CPEN 432
Real-Time System Design

Arpan Gujarati
University of British Columbia

Land Acknowledgement

- I would like to acknowledge that we are gathered today on the traditional, ancestral, and unceded territory of the Musqueam people.
  - UBC Vancouver Indigenous Portal
  - UBC Life blog: What you need to know about learning on Musqueam territory

Who am I?

- Arpan Gujarati
    - Research Associate in CS
    - https://arpangujarati.github.io/
- Education
    - BE from BITS Pilani, India
    - PhD from MPI-SWS, Germany
        - Saarland University
        - TU Kaiserslautern
- Research Interests
    - Real-time systems, distributed systems, fault tolerance, reliability analysis, and scheduling
    - Domains: Cyber-physical systems, datacenter systems

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

UNIVERSITÄT
DES
SAARLANDES

TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Overview

Course Staff and Logistics

- Staff
  - Instructor: Arpan Gujarati (arpanbg@cs.ubc.ca)
  - Teaching Assistant: Geetika Batta (geetikb@ece.ubc.ca)

- Course webpage: https://cpen432.github.io/

- Lectures and tutorials will be online (until further announcements)
  - Lectures: Mondays and Wednesdays from 9:30 AM to 11:00 AM
  - Tutorials: Fridays from 10:00 AM to 12:00 Noon

- We will use Piazza for course-related discussion
  - Piazza page: https://piazza.com/ubc.ca/winterterm22021/cpen432
  - Access code: LIU&LAYLAND

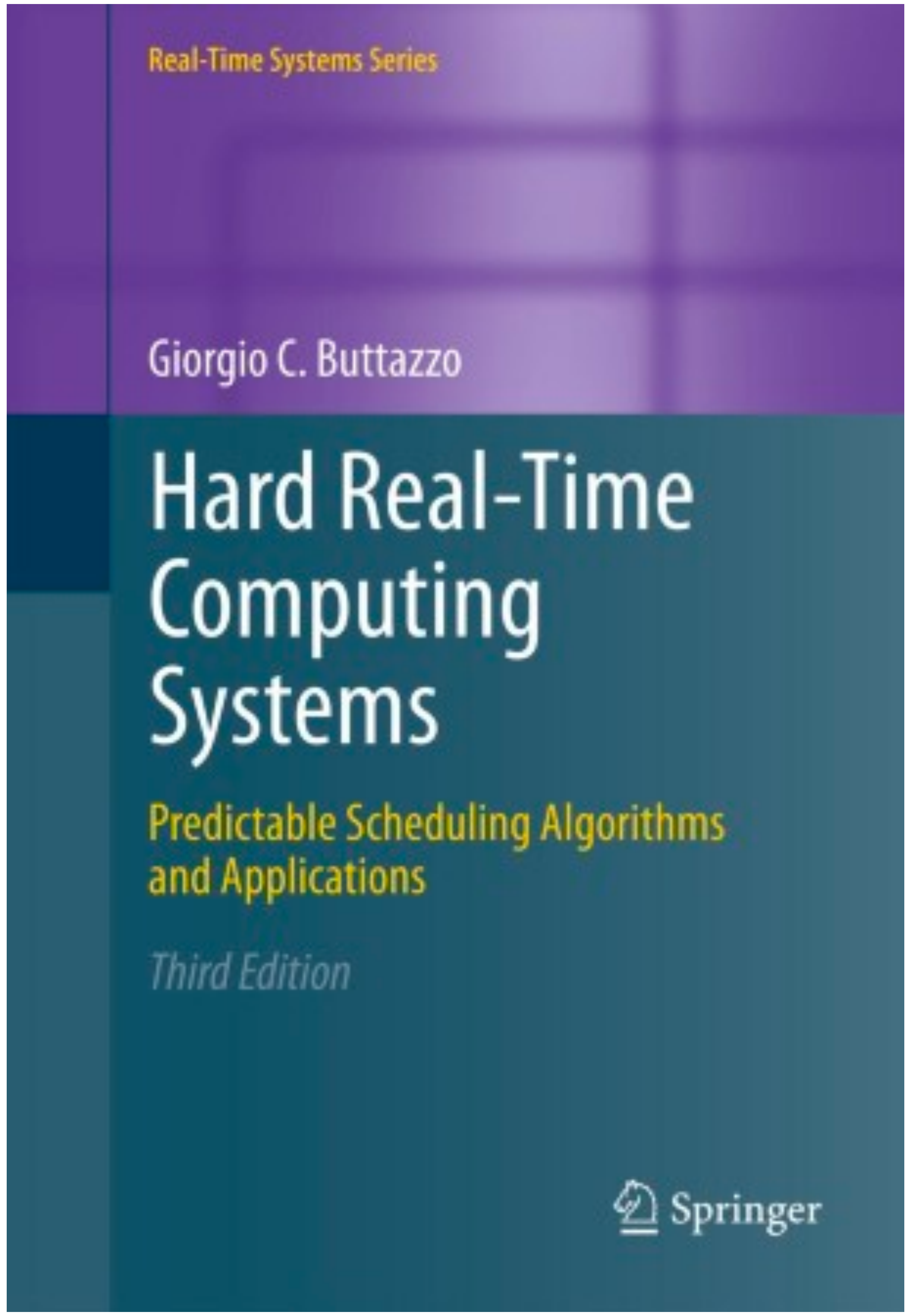
- Office hours
  - TBA. You may use the tutorial hours for now.

