INSTITUTO DE INGENIERÍA ELÉCTRICA

FACULTAD DE INGENIERÍA UDELAR

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Introducción y conceptos básicos

Sistemas embebidos para tiempo real

NEON
Network of Competence on Internet of Things

Co-funded by the Erasmus+ Programme of the European Union

Objetivos

- Definir sistemas embebidos e identificar sus características.
- Describir el hardware y explicar las consideraciones de diseño a tener en cuenta
- Definir sistemas de tiempo real e identificar sus características
- Indicar la importancia del uso de lenguaje C en los sistemas embebidos y de tiempo real

Agenda

- Sistemas embebidos
  - Definición / Características / Consideraciones de diseño / Ejemplos
- Sistemas de tiempo real
  - Definición / Características
- Sistemas embebidos y de tiempo real
  - Ejemplos

Actividad en grupo

- ¿Qué es un “sistema embebido”?
  - Objetivo:
    - Lograr una primera aproximación consensuada sobre el término “sistema embebido”
    - Escribir: Definición (tentativa) / Características
  - Grupos:
    - 3 a 4 participantes
  - Tiempo:
    - 5 minutos
  - Puesta en común.

¿Qué es un sistema embebido?

- “An embedded system is an **engineering artifact** involving computation that is subject to **physical constraints**. The physical constraints arise through two kinds of interactions of computational processes with the physical world: (1) **reaction to** a physical environment, and (2) **execution on** a physical platform. [...] Common reaction constraints specify **deadlines**, **throughput**, and **jitter**; they originate from the behavioral requirements of the system. Common execution constraints put bounds on available **processor speeds**, **power**, and **hardware failure rates;** they originate from the implementation requirements of the system.”

  Thomas A. Henzinger and Joseph Sifakis
  "The Embedded Systems Design Challenge”, Invited Paper,
  Formal Methods for Security and Trust in Industrial Applications, 2006.

¿Qué es un sistema embebido?

- Un sistema embebido es un dispositivo:
  - realiza procesado/computación
  - sujeto a **restricciones físicas:**
    - **reacción al** entorno físico,
    - **ejecución en** cierta plataforma física
- Restricciones:
  - reacción: **deadlines**, **throughput**, **jitter**...
  - ejecución: **velocidad del procesador**, **potencia/energía**, **tasa de falla de hardware**...

¿Qué es un sistema embebido?

- Dispositivos electrónicos que incorporan una computadora (normalmente un microprocesador) en su implementación.
- Microprocesador usado principalmente para simplificar el diseño del sistema y darle flexibilidad.

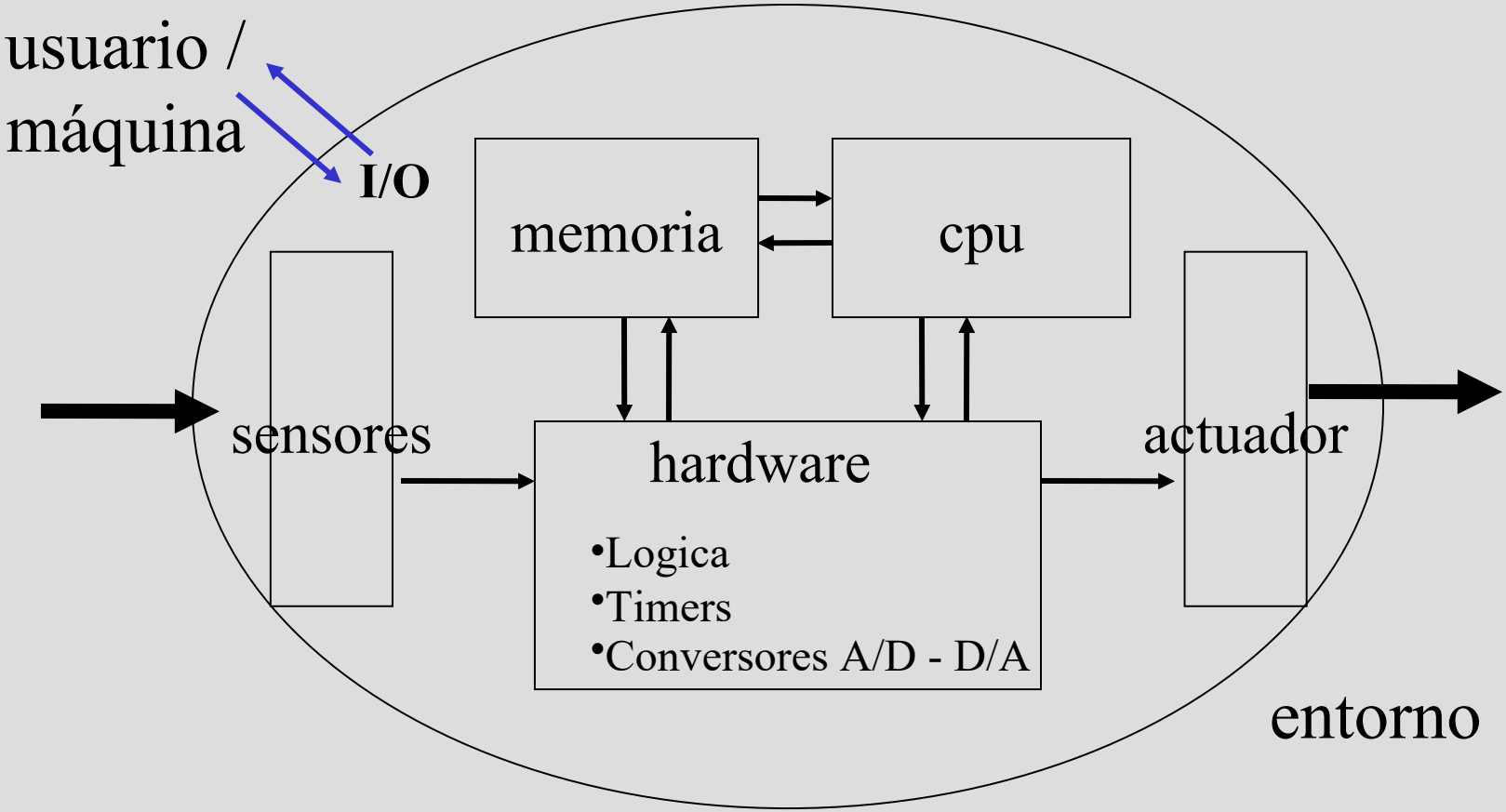
¿Qué es un sistema embebido?

- Características:
  - Requerimientos específicos y realiza tareas predefinidas.
  - Frecuentemente el usuario no es consciente que se usa una computadora.
  - Recursos limitados.
  - Costo reducido.

¿Qué es un sistema embebido?

