Lecture 02
Microcontrollers

CE346 – Microprocessor System Design

Branden Ghena – Spring 2021

Some slides borrowed from:
Josiah Hester (Northwestern), Prabal Dutta (UC Berkeley)

Northwestern

Class Updates

- Reminder: fill out lab hardware survey ASAP
  - The link is on Campuswire
- Starting orders after class
  - Let me know when people start receiving stuff
- My office hours 1-2pm today
  - Feel free to swing by

Today's Goals

- Explore microcontrollers
  - Their purpose
  - Their capabilities
  - Design tradeoffs
- Describe history and state-of-the-art for microcontrollers
- Explore the microcontroller(s) on our Microbits

Outline

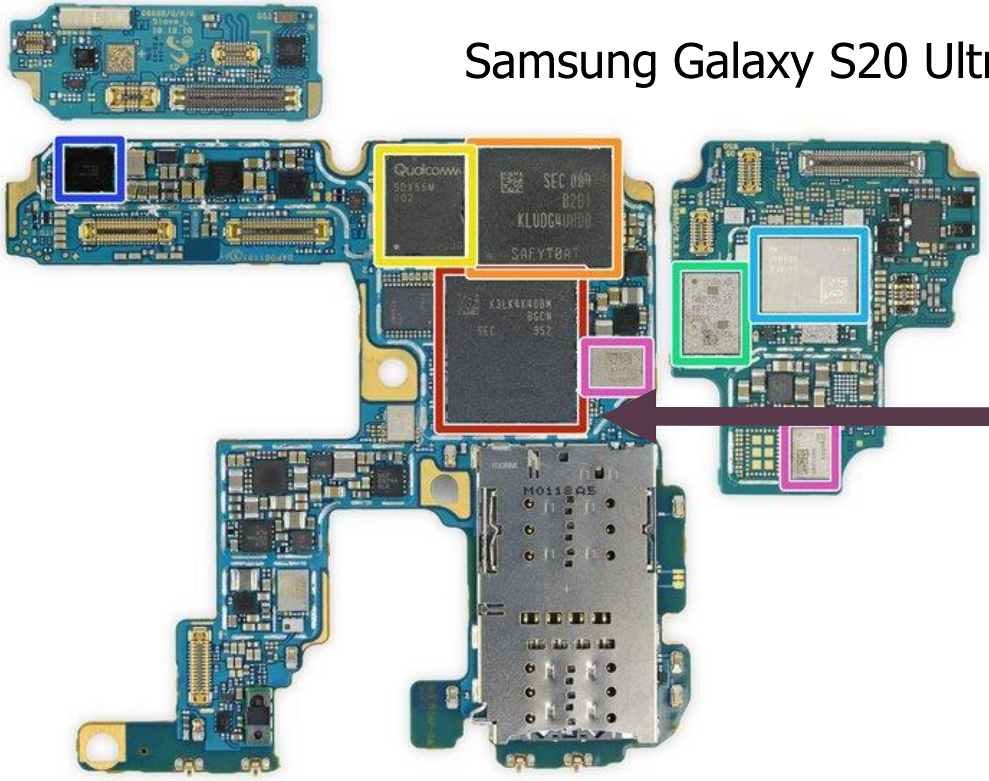
- **Microcontroller Background**
- Microcontroller Design
- Microcontroller History

