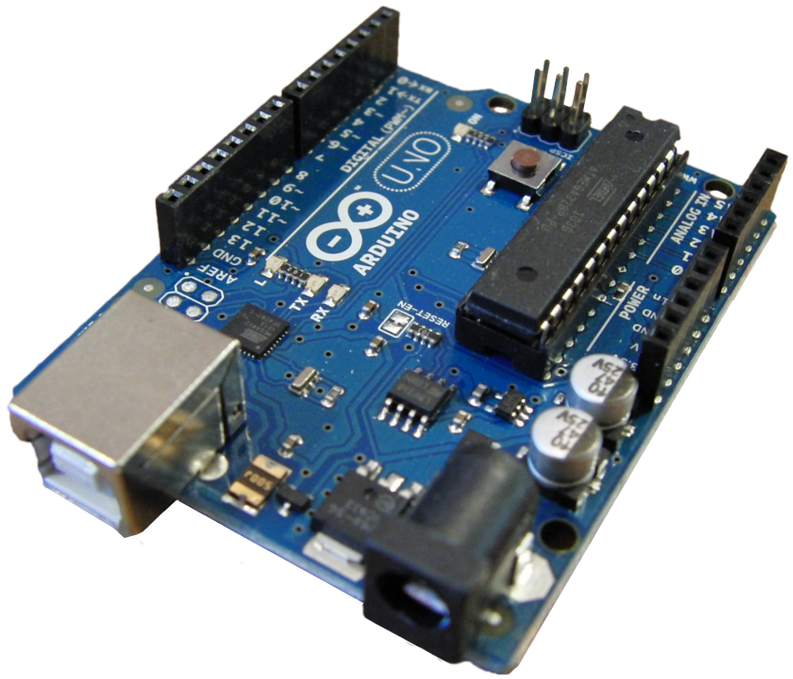


Microcontrollers

Associate Prof. András Libál

3rd year optional course, 2018 Fall Semester
Department of Mathematics and Computer Science
Babes-Bolyai University, Cluj

Contact

András Libál,

alibal@cs.ubbcluj.ro
andras.libal@gmail.com
andras.libal (facebook, if there is something urgent)

Mathematicum 210
(only if we have agreed on a date/time in email previously)

I am a Physicist (not a programmer) so we will learn about the underlying electronics and physical phenomena and signal processing as well (when necessary or aids in understanding)

I would like to emphasize understanding the hardware and being able to code in an efficient manner (fast, low energy consumption, low hardware requirements) so that you could use this knowledge later if you are building an embedded system/IoT project

Content of the course

We will study the Atmega 328P microcontroller, the Arduino IDE and language but we will also talk about lower level coding possibilities (accessing registers) in C.

Timers, PWM

Interrupts and real time

Serial Port, 1W,
TWI (I2C) SPI
communication

Variety of
Arduino **sensors**

Analog Digital
Conversion

low power apps, low cost hardware, utilizing hardware to the fullest

Internet of Things

Requirements, minimum requirements

Final grade 60% activity during the year
40% exam grade (written exam)

Activity during the year 60% = 6 labs (120 points) + 12 quizzes (120 points)

Every course begins with a quiz (5 questions) from the previous lecture (10 min.) this is worth 10 points

Every lab is worth 20 points. There will be an option to recover a lab. You have to be present at 75% of the labs (4 labs out of 6, this is a University requirement)

Minimum grade on the exam: 5.0 (it is a like big quiz with 44 questions)

12 quizes = 120 points
minimum 3.0 (36 pont)

6 labs+ 12 quizzes
= 240 pointd
minimum 5.0
(120 points)

Consequences of cheating

Cheating on the quiz (using any electronic device (cell phone etc)) is automatically a -40 points on that quiz

Cheating on the final exam (using any electronic device (cell phone etc)) when a second professor is present and in that person I have a witness to the event: disciplinary action in front of the ethics committee with possible expulsion from the University

Grades in a spreadsheet

Babes-Bolyai Tudomanyegyetem
Matematika es Informatika Szak
Informatika (magyar)

				Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q7+	Q8	Q9	Q10	Q11	SQ	Lab1	Lab2	Lab3	Lab4	Lab5	Lab6	Lab7	Quiz Atlag	Évközi Jegy	Vizsga	Vegso	Vegso
ID	Csaladnev	AK	Keresztnev																					Minimum 3	Minimum 5			kerekitve
															110	21.33							154	11.93909	11.4132	11	11.24792	10
1491	BAJCSI	I	ADÉL	10	8	9	10	10	7	8	1	10	10	8	8	18.33	22	22	22	22	20	22	22	10.63	10.7572	8.44	9.83032	10
1493	BALAZS	A	BRIGITTA	10	7	5	8	8	10	8	1	10	10	10	9.2	16.33	22	22	22	22	24	22	22	10.24818	10.7492	7.48	9.44152	9
1495	BARTHA	L	VIVIEN-EMŐKE	10	8	5	7	8	6	7	10	8	6	4	8	12.73	20	12	9	22	20	12	0	9.102727	7.8052	6.42	7.25112	7
1499	BIRÓ	S.J.	ENIKŐ	10	10	8	10	9.6	8	6	1	8	4	6	10	17.2	22	22	22	22	20	22	22	9.8	10.392	9.3	9.9552	10
1501	BÓNÉ	G	NORBERT-MÁTÉ	10	4	8	10	10	6.6	2	5	5.6	0	6	5.2	10.4	22	17	9	22	0	1	22	7.527273	7.032	6.56	6.8432	7
1502	BONÉ-GRÉCZI	Z	ONÉZIMOSZ-ZOLTÁN	10	2	0	6	8	8	3	0	6	8	4	5.2	0	22	12	9	22	0	12	0	5.472727	5.488	3.32	4.6208	3
1503	BORSAY	F. J.	ZSUZSANNA	0	5	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0.454545	0.2		0.12	10
1508	DEBRE	M	EMESE-REBEKA	5	5	10	9	8	10	9	1	10	10	8	7.6	14.93	22	22	22	22	24	22	22	9.775455	10.5412	5.88	8.67672	9
1518	FODOR	D	LÓRÁNT	10	6	6	6	8	10	6	0	6	10	8	5.2	11.53	22	22	19	22	22	22	22	8.43	9.7492	5.9	8.20952	8
1519	GAGYI	F	MÁTYÁS	0	0	0	0	0	0	0	0	0	0	0	0	0	22	22	19	0	0	0	0	0	2.52		1.512	2
1521	GOLICZA	P	LUKÁCS	10	4	7	8	0	10	6	0	7	0	2	0	0	20	17	22	11	22	20	22	4.872727	7.504	5.7	6.7824	7
1522	GOSULY		ROBERT	0	4	7	10	7	6	6	0	8	8	6	6	15	22	20	22	11	22	20	22	7.509091	8.864	7.18	8.1904	8
1526	KALLOS	A.A.	EDUARD-ATTILA	10	6	0	5	7	4	0	0	10	0	6	0	0	20	22	21	22	0	0	0	4.381818	5.328	4.86	5.1408	4
1528	KAPROS-BINDASZ	L	REGINA	10	6	8	6	8	10	8	0	8	0	4	0	0	20	17	22	11	22	20	22	6.181818	8.08	5	6.848	7
1531	KISS	E	ALPÁR	10	8	5	2	6	0	3	0	9	0	0	4	6.86	22	22	22	22	22	22	22	4.896364	8.3144	5.02	6.99664	7
1533	KOVÁCS	L.P.	PÉTER-RÓBERT	10	8	6	8	8	4	6	1	10	8	2	5.8	15.6	10	12	22	19	22	20	22	8.4	8.776	7.08	8.0976	8
1535	KUN	B	SZABOLCS-HUNOR	10	8	8	10	8	10	10	0	8.6	4	6	10	13.53	20	22	22	11	22	20	22	9.648182	9.8052	7.68	8.95512	9
1537	LÁZÁR	S	BOTOND	10	8	6	4	10	10	5	1	6	8	6	8	16.06	22	20	22	22	22	22	22	8.914545	10.0024	7.5	9.00144	9
1538	MÁTÉ	M	CSABA	10	2	8	8	9	10	8	5	6	0	6	5	9.86	20	22	20	11	22	20	22	7.896364	8.9544	5.52	7.58064	8
1540	MELEGA	T.F	ERVIN-TIBOR	10	5	9	2	2	4	4	1	0	0	0	0	0	20	16	22	22	22	20	22	3.381818	7.248	4.16	6.0128	4
1541	MERK	S	ARNOLD	10	2	8	8	6	8	4	10	8	6	6	8	13.66	10	12	22	19	22	20	22	8.841818	8.9704	5.48	7.57424	8
1543	MÉSZÁROS	S	ISTVÁN ÁBEL	10	2	8	5	6	4	6	10	3	5	6	8.2	13.46	20	16	22	22	22	20	22	7.896364	9.2344	6.4	8.10064	8
1545	MINOR-SOMLAY	K.A	SZILÁRD	10	7	7	7	0	6	7	1	7	4	0	5.2	13.4	22	22	20	22	22	22	20	6.818182	9	6.48	7.992	8
1546	MOCAN	SZ L	ZAKARIÁS	10	0	8	6	2	8	8	0	4	0	8	5.2	10.6	22	22	22	19	22	16	22	6.345455	8.592	6.94	7.9312	8
1553	NEMET	L	ORSOLYA	10	6	2	4	8	10	2	0	4	0	2	5.2	9.26	20	16	22	22	22	12	22	5.678182	7.9384	6.26	7.26704	7
1556	PATAKFALVI	CS	ÖRS-KRISZTIÁN	10	6	9	10	7.2	10	7	10	9	10	6	10	17.06	22	22	22	22	22	22	22	11.02364	11.0104	7.3	9.52624	10
1557	PELEI	A.-A.	ELÖD	5	0	0	0	0	0	0	0	7	2	2	5.4	9.2	20	0	0	11	0	20	22	2.781818	4.144		2.4864	3
1560	PETKES	I	ZSOLT-JÓZSEF	10	0	9	8	8	8	6	0	5	8	6	8.2	3.26	22	22	22	22	22	22	0	7.223636	8.4584	6.16	7.53904	8
1562	RACZ	A	ORSOLYA	10	7	6	7	10	9	9	0	8	6	10	7.2	11.86	22	22	22	19	22	16	22	9.187273	9.8424	6.14	8.36144	8
1563	RÁDULY	S	ZALÁN	10	8	4	8	6	10	8	10	8	4	4	6	11.86	22	20	22	22	22	22	22	8.86	9.9784	7.02	8.79504	9

You will get your grades in a spreadsheet on a dropbox link (I cannot make this publicly available because of GDPR regulations)

Course material, grades

There will be a link to a dropbox folder where you can find the grades spreadsheet, the course material and the Lab programs

Dropbox › Microcontrollers class

https://bit.ly/2y900KQ

Name

CH341SER.EXE

Keypad_decode.ino

Keypad.zip

Morse_code.ino

Motivation

Why would a programmer need to learn about microcontroller programming?

- **microcontrollers**

  The microcontroller is a small computer that was embedded into a system for a very specific purpose. It is usually only good for this purpose and it is not faster or more expensive than what is necessary for this specific purpose

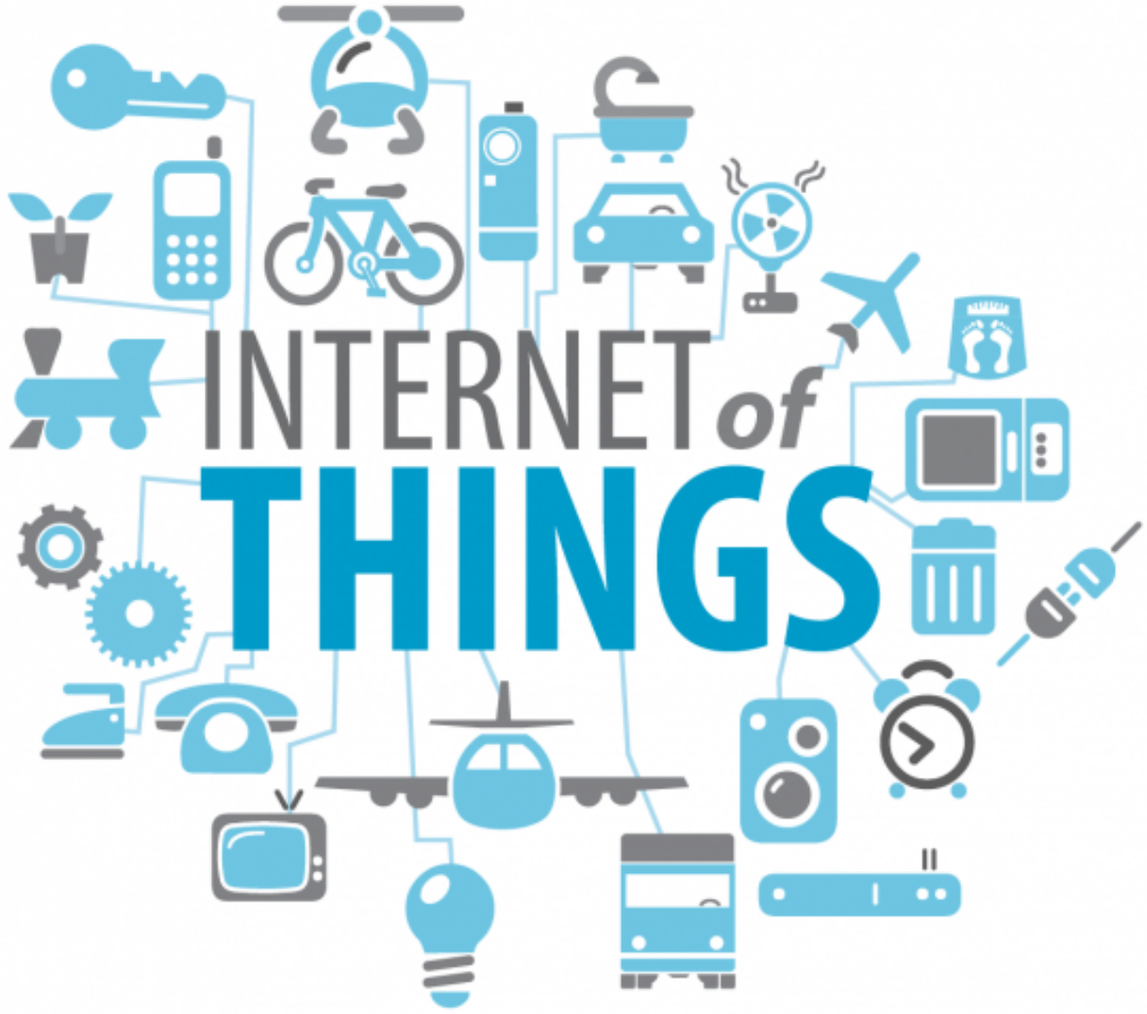
- **automatization**

These microcontroller systems are becoming cheaper, smaller and consume less and less power for the same computing capability. Because of this trend, they are making their way into more and more applications. More than half of the processors made today are microcontrollers - they sold 2 billion ($2x10^9$) microcontrollers in 1997 and more than 4 billion ($4x10^9$) in 2002. Most microcontrollers are 8 bit systems but with the development of IoT (internet of things) they are switching to 32 bit microcontrollers currently.