- Características: en general no tienen
  - teclados
    - si pueden tener botones o tecladito (keypads)
  - pantalla (monitor)
    - si pueden tener LEDs o pantallitas de LCD/LED
  - discos duros, CD’s, memorias USB
    - si pueden tener memorias ROM/RAM (o memoria SD)
    - si pueden tener conexiones (red, modems)

Sistemas embebidos: hardware



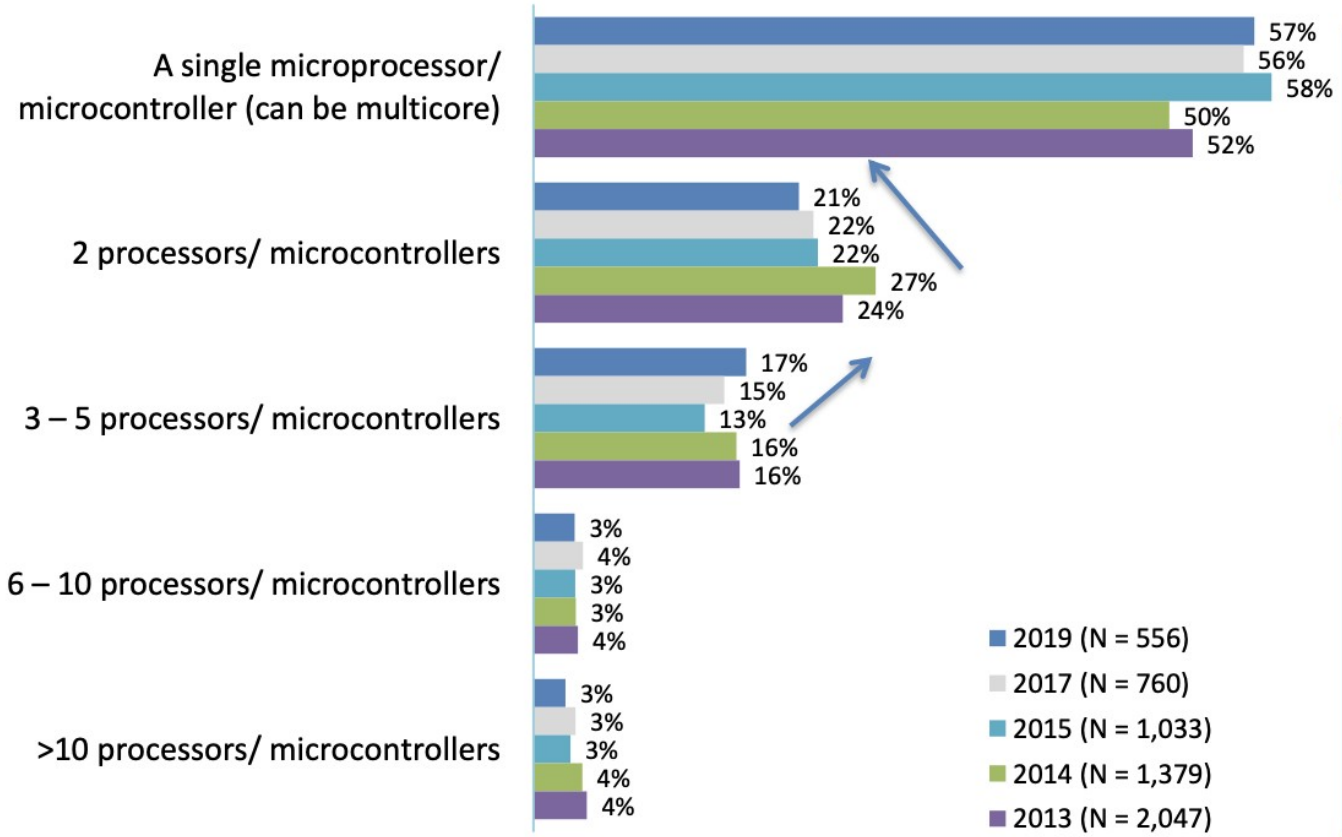
Sistemas embebidos: hardware

- Microprocesador (uP)
  - Chip que contiene la CPU
  - Ejemplos: Intel (PC), ARM, DSP
- Microcontrolador (uC)
  - Básicamente un uP con memoria y periféricos I/O
  - Diseñados para funciones especificas
  - Solución integral: reducción de partes, costo, consumo y/o tamaño.
  - Ejemplos: MSP430 (Texas Inst.), Atmega (Atmel), PIC (Microchip), ARM (múltiples fabricantes).
- SoC (System on chip) o SiP (System in a package)
  - Microcontrolador más radio
  - Procesador + co-procesadores (imágenes/video)

Sistemas embebidos: hardware

- Disponibilidad de un amplio rango de microprocesadores y/o microcontroladores
  - Costo y rendimiento varían dependiendo de:
    - tamaño de palabra/bus
    - memoria interna (tipos y tamaño)
    - periféricos
      - temporizadores, UART, puertos I/O, canales DMA, etc.
    - Soporte para memoria externa
    - velocidad de reloj

My current embedded project contains:



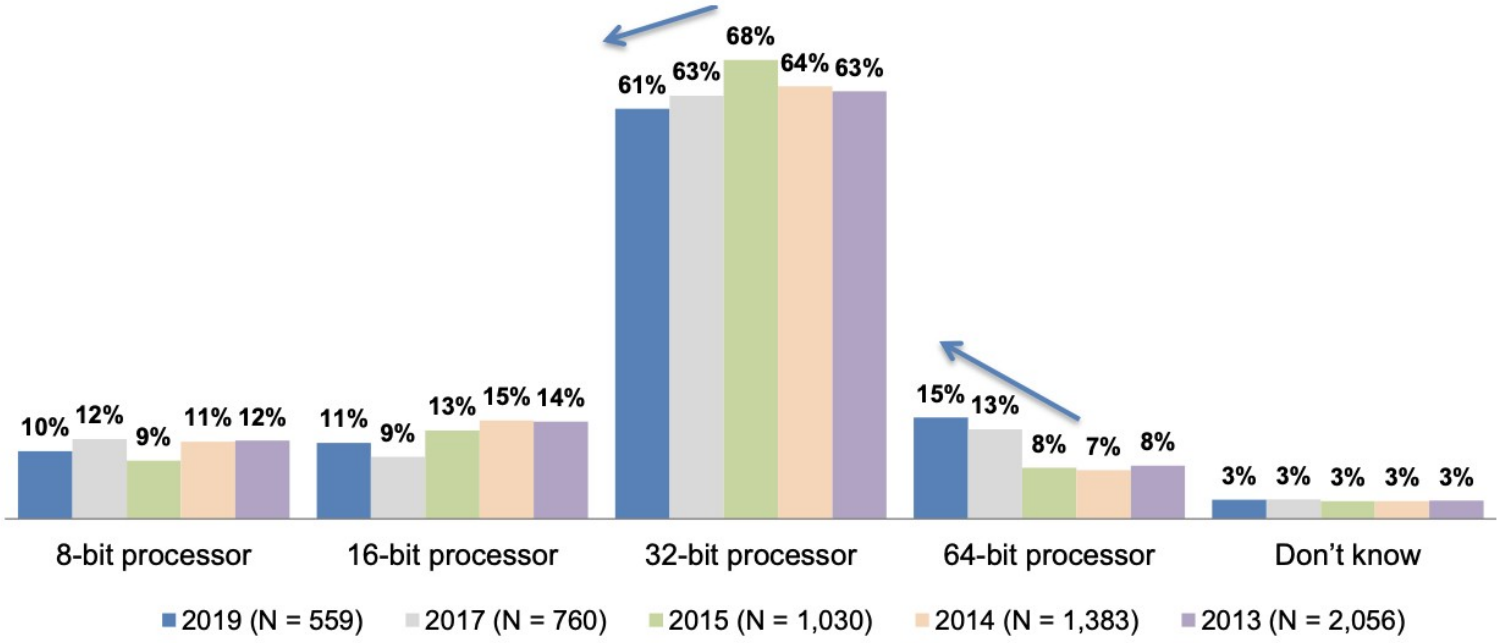
65% of EMEA designs contained a single processor.

