I. Introduction.

BE2M37MAM – Microprocessors

Stanislav Vítek

Czech Technical University in Prague

Outline

Part I

Course outline

Introduction

- Website

  https://cw.fel.cvut.cz/wiki/courses/be2m37mam

- Lectures and labs
  - Stanislav Vítek, viteks@fel.cvut.cz, B2-719

    http://mmtg.fel.cvut.cz/personal/vitek/
  - Lectures: C3-438
  - Labs: B2-717

Course purpose

Introduction to embedded software development

- Introduction to microcontrollers
- Firmware development
- Hardware debugging

Practical approach

- Presentation of peripherals during lectures
- Test on real hardware during lab sessions

Course goal

- Enable you to autonomously develop embedded projects

Lectures

Lecture **01** – Introduction to microcontrollers

Lecture **02** – STM32F4: getting started, GPIO peripheral

Lecture **03** – Peripherals: timers

Lecture **04** – Peripherals: ADC and DMA

Lecture **05** – Peripherals: USART

Lecture **06** – Peripherals: SPI and I2C

Lecture **07** – Peripherals: PWM, **project assigment**

Lecture **08** – Communication buses: RS485, Modbus, One-wire

Lecture **09** – Inter-processor communication

Lecture **10** – Realtime and parallel systems

Lecture **11** –

Lecture **12** –

Lab sessions / individual tasks

- Lab **01** IDE setup, LED blinking
- Lab **02** GPIO, light snake
- Lab **03** GPIO, display driver
- Lab **04** Timers, stopwatch
- Lab **05** ADC
- Lab **06** I2C bus
- Lab **07** Project start

Part II

Representation of a Numbers

II. Representation of a Numbers

Motivation

Integer Numbers

Negative Numbers

Motivation

- Why ever concern with internal representation of numbers in processor circuits?
  - Inside computers, everything is a number
  - But numbers usually stored with a fixed size

    8-bit bytes, 16-bit half words, 32-bit words, 64-bit double words
- Let's took **random two bytes** from memory:
  - If, in the case of processor circuit we got a set of bytes without an information about their meaning, we can not in any way distinguish their intention.
  - **Is it a piece of code or number – and which one?**
  - Opposite, when we decide to use distinct data type for number representation, we essentially have to know pros and cons of data type.

Motivation

- Misunderstanding the properties and limitations of the selected data type may result in:
  - Complete failure of otherwise correct algorithm
  - The significant reduction in performance

    e.g., poorly designed IIR filter in a 16-bit arithmetic can be unstable for signals with a bit resolution of about 6 bits!
  - The hidden errors, when the algorithm gives false results only at certain values of data
- Problems are caused especially by:
  - Overflow in integer / fixed point arithmetic
  - Rounding errors, or impossibility to represent distinct number in floating point arithmetic
    - financial operations
    - represent and count time in floating point or fixed point data type is virtually "evil" idea...

Example of serious bug – Patriot system

- When first designed, the primary targets were Soviet aircraft and cruise missiles travelling at speeds around MACH 2, and only operating at a few hours at a time

  Based on a velocity of target and time the tracking software calculates an area in the air space for where the system should look next for the incoming missile

- The time was represented by a 24 bit fixed point number, and measured as the number of tenth-seconds (0.1 s !!!)
- During the Operation Desert Storm, the system was deployed as defences operating **continuously**, tracking and intercepting Scud missiles travelling at speeds of approximately **MACH 5**
- After **100 hours** of operation, inaccuracy was roughly **0.34 s** leading in the trajectory prediction error of almost **700 m**...
- 25.2.1991 in Dhahran, Saudi Arabia a Patriot missile system failed, resulting in 28 dead and 98 injured...

II. Representation of a Numbers

Motivation

Integer Numbers

Negative Numbers

Positional Number Systems

- Place value system (positional notation)
  - The number is represented by a combination of characters (digits)
  - Base b is the number of values, which are represented by a character in a single position
  - n^{th} position has a weight b^n
- Example: a number 1023 may be represented
  - **Decimal** – in base 10
    $1 \times 10^3 + 0 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$
  - **Hexadecimal** – in base 16
    $3 \times 16^2 + 15 \times 16^1 + 15 \times 16^0$
  - **Binary** – in base 2
    $1 \times 2^{10} + 1 \times 2^9 + 1 \times 2^8 + 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

Classification by the number of bits

- to store a given value we have a limited number of positions – bits

  The same number can be encoded in different ways – especially applies for negative numbers and decimal numbers (BCD, DPD), Gray code and others.

