编 号

《嵌入式系统》

期末大作业

题目： 嵌入式设备监控系统

人工智能与计算机 学 院 计算机科学与技术 专 业

学　　号 1033180317

学生姓名 王汝宇

指导教师 陈志国

二〇二一年六月

摘 要

本次大作业设计了一个以S5P6818 ARM处理器为核心的嵌入式系统，实现通过传感器对原料密度、阀开关的状态监测，完成原料储罐的自动添水加料。各章节分别介绍了系统概要、硬件设计、软件设计以及心得体会。

目　录

摘 要 I
目　录 I
第 1 章 概述 1
1.1 设计背景 1
1.2 设计内容 1
第 2 章 硬件设计 3
2.1 系统架构 3
2.2 电路原理图 3
第 3 章 软件设计 5
3.1 软件流程图 5
3.1.1 总流程图 5
3.1.2 密度计控制 Watch_Rou() 5
3.1.3 液位开关 1 控制 Watch_Height() 5
3.2 软件代码 6
3.2.1 头文件及 define 6
3.2.2 初始化 GPIO 口 6
3.2.3 GPIO 控制 7
3.2.4 液位开关 1/阀门 3 控制 7
3.2.5 系统初始化 8
3.2.6 延时函数 8
3.2.7 密度计控制 9
3.2.8 主函数 10
第 4 章 结论与体会 11
4.1 预期运行结果 11
4.2 体会 11
参考文献 12

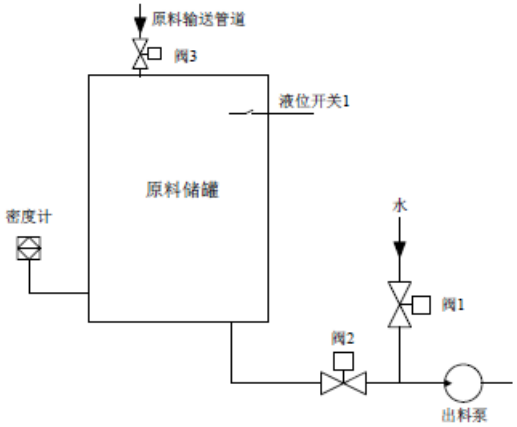
第 1 章 概述

1.1 设计背景

目前嵌入式系统的研究和应用中，ARM 芯片的使用越来越广泛。ARM 微处理器是由 ARM 公司提供 IP（Intellectual Property，知识产权）授权，交付多个芯片设计厂商整合生产的。在 2007 年，意法半导体（ST）公司成为第一个引入 ARM Cortex-M 授权的半导体厂商，开启了高性能、低成本、低功耗的 ARM 嵌入式芯片新时代，其生产的 STM32 系列微处理器是最流行的 Cortex-M 微处理器。ARM 嵌入式系统广泛应用于自动检测与控制、智能仪器仪表、机电一体化设备、汽车电子及日常消费电子产品中，其优越的性能和完善的开发环境得到广大电子工程师的青睐。同时，嵌入式技术不是单纯的软件技术，也不是单纯的硬件技术，是一门如何在一个特定的硬件环境上开发与构建特定的可编程软件系统的综合技术。嵌入式技术是在嵌入式系统的发展中应运而生的，它是依附于嵌入式系统，并推动嵌入式系统不断向前发展的核心动力。嵌入式是一种专用的计算机系统，作为装置或设备的一部分。通常，嵌入式系统是一个控制程序存储在 ROM 中的嵌入式处理器控制板。事实上，所有带有数字接口的设备，如手表、微波炉、录像机、汽车等，都使用嵌入式系统，有些嵌入式系统还包含操作系统，但大多数嵌入式系统都是由单个程序实现整个控制逻辑。嵌入式技术近年来得到了飞速的发展，但是嵌入式产业涉及的领域非常广泛，彼此之间的特点也相当明显。例如，很多行业：手机、PDA、车载导航、工控、军工、多媒体终端、网关、数字电视等。在了解嵌入式系统基础理论的前提下，掌握一些 ARM 处理器相关的汇编语言和 C 语言程序设计方法，熟悉基于 S5P6818 芯片的硬件接口设计方法，学会使用 ARM 集成开发环境，从而了解嵌入式系统的软硬件设计过程，才能为今后从事相关领域的应用和研究打好基础。

