UCDAVIS
UNIVERSITY OF CALIFORNIA

Internet of Things

Senior Design Project Course

Processing - Part 2

Lecturer: Avesta Sasan

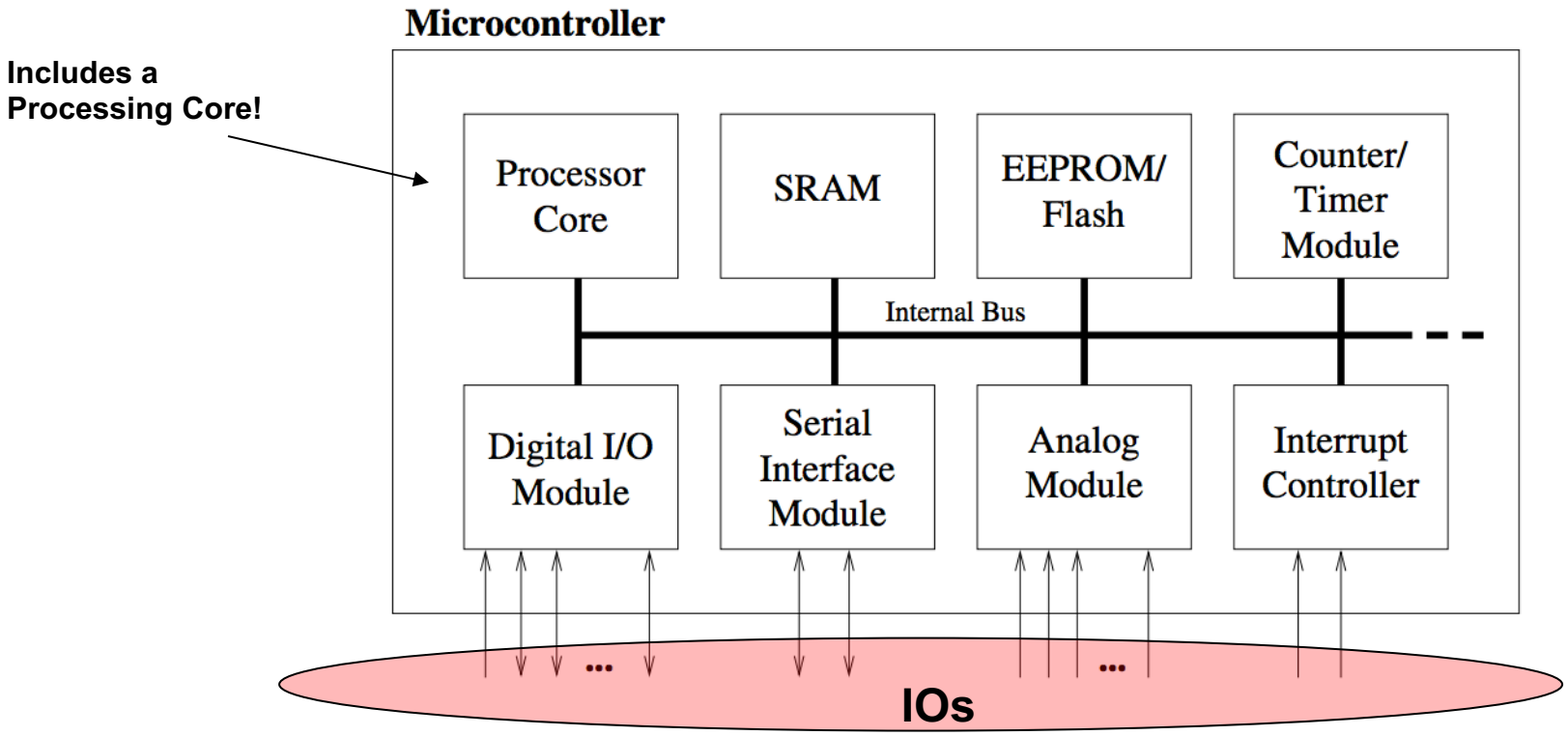
University of California Davis

Focus of Today's Lecture: **MCU**

Image source: http://www.cchc.cl/informacion-a-la-comunidad/industria-de-la-construccion/personaje/

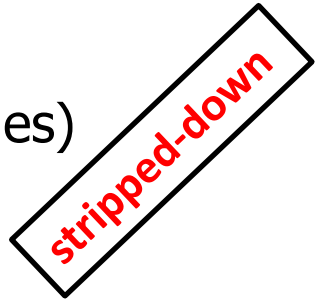
Microcontroller Basic Design (Review)

- All components are connected via an **internal bus**.
- All components are **integrated on one chip**.
- **Communicate** to outside world **via IOs**.



What is not Inside a MCU? (Review)

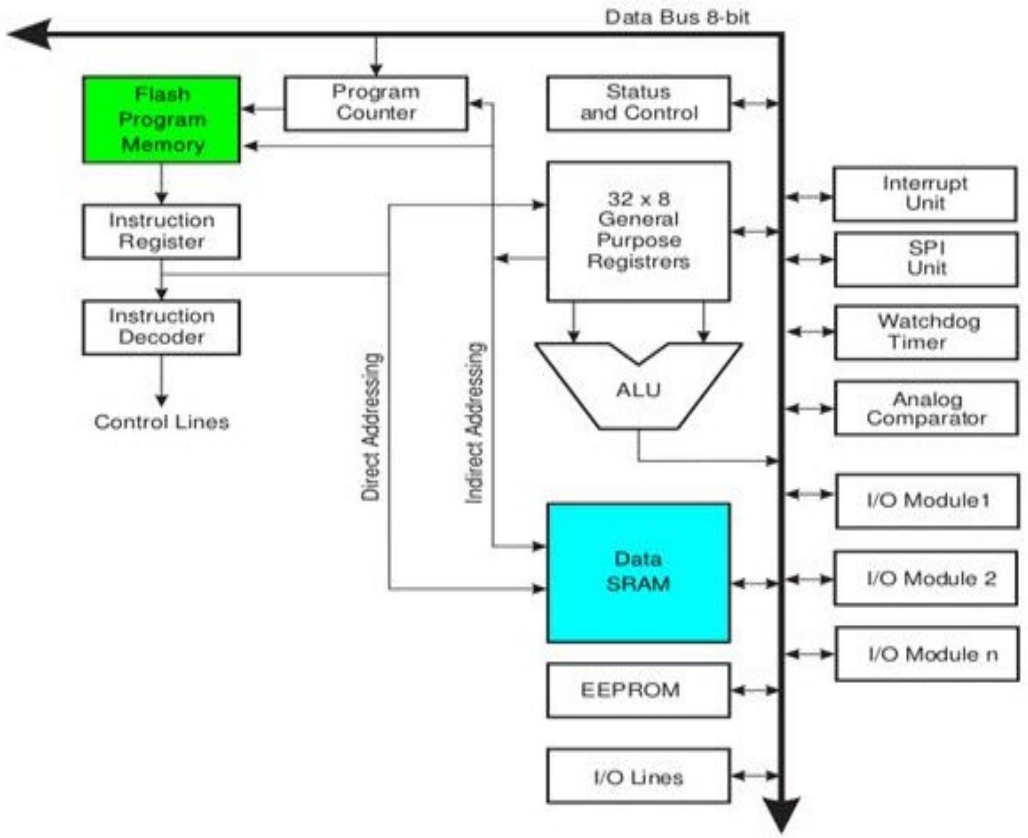
- No Cache!
- No MMU (maybe you see this in larger microcontrollers)
- No complicated pipeline (single or simple multicycle pipelines)
- No disk
- No FP ALU
- ….