- Microbit microcontroller

Source: iFixit

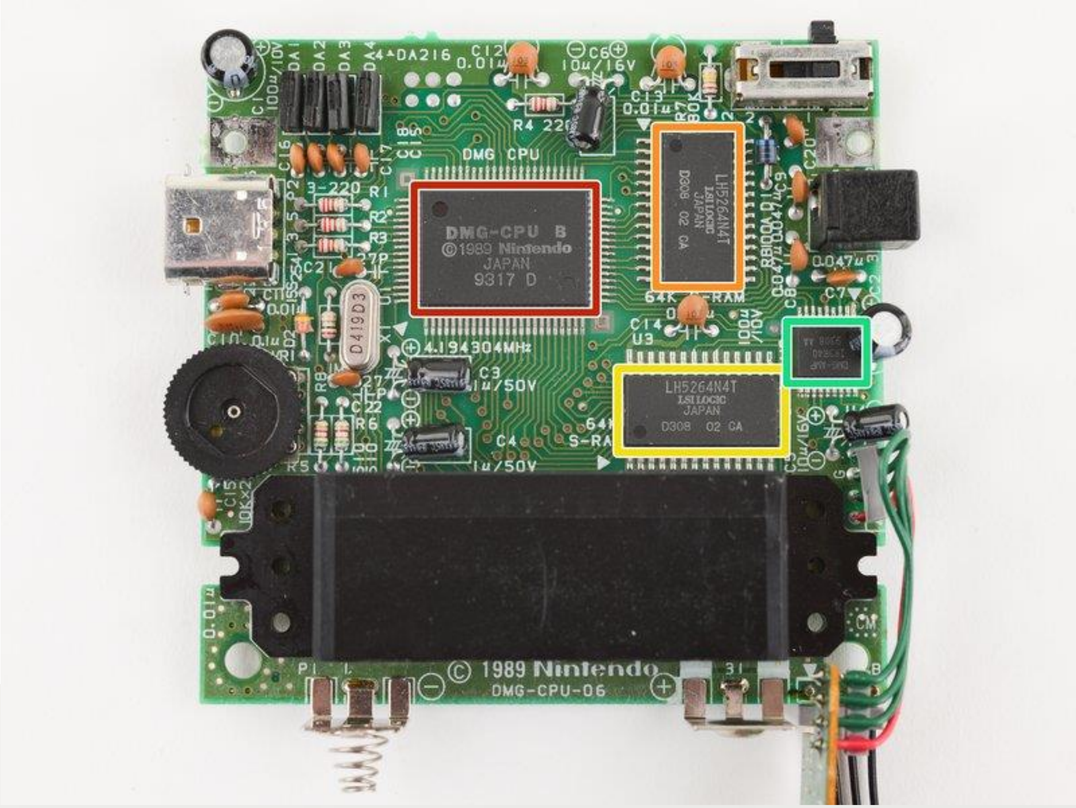
Source: iFixit

Samsung Galaxy S20 Ultra



Samsung K3LK4K40BM-BGCN 12 GB LPDDR5

RAM layered over Qualcomm 865 SoC



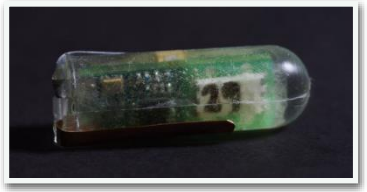
Source: iFixit

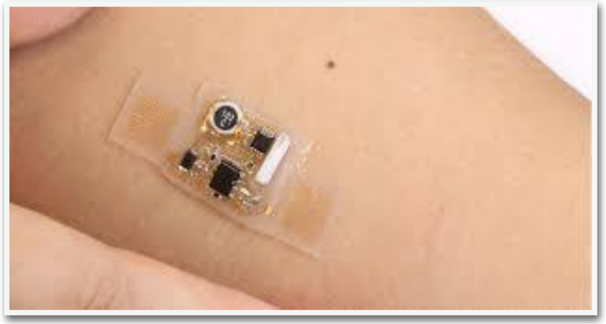
LR35902 4.19 Mhz CPU

…Z80 Microprocessor

The Nintendo Game Boy was initially released in Japan on April 21, 1989

...computing everywhere!





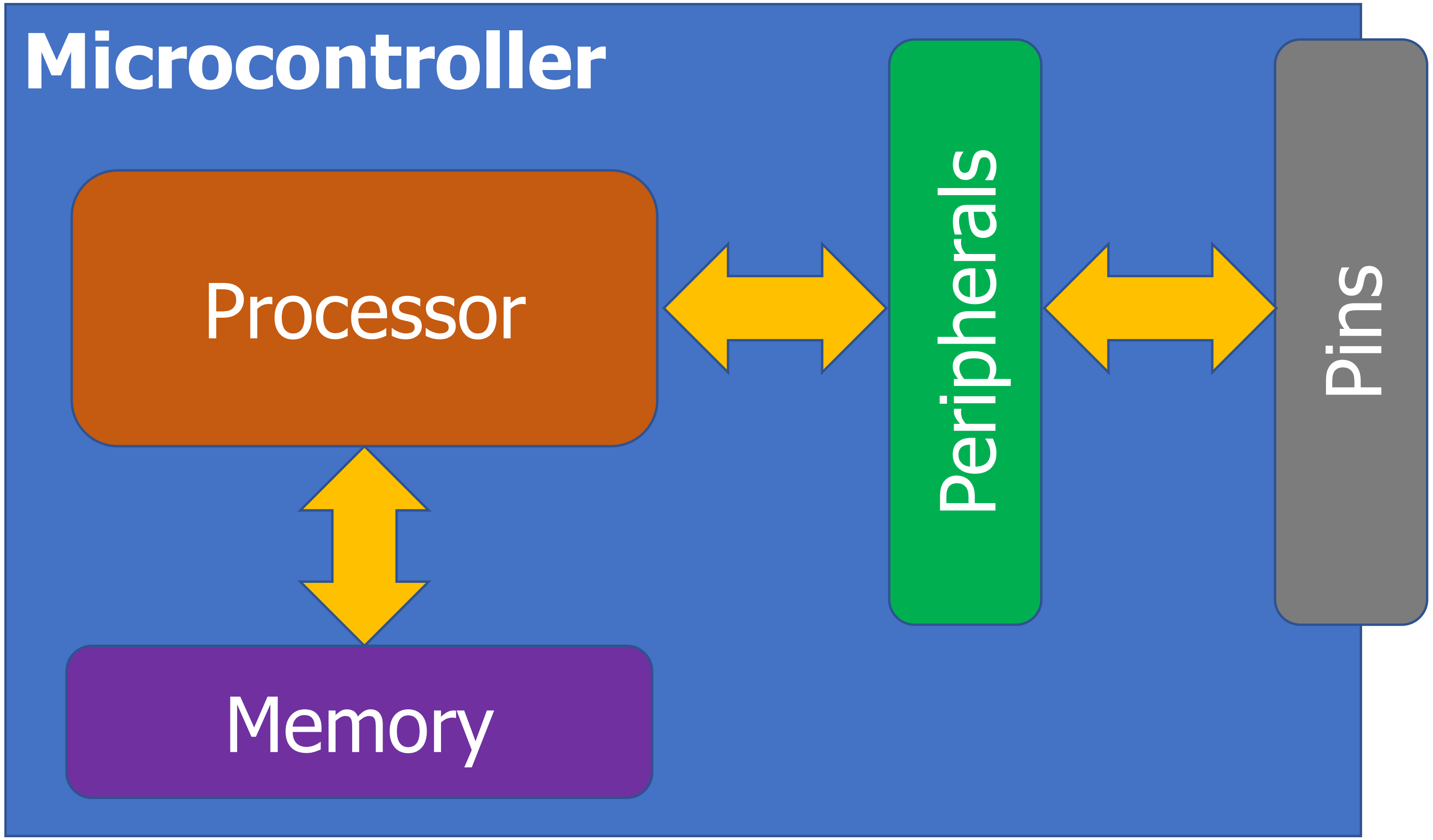
Categories of computer chips

- Microprocessor (MPU or Processor)
  - CPU + Bus/Pins
  - No memory or peripherals
- Microcontroller (MCU)
  - CPU + Pins + Memory + Peripherals
  - Peripherals: separate hardware units within chip
    - Often I/O interfaces
- System on a Chip (SoC)
  - Microcontroller + Extensive Peripherals
  - Example: Radios or ML Accelerators
  - Essentially multiple chips combined
- Evolution of increased complexity over time

Outline

- Microcontroller Background
- **Microcontroller Design**
- Microcontroller History
- Microbit microcontroller

Generic microcontroller block diagram



Processor

- Microcontroller needs to execute software instructions

- Emphasis: simple and reliable
  - i.e. not x86 CISC

- Often simple, in-order, short-pipeline RISC architectures
  - Literally a three-stage pipeline you likely learned in CS361

- Many different ISAs are possible
  - Custom ISAs and ARM (ARMv7E-M) are both common

Reminder: Instruction Set Architecture

- ISA includes the actual instructions as well as the model for how the computer interacts with memory

- **Does the ISA for a microcontroller matter if you're not programming in assembly?**

Reminder: Instruction Set Architecture

- ISA includes the actual instructions as well as the model for how the computer interacts with memory

