中控控制协议说明书（更新版）

1. 概述

本协议用于描述中控系统与数字电柜之间的通信规范。中控系统通过UDP协议向数字电柜发送控制指令，指令需以十六进制（Hex）格式发送，数字电柜根据指令执行相应的电路操作。

2. 通信参数

- **协议类型**: UDP
- **端口号**: 2021
- **数据格式**: 十六进制（Hex）

3. 数据格式

中控系统发送的指令为一个包含16个数字的字符串，格式如下：

[电柜编号, 电路1状态, 电路2状态, ..., 电路15状态]

- **电柜编号**: 表示目标电柜的编号，范围为0到255。
- **电路状态**: 表示对应电路的操作状态，取值范围为：
  - `0`: 关闭电路
  - `1`: 开启电路
  - `0xFF`: 无动作（保持当前状态）

指令需要转换为十六进制格式后发送。例如：

- 字符串指令：`[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]`
- 转换为Hex：`5B 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 5D`

4. 指令示例

- **示例1**: 控制第1号电柜的所有电路关闭
  - 字符串指令：`[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]`
  - Hex格式：`5B 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 5D`
- **示例2**: 控制第2号电柜的第1、3、5电路开启，其余电路关闭
  - 字符串指令：`[1,1,0,1,0,1,0,0,0,0,0,0,0,0,0,0]`
  - Hex格式：`5B 31 2C 31 2C 30 2C 31 2C 30 2C 31 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 2C 30 5D`
- **示例3**: 控制第3号电柜的第2、4电路开启，其余电路保持原状
  - 字符串指令：`[2,0xFF,1,0xFF,1,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF]`

- Hex格式：`5B 32 2C 30 46 46 2C 31 2C 30 46 46 2C 31 2C 30 46 46 2C 30 46 46 2C 30 46 46 2C 30 46 46 2C 30 46 46 2C 30 46 46 2C 30 46 46 2C 30 46 46 2C 30 46 46 5D`

5. 电柜响应

电柜在接收到指令后，会根据指令执行相应的电路操作，并将接收到的指令原样返回给中控系统，以确认指令已接收并执行。

6. 错误处理

- **无效电柜编号**: 如果电柜编号与当前电柜不匹配，电柜将忽略该指令。
- **无效电路状态**: 如果电路状态不是`0`、`1`或`0xFF`，电柜将忽略该电路的操作。

7. 代码实现

以下是中控系统将字符串指令转换为Hex格式并发送的示例代码（C语言）：

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <arpa/inet.h>
#include <unistd.h>

#define UDP_PORT 2021
#define BUFFER_SIZE 1024

// 函数：将字符串指令转换为Hex格式
void stringToHex(const char *input, uint8_t *output, int *output_len) {
    int len = strlen(input);
    *output_len = 0;
    for (int i = 0; i < len; i++) {
        output[(*output_len)++] = (uint8_t)input[i];
    }
}

int main() {
    int sockfd;
    struct sockaddr_in server_addr;
    char buffer[BUFFER_SIZE];
    uint8_t hex_buffer[BUFFER_SIZE];
    int hex_len;

    // 创建UDP套接字
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        exit(EXIT_FAILURE);
    }

    // 设置服务器地址
    memset(&server_addr, 0, sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(UDP_PORT);
```

```
    server_addr.sin_addr.s_addr = inet_addr("192.168.1.100"); // 电柜IP地址

    // 示例指令
    const char *command = "[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]";

    // 将指令转换为Hex格式
    stringToHex(command, hex_buffer, &hex_len);

    // 发送Hex指令
    sendto(sockfd, hex_buffer, hex_len, 0, (const struct sockaddr *)&server_addr,
sizeof(server_addr));
    printf("Sent: %s\n", command);

    // 关闭套接字
    close(sockfd);
    return 0;
}
```

8. 注意事项

- 确保中控系统与数字电柜的网络连接正常。
- 确保电柜编号与中控系统发送的指令一致。
- 电路状态的有效性检查应在发送指令前完成，以避免无效指令的执行。

9. 版本历史

- **V1.0**: 初始版本，定义基本通信协议和指令格式。
- **V1.1**: 更新为Hex格式发送指令。