A microcontroller is a (stripped-down) processor which is equipped with memory, timers, (parallel) I/O pins and other on-chip peripherals.

Example: Arduino Processor: (Review)

- Uses the Harvard architecture
  - The program code and program data have separate memories
- Single level pipeline to execute the instructions in order
- 32 x 8 bit general purpose registers
- Single clock cycle access time
- Single cycle ALU operation



Simple

Example: IBM Power5 (Review)

- 2 cores, out-of-order execution
- 100-entry instruction window in each core
- 8-wide instruction fetch, issue, execute
- Large, local+global hybrid branch predictor
- 1.5MB, 8-way L2 cache
- Aggressive stream based prefetching

Complex

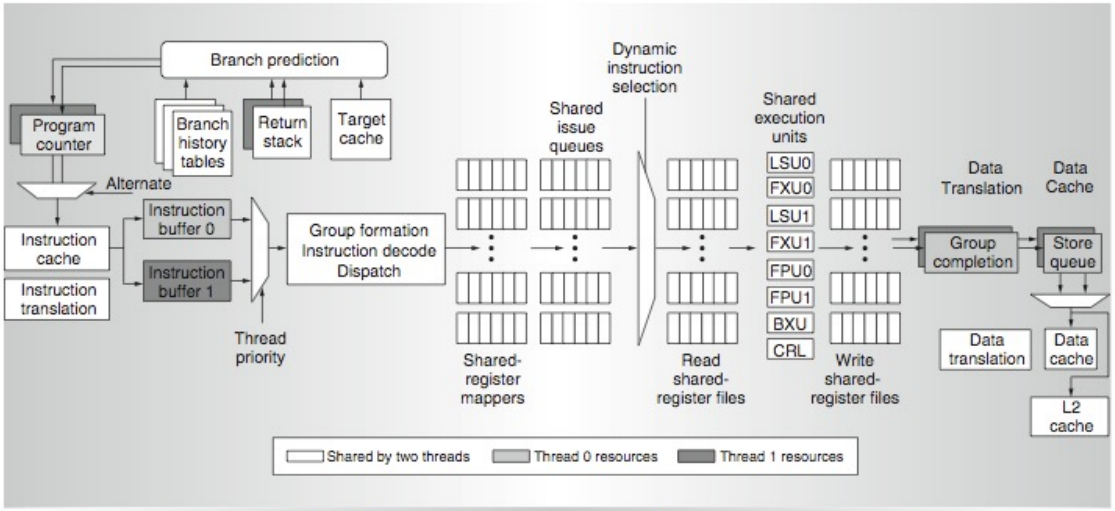
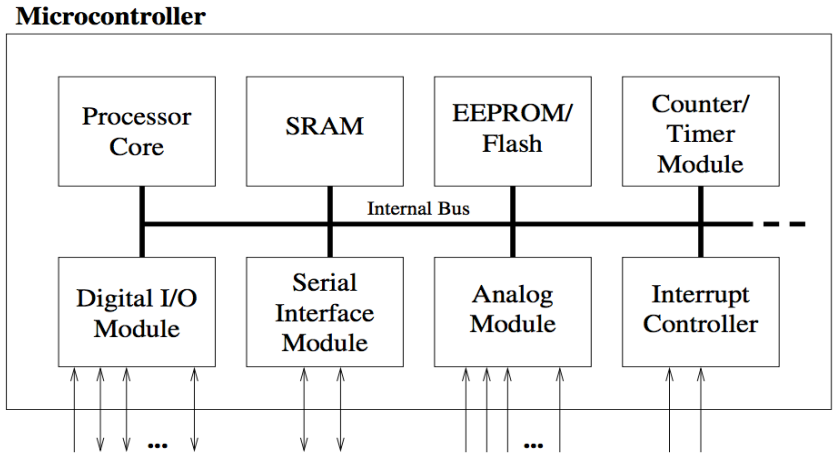
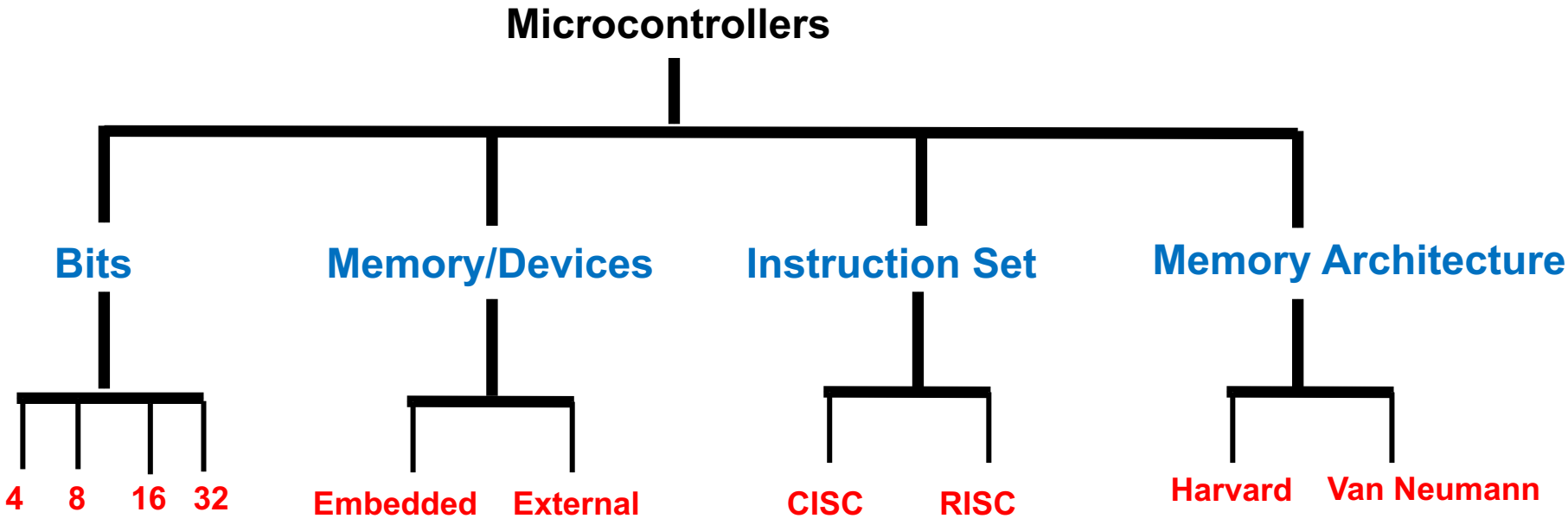


Figure 4. Power5 instruction data flow (BXU = branch execution unit and CRL = condition register logical execution unit).

Microcontroller Classification (Review)





Microcontroller vs. Microprocessor (Review)

Microprocessor

- CPU is stand-alone, RAM, ROM, I/O, timer are separate
- designer can decide on the amount of ROM, RAM and I/O ports.
- Expensive
- Versatility
- General-purpose
- High processing power
- High power consumption
- Instruction sets focus on processing-intensive operations
- Typically 32/64 – bit
- Typically deeply pipelined (5-20 stages)

Microcontroller

- CPU, RAM, ROM, I/O and timer are all on a single chip
- fixed amount of on-chip ROM, RAM, I/O ports
- For applications in which cost, power and space are critical
- Single (or limited) purpose (control-oriented)
- Low processing power
- Low power consumption
- Bit-level operations
- Instruction sets focus on control and bit-level operations
- Typically 8-16 bit
- Typically single-cycle/two-stage pipeline

Example:

- A MPU in a GP architecture, running at **600 MHz** has an average **CPI** (number of **C**lock needed **P**er **I**nstruction) **of 1.2** and a average power consumption of **400 mW.** It costs $100.