Readings

- Textbook
  - *Giorgio C. Buttazzo, Hard Real-Time Computing Systems, 3rd edition.*
  - The PDF is accessible via UBC library

- Material not covered in the text will be posted on the readings page (slides, research papers, etc.)
  - https://cpen432.github.io/readings/

- Optional
  - Edward A. Lee and Sanjit A. Seshia, Introduction to Embedded Systems, 2nd edition.
  - Available online at LeeSeshia.org



In this course, you will learn …

- What are real-time systems?
  - How do they differ from general-purpose systems?
- Tools and techniques to model, analyze, and design real-time systems
  - i.e., real-time systems toolbox
- Lots of embedded systems programming!
  - Reading datasheets and manuals

Lectures will cover the theory

Projects will cover the practical aspects

Prerequisites

- Understanding of operating systems
  - CPEN 331 (concepts and implementation) or equivalent courses
- Programming
  - A fair amount of experience with C programming (especially “pointers”)
  - Basic knowledge of assembly-language programming
- Math and logical reasoning
  - Algorithm design and analysis
  - Proving correctness and optimality
  - Basic probability
- Tools
  - Source code management software such as GitHub and GitLab
  - LaTeX for typesetting written assignments

Grading

- **Group projects (40%)**
  - Embedded systems programming
    - Learn from tutorials. We will provide all resources, but you will have to read them on your own!
  - We will provide all the necessary hardware (Raspberry Pi, USB-to-TTL cable, micro-SD card)
    - Groups of **5 (we have limited hardware!)**
  - Submission instructions TBA

- **Homework assignments (30%)**
  - Based on the material covered in the lectures
  - Individual work!
    - Discuss with colleagues, but **final write-up should be your own**
  - Must be typeset in LaTeX (we will provide a template)
    - Handwritten submissions will not be graded
  - Submission instructions TBA

- **Final examination (30%)**
  - Based on **both** project and lecture materials
  - Open-book

