to finish by their publicly stated date.

Over the few months the prototype design was refined, with the goal of trying to keep the board within $35 USD. During this process several features were cut to save costs [82]. Upton also embarked on a campaign to secure parts as cheaply as possible [82]. A version of the Linux OS, designed specifically for the Raspberry Pi, was also in development [82].

Once the team had the designs for the final product agreed upon, they originally planned to make 1000 unit in the first production run to gauge the market [86] but after the May 2011 showing received so much attention, they revised the initial run to 2,000 units [82]. The funds for the original production came from donations from the trustees of the Raspberry Pi Foundation [87]. The Raspberry Pi team then had to find a manufacturer, They ended up finding a business in China which could make the Raspberry Pi at a decent price [82]. Not long after securing this initial run in 2011, the Raspberry Pi team realised they had more demand for units than they could secure funds to manufacture, they estimated 10,000 units would need to be made for the launch. To help solve this problem, the Raspberry Pi team partners with two large companies right before the product launch, Element14 and RS. They both agreed to distribute and manufacture the Pi, so the Raspberry Pi team pivoted at this point to becoming a designing and licensing business and not a manufacturer of the Raspberry Pi [82]. This partnership secured worldwide distribution and manufacturing channels, something the Upton is very proud of as he says it let the Raspberry Pi grow a lot faster [82].

Steps Identified

1. Construct a working prototype to fulfil an idea or need. This prototype is taken as the first with a Broadcom chip, shown publicly in May 2011.
2. Form foundation or other body to hold licenses and other IP.
3. Add team members as needed to help grow the project. Focus on what skills they can provide when deciding on members.
4. Decide final product features and specifications based on prototype feedback.
5. Refine prototype to remove bugs/errors and keep end product cost down.
6. Develop software needed for launch, OS and applications.
7. Get agreements for parts at acceptable prices from suppliers.
8. Launch website and have online presence, try to generate awareness through media.
9. Self-fund a production run of the final product, 2000 units.
10. License out product to larger companies which can make and distribute the product better and faster

Challenges faced during productization

The Raspberry Pi productization process seems to have avoided most major problems, how much of this can be attributed to Upton waiting for a chip that suited the project is hard to decipher, but it most definitely helped. Whether waiting for a suitable chip is a challenge or not depends on the project.

The Raspberry team announced a release date before the final product was ready, this added a lot of pressure to meet their self-imposed deadline, but may have helped push the project along at a faster pace [81].

The Raspberry Pi was delayed in the first production run due to compliance laws in Europe around electronic interference compliance. The Raspberry Pi needed to be tested for electronic interference but the creators didn't realise this until it was too late to avoid delays [82]. The Foundation wanted to keep the cost of a board to $35USD as they believed it would be critical to achieving their goal of allowing as many people as possible to afford one. To keep the cost down, Upton had to cut several features during the development phase [82]. Upton also embarked upon an aggressive campaign to secure parts at low cost [82].

Relevant useful lessons gleamed from the study

1. Open-source hardware can be incredibly popular.

2. Check relevant laws and seek expert advice as early as possible to avoid delays in production.

3. Keeping the sale price of the product low can greatly increase market size and device uptake.

Challenges faced

1. Regulatory compliance

2. Strict product criteria on cost

3.2 Retro-computing case-studies

These case-studies look at a group of products that are inspired by 8-bit computers from the home computer era mentioned. Two of the biggest names from that time, Spectrum and Commodore are both reproduced in modern retro-computing projects which are studied in this section. These projects are retro-computing and feature 8-bit computers, so they are perfect for case-studies.

3.2.1 Sinclair ZX Spectrum Vega

What is it?
The Sinclair ZX Spectrum Vega is a game console inspired by the ZX Spectrum (another hugely popular home computer from the early home computer era, which was released in the United Kingdoms in 1982 by Sinclair Research. The ZX Spectrum Vega comes preloaded with 1000 games, most of which were written for the original ZX Spectrum. It swaps the full keyboard of the original for a directional thumb pad and 9 rubber buttons as shown in Figure 3.3. It has the endorsement of Sir Clive Sinclair, the founder of Sinclair Research. The Vega was created by Retro Computers and uses a hardware design created for the project and has a microcontroller at its core and custom-built software to allow it to play all games written for the ZX Spectrum [88].

When was it produced?
The Vega was released in April 2015, following a successful crowd-funding campaign on Indiegogo, a crowd-funding service. It cost £100 at launch but is no longer produced or sold by Retro Computers as they have run into financial and legal troubles relating to another product, the Vega+ and was recently wound up [89].

Why was it produced?
The Sinclair ZX Spectrum Vega was a commercial endeavour.

FIGURE 3.3: Sinclair ZX Spectrum Vega, a game console with 1000 preloaded games inspired by the ZX Spectrum from 1982.
Picture courtesy of Marco Tangerino

Process

The Sinclair ZX Spectrum Vega first came to public attention due to a crowd-funding campaign launched by it creators Retro Computer (RCL) on Indiegogo in 2014. The campaign was intending to raise the capital needed manufacture the first production run of the Vega. The campaign was successful, raising 149% of their intended goal or £155,682. Retro Computers Limited (RCL) claimed in the campaign that the design was complete already and that RCL had agreements in place to use the Sinclair brand as well as the rights to use 1,000 games that would be shipped with the Vega. The terms of the agreement with the games rights holders was that RCL can distribute the games on the Vega but for each Vega sold a donation must be made to a charity for 10% of the Vega purchase price. To secure the rights to use the Sinclair name, Paul Andrews, the managing director of RCL at the time, reached out to Sir Clive Sinclair to seek his approval, which was forthcoming with the provision that David Levy be able to join RCL and provide oversight on behalf of Clive.

Once the crowd-funding campaign had secured funding, RCL went about finding a manufacturer and in January 2015 they announced they had formed an agreement with SMS Electronics Ltd to manufacture the Vega. Later the same month, RCL announced that they have received suggestions on how to improve the Vega from various parties and they would be implementing two of them: adding extra buttons and adding a hardware interface and ability to upgrade the software. RCL also noted that the Vega had passed the required ECM (Electromagnetic compatibility) test in June 2015. Later the same month RCL posted on the Indiegogo campaign page that the first production run of the Vega had been complete but they could not legally send them to backers yet as they first needed a PEGI (Pan European Game Information) age rating. RCL announced that the first batch of Vegas were sent to backers on the 7th of August 2015.

Steps Identified

1. Construct a working prototype to fulfil an idea or need, in this case it was basically feature complete and finished, this included the electronics design as well as the tooling and case designs and software.

2. Form agreement with relevant license or rights holders to use their IP.

3. Launch crowd-funding campaign to raise capital for first production run.
4. Find and form agreement with manufacturer to produce product.
5. Refine prototype based on community suggestions.
6. Organise to have product tested as required by relevant laws. The Vega needed to pass an ECM test and a PEGI age rating test.
7. Launch website to sell product and have an online presence.
8. Send out finished product to backers.

Challenges faced during productization

The Indiegogo campaign was successful, raising 149% of the funding goal [88]. Retro Computers launched a crowd-funding campaign offering a product to backers which wasn't complete yet, since this campaign, there has been a successful legal challenge that states that this type of agreement is a sales contract and such they are legally obliged to deliver the product regardless of problems that could arise before the product is complete [90]. This ruling came from legal action against the Indiegogo crowd-funding campaign for the next product from Retro Computers, the Vega Plus game console. So by offering a product to backers in a crowd-funding campaign, Retro Computers exposed themselves to extra risk and it may have been better to word the crowd funding campaign differently to avoid this or just not offer the product to backers at all but rather sell it to them cheaper once it is finished (if it gets finished).

Another challenge for the Vega project was the delay due to PEGI age-rating tests not be conducted and the law in the market in which they where to be sold requiring them.

Relevant useful lessons gleamed from the study

Crowd-funding campaigns can be useful to raise capital but offering a as-yet unfinished product to backers exposes the campaign founders to an amount of risk in that they legally have to deliver the product and there may be unforeseen costs and challenges in producing it.

Challenges faced

1. Crowd-funding campaign to deliver product that had not been finished
2. PEGI rating testing delaying product

3.2.2 Spectrum Vega+

What is it?

The follow up of the Sinclair ZX Spectrum Vega mention earlier. It is, like it

FIGURE 3.4: Sinclair ZX Spectrum Vega Plus console, the poorly received culmination of a tumultuous crowd-funding campaign
Picture courtesy of Craig Wootton

predecessor, inspired by the Sinclair ZX Spectrum and also has the endorsement of Sir Clive Sinclair, although it is reported he has very little to do with the actual running of Sinclair Research [91]. The Vega Plus is a hand-held console with its own screen and internal battery as well as the ability to output its display to a TV, although this feature couldn't be made to work in a review of the Vega Plus [92]. The Vega+ has a microcontroller at its core which provides the computational power [93].

When was it produced?

In July 2018, Retro Computers sent out a number of Vega Plus consoles to backers. No other Vega Plus consoles have been produced.

Why was it produced?

Like its predecessor, the Vega Plus was a commercial endeavour.

Process

The Spectrum Vega+ was the next product offered by Retro Computers Limited (RCL) after the Spectrum Vega, mentioned above in section 3.2.1. Like the Vega, the Vega+ used a crowd-funding campaign on Indiegogo to raise capital to fund the first production run of 2,500 units and prepare for the second production run [93]. When the campaign was launched on 15th February 2016, RCL claimed they had a working prototype of the Vega+ and supplied video evidence to back up this claim [94][93]. The campaign closed on the 27th of March 2016 having successful reached its target; it raised 366% of the target or £512,790. But not long after the end of the campaign, two of the directors of RCL resigned citing "irreconcilable differences" between them and the remaining directors [95]. One of the directors, Chris Smith, was also the creator and owner of the firmware which was created to run the Vega+, when he left he took the rights to use them with him and refused RCL the rights to use it in the upcoming Vega+. This proved to be a difficult situation for RCL to recover from while under the leadership of the sole remain director, Dr David Levy. RCL eventually released a small number of the Vega+ consoles to no more than 400 of the over 4,500 backers of the campaign after Indiegogo threatened to send debt collectors after RCL to recover the funds unless backers received a Vega+

[96]. RCL has now been wound-up by creditors demanding payment and no more Vega+ consoles are ever likely to be made, leaving thousands of backers without the product they paid for and hundreds of thousands of pound unaccounted for [89][97]. By almost any metric it can be said the Vega+ project was an abject failure and such the process they used will be of little value.

Steps Identified

1. Construct a working prototype to fulfil an idea or need.
2. Launch crowd-funding campaign to fund first production run and prepare for second run
3. Lose rights to use prototype from first point, forced to remake the firmware because of this.
4. Add team members as needed, Levy brought in two new directors after Smith and Andrew resigned.
5. Make promises to backers about the imminent delivery of the product but don't deliver, this happened numerous times after the end of the crowd-funding campaign, which was during the development of the Vega+ with the new firmware.
6. Release a very small number of units to some backers after multiple threats of debt collectors from Indiegogo and legal challenges from backers trying to receive refunds.
7. Wind-up business and stop communicating with backers.

Challenges faced during productization

The Vega Plus crowd-funding campaign and subsequent unkept promises of delivery from Retro Computers has been reported as one of the worst crowd-funding disaster stories ever.

The Vega Plus Indiegogo campaign raising 366% of the funding goal, over half a million pounds (over $900,000 AUD). It followed another successful campaign, the Vega, so the team had proven results. The project also had the backing and involvement of home computer era luminaries Sir Clive Sinclair and designer Rick Dickinson. As well as a working prototype and completed designs for the Vega Plus and the proven expertise within Retro Computers to finish the product. It would be reasonable to give the Vega Plus project a fair chance of success in finishing and delivering the product to backers. What actually occurred can not be called successful by any metric and it is amazing to compare the very poorly received results with the potential at the beginning of the campaign.

The first and biggest challenge faced was that very early into the campaign, two of the directors of Retro Computers resigned, including the CTO, Chris Smith, who

was the designer of the Vega Plus prototype and held the rights to it. Smith took the rights with him when leaving the company and refused to allow RCL the right to use them. This left RCL having to start from scratch with the hardware and firmware design for the Vega Plus, after already promising over 4,000 backers they would deliver a product. This in turn lead to delays, Retro Computers made many failed promises of delivery of the Vega Plus in the ensuing months and years, causing some backers to become angry.

Several legal actions followed, most notably, a backer took Retro Computers to court claiming they broke a contract of sale with him over the Vega Plus [90]. Interestingly Indiegogo's terms explicitly state that any rewards to backers are perks or gifts, but the court ruled differently in part because the receipt to the backer for his pledge stated his order for a Vega Plus had been received and this formed a contract of sale between Retro Computers and the backer (not Indiegogo). Retro Computers now legally had to deliver the Vega Plus or refund the backers money, but Retro Computers didn't have a working prototype.

Retro Computers then lost the rights to use the 1,000 games that were included on the Vega, this was due to the license holders, Sky Media, withdrawing support due to the ongoing controversy surrounding the Vega Plus, which was already well behind it's delivery date with no progress to show [98]. Retro Computers eventually managed to scrape together a barely functional console using an open-source spectrum emulator called FUSE (Free Spectrum Emulator) and a mere handful of games, all made by a single developer that was closely associated with Retro Computers and the Vega Plus project [92]. Retro Computers claims they sent out 400 Vega Plus consoles to backers, with no word of when the rest are coming. One concerned backer did some investigating and found the number of consoles sent out to be closer to 50 [92]. Following this, Retro Computers was wound-up by an ex-director of Retro Computers, while functioning as the director of another company, claiming that Retro Computers owns them money [89]. So as it stands Retro Computers is closed, no more Vega Plus (or indeed Vega) consoles are likely to be made and a large amount of money has gone missing with very little to show for it. There is ongoing legal actions against various directors of Retro Computers about these issues.

Relevant useful lessons gleamed from the study

1. Keep control of needed intellectual property needed for the project, form agreements with holders of IP for there use in the project before making any promises of delivery of product.

2. Don't do anything to jeopardize any current contracts, such as Retro Computers did to lose the backing of Sky Media and thus the ability to include the games they had promised.

3. Be honest with backers and investors as to the state of the project, this is especially true for crowd-funding backers as these campaigns generally involve a large number of people.

4. Don't make claims or promises that cannot be kept in reasonable circumstances.

5. Crowd-funding campaigns offering products to backers can possible be seen to be a contract of sale between the backers and the campaign founders. This in turn forces the campaign founder to act according to the law regarding a contract of sale, including such things as refunds and guarantees of delivery.