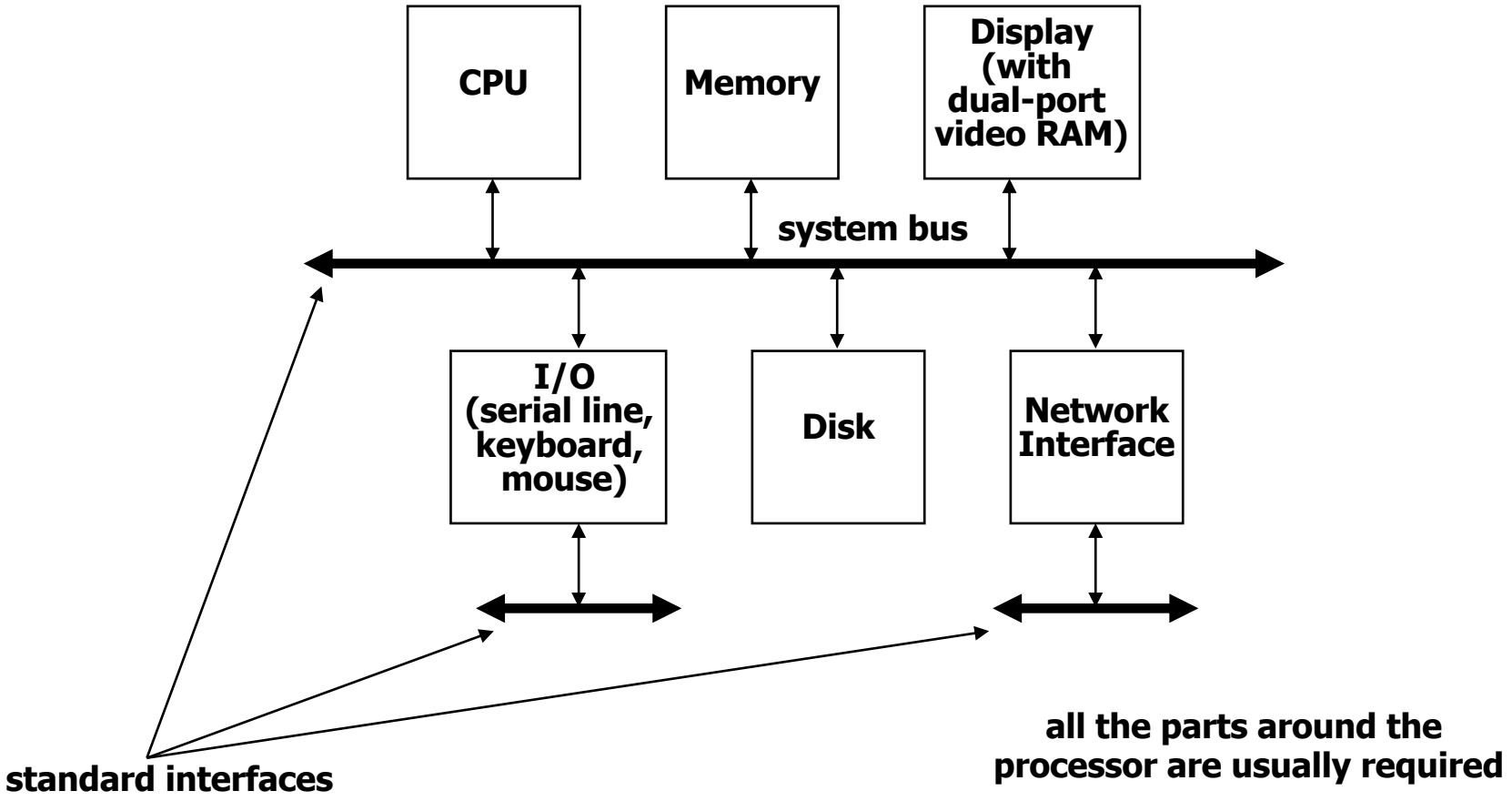
- A processor in a MCU running at **12 MHz** with a **two cycle datapath** has a power consumption of **2.4 mW**. It cost $0.96.
  - What is the associated CPI?

- Calculate their respective MIPS (**M**illions of **I**nstructions processed **P**er **S**econd)
  - **MPU:** $600{,}000{,}000\ \frac{clk}{s} * \frac{1}{1.2}\ \frac{Inst}{clk} = 500{,}000{,}000\ \frac{Inst}{s} = \mathbf{500}\boldsymbol{MIPS}$
  - **MCU:** $12{,}000{,}000\ \frac{clk}{s} * \frac{1}{2}\ \frac{Inst}{clk} = 6{,}000{,}000\ \frac{Inst}{s} = 6MIPS$

Example:

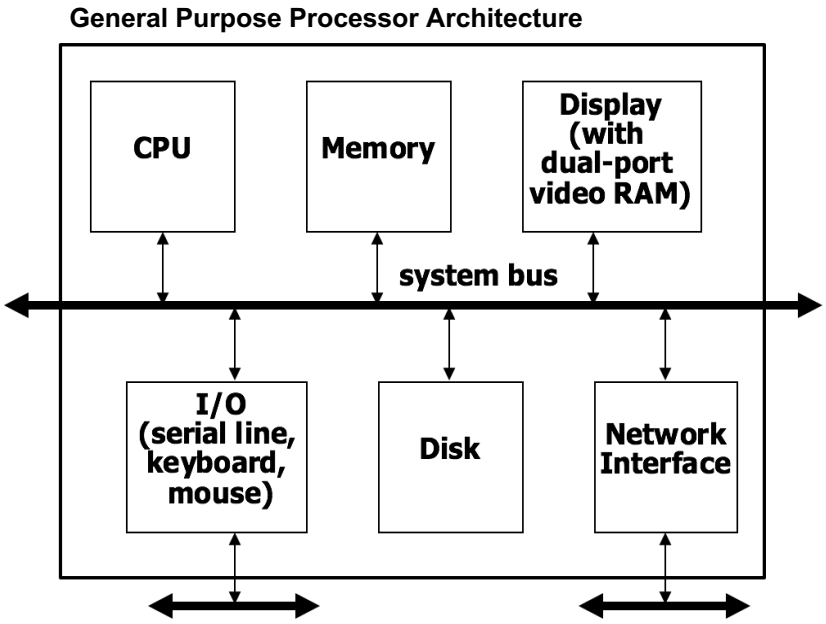
- Which one is more efficient in MIPS/mW?
  - **MPU:** $\frac{500\boldsymbol{MIPS}}{400\boldsymbol{mW}} = 1.25MIPS/mW = 1.25\frac{\frac{milion\ instructioin}{s}}{\frac{J}{s}} = 1.25\frac{milion\ instructioin}{j}$
  - **MCU:** $\frac{6\boldsymbol{MIPS}}{2.4\boldsymbol{mW}} = \boldsymbol{2.5MIPS/mW} = 2.5\frac{\frac{milion\ instructioin}{s}}{\frac{J}{s}} = 2.5\frac{milion\ instructioin}{j}$
- Which is more efficient in MIPS/$?
  - **MPU:** $\frac{500\boldsymbol{MIPS}}{\$100} = 5MIPS/\$ = 5\ \frac{milion\ instructioin}{j.s}$
  - **MCU:** $\frac{6\boldsymbol{MIPS}}{\$0.48} = \boldsymbol{12.5MIPS/\$} = 12.5\ \frac{milion\ instructioin}{j.s}$