Motivation

- **IoT (internet of things)**

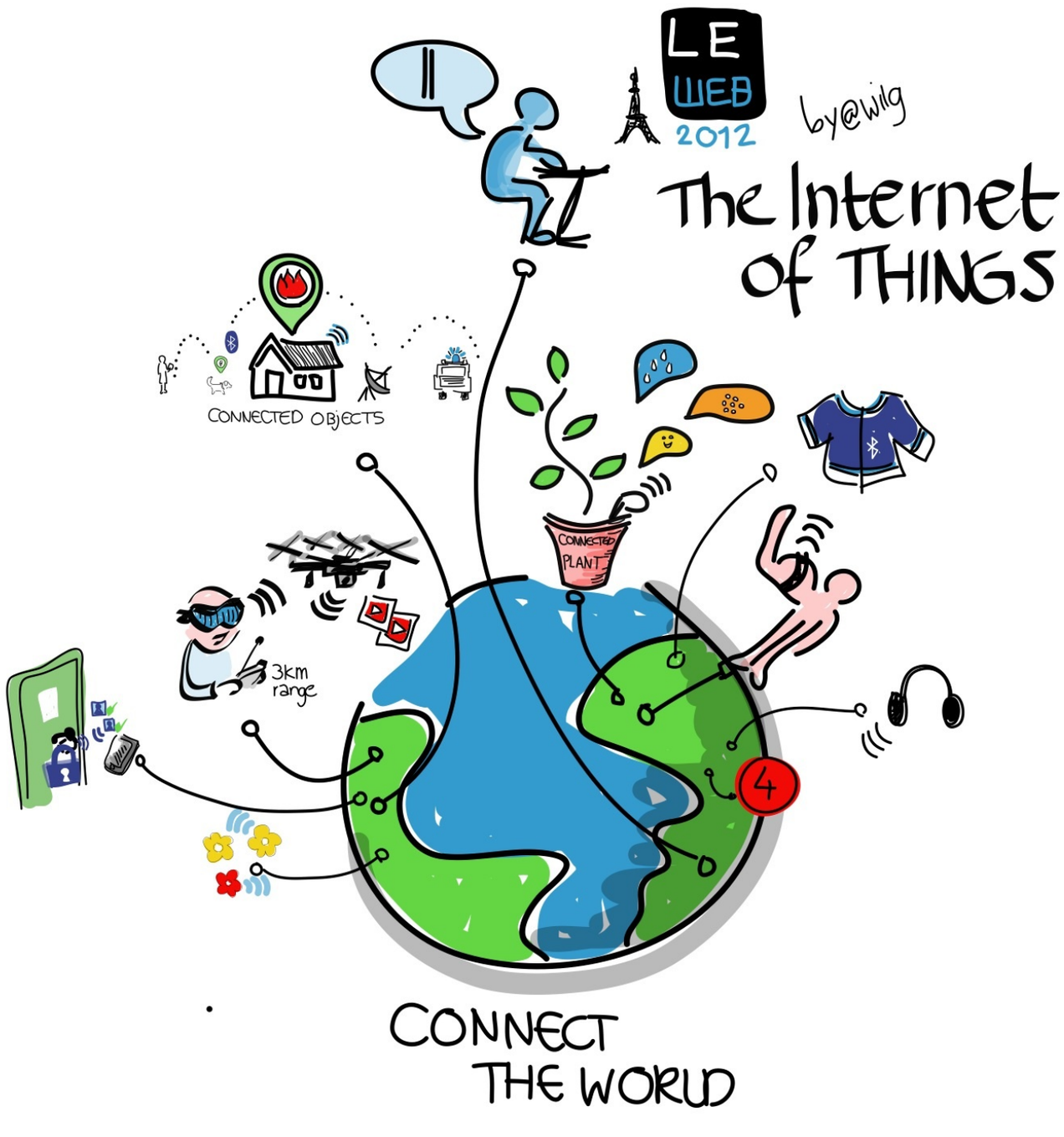
  These are systems built with microcontrollers that can connect to the internet. For example a thermometer, a heating system for an apartment, a camera, or an entire smart home system



Kevin Ashton, MIT
Internet of Things == a system where the Internet gets connected to the Physical World by using sensors and actuators connected to the web

- We can measure different things, we can control different things through the web
- We can integrate the world into the computational realm
- Cyber-physical systems: smart grid, smart home, intelligent transportation and smart cities. (it is estimated that by 2020 we will have about 50 billion different things connected to IoT)

Motivation



- **Smart Grid**
- **Smart Home**
- **Intelligent Transportation**
- **Smart Cities**

The information that we gather helps us using our resources better, use less energy for the same task, reduce waste in both generating (smart grid) and using (smart home) electric power, decrease traffic jams in cities etc.

Motivation

- **Smart Grid**
- **Smart Home**
- **Intelligent Transportation**
- **Smart Cities**

Smart Grid: controlling the electric grid. How much wind or solar will be produced, how much is the demand from the users: we can supplement energy by generating less waste. This information can also help in pricing the energy to facilitate user behavior (for example at night (or maybe during the day?) the electric energy could be cheaper to balance out the base load and also charge electric cars).

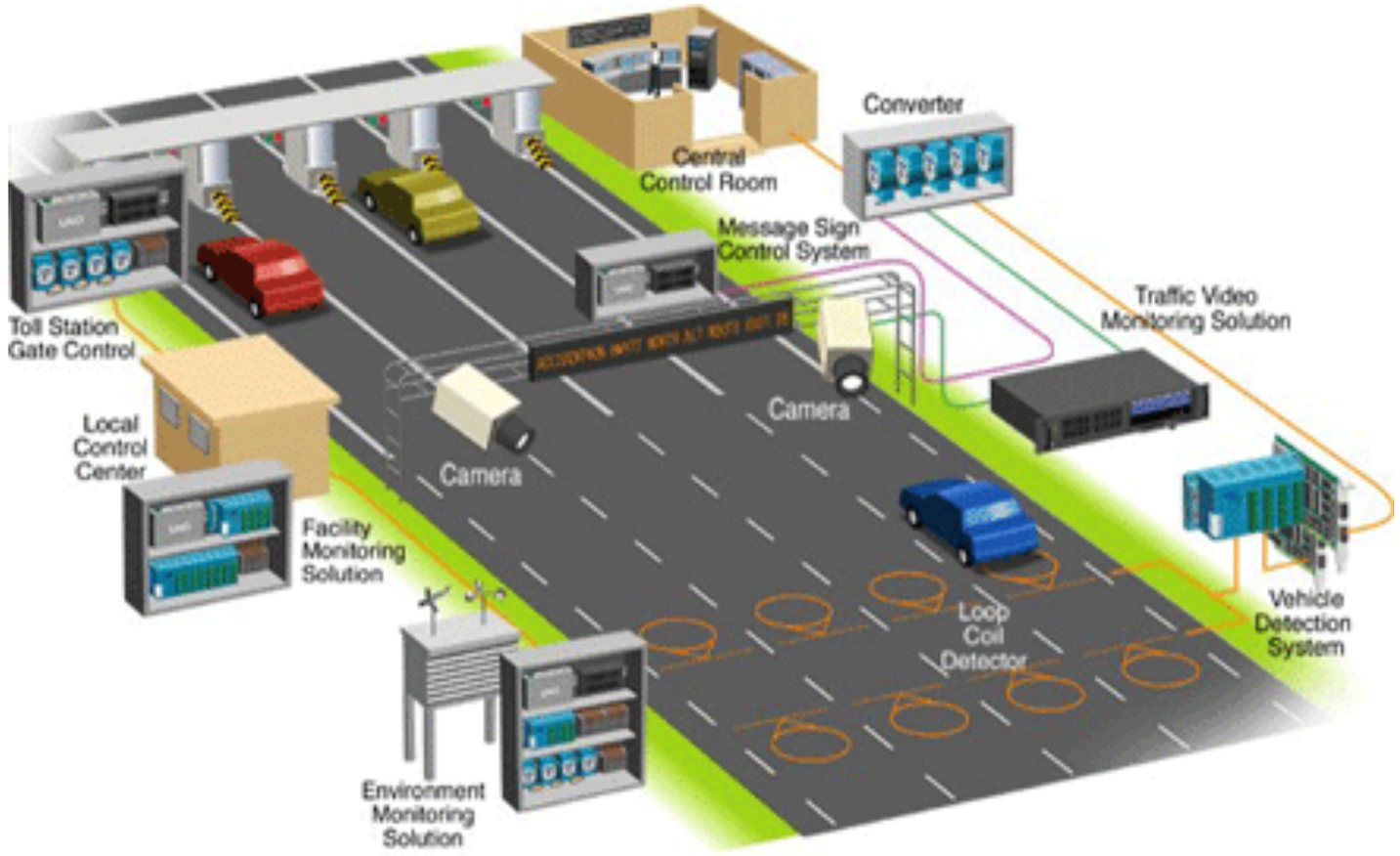
Motivation

- **Smart Grid**
- **Smart Home**
- **Intelligent Transportation**
- **Smart Cities**

Smart Home: setting the thermostat in the house, switching it on/off based on sensor data, setting the lights automatically based on sensors, ventialltion systems, monitoring the energy consumption of the home in a more detailed manner (having more data about the consumption even specific consumption), helping with shopping when the refrigerator knows the items inside it and knows their expiration date etc.

Motivation

- **Smart Grid**
- **Smart Home**
- **Intelligent Transportation**
- **Smart Cities**



Intelligent transportation: monitoring which road has what kind of traffic (sensors, cameras) setting the speed limit or the direction of the traffic lanes based on that or even setting the cost of entering the city based on that (Singapore). Based on weather reports set a speed limit on highways. Inside the city monitor the progress of buses and let users know when the next bus will arrive, etc.

Motivation

- **Smart Grid**
- **Smart Home**
- **Intelligent Transportation**
- **Smart Cities**

Amszterdam: állítható fényerő gyalogosforgalom alapján

Smart Cities: measuring pedestrian traffic the lights can be dimmed for example this happens in Amsterdam. Reading the water level/the humidity of the soil in the parks in the city so gardeners would know the current need/situation (Barcelona) or gathering data on people's movement before deciding on new bus routes. Having smart traffic lights that create a green wave for ambulances/ fire trucks and police. Santa Cruz, CA: the patterns of police car patrols are set based on the information gathered on crimes.

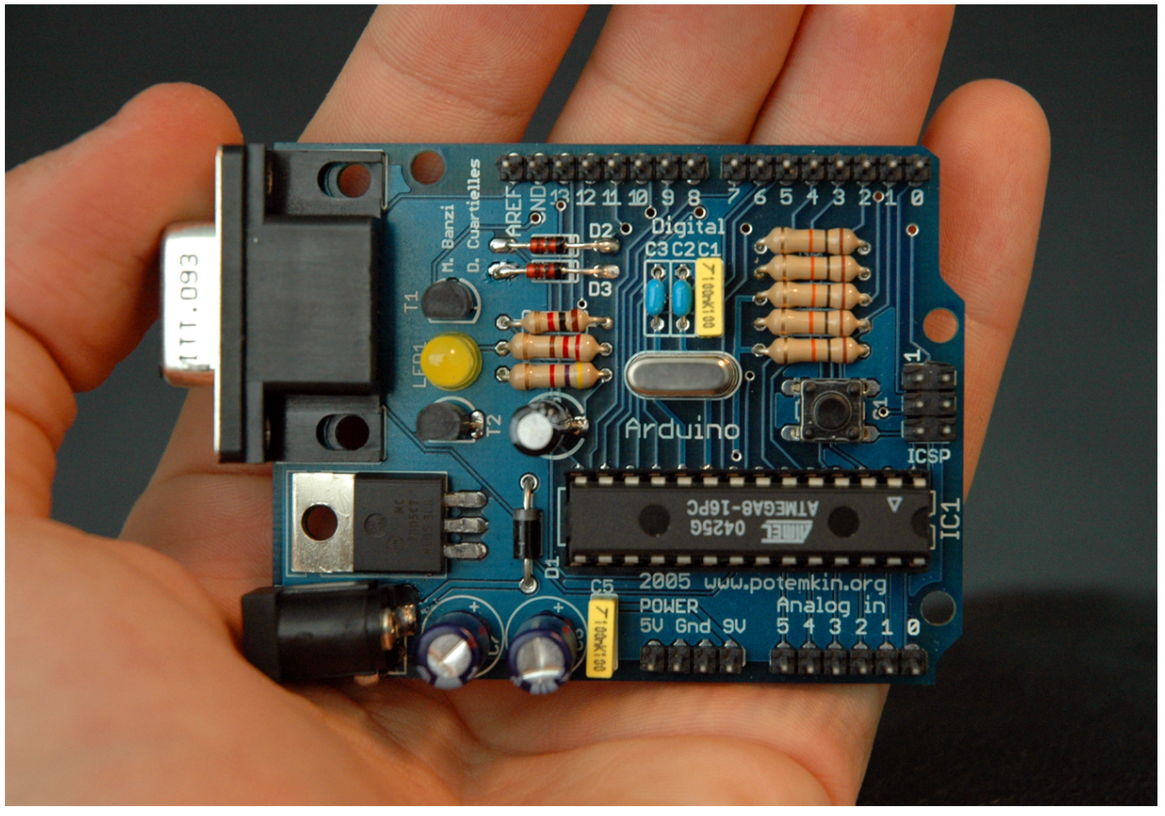
Microcontroller systems

Microcontroller: a computer that is written on a chip. The question is, how to choose the appropriate system: low consumption, small, cheap but it is capable of the task that we want it to perform

Wide scale of possibilities: simple, small, cheap, low computing power
Complex, complicated, expensive, high power consumption and high computing power

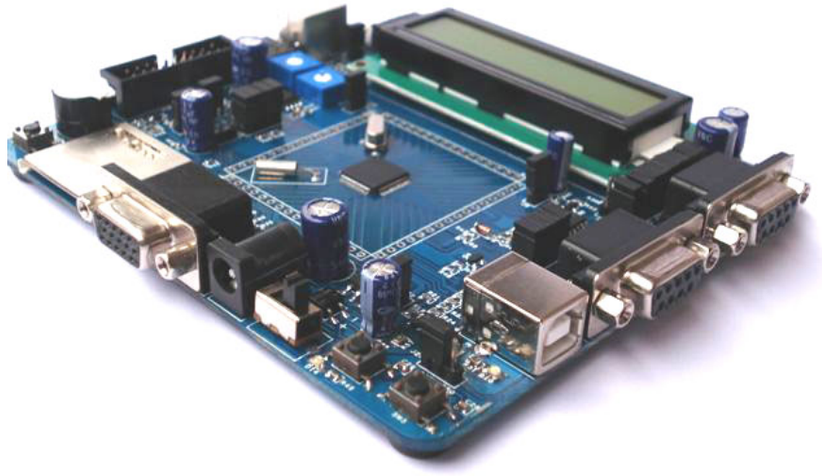
Older design/newer design

The new designs are pushing the old ones out of the market: PIC and 8051 are only found in education, Atmel is in the industry but switching to newer designs and ARM is also taking over covering a very large gamut from simple microcontrollers to processors for cell phones



Many times we want to save on resources: we want the minimum price, power consumption etc. for the task at hand. Sometimes reaction time matters we want the system to behave real time. For this purpose **it is useful to know about the internal structure of the microcontroller and it is good to know how to program it efficiently at low level (program registers in C)**

Embedded system



Microcontrollers are optimized in a different way than general purpose computers. They are usually **embedded=contains everything needed for their functioning**. They will not be augmented later with peripherals or used for a different purpose, they will not be reprogrammed etc.

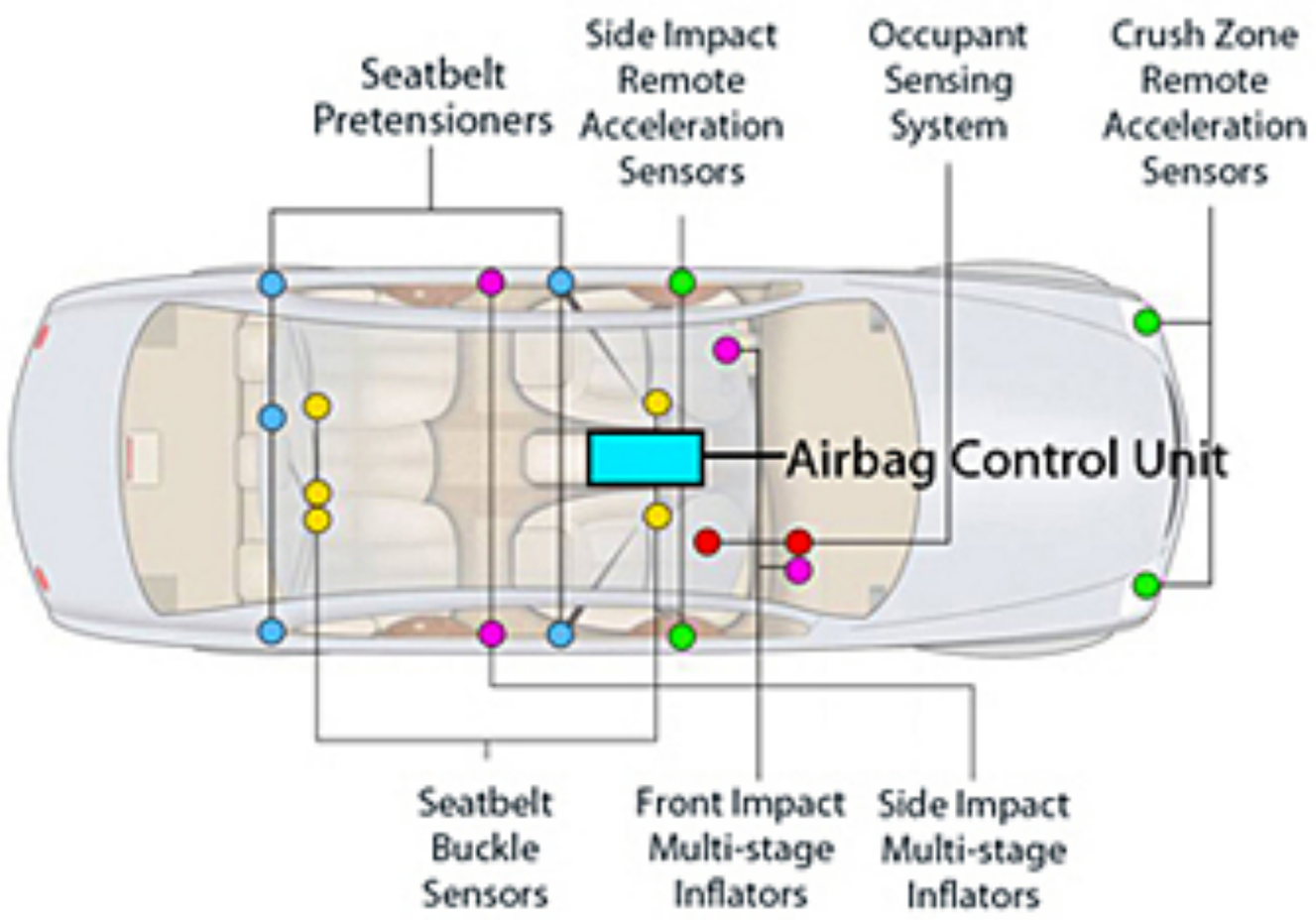
What does an embedded system contain? the processor
input/output (GPIO pins, connection possibilities accoring to different protocols)
memory: store the program (flash/ROM) (program)
small general purpose memory (RAM) (data, variables)
sensors, actuators, displays (ex. LCD), push buttons, what is needed for the task