The average number microprocessor/micro controllers per project was:

2.2 in 2019
2.3 in 2017
2.1 in 2015
2.4 in 2014
2.4 in 2013

EETimes embedded 2019 Embedded Markets Study

My current embedded project's main processor is a:

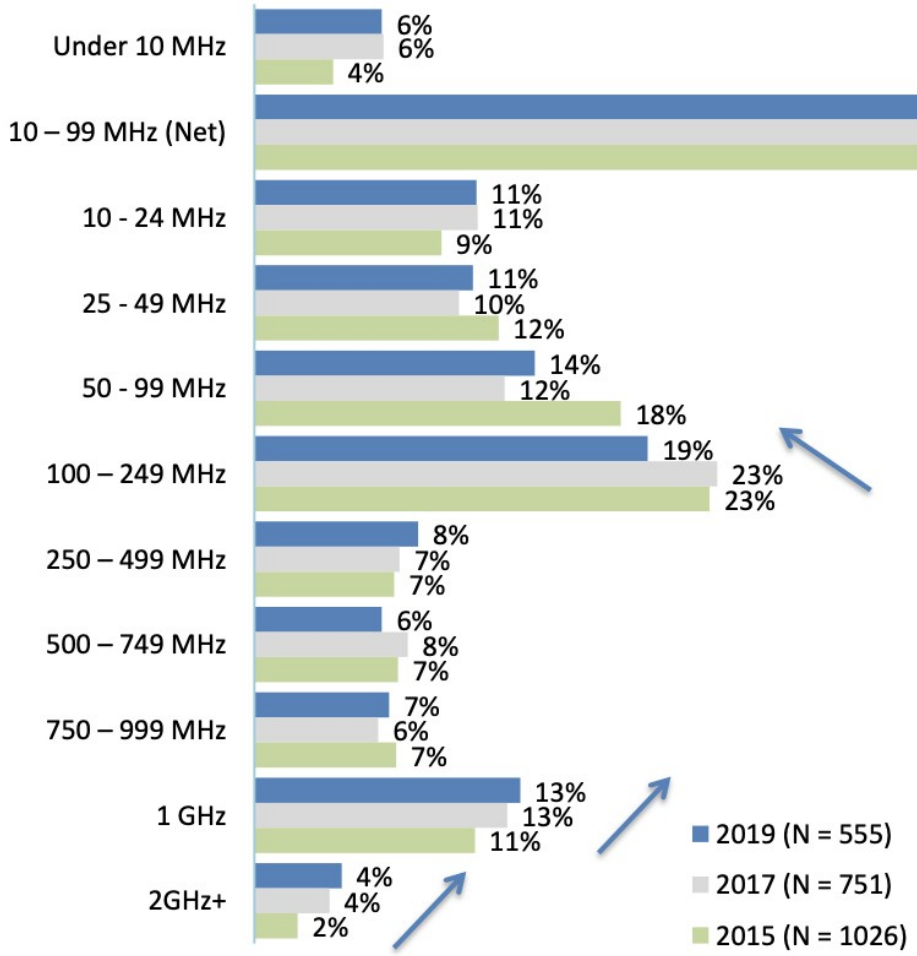


71% of **EMEA** users use 32-bit chips as their main processor.

Additional chips to the main processor	
Primarily 8-bit processors	19%
Primarily 16-bit processors	15%
Primarily 32-bit processors	55%
Primarily 64-bit processors	12%

EETimes embedded 2019 Embedded Markets Study

My current embedded project's main processor clock rate is:



The average processor clock rate was:
462 MHz in 2019
445 MHz in 2017
397 MHz in 2015
428 MHz in 2014

EETimes embedded

2019 Embedded Markets Study

Criterios generales de diseño

- Rendimiento (*performance*)
- Confiabilidad (*reliability*)
  - Misiones críticas (Critical Mission)
  - Riesgo de vida (Life-Threatening)
  - No hay usuario para rebootear ante una “pantalla azul”
- Disponibilidad (*availability*)
  - 24 hs /dia, 7 días/semana, 365 días/año
- Seguridad (*safety*)
  - Inocuo, no genera perjuicios para la vida o el ambiente.
- Seguridad (*security*)
  - Reducción de vulnerabilidades, protección contra amenazas que corren en software embebido

Otras consideraciones de diseño

- Consumo de energía (potencia)
  - ¿Cómo diseñamos para maximizar la duración de las baterías?
- Costo del sistema
  - Minimizar costo de fabricación (sobretodo cuando es producto de consumo, grandes volúmenes)
  - Mínima memoria, OS pequeño
- Balance de necesidades contrapuestas
  - Ejemplo: frecuencia de reloj
    - baja: para consumo
    - alta: para respuesta a eventos y procesamiento en ráfagas.

Sistemas embebidos: software

- Problemas diferentes a los presentes en PC.
  - Se debe:
    - responder a eventos externos
      - botones, lectura de sensores, etc.
    - resolver condiciones excepcionales sin intervención del usuario (M2M)
    - realizar varias tareas simultáneamente
    - cumplir con tiempos límites estrictos (deadlines)
    - no debe fallar jamás

Sistemas embebidos: software

- Se debe considerar.
  - Capacidad de procesamiento (*throughput*)
    - ¿El procesamiento es suficientemente rápido?
  - Tiempo de respuesta (*response time*)
    - ¿Se pude responder suficientemente rápido a los eventos?
  - “Testeable” (capacidad de prueba, *testability*)
    - ¿Se puede verificar que funciona bajo “toda” condición?
  - “Depurable” (*debugability*)
    - ¿Se pueden ubicar y resolver los errores de funcionamiento?
  - Confiable (*reliability*)
    - ¿Cubre las expectativas y necesidades de confiabilidad?

Ejemplos de sistemas embebidos

- Producto:
  Hunter Programmable Digital Thermostat.

- Microprocesador:
  4-bit.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplos de sistemas embebidos

- Producto:
  Vendo V-MAX 720
  vending machine.