Tutorials

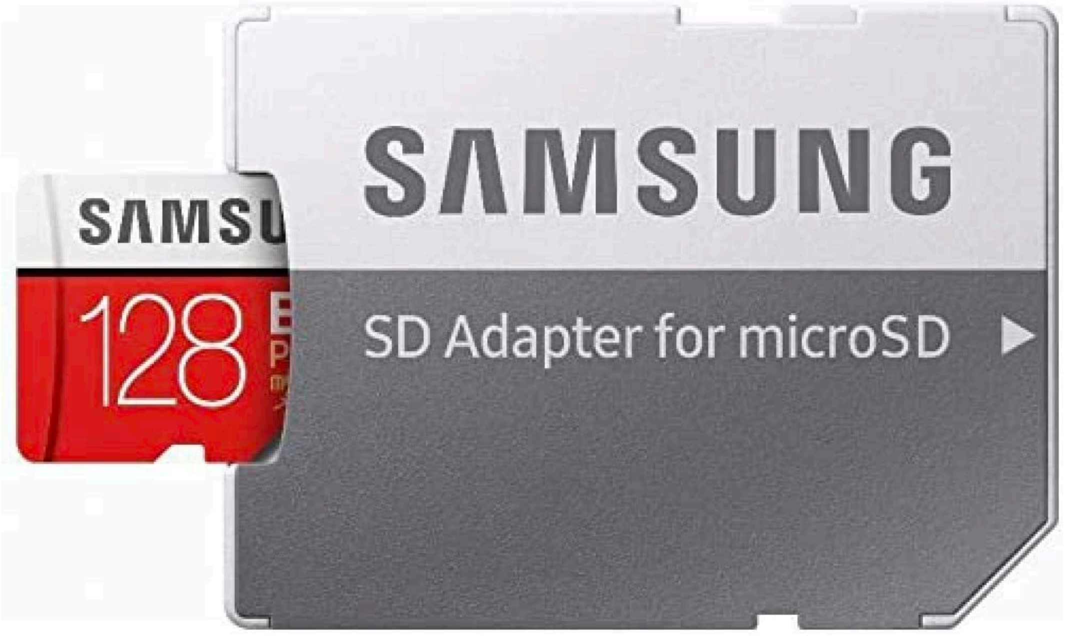
- Fridays
  - Session 1: 10:00 AM to 11:00 AM (**Arpan**)
  - Session 2: 11:00 AM to 12:00 Noon (**Geetika**, Arpan)
- Project grading (demos) and checkpointing
- Discuss homework assignment solutions (if needed)
- Questions related to the lecture material, etc.