- **Does the ISA for a microcontroller matter if you’re not programming in assembly?**
    - **Yes**
    - **Differences in instruction efficiency and amount are important**
    - **Differences in compiler support are VERY important**

Processor design choices

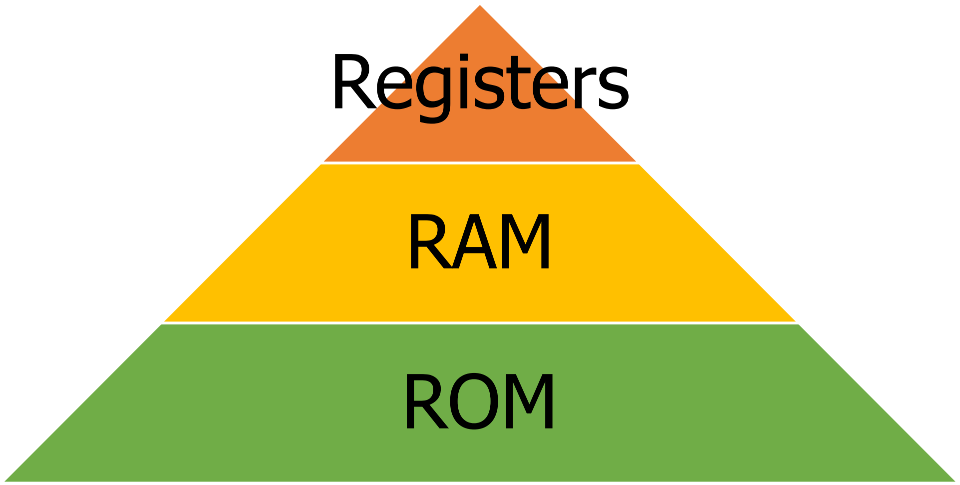
- How many bits is the architecture?
  - 8-bit and 16-bit not uncommon
  - 64-bit is very rare (who has that much memory?)

- What instructions are supported?
  - Normal stuff yes
  - What about single-cycle multiply?
  - What about floating-point operations?
  - Vector instructions?

- How fast does it run?
  - 1-100 MHz is common on modern systems (faster is more energy cost)
  - Occasionally MUCH slower (think 32 KHz)

Memory

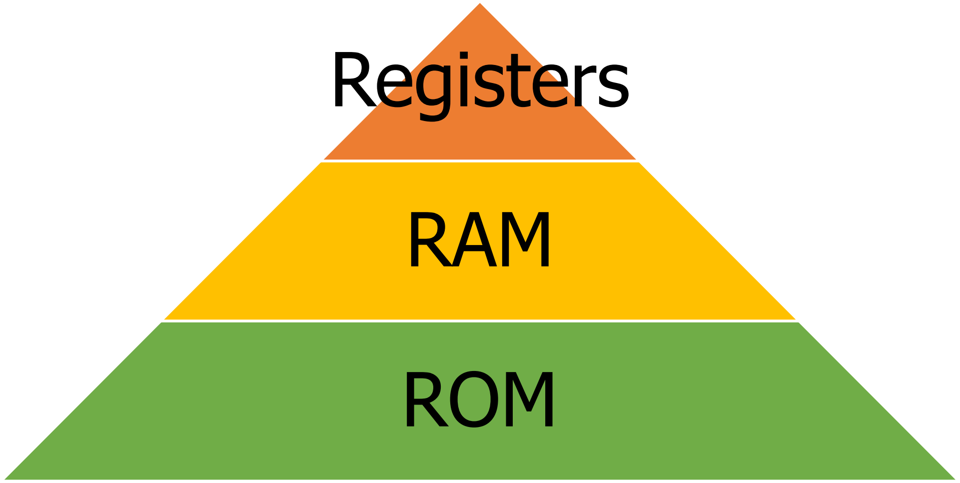
- Common memory hierarchy



- **What’s missing? Why?**

Memory

- Common memory hierarchy



- **What's missing? Why?**
  - **Cache**
  - **Makes timing unreliable**

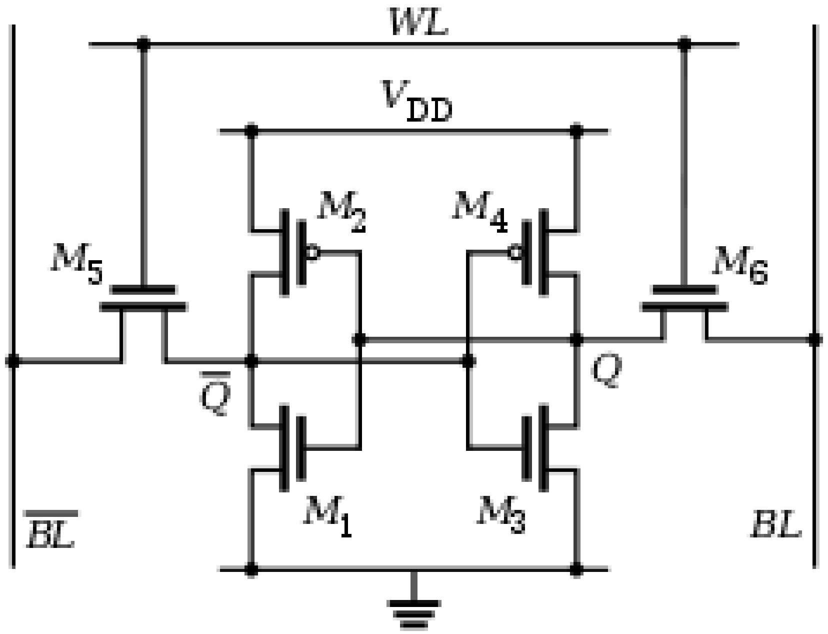
More modern microcontrollers often have optional instruction caches

Can be enabled by software

Volatile memory: SRAM

- Same design as registers in traditional computer systems
  - No need to refresh it -> lower energy costs

- Tradeoff:
  - More expensive
  - Physically larger



Non-Volatile memory: Flash (usually)

- Same design as SSDs and Flash drives

- Memory is stored with no energy costs
  - Reading is lowish energy and quick
  - Writing is high energy and very slow
    - Writing also has to occur in blocks (e.g. 512 bytes)

- Concern: Flash has a limited lifetime ~10,000 writes
  - Only writing to Flash when you load code? Essentially forever
  - Recording data to Flash every second? 2.8 hours

