5DV118 Computer Organization and Architecture Fall 2012

1 Course Staff

Instructor in charge:	Name:	Stephen J. Hegner
(kursansvarig)	Office:	C444 MIT-huset
	Telephone:	090 786 5760
	E-mail:	`hegner@cs.umu.se`
	URL:	`http://www.cs.umu.se/~hegner`
	Office hours:	1015-1045 on days (except Wednesdays) with a lecture given by Stephen J. Hegner
Instructor:	Name:	Lars Karlsson
	Office:	Contact hours held in D420 MIT-huset
	Telephone:	090 786 7024
	E-mail:	`larsk@cs.umu.se`
	URL:	`http://www.cs.umu.se/~larsk`
	Office hours:	1045 - 1150 on days with a lecture

- Stephen J. Hegner is available to answer questions about the lectures and the course material in general, but cannot answer detailed questions about what is or is not acceptable as an answer to an obligatory exercise.
- Questions regarding the exercises should be directed to Lars Karlsson.

2 Course Language

The official language of the course is English. The lectures will be given in English, and since one of the goals of this course is to develop skills in presenting technical ideas in English, all written work must be submitted in English as well. Furthermore, the questions on the final examination will be written in English; and answers must be written in English. For the final examination, it will be permitted to use an XX-English / English-XX dictionary, where XX is a natural language of the student's choice.

Questions may be asked during the lectures in either English or Swedish. However, as there will likely be students in the class who do not speak Swedish, questions posed in Swedish will have to be translated by the instructor into English. It is therefore preferable to ask questions in English whenever possible.

3 Course Literature

The official textbook for this offering of the course is the following.

1. David Patterson and John Hennessy, *Computer Organization and Design, The Hardware/Software Interface*, Revised Fourth Edition, 2012, Morgan Kaufmann, ISBN-13: 978-0-12-374750-1.

The third edition differs substantially from the fourth and is not recommended as a substitute. The (original) fourth edition differs in some details, including many exercises, from the revised fourth edition. It may be usable for the course, but the revised fourth edition is the official standard. Exercises from the revised edition may be assigned.

In addition to the course text, there will be relatively detailed overhead slides, based largely upon those used in courses given by Professor Mary Jane Irwin at the Pennsylvania State University, who has graciously granted permission to use those materials. They will be available for download on the course web page.

4 Course Content and Outline

The official kursplan is available by searching for `5DV118` on this link. A more offering-specific outline is shown below. The numbers shown in the single rectangular brackets (i.e., [..]) identify chapters in the textbook. The numbers in angle brackets ⟨..⟩ indicate the approximate number of 45-minute lecture periods which will be devoted to the topic.

- Reasonably detailed overhead slides will be available for many topics. The authoritative source for relevant (i.e., possible examination) material is the course lectures and these slides. In some cases, material not covered in the textbook may nonetheless be covered in lecture presentations.
- The slides alone are *not* a self-contained written record of all that will be covered during the class meetings. Students are responsible for all material which is covered in the course, regardless of whether or not it is found in the slides.
- The number of 45-minute lecture "hours" to be devoted to each topic is approximate, and in particular is rounded to the nearest integer. Adjustments will be made as the course progresses, and so the table below should not be used a definitive guide to which topics will be covered on which days.

1 Introduction [1] ⟨2⟩

2 The MIPS Architecture [2] ⟨7⟩

3 Arithmetic [3] ⟨4⟩

4 Processor Architecture [4.1-4.13] ⟨7⟩

5 Memory [5] ⟨4⟩

6 Secondary Storage [6] ⟨2⟩

7 Multi-Stuff and Parallelism [7] ⟨2⟩

8 Review ⟨2⟩

4.1 Online Materials

There web site for the course is located at

`http://www.cs.umu.se/kurser/5DV118/H12/index.html`

The following materials may be found on these pages.

