## 雨侦探

付炫钰；马文玉；房心茹；

**第一部分 设计概述**

* 1. 设计目的

随着时代不断发展，建设“智慧城市”的必要性日益突出，“海绵城市”作为智慧城市形成的创新表现，要求做到使城市既有“面子”，更有“里子”。而当我们目标锁定到现实生活，城市道路积水往往不具有良好的“弹性”，易给群众生产生活造成影响。

2021年河南郑州7.20特大暴雨造成巨大损失，为帮助避免此类事件的发生，设计“雨侦探”，通过传感器实现水压和水速的测量，采用基于龙芯开发板进行数据采集与转发，将数据上传至数据库，经过云平台对数据的处理，通过显示屏在指定地点进行显示。定位系统与地图相连，实现实地查看水量情况，当水量超过正常值，及时发出预警，“雨侦探”精准探测，各地互联，提前预防，更积极响应国家号召，助力构建智慧城市。

* 1. 应用领域

城市低洼地段由于雨水天气，积水不易排出，容易出现内涝和洪水，“雨侦探”的设计应用于城市道路积水的预警，实时监测低洼路段的积水情况，有效帮助城市管理者了解实时积水实况，为城市防灾减灾指挥调度决策提供数据支持，同时为公众出行提供方便，避免公众生命财产遭受损失。除此之外，可用于监控家庭地面积水、厨房漏水等问题；农村种植户使用产品，可帮助种植户足不出户了解田地水位状况。

* 1. 主要技术特点

1.液位传感器

液位传感器完成对积水量的测量，当液位变送器投入到被测液体中某一深度时，传感器迎液面受到的压力公式为：P = ρ .g.H + Po，通过导气不锈钢将液体的压力引入到传感器的正压腔，再将液面上的大气压 Po 与传感器的负压腔相连，以抵消传感器背面的 Po ，使传感器测得压力为：ρ .g.H ，通过测取压力 P ，可以得到液位深度。

2.通过NB-IoT模块完成信息转发

NB-IoT具备四大特点：一是便于实现广覆盖，二是具备支撑海量连接的能力，三是更低功耗，四是更低的模块成本，NB-IoT聚焦于低功耗广覆盖（LPWA）物联网（IoT），在积水探测器的实现中比WIFI通信模块更合适。

3.基于龙芯开发板的数据采集与转发

“雨侦探”的设计完成数据采集与转发，与其他开发板不同，基于龙芯的设计开发生态不太成熟，许多功能的实现都要开发者根据功能需求自行编写代码，没有丰富的库函数供开发者直接调用。基于龙芯的数据采集和转发涉及到相关传感器和通信模块的使用，这部分的驱动代码都需要手工编写和调试。

1. 云平台的使用

云平台来完成对数据的收集、处理和上传，同时保证在线服务比如实现实时查看水量和动态调整水量变化。

* 1. 关键性能指标

1.雨量测量的准确度

通过对模型的模拟测量，与实际人工测量作对比，计算雨量测量的误差，必要时通过一定的算法对数据进行处理，更好的反映实际雨量。

1. 测量雨量显示的实时性

因为要实现实时查看雨量，云平台运转速度快，通过对上传的数据进行处理来实现雨量的实时查看。但实际查看时可能存在一定延时，要采取多种方法使时间差最小。

* 1. 主要创新点

1.基于雨量信息分析的多层预警机制

雨量信息的分析结果及时上传到常用的地图导航软件，进一步通过数据分析，使用户实时准确了解相关信息，积水过于严重时预警做出强制预警，公众必须接受，并能够帮助及时开展救援。其次是地图软件做出的预警，包括等级预警以及路线预警。

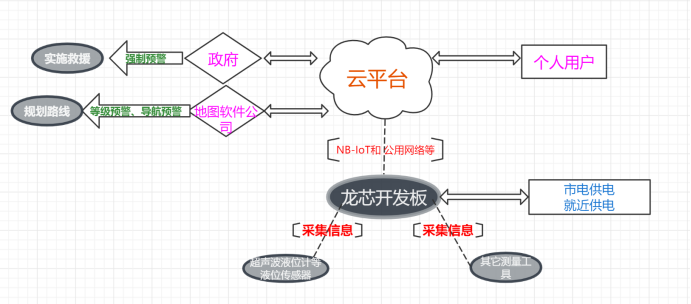
2.体积小巧，便于铺设

产品体积小，设置到城市中，不容易被损坏或盗窃，便于开展大量铺设，能够做到“想测就测，测完回收”，避免暴雨洪涝中因各种意外情况而不知水位情况的隐患。

**第二部分 系统组成及功能说明**

* 1. 整体介绍

本团队设计的这款产品采用国产龙芯开发板最新的龙芯1C SoC芯片，这款芯片具有功耗低、速度快的优点，整个产片采用的技术架构如图1所示。主要包含基于龙芯开发板的数据采集与转发、准确的实时预测和完备的预警机制、数据服务的粘合性等关键技术。



