Subject: Algorithm Acceleration in Heterogeneous Systems (AAHS)
Code: 32439
Institution: Escuela Politécnica Superior
Degree: Master's program in Research and Innovation in Information and Communications Technologies (I^2-ICT)
Level: Master
Type: Elective [High Performance Systems]
ECTS: 6

COURSE GUIDE: Algorithm Acceleration in Heterogeneous Systems (AAHS)

Academic year: 2012-2013

Program: Master's program in Research and Innovation in Information and Communication Technologies (I^2-CIT)
Center: Escuela Politécnica Superior
University: Universidad Autónoma de Madrid

Last modified: 2012/07/16
Status: Published 2012/09/13

1. ASIGNATURA / COURSE (ID)

Aceleración de Algoritmos en Sistemas Heterogéneos
Algorithm Acceleration in Heterogeneous Systems (AAHS)

1.1. Programa / program

Máster Universitario en Investigación e Innovación en Tecnologías de la Información y las Comunicaciones (I^2-TIC)

Master in Research and Innovation in Information and Communication Technologies (I^2-CIT) [Officially certified]

1.2. Course code

32439

1.3. Course areas

Computer Architecture

1.4. Tipo de asignatura / Course type

Optativa [itinerario: Sistemas de altas prestaciones]
Elective [itinerary: High-Performance Systems]

1.5. Semester

Second semester

1.6. Credits

6 ETCS

1.7. Language of instruction

The lecture notes are in English. The lectures are mostly in Spanish. Some of the lectures and seminars can be in English.

1.8. Recommendations / Related subjects

Knowledge of C language at an introductory level is useful to follow the laboratory assignments.

Related subjects are:

- Cálculo intensivo y manejo de datos a gran escala [Numerical and data-intensive computing]
- Sistemas reconfigurables avanzados [Advanced reconfigurable systems]
- Plataformas de computación en un chip [Computing platforms on a chip]

1.9. Lecturers

Add @uam.es to all email addresses below.

Lectures and labs:

Dr. Francisco Javier Gómez Arribas (Coordinator)
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Office: C-226
Tel.: +34 914972255
e-mail: francisco.gomez
Web: http://www.eps.uam.es/~fjgomez

Dr. Iván González Martínez
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Office: C-223
Tel.: +34 914976212
e-mail: ivan.gonzalez
Web: http://www.eps.uam.es/~igonzale

Dr. Gustavo Sutter
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Office: C-244
Tel.: +34 914976209
e-mail: gustavo.sutter
Web: http://www.arithmetic-circuits.org/gsutter

1.10. Objetivos de la asignatura / Course objectives

En esta asignatura se estudia como conseguir aceleración y aumento de prestaciones en sistemas de computación con arquitectura heterogénea. Se utilizan coprocesadores implementados en hardware reconfigurable y sistemas con coprocesadores gráficos, empleando respectivamente lenguajes de descripción hardware de alto nivel (HLL) y programación GPGPU.

This subject introduces acceleration and performance enhancement in computing systems with heterogeneous architectures. Coprocessors implemented in reconfigurable hardware, as well as graphic coprocessors, will be utilized, using respectively high-level hardware description languages (HLL) and GPU programming.

At the end of each unit, the student should be able to:

UNIT BY UNIT SPECIFIC OBJECTIVES	
UNIT 1.- Introduction to HW/SW systems	
1.1.	Understand the Models of Computation.
1.2.	Be familiar with some System-level design languages.
1.3.	Understand the basis of Design for Reconfigurable Systems using High-level Synthesis
1.4.	Describe the architecture and features or several Platforms for Reconfigurable Computing.
UNIT 2.- Synthesis from High-Level Languages (HLLs)	
2.1.	Understand why is interesting to use HLLs.
2.2.	Compare HLLs vs. HDLs.
2.3.	Describe the Panorama de HLLs.
2.4.	Differentiate the main features of: Impuse-C, Handel-C, AutoESL.
2.5.	Translate algorithm codification from C to Handel-C.
UNIT 3.- AutoESL	
3.1.	Understand the high-level synthesis flow of AutoESL.
3.2.	Apply directives to optimize design performance.
3.3.	Create a custom peripheral and add it to a processor system
3.4.	Get system level design experience using HLLs.
3.5.	Translate a C coded algorithm to AutoESL.
UNIT 4.- Heterogeneous Computing GPGPU Basis	
4.1.	Describe the main aspects of an architecture based on GPU coprocessors.
4.2.	Use of GPGPU: General-purpose computing on graphics processing units.
4.3.	Understand the CUDA Programming Model.
UNIT 5.- Algorithm techniques for scalable many-core programming	
5.1.	Understand the GPU limitations
5.2.	Take advantage of the GPU computing capabilities
5.3.	Learn how to optimize your application
UNIT 6.- Advanced GPU programming	
6.1.	Learn advanced GPU techniques

6.2.	Know new programming standards for GPUs
6.3.	Learn other GPU-related technologies

1.11. Course contents

PART I

1. **Introduction to Hardware/Software Systems**
    a. Models of Computation and Languages
    b. System-level design languages.
    c. Design for Reconfigurable Systems using High-level Synthesis
    d. Platforms for Reconfigurable Computing.

2. **Synthesis from High-Level Languages (HLLs)**
    a. Motivation for using Higher Level Languages.
    b. HLLs vs. HDLs.
    c. Panorama de HLLs.
        i. Impuse-C
        ii. Handel-C
        iii. AutoESL
    d. Handel-C Tutorial
        i. Getting started. Extensions to the C language
        ii. Parallelism and Scheduling
3. **AutoESL**
    a. Introduction and demo of an AutoESL generated core.
    b. Basic design flow and review generated output.
    c. Optimization
        i. Improving performance: use pipelining.
        ii. Improving area: directives to resource sharing.
    d. Handling Block and Port Level Protocols
        i. Data types
        ii. Coding Style
    e. Creating a Processor System

PART II

4. **Heterogeneous Computing: GPGPUs basis**
    a. System architecture with GPU graphics coprocessor
        i. SPMD with multiple cores.
        ii. Programming general purpose GPU (GPGPU).
    b. CUDA Architecture.
        4.2.1. CUDA programming environment.
        4.2.2. Programming model.
        4.2.3. CUDA API.

5. **Algorithm techniques for scalable many-core programming**
    a. CUDA Threading Model.

b. Memory and bandwidth.
c. Computation patterns.
d. Performance Measurement.
e. Debugging and Profiling.

6. **Advanced GPU programming**

   a. Dynamic Parallelism.
   b. Multi-GPU Programming.
   c. New programming standards
      i. OpenCL.
      ii. OpenACC.
      iii. NVIDIA GPUDirect.

1.12. Course bibliography

1. Handel-C Language Reference Manual. Embedded Solutions Limited: Version 2.1, 2004.
2. Practical FPGA Programming in C. David Pellerin; Scott Thibault, Ed Prentice Hall, 2005
3. SystemC 2.0.1 Language Reference Manual. Open SystemC Initiative : San Jose, California. 2003
4. Xilinx AutoESL High-Level Synthesis (HLS) Language Reference Manual 2012.
5. Programming massively parallel processors A hands-on Approach. David B. Kirk, Wen-mei W. Hwu. Morgan Kaufmann. 2010. ISBN 9780123814722. Ref_UAM: INF/681.324/KIR.
6. CUDA application design and development. Rob Farber. Ed. Morgan Kaufmann. ISBN 9780123884268. Ref_UAM: INF/681.3.06/FAR.

1.13. Coursework and evaluation

The course involves lectures, assignments, lab assignments, a seminar presentation and an exam.

It will also include lab practices and it is necessary to pass both practice and classroom assignments to pass the subject.

Finally, an extraordinary exam will also be available for those students who do not turn in the assignments.

In both the ordinary and the extraordinary exam period it is necessary to have a pass grade (≥ 5) in each of the exams to pass the course.