SRAM (RAM) versus Flash (ROM) memory

- Size and time difference between non-volatile and volatile memory is *significantly* reduced from traditional computing
  - Non-volatile store 2-10x larger than volatile
  - Single-cycle RAM. Maybe two-cycles to Flash (to read)

- Major difference: energy and writability
  - SRAM is low-energy to read and write (no refresh needed)
  - Flash is lowish-energy to read, but very high energy to write

- Hierarchical structure is not enforced
  - Same address space for RAM and Flash (very different from traditional)
  - Run instructions right inside of Flash
  - Keep variables in RAM (or `const` variables in Flash)

Memory design choices

- How much memory can we fit without making it too high cost or too high power?
  - The answer has slowly increased over time
  - 15 years ago: 2 KB RAM – 32 KB Flash
  - Today: 256 KB RAM – 1 MB Flash

- How to provide memory safety?
  - Usually no virtual memory (all addresses are real addresses)
  - Nothing stops arbitrary memory overwriting
  - Modern systems: Memory Protection Unit
    - Specify a small number of ranges and permissions

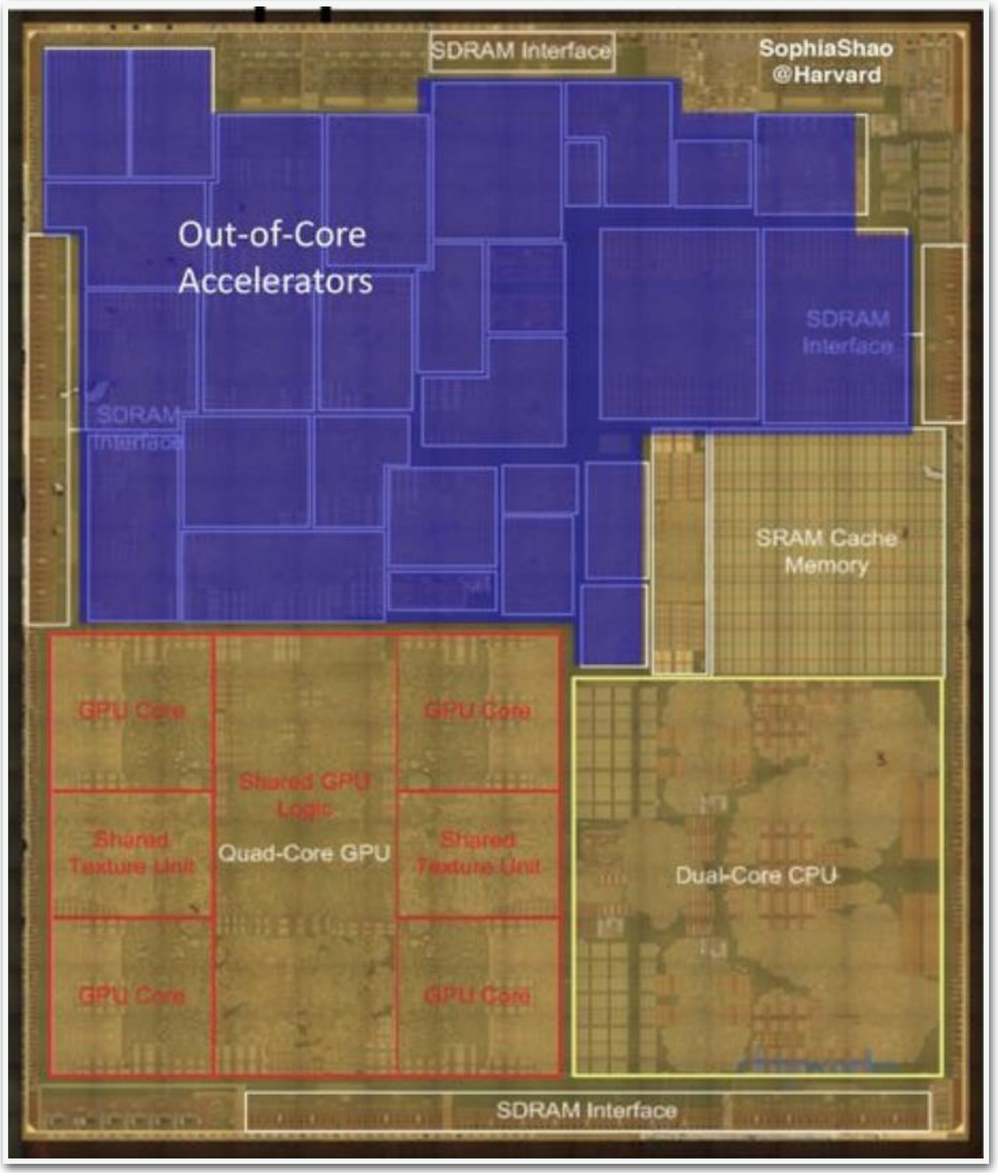
Peripherals

- Hardware modules the perform some action
- Common examples
  - Control digital input and output pins
  - Read analog inputs
  - Send messages in a given wire protocol (UART, SPI, I2C)
  - Set and check timers
- Less common examples
  - Cryptography accelerators
  - Complicated wire protocols (USB, CAN)
  - Wireless radios (BLE, 802.15.4, WiFi)
- We'll spend most of class learning various peripherals

Peripheral design choices

- More peripherals mean more use cases
  - But also means more cost for the chip
  - Peripherals occupy silicon
  - Most peripherals will go unused for a given application…

- Flexibility within a given peripheral
  - One capability is easier to use
  - Many capabilities are more useful

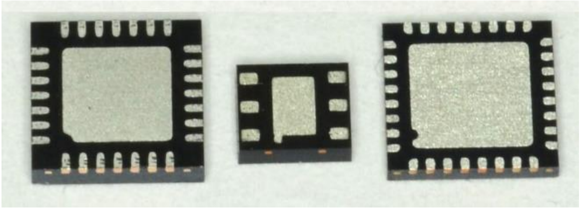


Apple A8 SoC

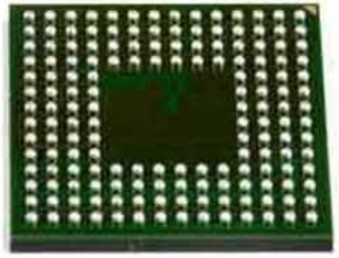
Pins

- Peripherals need to electrically connect to outside world
  - Attach to pins on the exterior of the chip