* 1. 各模块介绍

1. 基于龙芯开发板的数据采集与转发

“雨侦探”的设计完成数据采集与转发，与其他开发板不同，基于龙芯的设计开发生态不太成熟，许多功能的实现都要开发者根据功能需求自行编写代码，没有丰富的库函数供开发者直接调用。基于龙芯的数据采集和转发涉及到相关传感器和通信模块的使用，这部分的驱动代码都需要手工编写和调试。

1. 准确的实时预测和完备的预警机制

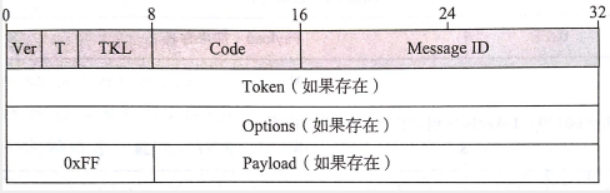
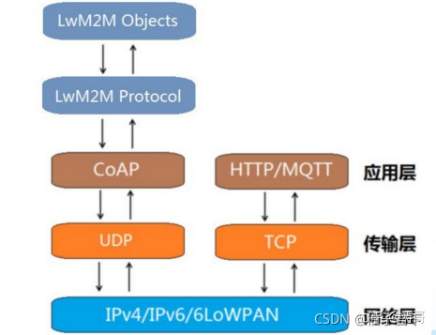
“雨侦探”采用的液位传感器能够实现较为准确的测量，数据经过云平台的数据整合及通过道路预测算法水位预测接下来一段时间的水位数据分析，并及时获取气象部门预测信息，能够最终做出及时、准确的预测。预警机制上，“雨侦探”采用两种途径预警，首先是政府途径，在积水较为严重以至于造成灾害时做出强制预警，其次是地图软件做出的预警，包括等级预警以及路线预警。

预测算法基于历史雨量和预报雨量的数据，基于神经网络算法分析，能够实现准确的对未来一段时间内水位变化的预测，以便针对不同客户提供预警服务。

1. 数据服务的粘合性

雨量信息的分析结果及时上传到常用的地图导航软件，进一步通过数据分析，将预测信息直接推送给用户，实现数据服务的粘合性很关键，粘合性好的可以与其他软件系统或者app快速融合，实现数据的共享，更及时方便地给用户推送等级预警和路线预警。

“雨侦探”通过数据采集得到的实时雨量信息，并依据Coap协议，通过NB-IOT模块进行转发。CoAP协议是一种面向资源受限的应用层协议，是一种轻量级协议，更好的适配龙芯开发板此类小型设备。最大限度避免数据传输中遇到的环境的不稳定性的问题，例如无线网络的带宽、时延、丢包等问题。