Challenges faced

1. Loss of agreement for use of intellectual property

2. Crowd-funding campaign to deliver product which is not yet completed

3. Use of 3rd party intellectual property

4. Adverse court finding that crowd-funding pledges constituted orders and were subject to contract law

3.2.3 ZX Spectrum Next

What is it?
The ZX Spectrum Next is a fully functional 8-bit computer that closely emulates the Sinclair ZX Spectrum from 1982. It achieves this by emulating the Spectrum in a FPGA board. The maker claim they will release the hardware design as open-source once it is complete. The case design is by Rick Dickinson the celebrated case designer of the original Spectrum series.

When was it produced?
It is currently under production. The Next is the focus of a Kickstarter crowd-funding campaign, which was launched in April 2017 and gave the original delivery date as January 2018. The campaign was well funded, raising £723,390 from over 3000 backers.

Why was it produced?
A commercial enterprise.

Process
The ZX Spectrum Next is an innovation of the ZX Spectrum from 1982. It is currently in the production phase of development after successfully raising £723,390 on a crowd-funding campaign on Kickstarter. When the crowd-funding campaign was launched on April 23rd 2017, the Next team had a working prototype with

FIGURE 3.5: ZX Spectrum Next, intended to be a fully functional iteration of the Sinclair ZX Spectrum. The makers claim it will be fully compatible with all software written for the original Spectrum
Picture from https://www.specnext.com/about/

most of the hardware design finished, they did add some features later when the crowd-funding campaign was so successful [99]. They also had CAD designs for the case and keyboard completed before the campaign [100]. With the capital raised from the crowd-funding campaign they intend to fund the first production run of the Spectrum Next. Not long after the end of the campaign which was on the 23rd of May 2017, The Next team released a website to act as a Next portal to house information for the community including hardware, software, drivers and firmware for the Next [101]. In April 2018, the Next team promised to have fortnightly updates on the Kickstarter page, this was in response to backers requests [102]. They also mentioned they have nearly decided on a manufacturer for the keyboard and the case [102]. The electronic manufacturer had been decided earlier [99]. The Next team also mentioned in April 2018, their plans for the box and manual that would be included with the Next [99].

Steps Identified

1. Construct a working prototype without case or keyboard.
2. Create finished CAD drawings of the keyboard and case.
3. Launch crowd-funding campaign to raise funds for the first production run.
4. Refine prototype to add new features, made possible by the crowd-funding campaign raising more than expected.
5. Regularly make updates to backers informing them of the progress made or problems faced.
6. Launch website portal to house community information and have online presence outside of Kickstarter.
7. Decide on manufacturers of keyboard, case and electronics.
8. Have manufacturers produce sample products for testing and quality control.
9. Adjust CAD drawing and make any other changes needed based on samples from manufacturers.

10. Secure components for manufacturing e.g. RAM, resistors, capacitors etc.
11. Finish work on manual and box designs and content.
12. Agree to start final production run with manufacturers.

Challenges faced during productization

The Next project has had trouble with the mechanical function of the keyboard, which has delayed production while they try to fix the issue. The last given expected delivery date was the 2nd quarter of 2019. There is also evidence the firmware the Next is running is still under development. The Next team also mentions problems with the exchange rate, as the value of the pound had been falling compared to the US dollar during the production period, this made it hard for them to source all the components within their budget [103]. The Spectrum Next team also had to consider the effects of Brexit [104]. The chosen manufacturer of the case unexpectedly left the project due to the then manager resigning, and the new manager not considering the project worthwhile for their business [105]. This caused some delay as the team had to find a new case manufacturer, the Next team was also worried about incurring storage costs because of this. If the keyboards were finished before the case then they would have to go into storage, costing the Next team extra money. It turned out that due to delays with the keyboard, this wasn't an issue. Another issue faced was due to a stretch goal (a stretch goal is an additional goal for the crowd-funding campaign that was above the first goal of funding the production), the inclusion of RAM sockets that would allow new RAM to be added without the need to solder, caused some problems as the devices are not in common usage any more and thus hard to buy or get manufactured. They did eventually find a manufacturer which still had the parts to make the sockets and that also agreed to make a small run for the Next [103].

Relevant useful lessons gleamed from the study

1. The time taken for quality control and testing of moving parts, such as buttons and keys, can escalate and endanger the entire project.

Challenges

1. Keyboard mechanism design issues
2. Crowd-funding currency change in value during project lifetime
3. Brexit
4. Supplier failure to deliver
5. Sourcing old component

FIGURE 3.6: C64 mini
Picture from https://retrogames.biz/thec64-mini/

3.2.4 C64 Mini

What is it?
A mini form-factor game console inspired by the Commodore 64. It also is a fully functional Commodore 64-like computer which can be control via a USB connected keyboard (not included). It has a non-functional keyboard and a functional joystick which connects to the console via a cable. The console connects to a TV to display in 720p resolution. The C64 Mini came pre-loaded with 64 games written for the Commodore 64, it also has the ability to load game ROMs from a USB flash stick. It was funded through a crowd-funding campaign on Indiegogo [106], the campaign was intended to raise funds to pay design work on the case and keyboard as well as pay for the first production run. The creators of the Mini are Retro Games, they originally planned on releasing a full-size and hand-held console form-factor, the console would have a built-in screen. During the productization process, on the 28th of April 2017, Retro Games pivoted to focus on the C64 Mini form-factor with the full-size version to follow later [107].

It's interesting to note that the resigned directors of Retro Computers (the company behind the unpopular Vega Plus campaign), Paul Andrews and Chris Smith (they resigned before the Vega Plus project went off the rails), are involved in the C64 Mini project, as well as Darren Melbourne, who worked on the C64 DTV [108].

The Mini has a ARM SoC at its core, which runs a software emulation of the C64 [107]. Retro Games have stated that they will open-source the designs as much as possible once the production is complete, currently they have not, even though the C64 Mini has been released, but the full-size version is still under production.

When was it produced?
Released in the first quarter of 2018.

Why was it produced?
A commercial endeavour.

Process
Retro Games didn't have a functional prototype at the crowd-funding campaign launch, it wasn't shown till August 2016, running on pre-production hardware. The crowd-funding campaign failed to reach its funding goal. After the crowd-funding campaign had finished, Retro Games tried to partner with a manufacturer and

distributor to secure extra funds and get the C64 Mini a worldwide retail release. They did manage a partnership, and transitioned to focusing on the C64 Mini form-factor, at the partners request. Retro Games worked on circuit design for 3 form-factors of the C64 range, as well as CAD designs for the case and keyboard. Some component selection and sourcing was also conducted during this time. Retro Games also sought agreements from game developers to use their games.

Steps Identified

1. Launch crowd-funding campaign.
2. Create functional prototype on pre-production hardware.
3. Find partner to invest, manufacture and distribute.
4. Re-evaluate with partner about best business direction (Retro Games focused on C64 Mini form-factor)
5. Secure rights to use games.
6. Finish circuit designs for final product and component selection and sourcing. Design with compliance in mind.
7. Create CAD drawings of case and keyboard.
8. Create manual and box design.
9. Test pre-production units and make amendments to product or processes as needed.
10. Start production run and send to backers.
11. Retail release.

Challenges faced during productization

The crowd-funding campaign did not meet the target goal of $150,00 USD, reaching 67% of the target or $100,611 USD. Retro Games delayed their production schedule because of this, but ultimately they reached an agreement with an international distributor and manufacturer which allowed them to continue without seeking further funding [109]. This collaboration with the aforementioned partner also lead Retro Games to rethink their plans, they pivoted to focus on the C64 mini version entirely and get that complete first before moving to the full size version.

Relevant useful lessons gleamed from the study

Regular communication with backers is important for a successful crowd-funding campaign, this includes listening to them and responding to their concerns and queries. Paul Andrews and Chris Smith must have been aware there could be potential backlash over their involvement in the Vega Plus quagmire (they were

both directors of Retro Computers when the Vega Plus campaign was launched but left a few months later). They remained honest about the situation and who they were and probably of most importance, gave regular weekly updates to backers describing progress made and relevant news as well as progress pictures whenever possible. They also, on several occasions, publicly answered specific questions from backers, which would go a long way to alleviating fears and concerns from the backers about a repeat of the Vega Plus campaign.

Challenges

1. Failure to reach crowd-funding goal

3.2.5 C64 DTV

What is it?
A single-board Commodore 64 clone housed within a joystick, which comes pre-loaded with 30 games. It was, unsurprisingly from the name, inspired by the Commodore 64. It features an ASIC (Application-specific Integrated Circuit) at its core [110] and has the possibility of after market modification due to exposed solder points on the board [111]. The C64 DTV was born from another project, the C-One which was also created by the DTV's designer, Jeri Ellsworth. The C-One is a single-board C64 clone made using FPGA technology.

When was it produced?
First released in 2004 with version 1, which support only NTSC Broadcasting Television standard. Version 2 came out in 2005 with additional PAL support

Why was it produced?
A commercial endeavour.

Process
The C64 DTV was an innovation of an earlier product, the C-One. The C-One is a single-board C64 clone that uses FPGA technology. The DTV is a smaller version of the C-One which fits into a accompanying joystick case, as shown in Figure 3.7. The DTV is aimed at letting users play Commodore 64 games, of which 30 were included. The DTV does not support the ability to load game ROMs after purchase but it is possible with some after-market modifications [112].

Steps Identified

1. Construct a working prototype to fulfil an idea or need, the prototype here is taken as being the prototype which lead to the C-One.

2. Partner with others that can help turn the prototype into a product, the C-One.

FIGURE 3.7: C64 DTV or C64 Direct-to-Television
Picture courtesy of Christian Wirth

3. Add team members based on skills, in this case to develop a games console based off the C-One.

4. Design and create the C64 DTV case including joystick and PCB.

5. Form partnership to market and distribute product.

Challenges faced during productisation

The C64 DTV was derived from another similar product, the C-One. Jeri Ellsworth designed an enhanced version of the Commodore 64 as a way to teach herself about programming FPGAs [110]. Ellsworth then partnered with Individual Computers to turn her FPGA design into a product, the C-One. Other manufacturers and retailers saw potential in the C-One and asked Ellsworth if she would consider making a smaller version that would fit in a joystick, with a focus on enabling users to play games, the result of which is the DTV, shown in Figure 3.7. Unfortunately it seems very little was published about the process from turning the C-One into the C64 DTV or from earlier efforts to complete the C-One.

Relevant useful lessons gleamed from the study

1. Retro games can sell quite well, nostalgia is a powerful incentive.

2. Extra features included in the device can help boost sales, these can be features that are not normally available but the device can be hacked or modified by the user to make them available. There are reports of user buying the DTV simply to access the C64-like computer running at its core.

3. ASICs are a technology that could be applied to this area.

4. There is a larger market for consoles that focus on playing games than other uses of a 8-bit computer.

Challenges

1. None identified

3.3 Retro-computing productization: process

This section consolidates the processes collected from the case studies above. A discussion of any interesting or useful conclusions drawn from the comparison follows.

3.3.1 Combined process from case studies

The following is a combination of the processes identified in the case studies. The steps that are common to all or most case studies are included, as well as two possible funding models.

Combined Process

1. Construct a working prototype.
2. Control IP.
3. Research relevant laws and regulatory obligations and make design changes to accommodate as well as budgeting for any costs likely to be incurred in future testing.
4. Decide final product features and specifications based on prototype feedback.
5. Add team members as needed to help grow the project. Focus on what skills they can provide when deciding on members.
6. Create CAD designs for case, keyboard and any other parts that need to be custom made for the final product.
7. Launch crowd-funding campaign or raise funds through private investors or businesses for first production run.
8. Refine prototype to remove bugs/errors and add any features that are required for the product launch.
9. Launch website to give a web presence and foster a community.
10. Find manufacturer for electronics, plastics (case, keyboard etc.) and source components.
11. Partner with or form agreements with distributors to increase the availability of the product and market the product.

Most of the discovered processes from the case studies agreed on an outline of a process to productization, which is listed above. One of the points of difference between the processes identified in the individual case-studies is how much of the design work was complete before announcing the release of the product to the public. The decision to announce earlier in the design process may be due to businesses constraints, if the project is in need of funds to continue, then announcing early may be worth the increased risk. The risk in announcing early is due to uncertainty in several factors: currency changes, law changes such as 'Brexit', unknown costs for manufacturing and design changes. The other main point of difference was to use a crowd-funding service to raise capital or to rely of private investor and/or business partnerships or a mixture of both. With regard to crowd-funding campaigns, it has been proven to be possible successfully raise funds, but a successful campaign, such as the Next, Vega and Vega+ had several common features. Successful in this context relates solely to the campaign's fund-raising effort, not the end result generated from the funds. The three campaigns named, all had a lot of information about the product as well as a working prototype shown in videos. All three campaigns also posted regular updates as well as responding to community requests and concerns.

It is hoped that the process described here will be useful for new retro-computing projects and form a basis for further research in this area.

3.4 Retro-computing productization: associated risks

The challenges identified in the case-studies are categorised into risks and ranked to produce a list of risks that are relevant to retro-computer projects. It is intended these risks be considered along side the typical project risks, which have been well researched and documented in previous research efforts.

For each identified risk, the challenges within the case-studies which fit within the risk category are mentioned. A description of the risk is given, and the factors which affect its potential damage and likelihood of occurring are also discussed.

The risks are ranked by their worst-case potential damage to the project and are shown in figure 3.9. The ranking is based off the following definitions:

1. Negligible - Slight time delay or cost on project.
2. Marginal - Moderate time delay or cost on project.
3. Critical - Extensive time delay or cost to project.
4. Catastrophic - Project cannot proceed.

