

CM1205: Architecture and Operating Systems

Structured Portfolio **(100%)**

Name: Semih Sarisoy | Date: 05/05/2023

## Week 1: Introduction and history of computer development

### Reflection on Learning Materials

During the first week of the CM1205 Architecture and Operating Systems module, we were introduced to the history of computers and their evolution. Although I had already learned most of these concepts in the previous CM1101 Computational Thinking module, I still found this week to be informative. I learned about the differences between computers and calculators, and how different computer architecture types were created to solve different problems. One of the most important architecture types we learned about is Von Neumann Architecture, which is used in most modern computers.

Diagram

Description automatically generatedTo further understand Von Neumann's Architecture, I watched a video on Computerphile (2018) by Professor Brailsford from Nottingham University. The video explained the five requirements needed to construct a computer using this architecture: Memory, Control Unit (CU), Arithmetic Logic Unit (ALU), Input and Output. It also delved into the history of architecture, detailing how it was invented by mathematician John Von Neumann after World War 2. Von Neumann started working on the architecture to help all the scientists who had gathered to build a general-purpose all-electronic computer. The Electronic Discrete Variable Automatic Computer (EDVAC) was one of the first machines to use Von Neumann's Architecture. A summer school at Moore School of Engineering was held to discuss the future of computers, specifically EDVAC, which led to the widespread adoption of the Von Neumann Architecture.

Overall, reflecting back to myself, this week's introductory module taught me valuable information about the history of computers and how they evolved into the machines we use today. I also gained a deeper understanding of Von Neumann Architecture and its significance in modern computing. This knowledge will be useful as I continue to study architecture and operating systems, and I am excited to explore more advanced topics in the weeks to come. Learning architectures used in computer will also be useful for my further modules such as CM1301 Principles, Tools and Techniques for Secure Software Engineering where we learn the black box testing which looks like Von Neumann Architecture since there is an input and an output where we use them to predict in software engineering. I also believe I will be using these fundamental knowledge in my future projects and jobs.

## Week 2: Structure of computers

### Reflection on Learning Materials

Diagram

Description automatically generatedThroughout this week, I revisited important concepts such as Von Neumann architecture and computer-level hierarchy, while learning new topics such as Logic Gates and Arithmetic Circuits. To increase my understanding of Arithmetic Circuits, based on my interests, I created a demo circuit on a full adder using logic.ly. Designing the circuit was challenging yet rewarding and testing it helped me understand how computers perform arithmetic operations at the hardware level. This project gave me a deeper understanding of circuits, logic gates, and transistors, and helped me understand the mindset of programmers and engineers. I also realized the importance of hands-on learning in understanding complex concepts and the significance of the building blocks of modern technology. Reflecting on myself, this experience helped me develop critical thinking skills. I now can use websites like logic.ly in my future tasks to create working diagrams and troubleshoot issues. Overall, this project has been a valuable learning experience that has deepened my understanding of computer hardware and helped me develop important skills for my future career. The hands-on experience of designing and testing a circuit has given me a better understanding of how computer hardware functions and has helped me appreciate the importance of hardware in computer technology.

### Reflection on Seminar Discussions

During this seminar, we were tasked with teaching older individuals about the history of computing. As part of a team of four, my teammates and I quickly realized that the task would be straightforward since we were already familiar with the subject matter. We began by constructing a set of talking points that would allow us to explain the history of computing to our audience. We focused primarily on the WW2 perspective since Ms. Marple, an older woman interested in history, was in attendance. My teammates decided to begin by explaining the work of Charles Babbage and Ada Lovelace, who built the first Turing-complete computer together. I followed by sharing the biography of Alan Turing, who built a computer for breaking German code during WW2. Thanks to our preparation, our team did not encounter any significant difficulties during the task. However, my teammates discovered that Ms. Marple was 20 years old when the first personal computer was invented, which was useful information to consider when teaching her. This discovery allowed us to adapt our teaching approach to better suit her understanding and knowledge level. Overall, the experience was a positive learning opportunity for all of us. Reflecting to myself, I realized that in the future, I can use common interests with others to explain the history of computing more effectively. For example, if someone is interested in knitting, I could use the example of punch cards, which are like knitting patterns, to explain how computers use them to store data. This task has taught me the importance of considering the audience when presenting complex information and how to adapt my teaching style to make it accessible and engaging for different people.