1. This syllabus, in both PDF and HTML.
2. The lecture slides for the course.
3. Descriptions of the obligatory exercises.
4. Information on auxiliary software, such as the Spim assembler and emulator.
5. Miscellaneous links to architecture-related things.
6. Some official documents required by the Department of Computing Science.

5 Course Materials Outline

The following is a list of those chapters and sections which will be covered in the course. For each chapter or section, a symbol is given which indicates the nature of coverage in the course. The meaning of these symbols is provided in the table below.

✔	Material will be covered in the course.
➲	Overview material, covered in concept but not detail
✘	Material will not be covered in the course.
✙	Review material, prior knowledge is expected.
✥	Material will be covered partially or selectively.
??	Coverage extent to be announced later.

Note the following:

- If an entire chapter is covered, no section-by-section breakdown is given.
- Entries have not been provided for sections entitled "Summary" or the like.

- In general, omitted items will not be covered. However, the possibility that some covered material may appear in an omitted chapter or section remains. In all cases, the lectures and course slides should be taken to be the definitive guides to the course material.
- Sections entitled “Concluding Remarks” and “Historical Perspective and Further Reading” are not listed. They are all classified as overview material.
- Sections entitled “Exercises” are not listed.

1 Computer Abstractions and Technology

- 1.1 Introduction ➲
- 1.2 Below Your Program ➲
- 1.3 Under the Covers ➲
- 1.4 Performance ✔
- 1.5 The Power Wall ➲
- 1.6 The Sea Change ✔
- 1.7 Real Stuff ➲
- 1.8 Fallacies and Pitfalls ✔
- 1.9 Concluding Remarks ➲

2 Instructions: Language of the Computer

- 2.1 Introduction ➲
- 2.2 Operations of the Computer Hardware ✔
- 2.3 Operands of the Computer Hardware ✔
- 2.4 Signed and Unsigned Numbers ✔
- 2.5 Representing Instructions in the Computer ✔
- 2.6 Logical Operations ✔
- 2.7 Instructions for Making Decisions ✔
- 2.8 Supporting Procedures in Computer Hardware ✔
- 2.9 Communicating with People ✔
- 2.10 MIPS Addressing for 32-Bit Immediates and Addresses ✔
- 2.11 Parallelism and Instructions: Synchronization ✔
- 2.12 Translating and Starting a Program ✔
- 2.13 A C Sort Example to Put It All Together ✔

- 2.14 Arrays vs. Pointers ✔
- 2.15 Advanced Material: Compiling C and Interpreting Java ✘
- 2.16 Real Stuff: ARM Instructions ➲
- 2.17 Real Stuff: ARM Instructions ✘
- 2.18 Fallacies and Pitfalls ➲

3 Arithmetic for Computers

- 3.1 Introduction ➲
- 3.2 Addition and Subtraction ✔
- 3.3 Multiplication ✔
- 3.4 Division ✔
- 3.5 Floating Point ✣
- 3.6 Parallelism and Computer Arithmetic: Associativity ✔
- 3.7 Real Stuff: Floating Point in the x86 ✘
- 3.8 Fallacies and Pitfalls ➲
- 3.9 Concluding Remarks ➲

4 The Processor

- 4.1 Introduction ➲
- 4.2 Logic Design Conventions ✔
- 4.3 Building a Datapath ✔
- 4.4 A Simple Implementation Scheme ✔
- 4.5 An Overview of Pipelining ✔
- 4.6 Pipelined Datapath and Control ✔
- 4.7 Data Hazards: Forwarding versus Stalling ✔
- 4.8 Control Hazards ✔
- 4.9 Exceptions ➲
- 4.10 Parallelism and Advanced Instruction-Level Parallelism ✣
- 4.11 Real Stuff ✘
- 4.12 Advanced Topic ✘
- 4.13 Fallacies ✔