3.4.1 Laws and regulations

This risk category is related to the laws and regulations of specific markets within which a retro-computing project wants to operate and sell products. The potential

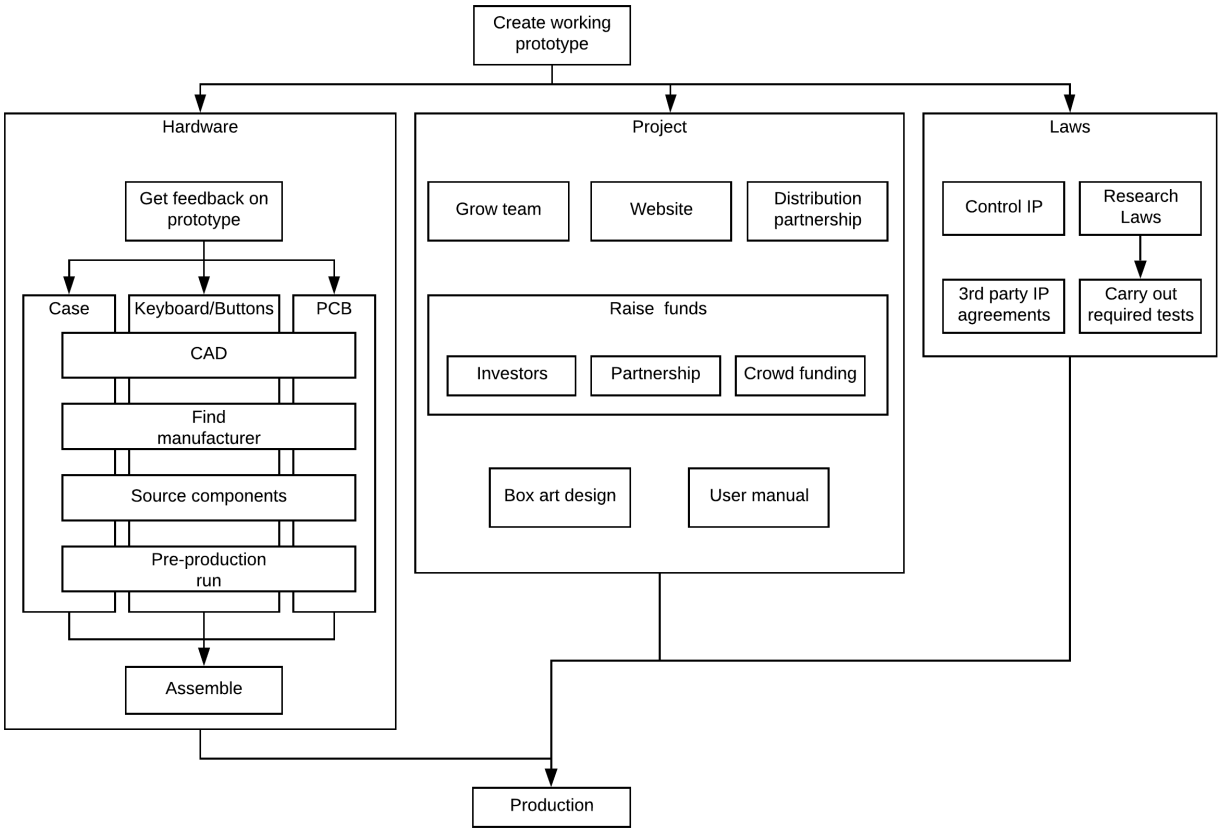

FIGURE 3.8: Retro-computer productization process. Small rectangles are tasks, larger rectangles are task areas.

damage from this risk comes from the delays and additional costs that complying with these laws and regulations may attract. This could be due to a project not being aware of required testing and as such not planning and conducting the required tests in a timely manner. Additionally, there could be further costs and delays if the product fails aforementioned tests and requires changes to the product in order to pass the test or tests. The worst-case scenario would involve the product failing a required test and the product being unable to be changed to pass.

Several case studies ran into problems complying with the required laws or regulations specific to the markets they wanted to enter. The Raspberry Pi experienced some delays because the team didn't think their product needed to undergo electronic emissions testing when in fact it did. By the time the team realised this was the case, it was too late to avoid any delay and still undergo the testing. The C64 Mini project also experienced a delay due to age-rating testing and certification.

A factors which affects the potential damage caused by this risk is how far along the productization process the project has progressed before the project becomes aware of all the required laws and regulations which they must abide by. If a product is practically complete and ready for release on the market, then any changes could be catastrophic. Inversely, if a product is still in the design phase when the required laws and regulations are researched, then any changes may be easily undertaken with minimal cost.

3.4.2 Brexit-like event

Brexit is the unofficial name given to the event of the United Kingdom leaving the European Union. This is a very specific event and will not need to be considered soon, as Brexit will have been finished. The Brexit-like event is used here to illustrate the type of large policy changes that can affect retro-computing projects if their customers, partners or their team is situated in a country undergoing an event such as Brexit. Large scale trade wars and economic recessions are other similar large scale events that should be considered.

In the case of Brexit, it is causing some concern for the Next team that a post-Brexit environment may have unforeseen laws and regulations they need to abide by to sell in European countries.

The factors which affect the likelihood of a Brexit-like event are too numerous to be discussed here in full, but the global political landscape and volatility in markets such as the share market can be useful indicators. One factor which affects the potential damage is the geographic region in which the event is occurring and if the retro-computing project in question is located in or intends to sell into that region.

3.4.3 Crowd-funding

There were several challenges associated with crowd-funding which were highlighted during the case studies. The specific areas are covered in depth below.

3.4.4 Crowd-funding: Contract of sale

This risk is associated with any retro-computing project that uses crowd-funding to generate funds. The risk comes from the potential for backers of the campaign to start legal proceedings against the campaign founders. These legal preceding are spurred by backers not receiving what they believe they are entitled to, namely the product offered to backers of the campaign. This could be because the product was never delivered, delivered late or there were significant changes to the product. The most visible example of this risk came from the case-study of the Vega Plus project.

A factor which affects the potential damage caused by this risk is the value of the items promised to backers of a crowd-funding campaign. If the backers are promised several relatively expensive products for backing then the potential damage is greater as each backers may bring a lawsuit seeking higher damages than if the promised products were less valuable.

The likelihood is increased by several factors, the amount and quality of the interactions between the campaign founders and the backers is one of them. If the campaign founders, the retro-computing project in this case, regularly post updates on the progress that is being made and interact honestly and are upfront about any problems, then the likelihood is reduced. If on the other hand the founders do not interact with the backers or repeatedly make promises which they do not keep, then the likelihood is increased.

This risk was given a catastrophic rating as the legal problems resulting from this event occurring could have stopped any of the crowd-funded projects in the case studies from being complete. Although it is arguable that if they got to the state where backers were successfully suing the campaign founders, then there's a good chance the project was already in trouble.

3.4.5 Crowd-funding: Currency

This risk is associated with the choice of currency chosen for a crowd-funding campaign. A crowd-funding campaign is normally set to a specific currency at the beginning, the risk comes from the chosen currency losing value in the time period before it's to be spent. The Spectrum Next case study uncovered this risk.

A factor which increases the likelihood of this occurring is the choice of currency, certain currencies are historically more stable. Retro-computing projects conducting research into recent currency trends before making a decision on currency, could reduce the likelihood of this even occurring. The markets and their respective currencies in which the funds are to be spent is another factor.

The potential damage is increased with time, the longer between the receiving and the spending of the funds increases the damage as the value of the currency has more time to change.

3.4.6 Crowd-funding: Failure to meet fund-raising goal

This risk is describing the event of a retro-computing project running a crowd-funding campaign to completion, but not reaching their pre-determined funding goal. This can lead to a retro-computing project not having the required funds to continue. This risk was highlighted by the C64 Mini case-study.

A factor which affects the likelihood of this event occurring is the quality, amount and trustworthiness of the media content used describing the retro-computing project. If potential backers decide that the retro-computing project has a high chance of success, then they are more likely to fund the campaign. To make this decision they would normally rely on the videos, written descriptions and other content the crowd-funding campaign founders would produce. Other factors include how much attention and media coverage the campaign receives, and how big of a market there is for the new product.

The potential damage caused by this risk is affected by the retro-computing projects funding model. If the project has the ability to generate funds from private investors or it can form a partnership with another business which can provide the funds, then the damage can be negligible. If the retro-computing project has no other way to fund its endeavours, then the harm can be catastrophic.

3.4.7 Sourcing old components

The use of old components can bring increased risk due to lack of supplies, manufacturing techniques or tools required to produce them. Old components is taken to mean components that are no longer frequently used to build modern electronic devices. Some retro-computing projects like to incorporate old components because those components were used on the computers which inspired them. The Spectrum Next experienced this problem, it's inspiration, the ZX Spectrum used ICs connected via sockets for RAM. This allowed the RAM to be changed without the use of tools.

Factors which affect the likelihood of this risk occurring are the components frequency of use in production and availability of supplies, amount others. The general rule being that the rarer a component is, the more it will cost to purchase or manufacture. As time goes by, the likelihood of this occurring will increase for any particular component, this is due to components being made redundant as new designs emerge.

The potential damage is increase for retro-computing project in which the aforementioned component is critical to the product. If the component cannot be replaced with an alternative as it would stop the retro-computing project reaching their goal of recreating or innovating a home computer, then the damage can be catastrophic. Alternatively, if the component forms parts of a feature which is not critical, then potential damage may be negligible as the feature could possibly be removed without stopping the project reaching their goal.

3.4.8 Use of 3rd party intellectual property

This risk is associated with the use of 3rd party intellectual property (IP) and the potential for the rights holders to undertake legal challenges against that use. All of the retro-computing projects from the case-studies used or attempted to use 3rd party games with their software. Every project studied handled this very well, except for one stand-out failure, the Spectrum Vega Plus.

The likelihood of this event occurring is affected by the rights holders attitude towards the use of their intellectual property. Retro-computing projects may find investigating recent legal challenges and how other projects have used the specific intellectual property beneficial. Some intellectual property used in previous retro-computing projects and home computer era computers, have ambiguous owners or have been regularly shared without challenge from the owners. This can make obtaining an agreement with the owners impossible or potentially unnecessary, but doesn't reduce the likelihood of the risk occurring. An agreement with the right holders will dramatically reduce this risk. The amount of IP used also affects the likelihood, the more rights holders which are affected, the higher the chance of this event occurring. The popularity and potential profit a retro-computing project generates is also a contributing factor, if a project is hugely successful and profitable then it is more likely the right holders mount legal challenges.

The potential damage is affected by the features of the retro-computing product that are reliant on the 3rd party IP. If the IP is used as a critical component of the product, then the damage from its forced removal could be catastrophic. If the IP is a minor feature or a component which can be replaced with a similar component, then the damage is reduced.

3.4.9 Open-source business

This risk is caused by a project trying to be both open-source and profitable, the open-source nature of the designs can make it easier for competition to enter the market. The Arduino project gave insight into how an open-source business can be successful. If a project changes the way income is generated, as the product grows in popularity, it can be profitable. In the beginning of a product's life-cycle an open-source project can generate income in much the same way as a more traditional proprietary-based technology business. By being the first to market an open-source project can beat any production competition and generate an income. Over time, if the product is popular enough, competitors may enter the market and use the open-source designs to manufacture their own products and reduce the possible income for the original project. By being ready to respond to this possibility by altering the business model and focussing more on accessories for the product, commissioned constancy work or speaking at presentations as examples, the retro-computing project can still make a profit.

The likelihood for this occurring is affected by the amount of designs from within a retro-computing project are open-source. If everything is open-source, the likelihood is increased due to the lessened cost to a competitor trying to produce the product. The likelihood is also increased if the retro-computing product produced is popular. The more popular it is, the more potential for profit a competing manufacturer could generate.

The potential damage is affected by the structure of the retro-computing project's team and its ability to adapt to changes. If a retro-computing project team is too entrenched around generating profit from selling an open-source product and too slow to change when competition emerges, then the damage can be catastrophic.

3.4.10 Loss of use of intellectual property

This risk is describing the event of intellectual property (IP) being withdrawn from use by its owners. This IP could have been created by team members of the project whom have changed there mind on allowing its use, or from 3rd parties how have given their agreement to use the IP but then have withdrawn it. The damage is caused by the legal challenges which can stem from the continued use of the IP, from the losses incurred in removing any IP from memory, documentation or marketing. This risk is most obvious from the Vega Plus case-study, in which the CTO left with their design of the prototype shown in the crowd-funding videos. The Arduino case study highlighted a case where the Arduino team had to endure legal challenges against the use of the Arduino trade mark.

This risk is dependant on the amount of IP used in the project and more importantly who holds ownership or who has agreement to use aforementioned IP. If the IP holder is an individual and their is no agreement in place to use the IP, or if the agreement allows the withdrawal of aforementioned IP without good reason, then the risk is increased. Inversely, if an entity associated with the project, such as the Raspberry Pi foundation, is the entity which holds ownership of the IP, then the risk is significantly reduced. An open-source ethos would also reduce the likelihood significantly as an open-source design is not possible to withdraw the right to use it.

3.4.11 Supplier failure

This risk describes the event of a supplier with which a retro-computing project has formed an agreement, not supplying what was agreed, either by being late, the supplied goods being wrong in some regard or the supplier being unable to provide any goods. This risk is not unique to retro-computing projects but is included because a case study into the Spectrum Next showed they experienced a delay when a manufacturer suddenly decided to not produce their product.

The potential damage and likelihood of this event occurring is specific to each supplier and each purchase. The likelihood of this event occurring can be reduced

by using reputable, established suppliers as well as undertaking due diligence into suppliers before entering into agreements. The potential damage occurred is dependant on the nature of the purchase which the supplier failed to deliver as well as the nature of the failure. If the purchase was of a highly critical component that the project needs to move forward, then the potential damage can be critical. If the purchase was for an incidental part which could be easily obtained from numerous suppliers then the potential cost would be negligible.

3.4.12 Open-source fair use

This risk is associated with the use of open-source work and the potential for damage to reputation from the way in which the open-source work is used and the treatment of the open-source works creator(s). The Arduino project experienced a slight challenge from certain individuals that considered the way in which Banzi used the open-source work that formed the start of the Arduino project, as unethical. This mainly stems from the fact that Banzi was the supervisor of a student's project that Arduino was started from. Individuals expressed that they felt it was unethical to use the work without inviting the student in question to join the Arduino team.

The likelihood is affected by the amount of open-source work from creators outside of the retro-computing project that is used within the project. The use of more open-source work increases the likelihood. Another factor is the relationship between the original creator of the open-source work and the retro-computing project, the closer the relationship the higher the likelihood.

The potential damage is relatively small for this risk, as it only has to do with loss of reputation and community backlash which may affect sales of a retro-computing project products but will likely not stop a project completely.

3.4.13 Physical production problems

This risk is to do with the problems that can occurring during the design, manufacture or construction of a retro-computing product and the additional costs they can cause. As new components are manufactured and tested, problems can be highlighted which need to be addressed before the product can be sold. These problems can cause delays and incur extra costs to a project. The Spectrum Next experienced a delay due to the mechanism of the keyboard buttons. The Spectrum Vega Plus had a problem with the design and construction of their buttons, causing delays.

This likelihood of this event occurring is increased by the amount of physical components and buttons required for a product. Also the nature of the buttons or other HID component, if it is a more standard component, following common practises, it will be less likely to suffer this event.