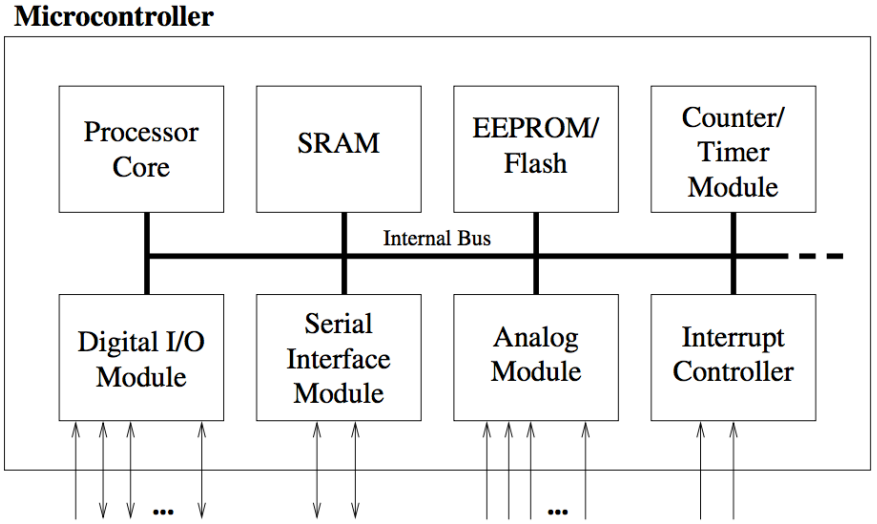
Typical General-Purpose Architecture



GPP vs MCU

- Complexity of which processing core (CPU) is higher?
- What is different about the way MCU and GPP communicate to outside?





Which is Used in IoT?

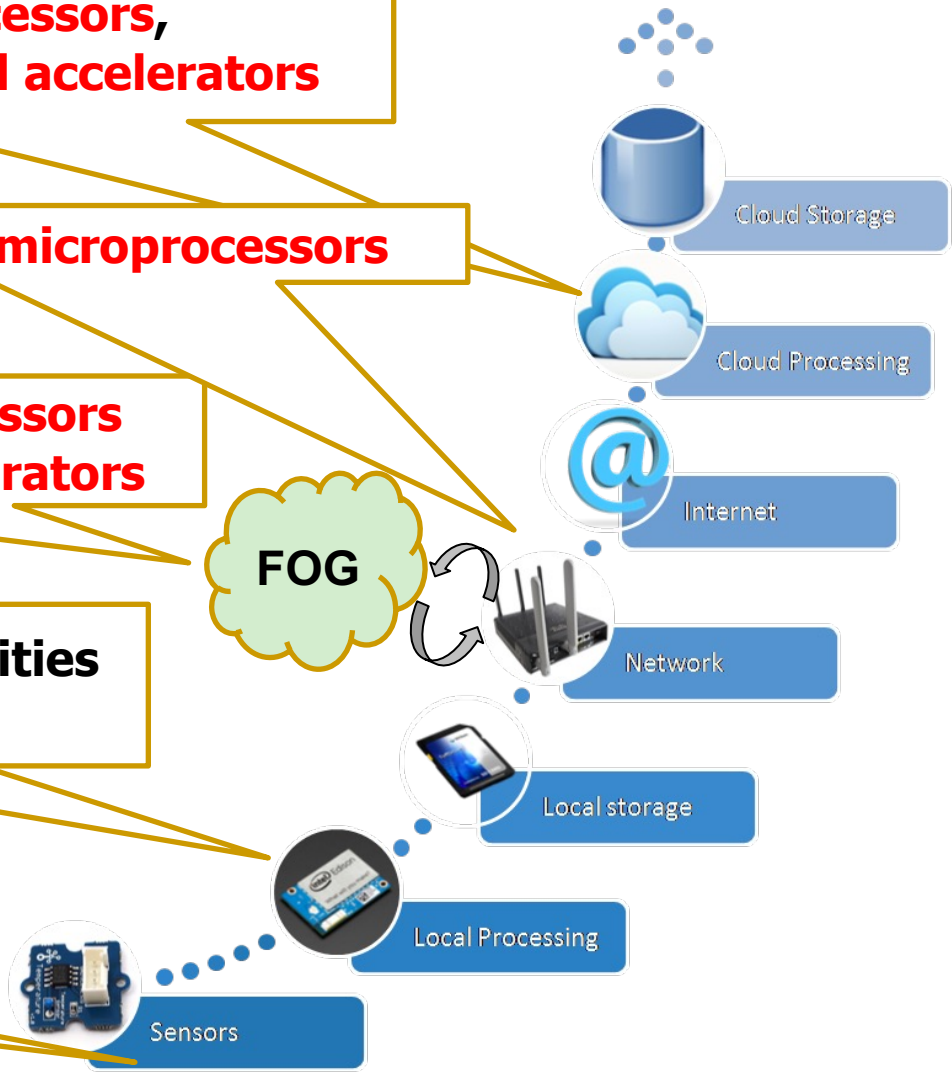
Many high-end microprocessors, assisted by all kinds of high-end accelerators

One or few mid-stength microprocessors

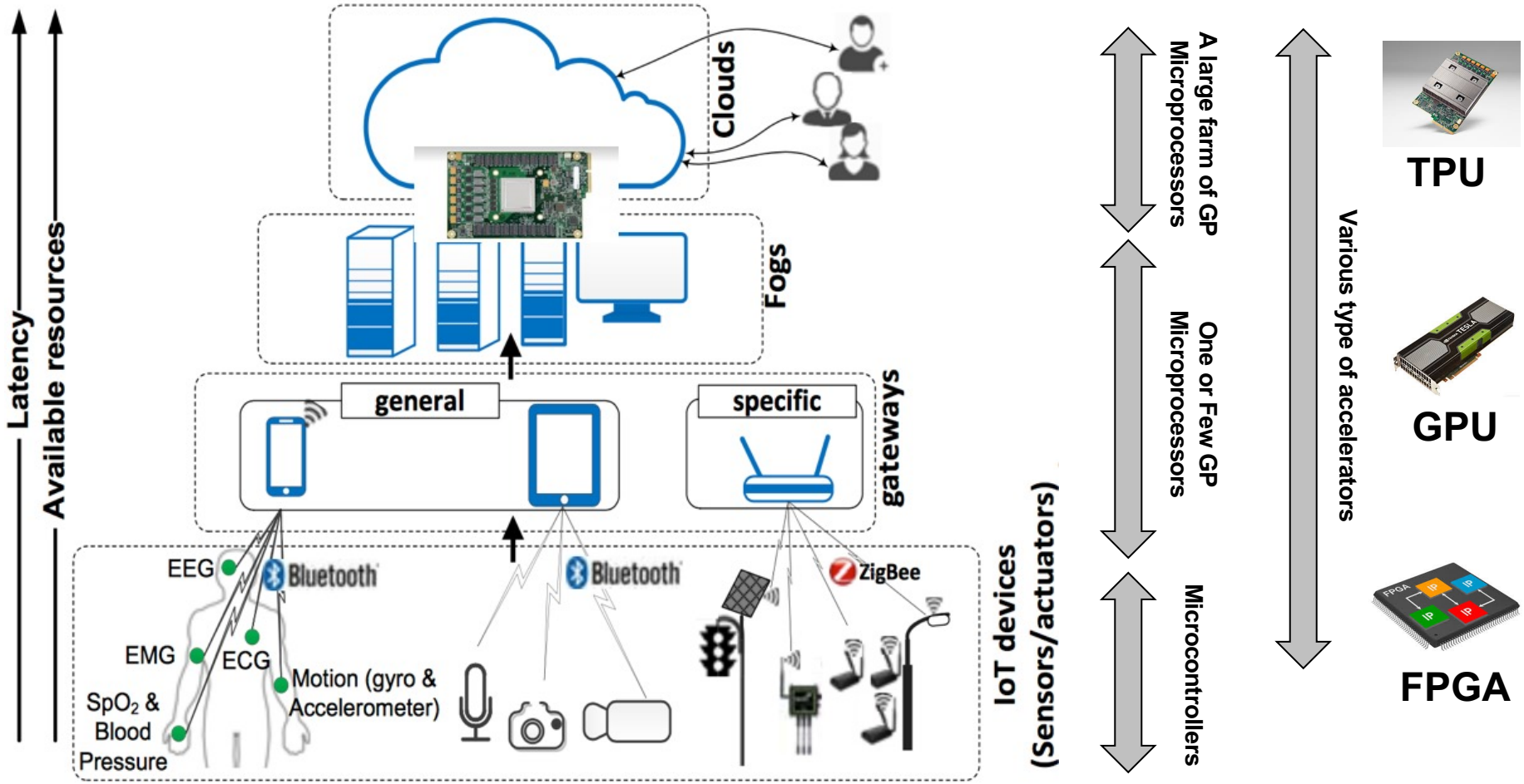
One or few semi-strong microprocessors maybe assisted by one or few accelerators

