A Little about ARM – The company

• Originally Acorn RISC Machine (ARM)
• Later Advanced RISC Machine
• Then it became ARM Ltd owned by ARM Holdings (parent company)
• In 2016 SoftBank bought ARM for $31 billion

ARM:
• Develops the architecture and licenses it to other companies
• Other companies design their own products that implement one of those architectures—including systems-on-chips (SoC) and systems-on-modules (SoM) that incorporate memory, interfaces, radios, etc.
• It also designs cores that implement this instruction set and licenses these designs to a number of companies that incorporate those core designs into their own products.

ARM Processors
• RISC based processors
• In 2010 alone, 6.1 billion ARM-based processor, representing 95% of smartphones, 35% of digital televisions and set-top boxes and 10% of mobile computers
• over 100 billion ARM processors produced as of 2017
• The most widely used instruction set architecture in terms of quantity produced

https://en.wikipedia.org/wiki/ARM_architecture
<table>
<thead>
<tr>
<th>Architecture</th>
<th>Core bit-width</th>
<th>ARM Holdings</th>
<th>Third-party</th>
<th>Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARMv1</td>
<td>32[1]</td>
<td>ARM1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARMv2</td>
<td>32[1]</td>
<td>ARM2, ARM250, ARM3</td>
<td>Amber, STORM Open Soft Core[39]</td>
<td></td>
</tr>
<tr>
<td>ARMv3</td>
<td>32[2]</td>
<td>ARM6, ARM7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARMv4</td>
<td>32[2]</td>
<td>ARM8</td>
<td>StrongARM, FA526, ZAP Open Source Processor Core[40]</td>
<td></td>
</tr>
<tr>
<td>ARMv4T</td>
<td>32[2]</td>
<td>ARM7TDMI, ARM9TDMI, SecurCore SC100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARMv5TE</td>
<td>32</td>
<td>ARM7EJ, ARM9E, ARM10E</td>
<td>XScale, FA626TE, Ferocoe, PJ1/Mohawk</td>
<td></td>
</tr>
<tr>
<td>ARMv6</td>
<td>32</td>
<td>ARM11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARMv6-M</td>
<td>32</td>
<td>ARM Cortex-M0, ARM Cortex-M0+, ARM Cortex-M1, SecurCore SC000</td>
<td>Microcontroller</td>
<td></td>
</tr>
<tr>
<td>ARMv7-M</td>
<td>32</td>
<td>ARM Cortex-M3, SecurCore SC300</td>
<td>Microcontroller</td>
<td></td>
</tr>
<tr>
<td>ARMv7E-M</td>
<td>32</td>
<td>ARM Cortex-M4, ARM Cortex-M7</td>
<td>Microcontroller</td>
<td></td>
</tr>
<tr>
<td>ARMv7-R</td>
<td>32</td>
<td>ARM Cortex-R4, ARM Cortex-R5, ARM Cortex-R7, ARM Cortex-R8</td>
<td>Real-time</td>
<td></td>
</tr>
<tr>
<td>ARMv8-R</td>
<td>32</td>
<td>ARM Cortex-R52</td>
<td>Real-time</td>
<td></td>
</tr>
<tr>
<td>ARMv7-A</td>
<td>32</td>
<td>ARM Cortex-A5, ARM Cortex-A7, ARM Cortex-A8, ARM Cortex-A9, ARM Cortex-A12, ARM Cortex-A15, ARM Cortex-A17</td>
<td>Qualcomm Krait, Scorpion, PJ4/Sheeva, Apple Swift</td>
<td>Application</td>
</tr>
<tr>
<td>ARMv8-A</td>
<td>32</td>
<td>ARM Cortex-A32</td>
<td></td>
<td>Application</td>
</tr>
<tr>
<td>ARMv8.1-A</td>
<td>64/32</td>
<td></td>
<td></td>
<td>Application</td>
</tr>
<tr>
<td>ARMv8.2-A</td>
<td>64/32</td>
<td>ARM Cortex-A55,[67] ARM Cortex-A75,[58]</td>
<td></td>
<td>Application</td>
</tr>
<tr>
<td>ARMv8.3-A</td>
<td>64/32</td>
<td></td>
<td></td>
<td>Application</td>
</tr>
</tbody>
</table>