Combined Risks

Challenge	Risk type	Harm severity
Regulatory compliance	Laws and regulations	Catastrophic
Crowd-funding to deliver product that doesn't exist yet	Crowd-funding contract of sale	Catastrophic
Sourcing old component	Sourcing old components	Catastrophic
Use of 3rd party intellectual property	Use of 3rd party intellectual property	Catastrophic
Failure to meet crowd-funding funding goal	crowd-funding failure to meet funding goal	Catastrophic
Project member leaving and taking their IP	Loss of right to use intellectual property	Catastrophic
Legal challenges to ownership of IP	Loss of right to use intellectual property	Catastrophic
Open-source business model	Open-source business	Critical
Supplier failure to deliver	Supplier failure	Critical
Attack on integrity from use of open-source work	Open-source fair use	Marginal
Keyboard mechanism design issues	Physical production problem	Marginal
Crowd-funding in currency other than USD	Crowd-funding currency	Marginal
Brexit	Brexit-like event	Marginal

FIGURE 3.9: Combined list of risk associated with a retro-computer project

The potential damage is critical and affected by factors such as the manufacturing process used, the cost of tools or moulds needed and the amount of HID components or other physical components are needed as well as their complexity.

Chapter 4

MEGA65: A retro-computing project

This chapter describes and outlines the current state of a retro-computing project, the MEGA65, mentioned in Chapter 2. This chapter first provides a description of the MEGA65 project and its intended products. Following the description, a use case study of the MEGA65 is conducted to further describe the MEGA65. The use case study is also employed to elicit the requirements needed to realise the use cases. This detailed information about the MEGA65 project is then used in the next chapter, Chapter 5, to evaluate the project against the risk list that was obtained from the case-studies in Chapter 3.

4.1 MEGA65 description

The MEGA65 is a evolution of the Commodore 65, built using FPGA technology at its core, more details are given in section 2.3.

The MEGA65 is planned to have two separate form-factors: a hand-held game console with two cellular modem sockets to allow for telephony, called MEGAphone (figure 4.2) and a desktop form-factor inspired by the look of the Commodore 65, figure 4.3.

During development of the MEGA65, the MEGA65 team got into discussions with the Museum of Electronic Games and Art (MEGA) and came to an agreement that MEGA will handle the physical development of the MEGA65 desktop version.

Both form-factors are to have the same FPGA chip and FPGA implementation. This means they are the same computer at the core and can run the same software. Both form-factors are intended to ship with the same software, both custom software made specifically for the MEGA65 and 3rd party software. Both form-factors are collectively referred to as the MEGA65.

The MEGAphone is to be developed further at Flinders University, with the original MEGA65 team members as well as involving students from the university. The desktop version is to have its physical development handled by MEGA. Physical development refers to the Printed Circuit Board (PCB) (figure 4.1), the case and the keyboard.

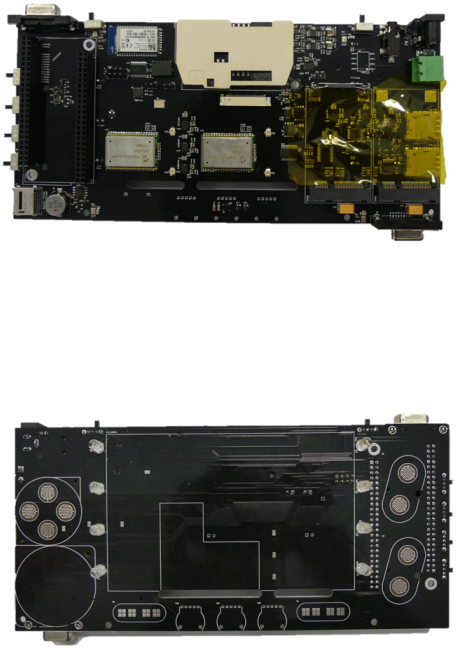

FIGURE 4.1: MEGAphone PCB revision 1 populated with most of the components. Top picture is the back of the MEGAphone, bottom is the front. Pictures taken March 2019

FIGURE 4.2: MEGAphone design concept.
Picture courtesy of MEGA65.org

FIGURE 4.3: MEGA65 desktop form factor design concept.
Picture courtesy of MEGA65.org

4.1.1 MEGA65 team, funding and goals

As it is relevant to the risk evaluation undertaken in Chapter 5, this section will briefly describe the MEGA65 team and funding model.

The MEGA65 team is centred around Dr. Paul Gardner-Stephen, the inventor and lead contributor to the MEGA65 project. Dr. Paul Gardner-Stephen is a senior lecturer at Flinders University. The MEGA65 project was started as a hobby and the goal of the project is not to generate a profit, as it was in many of the case studies.

The project is funded, in part, by the university because the project acts as a conduit for student learning. Many of the other project members are volunteers or students.

The MEGA65 project hopes and intends the MEGA65 to be able to function as a secure device and some work has been done to facilitate this.

4.1.2 Hardware

This section briefly describes the hardware, such as the human interface devices of the MEGAphone and desktop form-factor of the MEGA65.

The desktop form-factor is planned on featuring a built-in 3.5 inch floppy disk drive, like the Commodore 65 prototype. The desktop form factor will also feature a full-size keyboard. It will require a monitor to be connected to view its output. It will also have a SD drive.

It is planned that the MEGAphone will feature its own screen with internal battery and speakers, allowing for mobile usage. The screen will be a touch screen interface with additional controls provided by a d-pad and 4 buttons. The MEGAphone will also feature two sockets for the connection of cellular modems, and the required support to be able to make phone calls and send text messages. It will also have a SD card drive.

4.1.3 Software

The MEGA65 has had a range of custom software created for it, this section aims to list and describe that software. This will help further describe the MEGA65 as a product as well as give some explanation of the functions available to these pieces of software which will be helpful to carry out the use case studies.

Kickstart - the Hypervisor

Kickstart is the name of the hypervisor. It is unusual because it is also like a BIOS and boot-loader in a modern operating system, as well as a hypervisor. It was custom built for the MEGA65 and fits in 16 KB. On start up, the hypervisor loads the BASIC ROM. The BASIC ROM formed the entire operating system on the Commodore 64. Kickstart also handles kernel operations as such it handles all the freeze operations for the Freeze menu. The hypervisor also contains a few utilities housed within FPGA RAM, these can be run from the hypervisor by holding down the two Shift keys on reboot. One of these utilities is a config menu, which allows changing of certain settings. Another is the FDISK utility which can format a SD into the required MEGA65 format, ready for first use.

The Freeze Menu

The Freeze menu is a program which allows for a variety of useful features. It can be entered into from any program by holding down the Restore key for 1/2 a second, the main menu is shown in figure 4.4. When run, the Freeze Menu stops the current process or freezes it, and saves the state of the computer. This allows programs to be saved and swapped with other programs, then swapped back to, resuming from the original point, this feature is common in modern computers but it wasn't available on the C64. The idea was to get some of the same functionality as a 'freeze cartridge', a popular type of device in the Home Computer era. Some of the features available from the Freeze Menu are:

1. Restore and return to program which was running when Freeze Menu was run, by pressing F3.
2. Task switching.
3. View, select and load from disk images stored on a SD card.
4. Toggle video mode between NTSC and PAL.
5. Change CPU frequency between pre-set values: 1 MHz (C64 speed), 2 MHz (C128 speed), 3.5 MHz (C65 speed), 40 MHz (MEGA65 speed).
6. Change state of machine such as change register values which store the current amount of lives in a video game. This allows 'freeze cartridge' like 'cheats' to be used.

FIGURE 4.4: MEGA65 Freeze Menu main screen.
Picture courtesy of MEGA65.org

7. Setting options to set protected ROM and to enable the cartridge port.
8. Allow inspection of memory and allows changes to it. Action replay-like freeze cartridges used features like this to modify games.
9. Audio mixer.

Matrix Mode

An out-of-band machine state inspection tool which allows the user to inspect the RAM, ROM, and CPU, I/O registers.

Secure Mode

Secure mode is a special mode to allow the untrusted components of the MEGA65 to be excluded from the current working environment. This includes the cellular modem and SD card among others. Figure 4.5 shows the secure compartment and the untrustworthy components.

MegaWAT

MegaWAT is an open-source slide presentation and editing application [113]. It features:

1. Text editing with colours, attributes and typeface selection
2. Multiple slides
3. Presentation mode
4. Load and save
5. Changeable screen colour

MEGA65 IDE

There is also a MEGA65 IDE, which currently only allows viewing of text files.

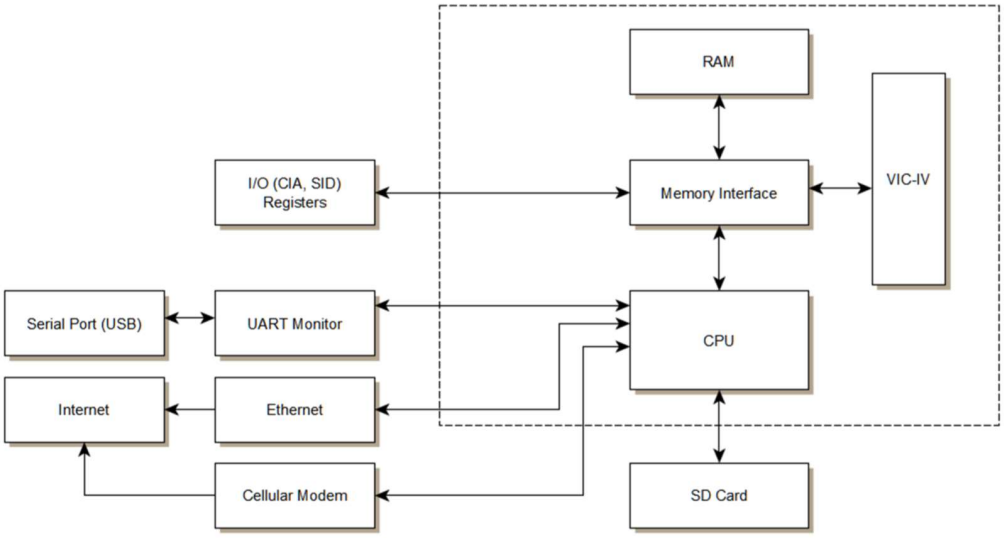

FIGURE 4.5: Secure compartment of the MEGA65. All untrustworthy components are outside of the container.
Picture courtesy of Timothy Kirby

It does currently support viewing of multiple files at once, up to 5 in separate windows, which it can be switched between and the files scrolled through.

Telephony software

This application allows phone calls to be made over a cellular modem, which the MEGAphone will have sockets to install two of them.

4.2 Use case study

This sections lists and discusses several use cases for the MEGA65, in both its form-factors, the desktop computer and the hand-held console. It is hoped that by studying the way the MEGA65 might possibly be used and extracting the features required to fulfil each use case, that a clear description of the MEGA65 can be formed. The requirements can then be used to help capture the current state of the MEGA65 in terms of its development. This list can then help with decision making regarding the minimum viable product for the MEGA65, but that is outside the scope of this thesis. The use cases were first decided upon by determining some roles or actors from the MEGA65's target market, with help from MEGA65 team members. These actors are described below, as well as a brief look at there likely use cases, as seen in Figure 4.6.

4.2.1 Actors

Gamer

The Gamer's goal is to be able to play retro games. The amount of games available and the ability to add more games to the device are valued by this actor. The ability to use additional HID devices, such as an external joystick is important to this actor.

8-bit Programmer

The 8-bit Programmer referred to as just Programmer from here on, has a goal of being able to run 8-bit programs in a C64 or C65 environment. They are interested in any feature which will add compatibility between the MEGA65 and C64 or C65 software, such as support for different opcodes or new video modes.

Technology Enthusiast

The Technology Enthusiast's goal is to try new technology and experience rare or peculiar technological devices, this group also contains 'modders' or 'hackers', people that like to make changes to products they have purchased. They are interested in any of the features but especially in the features that will add something unique or the ability to modify the product easily.

Privacy Motivated

The Privacy Motivated actor's goals is to be able to securely communicated and carry out a variety of tasks. They are interested in features that will let them thwart any malicious attempts to obtain data. Features that will allow the actor to perform checks on the MEGA65 and thus increase trust of the security of the device, will also be highly valued.

Educator

The Educator's goal is to teach computer skills and/or topics to others. They are interested in simple computer systems, as this will help highlight the fundamentals of a computer. They are interested in being able to easily write and run code. They are also interested in presenting slides and using 8-bit applications for teaching purposes.

4.2.2 Use case narratives

This section describes the use cases in a narrative form. The narrative tries to cover the most likely ways that users will interact with the MEGA65. Where the narrative is for a specific form-factor of the MEGA65 it will be stated, otherwise assume it is for both form-factors.

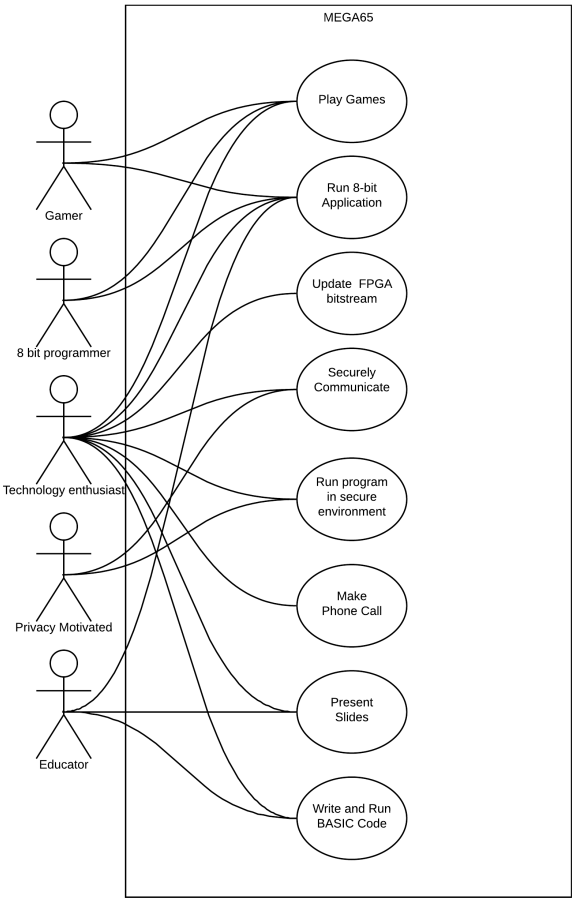

FIGURE 4.6: MEGA65 use cases

Play Games

The SD card is physically inserted into the MEGA65 before use, either by the actor or during production. The SD card must contain the games to be played and a version of BASIC which is compatible with the games to be played, e.g. Commodore 64 BASIC is needed to play a Commodore 64 game.