Storage width	Format name	Effective range
8	BYTE	0..255
16	WORD	0..65 535
32	DWORD	0..4 294 967 295
64	QWORD	0..18 446 744 073 709 551 615

Note: a WORD format had in past different width, ranging from 4 bits on Intel 4004 platform (nibble in present terminology), over presently exotic widths (22, 40, 50, ...) up to 64 bits in Cray or Alpha computers, including variable widths (IBM)

Binary Coded Decimal (BCD) encoding

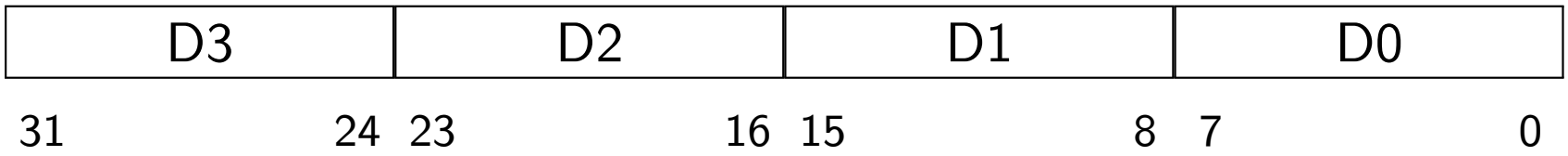
- class of binary encodings of decimal numbers
  - each decimal number is represented by fixed number of bits (4/8)
  - special bit patterns are usually used for a sign or for other indications (e.g., error, overflow)
- ten states representing BCD number are called **tetrades**
- Pros
  - more accurate representation and rounding of decimal number
  - ease of conversion into human-readable format
- Cons
  - low efficiency

bits	**efficiency (%)**	
	BCD	**packed**
8	3.9	39.1
16	0.15	15.26
32	0.00023	2.33

Binary Coded Decimal (BCD) encoding

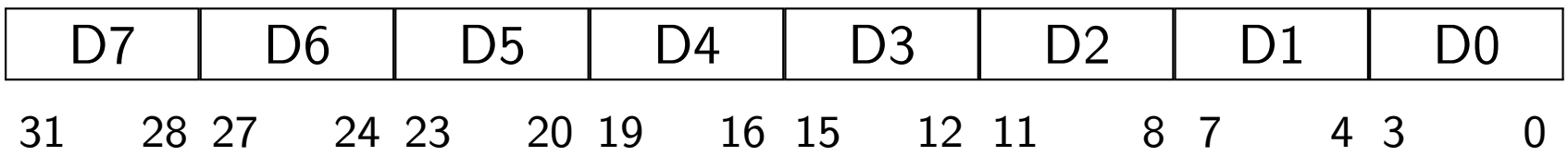
- BCD number stored in 32 bits

32 bits represent BCD number ranging from 0 ... 9 999



- packed BCD number stored in 32 bits

32 bits represent packed BCD number ranging from 0 ... 99 999 999



- compressed BCD (Densely Packed Decimal)
  - better utilization of a bus (speed!)
  - efficiency: 3 decimal places per 10 bits (97,7%)

x86 architecture supports 80 bit numbers in packed BCD cede (but used are just 72 bits (18 decimal places) + sign bit), but x87 coprocessor converts these numbers into floating point format!!!

