分类号 编 号

U D C 密 级



**本科生毕业设计（论文）**

**题 目： 工业物联网数据管理信息系统**

**与终端硬件设计**

**姓 名： 马思清**

**学 号： 11712610**

**系 别： 电子与电气工程系**

**专 业： 信息工程**

**指导教师： 虞亚军**

2021 年 月 日

**诚信承诺书**

1.本人郑重承诺所呈交的毕业设计（论文），是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料均真实可靠。

2.除文中已经注明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的作品或成果。对本论文的研究作出重要贡献的个人和集体，均已在文中以明确的方式标明。

3.本人承诺在毕业论文（设计）选题和研究内容过程中没有抄袭他人研究成果和伪造相关数据等行为。

4.在毕业论文（设计）中对侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

作者签名：

年 月 日

工业物联网数据管理信息系统与终端硬件设计

马思清

（电子与电气工程系 指导教师：虞亚军）

[摘要]： 本设计是关于工业企业物联网系统的一套完整搭建方案，涉及设备数据采集，信号处理，信息分析以及信息的可视化应用，涵盖了物联网信息收集、处理、展示 全流程。本方案的具体内容包括一个数据采集硬件设备，一个联接管理软件系统，与一个用户端应用程序。数据采集硬件按照工业标准设计，可采集工业设备的各种数据，实现了多种模拟量与传感器信号的接入，并搭载最新的窄带物联网（NB-IoT）技术，有着良好的无线通信性能。联接管理系统是一个运行在云服务器上的程序，收集各个硬件终端通过互联网上报的消息数据，对其中的信号进行预处理和分析，转储提取出的信息，并监控各个硬件终端的工作状态。用户端应用界面基于WEB技术设计，实现了多种方式的数据可视化展示，并允许用户异地远程监控设备工作情况。整套系统结合了电子技术，信息处理技术和计算机技术，能一次性部署到位，解决了本地设备数据难以联网所形成的信息孤岛问题，实现了工业设备的云监控和工业数据的云储存，解决了因不同厂商的硬件设备、云服务、可视化平台标准不一，难以协同开发的问题，有很好的实用价值和发展前景。

[ABSTRACT]: This design is a complete set of building scheme for industrial enterprise IOT system, involving equipment data acquisition, signal processing, information analysis and information visualization application, covering the whole process of IOT information collection, processing and display. The program includes a data acquisition hardware device, a connection management software system, and a client application. The data collection hardware is designed according to industrial standards, which can collect various data from industrial equipment and realize access to various analog and sensor signals, and equipped with the latest narrowband Internet of Things (NB-IoT) technology, which has good wireless communication performance. The connection management system is a program running on a cloud server that collects message data reported by each hardware terminal through the Internet, pre-processes and analyzes the signals therein, dumps the extracted information, and monitors the working status of each hardware terminal. The user-side application interface is designed based on WEB technology, which realizes data visualization display in various ways and allows users to remotely monitor the working condition of the equipment off-site. The whole system combines electronic technology, information processing technology and computer technology, and can be deployed at one time, solving the problem of information island formed by the difficulty of networking local equipment data, realizing cloud monitoring of industrial equipment and cloud storage of industrial data, solving the problem of different standards of hardware equipment, cloud services and visualization platforms of different manufacturers, making it difficult to develop in cooperation, and having good practical value and development prospects. .

[关键词]：物联网; 智能硬件; 云服务

**目录**

**1.工业物联网系统介绍.............................页码**

1.1 工作原理和模式................................页码

1.2 国内外发展现状................................页码

1.3 产品特点和创新................................页码

**2.系统全局设计.............................页码**

2.1需求分析..................................页码

2.1.1 使用场景与环境................................页码

2.1.2 产品功能及组成................................页码

2.1.3 成本及安全性................................页码

2.1.4 施工与部署特性................................页码

2.2设计方案..................................页码

1.3.1 信息采集................................页码

1.3.2 云端通信和管理................................页码

1.3.3 系统工作模式................................页码

**2.数据采集终端设计.............................页码**

2.1全局设计..................................页码

2.1.1性能指标................................页码

2.1.2工作模式................................页码

2.2 STM32L431主控模块设计..................................页码

2.2.1初始化配置................................页码

2.2.1.1芯片引脚配置................................页码

2.2.1.2芯片时钟配置................................页码

2.2.2 外部接口................................页码

2.2.2.1 UART配置................................页码

2.2.2.2 SPI配置................................页码

2.3 通信模块设计..................................页码

2.3.1 NB-IoT无线通信................................页码

2.3.1.1 NB-IoT技术简介................................页码

2.3.1.2中移动M5311模组...........................页码

2.3.1.3模组驱动电路设计...........................页码

2.3.2 RS485有线通信................................页码

2.3.2.1 Modbus协议简介..................................页码

2.3.2.2 RS485转换电路设计..................................页码

2.4传感器接入电路..................................页码

2.4.1多通道模拟信号采样电路................................页码

2.4.2.高精度高速信号采样电路.............................页码

2.5外围电路设计..................................页码

2.5.1 电源电路................................页码

2.5.2隔离和防护电路................................页码

2.6固件代码和PCB图纸..................................页码

**3.联接管理系统.............................页码**

3.1全局设计..................................页码

3.1.1 系统工作模式................................页码

3.1.2 软件组成架构................................页码

3.1.3 开发平台与框架...........................页码

3.2设备接入..................................页码

3.2.1.边缘节点接入................................页码

3.2.2 云平台物联网接口接入................................页码

3.2.3云平台直接接入................................页码

3.3数据分析和处理..................................页码

3.3.1数据分析组件................................页码

3.3.2算法定制化开发................................页码

3.4数据转移和储存..................................页码

3.4.1边缘设备缓存................................页码

3.4.2数据共享平台................................页码

3.4.3云端储存................................页码

3.5部署与功能拓展..................................页码

3.5.1云端部署方案................................页码

3.5.2组件升级方案................................页码

3.6 源代码及配置参数..................................页码

**4.客户端应用软件.............................页码**

4.1全局设计..................................页码

4.1.1 系统工作模式................................页码

4.1.2 软件组成架构................................页码

4.1.3 开发平台与框架...........................页码

4.2应用后端开发..................................页码

4.2.1 基于node.js技术的后端开发................................页码

4.2.2 数据流控制................................页码

4.3前端界面开发..................................页码

4.3.1 基于Express的前端开发................................页码

4.3.2 前端交互设计................................页码

4.4代码维护与功能拓展..................................页码

4.5 源代码与配置参数..................................页码

**5.实用案例.............................页码**

2.2.1三级标题(宋体四号)................................页码

2.2.2三级标题(宋体四号)................................页码

2.2.3三级标题(宋体四号)................................页码

2.3三级标题(宋体四号)..................................页码

**参考文献(宋体四号，加粗)...............................页码**

**附录(宋体四号，加粗)...................................页码**

**致谢(宋体四号，加粗)...................................页码**

1. 工业物联网系统介绍

1.1 研发背景

工业为自身和国民经济其他各个部门提供原材料、燃料和动力，是国民经济的重要支柱，在世界各国国民经济中起着主导作用【】。无论是能源采掘工业、原材料工业、或是加工工业，都大批量的使用机械电子设备来协助生产，提高设备的工作效率，优化生产工艺和生产流程是工业企业实现提质、降本、增效的根本手段。自20世纪以来，为了实现对生产过程更高效的把握，越来越多的精益生产模式被人们使用，人们希望通过加强对设备的监控管理，采集更多的生产过程数据来控制生产过程，并提升和优化设备的工作水平。

为实现这一目的，我们需要实时的获取尽可能多的生产数据，各种新技术持续不断的投入来尝试实现这类困难的要求。在信息时代之前，人们通过编写各种纸质的报表来手工记录，但随着电子技术和计算机技术的不断发展，多数生产设备都实现了电子化甚至是数字化，同时人们运用了各种传感器来收集信息，这些数据以电子信号的形式传输并储存在工厂的计算机内，这一进程被称为工业生产的数字化。

但在目前阶段，多数工业企业仅实现了不完全的数字化，甚至还没有开始进行数字化转型。并不是企业主没有意识到数字化管理的必要性，而多是因为企业规模有限，产品简单，不适合大规模机械化加工；或是因为存在必要的人工工序、加工设备较为原始，自动化水平低，难以实现数字化管理。如建材加工厂、电镀厂、包装制品厂、喷漆厂等，多为产能有限、布局分散的小微企业，其中生产设备如吊装设备、搅拌设备、燃气锅炉、各式压力容器等，都必须由工人进行操作。这类设备的复杂程度和自动化水平较低，需要对设备硬件进行升级或加装传感器才能实现参数的数字化读取。当大数据和智能制造时代来临，这些企业迫切需要一种解决方案，来在不破坏原有生产秩序的前提下实现智能化和数字化管理，以享受信息产业高速发展所带来的红利。

本文所介绍的工业物联网数据管理信息系统与终端硬件（以下统称工业物联网系统）正是在这种背景下设计和开发，其所要解决的问题就是帮助上述困难企业实现完全的数字化生产和管理。实现这一目的的一个重要途径是使用物联网技术，即赋予生产设备联网属性，实现系统的智能化感知、识别和管理。工业物联网系统面向传统机电设备和孤立的传感器单元设计了数字化数据采集模块，同时依靠模块的网络通信功能，实现设备与设备间的互联互通，打破各个生产设备间的数据壁垒，在终端汇聚海量的设备数据。

依靠这一强大功能，企业可以以数据为导向有效优化生产过程，高效监管生产设备，获取更多有价值的生产信息，实现精细化管理。此外，工业物联网系统还会融合先进的通信技术以及更强大的云端算力，使用先进的通信技术实现高速可靠的数据传输，同时依靠云计算平台对海量数据进行高效处理，让工厂管理脱离现场，实现电子化生产报表和线上设备监控。

1.2国内外发展现状

近年来关于工业物联网系统的设计研究有着很大的进展，但多是以定制化的形式聚焦于高端产能用户。一些大型的制造业企业，通过自主研发的方式为自己的工厂设计了配套的物联网软硬件系统，如美的公司的MeiCloud系统【】等，但这些系统都是高度定制且不完全开放，很难适用于形态多样的中小微制造企业。对于中小企业来说，依靠自身能力整合软硬件资源，来设计并搭建自己的物联网系统仍十分困难。

为应对企业的定制需求，一线云计算业务服务商如华为云【】、阿里云【】推出了有关工业物联网的产品，内容涉及物联网数据储存、在线数据分析、物联网应用开发等领域。用户在获取第一手原始数据后，可以很方便的上传至云平台，并在云端完成数据转储和分析等业务。但这类云服务产品在数据采集末端没有提供与之配套的即插即用的硬件解决方案。企业在使用这一类产品前，需要从其他供应商购买或自行研发数据采集终端，并自行搭建内部网络，对于尚未实现数字化生产的中小工业企业来说，定制化方案的门槛较高，要实现以上要求较为困难。

对于缺乏自主研发工业物联网系统能力的企业，购买第三方提供的工业物联网产品也是一种可行的解决方案。面向这类需求，一些科技企业推出了定制化的工业物联网产品，如卡奥斯(Cosmoplat)公司的高端智能装备解决方案【】，提供包括智能工厂规划咨询、智能装备设计与制造、系统集成、产线运维等智能化服务。这类定制化施工方案包括了硬件和软件系统的建设，但对于中小微企业来说这类高端解决方案过于昂贵，且不适合规模较小的使用场景。

思科（Cisco），研华（Advantech）等硬件制造商提供了多种适用于工业物联网的硬件设备，如交换机【】、网关【】等，并提供了有关硬件设备接入和管理的软件产品。已经实现生产数字化管理的大中型工业企业可以很方便的通过这类硬件产品实现物联网的组建，但对于尚未实现数字化管理的小微工业企业，这类硬件无法与未实现数字化控制的生产设备连接。这些企业若想享受大数据生产管理所带来的红利，则必须先对现有的生产设备进行一定程度的数字化改造以达到与标准物联网设备进行通信的基本硬件要求。如果仅仅是为了实现厂内设备的联网管理，对整个生产模式和设备都进行修改和更换，这对于管理能力有限的小微企业来说是不可接受的。

综上所述，目前全球关于工业物联网系统的产业布局尚处于一种分散的局面，各家厂商间的产品缺乏互相适配的标准，尚没有形成一种成熟的生态。反观发展较早的生产自动化产业，已经形成了完整的解决方案模式，各大厂家间的产品实现了一定程度的通用。无论是核心设备的研发厂商还是基层的运维、安装服务提供商，都能以一种标准化的工作方式参与其中。未来工业物联网产业终将形成类似的标准化生态，各家厂商分工合作，提供可兼容、功能全面、适用面广的工业物联网软硬件产品，但这一周期是漫长的。我们希望能在这一关键的转型时期为广大中小工业企业提供一个实用性强、成本低、的物联网改造方案。

1.3产品特点和创新思路

结合上文分析，工业物联网产业尚处于发展初级阶段，无论是物联网硬件制造商、物联网应用供应商，还是云计算服务商，都无法提供一种低成本的囊括端设备数据采集、设备间通信、云端接入、管理软件和数据应用软件的工业物联网解决方案。对于多数小微工业企业来说，整合各家的产品并搭建自己的工业物联网系统十分困难，需要专门聘请相关的技术人员进行设计。面向众多小微工业企业的需求，我们需要开发一种低成本的、通用化的工业物联网产品，在使用过程中不用考虑设备通信、数据处理等专业化问题，企业依靠自身的技术力量就能完成安装部署，以极低的前期投入成本享受到完整全面的智能化大数据管理体验。

实现上述目标的一个关键思路是全栈化。全栈（full stack）概念来源于软件工程学，意味着开发者或产品拥有系统的各个层次上理解问题和解决问题的能力，或者说实现和管理系统中各个组成部分的能力。一个全栈化的物联网系统遍布终端和末端，内容包括终端采集硬件、通信设备、管理平台、可视化界面等，一次部署即可实现一个完整物联网系统的全部能力。对于用户来说，设备通信、数据处理等中间环节被封装在系统内部，在部署和施工过程中无需考虑不同供应商产品间的通信方案与协同配合，相比目前需要用户自行购买软硬件组件并进行配置的工业物联网产品，有着复杂度低、易于施工的突出优势。极大的降低了物联网系统使用难度和部署成本。

工业物联网产品难以在小规模工厂推广的一个主要原因是其对原有设备自动化能力要求过高，而小型工厂往往都以人工生产为主，企业也不愿意投入大量资金，冒着成本增加和可靠性降低的风险来升级自动化生产设备。对此，我们希望使用一个易适配的、通用化的数据采集硬件来对现有工厂设备施行无损升级。数据采集硬件并联与工业设备的传感器网络，无需设备本身拥有数控能力；在不改变原设备人工控制能力的前提下，接入设备现有的传感器，实现设备数据联网。这种方案帮助了企业实现生产数据数字化、无纸化监控，同时也不破坏原有的传统生产模式。

1.4工作模式

面向工业设备的物联网系统需要具有三个基本功能需求：1、生产设备与局域网或互联网间的数据传输；2、对设备数据的管理和储存；3、面向设备管理者的操作界面。

分别涉及前期数据获取和采集，中段数据分发与解析，以及末端的数据可视化，其中包含多种软件和硬件设备。根据属性和功能的不同，可以把整个系统拆分为三个模块：数据采集终端、联接管理系统、客户端应用。

数据采集终端是一个安装在设备侧的数据采集和传输装置，它的作用是协助传统工业设备拥有获取自身参数的能力，实现设备信息的数字化读取。数据采集终端需要在电控水平较低的工厂部署，没有现成的设备通信接口使用。所以我们设计了一系列供传感器接入的模拟量接口，如获取设备的启停和荷载信息，可以通过检测设备工作电流实现；获取容器内的介质或环境信息、可以通过各式温度、压力、浓度传感器实现；机械加工中手动机床的加工轨迹、吊车的运行轨迹可以通过光栅或接触开关来获取。这些信息经过相应的传感器转换为模拟电信号，数据采集终端需要收集各种形式的模拟信号，并在内部进行模数转换后进行储存。在通信方面，数据采集终端使用最新的窄带物联网（NB-IoT）【】通信技术接入互联网，无需在厂内布设网线和交换机，也无需安装无线路由器，大大简化了设备部署难度，拥有基本电工知识的工人即可独自完成设备的安装部署。

联接管理系统是一个网络中间件，运行于远程服务器，管理用于终端硬件接入的网络端口，和外部应用程序的连接请求。在端口获取到某个终端设备上报的信息后，联接管理系统内的算法会首先对设备进行鉴权，随后自动对合法信息进行通信协议的转换，并根据设备的不同类型将设备信息分流至不同的储存空间进行储存。此外，联接管理系统还拥有一个数据分析进程，定时读取缓存内新产生的数据，并运行一系列时序分析算法，用于检测设备工作状态、生成有关设备运行情况的多种统计数据。联接管理系统还用于一个上行通信接口同外部应用程序进行通信，外部应用程序可以通过这一接口读取设备信息和数据分析结果。

客户端应用是一个网页程序，通过上文所述的通信接口从联接管理系统中读取所需数据，用户可以在联网条件下通过PC和移动端设备的浏览器打开客户端应用，查看当前在线的采集终端信息，并查看各种生产参数曲线等。

这一系统实现了最基本的工业设备数据获取、处理和呈现的全过程，在此框架上可以进行更深一步的开发，例如同设备制造厂商合作，将数据采集终端的功能集成在设备内部，或者添加更多定制化的数据分析算法进入联接管理系统。也可以开发更多形式的客户端应用，实现更高级的数据可视化效果等等。

1. 数据采集终端设计

2.1 全局设计

数据采集终端用于采集生产过程中产生的各种数据，轻便可靠，易于大批量安装。设备实际上是一种无线联网的数据采集模块，可以收集多种类型的传感器信号，同时支持人工录入数据。该设备可以以无线方式接入互联网，也可使用串行接口接入厂区现有的有线网络。

