SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

南方科技大学

0 - STM32 Introduction

System Design and Intelligent Manufacture
2019 Spring

0 **Preparation**

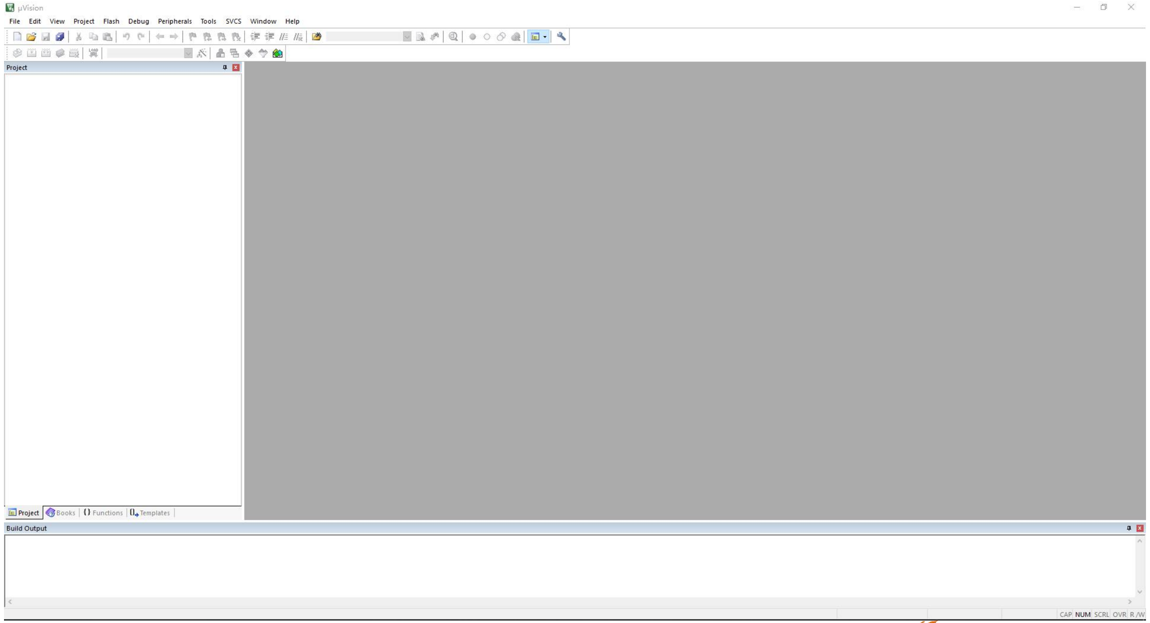
1 **Hardware Introduction**

2 **GPIO: Theory and Practice**

3 **Assignment**

◆What is “MDK”？

RealView MDK is developmented by Keil. It can provide an environment for processor based on Cortex, ARM7, ARM9.


μVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project
Project Books Functions Templates
Build Output
CAP NUM SCRL OVR R/W

◆Tips：

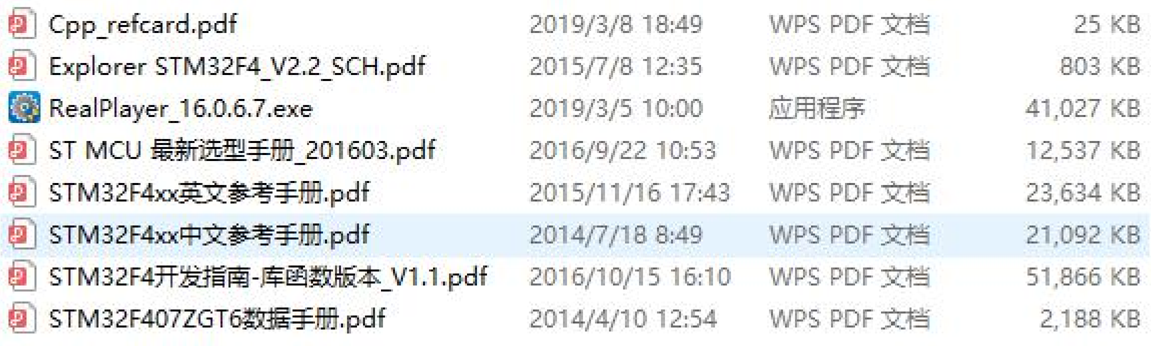
1）Installation path should be in English.

2) System user name should be in English.

3）Don't install multiple MDK(Keil) in the same path.

4）Don't forget to load package of chip.