Weighted tetrade codes

- A digit in the BCD code is represented by 4 bits
  - natural one uses weights 8, 4, 2, 1
  - other BCD codes may use another weights, even not in power of 2

code	**weights**	**example** 6_{10}
BCD	8421	0110
Aiken	2421	1100
	4221	1100
	5311	1001
	$74\bar{2}\bar{1}$	1001

II. Representation of a Numbers

Motivation

Integer Numbers

Negative Numbers

Possible coding of negative numbers

- **Sign** – naive idea with a most significant bit representing sign
  - 0 – positive number
  - 1 – negative number
  - for the representation of integers with exception of BCD code is rarely used (more complicated implementation – negative and positive zero)
- One's complement
- **Two's complement** – used by almost all computers today
  - simple hardware implementation of arithmetic operations
  - positive zero
  - N bits gives range -2^{N-1} to $2^{N-1} - 1$ numbers
- Biased zero code – exponent in floating point number representation

Two's Complement

- Conversion

  $X_2^{**} = \overline{X}_2 + 1$

  - negate all bits
  - add 1

  $-a_2^{**} = 2^N - a$

  - Example: $-50_{10} = 256_{10} - 50_{10} = 206_{10} = 1000\ 1110$
- What values can take 8-bit binary number in two's complement?

  $0000\ 0000_2 = 0_{10}$

  $0111\ 1111_2 = 127_{10}$

  $1000\ 0000_2 = -128_{10}$

  $1111\ 1111_2 = -1_{10}$

Part III

A Brief History of Computers

III. A Brief History of Computers

Mechanical Computers

Historical digital computers

Mechanical calculators

- allowed basic math operations (+, -, *, /);

1617 John Napier created his rods (or bones); slide rule was developed around 1630 (and modified in 1654 so that it allowed calculation of trigonometric functions, logarithms, roots and powers) and was used until the mid-of 20th century;

1623 Wilhelm Schickard invented his "calculating clock" (a mechanical calculator with rotating wheels)

1643 Blaise Pascal invented mechanical calculator "Pascaline" (the first calculator to be used in an office, commercialized and patented), improved by Liebnitz in 1674, (automatic multiplication and division);

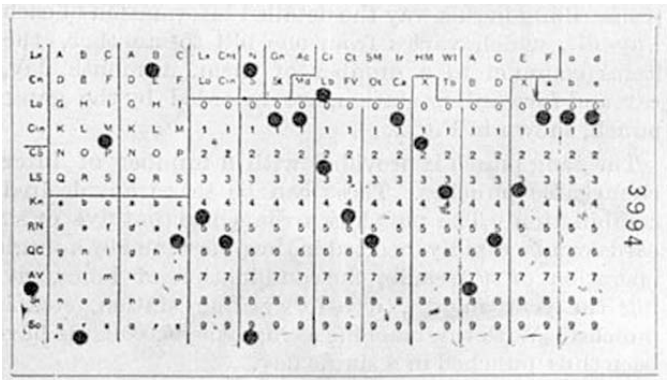
1850 Parmalee invented first calculator with keyboard, which was improved and supplemented by printing at 1885 by Burroughs and in 20th century, the calculator was supplemented with electric motors and in that manner survived until the second half of the 20th century.

Punched cards

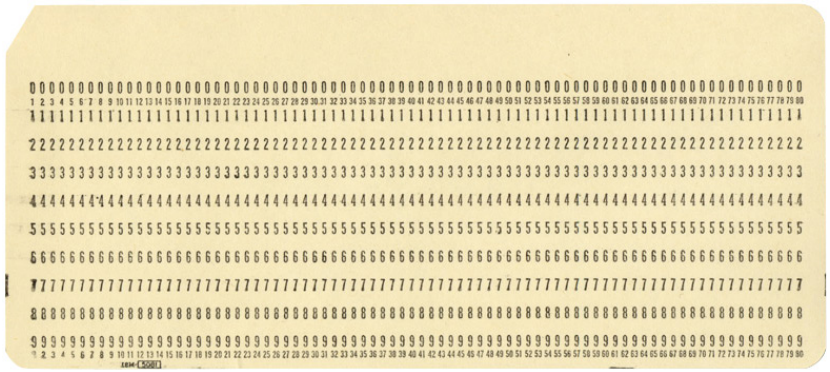
- used for payroll processing, stock accounting and statistical reports, entering data into computers

1725 were historically first ever used for the controlling of looms in France

1886 the first mechanical counting machine, based on the cards was developed by Hollerith in the USA (CTR Corporation, in 1924 its name was changed to IBM).



Herman Hollerith's design (1890), circular holes in 22 columns with eight punch positions each.