2.1.1设计指标

数据采集终端的主要设计指标如下表所示：

**表1 主要设计指标**

|  |  |
| --- | --- |
| 项目 | 设计要求 |
| 体积限制 | PCB面积<0.01m2, 外壳< |
| 设备接入方式 | 电流、电压、串行接口 |
| 电流模拟量输入 | 量程4-20mA， 分辨率0.01mA，8通道 |
| 电压模拟量输入 | 量程0-1V，分辨率0.01V，8通道 |
| 有线通信 | RS232，RS485（Modbus） |
| 无线通信 | 窄带物联网（NB-IoT） |
| 安全性 | IP67级防护，防雷击电路，防输入过载电路 |
| 人机交互 | 0.96寸OLED显示屏 |

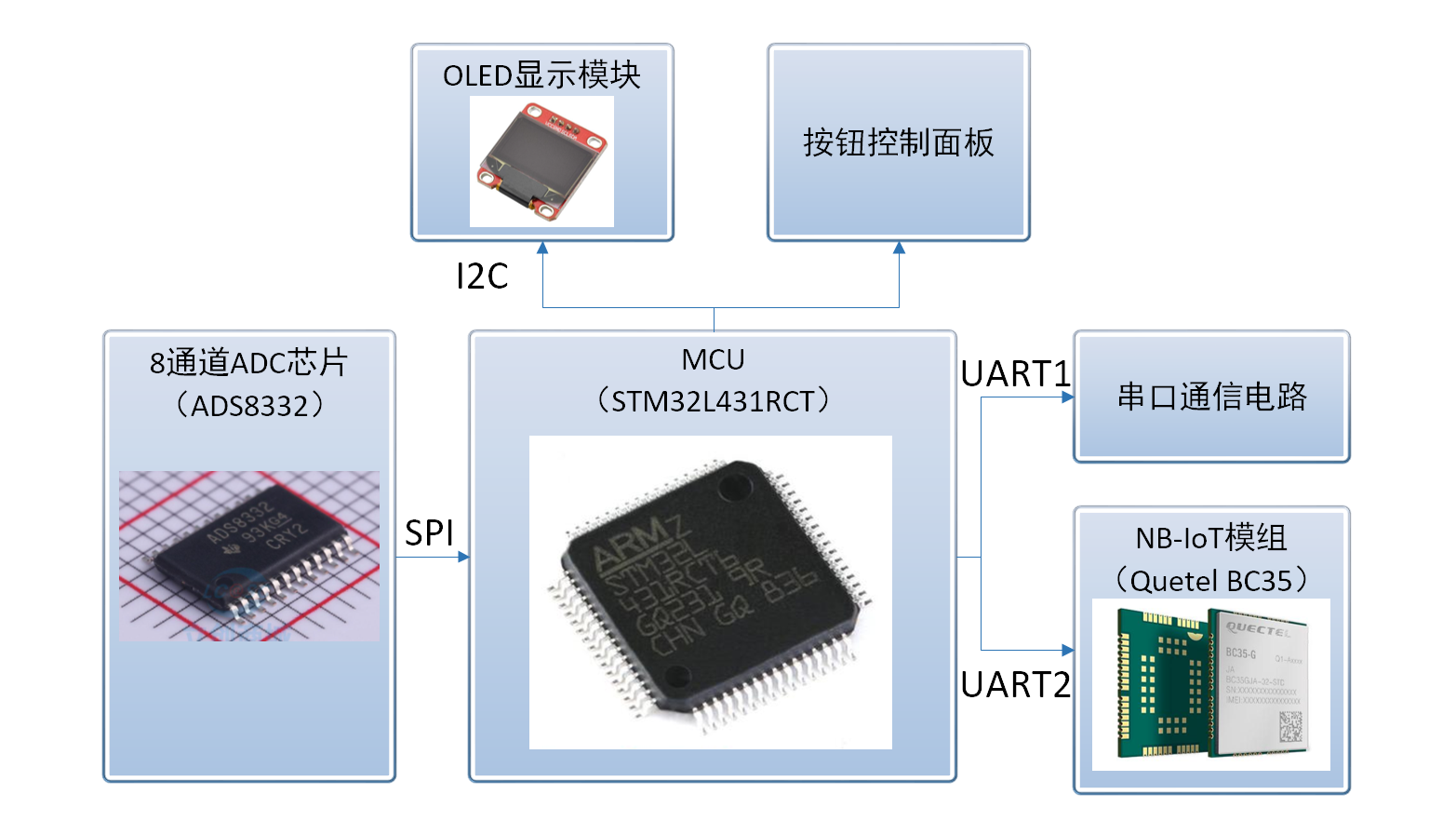
数据采集终端共提供了三种设备接入方式，可以供各种工业传感器或带有RS485串口的工控设备接入。电流接入模式主要面向输出信号为电流形式的工业传感器，如压力传感器、交流电变送器、电磁流量计等；电压接入模式主要面向输出信号为电压形式和开关量形式的工业传感器，如热电偶、光电开关等；此外，为了满足一些较为复杂仪器仪表和可编程逻辑控制器（PLC）的接入需求，数据采集终端亦提供了RS485端口，可实现采集终端与待测设备间的双向通信。

在通信能力方面，数据采集终端提供了两种通信方式，分别串行通信和窄带物联网（NB-IoT）无线通信。串行接口支持RS232或RS485两种接口标准，主要用于调试和固件更新，NB-IoT无线模块负责实现采集终端和云服务器间的数据透传。在人机交互方面，硬件终端提供了一个0.96寸OLED显示屏用于显示设备工作状态和通信参数，此外还配备了三个输入按钮，分别用于设备复位，通信测试和菜单选择，给予了使用者极大的便利。

在安全性方面，数据采集终端配备了防雷击电路和防输入过载电路，可抵御因接地电势剧烈变化导致的电网浪涌冲击，和传感器故障产生的过载电信号，充分考虑了恶劣复杂的工业使用环境，保证终端始终可以可靠且稳定的工作。

2.1.2组成结构

数据采集终端基于嵌入式技术设计，使用低功耗的STM32L4系类微处理器作为控制单元，使用独立的AD采样芯片和通信模组，并配备三色OLED显示屏。数据采集终端采用模块化的形式设计，模块与模块间使用串行接口通信，其组成模式如下图所示：



其中，ADC模块负责将接收到的电压和电流模拟信号转换为数字信号，并通过串行数据接口同MCU进行通信。ADC芯片的型号为德州仪器ADS8332，是一款16位，8通道的ADC芯片，最高采样率为500k/s。无线模块使用移远通信BC35型模组，这是一款专为物联网应用设计的NB-IoT通信模组，体积小，功能丰富。关于具体的设备选型规范和各模块的详细设计方案将在下文进行讨论。

2.2 STM32L431主控模块

主控模块由中央处理器和外围接口电路组成，负责协调各个子模块的工作，收集各个子模块上传的信息和数据，并在内部对其进行整合和处理。主控模块需要实现稳定的不间断工作，对中央处理器和外围元器件的综合性能有着一定的要求。此处选用STM32L431RCT作为主控模块的中央处理器，以满足高计算性能、高稳定性以及低功耗的设计要求。以下是对这款芯片以及主控模块设计的详细介绍。

STM32系列微处理器是目前世界上最为流行的ARM架构微处理器，由意法半导体公司研发，专门面向高性能、低成本、低功耗的嵌入式应用场景【】。STM32L4x1系列属于基本型超低功耗STM32处理器，在满足设计要求的前提下在功耗、性能以及成本之间作了很好的平衡。下表为该芯片的基本属性。

**表1 STM32L431RCT低功耗微处理器性能参数表**

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 值 | 参数 | 值 |
| 型号 | STM32L431RCT | 内核型号 | Cortex-M4 |
| 封装形式 | LQFP | 位数 | 32位 |
| 引脚数 | 64 | 内存大小 | 256KB |
| 尺寸 | 12mil×12mil×1.6mil | 通信接口数量 | 16 |

数据来源：STM32L431xx Datasheet - production data意法半导体

2.2.1初始化配置

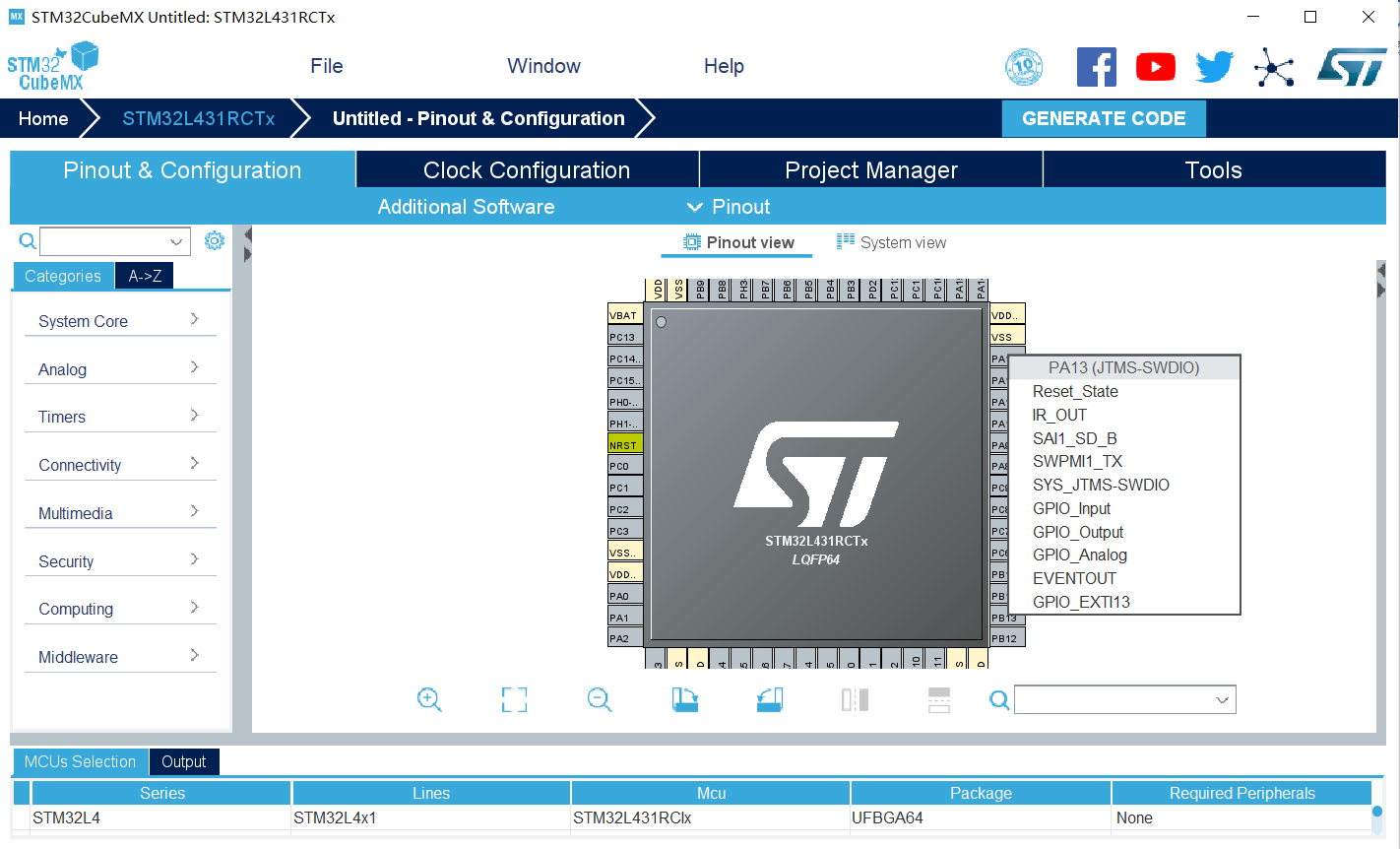
STM32系列微处理器的初始化配置主要是针对芯片的硬件属性，包括芯片引脚配置、时钟配置、外设接口配置等。这些初始化参数以配置文件的形式保存，在编译时以编译参数的形式提供给编译器。开发者可以直接编写C代码配置文件，但需要对MCU硬件结构和原理有着很深入的理解。此外，开发者也可以通过初始化代码生成器来完成STM32的初始化设置，如ST公司推出的代码生成工具STM32CubeMX。

STM32CubeMX是ST公司针对STM32系列微处理器开发的一款初始化代码生成工具【】，它集成了多个软件平台，包括STM32Cube HAL集成库以及TCP/IP,USB,RTOS等通信中间件，支持全系列STM32芯片。STM32CubeMX拥有一个图形化的操作界面用于配置微处理器的初始参数，并自动生成其所对应的初始化C代码及工程文件。开发者可以在生成的工程目录下调用其生成的库函数实现函数式开发，无需关心函数底层的实现。

微处理器的初始化配置项目根据功能不同可以分为三类，分别是引脚功能配置，时钟配置以及外设配置。以下是数据采集模块的微处理器初始化配置说明。

2.2.1.1引脚配置

STM32L431RCT共有64个引脚，除去11个电源引脚和1个RESET引脚之外，其他引脚均支持多种功能，如通信、输入输出、模数转换等，开发者在编写程序前，需要预先指定各引脚的功能，这一过程称作引脚初始化配置。引脚配置信息以c代码的形式储存于工程文件中，开发者可以按照一定的规则直接修改相应的c代码来配置引脚功能，但这种方法十分繁琐且容易出错，对此，STM32CubeMX软件提供了引脚配置代码自动生成功能，与图形化的开发界面，极大的提高了系统开发效率。



芯片引脚的初始化过程主要包括分配引脚类型和配置工作参数。引脚类型包括：通信引脚、输入输出引脚、通信引脚、ADC引脚和其他功能性引脚等。不同类型的引脚存在工作电平、工作频率以及电气特性的差别，必须妥善的进行初始化设定。

通信引脚包括UART（通用异步收发传输器）【】引脚和SPI（串行外设接口）【】，这两者均属于串行通信协议。UART是一种通用串行数据总线，用于异步通信。该总线双向通信，可以实现全双工传输和接收。在嵌入式设计中，UART用于主机与辅助设备通信【Michael M S. Universal asynchronous receiver/transmitter: U.S. Patent 5,140,679[P]. 1992-8-18.】。此处我们设置了两组UART引脚，分别作用于电脑和采集模块、采集模块和Nb-IoT之间的通信。

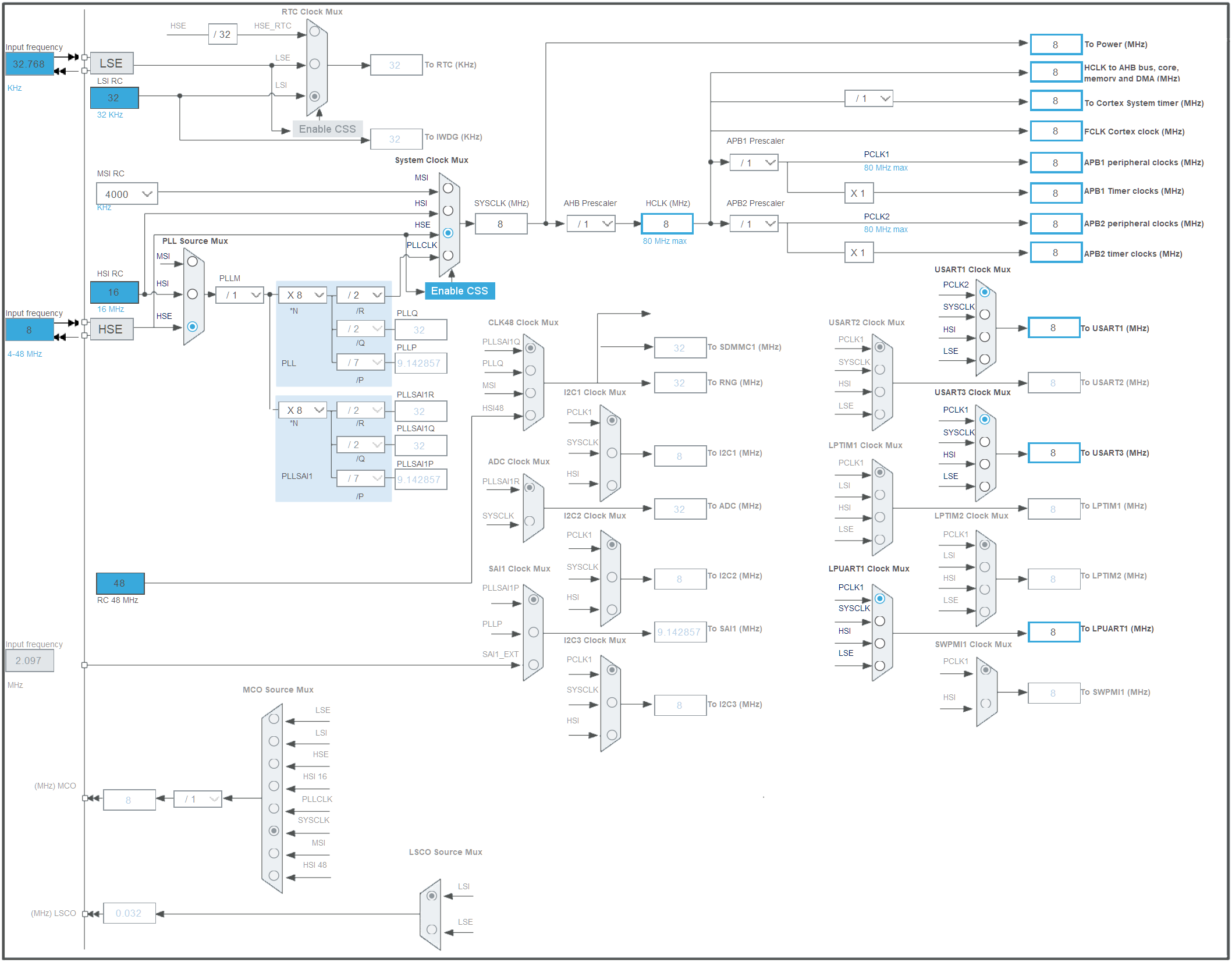
|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 功能 |
| PA9 | USART1\_TX | UART | 外部通信 |
| PA10 | USART1\_RX |
| PC4 | USART3\_TX | UART | NB-IoT模组通信 |
| PC5 | USART3\_RX |
| PB1 | KEY1 | GPIO | 按钮1 |
| PB3 | KEY2 | 按钮2 |
| PA1 | SPI1\_SCK | SPI | ADC芯片通信 |
| PA6 | SPI1\_MISO |
| PA7 | SPI1\_MOSI |
| PB7 | GPIO\_PIN\_7 | GPIO | OLED屏幕虚拟I2C接口 |
| PB6 | GPIO\_PIN\_6 |
| PC13 | LED | LED指示灯 |

