# 电气量测控平台系统需求分析与架构设计技术方案

**适配平台：RK3568 嵌入式平台（ubuntu 22.04系统）**
**技术栈：FastAPI + HTTP + WebSocket + Python + Web前端**

# 一、文档概述

## 1.1 文档目的

本文档基于《电气量测控平台显示界面设计和数据接口说明》、`gui.h`头文件及项目功能描述，明确**RK3568嵌入式平台**上Python编写、Web界面展示的电气量测控配置软件的功能、性能、接口、数据存储需求，完成系统架构设计，为软件开发、测试及交付提供技术依据。

## 1.2 适用范围

本文档适用于**RK3568嵌入式硬件**，以Python为开发语言、FastAPI为服务框架、Web为交互界面的电气量测控配置软件，涵盖参数配置、实时数据交互、状态监测、报警管理等全流程功能设计。

## 1.3 参考资料

1. 《电气量测控平台显示界面设计和数据接口说明》
2. `gui.h`嵌入式程序头文件
3. 项目功能需求描述

# 二、系统需求分析

## 2.1 运行环境需求

- **硬件环境**：RK3568嵌入式开发板（支持Linux系统、文件系统、SQLite数据库）
- **软件环境**：Linux操作系统、Python 3.8、**FastAPI**、Uvicorn ASGI服务、SQLite数据库、JSON文件存储支持
- **交互环境**：Web浏览器（支持WebSocket、动态数据渲染、按钮交互、表格/波形展示）

## 2.2 功能需求

### 2.2.1 参数配置管理功能

实现测控设备全维度参数配置，配置项支持**JSON文件本地存储**，关键参数通过接口下发至C语言嵌入式程序，具体配置项如下：

1. **基础设备配置**
    - 设备软件版本信息展示与配置
    - 通讯参数设置：串口、网口参数配置
    - 操作密码设置：支持密码修改、本地JSON文件存储
2. **装置通道配置**
    - AI通道设置：模拟量输入配置、信号类型、所属线路、数据阈值
    - AO通道设置：模拟量输出配置

    - 定值设置：电压/电流/功率等超限限值、检测延时、报警输出配置
3. **系统管理配置**
    - 对时设置：参数实时下发嵌入式程序
    - 灯光设置：屏幕亮度、屏保时间，参数实时下发嵌入式程序

### 2.2.2 实时数据动态显示功能

1. **数据读取周期**：间隔**0.5秒**从C语言控制系统读取实时数据
2. **数据类型**：模拟量、开关量、AI采集数据
3. **推送方式**：**WebSocket主动推送**至Web界面，动态刷新展示
4. **存储规则**：动态数据仅内存缓存，**不保存到本地**

### 2.2.3 设备状态实时监测功能

从嵌入式系统动态读取设备状态，Web界面实时展示：

- 装置自检状态（正常/异常）
- 网络1状态、网络2状态
- 串口1状态、串口2状态

### 2.2.4 报警事件管理功能

1. **报警数据采集**：实时获取嵌入式设备报警事件
2. **数据存储**：报警事件保存至**SQLite数据库**
3. **展示方式**：**WebSocket实时推送**报警至Web界面，支持弹窗提醒+历史查询

## 2.3 数据交互需求

### 2.3.1 上行数据（嵌入式系统→Python服务）

- 实时数据：0.5秒/次，模拟量、开关量、AI采集数据
- 状态数据：装置自检、网络、串口状态
- 报警数据：实时报警事件数据

### 2.3.2 下行数据（Python服务→嵌入式系统）

- 配置数据：设备参数、通道配置、定值、对时、灯光参数
- 控制指令：开关量分/合控制信号

## 2.4 数据存储需求

1. **JSON文件存储**：设备版本、通讯参数、密码、AI/AO配置、定值配置等**非实时参数**
2. **SQLite数据库**：报警事件数据（支持历史查询）
3. **动态数据**：仅内存缓存，不持久化存储

## 2.5 性能需求

1. **数据刷新率**：实时数据读取与WebSocket推送间隔≤0.5秒
2. **响应速度**：参数配置下发、开关控制指令响应时间≤1秒
3. **稳定性**：7×24小时连续运行
4. **兼容性**：适配RK3568硬件，兼容嵌入式C程序接口