- Microprocesador:
  8-bit Motorola 68HC11.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de sistemas embebidos

- Producto:

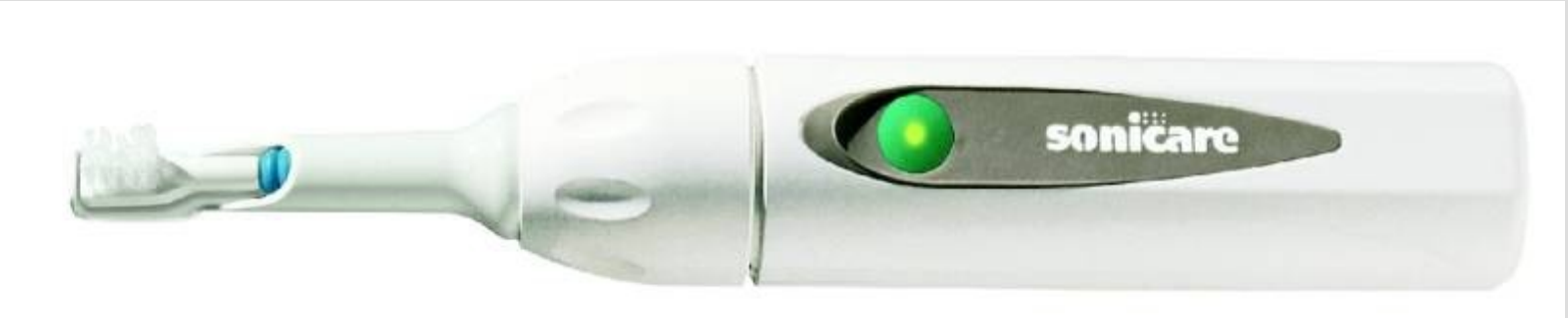
  CoinCo USQ-712 coin changer.

- Microprocesador:
  8-bit Motorola 68HC912.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplos de sistemas embebidos

- Producto:
  Sonicare Plus toothbrush.
- Microprocesador:
  8-bit Zilog Z8



Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de sistemas embebidos

- Producto:
  Miele dishwashers.

- Microprocesador:
  8-bit Motorola 68HC05

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de productos

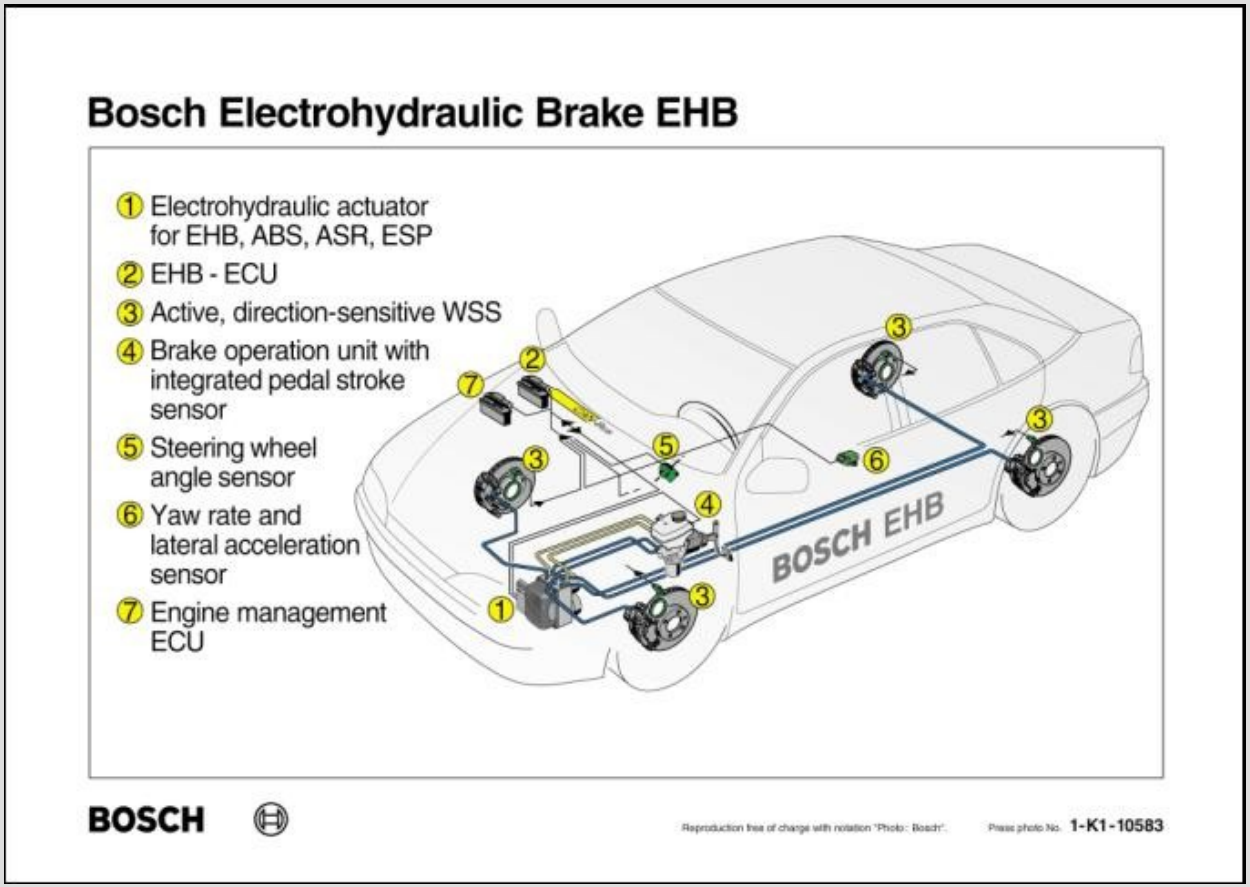
www.fitbit.com

- Incluyndop tecnologías de comunicación inalámbrica (Bluetooth low energy, wifi, etc.)

www.sticknfind.com

www.meterplug.com

Ejemplo de sistemas embebidos



Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de sistemas embebidos

- Producto: NASA's Mars Sojourner Rover.

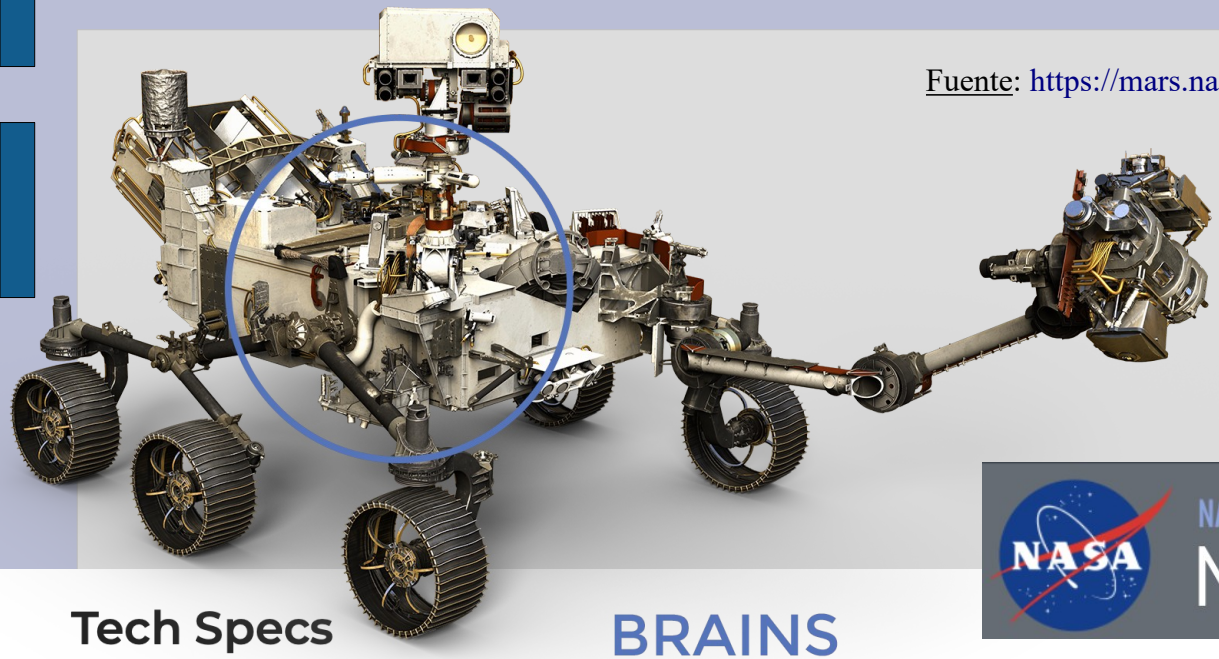
  “First wheeled vehicle to rove another planet”