Acknowledgements

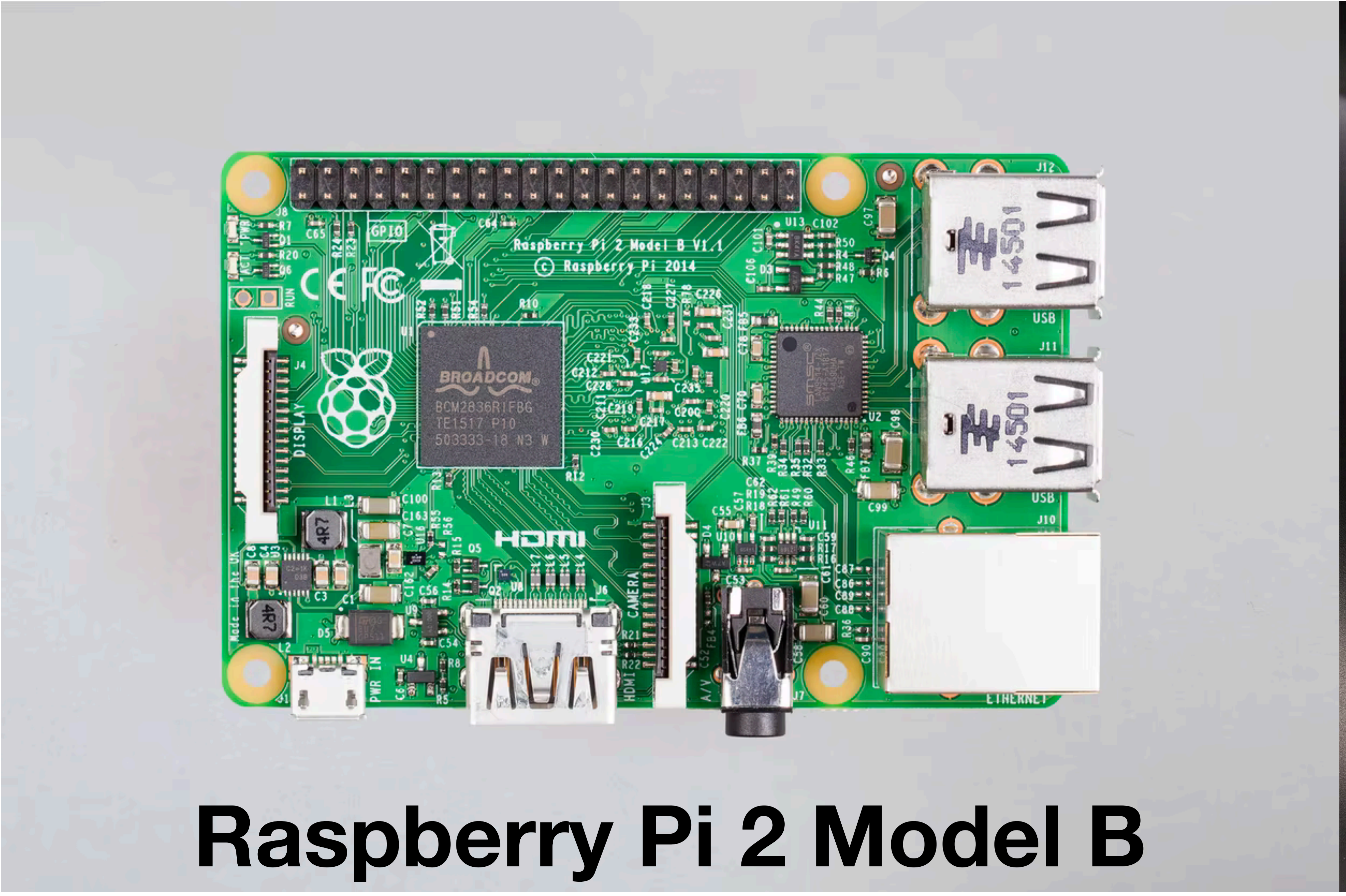
- The course is adopted from CPEN 432 (2020-21) by Bader Alahmad
- Some slides are adopted from Foundations of Cyber-Physical Systems by Rupak Majumdar and Björn Brandenburg at MPI-SWS (2014-15)

Projects

Hardware



microSD Card



Raspberry Pi 2 Model B
+ Power Adapter

USB to TTL

Project Timeline (dates are tentative!)

#	Title	Release	Deadline*
1	Bare-metal programming: UART, timer, printing, code optimization	17 January	6 Febuary
2	More bare-metal: Interrupt handling, application, testing and validation	7 February	27 February
3	Port FreeRTOS to RPi; implement periodic tasks, admission control, and resource access control	28 February	20 March
4	Supporting aperiodic tasks and multiprocessor scheduling	21 March	10 April

* Deadline of 6 February implies 11:59 PM Vacouver local time on 6 February

Late Project Submissions

- Each group is allocated an automatic extension of 4 calendar days
  - Groups can use the extension on any project component in increments of 1 day
    - E.g., you can hand in one project assignment 4 days late, or one assignment 2 days late and two assignments 1 day late, or 4 project assignments 1 day late
  - This extension is to be used for incidental occurrences, so **please do not ask us for any more extensions due to any incidental circumstances**
  - If a student moves to a different group, the minimum remaining extension time in the student's former and new groups will be considered

- **No credit for late submissions beyond the automatic extensions**

Action Items (complete by 16 January)

1. Form groups of **five**
   - You may use Piazza to find group members
   - Provide **Gitlab** usernames of all group members to **both instructors over email**
   - Use subject line “[CPEN 432] Project Group Registration”

2. Collect hardware
   - Pick a five minute slot during the tutorial hour on 14 January
   - Schedule a meeting with **both instructors over email** for collecting the hardware
   - Use subject line “[CPEN 432] Hardware Collection”

Cyber-Physical Systems

List all domains/applications where computing systems are used today …

1. My vacuum cleaner iRobot
   contains a tiny computer!

2. ???

