IUST

Iran University of
Science and Technology

Department of
Computer Engineering,
Iran University of
Science and Technology

Embedded Systems: The Future Computation Systems

Presenter:

Amir Mahdi Hosseini Monazzah

monazzah@iust.ac.ir

Invited Talk at Sharif University of Technology

1400/10/22

CPS LAB

IUST

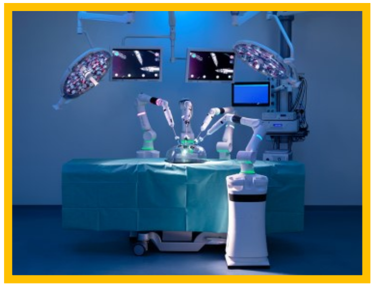
Cyber-Physical Systems Laboratory

Overview

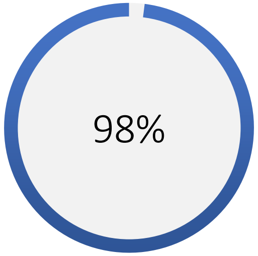
- What are ESs?
- ES components
  - Processors
  - Memories
  - Peripherals
- ES design
- Research potential
- Job potential
- Conclusion

What is an Embedded System?

- Embedded systems
  - Information processing systems embedded into a larger product
- Two types of computing
  - Desktop – produced millions/year
  - Embedded – billions/year
- Non-Embedded systems
  - PCs, servers, and notebooks
- The future of computing!
  - Automobiles, entertainment, communication, aviation, handheld devices, military and medical equipments

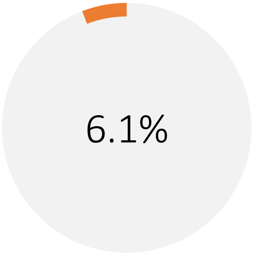


Why Embedded System is Important?



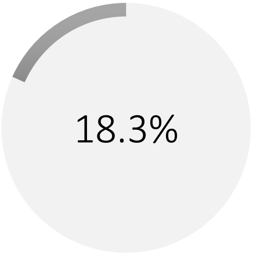
Widespread

Ninety-eight percent of all microprocessors manufactured are utilized in embedded systems [1].



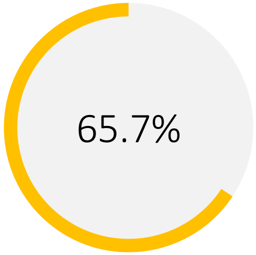
CAGR

The embedded system market size is expected to reach USD 116.2 billion by 2025 [2]!



Automotive Inds.

The automotive industry is likely to account for a promising share of 18.3% of the overall market by 2021 [3].

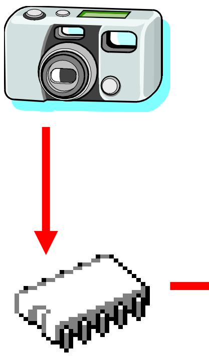


Relation to IoT

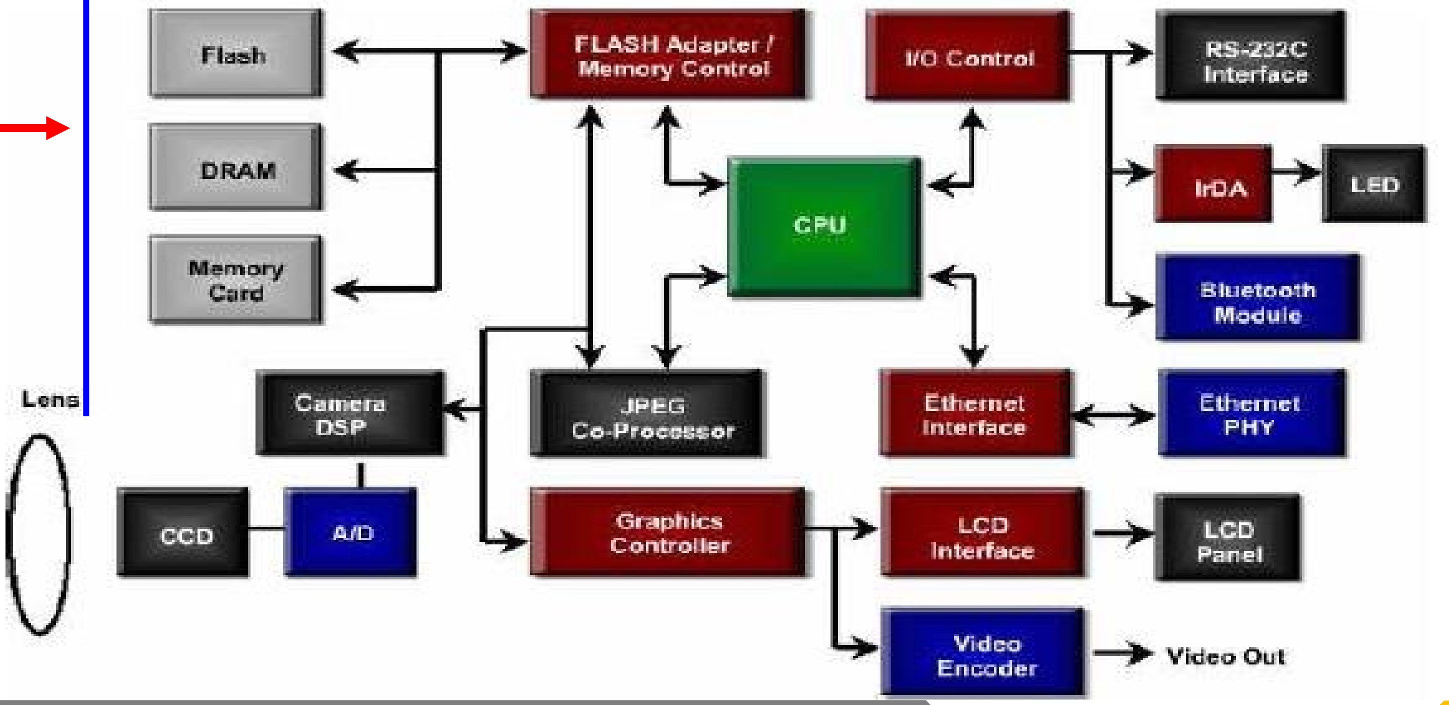
In future projects with over 65% indicating that they will have one or more projects devoted to IoT [4].