- More pins allow for more connections
  - At increased cost and size



- Pin layouts can add more pins for cheaper
  - But make soldering and debugging difficult



Internal connections to pins

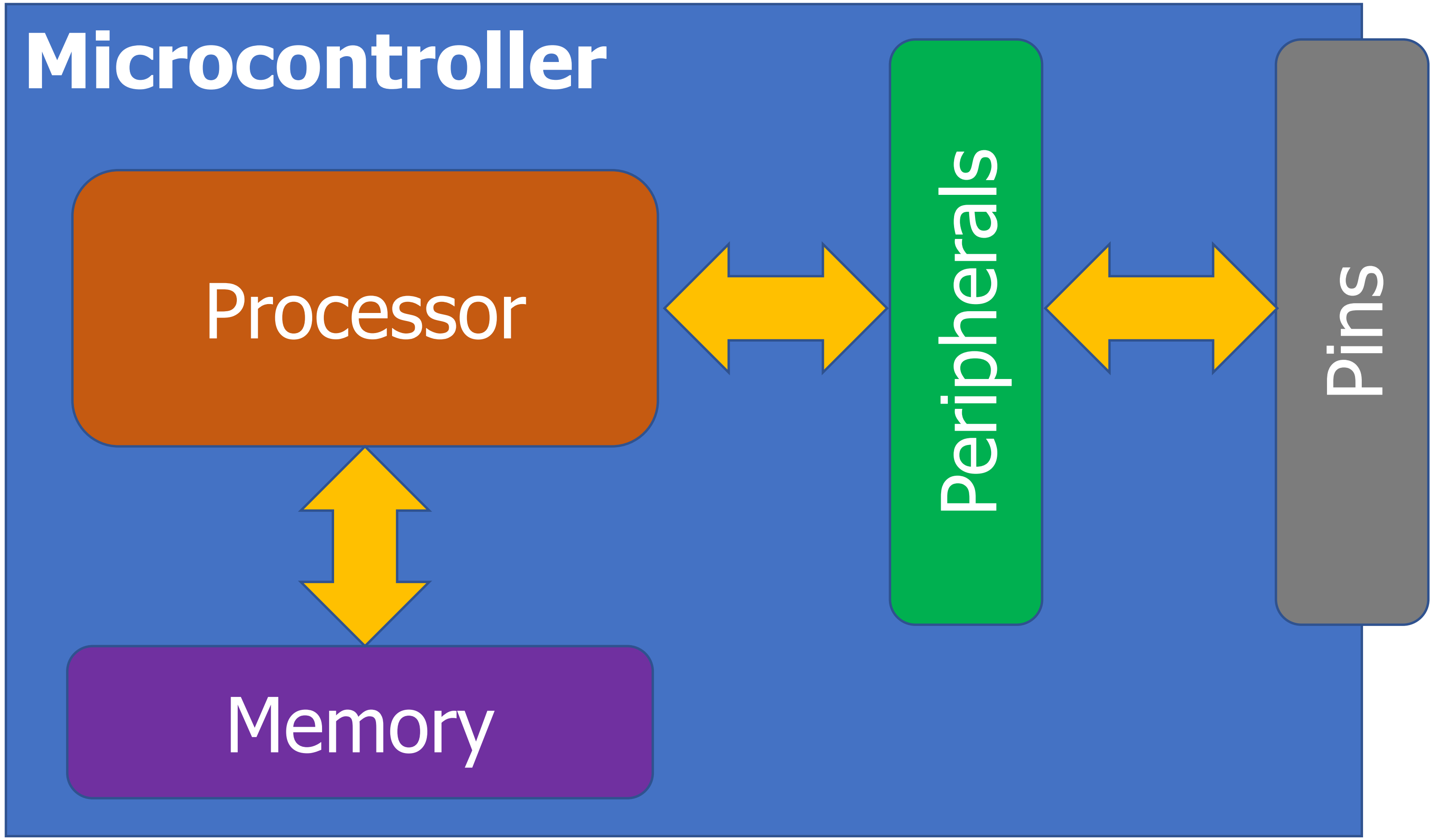
- Peripherals need to connect to external pins
  - Can any peripheral connect to any pin, or are there limited mappings?
  - Modern microcontrollers allow any-to-any connections

- Older MCUs had mapping tables and pin selection was more challenging

Table 3-1. 100-pin GPIO Controller Function Multiplexing (Sheet 1 of 4)

