|  |  |
| --- | --- |
| HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY AND EDUCATION  **Faculty of Information Technology** | **Major: Information Technology**  **Level: Undegraduate**  **Program: Information Technology** |

**COMPUTER ARCHITECTURE AND ASSEMBLY LANGUAGE COURSE SYLLABUS**

1. **Vietnamese name:** Kiến trúc máy tính và Hợp ngữ
2. **English name:** Computer Architecture and Assembly Language **Course** **number:** CAAL240180
3. **Credit number:** 3 credits (2/1/4) (2 credits on class, 1 credit on laboratory, 4 credits at home)
4. **Instructors:**

1. Nguyễn Đăng Quang

2. Đinh Công Đoan

1. **Prerequisite:**

Students have completed Introduction to Programming course

1. **Course Description**

This course provides students with basic knowledge of Computer Architecture, those attributes that have direct impact on the logical execution of a program, the operational units and their interconnection that realize the architectural specifications, memory organization, addressing modes. understanding the stack frame when an assembly code is called from the high-level language program.

This course is also to provide a comprehensive introduction to programming in assembly language on x86-64 processor. The students will learn how a stack frame is organized when an assembly code is called from the high-level language program. They will learn how to debug an executable file with gdb.

1. **Course Goals**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Goals*** | ***Goal description***  *(The course aims to provide students with:)* | **Goals** | ***Competency level*** |
| **G1** | An ability to explain technical terms, concepts, structural and functional view of computer, understand the complex trade-offs between CPU clock speed, cache size, bus organization, number of core processors | 3 | 3 |
| **G2** | Competency to assemble, debug programs at machine language level. | 3 | 3 |
| **G3** | Teamwork skill, presentation skill | 3 | 3 |
| **G4** | Ability to apply knowledge of computer architecture to other areas of computing curriculum and real environment. | 1.3  1.4  1.5 | 2  3  3 |

1. **Learning outcomes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Course objectives** | | | **Description**  *(After completing program, students are able)* | **ELO** | **Competency levels** |
| **G1** | G1.1 | To describe computer structural and functional view of computer components, Understand the complex trade-offs between CPU clock speed, cache size, bus organization, number of core processors | | 1.3 | 2 |
| G1.2 | To describe the interaction among components, peripherals, interrupt and I/Os | | 1.3 | 2 |
| G1.3 | To describe the x86-64 instruction set architecture | | 1.4 | 2 |
| G1.4 | To describe the structure of an assembly program, | | 1.5 | 3 |
| **G2** | G2.1 | To write assembly code, to understand stack frame and calling convention an assembly code from the high-level language program | | 2.2 | 3 |
| G2.2 | To improve programing skill and debug program at machine language level | | 2.3 | 3 |
| **G3** | G3.1 | To collaborate and sharing ideas among group members | | 1.2 | 2 |
| G3.2 | To improve English reading skill | | 1.2 | 3 |
| **G4** | G4.1 | To enhance knowledge on other subjects in curriculum | | 1.3 | 3 |
| G4.2 | To explain various aspects of software programming regarding the underlying hardware technologies. | | 1.4 | 3 |

1. **Ethics:**

Students must demonstrate that they accomplished themselves all exercises and projects. Those students who cheat or commit plagiarism will be imposed punishment.