Example

- An example ES!

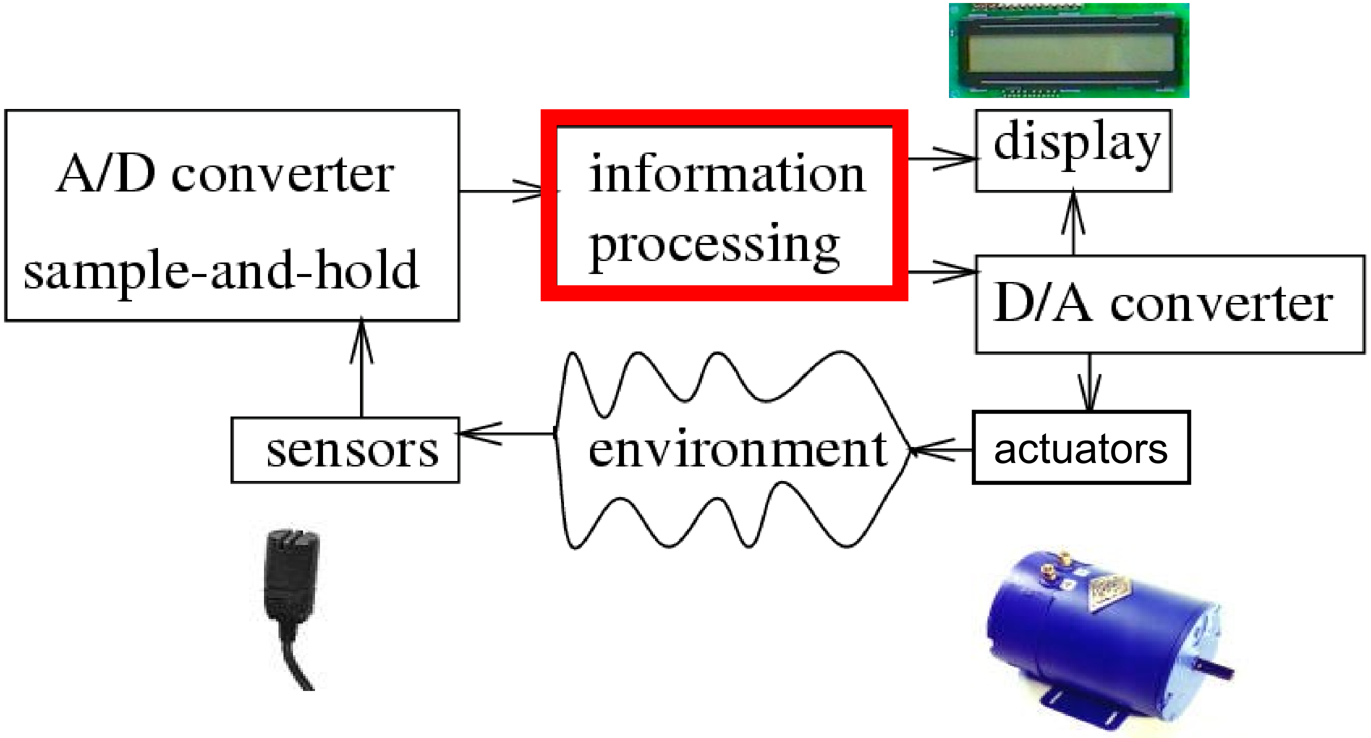


Digital Camera Block Diagram



Big Picture

- Simplified block diagram of an ES



What are Embedded System’s Components?

- Analog Components
  - Sensors, Actuators, Controllers, …
- Digital Components
  - Processor, Coprocessors
  - Memories
  - Controllers, Buses
- Converters – A2D, D2A, …
- Software
  - Application Programs
  - Exception Handlers

Hardware

Software

Processors (Microprocessors)

- Execute programs
  - Serial execution of instructions
  - Simple, universal
- Instruction execution engine: fetch/execute cycle
  - Flow of control determined by modifications to program counter
  - Instruction classes:
    - Data: move, arithmetic and logical operations
    - Control: branch, loop, subroutine call
    - Interface: load, store from external memory

Processors (Microprocessors)

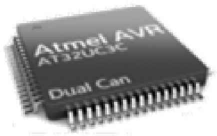
- Traditional architecture goal: Performance
  - Caches
  - Branch prediction
  - Multiple/Out of Order (OoO) issue

Embedded Processors (Microcontrollers)

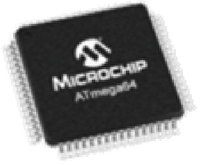
- Processor optimized for low cost
  - No cache
  - Small memory
  - No disks
  - 4 bit/8 bit/16 bit
  - No FP
  - No complicated datapath
  - Multicycle instruction interpretation
  - Simple/no operating system
  - Programs are static

8Bit
Microcontroller

16Bit
Microcontroller



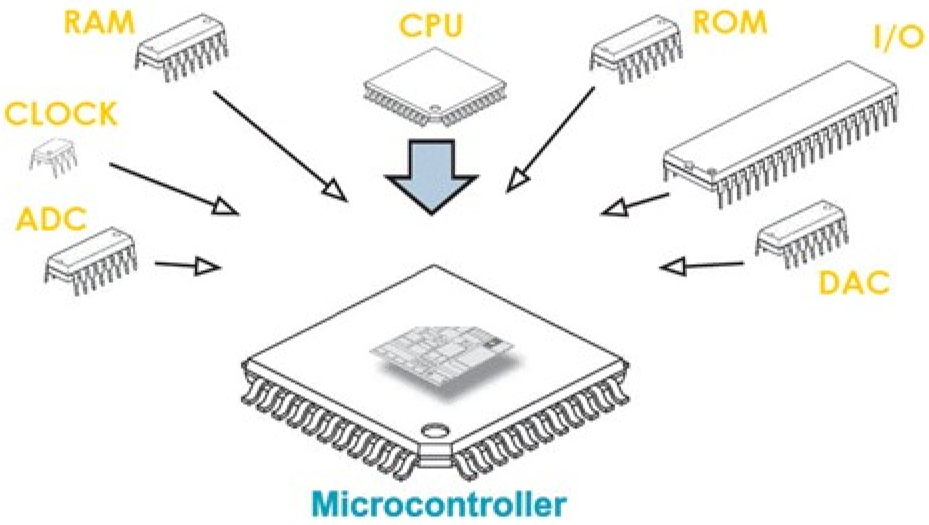
32Bit
Microcontroller



64Bit
Microcontroller

Embedded Processors (Microcontrollers)