Handbooks and Resources



Cpp_refcard.pdf	2019/3/8 18:49	WPS PDF 文档	25 KB
Explorer STM32F4_V2.2_SCH.pdf	2015/7/8 12:35	WPS PDF 文档	803 KB
RealPlayer_16.0.6.7.exe	2019/3/5 10:00	应用程序	41,027 KB
ST MCU 最新选型手册_201603.pdf	2016/9/22 10:53	WPS PDF 文档	12,537 KB
STM32F4xx英文参考手册.pdf	2015/11/16 17:43	WPS PDF 文档	23,634 KB
STM32F4xx中文参考手册.pdf	2014/7/18 8:49	WPS PDF 文档	21,092 KB
STM32F4开发指南-库函数版本_V1.1.pdf	2016/10/15 16:10	WPS PDF 文档	51,866 KB
STM32F407ZGT6数据手册.pdf	2014/4/10 12:54	WPS PDF 文档	2,188 KB

BBS and Pages

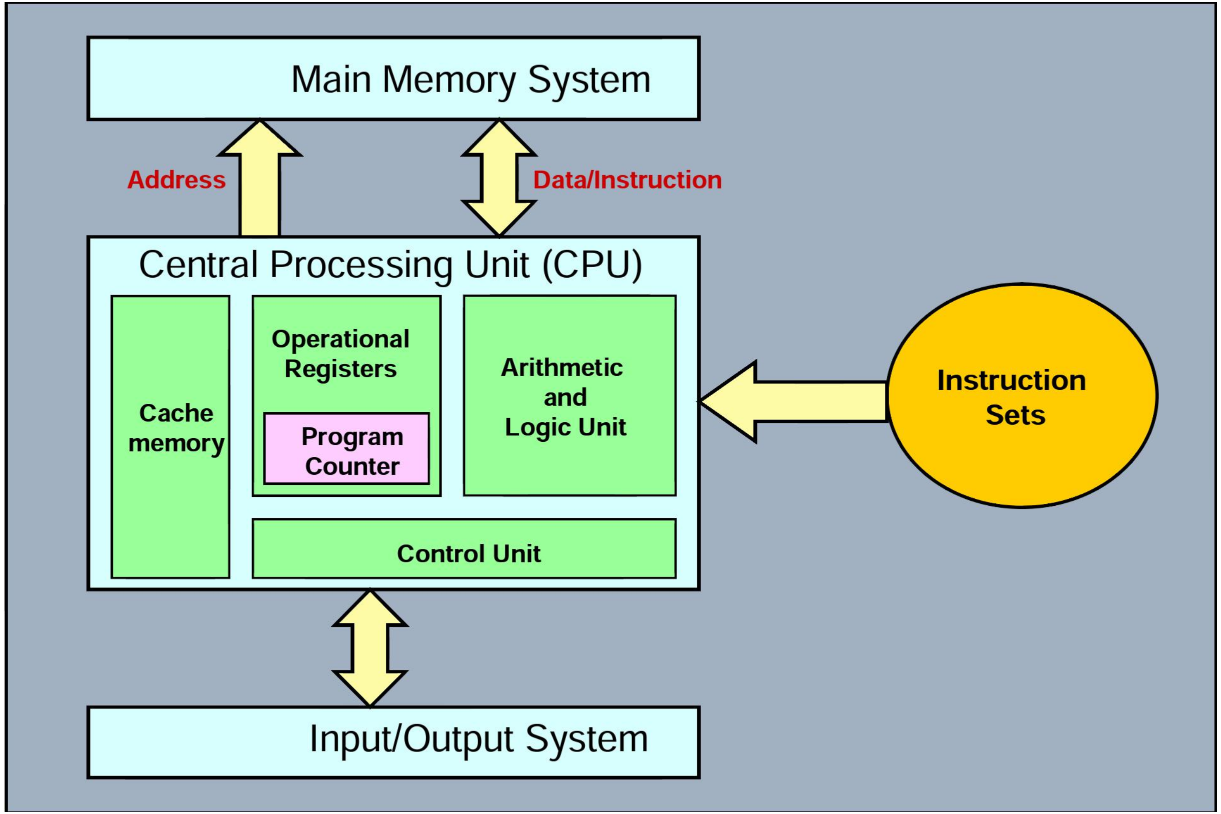
https://www.stmcu.com.cn/

http://www.stmcu.org.cn/module/forum/forum.php

