Embedded Systems

1 - Introduction

© Lothar Thiele

Computer Engineering and Networks Laboratory

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture Organization

Herbstsemester 2021

227-0124-00L Embedded Systems

Daten der Belegungseinschränkung

Platzzahl	Aktuelle Belegung
200	200

Anzahl Studierende in der Warteliste zum Zeitpunkt 14.09.2021 13:23:37: 122

Organization

WWW: https://www.tec.ee.ethz.ch/education/lectures/embedded-systems.html

Lecture: Lothar Thiele, thiele@ethz.ch; Michele Magno <michele.magno@pbl.ee.ethz.ch>

Coordination: Seonyeong Heo (ETZ D97.7) <seoheo@ethz.ch>

References:

- *P. Marwedel:* Embedded System Design, Springer, ISBN 978-3-319-85812-8/978-3-030-60909-2, 2018/2021.
- *G.C. Buttazzo:* Hard Real-Time Computing Systems. Springer Verlag, ISBN 978-1-4614-0676-1, *2011.*
- *Edward A. Lee and Sanjit A. Seshia:* Introduction to Embedded Systems, A Cyber-Physical Systems Approach, Second Edition, MIT Press, ISBN 978-0-262-53381-2*, 2017.*

Sources: The slides contain ideas and material of J. Rabaey, K. Keuzer, M. Wolf, P. Marwedel, P. Koopman, E. Lee, P. Dutta, S. Seshia, and from the above cited books.

Organization Summary

- **Lectures** are held on Mondays from 14:15 to 16:00 in ETF C1 until further notice. Life streaming and slides are available via the web page of the lecture. In addition, you find audio and video recordings of most of the slides as well as recordings of this years and last years life streams on the web page of the lecture.
- **Exercises** take place on Wednesdays and Fridays from 16:15 to 17:00 via Zoom. On Wednesdays the lecture material is summarized, hints on how to approach the solution are given and a sample question is solved. On Fridays, the correct solutions are discussed.
- **Laboratories** take place on Wednesdays and Fridays from 16:15 to 18:00 (the latest). On Wednesdays the session starts with a short introduction via Zoom and then questions can be asked via Zoom. Fridays are reserved for questions via Zoom.

Further Material via the Web Page

Lecture Slides

All lecture slides are available for download as a bundle:

- Embedded Systems lecture slides [single page format] ⤓
- Embedded Systems lecture slides [4on1 page format] ⤓

Lecture Recordings

Life Recordings Autumn 2021

The life recordings of the lectures in Autumn Semester are available at the following link: Embedded Systems Life Recordings AS 2021.

Life Recordings Autumn 2020

The life recordings of last years lecture are available at the following links:

1. Lecture 1: Chapters 1. Introduction and 2. Software Development
2. Lecture 2: Chapters 2. Software Development and 3. Hardware-Software Interface

Audio and Videos of Selected Chapters

Some of the chapters are documented via carefully recoreded videos. They contain some of the slides as well as audio explanations.

- 1. Introduction
- 2. Software Development
- 3. Hardware Software Interface

Exercises and Laboratory

Generic Documents	
Embedded System Companion	Supplementary Material
Remote Installation Instructions	
Documents for Lab 0	
Handout	Source (code)
Slides and videos	Solution (code and handout)
Documents for Lab 1	
Handout	Source (code)
Slides and videos	Solution (code and handout)
Documents for Lab 2	
Handout	Source (code)
Slides and videos	Solution (code and handout)
Documents for Lab 3	

When and where?

Schedule

	When	Where
Lectures	Monday 14:15 - 16:00	ETF C1
Exercises	Wednesday 16:15 - 17:00 Friday 16:15 - 17:00	Zoom Zoom
Labs	Wednesday 16:15 - 18:00 Friday 16:15 - 18:00	Zoom Zoom

Timetable

Date	Lecture	Exercice	Lab
27.09.2021	1. Introduction 2. Software Development		
29.09./01.10.2021			0. Prelab [MM]
04.10.2021	3. Hardware-Software Interface		
06./08.10.2021			1. Bare Metal Program-

What will you learn?

- Theoretical foundations and principles of the analysis and design of embedded systems.
- Practical aspects of embedded system design, mainly software design.

The course has three components:

- *Lecture:* Communicate principles and practical aspects of embedded systems.
- *Exercise*: Use paper and pencil to deepen your understanding of analysis and design principles .
- *Laboratory (ES-Lab)*: Introduction into practical aspects of embedded systems design. Use of state-of-the-art hardware and design tools.

Please read carfully!!

- https://www.tec.ee.ethz.ch/education/lectures/embedded-systems.html

Exercises and Laboratory

We urgently ask all students to do the laboratory on their own hardware. For this, we provide you with a virtual machine that has all the necessary software already pre-installed. You can find the installation instructions on GitLab. We have tested this setup on PCs and Laptops with an USB port that run Windows 10, macOS Catalina, as well as Linux Mint and Linux Ubuntu 18.04 and 20.04; in general, all platforms which can run VirtualBox should work. In exceptional circumstances where this is not possible, students are allowed to use the computers in ETZ D61.1 or ETZ D96.1 during the regular laboratory hours (Wednesday or Friday 16.15 – 18.00). In such a case, please send an email with your name and Legi number to the lecture coordinator. You will receive a time slot and room allocation that guarantees that the maximum occupation of the computer rooms is respected. You are not allowed to enter ETZ D61.1 or ETZ D96.1 during the laboratory hours if you do not have an allocated slot.

What you got already…

Be careful and please do not …

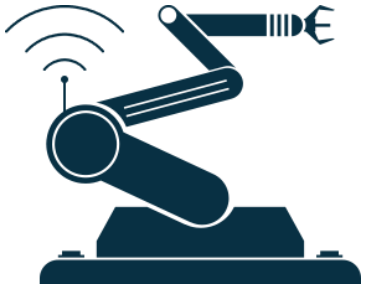
You have to return the board at the end!

Embedded Systems - Impact

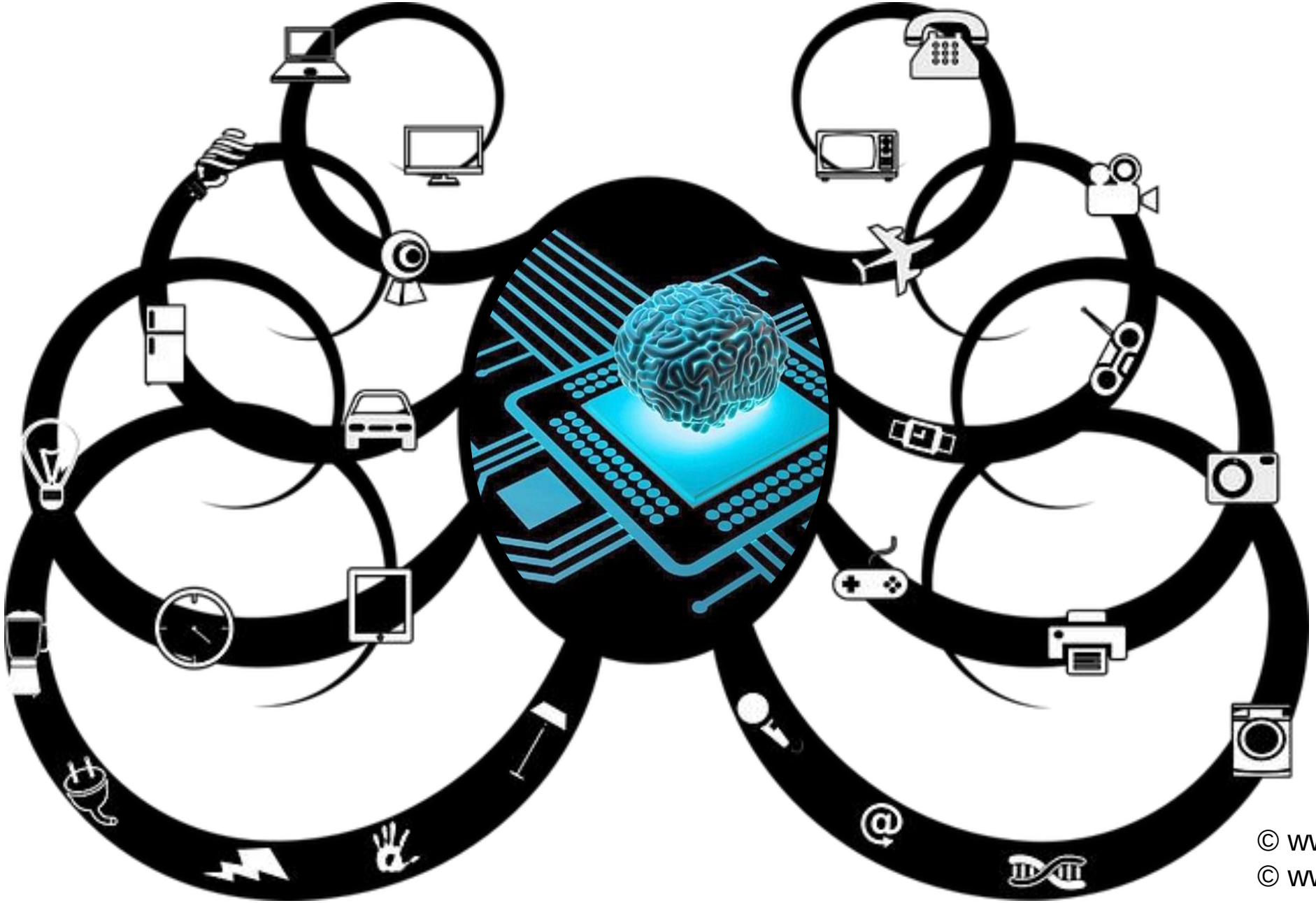
Embedded Systems

Embedded systems (ES) = **information processing systems embedded into a larger product**

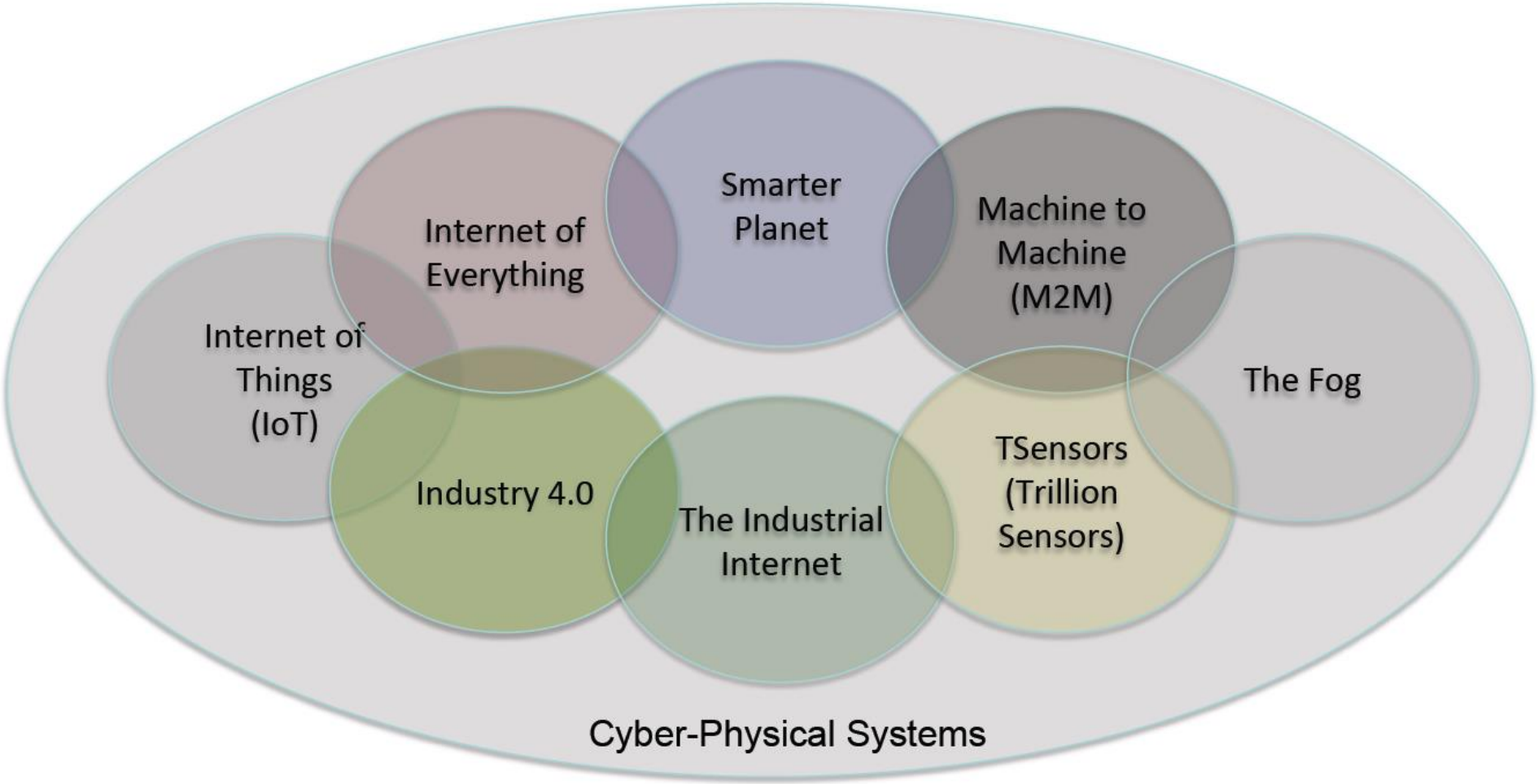
Examples:



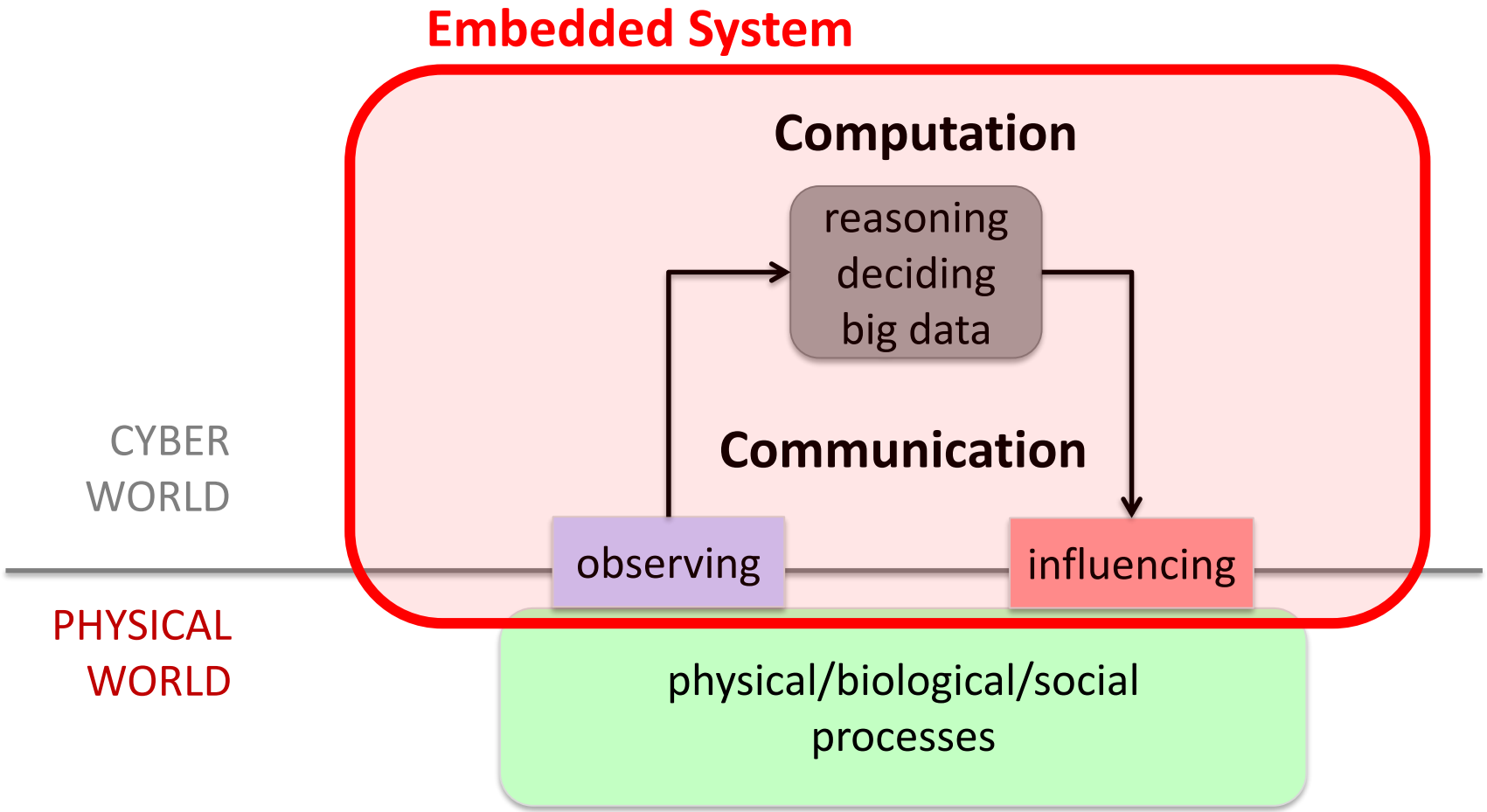
Often, the main reason for buying is not information processing



Many Names – Similar Meanings



Embedded System



Use feedback to influence the dynamics of the physical world by taking smart decisions in the cyber world

Reactivity & Timing

Embedded systems are often reactive:

- Reactive systems must react to stimuli from the system environment :

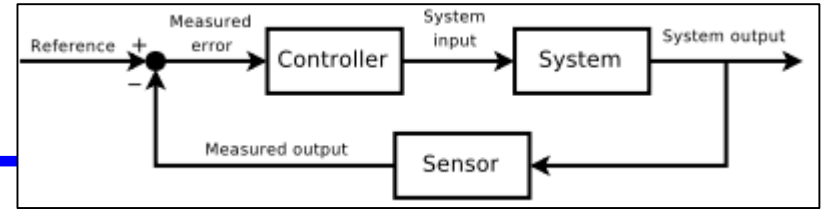
> *„A reactive system is one which is in continual interaction with is environment and executes at a pace determined by that environment“ [Bergé, 1995]*

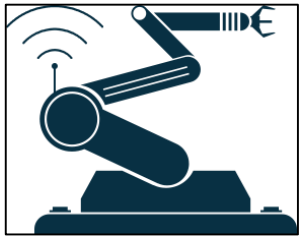
Embedded systems often must meet *real-time constraints:*

- For hard real-time systems, right answers arriving too late are wrong. All other time-constraints are called soft. A *guaranteed system response* has to be explained without statistical arguments.

> *„A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe“ [Kopetz, 1997].*

Predictability & Dependability





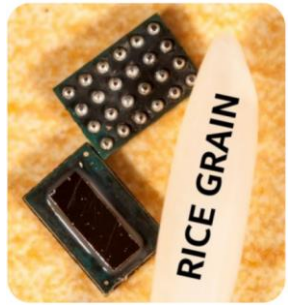
CPS = cyber-physical system

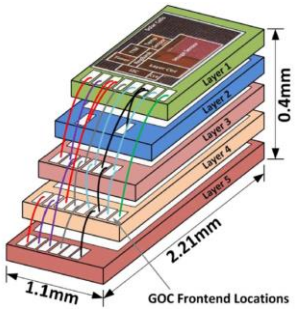
"It is essential to *predict* how a CPS is going to behave under any circumstances […] *before* it is deployed."[Maj14]

"CPS must *operate dependably*, safely, securely, efficiently and in real-time."[Raj10]

[Maj14] R. Majumdar & B. Brandenburg (2014). Foundations of Cyber-Physical Systems.
[Raj10] R. Rajkumar et al. (2010). Cyber-Physical Systems: The Next Computing Revolution.

Efficiency & Specialization





- Embedded systems must be *efficient:*
  - *Energy* efficient
  - *Code-size* and *data memory* efficient
  - *Run-time* efficient
  - *Weight* efficient
  - *Cost* efficient

Embedded Systems are often *specialized* towards a certain application or application domain:

- Knowledge about the expected behavior and the system environment at design time is exploited to *minimize resource usage* and to *maximize predictability and reliability*.

Comparison

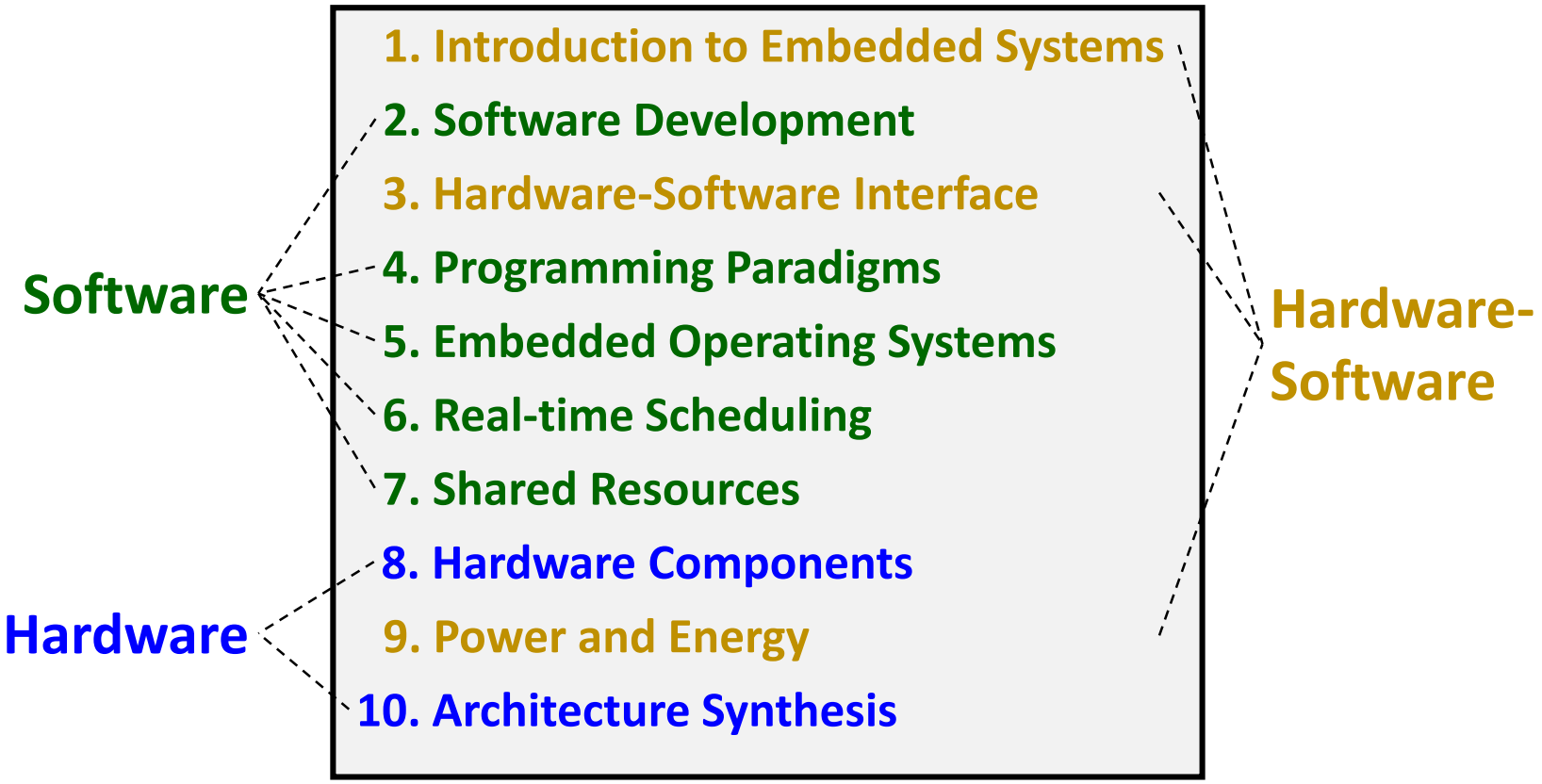
Embedded Systems:

- Few applications that are known at design-time.
- Not programmable by end user.
- Fixed run-time requirements (additional computing power often not useful).
- Typical criteria:
  - cost
  - power consumption
  - size and weight
  - dependability
  - worst-case speed

General Purpose Computing

- Broad class of applications.
- Programmable by end user.
- Faster is better.
- Typical criteria:
  - cost
  - power consumption
  - average speed