Examples

Let's think for every use case on what we need to measure (sensors) and what we need to control (actuators)

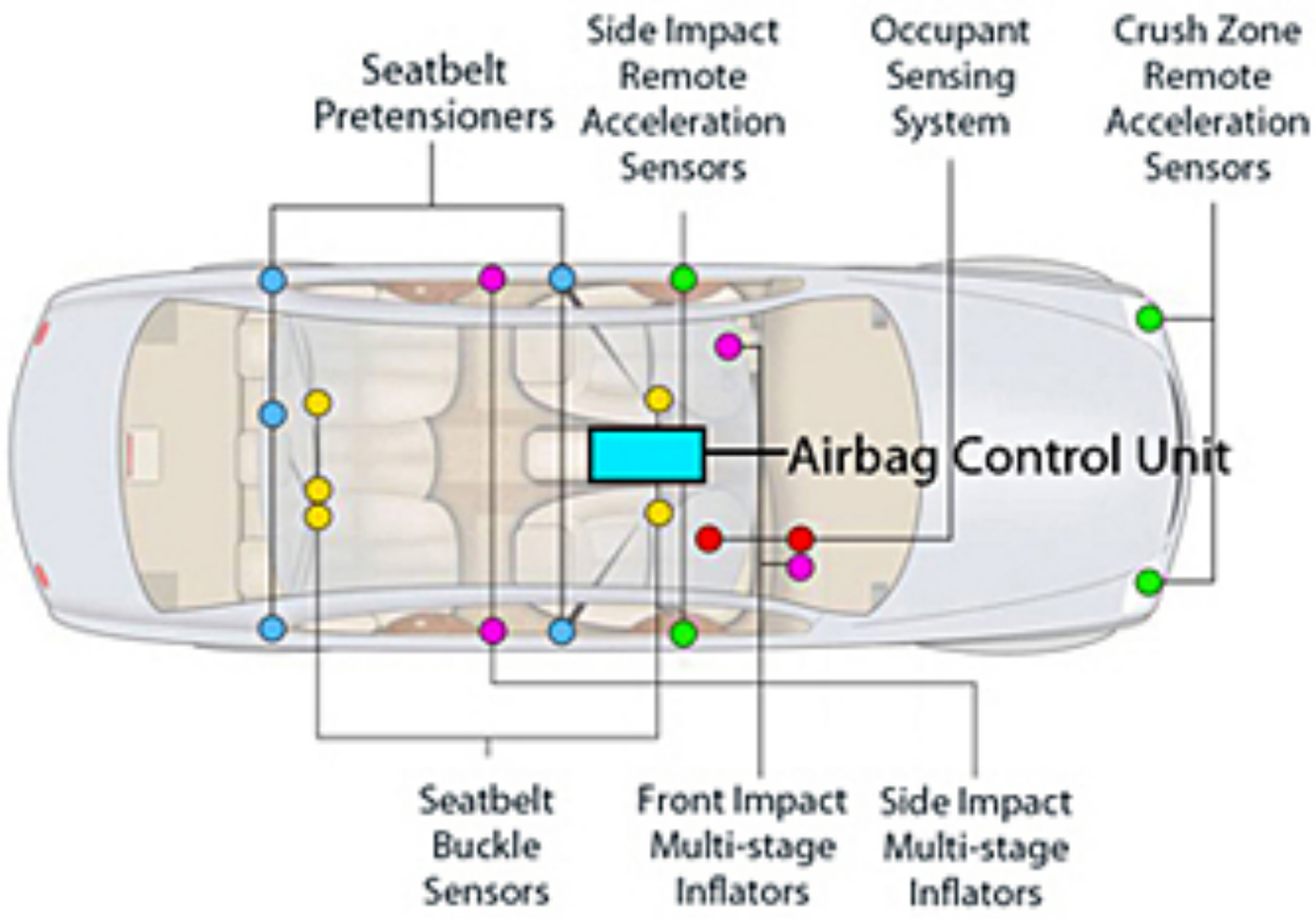
Washing machine
Thermostat
Electric oven
Bread making machine

Microcontrollers in cars



What is critical in such a microcontroller system for example the one that controls the airbags? What would be most important?

Microcontrollers in cars



What is critical in such a microcontroller system for example the one that controls the airbags? What would be most important?

The critical part is the **real time functioning**: we can only wait a certain very short time after impact before the airbags are deployed. The code should not delay too much. Some systems even start pulling the seatbelts, straightening the seats and closing the windows shut when they decide that a collision is inevitable (Mercedes), and once airbags are deployed and a serious crash is detected the system might even dial emergency (ambulances) automatically

In programming sometimes the Time to Market matters (showing off a concept to a customer) so the most important part is the comfort of the programmer and the ease of programming. In these microncontroller systems the fast error-free operation is more critical.

Optimizing microcontrollers

What is critical in the microcontroller system built into a car key fob? What would the main concern be to keep the customer satisfied with the product?

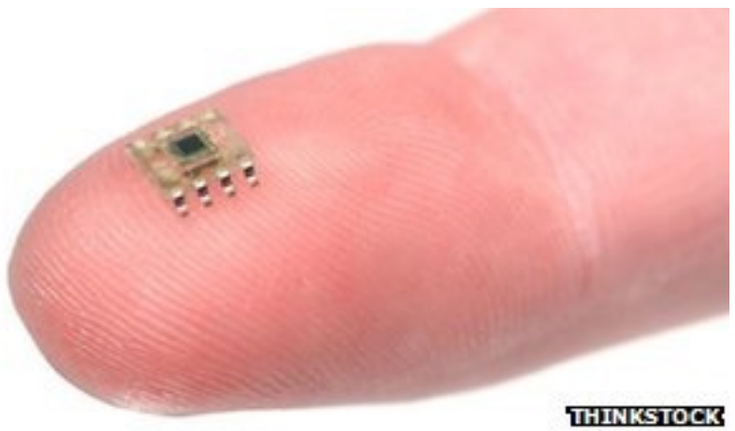
Optimizing microcontrollers

What is critical in the microcontroller system built into a car key fob? What would the main concern be to keep the customer satisfied with the product?

The crictical part is the **low consumption**: we don't want it draining the battery when it is not in use. There are different sleep modes that the microcontroller has, in these modes almost everything is switched off except one part that can wake the system when the user pushes a button.

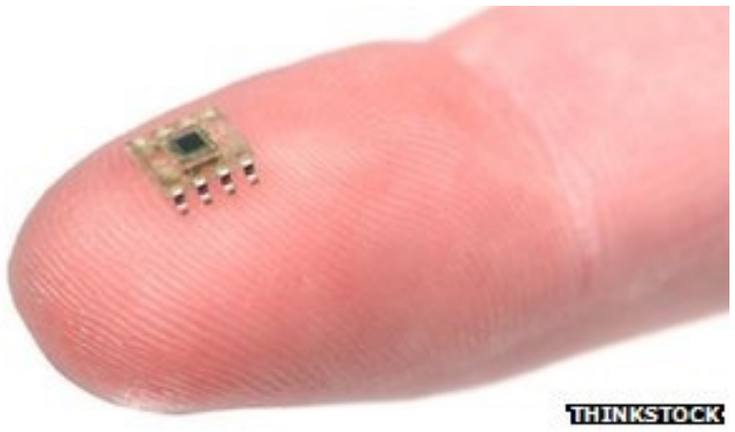
To write such a code we need to know the internal settings of the microcontroller to be able to switch off unwanted functionality (this is done sometimes to save power even when we are not using sleep mode) and also to put the microcontroller in a sleep mode.

Optimizing microcontrollers



What is important in a microcontroller that we want to embed into a drone?

Optimizing microcontrollers

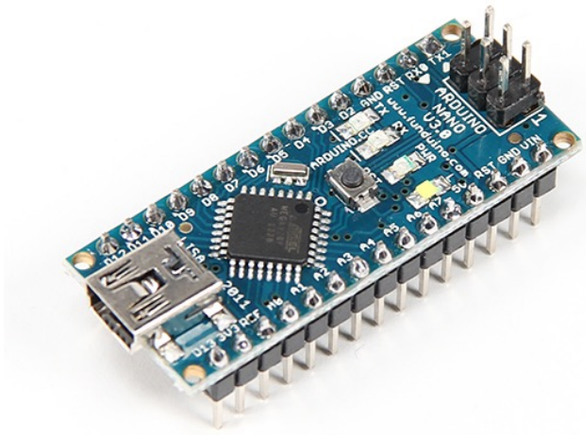


What is important in a microcontroller that we want to embed into a drone?

The critical part is **low power consumption** but we also need to find a solution that has a **small** form factor and also it is important that it would be **light (small weight).** This inevitably will mean a restriction on the memory, on the computing power etc of the chip so the code needs to use the processor's capabilities fully.

To do that we need to know the microcontroller's inner workings and we need to be able to program it low level or at least understand what is going on low level in the microcontroller.

Optimization



General Purpose Computer:

it should be good for many different purposes, you should be able to attch it to many different and varied peripherial units

should have a high computational output

small consumption (for desktops this is less of a criteria)

It should be as good as possible (screen resolution, clock speed, number of cores, ports available etc.)

Microcontroller system:

Should be good for the task at hand. Because it will be built in, it is likely that the software will be the same throughout the lifecycle of the product.

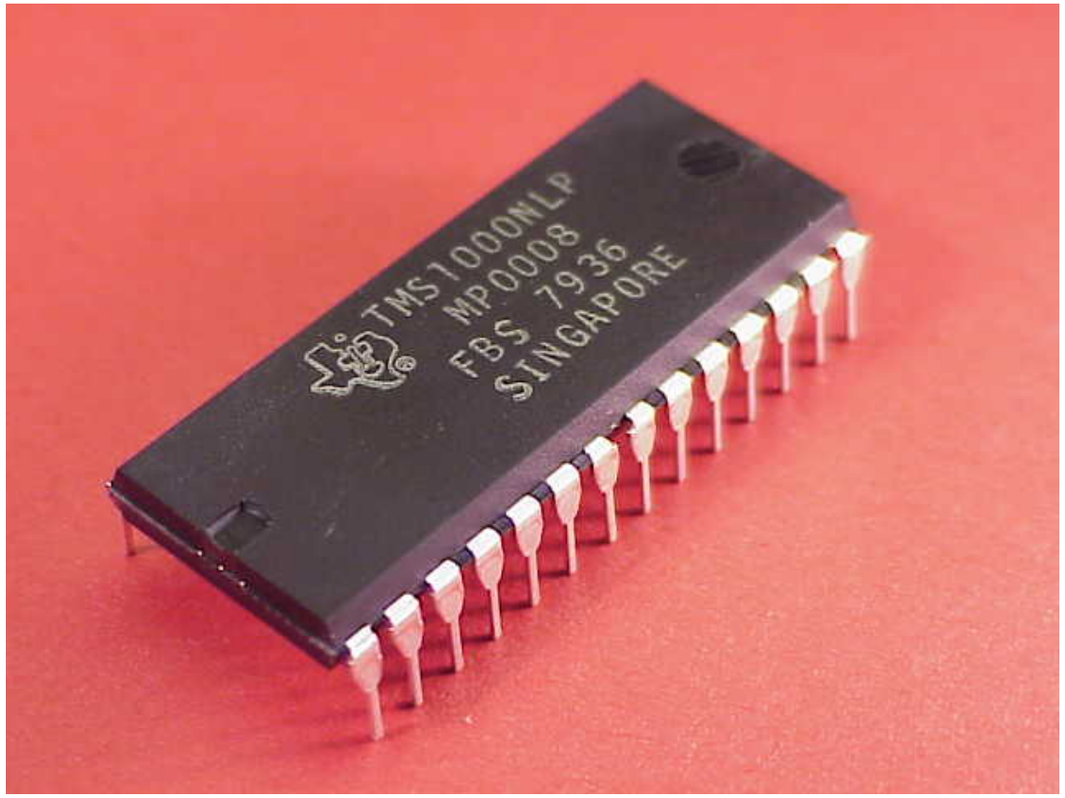
Cheap (around $1)

Small size

Low consumption

It should be robust (industrial applications): should be able to withstand vibrations, dust, humidity, temperature changes etc.

History

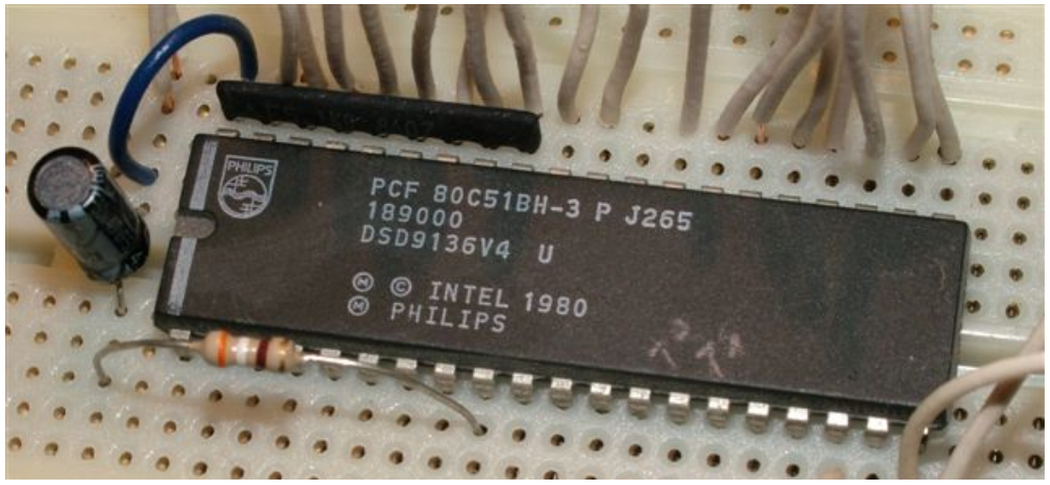


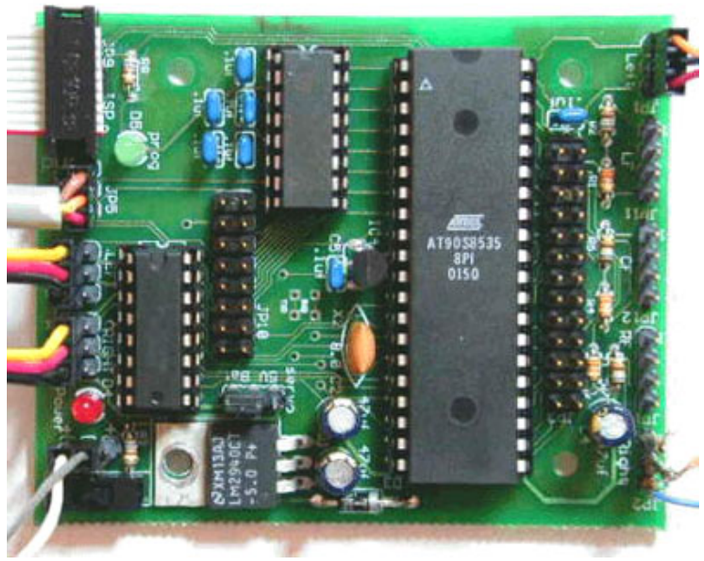
1971 Intel 4004 és 8008 ancestors of modern computers - these were very expensive not the chips themselves but all the peripheric units built around them

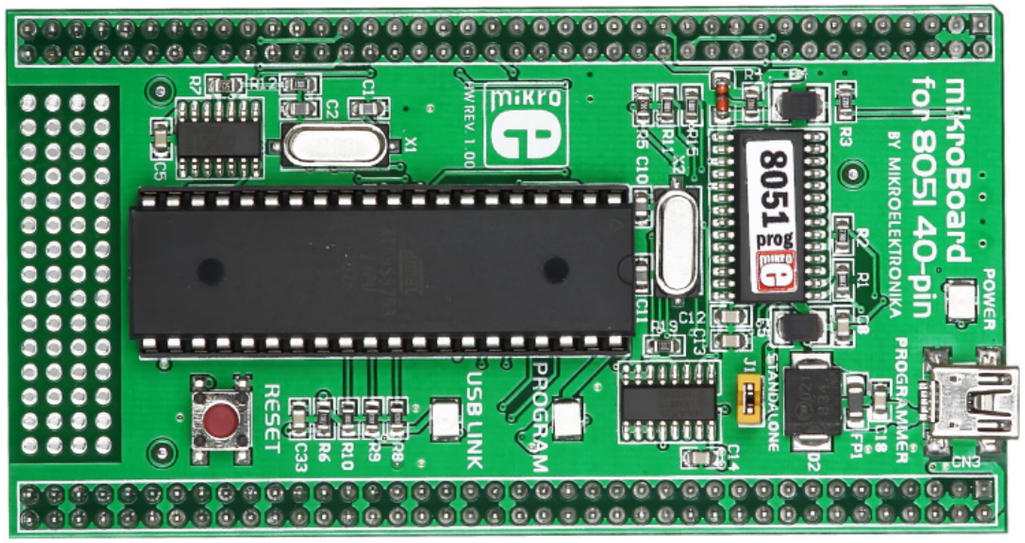
They developed a different line TMS1000, then the Intel responded with the Intel 4048 - this was built in to the keyboards these are the ancestors of today's microcontrollers

We can say that general purpose computers and microcontrollers appeared at about the same time, they started developing them at the same time