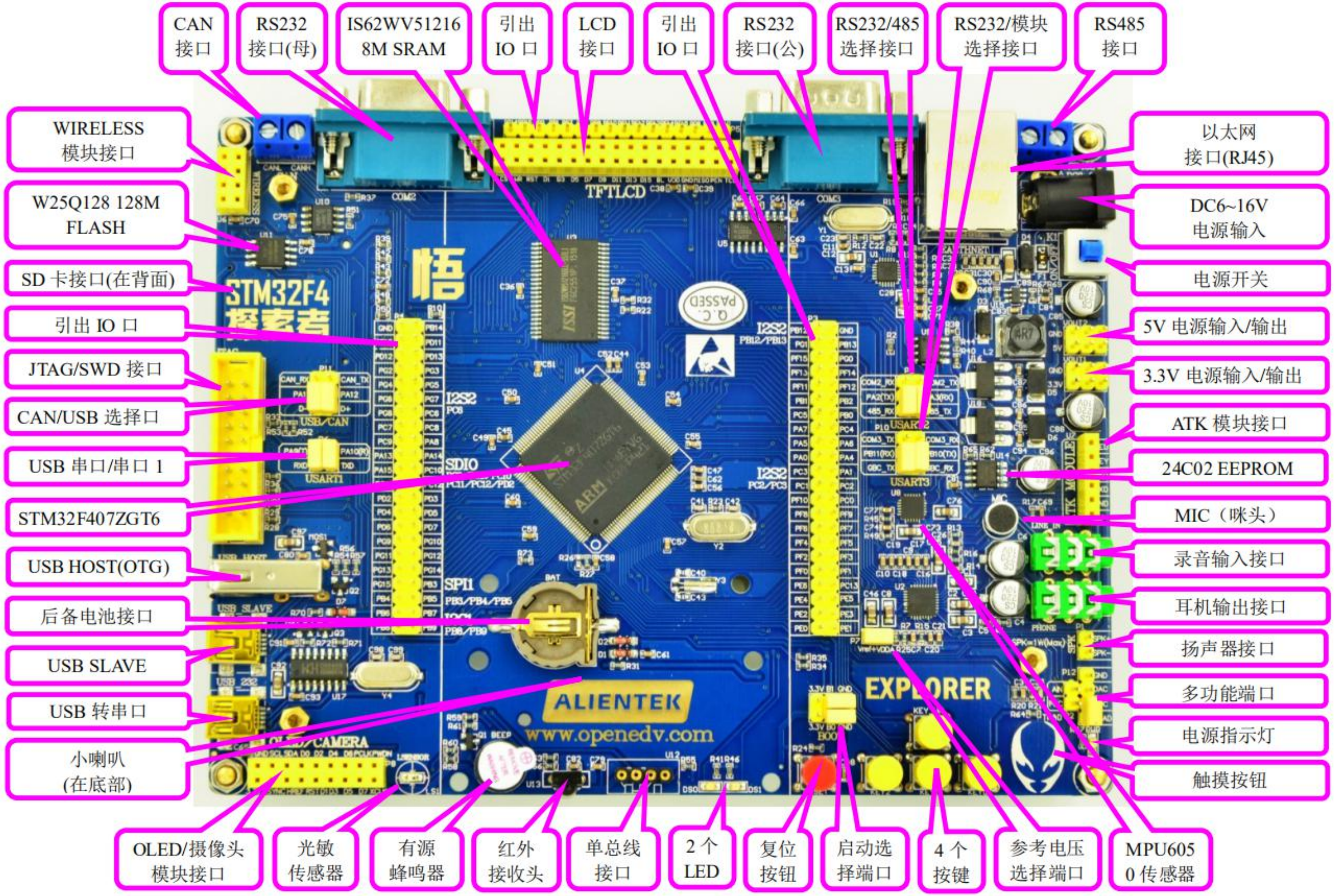
http://bbs.21ic.com/icfilter-typeid-35-226.html

http://firebbs.cn/forum.php

Hardware Introduction



	Knowledge
CSE	Computer Architecture, Embedded Programming, C++ Programming, Operating System, Fundamentals of Compiling
EEE	Fundementals of Circuits, Artificial Circuits, Digital Circuits
ME	/



CAN 接口

RS232 接口(母)

IS62WV51216 8M SRAM

引出 IO 口

LCD 接口

引出 IO 口

RS232 接口(公)

RS232/485 选择接口

RS232/模块 选择接口

RS485 接口

WIRELESS 模块接口

W25Q128 128M FLASH

SD 卡接口(在背面)

引出 IO 口

JTAG/SWD 接口

CAN/USB 选择口

USB 串口/串口 1

STM32F407ZGT6

USB HOST(OTG)

后备电池接口

USB SLAVE

USB 转串口

小喇叭 (在底部)

以太网 接口(RJ45)