Lecture Overview



Components and Requirements by Example

Components and Requirements by Example

- Hardware System Architecture -

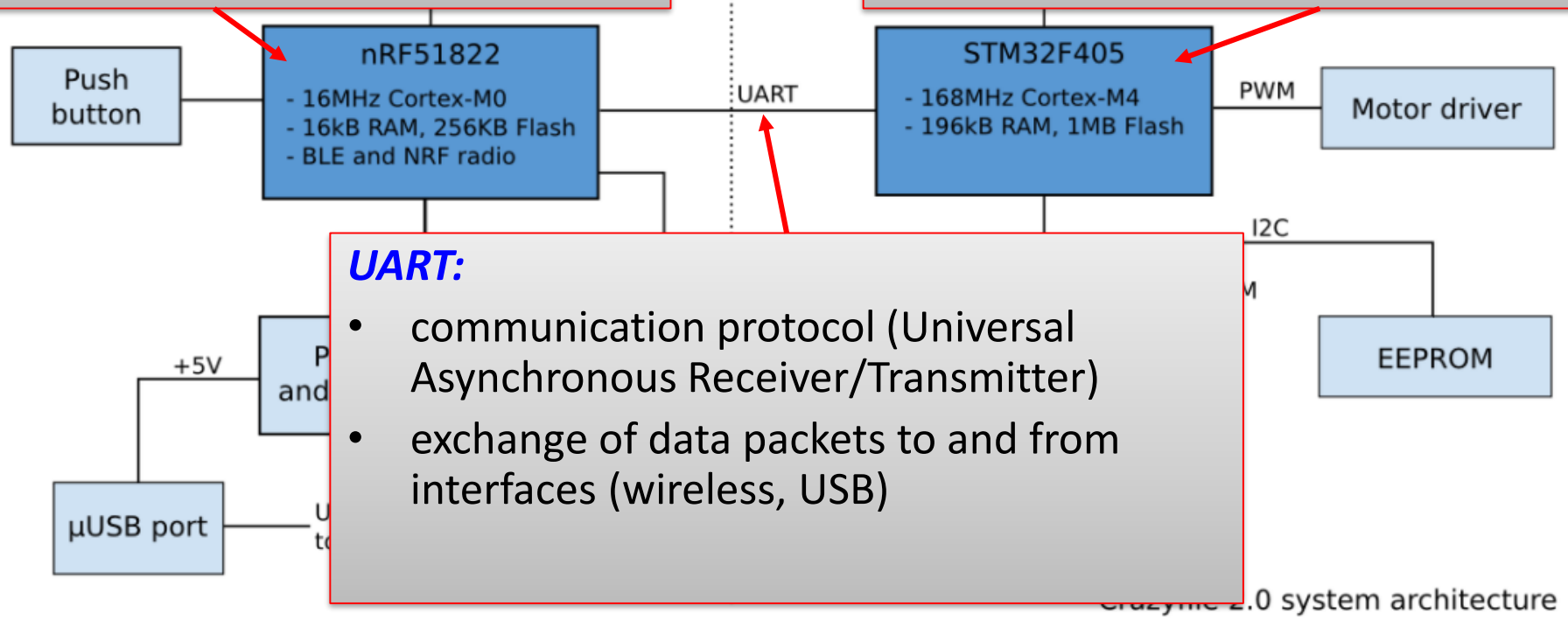
High-Level Block Diagram View

low power CPU

- enabling power to the rest of the system
- battery charging and voltage measurement
- wireless radio (boot and operate)
- detect and check expansion boards

higher performance CPU

- sensor reading and motor control
- flight control
- telemetry (including the battery voltage)
- additional user development
- USB connection



UART:

- communication protocol (Universal Asynchronous Receiver/Transmitter)
- exchange of data packets to and from interfaces (wireless, USB)

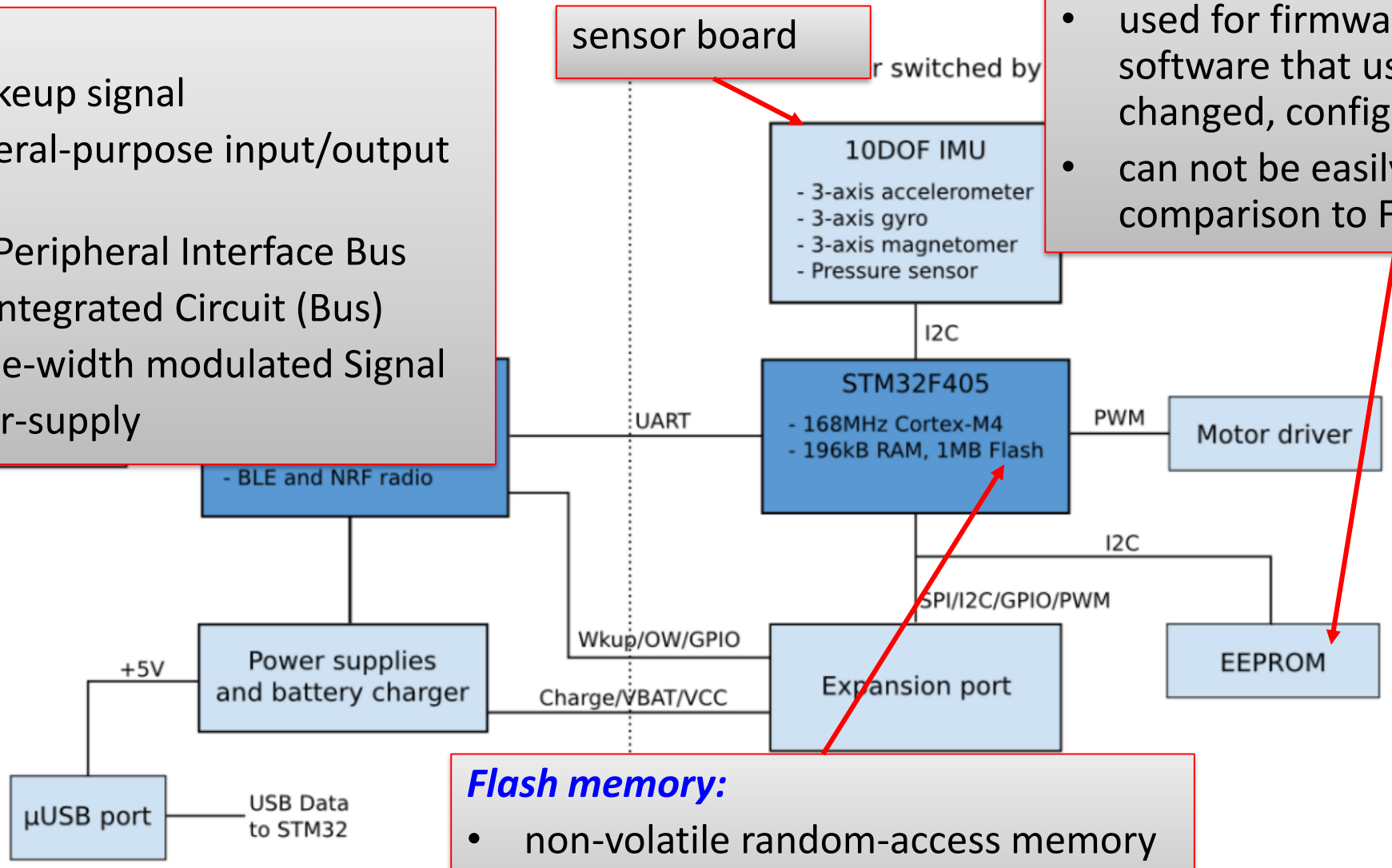
High-Level Block Diagram View

Acronyms:

- Wkup: Wakeup signal
- GPIO: General-purpose input/output signal
- SPI: Serial Peripheral Interface Bus
- I2C: Inter-Integrated Circuit (Bus)
- PWM: Pulse-width modulated Signal
- VCC: power-supply

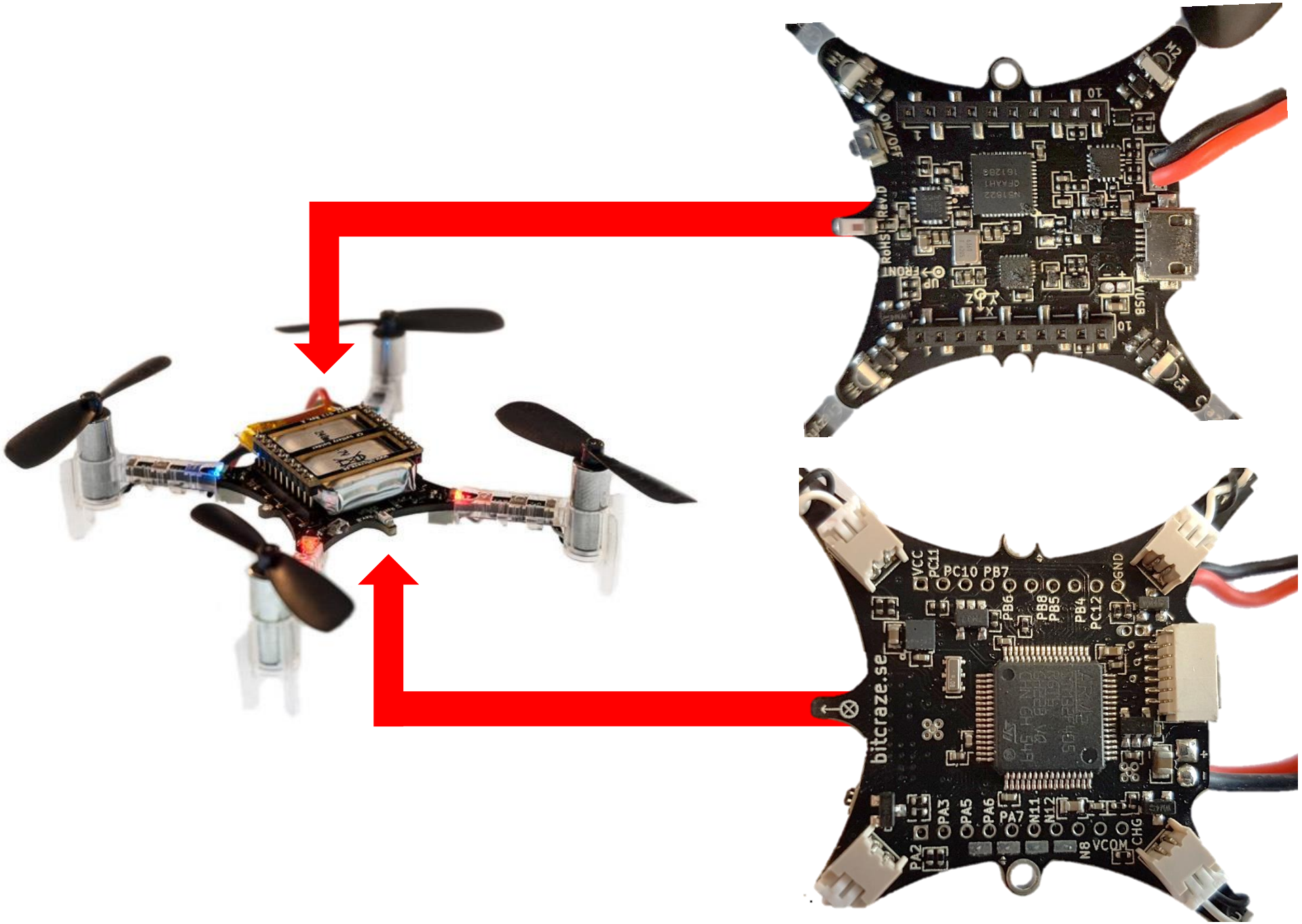
EEPROM:

- electrically erasable programmable read-only memory
- used for firmware (part of data and software that usually is not changed, configuration data)
- can not be easily overwritten in comparison to Flash

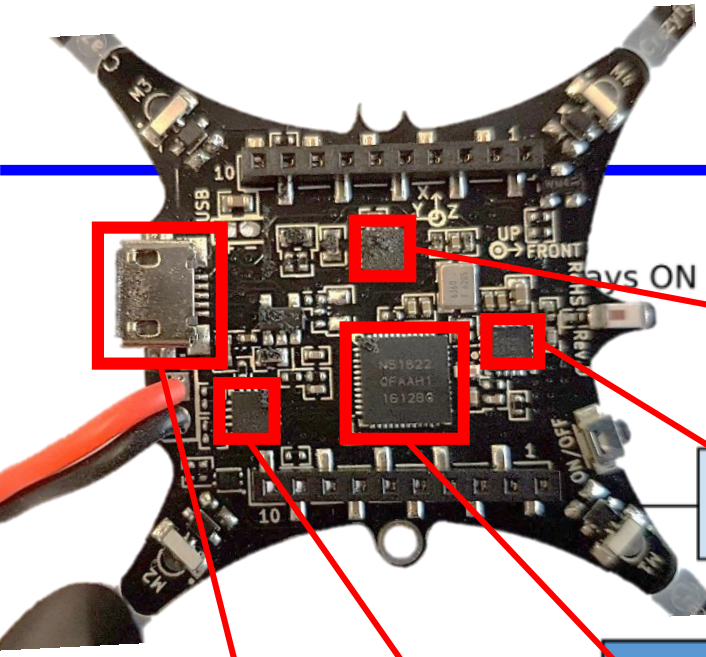


Flash memory:

- non-volatile random-access memory for program and data


ON/OFF
RoHS
REV.D
FRONT
USB
bitcraze.se
VCC
PC11
PC10
PB7
GND
PB6
PB8
PB5
PB4
PC12
PA2
PA3
PA5
PA6
PA7
CN11
CN12
N8
VCOM
CHG

High-Level Physical View



ays ON power domain

Power switched by nRF51 (VCC)

RF power amplifier

10DOF IMU
- 3-axis accelerometer
- 3-axis gyro
- 3-axis magnetomer
- Pressure sensor

