Arduino

www.smartprj.com

新车间网址

图形化编程软件 ArduBlock

Virtual breadboard

Proteus

```
int ledPin = 3;
int delayTime = 1000;

void setup()
{
    pinMode(ledPin, OUTPUT);
    for (int i=0; i<10; i++)
        Serial.println(i);
}

void loop()
{
    digitalWrite(ledPin, HIGH);
    delay(delayTime);
    digitalWrite(ledPin, LOW);
    delay(delayTime);
}
```

支持宏定义 `#define` 和关键词 `const`.

```
byte A = 8; // 字节型, 存储 0-255 的数字.
char B = char(A);
word C = word(B);

word(H, L); // H 为高阶字节, L 为低阶字节.
```

支持结构体 `struct` 和数组, 变量和运算符与 c 类似.

常用函数 (见 70 页).

PWM (Pulse Width Modulation): 脉冲宽度调制.

第 3 章 硬件

3.1 单片机简介

计算机经典结构: 运算器, 控制器, 存储器, 输入设备, 输出设备.

单片机

- I/O 接口电路
- CPU: 运算器, 控制器, 寄存器
- 存储器

3.2 Atmel AVR 单片机

3.2.1 Arduino 与 AVR

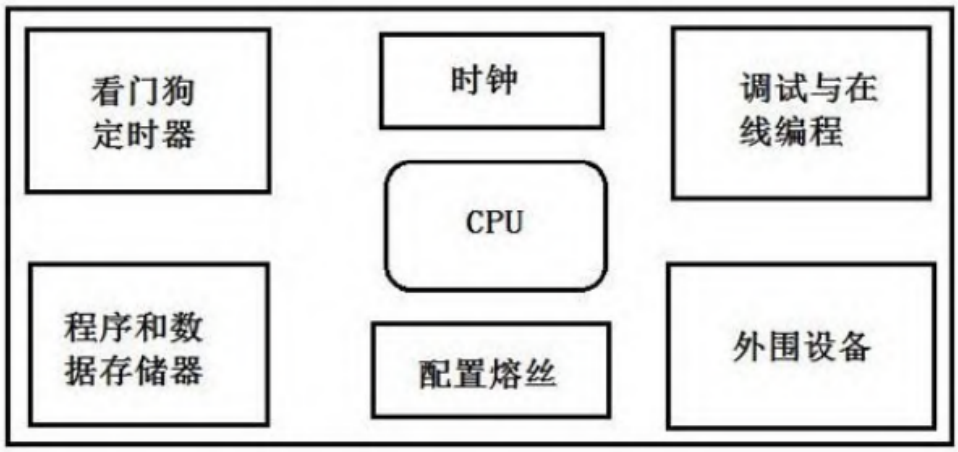

图3-2　AVR ATmega328功能部分

熔丝位状态包括 Unprogrammed (禁止), 表示 1, Programmed (允许), 表示 0.

电压范围 1.8-5.5 V.

3.2.2　芯片封装

3.2.3　管脚定义及指令系统

3.2.4　AVR内核

3.2.5　片内外围设备介绍

3.3 电子技术基础学习

3.3.1　电路图

3.3.2　电子元件

发光二极管: 长引脚为正极

3.3.3　基本工具介绍

- 万用表
- 电烙铁、焊锡和松香

- 万能板
- 剥线钳

第 4 章 Arduino 示例演练

4.1 制作 LCD 温度显示器

4.2 再谈 Arduino 语言

4.2.1 位操作

```
int x = 0x81;                //10000001
int n3_bit = bitRead(x, 7); //从右开始，从 0 开始的第 7 位 (这里是 1)
int change_bit(x, 1, 0);    //第 1 位变成 0
```

4.2.2 数学函数

函数	说明
max(x, y)	
min(x, y)	
abs(x)	
sq(x)	平方
pow(base, exponent)	次方
sqrt(x)	平方根
sin(rad)	正弦 (还有 cos, tan)
constrain(x, a, b)	约束范围: x>b 则返回 b
map(x, fromL, fromH, toL, toH)	改变范围
log(x)	自然对数

4.2.3 随机函数

```
randomSeed(seed);
int num = random(10);       // 0-9
int num = random(1, 10);    // 1-9
```