DC6~16V 电源输入

电源开关

5V 电源输入/输出

3.3V 电源输入/输出

ATK 模块接口

24C02 EEPROM

MIC（咪头）

录音输入接口

耳机输出接口

扬声器接口

多功能端口

电源指示灯

触摸按钮

OLED/摄像头 模块接口

光敏 传感器

有源 蜂鸣器

红外 接收头

单总线 接口

2 个 LED

复位 按钮

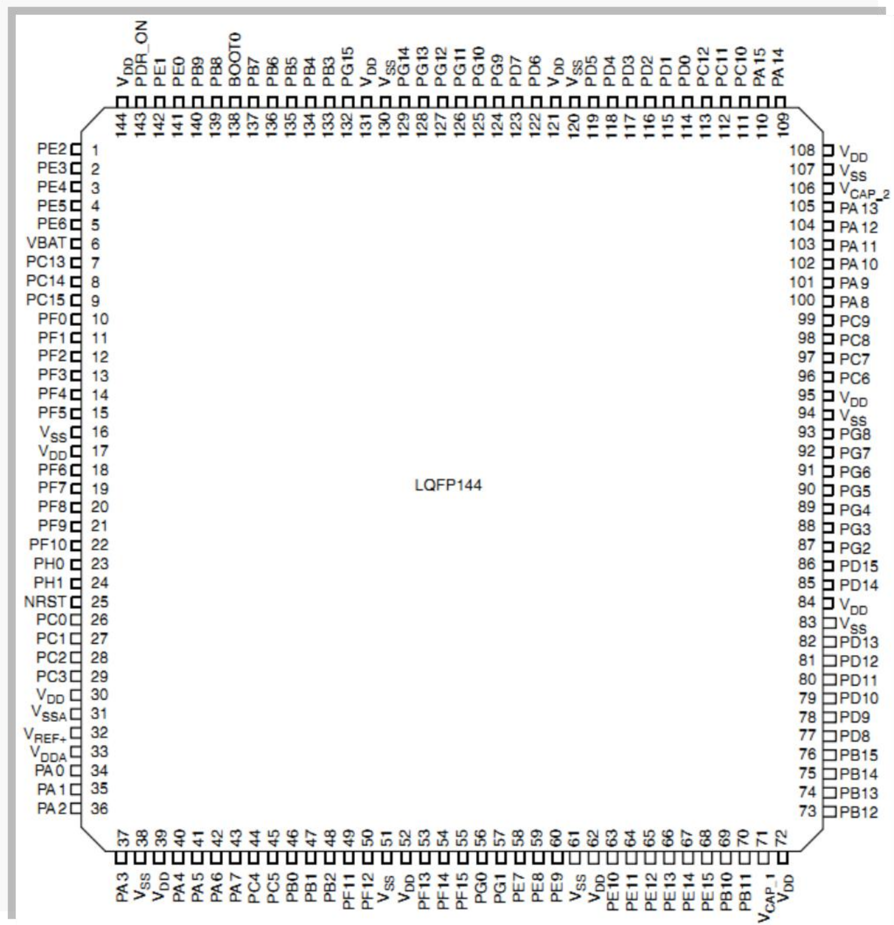
启动选 择端口

4 个 按键

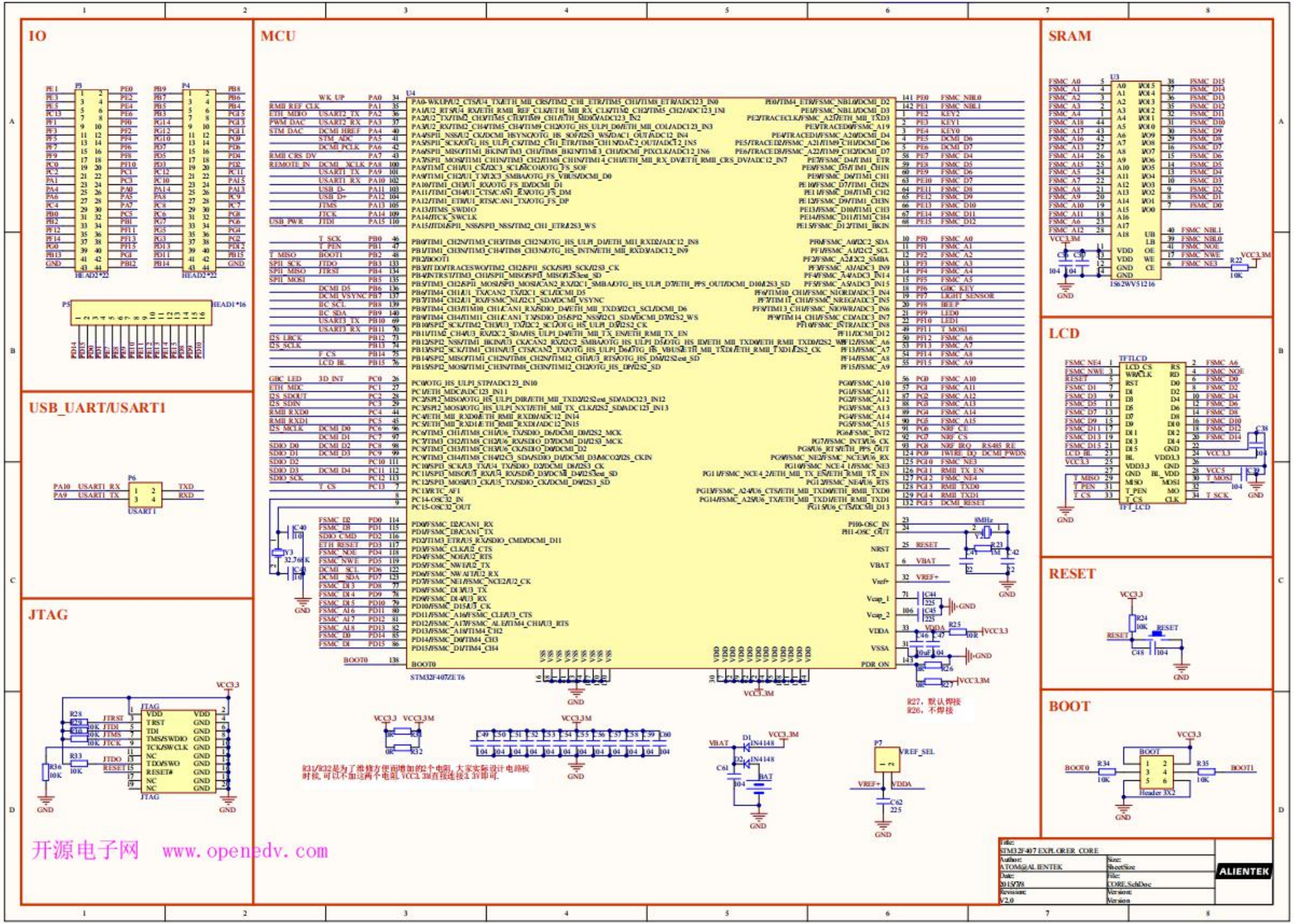
参考电压 选择端口

MPU605 0 传感器

STM32F407ZGT6
HPAFFVQ
ARM
K0RHP430



LQFP144
1 PE2
2 PE3
3 PE4
4 PE5
5 PE6
6 VBAT
7 PC13
8 PC14
9 PC15
10 PF0
11 PF1
12 PF2
13 PF3
14 PF4
15 PF5
16 V_{SS}
17 V_{DD}
18 PF6
19 PF7
20 PF8
21 PF9
22 PF10
23 PH0
24 PH1
25 NRST
26 PC0
27 PC1
28 PC2
29 PC3
30 V_{DD}
31 V_{SSA}
32 V_{REF+}
33 V_{DDA}
34 PA0
35 PA1
36 PA2
37 PA3
38 V_{SS}
39 V_{DD}
40 PA4
41 PA5
42 PA6
43 PA7
44 PC4
45 PC5
46 PB0
47 PB1
48 PB2
49 PF11
50 PF12
51 V_{SS}
52 V_{DD}
53 PF13
54 PF14