## 2.6 通信服务需求

1. **HTTP服务**：基于FastAPI提供**RESTful API**，用于参数配置、状态查询、报警查询、控制指令
2. **WebSocket服务**：基于FastAPI提供双工通信，用于**实时数据、报警数据主动推送**至Web界面

# 三、系统架构设计

## 3.1 总体架构

系统采用**分层架构 + 前后端分离**设计，从下到上分为：
**硬件层 → 嵌入式C程序层 → Python FastAPI服务层 → Web界面层**

```
Web界面层（HTML/CSS/JS）
        ↓↑（HTTP + WebSocket）
Python FastAPI服务层（RESTful接口 + WebSocket推送 + 数据处理）
        ↓↑（内部进程/核间通讯）
嵌入式C程序层（数据采集 + 控制执行）
        ↓
硬件层（RK3568 + 测控硬件 + 通讯模块）
```

## 3.2 分层详细设计

### 3.2.1 硬件层

- 核心硬件：RK3568嵌入式开发板
- 外设模块：AI/AO采集模块、开关量模块、网络/串口模块、显示屏

### 3.2.2 嵌入式C程序层

- 数据采集模块：20ms周期采集模拟量、开关量、AI数据
- 控制执行模块：接收参数与指令，执行配置、对时、灯光、开关控制
- 状态监测模块：自检、网口、串口状态实时上报
- 报警生成模块：异常事件生成并上传
- 通讯模块：与Python服务双向数据交互

### 3.2.3 Python FastAPI服务层（核心）

采用**FastAPI + Uvicorn**搭建，提供**HTTP接口**和**WebSocket推送**双服务：

1. **RESTful API模块**：处理配置提交、状态查询、报警查询、控制指令
2. **WebSocket推送模块**：0.5秒推送实时数据、实时报警数据到Web端

3. **参数配置模块**：JSON文件读写、配置下发C程序
4. **实时数据模块**：定时读取C程序数据，缓存并通过WebSocket推送
5. **报警管理模块**：接收报警→存储SQLite→WebSocket推送前端
6. **设备状态模块**：采集并缓存设备运行状态

### 3.2.4 Web界面层

- 菜单区：配置、状态、数据、报警、控制入口
- 状态区：自检、网络、串口实时状态
- 主显示区：实时数据表格、配置表单、控制按钮、报警列表
- 报警区：WebSocket实时弹窗+滚动显示

## 3.3 数据流向设计

1. **实时/报警数据流向**
   嵌入式C程序 → Python → WebSocket主动推送 → Web界面（0.5秒刷新）
2. **参数配置流向**
   Web界面 → HTTP POST → Python → JSON存储 + 下发C程序
3. **状态/查询流向**
   Web界面 → HTTP GET → Python → 返回数据展示

# 四、接口详细设计（RESTful + WebSocket）

## 4.1 RESTful API 接口详情（完整参数+返回值）

所有接口统一返回格式：

```
{
  "code": 200,          // 200=成功，400=参数错误，500=服务异常
  "msg": "操作成功",
  "data": {}            // 业务数据
}
```