1.2 设计内容

某单位拟对原料储罐设备进行控制，系统装置如下图所示：



已知在原料罐上安装了 1 个密度计（密度计测量范围：0.5~2.0 g/cm3），密度计为电压输出模式，输出电压范围：0~5V；同时还安装了 1 个液位开关，当液位超过液位开关 1 所在位置时开关处于闭合状态（否则处于断开态）。液位开关由 CPU 的 GPIO 口进行状态采集。出料泵采用 CPU 的 PWM 进行控制，初始时 PWM 输出占空比为 0%，泵处于停止状态。阀 1、阀 2 和阀 3 由 CPU 的 GPIO 口进行控制（GPIO 口输出 1，阀开启；GPIO 口输

出 0，阀关闭），初始时阀 1、阀 2 和阀 3 都处于关闭状态。

系统的控制程序如下：

1）系统对密度计、液位开关 1 进行实时采样，并将密度数据和液位开关 1 的状态通过适当的通讯接口传送到计算机 A，并在计算机 A 上显示出来。阀 3 由液位开关 1 控制，如果液位低于液位开关 1 所在位置，阀 3 开启，否则阀 3 关闭。

如果原料密度低于 1.6 g/cm3，阀 1 和阀 2 关闭，控制“出料泵”的 PWM 输出占空比为 0%，“出料泵”处于停止状态（PWM 输出占空比为 0%）。当密度计检测到原料密度达到 1.6 g/cm3 时，打开“阀 2”并启动“出料泵”将原料采出，此时控制“出料泵”的 PWM 输出占空比为 95%，10 分钟后进入下一步；