https://en.wikipedia.org/wiki/ARM_architecture
# ARM Family and Architecture

<table>
<thead>
<tr>
<th>CORE</th>
<th>Application Cortex Processors</th>
<th>Embedded Cortex Processors</th>
<th>Classic ARM Processors</th>
<th>Cortex-A15</th>
<th>Cortex-A9</th>
<th>Cortex-A8</th>
<th>Cortex-A7</th>
<th>Cortex-M7</th>
<th>Cortex-R7</th>
<th>SC300</th>
<th>SC00</th>
<th>Cortex-M4</th>
<th>Cortex-M1</th>
<th>Cortex-M0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ARM11MP</td>
<td>Cortex-A5</td>
<td>ARM7/116</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ARM926</td>
<td></td>
<td>ARM968</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ASC100</td>
<td></td>
<td>ARM1136J</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ARM7TDMI</td>
<td></td>
<td>ARM946</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ARM1156T2</td>
<td></td>
<td>ARM1156T2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Family</th>
<th>ARM7TDMI</th>
<th>ARM9E</th>
<th>ARM11</th>
<th>Cortex-A/R</th>
<th>Cortex-M</th>
<th>Cortex-M</th>
<th>Cortex-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>ARMv4T</td>
<td>ARMv5TJ</td>
<td>ARMv6</td>
<td>ARMv7A/R</td>
<td>ARMv7M/ME</td>
<td>ARMv8M</td>
<td></td>
</tr>
</tbody>
</table>

- **ARM Family**
  - ARM7TDMI
  - ARM9E
  - ARM11
  - Cortex-A/R
  - Cortex-M
  - Cortex-M

- **Application Cortex Processors**
  - Cortex-A15
  - Cortex-A9
  - Cortex-A8

- **Embedded Cortex Processors**
  - ARM11MP
  - Cortex-A5
  - Cortex-M7

- **Classic ARM Processors**
  - ARM926
  - ARM968
  - ARM1136J
  - ARM7TDMI
  - ARM946
  - ARM1156T2

- **ARM Architecture Versions**
  - ARMv4T
  - ARMv5TJ
  - ARMv6
  - ARMv7A/R
  - ARMv7M/ME
  - ARMv8M
ARM FAMILY TREE

Cortex-A
- Cortex-A9
  - Cortex-A15
    - ARMv7-A
    - High performance 32-bit CPU with enterprise class feature set
  - Cortex-A17
    - ARMv7-A
    - High performance 32-bit CPU with lower power and smaller area
  - Cortex-A57
    - ARMv8-A
    - Highest performance
    - 64/32-bit CPU

Cortex-A
- Cortex-A5
  - ARMv7-A
  - Smallest and lowest power CPU
- Cortex-A7
  - ARMv7-A
  - High efficiency
  - 32-bit CPU
  - big.LITTLE™ compatible
- Cortex-A53
  - ARMv8-A
  - High efficiency
  - 64/32-bit CPU
  - big.LITTLE™ compatible

Cortex-R
- Cortex-R4
  - Real-time standard
- Cortex-R5
  - Functional safety
- Cortex-R7
  - High performance
  - 4G modem and storage
  - Enhanced system integration features

Cortex-M
- Cortex-M0
  - Lowest cost
  - Lowest power
- Cortex-M0+
  - Highest energy efficiency
- Cortex-M3
  - Performance efficiency
- Cortex-M4
  - Mainstream Control & DSP
- Cortex-M7
  - Maximum Performance
  - Control & DSP
  - Control
ARM Cortex Processors

• ARM Cortex-A family:
  • Applications processors
  • Support OS and high-performance applications
  • Such as Smartphones, Smart TV

• ARM Cortex-R family:
  • Real-time processors with high performance and high reliability
  • Support real-time processing and mission-critical control