- Low performance
  - 1 MIPS is enough if 1 ms is the time scale
- Integrate on a single chip



Microprocessors vs. Microcontrollers

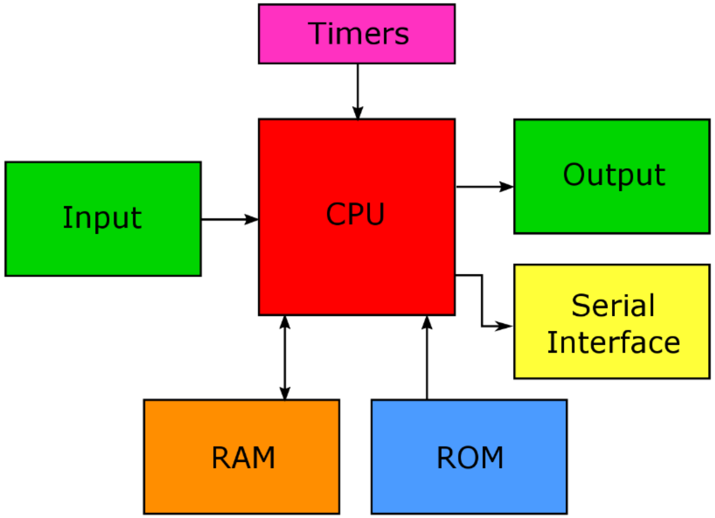
- Microprocessors
  - Programmed by user
  - New applications are developed routinely
  - General-purpose
  - Must handle a wide ranging variety of applications
  - Interacts with environment through memory
  - All devices communicate through memory
  - DMA operations between disk and I/O devices
  - Dual-ported memory (as for display screen)
  - Oblivious to passage of time (takes all the time it needs)

Microprocessors vs. Microcontrollers

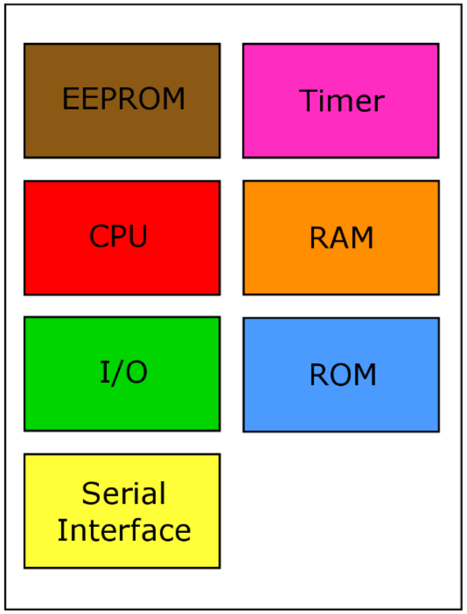
- Microcontrollers
  - Programmed once by manufacturer of system
  - Executes a single program (or a limited suite) with few parameters
  - Task-specific
  - Can be optimized for specific application
  - Interacts with environment in many ways
  - Direct sensing and control of signal wires
  - Communication protocols to environment and other devices
  - Real-time interactions and constraints

Microprocessors vs. Microcontrollers

Microprocesser: CPU
and several supporting chips.

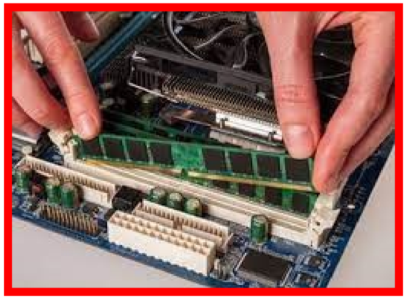


Microcontroller: CPU
on a single chip.

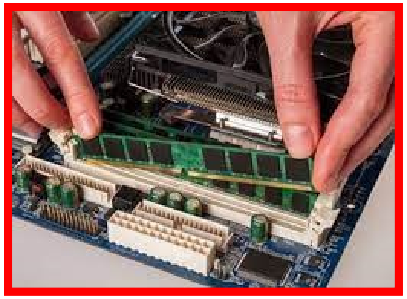


Memories

- What is a memory?
  - Artifact that stores bits
  - Storage fabric and access logic
- Write-ability
  - Manner and speed a memory can be written
- Storage-permanence
  - Ability of memory to hold stored bits after they are written
- Many different types of memories
  - Flash, SRAM, DRAM, STT-MRAM, etc.