2）打开“阀 1”进行洗水操作（防止“出料泵”堵塞），此时控制“出料泵”的 PWM 输出占空比为 75%，30 秒后进入下一步；

3）关闭“阀 2”，此时控制“出料泵”的 PWM 输出占空比为 55%，30 秒后进入下一步；

4）关闭“阀 1”和“出料泵”（PWM 输出占空比为 0%），然后返回第 1 步。

以 S5P6818 的 ARM 处理器为核心，设计嵌入式系统。

第 2 章 硬件设计

2.1 系统架构

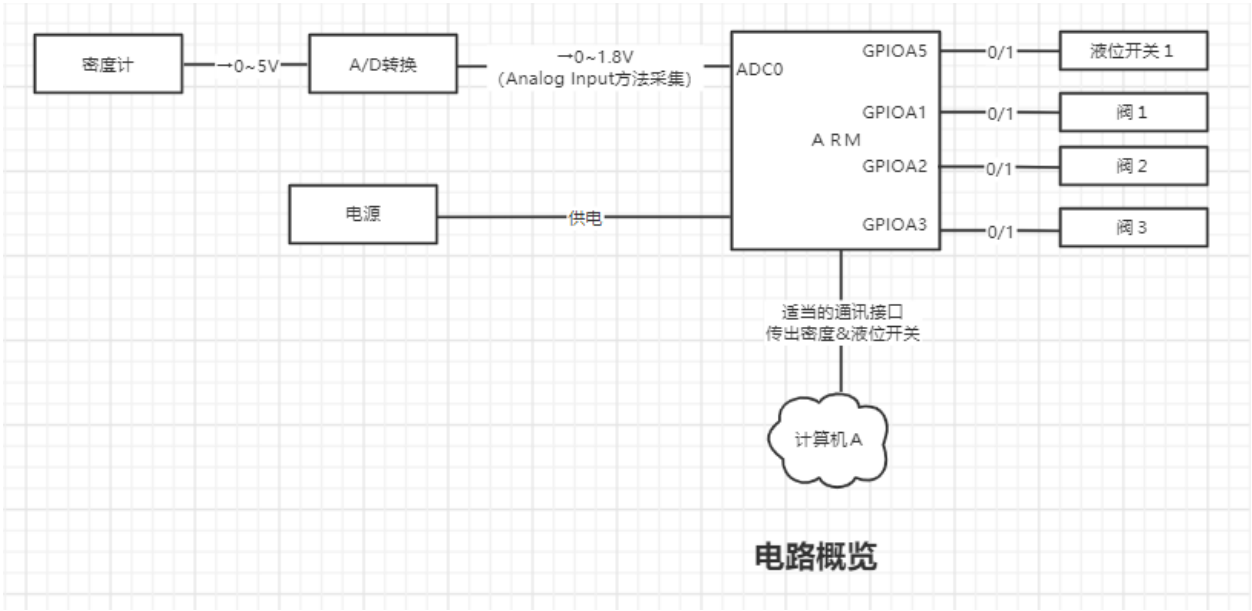


图2-1 电路概览

1、密度计将0.5~2.0 g/cm3的密度信息转换为0~5V的电压信息，输出到A/D转换芯片中，此处A/D转换芯片充当一个变压器的作用，将0~5V的电压转换为0~1.8V的电压，再输送到ARM中