5DV118 Syllabus, page 6

5 Large and Fast: Exploiting Memory Hierarchy

- 5.1 Introduction ➲
- 5.2 The Basics of Caches ✔
- 5.3 Measuring and Improving Cache Performance ✔
- 5.4 Virtual Memory ✔
- 5.5 A Common Framework for Memory Hierarchies ✔
- 5.6 Virtual Machines ➲
- 5.7 Using a Finite-State Machine to Control a Simple Cache ✥
- 5.8 Parallelism and Memory Hierarchies: Cache Coherence ✥
- 5.9 Advanced Material: Implementing Cache Controllers ✘
- 5.10 Real Stuff ➲
- 5.11 Fallacies and Pitfalls ✔

6 Storage and Other I/O Topics

- 6.1 Introduction ➲
- 6.2 Dependability, Reliability, and Availability ✔
- 6.3 Disk Storage ✔
- 6.4 Flash Storage ✔
- 6.5 Connecting Processors, Memory, and I/0 Devices ✔
- 6.6 Interfacing I/O Devices to the Processor, Memory and Operating System ✔
- 6.7 I/O Performance Measures: Examples from Disk and File Systems ➲
- 6.8 Designing an I/O System ➲
- 6.9 Parallelism and I/O: Redundant Arrays of Independent Disks ✔
- 6.10 Real Stuff ✘
- 6.11 Advanced Topics: Networks ✘
- 6.12 Fallacies and Pitfalls ✔

7 Multicores, Multiprocessors, and Clusters

- 7.1 Introduction ➲
- 7.2 The Difficulty of Creating Parallel Processing Programs ✔
- 7.3 Shared Memory Multiprocessors ✥

- 7.4 Clusters and Other Message-Passing Multiprocessors ✣
- 7.5 Hardware Multithreading ✣
- 7.6 SISD, MIMD, SIMD, SPMD, and Vector ✔
- 7.7 Introduction to Graphics Processing Units ✣
- 7.8 Introduction to Multiprocessor Network Topologies ✣
- 7.9 Multiprocessor Benchmarks ✣
- 7.10 Roofline: A Simple Performance Model ??
- 7.11 Real Stuff ✘
- 7.12 Fallacies and Pitfalls ✔

A Graphics and Computing CPUs ✘

B Assemblers, Linkers, and the SPIM Simulator ✣

C The Basics of Logic Design (on CD) ✙

D Mapping Control to Hardware (on CD) ✘

E A Survey or RISC Architectures for Desktop, Server, and Embedded Computers (on CD) ✘

6 Laboratory Schedule and Computer Resources

There is no official laboratory booking for the course, nor any in-laboratory instruction. In general, when not reserved by a course, the computer laboratories of the department are open for use by students for their coursework.

The assembler and simulator *Spim* (command-line version) and *XSpim* (X-windows version) is available on the departmental computers for the purposes of learning the MIPS instruction set and testing the correctness of the programming exercises. However, the programming exercises themselves will be carried out using standard programming languages such as `C` or `Java`, and the final, submitted versions must run under the Unix/Linux environment of the department.

The newer version *QTSpim* is installed on the departmental MS Windows computers. Unfortunately, QTSpim is incompatible with the current version (Squeeze) of stable Debian Linux (but it works fine with Ubuntu Linux 10.10 and newer, if you wish to install it on your personal machine.)

7 Course Schedule

The table below identifies the course meeting times and places, together with the nature of the meeting. The key ”L” denotes a lecture, “R” denotes a review session, while ”E” denotes an examination booking.

For each lecture, the topics to be covered are identified via the outline header number of Section 4 of this syllabus. So, for example, on November 30 topics of 4, Processor architecture, will be covered. This is only an approximate assignment of meeting times to topics, and it may be altered as the course progresses.

Rooms whose identifiers begin with the letter *M* are located in MIT-huset.