55 PF15
56 PG0
57 PG1
58 PE7
59 PE8
60 PE9
61 V_{SS}
62 V_{DD}
63 PE10
64 PE11
65 PE12
66 PE13
67 PE14
68 PE15
69 PB10
70 PB11
71 V_{CAP_1}
72 V_{DD}
73 PB12
74 PB13
75 PB14
76 PB15
77 PD8
78 PD9
79 PD10
80 PD11
81 PD12
82 PD13
83 V_{SS}
84 V_{DD}
85 PD14
86 PD15
87 PG2
88 PG3
89 PG4
90 PG5
91 PG6
92 PG7
93 PG8
94 V_{SS}
95 V_{DD}
96 PC6
97 PC7
98 PC8
99 PC9
100 PA8
101 PA9
102 PA10
103 PA11
104 PA12
105 PA13
106 V_{CAP_2}
107 V_{SS}
108 V_{DD}
109 PA14
110 PA15
111 PC10
112 PC11
113 PC12
114 PD0
115 PD1
116 PD2
117 PD3
118 PD4
119 PD5
120 V_{SS}
121 V_{DD}
122 PD6
123 PD7
124 PG9
125 PG10
126 PG11
127 PG12
128 PG13
129 PG14
130 V_{SS}
131 V_{DD}
132 PG15
133 PB3
134 PB4
135 PB5
136 PB6
137 PB7
138 BOOT0
139 PB8
140 PB9
141 PE0
142 PE1
143 PDR_ON
144 V_{DD}