**表1 主控芯片引脚功能分配表**

2.2.1.2芯片时钟配置

时钟是微处理器的必备部件之一，为微处理器提供标准的工作频率，推动微处理器内各个部分执行响应的指令。STM32L431采用多时钟源设计，目的是为不同工作频率的外设提供不同的时钟信号，以达到性能和功耗的最优化。这些时钟源最初都由同一个系统时钟信号分频得来，开发人员可以通过修改配置文件来定义芯片内部的分频机制，以产生不同种类外设所需的不同频率的时钟信号。

我们可以通过STM32CubeMX软件的图形化时钟配置界面来快速、直观的配置芯片时钟。下图是STM32CubeMX中显示的STM32L431RCT型芯片的时钟系统图：



图中左上部分显示了STM32的四个独立时钟源，分别为HIS（高速内部时钟），HSE（高速外部时钟），LSI（低速内部时钟），LSE（低速外部时钟），其中高速时钟经过分频或倍频处理可作为系统时钟使用。此处我们使用HSE（高速外部时钟）作为主系统时钟，外部时钟信号由独立的晶振提供，其精度和稳定性优于微处理器自带的内部时钟晶振。在使能HSE（高速外部时钟）后，来自HSE的晶振脉冲经过AHB分频器后变为主时钟HCLK信号，随后被复用为多路，其中一路时钟信号PCLK1将被用于UART1，UART3和SPI等外设。

为了节省系统功耗，我们选用较低频率的8MHz晶振作为时钟源，不经过分频和倍频即可提供给主时钟HCLK和外设时钟PCLK1使用。图中右侧可见各外设的工作频率，都为8MHz，与主时钟保持一致。

2.2.2 内部通信参数

数据采集终端内部各模块使用串行总线进行通信，MUC通过串行总线接收无线通信模块或AD模块发会的数据，也通过串行总线驱动OLED屏幕显示字符内容。数据采集终端共使用了三种不同的串行总线，分别为通用异步收发器（UART）、串行外设接口（SPI）和I2C总线。下面将对这三种不同总线的功能及配置情况做详细说明。

UART全称为通用异步收发传输器（Universal Asynchronous Receiver、Transmitter），俗称“串口“，是一个完成串并转换的硬件，也同时负责数据格式的编码和解析，具有硬件和协议的双重属性【】。USART时UART的低功耗版本，在数据采集终端中共使用了两个USART，UASRT1是数据采集终端的主串口，用于同外部设备通信，如工控主机、PLC等。 USART3用于MCU与NB模块间的通信，传输不同的AT指令。UART的通信规则十分简单，一个数据包即是一个字节流，内部都是有效通信数据，没有定义任何标记位或校验位。由于接收到的报文长度未知，且报文不一定带有结束符，所以必须使用一定机制来判断一段数据的接收是否结束。一种做法是使用超时接收：当串口超过一定时间未接收到数据，即可判断数据接收完毕。STM32在硬件上为此类算法做了专门的优化，在检测到UART总线空闲时，状态寄存器（USART\_SR）中的IDLE（串口空闲中断）位会被置位，并产生一个中断，告知系统一个完整的串口数据帧已被接收。通过在中断程序内编写算法即可实现一个完整串口数据包的读取，其优点十分明显：接收数据的过程不会与主程序产生冲突，不会对运行速度造成较大的影响。以下是两个UART串口的详细配置参数：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 波特率 | 字长 | 校验位 | 停止位 |
| USART1 | 115200 | 8bit | 0 | 1 |
| USART3 | 9600 | 8bit | 0 | 1 |
|  | 全局中断 | 模式 | 流控 | 中断优先级 |
| UASRT1 | 使能 | 异步 | 无 | 1 |
| USART3 | 使能 | 异步 | 无 | 2 |

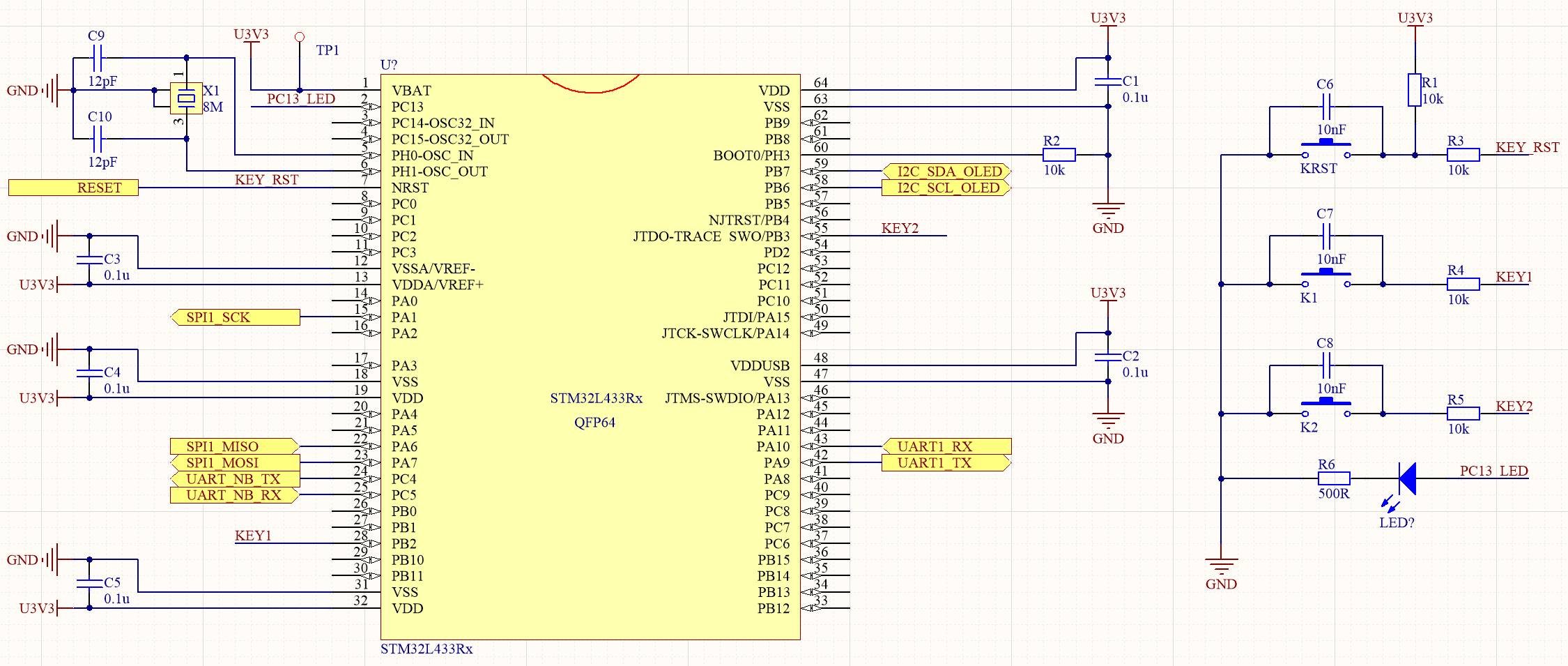
**表1 USART1配置项目**

I2C总线由两条信号线组成，其中一条用于传输数据（SDA），另一条用于传输时钟信号（SCL），它是一种同步通信协议，即通信的AB双方以时钟线（SCL）连接，接收方从机工作在发送方主机传输的时钟信号下。MCU使用I2C协议与OLED屏幕进行通信。关于I2C协议的详细内容不在本文讨论范围内，在此不做过多介绍。

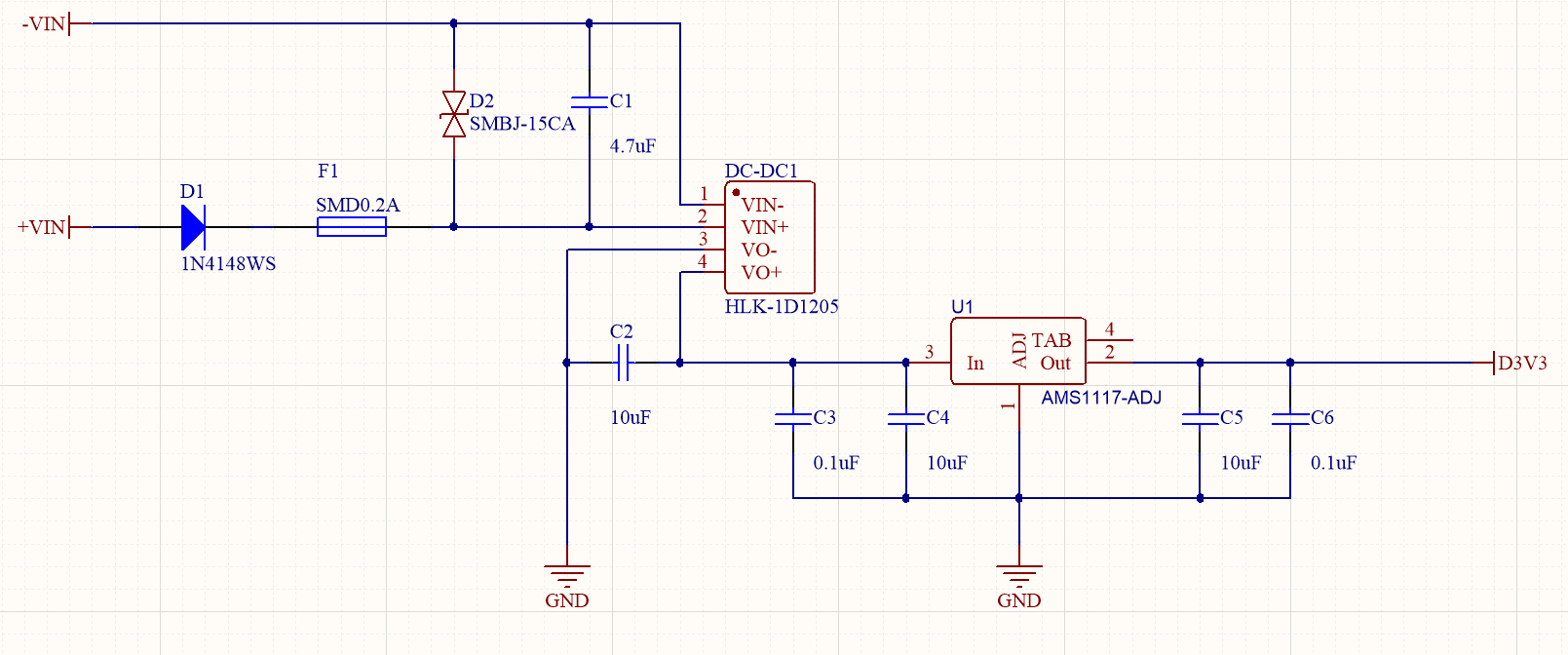
SPI总线用于MCU与ADC芯片ADS8332间的数据通信，

2.2.2 外围电路设计

芯片外围电路需要为芯片提供稳定的电压源与振动源以保证芯片的基本工作需求。图XX是STM32L431RCT主控芯片的外围电路设计图，图中可见起振电路、按钮控制电路和芯片接口配置情况。STM32L431的工作电平为3.3v，共有6个电源输入引脚。由于引脚在电平翻转时会产生尖峰电流，为获得良好的工作效果，每组电源引脚都配备了0.1uF的去耦电容来减小尖峰电流对旁路器件的影响。此外，为缓解按键在复杂使用环境中可能出现的抖动效应，每组按键都并联了大小为10nF的消抖电容。



图XX是数据采集模块的电源电路，额定输入电压12V，输出电压3.3V，如图XX所示，电路由左至右分别为输入保护电路、电压隔离电路和降压稳压电路。输入保护电路具有防过载、防反接、防静电等保护功能，由 1N4148二极管，0.2A自恢复保险丝与SMBJ-15CA瞬态电压抑制二极管（TVS）组成。1N4148与自恢复保险丝串联构成了一个基本的防反接和防过载电路，随后在输出末端并联反向击穿电压为15V的TVS二极管，在极端情况下可以瞬间反向导通，保证输出电压被钳位在安全范围内。为了实现更好的隔离效果，在保护电路末端添加了额定功率为1W的DC/DC模块电源HLK-1D1205，可以实现输入端与输出端之间的电压隔离，进一步提升抗干扰性能。由于HLK-1D1205的输出电压较高且精度存在较大偏差（5V15%），在模块电源输出末级又添加了三端稳压集成电路AMS1117-3.3，最终输出稳定的3.3V电压。



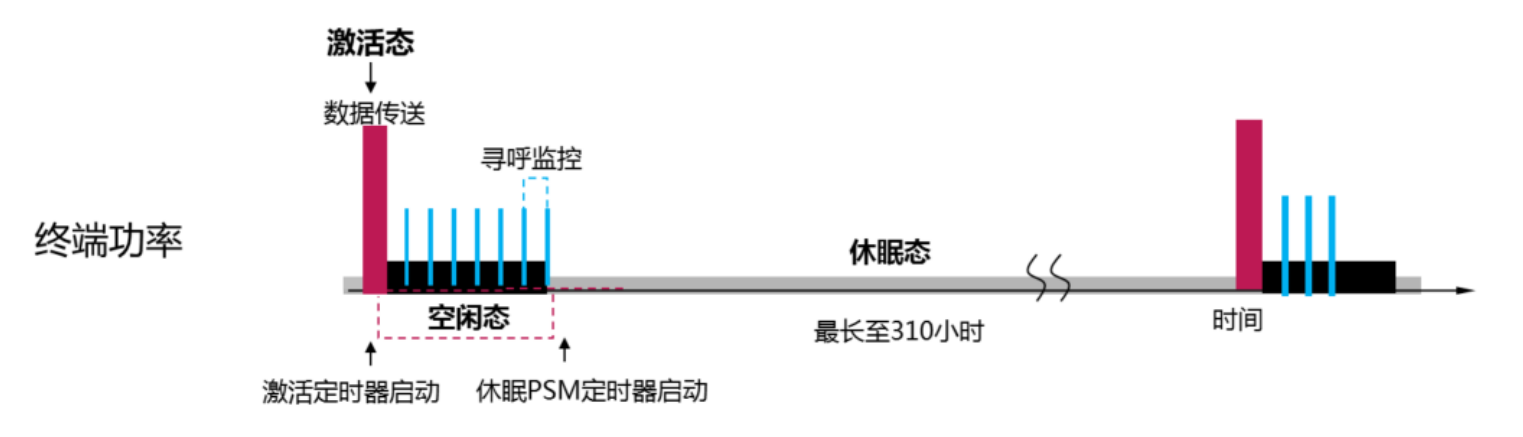
2.3 通信模块设计

2.3.1 NB-IoT无线通信

2.2.1.1 NB-IoT技术简介

NB-IoT（Narrow Band IoT）是一种构建于蜂窝网络中的新兴通信技术，在2017年实现商用，主要用于低功耗广覆盖的物联网应用中。NB-IoT的特点是覆盖范围广，功耗低，稳定性高，成本低，支持LTE制式平滑演进，可以根据不同运营商的需求灵活调整频段和部署模式。

NB-IoT的物理层实现与LTE类似，但更加精简。其上行技术使用SC-FDMA（单载波频分多址），下行技术为OFDMA，系统带宽为180kHZ，可以布置于LTE系统的保护带或边缘无用的频带，在手机信号覆盖范围内都可以正常通信。此外，NB-IoT专门针对物联网的严苛应用场景进行了算法优化，添加了重传机制，加上较窄的带宽，相比GSM有20dB+增益【华为liteos p190】，拥有极强的穿透性且可以覆盖更广的地域。



来源：华为技术有限公司

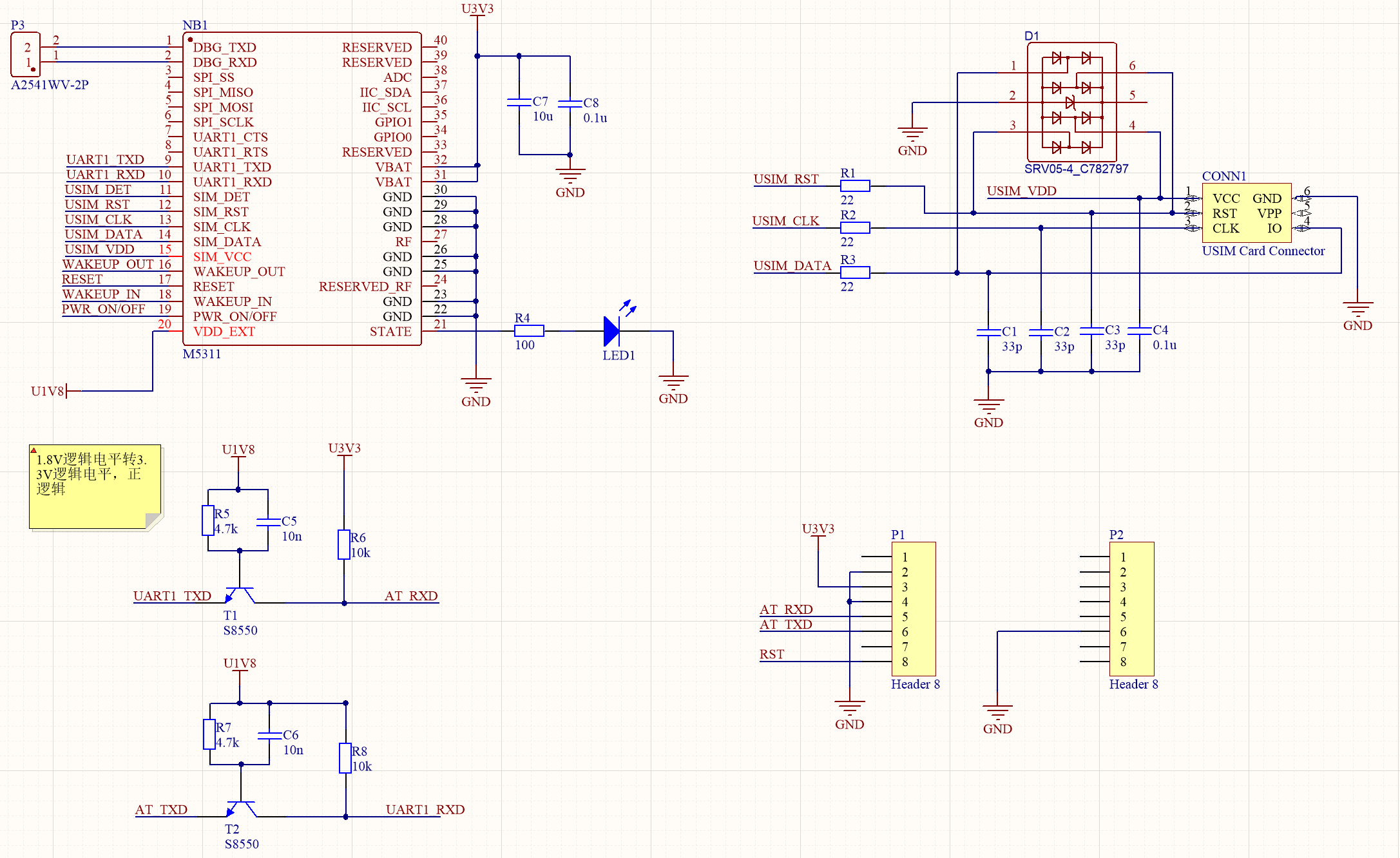
NB-IoT主要依靠动态休眠的通信模式来实现超低功耗。不同于手机需要随时侦听网络中可能发起的请求，NB-IoT模组在发送数据包后，会立刻进入休眠状态，直到下一次有上报数据请求的时刻再开始新一轮的通信活动。由于物联网数据采集的行为习惯，多数时间模组都会处于休眠状态，此时的功耗将非常低。此外3GPP组织还定义了NB-IoT的空闲态eDRX（扩展非连续接收）功能，将寻呼周期从传统的2.56秒拓展到2.92小时，减少寻呼行为所造成的功耗【】。