4.2.4 高级输入输出

```
tone(3, 400);       //引脚 3 输出 400 Hz 的声音
tone(3, 400, 32);   //持续 32，时间单位未知
noTone(3);          //引脚 3 声音停止
```

4.2.5 时间函数

```
delay(1000);                //延时 500 ms
delayMicroseconds(value);   //延时 value 微秒
```

记录程序从运行开始执行的时间

```
unsigned long currentMillis = millis();     //最长 9 h 22 min.
```

4.2.6 中断

- 外部中断: 如键盘中断, 打印机中断. 需要中断源的中断请求.
- 定时中断: 自动进行, 无需中断源的中断请求.

4.2.7 中断的使用

关中断函数与开中断函数

```
program;
Interrupt();
program;        //不可被中断的函数
noInterrupt();
program;
```

1 外部中断

外部中断函数

```
attachInterrupt(interrupt, function, mode);     // 中断使能函数
detachInterrupt(interrupt);                     // 中断禁止函数
```

- interrupt: 中断号, UNO 只能使用 0 或 1, 即代表 D2 与 D3 口.

  可以使用函数 `digitalPinToInterrupt(pin)`, 即输入 2 返回 0, 输入 3 返回 3.
- Function: 中断发生时调用的函数 (不能有返回值)
- Mode: 中断触发模式
  - LOW: 当针脚输入为低时, 触发中断.
  - CHANGE: 当针脚输入发生改变时, 触发中断.
  - RISING: 当针脚输入由低变高时, 触发中断.
  - FALLING: 当针脚输入由高变低时, 触发中断.

---

2 定时中断

常用的库: FlexiTimer2.h, MsTimer2.h

1. 设置定时中断函数

```
void set(unsigned long ms, void (*f)());
```

2. 开启定时中断函数和关闭定时中断函数

```
program;
MsTimer2::set(500, blink);  // 每 500 ms 执行一次 blink 函数
MsTimer2::start();
program;             // 需要执行定时中断的代码段
MsTimer2::stop();
program;
```

e.g.

```
#include <MsTimer2.h>   // 定时器库的头文件
int ledPin = 13;
volatile int state = LOW;

void setup(){
    pinMode(ledPin, OUTPUT);
    //attachInterrupt(0, blink, CHANGE);    // 设置触发类型为 CAHNGE
    MsTimer2::set(500, blink);  // 设置中断函数, 每 500ms 进入一次
    MsTimer2::start();          // 开始计时
}

void loop(){
    digitalWrite(ledPin, state);
}

void blink(){
    state = !state;
} // 中断服务程序
```

4.3 实例

火焰报警器.

蜂鸣器: 压电式蜂鸣器, 电磁式蜂鸣器

4.4 Arduino 与传感器的互动

1. 厨房的危险报警系统.
2. 远程控制的系统.

4.5 用 Arduino 驱动电机

4.5.1 电机简介

电机分类

- 直流电机, 交流电机.
- 直流电机, 同步电机, 异步电机.
- 驱动用电机, 控制用电动机.

4.5.2 Arduino 与直流电机

直流电机不能直接接到 Arduino (数字引脚的最大输出电流为 40 mA).

需要一个电源额外给直流电机供电, 并使用三极管作为开关控制电机. 为了避免反向电压的危害, 还需要用到二极管.

4.5.3 Arduino 与步进电机

将电脉冲转化为角位移.

4.5.4 Arduino 与舵机

棕色为接地线 (GND), 红色为电源正极线 (VCC), 橙色为信号线 (PWM).

舵机的转动角度是通过调节 PWM (脉冲宽度调制) 信号的占空比来实现的.

Arduino 控制舵机的方法:

1. 通过 Arduino 的普通数字传感器接口产生占空比不同的方波, 模拟 PWM 信号.
2. 使用 Servo 库的函数进行控制.

   这样只能控制 2 路舵机, 因为 Arduino 自带函数只能利用数字 9, 10 接口.

由于 Arduino 的驱动能力有限, 控制多个舵机时需要外接电源.

4.6 Arduino 访问网络

与 W5100 或 ENC28J60 网络扩展板配合即可连接网络.