What is a cyber-physical system? (3/4)

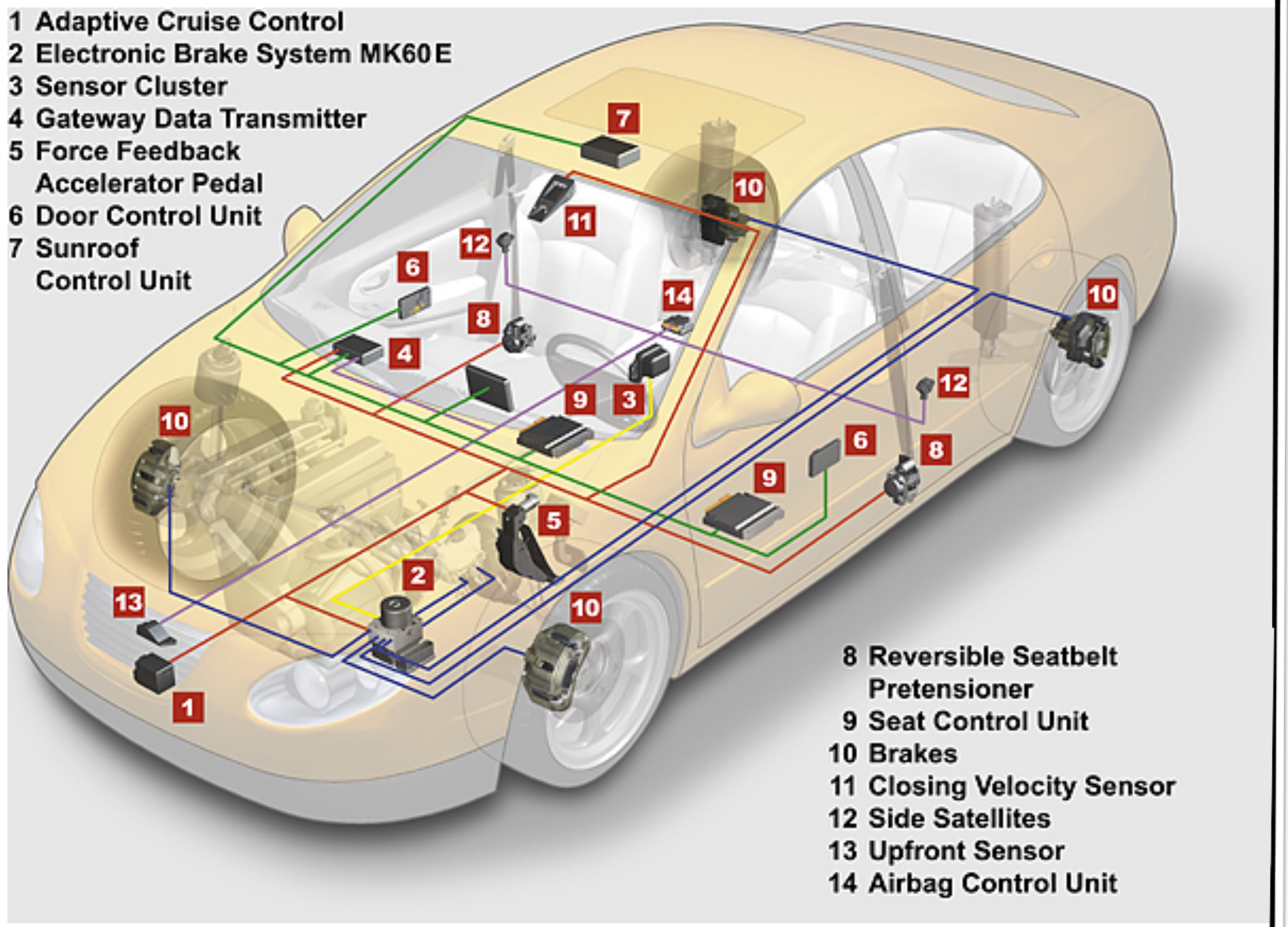
"Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa. The design of such systems, therefore, requires understanding the joint dynamics of computers, software, networks, and physical processes. It is this study of joint dynamics that sets [the discipline of cyber-physical systems] apart."[Der11]

— *Patricia Derler et al., 2011*

[Der11] P. Derler et al. (2011). Modeling Cyber–Physical Systems.