对比其他可能的物联网通信方案，NB-IoT有非常显著的优势。相比有线通信方案，无线通信方案极大的降低了组网成本，且可以灵活布置终端机的位置，新终端机也可以不受基础硬件的限制快速的部署上线。在无线通信方案中，NB-IoT基于现有的蜂窝移动网络，覆盖范围广，网络质量好。对比WiFi方案，NB-IoT不需要在厂区内部布设有线网络和无线路由器；对比传统的无线电通信方案，NB-IoT不需要配置高功率的发射机也无需申请合法的通信频道许可；对比LoRa（Long Range Radio远距离无线电）方案【】，NB-IoT无需在厂区内部自行搭建LoRa网络，也不需要配置专门的网关设备。总而言之，NB-IoT是一种十分合适的工业物联网系统通信手段。

2.3.1.3 模组驱动电路设计

数据采集终端使用移远通信BC-35G模块接入NB-IoT网络。BC-35G是移远通信（QUECTEL）于2018年发布的一款单频段LTE Cat NB2 模块，使用海思Boudica 150芯片【】。此款模块在移远通信BC系列NB-IoT模块中属于中端型号，支持450-470MHz，696-960MHz，1695-2180MHz等多个通信频段，且支持CoAP/LwM2M/TCP/MQTT等多种物联网常见协议。BC-35G使用了华为海思的基带芯片，已通过华为物联网平台接入认证，相比其他厂商生产的NB-IoT模块，BC-35G可以直接使用内置AT指令接入华为物联网平台，无需复杂的接入认证程序。

为保障BC-35G模块的正常工作，需要设计相应的外围电路，如供电电路、串口电路、天线电路等，此外由于BC-35G使用外置USIM接口，设计者还需要单独设计外置USIM卡电路，以保障模块正常入网。由于BC-35G模块高度集成化，体积较小，耐压和抗静电能力较弱，外围驱动电路需要优异的抗干扰性能以保证其在工业环境下的正常工作。



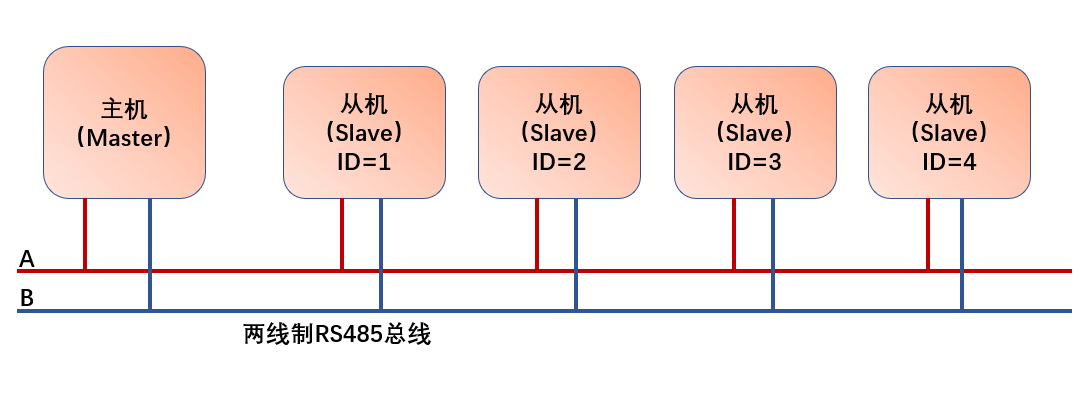
**图1 XX图**（中文黑体，英文为Times New Roman五号加粗，居中）

* + 1. RS485有线通信

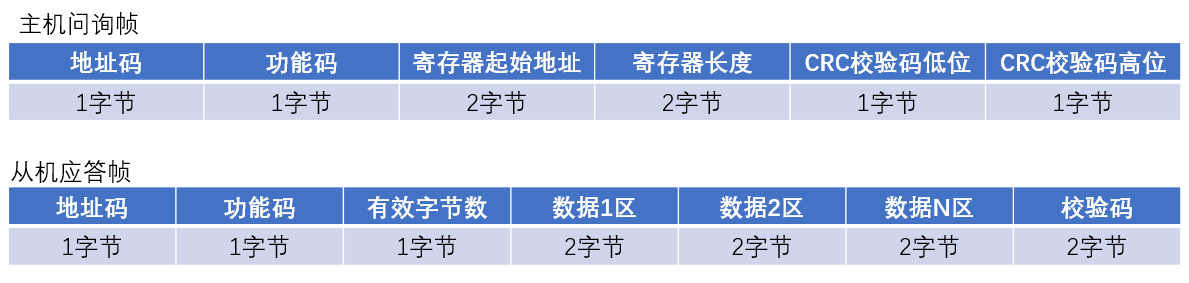
2.3.2.1 RS485及Modbus协议简介

RS485是一种工业界常用的串口通信标准，因其布线简单、抗干扰能力强、通信距离远，多数工控产品都会配备支持RS485的串口接口用于设备间通信。RS485最初的研发目的是为了替代过时的RS232标准，不同于RS232的电平信号，RS485使用差分信号传输字节流，不需要一个基准电势作为参考信号，具有更强的抗干扰能力，传播距离也大大增长。此外RS485标准结合Modbus协议还可以实现多设备组网，使用总线拓扑结构时最多可以连接32个子节点，相比传统串口的单点通信方式有着很大的性能提升。

RS485标准通常搭配Modbus协议使用，Modbus协议是工业领域通信协议的业界标准，是工业电子设备之间最常用的连接方式之一。Modbus支持多种传输介质，最常用的是使用以太网进行传输的Modbus TCP以及使用RS485总线进行传输的Modbus RTU。在Modbus是一种主从形式的通信协议，在一条总线上配备有一个主机（Master）和多个从机（Slave），在实际应用中，主机对应区域内的核心工控电脑，从机对应区域内各设备的控制单元。图XX描述了一个拥有1个主机和4个从机的Modbus通信网络。



Modbus协议是一种十分轻量化的通信协议，以使用RS485标准传输的Modbus RTU协议为例，其通讯格式如下：



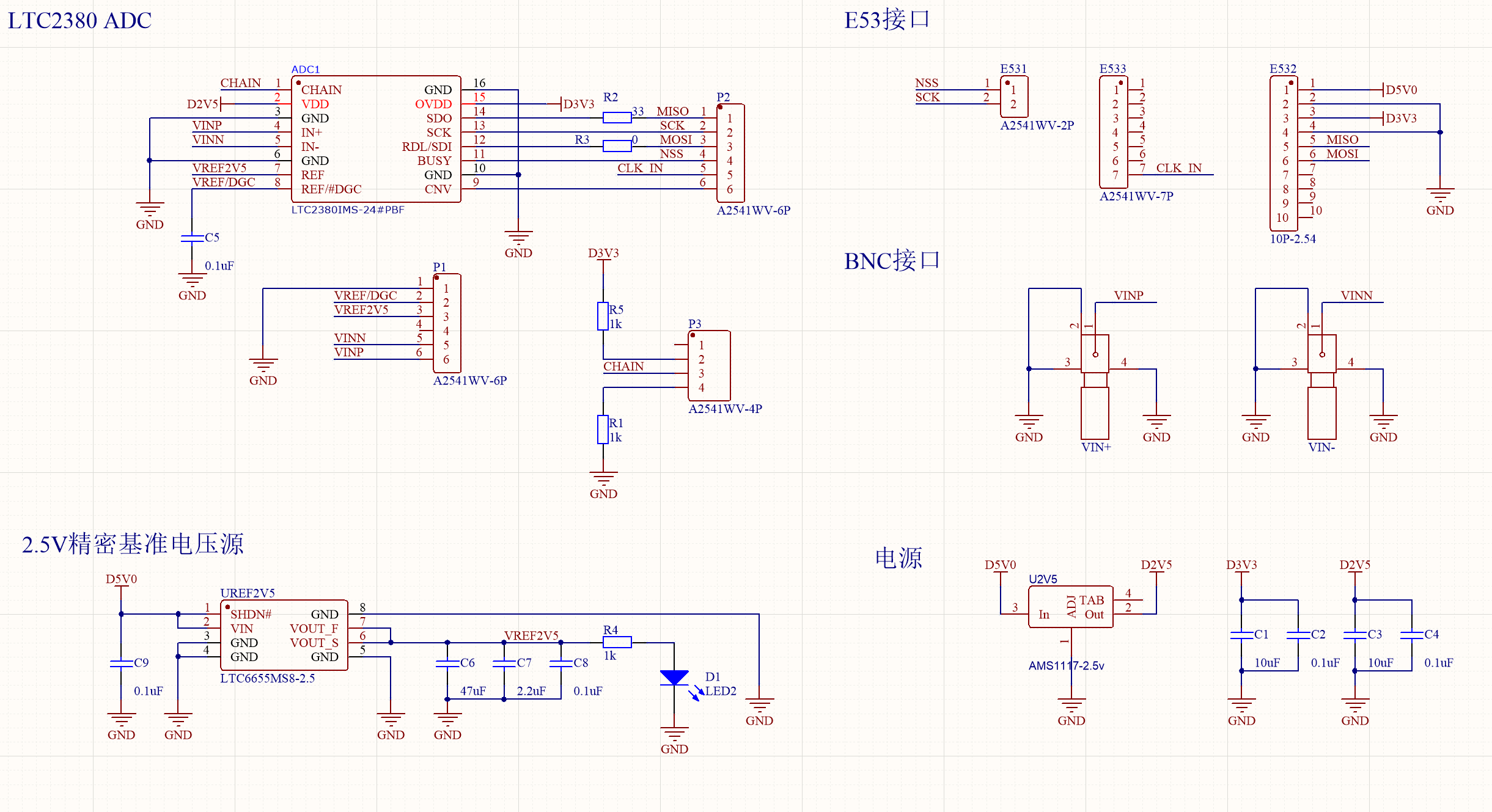
其中，地址码为从机的地址标识，在一个通信网络中是唯一的，由设计者决定；功能码是主机所发的功能指示，如读写寄存器等；数据区是通信具体数据；CRC码是使用CRC算法生成的两字节长度的校验码。我们将在本节后半段介绍Modbus协议在嵌入式设备中的具体实现算法。

2.3.2.2 RS485转换电路设计

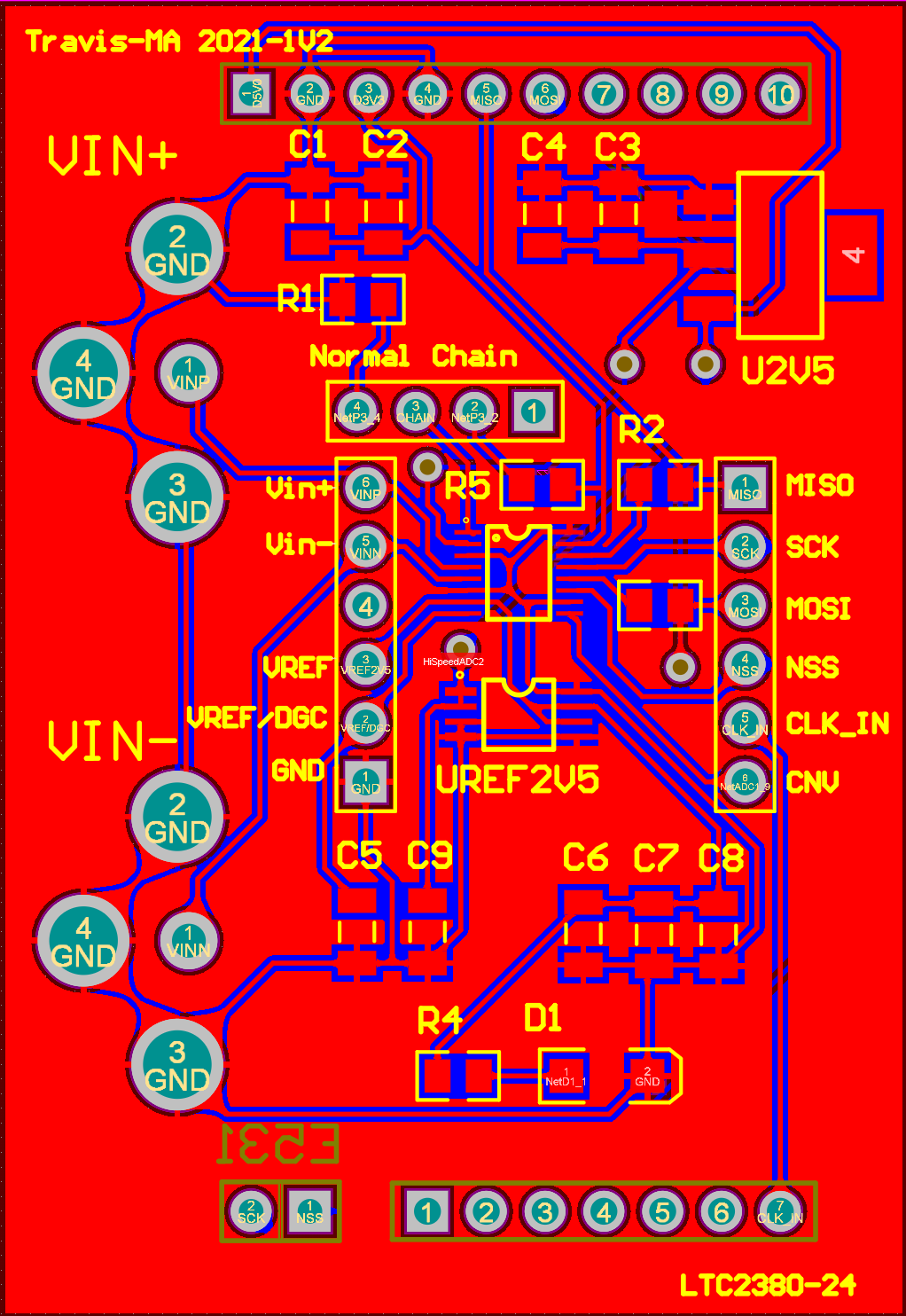
RS485标准使用差分通信，且工作电平与UART电平存在差异，需要专门设计一种RS485转UART电路。

2.4 传感器接入电路

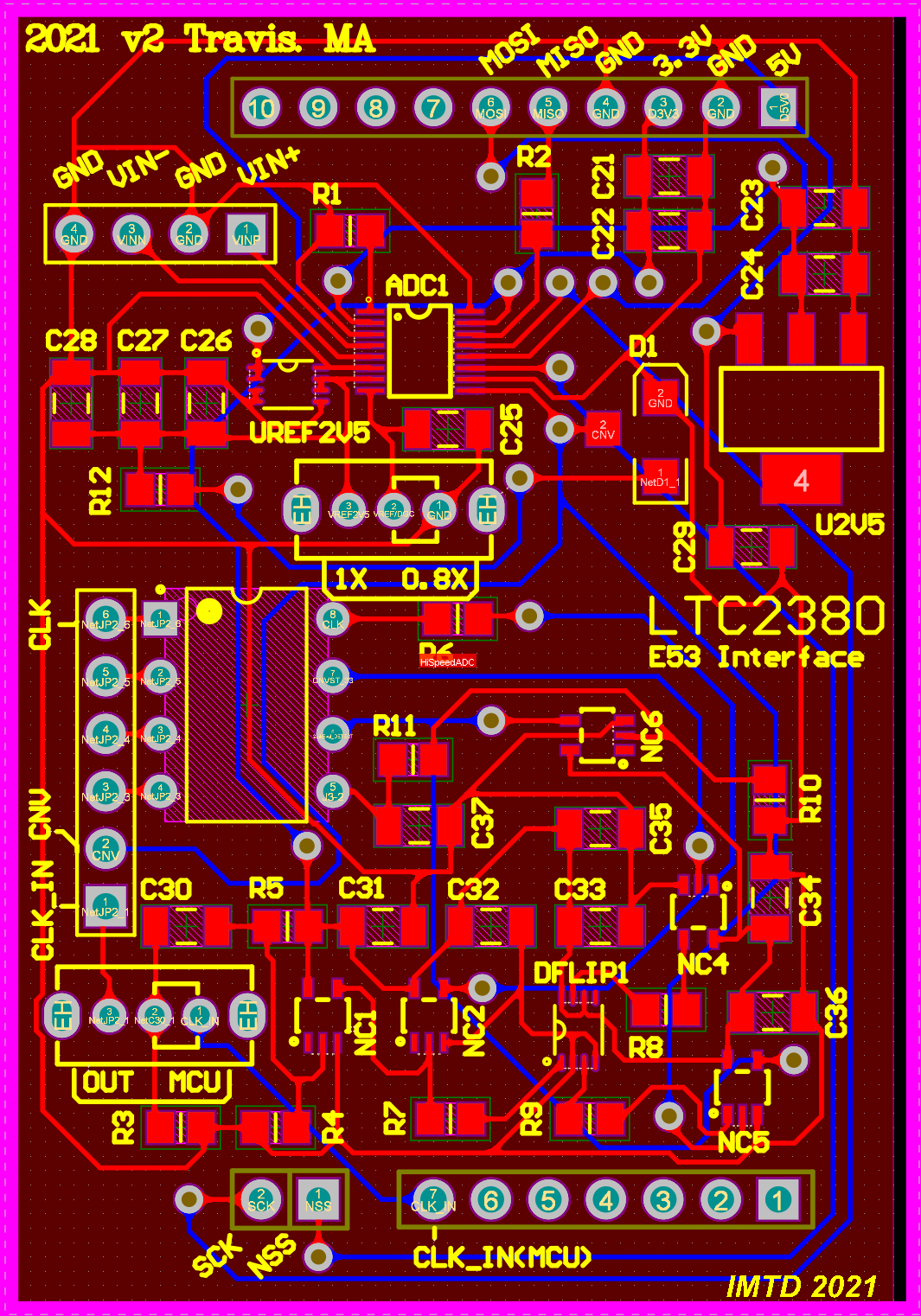
数据采集模块有8路传感器接口，通过接口上的开关可以设置电压接入模式和电流接入模式，其中电压信号范围为0-1V，电流信号范围为4-20mA。数据采集模块使用了独立的模数转换（ADC）模块来实现高精度的模拟信号采样