- Microprocesador:

  8-bit Intel 80C85

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de sistemas embebidos



Fuente: https://mars.nasa.gov/mars2020/spacecraft/rover/brains/ , 1/3/2021

Microprocesador de 32 bits (2001)

NASA Science
MARS 2020 MISSION
PERSEVERANCE ROVER

Tech Specs

BRAINS

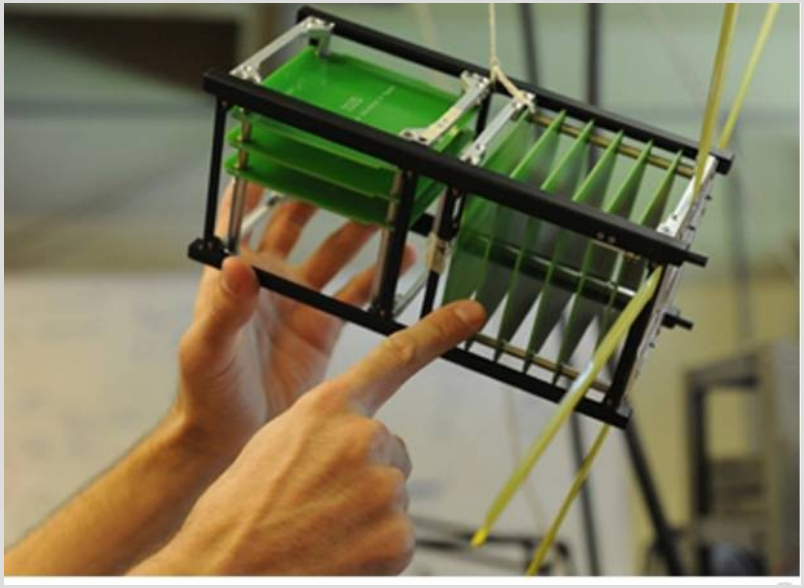
Processor	• Radiation-hardened central processor with PowerPC 750 Architecture: a BAE RAD 750 • Operates at up to 200 megahertz speed, 10 times the speed in Mars rovers Spirit and Opportunity's computers
Memory	• 2 gigabytes of flash memory (~8 times as much as Spirit or Opportunity) • 256 megabytes of dynamic random access memory • 256 kilobytes of electrically erasable programmable read-only memory

Ejemplo de sistemas embebidos

- Producto:
  GloboSat01 IIE (2008)

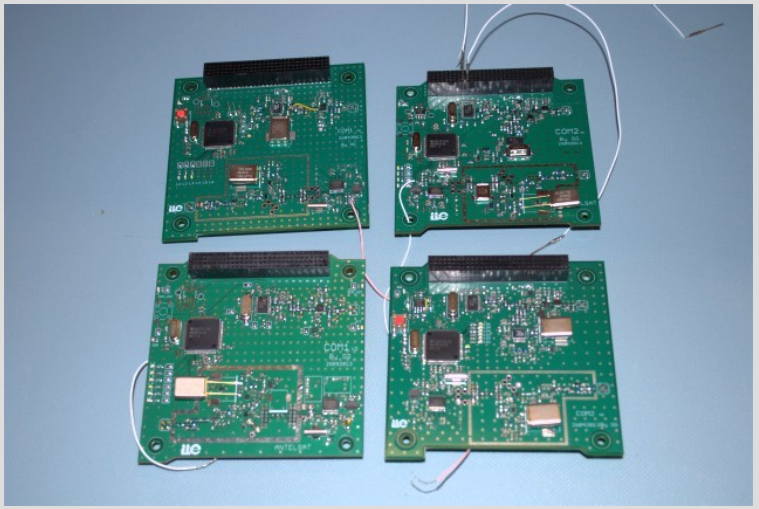
- Microprocesadores:
  - ATmega2560 (aplicación)
  - PIC16F877A (telemetría)

Ejemplo de sistemas embebidos



- Producto:
  AntelSat (IIE + ANTEL)

