第九讲
Arduino（I）
Lecture 9
Arduino (I)

明玉瑞 Yurui Ming

yrming@gmail.com

声明
Disclaimer

- 本讲义在准备过程中由于时间所限，所用材料来源并未规范标示引用来源。所引材料仅用于教学所用，作者无意侵犯原著者之知识版权，所引材料之知识版权均归原著者所有；若原著者介意之，请联系作者更正及删除。

  The time limit during the preparation of these slides incurs the situation that not all the sources of the used materials (texts or images) are properly referenced or clearly manifested. However, all materials in these slides are solely for teaching and the author is with no intention to infringe the copyright bestowed on the original authors or manufacturers. All credits go to corresponding IP holders. Please address the author for any concern for remedy including deletion.

微控制器与微处理器
Microcontroller and Microprocessor

- 微处理器与微控制器的集成示例如下图所示：

  Examples of microprocessor and microcontroller integration are as follows:

- Single-board computers

Raspberry Pi

Beagle Bone

Jetson

- Microcontroller Units (MCU, µC)

PIC18F

AVR

MICROCHIP

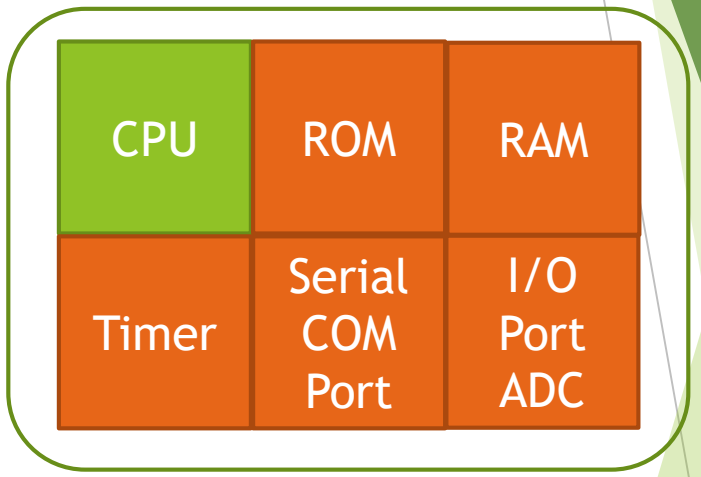
arm

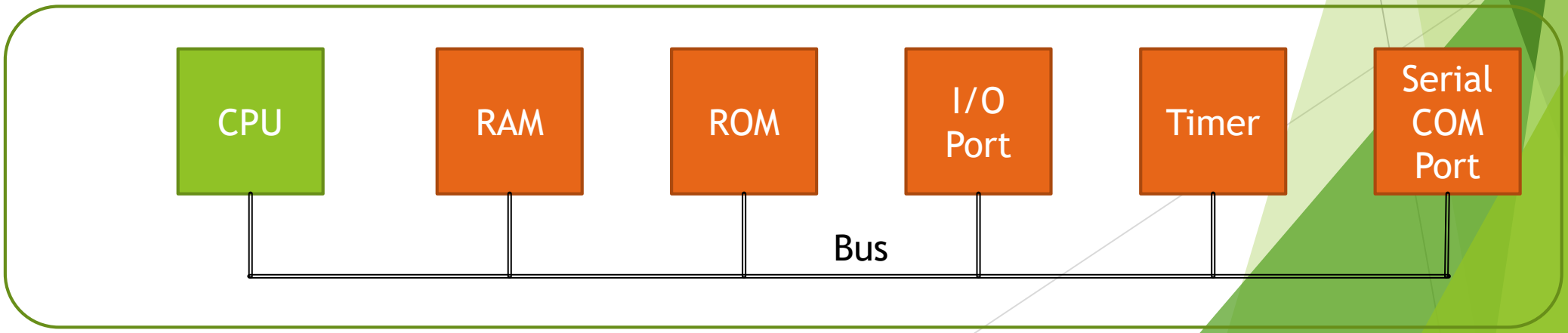
微控制器与微处理器
Microcontroller and Microprocessor

- 微处理器与微控制器的结构示例如下图所示：

Examples of microprocessor and microcontroller architecture are as follows:

Microcontroller→
Microprocessor↓

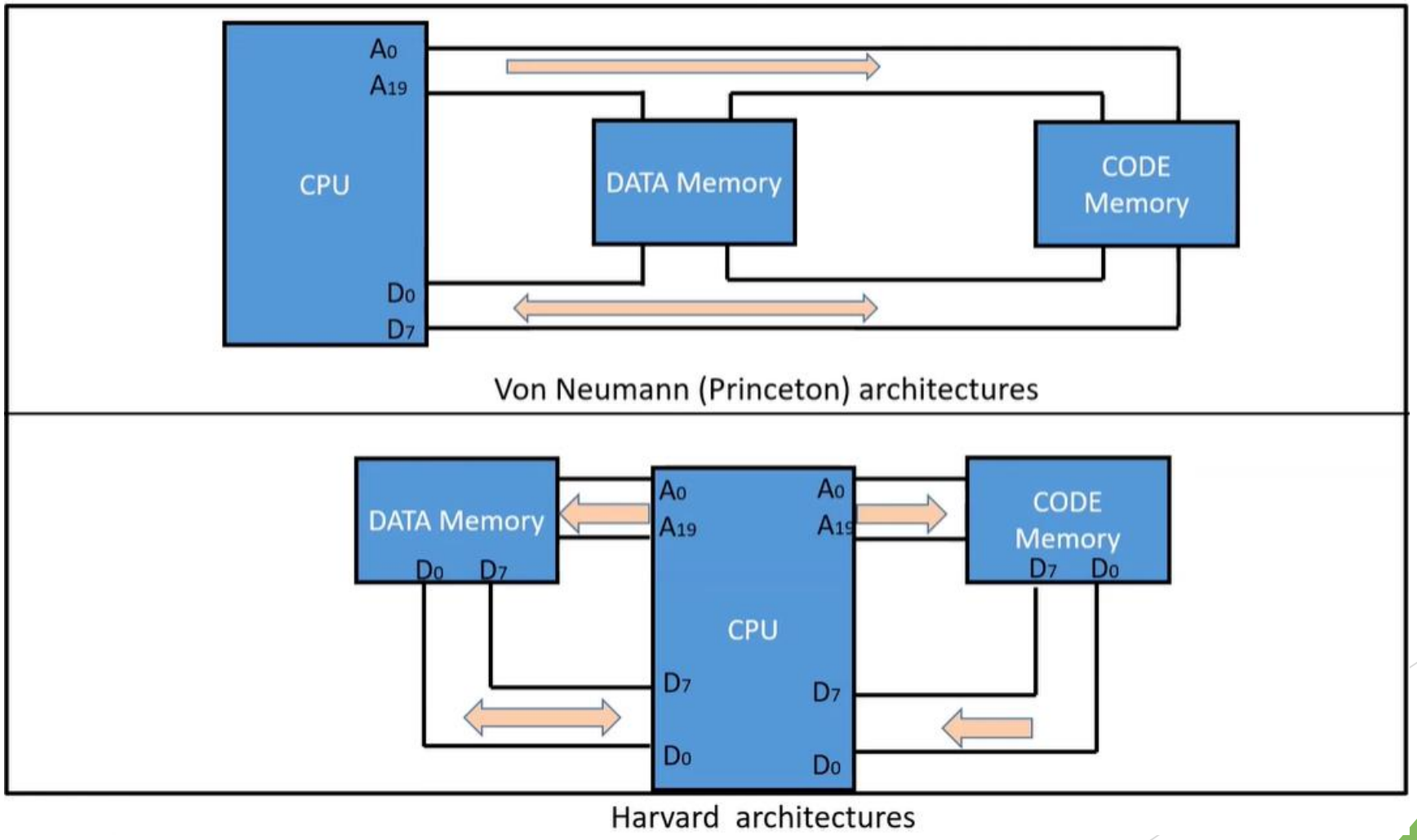




微控制器与微处理器
Microcontroller and Microprocessor

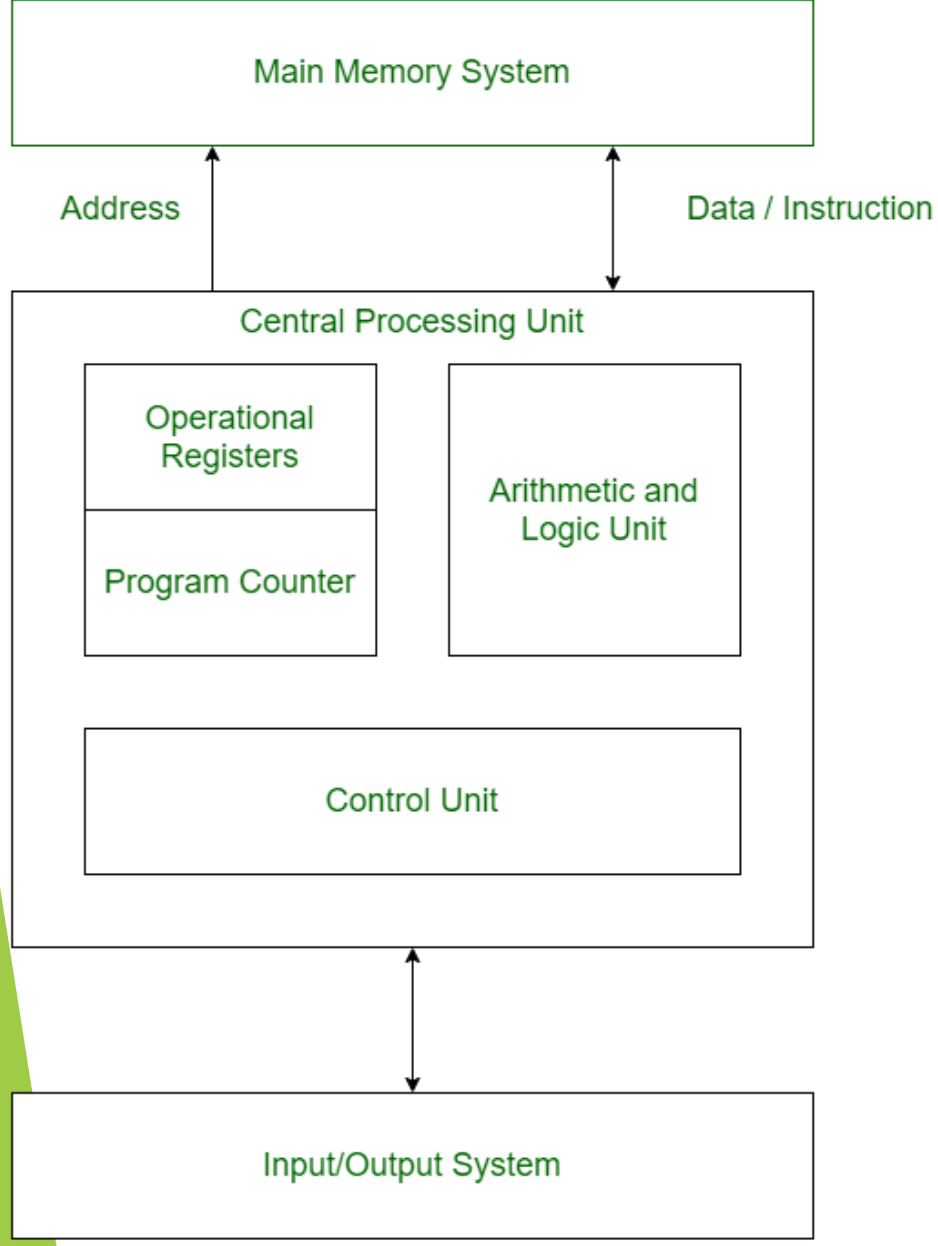
Microprocessor	Microcontroller
Contains only CPU: ROM, RAM, I/O, timer are separately provided	CPU, ROM, RAM, I/O, timer are integrated on one chip
Designer decides on the amount of ROM, RAM and I/O Ports	Fix amount of on-chip ROM, RAM and I/O ports
Usually Doesn’t support bit addressability	Supports bit addressability
General purpose and better for multitasking	Single purpose and weak for multitasking
Comes with von Neumann architecture and usually high speed and high cost	Comes with Harvard architecture and usually low speed and low cost
Microprocessor based system requires more hardware to be interfaced	Microcontroller based system requires less hardware to be interfaced

微控制器与微处理器
Microcontroller and Microprocessor

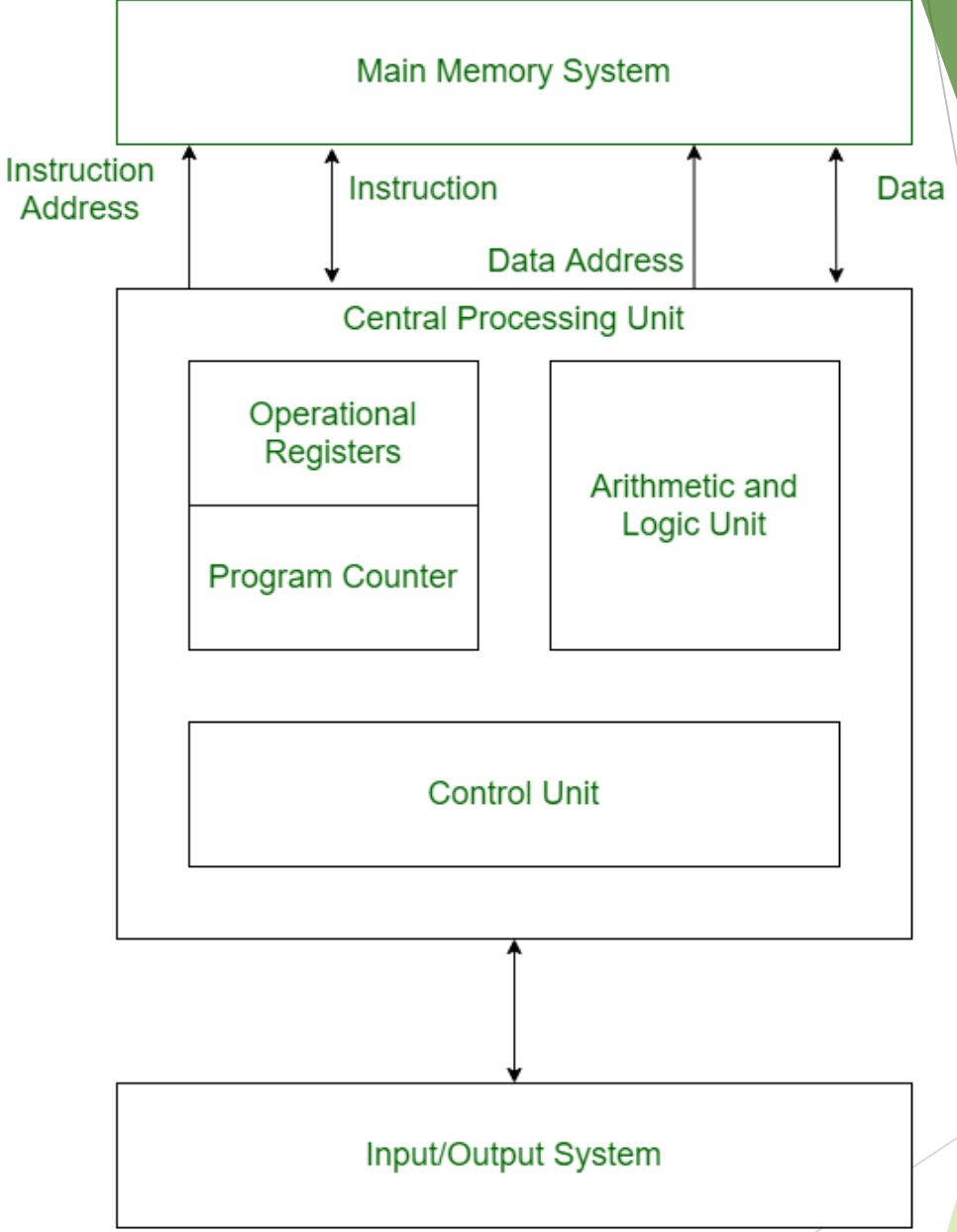


←Microprocessor

←Microcontroller



Von Neumann Architecture

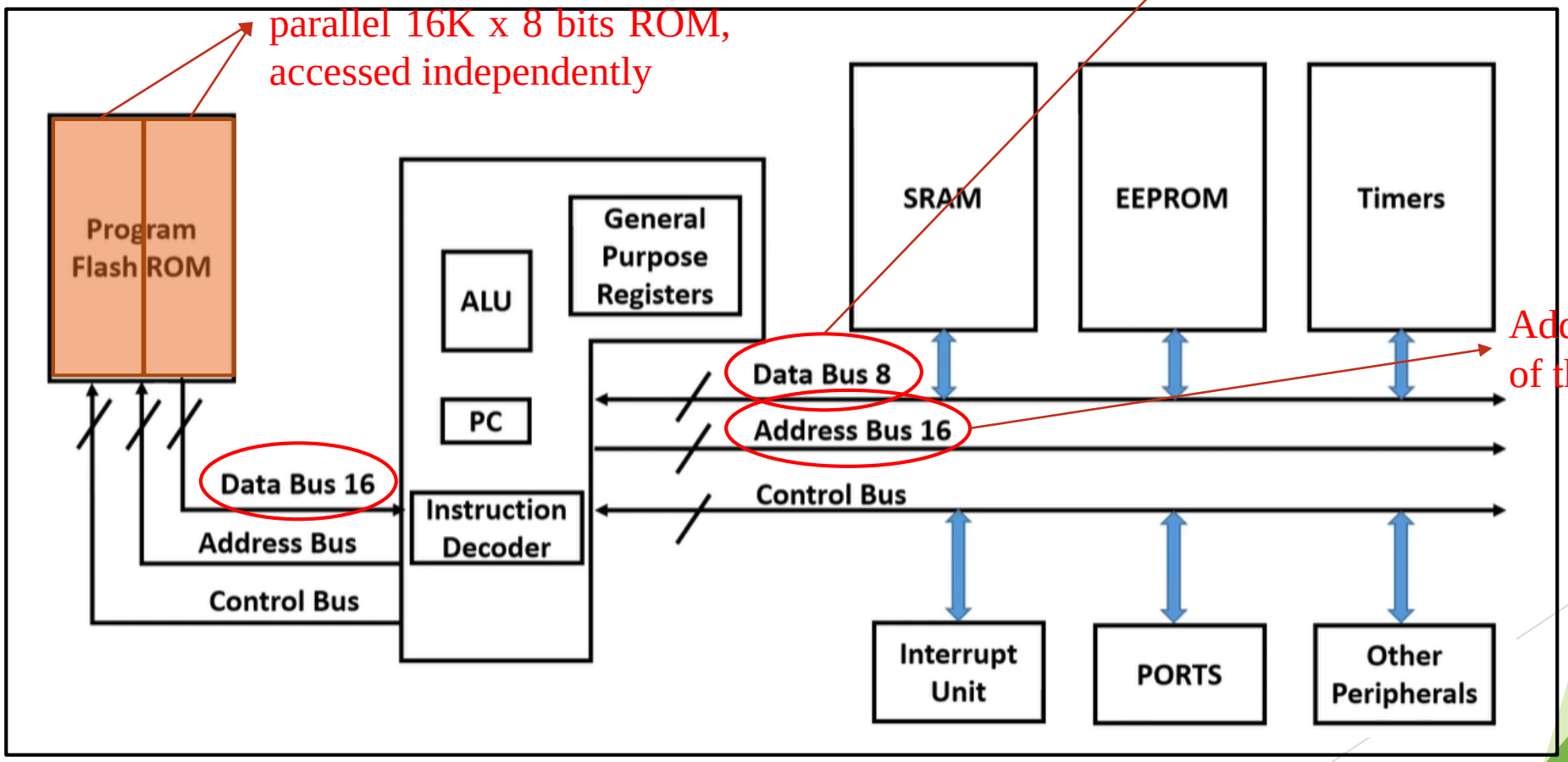


Harvard Architecture

微控制器与微处理器
Microcontroller and Microprocessor

8-bit microcontroller, so the data bus width is 8.