History





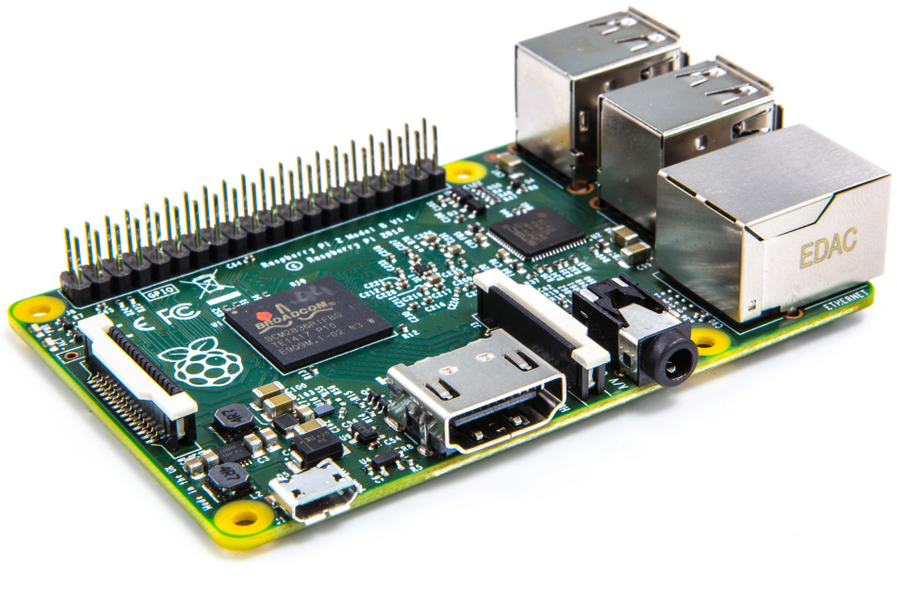


8084, 80C51 illetve 8051. A 8051et régebb autóiparban használták és még ma is oktatásban

PIC - only in education
Atmel microcontrollers - they are used in the industry and the Atmel company is developing new lines continuously - we will learn about this line of microcontrollers (this is what is built in to the Arduino)

ARM: wide possiblities from very simple processors to processors built into mobile phones

Rapsberry Pi and similar systems

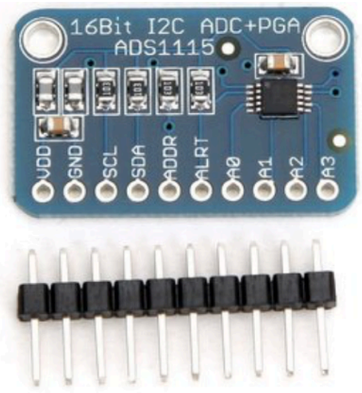


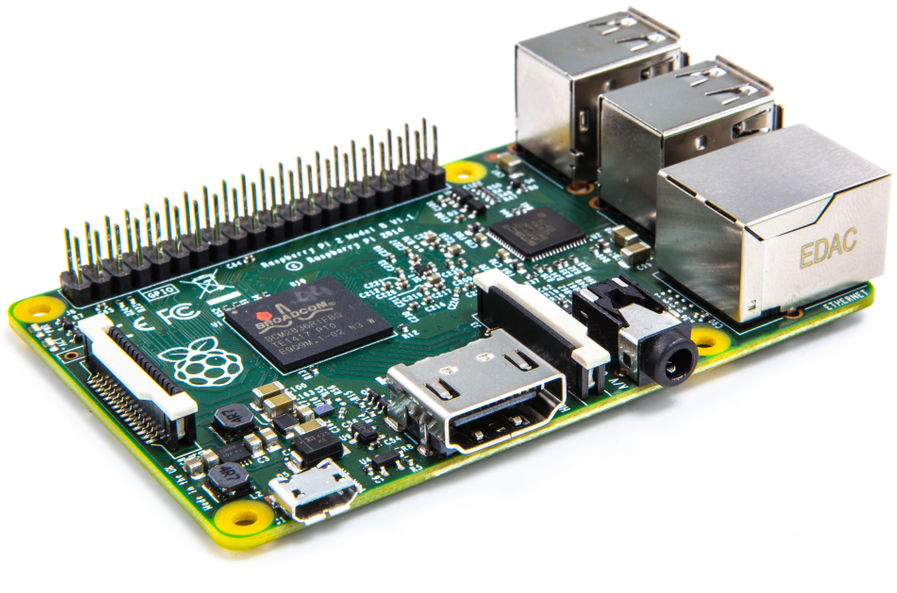
A Rapsberry Pi 2 can run a full Hd 1080ps video, it has usb ports (4 ports) it has an ethernet port and it runs linux. The processor is an 900MHz ARM Cortex-A7 chip and it has 1Gb of memory and it costs about $35

Why would I learn how to program a microcontroller in C if I can just buy a rapsberry pi and program conveniently in the only language I know and I am familiar with (for example in Java)

We have given an answer to that already. You might solve many embedded projects with a rapsberry by just buying an expensive hardware and not caring about power consumption. But even in such cases some things that we learn (for example how to convert an analogue signal to digital) will help you. But in some cases you will be able to deliver much better results by just paying a little attention to what is happening on a lower level and choosing a hardware that is cheaper and more suited to the task.

Rapsberry Pi and similar systems





A Rapsberry Pi 2 can run a full Hd 1080ps video, it has usb ports (4 ports) it has an ethernet port and it runs linux. The processor is an 900MHz ARM Cortex-A7 chip and it has 1Gb of memory and it costs about $35

Why would I learn how to program a microcontroller in C if I can just buy a rapsberry pi and program conveniently in the only language I know and I am familiar with (for example in Java)

We have given an answer to that already. You might solve many embedded projects with a rapsberry by just buying an expensive hardware and not caring about power consumption. But even in such cases some things that we learn (for example how to convert an analogue signal to digital) will help you. But in some cases you will be able to deliver much better results by just paying a little attention to what is happening on a lower level and choosing a hardware that is cheaper and more suited to the task.

Real Time functioning

Many times we want real time functionality from our embedded systems. To achieve that, the microcontrollers are built with

predictable latency : this means that we can predict how many clock cycles will it take to get an answer from the microcontroller. Depending on the previous operation it is doing this could be 1-2 or max 8 clock cycles and then it will respond

general purpose processor:
3 Ghz 0.33 ns one clock cycle
mikrokontroller:
16 Mhz 62.5 ns one clock cycle

With a general purpose processor we could have some other things happening, some interrupt, another application grabbing the processor for a while, a cache miss and looking for data in the memory for 40-300 cyctles etc. We are not guaranteed and answer in real time. With a microcontroller we are.

To achieve predictable latency we need to have operations that take only a few clock cycles. We cannot have very complex operations in the processor. This is why most microcontroller systems are RISC (reduced instruction set computing) architecture processors.

Real Time and ISR (Interrupts)

To have **real time** functioning, it is important to eliminate the possibility of something else grabbing the processor. This is done easily as all the interrupts are set by us we can set which interrupts are possible and which are not. Tio achieve real time we can have no interrupts at all or maybe one interrupt that can access the processor (and we put our critical function in that interrupt). In our system interrupts cannot be nested and interrupt call cannot be made if the processor is already in an interrupt.

We can set all interrupts on/off and we have complete control over what can happen in the system. (Atmel Atmega16 or Atmega 32 have a total of 21 interrupts as listed here).

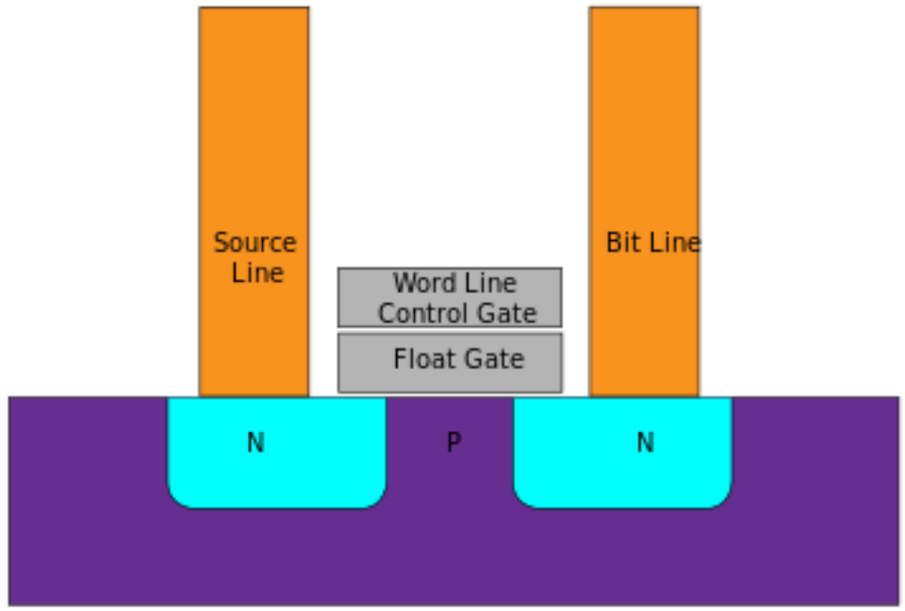
ISR: Interrupt Service Routine

Table 18. Reset and Interrupt Vectors

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	INT2	External Interrupt Request 2
5	$008	TIMER2 COMP	Timer/Counter2 Compare Match
6	$00A	TIMER2 OVF	Timer/Counter2 Overflow
7	$00C	TIMER1 CAPT	Timer/Counter1 Capture Event
8	$00E	TIMER1 COMPA	Timer/Counter1 Compare Match A
9	$010	TIMER1 COMPB	Timer/Counter1 Compare Match B
10	$012	TIMER1 OVF	Timer/Counter1 Overflow
11	$014	TIMER0 COMP	Timer/Counter0 Compare Match
12	$016	TIMER0 OVF	Timer/Counter0 Overflow
13	$018	SPI, STC	Serial Transfer Complete
14	$01A	USART, RXC	USART, Rx Complete
15	$01C	USART, UDRE	USART Data Register Empty
16	$01E	USART, TXC	USART, Tx Complete
17	$020	ADC	ADC Conversion Complete
18	$022	EE_RDY	EEPROM Ready
19	$024	ANA_COMP	Analog Comparator
20	$026	TWI	Two-wire Serial Interface
21	$028	SPM_RDY	Store Program Memory Ready

http://stackoverflow.com/questions/24128926/what-happens-when-an-isr-is-running-and-another-interrupt-happens

Embedded:
On Chip Memory



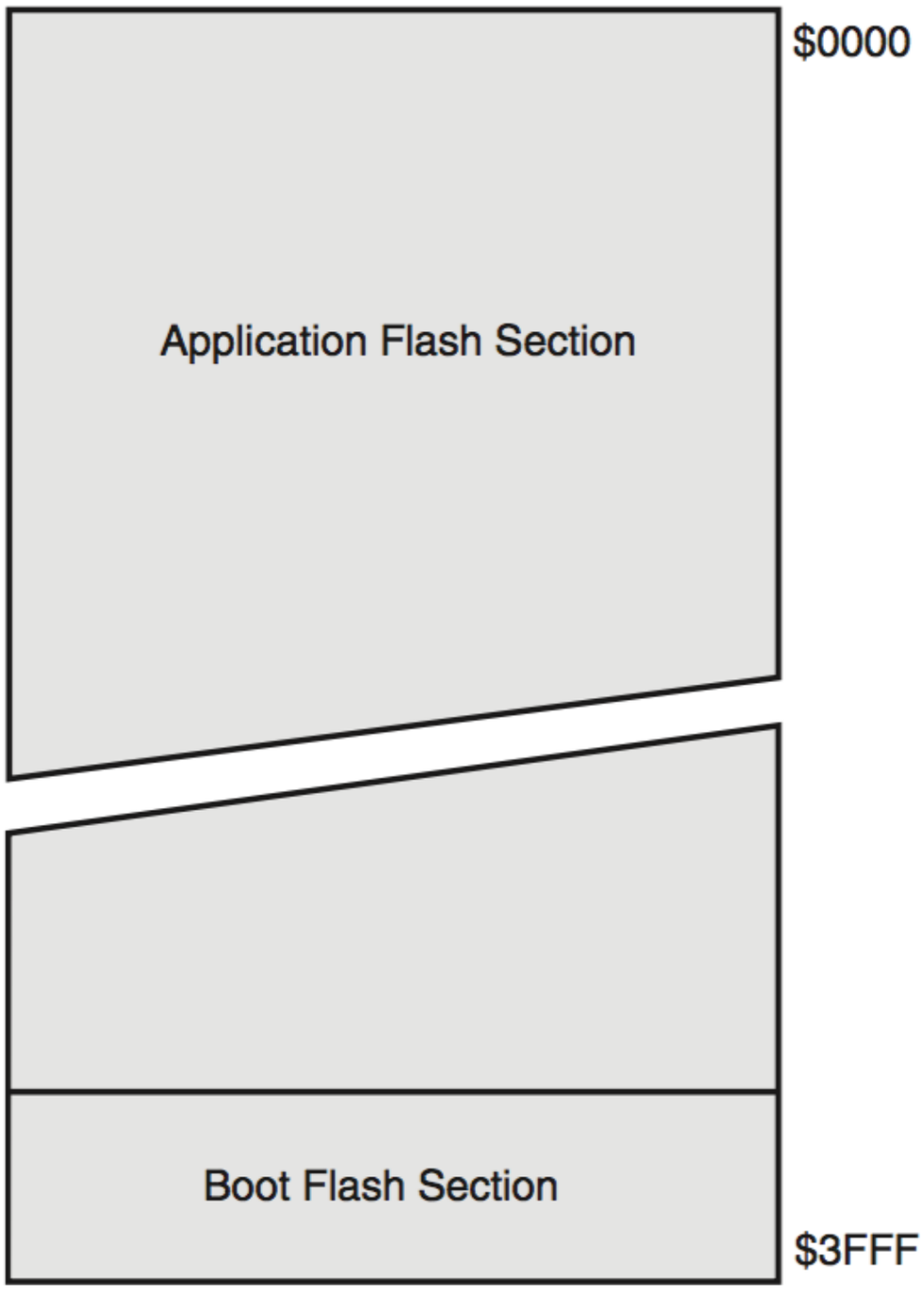
In an embedded system we have the memory in the same place as the processor. This is called on chip memory. The type of memory used for this is flash memory.

There were other types of memory such as ROM, PROM, EPROM, EEPROM in the past. EEPROM is still used today, there is a small portion of EEPROM built into the Arduino that you can use for logging data or storing parameter settings. EEPROM is also available as an external small chip that we will program using the TWI protocol.

One bit of a flash memory
This retains its value even if power is lost

The code we write on the microcontroller remains in the flash memory. When we reset the board the same code will start executing again.

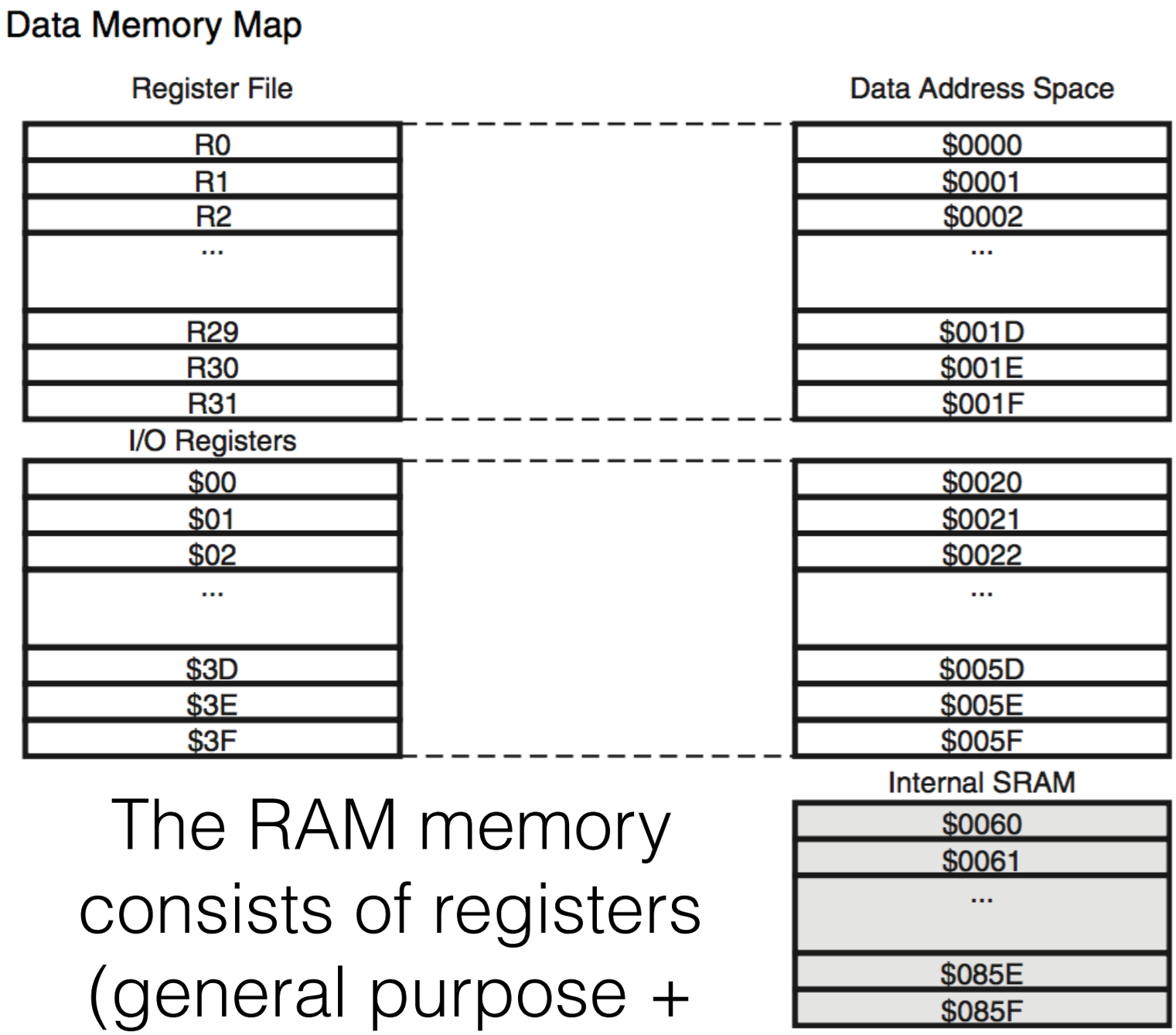
Embedded: On Chip Memory



Some part of the on chip memory will contain the code that we program the microcontroller with. When programming a microcontroller it is not a bad practice to hard code data into the program: this is written on the flash memory part that is larger than the volatile RAM memory. The volatile RAM memory can be as small as 1k but the flash memory holding the program (and hard coded data) can be16K, 32K this is what the name of the processor means the size of the flash mempory Atmel Atmega 16, 32 , 64 Atmega 328P (the one in the Arduino) also means 32k flash memory.