Microcontrollers with various abilities depending on the application

Very limited function ICs or controllers in MEMs

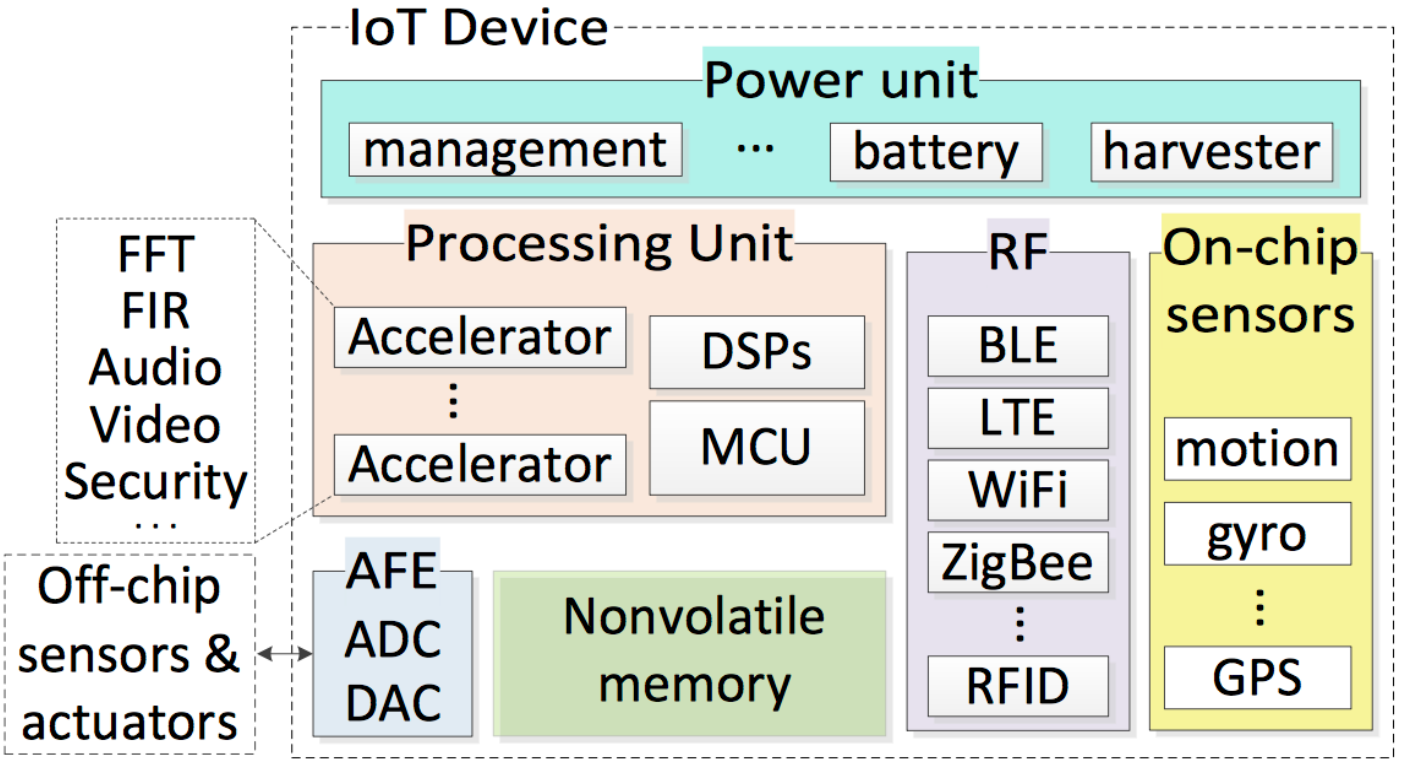


IoT Chain Computation Layers



F. Samie, L. Bauer and J. Henkel, "IoT technologies for embedded computing: A survey," *2016 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Pittsburgh, PA, 2016, pp. 1-10.

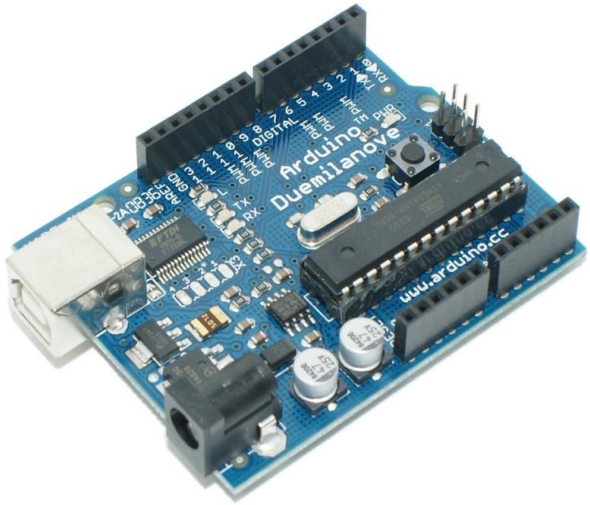
General Arch. of an IoT Device:



Lets look at some examples!

Arduino Microcontroller

- Inexpensive ($6 - $50 depending on package!)
- Small size
- Easily Programmable
- Easily connectable
- Open source with big developer community
- Simple to use software
- Easy to augment the functionality
  - Wire directly into the pins on the Arduino board
  - Stack chips called "**shields**" on top of the base unit.