Flash ROM consists of two parallel 16K x 8 bits ROM, accessed independently



Address bus is not necessary of the same width as data bus.

Harvard Architecture in AVR

指令集
Instruction Set

- 精简指令集架构(RISC)—通过使用由几个基本步骤组成的指令集来简化硬件，用于加载、处理和存储等操作。例如使用加载命令加载数据，使用存储命令存储数据。其以每个程序的指令数为代价减少每条指令的周期来提高 CPU 性能

  Reduced Instruction Set Architecture (RISC) – To make hardware simpler by using an instruction set composed of a few basic steps for loading, evaluating, and storing operations. Just like a load command will load data, a store command will store the data. It is by reducing the cycles per instruction at the cost of the number of instructions per program to increase the CPU performance.

- 复杂指令集架构(CISC) —一条指令将执行所有加载、评估和存储操作，就像乘法命令将执行加载数据、评估和存储数据等操作一样，因此它很复杂。其用最小化每个程序的指令数、增加每条指令的周期数来提高 CPU 性能。

  Complex Instruction Set Architecture (CISC) – A single instruction will do all loading, evaluating, and storing operations. Just like a multiplication command will do stuff like loading data, evaluating, and storing it, hence it’s complex. It is by minimizing the number of instructions per program but at the cost of an increase in the number of cycles per instruction to increase the CPU performance.

RISC处理器
RISC Processor

- RISC处理器具有固定的指令大小。例如，AVR主要使用 2 字节指令，很少使用 4 字节，而8051的CISC使用1、2或3字节指令。尽管有限的指令数量可能会导致编程困难，但其95% 的指令在1个时钟周期内执行。

  RISC Processors have a fixed instruction size. For, example, AVR uses mostly 2 byte instruction and very few 4 bytes as opposed to CISC in 8051 which uses 1, 2 or 3 byte instructions. Although the limited number of instructions might incur difficulty in programming, however, 95% of its instructions are executed within 1 clock cycle.

- RISC处理器提供了大量的寄存器。大多数RISC处理器提供32个寄存器用于存储数据而不是使用堆栈。同时，它使用单独的地址总线和数据总线来访问代码和数据，从而提供了很好的速度。注意，RISC使用加载/存储机制，也即RISC不能直接处理内存数据。

  It provides large number of registers. Most RISC processor provides 32 registers to be used to storing data instead of using stack. Meanwhile, it uses separate address buses and data buses for accessing code and data, which renders a good speed. Notably, RISC uses load/store mechanism, which means it cannot perform operations directly with memory.

AVR控制器
AVR Microcontroller

- AVR 最初由Atmel公司于1996年开发，该公司后来在2016年被Microchip Technology收购。AVR是Advanced Virtual RISC或Alf and Vegard RISC的缩写。

  AVR was originally developed by Atmel corporation in 1996, which was later acquired by Microchip Technology in 2016. AVR is an initialism either of Advanced Virtual RISC or Alf and Vegard RISC.

- AVR是基于RISC的8位单芯片微控制器，采用哈佛架构。它具有代码ROM、数据ROM (EEPROM)、RAM、I/O和定时器。它还支持CAN、I2C、SPI、USB、USART等一些串行接口。AVR产品主要分为四大类，即Tiny （如ATtiny25）、Mega（如ATmega32）、Special Purpose（如AT90CAN128）、Classic（如AT90S2313）。

  AVR is an 8 bit RISC based single chip microcontroller with Harvard architecture. It has code ROM, data ROM (EEPROM), RAM, I/O and timers. It also supports some serial interfaces such as CAN, I^2C, SPI, USB, USART. AVR family falls into four categories, namely, Tiny (such as ATtiny25), Mega (such as ATmega32), Special Purpose (such as AT90CAN128), Classic (such as AT90S2313)

AVR控制器
AVR Microcontroller

- 常用的Arduino Uno的微控制器为ATmega32。Mega系列相关对比如下图所示：

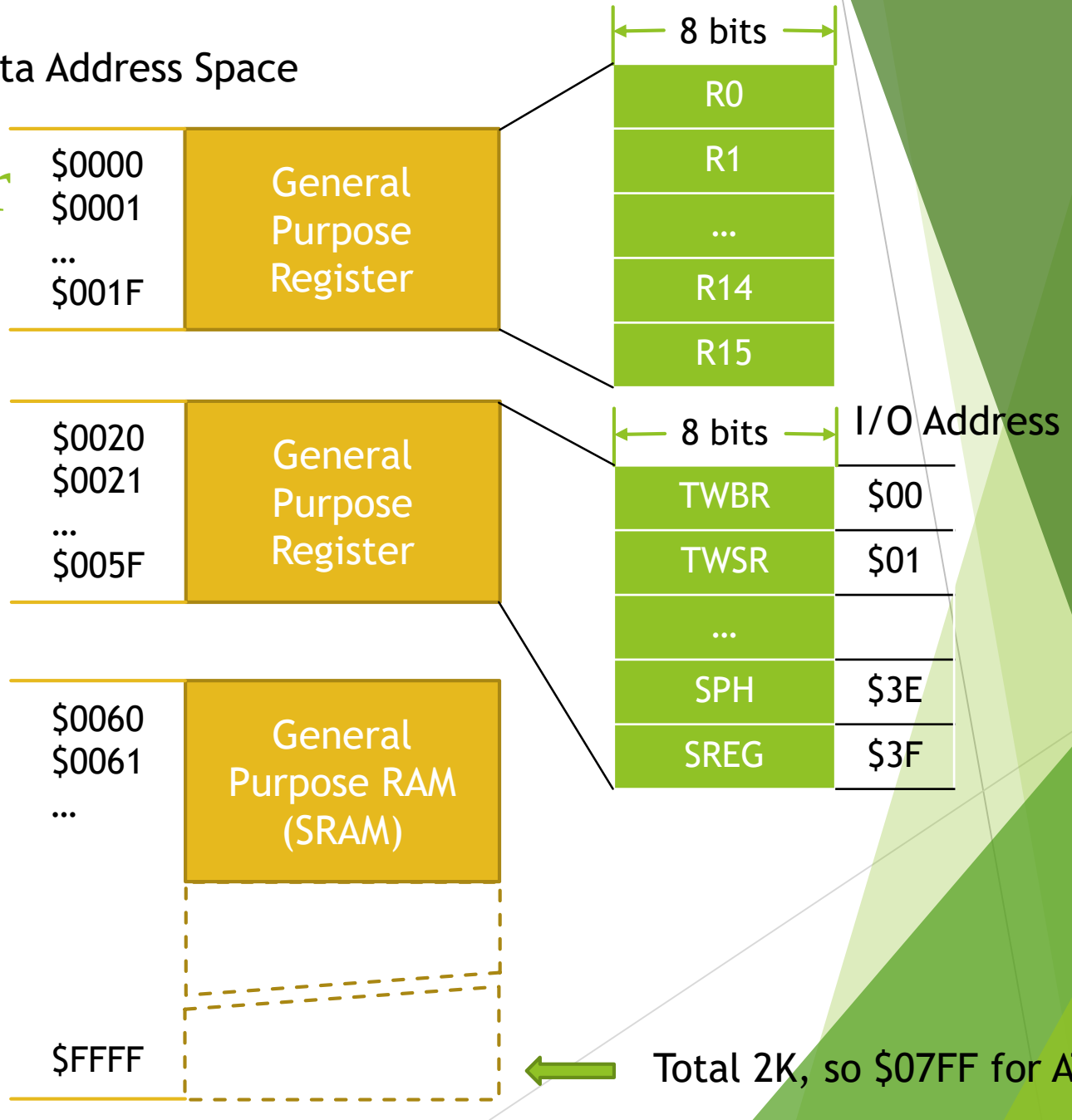
  The microcontroller of the commonly-used Arduino Uno is ATmega32. The comparison of AVR Mega series is in the following figure:

Part Number	Code ROM	DATA RAM	DATA EPROM	I/O pins	ADC	Timers	Pin numbers & Packages
ATmega8	8K	1K	0.5K	23	8	3	TQFP32,PDIP28
ATmega16	16K	1K	0.5K	32	8	3	TQFP44,PDIP40
Atmega32	32K	2K	1K	32	8	3	TQFP44,PDIP40

AVR控制器
AVR Microcontroller

- ATmega32的RAM 架构如右图所示:

  The RAM architecture of ATmega32 is shown in the right figure:



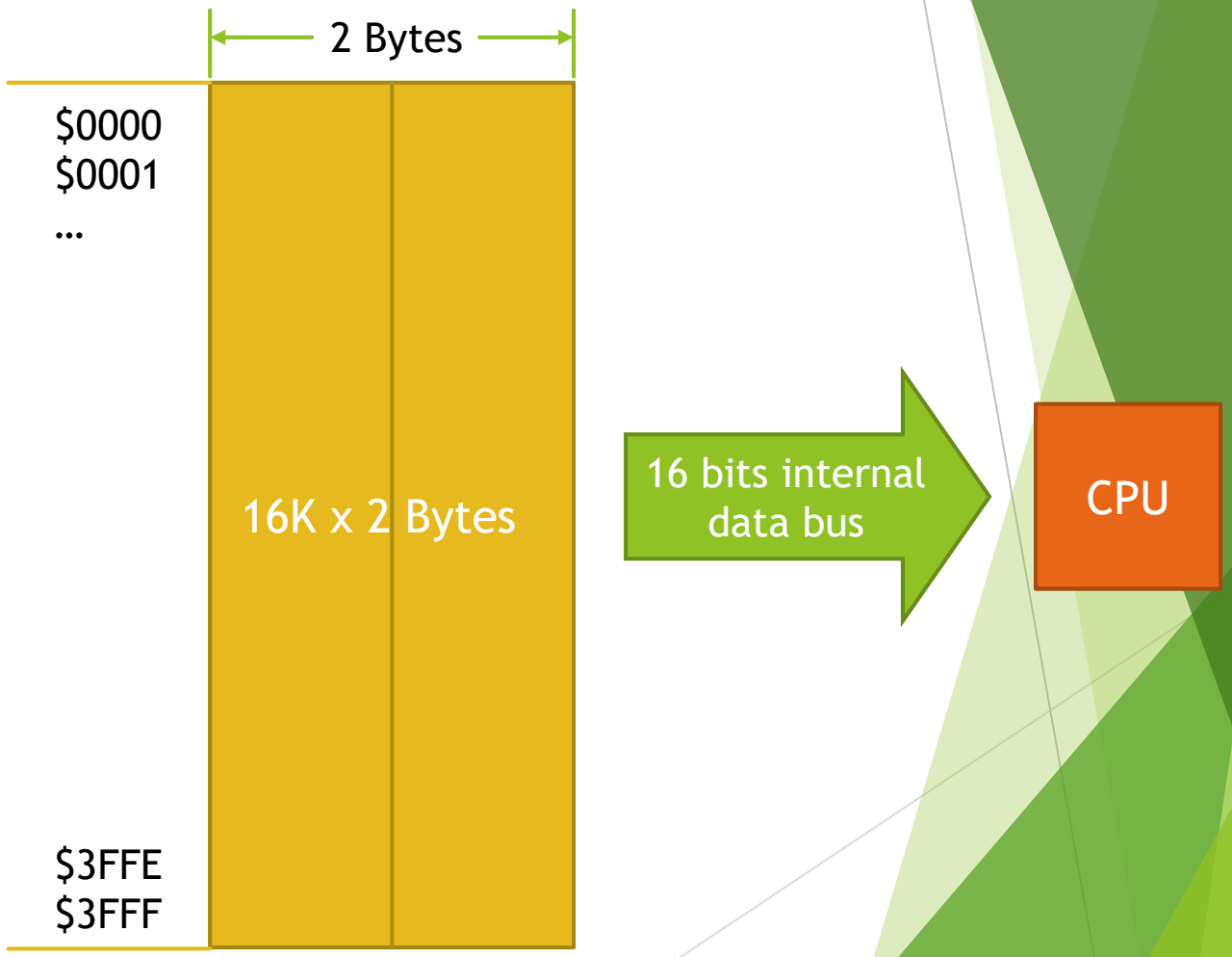
AVR控制器
AVR Microcontroller

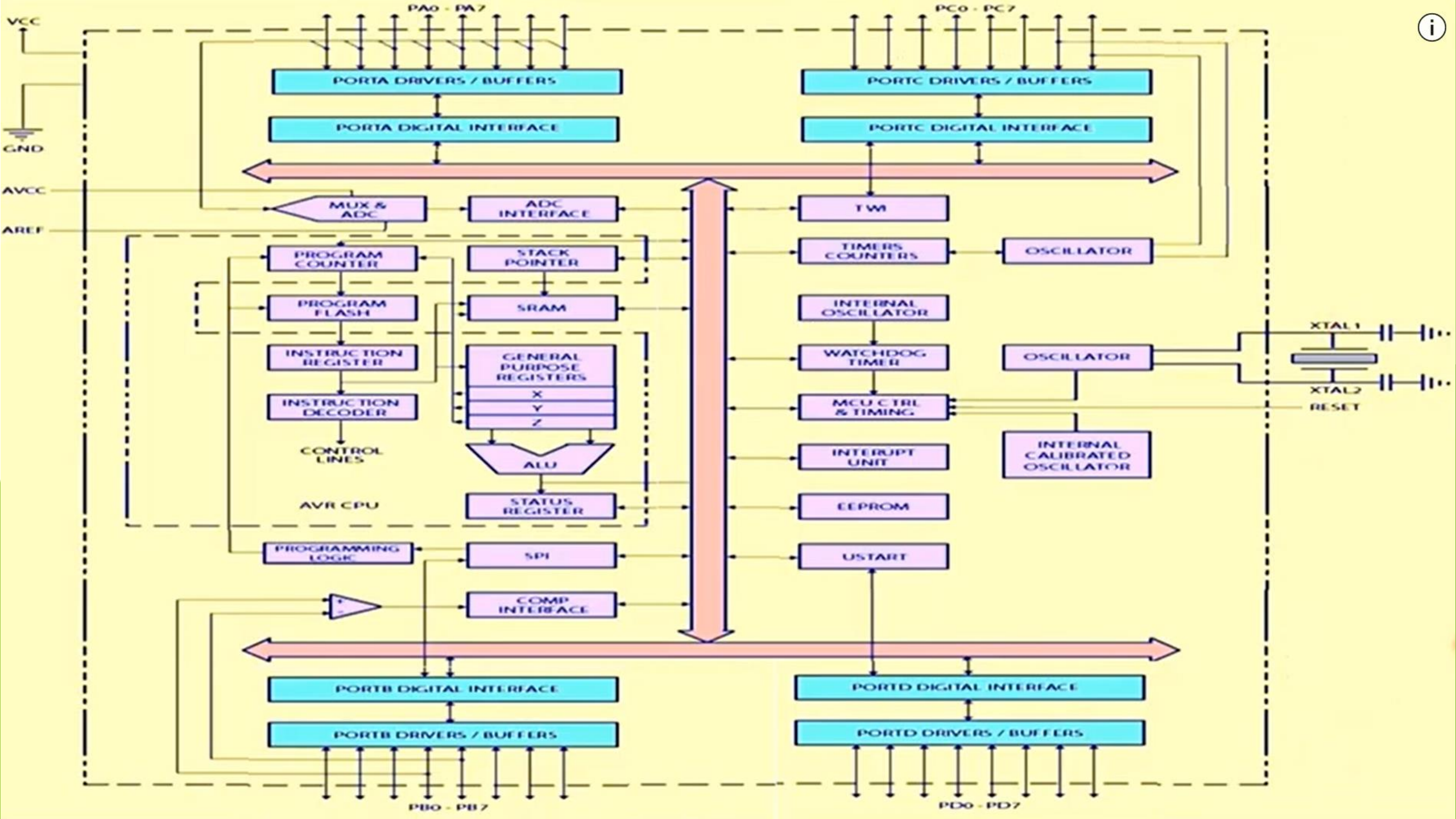
- ATmega32的ROM （32K） 架构如右图所示:

  The ROM (32K) architecture of ATmega32 is shown in the right figure:

- CPU每次只处理8位或1字节数据。

  The CPU only processes 8 bits or 1 byte data at a time.

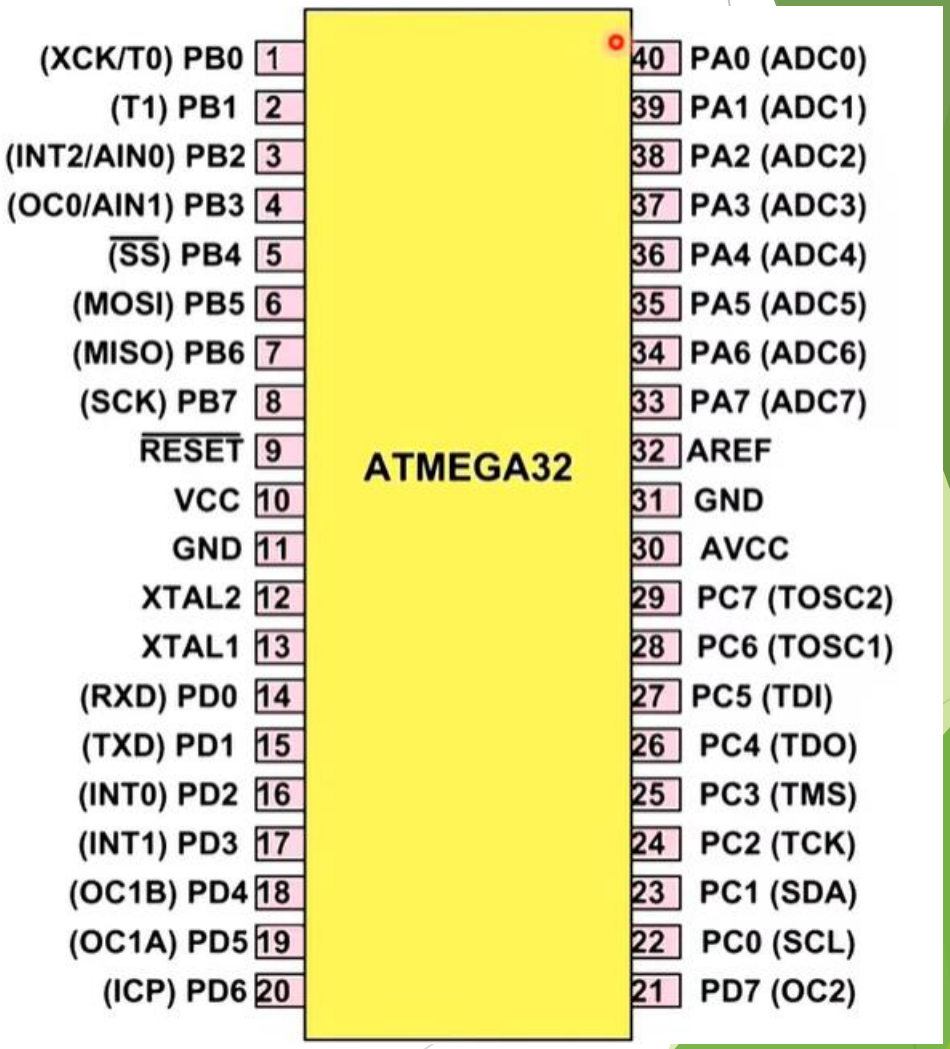



VCC
GND
AVCC
AREF
PA0 - PA7
PORTA DRIVERS / BUFFERS
PORTA DIGITAL INTERFACE
PC0 - PC7
PORTC DRIVERS / BUFFERS
PORTC DIGITAL INTERFACE
MUX & ADC
ADC INTERFACE
TWI
PROGRAM COUNTER
STACK POINTER
TIMERS COUNTERS
OSCILLATOR
PROGRAM FLASH
SRAM
INTERNAL OSCILLATOR
INSTRUCTION REGISTER
GENERAL PURPOSE REGISTERS
X
Y
Z
WATCHDOG TIMER
OSCILLATOR
XTAL1
XTAL2
INSTRUCTION DECODER
CONTROL LINES
ALU
MCU CTRL & TIMING
RESET
INTERNAL CALIBRATED OSCILLATOR
AVR CPU
STATUS REGISTER
INTERUPT UNIT
EEPROM
PROGRAMMING LOGIC
SPI
USTART
COMP INTERFACE
PORTB DIGITAL INTERFACE
PORTD DIGITAL INTERFACE
PORTB DRIVERS / BUFFERS
PORTD DRIVERS / BUFFERS
PB0 - PB7
PD0 - PD7