**图1 XX图**（中文黑体，英文为Times New Roman五号加粗，居中）



**图1 XX图**（中文黑体，英文为Times New Roman五号加粗，居中）



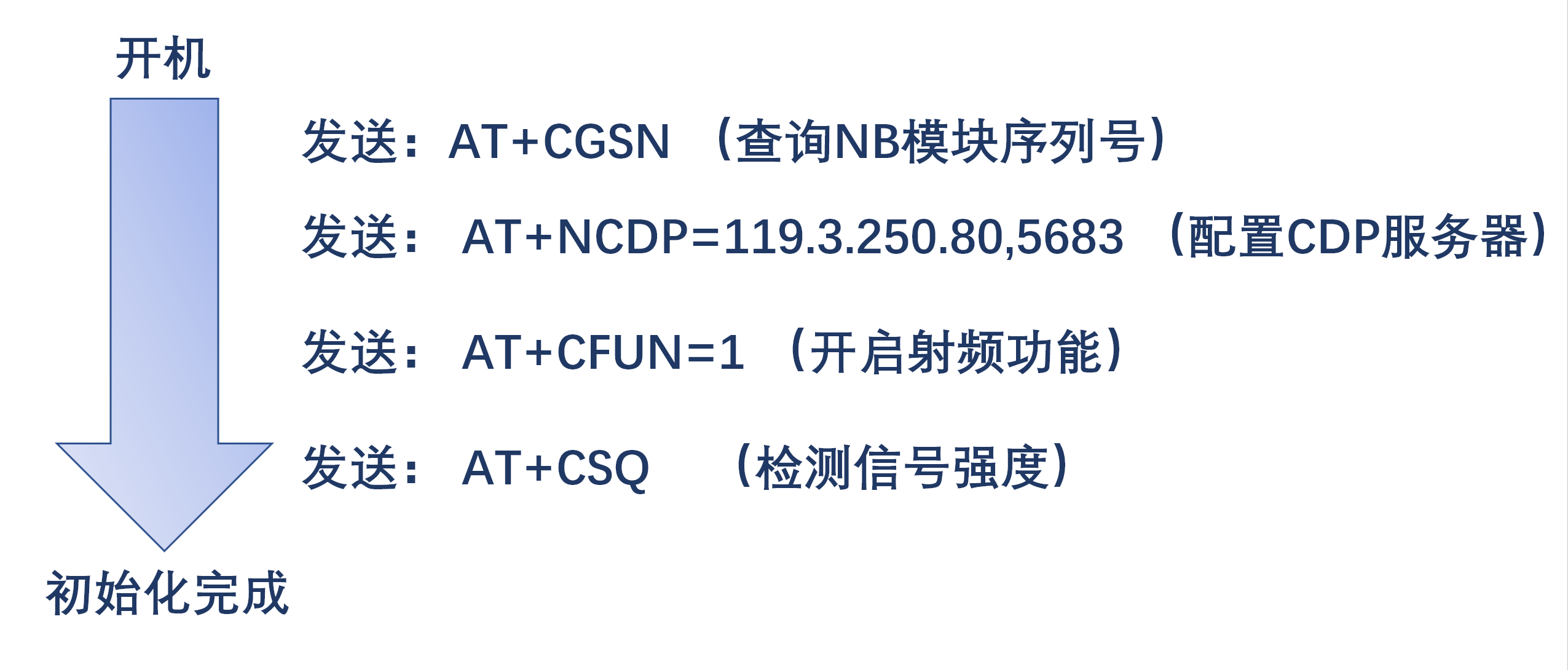
2.6 工作机制

3.1.1模拟信号采样

联接管理系统所要实现的具体功能有如下：1、与运营商网络连接，定时下载转储数据采集终端

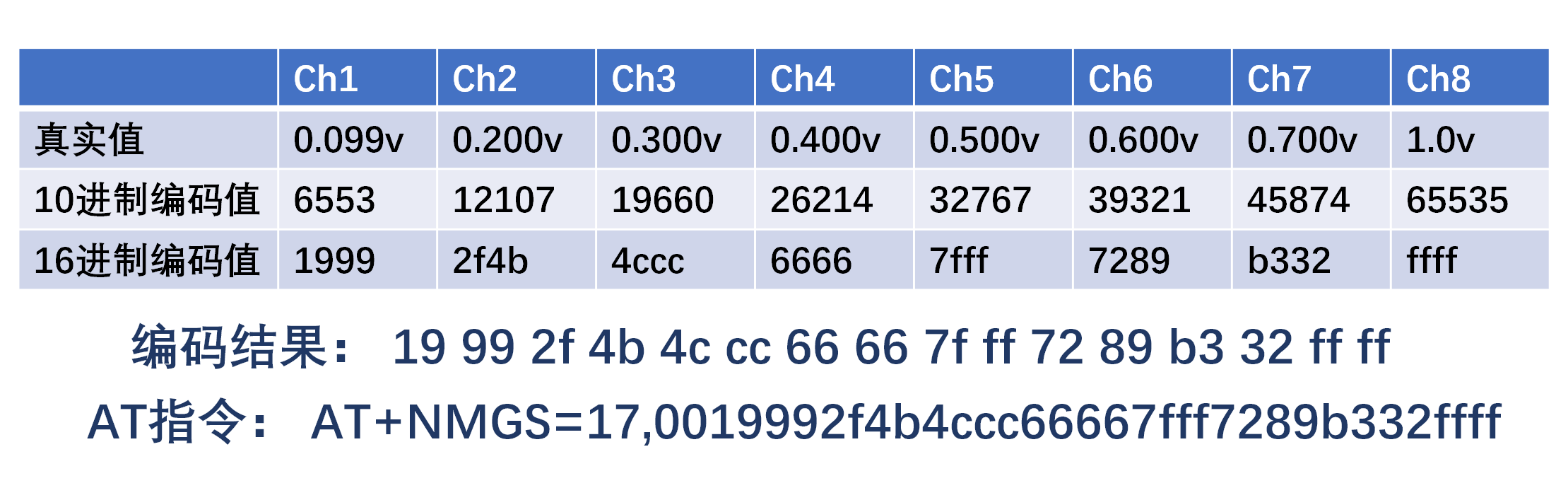
3.1.1无线通信

MCU与NB-IoT模块间使用串口USART3连接，通过AT命令来控制与NB-IoT模块间的交互行为。在数据采集模块开机后，NB-IoT模块开始初始化，这个时间约为10秒。MCU会在NB-IoT结束初始化后发送AT指令读取NB-IoT模块的内部参数，验证初始化是否成功。下图展示了数据采集终端开机后无线通信模块的初始化过程：



其中IP地址119.3.250.80：5683是华为云设备接入服务CDP服务器的CoAP协议接入地址。发送“AT+CSQ”查询信号强度将返回模块从基站接收到的信号强度指示RSSI和信道误码率BER，这些参数将被显示在OLED屏幕上供用户参考。

数据采集终端使用AT指令将各通道的采样值发送至CDP服务器，其命令格式为：“AT+NMGS=<length>,<data>[,<seq\_num>]”其中length字段是发送数据的长度，data字段是字符串类型的待传输数据，其格式为16进制。每个通道的采样值被编码为2字节供4位的16进制数据，最大值为ffff，共有8个通道，数据字段的总长度为16字节。编码样例参考图XX。



3.1.1 Modbus通信

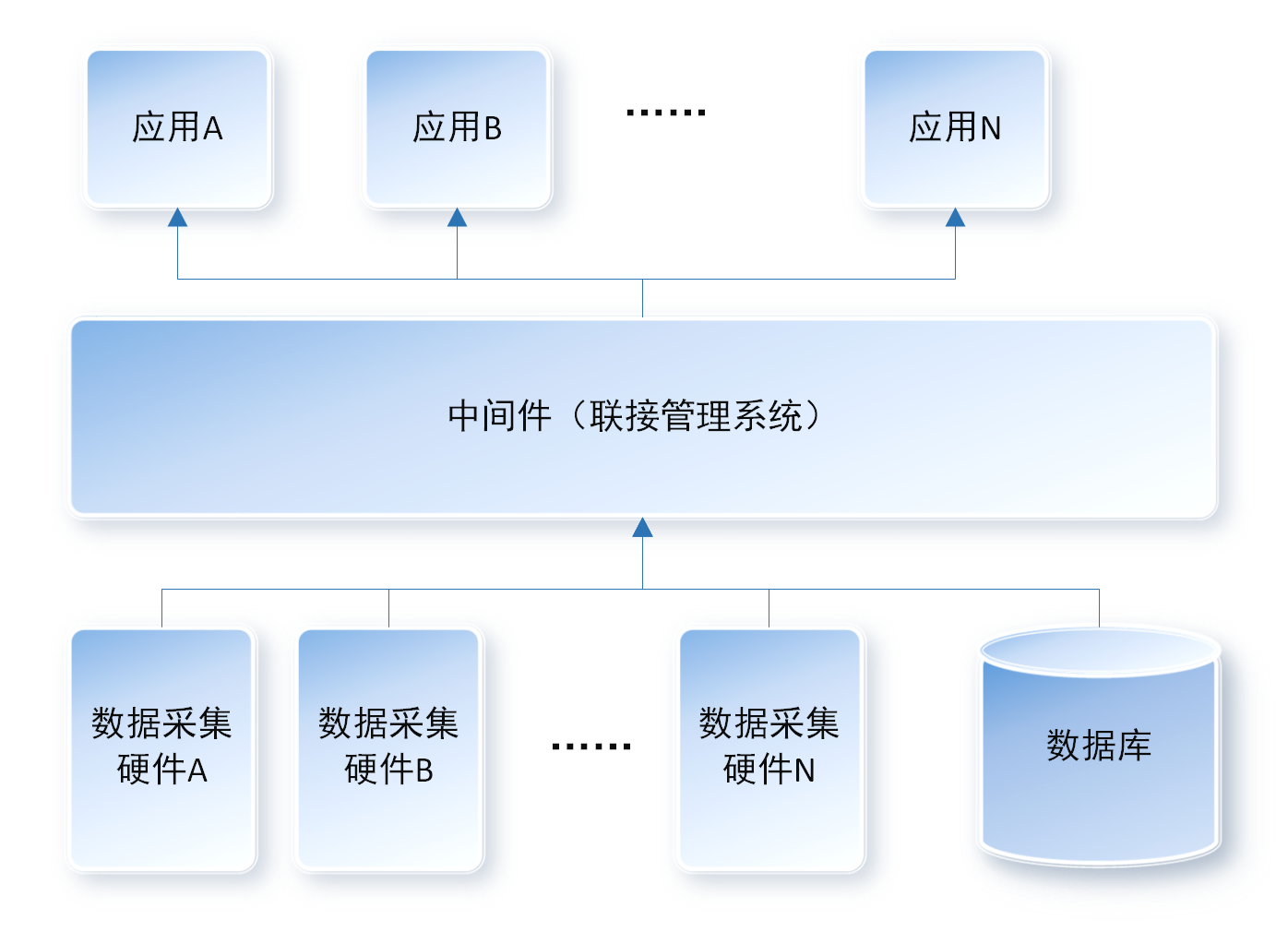
联接管理系统所要实现的具体功能有如下：1、与运营商网络连接，定时下载转储数据采集终端

2.6 固件代码和PCB图纸

中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。

1. 联接管理系统

联接管理系统是一个运行在云服务器上的程序系统，提供虚拟网络接口供分布在各地的数据采集终端接入，并对其上传的数据进行分析和管理。可以将联接管理系统视为一个中间件【】，即一种独立的系统软件或服务程序，位于终端硬件、基本网络服务和数据库之上，在应用软件之下，衔接网络上应用系统的各个部分或不同的应用。联接管理系统这一中间件正是承担了数据终端和客户端界面间的信息转发和处理工作，为数据终端硬件提供了通信支持，也为上层应用软件提供了获取信息的接口（图XX），能使物联网数据的整合和分析任务在一个相对封闭和独立的且易于实现高性能的计算空间中完成。

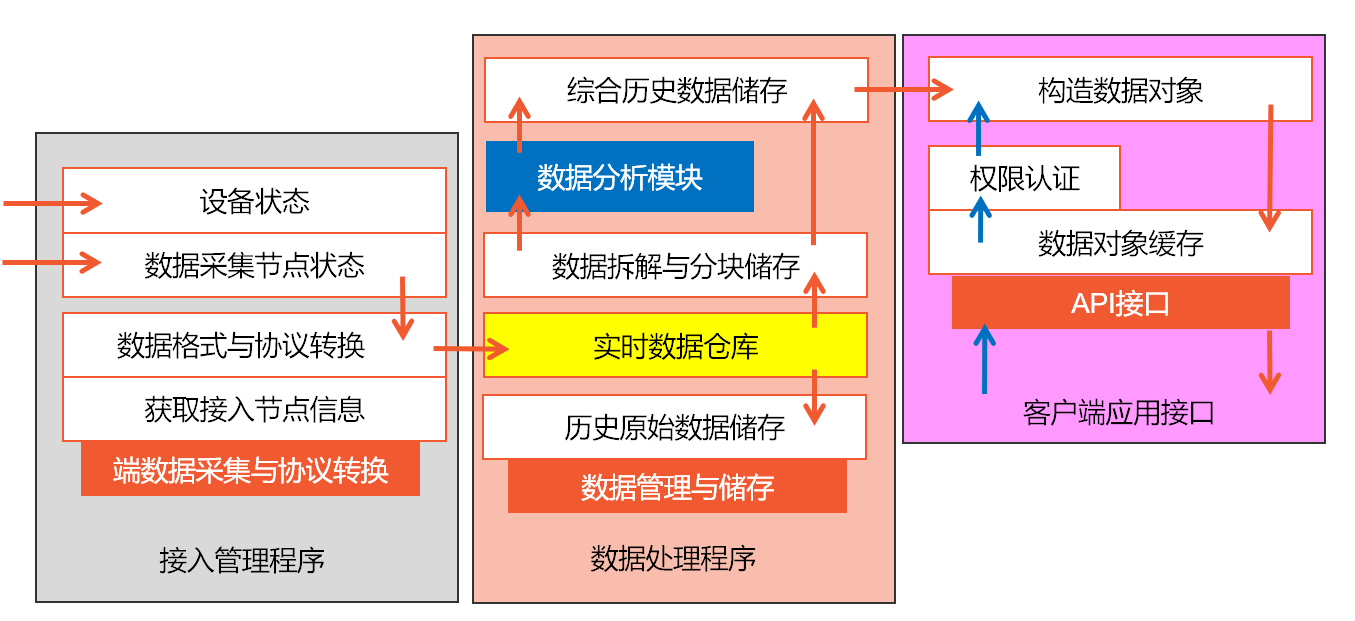


3.1 全局设计

3.1.1系统工作模式

联接管理系统所要实现的具体功能有如下：1、与运营商网络连接，定时下载转储数据采集终端上报的信息；2、对终端数据进行数据格式和协议的转换；3、按照一定规则对数据进行拆解和分块储存，并做双重备份；4、对物联网数据进行实时和非实时的分析。 5、提供面向应用侧的API接口，支持客户端应用程序在线获取信息。每一个功能都拥有其对应的工作模式，都涉及到多个软件的协同工作。

以上功能被拆分为三个主要程序，分别为接入管理程序，数据处理程序和客户端应用接口程序，三个主程序异步运行，互不干扰。下图描述了数据包在三个主程序之间的传递过程和程序内部的工作模式。



接入管理程序通过监听一个特定的网络端口来获取数据采集终端定时上报的数据，数据采集终端通过NB-IoT网络与互联网连接，按照预设的上报周期向联接管理系统的设备接入端口发送按照Modbus协议编码的字符串，字符串中包含有数据值，设备识别码，数据类型，采集时间等参数，在被联接管理系统捕获后，按照一定规则将数据转储至云端缓存中。使用Modbus协议进行编码的好处是它拥有较小的信息冗余，可以大幅减小数据包的大小。此外，接入管理程序在接收数据时会对Modbus字符串进行解析，并以json格式的字符串保存，使用json格式将有利于后期信息的处理，因为多数高级程序语言都有针对json格式开发的插件。