EtherCard 库文件

4.6.1 Arduino 连接网络

1 ether.begin()

功能: 连接网络

```
ether.begin(sizeof Ethernet::buffer, mymac [, 10]);
```

- buffer 为缓冲大小
- mymac 为 MAC 地址
- 可选第 10 引脚

连接成功则返回 1, 否则返回 0

---

2 ether.dhcpSetup()

功能: 寻找服务器获取地址

```
ether.dhcpSetup();
```

成功则返回 1, 若 30 s 仍未获得 IP 地址, 则返回 0.

---

3 ether.staticSetup()

功能: 配置静态 IP 地址

```
ether.staticSetup(myIP [, gwIP, dnsIP]);
```

- myIP 表示要设定的 IP 地址.
- gwIP 代表网关, dnsIP 代表 DNS.

4 ether.printIp()

功能: 在串口上打印 IP 地址

```
ether.printIp("My IP is:", ether.myip);
ether.printIp(ip);
```

5 ether.packetReceive()

功能: 从网络接受一个新传入的数据表.

返回值: 接收到的数据包大小.

6 ether.packetLoop()

功能: 对接收到的信息作出回应, 包含 ping 请求.

```
ether.packetLoop(len);
```

- len 表示接收到的数据包的大小, 类型为 word 性.

返回值: 数据在缓冲区中的偏移量.

4.6.2 Arduino 与 Yeelink

Yeelink 是一个网络服务平台.

4.6.3 Arduino 和 Web 服务器通信

4.7 Arduino 与无线通信

第 5 章 Arduino 项目训练

5.1 项目 1 智能家居

5.2 项目 2 遥控小车

5.3 项目 3 机械手臂

5.4 项目 4 贪吃蛇

第 6 章 媒体互动制作

6.1 Processing

Arduino Examples

01 Basics

```
Serial.begin(9600);     //主串口
Serial1.begin(9600);    //其它串口
Serial.end();           //禁止串口传输，无参数

Serial.println("Hello");        //有换行
Serial.print("World");           //无换行
delay(1000);

pinMode(LED_BUILTIN, OUTPUT);
digitalWrite(LED_BULTIN, HIGH);     // LOW 为 0 V
analogWrite(pin, value);            // value 为 0~255
analogReference(type);

int buttonState = digitalRead(button);      //数字信号
int sensorValue = analogRead(A0);           //模拟信号，读一次需要 1 微秒
```

02 Digital

```
unsigned long currentMillis = millis();     //已运行的时间
//为了避免按一次按钮响应两次（噪音），可以使用这个函数判断两次信号的时间间隔

pinMode(2, INPUT_PULLUP);       //读入数据后输出到串口监视器上
//按钮被按下是 LOW，对应的输出电压应该为 HIGH

#include "pitches.h"            //自己编写的含音符宏定义的头文件
tone(pin, frequency, duration); //无需包含头文件 (八分音符)
noTone(8);                      //无需包含头文件. Arduino 一次只能产生一个声音.

f = map(x, a1, b1, a2, b2);
```

03 Analog

```
f = constrain(x, min, max);

analogReference(type);      // 配置参考电压，type 取值如下：
// DEFAULT（默认 5 V），或 INTERNAL（低功耗模式，1.1），或 EXTERNAL（扩展模式，0~5）
```

04 Communication

```
while(!Serial);         // 等待串口连接

Serial.write(ch);       // 输出 ASCII 值
Serial.print(ch);       // 输出十进制数
Serial.print(ch, HEX);  // 输出十六进制数，还可以是 OCT, BIN, DEC

if (Serial.available()){
    brightness = Serial.read();
    analogWrite(ledPin, brightness);
}// 从串口监视器输入数据
```

每次有新数据时，串口事件的函数会被调用

```
void serialEvent(){
    while (Serial.available()){
        Serial.println(Serial.read());
    }
}
```

05 Control

06 Sensors

- knoch sensor (piezo)

```
const int knockSensor = A0;
const int threshold = 100;
int ledState = LOW;
// in the loop()
if (sensorReading >= threshold){
    ledState = !ledState;
    digitalWrite(ledPin, led State);    // knock
}
delay(100); // to avoid overloading the serial port buffer
```