ATmega32管脚示意图
ATmega32 Pin Diagram

- 从右图可见，ATmega32提供了丰富的管脚，如何利用这些管脚实现复杂的系统功能，是利用AVR系列微控制器进行产品设计需要考量的地方。这是由于管脚可能不能直接用，需要外加接口电路才能与其他模块或系统交互。

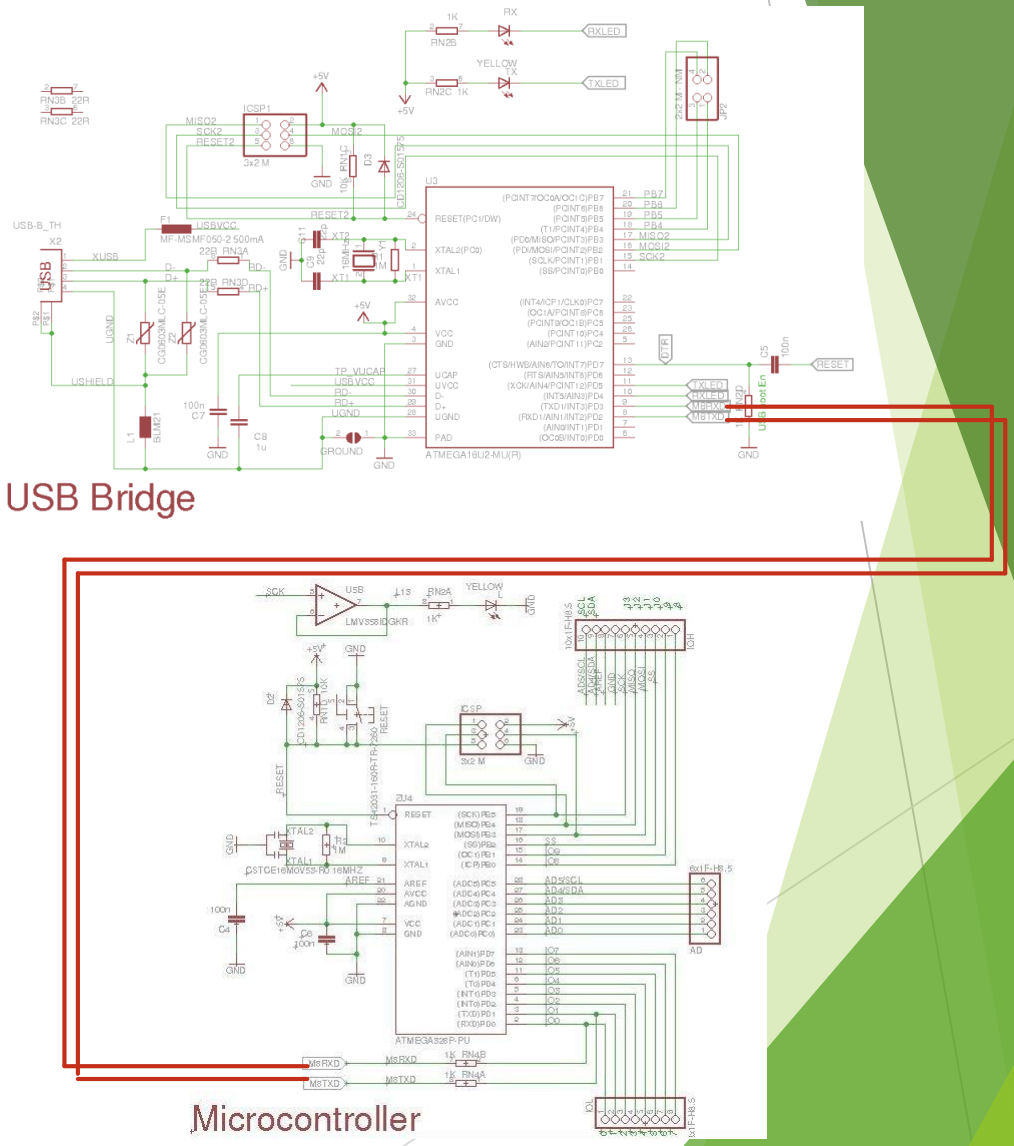
  The right figure indicates that ATmega32 provides rich pins. How to make good use of these pins to implement complicated system functionalities is the key to take advantage of AVR series microcontrollers in product design. This is because the pins may not be used directly, and additional interface circuits are required to interact with other modules or systems.



ATmega32管脚示意图
ATmega32 Pin Diagram

- 例如，ATmega32并不具有原生的USB通信的功能，如果想用RXD与TXD两个管脚，通过USB接口与PC通讯，则需要额外的USB接口电路，这可以通过额外一块ATmega16U2完成。

  For example, ATmega32 does not support communication via USB natively. If you want to use two pins, RXD and TXD, to communicate with PC through USB interface, you need an additional USB interface circuit, which can be done by an additional ATmega16U2.