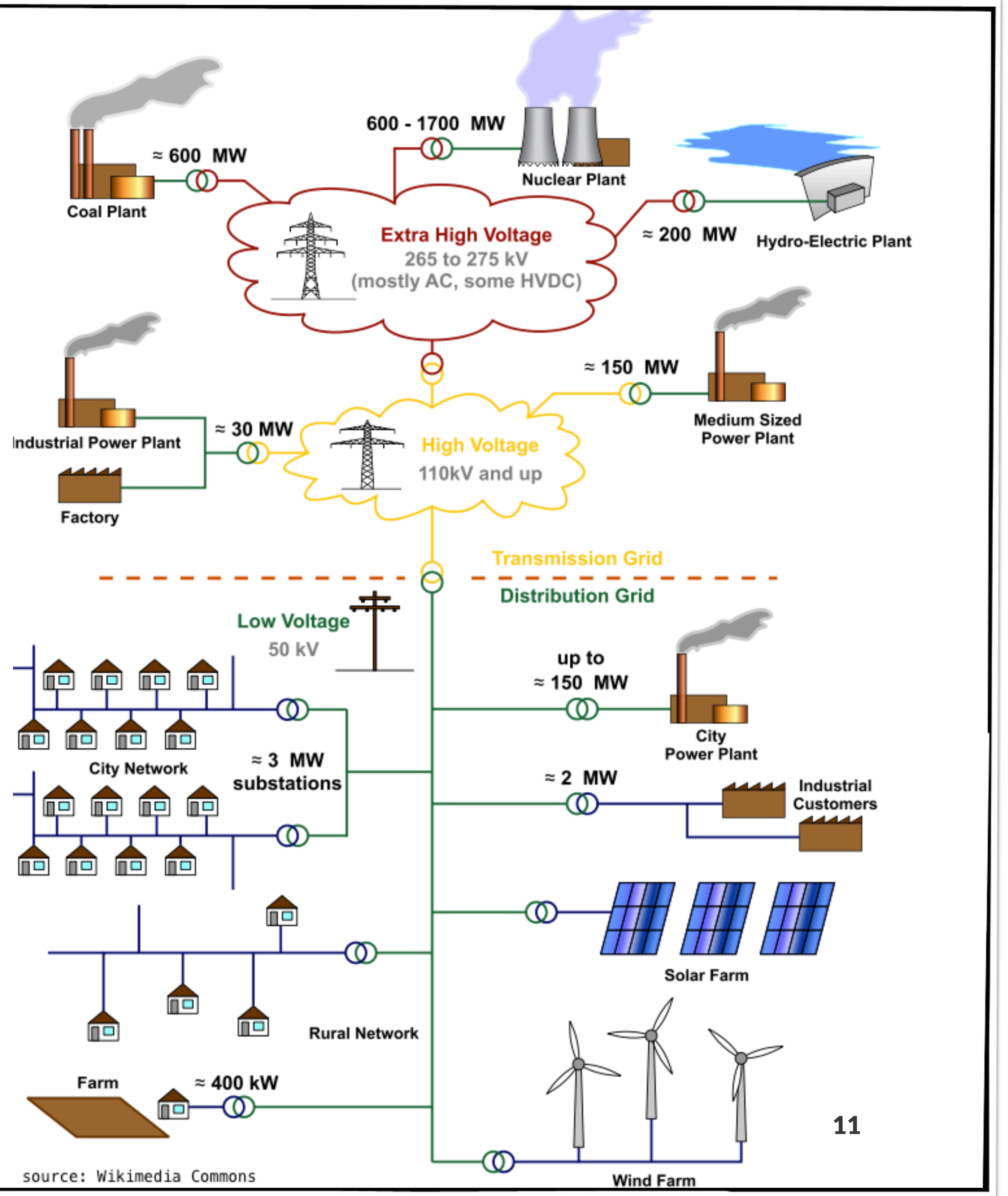
Example CPS (1/3): Automotive Systems

- distributed sensors and actuators
- many driver assistance systems: anti-lock breaking system (ABS), electronic stability control (ESC), adaptive cruise control, adaptive light control, lane departure warning, X-by-wire ... → **complex interactions**
- Safety determined by physics!



Example CPS (2/3): Power Grid

- constantly changing demand and supply
- limited power line capacity
- supply must be controlled and adjusted
- inherently distributed and time-critical
- possibility of cascading failures
  (➞ 2003 US & Canada power outage)
- The physics of power transmission determine and limit safe operation.



source: Wikimedia Commons

credit: CLAAS E-Systems KGaA mbH & Co KG

Example CPS (3/3): Precision Agriculture

- current agricultural practices waste large amounts of water, seed, fertilizer, and toxins
- large negative economic and environmental impact
- **precision agriculture**: use GPS, drones, and autonomous vehicles to cut down on waste
- distributed sensing, mapping, decision making, and control

Challenges Specific to CPS …

1. Modeling physics is hard!

2. ???

Cyber-Physical Systems — Challenges (1/3)

- physical processes are **continuous** (typically modeled as differential equations), whereas software and computers are **discrete** (typically modeled as automata)
- nature is concurrent — **concurrency** is inherent in CPSs
- nature doesn't stop or slow down — **timeliness** is critical
- communication and computation takes time — **implementation performance** matters and cannot be "abstracted away"

Cyber-Physical Systems — Challenges (2/3)

- stuff happens — must deal with noise, delays, breakdowns, …
- the digital components are often **distributed** — partial failures
- limited energy, size, weight, cost budgets — **resource constraints**
- CPS are often part of a larger system — **open systems** interfacing with other, independent systems
- attackers might target critical infrastructure — **security** concerns

Cyber-Physical Systems — Challenges (3/3)

- often too large and **too important for trial and error** — cannot build a copy of the power grid, cannot afford autonomous vehicle failures
- by definition CPSs, interact with and control the real world — many CPSs are thus **safety-critical**
- **certification** requirements — must meet stringent correctness and documentation criteria

Focus of this Class

Real-Time System Design

CPS vs. General-Purpose Computing

"When studying CPS, certain key problems emerge that are rare in so-called general-purpose computing. For example, in general-purpose software, **the time it takes to perform a task** is an issue of **performance**, not correctness. It is not incorrect to take longer to perform a task. It is merely less convenient and therefore less valuable. In CPS, the time it takes to perform a task may be **critical** to [the] correct functioning of the system."[Der11]

