

به نام خدا

برنامه‌ریزی تراشه AVR

معرفی ابزارهای کاربردی کار با تراشه‌های AVR (Toolchain)

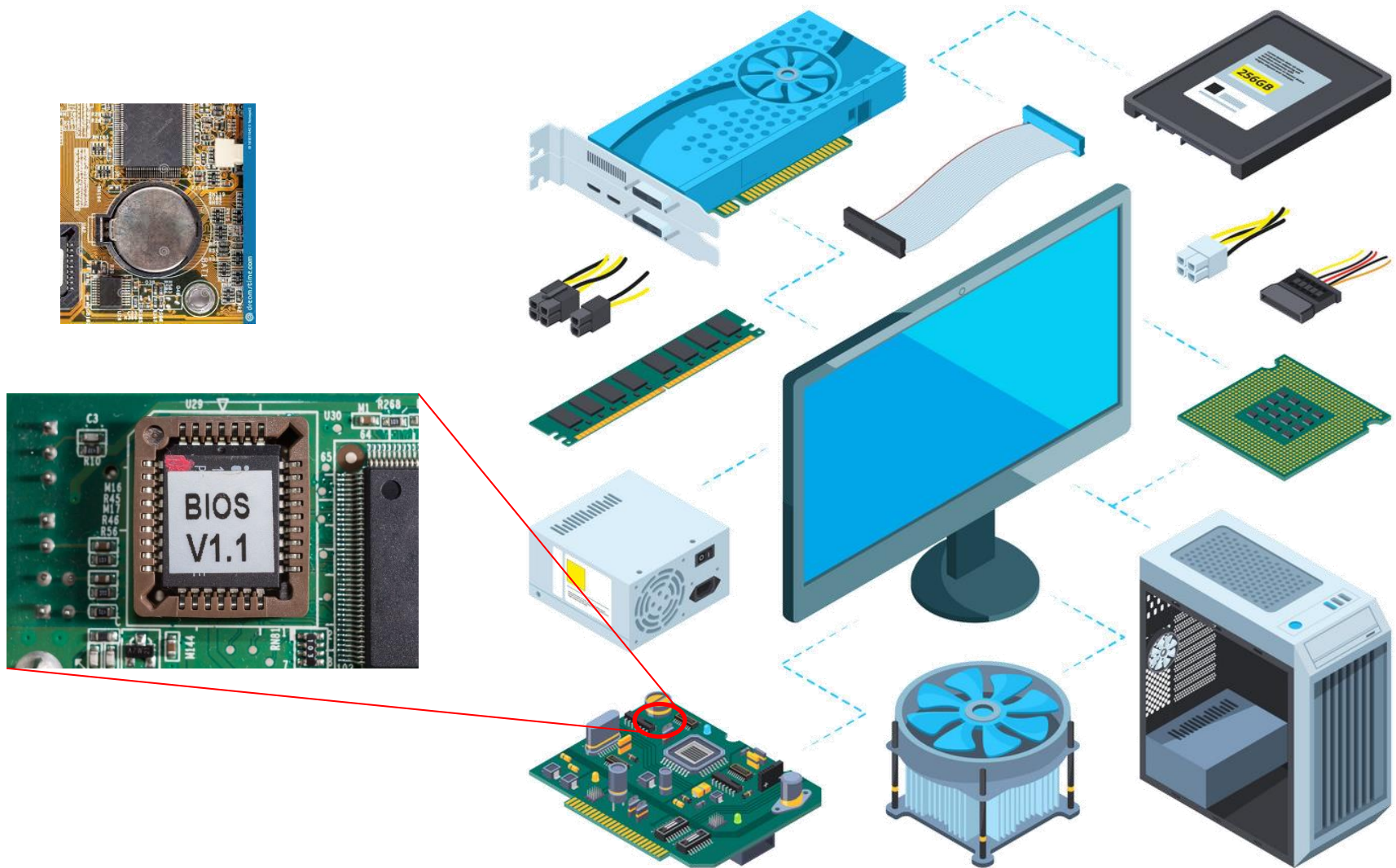
Dr. Aref Karimafshar
A.karimafshar@ec.iut.ac.ir



کامپیوترهای آماده به کار

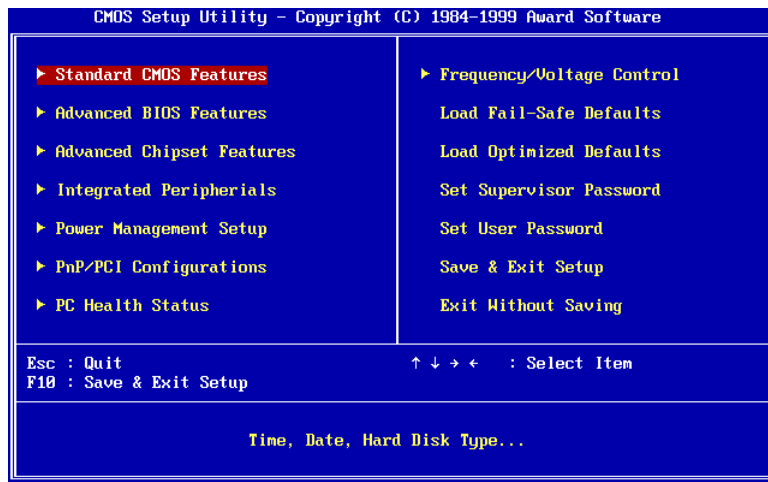


سرهم کردن قطعات



راه انداز سیستم

- BIOS (basic input/output)
 - Motherboard firmware
 - Software which runs at a lower level than operating system
 - Tells the computer
 - What drive to boot from
 - How much RAM you have
 - Controls other key details like CPU frequency