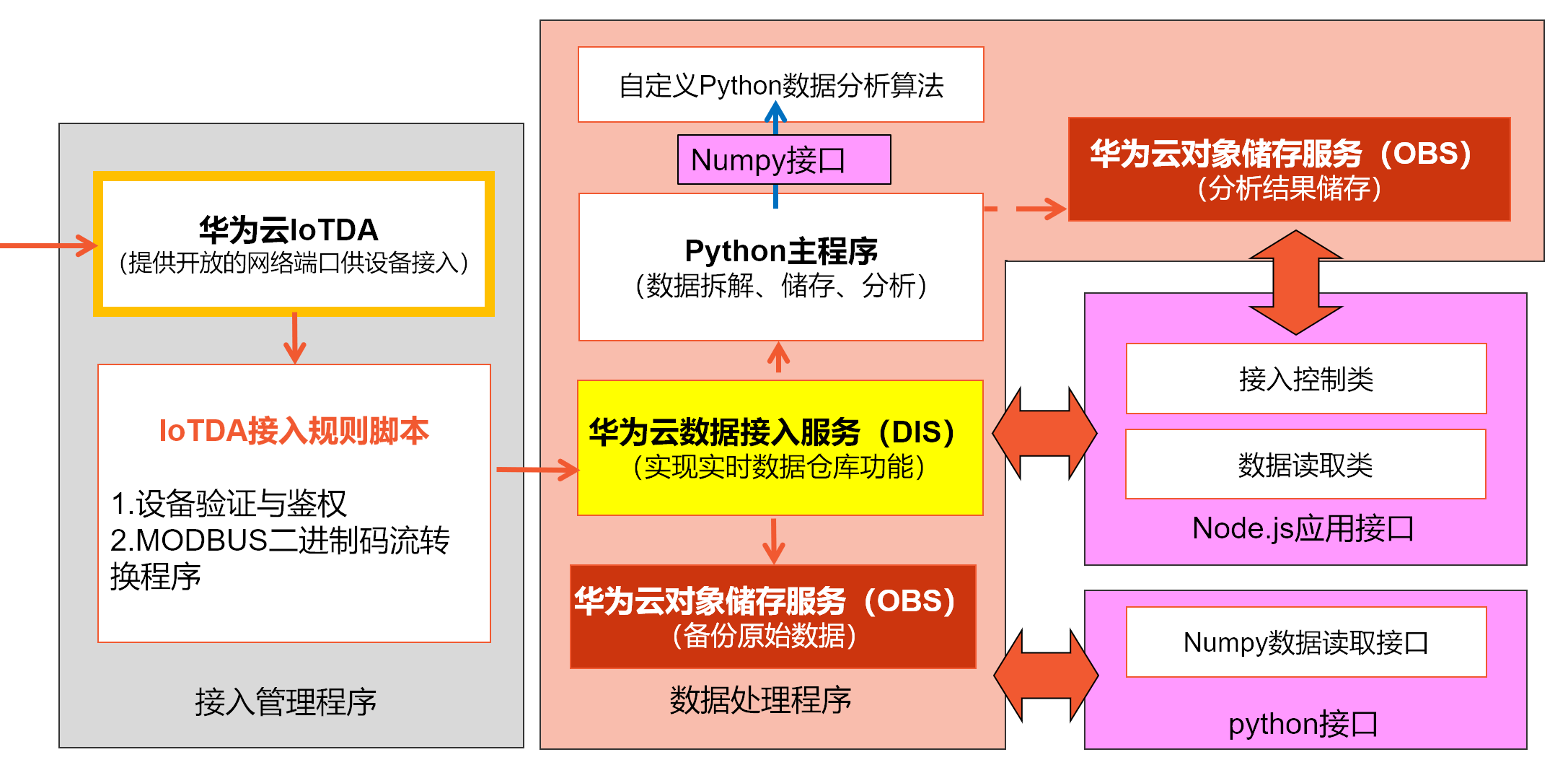
数据处理程序会定时的读取缓存中的json字符串，按照数据记录的发生时间和信息类型拆分数据包，并储存至实时数据缓存中。数据经过拆解和分类储存后，后期数据处理进程就可以直接按照类型索引在数据库中查找有关数据，而不用遍历原始的未拆解的json字符串来筛选信息，可以降低程序的复杂性并提升运行速度。此外，数据处理程序亦可以对物联网数据进行分析，通过调用一个独立的数据分析模块来实现。开发者可以在数据分析模块中实现各种分析算法，如对实时数据缓存中的数据进行时域上的分析，包括根据特征识别不同的生产阶段，并计算出每一个生产阶段中数据的各种统计量。经过分析处理后的结果由数据处理程序上传至云端储存器长期储存。

客户端应用接口部署于应用侧，拥有实时数据缓存和云储存的访问权限，可以根据客户端应用的需求调用所需数据资源。

3.1.2软件组成架构

软件采用了一种扁平化、低耦合、低成本的架构设计方案。其中扁平化的特点体现在系统内各个模块都是以一种平等的方式协同工作，不存在中央控制机制。这种平等协同工作方式的实现依赖于各个模块都设置了输入和输出的缓存机制，使得各个模块可以按照不同的工作周期独立工作，而不是等待上一步完成后再被前一个进程调用。因为其弱化了进程间的相互控制和依赖，所以这种方式可以很大程度降低模块间的耦合度。这种扁平化、低耦合的架构十分适合物联网数据处理，因为各个模块独立工作，每个模块的工作机制相对简单、易于实现高可靠性，可以满足较长时间的无托管运行。这种架构也存在缺点，即整个系统响应速度受模块间缓存影响较为缓慢，但由于物联网数据更新周期长（秒级），对实时性要求不高，因此采用本架构并不会对软件综合性能产生较大的影响。

软件组成架构如图XX所示，除去数据分析算法模块与主程序存在调用关系外，其他模块间都是以缓存的形式进行异步的数据交互。在具体实现方案中，我们使用了多种商用技术，如开发接入管理程序使用的华为云IoTDA【】平台，华为云DIS【】和OBS【】储存服务等，关于它们的详细介绍将在下文给出。这些开发平台框架在很大程度上减轻了开发压力，同时具备较高的性能和稳定性，且商用数据接入接口和云储存服务相比自建服务器有着明显的成本优势。



整套系统被部署于由华为云提供的远程服务器上，其中接入管理程序在华为云IoTDA的自有服务器运行并接受管理，python主程序与数据分析算法在一台单独的云服务器上运行。根据前文所述的缓存耦合机制，所有模块间的数据交换都需要通过云储存介质，即模块输入和输出数据都是从云储存介质中提取，两个模块间无直接的数据传输通道，当其中一个模块出现故障时，其他模块仍能从云储存介质中提取数据保持自身的正常工作。如当python主程序阻塞时，接入管理程序仍能正常转储设备上报的物联网数据，保证数据不会丢失。

3.1.3开发平台与工具

联接管理系统包括多个程序组件，分别实现设备接入、数据储存、数据分析等软件功能，实现这些功能依赖于一些商用软件开发平台和工具。其中具有代表性的，如华为云对象储存服务（OBS）与数据接入服务（DIS）在时序数据的高效转储中发挥了主要作用，以下将进行简要介绍。

对象储存服务（Object Storage Service，简称OBS），是华为技术有限公司开发的一款稳定、安全、高效、易用的云储存服务，其特性是可以储存任意数量和形式的非结构化数据【】。使用OBS服务来实现物联网数据的云储存相比于本地储存更为安全，且支持多协议访问。其安全性体现在于OBS中的特定数据由访问密钥（AK/SK）保护，无关用户在未经管理员授权的情况下无OBS的访问权限。它还提供了基于HTTP/HTTPS协议的Web服务接口，用户可以随时随地连接到Internet的电脑上，通过OBS管理控制台或各种OBS工具访问和管理存储在OBS中的数据【】。联接管理系统基于OBS服务实现物联网数据的长期储存，关于数据储存的机制将在3.X“数据储存”中详细说明。

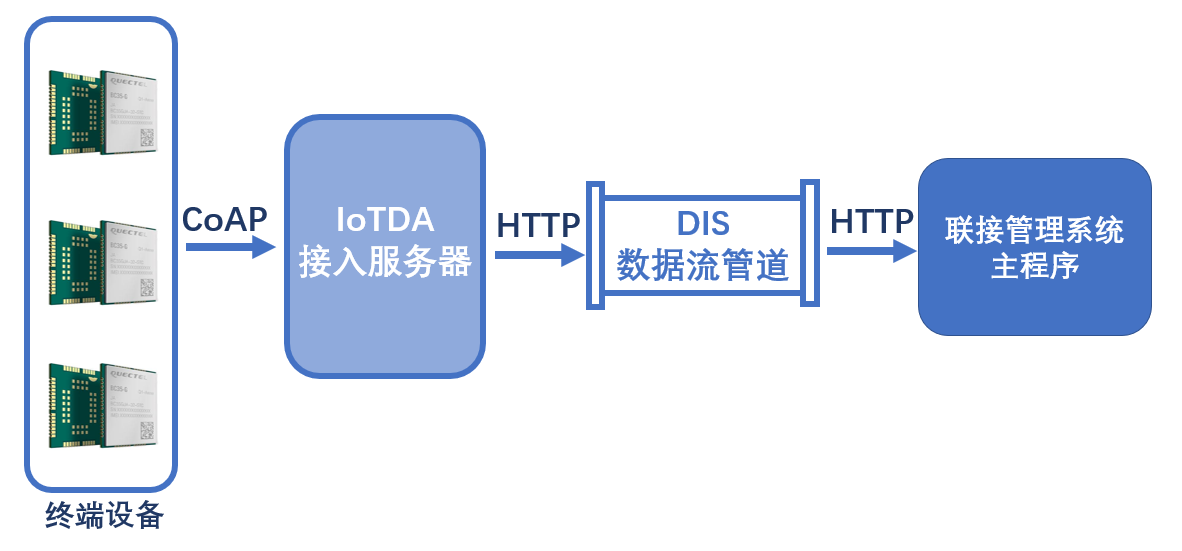
数据接入服务（Data Ingestion Service，简称DIS），是华为技术有限公司开发的一款云服务，为处理或分析数据流数据的自定义应用程序构建数据流管道，解决云服务外的数据实时传输到云服务内的问题【】，可用于快速构建实时数据应用。联接管理系统的一个基本功能是物联网设备的接入，通过DIS通道可将物联网设备上报的数据实时传输到云服务内，且联接管理系统的其他进程也可以实时的从DIS通道中读取数据。关于物联网设备通过DIS接入云服务的细节可参阅3.X“设备接入原理”。

3.2 数据转储机制

3.2.1设备接入原理

数据采集终端通过NB-IoT技术接入联接管理系统，并使用CoAP协议将业务数据上报到云端。受限制的应用协议（Constrained Application Protocol,检测CoAP）协议是一种小巧的应用层协议【】，多被运用于小型设备或物联网设备。CoAP物联网设备使用的CoAP协议与浏览器使用的HTTP协议类似，都用于网络应用层的信息交换。可以将CoAP协议看作为一种简化的HTTP协议，HTTP是一种较为复杂的应用层协议，运行于TCP之上，对硬件的要求较高，而CoAP协议运行于UDP之上，且数据包非常小巧，适合嵌入式设备运行。 NB-IoT模组提供了一系列关于CoAP协议的AT指令，可以向特定服务器发送CoAP报文，我们只需在运行联接管理系统的服务器中设定一个网络端口用于接收各个NB-IoT模组上报的CoAP报文，即可接收各数据采集终端上报的业务数据。

华为云针对这种典型的应用场景开发了“IoT设备接入云服务”，提供了一个稳定的供CoAP协议接入的CDP(客户数据平台)服务器端口，在接收到数据采集终端上报的CoAP报文后，会自动对其进行解码，并将其中的有效信息转储至DIS服务。联接管理系统的主进程通过该CDP服务器即可获得实时的业务数据。图XX为设备接入过程的示意图。



使用DIS实现设备接入无需自行搭建CDP服务器用于监听CoAP报文，也无需自行编写CoAP报文的解码程序来提取报文中的有效数据，大大减轻了开发难度。使用成熟稳定的商用CDP服务也有助于提升系统整体的可靠性。

3.4.3数据储存

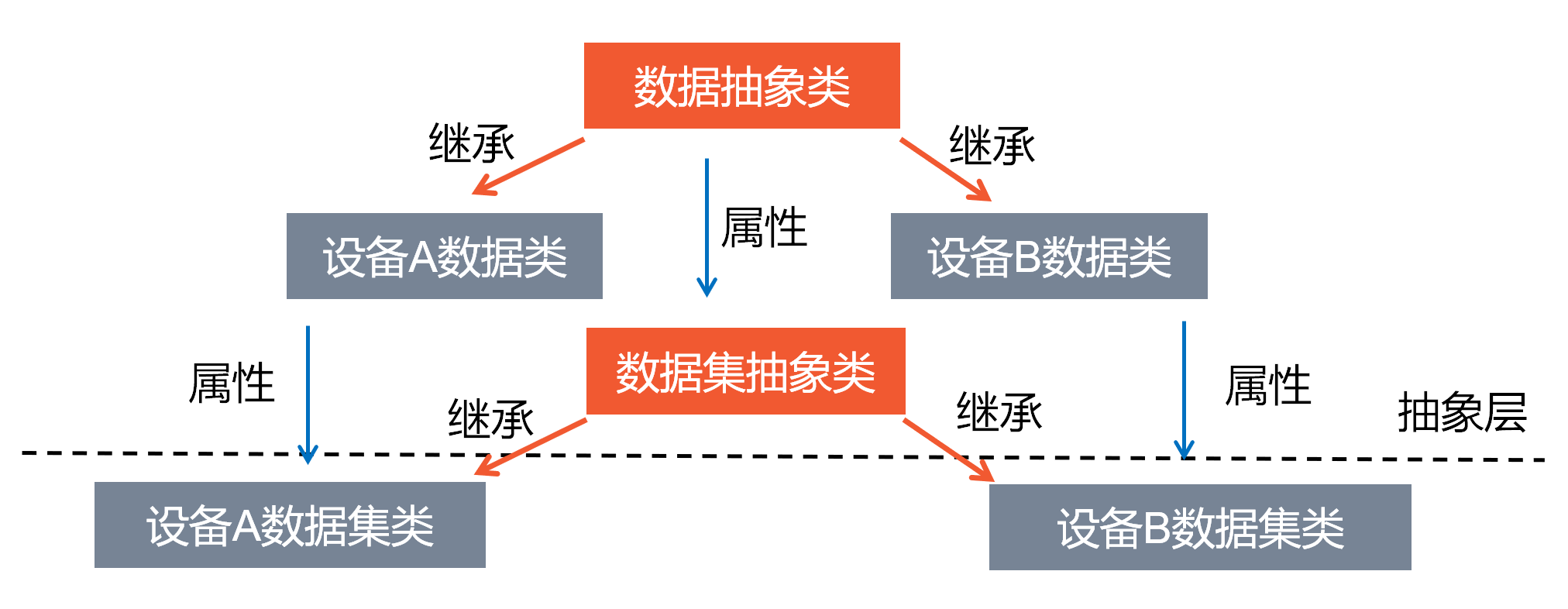
物联网数据以json格式文件储存在OBS仓库中，使用文件方式储存相较于传统的关系数据库有其独特优势。由于物联网数据具有明显的时序特性，传统的关系数据库很难对这类数据实现高效的储存。如果使用关系数据库来储存物联网时序数据，数据的唯一性由其时间决定，这也就意味着数据表的主键必须设定为数据产生的时间，这样一来，数据表的行数就与记录数据的条数成正比。当数据条数随时间增长，数据表的行数会达到一个巨大的数量级，此时关系数据库的工作性能将大大降低。通过分析物联网时序数据的存储与读取特性，可以发现其储存行为是一次性的，即储存过后不会被更改；其读取行为多是根据一个确定的时间范围进行检索。使用文件可以很低成本的实现大量数据的储存，同时也可以按照数据产生的不同时间来构建文件系统，当使用时间范围检索数据时，可以很方便的定位到所需时间范围的文件，实现不亚于关系数据库的查询性能。