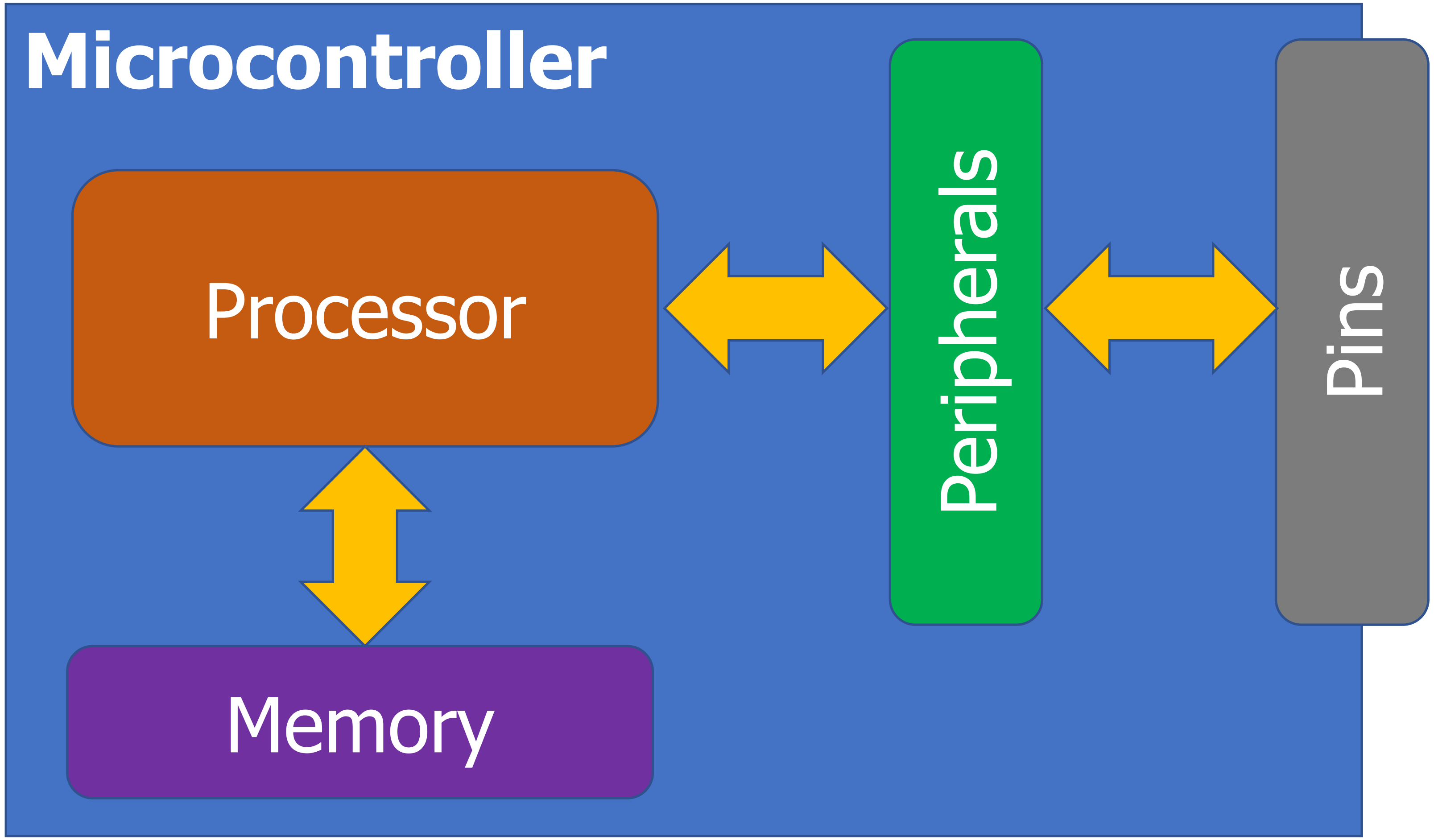
ATSAM4LC		ATSAM4LS		Pin	GPIO	Supply	GPIO Functions						
QFN	VFBGA	QFN	VFBGA				A	B	C	D	E	F	G
5	B9	5	B9	PA00	0	VDDIO							
6	B8	6	B8	PA01	1	VDDIO							
12	A7	12	A7	PA02	2	VDDIN	SCIF GCLK0	SPI NPCS0					CATB DIS
19	B3	19	B3	PA03	3	VDDIN		SPI MISO					
24	A2	24	A2	PA04	4	VDDANA	ADCIFE AD0	USART0 CLK	EIC EXTINT2	GLOC IN1			CATB SENSE0
25	A1	25	A1	PA05	5	VDDANA	ADCIFE AD1	USART0 RXD	EIC EXTINT3	GLOC IN2	ADCIFE TRIGGER		CATB SENSE1
30	C3	30	C3	PA06	6	VDDANA	DACC VOUT	USART0 RTS	EIC EXTINT1	GLOC IN0	ACIFC ACAN0		CATB SENSE2

What haven’t we talked about?



What haven’t we talked about?

- **Power**
- **Programming**
- **Others?**
  - Clocks
  - Antennas



Powering microcontrollers

- Usually require a specific voltage
  - E.g. 5 volts, 3.3 volts, 1.8 volts
  - Must be stable and supply enough current (or MCU "browns out")
  - Noisy power supply can be an issue

- Some microcontrollers have wider ranges of acceptable voltages
  - Need to pay attention to acceptable range on I/O though

Programming microcontrollers

- JTAG (Joint Test Action Group)
  - Hardware built into the microcontroller for testing purposes
  - Can arbitrarily read/write memory
  - Can single step process too, at runtime!
    - GDB can connect to it! (sort of)

- Bootloaders
  - Software runs on the microcontroller at boot that waits a short time for someone to contact it and upload code
    - Via I/O pins
  - Convenient, but sometimes unreliable

Outline

- Microcontroller Background
- Microcontroller Design
- **Microcontroller History**
- Microbit microcontroller

Older microcontrollers (90s-00s)

- Focus: cheap, small computer systems

- PIC
  - 8, 16, and 24-bit MIPs architecture
  - PICAXE available with Basic interpreter

- AVR
  - 8-bit custom architecture (AVR architecture)
  - Used in Arduinos
  - AT Tiny 4: $0.30 per unit
    - 4 I/O pins, 32 Bytes RAM, 512 Bytes ROM

More capable microcontrollers (00s-10s)

- Focus: diversify into extreme low power and more capable systems

- MSP430
  - 16-bit custom architecture (MSP430 architecture)
  - Capable, but also extremely low power
    - <1 μA sleep current

- STM32, Atmel SAM series (Cortex-M0, M3, M4, M4F)
  - 32-bit ARM architecture (ARMv7)
    - Leverage success of ARM on smartphones
  - Every peripheral under the sun. Plus a variety of memory choices

Cortex M Series

- A number of options for increasing capabilities
- In practice:
  - Cortex-M0+ for low-end systems
  - Coretx-M4 for high capability systems

ARM Cortex-M instruction variations

Arm Core	Cortex M0[2]	Cortex M0+[3]	Cortex M1[4]	Cortex M3[5]	Cortex M4[6]	Cortex M7[7]
ARM architecture	ARMv6-M[9]	ARMv6-M[9]	ARMv6-M[9]	ARMv7-M[10]	ARMv7E-M[10]	ARMv7E-M[10]
Computer architecture	Von Neumann	Von Neumann	Von Neumann	Harvard	Harvard	Harvard
Instruction pipeline	3 stages	2 stages	3 stages	3 stages	3 stages	6 stages
Thumb-1 instructions	Most	Most	Most	Entire	Entire	Entire
Thumb-2 instructions	Some	Some	Some	Entire	Entire	Entire
Multiply instructions 32x32 = 32-bit result	Yes	Yes	Yes	Yes	Yes	Yes
Multiply instructions 32x32 = 64-bit result	No	No	No	Yes	Yes	Yes
Divide instructions 32/32 = 32-bit quotient	No	No	No	Yes	Yes	Yes
Saturated instructions	No	No	No	Some	Yes	Yes
DSP instructions	No	No	No	No	Yes	Yes
Single-Precision (SP) Floating-point instructions	No	No	No	No	Optional	Optional
Double-Precision (DP) Floating-point instructions	No	No	No	No	No	Optional

Modern system-on-chips (10s-?)