I2C

Push button

nRF51822
- 16MHz Cortex-M0
- 16kB RAM, 256KB Flash
- BLE and NRF radio

UART

STM32F405
- 168MHz Cortex-M4
- 196kB RAM, 1MB Flash

PWM

Motor driver

I2C

SPI/I2C/GPIO/PWM

Wkup/OW/GPIO

Power supplies and battery charger

+5V

Charge/VBAT/VCC

Expansion port

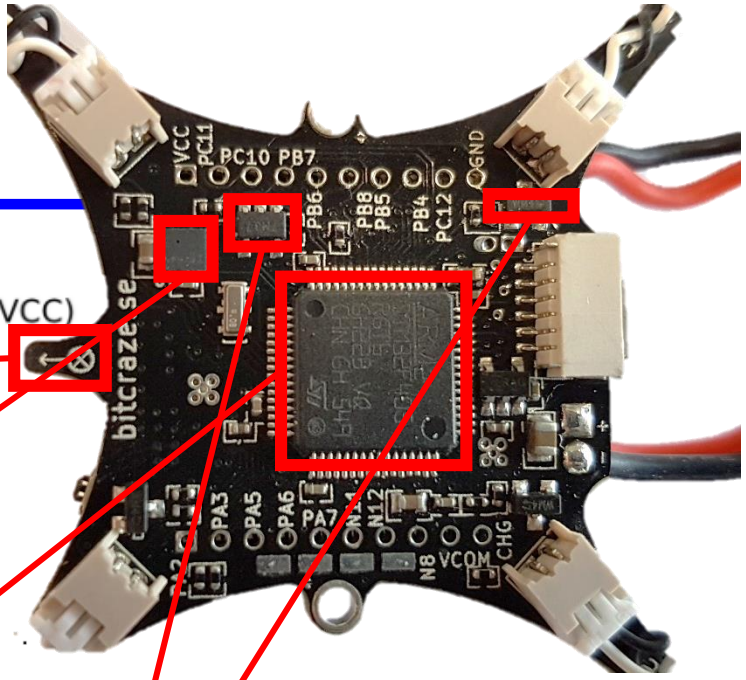
EEPROM

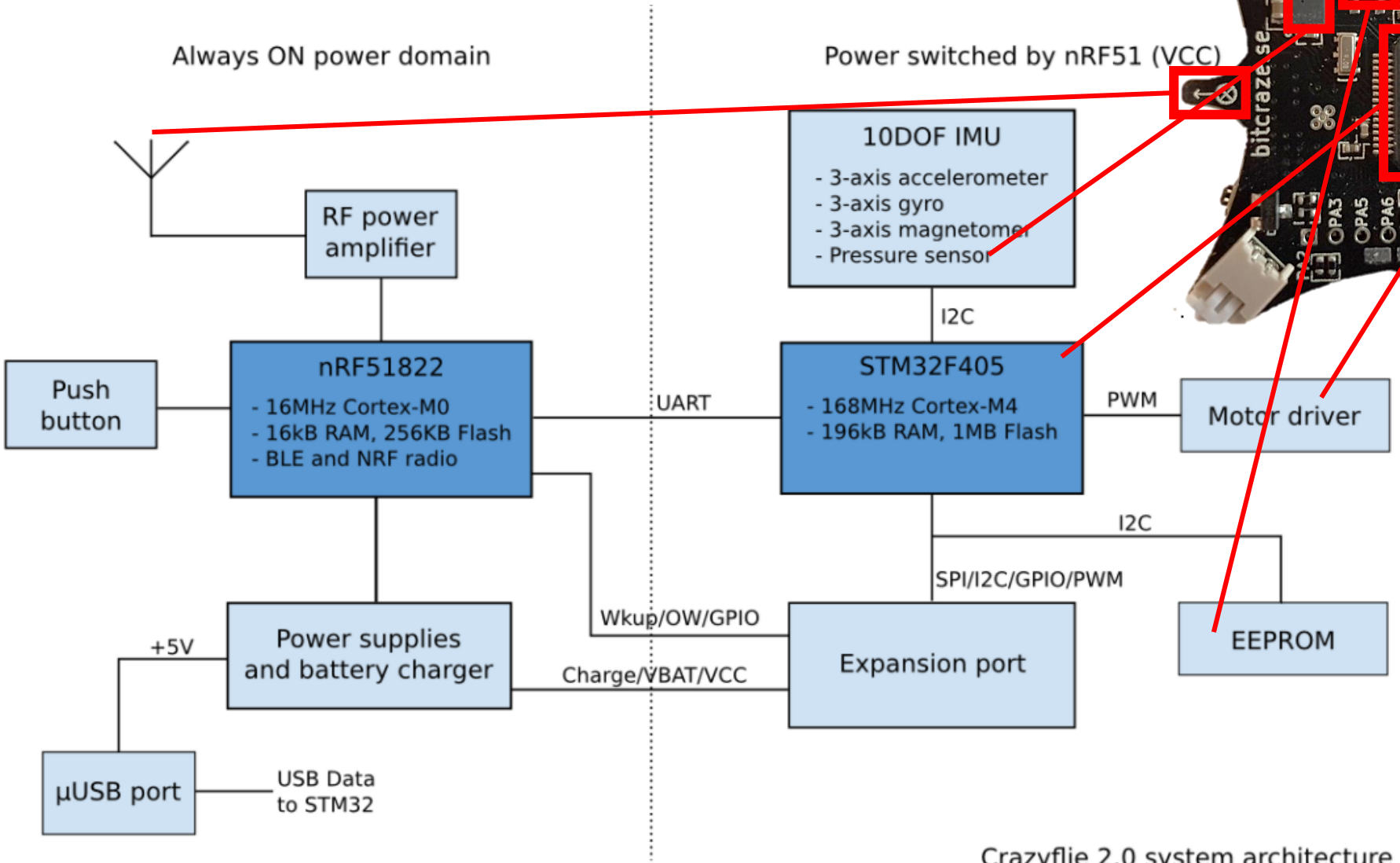
µUSB port

USB Data to STM32

Crazyflie 2.0 system architecture

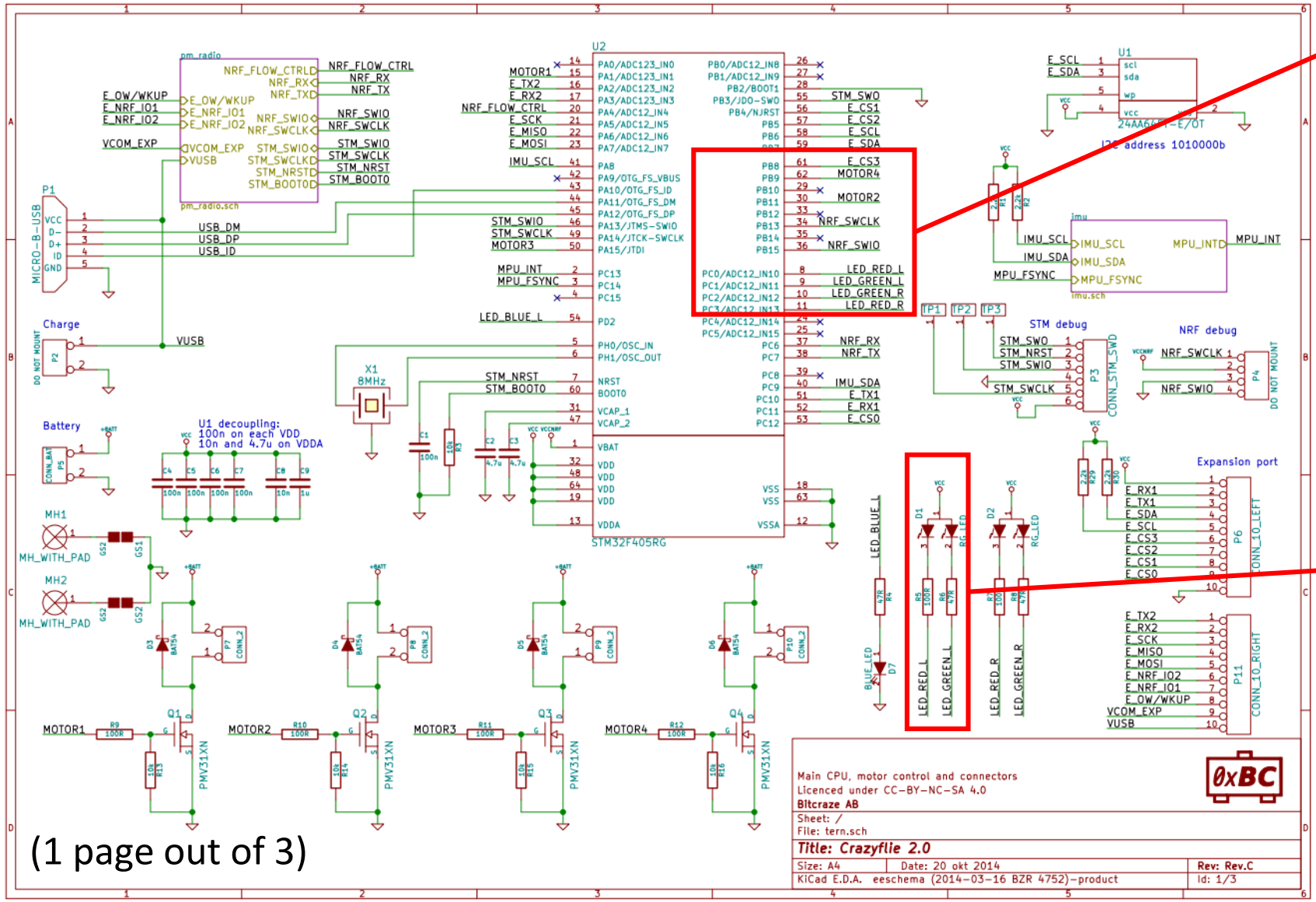
High-Level Physical View



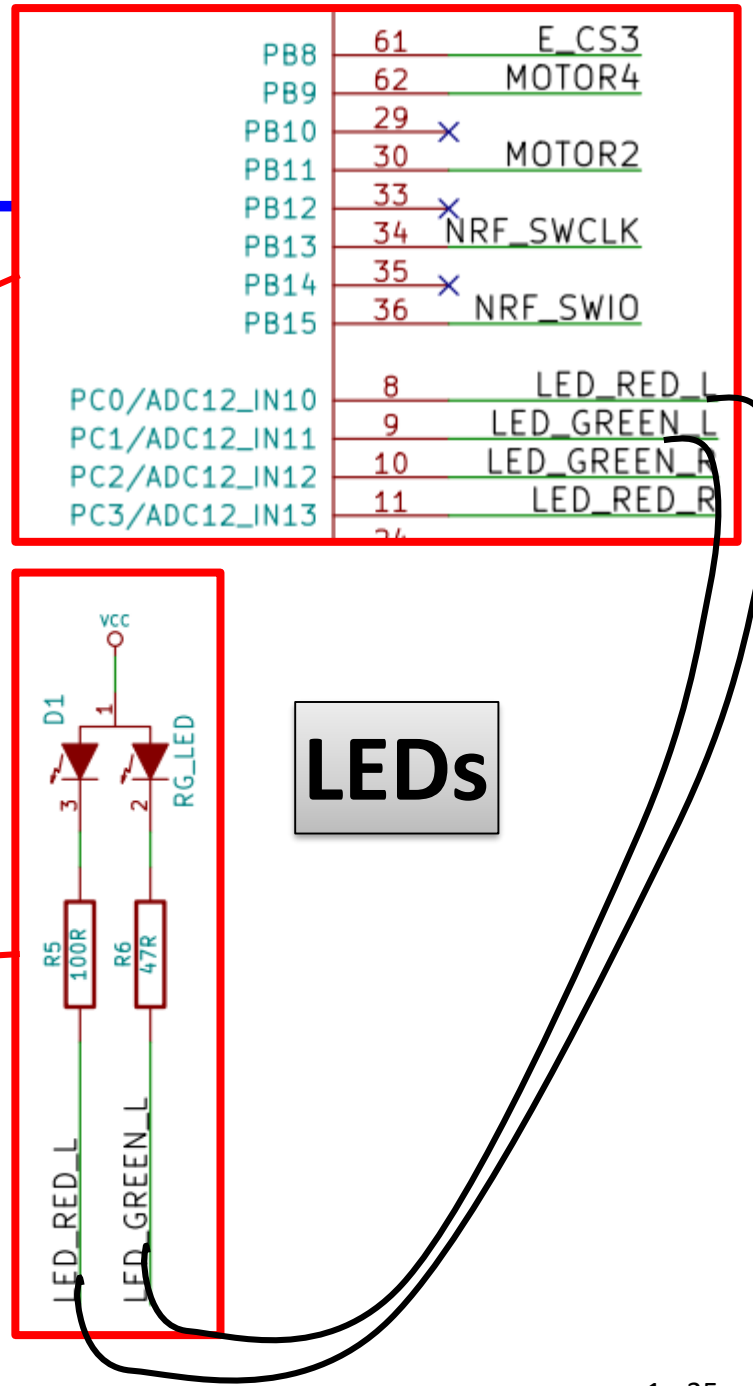


Crazyflie 2.0 system architecture

Low-Level Schematic Diagram View

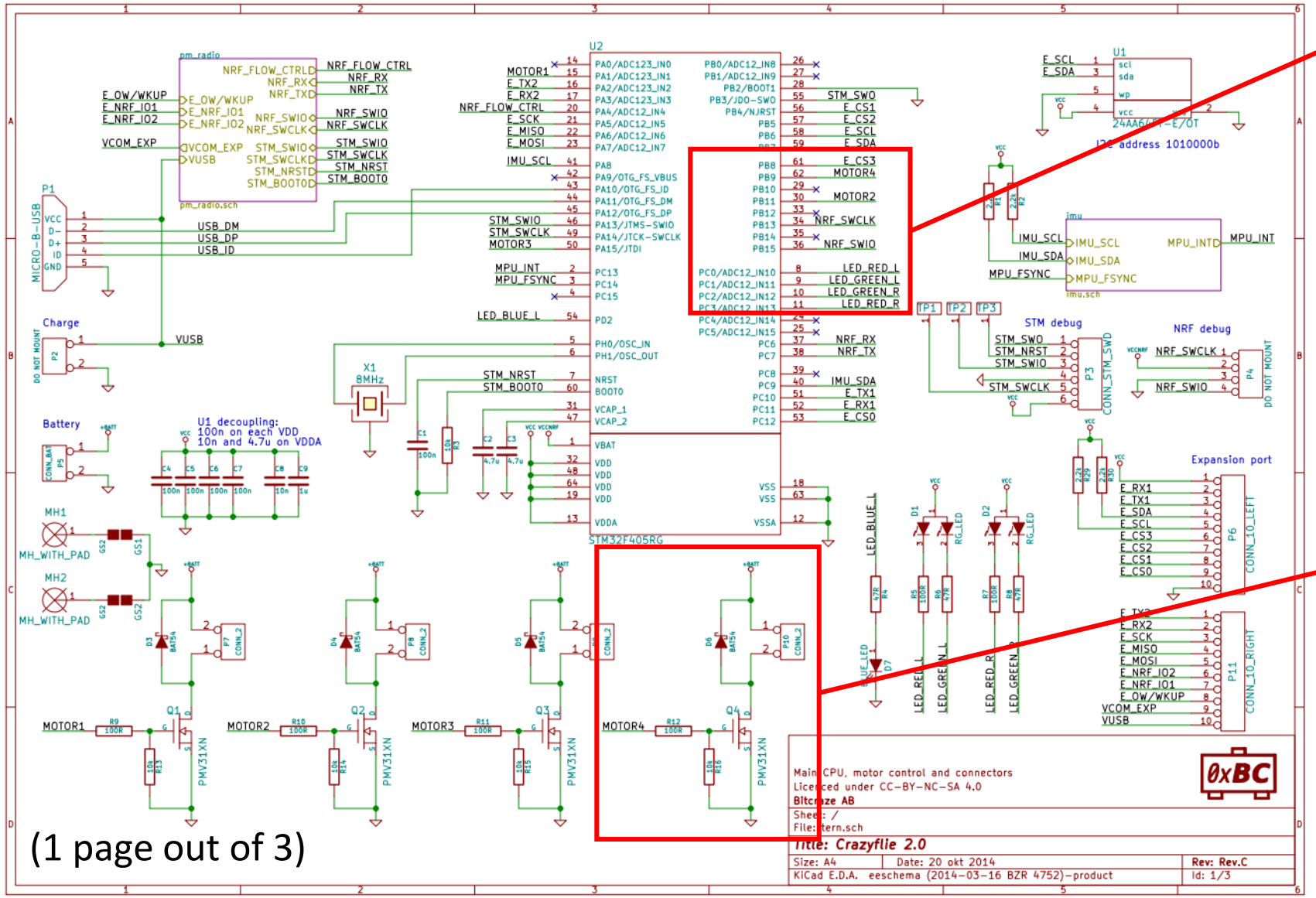


(1 page out of 3)

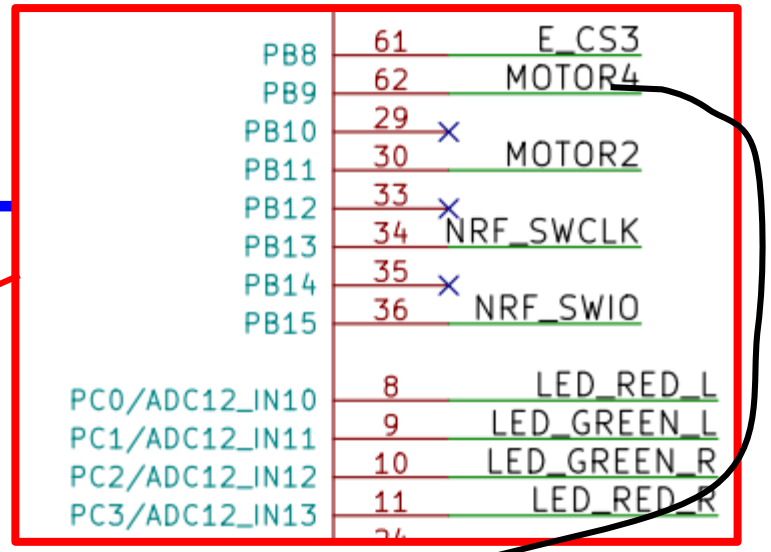