• ARM Cortex-M family:
  • Microcontroller
  • Cost-sensitive, support SoC
• Cortex-M is a great trade-off between performance, cost, efficiency; used for IoT, various applications.
• Has on-chip peripherals
• Core is licensed by ARM
CORTEX-M: CORE + Peripherals

• Core
  • Memory
    • FLASH: Non-Volatile / Instruction memory
    • SRAM/DRAM: Volatile / data memory
  • Processor
    • ALU
    • Processor Control Unit (CPU)
  • Registers
    • Special Purpose Registers
    • General Purpose Registers
  • Buses
    • Data Bus
    • Instruction Bus
    • Bus bridge to connect diff. buses
    • Advanced High-performance Bus (AHB)
    • Advanced Peripheral Bus (APB)
  • GPIO

• Peripherals
  • ADC
  • LCD Controller
  • SPI
  • I2C
  • Etc.
Core Architecture

Von-Neumann

Instructions and data are stored in the same memory.

- Simple and inexpensive
- Access to data or instruction, one at a time

Harvard

Data and instructions are stored into separate memories.

- Faster
- More energy efficient
- Different bus sizes
Core Architecture

**Von-Neumann**

Instructions and data are stored in the same memory.

---

**Harvard**

Data and instructions are stored into separate memories.

---
ARM Simplified Block Diagram
System on Chip (SoC)

http://www.microdigitaled.com/ARM/ASM_ARM/PowerPoints/ARM_ASM_ppts.htm
ARM Cortex-M4 Organization (STM32L4)
Memory

- Memory is arranged as a series of “locations”
  - Each location has a unique “address”
  - Each location holds a byte (*byte-addressable*)
  - e.g. the memory location at address 0x080001B0 contains the byte value 0x70, i.e., 112

- The number of locations in memory is limited
  - e.g. 4 GB of RAM
  - 1 Gigabyte (GB) = $2^{30}$ bytes
  - $2^{32}$ locations ➔ 4,294,967,296 locations!

- Values stored at each location can represent either program data or program instructions
  - e.g. the value 0x70 might be the code used to tell the processor to add two values together

<table>
<thead>
<tr>
<th>Data</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>70</td>
<td>0x080001B0</td>
</tr>
<tr>
<td>BC</td>
<td>0x080001AF</td>
</tr>
<tr>
<td>18</td>
<td>0x080001AE</td>
</tr>
<tr>
<td>01</td>
<td>0x080001AD</td>
</tr>
<tr>
<td>A0</td>
<td>0x080001AC</td>
</tr>
<tr>
<td>0x00000000</td>
<td></td>
</tr>
</tbody>
</table>
Memory Mapping

Answer the following questions:
- What is the size of the EEPROM?
- What is the size of the Flash?
- Which Memory portion is non-volatile?
- What does SRAM generally used for in an ARM core processor?
- Where is 0x743 address?
- Where is 0x1000AB address?
Processor Registers

- Fastest way to read and write
- Registers are within the processor chip
- A register stores 32-bit value
- Cortex M (STM32L) has
  - R0-R12: 13 general-purpose registers
  - R13: Stack pointer (Shadow of MSP or PSP)
  - R14: Link register (LR)
  - R15: Program counter (PC)
- Special registers (xPSR, BASEPRI, PRIMASK, etc.) - more later
Program Execution

- Program Counter (PC) is a register that holds the memory address of the next instruction to be fetched from the memory.

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>PC</th>
</tr>
</thead>
<tbody>
<tr>
<td>4770 0x080001B4</td>
<td>188B 0x080001B0</td>
</tr>
<tr>
<td>2000 0x080001B2</td>
<td>2201 0x080001AE</td>
</tr>
<tr>
<td>188B 0x080001B0</td>
<td>2100 0x080001AC</td>
</tr>
</tbody>
</table>

PC = 0x080001B0
Instruction = 188B or 2000188B or 8B180020
Three-state pipeline: Fetch, Decode, Execution

- **Pipelining** allows hardware resources to be fully utilized
- One 32-bit instruction or two 16-bit instructions can be fetched.