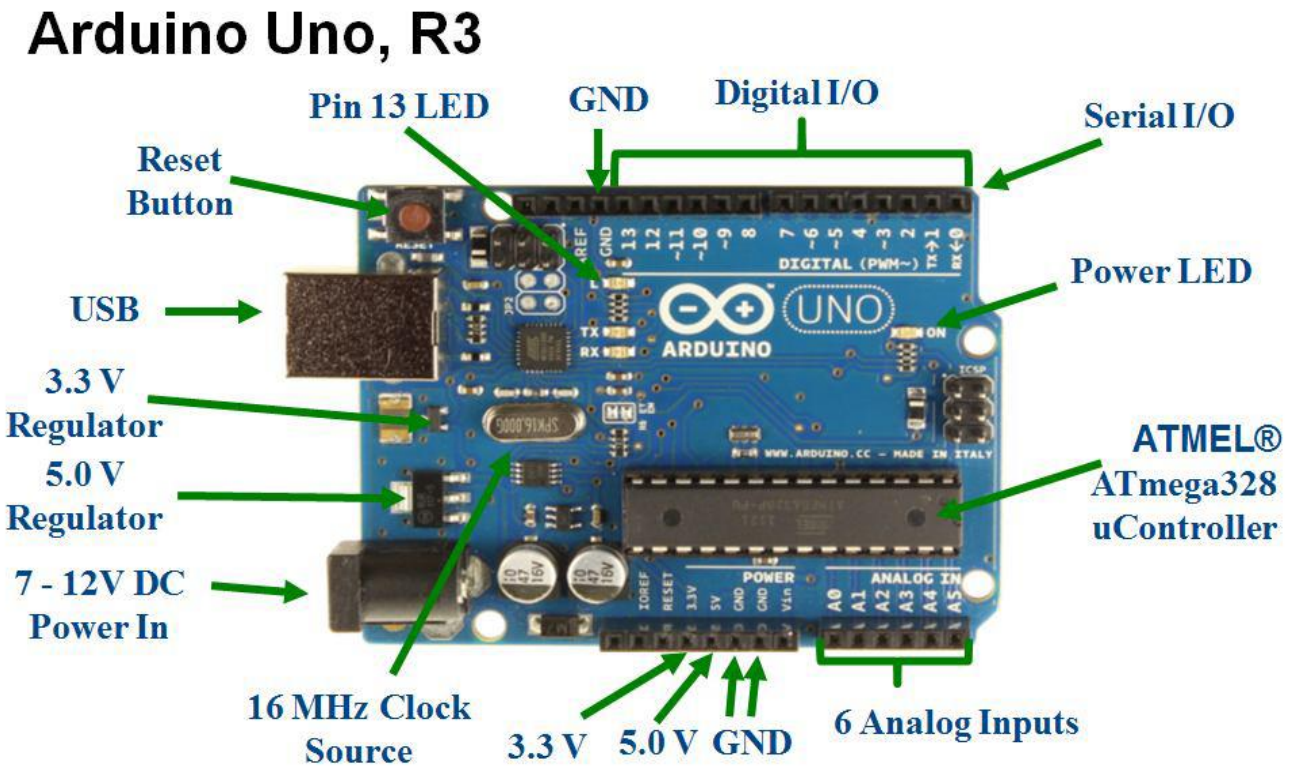
Arduino

- 意大利伊夫雷亚交互设计学院为给学生新手和专业人士提供一种低成本且简单的工具，于2005年创建了Arduino项目，以方便使用传感器和执行器与环境进行交互。

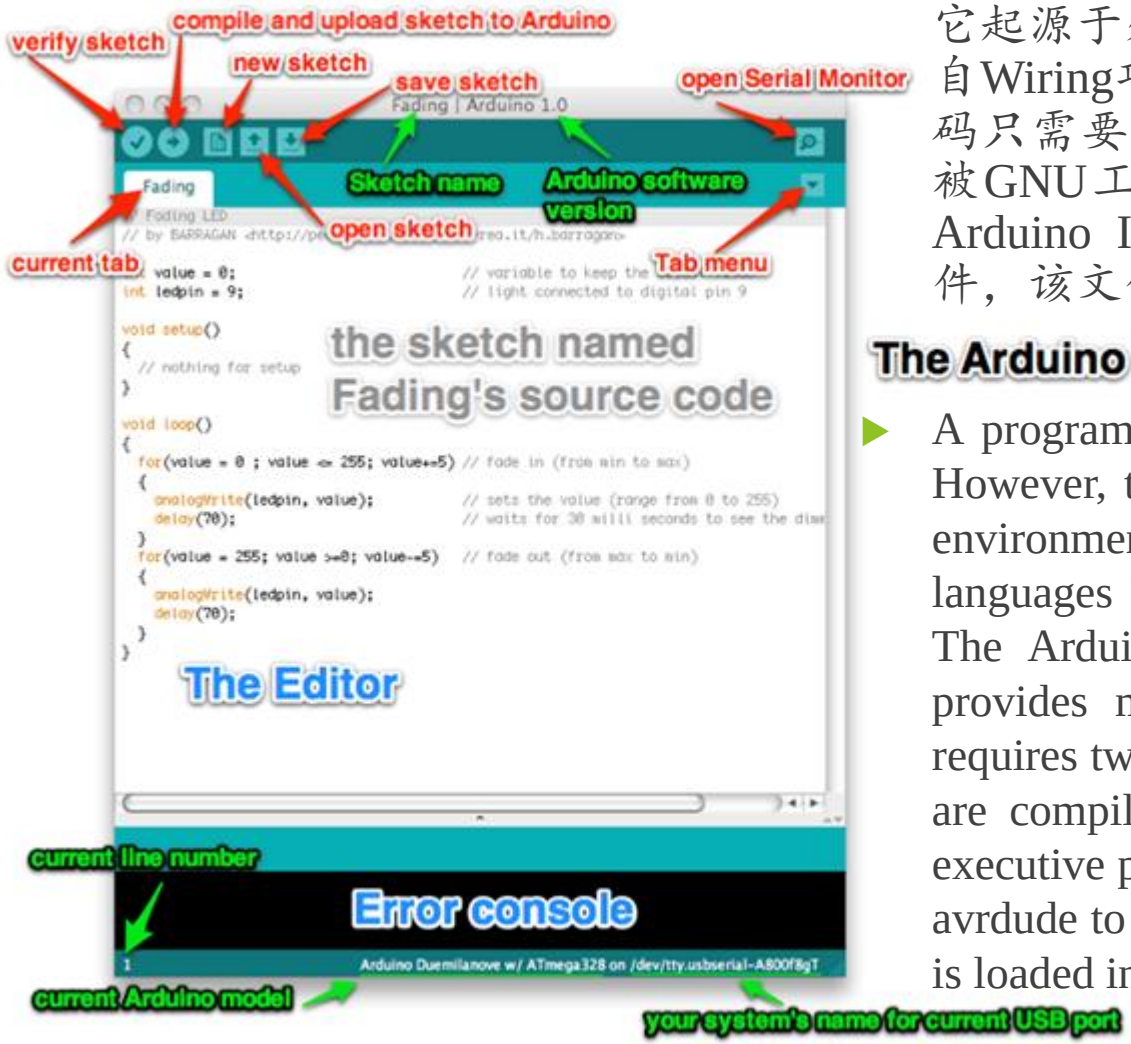
  The Arduino project began in 2005 as a low-cost and easy tool for student novices and professionals at The Interaction Design Institute Ivrea, Italy, to create devices that interact with their environment using sensors and actuators.

- Arduino板设计使用各种微处理器和控制器，并配备了一组数字和模拟输入/输出(I/O)引脚。最常使用的便是AVR系列微处理器，并且引出了尽可能多的管脚以供使用。

  Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins. The most commonly-used microprocessors are the AVR series with as many as pins exposed for interfacing with.



Arduino



- 尽管可以用任何编程语言编写Arduino硬件的程序，但Arduino项目的不同之处在于其提供了使用 Java 编程语言编写的跨平台应用程序Arduino IDE。它起源于处理和布线语言，但也支持C和C++语言。Arduino IDE提供了来自Wiring项目的软件库，包括许多常见的输入和输出程序。用户编写的代码只需要编写用于启动程序和主程序循环的两个基本函数。这两个函数被GNU工具链编译并与主程序main()链接成可执行的循环执行程序。Arduino IDE使用程序avrdude将可执行代码转换为十六进制编码的文本文件，该文件由板固件中的加载程序加载到Arduino板中。

The Arduino IDE

- A program for Arduino hardware may be written in any programming language. However, the Arduino project provides the cross-platform integrated development environment (IDE) that is written in Java. It originated from the IDE for the languages Processing and Wiring, but supports the languages C and C++ as well. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

Arduino

- 用Arduino IDE编写的程序有一个特殊的名字，称为草图，以文本文件的形式保存在计算机上，文件扩展名为 .ino。

  A sketch is a program written with the Arduino IDE. Sketches are saved on the computer as text files with the file extension .ino.

- 一个最小的 Arduino C/C++ 程序只包含两个函数：
  - setup()：在上电或复位后启动草图时调用一次此函数。它用于初始化变量、输入和输出引脚模式以及草图中所需的其他库。
  - loop()：setup()函数退出（结束）后，loop()函数在主程序中反复执行。它控制电路板，直到电路板断电或复位，它类似于控制结构while (1).

  A minimal Arduino C/C++ program consists of only two functions:
  - setup(): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
  - loop(): After setup() function exits (ends), the loop() function is executed repeatedly in the main program. It controls the board until the board is powered off or is reset. It is analogous to the control structure while(1).

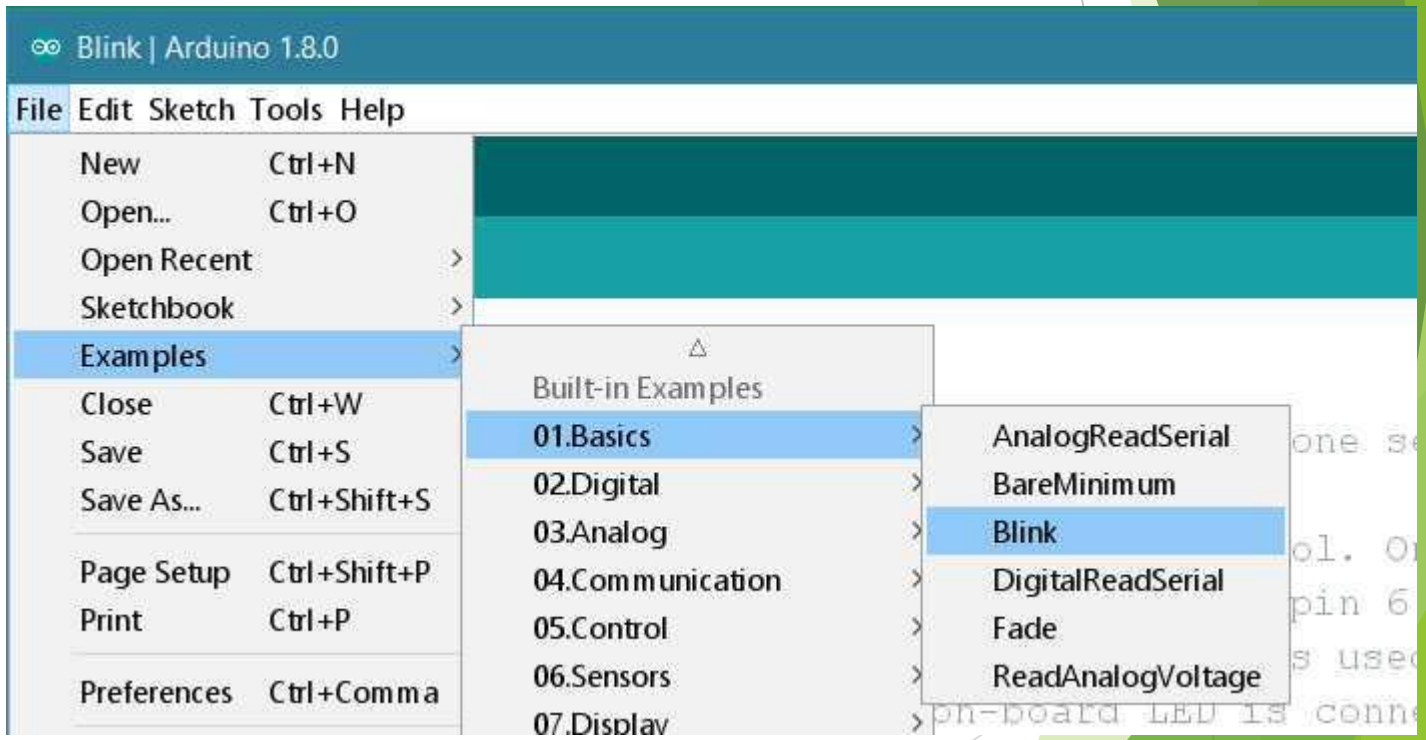
Arduino IDE 使用步骤
Steps to Use Arduino IDE

- 使用A-B型USB电缆连接电脑与Uno板；

  Connect your Uno board with PC using an A-B type USB cable;

