

DFDV3100 – 2018/19

Computer System Architectures and VHDL Programming



Universitetet
i Sørøst-Norge

DFDV3100 – 2018/19
(2nd part: Architectures)

Introduction

<http://bit.ly/dfdvd3100-08a>

José Manuel Martins Ferreira | josemmf@usn.no

Professor

Fakultet for teknologi, naturvitenskap og maritime fag

Introductory information

- **Code:** DFDV3100
- **Name:** Computer System Architectures and VHDL programming (2nd part: Architectures)
- **Content:** Introduction to digital technical components and building blocks, VHDL and software for simulation, analysis and synthesis of digital circuits / systems, and selected key issues in modern computer architecture.

Learning outcomes: Knowledge

- **Knowledge:** The student has knowledge of digital systems structure and function; has knowledge of VHDL description of digital circuits; has knowledge of principles used in modern computers

Learning outcomes: Skills

- **Skills:** The student can simulate, analyze and synthesize digital systems; can implement a digital system described with VHDL in an FPGA circuit

Learning outcomes: Competence

- **General competence:** The student has a basis to follow future developments in computer architecture

Further information

- **Course syllabus:** http://bit.ly/dfd3100_18-19
- Learning activities comprise lectures, tutorials, and project (coursework assignment)
- Learning resources will come from Canvas and Library

Assessment

- Written exam, 4 hours (100%)
- Access to final exam requires successful completion of the compulsive coursework assignment
- Exam questions will be based on Canvas discussions


About me

- **Professional profile:**

- Extended CV: <http://bit.ly/josemmf>
- ORCID: http://bit.ly/josemmf_OR
- Linkedin: http://bit.ly/josemmf_LI
- ResearchGate: http://bit.ly/josemmf_RG



Thanks for
your attention



DFDV3100 – 2018/19

Computer System Architectures and VHDL Programming



Universitetet
i Sørøst-Norge

DFDV3100 – 2018/19
(2nd part: Architectures)

Work plan and T&L model

<http://bit.ly/dfdvdv3100-o8b>

José Manuel Martins Ferreira | josemmf@usn.no

Professor

Fakultet for teknologi, naturvitenskap og maritime fag

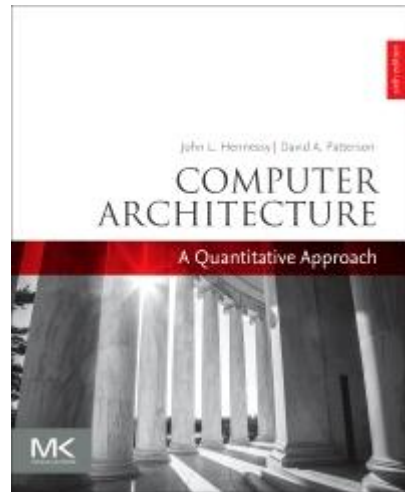
Learning resources

- **Course content on Canvas**

Comprising lecture materials and discussions (main learning resource)

- **Book: [Computer Architecture | A Quantitative Approach](#)**

We will address the subjects covered by the first 5 chapters (complementary learning resource)



Book chapters to be covered

- We will address the five initial chapters of the book
- None of these chapters will be entirely covered (both for reasons of time, and also because they go into much further detail than what is expected in this course)
- Each chapter will be considered to an extent that depends on the subject

Teaching and learning model


- Lectures will be recorded as short video presentations made available in Canvas together with **discussions**
- The weekly workload assumes that students go through all lectures and discussions posted for that week
- Classes = meetings with each group to discuss exercises (4 x 45 min weekly); held on-campus and online

Schedule

W41: 09.10 + 12.10 OC	Fundamentals of quantitative design and analysis
W42: 16.10 + 19.10 OC	Memory hierarchy design
W43: 23.10 + 26.10 OL	Instruction-level parallelism and its exploitation
W44: 30.10 + 02.11 OL	Data-level parallelism in vector, SIMD, and GPU architectures
W45: 06.11 + 09.11 OC	Thread-level parallelism
W46: 13.11 + 16.11 OC	Compulsive coursework assignment
W47: 20.11 + 23.11 OL	Compulsive coursework assignment