- accelerator

```
// in the loop()
int pulseX = pulseIn(xPin, HIGH);   // read pulse from x-axis
int pulseY = pulsein(yPin, HIGH);
int accelerationX = (pulseX/10 - 500) * 8;  // the gravity is g
int accelerationY = (pulseY/10 - 500) * 8;
```

- ping sensor (超声波测距仪)

```
void loop(){
    pinMode(pingPin, OUTPUT);
    digitalWrite(pingPin, LOW);
    delayMicroseconds(2);       // 微秒
    digitalWrite(pingPin, HIGH);
```

```
    delayMicroseconds(5);
    digitalWrite(pingPin, LOW);

    pinMode(pingPin, INPUT);
    long duration = pulseIn(pingPin, HIGH);
    long distance = microsecondsToCentimeters(duration);
    // 省去输出的部分
    delay(100);
}

long microsecondsToCentimeters(long microseconds){
    return microseconds / 29 / 2;
}// 340 m/s or 29 cm/s
```

07 Display

- bar graph

```
void loop(){
    int sensorReading = analogRead(analogPin);
    int ledLevel = map(sensorReading, 0, 1023, 0, ledCount);
    for (int thisLed = 0; thisLed < ledCount; thieLed++){
        if (thisLed < ledLevel){
            digitalWrite(ledPins[thisLed], HIGH);
        }
        else{
            digitalWrite(ledPins[thisLed], LOW);
        }
    }
}
```

- row-column scanning an 8×8 LED matrix with X-Y input

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};
const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};
int pixels[8][8];

void setup(){
    for (int thisPIn = 0; thisPin < 8; thisPin++){
        pinMode(col[thisPin], OUTPUT);
        pinMode(row[thisPin], OUTPUT);
        digitalWrite(col[thisPin], HIGH);   // turn off the LEDS
    }
    for (int x=0; x<8; x++){
        for (int y=0; y<8; y++){
            pixels[x][y] = HIGH;
        }
    }
}
```

```
void loop(){
    readSensors();      // read input
    refreshScreen();    // draw the screen
}

void readSensors(){
    pixels[x][y] = HIGH;    // turn off the last position
    x = 7 - map(analogRead(A0), 0, 1023, 0, 7);
    y = map(analogRead(A1), 0, 1023, 0, 7);
    pixels[x][y] = LOW;     // screen refresh
}

void refreshScreen(){
    for (int thisRow = 0; thisRow < 8; thisRow++){
        digitalWrite(row[thisRow], HIGH);
        for (int thisCol = 0; thisCol < 8; thisCol++){
            int thisPixel = pixels[thisRow][thisCol];
            digitalWrite(col[thisCol], thisPixel);
            if (thisPixel == LOW){
                digitalWrite(col[thisCol], HIGH);
            }
            digitalWrite(row[thisRow], LOW);
        }
    }
}
```

08 Strings

```
// 字符操作
isAscii(ch);            // ASCII 字符
ifDigit(ch);            // 数字
isAlpha(ch);            // 字母
isAlphaNumeric(ch);     // 字母或数字
isLowerCase(ch);        // 小写字母
isUpperCase(ch);        // 大写字母
isWhtespace(ch);        // 空格
isControl(ch);          // 控制字符
isPrintable(ch);        // 可打印字符
isGraph(ch);            // 非空格可打印字符
isPunct(ch);            // 符号
isSpace(ch);            // 空字符，包括 \0，空格等
isHexadecimalDigit(ch); // 十六进制数

// 创建字符串
String str = "Hello";
String str = String('a');
String str = String("Hello");
String str = String(str2 + " world");
String str = String(32, DEC);       // 可以没有 DEC
String str = String(3.1415, 2);     // 四舍五入

// 连接字符串
String str = String("Num: ") + 123 + "ABC" + 'a' + str2[2];
str.concat(str2);
str.concat(millis());
```

```
// 改变字符串
str.trim();          // 去除首尾空字符
str.toUpperCase();
str.toLowerCase();
str.setCharAt(n, ch);

str.reserve();          // 预留内存
str.remove(index);
str.remove(index, count);
str.replace(substr1, substr2);