Here we can see that part of the memory is occuppied by the bootloader. This is a small code that checks if our incoming data is actually a coding data instead of communication and makes it possible to program the microcontroller through the same usb port where the data transfer happens. Pro: easier Con: takes up a bit of space.

Embedded: On Chip Memory



The RAM memory consists of registers (general purpose + specific)

The variables we use in our code are being stored in fast SRAM type (volatile) memory. This is where the registers are also stored that are specific in the system (they have specific names) and we can use them to give commands (control registers) check on the status of things (status registers) or transmit and receive data (data registers).

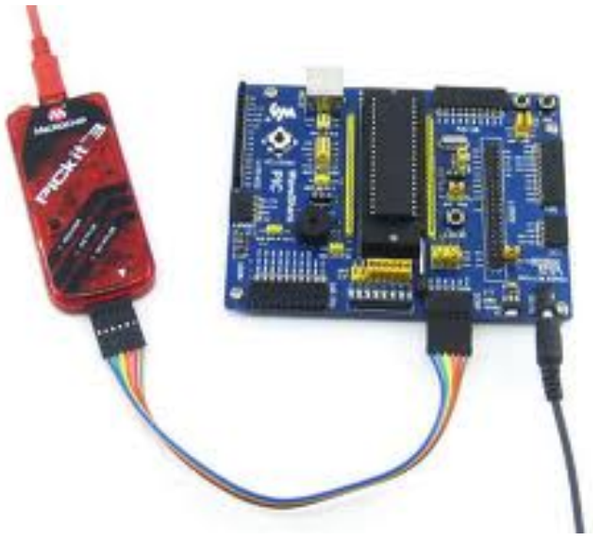
The Atmel microcontroller is a register based architecture.

Programming the microcontoller

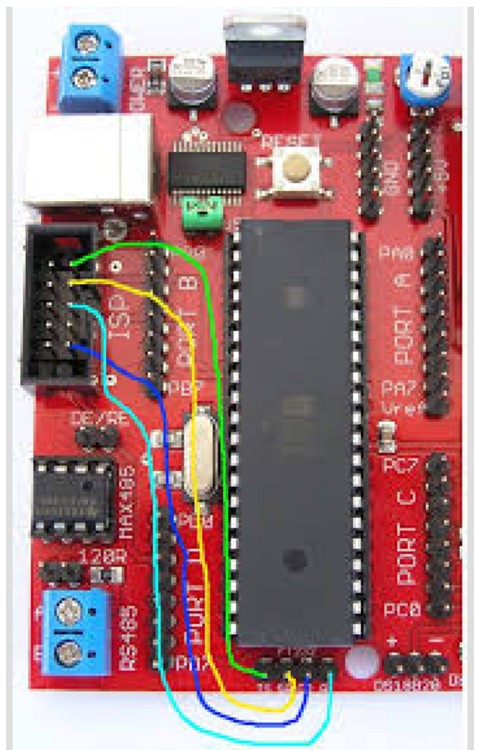
We will write the code in C language or the Arduino language (whoch is C + some libraries). The coding is done more easily in the **Arduino IDE** which includes a lot of pre-writtren examples and has the drivers for many different Arduino boards. The uploading of the code is also easy in this environment because it has a compiler and a built-in avrdude programmer code that can communicate with the bootloader. The Arduino IDE is free and it is available in all major platforms (Windows, Linux and Mac).

Another alternative for Atmel, ARM and other microcontrollers using the **Atmel Studio**, this looks like Visual Studio-ra épül but it is available only under Windows.

Another alternative is to use the **Atom editor** and dowload the **PlatformIO IDE** package that allows for the programming of microcontrollers in this environment.



- programming with a programmer
  advantages: more flash space.
  disadvantages: you need the separate programmer board



- with boot loader
  advatages: you only need the one usb cable and one usb slot to do both programming and serial communication (emulated on the USB) The program in the background doing the programming on the computer end is the avrdude (built in to the Arduino IDE, also available separately as a command line tool)

code

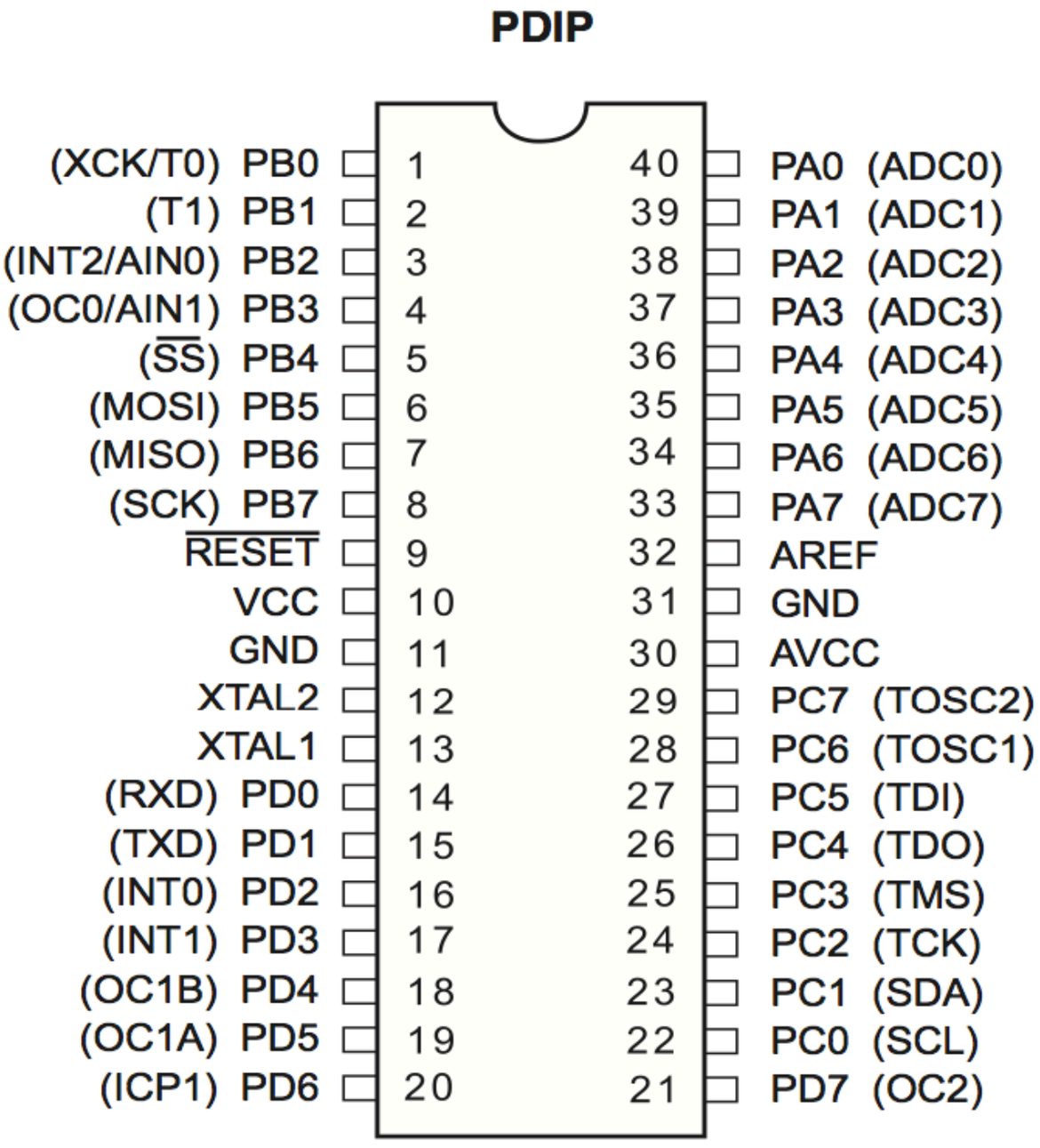
compiling

hex file

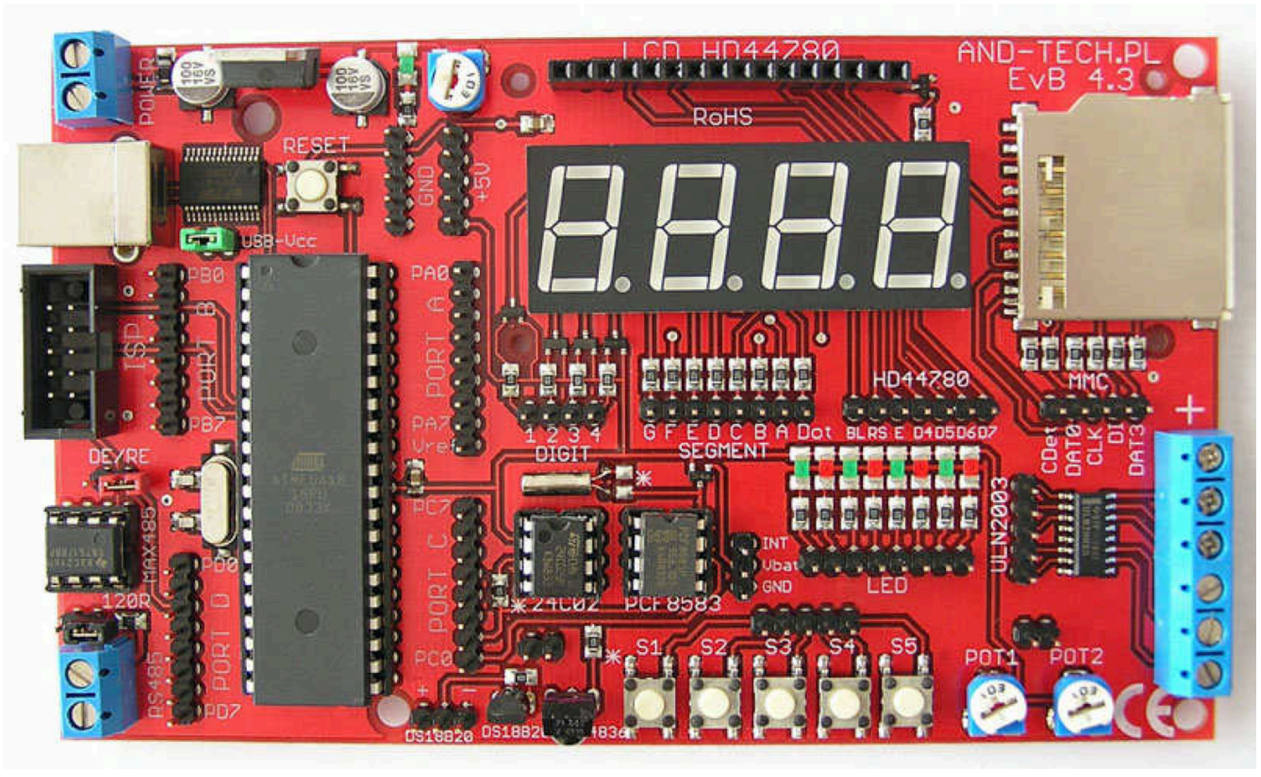
programmer or bootloader

program on the µcontroller

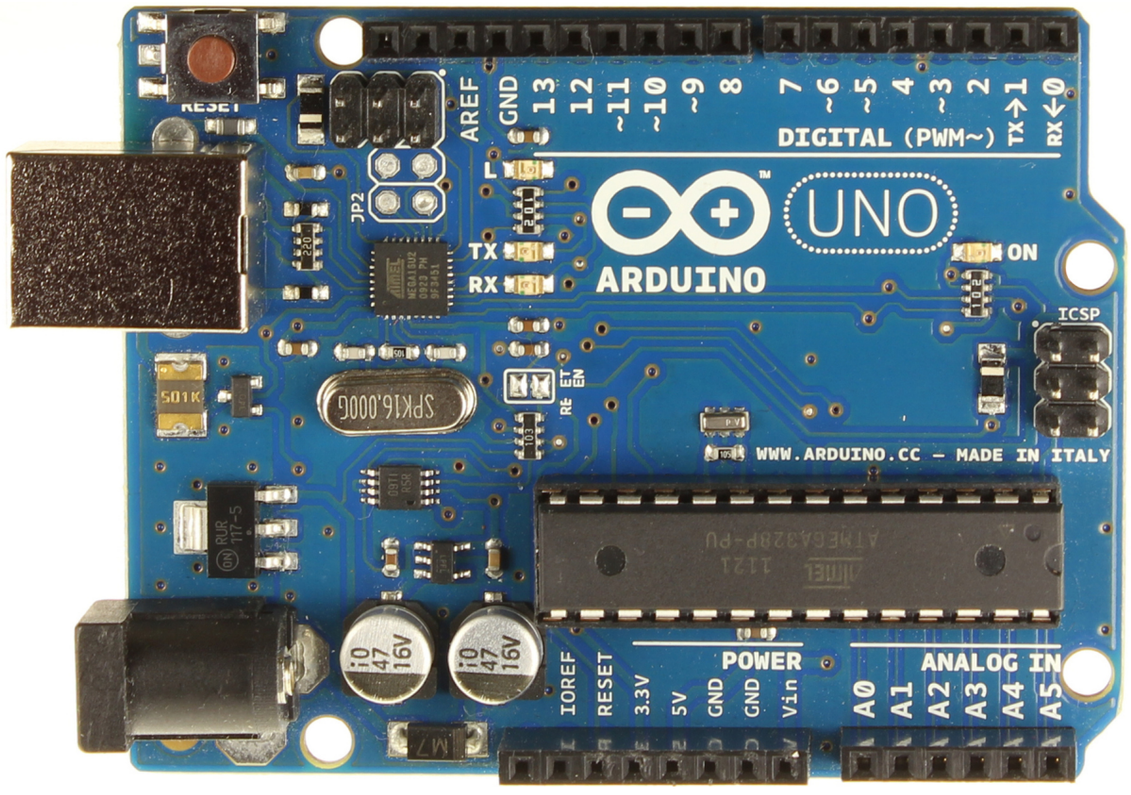
Programming the microcontoller



This is the EvB 4.3as board that I used a couple years ago on the Labs. It has peripherial uints already built in.