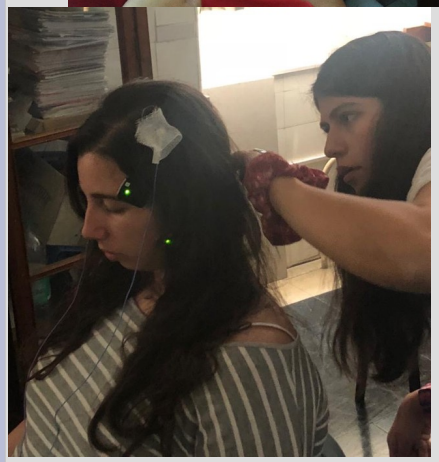
- Microprocesadores:
  - MSP430



Ejemplo de sistemas embebidos

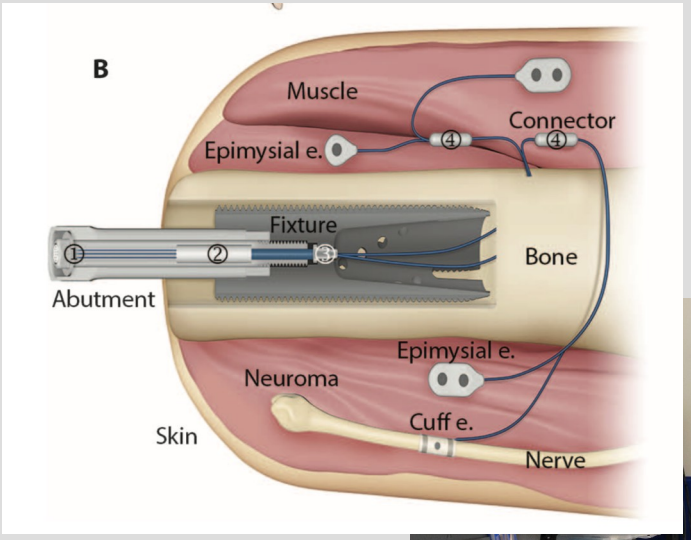
EEG inalámbrico

Causa et al, LASCAS 2018

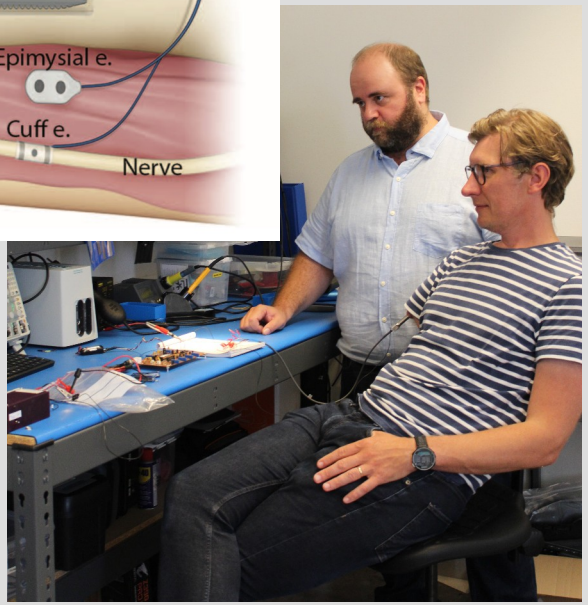


Cabrera et al, SABI 2020

Control de prótesis neurales



Ortiz-Catalan et al, ST MED 2014



Ejemplo de sistemas embebidos

- Algunos proyectos Sisem de años anteriores:
  - Calentador de agua inteligente (SmartSUN, 2019)
  - Monitoreo y control de una estufa de leña de alto rendimiento (2019)
  - Juego de la viborita en pantallita OLED (CCuP, 2019)
  - Periférico Basado En Sensor Capacitivo (TouchCap, 2020)
  - Determinación de la dinámica espacial de una Oveja (2020
  - Sistema de Invernadero Automatizado (SINVA, 2020)
  - Adquisidor de señales de signos vitales (2020)
  - Mouse controlado por gestos (GesturesMouse, 2019)
  - Medida de la impedancia transtorácica en pacientes pediátricos (2019)
- Ver detalles en página web del curso.

Actividad en grupo

- ¿Qué es un “Sistema de tiempo real”?
  - Objetivo:
    - Reflexionar sobre los sistemas con características “tiempo real” y aquellos que no las tienen.
    - Escribir: Definición (tentativa) / Características
  - Grupos:
    - 3 a 4 participantes
  - Tiempo:
    - 5 minutos
  - Puesta en común.

¿Que es un sistema de tiempo real?

- Definición:
  - Un sistema de tiempo real es aquel en el que la corrección del resultado depende tanto de su **validez lógica** como del **instante** en que se produce.
- Características típicas:
  - Sistema reactivo
    - el sistema reacciona ante eventos externos
    - los eventos detectados en las entradas provocan otros eventos como salidas.
  - Restricciones de tiempo
- Atención: tiempo real ≠ rápido.

¿Que es un sistema de tiempo real?

- Definición:
  - "Un sistema de tiempo real es un sistema que debe satisfacer **tiempos de respuestas explícitos**, bajo riesgo de consecuencias graves, incluyendo el fallo."
    - Si un tiempo limite no se cumple, hay consecuencias.
- Ejemplos:
    - Sistemas de gestión y comando (tráfico aéreo, espacial, planta nuclear, industrial).
    - Sistemas multimedia en red. Por ejemplo: degradación de audio/video en transmisión en vivo de un partido de fútbol.
- Clasificación para diferenciar qué tan críticos son los tiempos límite.

Clasificación

- Sistema de tiempo real “blando” (Soft Real-Time System)
  - Perder un deadline produce la degradación del sistema
  - Sistemas de transacciones on line (pagos, reservas), transmisición de audio/video.
- Sistema de tiempo real “duro” (Hard Real-Time System)
  - Perder un deadline puede producir la falla del sistema
  - Sistemas de gestión y comando: tráfico aéreo, espacial, planta nuclear, aplicaciones militares, automóviles.

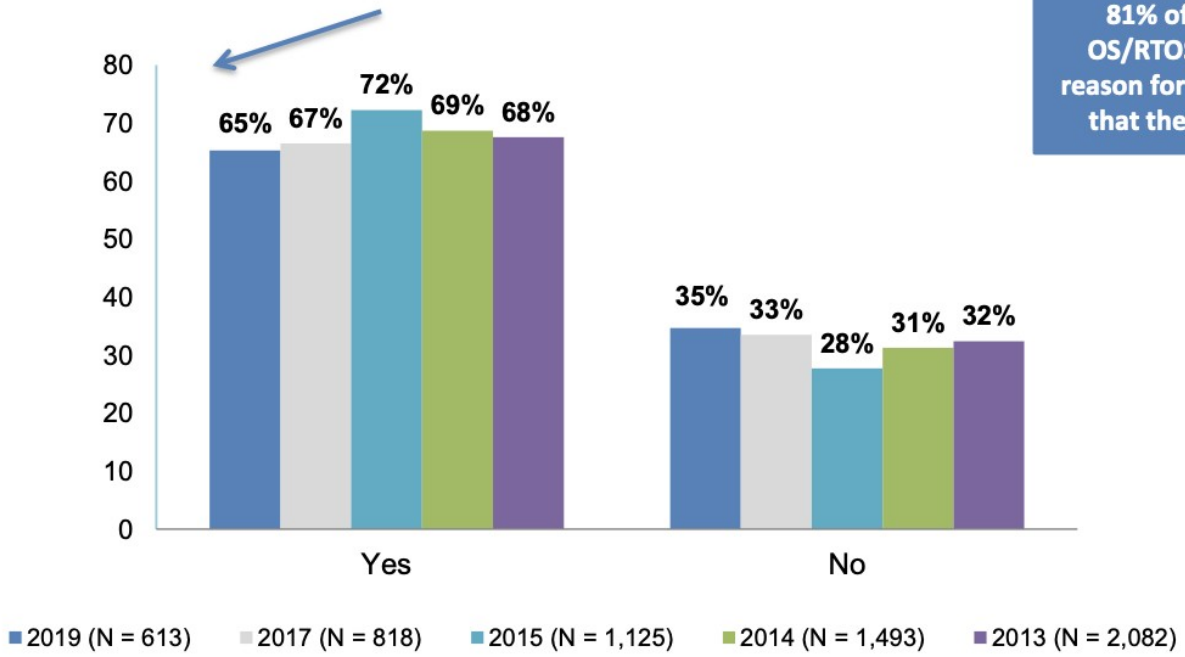
Sistemas de Tiempo Real

- ¿Todos los sistemas embebidos son de tiempo real? No.
- La previsibilidad en su funcionamiento es lo más importante en sistemas de tiempo real.
- Enfoque más radical: hago una sola cosa a la vez. No siempre se puede. ¿Entonces?

¿Qué significa multitarea?

- Sistemas embebidos: en general tienen varias I/O que deben responder a eventos independientes (asíncronos) y un solo core (microprocesador)
- Separar el código en tareas simplifica la programación, pero es necesario un mecanismo para conmutar entre tareas.
- *Concurrencia* es la apariencia de la ejecución simultánea de múltiples tareas.
- Veremos durante el curso diferentes arquitecturas: desde simples hasta RTOS (Real Time Operating Systems).