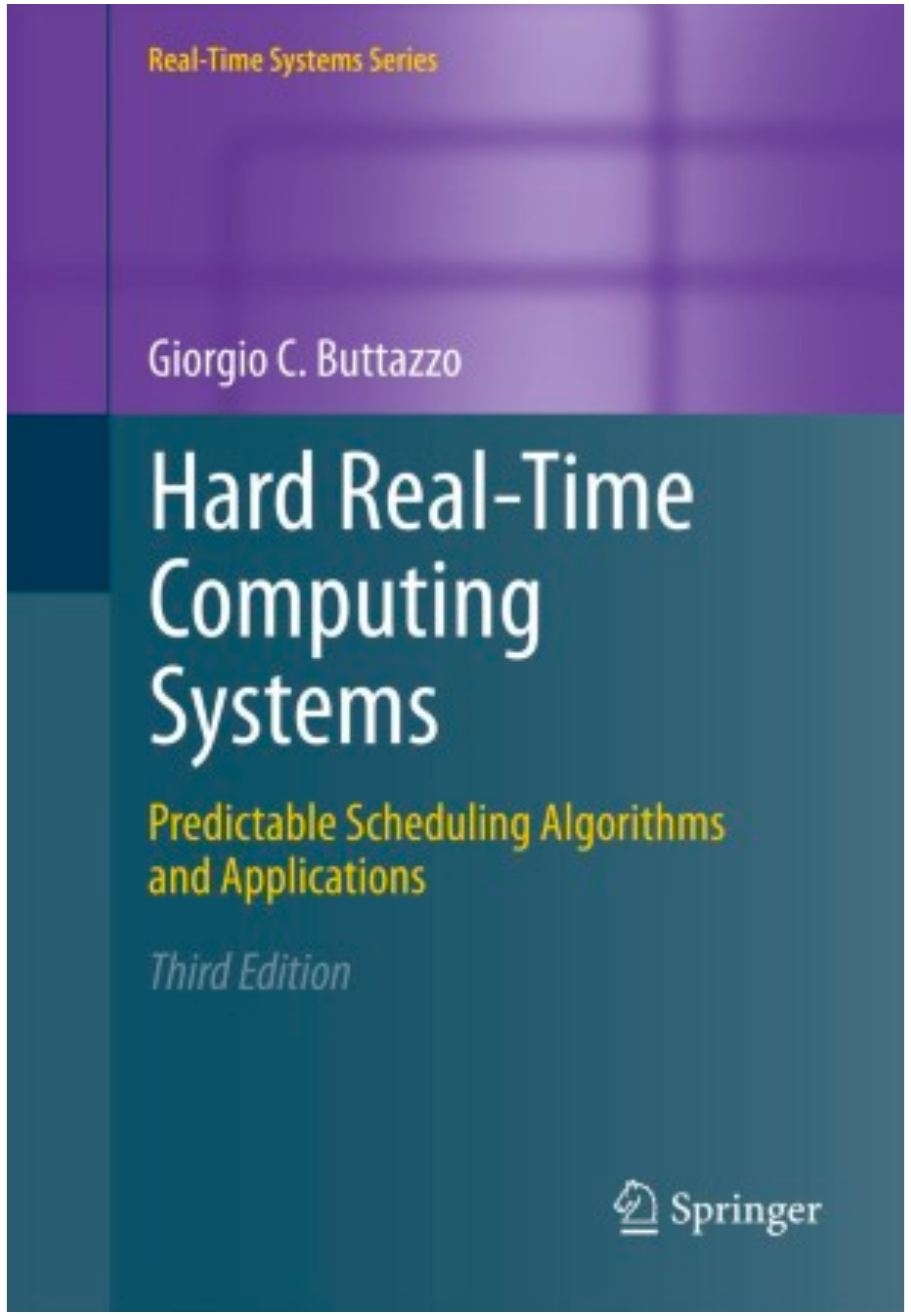
— *Patrica Derler et al., 2011 (emphasis added)*

[Der11] P. Derler et al. (2011). Modeling Cyber–Physical Systems.

1.1 INTRODUCTION

Real-time systems are computing systems that must react within precise time constraints to events in the environment. As a consequence, the correct behavior of these systems depends not only on the value of the computation but also on the time at which the results are produced [SR88]. A reaction that occurs too late could be useless or even dangerous. Today, real-time computing plays a crucial role in our society, since an increasing number of complex systems rely, in part or completely, on computer control. Examples of applications that require real-time computing include the following:

- Chemical and nuclear plant control,
- control of complex production processes,
- railway switching systems,
- automotive applications,



Goal

- Model and analyze the timing properties of real-time systems in advance
- Learn system design principles that make such *a priori* analysis possible
- Understand the interplay between timing and other CPS challenges

This course: Real-Time Toolbox

- Basic concepts
  - Different types of deadlines, priorities, blocking, etc.
- Estimating worst-case execution times
- Estimating worst-case response times
- Periodic versus aperiodic task models and their implications
- Optimality results in real-time scheduling
- Blocking, synchronization, and resource access protocols
- Reliability and fault tolerance