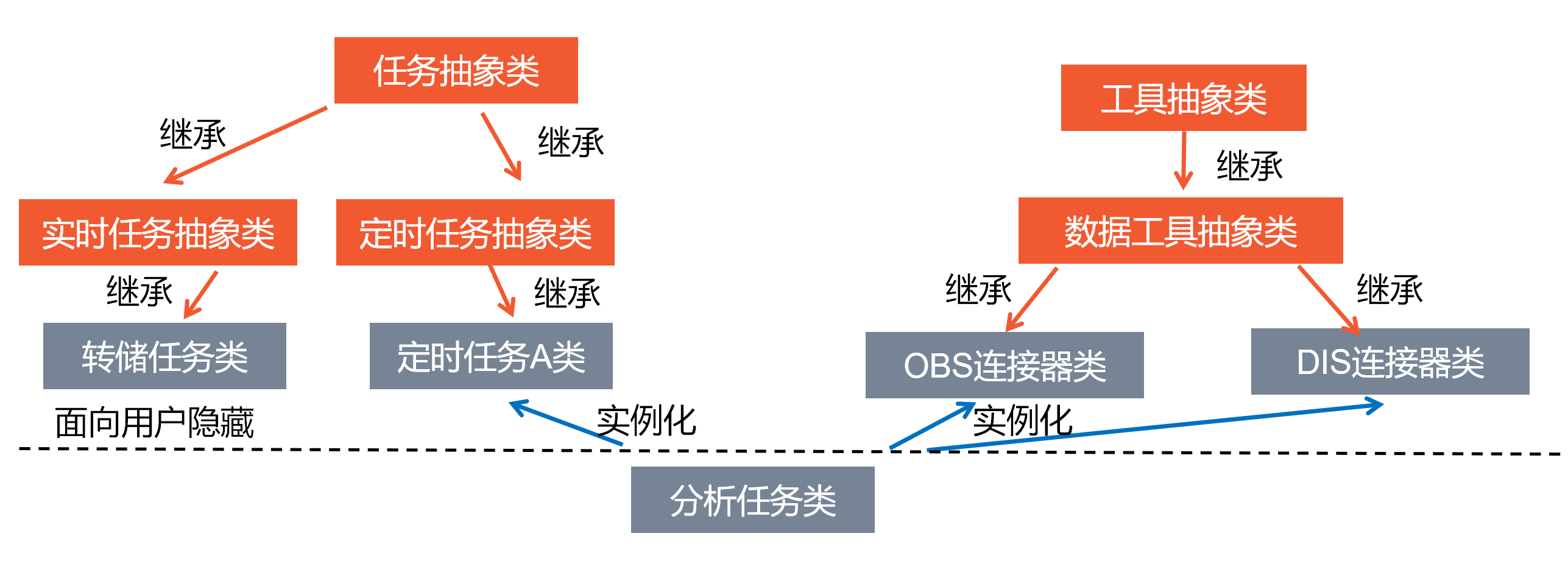
3.3 数据分析和处理

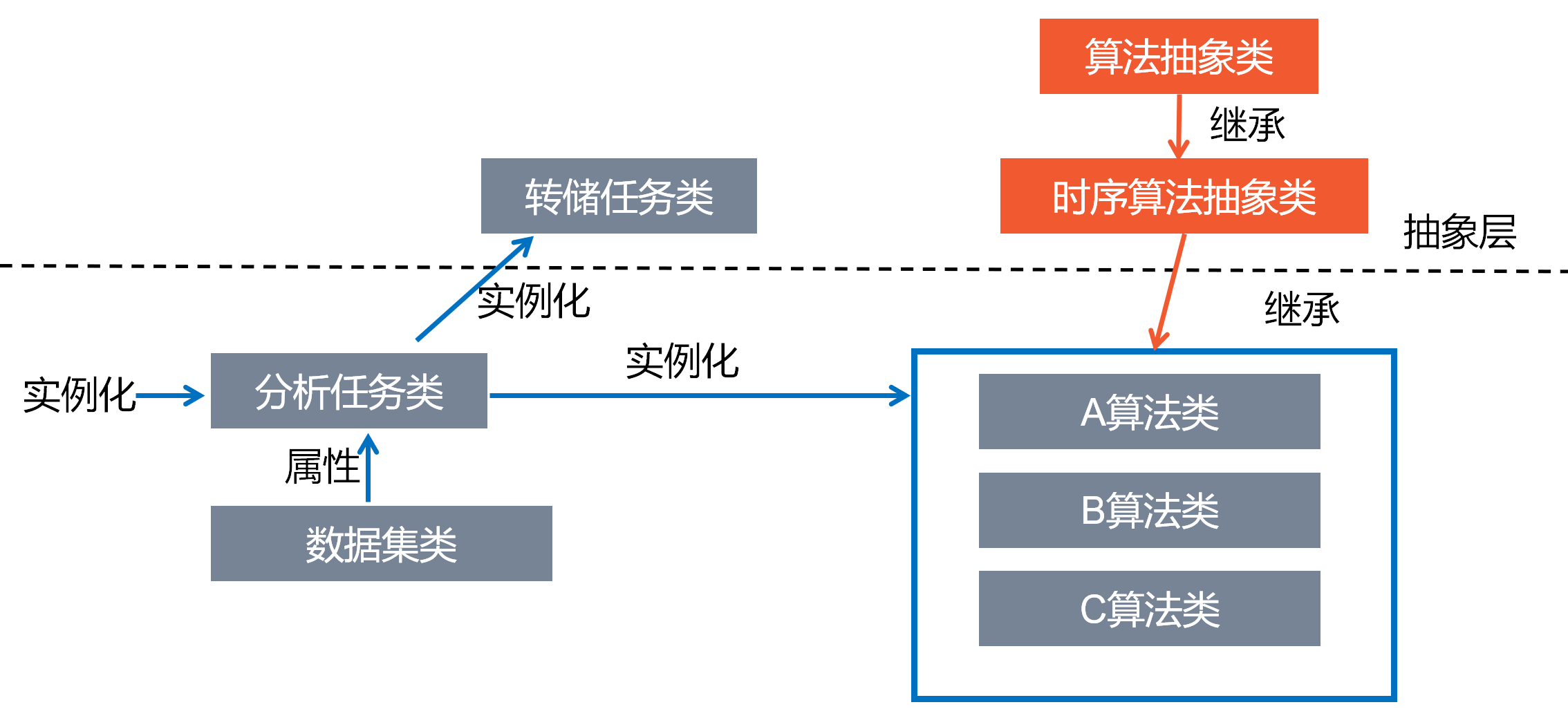
中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。

3.3.1数据分析组件

中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。







3.3.2算法定制化开发

中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。

3.5 部署与功能拓展

3.5.1云端部署方案

中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。

3.5.2组件升级方案

中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。

3.6 源代码及配置参数

1. 客户端应用软件

客户端应用软件基于物联网核心应用场景设计，将后台储存的物联网数据可视化，并提供控制接口供用户和设备管理人员使用。本客户端应用软件根据典型的工业物联网应用场景设计了多种类型的设备状态监控界面，可显示实时数据和历史数据曲线，同时可在此基础上开发多种数据分析界面。软件基于WEB技术实现，在浏览器上运行，满足了PC、手机、PAD多种终端的使用需求。

4.1 全局设计

4.1.1系统工作模式

软件基于WEB技术开发，支持多平台运行，拥有独立的后端（服务器端）和前端（客户端）进程，后端进程运行于远程服务器，负责与数据库对接，并响应前端进程的请求。前端进程运行于客户浏览器端，负责在本地渲染网页效果，并将后端发回的数据布设在网站界面中。这种工作模式的好处是，后端程序运行于较高性能的远程服务器，拥有更强的计算能力和更高的网络链路速度，可以更快速的从远程数据库中读取数据并进行预处理。预处理过程包括数据包的重组，压缩等，尽量减少数据中的冗余部分。前端程序仅需编写界面的交互逻辑，而不用考虑大流量数据的传输和处理，仅需接收后端程序缓存完成的数据包即可，减轻了浏览器的工作负担，同时简化了程序代码，使其能更快的加载和运行。

4.1.2软件架构和运行逻辑

整个软件系统拥有两个主进程，分别运行于服务器和客户端浏览器，每个主进程中包含若干子线程，共同维护软件的运作。服务器进程需要拥有对接联接管理系统的上行接口，此处通过调用与联接管理系统共享信息的华为云OBS对象储存服务API来实现，这种方案的优势将在下文描述。服务器进程也需要对客户端进程发送的不同HTTP请求进行监听，并返回对应的数据包，所以服务器进程也需要对来自上行接口的数据进行重新整理和压缩，降低数据包大小以节省网络带宽。我们使用json格式的字符串作为服务器进程的通用数据传输格式，即向上行API获取的数据和向客户端发送的数据都以json格式编码。总体上来说，服务器进程可以分为三个子线程，线程1按照一定的周期向上行API调取原始数据，调取到的原始数据以文件形式保存在本地。线程2按照一定的周期对原始数据进行重新整理和压缩，这些处理过后的数据也以文件形式保存在本地，在客户端请求到来时作为HTTP的响应内容发送给客户端。线程3负责监听网络中的HTTP请求，并返回数据包至客户端。三个线程以异步形式运行，线程3开始运行的时间是一个随机值，取决于客户端何时返回请求。线程1和2的工作实际上是一个连续过程，但此处将其分为两个独立线程，原因是当线程1因为数据量过大或网络故障发生阻塞，线程2仍可以保证在一个运行周期过后返回最新数据的处理结果。

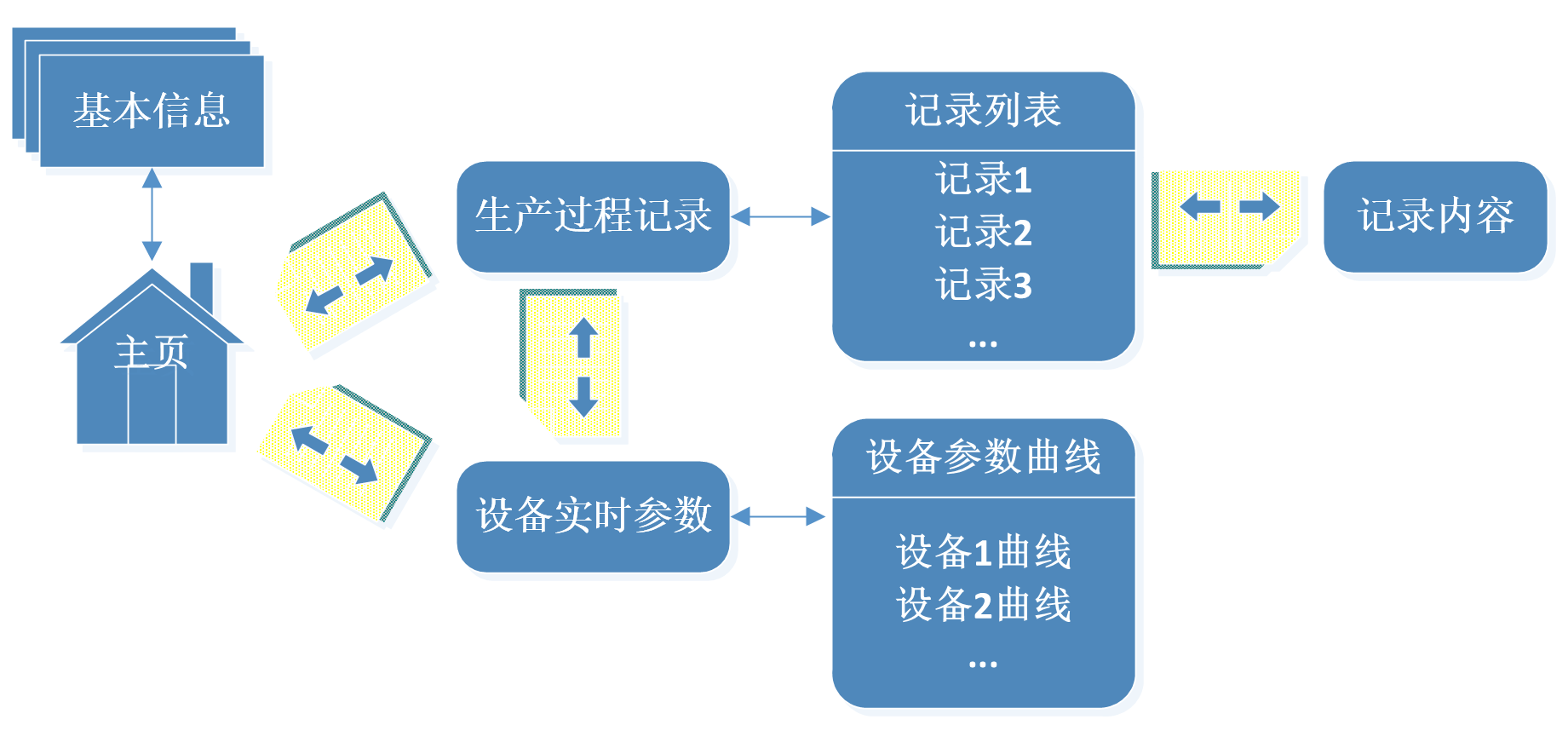
客户端进程的执行文件在网页加载初期被发送到用户浏览器上，同时传送的还有网页的静态文件，包括html文档，CSS文件，图片等。之后用户浏览器会运行客户端进程，解析网页静态文件，并根据用户的不同操作向服务器发送不同的HTTP请求。客户端进程中同样拥有异步运行的独立线程，如在网页加载初期，渲染网页静态文档和向服务器请求原始数据由两个分立的线程负责，这种架构保证了网站运行的流畅性。

4.1.3开发平台与框架

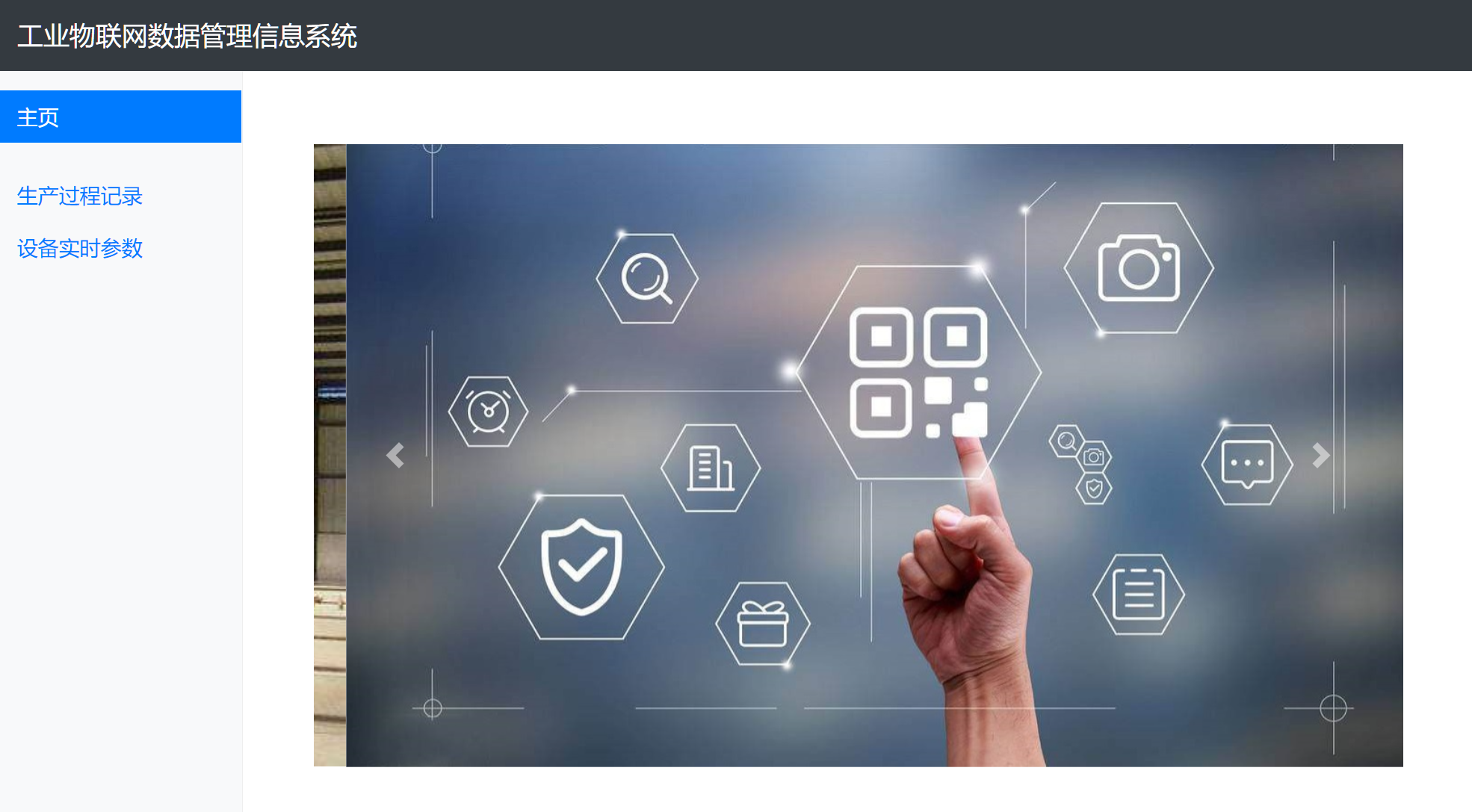
随着软件工程技术的发展，越来越多的软件通过不同的开发平台和框架实现，开发平台和框架是对一类形式的软件的抽象，它规定了一种文件结构形式或代码样式，使得在此基础上开发软件更加快速方便，同时拥有更好的性能。同样，物联网系统的客户端应用软件同样基于一些开发平台和框架实现：服务器端与客户端都基于node.js【】框架开发，node.js是一个开源与跨平台的 JavaScript 运行环境，可以不依赖浏览器运行JavaScript程序，这样一来，服务器端和客户端都可以用JavaScript一种语言来编写，通过标准的node.js接口通信，大大简化了系统的复杂程度。此外，客户端也使用了express【】和bootstrap【】框架用于网页交互的设计，实现了高性能的网页渲染和生动的交互效果。此外，客户端的数据可视化图表使用了由百度公司开源的 ECharts【】 技术，基于此技术，可以以低成本实现惊艳的数据可视化效果。

4.2 应用功能设计

工业设备物联网客户端拥有基本的设备参数监控功能与数据分析结果查看功能。在网页设计中，我们将这两个主要功能分为两个界面，分别是“设备实时参数”和“生产过程记录”。下图是网站的总体页面设计图。



其中“主页”页面可用于放置介绍企业基本信息的静态图片或文字（图xx），同时带有通向“设备实时参数”和“生产过程记录”页面的链接。“设备实时参数”页面（图xx）中放置有显示设备参数曲线的图表组件，可以自行编辑和缩放，并带有基本的统计功能。“生产过程记录”页面（图xx）根据相应规则将时序的生产数据分为多个记录集，以列表形式排列在页面上，通过点击子记录按钮，可以进入“记录内容”页面（图xx）查看该记录的详细信息，包括相应设备参数曲线，以及系统计算得出的各种数据分析结果。



4.2.1设备监控功能

通过“设备实时参数”页面我们可以对联网设备实现实时监控，界面中有多个独立的图表组件，每一个图表都对应一台设备，图表中有多条曲线，可以显示多种设备变量随时间变化的情况。用户可以对图表进行简单操作，如缩放时间轴，隐藏或显示变量曲线，点击数据点查看更多内容等。同时，每个图表组件都可以自由拖拽变换位置，用户可以将两个相同类型设备的监控图排列在同一区域，便于用户对比分析。



4.2.1分析记录查看功能

通过“生产过程记录”页面，我们可以按照日期查找当天的生产记录，通过点击界面上的日历组件，随后系统会自动返回对应日期的生产记录，后台分析程序通过分析当日的设备参数曲线，自动找出不同生产阶段的分界点，再根据这些分界点将一天的生产过程分为多条记录。用户可以点击每条记录后的“查看”按钮进入记录内容界面，该界面会显示更多详细信息供用户查看。





4.2 应用实现方案

4.2.1基于node.js/Express技术的网站开发

WEB应用的后端进程运行在网络服务器上，通过监听特定端口传回的客户端HTTP报文，来处理客户端的请求，之后再通过HTTP协议回复客户端。运行在服务器端的后端进程可能需要在同一时间处理多个客户端发来的请求，这就意味着需要在同一时间处理若干个并发连接。此外，后端进程应该在处理请求的空闲时间缓存来自上级数据库的数据，避免在客户端提出请求时再进行数据库读取，从而延长响应时间。以上问题对后端进程的使用性能都有着很大的影响，需要在设计时得到妥善解决。