The actors turns on the MEGA65 and then enters the Freeze Menu once the MEGA65 has booted. From the Freeze Menu, the actor navigates to the SD card disk image viewer and loads the image with the desired game on it. While still in the Freeze Menu the actor changes the computer mode to Commodore 64 and then leaves the Freeze Menu. The actor then leaves the Freeze Menu and loads and runs the game from BASIC command prompt. Once the game has loaded, the actor plays the game using the keyboard on the desktop form-factor or the buttons and d-pad (directional pad, 4-way button commonly used on games consoles to control movement) on the MEGAphone form-factor. The actor changes the volume of the game sounds.

Play Games: Variation

After playing briefly the actor wants to change the CPU mode to the Commodore 64 speed, so the actor enters the Freeze Menu and then changes the CPU mode to the Commodore 64 speed. The actor then leaves the Freeze Menu, resuming the game at the point where the Freeze Menu was entered earlier.

The actor wishes to control the game using an external joystick. Actor enters Freeze Menu from the game. Actor physically connects joystick via joystick port. Actor resumes game from Freeze Menu and tests joystick, if it works then no more needs to be done, if not, it is most likely that the actor needs to enable the joystick controls from the MEGA65 settings. To do this the actor enters the Freeze Menu and saves the state of the current game being played. Then the actor restarts the MEGA65 and enters the Hypervisor menu and from there the configuration menu and then actor enables the joystick controller option. The actor then leaves the Hypervisor and once the MEGA65 has finished booting, the actor enters the Freeze Menu and loads the saved state of the game they were playing.

Upon loading the game, the actor notices some strange behaviour in the graphics, they appear to be odd colours and are flashing in a strange way. The actor thinks this is most likely due to the video mode being used is the wrong type for the game. The actor enters the Freeze Menu and changes the display mode from NTSC to PAL and then leaves the Freeze Menu, resuming the game, which now looks as expected.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card

3. Power switch
4. Ability to navigate through menus and play games i.e human interface device. For the desktop form-factor this would be a full-sized keyboard. For the MEGAphone form-factor menu interactions, it could be either buttons or a touch screen or both. Playing a game would require the d-pad and at least one button on the MEGAphone.
5. Speaker
6. Volume controllers
7. Joystick port

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. Game software
4. Freeze Menu - ability to view SD card contents and load images
5. Freeze Menu - ability to enter Freeze Menu from any program
6. Stable MEGA65 FPGA components with ability to boot and run game when supplied with the relevant BASIC ROM
7. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
8. Speaker drivers
9. Volume control drivers
10. Freeze Menu - ability to change CPU modes to decrease speed to Commodore 64 levels
11. Freeze Menu - ability to "freeze" a program and then reload its state on request
12. Joystick drivers
13. Hypervisor - ability to turn on/off joystick support
14. Freeze Menu - ability to change display mode between NTSC and PAL
15. Video controller capable of doing both NTSC and PAL display modes

Run 8-bit Application

This use case is practically the same as the Play Games use case except an application is loaded from the SD card not a game. The differences between Play Games use case are: the actor would not want to use a joystick and the application may not need sound. Another difference is the application may be more responsive and faster when the MEGA65 is run at the fastest speed. The MEGAphone would require a keyboard to effectively use most applications.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card
3. Power switch
4. Ability to navigate through menus and applications i.e human interface device. For the desktop form factor this would be a full sized keyboard. For the MEGAphone form factor menu interactions, it could be either buttons or a touch screen or both. Controlling an application would require a keyboard.
5. MEGAphone - keyboard port

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. 8-bit application software
4. Freeze Menu - ability to view SD card contents and load images
5. Freeze Menu - ability to enter Freeze Menu from any program
6. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
7. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
8. Freeze Menu - ability to change CPU modes to increase speed
9. Freeze Menu - ability to "freeze" a program and then reload its state on request
10. Keyboard drivers

Update FPGA bitstream

This use case is unique in that it needs another computer to perform to modifications to the FPGA components. That process will not be covered here as it doesn't relate to the MEGA65 specifically. This use case will start after the FPGA component has been successfully modified.

The actor connects the MEGA65 to the modern computer which houses the updated bitstream via a USB cable. The actor powers the MEGA65 but does not boot it up. The actor then begins the process of copying the new bitstream from the modern computer onto the MEGA65. Once this is complete, the actor disconnects the USB cable then boots up and uses the MEGA65 with the updated bitstream.

Requirements needed to meet narrative

The requirements relating to the additional computer required to carry out the FPGA modifications are not listed here.

Hardware

1. FPGA USB connection
2. Power switch
3. Ability to navigate through menus and applications i.e human interface device. For the desktop form factor this would be a full sized keyboard. For the MEGAphone form factor menu interactions, it could be either buttons or a touch screen or both.

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. Freeze Menu - ability to view SD card contents and load images
4. Freeze Menu - ability to enter Freeze Menu from any program
5. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
6. Human interface device firmware/drivers created, installed and configured to work with relevant hardware

Make Phone Call

This use case is only applicable to the MEGAphone form factor, it is assumed there is a 4G cellular modem installed with an active SIM card also installed.

The actor turns on the MEGAphone, after booting the actor enters the Freeze Menu and navigates to the SD card disk image viewer. The actor selects and loads the image with the telephony software on it. The actor then leaves the Freeze Menu

and runs the telephony software with a BASIC command. Once the telephony software has loaded, the actor selects a phone number from the saved contacts list. MEGAphone begins a call to the given number, the actor can hear the other person on the call and be heard. The actor adjusts the receiving volume to better hear the other caller.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card
3. Power switch
4. Ability to navigate through menus and telephony software i.e human interface device. For the MEGAphone form factor, it could be either buttons or a touch Screen or both. For the desktop form factor a keyboard would be the control method
5. Cellular socket
6. 4G cellular modem
7. Volume controls
8. Speaker
9. Microphone

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. Telephony software
4. Freeze Menu - ability to view SD card contents and load images
5. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
6. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
7. Cellular modem drivers
8. Speaker drivers

9. Microphone drivers

Securely Communicate

This use case is closely related to the Make Phone Call use case and it is specific to the MEGAphone form-factor.

The actor turns on the MEGAphone and enters the Freeze Menu. From the Freeze Menu the SD image with the telephony software is loaded, then Freeze Menu is left, returning the actor the the BASIC environment. The actor now loads the telephony software from the BASIC environment. The actor has three ways in which they can securely communicate: encrypted phone call, encrypted text message or send an encrypted file via a text message. To achieve any of these the actor uses the telephony software to select the desired communication medium and then selects the option to encrypt the contents. The actor then carries out the phone call or writes and sends the text message.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card
3. Power switch
4. Ability to navigate through menus and telephony software i.e human interface device. For the MEGAphone form factor, it could be either buttons or a touch Screen or both. For the desktop form factor a keyboard would be the control method
5. Cellular socket
6. 4G cellular modem
7. Volume controls
8. Speaker
9. Microphone

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. Telephony software with encryption capabilities
4. Freeze Menu - ability to view SD card contents and load images

5. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
6. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
7. Cellular modem drivers
8. Speaker drivers
9. Microphone drivers

Run Program in Secure Environment

This use case could have many variations but the most important aspects are captured here in a single description, in which the actor wants to type up a new text document then encrypt it, all while being as securer from malicious actors as possible.

The actor turns on the MEGA65, after booting the actor loads their choice of text editing software. The actor then enters "Matrix mode" and requests a secure container to work in, referred to as secure mode in section 4.1.3. Still within "Matrix mode", the actor inspect the state of the machine for suspicious activity. Leaving "Matrix mode" but still within the secure container, the actor returns to the text editor and types out the document and then encrypts it. The actor leaves secure mode, relinquishing the secure container.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card
3. Power switch
4. Ability to navigate through menus and applications i.e human interface device. For the desktop form factor this would be a full sized keyboard. For the MEGAphone form factor menu interactions, it could be either buttons or a touch screen or both. Entering text would require a keyboard.
5. MEGAphone - keyboard port

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. Text editing software

4. Freeze Menu - ability to view SD card contents and load images
5. Freeze Menu - ability to enter Freeze Menu from any program
6. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
7. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
8. Keyboard drivers
9. Matrix mode
10. Ability to create secure compartments
11. Ability to encrypt a text document

Present Slides

The actor boots up the MEGA65 and enters the Freeze Menu, then they select and load the SD image with MegaWAT on it. Leaving the Freeze Menu, the actor loads MegaWAT from the BASIC environment. The actor returns to the Freeze Menu and changes the CPU speed to the fastest available. An external monitor is connected via the video output port.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card
3. Power switch
4. Ability to navigate through menus and applications i.e human interface device. For the desktop form factor this would be a full sized keyboard. For the MEGAphone form factor menu interactions, it could be either buttons or a touch screen or both. Entering text on slides would require a keyboard.
5. MEGAphone - keyboard port
6. Video output port

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. MegaWAT software

4. Freeze Menu - ability to view SD card contents and load images
5. Freeze Menu - ability to enter Freeze Menu from any program
6. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
7. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
8. Freeze Menu - ability to change CPU modes to increase speed
9. Keyboard drivers

Write and Run BASIC Code

The actor turns on the MEGA65, after it boots it enters straight into the BASIC command prompt. The actor uses a keyboard to enter and run BASIC code.

Requirements needed to meet narrative

Hardware

1. SD card slot
2. SD card
3. Power switch
4. Ability to navigate through menus and applications i.e human interface device. For the desktop form factor this would be a full sized keyboard. For the MEGAphone form factor menu interactions, it could be either buttons or a touch screen or both. Entering code would require a keyboard.
5. MEGAphone - keyboard port

Software

1. SD card controller firmware and/or FPGA components to enable reading from SD card
2. Relevant version of BASIC
3. Freeze Menu - ability to view SD card contents and load images
4. Stable MEGA65 FPGA components with ability to boot and run application when supplied with the relevant BASIC ROM
5. Human interface device firmware/drivers created, installed and configured to work with relevant hardware
6. Keyboard drivers

4.3 State of the MEGA65 as of July 2018

This section highlights the state of the MEGA65's development at the beginning of this thesis project. This is useful to help illustrate the MEGA65 project as a whole as well as to indicate its current state of completeness. This was achieved by using the requirements gathered from the use case study above and then eliciting the state of the MEGA65 from the MEGA65 team. Table 7.1 shows the requirements and whether they were able to be met as of July 2018. A brief discussion of some of the Table 7.1 entries follows.

The desktop form-factor is practically complete with a working keyboard which is used to navigate through the menus. The MEGAphone is still in development and the touch screen and buttons are not yet working. The MEGAphone can be controlled via an external keyboard.

The MEGAphone doesn't have speakers attached but the supporting circuitry and components are present. The volume control thumb knobs are also not attached.

TABLE 4.1: MEGA65 state of completeness as of July 2018

Requirement	**Desktop**	**MEGAphone**
Read from SD card	yes	yes
Power switch	yes	yes
Navigate through menu with HID	yes	partial
Can produce audio	yes	no
Can receive audio from cellular modem	N/A	no
Ability to adjust speaker volume	yes	partial
Can connect external joystick	yes	yes
Can connect external keyboard	yes	yes
Can connect USB cable for FPGA data transfer	yes	yes
Ability to change CPU speeds	yes	yes
Commodore 64 CPU speed available	yes	yes
35-50 MHz CPU speeds available	yes	yes
Ability to 'freeze" a program then resume it	yes	yes
Matrix mode	no	no
Ability to request a secure compartment	no	no
Ability to encrypt a text document	no	no
Game software	yes	yes
8-bit application	yes	yes
Text editing software	yes	yes
MegaWAT software	no	no
BASIC ROM	yes	yes
Telephony software	N/A	yes
Text messaging software	N/A	yes
Ability to encrypt phone calls	N/A	no
Ability to encrypt text messages and attached files	N/A	no

Chapter 5

Risk evaluation of the MEGA65 project in July 2018

This chapter aims to provide useful advice to the MEGA65 project, with the aim of reducing the project's risk exposure. To achieve this, the detailed snapshot of the project's progress at the point in time of July 2018, shown in Chapter 4, is used to evaluate the MEGA65 project's risk levels for risks identified in Chapter 3.

The layout of this chapter is that each risk identified in the risk list is given a section and within each section the MEGA65 project is scrutinised to determine its risk level to that specific risk. Following the risk evaluation, for the identified high risk areas, some advice that could reduce the project's risk level is given. This advice is in the form of alternate solutions or methods of doing things and their potential risk reduction.

5.1 Risk evaluation

For each risk identified in the risk list, figure 3.9, there is a discussion of the factors which may change the risk level and are relevant to the MEGA65 project. These factors could affect the potential damage or the likelihood of the risk.

The risk evaluation of the MEGA65 project is carried out following the guidelines given in *AS ISO 31000:2018 Risk management — Guidelines* [114] and categorised using the definition of damage listed in the table at the start of Chapter 3. The risk is determined using the risk matrix Figure 5.1.

	Negligible	Marginal	Critical	Catastrophic
Certain	High	High	Extreme	Extreme
Likely	Moderate	High	High	Extreme
Possible	Low	Moderate	High	Extreme
Unlikely	Low	Low	Moderate	Extreme
Rare	Low	Low	Moderate	High

FIGURE 5.1: Risk matrix showing levels of risk for given likelihood and harm severity.

5.1.1 Laws and Regulations

The MEGA65 products, the MEGAphone and the desktop version, will have to comply with the relevant laws and regulations of the countries in which they are sold in. These laws and regulations are specific to each country, but the general definition and intent of the laws and regulations is similar in lots of places around the world, at least in regard to electronic consumer products. This similarity in laws and regulations comes, in part, from international organisations and standards being adopted or used as a starting point for country specific standards. While the potential damage to a project is catastrophic as it could force a project to be abandoned in the most extreme cases, the uncertainty is fairly low as the laws and regulations are placed in the public domain before being enacted and should be unambiguous. Two cases of laws and regulations causing problems were identified during the case studies. The Raspberry Pi was delayed due to electronic interference testing and the Spectrum Vega was delayed due to age rating testing. While both of these examples are European specific, they similar nature of the laws and regulations of many market places make them relevant for a large portion of the world.