---

### 1. GET /api/real-time-data

**作用**：获取最新一次实时数据（备用，主用WebSocket）
**参数**：无
**返回data**：

```
{
  "code": 200,
  "msg": "获取实时数据成功",
  "data": {
    "line_list": [
      {
        "line_no": 1,
        "pri_val": {
```

```
      "Ua": 6000.0,
      "Ub": 6002.5,
      "Uc": 5998.3,
      "Ia": 150.2,
      "Ib": 148.9,
      "Ic": 151.6,
      "Pa": 820.5,
      "Pb": 815.3,
      "Pc": 825.7,
      "Pt": 2461.5,
      "Qa": 120.3,
      "Qb": 118.6,
      "Qc": 122.9,
      "Qt": 361.8,
      "Sa": 835.2,
      "Sb": 829.7,
      "Sc": 840.3,
      "St": 2505.2,
      "PFa": 0.98,
      "PFb": 0.97,
      "PFc": 0.99,
      "PFt": 0.98,
      "Uab": 10400.1,
      "Ubc": 10395.6,
      "Uca": 10398.2,
      "frq": 50.00
    },
    "sec_val": {
      "Ua": 57.6,
      "Ub": 57.7,
      "Uc": 57.5,
      "Ia": 1.2,
      "Ib": 1.18,
      "Ic": 1.22,
      "Pa": 68.2,
      "Pb": 67.8,
      "Pc": 68.6,
      "Pt": 204.6,
      "Qa": 9.8,
      "Qb": 9.6,
      "Qc": 10.1,
      "Qt": 29.5,
      "Sa": 69.5,
      "Sb": 68.9,
      "Sc": 70.1,
      "St": 208.5,
      "PFa": 0.98,
      "PFb": 0.97,
      "PFc": 0.99,
      "PFt": 0.98,
      "Uab": 100.0,
      "Ubc": 99.8,
      "Uca": 99.9,
```

```
        "frq": 50.00
      }
    },
    {
      "line_no": 2,
      "pri_val": {},
      "sec_val": {}
    },
    {
      "line_no": 3,
      "pri_val": {},
      "sec_val": {}
    },
    {
      "line_no": 4,
      "pri_val": {},
      "sec_val": {}
    }
  ],
  "switch": {
    "di1": 1,
    "di2": 0,
    "di3": 1,
    "di4": 0,
    "di5": 1,
    "di6": 0,
    "di7": 1,
    "di8": 0,
    "di9": 1,
    "di10": 0,
    "di11": 1,
    "di12": 0,
    "do1": 1,
    "do2": 0,
    "do3": 1,
    "do4": 0,
    "do5": 1,
    "do6": 0,
    "do7": 1,
    "do8": 0,
    "do9": 1,
    "do10": 0,
    "do11": 1,
    "do12": 0
  },
  "ai_collect": {
    "ai1": 4.52,
    "ai2": 3.86,
    "ai3": 5.21,
    "ai4": 2.99,
    "ai5": 6.10,
    "ai6": 1.75,
    "ai7": 3.22,
```

```
      "ai8": 4.91,
      "ai9": 2.18,
      "ai10": 5.47,
      "ai11": 3.06,
      "ai12": 4.33
    }
  }
}
```

---

## 2. GET /api/device-status

**作用**：获取设备当前运行状态
**参数**：无
**返回data**：

```
{
  "self_check": "正常",
  "net1": "正常",
  "net2": "正常",
  "uart1": "正常",
  "uart2": "断开"
}
```

---

## 3. POST /api/config/device

**作用**：提交设备基础配置（版本、通讯、密码）
**请求body参数**：

```
{
  "password": "123456",
  "hardware_version": {
    "board_version": "B001.001.001",
    "display_version": "S001.001.001",
    "other_version": "Y001.001.001"
  },
  "software_version": {
    "display_program": "001.001.001",
    "communication_program": "001.001.001",
    "measurement_program": "001.001.001"
  },
  "net": [
    {
      "nic": "网卡一",
      "ip": "",
      "mask": "",
      "gateway": "",
      "protocol": ""
    },
    {
      "nic": "网卡二",
```

```
      "ip": "192.168.1.56",
      "mask": "255.255.255.255",
      "gateway": "192.168.1.56",
      "protocol": "Modbus TCP"
    }
  ],
  "uart": [
    {
      "port": "COM1",
      "baud": 0,
      "parity": "",
      "data_bits": 0,
      "stop_bits": 0,
      "protocol": ""
    },
    {
      "port": "COM2",
      "baud": 115200,
      "parity": "NONE",
      "data_bits": 8,
      "stop_bits": 1,
      "protocol": "Modbus RTU"
    }
  ]
}
```


**返回data**：`{"save_path":"/config/device.json","send_status":"成功"}`

---

## 4. POST /api/config/channel

**作用**：提交AI/AO通道配置(AI：12通道，AO：12通道)
**请求body参数**：


```
{
  "ai_channel": [{"ch":1,"singal_type":"4-
20mA","line_no":1,"type":"UA","limit_low":0,"limit_high":20}],
  "ao_channel": [{"ch":1,"singal_type":"1-
5v","line_no":2,"type":"UA","limit_low":0,"limit_high":20}]
}
```