## Week 3: assembly Language (Overview)

### Reflection on Learning Materials

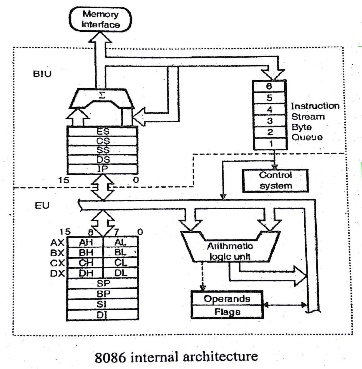
This week, after completing the initial slides, I watched "What Is Assembly Language?" by "javidx9" to understand how assembly language communicates with the hardware. The video explained the role of CPU registers and how RAM stores data in addresses and data fields. One critical thinking skill I developed was the ability to identify components of a computer system. I appreciated assembly language's low-level communication with hardware compared to high-level languages. This understanding highlights the complexity of computer systems and the importance of low-level comprehension. The video introduced me to how loops and arithmetic operations in assembly language work with registers. I found it fascinating how two registers could multiply numbers and how it relates to high-level languages. This connection between assembly language and high-level languages revealed similarities between programming languages and how they can solve complex problems. Reflecting on my learning experience, I realized that what I learned is vital to understanding computer science. This video deepened my appreciation for programming languages in computer systems and how they interact with hardware. I am eager to continue learning assembly language and other low-level programming languages. In conclusion, "What Is Assembly Language?" has helped me understand assembly language's role in programming computer systems and how it communicates with the hardware. This provided me with a good introduction to Assembly Language which developed my critical thinking skills and increased my appreciation for computer systems' complexity and the importance of understanding how they work at the lowest level just before machine code.

### Reflection on Seminar Discussions

This week's task was to explain to a customer how to compare CPUs beyond brand names and prices. We explained technical specifications, such as core count, clock speeds, memory support, instruction set extensions, hardware security features and more. In the process, I learned the importance of critical thinking and providing value to customers by helping them make informed decisions. We compared the i5 and i7 CPUs and found that the i5 CPU, which was five generations ahead, performed better due to its greater number of transistors and better instructions. Reflecting on this experience, I realized that technical knowledge and critical thinking skills are essential in computer science. Understanding technical specifications enable us to provide valuable advice to customers and make informed decisions. I also gained an appreciation for the importance of considering a CPU's instruction set extensions. The i5's support for AVX2 and Intel Boot Guard made it superior to the i7 in terms of performance and security. With this task, I have used my pre-knowledge to critically think about how to explain some of that knowledge to someone who doesn't know. Overall, this task taught me the importance of having a deep understanding of technical specifications when making decisions related to computer hardware. In the future I can use what I learned from this task to build myself a computer by comparing parts and deciding what to use. I also realized that being able to effectively communicate this knowledge to customers is critical in providing value and building trust.