Clock

Instruction i

Instruction Fetch | Instruction Decode | Instruction Execution

Instruction i + 1

Instruction Fetch | Instruction Decode | Instruction Execution

Instruction i + 2

Instruction Fetch | Instruction Decode | Instruction Execution

Pipeline of 32-bit instructions
Loading Code and Data into Memory

Dissection of a C Program: Copying an Array

```
int counter;
int a[5] = {1, 2, 3, 4, 5};
int main(void){
    int i;
    int b[5];
    counter = 0;
    for (i = 0; i < 5; i++){
        b[i] = a[i];
        counter++
    }
    while(1);
}
```
ARM Register and ALU
Machine codes are stored in memory
Fetch Instruction: pc = 0x08001AC
Decode Instruction: 2100 = MOVSR1, #0x00
Execute Instruction:
MOV S r1, #0x00
Fetch Next Instruction: pc = pc + 2
Decode & Execute: 2201 = MOVS r2, #0x01
Fetch Next Instruction: pc = pc + 2
Decode & Execute: 188B = ADDS r3, r1, r2
Fetch Next Instruction: $pc = pc + 2$
Decode & Execute: $2000 = \text{MOVS r0, #0x00}$
ARM Applications....
The A6 processor is the first Apple System-on-Chip (SoC) to use a custom design, based off the ARMv7 instruction set.
iPhone 6 Teardown

The A8 processor is the first 64-bit ARM based SoC. It supports ARM A64, A32, and T32 instruction set.

http://www.ifixit.com
iPhone 7
Teardown

A10 processor:
• 64-bit system on chip (SoC)
• ARMv8-A core
Apple Watch

• Apple S1 Processor
  • 32-bit ARMv7-A compatible
  • # of Cores: 1
  • CMOS Technology: 28 nm
  • L1 cache 32 KB data
  • L2 cache 256 KB
  • GPU PowerVR SGX543
Kindle HD Fire

Texas Instruments OMAP 4460 dual-core processor

http://www.ifixit.com
Fitbit Flex Teardown

STMicroelectronics 32L151C6 Ultra Low Power ARM Cortex M3 Microcontroller

Nordic Semiconductor nRF8001 Bluetooth Low Energy Connectivity IC

www.ifixit.com
Samsung Galaxy Gear

source: ifixit.com

• STMicroelectronics STM32F401B ARM-Cortex M4 MCU with 128KB Flash
Pebble Smartwatch

- STMicroelectronics STM32F205RE ARM Cortex-M3 MCU, with a maximum speed of 120 MHz

[source: ifixit.com]
Oculus VR

- Facebook’s $2 Billion Acquisition Of Oculus in 2014
- ST Microelectronics STM32F072VB **ARM Cortex-M0** 32-bit RISC Core Microcontroller

[source: ifixit.com]
HTC Vive

STMicroelectronics
32F072R8 **ARM Cortex-M0**
Microcontroller

(source: ifixit.com)
Nest Learning Thermostat

- ST Microelectronics **STM32L151VB** ultra-low-power 32 MHz ARM Cortex-M3 MCU

source: ifixit.com
Samsung Gear Fit Fitness Tracker

- STMicroelectronics STM32F439ZI 180 MHz, 32 bit ARM Cortex-M4 CPU

source: ifixit.com
A Little About STM32

STM32 is a family of 32-bit microcontroller integrated circuits by STMicroelectronics.

The STM32 chips are grouped into related series that are based around the same 32-bit ARM processor core, such as the Cortex-M7F, Cortex-M4F, Cortex-M3, Cortex-M0+, or Cortex-M0.