**返回data**：`{"save_path":"/config/channel.json","send_status":"成功"}`

---

## 5. POST /api/config/line_alarm_setting

**作用**：提交定值报警阈值配置
**请求body参数**：


```
{
      "line_no": 1,
      "over_limit_alarm": [
```

```
        {
          "category": "电压",
          "limit": 180,
          "delay": 180,
          "output_node": "开出1",
          "enabled": true
        },
        {
          "category": "电流",
          "limit": 180,
          "delay": 180,
          "output_node": "开出1",
          "enabled": true
        },
        {
          "category": "差流",
          "limit": 180,
          "delay": 180,
          "output_node": "开出1",
          "enabled": false
        },
        {
          "category": "功率",
          "limit": 180,
          "delay": 180,
          "output_node": "开出1",
          "enabled": true
        },
        {
          "category": "频率",
          "limit": 180,
          "delay": 180,
          "output_node": "开出1",
          "enabled": false
        }
      ],
      "fault_alarm": [
        {
          "category": "PT断线",
          "delay": 180,
          "output_node": "开出1",
          "enabled": true
        },
        {
          "category": "CT断线",
          "delay": 180,
          "output_node": "开出1",
          "enabled": false
        }
      ]
    }
```


**返回data**：`{"save_path":"/config/setting.json","send_status":"成功"}`

## 5. POST /api/config/ai_alarm_setting

**作用**：提交AI报警设置
**请求body参数**：

```
[
  {
      "channel_no": 1,
      "singal_type": "4-20mA",
      "limit_low":0,
      "limit_high":20,
      "delay": 180,
      "output_node": "开出1",
      "enabled": true
    },
 {
      "channel_no": 2,
      "singal_type": "1-5v",
      "limit_low":0,
      "limit_high":20,
      "delay": 180,
      "output_node": "开出1",
      "enabled": true
    },
    ]
```

**返回data**：`{"save_path":"/config/setting.json","send_status":"成功"}`

## 6. POST /api/config/system

**作用**：提交系统对时、灯光配置
**请求body参数**：

```
{
  "time_sync": "auto",
  "brightness": 80,
  "screen_saver": 60
}
```

**返回data**：`{"send_status":"成功"}`

---

## 7. GET /api/alarm/list

**作用**：分页查询历史报警
**参数**：`page=1&size=20`
**返回data**：

```
[
{
  "id": 1,
  "alarm_type": "operate_alarm",
  "time": "2025-01-01 12:00:00",
  "no": "",
  "type": "",
  "content": "告警内容",
  "level": "告警等级"
}
]
```