IBM 80-column card (1928)

Analytical Engine

- Charles Babbage (1834 – 1871), powered by a steam engine
- first programmable computer (using punched cards)
- it has mechanical memory (1000 50-digit decimal numbers), arithmetic operations, supported conditional branching of a program
- programming was ensured by Countess Augusta Ada Lovelace
- due to financial problems, the machine was not completed.

Harvard Mark 1

- Harvard University and IBM (1944), powered with 4kW motor
- Harvard's architecture, fixed-point arithmetic
  - program memory: not present

    computer processed program instructions from paper tape
  - data memory: 72 numbers, each 23 decimal digit long
- input: 60 sets of 24 mechanical switches
- output: mechanical typewriter
- conditional branches not supported

Analog computers

- mechanical integrators
  - calculations of differential equations for the control of artillery fire (1876 Kelvin Thomson) and bombings (2nd World War, Korea, Vietnam, "Norden bombsight")
- tube and transistor integrators
  - used for fire control and simulation of mechanical systems
  - the programming was performed by simple interconnecting of dedicated programming terminals by wires
  - input – from the coordinate reader, output – the plotter
  - Producers: Electronic Associates, Aritma (Meda), Tesla Pardubice (APX), ...

III. A Brief History of Computers

Mechanical Computers

Historical digital computers

Generation 0

- used electromagnetic relays, including memory
- Zuse Z2 1939 Z3, 1947 Mark 2, partially Mark 3

Generation 1 (1/2)

- used tubes, served especially for the army

  the development of hydrogen bombs, artillery ...

- worked with decimal numbers, Fortran programming language
- the memory
  - cathode tubes
    - modified television screen (CRT); each pixel was a single bit of information, represented by the charge; required refreshing, analogous to today's DRAM memories; information has been read by the sensor on the outer side of the screen
  - mecury delay lines, magnetic drums, ferrite cores
- had modular design, ring counters, flip-flops, 20 accumulators with 10-digit registers.
- arithmetic was carried out by counting the pulses and was about 1000 times faster than the Harvard Mark 1

Generation 1 (2/2)

- Colossus – Mark 2 (1944 England)
- ENIAC (1946, Penn State University, USA)
  - used hybrid construction with 17,000 vacuum tubes, but also the 1500 relay.
  - considered the first fully programmable (commercially successful) computer
- EDVAC (1949, Ballistics Research Laboratory, USA)
  - first time used von Neumann architecture
  - binary numbers
- IBM 700 series (1953)

Generation 2

- transistors, input – typewriter, output – printer
- the data format was not still standardized
- programming languages Algol and Cobol
- IBM series 7000 (1958, scientific computing)

  IBM 7030 had 64 bit words
- IBM series 1400 (1959, banking systems)

  IBM 1401 used 6-bit "bytes"
- CDC 6600 (1964)
  - the first successful supercomputer

    built under the leadership of Seymour Cray
  - RISC architecture, running at 10 MHz

    however, using 4-phase signaling, the effective frequency were 40 MHz
  - CPU had 60 bits data registers and 18 bits address registers
  - Instructions had either 15 or 30 bits
  - Input and output were secured by 10 peripheral processors with a 12-bit "bytes" and 6 bit characters.

Generation 3

- integrated circuits, input – keyboard, output – display
- pernament memory: magnetic disc (28MB in 1966)
- IBM 360 series ()
  - 32-bit data bus
  - supported 64-bit data width for floating point calculations
- Cray 1 (1976)
  - 80 MHz CPU, 1 MB RAM, 64 bit data, 24 bit addresses
  - design optimized in terms of delay and reflection on the wires

this concept was partially used in Petium class processors

Part IV

Microcomputers and Microcontrollers

IV. Microcomputers and Microcontrollers

Organization and Architecture

- **Computer architecture** – refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program.
  - Architectural attributes – instruction set; the number of bits used to represent various data types (e.g., numbers, characters); I/O mechanisms; techniques for addressing memory.

    It is an architectural design issue whether a computer will have a multiply instruction.
- **Computer organization** – refers to the operational units and their interconnections that realize the architectural specifications.
  - Organizational attributes transparent to programmers – control signals; interfaces between the computer and peripherals; the memory technology used.