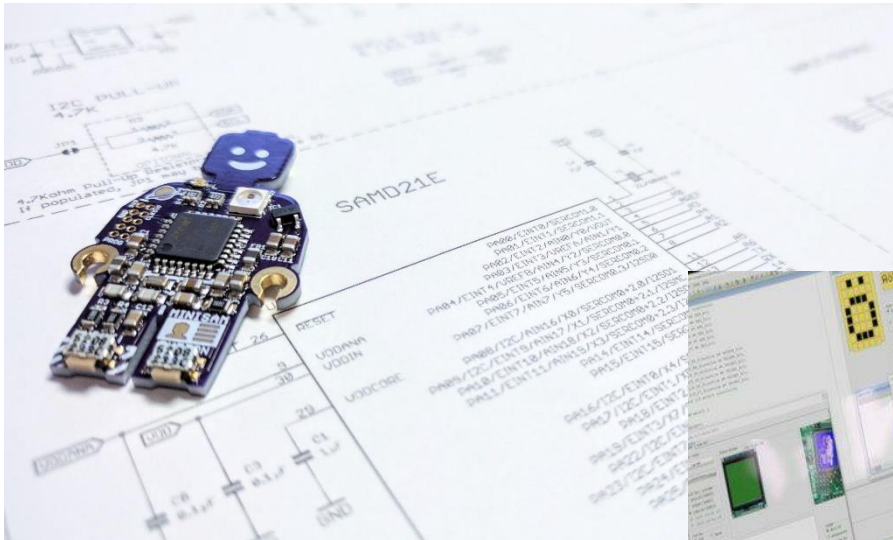


میکروکنترلر

- میکروکنترلر

- خالی از هر گونه برنامه

- نیازمند برنامه ریزی

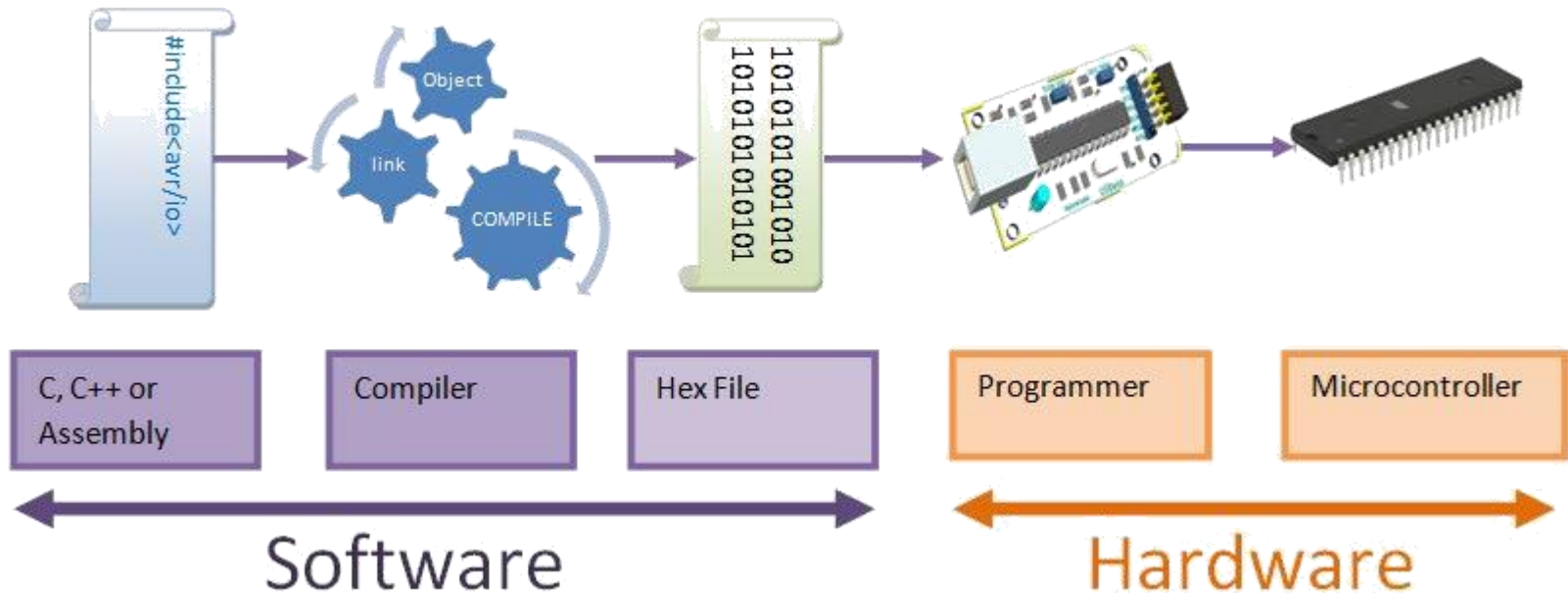


میکروکنترلر

- برنامه ریزی

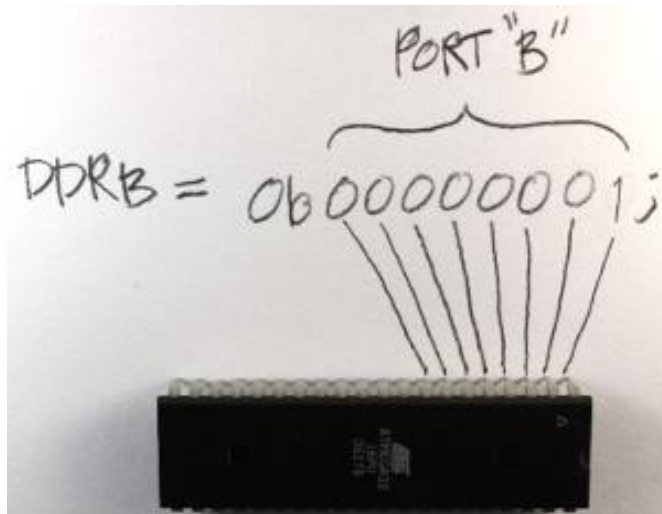
- ابزارهای فیزیکی

- نرم افزارهای مدیریت و برنامه ریزی



Program & Data

- Program
 - a sequence of instructions specifying how the data is to be processed.
- Data
 - input to the program, either supplied during runtime or pre-stored in the computer



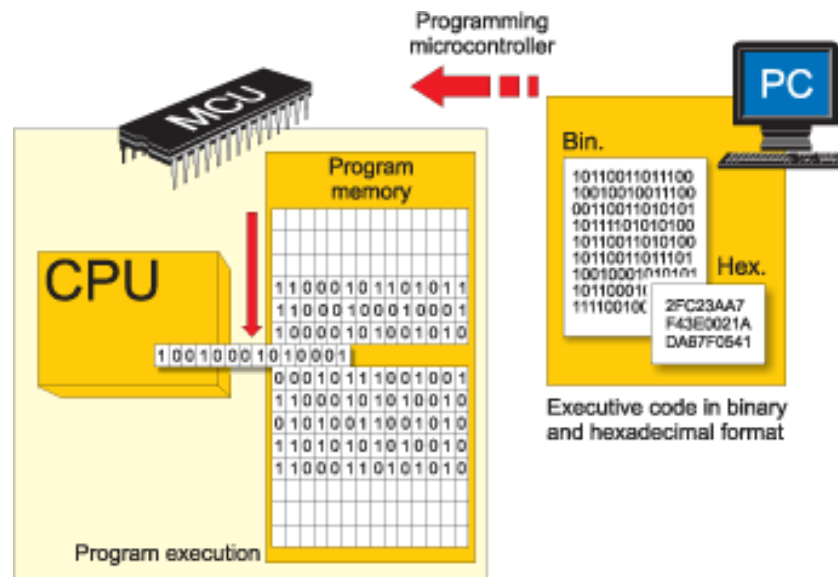
DATA_BUFFER	EQU	0x20	
DATA_BUFFER2	EQU	0x21	
CMD_BUFFER	EQU	0x22	
REF_BUFFER	EQU	0x24	
ADDR_INDEX	EQU	0x25	;STARTING ADDRESS IN EEPROM
ADDR_L	EQU	0x26	;STARTING ADDRESS L
ADDR_H	EQU	0x27	;STARTING ADDRESS H
COUNTER_INDEX	EQU	0x29	;COUNTER
BIT_INDEX	EQU	0x2A	;BIT INDEX
CMD_SET_CURSOR	EQU	21H	;SET CURSOR
CMD_TXHOME	EQU	40H	;SET TXT HM ADD
CMD_TXAREA	EQU	41H	;SET TXT AREA
CMD_GRHOME	EQU	42H	;SET GR HM ADD
CMD_GRAREA	EQU	43H	;SET GR AREA
CMD_OFFSET	EQU	22H	;SET OFFSET ADD
CMD_ADPSET	EQU	24H	;SET ADD PTR
CMD_SETDATA_INC	EQU	0COH	;WRITE DATA AND INCREASE ADP
CMD_AWON	EQU	0BOH	;SET AUTO WRITE MODE
CMD_AWROFF	EQU	0B2H	;RESET AUTO WRITE MODE

Program

- The microcontroller executes user program loaded in its memory.
- In the case of Harvard architecture
 - program memory is usually of ROM type (PROM, OTP ROM, EPROM, Flash...)
- Program in ROM memory is called executable code, or firmware
- Executable code is sequence of zeros and ones – binary code
- It is organized in 8, 12, 14, 16, 24, 32-bit wide words
 - depending on the microcontroller's architecture.
- Every word is considered by the CPU as a command being executed during the operation of the microcontroller.
- Set of commands understood by CPU is called machine language
- Machine language is CPUs native language

Program

- For practical reasons, as it is much easier for us to deal with hexadecimal number system, the executable code is often represented as a sequence of hexadecimal numbers called a Hex code.
- Executable code can be written directly using text editor, and stored into the Flash memory by appropriate software tool – flash programmer



Programming Languages (PL)

```
27bdfdd0 afbf0014 0c1002a8 00000000 0c1002a8 afa2001c 8fa4001c
00401825 10820008 0064082a 10200003 00000000 10000002 00832023
00641823 1483ffffa 0064082a 0c1002b2 00000000 8fbf0014 27bd0020
03e00008 00001025
```

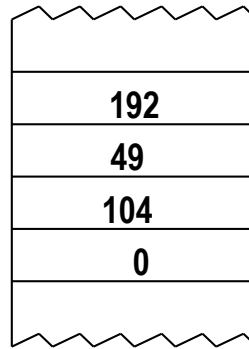
Can You Understand This?

```
addiu    sp,sp,-32
sw       ra,20(sp)
jal      getint
nop
jal      getint
sw       v0,28(sp)
lw       a0,28(sp)
move     v1,v0
beq      a0,v0,D
slt      at,v1,a0
A: beq    at,zero,B
nop
b        C
subu     a0,a0,v1
B: subu   v1,v1,a0
C: bne    a0,v1,A
slt      at,v1,a0
D: jal    putint
nop
lw       ra,20(sp)
addiu    sp,sp,32
jr       ra
move     v0,zero
```

Is This Better?

PL: Machine Languages

- A CPU only understands its own machine language => *portability* problem
 - In Motorola 68000, the following 4 bytes is an instruction to move the value in register D3 to memory address 104.



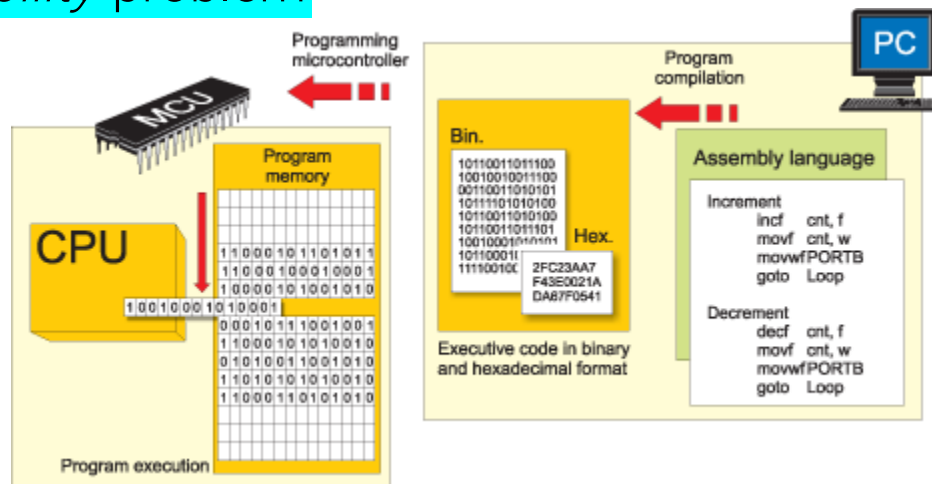
- In Intel 80486, the same sequence of bytes represents different instructions
- Coding is tedious and error prone

PL: Assembly Languages

- As we know process of writing executable code using machine language is very difficult
- Binary/Hex words from machine language vocabulary can be represented in the form of meaningful abbreviations
- These abbreviations are called assembly instructions
- Set of assembly instructions is called assembly language.
- Programs written using assembly instructions, in order to be executed by microcontroller, should be translated (compiled) into executable code
- Compiling can be performed using special PC tool called Assembler