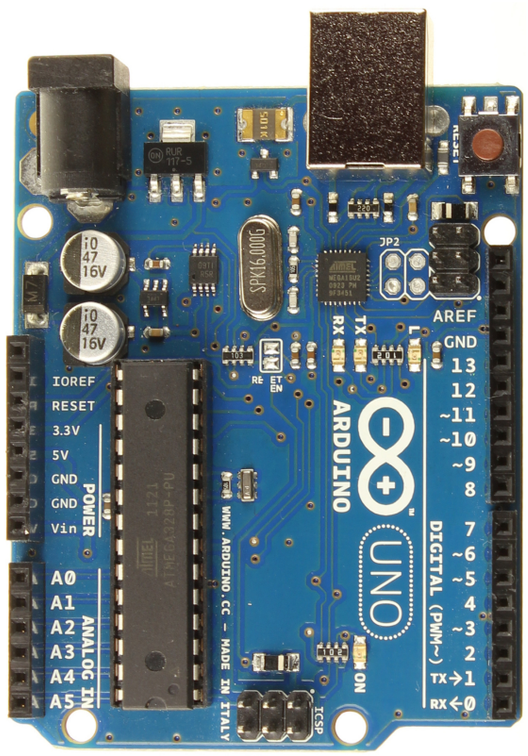
Programming the microcontoller



Arduino Uno with an Atmel Atmega 328P chip

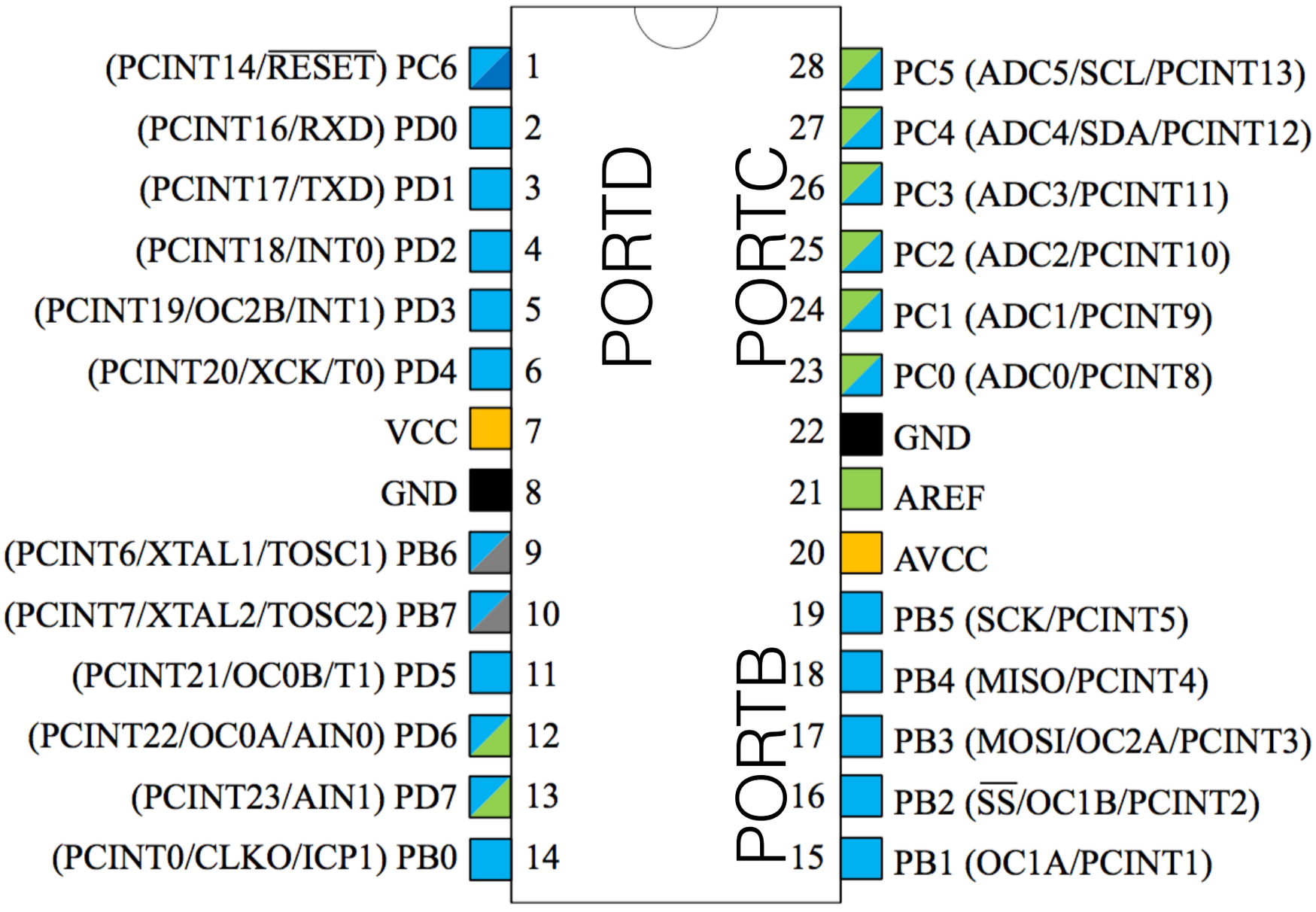
This is the Arduino Uno board that contains an Atmel Atmega 328P chip. This is very similar to the Atmel Atmega 32 chip on the EvB 4.3 board the only difference is in the number of PORTs and PWM outputs

Arduino Uno Pins



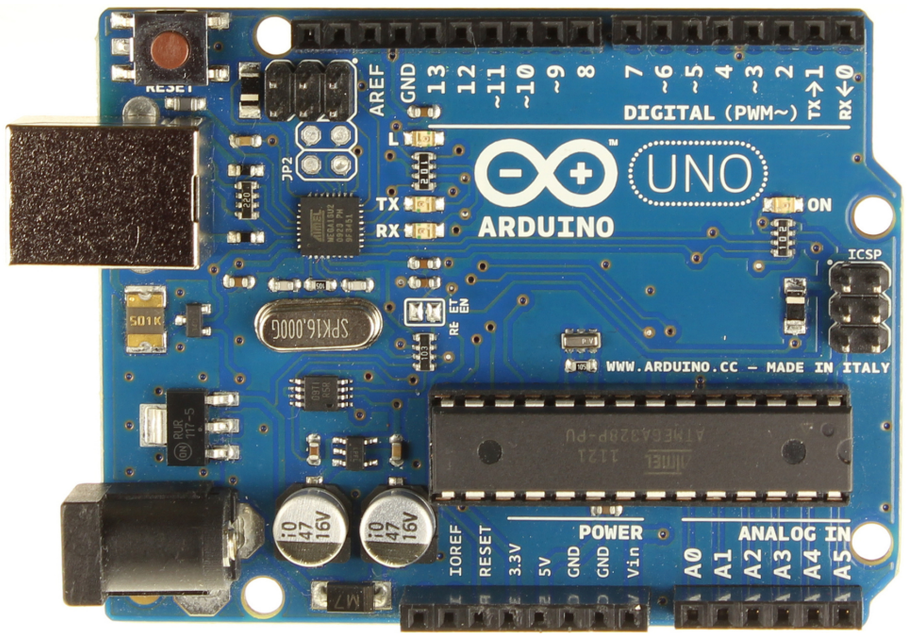
Pin-out

Figure 5-1. 28-pin PDIP



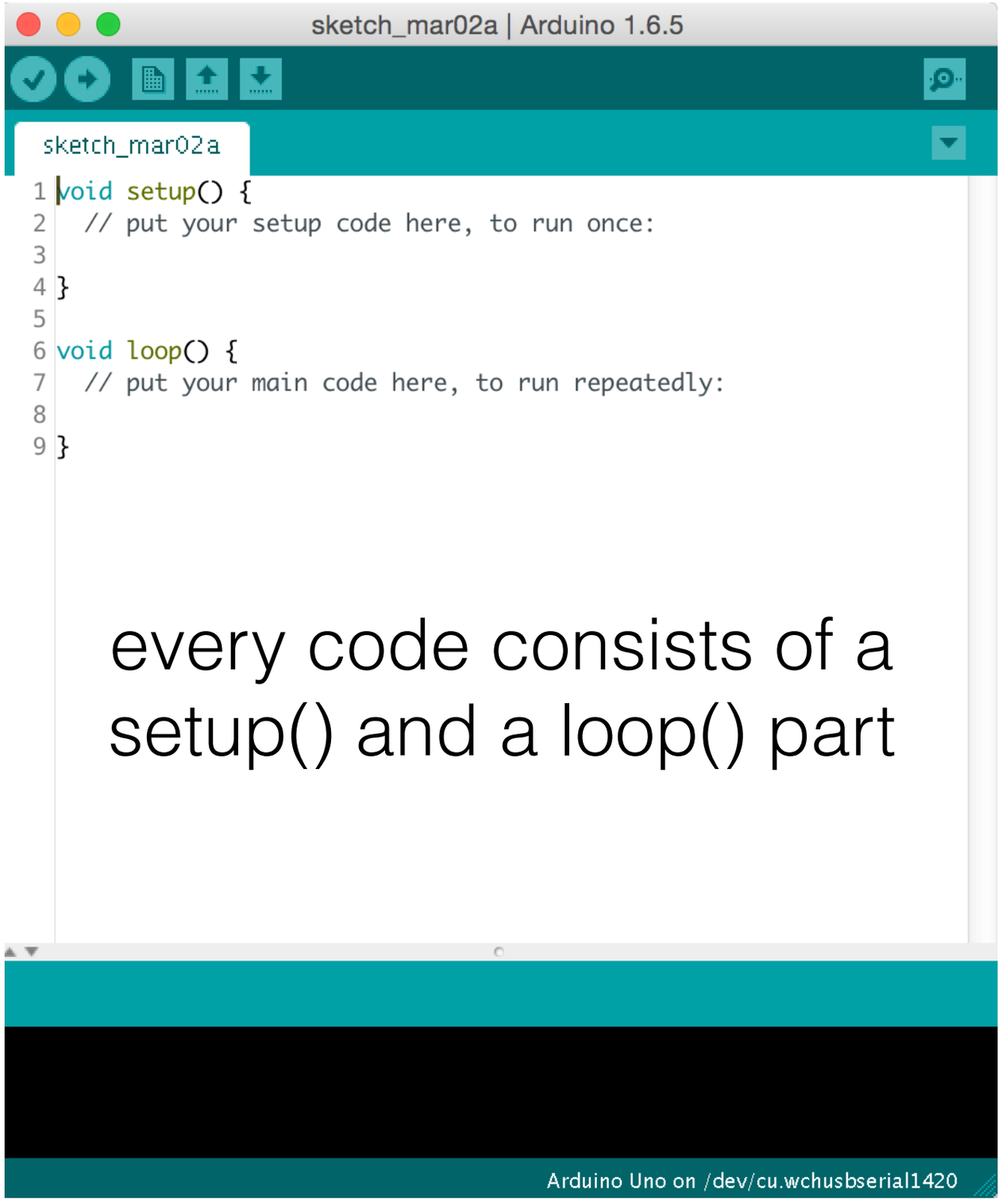
the Arduino philosophy is to provide extensions separately

Arduino IDE



IDE available for:
Windows, Linux, Mac
https://www.arduino.cc

Works with the Arduino bootloader through the COM port

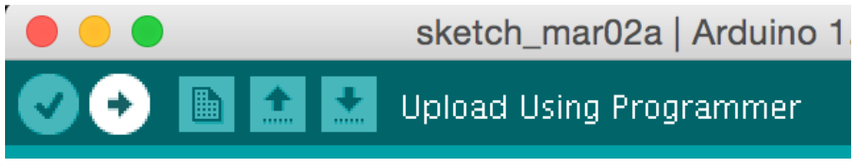


every code consists of a setup() and a loop() part

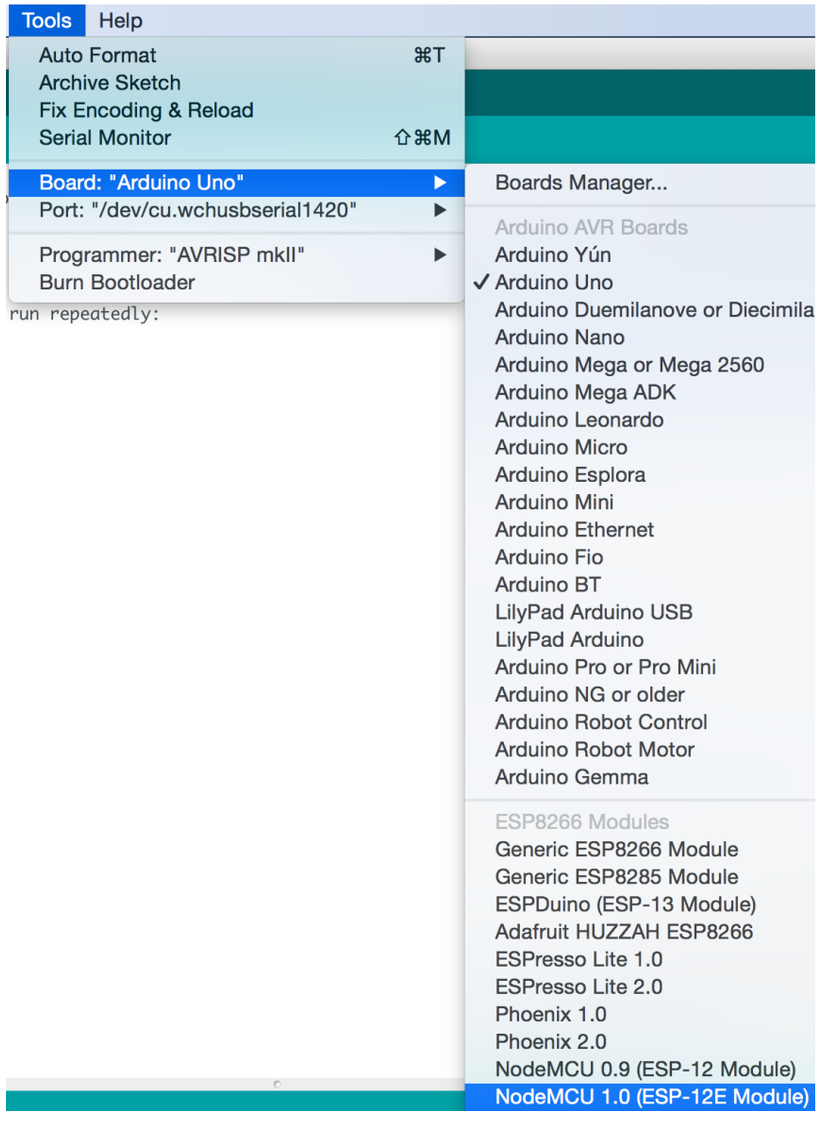
Arduino IDE: we can compile the code and we can upload the code with built-in avrdude

Arduino IDE

Verify: compile the code

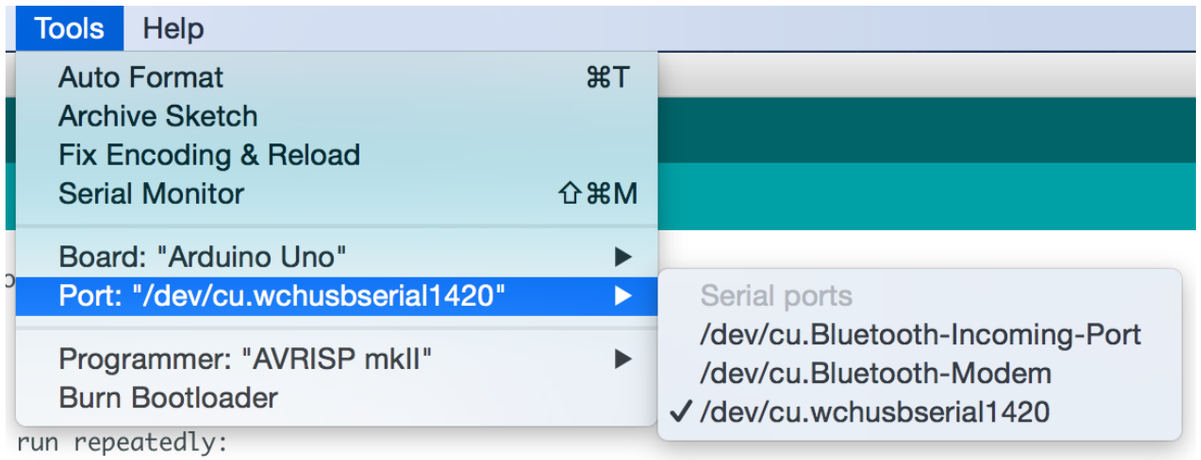


Upload: compile and upload



Tools menu: Board
We can choose the Board that we compile for. It has all the Arduino Boards and it is possible to add new boards that are not present in the list like the Node MCU wireless development board

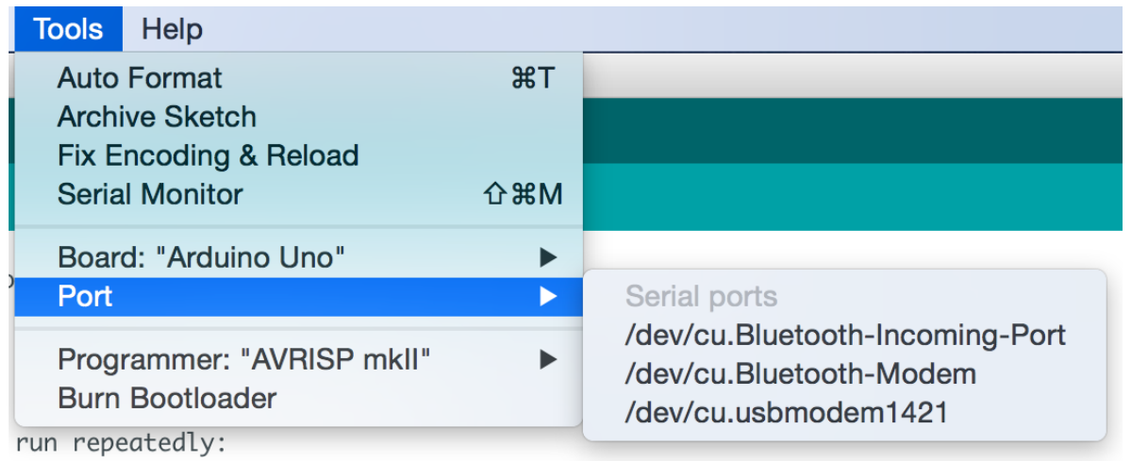
Arduino IDE



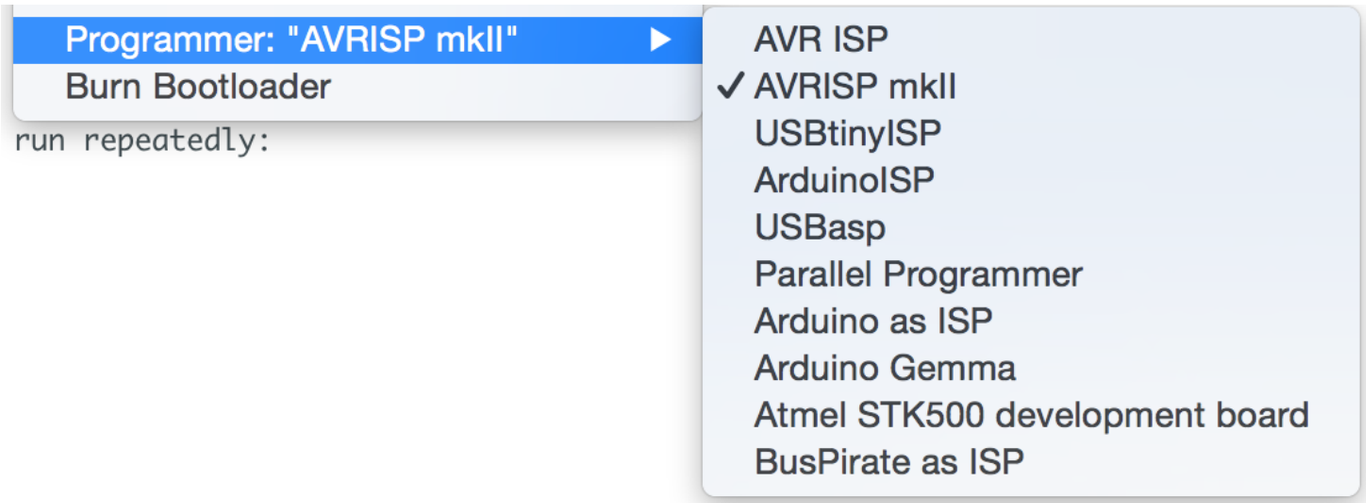
Tools menu: Port

Here we get a list of all boards that are connected and have been recognized. On a mac it shows up as /dev/ wchusbesrial1420 for example, on a Windows as a COM3 and on a Linux as a tty* port