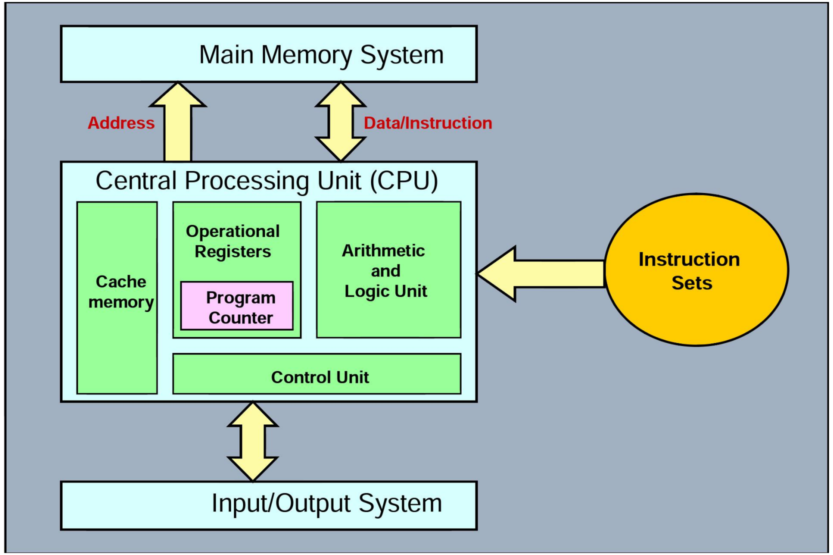


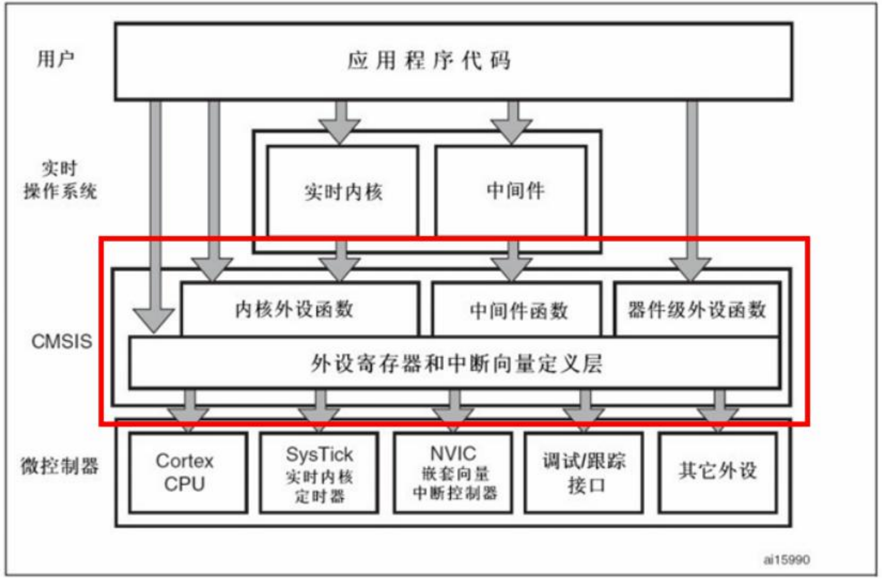
R27，默认焊接
R26，不焊接

R31/R32是为了维修方便而增加的2个电阻，大家实际设计电路板时候，可以不加这两个电阻，VCC3.3M直接连接3.3V即可。

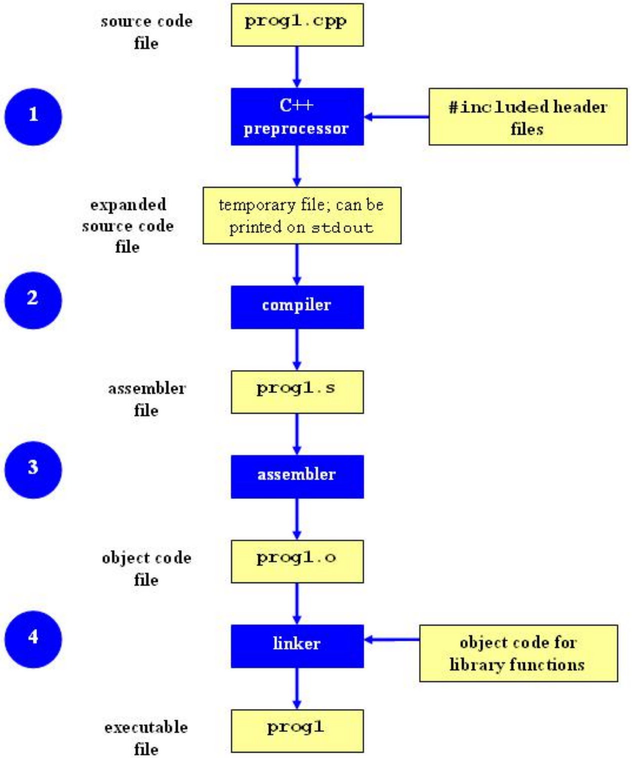
Title: STM32F407 EXPLORER CORE	
Author: ATOM@ALIENTEK	Size: SheetSize
Date: 2015/7/8	File: CORE.SchDoc
Revision: V2.0	Version: Version