2、液位开关1、阀1、2、3分别接入ARM的GPIO5、1、2、3接口，将阀门的开关信息转为0/1数值

3、ARM与计算机A通过适当的通讯接口连接，将密度信息与液位开关1的信息通过计算机A的上位程序传入计算机A。

2.2 电路原理图

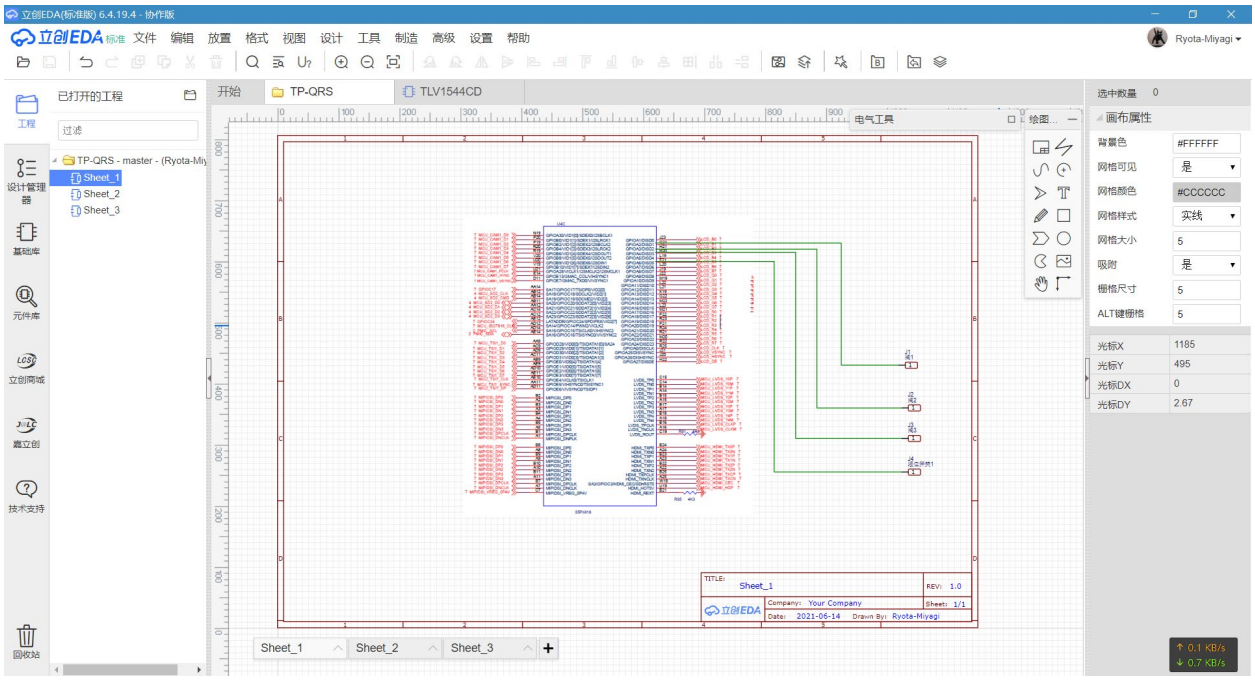


图2-2 ARM——阀门连接

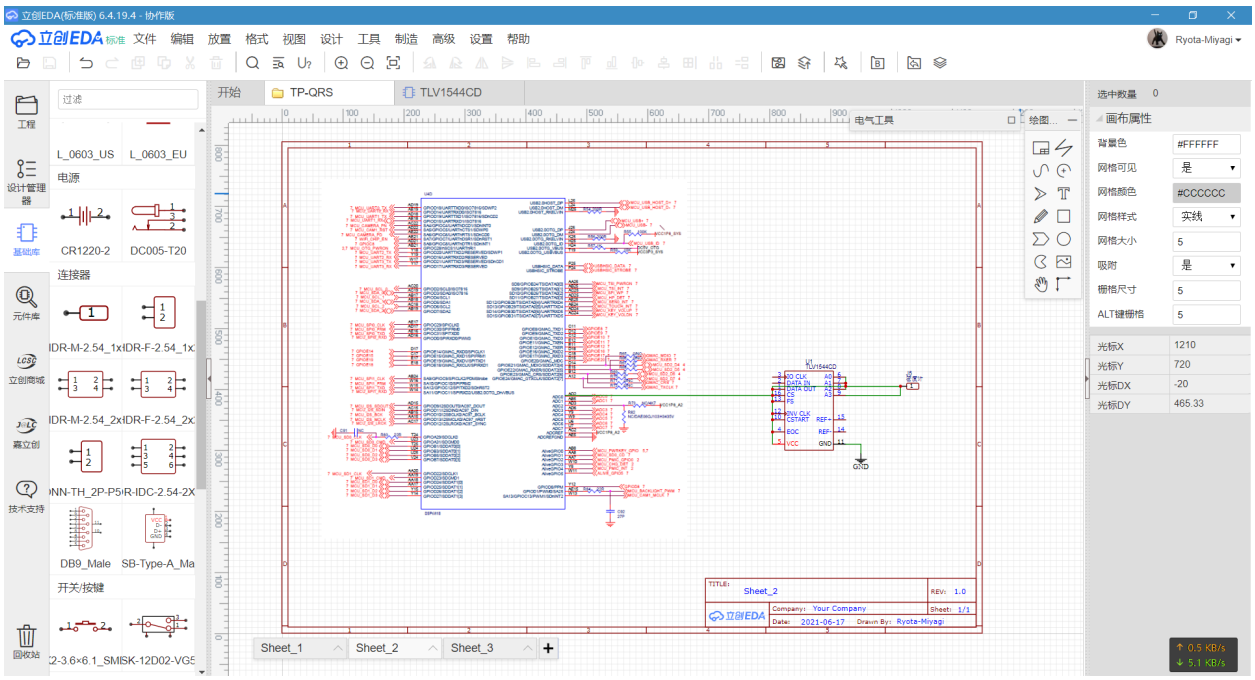


图2-3 密度计——A/D芯片——ARM连接

第 3 章 软件设计

3.1 软件流程图

3.1.1 总流程图

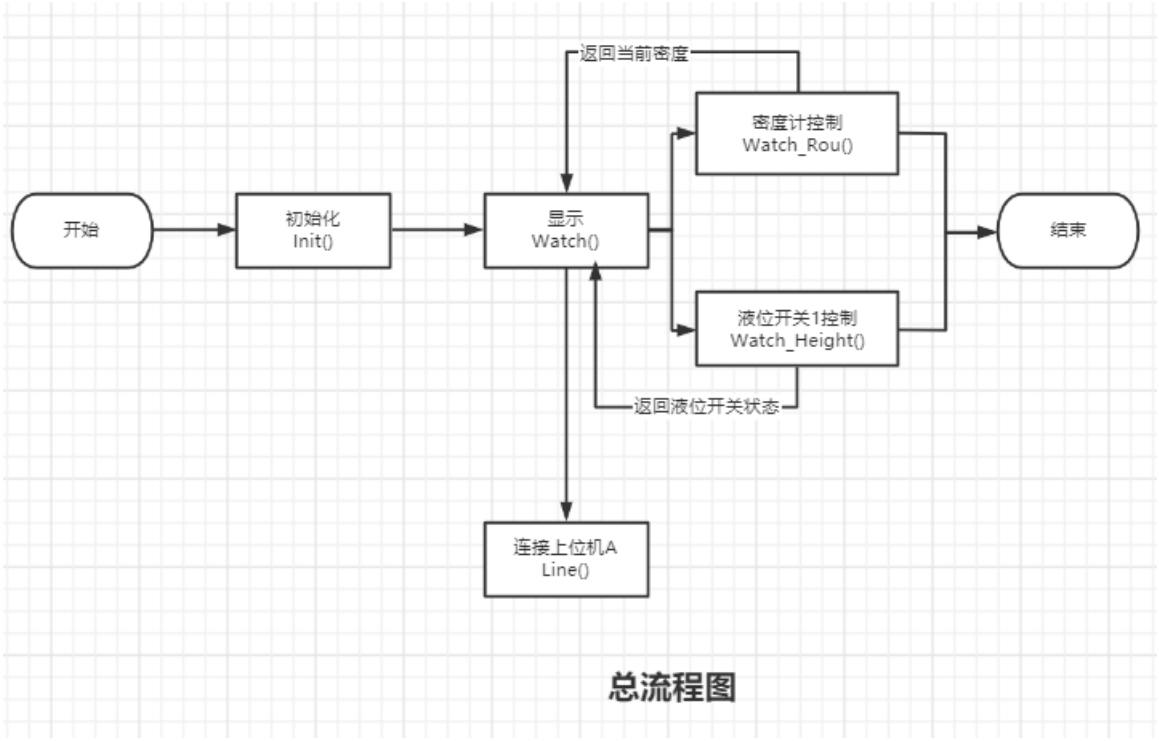


图 3-1 总流程图

3.1.2 密度计控制 Watch_Rou()

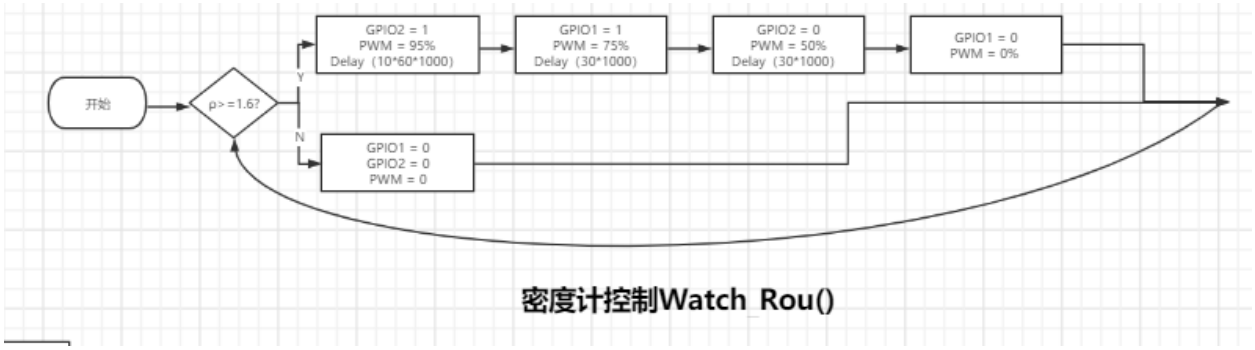


图 3-2 密度计控制函数

3.1.3 液位开关 1 控制 Watch_Height()

图 3-3 液位开关控制函数

3.2 软件代码

3.2.1 头文件及 define

```
#include <main.h>
#define GPIOFA  (*(volatile unsigned int*)0xC001A020)
#define GPIOFB  (*(volatile unsigned int*)0xC001A004)
#define GPIOFC  (*(volatile unsigned int*)0xC001A000)
#define TO_PERIOD_NS(freq)  ((1000000000UL)/(freq))