1. **Plan:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Week** | **Content** | **ELO** | Teaching Methods | Assess.  Methods |
| 1,2 | **Chapter 1: Introduction** (4/0/8) |  |  |  |
| **Lecture’s content**   * Computer Organization vs. Architecture * Operational structures of computer components * Multi-level machine * Evolution of Computer * Von Neuman (IAS) machine * Microprocessor speed techniques * Performance balance * Multi-core processor * Performance assessment (CPI, MIPS, Execution time of program) * Amdahl’s law | G1.1,  G1.2  G3.1  G3.2 | Presentation | Questions & Case studies |
| **Tasks for students at home**:   * Group work on CISC and RISC Computer * Exercise on computer assessment and speed enhancement when moving from single-core to multi-core processor | G1.1,  G1.2  G3.1  G3.2 |  |  |
| 3 | **Chapter 2*:* Top-level view of Computer functions and interconnections** *(4/0/8)* |  |  |  |
| **Lecture’s content**   * Computer functions * Instruction cycle * Program execution * Interrupt * Instruction cycle with interrupt * Interrupt vector table * Interconnections * Memory module * I/O module * Processor module * Buses * Bus arbiter * Bus hierarchy * Intel QPI * PCIe * **Practice Lab**   + Watch the fetch-decode-execute video [Fetch-Decode-Execute Cycle](https://www.youtube.com/watch?v=XM4lGflQFvA)  + Undertanding the MBR, MAR, PC, IR and Program execution with MARIE Simulator. | G2.1.  G2.2  G4.1  G4.2 | Presentation | Questions & Case studies |
| **Tasks for students at home**  + Running simulation of programs with MARIE Simulator  + Exercises on computing data transfer rate for processors with various register bit-size, databus width and different clock speed: 3.3, 3,5, 3.6, 3.7, 3.11 – section 3 of [1] | G2.1, G2.2  G4.1  G4.2 |  |  |
| 4 | **Chapter 3: Computer Memory (8**/0/16) |  |  |  |
| **Lecture’s content**  **Computer memory overview**   * Key characteristics * Memory hierarchy * Locality of reference   **Cache memory**   * Principle * Mapping schemes   + Direct mapped cache   + Associative cache   + Set associative cache * Replacement policies * Cache write policies   **Virtual Memory**   * Paging * Advantages & Disadvantages of Paging and Virtual Memory * Segmentation * Paging combine with Segmentation   **Practice lab**:  **+** Watchvideo from Dr. Harry Potter  + Investigate cache strategies with Memory Tutorial Simulator   * **Internal memory** * Semiconductor memory * SRAM, DRAM * ROM, PROM, EPROM, EEPROM, flash memory * Chip logic * Memory packaging * **External memory** * Magnetic disk * Disk performance parameter: seek time, rotational latency, access time, transfer time * RAID * SSDs, SSD practical issues * Optical memory | G2.2  G4.2 | Presentation |  |
| **Tasks for students at home**   * Watch video, review cache mapping * Group work on Internal memory and External memory * Excercises on building up memory module from chips | G2.2, G3.1  G3.2  G4.2 |  |  |
|  | **Chapter 4: The Central Processing Unit** (4/0/8) |  |  |  |
| 5 | **Lecture’s content**   * CPU Complexity * x86-64 Registers * x86 Memory segmentation * x64 flat memory model * Addressing modes | G2.4  G4.3, G4.4 | Problem-Based |  |
| **Tasks for students at home**   * Registers & memory view with gdb (tutorial) | G2.4  G4.2, G4.3, G4.4 | Problem-Based | Lab Submis-sion |
| 6 | **Practice lab** (0/4/0) |  |  |  |
| * Practice registers & memory view with gdb |  | Problem-Based | Lab Submis-sion |
|  | **Chapter 5: Data representation** (2/1/4) |  |  |  |
| 7 | **Lecture’s content**   * Integer representation * Unsigned & signed addition * Floating-point representation * Characters & Strings * Excercises |  | Presentation |  |
| **Tasks for students at home**:  Doing Quiz in chapter 3.0 of [3] |  |  |  |
|  | **Chapter 5: Assembly language** |  |  |  |
| 8 | **Lecture’s content**  **NASM assembler**   * Comments * Numeric values * Constants   **Program Memory sections declarations**  + Memory layout  + Data section  + BSS section  + Text section  + Example program  + Excercises  **Assemble/Link/Load**  + Assemble  + Linker  + Loader  + Load in gdb  **Console input/output**   * Calling system services * Input * Output   **Practice lab**  Practice compiling simple nasm programs then watch in gdb |  |  |  |
|  | **Tasks for students at home**  Doing suggested projects from chapter 6 of [3] |  |  | Lab submisison |
|  | **Chapter 6: Instruction Set Overview** |  |  |  |
| 9, 10, 11 | **Lecture’s Content**   * Notational convention * Data movement * Addresses & Values * Conversion instructions * Integer arithmetic * Logical instructions * Control instructions * Iterations   **Excercises:** Quiz questions from chapter 7 of [3] |  |  |  |
|  | **Tasks for students at home**:  Doing all suggested projects in chapter 7.0 of [3] |  |  | Lab Submis-sion |
|  | **Chapter 7: Addressing Modes & Stack processing** |  |  |  |
| 12 | **Lecture’s Content**  **Addressing modes**   * Addresses & values * Example: List summation * Example: Pyramid areas & Volumes * Excercises   **Stack processing**   * Stack * Stack instructions * Stack implementation * Examples * Excercises   **Command line Arguments**   * Parsing Command line arguments * High-level language Example * Argument Count and Argument Vector |  |  |  |
| **Tasks for students at home**:  Doing all suggested projects in chapter 8,9,16 of [3] |  |  | Lab Submis-sion |
|  | **Chapter 8: Macros & Functions** |  |  |  |
| 13, 14 | **Lecture’s Content**  **Macros**   * Single-line macro * Multi-line macro * Example * Debugging macros * Excercises   **Functions**   * Overview, declaration * Debugger next, step commands * Calling convention (parameters, registers) from high-level programming language * Stack frame * Examples * Excercises |  |  |  |
| **Tasks for students at home**:  Doing quiz questions in chapter 11,12 of [3]  Doing all suggested projects in chapter 11,12 of [3] |  |  | Lab Submis-sion |
|  | **Chapter 9: System Services** |  |  |  |
| 15 | **Lecture’s Content**  **Overview**   * Calling system services * Newline Character   **Console output**   * Example   **Console input**   * Examples   **File Operations**   * Excercises |  |  |  |
| **Course Review** |  |  |  |

1. **References**

**-** Lecture notes:

Nguyễn Đăng Quang, Computer Architecture and Assembly language, 2017

**-** References:

1. William Stallings, Computer Organization and Architecture, 9th Edition, Pearson Education, Prentice-Hall, 2013, ISBN-13: 978-0-13-293633-0

2. Linda Null, Julia Lobur, The Essentials of Computer Organization and Architecture, Jones and Bartlett Publishers, 2003

3. x86-64 Assembly Language Programming with Ubuntu

4. nasm online:

<https://www.jdoodle.com/compile-assembler-nasm-online>

<https://rextester.com/l/nasm_online_compiler>

5. Ubuntu vmware machine for labs [16.04](https://drive.google.com/file/d/1HxdUhq-J_-_QKyjngpH9m6Kmuvy0_68a/view?usp=sharing), [12.04](http://www.cis.syr.edu/~wedu/SEEDUbuntu12.04.zip)

1. **Assessesments:**

- Grade scale: **10**

- Plan for assessments:

1. **Approval time:**
2. **Approved by:**

|  |  |  |
| --- | --- | --- |
| **Dean of faculty** | **Head of Department** | **Instructor** |