// 比较
if (str1 == str2);                  // 或 !=, <, > 等
if (str1.equals(str2));             // ==
if (str1.equalsIgnoreCase(str2));   // 忽略大小写 ==
if (str1.compareTo(str2));          // 比较 >=<
if (str1.endWith(str2));            // 结尾字符串
if (str1.startsWith(str2));         // 开始字符串

int len = str.length();     // 长度
int n = str.toInt();        // 转为 int
float a = str.toFloat();    // 转为 float
double a = str.toDouble();  // 转为 double
char ch = str.charAt(n);    // 获取 char
char *s = str.c_str();      // 转为 char*

str.toCharArray(buf);
str.toCharArray(buf, len);  // 复制到 buf
String substr = str.substring(from);
String substr = str.substring(from, to);    // [from, to)

int index = str.indexOf(val);       // val 为 char 或 String
int index = str.indexOf(val, from); // 从第 from 个字符开始
int index = str.lastIndexOf(val);   // 没有则返回 -1
int index = str.lastIndexOf(val, from);
```

09 USB

这一章示例不能使用 UNO 板子.

- keyboard logout

```
#include "Keyboard.h"
#define OSX 0
#define WINDOWS 1
#define UBUNTU 2
int platform = WINDOWS;

void setup(){
    pinMode(2, INPUT_PULLUP);   // make pin 2 an input and turn on the pull-up
resistor so it goes high unless connected to gound.
    Keyboard.begin();
}
```

```
void loop(){
    while (digitalRead(2) == HIGH){
        delay(500);     // do nothing until pin 2 goes low
    }
    delay(1000);

    switch (platform) {
        case OSX:
            Keyboard.press(KEY_LEFT_GUI);
            // Shift + Q, logs out:
            Keyboard.press(KEY_LEFT_SHIFT);
            Keyboard.press('Q');
            delay(100);
            Keyboard.releaseALL();
            // enter:
            Keyboard.write(KEY_RETURN);
            break;
        case WINDOWS:
            // Ctrl + ALt + Del:
            Keyboard.press(KEY_LEFT_CTRL);
            Keyboard.press(KEY_LEFT_ALT);
            Keyboard.press(KEY_DELETE);
            delay(100);
            Keyboard.releaseAll();
            // Alt + l:
            delay(2000);
            Keyboard.press(KEY_LEFT_ALT);
            Keyboard.press('l');
            Keyboard.releaseAll();
            break;
        case UBUNTU:
            // Ctrl + Alt + Del:
            Keyboard.press(KEY_LEFT_CTRL);
            Keyboard.press(KEY_LEFT_ALT);
            Keyboard.press(KEY_DELETE);
            delay(1000);
            Keyboard.releaseAll();
            // Enter to confirm logout:
            Keyboard.write(KEY_RETURN);
            break;
    }

    while (true);   // do nothing
}
```

- keyboard message

```
#include "Keyboard.h"
const int buttonPin = 4;
int previousButtonState = HIGH;
int counter = 0;

void setup(){
    pinMode(buttonPin, INPUT);
    Keyboard.begin();
```

```
}

void loop(){
    int buttonState = digitalRead(buttonPin);
    if (buttonState != previousButtonState &&
       buttonState == HIGH) {
        ++counter;
        Keyboard.print("You pressed the button ");
        Keyboard.print(counter);
        Keyboard.println(" times.");
    }
    previousbuttonState = buttonState;
}
```

- keyboard reprogram

```
#include "Keyboard.h"
// char ctrlKey = KEY_LEFT_GUI; // for OSX
char ctrlKey = KEY_LEFT_GUI;    // for Windows and Linux

void setup(){
    pinMode(2, INPUT_PULLUP);
    Keyboard.begin();
}

void loop(){
    whie (digitalRead(2) == HIGH) {
        delay(500);     // do nothing until pin 2 goes low
    }
    delay(1000);

    // new document:
    Keyboard.press(ctrlKey);
    Keyboard.press('n');
    delay(100);
    Keyboard.releaseAll();
    delay(1000);

    // select all
    Keyboard.press(ctrlKey);
    Keyboard.press('a');
    delay(500);
    Keyboard.releaseAll();
    Keyboard.write(KEY_BACKSPACE);
    delay(500);