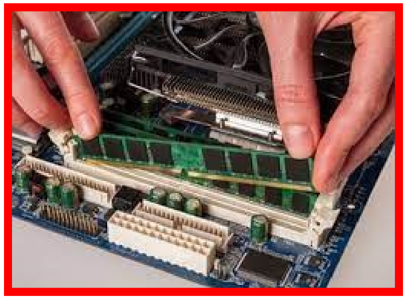


Memories



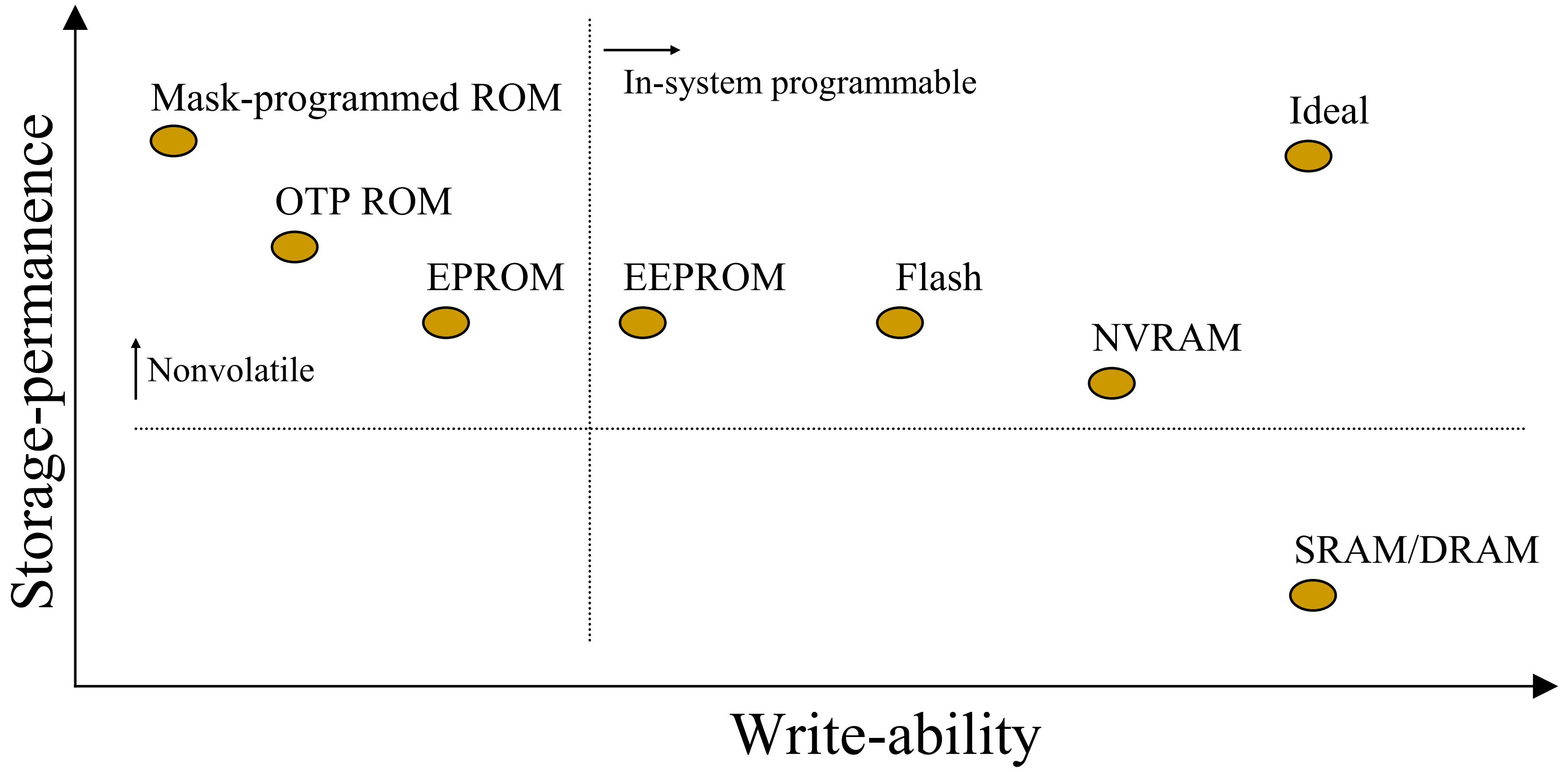
- Ranges of write-ability
  - High end
    - Processor writes to memory simply and quickly
    - E.g., RAM
  - Middle range
    - Processor writes to memory, but slower
    - E.g., FLASH, EEPROM
  - Lower range
    - Special equipment, “programmer”, must be used to write to memory
    - E.g., EPROM, OTP ROM
  - Low end
    - Bits stored only during fabrication
    - E.g., Mask-programmed ROM

Memories



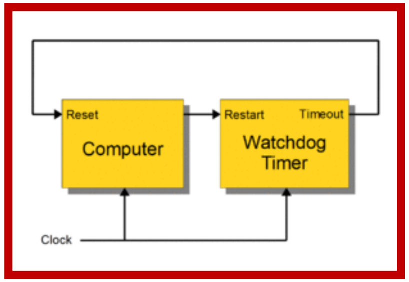
- Range of storage-permanence
  - High end
    - Essentially never loses bits
    - E.g., mask-programmed ROM
  - Middle range
    - Holds bits days/months/years after memory's power source turned off
    - E.g., NVRAM
  - Lower range
    - Holds bits as long as power supplied to memory
    - E.g., SRAM
  - Low end
    - Begins to lose bits almost immediately after written
    - E.g., DRAM

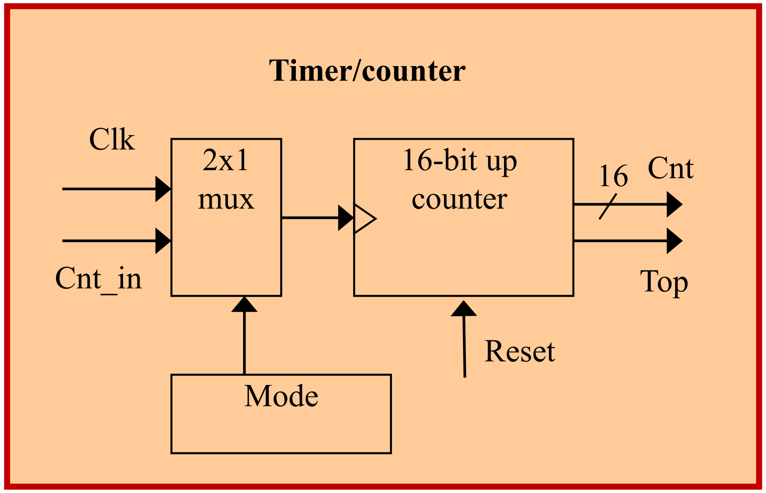
Memories


Storage-permanence
Write-ability
In-system programmable
Mask-programmed ROM
OTP ROM
EPROM
EEPROM
Flash
NVRAM
Ideal
Nonvolatile
SRAM/DRAM

Peripherals

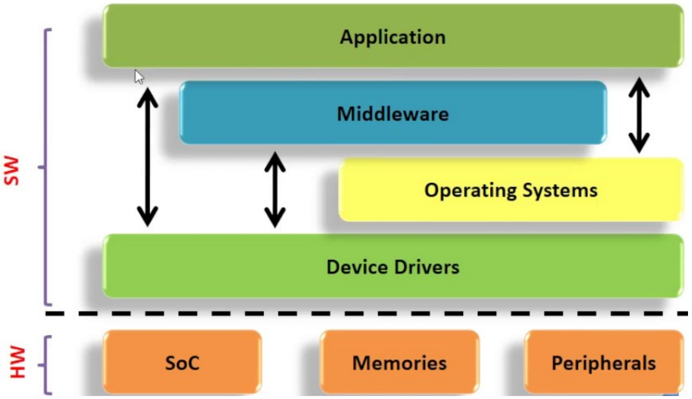
- Perform specific computation task
- Custom single-purpose processors
  - Designed by us for a unique task
- Standard single-purpose processors
  - “Off-the-shelf”
  - Pre-designed for a common task