- 鼓励以示例程序为蓝本进行修改，如闪烁程序；

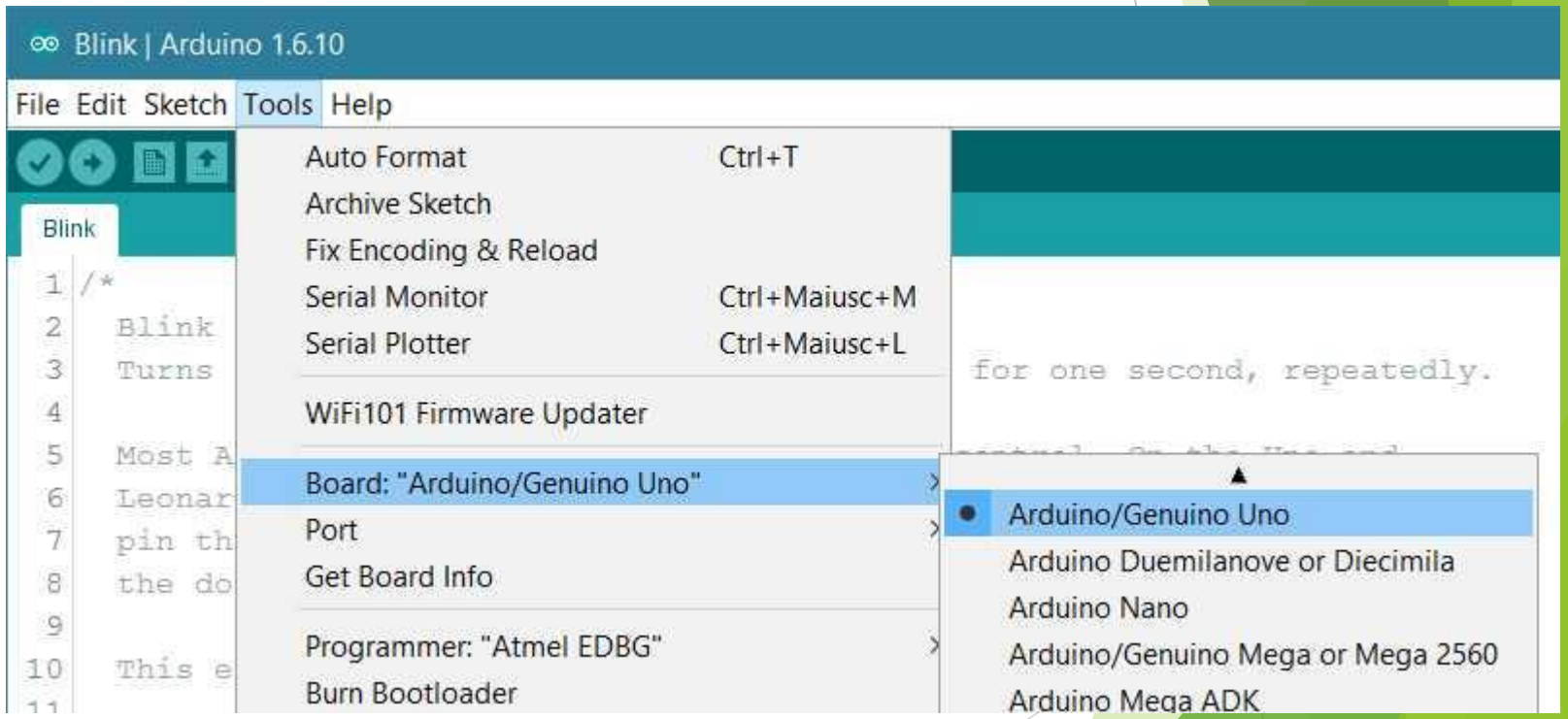
  It is encouraged to develop based on the example sketch, for example, the Blink example.



Arduino IDE 使用步骤
Steps to Use Arduino IDE

- 选择板卡类型和端口，通过工具 > 板菜单中选择与手头Arduino板相对应的条目；

  Select your board type and port. Select the entry via Tools > Board menu that corresponds to your Arduino board;



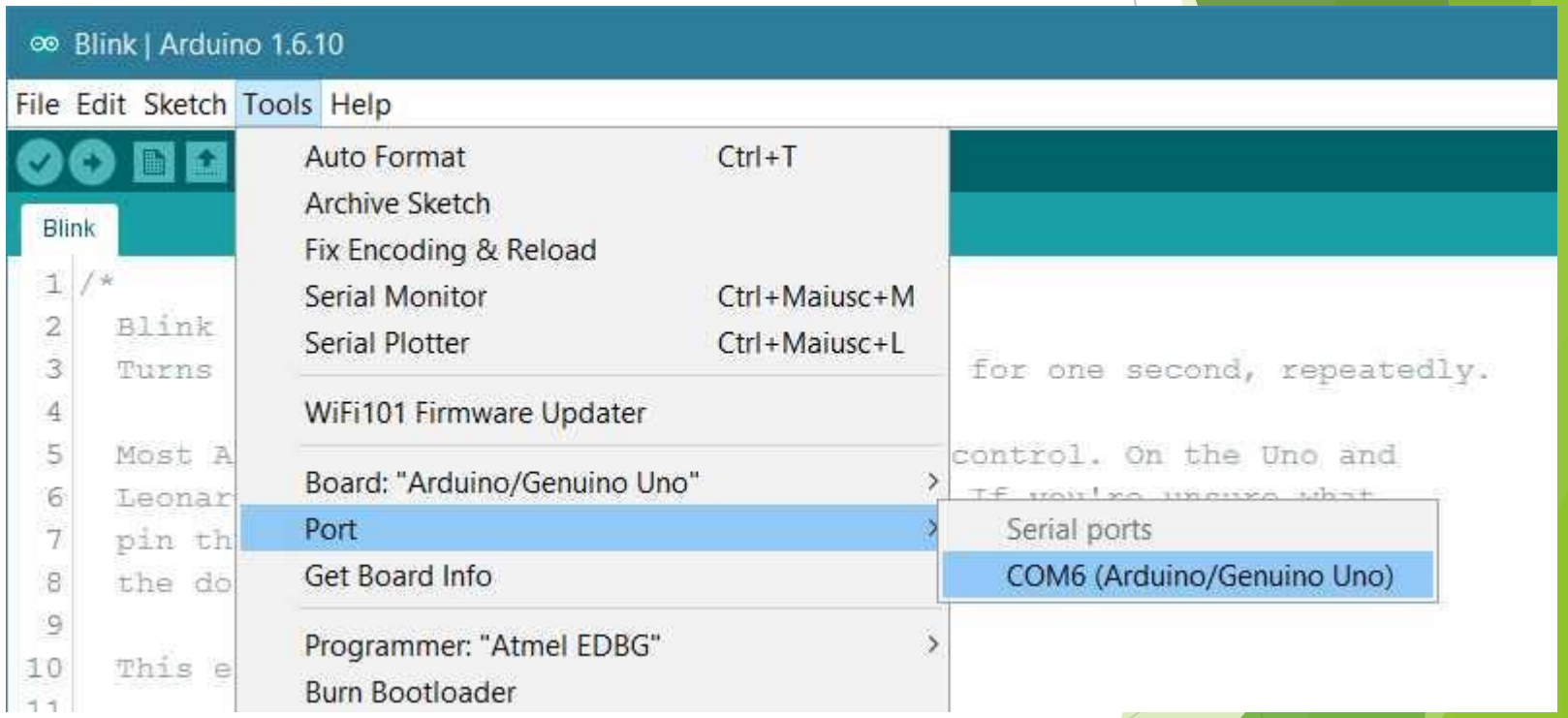
Arduino IDE 使用步骤
Steps to Use Arduino IDE

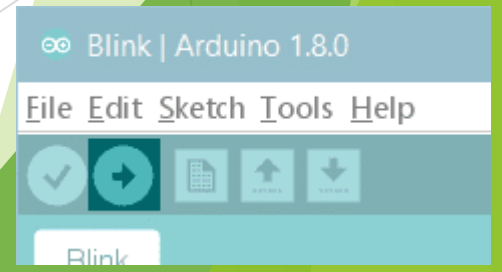
- 从工具|串行设备菜单中选择板的串行端口，通常是 COM3 或更高端口。

  Select the serial device of the board from the Tools | Serial Port menu. This is likely to be COM3 or higher.

- 编译上传程序。

  Compile and upload the program.

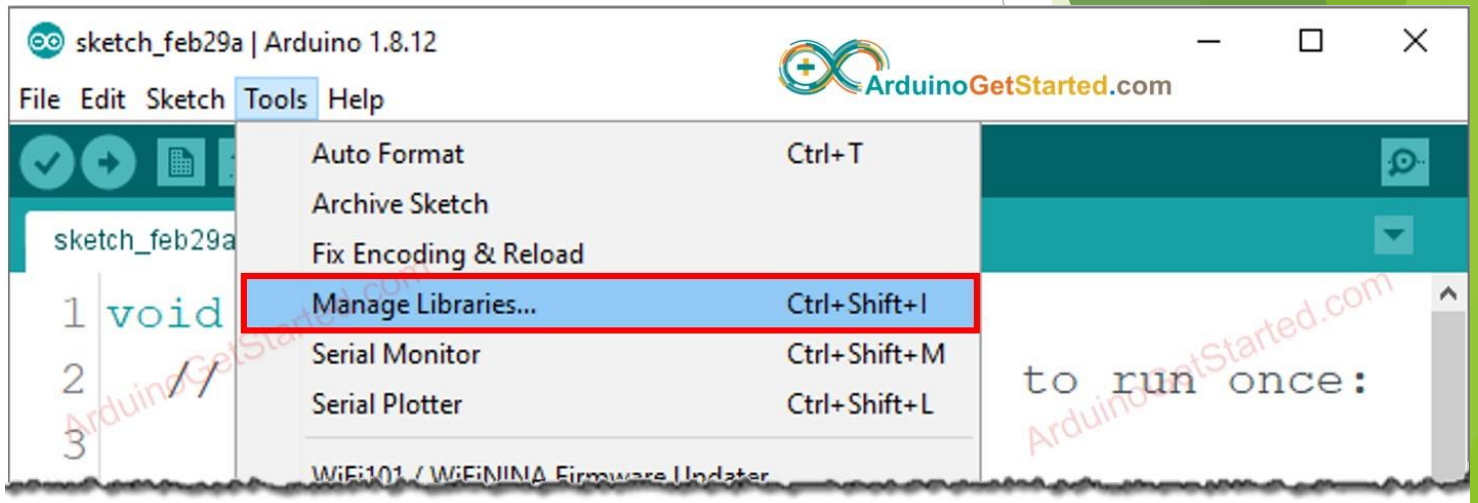


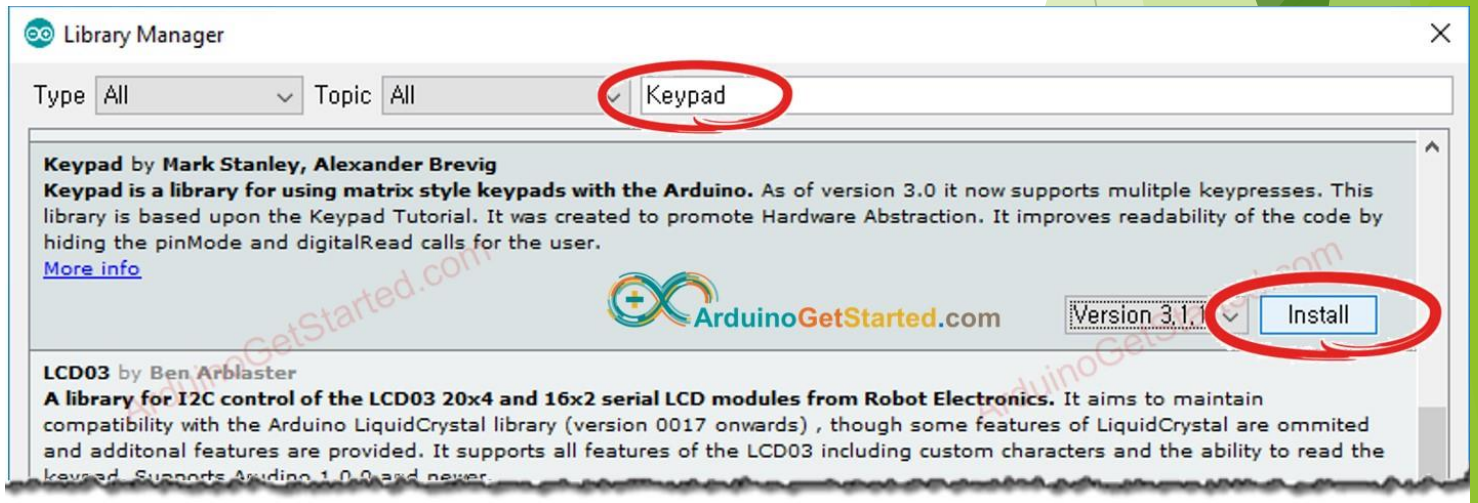


Arduino IDE 使用步骤
Steps to Use Arduino IDE

- 如果需要安装额外的库的话，可以通过菜单工具|管理库菜单进行，例如安装按键支持。

  Extra library on Arduino IDE can be installed via Tools | Manage Libraries menu, for example, the keypad library.