    It is an organizational issue whether that instruction will be implemented by a special multiply unit or by a mechanism that makes repeated use of the add unit of the system.

Main Structural Components

- **Central processing unit (CPU)** – controls the operation of the computer and performs its data processing functions; often simply referred to as processor.
  - Control unit – controls the operation of the CPU
  - Arithmetic and logic unit (ALU) – data processing functions
  - Registers – provides storage internal to the CPU
  - CPU interconnection – some mechanism that provides for communication among the control unit, ALU, and registers
- **Main memory** – stores data.
- **I/O** – moves data between the computer and its external environment.
- **System interconnection** – some mechanism that provides for communication among CPU, main memory, and I/O.

  A common example of system interconnection is by means of a **system bus**, consisting of a number of conducting wires to which all the other components attach.

Other Structural Components

- **Interrupt system** – it simplifies of peripherals, and improves utilization of machine time since CPU does not need to check in regular intervals their state. Instead, peripheral device informs CPU by an interrupt that it has new data for processing.
- **Address decoder** – generally required for generating an control signals for memories and memory mapped I/O peripherals from addresses.
- **Clock generator** – must be connected to the CPU.

IV. Microcomputers and Microcontrollers

Introduction

Computer Architectures

Instruction Set Architecture (ISA)

Microcontrollers

ARM microcontrollers

Harvard Architecture (1/2)

- proposed by Howard Hathaway Aiken in 1937

  inspired by Babbage's machine

- first realization Harvard Mark I

  original codename Automatic Sequence Controlled Calculator or just ASCC

- architecture includes five basic blocks

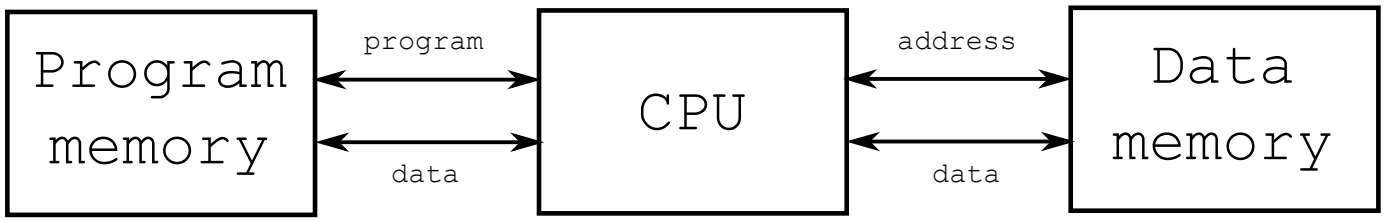
  or six, if we consider I/O as two separate parts

- an apparent and most important difference from Von Neumann architecture is splitting of memory on two separate parts
  - program memory – instructions, may also store constants and input gates

    In Harvard Mark I computer the "program memory" was a punched paper tape, from which the program was read and directly executed.
  - data memory – variables, optionaly I/O space

    Data words may or may not have a different width (different number of bits) than instructions have.

Harvard Architecture (1/2)



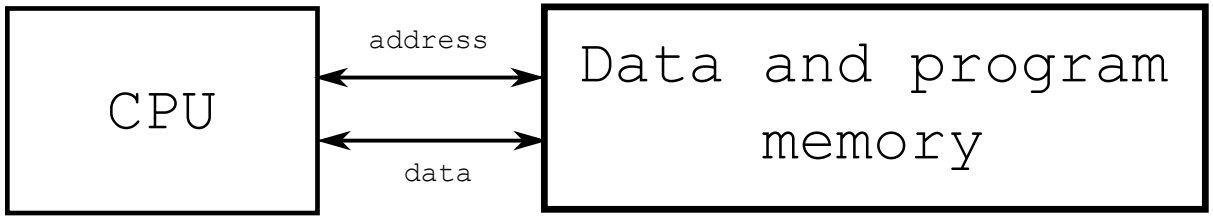
Pros and Cons

- Due to memory allocation program and data memory can have separate buses so both instruction and data may be read at the same time – higher program processing speed.
- Higher security, as the data could not rewrite a piece of code as a buffer overflow attack, for example.