#define TO_DUTY_NS(duty, freq)  (duty ? TO_PERIOD_NS(freq)/(
100/duty) : 0)
```

3.2.2 初始化 GPIO 口

```
//初始化GPIO口
void Init_GPIO(){
    unsigned int reg =
0;
    //液位开关1 = 5
    reg = GPIOFA;
    reg &= ~(0x3 << 5);
    GPIOFA = reg;
    reg = GPIOFB;
    reg &= ~(0x1 << 5);
    reg |= (0x1 << 5);
    GPIOFB = reg;
    //阀1 = 1
    reg = GPIOFA;
    reg &= ~(0x3 << 1);
    GPIOFA = reg;
    reg = GPIOFB;
    reg &= ~(0x1 << 1);
    reg |= (0x1 << 1);
    GPIOFB = reg;
    //阀2 = 2
    reg = GPIOFA;
    reg &= ~(0x3 << 2);
    GPIOFA = reg;
    reg = GPIOFB;
    reg &= ~(0x1 << 2);
    reg |= (0x1 << 2);
    GPIOFB = reg;
    //阀3 = 3
    reg = GPIOFA;
    reg &= ~(0x3 << 3);
    GPIOFA = reg;
    reg = GPIOFB;
    reg &= ~(0x1 << 3);
    reg |= (0x1 << 3);
    GPIOFB = reg;
}
```