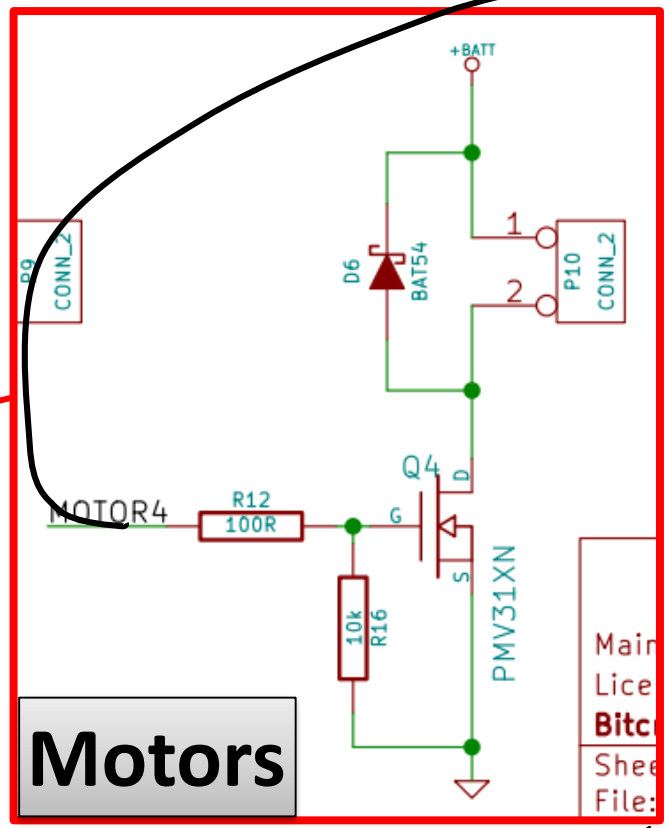
LEDs

Low-Level Schematic Diagram View



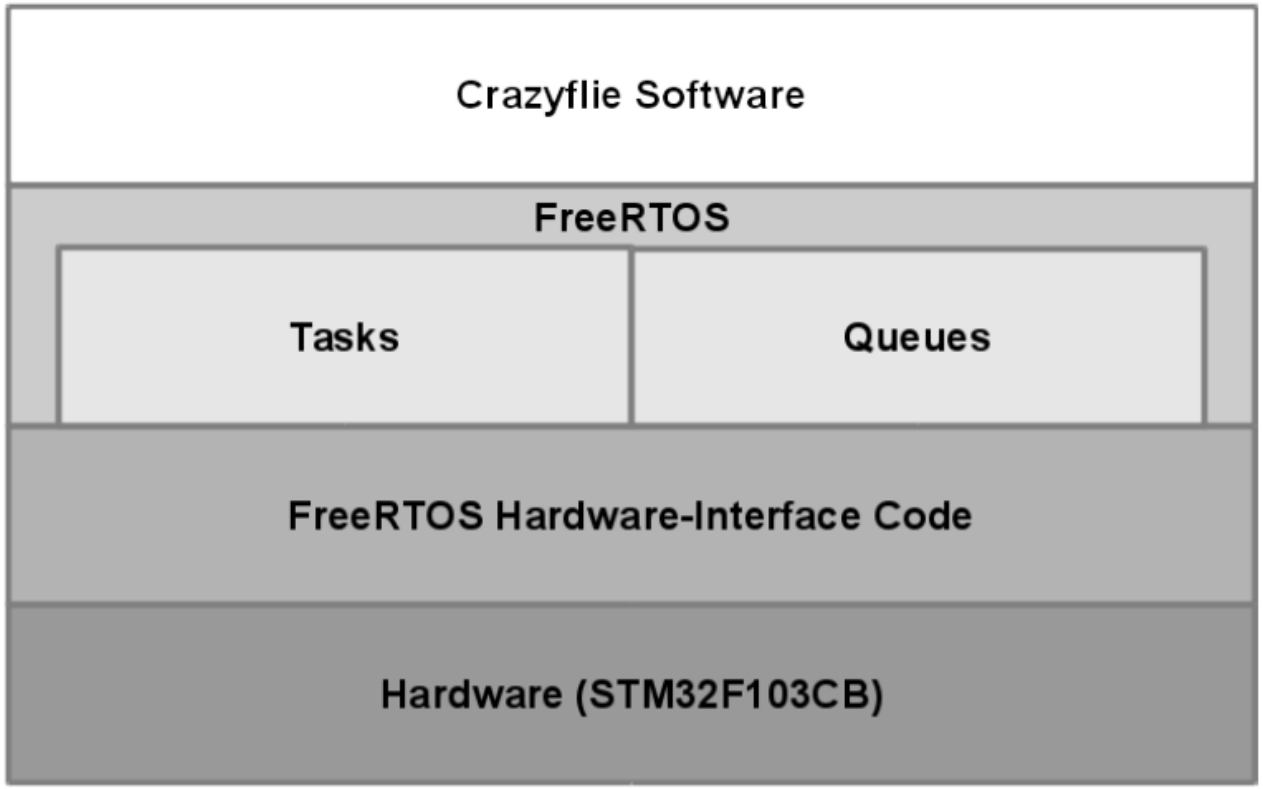
(1 page out of 3)





High-Level Software View

- The software is built on top of a *real-time operating system* “FreeRTOS”.
- We will use the same operating system in the ES-Lab … .



High-Level Software View

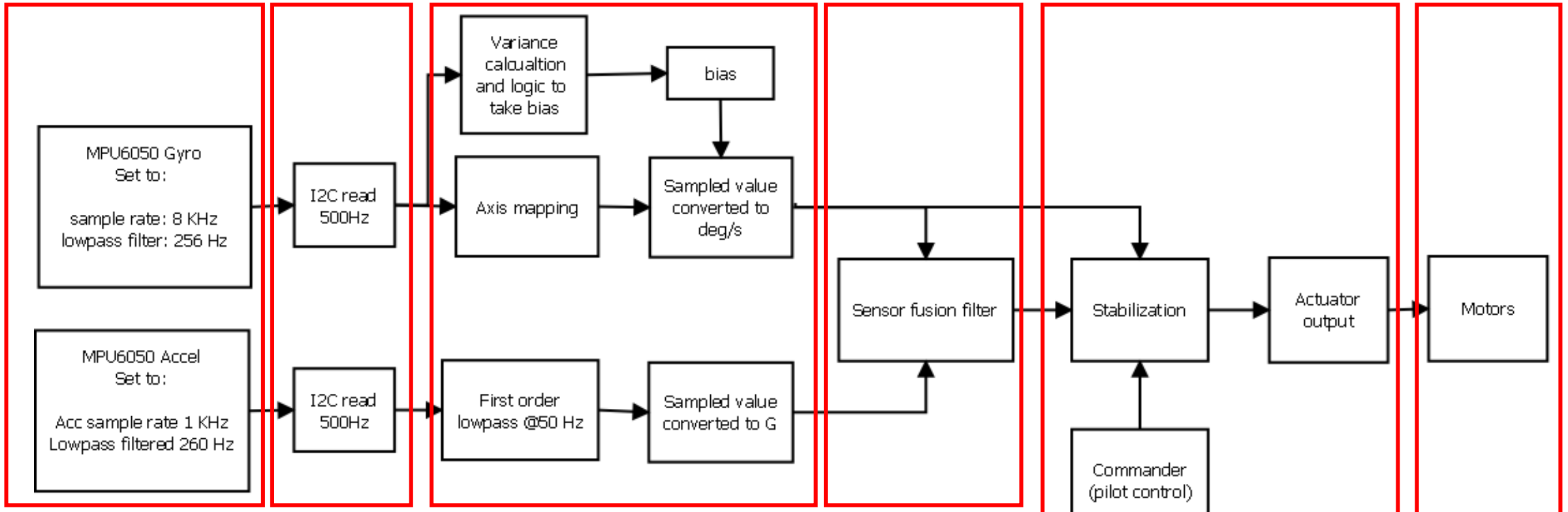
The *software architecture* supports

- *real-time tasks* for motor control (gathering sensor values and pilot commands, sensor fusion, automatic control, driving motors using PWM (pulse width modulation, …) but also

- *non-real-time tasks* (maintenance and test, handling external events, pilot commands, …).