The MEGA65 project is rated as a marginal level of damage to the project from a law and regulation related risk. This is due to the projects ability to adapt to change quite well and it doesn't have a strict time or cost constraints. If a product was found to breach electronic interference laws for example, the MEGA65 team would be able to redesign and test the new product with only marginal delays in time and cost. The MEGA65 team is aware of this risk and of the most notable examples of these laws, such as electronic interference and age rating testing, but not of the specific laws of each market they are intending to operate in. The project has indicated that they are intending to carry out specific research into the laws and regulations of certain markets. Until this research has been conducted, the project has been rated at a likely level of probability of occurring.

Category	**Rating**
Likelihood	Likely
Possible Damage	Marginal
Risk	High

5.1.2 Brexit-like events

There are no foreseeable Brexit-like events that would be likely to affect the MEGA65 project. Brexit itself is unlikely to affect the MEGA65 project as the project is largely based in Australia, with the desktop version being made in Germany. The potential market for the MEGA65 in the United Kingdom is small and there will likely be some level of agreement with current laws and regulations within the

United Kingdom to reduce to the disruption to the market during the transition out of the EU. The project has been rated as unlikely to experience this type of risk. The potential damage to the MEGA65 project was rated as marginal.

Category	**Rating**
Likelihood	Unlikely
Possible Damage	Marginal
Risk	Low

5.1.3 Crowd-funding

The MEGA65 is considering using crowd-funding to raise funds but importantly they will wait until the design and component selection is complete before launching the campaign.

5.1.4 Crowd-funding: contract of sale

The fact the the MEGA65 team are waiting until much later in the productization process to launch the crowd-funding campaign greatly reduces the risks associated with it. The risk comes from forming a contract of sale between MEGA65 project and backers of the crowd-funding campaign and that contract not being fulfilled. This could lead to legal proceedings against the MEGA65 project. The fact that the MEGA65 project won't launch a crowd-funding campaign until a much later stage in productization means the chance that the MEGA65 project will fail to deliver the product is greatly reduced. This is due to the uncertainty factor being reduced as a lot of the design and component selection is done, as such is not uncertain but certain. The MEGA65 project is rated as having an unlikely probability that it would cause a critical amount of damage to the project.

Category	**Rating**
Likelihood	Unlikely
Possible Damage	Critical
Risk	Moderate

5.1.5 Crowd-funding: currency

If the MEGA65 team does decide to go the crowd-funding route they will need to decide on what currency to operate in. The risk here is the currency exchange rate drops after receiving the funds and now the actual purchasing power of the funds may be reduced. This risk is increased with time, the longer you hold onto the funds without spending them the more chance the exchange rate will fluctuate. As the MEGA65 team will wait until the product is much further along in the productisation process before crowd-funding this time period will be significantly reduced. There are also some currency which are more stable, in general, and

research into these would also reduce the risk by allowing the MEGA65 team to use a stable currency to raise funds in. Another aspect to consider is the currency in which the majority of the funds will be spent i.e. if most of the parts and components are coming from USA companies then it may make sense to use the USD as the crowd-funding currency. The MEGA65 project is rated as unlikely to experience these risks and it would most likely produce a marginal level of damage if it did occur.

Category	**Rating**
Likelihood	Unlikely
Possible Damage	Marginal
Risk	Low

5.1.6 Crowd-funding: failure to meet funding goal

Due to the funding model of the MEGA65 project their is a lot more flexibility in the way the project moves forward after this event, if it did occur. The MEGA65 project has no strict criteria on cost or time to release and should only suffer a delay in time if it did not reach the predetermined campaign goal of a crowd-funding campaign. The MEGA65 project is rated as having a possible chance of this occurring, this depends heavily on the reception of the MEGA65 products to the public and how much publicity the campaign receives. The MEGA65 project is rated as likely to receive a negligible amount of damage from this event, should it occur.

Category	**Rating**
Likelihood	Possible
Possible Damage	Negligible
Risk	Low

5.1.7 Sourcing old components

The MEGA65 project plans to use a 3.5" floppy disk drive as a component in the desktop version of the MEGA65. This is because the Commodore 65 prototype featured an internal 3.5" floppy disk drive. This use of old components brings increased risk due to the part not being in mass-production or common usage as such the availability of stock may be limited and the price may increase dramatically if the component is rare or becomes rare. The risk becomes even greater if the component is no longer in production and it needs to be manufactured for the project. This risk also increases with time as it is almost certain, given a long enough time period, that every component will become obsolete and thus expensive to purchase or produce. The MEGA65 is certain to be exposed to this risk. The project is rated as having a marginal amount of damage caused due to the 3.5" inch floppy still being in production and available to purchase.

Category	Rating
Likelihood	Certain
Possible Damage	Marginal
Risk	High

5.1.8 Use of 3rd party intellectual property

The MEGA65 project has exposure to this risk in several areas. Due to the nature of the MEGA65 project, and retro-computing projects in general, being inspired by products from the past, there is commonly intellectual property that the retro project wants to use or imitate to better replicate their inspiration. This is true for the MEGA65 project, being inspired by the Commodore 64 and 65, there are several areas in which being similar to these machine would greatly help the MEGA65 project. They are listed here:

1. Commodore logo on Commodore key
2. Commodore 64 software including games
3. Commodore 64 character set
4. Commodore 64 BASIC ROM
5. Commodore kernel ROM

Commodore logo on Commodore key

The Commodore 64 and 65 featured a special Commodore key with a Commodore icon on it (sometimes referred to as "C="). The MEGA65 desktop form-factor will need a key to function as the Commodore key if it to run Commodore 64 software. It would help users of the MEGA65 understand what the key is used for if the icon on it made users think of the Commodore key. An exact copy of the Commodore icon would fulfil this perfectly but it is a trademark, along with the Commodore name, as such they both carry risk in using. The MEGA65 project is certain to encounter this risk and there is a critical amount of potential damage that could be caused in using the Commodore icon.

Category	Rating
Likelihood	Certain
Possible Damage	Critical
Risk	Extreme

Commodore 64 software including games

The software in question is specifically applications such as games, text editors or other a productivity software as some non-exhaustive examples, not the kernel or BASIC environment software. The MEGA65 aims to be compatible with as many

Commodore 64 software titles as possible. This allows a more authentic experience for those seeking to recreate their Commodore 64 experiences. To achieve this the MEGA65 needs software to run on it and specifically it needs the software from the home computer era, created for the Commodore 64, to truly recreate the experience. This software is intellectual property as such its use without an agreement brings risks from legal action taken by the IP holders to protect their property. The MEGA65 project is planning on only including with their products, 3rd party software with which they have implicit agreement from the rights holders to use their intellectual property. This could either be from the rights holders stating their software is free to use to the public or through the MEGA65 team reaching out to the rights holders and them forming an agreement. Because of this, the MEGAG65 project has been rated as having a rare likelihood of experiencing this event.

The potential damage caused by this event comes in three parts:
a) The loss of user experiences and value to the users of the MEGA65
b) The loss of time spent removing games from memory, documentation and promotional material
c) The loss of time and money responding to legal challenges and rulings

The MEGA65 project would be expected to experience negligible damage from a) and b). The damage from c) would be marginal as the MEGA65's market is likely to be small so any potential losses to rights holders would also be small which would suggest and legal ruling awarding the rights holders for loss of earnings would also be small.

Category	**Rating**
Likelihood	Rare
Possible Damage	Marginal
Risk	Low

Commodore 64 character set

The MEGA65 requires a set of character bitmaps stored in memory, this is used to display text on the screen. The Commodore 64 character set would be ideal for this purpose as it is readily available from numerous website and would help the MEGA65 replicate the Commodore 64 more closely. The Commodore 64 character set is intellectual property, as such its use in the MEGA65 project brings risk. The MEGA65 project currently does not have an agreement to use the Commodore 64 character set. There is a certain likelihood of this occurring and a marginal level of damage that could occur.

Category	Rating
Likelihood	Certain
Possible Damage	Marginal
Risk	High

Commodore 64 BASIC ROM

The MEGA65 requires a BASIC ROM which will provide the BASIC environment into which it boots into on start up. This ROM, of which there are numerous versions specific to differing versions of BASIC, should be a ROM which provides the greatest amount of compatibility with Commodore 64 and 65 software, as the user will derive greater facility from it. The Commodore 64 BASIC ROM would be the ideal candidate but it is intellectual property and will attract risk if used. The MEGA65 project currently does not have an agreement to use the Commodore 64 BASIC ROM. The MEGA65 project is certain to experience this risk with the potential damage being critical.

Category	Rating
Likelihood	Certain
Possible Damage	Critical
Risk	Extreme

Commodore 64 kernel ROM

The MEGA65 requires this ROM for basic functionality and it is of critical importance to the compatibility of the MEGA65 with Commodore 64 software that the kernel used function as the Commodore 64 kernel does. The ideal candidate kernel if only the functionality of the MEGA65 is considered, is the Commodore 64 kernel. This kernel is intellectual property as such it will attract a risk if used without an agreement with the rights holder. The MEGA65 project does not currently have an agreement with the rights holder. The potential damage from this risk is from the rights holder undertaking legal action against the MEGA65 project. The likelihood of this event occurring is considered unlikely as the ROM has been available online for decades with no noticeable attempt to control its distribution. The potential damage is catastrophic as the MEGA65 could not function without a kernel and if the Commodore 64 kernel was not able to be used suddenly, the MEGA65 would effectively be useless until a replacement was found.

Category	Rating
Likelihood	Unlikely
Possible Damage	Catastrophic
Risk	Extreme

5.1.9 Open-source business

The MEGA65 project will be exposed to this risk, which is related to competitors being able to produce and sell the MEGA65 with no money being generated for the MEGA65 team. This means it can be harder to operate as a traditional electronics business in which its products are propriety and they can discourage competitors using there designs with legal challenges. The MEGA65 project is likely to not have a large market, as such the cost to enter the market is quite high compared to the potential profits. This will help reduce the likelihood of this occurring as long as the assumption that the MEGA65's market is small, remains true. The MEGA65 project is rated as possible for the likelihood of this occurring and a critical level of damage could be caused.

Category	**Rating**
Likelihood	Possible
Possible Damage	Critical
Risk	High

5.1.10 Loss of intellectual property

This relates to the loss of agreement to use intellectual property that is require for the MEGA65. This occurred the most dramatically in the Vega Plus case study where the CTO left RCL, the makers of the Vega Plus, and refused to let them use his prototype. The MEGA65 is entirely shielded from this due to the open-source nature of the designs used throughout the entire project. No team member owns any part of the design as such they cannot withdraw their agreement for MEGA65 to use it. The MEGA65 project is rated as having a rare chance (the lowest) of occurring this event and of it doing negligible damage.

Category	**Rating**
Likelihood	Rare
Possible Damage	Negligible
Risk	Low

5.1.11 Supplier failure

The MEGA65 is exposed to this risk whenever they purchase from a 3rd party. The desktop version which is undergoing physical development with MEGA, and is intended to enter a pre-production stage of development soon, is rated as having a marginal rating of potential damage should this event occur. This is due to the fact the while the desktop version is in pre-production it will have several different components being manufactured by different suppliers. These components need to be assembled together to form the MEGA65 desktop version once they are all complete. A delay in any of the components could cause a time delay in the overall

construction of the MEGA65 desktop version preproduction run. A delay could also incur additional costs if a new manufacturer has to be found or if there are storage costs caused by the delay. The desktop version is rated as having a unlikely chance of this event occurring as the suppliers chosen are established businesses. The other MEGA65 form factor, the MEGAphone, is undergoing the alpha stage of production as such the purchases are currently small as they are focused on a single prototype. This reduces the potential damage to negligible during this phase. As the MEGA65 products enter full production the risk increases. The MEGA65 team members are aware of this risk and of the preference to use established businesses and undertake due diligence. With all the above factors considered, the MEGA65 project is rated as having a marginal level of potential damage and a unlikely chance of this event occurring.

Category	**Rating**
Likelihood	Unlikely
Possible Damage	Marginal
Risk	Low

5.1.12 Open-source fair use

This should not be an issue for the MEGA65 project as all of the open-source components are created by project members or freely available to be used in projects such as the MEGA65.

Category	**Rating**
Likelihood	Rare
Possible Damage	Negligible
Risk	Low

5.1.13 Physical production problems

The MEGA65 project is exposed to this risk in both its form-factors. The desktop version will require physical mechanical components such as a keyboard and case. Both of these components have the potential to experience problems during manufacturing, assembly or testing. These problems such as incorrect material, incorrect size, rough edges or wrong shape to name a few, could potentially cost the project lost time and money. The desktop version has a possible chance of this event occurring and a negligible level of potential damage. The MEGAphone form factor which is much less advanced in its physical design compared to the desktop version, is more exposed to this risk. This is due to the case design having not been created or tested as well as the use of touch screen and button interface, all of which add complexity to the design. The potential damage for the overall project is rated as marginal and it has a possible chance of occurring.

Category	Rating
Likelihood	Possible
Possible Damage	Marginal
Risk	Moderate

5.2 Recommendations for reducing the MEGA65 project's risk exposure

In this section, practical, actionable advice will be given on how the MEGA65 project can reduce their exposure to the risks identified in Chapter 3. Using the risk evaluation of the MEGA65 project as of July 2018, as a starting point, this section aims to reduce the risk by focusing on the highest risk areas and suggesting strategies to reduce that risks.

5.2.1 Laws and Regulations

The MEGA65 project is aware of the need to undertake research into the laws and regulations of the specific markets into which they want to sell the MEGA65. To reduce their risk they must undertake aforementioned research. The potential damage to the MEGA65 project is increased the further the development goes forward without this research. Assuming the research is conducted the likelihood is reduced to rare and a potential risk marginal if the research is conducted in a timely manner.

5.2.2 Sourcing old components

The MEGA65 projects use of floppy disk drives opens up the potential for damage in the future if the become obsolete (which they arguable are already). As they are manufactured less and become rarer, their unit price would be expected to increase. To compensate for this a large inventory of the disk drives could be sourced much earlier than required for production. This could bring twofold benefits in the unit price may come down with a large unit number purchase and the MEGA65 is shielded from component volatility in the medium term (until the stocked drives run out). The downside to this strategy is storage costs as well as increase upfront costs in purchasing the drives.

5.2.3 Use of 3rd party intellectual property

The use of 3rd party software has opened up the MEGA65 project to some high risks. The use of the software is almost unavoidable in cases, if the MEGA65 is to be a true innovation of the Commodore 65 and Commodore 64 compatible. The main areas identified above are the Commodore icon on the Commodore keyboard key and the ROMs required for Commodore 64 compatibility: the character set ROM, BASIC ROM and the kernel ROM. The only options to reduce the risk are to either entered

into an agreement with the rights holder and barring that simple not use them in the MEGA65. Suitable replacements may be found in some cases. The Commodore icon could potential be changed to a MEGA65 inspired design. The character set, while being closely linked to the Commodore 64 experience, could be swapped for another more readily available set. The kernel and BASIC ROM are less likely to have suitable replacements readily available. Another strategy would be to create the ROMs within the MEGA65 project. This would completely eliminate the potential risk, as long as the replacements didn't break any copyright laws.