---

### 8. POST /api/control/switch

**作用**：下发开关量控制指令
**请求body参数**：

```
{"ch":1,"action":1}  // 1=合，0=分
```

**返回data**：`{"control_status":"执行成功"}`

---

## 4.2 WebSocket 接口（主动推送）

### 1. WebSocket 地址：`ws://ip:port/ws/real-time`

**推送内容**：实时数据（0.5秒/次）

```
{
  "type": "real_time",
  "data": {
    "line_list": [
      {
        "line_no": 1,
        "pri_val": {
          "Ua": 6000.0,
          "Ub": 6002.5,
          "Uc": 5998.3,
          "Ia": 150.2,
          "Ib": 148.9,
          "Ic": 151.6,
          "Pa": 820.5,
          "Pb": 815.3,
          "Pc": 825.7,
          "Pt": 2461.5,
          "Qa": 120.3,
          "Qb": 118.6,
          "Qc": 122.9,
          "Qt": 361.8,
          "Sa": 835.2,
```

```
        "Sb": 829.7,
        "Sc": 840.3,
        "St": 2505.2,
        "PFa": 0.98,
        "PFb": 0.97,
        "PFc": 0.99,
        "PFt": 0.98,
        "Uab": 10400.1,
        "Ubc": 10395.6,
        "Uca": 10398.2,
        "frq": 50.00
      },
      "sec_val": {
        "Ua": 57.6,
        "Ub": 57.7,
        "Uc": 57.5,
        "Ia": 1.2,
        "Ib": 1.18,
        "Ic": 1.22,
        "Pa": 68.2,
        "Pb": 67.8,
        "Pc": 68.6,
        "Pt": 204.6,
        "Qa": 9.8,
        "Qb": 9.6,
        "Qc": 10.1,
        "Qt": 29.5,
        "Sa": 69.5,
        "Sb": 68.9,
        "Sc": 70.1,
        "St": 208.5,
        "PFa": 0.98,
        "PFb": 0.97,
        "PFc": 0.99,
        "PFt": 0.98,
        "Uab": 100.0,
        "Ubc": 99.8,
        "Uca": 99.9,
        "frq": 50.00
      }
    },
    {
      "line_no": 2,
      "pri_val": {},
      "sec_val": {}
    },
    {
      "line_no": 3,
      "pri_val": {},
      "sec_val": {}
    },
    {
      "line_no": 4,
```

```
        "pri_val": {},
        "sec_val": {}
      }
    ],
    "switch": {
      "di1": 1,
      "di2": 0,
      "di3": 1,
      "di4": 0,
      "di5": 1,
      "di6": 0,
      "di7": 1,
      "di8": 0,
      "di9": 1,
      "di10": 0,
      "di11": 1,
      "di12": 0,
      "do1": 1,
      "do2": 0,
      "do3": 1,
      "do4": 0,
      "do5": 1,
      "do6": 0,
      "do7": 1,
      "do8": 0,
      "do9": 1,
      "do10": 0,
      "do11": 1,
      "do12": 0
    },
    "ai_collect": {
      "ai1": 4.52,
      "ai2": 3.86,
      "ai3": 5.21,
      "ai4": 2.99,
      "ai5": 6.10,
      "ai6": 1.75,
      "ai7": 3.22,
      "ai8": 4.91,
      "ai9": 2.18,
      "ai10": 5.47,
      "ai11": 3.06,
      "ai12": 4.33
    }
  }
}
```

### 2. WebSocket 地址：`ws://ip:port/ws/alarm`

**推送内容**：实时报警事件（触发即推）

```
{
  "alarm_type": "line_alarm",
  "time": "2025-01-01 12:00:00",
  "no": "1",
  "type": "",
  "content": "告警内容",
  "level": "告警等级"
},
{
  "alarm_type": "ai_alarm",
  "time": "2025-01-01 12:00:00",
  "no": "2",
  "type": "",
  "content": "告警内容",
  "level": "告警等级"
},
{
  "alarm_type": "operate_alarm",
  "time": "2025-01-01 12:00:00",
  "no": "3",
  "type": "",
  "content": "告警内容",
  "level": "告警等级"
}
```

# 五、数据存储设计

## 5.1 JSON配置文件

- 路径：`/config/`
- 文件：`device.json`、`channel.json`、`setting.json`

## 5.2 SQLite报警数据库

表：`alarm_event`

| 字段 | 类型 | 说明 |
| --- | --- | --- |
| id | INTEGER | 主键 |
| alarm_type | TEXT | 告警类型（line_alarm,ai_alarm,operate_alarm） |
| time | DATETIME | 事件发生时间 |
| no | TEXT | 线路号/通道号 |
| type | TEXT | 类型 |

| 字段 | 类型 | 说明 |
| --- | --- | --- |
| content | TEXT | 告警详情 |
| level | TEXT | 告警等级 |

# 六、安全与可靠性设计

1. **密码加密存储**，接口权限校验
2. WebSocket断线自动重连
3. 配置文件修改自动备份
4. 服务异常自动重启

# 七、总结

本方案基于**RK3568平台**，采用**FastAPI + HTTP + WebSocket**架构，满足电气量测控平台全部需求：

- RESTful API 负责配置、查询、控制
- WebSocket 负责实时数据、报警数据主动推送
- JSON 存储配置，SQLite 存储报警
- 0.5秒实时刷新，界面无卡顿、数据不丢失

系统架构轻量化、高性能，完全适配嵌入式平台运行。

---