    Keyboard.println("Hello world");

    while (true);
}
```

- keyboard serial

```
#include "Keyboard.h"

void setup(){
    Serial.begin(9600);
    Keyboard.begin();
}

void loop(){
    if (Serial.available() > 0){
        char inChar = Serial.read();
        Keyboard.write(inChar + 1);
    }
}
```

- Keyboard and mouse control
- Button mouse control
- Joy stick mouse control

UNO 1 EEPROM

```
#include <EEPROM.h>
EEPROM.length();        // 不同板子有不同的 EEPROM 内存, 我的是 1024
EEPROM.write(i, 0);     // 置为零
```

UNO 2 Software

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11);    // RX, TX
// RX is digital pin 10 (connected to TX of other device)
// TX is digital pin 11 (connected to RX of other device)
mySerial.begin(4800);
mySerial.println("Hello");
if (mySerial.available()){
    Serial.write(mySerial.read());
}
if (Serial.available()){
    mySerial.write(Seiral.read());
}

SoftwareSerial portOne(8, 9);
portOne.listen();
```

FastLED

```
#include <FastLED.h>
```

Arduino References

Serial 的函数

```
if (Serial);        // If the specified Serial port is ready
while (!Serial);    // wait for Serial port to connect, needed for native USB

int num = Seiral.available();   // Get the number of characteres available for
reading from the Serial port.
int num = Serial.availableForWrite();

Serial.begin(9600);
Serial.end();

Serial.find(ch);            // reads data until the target is found
Serial.find(ch, length);    // all of them return bool value
Serial.findUntil(target, termial);

int num = Serial.parseInt();    // Looks for the next valid integer

Serial.peek();
Serial.flush();
```

教材的速查表

```
// 时间函数
unsigned long millis(void);     // 毫秒, 50 天时溢出, 从零开始
unsigned long micros(void);     // 微秒, 70 min 后溢出
void delay(unsigned long ms);
void delayMicroseconds(unsigned int us);

// 数字 I/O
void pinMode(uint8_t pin, uint8_t mode);
void digitalWrite(uint8_t pin, uint8_t value);
int digitalRead(uint8_t pin);

// 模拟 I/O
void analogReference(uint8_t type);
int analogRead(uint8_t pin);
void analogWrite(uint8_t pin, int value);

// 高级 I/O
void tone(uint8_t pin, unsigned int frequency, unsigned long
```

```
duration);
void notone(uint8_t pin);

// 数学函数
#define min(a, b) ((a)<(b)?(a):(b))
#define max(a, b) ((a)>(b)?(a):(b))
abs(x);
#define constrain(amt, low, high) ((amt)<(low)?(low):((amt)>(high)?(high):
(amt)))
long map(long value, long fromLow, long fromHigh, long toLow, long toHigh);
double pow(float base, float exponent);
double sqrt(double x);

// 三角函数
float sin(float rad);
float cos(float rad);
float tan(float rad);

// 随机数函数
void randomSeed(unsigned int seed);
long random(long sup);
long random(long inf, long sup);

// 位操作函数
#define lowByte(w) ((w) & 0xff)
#define highByte(w) ((w) >> 8)
#define bitRead(value, bit) ((value) >> (bit) & -x01)
#define bitWrite(value, bit, bitvalue) (bitvalue ? bitSet(value, bit) :
bitClear(value, bit))
#define bitSet(value, bit) ((value) |= (1UL << (bit)))
#define bitClear(value, bit) ((value) &= ~(1UL << (bit)))
#define bit(b) (1 << (b))

// 中断函数
void attachInterrupt(uint8_t interruptNum, void(*)(void)userFunc, int mode);
// interruptNum 是中断类型 (0 或 1)
// mode 是触发模式 (LOW, CHANGE, RISING, FALLING)
void detachInterrupt(uint8_t interruptNum);
#define interrupts() sei()
// 重新启用中断 (使用 noInterrupts 命令后将被禁用)
#define noInterrupts cli()
```

设置

编译文件默认文件夹.

串口调试助手输出换行符：

工具 - 选项 - 输出 - line feed，设置完每次发送都会自带换行符