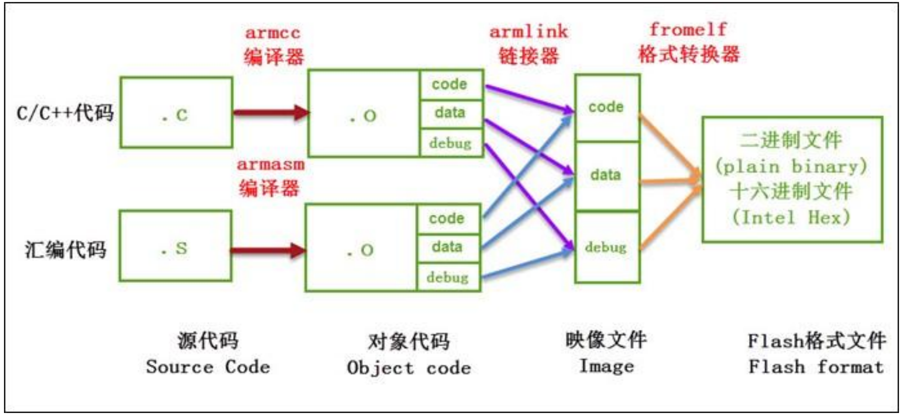
ALIENTEK


Main Memory System
Address
Data/Instruction
Central Processing Unit (CPU)
Cache memory
Operational Registers
Program Counter
Arithmetic and Logic Unit
Control Unit
Instruction Sets
Input/Output System


用户
应用程序代码
实时操作系统
实时内核
中间件
CMSIS
内核外设函数
中间件函数
器件级外设函数
外设寄存器和中断向量定义层
微控制器
Cortex CPU
SysTick 实时内核定时器
NVIC 嵌套向量中断控制器
调试/跟踪接口
其它外设
ai15990

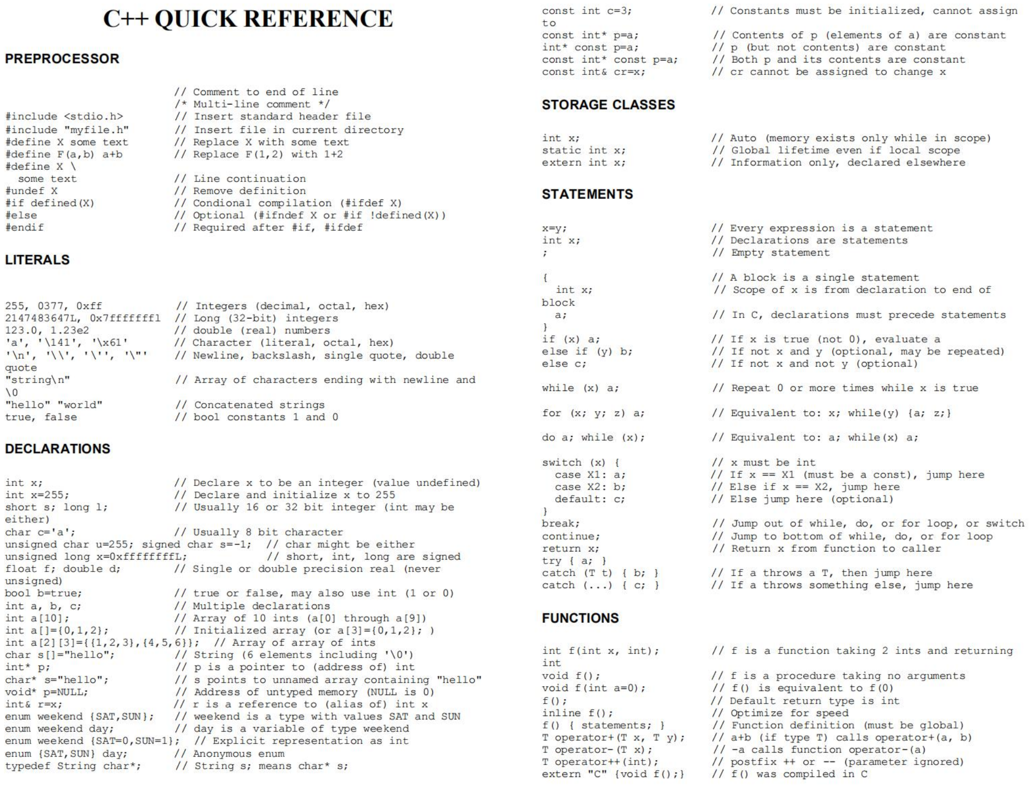
Programming





Details:
https://www.cnblogs.com/firege/p/5806134.html

Simple C++



C++ QUICK REFERENCE

PREPROCESSOR

```
                           // Comment to end of line
                           /* Multi-line comment */
#include <stdio.h>         // Insert standard header file
#include "myfile.h"        // Insert file in current directory
#define X some text        // Replace X with some text
#define F(a,b) a+b         // Replace F(1,2) with 1+2
#define X \
  some text                // Line continuation
#undef X                   // Remove definition
#if defined(X)             // Condional compilation (#ifdef X)
#else                      // Optional (#ifndef X or #if !defined(X))
#endif                     // Required after #if, #ifdef
```

LITERALS

```
255, 0377, 0xff            // Integers (decimal, octal, hex)
2147483647L, 0x7fffffffl   // Long (32-bit) integers
123.0, 1.23e2              // double (real) numbers
'a', '\141', '\x61'        // Character (literal, octal, hex)
'\n', '\\', '\'', '\"'     // Newline, backslash, single quote, double
quote
"string\n"                 // Array of characters ending with newline and
\0
"hello" "world"            // Concatenated strings
true, false                // bool constants 1 and 0
```