Thanks for
your attention



DFDV3100 – 2018/19

Computer System Architectures and VHDL Programming

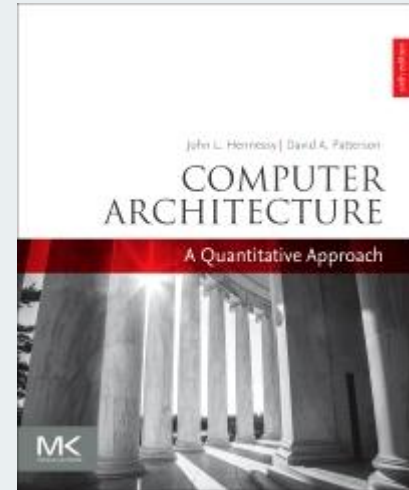
DFDV3100 – 2018/19
(2nd part: Architectures)

Universitetet
i Sørøst-Norge

Fundamentals of quantitative design and analysis - 01

<http://bit.ly/dfdvdv3100-o8c>

José Manuel Martins Ferreira | josemmf@usn.no
Professor
Fakultet for teknologi, naturvitenskap og maritime fag



Learning outcomes

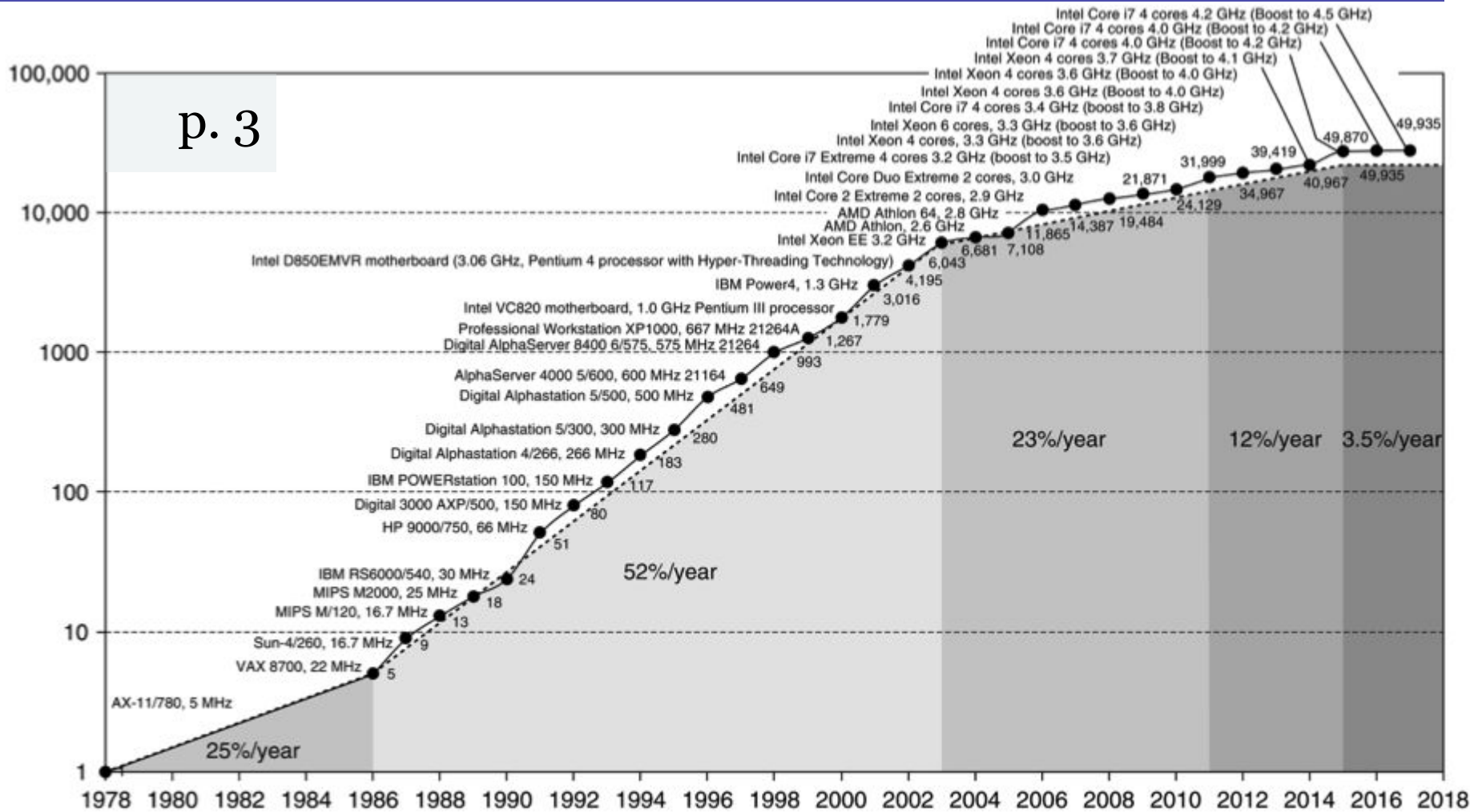
- You'll be able to
 - Briefly describe the evolution of computing performance throughout the last 40 years
 - Distinguish the main classes of computers and classes of parallelism
 - Explain what is an instruction set architecture and describe the RISC-V instruction set