## Week 4: assembly Language AND INTEL 8086 ASSEMBLY LANGUAGE

### Reflection on Learning Materials

During this week I have been introduced to some concepts in the Assembly Language of Intel 8086 microprocessors and the architecture of Intel 8086. The slides for this week briefly explained the architecture of the 8086 Microprocessor and certain registers used in 8086. I wanted to dive deeper into the 8086 architecture and understand what is going on with a more detailed explanation. I have watched a video from the “Bharat Acharya Explanation” channel called “8086 Microprocessor Architecture” which explains the 8086 architecture and how it processes Assembly instructions. One of the most interesting concepts that the video explained was how to register A can be used as either AX which holds a 16-bit number or as AH and AL which both hold an 8-bit number. This was also explained in the slides. Another concept which was in the video was how the control section decodes opcode (e.g., MOV, AH) but does not decode the operand. Learning the architecture of 8086 filled some gaps in my knowledge about registers, flags and how they work. I will apply what I learned in my future work and assignments. As a result of this new information, I learned about CPU architecture from now on before any programming-related task (e.g., Assembly) I will check the CPU’s architecture to understand the compilers and assembly language better. After understanding the architecture, I started doing the lab assessment which included some assembly programming. At first, I struggled a little bit because assembly is a low-level programming language, and it requires some changes through Visual Studio before running such as changing 64x to 84x since 8086 and similar Intel processors use x86 architecture which does not work with 64x which is for more modern CPU’s. But because I had already understood the architecture of 8086, I was able to program assembly on another intel processor which had similar architecture without too much trouble. Using visual studio code, I was able to see the registers and flags change as assembly language was run step by step. This experience helped me understand the system structure by running the assembly step by step. I will be applying this assembly experience in the future since learning assembly language is always going to be relevant, especially 8086 architectures considering it is still relevant from 1978 when it was first introduced. Critically thinking about the assembly language, I learned that it works step by step which made me understand how CPU processes instructions step by step where first converts programs into raw binary codes 0’s and 1’s. These contributed to my learning in a positive way since learning how a CPU (which is known as the brain of the computer) executes instructions and does calculations with ALU. In the future, I am planning on learning more modern architecture and writing assembly on it where the information I learned in this task will be helpful. I am planning on learning the Nehalem architecture of intel and programming some assembly on it since it supports both 64x and 86x platforms. Overall, I learned new things about CPU architecture and assembly language this week.

## Week 5: ADDRESSING MODES, LOOPS AND JUMPS

### Reflection on Learning Materials

Text

Description automatically generatedThis week we were introduced to some programming concepts, that we learned from high-level languages, with assembly. After completing the slides and learning the materials to get more familiar with the basics I decided to read a paper called “Introduction to Microprocessors and Assembly Language” which explains segments of an assembly program, if-else statements, and loops in assembly language. According to the paper, assembly code has 3 segments stack, data, and code. The stack segment contains data in storage, the code segment has the actual code, data segment contains initial static and global variables. The paper then explains the Structured program theorem which proves any logic problem can be solved with only selection and iteration which is if-else statements and loops. This is an idea used by assembly since assembly relates to all modern languages nowadays because compilers of the C/C++ family include assembly code. The paper then starts explaining the if-statements and some useful commands such as jg, jl, and jne which correspond to jump if greater, jump if less and jump if not equal to zero which then eventually corresponds to mathematical comparison operators <= and >=. Loops are done by using if-else statements inside where the structure is done control loop operator incrementing an operator and comparing with an integer/value and each time a certain condition is reached this could be the control loop operator being smaller or bigger than a certain integer. For achieving loops in assembly the paper uses jmp, inc and cmp operators which correspond with an unconditional jump, incrementing by one and comparing. For the Lab Exercise part of this week, we had 3 exercises first one was about if-else statements, the second one was about loops, and the third exercise was about taking a keyboard input and printing to screen using assembly language by combining two files one for input/taking keyboard input and the other for output/printing to screen. When doing the lab exercises, I had some issues such as Visual Studio not working for no reason when I am trying to open a new file, but I overcome these issues by reverting the settings. I believe troubleshooting these problems that occur increases my experience with general computer and assembly language and how it works. I have heard about the use of assembly language in many fields of computer science such as making an operating system where one example is MenuetOS or making a video game such as RollerCoaster Tycoon game where 99% of the game was written with x86 assembly. The assembly might be a harder language to learn because its syntax is closer to true machine language which is 0s and 1s but it also has a lot of advantages such as aiding engineers in reverse engineering, controlling the hardware better and the general performance benefits. Reflecting on my work I am planning on using assembly in projects such as my CM1301 Principles Tools and Techniques for secure Software Engineering module since assembly language is one of the best languages for reverse engineering other software that was not written by me. Overall, I think assembly allows us to understand computers in a much more detailed way and it is a super useful language for those who want more details.

## Week 6: PROCEDURES AND SUBROUTINES, STRING INSTRUCTIONS

### Reflection on Learning Materials

This week we were introduced to procedures, subroutines (which are like functions or methods in high-level languages) and string instructions in assembly language. This week the slides also included code examples to help us understand the materials better. But before checking the lab exercise materials I decided to first improve my assembly language knowledge by reading “Guide to Assembly A concise Introduction” book written by James T. Streib. Part 7 and Part 9 of the book were about procedures and strings. The book starts by explaining that a procedure is used to not repeat code. According to the book, procedures can have more than one return statement but it is not advised since it might break the program.