PL: Assembly Languages

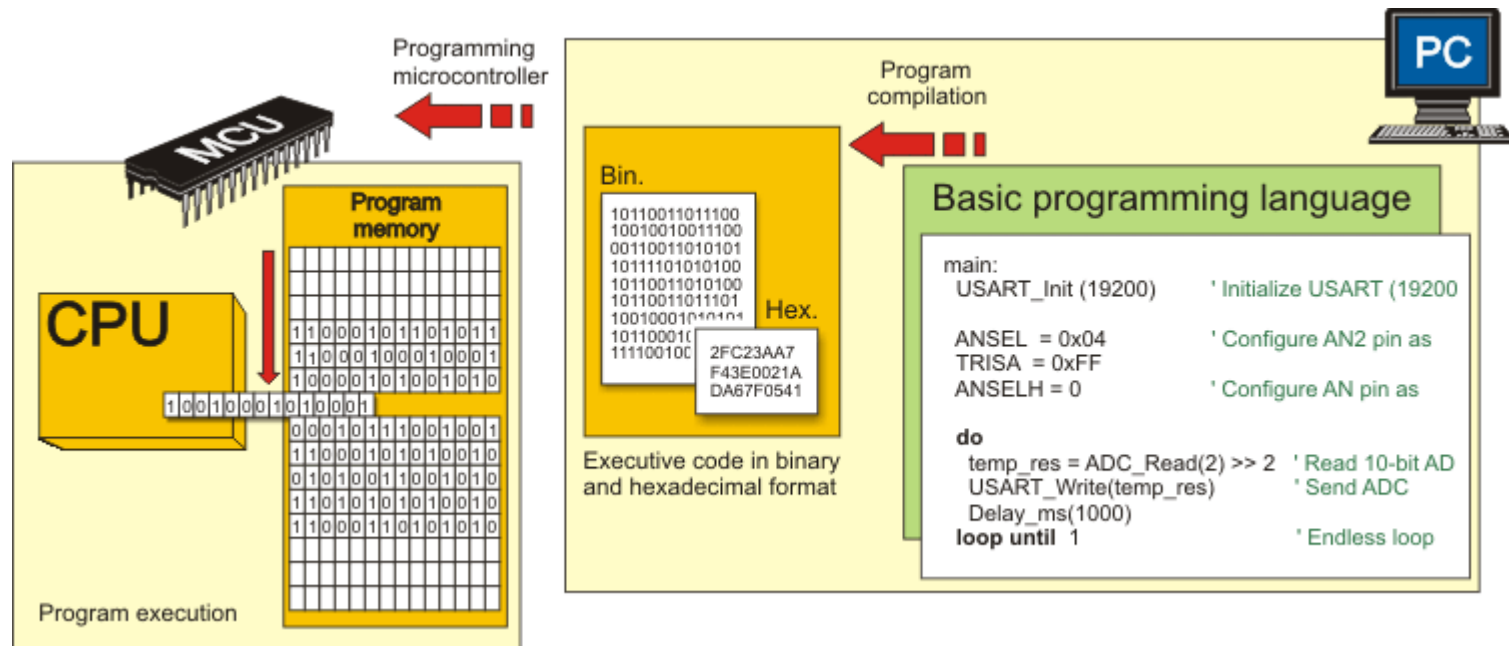
- Use English-like abbreviations
 - e.g. the MC68000 machine instruction is written as:
`MOVE D3, 104`
 - Slightly easier for human to understand
 - Different CPUs have different instruction sets
 - e.g. `MOVE D3, 104` is not a valid instruction in Intel 80486 because it doesn't even have a register called D3
- => *portability* problem



PL: High-level Languages

- Close(r) to human language
- One single statement accomplishes substantial tasks
- Need a compiler/linker
 - to translate into machine language
- More portable
 - the same program (more or less) works for different machines
- e.g.
 - Fortran, COBOL, Basic, Pascal, Ada, Modula, C, C++, Lisp, Prolog, Java, Perl...

PL: High-level Languages



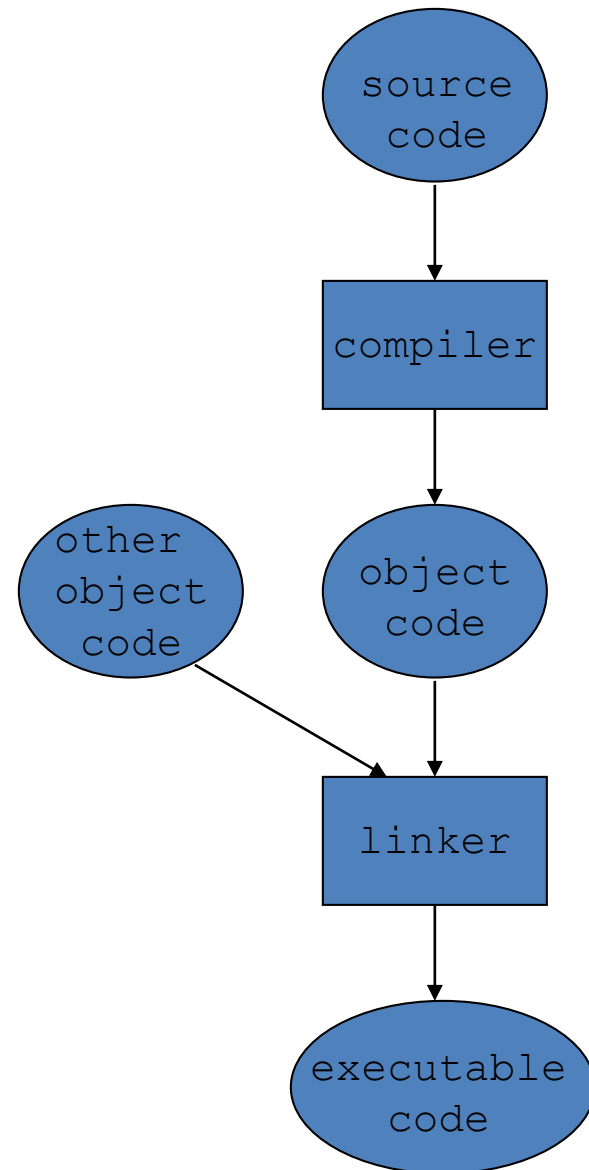
PL: Compiler & Linker

- **Compiler**

- Translates a program in high-level language into an *object program* (or *object code*). The original program is called the *source program/code*.

- **Linker**

- Combines the object code of a program with other pre-compiled object codes to generate an *executable code* which is in machine language



From source code to Hex file and programming



```
program test
main:
    TRISA = 0x00      ' Configure PORTA pins as outputs
    PORTA = 0xFF      ' Turn PORTA LEDs ON
end.
```

Program written in Basic

Compiling program into assembly code

ADDRESS	OPCODE	ASM

		GOTO 3
0x0000	0x2803	
main:		
		TRISA = 0x00 ' Configure PORTA pins as outputs
0x0003	0x1683	BSF STATUS, 5
0x0004	0x1303	BCF STATUS, 6
0x0005	0x0185	CLRF TRISA
		PORTA = 0xFF ' Turn PORTA LEDs ON
0x0006	0x30FF	MOVLW 255
0x0007	0x1283	BCF STATUS, 5
0x0008	0x0085	MOVWF PORTA
0x0009	0x2809	GOTO \$+0

Compiled program

Compiling program into a hex code

```
:020000000328D3
:0E000600831603138501FF308312850009283D
:04400E00F22F000786
:00000001FF
```

Executable code of the program (hex code)

Loading program into the microcontroller



Toolchain

- A collection of tools/libraries
 - To create applications for AVR microcontrollers
 - Compiler
 - Assembler
 - Linker
 - Standard C library (AVR-libc)
- For AVR
 - There are free software tools available
 - Even some of it is open-source!

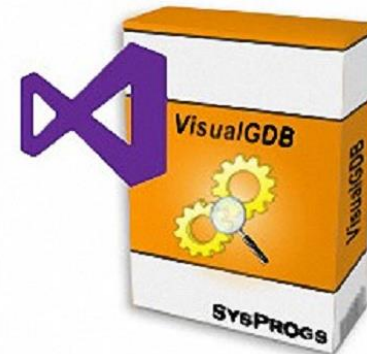


IDE

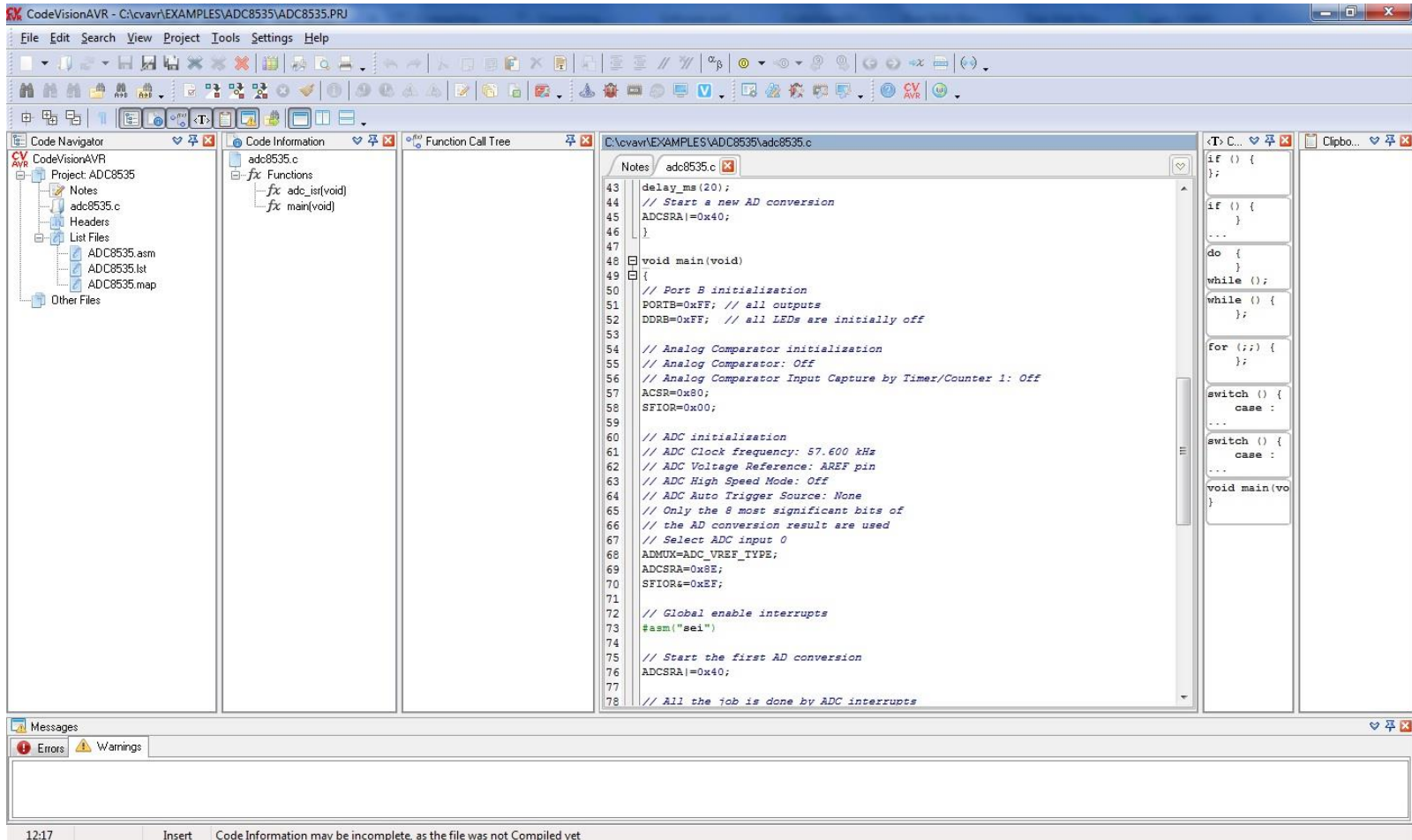
- Integrated development environment (IDE)
 - CodeVisionAVR
 - BASCOM-AVR
 - WinAVR
 - Visual Studio
 - Eclipse (AVR Eclipse Plugin)
 - Atmel Studio (Microchip Studio)