Von Neumann (Princeton) Architecture (1/2)

- Derived from a computer architecture proposal from 1945 "First Draft of a Report on the EDVAC" from Hungarian-American mathematician John von Neumann
- It has only one memory, shared both by program and data
  - this memory is called operational memory
  - since both instructions and data shares the same memory space, they must have the same word width, and control unit can distinguish between instructions and data only by their order or address
  - data share the same address, data and control buses
- ALU performs all arithmetic calculations and evaluates logic expressions, in contrast to many Harvard architecture based processors. The operation is sequential so it is more difficult to deal with parallelism in design.

Von Neumann (Princeton) Architecture (1/2)



Pros and Cons

- Since instructions and data share a common bus, an instruction fetch and a data operation cannot occur at the same time. So operation of microcomputer with von Neumann architecture is generally slower than a system with Harvard architecture operating at the same clock frequency **Von Neumann bottleneck**.
- Program can modify itself. This is associated with serious safety risk, a **buffer overflow attack**.

The design of a Von Neumann architecture is simpler, may be cheaper.

IV. Microcomputers and Microcontrollers

Instruction Set Architecture (ISA)

- It is a part of the computer architecture, describes an essential programming model.
- It separates programmer (or compiler) from the actual physical implementation of a computer.
- The same ISA can be defined on computers with a completely different microarchitecture. For example, for Intel and AMD, we can write an identical code, even though the internal structure of both chips is entirely distinct.
- ISA defines:
  - Which operations are supported by the processor.
  - Data types, registers, addressing modes, memory organization.
  - Interrupts and exceptions, operations of input and output.

RISC vs. CISC

- The RISC instruction set contains a relatively small number of simple instructions (33-150)

  CISC has a large number of instructions, which can be greater than 1000 on PC

- The size of the RISC instructions is usually a single word (depending on the width of the bus carrying instructions)

  CISC instruction length varies (from 1 byte to 15 bytes on PC)

- Execution of RISC instruction is in the same number of machine cycles, usually directly proportional number of words of instruction

  Execution time of distinct instructions varies on CISC processors significantly, even if they are of the same length, depending on complexity of a microcode microprogram

- Programs RISC processors are usually faster, but due to more instructions need more storage space.
- Program execution time on RISC processors may be precisely determined

  each instruction on CISC processor has a different execution time

IV. Microcomputers and Microcontrollers

What are microcontrollers?