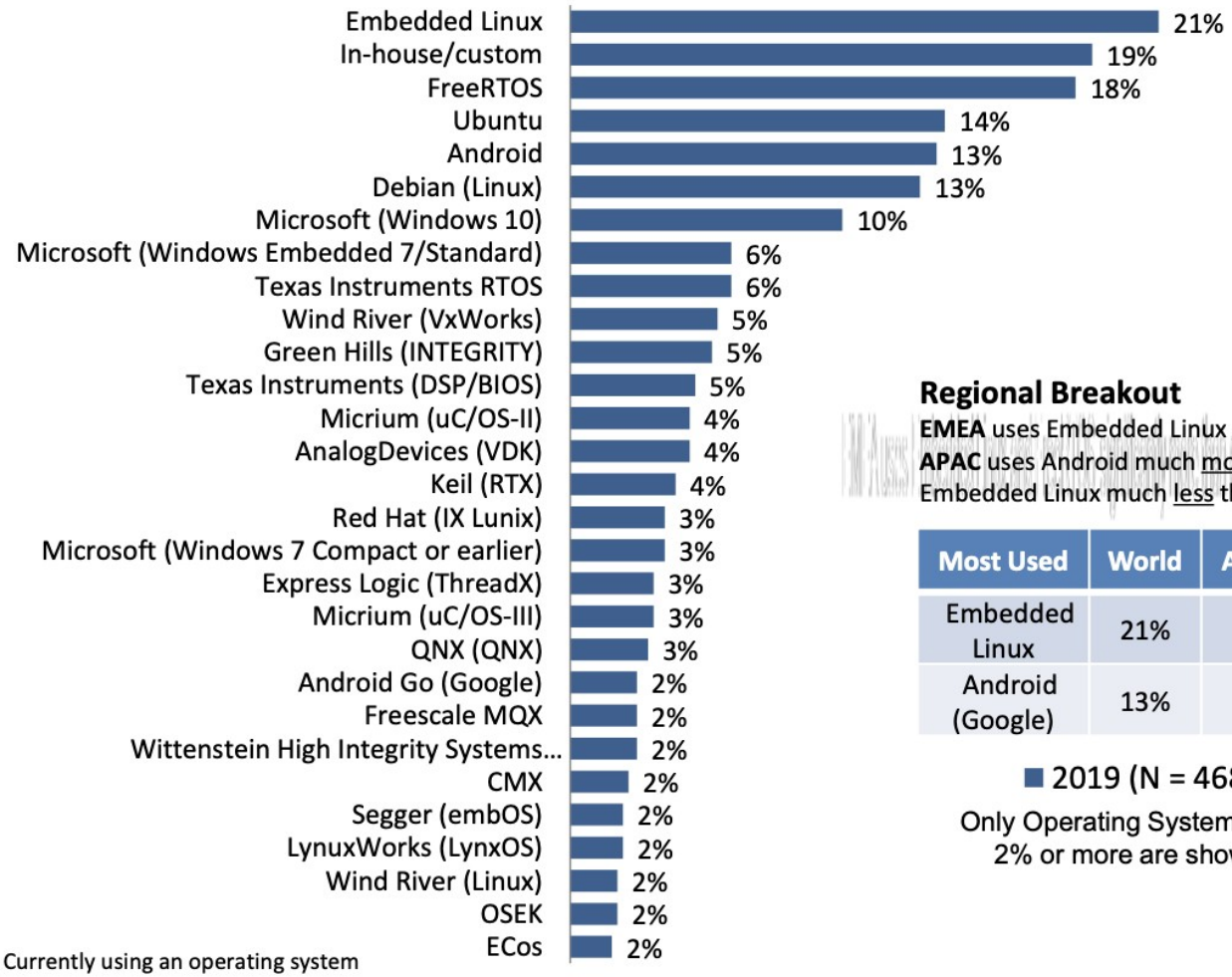
Does your current embedded project use an operating system, RTOS, kernel, software executive, or scheduler of any kind?



81% of those not using OS/RTOSes, said the main reason for NOT using is simply that they are not needed.

EETimes embedded 2019 Embedded Markets Study

Please select ALL of the operating systems you are currently using.



Base: Currently using an operating system

Regional Breakout

EMEA uses Embedded Linux much more than other regions.
APAC uses Android much more than other regions and uses Embedded Linux much less that others.

Most Used	World	Americas	EMEA	APAC
Embedded Linux	21%	21%	30%	15%
Android (Google)	13%	9%	14%	27%

2019 (N = 468)

Only Operating Systems with 2% or more are shown.