High-Level Software View

Block diagram of the stabilization system:

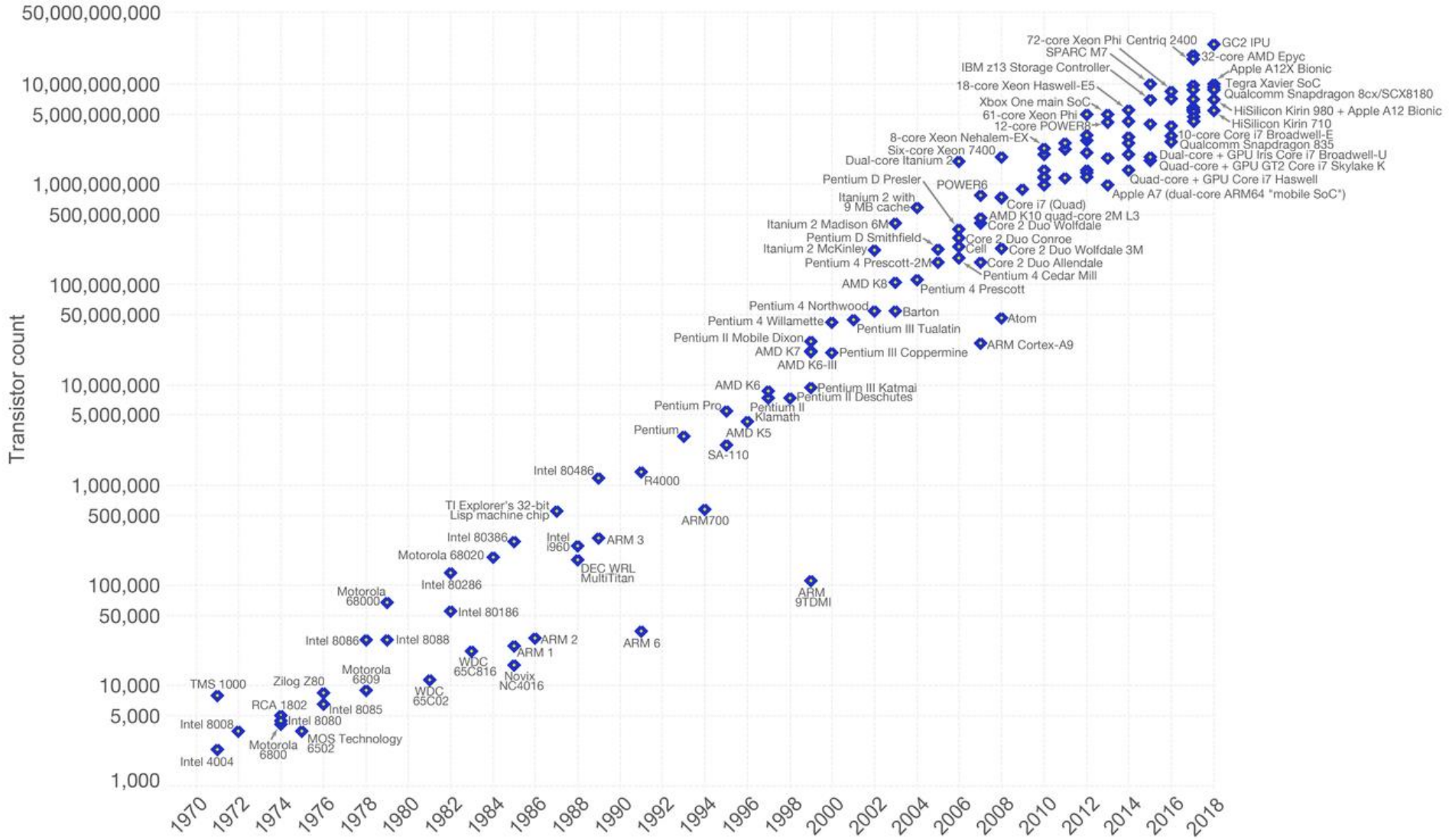


Components and Requirements by Example

- Processing Elements -

What can you do to increase performance?

From Computer Engineering

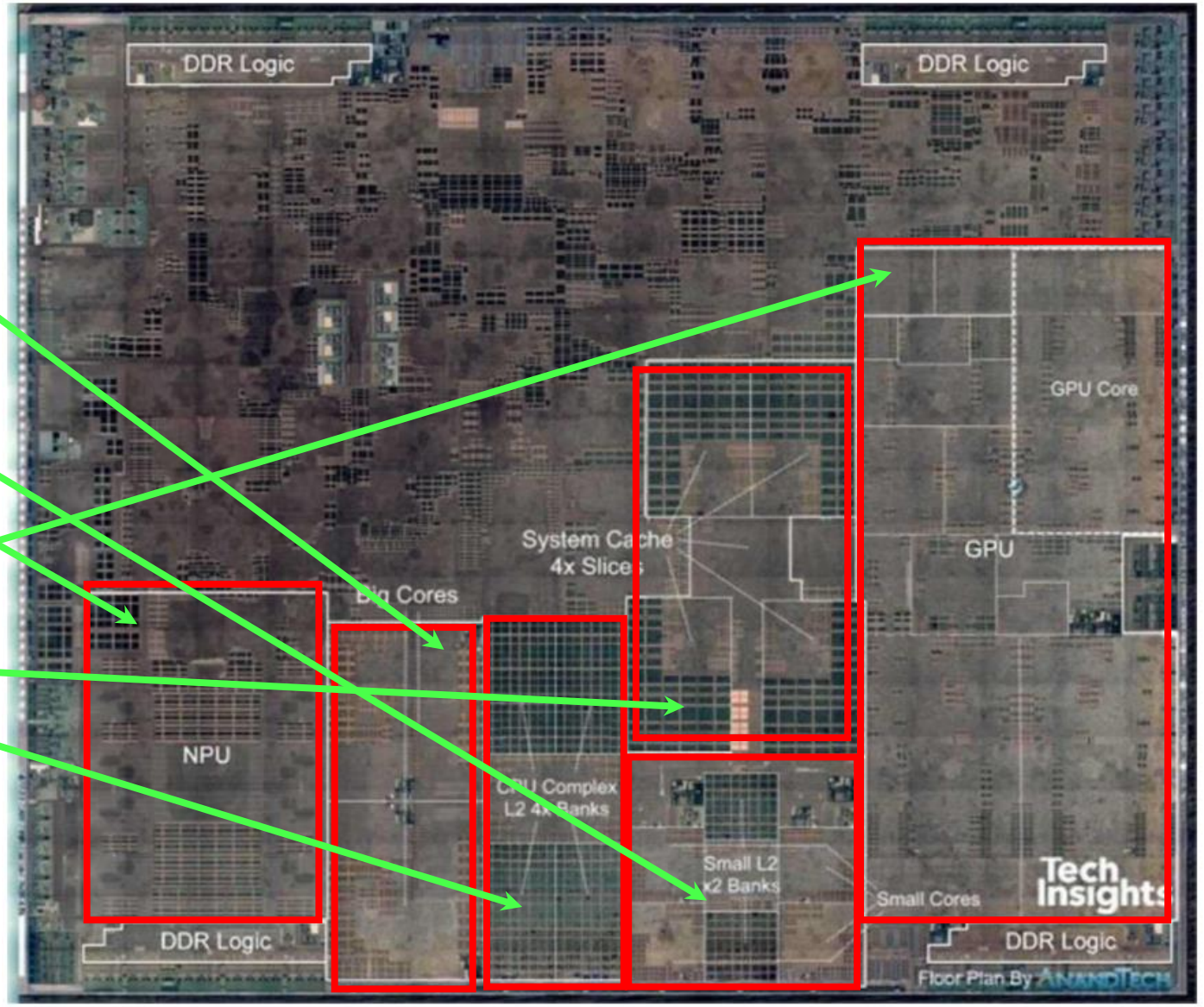


Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

From Computer Engineering

iPhone Prozessor A12

- 2 processor cores - high performance
- 4 processor cores - less performant
- Acceleration for Neural Networks
- Graphics processor
- Caches



What can you do to decrease power consumption?

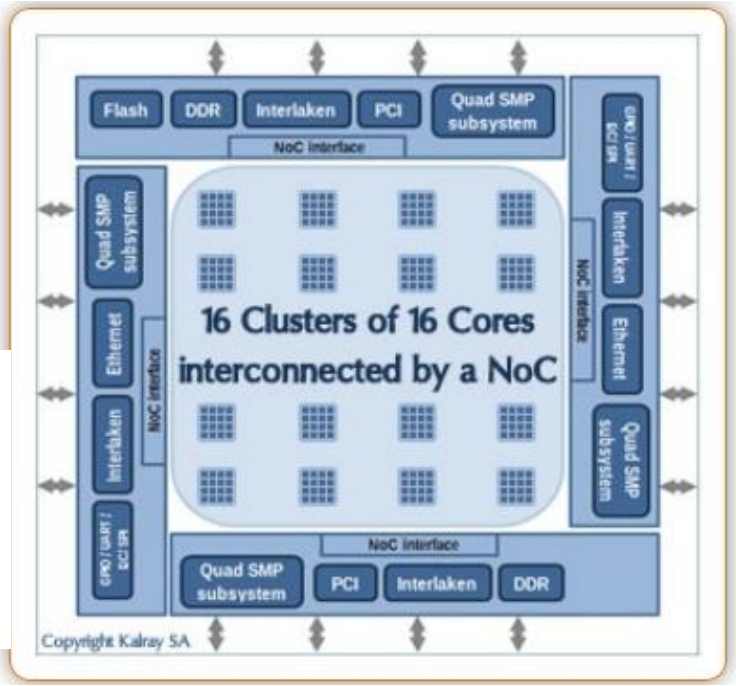
Embedded Multicore Example

Trends:

- Specialize multicore processors towards real-time processing and low power consumption (parallelism can decrease energy consumption)
- Target domains:

IMAGE & AUDIO · SIGNAL PROCESSING · DATA SECURITY · SCIENTIFIC COMPUTING · CONTROL COMMAND

Core Generation	Number of Processing Cores	GFLOPS/W	GOPS/W
Andey	256	25	75
Bostan (2014)	256	50	80
Coolidge (2015)	64/256/1024	75	115

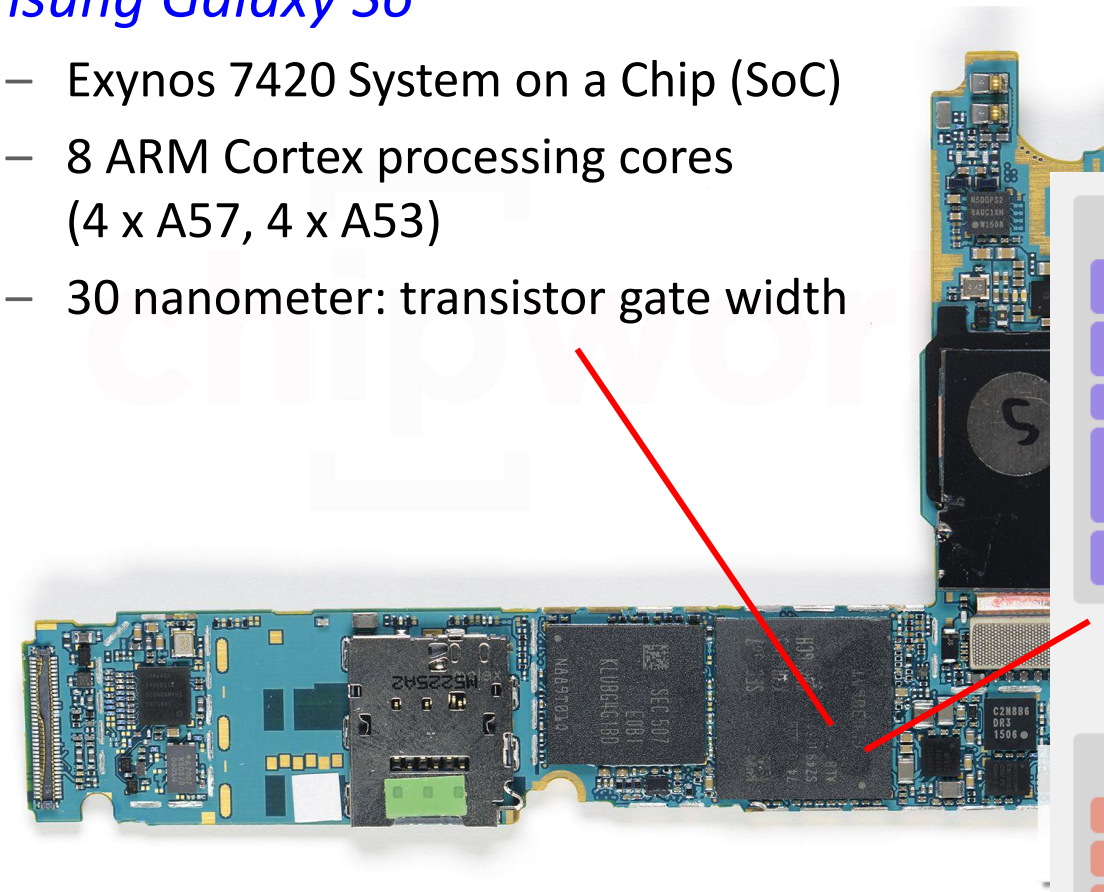


Why does higher parallelism help in reducing power?