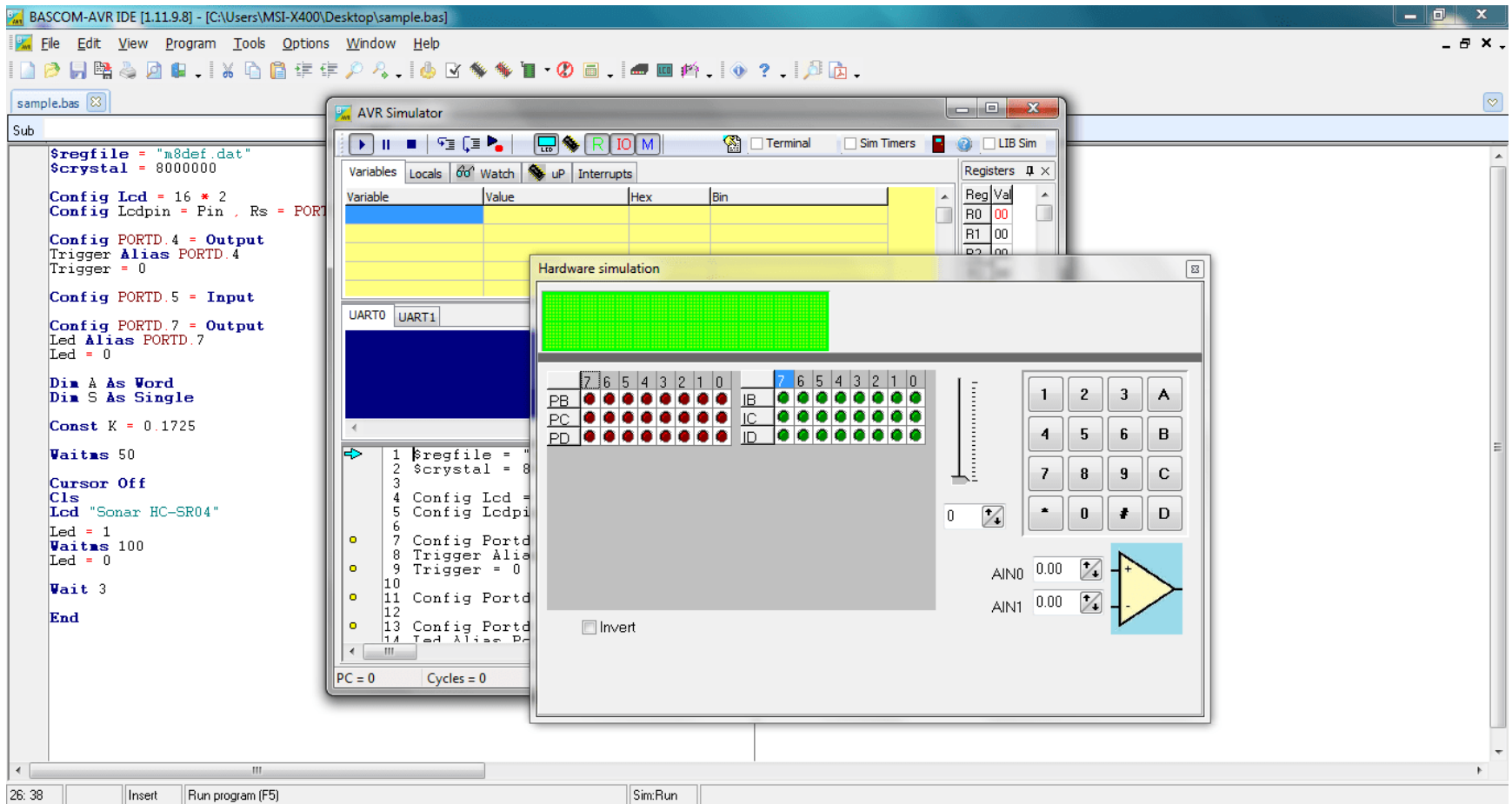
WINAVR



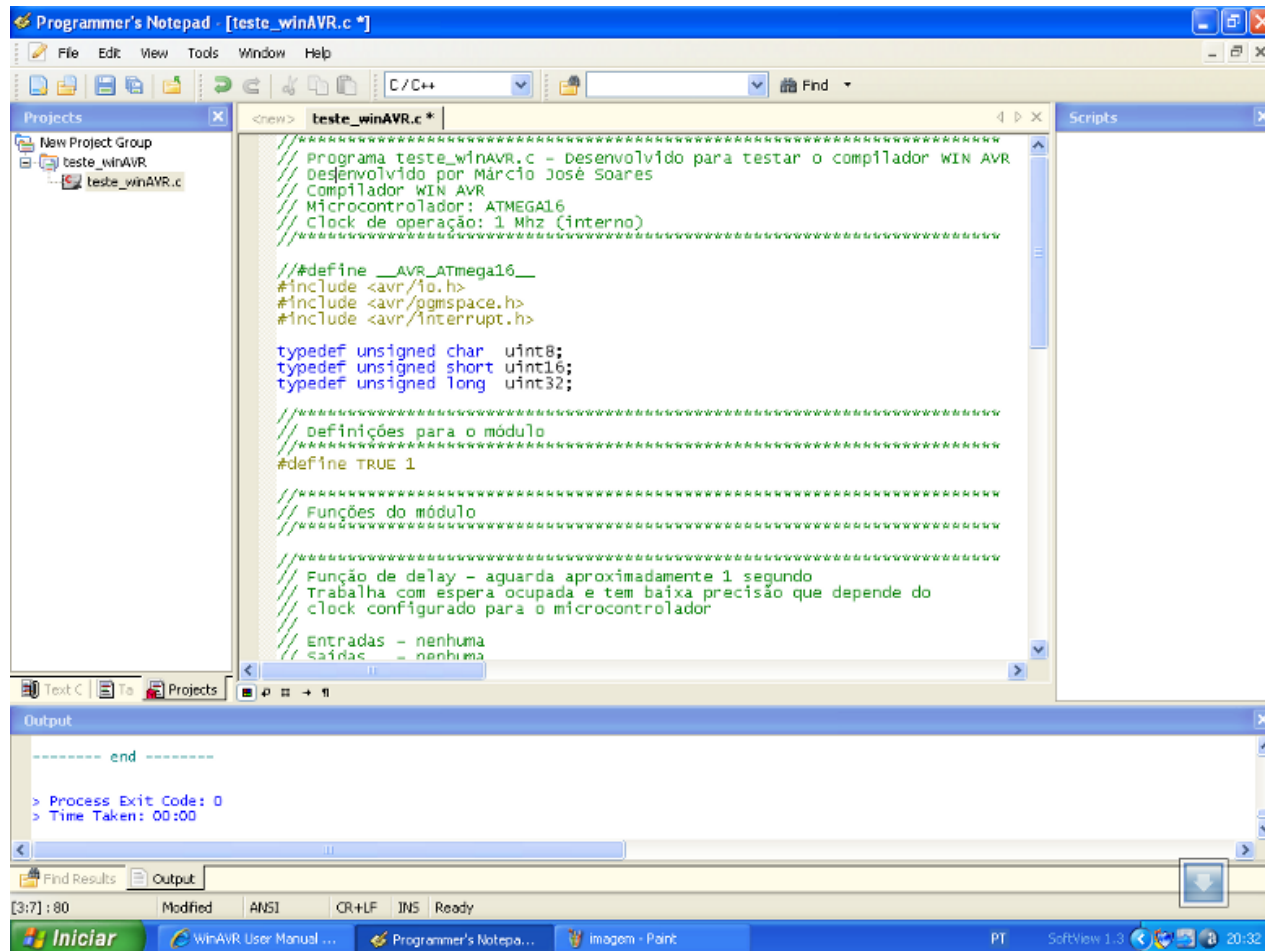
CodeVisionAVR



BASCOM-AVR



WinAVR

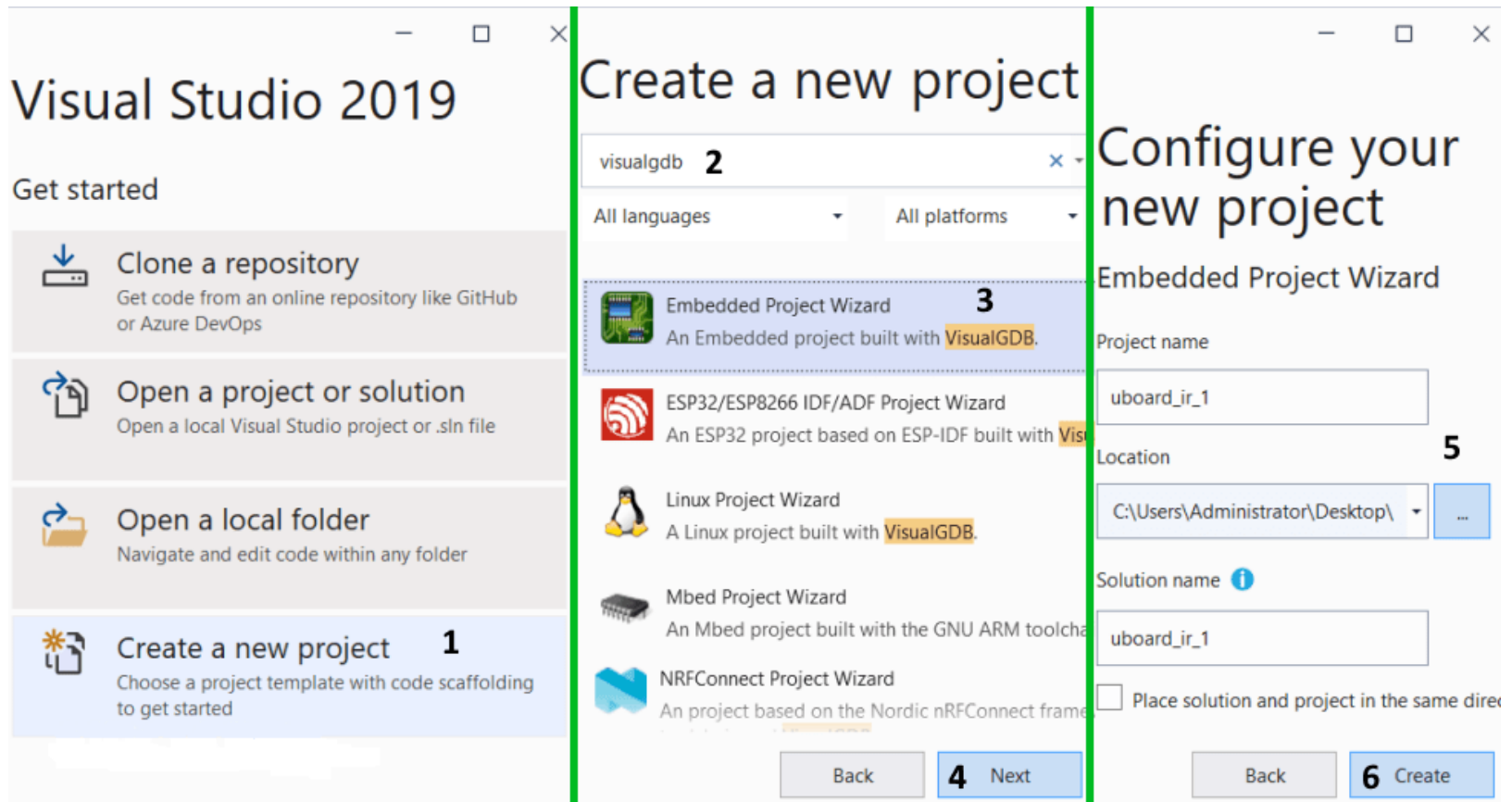


BASCOM vs. CodeVision



Visual Studio

- افزونه‌ی VisualGDB



شركة Atmel

- Atmel (advanced technology for memory and logic)



Logo from 1984 to 2012

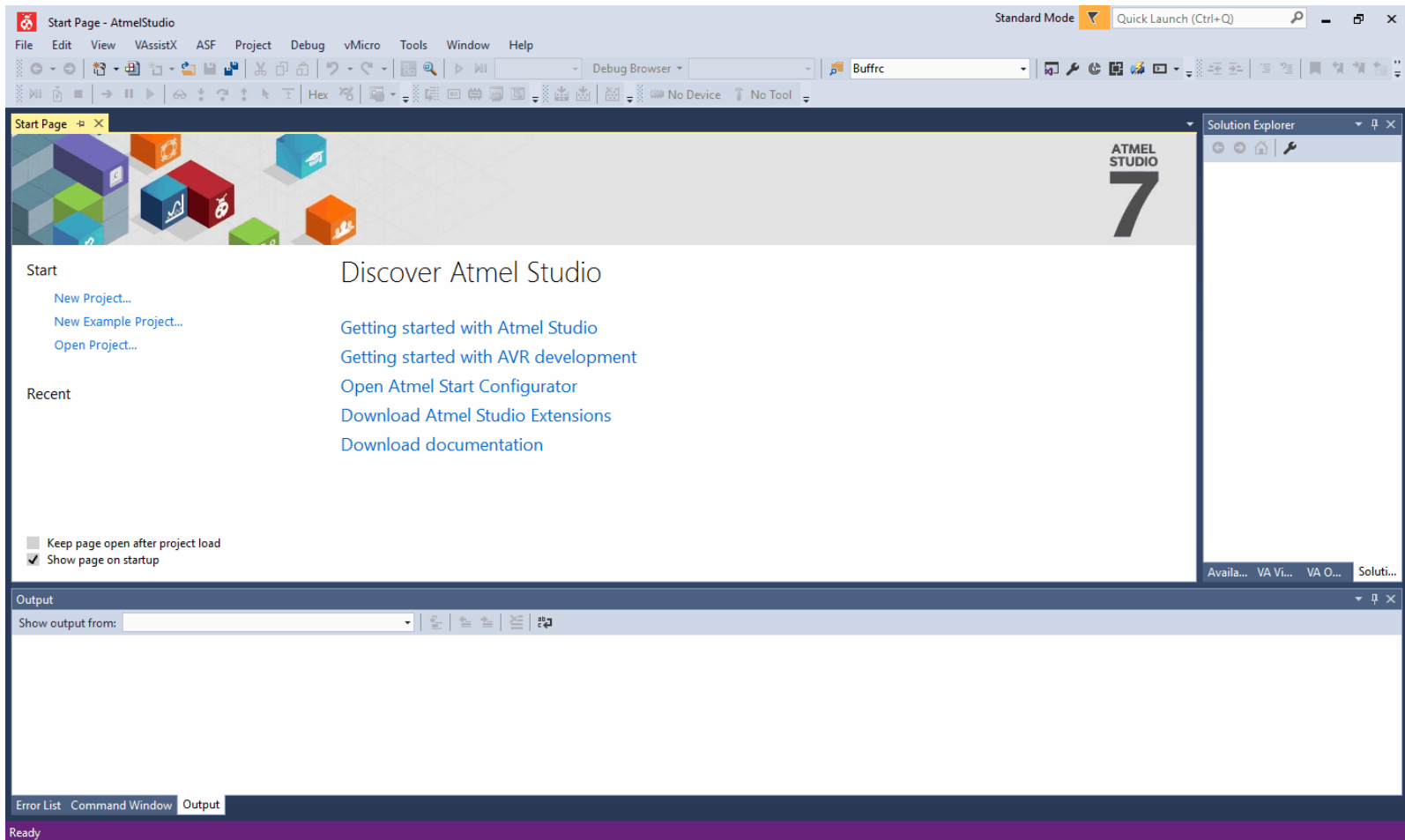
- Founded in 1984
- By George Perlegos



- Worked in the memory group of intel in the 1970s
- Co-founded seeq technology to manufacture **EPROM** memory
- Atmel was initially operated as a fabless company
 - 1989, purchase fabrication facility
- In 2016, acquired by microchip technology
 - Previously heard offers, in 2008 (Microchip Technology) and 2015