5.2.4 Open-source business

This risk increases with MEGA65's popularity and market size. At a certain threshold of unit sales other businesses will consider manufacturing the MEGA65 from the open-source designs and selling it themselves. This will not generate any revenue for the MEGA65 team and may be a problem in the long term, depending on the MEGA65's popularity. It may be worth trademarking the MEGA65 name and then later licensing its use to generate profit. The MEGA65 team should also consider branching out into accessories for the MEGA65 as a strategy for dealing with this risk. Dr. Paul Gardner-Stephens and other core team members could also consider commissioned advisory roles and presentations as another strategy.

Chapter 6

Risk evaluation of the MEGA65 project in May 2019

After a 10 month period, the MEGA65 project was re-evaluated against the risks identified in Chapter 3. This period of time gave the project sufficient time to act on the suggestions given at the end of Chapter 5. By re-evaluating the MEGA65 project and comparing the result with the evaluation from July 2019, it is hoped that a conclusion can be drawn on whether the MEGA65 project reduced its risk exposure or not.

6.1 Laws and regulations

The MEGA65 project has undertaken some research into the laws and regulation which need to be upheld in some of their key markets in which they expect to operate. With the knowledge gained from the research, the uncertainty in the relevant laws and regulations is reduced, as such the overall risk to the project is reduced. Not every market was researched, and the unknown markets still expose the project to risk if the MEGA65 where to be sold within it. The MEGA65 project is rated as having an unlikely chance of occurring and if it causing a marginal level of potential damage.

Category	**Rating**
Likelihood	Unlikely
Possible Damage	Marginal
Risk	Low

6.2 Sourcing old components

During the interim 10 months the MEGA65 project purchased 700 3.5" floppy disk drives and has them in storage in Germany. These disk drives are to be used as components during the assembly of the MEGA65 desktop version. This action of purchasing a large surplus of stock allows the MEGA65 project to avoid this risk in the medium term. When the stock runs out the risk will increase, as such the risk is largely tied to the MEGA65's popularity and sales volume. If the MEGA65

becomes hugely popular then this risk could increase dramatically. If the MEGA65 sells as expected by the MEGA65 project team, then 700 drives will be sufficient for the medium to long term. The potential damage the MEGA65 project is rated as marginal and the chance of it occurring possible.

Category	**Rating**
Likelihood	Possible
Possible Damage	Marginal
Risk	Moderate

6.3 Use of 3rd party intellectual property

The MEGA65 project has enacted some of the suggestions from Chapter 5. The steps undertaken in the identified risk areas are discussed below.

6.3.1 Commodore logo

The MEGA65 team decided to use their own design for the logo on the Commodore keyboard key. They have created and used the design for the pre-production run of the desktop form factor. This new design should completely remove the potential for this risk to effect the MEGA65 project.

6.3.2 Commodore 64 character set

The MEGA65 have decided to replace the Commodore character set with one they have created themselves from other freely available character sets and their own work. By removing the Commodore character set the risk is eliminated.

6.3.3 Commodore BASIC ROM and kernel ROM

The MEGA65 team have decided to created their own BASIC and kernel ROMs. This reimplementation is currently under way and can provide basic functionality at this stage. By removing the Commodore ROMs from the MEGA65, the risk is entirely eliminated.

Category	**Rating**
Likelihood	Rare
Possible Damage	Negligible
Risk	Low

6.4 Open-source business

The MEGA65 team have not trademarked the MEGA65 name and are operating under the assumption that the MEGA65 will not be hugely popular. The MEGA65

team have been made aware of the strategies listed in Chapter 5, these strategies are only sensible if the MEGA65 becomes hugely popular and cannot be enacted to much benefit before. Because the MEGA65 team is aware of these strategies the MEGA65 project's potential damage and chance of occurring are both lowered compared to July 2018.

Category	**Rating**
Likelihood	Unlikely
Possible Damage	Moderate
Risk	Low

Chapter 7

Results and discussions

This chapter is dedicated to discussing the results of this thesis, specifically the comparison of risk evaluations conducted on the MEGA65 project, how the thesis answered the research questions and a consideration of the hypothesis.

7.1 Comparing risk evaluations from July 2018 with May 2019

By comparing the risk evaluation of the MEGA65 project from July 2018 with the evaluation from May 2019 it can be seen how the MEGA65 project's risk exposure changed during that time. Table 7.1 shows the risk evaluation results for the risk areas in which the MEGA65's risk exposure changed between July 2018 and May 2019.

It can be seen that 7 of the total 17 risks identified in chapter 5 have been reduced.

TABLE 7.1: Comparing risk evaluations from July '18 and May '19

Risk	**July 2018**	**May 2019**	**Change**
Laws and regulations	High	Low	Reduced
Sourcing old components	High	Moderate	Reduced
Commodore logo	Extreme	Low	Reduced
Commodore character set	High	Low	Reduced
Commodore BASIC ROM	High	Low	Reduced
Commodore kernel ROM	High	Low	Reduced
Open-source business	High	Low	Reduced

7.2 Discussion of results

The results from table 7.1 show that the risk exposure for the MEGA65 project decreased over the ten month period between the first and second risk evaluation. This reduction was caused by the MEGA65 team undertaking actions to mitigate the risks. These actions where spurred by the advice given to the MEGA65 team after the first risk evaluation.

Directly proving how effective the risk evaluation and subsequent advice were in improving the outcomes for the MEGA65 project is difficult, as it is a sample size of one and there is no practical way to have a control group to study. Also, the ten month time frame made it impossible to determine the final outcomes of the project, as such the value of the results is lessened.

7.3 Research questions

This section considers each research question in turn. For each question, an answer and a discussion of how the answer was achieved, is given.

7.3.1 What does the retro-computing project productization process look like?

The retro-computing process is discussed and shown in section 3.3. This process was elicited from the case studies of retro-computing projects seen in chapter 3.

7.3.2 What risk and challenges are associated with a retro-computing project?

The risk and challenges are discussed in section 3.4. The challenges are obtained from the case-studies into retro-computing projects, shown in chapter 3. The risk are then derived from these challenges and shown in figure 3.9.

7.3.3 What is the MEGA65 project?

The MEGA65 project is discussed in detail in chapter 4. The research into the MEGA65 project was largely conducted via conversations with the team members due to the nature of the productization process limiting the amount of documentation generated. A use case study was conducted on the MEGA65 to determine the requirements needed to fulfil the use cases. From these requirements, a snapshot of the state of the MEGA65 as of July 2018 was obtained.

7.3.4 How exposed to risks was the MEGA65 project in July 2018?

A risk evaluation of the MEGA65 project, against the identified risks, was conducted in section 5.1. This evaluation directly answers this question.

7.3.5 What can be done to reduce the MEGA65 project's risks?

Advice on how the MEGA65 project can reduce its risk exposure was given in section 5.2. This advice was arrived at by considering the factors which affect each risk. These factors are outlined in section 3.4. For each factor, possible methods to positivity affect it in regards to risk where determined.

7.3.6 Did the MEGA65 project reduce its risk after 10 months?

A comparison of the two risk evaluations of the MEGA65 project show that the project did reduce its risk over 10 months, table 7.1. The second evaluation focused on the risk areas which where identified as high risk within the first risk evaluation.

7.4 Hypothesis

This section considers the hypothesis stated at the beginning of this thesis in section 1.5. The hypothesis states that outcomes to retro-computing projects will be improved with the creation of a body of knowledge devoted to the retro-computing productization process. To prove this hypothesis the retro-computing project, MEGA65 was studied.

A risk evaluation of the MEGA65 project in July 2018 was conducted. This evaluation focused on the risks identified in the retro-computing case studies. From this evaluation the high risk areas where identified, and advice given on how to mitigate these risks. A second risk evaluation was conducted 10 months later and the results of both compared.

The advice drew from the body of knowledge contained in the case studies and their resulting process and risk discussions contained in chapter 3. The risks with which the MEGA65 was evaluated against where directly resultant from the body of knowledge.

Considering the MEGA65 project had a reduced risk exposure as a result of the advice given in this thesis. This reduced risk exposure is an improved outcome for the MEGA65 project. And considering the advice and risks where both associated with the body of knowledge on retro-computing projects contained in chapter 3, the hypothesis can be considered correct.

Chapter 8

Conclusion and future direction of research

This chapter provides a conclusion and directions for future research.

8.1 Conclusion

As mentioned in section 7.1, the results from the comparison of the risk evaluations show that the MEGA65 project has reduced its risk exposure in the interim ten months, table 7.1. This reduction in risk exposure is an improved outcome for the MEGA65 project. This improved outcome is partly due to the body of knowledge within this thesis, and specifically the identified retro-computing risks and the risk evaluation of the MEGA65, based on these risks. This improved outcome for the MEGA65 retro-computing project proves the hypothesis stated in Chapter 1.

As stated in section 1.7, the contributions this thesis makes to retro-computing projects is categorised as follows:

1. Creates a body of knowledge exploring the retro-computing project productization process.
2. Creates a body of knowledge discussing the risks and challenges associated with retro-computing projects.
3. A risk evaluation of a specific retro-computing project, the MEGA65.
4. Practical, actionable recommendations to reduce the MEGA65 project's risk exposure.
5. Evidence of the benefit of those recommendations, gathered through a follow-up risk evaluation of the MEGA65 several months after providing the recommendations.

8.2 Future direction of research

A possible future direction of research is further testing of the hypothesis, by undertaking another study of the MEGA65 at the end of the productization process,

to determine if the MEGA65 did suffer from any of the events described by the risks identified in the case studies. This body of knowledge could also be given to other retro-computing projects at the beginning of their productization process. The projects could then be studied to determine if the body of knowledge provides any benefit to the outcomes of these projects.

Another direction that research could continue is to further elaborate on the body of knowledge contained within this thesis. As there was no prior work done in this area, the aim was to cover a broad area with as much depth as was possible within the time frame. This leaves a lot of areas within that could be potential areas of future research. The identified risks could be elaborated into many more risk categorises with much more description. The process identified in the case study could also be elaborated with each step offering a possible area of research.

Bibliography

[1] E. E. Ogheneovo, "On the relationship between software complexity and maintenance costs," *Journal of Computer and Communications*, vol. 02, p. 1, 2014. http://file.scirp.org/Html/1-1730107_51631.htm.

[2] C. Magerkurth, T. Engelke, and M. Memisoglu, "Augmenting the virtual domain with physical and social elements: towards a paradigm shift in computer entertainment technology," in *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pp. 163–172, ACM, 2004.

[3] J. Suominen, M. Reunanen, and S. Remes, "Return in play: The emergence of retrogaming in finnish computer hobbyist and game magazines from the 1980s to the 2000s," *Kinephanos. http://www. kinephanos. ca/2015/emergence-of-retrogaming*, 2015.

[4] M. of Electronic Games and e. Art, "Mega65 - (most probably) the best computer," 2019.

[5] R. W. Veryzer Jr, "Discontinuous innovation and the new product development process," *Journal of Product Innovation Management: AN INTERNATIONAL PUBLICATION OF THE PRODUCT DEVELOPMENT & MANAGEMENT ASSOCIATION*, vol. 15, no. 4, pp. 304–321, 1998.

[6] K. Imai, I. Nonaka, and H. Takeuchi, *Managing the new product development process: how Japanese companies learn and unlearn*. Division of Research, Harvard Business School Boston, MA, 1984.

[7] M. A. Schilling and C. W. Hill, "Managing the new product development process: strategic imperatives," *Academy of Management Perspectives*, vol. 12, no. 3, pp. 67–81, 1998.

[8] J. M. Morgan and J. K. Liker, *The Toyota product development system*, vol. 13533. New York: Productivity Press, 2006.

[9] D. Zahay, N. Hajli, and D. Sihi, "Managerial perspectives on crowdsourcing in the new product development process," *Industrial Marketing Management*, vol. 71, pp. 41–53, 2018.

[10] A. F. Sommer, C. Hedegaard, I. Dukovska-Popovska, and K. Steger-Jensen, "Improved product development performance through agile/stage-gate hybrids: The next-generation stage-gate process?," *Research-Technology Management*, vol. 58, no. 1, pp. 34–45, 2015.

[11] J. Stark, "Product lifecycle management," in *Product lifecycle management (Volume 1)*, pp. 1–29, Springer, 2015.

[12] B. Rajagopalan and B. L. Bayus, "Exploring the open source product development bazaar," in *Review of Marketing Research*, pp. 78–94, Routledge, 2017.

[13] A. Chahin, J. Hoffmeister, K. Paetzold, N. Noori, X. Vilasis Cardona, *et al.*, "A practical approach to structure the product development process using network theory," in *DS 84: Proceedings of the DESIGN 2016 14th International Design Conference*, pp. 1235–1242, 2016.

[14] A. Gupta and H. M. D. Toong, "The first decade of personal computers," *Proceedings of the IEEE*, vol. 72, no. 3, pp. 246–258, 1984.

[15] J. S. C. Kilby, "Turning potential into realities: The invention of the integrated circuit (nobel lecture)," *ChemPhysChem*, vol. 2, no. 8-9, pp. 482–489, 2001.

[16] G. E. Moore, "Cramming more components onto integrated circuits," *Proceedings of the IEEE*, vol. 86, no. 1, pp. 82–85, 1998.

[17] I. Corporation, "The story of the intel® 4004," 2018. https://www.intel.co.uk/content/www/uk/en/history/museum-story-of-intel-4004.html.

[18] Intel, "Intel chips timeline poster," 2012. https://www.intel.com/content/dam/www/public/us/en/documents/corporate-information/history-intel-chips-timeline-poster.pdf.

[19] R. C. Dorf, *The Engineering handbook*. Boca Raton: Boca Raton : CRC Press, 1995.

[20] J. Brennan and M. Molloy, "Microcomputers," *Interfaces*, vol. 13, no. 1, pp. 28–39, 1983.

[21] R. Sugarman, "Does the country need a good s20 microprocessor?," 1975. https://www.commodore.ca/gallery/magazines/misc/mos_605x_team_eetimes_august_1975.pdf.

[22] S. Cass, "The golden age of basic," *IEEE Spectrum: Technology, Engineering, and Science News*, 2014.