DECLARATIONS

```
int x;                     // Declare x to be an integer (value undefined)
int x=255;                 // Declare and initialize x to 255
short s; long l;           // Usually 16 or 32 bit integer (int may be
either)
char c='a';                // Usually 8 bit character
unsigned char u=255; signed char s=-1;  // char might be either
unsigned long x=0xffffffffL;            // short, int, long are signed
float f; double d;         // Single or double precision real (never
unsigned)
bool b=true;               // true or false, may also use int (1 or 0)
int a, b, c;               // Multiple declarations
int a[10];                 // Array of 10 ints (a[0] through a[9])
int a[]={0,1,2};           // Initialized array (or a[3]={0,1,2}; )
int a[2][3]={{1,2,3},{4,5,6}};  // Array of array of ints
char s[]="hello";          // String (6 elements including '\0')
int* p;                    // p is a pointer to (address of) int
char* s="hello";           // s points to unnamed array containing "hello"
void* p=NULL;              // Address of untyped memory (NULL is 0)
int& r=x;                  // r is a reference to (alias of) int x
enum weekend {SAT,SUN};    // weekend is a type with values SAT and SUN
enum weekend day;          // day is a variable of type weekend
enum weekend {SAT=0,SUN=1};  // Explicit representation as int
enum {SAT,SUN} day;        // Anonymous enum
typedef String char*;      // String s; means char* s;
const int c=3;             // Constants must be initialized, cannot assign
to
const int* p=a;            // Contents of p (elements of a) are constant
int* const p=a;            // p (but not contents) are constant
const int* const p=a;      // Both p and its contents are constant
const int& cr=x;           // cr cannot be assigned to change x
```

STORAGE CLASSES

```
int x;                     // Auto (memory exists only while in scope)
static int x;              // Global lifetime even if local scope
extern int x;              // Information only, declared elsewhere
```

STATEMENTS

```
x=y;                       // Every expression is a statement
int x;                     // Declarations are statements
;                          // Empty statement

{                          // A block is a single statement
  int x;                   // Scope of x is from declaration to end of
block
  a;                       // In C, declarations must precede statements
}
if (x) a;                  // If x is true (not 0), evaluate a
else if (y) b;             // If not x and y (optional, may be repeated)
else c;                    // If not x and not y (optional)

while (x) a;               // Repeat 0 or more times while x is true

for (x; y; z) a;           // Equivalent to: x; while(y) {a; z;}

do a; while (x);           // Equivalent to: a; while(x) a;

switch (x) {               // x must be int
  case X1: a;              // If x == X1 (must be a const), jump here
  case X2: b;              // Else if x == X2, jump here
  default: c;              // Else jump here (optional)
}
break;                     // Jump out of while, do, or for loop, or switch
continue;                  // Jump to bottom of while, do, or for loop
return x;                  // Return x from function to caller
try { a; }
catch (T t) { b; }         // If a throws a T, then jump here
catch (...) { c; }         // If a throws something else, jump here
```

FUNCTIONS

```
int f(int x, int);         // f is a function taking 2 ints and returning
int
void f();                  // f is a procedure taking no arguments
void f(int a=0);           // f() is equivalent to f(0)
f();                       // Default return type is int
inline f();                // Optimize for speed
f() { statements; }        // Function definition (must be global)
T operator+(T x, T y);     // a+b (if type T) calls operator+(a, b)
T operator-(T x);          // -a calls function operator-(a)
T operator++(int);         // postfix ++ or -- (parameter ignored)
extern "C" {void f();}     // f() was compiled in C
```

Programming in STM32

10 mins

GPIO

输入：

模拟输入-获得外部的模拟信号，输入不经过输入数据寄存器
浮空输入-输入完全由外部决定，此时IO的电平状态未知
上拉输入-外部无输入时，为高电平
下拉输入-外部无输入时，为低电平

输出：

开漏输出-输出0时为GND，输出1时，由外接上拉电阻决定
推挽输出-输出0时为GND，输出1时为VCC
复用开漏输出-片内外设功能（TX1, MOSI, MISO, SCK, SS）
复用推挽输出-片内外设功能（I2C的SCL, SDA）

```
void LED_Init(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOF, ENABLE); //使能 GPIOF 时钟

    //GPIOF9,F10 初始化设置
   GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9 | GPIO_Pin_10;           //LED0 和 LED1 对应 IO 口
   GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;                      //普通输出模式
   GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;                      //推挽输出
   GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;               //100MHz
   GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;                       //上拉

   GPIO_Init(GPIOF, &GPIO_InitStructure);                                  //初始化 GPIO

   GPIO_SetBits(GPIOF,GPIO_Pin_9 | GPIO_Pin_10);                    //GPIOF9,F10 设置高，灯灭
}
```