Text, letter

Description automatically generatedThe book also explains what a “dead code” means which is the code that comes after the return statement. A dead code will not run each time a procedure is called since it is after the return statement it will consume space but do nothing. A good programmer should avoid writing dead code in any programming language since it is wasteful and makes debugging hard. The book introduces strings with assembly by explaining that strings are just an array of bytes. According to the book, strings can be indexed by edi and esi registers. And then the book introduced me to the string instructions such as movsb, scasb, lodsb, and stosb. Learning these concepts will help me with my problem-solving and critical-thinking skills. Reflecting to myself I learned that some of the high-level concepts are constructed by assembly language and learning assembly language helps me understand any language better since almost all languages share the same structure and assembly is one of the first languages that helped computer programmers save time since they don’t have to learn binary to operate computers. After finishing the strings part of the book, I started doing the lab project. The first part of the lab project required us to print an input name after a quote. This is one of the programming questions for beginners. Since I was still getting used to assembly language the question made me go out of my comfort zone and research different ways of approaching it. Eventually, I realized that if I divide the question into smaller parts just like how algorithms work it would make my task easier. So first I had taken the user's name as input from the user. Then I have written a piece of code that let me print the quote. Afterwards, I used another piece of code to help me print the taken input. Overall reflecting back to myself, this lab exercise has taught me a lot since compared with other high-level languages assembly requires direct hardware interaction. I was able to see how a simple program was able to change the registers and flags values to store and retrieve information. I believe my problem-solving skills were significantly increased with this programming exercise. I will be able to use these problem-solving and critical-thinking skills to complete other projects in the future. I was also able to understand the types of languages better such as functional, object-oriented, imperative, and declarative. Currently, I am planning on solving some other code exercises that I solved with other high-level languages using assembly.

## Week 8: Memory, Secondary memory, I/o bus

### Reflection on Learning Materials

### For this week’s topic, we were introduced to computer memory and the hierarchy of computer memory. The slides have explained the hierarchy as registers, cache, main memory (RAM), Magnetic Discs (Hard drives) and Tapes/Optical Discs. Another part which caught my interest was the secondary memory which is used for long-term storage. One example of this is SSD and another more popular example is the hard disc.

Diagram

Description automatically generatedWe were introduced to hard drives in slides, but I watched a TED-Ed video called “How Hard Drives Work?” to understand further. The video explains how hard disc drives can store data and what should we do to increase the storage space and memory speed. It also explains some techniques used in HDD such as thin film lithography to shrink reader/writer and also perpendicular recording to prevent data loss. Reflecting on myself I think understanding hardware is an important step to understanding how computers work. I believe another important thing to consider is the demand for some technology to improve or invents it. Back in the days when computers did not have secondary memory, they were not used for long-term storage only short-term calculations (e.g. for Astronomy). When our tasks got complex scientists decided to use long-term storage in computers (e.g., Hard Drives). And after some time, HDDs started to get replaced by SSDs since they are faster than mechanical parts of HDDs. Hardware will always get replaced by other hardware but learning how hardware evolved is always going to be important for being able to think critically and find better ways to achieve goals.

### Reflection on Seminar Discussions

Chart

Description automatically generated with medium confidenceThis week in our seminar discussion task we were required to design an infographic sheet for registers in the 8086 processor. We were advised to use PowerPoint since it is easy enough for beginner designers. I decided to focus on the 4 general-purpose registers that are also used to store data. Accumulator, base, count, and data registers are used in 8086 processors to both have a separate task from each other and can be used to store data. When storing data with these registers they are separated into two sections H and L which stand for low and high bytes. Either a 16-bit quantity of X can be used or two sections H and L for two 8-bit quantities. What makes these registers useful is also shown in the infographic. For a high amount of small numbers, two sections are more efficient for a lower amount of large numbers using them as one quantity is useful. Reflecting on myself, I have learned a lot from this task since it allowed me to visually understand how registers work and how they can be separated into two parts or stay as a single section. Scientists and Engineers use their problem-solving, critical thinking and decomposition skills to use materials efficiently, including hardware and software. I believe I have gained problem-solving, critical thinking and decomposition skills from this task since I visually understood the idea of a register rather than on paper. Overall, this task was quite educational and important to understand registers and how they work better.