3.2.3 GPIO 控制

```c
//控制GPIO
void Control_GPIO(int n, int flag) {
    unsigned int reg = 0;
    if (flag == 1) {
        reg = GPIOFC;
        reg &= ~(0x1 << n);
        reg |= (0x1 << n);
        GPIOFC = reg;
    }else {
        reg = GPIOFC;
        reg &= ~(0x0 << n);
        reg |= (0x0 << n);
        GPIOFC = reg;
    }
}
```

3.2.4 液位开关 1/阀门 3 控制

```c
//液位开关1控制
int Watch_Height() {
    unsigned int reg = 0;
    reg = GPIOFC;
    if(((reg >> 5) & 1) == 1) Control_GPIO(3,0)
    if(((reg >> 5) & 1) == 0) Control_GPIO(3,1)
    return ((reg >> 5) & 1);
}
```

3.2.5 系统初始化

```c
//系统初始化
static void Init(void)
{
    pwm_config_u(2, TO_DUTY_NS(0, 1000), 1000000);
    malloc_init();
    s5p6818_pwm_init();
    s5p6818_serial_initial();
}
```

3.2.6 延时函数

```c
//延迟函数
void Delay(int ms) {
    int i, j = 0;
    for (i = 0; i < 4; i ++) {
        for (j = 0; j < 514; j ++){
        }
    }
}
```

3.2.7 密度计控制

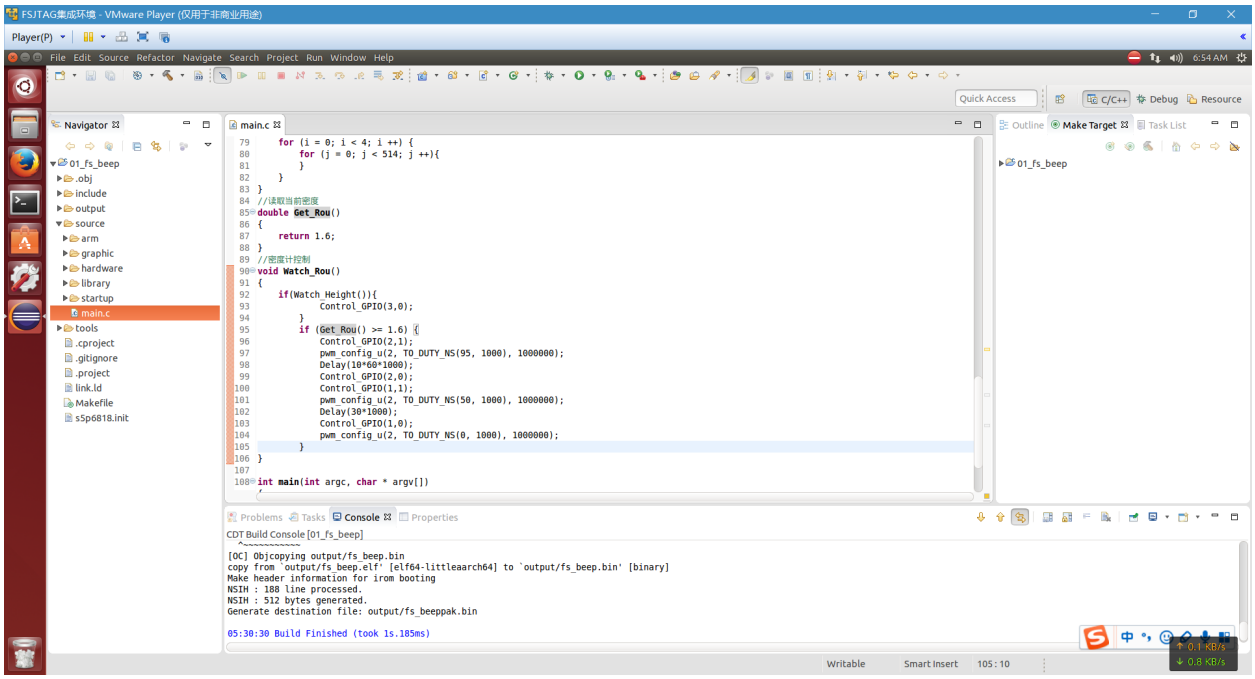
```
//密度计控制
void Watch_Rou()
{
    if(Watch_Height()){
            Control_GPIO(3,0);
        }
        if (getDensity() >= 1.6) {
            Control_GPIO(2,1);
            pwm_config_u(2, TO_DUTY_NS(95, 1000), 1000000);
            Delay(10*60*1000);
            Control_GPIO(2,0);
            Control_GPIO(1,1);
            pwm_config_u(2, TO_DUTY_NS(50, 1000), 1000000);
            Delay(30*1000);
            Control_GPIO(1,0);
            pwm_config_u(2, TO_DUTY_NS(0, 1000), 1000000);
        }
}
```