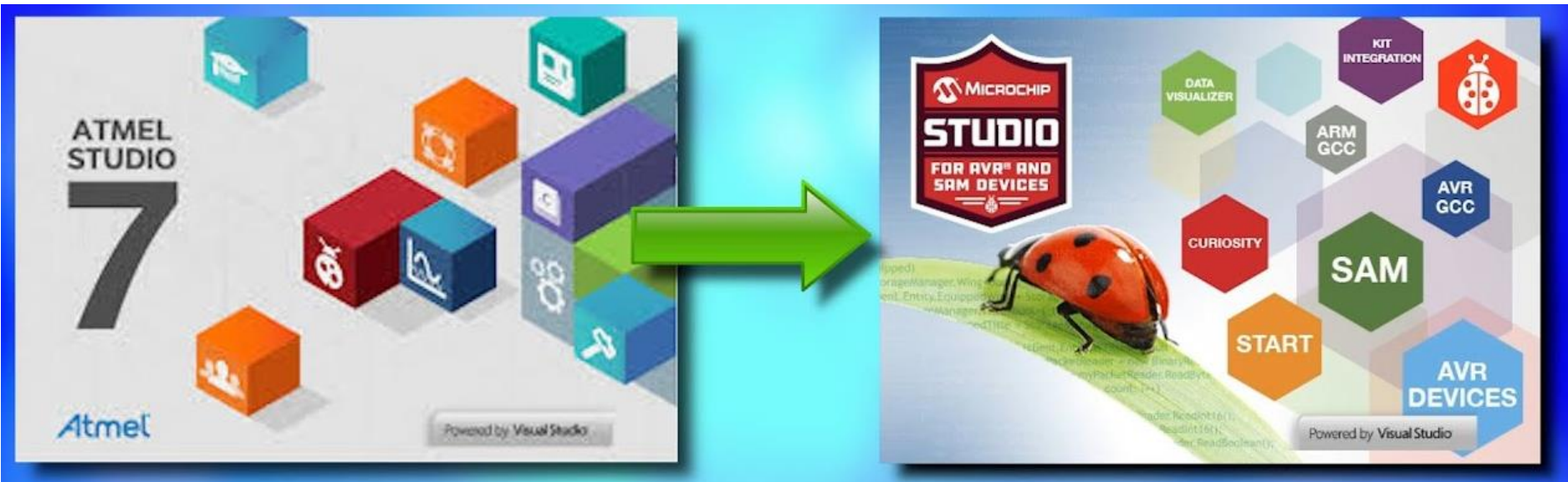


Atmel Studio



Atmel Studio

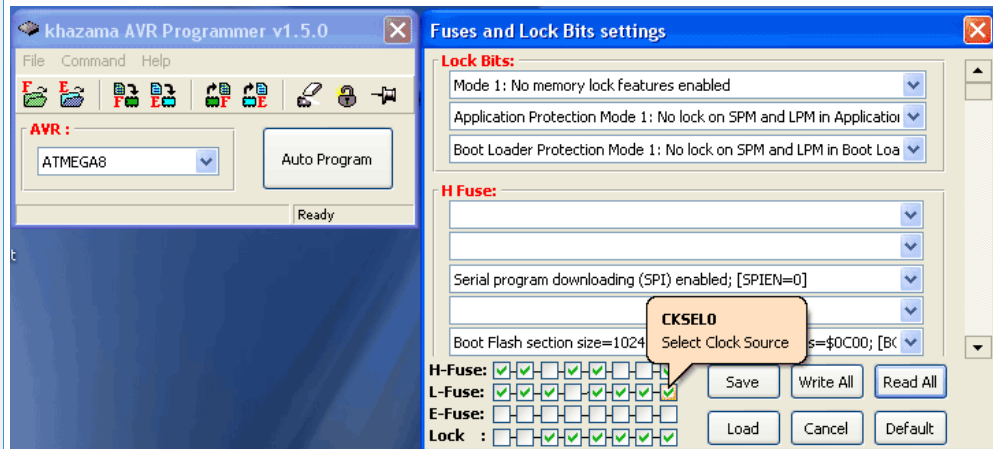
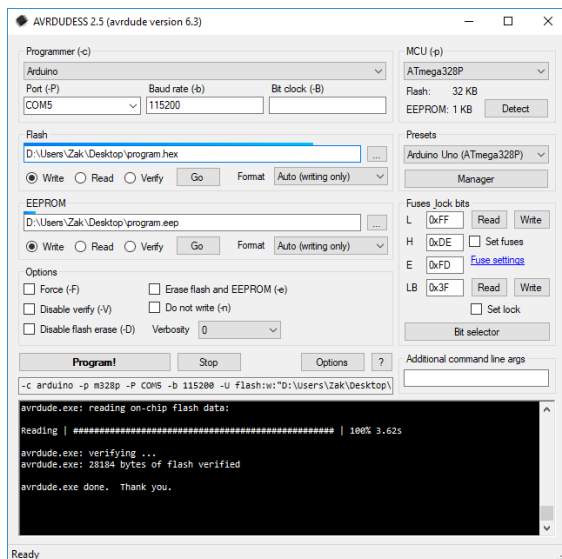
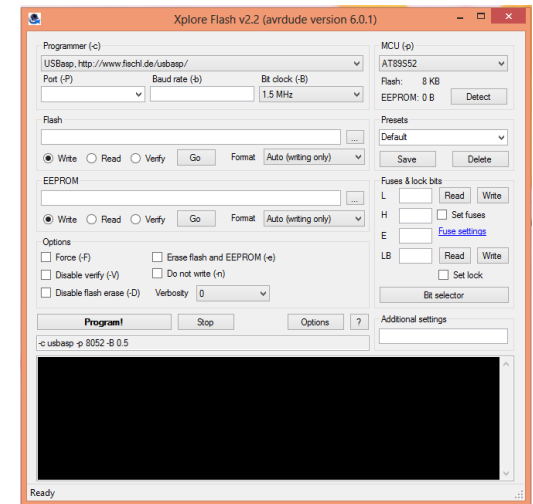
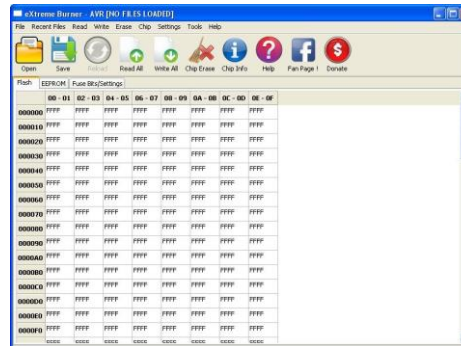
- Atmel Studio —————> **Microchip Studio**
 - After Acquired by Microchip Technology