- Focus: increase memory and capability
  - Include radios in the same chip

- nRF51 and nRF52 series
  - 32-bit ARM microcontrollers (Cortex M)
  - Include 2.4 GHz radio: Bluetooth Low Energy and 802.15.4/Thread

- Others have followed this same path
  - TI CC26XX
  - Apollo3
  - STM32WL

Future microcontrollers

- Multi-core systems
  - Not for performance, but for separation of concerns
    - Run radio code on one core
    - Application code goes on the other core
  - Also allows BIG.little architecture
    - Higher performance core when interpreting data
    - Lower performance core for sampling sensors
  - nRF53 series is already doing this
    - But support is still nascent

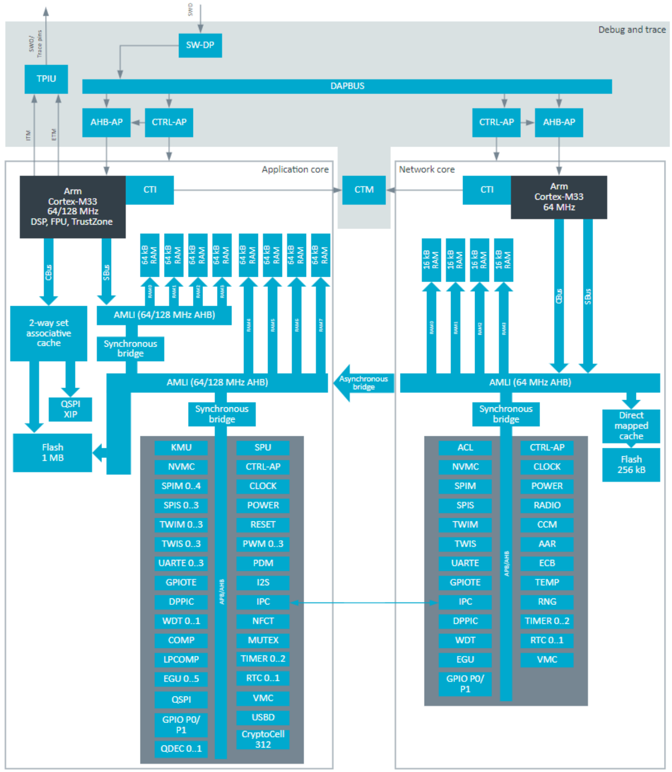
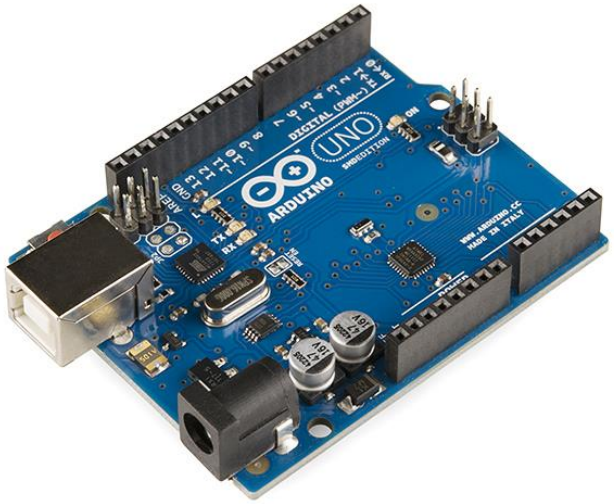


Figure 1. Simplified block diagram

Popular components rarely die



- Majority of popular older options still exist
  - Designers can trade off cost, capability, and efficiency alongside "modern features"
- Upgrading for an existing product is unlikely
  - Large cost, little benefit

- Example: Arduino still primarily uses AVR
  - Works just fine for its needs
  - Also releases new boards with nRF52s (Arduino Nano 33 BLE)

Outline

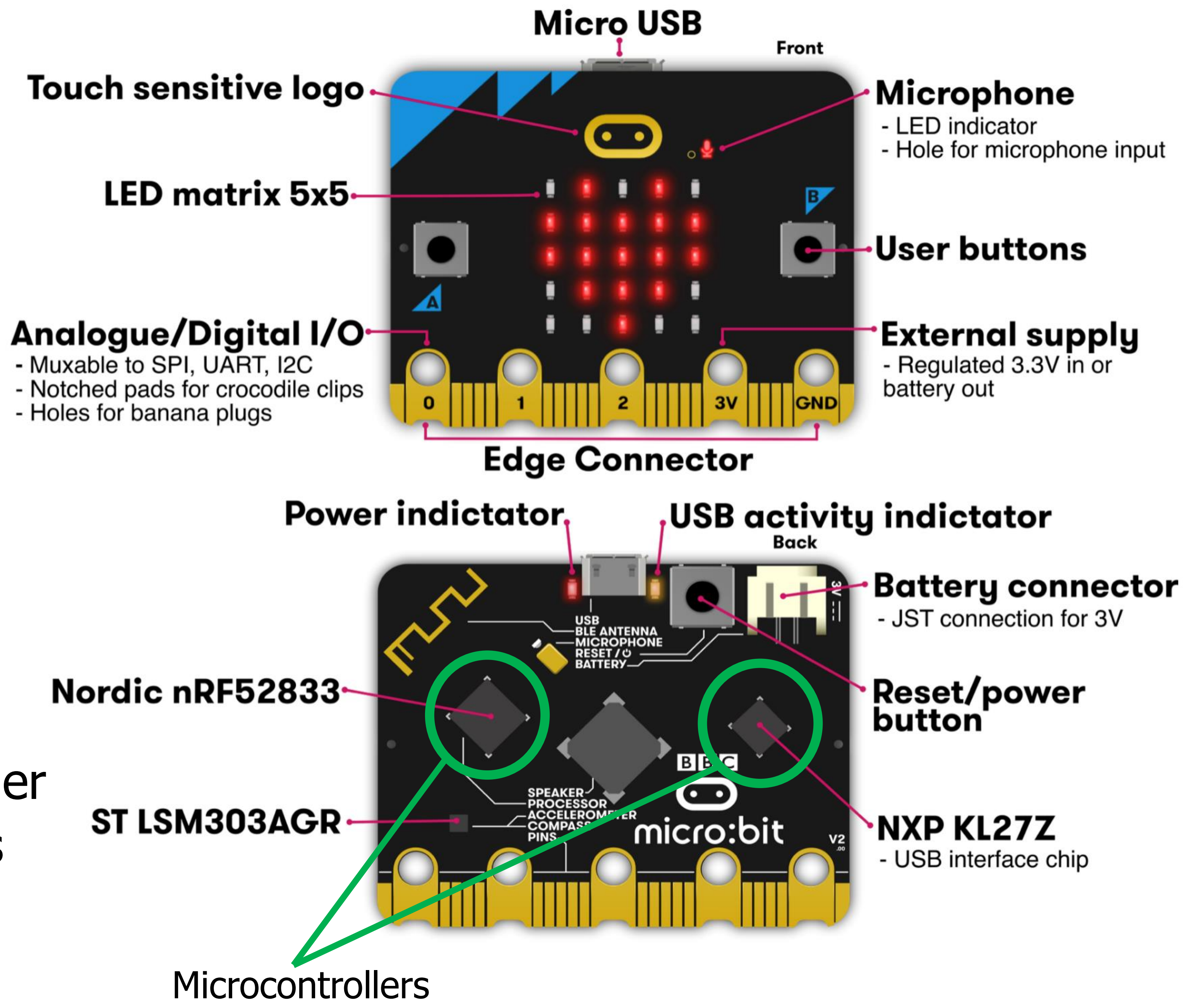
- Microcontroller Background
- Microcontroller Design
- Microcontroller History