3.2.8 主函数

```
int main(int argc, char * argv[])
{
    Init();
    while(1) {
        Watch_Rou();
    }
    return 0;
}
```

第 4 章 结论与体会

4.1 预期运行结果

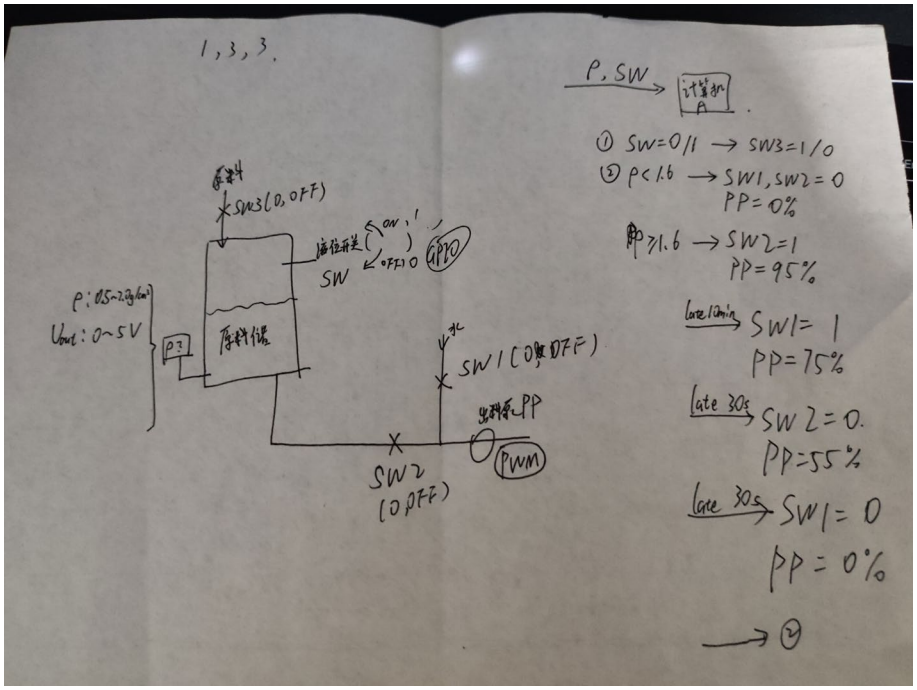


在 Eclipse 中编译通过，无错误

4.2 体会

在本次大作业中，温习了电路图的绘制方法，以 S5P6818 实验板为基础，更加深入的学习了 ARM 内核的嵌入式系统编译方法，还顺带复习了 C 语言。在查找底板电路图、查找引脚和串口控制方法的过程中也熟练掌握了查阅技术手册及相关参考文献的能力。

除此之外，在设计过程中也经历了数次修改与斟酌，也为以后的学习增长了相关经验。



参考文献

[1] 张卫平．开关变换器的建模与控制[M]．北京：中国电力出版社，2006，15-88.

[2] 伍言真．DC/DC 开关变换器建模分析及其变结构控制方法的研究[D]．广州：华南理工大学，1998.

[3] Takagi T, Sugeno M. Fuzzy identification of systems and its applications to modeling and control [J]. IEEE Trans on Systems, Man and Cybernetics, 1985, 15(2): 116-132.

[4] Sugeno M, Kang G T.　Fuzzy modeling and control of multilayer incinerator [J]. Fuzzy Sets Syst., 1986, 18(2): 329-346.