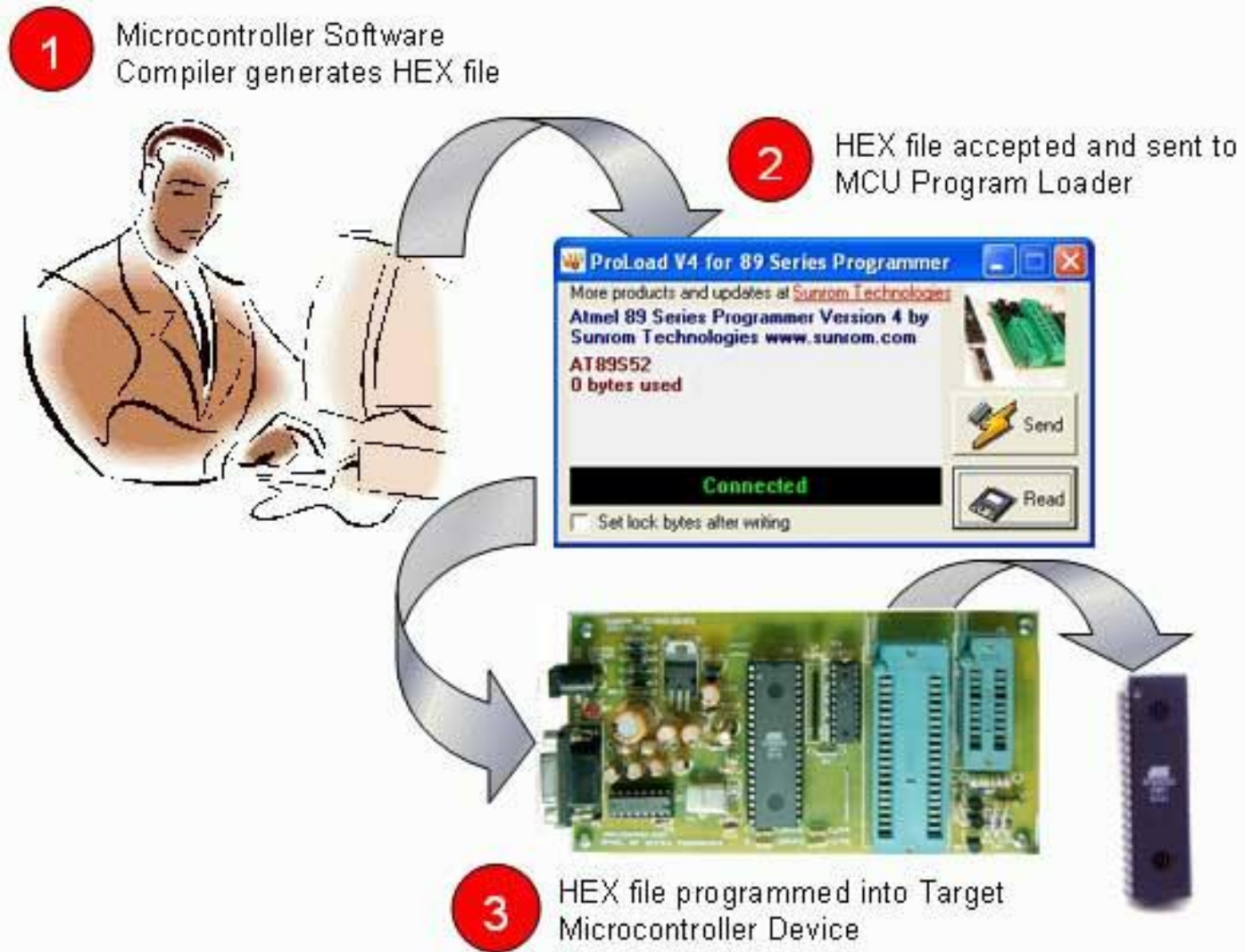
Toolchain

- Programming Software

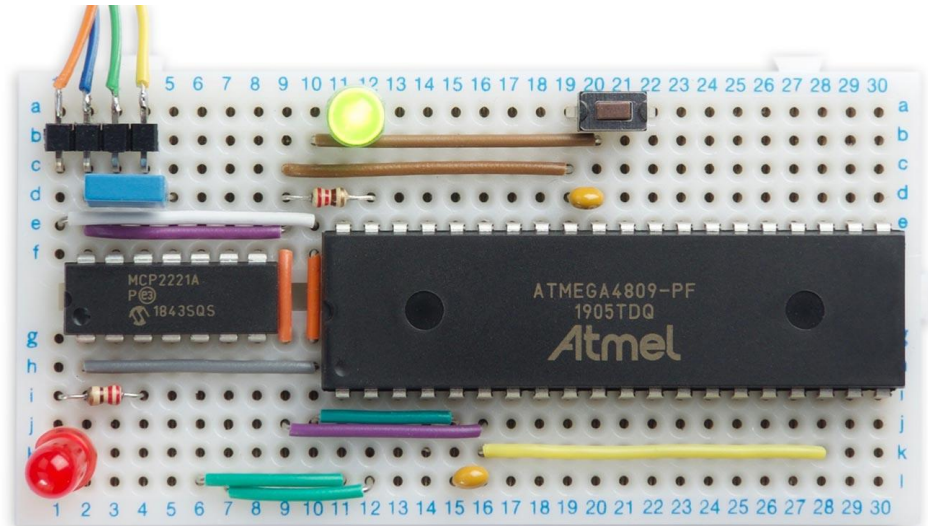
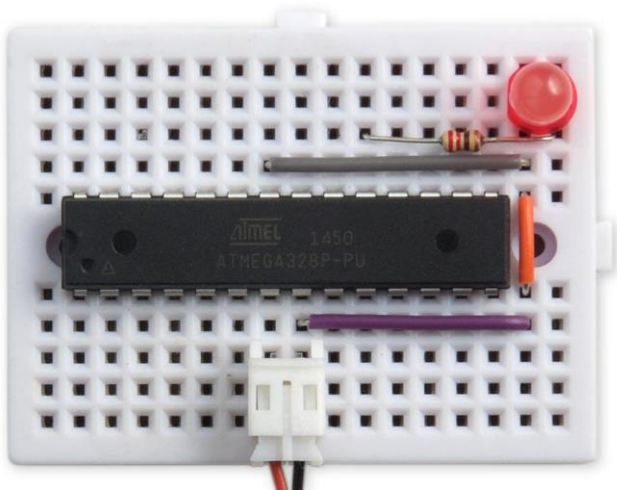
- Xplore Flash
- Extreme Burner
- Khazama
- AVRDUDE (AVRDUDESS – A GUI for AVRDUDE)



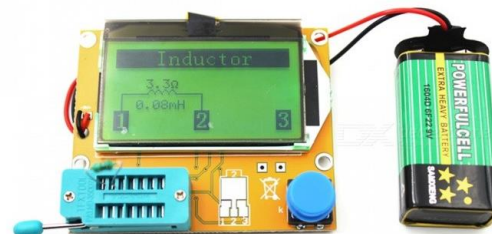
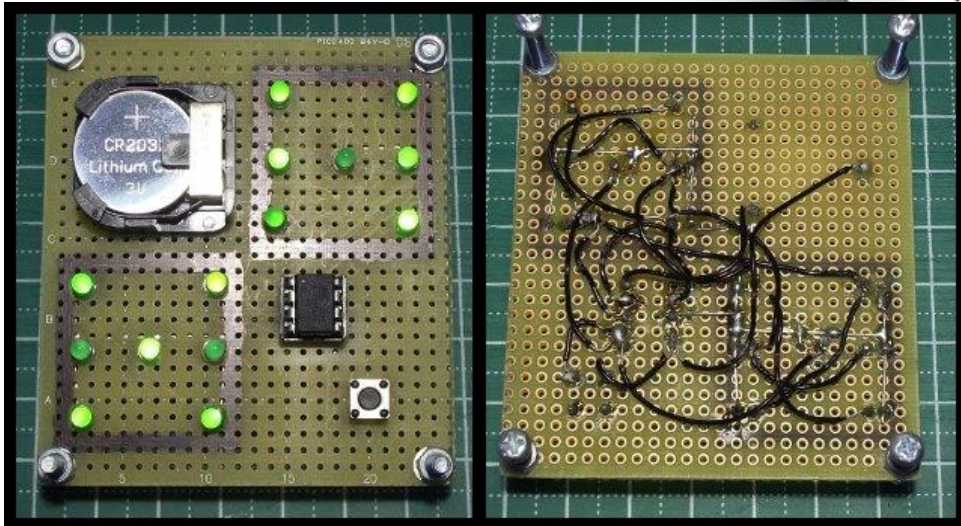
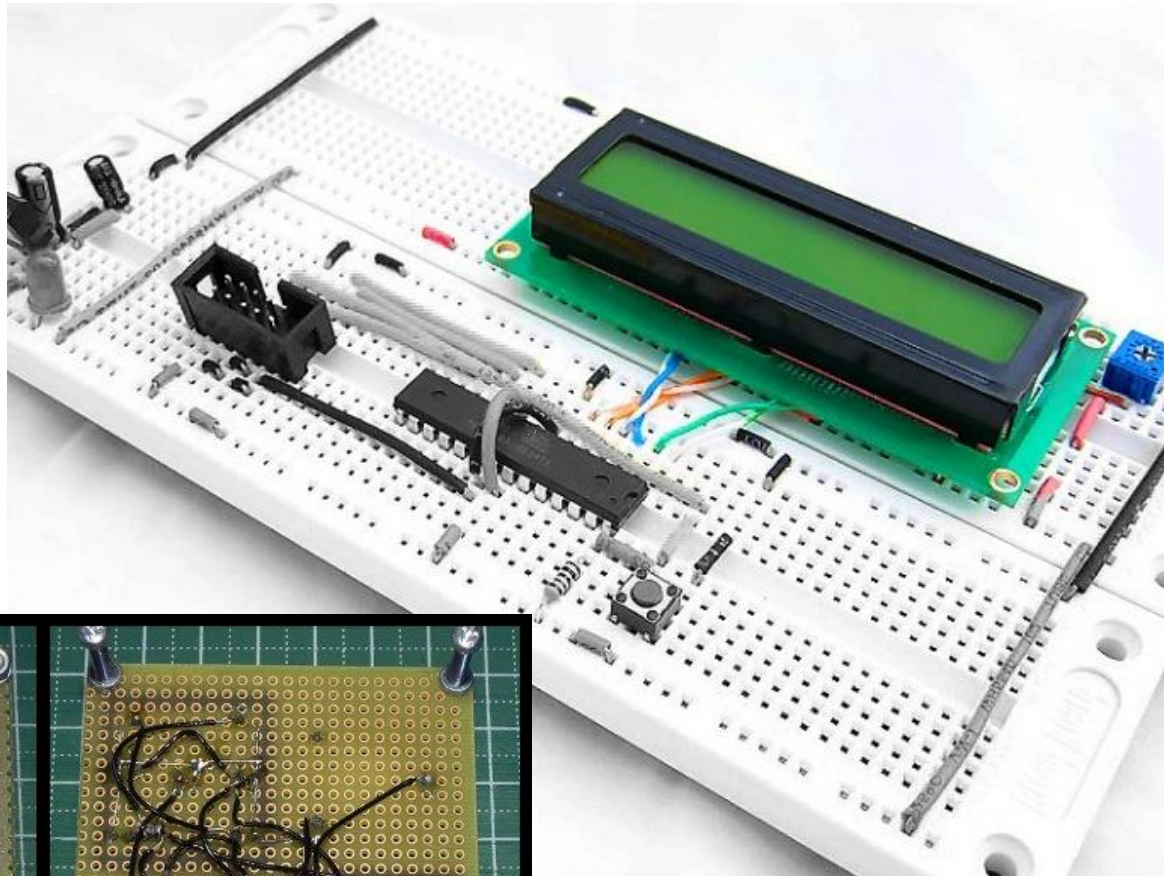
Program AVR μ C