Historical perspective

- Processor performance improved by 50,000 in the last 40 years
- In 2004 Intel declared that the road to higher performance would be via multiple processors per chip rather than via faster uniprocessors
- Due to the end of Moore's law and Dennard scaling, performance is now only doubling every 20 years

p. 3

Performance (vs. VAX-11/780)



Classes of computers

pp. 6 + 7-10

Feature	Personal mobile device (PMD)	Desktop	Server	Clusters/warehouse-scale computer	Internet of things/embedded
Price of system	\$100–\$1000	\$300–\$2500	\$5000–\$10,000,000	\$100,000–\$200,000,000	\$10–\$100,000
Price of microprocessor	\$10–\$100	\$50–\$500	\$200–\$2000	\$50–\$250	\$0.01–\$100
Critical system design issues	Cost, energy, media performance, responsiveness	Price-performance, energy, graphics performance	Throughput, availability, scalability, energy	Price-performance, throughput, energy proportionality	Price, energy, application-specific performance

NB.: The ability to run third-party software is the dividing line between non-embedded and embedded computers

Classes of parallelism

- Parallelism in applications (2)
 - [Data-level parallelism (DLP)] [Task-level parallelism]
- Ways to exploit it in computer hardware (4)
 - [Instruction-level parallelism (ILP)] [Vector architectures, graphic processor units (GPU), and multimedia instruction sets] [thread-level parallelism] and [request-level parallelism]

Parallel computing classification

- Targeting data-level and task-level parallelism:
 - SISD (uniprocessor exploiting ILP)
 - SIMD (same instruction to multiple items of data)
 - MISD (no commercial implementation)
 - MIMD (targets task-level parallelism)

NB.: Many parallel processors are hybrids of the SISD, SIMD, MIMD classes

Instruction set architecture (ISA)

- *Computer architecture* was initially used to mean instruction set design
- The designer's job, however, encompasses much more
- The expression *instruction set architecture* is herein used to mean the programmer-visible instructions and serves as the boundary between software and hardware

The seven dimensions of an ISA


- Class of ISA (most popular: register-memory, load-store)
- Memory addressing (normally byte addressing)
- Addressing modes (for registers, constants, memory)
- Types + sizes of operands (ASCII 8-bit, Unicode 16-bit, etc)
- Operations (data transfer, arithmetic, logical, etc)
- Control flow instructions (conditional branches, jump, etc)
- Encoding an ISA (fixed length, variable length)

Learning task

- Go through
 - The RISC-V registers, names, usage and calling conventions presented in p. 13
 - The subset of RISC-V instructions presented in p. 15
 - The floating point instructions presented in p. 16
 - The RISC-V ISA formats in p. 16