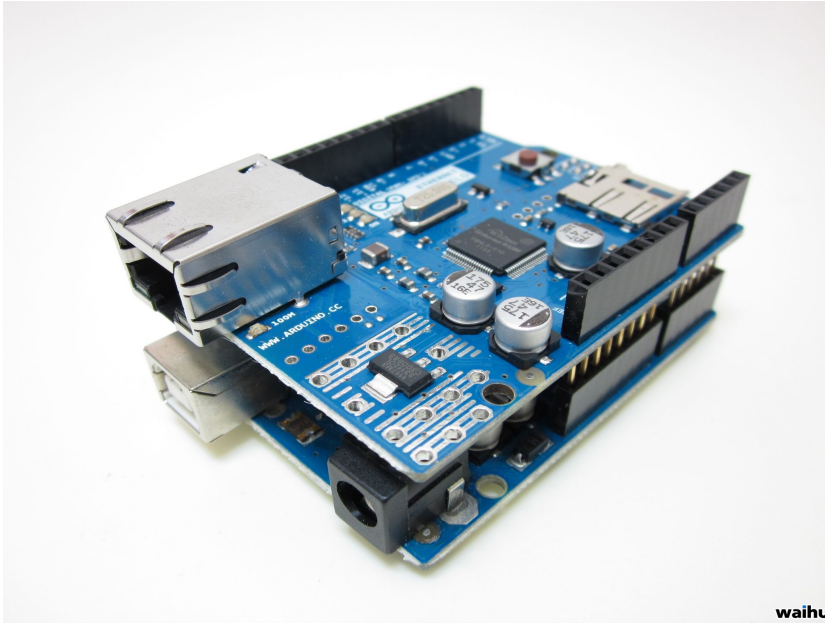


- **https://www.arduino.cc**

ARDUINO

Arduino Ethernet Shield

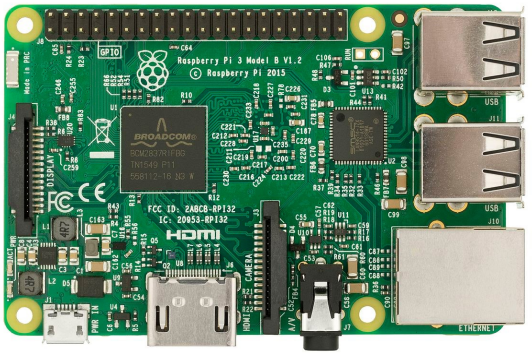
- Extends the Microcontroller functionality:
  - Connect your Arduino board to the internet.
- Open source
- Simple to use software
- You can **keep stacking** the shields!



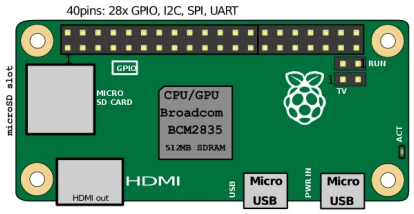
ARDUINO

Raspberry Pi

- It is a computer
- It runs Linux
- More software oriented programming
- Embeds a full Networking System
- It is born in the United Kingdom to promote teaching of basic computer science.
- https://www.raspberrypi.org

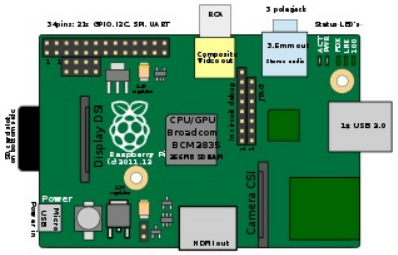


Pi Zero



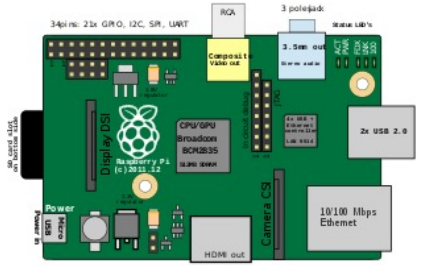
2013

Model A



2015

Model B



Raspberry Pi vs Arduino

Hardware

A microcontroller motherboard

run one program at a time, over and over again

begins executing code when turned on and stops when you pull the plug

much easier to connect analog sensors

Software and Networking system

A general-purpose computer

Can run multiple programs

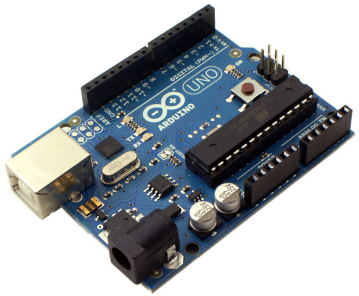
Need 5V supply to remain on, and is shut down via a software process

Built-in Ethernet port

requires software to effectively interface with other devices

Good for Sensors

https://www.arduino.cc



Arduino
$25
ATmega328

http://chipkit.net

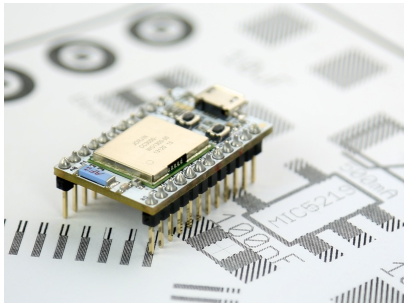
ChipKIT
$30
PIC

http://www.ti.com/lsds/ti/tools-software/launchpads/launchpads.page

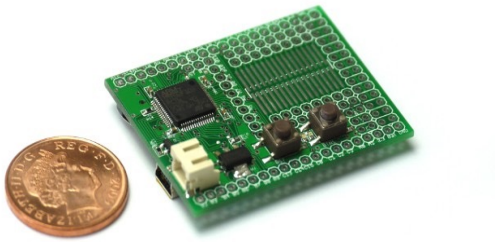
LaunchPad
$4
MSP430

Good for Sensing & Processing

STM32
$30
ARM Cortex M0, M3, M4



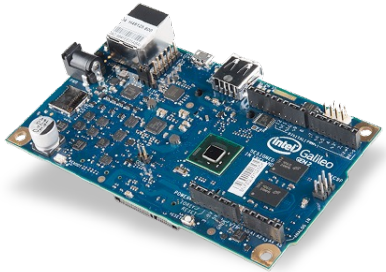
Particle
$35
ARM
WiFi Internet



Espruino
$30
ARM
Javascript

Good for Processing & Networking

Raspberry Pi
$35
900 MHz ARM CPU
250 MHz GPU
1 GB RAM
Compute Module



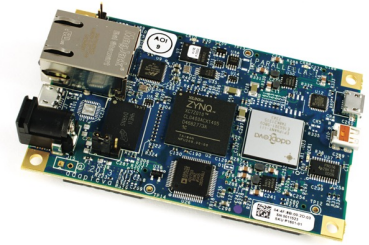
Intel® Galileo
$50
400 MHz Quark x86
256 MB RAM