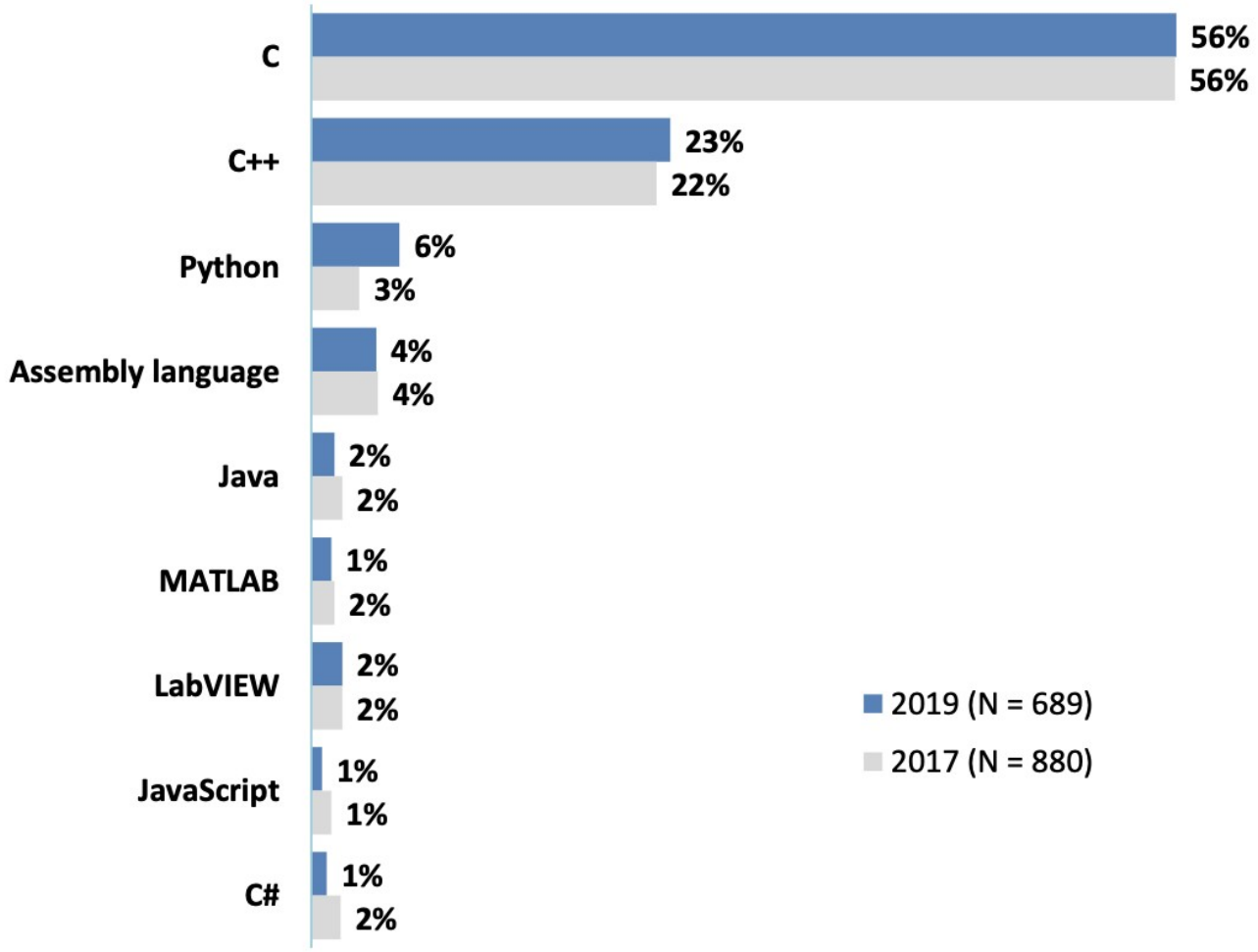
EETimes embedded

2019 Embedded Markets Study

El lenguaje C

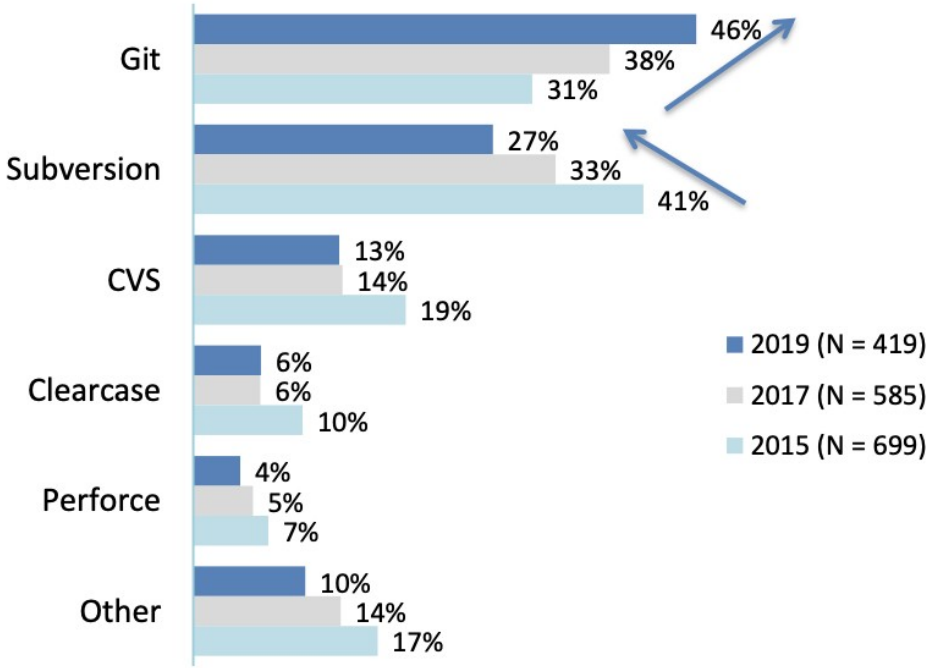
- Es la *lingua franca* de los sistemas embebidos
  - Compiladores C disponibles para todos los uC
  - C es suficientemente de bajo nivel
- C fundamental para el curso
  - Laboratorios hechos en C
  - El texto asume “conocimientos de lectura”
- C es razonablemente portable, pero no totalmente
- Otros:
  - C++ es más complicado y no universalmente soportado
  - EC y EC++

My *current* embedded project is programmed mostly in:



EETimes embedded 2019 Embedded Markets Study

Which of the following Version Control software systems do you currently use?



EETimes embedded 2019 Embedded Markets Study

Bibliografía

- “An Embedded Software Primer”, David E. Simon
  - Chapter 1: A First Look ate Embedded Systems
- Real-Time Embedded Systems, Xiaocong Fan
  - Chapter 1: Introduction to Embedded and Real-Time Systems
- "Embedded market study", Aspencore, 2019.

El tiempo pasa....

Ejemplo de productos

- Producto:
  Palm Vx handheld.

- Microprocesador:
  32-bit Motorola
  Dragonball EZ.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

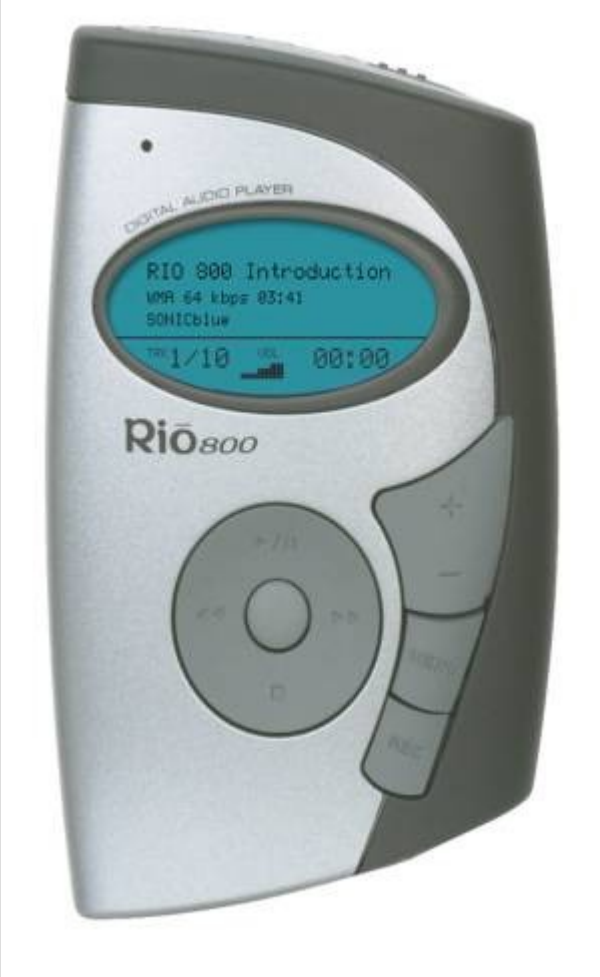
Ejemplo de productos

- Producto:
  Motorola i1000plus iDEN Multi-Service Digital Phone.

- Microprocesador:
  Motorola 32-bit MCORE.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de productos



- Producto:
  Rio 800 MP3 Player.

- Microprocesador:
  32-bit RISC.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de productos

- Producto:
  RCA RC5400P DVD player.

- Microprocesador:
  32-bit RISC.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de productos

- Producto:
  IBM Research's Linux wrist watch prototype.

- Microprocesador:
  32-bit ARM RISC.

Fuente: Daniel W. Lewis, "Fundamentals of Embedded Software"

Ejemplo de sistemas embebidos

- Producto:
  Garmin StreetPilot GPS Receiver.

- Microprocesador:
  16-bit.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de sistemas embebidos

- Producto:
  Sony Aibo ERS-110
  Robotic Dog.

  Microprocesador:
  64-bit MIPS RISC.

Fuente: Daniel W. Lewis, “Fundamentals of Embedded Software”

Ejemplo de sistemas embebidos

- Producto:
  SiAgro2 (2007)
- Microprocesador: MSP430