Thanks for
your attention



DFDV3100 – 2018/19

Computer System Architectures and VHDL Programming

DFDV3100 – 2018/19
(2nd part: Architectures)

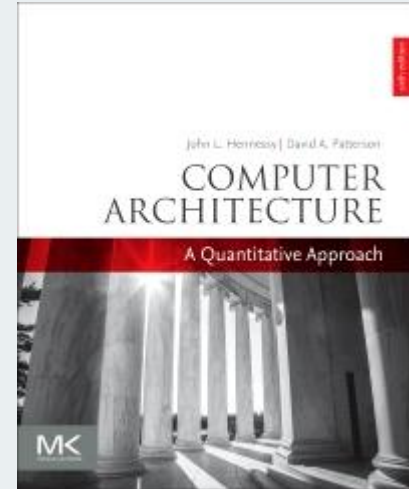


Universitetet
i Sørøst-Norge

Fundamentals of quantitative design and analysis - 02

<http://bit.ly/dfdvd3100-o8d>

José Manuel Martins Ferreira | josemmf@usn.no
Professor
Fakultet for teknologi, naturvitenskap og maritime fag



Learning outcomes

- You'll be able to
 - Describe the semantics of frequent expressions such as *computer implementation* and *performance*
 - Explain why computing performance is very much technology-bound
 - Summarise the main techniques used to improve energy efficiency in computer architectures

Computer implementation

- Comprises two main components
 - Organisation: high-level aspects e.g. memory system and interconnect, CPU
 - Hardware: detailed logic design and packaging technology
- The term architecture covers ISA, organisation, and hardware

Changes in implementation tech

- IC logic technology (Intel Core i7: 1,750,000,000 trans.)
- Semiconductor DRAM (currently 16 Gb)
- Semiconductor Flash or EEPROM (8-10 times cheaper per bit than DRAM)
- Magnetic disk technology (8-10 times per bit cheaper than Flash)
- Network technology (switches and transmission system)

Performance

- Two important concepts
 - Bandwidth: the total amount of work done in a given time
 - Latency: the time between the start and the completion of an event

Technology limits

- The feature size (minimum size of a transistor or wire) decreased from 10 μm in 1971 to 0.016 μm in 2017 (7 nm chips are already underway)
- Performance improvements are bound by technology limitations that caused the end of Moore law and Dennard scaling

Performance milestones

Microprocessor	16-Bit address/ bus, microcoded	32-Bit address/ bus, microcoded	5-Stage pipeline, on-chip I & D caches, FPU	2-Way superscalar, 64-bit bus	Out-of-order 3-way superscalar	Out-of-order superpipelined, on-chip L2 cache	Multicore OOO 4-way on chip L3 cache, Turbo
Product	Intel 80286	Intel 80386	Intel 80486	Intel Pentium	Intel Pentium Pro	Intel Pentium 4	Intel Core i7
Year	1982	1985	1989	1993	1997	2001	2015
Die size (mm ²)	47	43	81	90	308	217	122
Transistors	134,000	275,000	1,200,000	3,100,000	5,500,000	42,000,000	1,750,000,000
Processors/chip	1	1	1	1	1	1	4
Pins	68	132	168	273	387	423	1400
Latency (clocks)	6	5	5	5	10	22	14
Bus width (bits)	16	32	32	64	64	64	196
Clock rate (MHz)	12.5	16	25	66	200	1500	4000
Bandwidth (MIPS)	2	6	25	132	600	4500	64,000
Latency (ns)	320	313	200	76	50	15	4