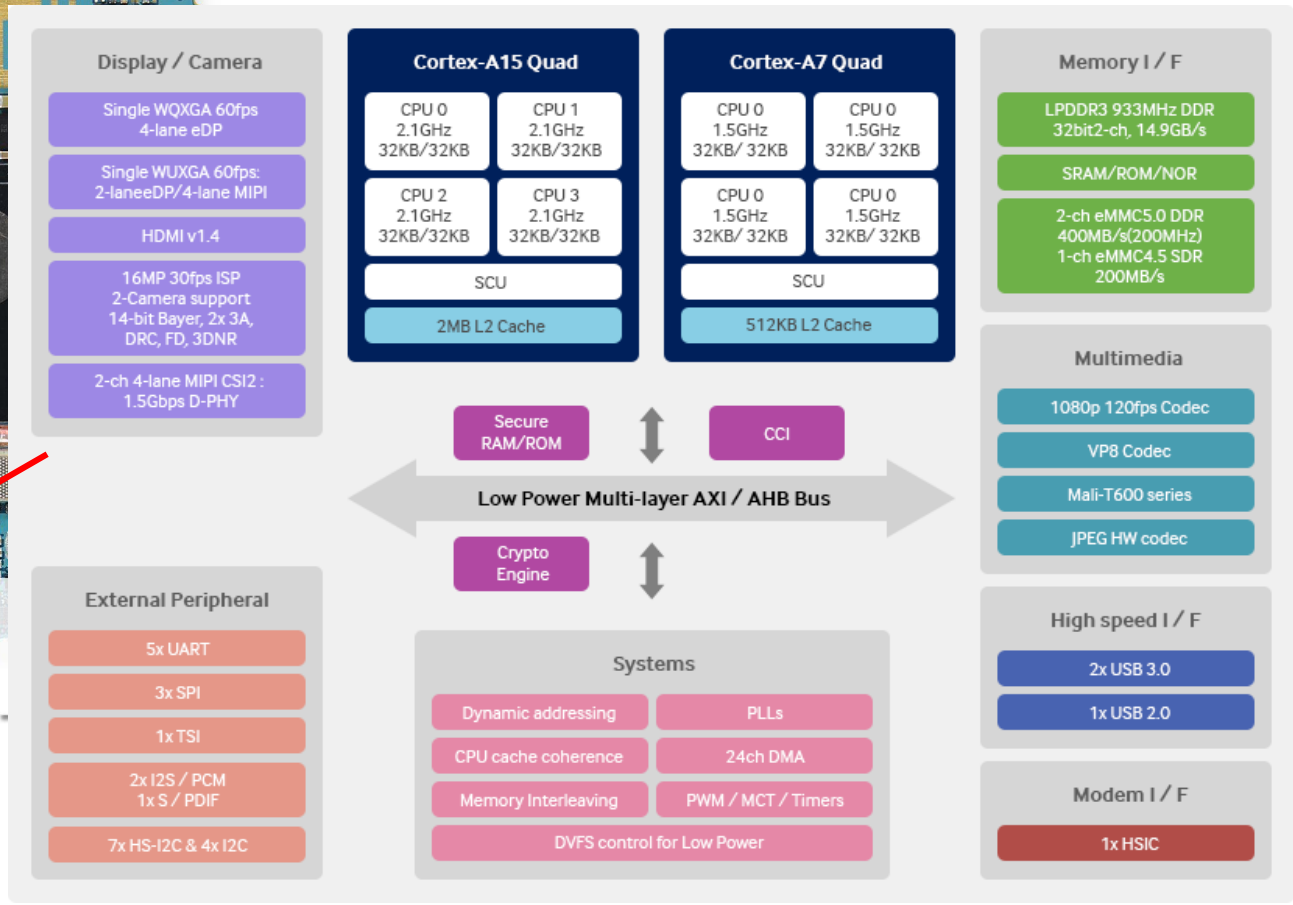
System-on-Chip

Samsung Galaxy S6

- Exynos 7420 System on a Chip (SoC)
- 8 ARM Cortex processing cores (4 x A57, 4 x A53)
- 30 nanometer: transistor gate width



Exynos 5422

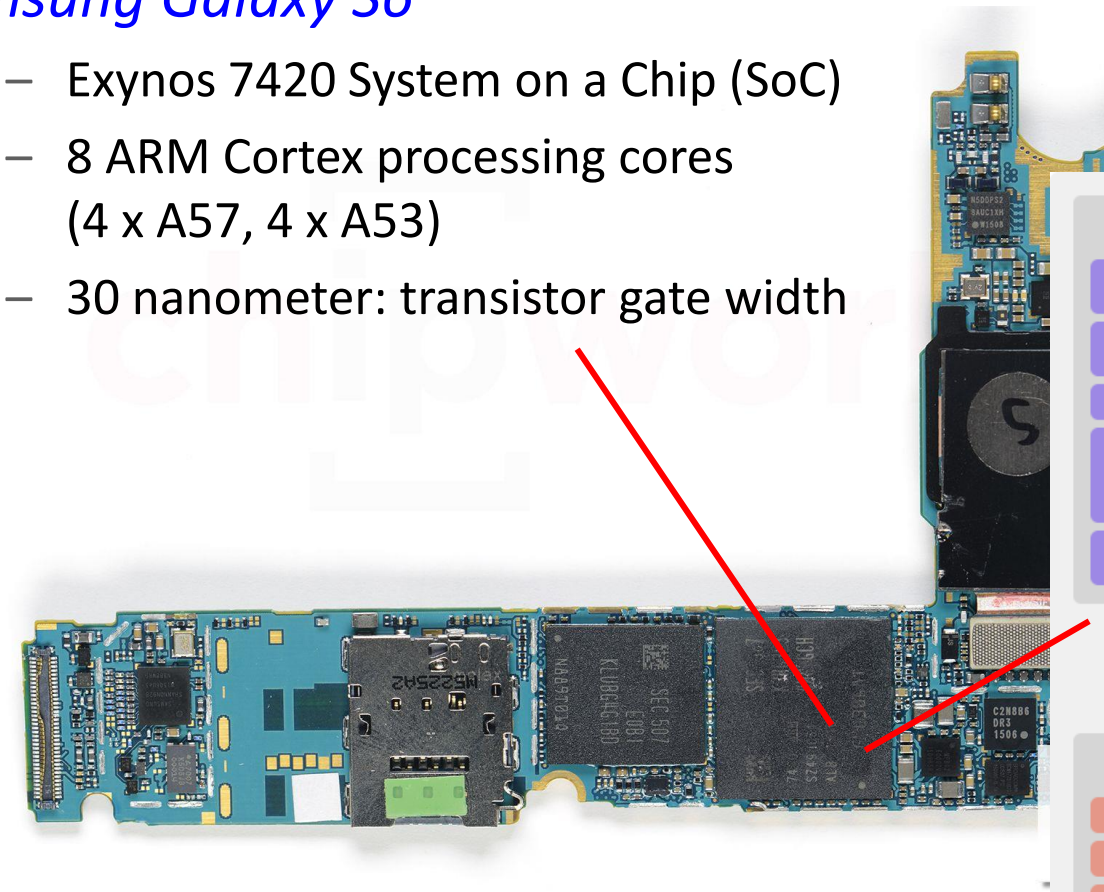


How to manage extreme workload variability?

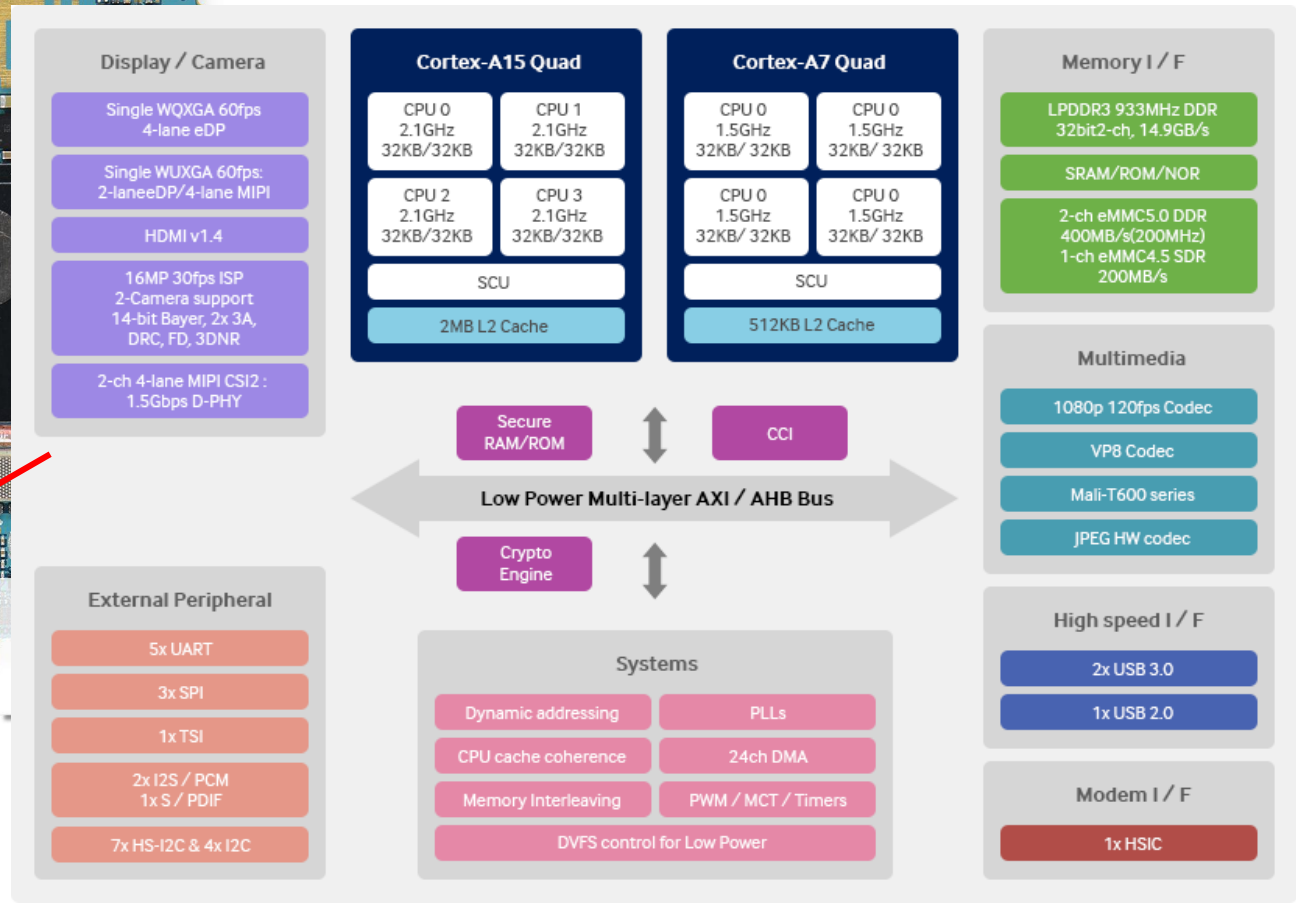
System-on-Chip

Samsung Galaxy S6

- Exynos 7420 System on a Chip (SoC)
- 8 ARM Cortex processing cores (4 x A57, 4 x A53)
- 30 nanometer: transistor gate width