[23] T. E. Kemeny, John G.; Kurtz, "Basic: a manual for basic, the elementary algebraic language designed for use with the dartmouth time sharing system," 1964. http://bitsavers.trailing-edge.com/pdf/dartmouth/BASIC_Oct64.pdf.

[24] M. Swalwell, "Questions about the usefulness of microcomputers in 1980s australia," *Media International Australia incorporating Culture and Policy*, pp. 63+, May 2012.

[25] "How many commodore 64 computers were really sold? | pagetable.com," 2018. https://www.pagetable.com/?p=547.

[26] G. W. R. Limited, "Guinness world records: Commodre 64 record search results," 2018. http://www.guinnessworldrecords.com/search/applicationrecordsearch?term=commodore+64&contentType=record.

[27] *InfoWorld*, vol. 4, No. 4. InfoWorld Media Group, Inc., 1982. https://books.google.com.au/books?id=dT4EAAAAMBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q=commodore%2064&f=false.

[28] G. Rottensteiner, "Georg rottensteiner - gr games," 2013. http://www.georg-rottensteiner.de/en/index.html.

[29] "Lemon64.com - all about commodore 64," 2018. https://www.lemon64.com.

[30] "Rgcd," 2018. https://www.rgcd.co.uk.

[31] "Run magazine issue 1986-special," 1986. https://archive.org/stream/run-magazine-1986-special/Run_Issue_Special_2_1986#page/n73/mode/2up.

[32] B. Hill, "Ultra-rare working commodore c65 prototype pops up on ebay for 25k euro https://hothardware.com/news/ultra-rare-working-commodore-c65-prototype-pops-up-on-ebay-for-25k-eu.

[33] "Compute! gazette issue 75," 1989. https://archive.org/details/1989-09-computegazette/page/n0.

[34] C. Kaiser, "Secret weapons of commodore: The commodore 65," 2018. http://www.floodgap.com/retrobits/ckb/secret/65.html.

[35] "r/c64 - mega 65: Commodore 65 remake gets a physical release," 2018. https://www.reddit.com/r/c64/comments/33kjp8/mega_65_commodore_65_remake_gets_a_physical.

[36] P. Gardner-Stephen, "Some screen shots," 2018. http://c65gs.blogspot.com/2014/01/some-screen-shots.html.

[37] P. Gardner-Stephen, "Motivations and goals for the project," 2018. http://c65gs.blogspot.com/2014/01/motivations-and-goals-for-project.html.

[38] P. Gardner-Stephen, "Mega65 fpga computer user manual," 2016. https://github.com/MEGA65/mega65-core/blob/master/doc/usermanual0.md#11-purpose.

[39] "Introducing the mega65 (8-bit) computer | mega - museum of electronic games & art," 2018. http://www.m-e-g-a.org/mega65-introduction.

[40] D. P. Gardner-Stephen, "Making a c64/c65 compatible computer - april, 2015," 2018. http://c65gs.blogspot.com/2015/04.

[41] G. E. Moore, "Progress in digital integrated electronics [technical literaiture, copyright 1975 ieee. reprinted, with permission. technical digest. international electron devices meeting, ieee, 1975, pp. 11-13.]," *Solid-State Circuits Society Newsletter, IEEE*, vol. 11, no. 3, pp. 36–37, 2006.

[42] www.techpowerup.com, "Amd ryzen threadripper 2990wx," 2018. https://www.techpowerup.com/cpudb/2069/ryzen-threadripper-2990wx.

[43] H. Iwai, "Roadmap for 22nm and beyond (invited paper) vol. 86, no. 7, pp. 1520–1528, 2009. http://www.sciencedirect.com/science/article/pii/S0167931709002950.

[44] A. b. Huang, "The death of moore's law will spur innovation," *IEEE Spectrum: Technology, Engineering, and Science News*, 2015. https://spectrum.ieee.org/semiconductors/design/the-death-of-moores-law-will-spur-innovation.

[45] M. R. Lyu, "Book: Handbook of software reliability engineering," 1996. http://www.cse.cuhk.edu.hk/~lyu/book/reliability.

[46] B. Schneier, "Software complexity and security," 2000. https://www.schneier.com/crypto-gram/archives/2000/0315.html#8.

[47] B. Schneier, "A plea for simplicity: You can't secure what you don't understand.," 1999. https://www.schneier.com/essays/archives/1999/11/a_plea_for_simplicit.html.

[48] H. B. Lawson, "The march into the black hole of complexity," *Commun. ACM*, vol. 61, no. 5, pp. 43–45, 2018.

[49] P. Gelsinger, D. Kirkpatrick, A. Kolodny, and G. Singer, "Such a cad!," *Solid-State Circuits Magazine, IEEE*, vol. 2, no. 3, pp. 32–43, 2010.

[50] R. W. Shirey, "Internet security glossary, version 2," 2018. https://tools.ietf.org/html/rfc4949.

[51] S. Australia, "Iso/iec 27005 information technology - security techniques - information security risk management," 2011.

[52] S. Australia, "Iso/iec 18028.1 information technology - security techniques - it networks security part 1: Network security management," 2008.

[53] S. Australia, "Iso/iec 18028.5 information technology - security techniques - it networks security part 5: Securing communications across networks using virtual private networks," 2008.

[54] P. Apps, "Firms lose more to electronic than physical theft," 2018. https://www.reuters.com/article/us-crime-fraud/firms-lose-more-to-electronic-than-physical-theft-idUSTRE69H25820101018.

[55] S. N. M. o. A. History, "Log book with computer bug," 2018. http://americanhistory.si.edu/collections/search/object/nmah_334663.

[56] F. Cohen, "Computer viruses: Theory and experiments," *Computers & Security*, vol. 6, no. 1, pp. 22–35, 1987.

[57] E. Filiol, *Computer viruses: from theory to applications*. Paris: Paris: Springer Paris, 2005.

[58] J. Shoch and J. Hupp, "The "worm" programs-early experience with a distributed computation," *Communications of the ACM*, vol. 25, no. 3, pp. 172–180, 1982.

[59] A. Young and Y. Moti, "Cryptovirology: extortion-based security threats and countermeasures," 1996.

[60] "Computer security threats: A brief history - direct2dell," 2014. https://blog.dell.com/en-us/computer-security-threats-a-brief-history.

[61] "Malware of the 1980s: A look back at the brain virus and the morris worm," 2018. https://www.welivesecurity.com/2018/11/05/malware-1980s-brain-virus-morris-worm.

[62] J. K. Reynolds, "Helminthiasis of the internet," 1989. https://tools.ietf.org/html/rfc1135.

[63] "Website for storing digital currencies hosted code with a sneaky backdoor," 2019. https://arstechnica.com/information-technology/2019/05/website-for-storing-digital-currencies-hosted-code-with-a-sneaky-backdoor.

[64] "Cyber security – parliament of australia," 2013. https://www.aph.gov.au/About_Parliament/Parliamentary_Departments/Parliamentary_Library/pubs/BriefingBook44p/Cyber.

[65] "Cybersecurity," 2012. https://www.dhs.gov/topic/cybersecurity.

[66] "How digital detectives deciphered stuxnet, the most menacing malware in history," 2018. https://arstechnica.com/tech-policy/2011/07/how-digital-detectives-deciphered-stuxnet-the-most-menacing-malware-in-history.

[67] D. E. Sanger, "Obama ordered wave of cyberattacks against iran," *N. Y. Times*, 2012. https://www.nytimes.com/2012/06/01/world/middleeast/obama-ordered-wave-of-cyberattacks-against-iran.html.

[68] "Australians warned not to be complacent as ransomware victims come forward," 2017. https://www.abc.net.au/news/2017-05-14/ransomware-cyberattack-threat-lingers-as-people-return-to-work/8525554.

[69] G. U. o. Technology, "Meltdown and spectre," 2018. https://meltdownattack.com/.

[70] B. Schneier, "Cryptography is harder than it looks," *IEEE Security & Privacy*, vol. 14, no. 1, pp. 87–88, 2016.

[71] "Arduino - aboutus," 2019. https://www.arduino.cc/en/Main/AboutUs.

[72] D. Kushner, "The making of arduino," *IEEE Spectrum: Technology, Engineering, and Science News*, 2011.

[73] H. Barragán, "The untold story of arduino," 2016. https://arduinohistory.github.io.

[74] H. Barrag n, "Wiring," 2019. http://wiring.org.co.

[75] "Arduino the documentary (2010) english hd," 2019. https://vimeo.com/18539129.

[76] C. Thompson, "Build it. share it. profit. can open source hardware work? wired," 2008. https://www.wired.com/2008/10/ff-openmanufacturing.

[77] "Arduino srl to distributors: "we're the real arduino" 2015. https://hackaday.com/2015/03/28/arduino-srl-to-distributors-were-the-real-arduino.

[78] "Two arduinos become one https://blog.arduino.cc/2016/10/01/two-arduinos-become-one-2.

[79] "Raspberry pi's $35 linux computer on track to launch later this month," 2019. https://arstechnica.com/information-technology/2012/02/raspberry-pis-35-linux-computer-on-track-to-launch-later-this-month.

[80] "Raspberry pi credit-card sized linux pcs are on sale now, $25 model a gets a ram bump," 2019. https://www.engadget.com/2012/02/29/raspberry-pi-credit-card-sized-linux-pcs-are-on-sale-now-25-mo.

[81] N. Heath, "Inside the raspberry pi: The story of the $35 computer that changed the world," *TechRepublic*, 2019. https://www.techrepublic.com/article/inside-the-raspberry-pi-the-story-of-the-35-computer-that-changed-the-world.

[82] T. C. f. C. History, "Eben upton - the story of raspberry pi," 2016. https://www.youtube.com/watch?v=UCt6d0SCxO4.

[83] "Raspberry pi - 2006 edition - raspberry pi https://www.raspberrypi.org/blog/raspberry-pi-2006-edition.

[84] "Raspberry pi foundation - about us https://www.raspberrypi.org/about.

[85] R. Cellan-Jones, "Raspberry pi - the £15 computer," 2011. https://www.youtube.com/watch?v=pQ7N4rycsy4.

[86] htxt.africa, "The history of raspberry pi by eben upton - raspberry jam south africa," 2017. https://www.youtube.com/watch?v=5hSCw9O1tok.

[87] W. Staff, "Raspberry pi's secret: 'sell out a little to sell a lot' wired," 2018. https://www.wired.com/2012/09/raspberry-pi-insider-exclusive-sellout-to-sell-out.

[88] "Sinclair zx spectrum vega https://www.indiegogo.com/projects/sinclair-zx-spectrum-vega#.

[89] G. Corfield, "Is this a wind-up? planet computers boss calls time on zx spectrum reboot firm https://www.theregister.co.uk/2019/02/05/retro_computers_ltd_wound_up_private_planet.

[90] G. Corfield, "Crowdfunding small print binned as retro computers ltd loses court refund action https://www.theregister.co.uk/2018/01/31/crowdfunding_court_case_refund_retro_computers_zx_vega_plus.

[91] G. Corfield, "Zx spectrum reboot scandal: Directors quit, new sack effort started https://www.theregister.co.uk/2018/08/28/zx_spectrum_reboot_directors_quit.

[92] G. Corfield, "Zx spectrum vega+ blows a fuse: It runs open-source emulator the register," 2018. https://www.theregister.co.uk/2018/08/09/zx_spectrum_vega_plus_hands_on_review.

[93] "The sinclair zx spectrum vega plus console https://www.indiegogo.com/projects/the-sinclair-zx-spectrum-vega-plus-console#.

[94] "Vega+ prototype - virtual keyboard https://www.flickr.com/photos/89823678@N03/32060275114/in/album-72157680247094916.

[95] "Chris smith and paul andrews resign as retro computers directors gamesindustry.biz," 2019. https://www.gamesindustry.biz/articles/2016-06-21-chris-smith-and-paul-andrews-resign-as-retro-computers-directors.

[96] G. Corfield, "Indiegogo grants zx spectrum reboot firm another two weeks to send a console https://www.theregister.co.uk/2018/06/01/indiegogo_extends_zx_spectrum_vega_plus_deadline.

[97] G. Corfield, "Zx spectrum reboot scandal: Directors quit, new sack effort started https://www.theregister.co.uk/2018/08/28/zx_spectrum_reboot_directors_quit.

[98] "Vega+ to be stripped of sinclair brand," 2019. https://www.bbc.com/news/technology-45024267.

[99] "Update 15: Time for a mighty update! · zx spectrum next 2019. https://www.kickstarter.com/projects/1835143999/zx-spectrum-next/posts/1946308.

[100] "Zx spectrum next https://www.kickstarter.com/projects/1835143999/zx-spectrum-next/description.

[101] "Zx spectrum next – the official portal for all things next," 2019. https://www.specnext.com.

[102] "Update 27: More information on case production and games! · zx spectrum next https://www.kickstarter.com/projects/1835143999/zx-spectrum-next/posts/2162757.

[103] "Update 18: Latest progress and shipping date · zx spectrum next kickstarter," 2019. https://www.kickstarter.com/projects/1835143999/zx-spectrum-next/posts/2009473.

[104] "Update 42: Kicking off 2019 · zx spectrum next https://www.kickstarter.com/projects/1835143999/zx-spectrum-next/posts/2391252.

[105] "Update 40: Project status: an x-ray (and the latest) · zx spectrum next kickstarter," 2019. https://www.kickstarter.com/projects/1835143999/zx-spectrum-next/posts/2347612.

[106] "Thec64 - computer and games console https://www.indiegogo.com/projects/thec64-computer-and-games-console#.

[107] "Thec64 - computer and games console https://www.indiegogo.com/projects/thec64-computer-and-games-console#/updates/all.

[108] "The commodore 64 bounces back to life as a direct-to-tv™ plug and play joystick! https://www.gamesindustry.biz/articles/the-commodore-64-bounces-back-to-life-as-a-direct-to-tv-plug-and-play-joystick.

[109] "Thec64 - computer and games console https://www.indiegogo.com/projects/thec64-computer-and-games-console#/updates/all.

[110] "Commodore info page - articles: C64 - dtv [en]," 2019. https://www.richardlagendijk.nl/cip/article/item/c64dtv/en.

[111] "C64dtv in original c64 case," 2019. http://joco.homeserver.hu/mmc2iecKB.

[112] "C-64 dtv hacking," 2016. http://www.chrismcovell.com/c64dtv.html.

[113] "Mega65/megawat https://github.com/MEGA65/MegaWAT.

[114] S. Australia, "As iso 31000:2018 risk management—guidelines," 2018.