Intel® Edison
$70
1 GHz Dual Core Atom x86
1 GB RAM
WiFi
BLE
4 GB Flash

Good for Processing and Network

Beaglebone Black
$45
1 GHz ARM, GPU
512 MB RAM
4 GB Flash

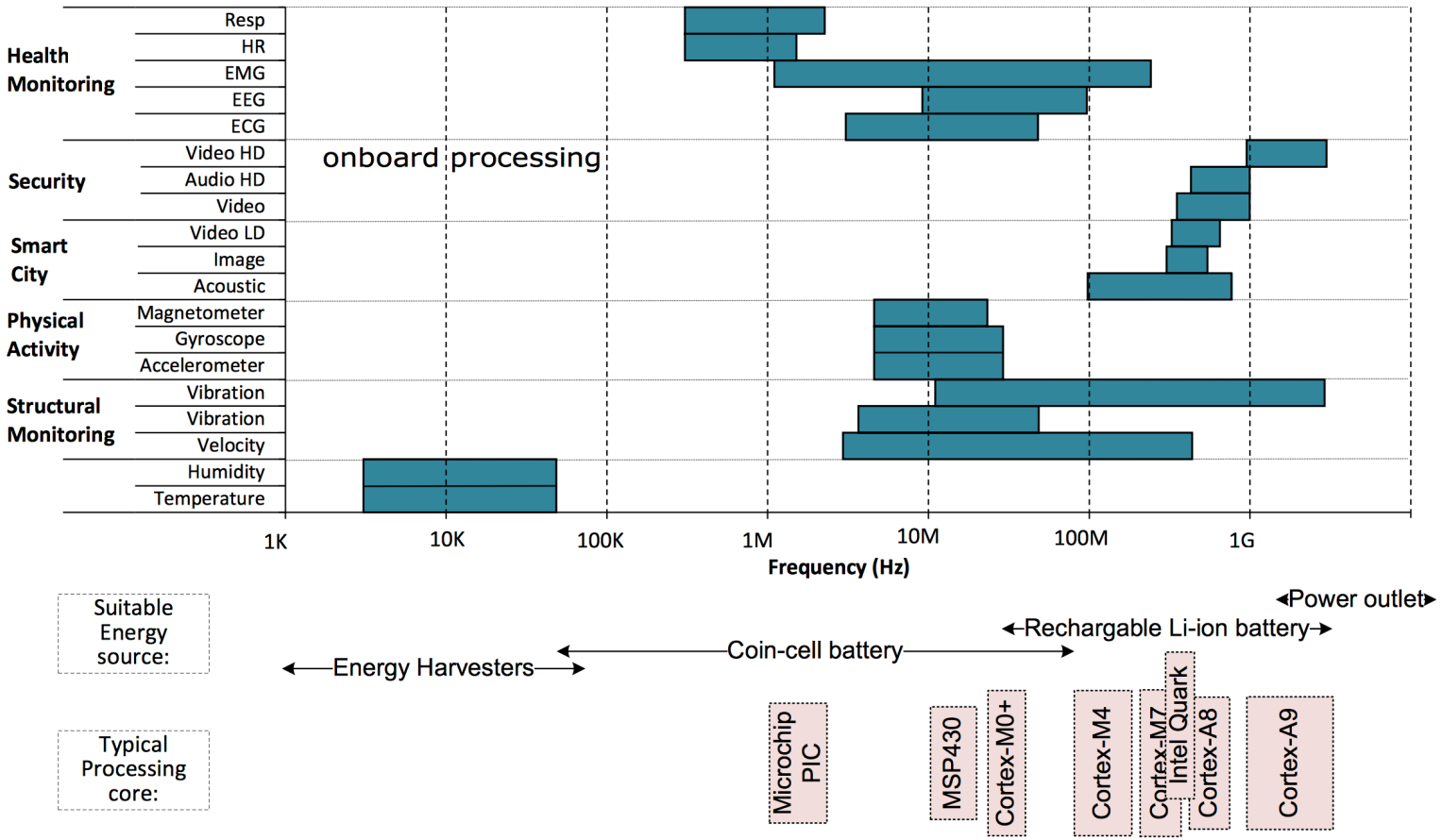
UDOO Neo
$50
i.MX 6 Solo ARM, GPU
ARM M4
512 MB or 1 GB RAM



Parallella
$99
1 GHz Dual Core Zynq ARM
16 or 64 Epiphany CPUs

Processing Sensor Data

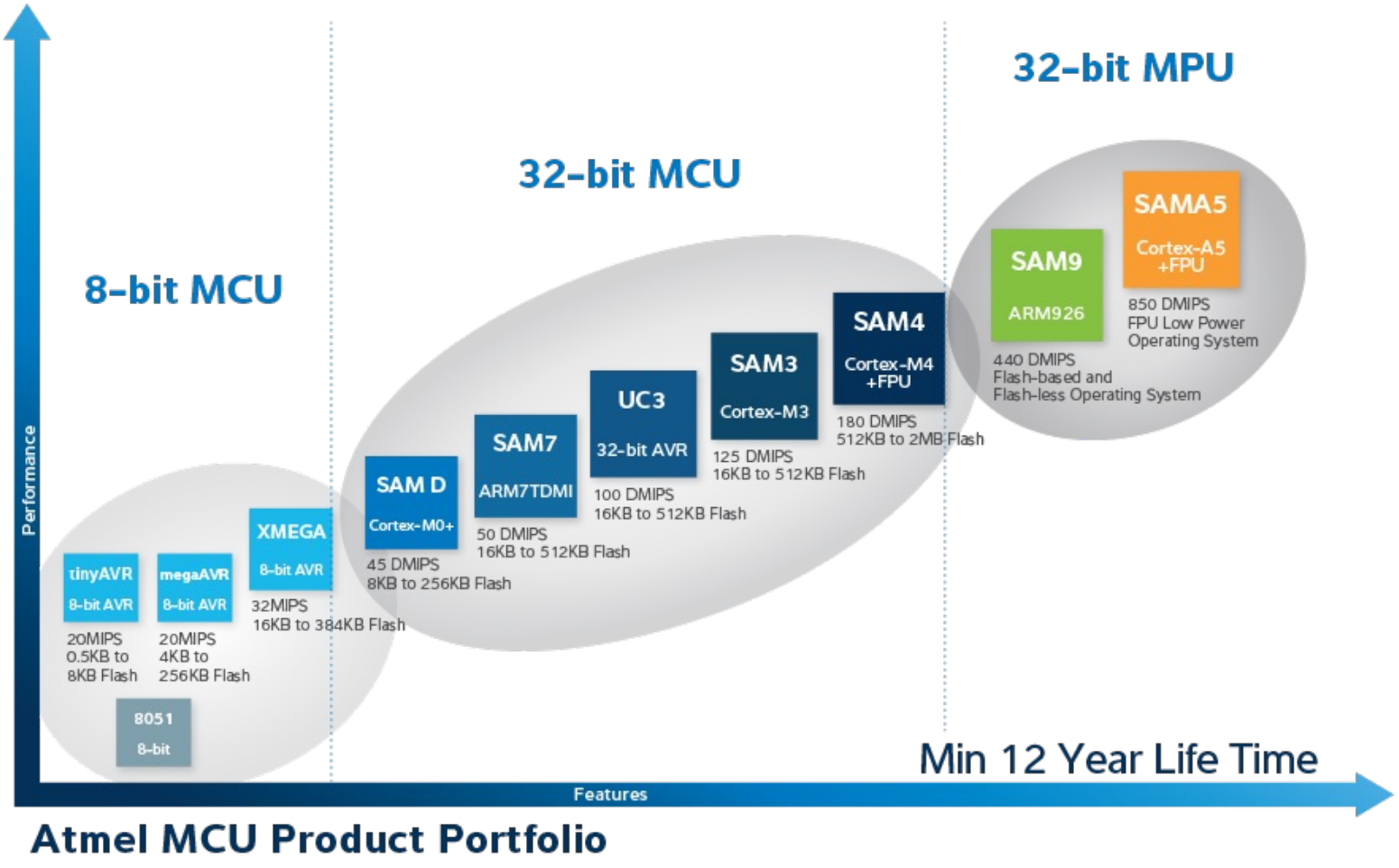
- The number of cycles (i.e. required frequency) to fully process the IoT sensors



F. Samie, L. Bauer and J. Henkel, "IoT technologies for embedded computing: A survey," *2016 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Pittsburgh, PA, 2016, pp. 1-10.