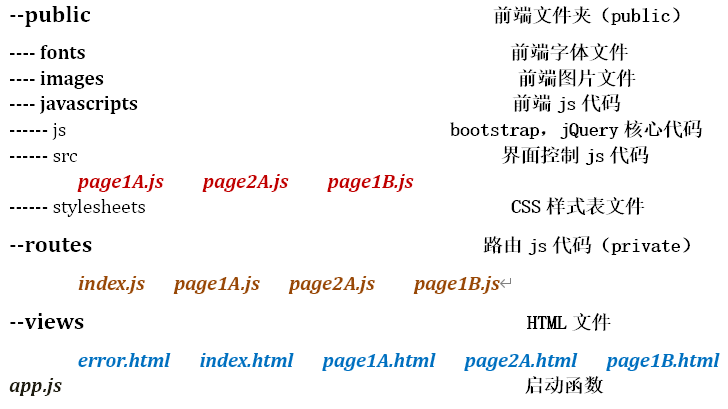
传统的后端程序代码使用java或python等高级语言编写，开发者需要自行实现用于处理异步并发响应的多线程算法，无论是使用阻塞线程或引入管理线程来处理并发事件，实现起来都较为繁琐，受限于开发者有限的编程水平，这类算法很容易出错且难以达到很好的性能。

为了克服以上难题，我们选择Node.js技术开发后端应用。Node.js是一个开源与跨平台的JavaScript运行环境【】，拥有独立的Chrome V8引擎，使其可以不依赖浏览器独立运行JavaScript程序。由于JavaScript是一门专门为网络编程创建的编程语言，使用JavaScript开发服务器端程序有着天然的优势，很多专为网络编程设计的语法特性和函数可以直接使用，而无需像其他高级语言那样需要调用特定的插件和库。此外，相比传统开发方式，Node.js拥有更加强大的异步并发处理能力，同时支持各种最新的网络编程技术。Node.js异步运行的特点贯穿于整个编程过程中，如大量的使用回调函数，使得它再执行I/O操作（网络读取、访问数据库或文件系统）时，会在响应返回时恢复操作，而不是阻塞线程使得CPU循环等待【】。

Express 是一个保持最小规模的灵活的 Node.js Web 应用程序开发框架，为 Web 和移动应用程序提供一组强大的功能【】。Express框架集成了许多中间件可直接供用户使用，例如解析json文件、解析cookie、解析日志等工作都可以通过调用中间件来完成。使用Express框架来开发网站可以大大减轻开发者的负担，开发者可以把工作重心完全放在实现网站核心功能算法上。

4.2.2程序架构

本客户端软件按照标准的Node.js网页应用架构设计，代码文件包括运行在客户浏览器上的界面脚本，以及运行于服务器端的路由控制脚本，服务器程序由一个独立的启动脚本启动，下图显示了本软件的代码文件结构：



其中字段index代表“主页”，page1A代表“设备实时参数”页面，page2A代表“生产过程记录”页面，page1B代表“记录内容”页面。我们设置了一个public文件夹来存放网站的静态文件，包括字体、图片、CSS文件和前端代码，这些静态文件对网站访问者开放，所以必须与其他代码文件隔离。此外，我们设置了routes文件夹来存放路由控制器代码。Express会根据前端返回的不同URL来调用对应页面的路由控制器来处理响应，网站后端的大部分逻辑实现都在这些路由控制器代码中。views文件夹单独用于放置html文档，每个html文档都描述了一个页面的组件布局和初始化过程。app.js文件中有整个后端程序的启动入口，以及各种初始化代码。

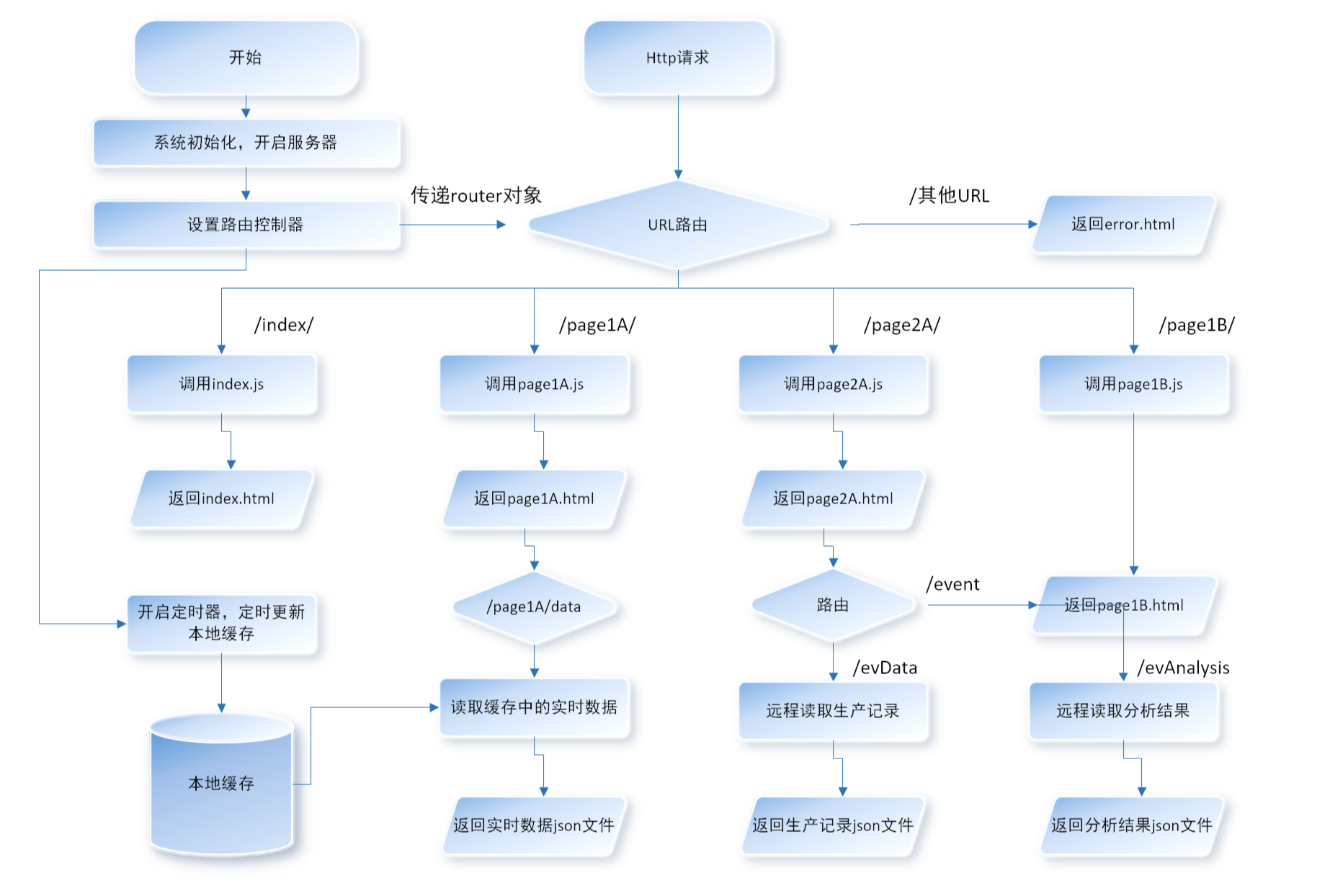
4.2.2后端程序工作模式

后端程序运行于远程服务器，主要负责处理前端发回的各种响应请求，不同请求使用不同的URL来标记，后端主程序通过识别不同的请求URL会交给路由处理来转发给相应的路由控制器（图xx棕褐色代码文件）。 在路由控制器中，Express框架使用回调函数来处理页面请求和响应，通过调用request和response对象中的指定函数来获取收到的请求URL或发回响应数据。 客户端软件在一共放置了四个路由控制器程序，分别处理主页、生产记录、实时参数、记录内容页面发回的不同URL，下表列出了网站URL的路由规则。

**表1 网站路由表**

|  |  |  |
| --- | --- | --- |
| URL | 目标页面 | 请求描述 |
| /index | index | 返回主页 |
| /page1A | page1A | 进入设备参数页面 |
| /page2A | page2A | 进入生产记录页面 |
| /page1A/data | -- | 请求实时数据json文件 |
| /page2A/event | page1B | 进入记录内容界面 |
| /page2A/evData | -- | 请求生产记录列表json文件 |
| /page2A/evAnalysis | -- | 请求记录分析结果json文件 |

图X显示了后端程序的完整工作模式，此处实时数据和分析结果数据由联接管理系统上传至华为云OBS（对象储存服务）服务器中，网站后端程序通过双重加密验证访问OBS服务器，下载所需数据。由于实时数据量巨大，在客户请求时再进行远程读取可能会花费较长的时间，所以我们单独设置了一个缓存程序，通过预先设置的定时器控制，在空闲时间下载实时数据至本地，在前端访问时直接从本地缓存中读取实时数据，虽然存在一定的更新延迟，但由于物联网数据的更新多以秒或分钟计算，所以对数据完整性的影响并不大，但这种方式能显著加快网站响应速度，对于提升整体使用体验有很大的帮助。除实时数据是通过缓存形式读取外，其他类型的数据都是现场连接远程服务器读取，因为数据量较小，所以延迟基本可以忽略不计。



4.3.1基于Bootstrap/Echarts的前端开发

在编写前端程序时，我们运用了Bootstrap和jQuery框架来实现更高级的前端交互效果。Bootstrap是Twitter推出的一个用于前端开发的开源工具包【】。它由Twitter的设计师Mark Otto和Jacob Thornton合作开发,是一个CSS/HTML框架。Bootstrap集成了一系列美观且功能丰富的网页布局组件，可以实现更高级的交互效果。Bootstrap在前端交互中的应用实例如图XX所示。

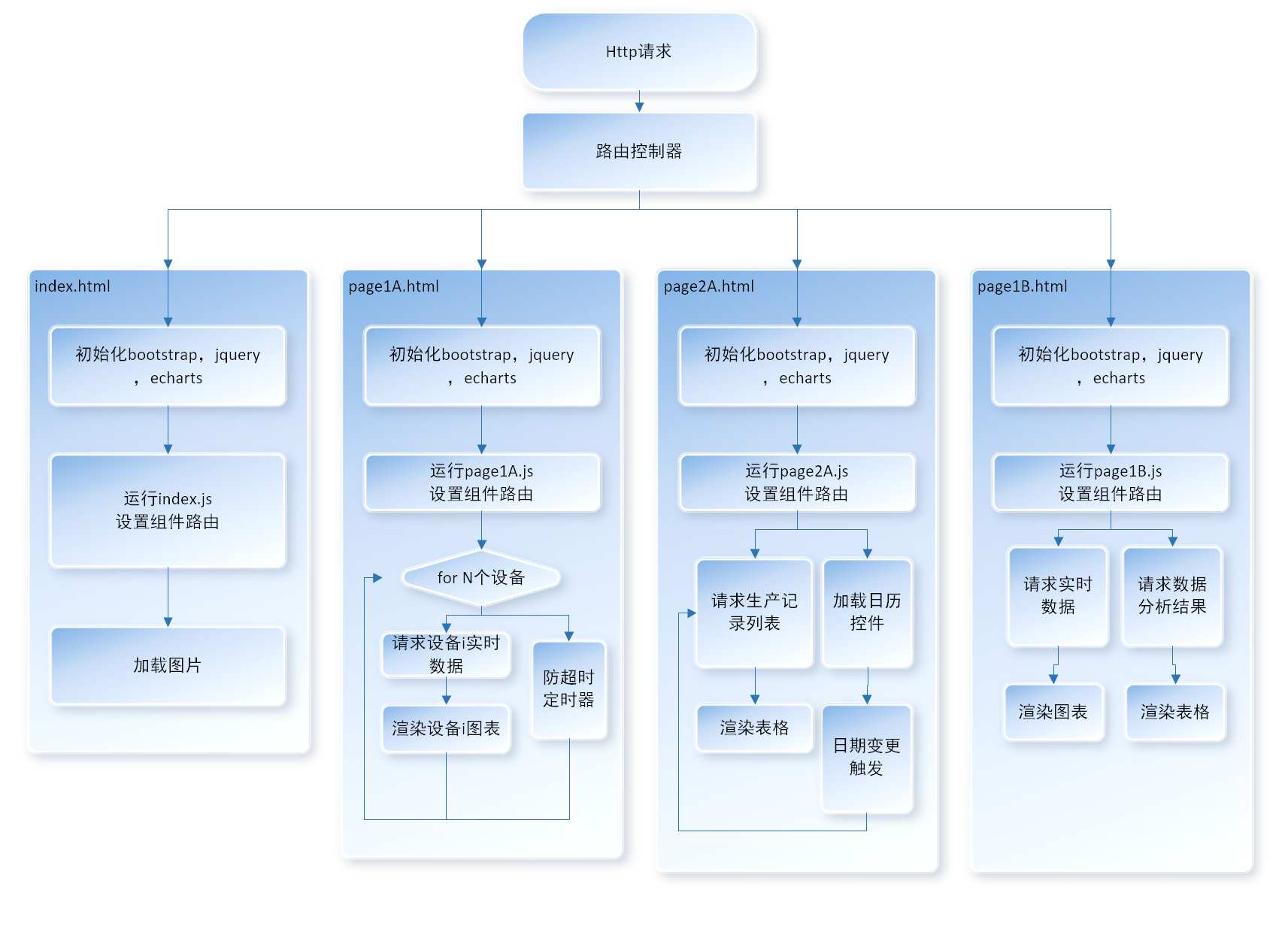


本应用借助了百度公司开源的Echarts【】技术来实现前端丰富多样的图表效果。Echarts是百度公司面向数据可视化应用开发而开源的Javascript插件，通过Echarts可以很方便的实现带有高级缩放和反馈功能的图表。Echarts在前端交互中的应用实例如图XX所示。

4.3.2前端程序工作模式

界面脚本代码文件与每一个界面html文档对应，负责控制界面中各个图表和组件，以及界面效果加载等，这部分代码在网页加载时被发送到客户机上，随后在客户端浏览器运行，这就要求严格控制代码文件大小并简化内部算法，以适应不同性能的浏览器。我们通过将耗时耗力的数据规整和表格渲染任务被安排在服务器端完成，减轻浏览器的工作压力，但这种方式可能会导致页面加载时间延长，因为浏览器必须等待服务器返回预处理好的结果才能进行画面下一步的渲染，为此，我们设计了一个分段数据传输算法来提升页面加载速度。在页面加载的不同阶段，客户端算法与服务器端算法将会进行多次双向通信，按照数据量大小分批次发送数据文件，客户端进程可在数据传输时同时进行界面渲染，这样大大提升了页面的整体加载速度。

前端工作模式图（图xx）描述了每一个页面在浏览器打开时的加载步骤，可以看出在page1A页面加载时，设备实时数据是分段传输并渲染的。此外在其他页面的加载过程中，我们充分应用了node.js框架的异步处理特性，尽量将多个加载和渲染任务并行完成以提高页面的整体加载速度。



4.4 代码维护与功能拓展

考虑到后期代码的维护与功能拓展问题，在设计网页客户端时，我们充分遵循了MVC设计模式【】，这种设计模式提倡将用户界面与后端逻辑代码完全分开，并在中间编写一个专用的控制器对象来进行前后端参数的调度，以实现前端页面代码的良好封装，但同时可以通过修改控制器对象来继续利用原有的逻辑函数。的在软件设计过程中，依据这种设计模式，数据远程读取和各种预处理算法的逻辑代码都后端实现，每一个前端页面的控制代码与其他页面几乎没有任何逻辑关联，实现了良好的封装性。当后期开发中需要继续添加新的功能或增加新的前端页面时，由于算法逻辑和各种参数的传递都在后端完成，很容易实现跨页面的分享数据，同时也可以使用原先的功能函数。

4.5 源代码与配置参数

中文为宋体小四号字，段落首行缩进2字符，行距1.5倍。

表序在表题左方，不加标点，中间空一格，标题末尾不加标点。全文表格可统一编序，也可按章节编序，表序须连续。

表的示例如下：

**表1 XX表**（中文黑体，英文为Times New Roman五号加粗，居中）

|  |  |  |
| --- | --- | --- |
| XX（宋体五号） | XX | XXX  表格内容中文为宋体五号，英文为Times New Roman五号 |
| 1 | 11 | 111 |
| 2 | 22 | 222 |
| …… | …… | …… |

数据来源：......( 注于表下方，宋体五号，相对表格左下角缩进2个汉字。)

脚注（也可在论文篇末作尾注），字号小五，中文宋体英文Times New Roman。

图的示例如下：

图片

**图1 XX图**（中文黑体，英文为Times New Roman五号加粗，居中）

图序和图题居于图的下方正中，图序须连续。可用全文统一或按章节编序，但无论用哪种方式，应和表格、公式的方式统一。

结束语：（可选项，**中文黑体，英文Times New Roman，三号**）

......（正文内容格式：中文为宋体，英文为Times New Roman，均为小四号字，段落首行缩进2字符，行距1.5倍，下同。）

**参考文献**（黑体三号，另起一页）

[1] 作者．文献名[M]．出版地：出版者，出版年：起止页码（整体引用时不注）．(图书文献适用)

[2] 作者．文献名[J] ．刊名，年，卷（期）：起止页码．（期刊文献适用）

[3] 作者．文献名[N]．报纸名，出版日期（版次）．（报纸文献适用）

[4] 标准编号，标准名称[S]．（标准、法规文献适用）

[5] 作者．文献名[文献类型标识/载体类型标识]．出版地：出版者，出版年：起止页码（当整体引用时不注）．(载体类型标识为 “DK”、“MT”和“CD”，分别对应磁盘、磁带和光盘电子文献适用）

[6] 作者．文献名[文献类型标识/ OL]．（发表或更新日期）．[引用日期]．电子文献网址．(在线电子文献适用）

中文用宋体，英文用Times New Roman，均为五号字体。

附录（黑体三号字，为可选项，另起一页）

附录A

附A1

......（内容格式：中文为宋体，英文为Times New Roman，均为小四号字，段落首行缩进2字符，行距1.5倍。）

附录B

附录C

致谢（黑体三号字，为可选项，另起一页）

......（内容格式：中文为宋体，英文为Times New Roman，均为小四号字，段落首行缩进2字符，行距1.5倍。）