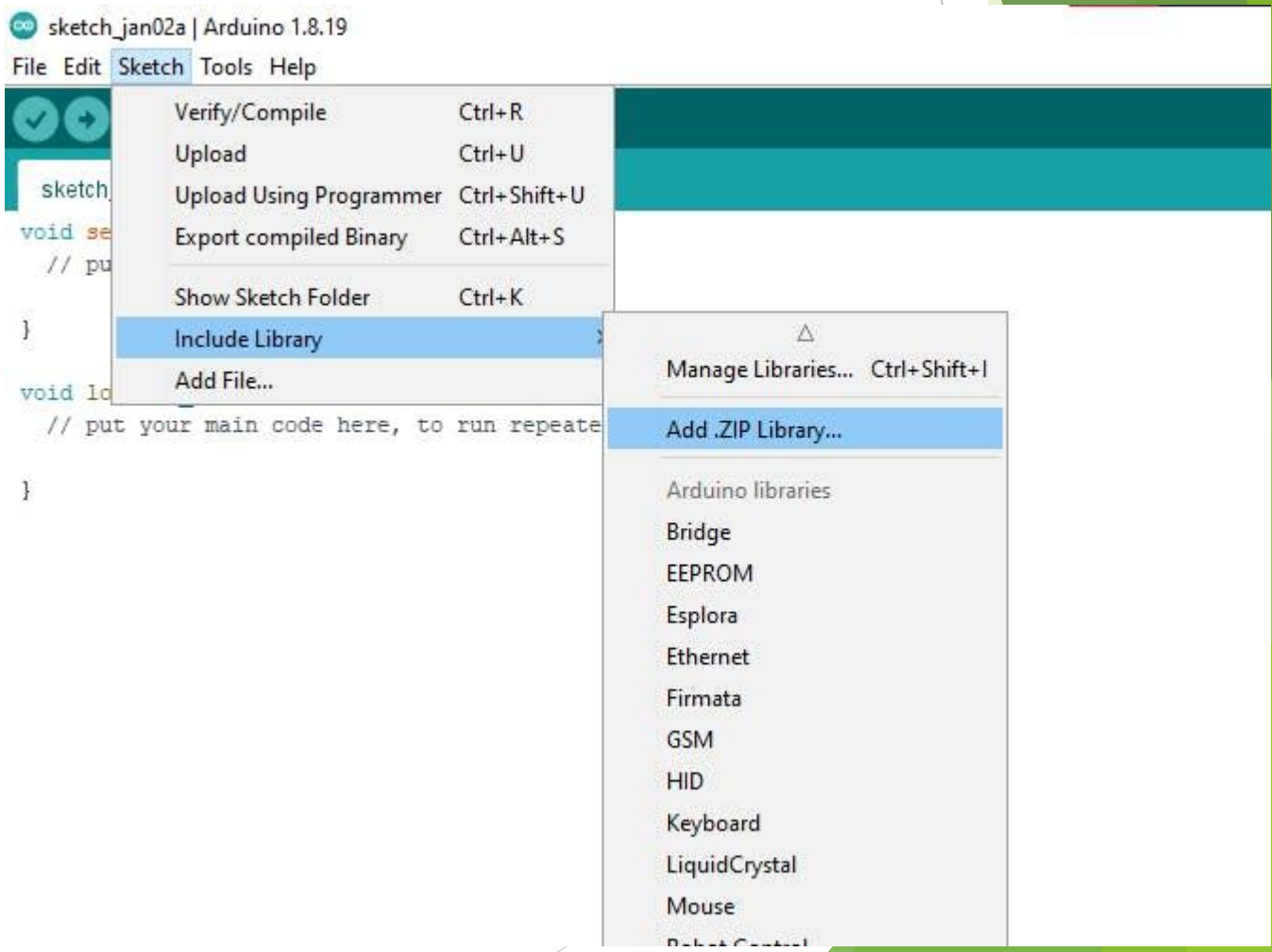




Arduino IDE 使用步骤
Steps to Use Arduino IDE

- 一些第三方库可能不是由库管理器直接管理的，则可以通过直接安装压缩文件安装库。如果需要安装额外的库的话，可以通过菜单项目|加载库>管理库进行。

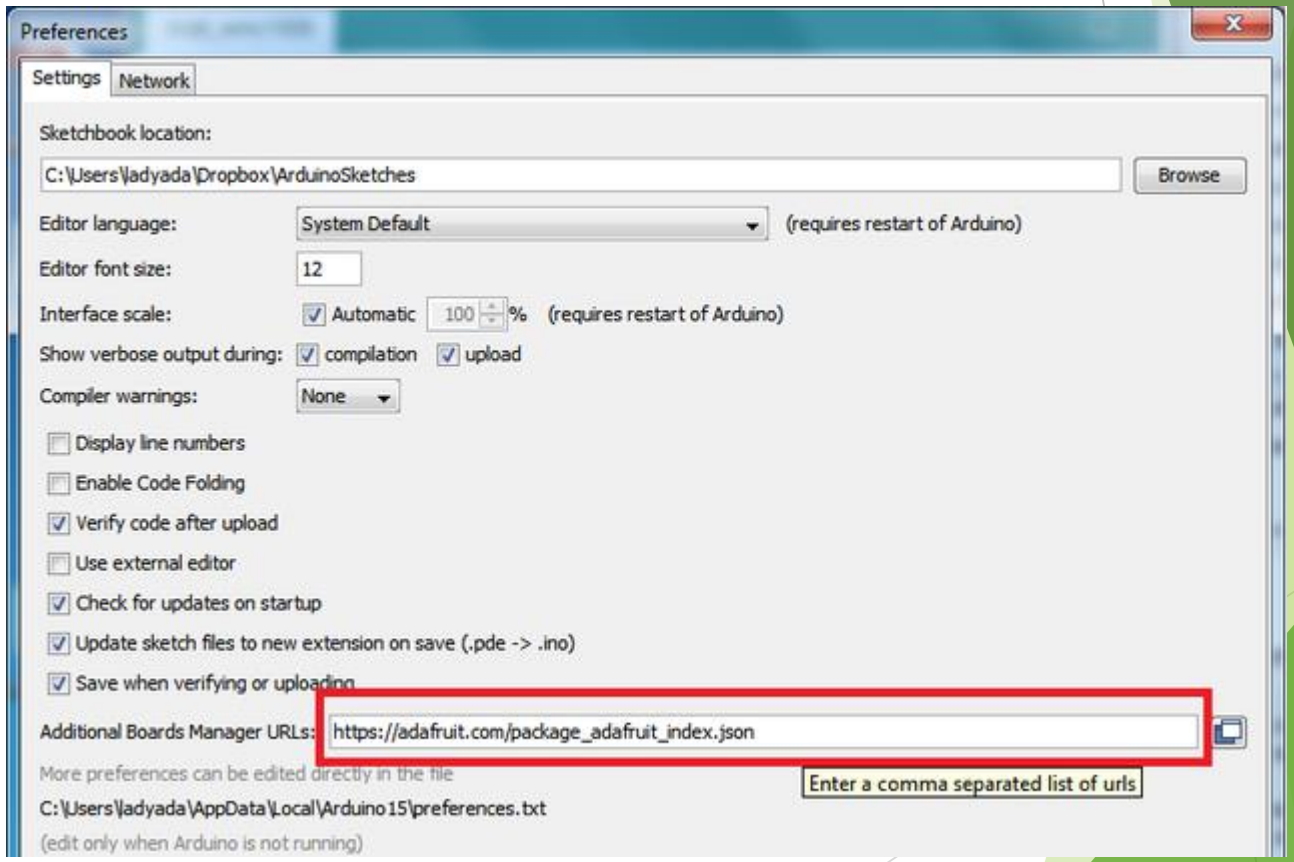
  Some third-party libraries may not be directly managed by the library manager, you can add the library by installing the zip file directly. If you need to install additional libraries, you can do it through menu Sketch | Include Library > Add .ZIP Library.



Arduino IDE 使用步骤
Steps to Use Arduino IDE

- 如果要安装一些第三方的开发板，如SparkFun RedBoard，则可以通过如下步骤：File > Preferences > Additional Boards Manager URLs，粘贴第三方链接。

  If third-party boards, like SparkFun RedBoard, need to be added to the Arduino IDE, one can configure the Board Manager via File > Preferences > Additional Boards Manager URLs, and paste the link.



Arduino IDE 使用步骤
Steps to Use Arduino IDE

- 仍以SparkFun RedBoard为例，单击工具 > 开发板 > 开发板管理器...从下拉菜单中选择 “贡献”类型。单击 SparkFun AVR 板，然后单击安装 。

  Exemplified by SparkFun RedBoard, click on Tools > Board > Boards Manager..., then select the Type as "Contributed" from the drop down menu. At last, click on the SparkFun AVR Boards and then click Install.