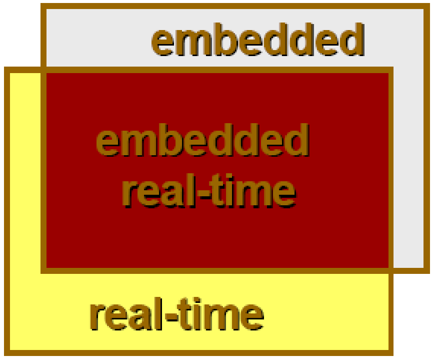
Software

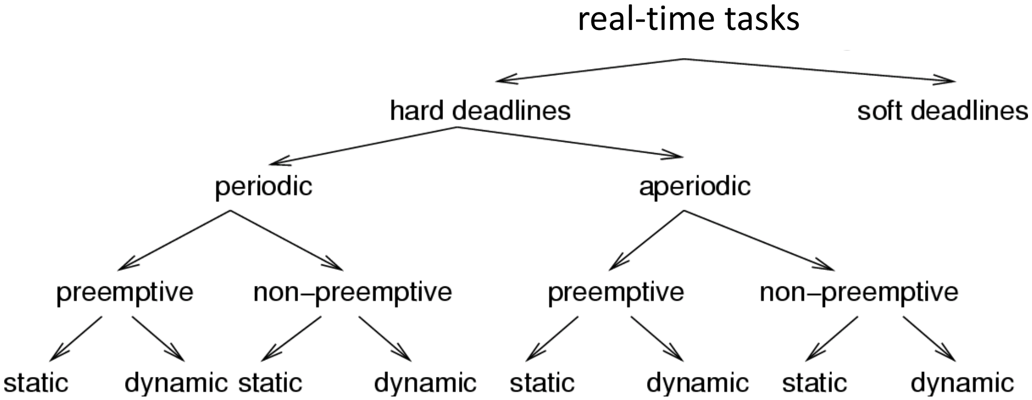
- ES softwares can be categorized in
  - Operating systems
  - Middleware
  - Real-time data bases
  - Standard software (MPEG-x, GSM-kernel, …)
  - Device drivers



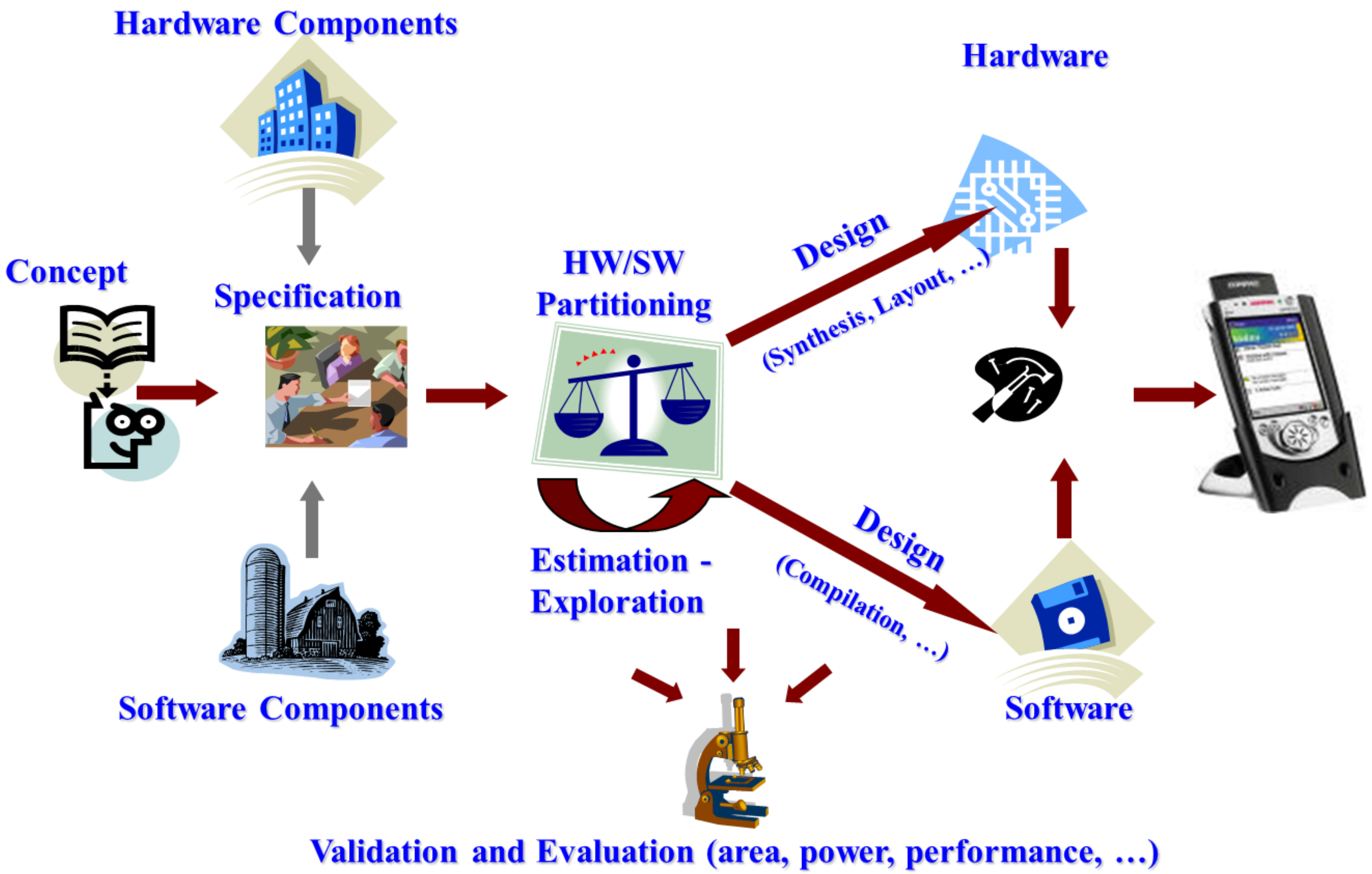
Software

- Most of embedded systems perform real-time tasks
  - Real-time system means that the system is subjected to real-time
    - Response should be guaranteed within a specified timing constraint
    - System should meet the specified deadline