Week	Day	Date	Time	Type	Room	Topics	Lecturer
45	We	Nov 07	0815-0955	L	MA166	1	Hegner
45	Fr	Nov 09	0815-0955	L	MA146	2	Hegner
46	Tu	Nov 13	0815-0955	L	MA166	2	Hegner
46	Fr	Nov 16	0815-0955	L	MA166	2	Hegner
47	Tu	Nov 20	0815-0955	L	MA166	2,3	Hegner
47	We	Nov 21	0815-0955	L	MA166	3	Hegner
47	Fr	Nov 23	0815-0955	L	MA166	3,4	Hegner
48	Tu	Nov 27	0815-0955	L	MA166	4	Hegner
48	Fr	Nov 30	0815-0955	L	MA166	4	Hegner
49	Tu	Dec 04	0815-0955	L	MA166	4	Hegner
49	We	Dec 05	0815-0955	L	MA166	5	Hegner
49	Fr	Dec 07	0815-0955	L	MA146	5	Hegner
50	Tu	Dec 11	0815-0955	L	MA166	6	Hegner
50	Fr	Dec 14	0815-0955	L	MA156	7	Karlsson
02	Tu	Jan 08	0815-0955	R	MA166	8	Karlsson
03	We	Jan 16	0900-1500	E	Skrivsal 6	Final examination	
14	We	Apr 03	0900-1500	E	Skrivsal 7	Final examination	
24	Tu	Jun 11	0900-1500	E	Skrivsal 2	Final examination	

The lectures will run 0815-0900 and 0910-0955. The reason for the five-minute shift and shorter break than is traditional is to avoid collisions with courses from other faculties, which often begin on the hour (hh00).

8 Prerequisites

The formal requirements are listed in the course plan, which may be found by searching for 5DV118 at the following link. They include the following.

1. A knowledge of programming in C in the Unix/Linux environment. This requirement is met by the formal prerequisite of the course *Systemnäraprogrammering* (Systems Programming).

2. A knowledge of digital design (combinational and sequential computer logic). This requirement is met by the formal prerequisite of the course *Digitalteknik* (Digital design).

3. A thorough knowledge of data structures and algorithms, as presented in the course *Datastrukturer och algoritmer* (Data Structures and Algorithms), which is a prerequisite for the course in systems programming, and so an implied prerequisite.

4. A knowledge of discrete mathematics and the formal foundations of computer science. This requirement is met by the course *Diskret matematik* (Discrete Mathematics).

5. A broad background in computer science consisting of at least 60 credits (one full year), or at least two years of study in a related discipline.

These requirements should be met by students who are following a normal path of study in one of the programs of the Department of Computing Science. However, students from other disciplines, as well as students from other universities, who are considering this course should understand that the level of sophistication required in these topics is relatively high, and often not met by persons who work in other disciplines or have followed a less technical course of study in computer science, even if they have a fair amount of practical programming experience.

9 Grading System

This course has two parts (*moment* in Swedish), a conceptual part (*teoridelen* in Swedish) and an exercise part (*laborationsmoment* in Swedish).

The only possible grades for the exercise part are S (Satisfactory; G=*Godkänd* in Swedish) and U (Unsatisfactory, *Underkänd* in Swedish). The grade on this part will be determined entirely by two obligatory programming projects. Each project will be graded as S (Satisfactory) or U (Unsatisfactory). To obtain the grade of S for the exercise part of the course, it is both necessary and sufficient to obtain the grade of S on both projects. In addition, for each obligatory project it will be possible to earn a maximum of 75 quality points. These points will be assigned based upon the overall quality and correctness of the work. Fifty of these points will be based upon the quality of the basic parts of the project, with an additional 25 bonus points possible for implementing extra features. Thus, a maximum of 150 points may be earned on the two obligatory projects.

There will also be two non-obligatory problem exercises, which will provide practice in solving typical problems on such topics as pipeline and cache analysis. Each will have a maximum point total of 50, so a maximum of 100 points may be earned on the two non-obligatory exercises.

The examination will have a total of 1000 points.