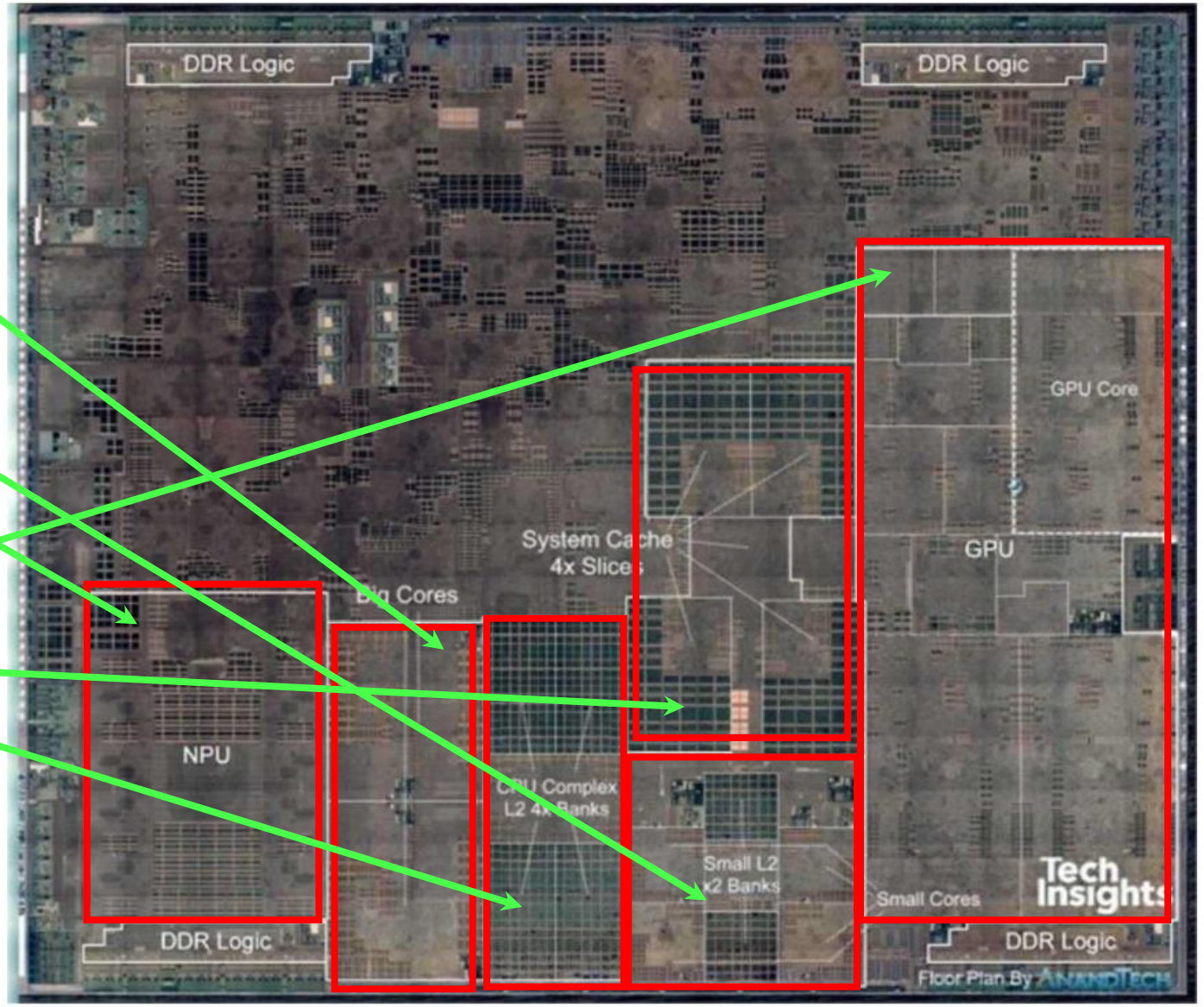
Exynos 5422



From Computer Engineering

iPhone Prozessor A12

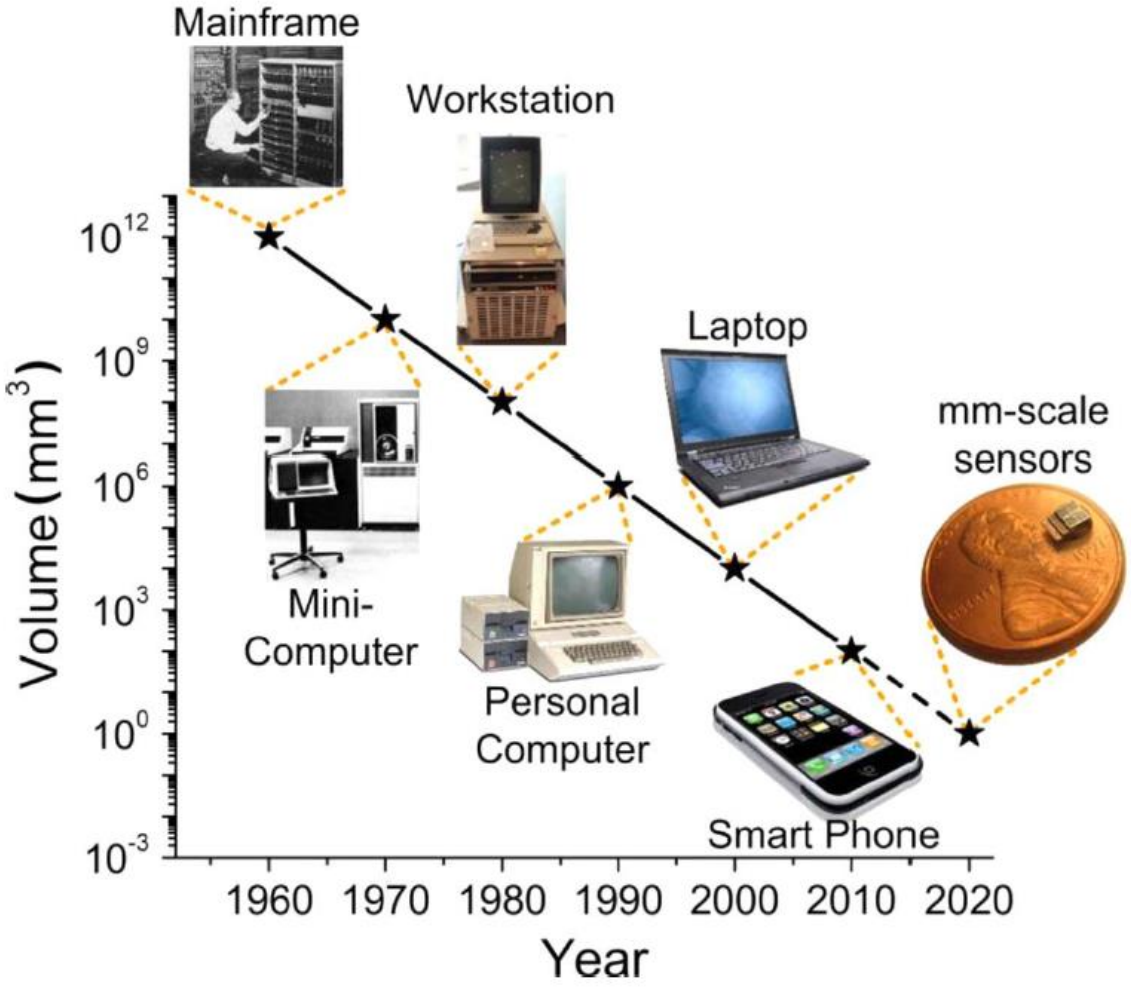
- 2 processor cores - high performance
- 4 processor cores - less performant
- Acceleration for Neural Networks
- Graphics processor
- Caches



Components and Requirements by Example

- Systems -

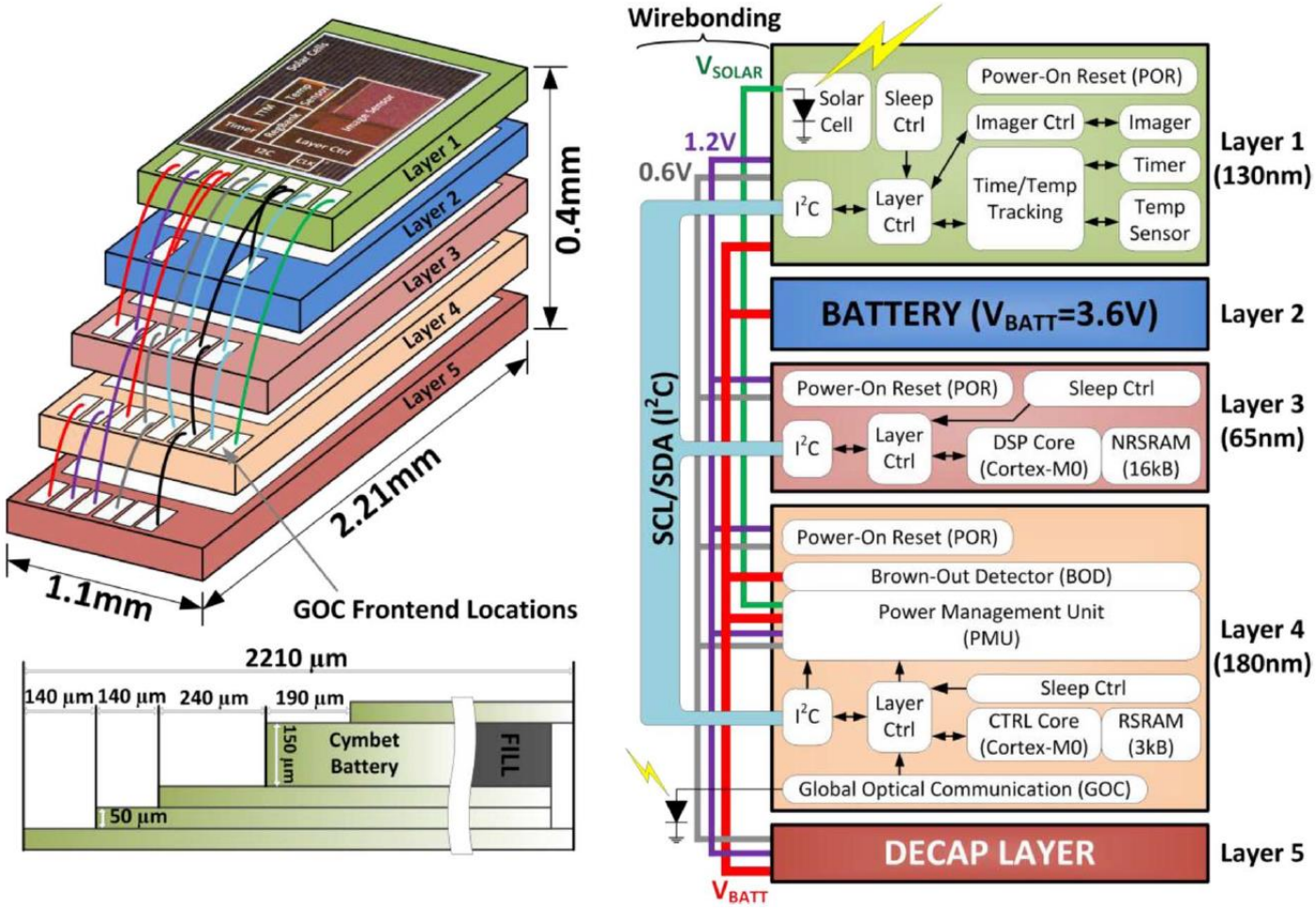
Zero Power Systems and Sensors



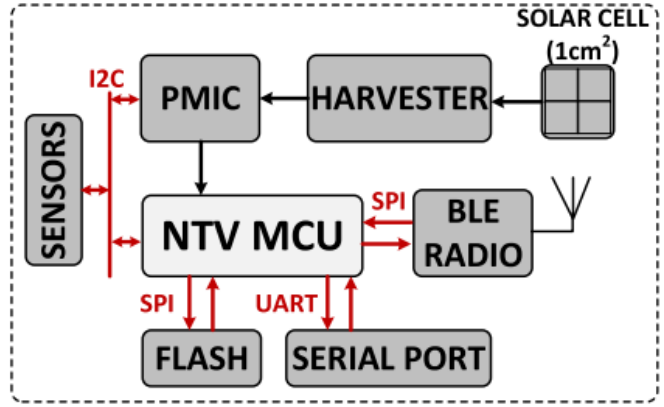
Streaming information to and from the physical world:

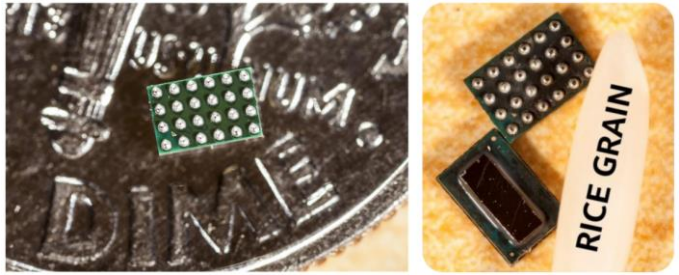
- "Smart Dust"
- Sensor Networks
- Cyber-Physical Systems
- Internet-of-Things (IoT)

Zero Power Systems and Sensors



IEEE Journal of Solid-State Circuits, Jan 2013, 229-243.





IEEE Journal of Solid-State Circuits, April 2017, 961-971.

Trends …

- *Embedded systems are communicating with each other,* with servers or with the cloud. Communication is increasingly wireless.

- *Higher degree of integration* on a single chip or integrated components:
  - Memory + processor + I/O-units + (wireless) communication.
  - Use of networks-on-chip for communication between units.
  - Use of homogeneous or heterogeneous multiprocessor systems on a chip (MPSoC).
  - Use of integrated microsystems that contain energy harvesting, energy storage, sensing, processing and communication ("zero power systems").
  - The complexity and amount of software is increasing.

- *Low power and energy constraints* (portable or unattended devices) are increasingly important, as well as temperature constraints (overheating).
- There is increasing interest in *energy harvesting* to achieve long term autonomous operation.