Design Flow

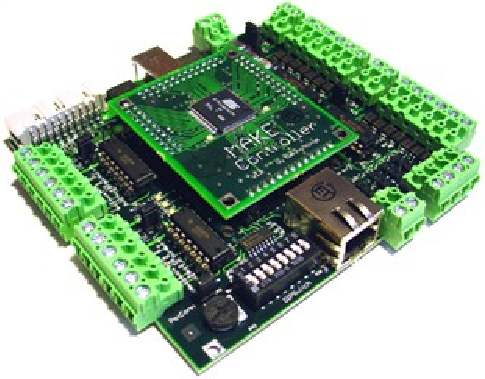

Hardware Components
Concept
Specification
HW/SW Partitioning
Design
(Synthesis, Layout, …)
Hardware
Estimation - Exploration
Design
(Compilation, …)
Software
Software Components
Validation and Evaluation (area, power, performance, …)

Classical Method

- Developing an application via microcontroller
  - Software development
    - We can use different IDEs exist for different microcontrollers like codevision, keil
  - Hardware development
    - Simulation
      - We can use some simulators like Thinkercad (online) or Proteus
    - Implementation
      - We can use different microcontrollers from different brands

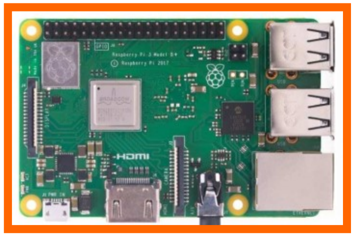
Recent Method

- Developing an application via embedded development board
- What is an embedded development board?
  - A microcontroller built onto a single printed circuit board.
  - Provides all of the circuitry necessary for a useful control task
    - A microprocessor
    - I/O circuits
    - A clock generator
    - RAM
    - Stored program memory
    - Any necessary support ICs

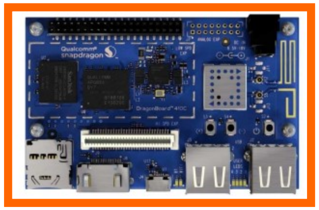


ES Development Boards

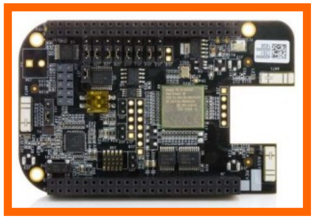
- Raspberry Pi 3 B+



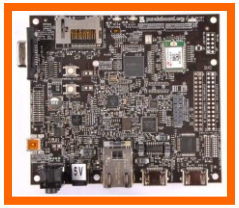
- Qualcomm Snapdragon



- BeagleBone Black



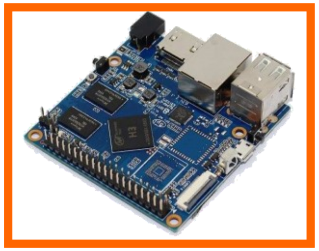
- PandaBoard



- Intel Galileo Gen 2

- Arduino Mega 2560

- Banana Pi M2+



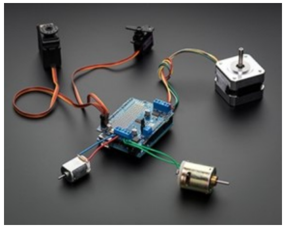
- …

ARDUINO

Raspberry Pi

Shields

- Peripherals for ES development boards
  - They are ES themselves!
    - TFT touch Screen
    - Data logger
    - Motor/Servo shield
    - Ethernet shield
    - Audio wave shield
    - Cellular/GSM shield
    - WiFi shield
    - ...many more



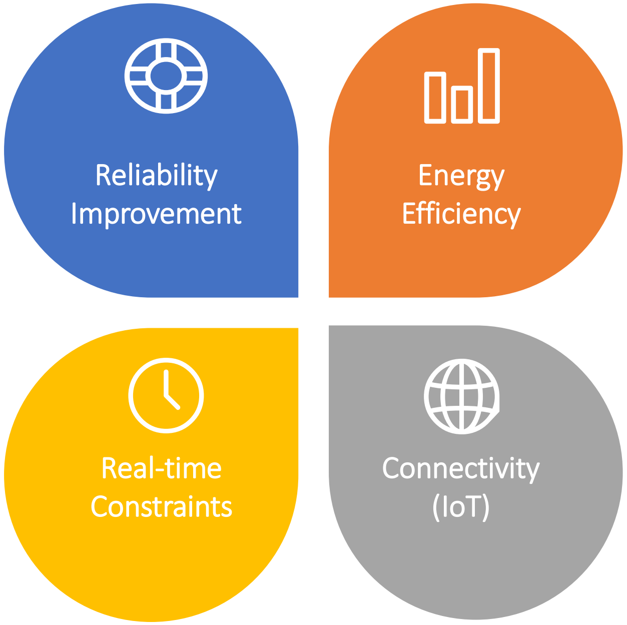
ES Characteristics

- Dependability
  - Reliability, Maintainability, Availability, Safety, Security
- Energy efficiency
- Performance
- Real-time constraints
  - For real-time systems, right answers arriving too late are wrong.
- Weight efficient, cost efficient, code-size efficient
- Dedicated towards a certain application
  - Minimize resources, maximize robustness

ES Characteristics

- Dedicated user interface
  - No mouse, keyboard and screen
- Frequently connected to physical environment through sensors and actuators.
- Hybrid systems (analog + digital parts).
- Not every ES has all of the above characteristics

Some Popular ES Research Areas



Reliability Improvement

Improving the probability of ES correct functionality during time [0,t] if it had correct functionality at time 0.

Energy Efficiency

Improving the energy usage of ES so as we don't need to replace its battery frequently.

Some Popular ES Research Areas

Reliability Improvement
Energy Efficiency
Real-time Constraints
Connectivity (IoT)

Real-time Constraints

Trying to guarantee the task's deadlines in an ES through appropriate resource allocations.

Connectivity (IoT)

Improving packet delivery ratio of ES connected to each other in a Low-power Lossy Network (LLN) infrastructures.