Definition

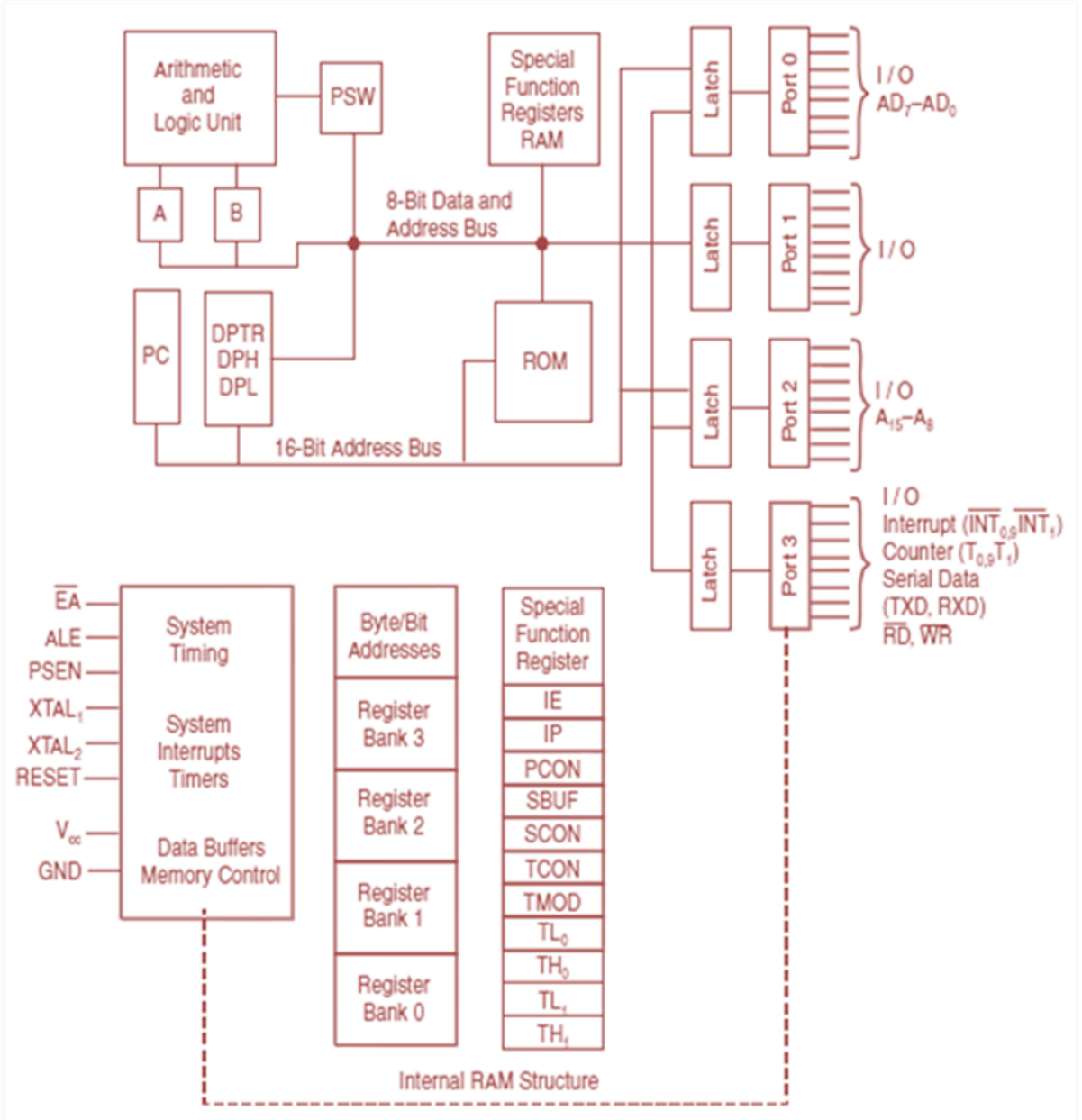
- A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.

- A microcontroller has peripherals, a processor no
- A microcontroller has an integrated program memory, a processor no
- A microcontroller usually needs few additional components to run, a processor many
- Generally speaking, microcontrollers are less aimed at performing heavy processing, more at interfacing with other devices

Microcontroller vendors overview

- Intel developed the 8051 architecture in 1980, still in use
- Microchip is well known for their PIC, probably the most popular microcontroller
- Atmel produces the AVR family, that powers Arduino boards
- ARM based microcontrollers are available from many vendors
- Parallax produces the Propeller, a multi-core microcontroller
- Texas Instruments has the MSP430, targeted to ultra-low-power applications
- Cypress Semiconductors produces the CY8Cxxxxx family of PSoC (programmable system on chip)
- Analog Devices sells precision analog ADuC microcontrollers
- Rabbit, Dallas, Freescale, Lattice, In neon, ST Microelectronics, Toshiba, Zilog...