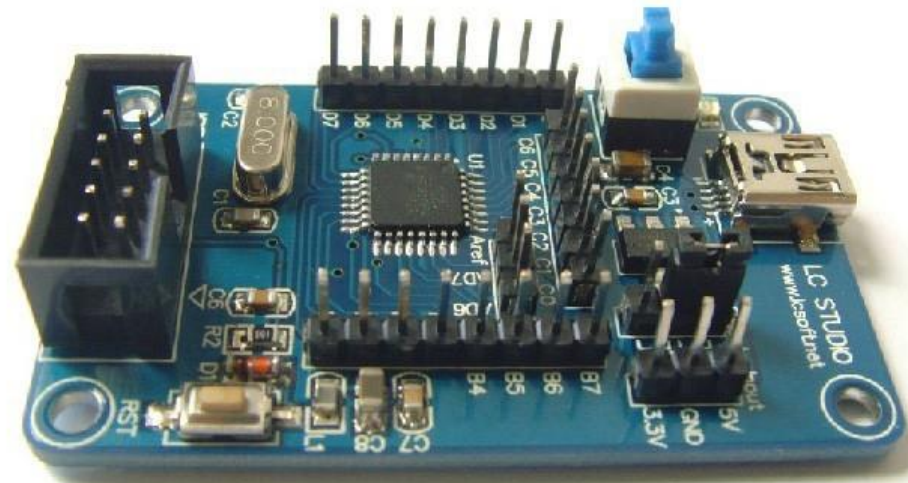
سخت افزار مورد نیاز



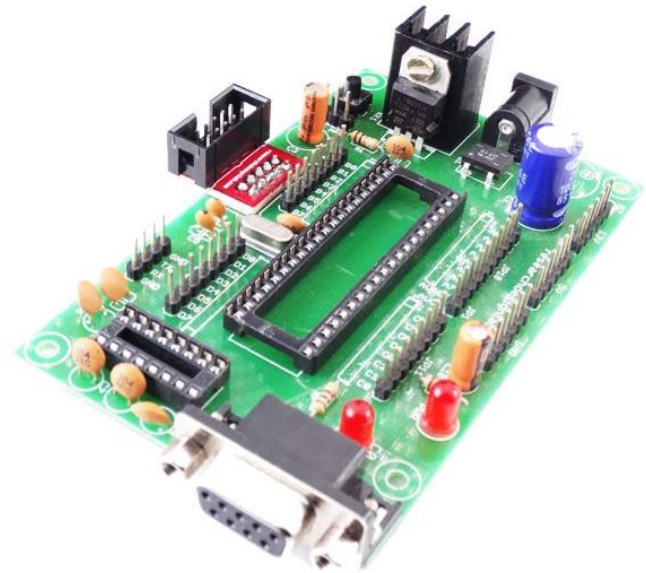
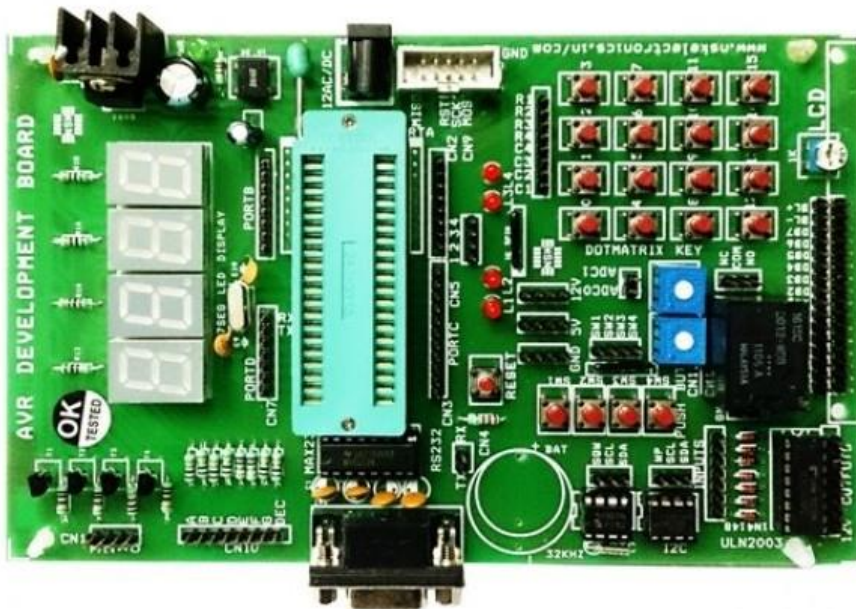
سخت افزار مورد نیاز



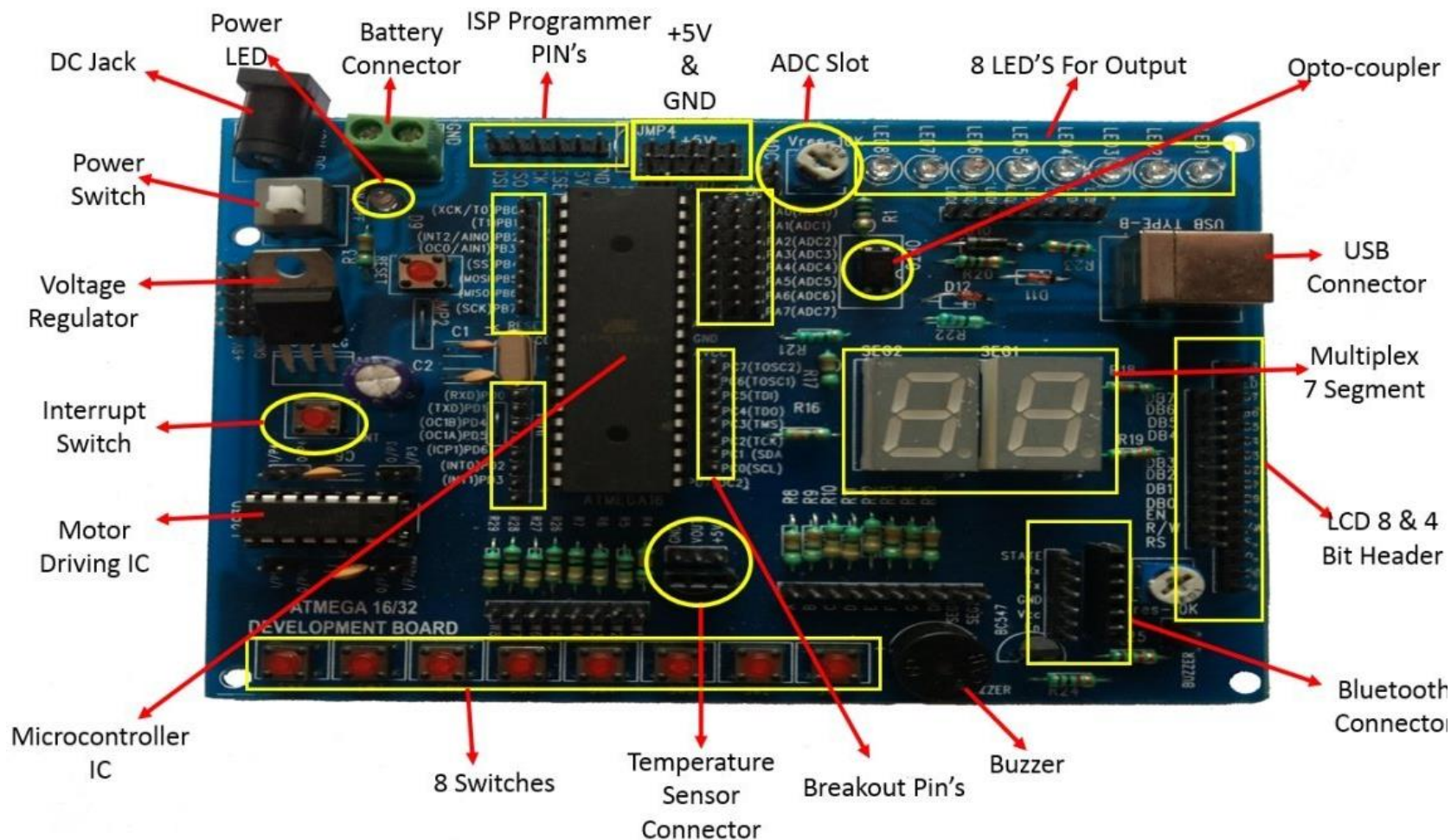
مدار کامل



برد توسعه



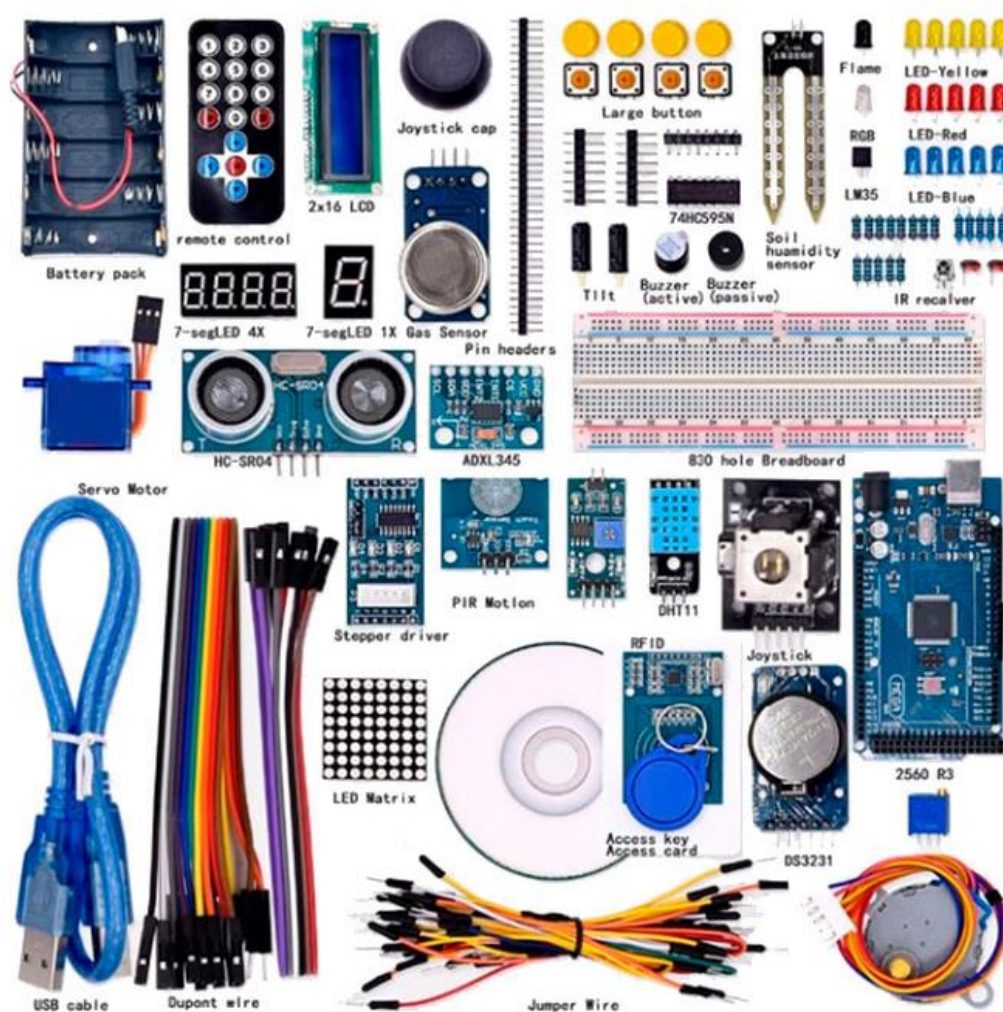
برد توسعه



برد آردوینو



برد آردوینو





محیط توسعه آردوینو

A screenshot of the Arduino IDE interface. The window title is "Blink | Arduino 1.8.5". The top toolbar shows icons for checking, running, uploading, and downloading. The main text area displays the Blink example code. The code includes a comment about the public domain, a URL to the Arduino tutorial, and the setup and loop functions. The setup function initializes the LED_BUILTIN pin as an output. The loop function turns the LED on and off with 1000ms delays. The bottom status bar shows "32" and "Arduino/Genuino Uno on COM1".

```
Blink $
This example code is in the public domain.

http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {$
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}
```

پایان

موفق و پیروز باشید