The final point total F for the course is computed as

$$F = \max(E,\quad 0.8 \times E + (L + P))$$

with E the number points earned on the examination, L the number of points earned on the laboratory exercises, and P the number of points on the non-obligatory exercises. The final grade on the conceptual part of the course is computed as follows.

Number of points	Grade
$F \geq 800$	5 (med beröm godkänd – excellent)
$650 \leq F < 800$	4 (icke utan beröm godkänd – very good)
$500 \leq F < 650$	3 (godkänd – satisfactory)
$F < 500$	U (underkänd – unsatisfactory)

In addition, to pass the course, a minimum of 500 points on the examination is necessary, regardless of how many points are earned on the exercises. Thus, exercise points can only be used to increase the grade from 3 to 4, or from 4 to 5. They cannot be used to rescue a performance of less than 50% on the examination.

Students who completed the exercise part of the course in a previous year may nevertheless submit exercises for points.

10 Obligatory Work

10.1 General Remarks on the Obligatory Projects

- The exercises may be completed in groups, and collaboration is permitted on the software exercises, roughly as described in the documents *Riktlinjer vid labgenomförande (Policy for Obligatory Exercises)* and *Hederskodex (Honor Code)*. More details will be provided later, when the descriptions of these exercises are distributed.

- The exercises may be submitted individually, or two or three persons may submit one solution. However, once a solution is submitted, only those named on the submission will receive credit for it. Partners in solution may not be added after the initial submission.

- Each exercise will have a submission deadline. For late submissions, the number of points awarded will be $(1 - 0.1 \times p)$, where p is the number of working days which the submission is late. Thus, a submission which is more than two weeks late cannot receive any points at all.

- All written submissions must be in English.

- All obligatory exercises must be submitted for grading on or before the third and final examination (during June, 2013). No exercises will be accepted after that date. After that date, the student must re-register for the next offering of the course and complete the exercise part according to the rules for that part.

10.2 Obligatory Work Completed in Previous Years

- The laboratory part of 5DV118 is similar to, but not identical to, that of the old course 5DV008. See Sec. 12.1 below for rules of transfer between 5DV008 and 5DV118.

- Credit for individual exercises may not be carried over from previous years. A student who does not already have a satisfactory grade recorded for the exercise part of 5DV008 must complete all requirements for that part as defined by this offering of the course.
- Quality point from previous years may not be carried over to the current year. Students who have already completed the laboratory exercises for previous years may however re-submit solutions for the current year in order to obtain quality points.

11 Non-Obligatory Exercises

More information on these exercises will be provided when they are distributed. However, the deadlines will be reasonably strict and late submissions risk not be accepted.

12 Relationship to 5DV008

The discontinued course 5DV008, Computer Architecture, was similar in content to 5DV118. The main differences are the following:

- The new course, 5DV118, assumes a knowledge of digital design, and such knowledge will be used in support of the presentation of several topics.
- As 5DV118 is classified as a course at the advanced level, the examinations will be somewhat more comprehensive.
- One of the goals of 5DV118 is to promote the development of technical writing skills in English. To this end, all written work, including answers on the final examination, must be in English. Furthermore, substantially greater weight will be placed on the written parts, particularly the user manuals, for the two obligatory exercises.

12.1 Use of Results from 5DV008

The decision of the Department of Computing Science regarding use of results from 5DV008 is roughly as follows.

- If the conceptual (“theory”) part of 5DV118 and the exercise (“laboratory”) part of 5DV008 is completed, the student will receive an overall grade for 5DV118.
- If the conceptual (“theory”) part of 5DV008 and the exercise (“laboratory”) part of 5DV118 is completed, the student will receive an overall grade for 5DV008.

In other words, the conceptual part governs the identity of the course for which a composite grade is given. For more comprehensive information, including the official rules, follow the links below.

Swedish: `http://www.cs.umu.se/student/kursutbud/nedlagda-kurser/`

English: `http://www.cs.umu.se/english/student/courses/discontinued-courses/`