Intel 8051 architecture – 1980

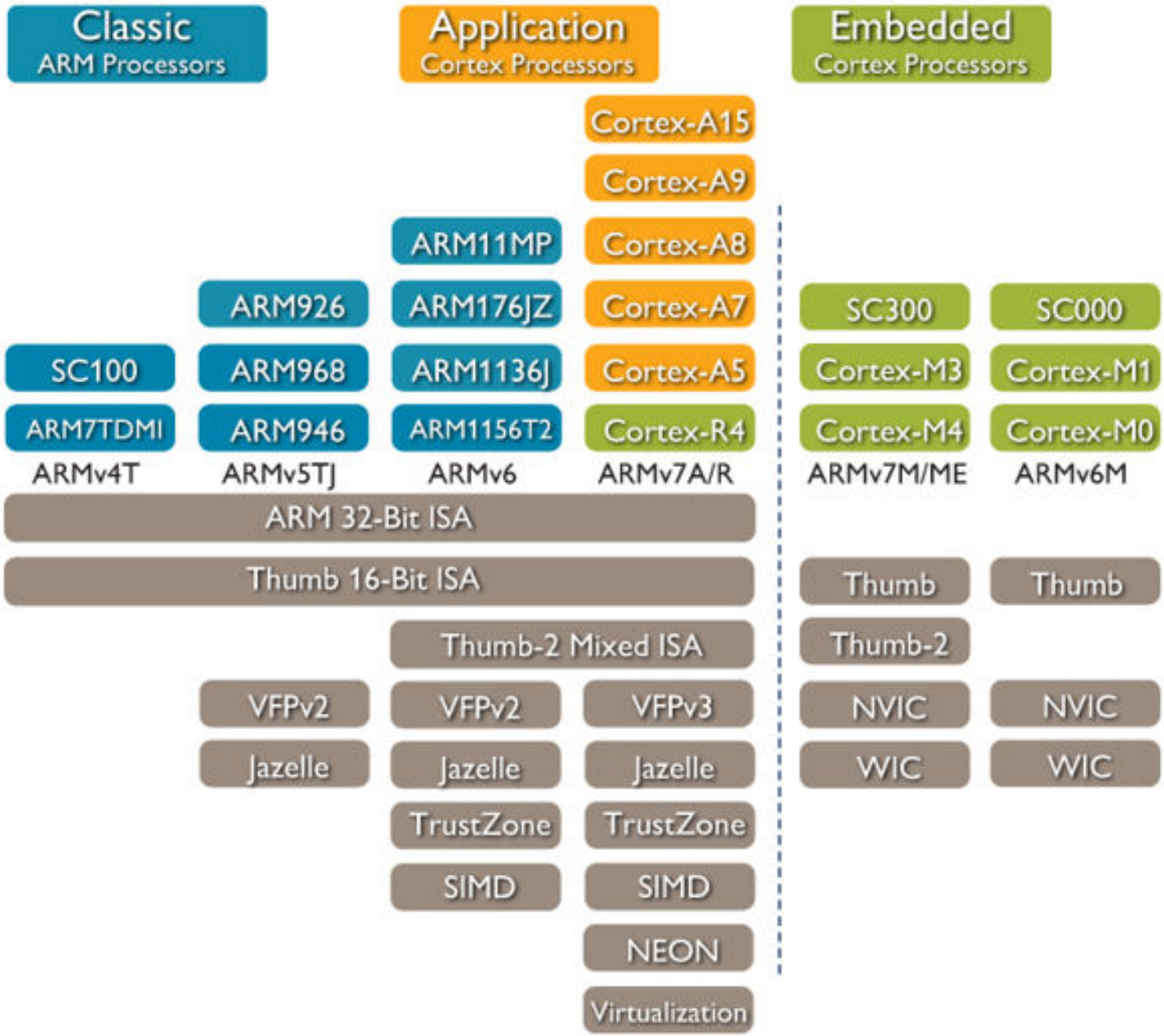


IV. Microcomputers and Microcontrollers

ARM: Advanced Risc Machines

- Founded in 1990 by Acorn, Apple and VLSI
- Headquarters in Cambridge, United Kingdom
- ARM is not a manufacturer, it licenses intellectual property to manufacturers
  - Alcatel-Lucent, Apple Inc., Atmel, Broadcom, Cirrus Logic, Digital Equipment Corporation, Freescale, DEC, LG, Marvell, Microsoft, NEC, Nintendo, Nvidia, Sony, NXP, Oki, ON Semiconductor, Qualcomm, Samsung, Sharp, STMicroelectronics, Texas Instruments, VLSI Technology, Yamaha and more
- In 2010, over 95% of the smartphones sold used an ARM processor
- As of 2009, ARM processors account for approximately 90% of all embedded 32-bit RISC processors

ARM family and architecture



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

Where to find information? (1/2)

Almost every component has a datasheet, that reports:

- the description of the component
- the pinout of the component
- electrical specifications (max/min voltage supply, current, ...)
- performance details (frequency response, noise, ...)
- thermal characteristics
- mechanical characteristics, technical drawing and footprint
- generally, a reference design

Where to find information? (2/2)

Reference manual

- extensive description of the device
- modes of operation
- configuration details
- registers map

Application note

- common applications
- example configurations
- complete systems using the specific components
- demo boards