Wide Range of MCU Choices

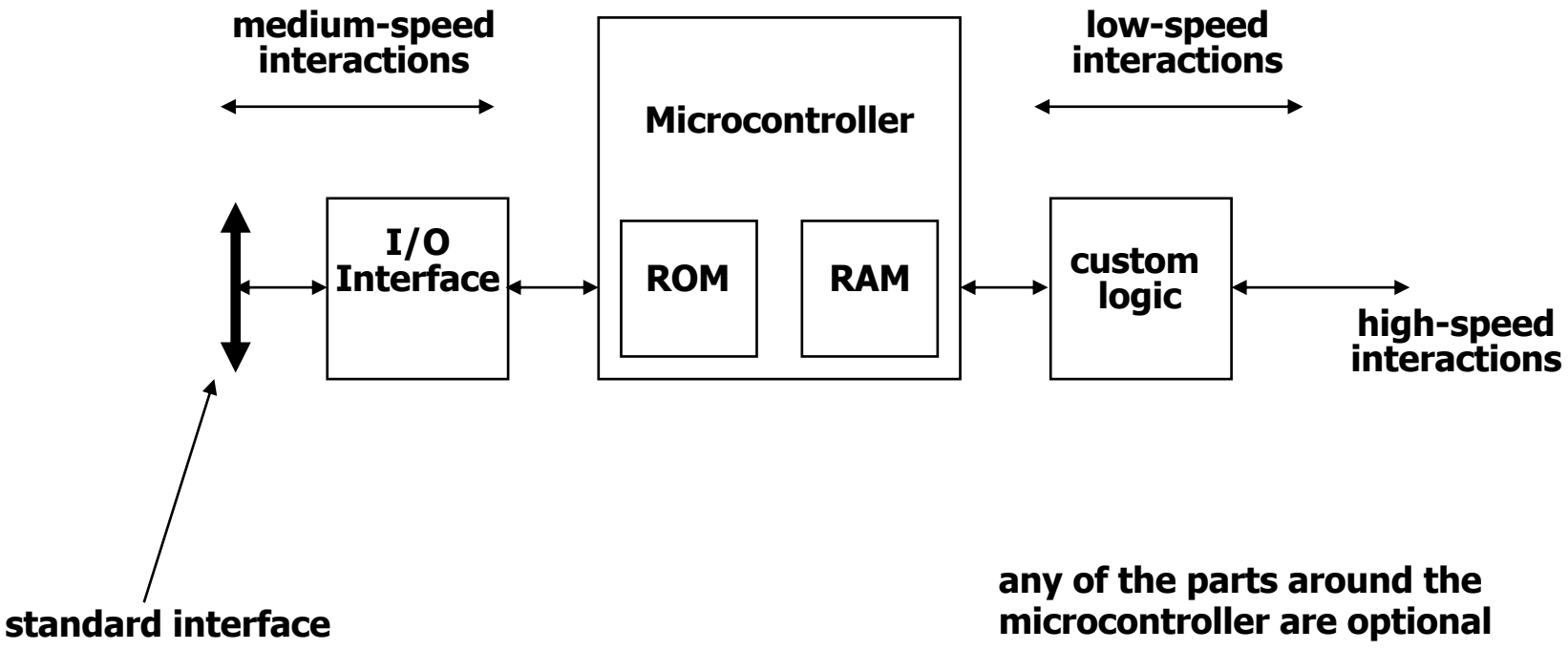
- As an example, see how many MCUs are offered by Atmel



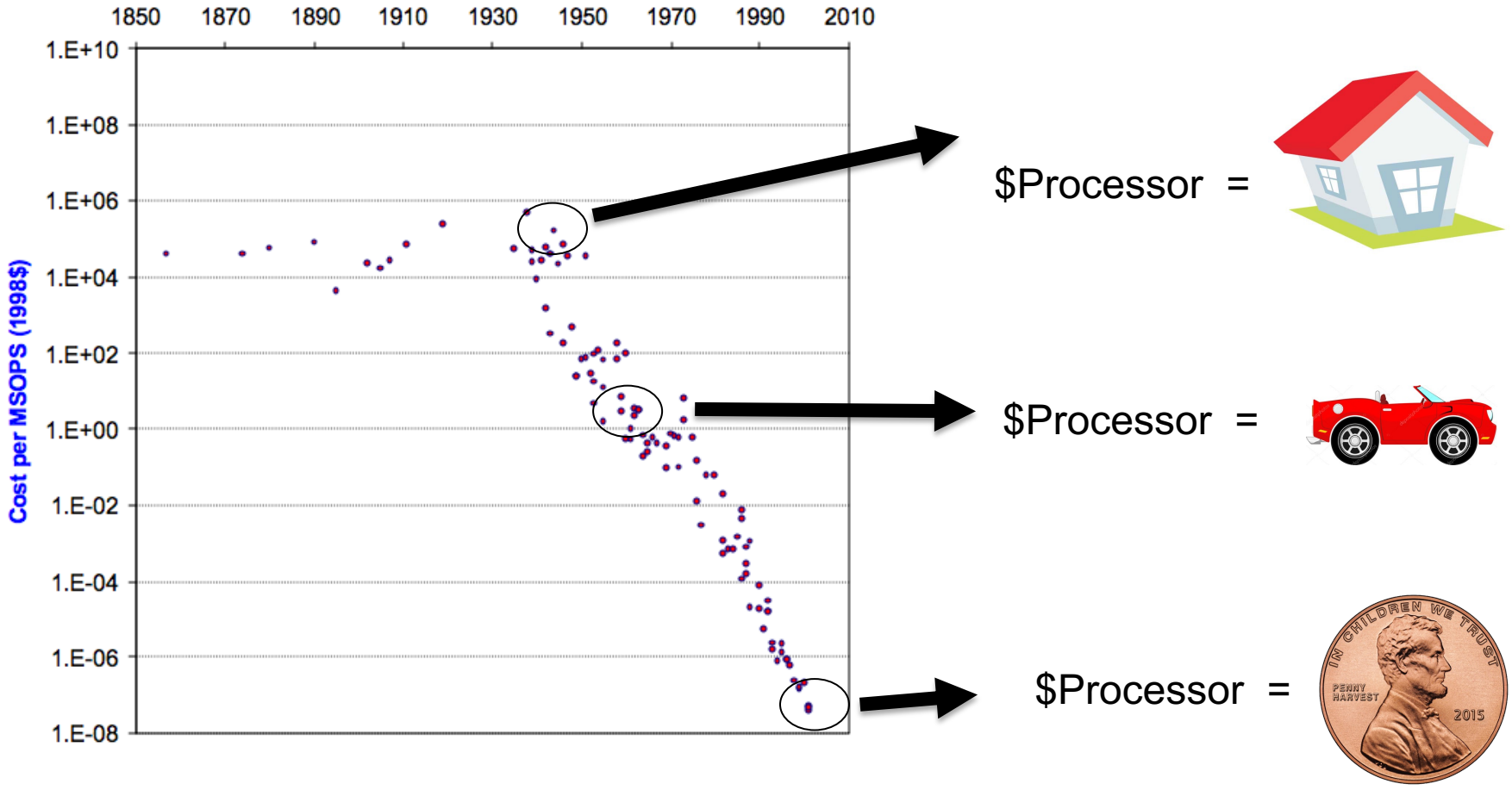
The End!


WHEEEEAH...
THANK
GOODNESS!
H. Brown
CARTOONSTOCK
.com
Search ID: hbrn1957

Typical task-specific architecture



Cost of Processing Drops Quickly!



Similar trend happened to sensors!

This allows us to put few sensors and a processor in any and every object!