If it does not recognize your board you might need a driver installed (genuine boards get recognized better)



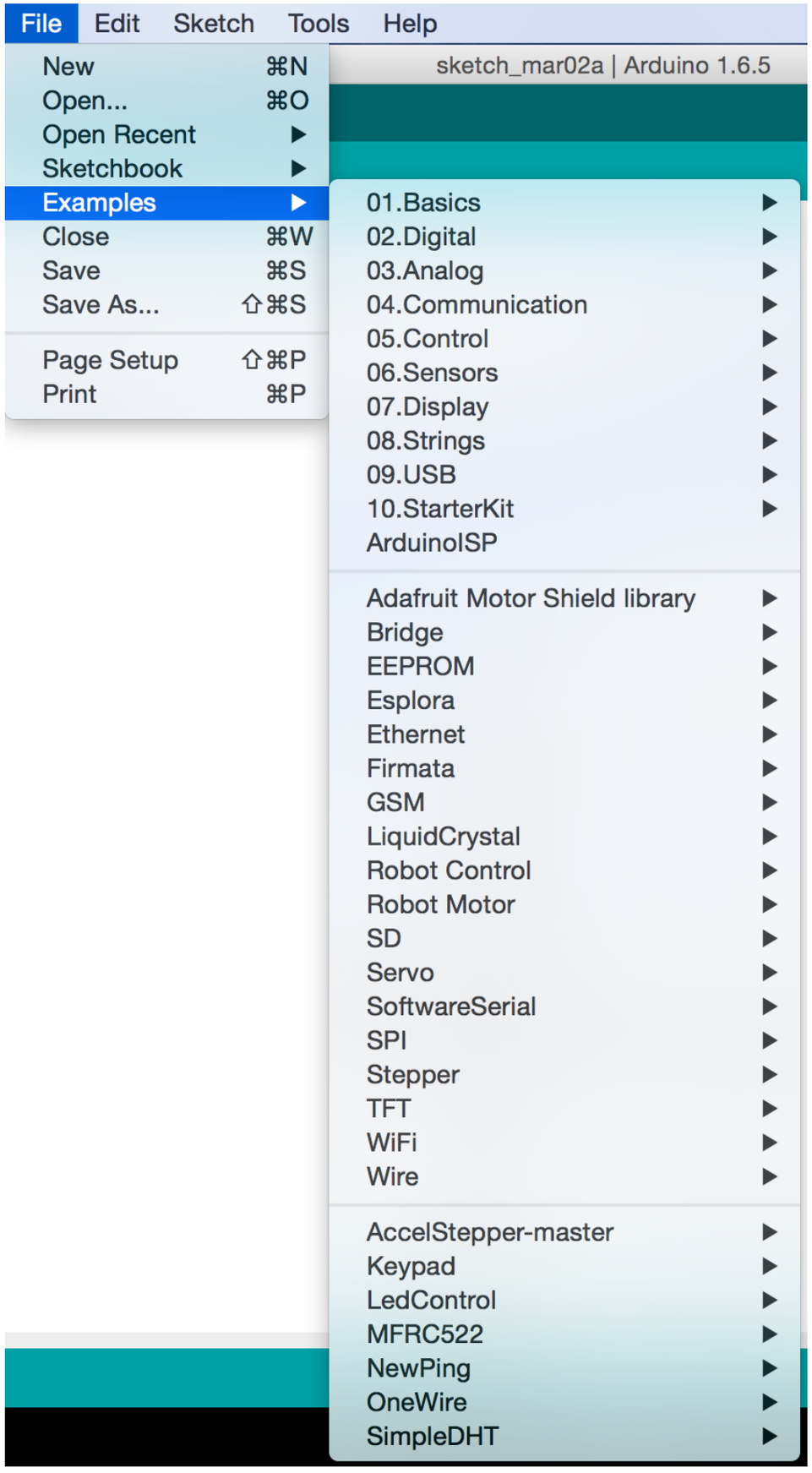
Both the Board and the Port need to be set up correctly to program the board



Arduino IDE

What kind of a programmer we use to upload the code. This is the standard AVR ISP bootloader but the Arduino IDE can also use other types of programmers

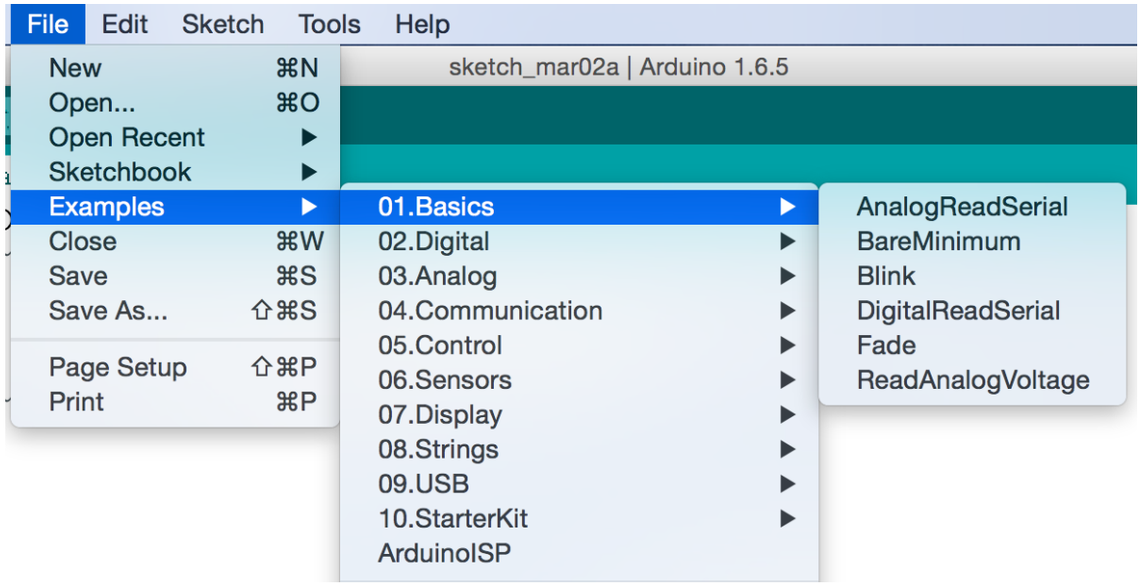
A Burn Bootloader - we will not use it because the Arduino chip usually comes with a bootloader from the factory, however on other microcontrollers you might need to burn a bootloader first

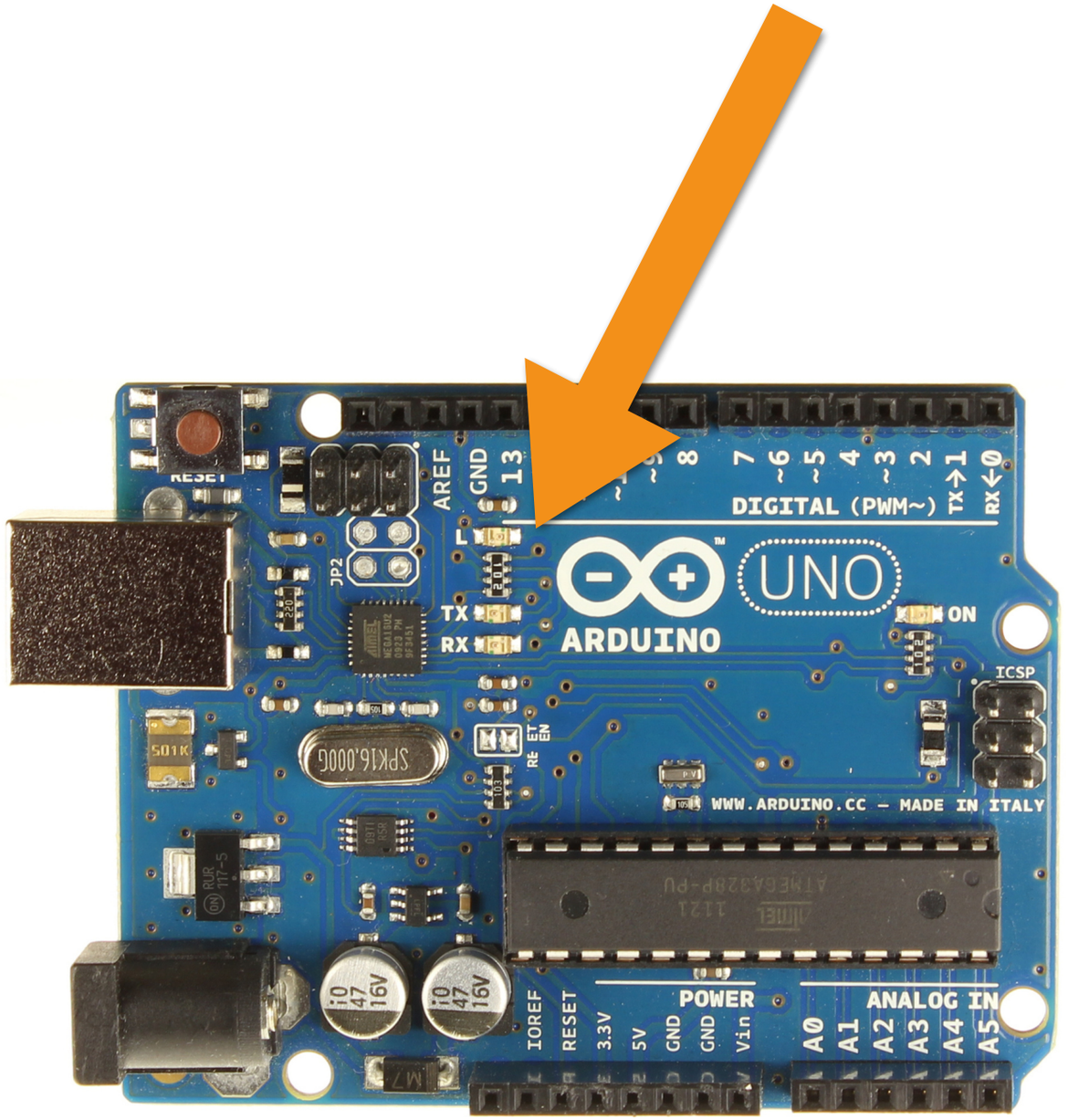


Arduino IDE : Example programs

A great advantage of the Arduino platform and Arduino IDE is that there are many example codes already in the package. The user base of Arduino is also very large, so any kind of extension, shield, sensor etc. that is used with Arduino likely has a code and examples available on the internet already.

Arduino IDE : Blink example





The simplest example is the Blink code, this uses the PIN number13 (PORTB5) that has an attached LED to it to create blinking

Arduino IDE : Blink example

```
17 // the setup function runs once when you press reset or power the board
18 void setup() {
19   // initialize digital pin 13 as an output.
20   pinMode(13, OUTPUT);
21 }
22
23 // the loop function runs over and over again forever
24 void loop() {
25   digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
26   delay(200);               // wait for a second
27   digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
28   delay(1000);              // wait for a second
29 }
```

setup sets the PIN nr.13 to be an OUTPUT
These PINs have a general purpose we can use them both as an output or an input pin
(**GPIO General Purpose I/O**)

Arduino IDE : Blink example

```
17 // the setup function runs once when you press reset or power the board
18 void setup() {
19   // initialize digital pin 13 as an output.
20   pinMode(13, OUTPUT);
21 }
22
23 // the loop function runs over and over again forever
24 void loop() {
25   digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
26   delay(200);               // wait for a second
27   digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
28   delay(1000);              // wait for a second
29 }
```

In the loop we write data to this PIN
HIGH (logical 1 or around 5V)
LOW (logical 0 or GND, 0V)
According to this the LED connected to this pin with a resistor to the GND will either light up or remain dark.

Arduino IDE : Blink example

```
17 // the setup function runs once when you press reset or power the board
18 void setup() {
19   // initialize digital pin 13 as an output.
20   pinMode(13, OUTPUT);
21 }
22
23 // the loop function runs over and over again forever
24 void loop() {
25   digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
26   delay(200);               // wait for a second
27   digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
28   delay(1000);              // wait for a second
29 }
```

The microcontroller stays in the loop permanently and repeats it so it would repeat the loop and do 16 million instructions (or so) each second. So we need to put in a delay in ms to be able to see the blinking of the LED.

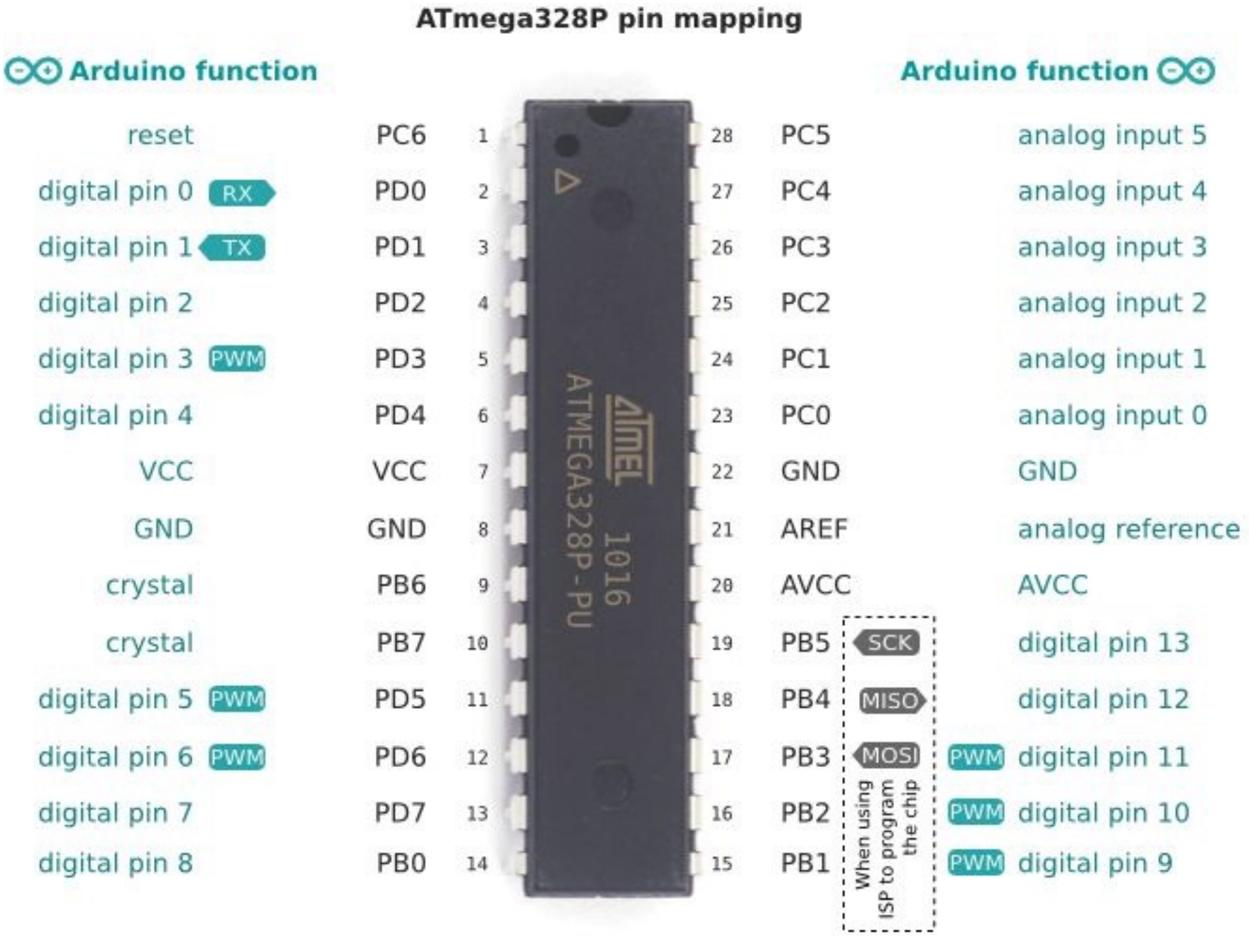
GPIO pins

The delay() function is a higher level function that actually uses the Timer section of the microcontroller. We will look under the hood and learn about the details of the timers on the next course. Let's first examine the low level coding of the GPIO pins.