- **Microbit microcontroller**

Micro:bit v2

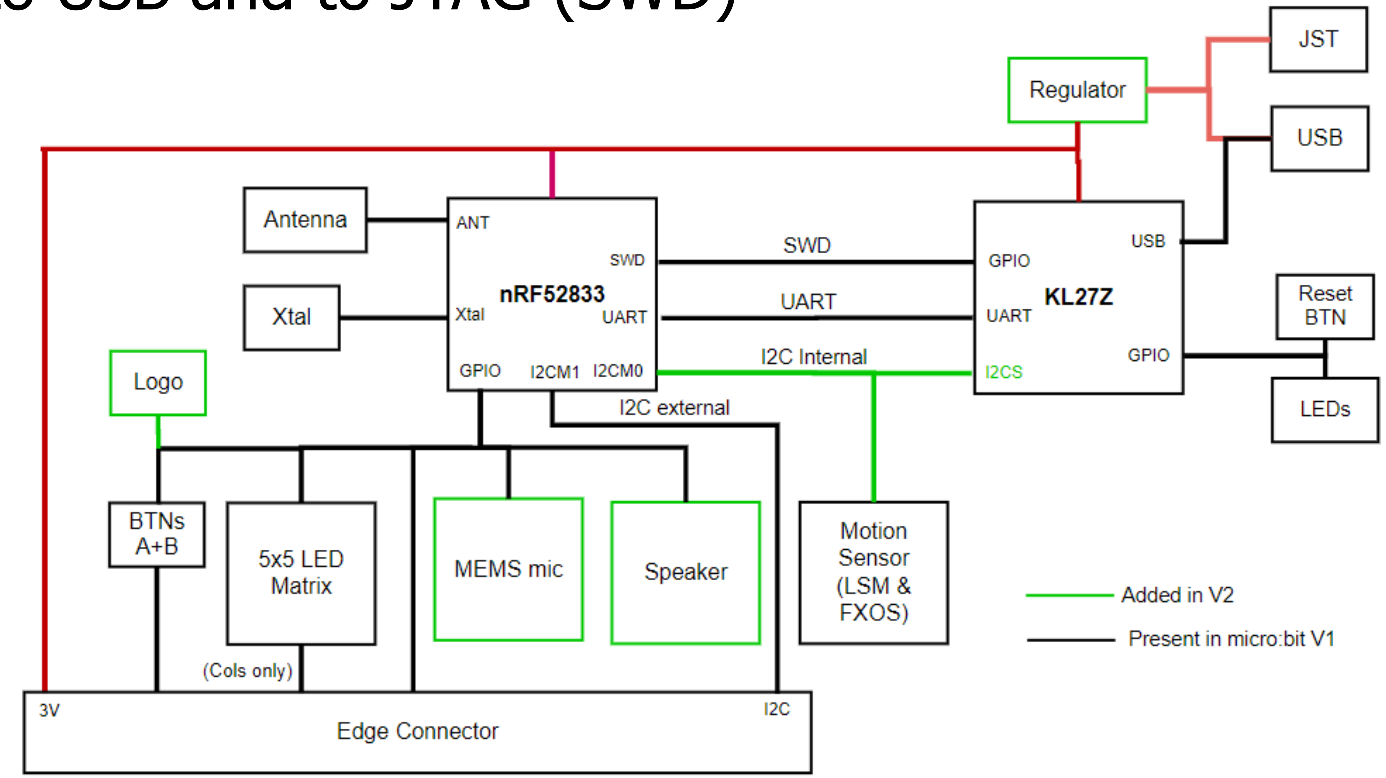
- Circuit board
  - Entire thing
  - a.k.a "Dev Board"
  - a.k.a PCB
    (Printed Circuit Board)

- Microcontroller
  - The computer on it
  - Microbit has two
    - One as a programmer
    - One for applications



KL27Z – Programmer on the Micro:bit v2

- 32-bit ARM Cortex-M0+ microcontroller
  - 48 MHz core, 16 KB RAM, 256 KB Flash
- Acts as a programming interface to nRF52833
  - Connects to USB and to JTAG (SWD)



nRF52833 – Microcontroller on the Micro:bit v2

- 32-bit ARM Cortex-M4F microcontroller
  - 64 MHz core, 128 KB RAM, 512 KB Flash
  - Floating point support
  - 2.4 GHz Radio: Bluetooth Low energy / 802.15.4
  - Various peripherals
    - ADC, PWM
    - I2C, UART, SPI, USB
    - RNG, 32-bit Timers, Watchdog, Temperature
    - Up to 42 I/O pins

To the datasheet!

- nRF52833 Product Specification
  - Online: https://infocenter.nordicsemi.com/index.jsp?topic=%2Fps_nrf52833%2Fkeyfeatures_html5.html

  - PDF: https://infocenter.nordicsemi.com/pdf/nRF52833_PS_v1.3.pdf

Outline

- Microcontroller Background
- Microcontroller Design
- Microcontroller History

- Microbit microcontroller(s)