## Week 9: Memory, Secondary memory, I/o bus (cont’)

### Reflection on Learning Materials

A picture containing indoor, electronics

Description automatically generatedThis week we continued to learn about memory and input/output devices which transfer data to/from the computer. This week the slides introduced us to different storage devices that store data such as CDs (Compact Discs) and SSD (Solid-State Drives). I was fascinated by SSDs since they use V-NAND which stands for vertical NAND. Since there is a vertical part in SSDs, I decided to watch a video called “How do SSDs work?” from “Branch Education” that explains how SSDs work. The video starts by explaining the V-NAND technology and how every SSD includes one. It then starts explaining charge flash trap technology which allows SSDs to trap electrons and store charge as bits. In the video 3-bit version of charge flash trap memory is used as an example where 8 different charge levels occur because of 23. To read the stored data control gate sends the charge stored into the bitline. Reflecting on my learning I now understand how SSDs work at the microscopic level which will help me understand why SSDs perform some different processes such as garbage collection. Learning how electrons act at the microscopic level helped me increase my critical thinking skills since manipulating electrons in the correct way is the most important step to solving the hardware problems of a computer. Technology gets improving every day and keeping up is crucial for scientists and engineers. Overall, I believe learning SSDs was crucial for my hardware knowledge and it taught me a lot.

### Reflection on Seminar Discussions

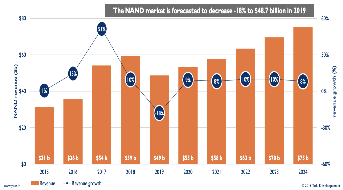
### For this week’s seminar discussion, we were given a scenario to be an HR manager in a company that has an inefficient organizational structure. I decided to research popular organizational structures used by businesses and take some notes to choose the best one. At first, I had some challenges understanding organizational structure since it is a different area of research, but I overcame this by reading more. Eventually, I decided on using a hierarchical structure since it looks very similar to how computers store data in folders. The hierarchical structure is an organizational structure where employees are assigned into groups and assigned a supervisor for each group. This looks like the folder structure in computers where each folder is inside another folder therefore the data does not mix with different types. Each folder has a name and specific data inside which corresponds with the name. An example could be the pictures folder which includes pictures taken. This folder is also managed by the home directory and the home directory is managed by the system. And this goes on until the most crucial root directory corresponds to the manager in businesses. I believe learning manager tasks will always be helpful for new opportunities since a manager is the heart of a business and plays a really important role. Reflecting on myself, learning organizational structure helped me understand how managers manage the company. Overall learning about organizational structure in this discussion will be useful for me since my future job can also require some business knowledge.

## Week 10: Introduction to operating systems, cpu scheduling

### Reflection on Learning Materials

This week we were introduced to operating systems and CPU scheduling. We were introduced to operating systems, computer systems and CPU scheduling. This week one of the topics that caught my interest was the operating system itself. The operating system (OS) is software that lets users control the hardware that they own. The slides explain OS as the intermediate between users and hardware. Different operating systems have different scheduling algorithms, support for programs and applications and hardware support. I have read the first chapter of “Operating Systems in Depth” book to find more details. And I have come across two new mechanisms traps and interrupts. A trap is a software-generated unintended request that occurs because of an error such as division by zero. An interrupt, unlike a trap, on the other hand, is an independently handled request from an external device. Reflecting on my learning and thinking I think learning how operating systems, scheduling algorithms and mechanisms improve my ability to choose the correct OS for the correct workload. Such as which OS should be used for hosting a web page or which OS uses the least amount of memory while being the most efficient. Learning these skills will improve my general understanding and help me in my future jobs and projects.