GPIO: General Purpose Input/Output
Az Atmel Atmega 328P has 3 such PORTs each with 8 pins
PORTB, PORTC and PORTD

PORTB contains the
PORTB0, PORTB1, PORTB2 … PORTB7 pins
8 bit - 1 byte, this corresponds to one register

GPIO pins


ATmega328P pin mapping
Arduino function
Arduino function
reset PC6 1 28 PC5 analog input 5
digital pin 0 RX PD0 2 27 PC4 analog input 4
digital pin 1 TX PD1 3 26 PC3 analog input 3
digital pin 2 PD2 4 25 PC2 analog input 2
digital pin 3 PWM PD3 5 24 PC1 analog input 1
digital pin 4 PD4 6 23 PC0 analog input 0
VCC VCC 7 22 GND GND
GND GND 8 21 AREF analog reference
crystal PB6 9 20 AVCC AVCC
crystal PB7 10 19 PB5 SCK digital pin 13
digital pin 5 PWM PD5 11 18 PB4 MISO digital pin 12
digital pin 6 PWM PD6 12 17 PB3 MOSI PWM digital pin 11
digital pin 7 PD7 13 16 PB2 PWM digital pin 10
digital pin 8 PB0 14 15 PB1 PWM digital pin 9
When using ISP to program the chip
ATMEGA328P-PU
1016

GPIO pins

Each GPIO PORT has 3 dedicated registers
For example the B PORT is assigned the

DDRB register (that sets the pins to input or output)
PORTB (this is the data register for the port, this is where we write the information when it is an output)

PINB (this is a read only register this is where we read the value from when it is an input)

GPIO pins

```
1
2  #include <avr/io.h>
3  #include <util/delay.h>
4
5  #define BLINK_DELAY_MS 1000
6
7  int main (void)
8  {
9   /* set pin 5 of PORTB for output*/
10  DDRB |= _BV(DDB5);
11
12  while(1) {
13   /* set pin 5 high to turn led on */
14   PORTB |= _BV(PORTB5);
15   _delay_ms(BLINK_DELAY_MS);
16
17   /* set pin 5 low to turn led off */
18   PORTB &= ~_BV(PORTB5);
19   _delay_ms(BLINK_DELAY_MS);
20  }
21 }
```

Same BLINK code but on a low level

instead of writing pinMode(13, OUTPUT), I set the a DDRB register at the correct location of PORTB (5) to 1 (output)

PORTB = PORTB | 0b00100000;

GPIO pins

```
1
2 #include <avr/io.h>
3 #include <util/delay.h>
4
5 #define BLINK_DELAY_MS 1000
6
7 int main (void)
8 {
9  /* set pin 5 of PORTB for output*/
10  DDRB |= _BV(DDB5);
11
12  while(1) {
13   /* set pin 5 high to turn led on */
14   PORTB |= _BV(PORTB5);
15   _delay_ms(BLINK_DELAY_MS);
16
17   /* set pin 5 low to turn led off */
18   PORTB &= ~_BV(PORTB5);
19   _delay_ms(BLINK_DELAY_MS);
20  }
21 }
```

DDB5 simply contains the value 5
_BV(5) means bitwise shift to left
1<<5, so it means 0b00100000

Macro _BV

```
#define _BV(bit) \
        (1 << (bit))
```

```
#include <avr/io.h>
```

Bitwise OR - why we have this?
we want to change only bit nr. 5
the rest will stay the same
(OR 0 does not change the bit)

GPIO pins

```
1
2  #include <avr/io.h>
3  #include <util/delay.h>
4
5  #define BLINK_DELAY_MS 1000
6
7  int main (void)
8  {
9   /* set pin 5 of PORTB for output*/
10  DDRB |= _BV(DDB5);
11
12  while(1) {
13   /* set pin 5 high to turn led on */
14   PORTB |= _BV(PORTB5);
15   _delay_ms(BLINK_DELAY_MS);
16
17   /* set pin 5 low to turn led off */
18   PORTB &= ~_BV(PORTB5);
19   _delay_ms(BLINK_DELAY_MS);
20  }
21 }
```

PORTB5 just means 5

PORTB |= 0b00100000;

sets the 5th bit to 1 and leaves the rest of them alone

PORTB &= ~0b00100000;

~ is bitwise negation

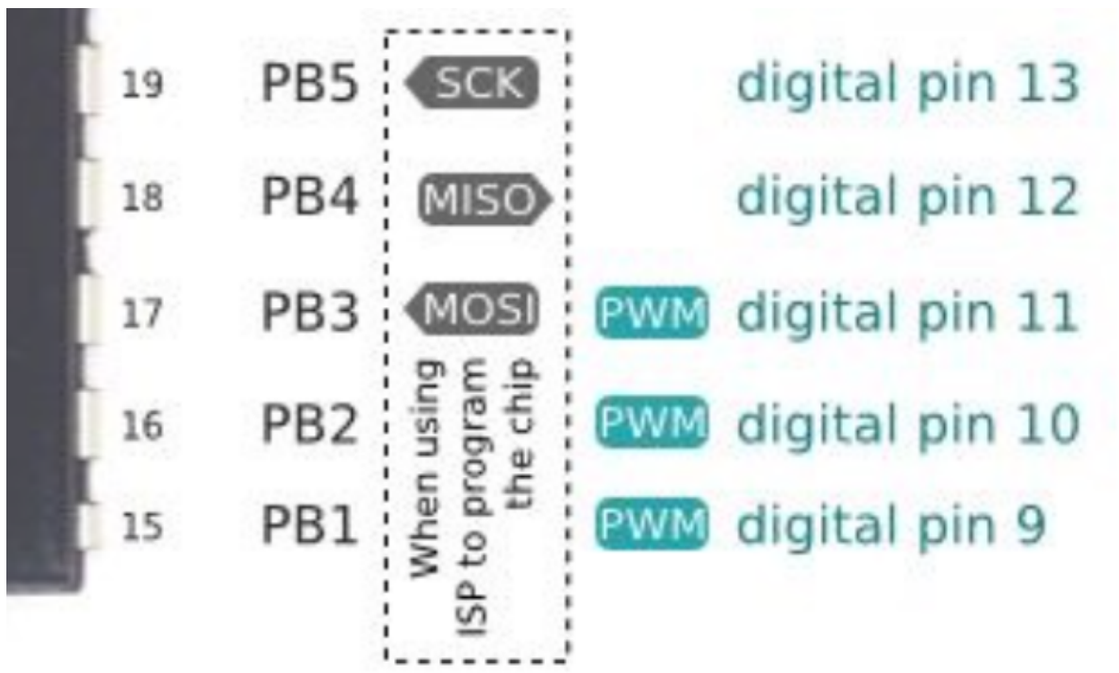
this means the following:

PORTB &= 0b11011111;

this means all bits remain the same except the 5th that gets set to 0

GPIO pins

What is the difference between:
PORTB |= 0b0010100;
from:
digitalWrite(13,HIGH);
digitalWrite(11, HIGH); ?



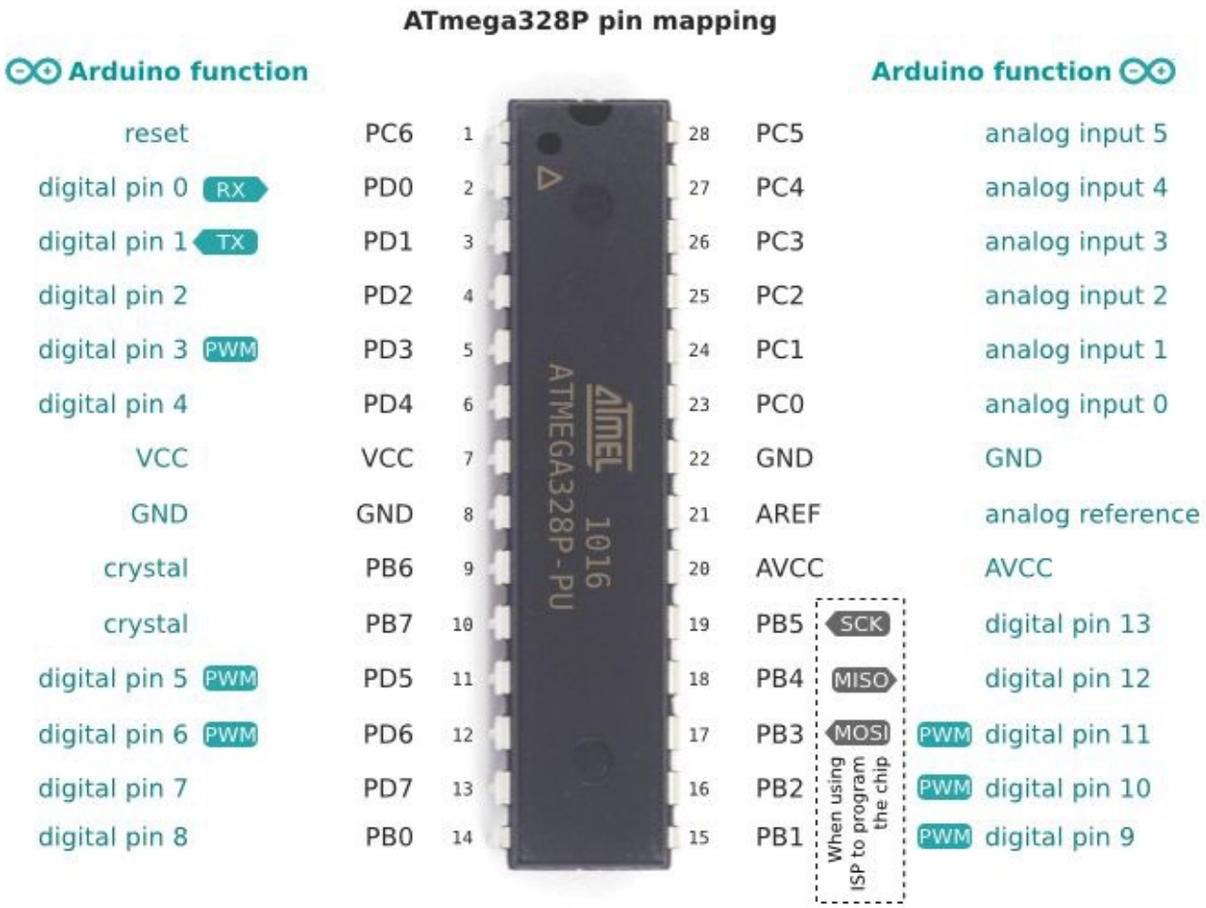
Mapping is ok,
13==PORTB5 and
11==PORTB3, that is
not where
the difference is

GPIO pins

What is the difference between:
PORTB |= 0b0010100;
from:
digitalWrite(13,HIGH);
digitalWrite(11, HIGH); ?

The difference is that the first line is executed in 1 clock cycle while the 2 lines below might execute in more depending on how the library was implemented. The code on the microcontroller is also going to be shorter we have a more direct control over it.

GPIO pins

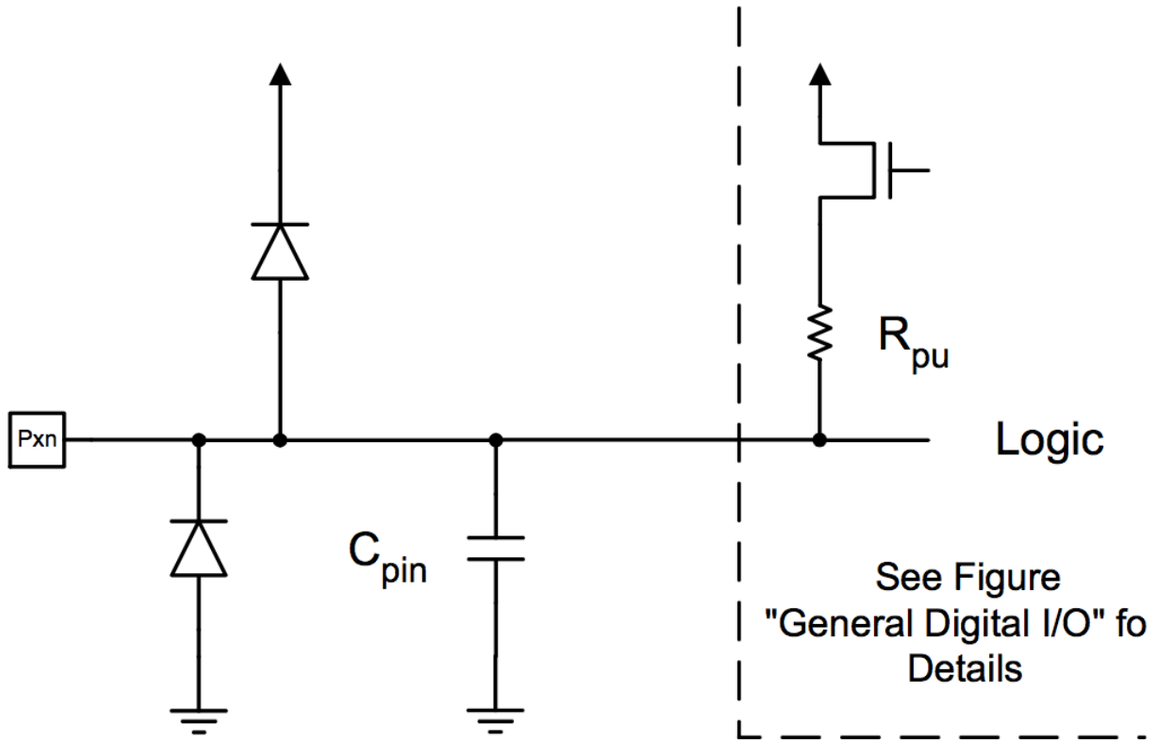


Low level: we try using the mapping that is why we use pre-defined constants as DDRB5, PORTB5 that also make the code more readable

What do we need to pay attention to on the low level coding? You need to know what you are doing if you set DDRD Pin nr. 0 to output (1), then the serial port will not work any more because this is the RX bit in serial communication. We usually avoid using bits 0,1 from PORTD for this reason (they are RX,TX in serial communication)

GPIO pins

Figure 18-1. I/O Pin Equivalent Schematic



▼ 18. I/O-Ports

- 18.1. Overview
- ▶ 18.2. Ports as General Digital I/O
- ▶ 18.3. Alternate Port Functions
- ▶ 18.4. Register Description

Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

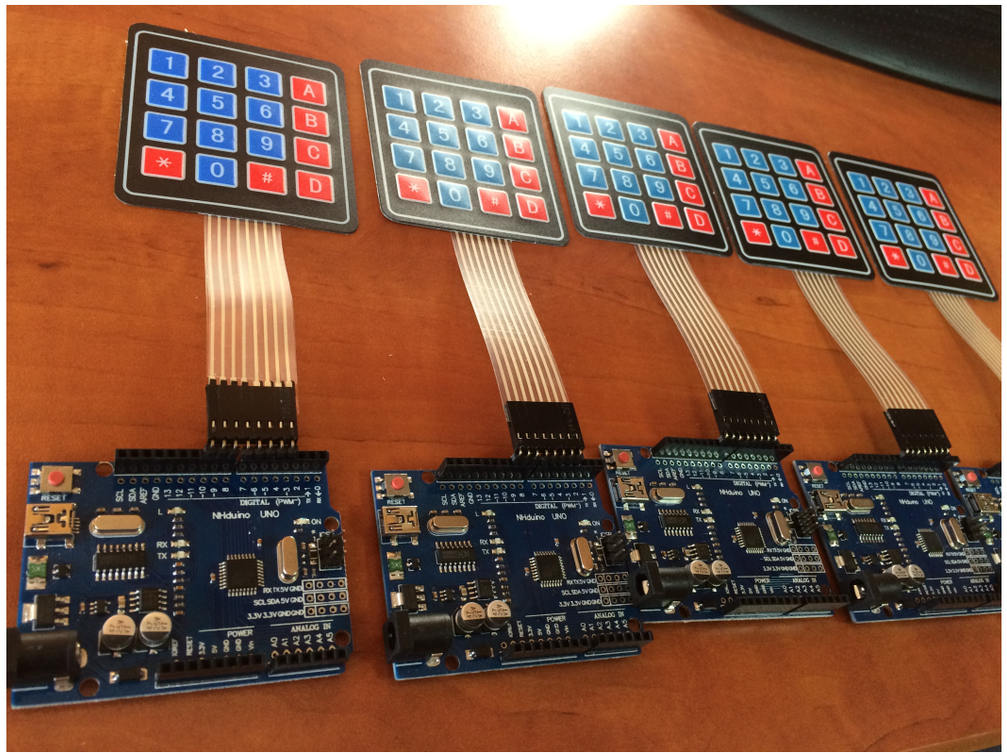
A detaild description of the GPIO ports is in the Atmel Atmega 328P data sheet pdf in chapter nr. 18. between pages 97-101 and we can find the secondary mapping of each PIN on pages 102-124 with the description of the PORT, PIN and DDR registers

Lab #1: GPIO

We modify the Blink program on the first lab to check for connection, and if the IDE is working, and also to see the simple structure of the code

Then we modify this to send Morse code with the LEDs

Finally we will use Keypads to illustrate digital inputs we will set a password using the keypads



The keypad is in fact a 4x4 switch array that needs to be continuously scanned for changes. How would you implement this?

Lab #1: GPIO

The inputs for the keypad are on the D3..D10lines