Our Research Profile: Interdisciplinary Area I

01

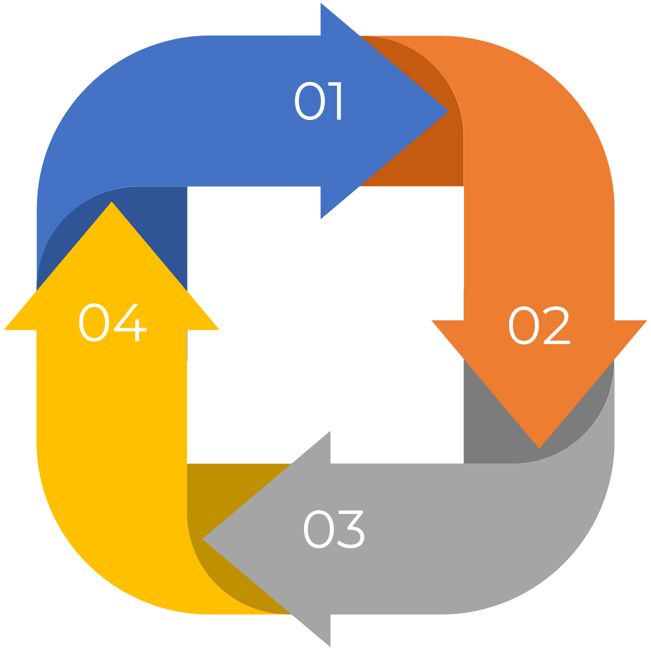
Using Non-volatile Memory

Replacing volatile memories in ES with NVMs to improve energy consumption

02

New Reliability Challenges

For example STT-MRAM suffers from limited endurance, read/write error and retention failure



03

Our Contribution

Providing reliability improvement method to alleviate mentioned reliability challenges

04

Verifications

Demonstrating that our approach alleviate energy consumption and improve reliability of NVM as well

Most Relevant Publications:

- 2013-FTSPM-DSN
- 2015-LATED-EDCC
- 2016-LER-TDMR
- 2017-A2PT-TPDS
- 2017-WIPE-ASPDAC
- 2017-AWARE-TETC
- 2017-OPTIMAS-TDMR
- 2017-RI-COST-CADS
- 2018-ORIENT-DATE
- 2018-EHMCC-ReCoSoC
- 2019-ACACHE-TCASII
- 2019-REACT-TMAG
- 2019-CLEAR-MJ
- 2019-COACH-TETC
- 2020-ROCKY-TC

Our Research Profile: Interdisciplinary Area II

01

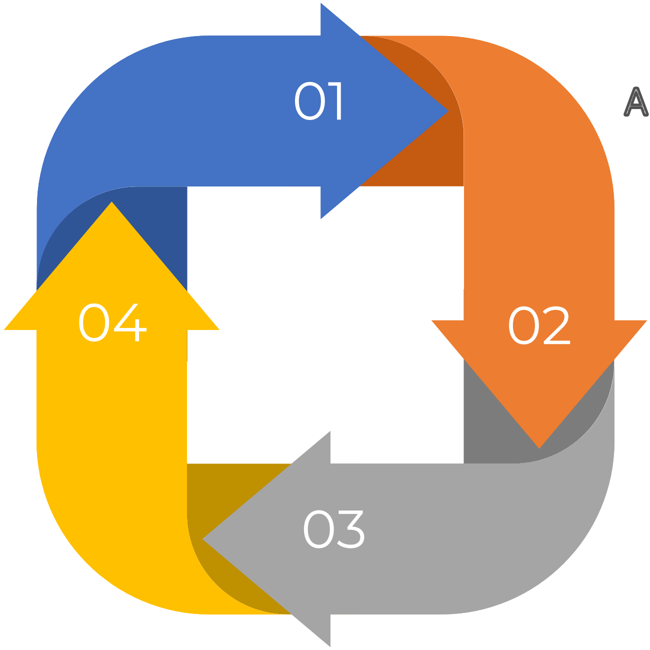
Using Non-volatile Memory

Replacing volatile memories in ES with NVMs to improve energy consumption

02

New Knobs

You don't need to waste energy for approximate data to keep reliability of them high



Most Relevant Publications:

2017-QuARK-ISLPED
2020-CAST-TCAD
2020-NOSTalgy-TC
2021-FlexCache-Springer

Approximate Computing

03

Our Contribution

Tuning of read/write voltage (current) of STT-MRAM based on data vulnerabilities

04

Verifications

Demonstrating that our approach alleviate energy consumption while it keeps the quality thresholds

Our Research Profile: Interdisciplinary Area III

01

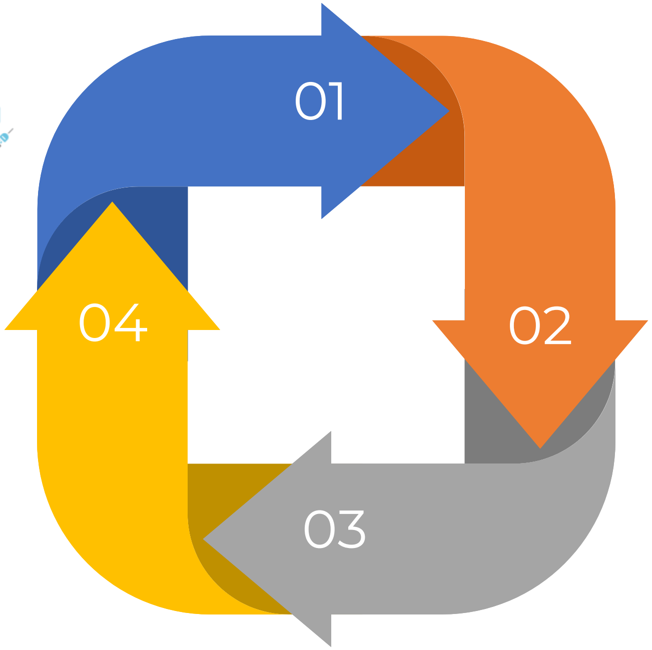
ES as IoT Nodes

Most of IoT nodes are indeed ES devices and should operate in an unmanned manner

02

LLN Networks

ES devices should connect each others in harsh environments in an energy-efficient manner



03

Our Contribution

Providing the most reliable-energy efficient ways to deliver the packet

04

Verifications

Demonstrating that our approach alleviate energy consumption while it benefits from acceptable PDR

Most Relevant Publications:

2017-RSIoT-ICSRS
2018-OF-RTEST
2019-PEDAL-SAC
2019-ERPL-MM
2019-TRMESIoT-AICT

2020-REFER-RTEST
2020-ELITE-IoTJ
2020-IMMRPL-ACCESS
2021-CBDERPL-SysCon
2021-ARMOR-IoTJ

Our Research Profile: Interdisciplinary Area IV

01

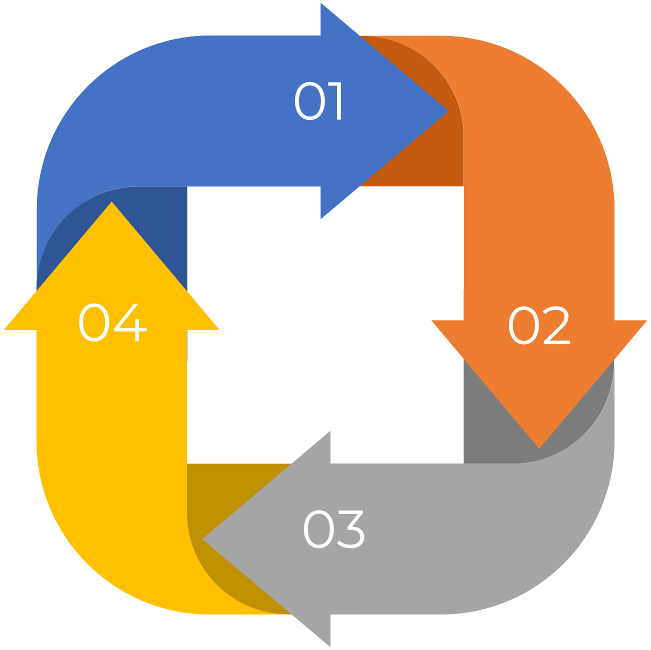
Reactive Systems

ESs are commonly reactive systems that should do a task when something happen

02

Deadlines

Most of reactive systems are also real-time and their considered tasks have deadlines.



Most Relevant Publications:

2020-READY-TC

03

Our Contribution

Providing task management algorithms to meet deadlines in an energy efficient manner

04

Verifications

Demonstrating that our approach alleviate energy consumption while it meets the deadlines

What Skills Do I Need to Work in ES Dev.?

Embedded Software

Design and implement embedded software using C and C++ (or another programming language)

Peripherals

Understand interfacing peripherals, compilers, vision control and text editors for writing code

Assemblers

Understand assemblers to convert code, libraries, debuggers and simulators

Hardware Skills

Understand embedded hardware systems and electronics schematics

Creativity

Communicate and problem solve

More information can be reached here!

6 Career Opportunities in ES Development

- Microcontroller firmware engineer
  - In general, an embedded software engineer is a person who is proficient in microcontrollers and writes firmware for microcontrollers.
    - Design and implement embedded software using C and C++
    - Design devices, including label printers, medical devices, automobile control parts and game controllers
- Embedded Linux engineer
  - An embedded Linux engineer takes care of low-level development activities.
    - Develop low-layer activities with strong embedded constraints
    - Test developed modules
    - Run unit tests

More information can be reached here!

6 Career Opportunities in ES Development

- Embedded applications engineer
  - Managing applications is a big part of what embedded developers do.
    - Manage embedded software frameworks
    - Work on open source stacks and applications
    - Work with different programming languages, including Embedded C and Python
- Embedded network engineer
  - Embedded network engineers take care of various network devices like routers, access points, firewalls, network backend infra, bridges and switches.
    - Work on optimization of packet data transfer within a network
    - Work on network layers L3, L4-bridging and muxing
    - Manage network processors

More information can be reached here!

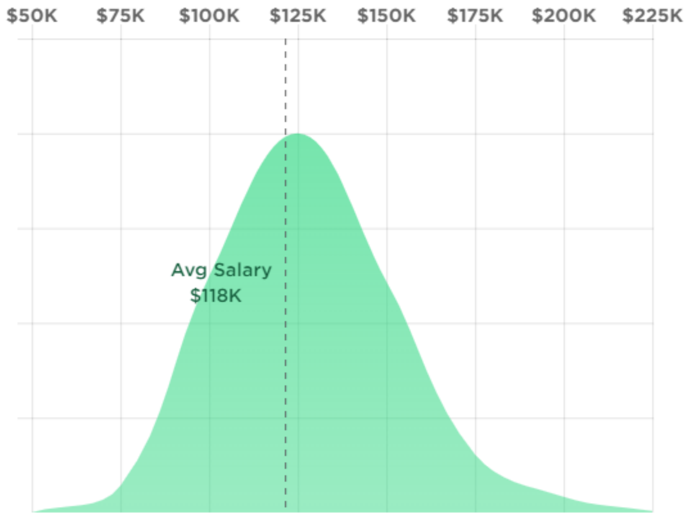
6 Career Opportunities in ES Development

- Embedded IoT application developer
  - With the growth of the internet of things (IoT), embedded IoT application professionals are more relevant and in demand than ever.
    - Design and implement embedded software using C and C++
    - Validate new product solutions compliance to standards
    - Use version control, test-driven development, mobbing and other best practices
- Cybersecurity embedded developer
  - One should use her/his technical expertise to ensure the safety of ES.
    - Design APIs
    - Understand hardware security modules, PKI, transport layer security and common application security vulnerabilities
    - Test and debug

More information can be reached here!

How Much Money Can I Make?

- According to Hired.com, embedded software engineers make an average annual salary of $121,000 in US.
  - But remember, there are a lot of factors play when it comes to average salaries, including location, industry, organization size and more.



More information can be reached here!

Conclusion

What did we Talk about?

ES Definition

Information processing systems embedded in to a larger product.

ES Components

Processing elements (hardware) + software (drivers, applications, OS) + Peripherals (shields)

ES Design

Classical method, development board method, ES design characteristics that should be considered

Research Areas and Job Positions

Most important research areas, IUST CPS-LAB research direction, job skills and job opportunities

THANK YOU

Cyber-Physical Systems Laboratory

cps.iust.ac.ir ☎ +98 (21) 73225350 ✉ monazzah@iust.ac.ir

Room 121, Department of Computer Engineering, Iran University of Science and Technology, University Road, Hengam Street, Resalat Square, Narmak, Tehran, IRAN 16846-13114.