### Reflection on Seminar Discussions

This week our task in the seminar was planning to manufacture NAND flash memory in the UK. To start my analysis of NAND chips I decided to research the process of how they are manufactured. In summary, they are manufactured in a clean factory (fab) where chips are etched into wafers which are then sent for testing and afterwards the wafers are sliced into individual die and then packaged after whole testing is done. Next, I looked at the raw materials required for NAND chips and found out that other than copper and silicon all the other raw elements and materials can be produced in the UK. For copper, I decided to buy them from Australia since both countries have a good relationship. As for silicon Norway is the 4th biggest producer and it can be bought from them since their population is very little compared to the amount of silicon produced. And finally, I had to analyze the market share to see if we can compete with existing players in the market. The market had a big fall after the Covid-19 outbreak since everyone was forced to work from home more hardware was produced but currently the market is safe to start a new manufacturing company. Reflecting on myself, I have learned a lot about manufacturing hardware and how the market works from this task. I think these skills that I learned will be useful in the future if I ever require to produce my own hardware or work in a hardware company. I also found out which countries and companies lead the industry for raw materials and manufacturing processes. Also learning about how hardware is manufactured and tested helped me understand more about the devices that I use currently and how they were made before I started using them.

## Week 11: Process and Threads, Process Synchronization, memory management

### Reflection on Learning Materials

A diagram of a process

Description automatically generated with low confidenceThis week we were introduced to process and threads, process synchronization and memory management. This week’s slides included a topic that I heard about before but not researched deeply which is deadlock. Deadlock can be defined as “When two or more threads are waiting for each other to release the resources they need and get stuck forever” Therefore it is obvious that this must be avoided. I have found a website called scaler.com which explains this concept in detail by giving tips and tricks to recover and avoid deadlock completely. According to the website, the deadlock occurs because of four reasons: Mutual Exclusion, Hold and Wait, No Preemption and Circular Wait which was also explained in the slides. According to the website to avoid deadlock completely we can use deadlock avoidance algorithms that test before executing a process. And for recovery aborting all processes one by one or at once can help solve the deadlock. Reflecting on my own learning I believe learning deadlock is important since it is one of the most frequently asked interview questions. And it also helps me to improve my debugging skills. It is also one of the most common problems in parallel computing and multiprocessing systems. Overall, I believe learning deadlock and how to solve it or avoid it completely will be useful in my future projects since I expect to debug my code in a professional environment.

### Reflection on Seminar Discussions

This week our seminar was about solving the tragedy of commons using concepts from CPU scheduling. The tragedy of commons is when the lack of property rights leads to the overuse of resources. One example could be berries in parks since they are public sometimes, they are exhausted by individuals which harm individuals again. Although this is an issue in CPU scheduling this is not a problem at all. In CPU scheduling algorithms there is mostly an arrangement technique applied to make an efficient queue. This could be the shortest job first since it can finish faster. Therefore, a solution would be to put some law such that people can enter a queue either online or from a physical place and the person with the shortest job can use it first. Or bring a temporary ban if the resource is about to finish so it can reproduce again. Now in computers such a problem does not exist because CPU, cache and memory are usually so fast which ends up resources never finishing. However, in resource-intensive tasks, this could be applied. Reflecting on myself, I think I have learned new concepts in economics and how we can apply computer architecture concepts to solve problems in other areas. This is a useful skill since I could apply this when I approach a new concept that is outside my area. Having an engineer’s or scientist’s mind is useful in every area since an engineer’s aim is to find the cheapest solution to a problem and a scientist’s aim is to try every solution to find the best one. Overall, I have learned a lot from this week’s seminar, and this will be useful in my future jobs, tasks and projects.

References:

<https://www.youtube.com/watch?v=Ml3-kVYLNr8>

<https://logic.ly/demo/>

<https://www.youtube.com/watch?v=1FXhjErUz58>

<https://www.youtube.com/watch?v=DmwOSdwzZ3E>

<http://www.vpscience.org/materials/US06CCSC04_4.pdf>

[Guide to Assembly Language A Concise Introduction](https://librarysearch.cardiff.ac.uk/discovery/fulldisplay?docid=alma9911786308502420&context=L&vid=44WHELF_CAR:44WHELF_CAR_VU1&lang=en&search_scope=CU_Search_ALL&adaptor=Local%20Search%20Engine&tab=CSCOP_EVERYTHING&query=any%2Ccontains%2CGuide%20%20to%20Assembly%20%20Language) – James T. Streib

<https://www.youtube.com/watch?v=wteUW2sL7bc>

<https://www.youtube.com/watch?v=5Mh3o886qpg>

<https://learning.oreilly.com/library/view/operating-systems-in/9780471687238/>

<https://www.scaler.com/topics/operating-system/deadlock-in-os/>