**第三部分 完成情况及性能参数**

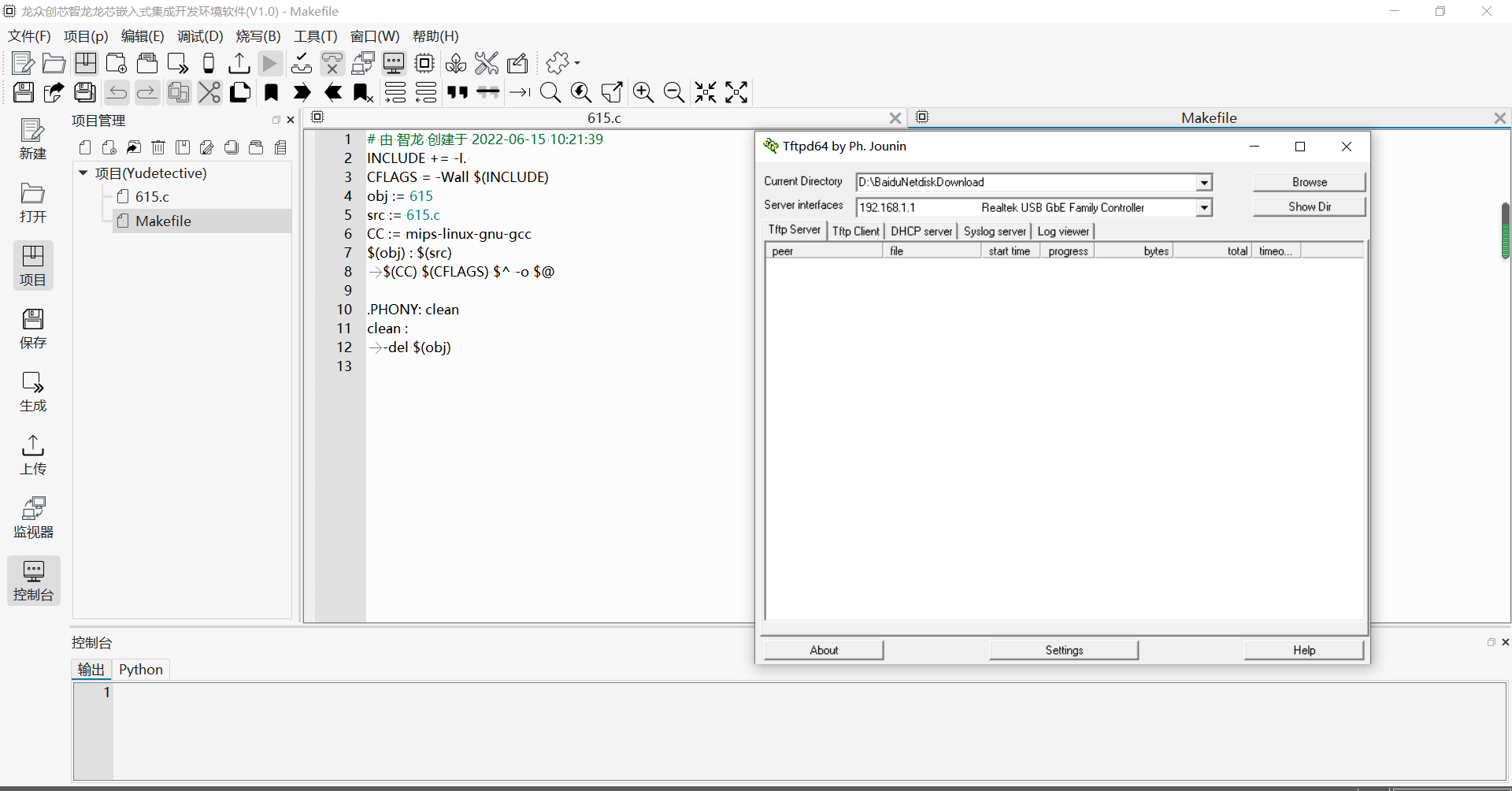
3.1硬件部分

3.1.1龙芯1C智龙开发板与windows系统电脑连接

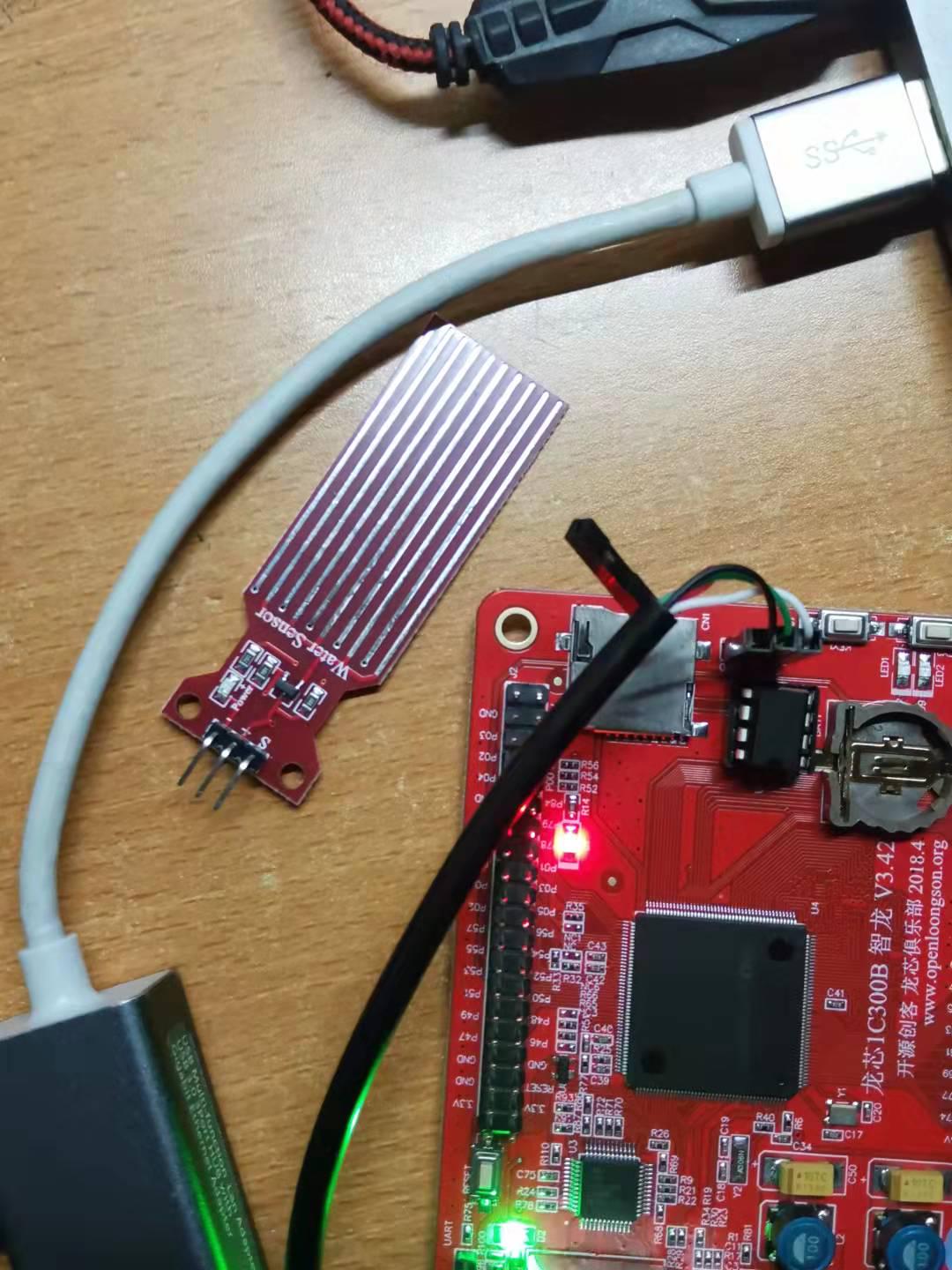


3.1.2智龙IDE

智龙集成开发环境是针对智龙 1C 系列开发板的一个集成式开发环境。统一简称为 IDE 或智龙 IDE。



3.1.3水位传感器（Water Sensor）是一款简单易用、性价比较高的水位/水滴识别检测传感器，其是通过具有一系列的暴露的平行导线线迹测量其水滴/水量大小从而判断水位。Water Sensor水位传感器不仅可以测量水位，还可以测量是否有水，当测量是否有水时，直接检测输出端引脚，若检测为0，则显示没有水，若检测到1，则有水；当检测水位变化时，需要进行ADC采集，利用函数进行模拟电压值到水位的转换。



目前阶段完成的代码与功能如下所示：

1. 检测是否有水

#include "water.h"

#include "delay.h"

#define WA GPIO\_Pin\_11

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 配置LED用到的I/O口 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Water\_GPIO\_Config(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd( RCC\_APB2Periph\_GPIOB, ENABLE); // 使能端口时钟

GPIO\_InitStructure.GPIO\_Pin = WA;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB, &GPIO\_InitStructure); //初始化端口

//GPIO\_SetBits(GPIOB, WA); // 关闭所有LED

}

int Water\_state(GPIO\_TypeDef\* GPIOx, uint16\_t GPIO\_Pin\_x)

{

Delay\_ms(10);

if( GPIO\_ReadInputDataBit(GPIOx, GPIO\_Pin\_x) == 0)

{

return 1;

}

else

{

return 0;

}

}

if( Water\_state(GPIOB, GPIO\_Pin\_11)== 0)