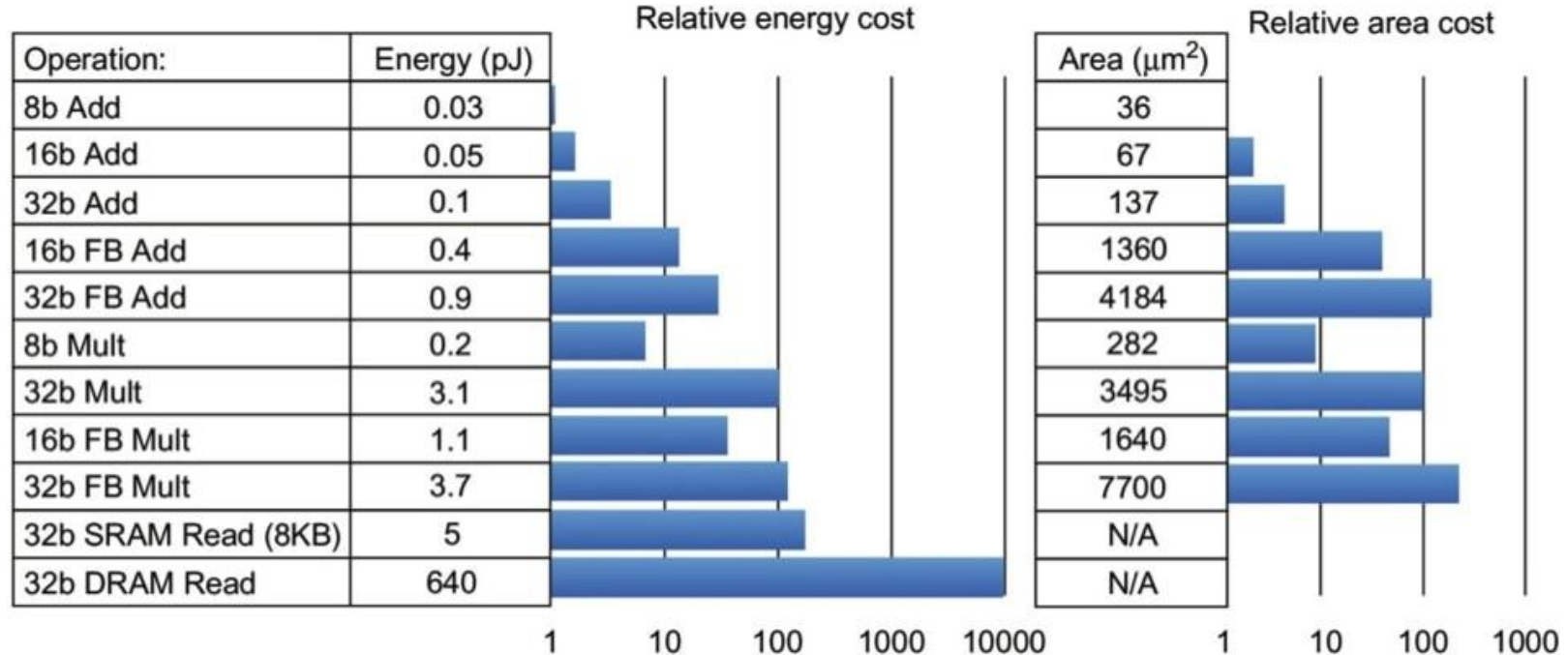
Power and energy

- NB.: Power = Energy per unit time ($1 \text{ W} = 1 \text{ J/s}$)
- The biggest challenge faced by computer designers (microprocessors have hundreds of pins and multiple interconnection layers for power and ground)
- Intel Core i7-6700K @4 GHz consumes 95 W

Improving energy efficiency

- Turn-off the clock of inactive modules
- Frequently there is no need to operate at the highest clock frequency and voltages (DVFS)
- Design for the typical case (use safety mechanisms to degrade performance in case of excessive power use)
- Overclocking (run c. 10% faster for short periods)

Energy and die area



Amdahl's law

- The performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used
- Amdahl's law is applicable beyond performance (e.g. to reliability)

Learning task

- Check Wikipedia's [CPU power dissipation](#) page
- Use Amdahl's law to explain the following statement found in that page: “Many multi-threaded development paradigms introduce overhead, and will not see a linear increase in speed when compared to the number of processors”



Thanks for
your attention