Internally, each microcontroller consists of the processor core, static RAM memory, flash memory, debugging interface, and various peripherals.

https://en.wikipedia.org/wiki/STM32
<table>
<thead>
<tr>
<th>Product Name</th>
<th>STM32F410P</th>
<th>STM32F412P</th>
<th>STM32F411P</th>
<th>STM32F410T</th>
<th>STM32F411T</th>
<th>STM32F410k</th>
<th>STM32F411k</th>
<th>STM32F409G</th>
<th>STM32F407ZG</th>
<th>STM32F405ZG</th>
<th>STM32F402ZG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash (Kb)</td>
<td>192</td>
<td>256</td>
<td>512</td>
<td>192</td>
<td>256</td>
<td>512</td>
<td>192</td>
<td>256</td>
<td>512</td>
<td>192</td>
<td>256</td>
</tr>
<tr>
<td>RAM (Kb)</td>
<td>1 M</td>
<td>2 M</td>
<td>4 M</td>
<td>1 M</td>
<td>2 M</td>
<td>4 M</td>
<td>1 M</td>
<td>2 M</td>
<td>4 M</td>
<td>1 M</td>
<td>2 M</td>
</tr>
<tr>
<td>RUN current</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
<td>360 mA</td>
</tr>
<tr>
<td>STOP current</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
<td>120 µA</td>
</tr>
<tr>
<td>Small package (mm)</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
<td>4X5</td>
</tr>
<tr>
<td>FSLC (NOR/PSRAM/LCD support)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QSPI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFSDM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAN 2.0B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRNG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMA Batch Acquisition Mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USB 2.0 OTG FS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access lines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
STM32 Nucleo Family
<table>
<thead>
<tr>
<th>Feature</th>
<th>NUCLEO F401</th>
<th>Arduino UNO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller</td>
<td>STM32F401 32-bit</td>
<td>ATMEGA 328 8-bit</td>
</tr>
<tr>
<td>Family</td>
<td>ARM Cortex-M4</td>
<td>AVR</td>
</tr>
<tr>
<td>Clock Frequency</td>
<td>84 MHz</td>
<td>16 MHz</td>
</tr>
<tr>
<td>Flash memory</td>
<td>512 Kb</td>
<td>32 Kb</td>
</tr>
<tr>
<td>SRAM</td>
<td>96 Kb</td>
<td>2 K</td>
</tr>
<tr>
<td>EEPROM memory</td>
<td>-</td>
<td>1 Kb</td>
</tr>
<tr>
<td>PWM</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>Analog inputs</td>
<td>16</td>
<td>6</td>
</tr>
<tr>
<td>Digital Pin</td>
<td>47</td>
<td>14</td>
</tr>
<tr>
<td>I2C modules</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>USART modules</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>SPI modules</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Timer</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>Floating point UNIT</td>
<td>One</td>
<td>No</td>
</tr>
<tr>
<td>Maximum voltage supported</td>
<td>5V</td>
<td>5V</td>
</tr>
<tr>
<td>USB OTG</td>
<td>One</td>
<td>No</td>
</tr>
<tr>
<td>Dimensions</td>
<td>68mmx80mm</td>
<td>53mmx68mm</td>
</tr>
<tr>
<td>Price</td>
<td>€ 10</td>
<td>€ 20</td>
</tr>
</tbody>
</table>
Other ARM Chips

<table>
<thead>
<tr>
<th>Company</th>
<th>Device</th>
<th>Flash (K Bytes)</th>
<th>RAM (K Bytes)</th>
<th>I/O Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atmel</td>
<td>AT91SAM7X512</td>
<td>512</td>
<td>128</td>
<td>62</td>
</tr>
<tr>
<td>NXP</td>
<td>LPC2367</td>
<td>512</td>
<td>58</td>
<td>70</td>
</tr>
<tr>
<td>ST</td>
<td>STR750FV2</td>
<td>256</td>
<td>16</td>
<td>72</td>
</tr>
<tr>
<td>TI</td>
<td>TMS470R1A256</td>
<td>256</td>
<td>12</td>
<td>49</td>
</tr>
<tr>
<td>Freescale</td>
<td>Mk10DX256VML7</td>
<td>256</td>
<td>64</td>
<td>74</td>
</tr>
</tbody>
</table>