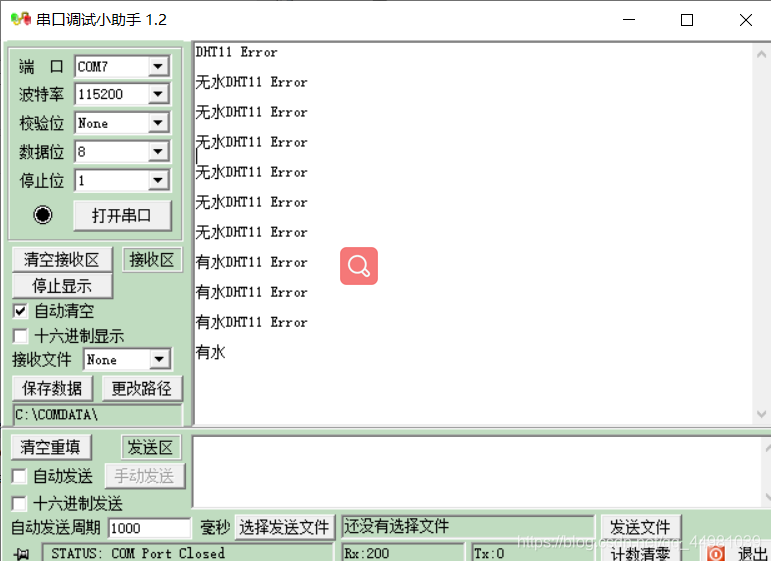
{

printf("\r\n有水");

}

else

printf("\r\n无水");



1. 测量水位

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 文件名 ：adc.c

\* 描述 ：DMA方式读取ADC值应用函数库

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "adc.h"

#define ADC1\_DR\_Address ((u32)0x4001244C)

\_\_IO u16 ADC\_ConvertedValue;

/\*配置采样通道端口 使能GPIO时钟 设置ADC采样PA0端口信号\*/

void ADC1\_GPIO\_Config(void)

{ GPIO\_InitTypeDef GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE);

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_0;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AIN; //GPIO设置为模拟输入

GPIO\_Init(GPIOA, &GPIO\_InitStructure);

}

/\*配置ADC1的工作模式为MDA模式 \*/

void ADC1\_Mode\_Config(void)

{

DMA\_InitTypeDef DMA\_InitStructure;

ADC\_InitTypeDef ADC\_InitStructure;

RCC\_AHBPeriphClockCmd(RCC\_AHBPeriph\_DMA1, ENABLE); //使能MDA1时钟

/\* DMA channel1 configuration \*/

DMA\_DeInit(DMA1\_Channel1); //指定DMA通道

DMA\_InitStructure.DMA\_PeripheralBaseAddr = ADC1\_DR\_Address;//设置DMA外设地址

DMA\_InitStructure.DMA\_MemoryBaseAddr = (u32)&ADC\_ConvertedValue; //设置DMA内存地址，ADC转换结果直接放入该地址

DMA\_InitStructure.DMA\_DIR = DMA\_DIR\_PeripheralSRC; //外设为设置为数据传输的来源

DMA\_InitStructure.DMA\_BufferSize = 1; //DMA缓冲区设置为1；

DMA\_InitStructure.DMA\_PeripheralInc = DMA\_PeripheralInc\_Disable;

DMA\_InitStructure.DMA\_MemoryInc = DMA\_MemoryInc\_Disable;

DMA\_InitStructure.DMA\_PeripheralDataSize = DMA\_PeripheralDataSize\_HalfWord;

DMA\_InitStructure.DMA\_MemoryDataSize = DMA\_MemoryDataSize\_HalfWord;

DMA\_InitStructure.DMA\_Mode = DMA\_Mode\_Circular;

DMA\_InitStructure.DMA\_Priority = DMA\_Priority\_High;

DMA\_InitStructure.DMA\_M2M = DMA\_M2M\_Disable;

DMA\_Init(DMA1\_Channel1, &DMA\_InitStructure);

/\* Enable DMA channel1 \*/

DMA\_Cmd(DMA1\_Channel1, ENABLE); //使能DMA通道

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_ADC1, ENABLE); //使能ADC1时钟

/\* ADC1 configuration \*/

ADC\_InitStructure.ADC\_Mode = ADC\_Mode\_Independent; //使用独立模式，扫描模式

ADC\_InitStructure.ADC\_ScanConvMode = ENABLE;

ADC\_InitStructure.ADC\_ContinuousConvMode = ENABLE; //无需外接触发器

ADC\_InitStructure.ADC\_ExternalTrigConv = ADC\_ExternalTrigConv\_None; //使用数据右对齐

ADC\_InitStructure.ADC\_DataAlign = ADC\_DataAlign\_Right;

ADC\_InitStructure.ADC\_NbrOfChannel = 1; // 只有1个转换通道

ADC\_Init(ADC1, &ADC\_InitStructure);

/\* ADC1 regular channel11 configuration \*/

ADC\_RegularChannelConfig(ADC1, ADC\_Channel\_0, 1, ADC\_SampleTime\_55Cycles5); //通道1采样周期55.5个时钟周期

/\* Enable ADC1 DMA \*/

ADC\_DMACmd(ADC1, ENABLE); //使能ADC的DMA

/\* Enable ADC1 \*/

ADC\_Cmd(ADC1, ENABLE); //使能ADC1

/\* Enable ADC1 reset calibaration register \*/

ADC\_ResetCalibration(ADC1);

/\* Check the end of ADC1 reset calibration register \*/

while(ADC\_GetResetCalibrationStatus(ADC1));

/\* Start ADC1 calibaration \*/

ADC\_StartCalibration(ADC1);

/\* Check the end of ADC1 calibration \*/

while(ADC\_GetCalibrationStatus(ADC1));

/\* Start ADC1 Software Conversion \*/

ADC\_SoftwareStartConvCmd(ADC1, ENABLE); //开始转换

}

/\*初始化ADC1 \*/

void ADC1\_Init(void)

{

ADC1\_GPIO\_Config();

ADC1\_Mode\_Config();

}

AD\_value = 3300000/4096\*ADC\_ConvertedValue/1000;

val=(exp(0.0056\*AD\_value))\*0.467;

printf("水位深度 = %d mm \r\n", val);

Delay\_ms(1000);



3.2云平台部分

云平台部分，我们选择的是阿里云S6 1核2G服务器，优势如下：



以及价格便宜，适合学生使用，现只租赁一个月的时间，可以支撑我们完成实验。



本项目中云平台的首要作用是整合传感器收集的数据，并进行算法运算。

整合数据并尽可能化为图像输出，向用户展示水位的变化趋势以及实时显示水位情况。目前云平台的各项功能还在调试学习中，已经学习如何连接设备与云平台，并在其中传送数据，但是由于进展问题还未实际检测。

整个设计还有部分功能未实现，性能参数等还有待测试。

**第四部分 总结**

* 1. 可扩展之处

1.对雨量的预测，除了实时查看雨量，实现雨量的预测也是现实应用中的需要，对雨量的预测还要分时间段实现，根据现有的数据对未来一段时间的雨量进行预测，可以通过结合雨量预测算法来实现。现有的雨量水位预测算法基本都基于河流、洪水等现实情况，我们可以根据需要更改一些参数设置以及综合国家气象平台出台的数据，得到我们需要的道路水位预测算法。

2.在有相应条件的情况下，可以设置一定的紧急处理机制，例如水泵等，来实现对积水的紧急处理。在实际生产生活中，积水可能造成的较为严重的影响，这就需要提前准备好应急措施，也可以结合对雨量的预测，提前制动，在雨量有持续大量上涨的趋势时对雨量进行处理，防止积水情况进一步恶化。

* 1. 心得体会

最初定好选题，最开始不知道从何下手，后来看到大家逐渐理出了参赛的头绪，感到有些压力，在指导老师的引导下，首先确定了参赛主题，小组成员初步明确和计划了每一阶段的工作，小组一起查找资料，确定开发板和外设，确定整个设计的不同模块，后分步骤一点一点的进行，在整个过程中无论是能力、知识，在小组的交流合作中都得到很大的提高。

首先，积极的参与小组活动，小组成员的合作能力有了很大的提高，与小组成员相互配合，使我们之间有了很大的默契与信赖，我们彼此激励相互促进，培养了我们的合作意识，同时培养了自己自主完成任务的能力。

在比赛准备过程中，学会了收集和筛选材料。合作中学会了根据自己的比赛主题，合理解决所需的各种材料，并初步学会运用这些材料，通过分析、整理、综合等多种方法解决问题。学会了用富有条理性、逻辑性的语言来叙述小组讨论的结论，阐述小组每一部分需要完成的内容，描述小组下一步需要开展的工作并及时进行总结。在讨论中，各个成员能主动发表自己对问题的观点，能结合小组成员的观点，及时调整自己的思路，并从中产生更为新颖的想法，合作的意识及合作的能力得到了初步的养成。

同时在比赛过程中学会了要及时整理出现的问题，通过和指导老师交流和后期查找资料、修正来不断解决问题，过程中可能出现很多不会的地方，尤其是最开始的时候，但要戒骄戒躁，一步一步来克服，积少成多，逐渐实现起来变得更容易。