**交叉带分拣系统通讯协议**

**版本V2.0.0.8**

目录

[1 分拣系统通讯简介 4](#_Toc1959409301)

[1.1通讯系统框图 4](#_Toc409024669)

[1.2通讯协议格式 4](#_Toc1367584538)

[1.3通讯规则 5](#_Toc1027361114)

[2 车载系统与WCS通讯命令详解 6](#_Toc509221229)

[2.1旧版车载协议 6](#_Toc209720895)

[2.1.1车载上线命令 6](#_Toc1233526895)

[2.1.2车载运行命令信息（速度/上料/下料/IO四包数据合一） 6](#_Toc1906133043)

[2.1.3下料口及供包台初始化配置信息 10](#_Toc1106687094)

[2.1.4 车载心跳命令 11](#_Toc1719337293)

[2.1.5分拣许可命令 12](#_Toc328998841)

[2.1.6下料上包配置参数命令 13](#_Toc40409481)

[2.1.7 WCS上线命令 13](#_Toc806244513)

[2.1.8上料运行时参数 14](#_Toc643780909)

[2.1.9测试小车正转/反转 15](#_Toc583564325)

[2.1.10 设置强排口参数 16](#_Toc1688283082)

[2.1.11车载can通讯报警（0x1138）针对旧车载协议 17](#_Toc1745478578)

[2.2 新版车载协议 18](#_Toc304323708)

[2.2.1车载上线命令(0x1101) 18](#_Toc1810988713)

[2.2.2车载运行命令信息（上料/下料/IO三包数据合一）(0x1110) 19](#_Toc1967928988)

[2.2.3 车载故障反馈命令(0x1111) 21](#_Toc883728658)

[2.2.4 车载执行异常反馈命令（0x1112） 23](#_Toc133677705)

[2.2.5 车载心跳命令(0x1150) 24](#_Toc1081973154)

[2.2.6分拣许可命令(0x1121) 24](#_Toc1650187272)

[2.2.7配置参数命令(0x1135) 25](#_Toc1491667885)

[2.2.8 WCS上线命令(0x1120) 26](#_Toc1659586237)

[2.2.9测试小车正转/反转 (0x1133) 26](#_Toc1764849554)

[2.2.10 设置强排口参数(0x1136) 27](#_Toc1454343968)

[2.2.11 头车位置信息(0x1137) 28](#_Toc1519828932)

[2.2.12 车载控制器光电位置信息(0x1139) 29](#_Toc408323137)

[3 地面主控系统与WCS通讯命令详解 31](#_Toc1256655282)

[3.1设置速度控制参数命令 31](#_Toc1331754585)

[3.2地面主控系统实时信息命令 32](#_Toc817347806)

[3.3把指定小车开到维修区 33](#_Toc476756172)

[3.4地面主控系统版本信息 34](#_Toc211632052)

[3.5地面主控系统位置确认信息 34](#_Toc1326569035)

[4 上料系统与WCS通讯命令详解 36](#_Toc686477067)

[4.1待上料小车命令 36](#_Toc1445158947)

[4.2供包系统匹配上包信息命令 37](#_Toc1085218431)

[4.3供包通信转换系统与供包台485通信信息命令 38](#_Toc1793164162)

[5下料口IO控制系统与WCS通讯命令详解 41](#_Toc1017012592)

[5.1下料口按键实时状态 41](#_Toc1414217272)

[5.2下料口上线命令 42](#_Toc1833573643)

[5.3 下料口打印命令 43](#_Toc1823257106)

[5.4 SCADA中关闭滑槽口 44](#_Toc2057998181)

[5.5 清空下料口命令 45](#_Toc269654320)

[5.6 下料口包裹数量实时状态（20220707-燕聪修订） 45](#_Toc1364056540)

[5.7 一键开关命令（20200620-燕聪修订） 47](#_Toc1655993112)

[5.8 下料口操作无效命令（20200624-燕聪修订） 48](#_Toc573978028)

[5.10下料口系统版本信息 49](#_Toc1027561605)

[5.11 WCS发送摆臂位置命令（0x140C） 50](#_Toc1476438452)

[5.12 摆臂实时位置（0x140D） 51](#_Toc1457706687)

[5.13 摆臂实时状态（0x140E） 52](#_Toc1161239310)

[5.14 格口特殊操作方式命令 53](#_Toc410927958)

[5.15 摆臂类型配置信息 54](#_Toc960410311)

[5.16 摆臂基础参数配置 55](#_Toc505423547)

[5.17 下料口从板卡can通讯报警（0x1413） 56](#_Toc2070514196)

[6 PLC控制系统与WCS通讯命令详解 57](#_Toc577776218)

[6.1 IOB信息、急停按钮、报警灯状态、电机状态、防碰撞、48伏故障 57](#_Toc1959767515)

协议文档修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日 期 | 版本 | 说 明 | 负责人 |
| 2021/12/06 | 2.0.0.1 | 车载协议部分均有修改 | 马笑 |
| 2022/3/2 | 2.0.0.2 | （下料口）新增潮汐滑槽摆臂协议5.11、5.12、5.13 | 马笑 |
| 2022/5/11 | 2.0.0.3 | 新增2.12车载can通讯报警 | 丁佳慧 |
| 2022/5/24 | 2.0.0.3 | 修改2.3、2.4、2.7 | 燕聪 |
| 2022/5/26 | 2.0.0.4 | 修改5.4（新增格口闪烁状态，德邦用） | 马笑 |
| 2022/5/30 | 2.0.0.4 | 增加2.2的字段说明解释 | 马笑 |
| 2022/5/30 | 2.0.0.5 | 重构章节2的布局，新旧版本合并 | 马笑 |
| 2022/6/8 | 2.0.0.5 | 修改5.4（新增取消格口闪烁状态，德邦用） | 马笑 |
| 2022/6/14 | 2.0.0.5 | 修改5.4（新增集包失败，马来虾皮用） | 丁佳慧 |
| 2022/07/07 | 2.0.0.5 | 修改5.6( 新增3层的包裹数量) | 燕聪 |
| 2022/10/09 | 2.0.0.6 | 新增3.5（验证车头位置错序问题） | 燕聪 |
| 2022/10/14 | 2.0.0.7 | 新增5.14（LAZADA需要区分格口操作方式） | 马笑 |
| 2023/1/4 | 2.0.0.8 | 新增2.2.12（车载从板加计数、复位光电） | 燕聪 |
| 2023/1/10 | 2.0.0.8 | 新增5.17（下料口从板卡can通讯报警） | 燕聪 |

# 分拣系统通讯简介

## 1.1通讯系统框图

分拣系统各硬件模块间的通讯接口框图如下：



## 1.2通讯协议格式

各硬件模块与WCS采用TCP和UDP协议通讯，其中，WCS与车载、IO下料口、供包台、主线速度控制系统都是采用UDP通讯协议，WCS与相机之间采用TCP/IP通讯协议，WCS模块为客户端，主动连接相机，两者之间保持长连接状态。

数据帧格式：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | 帧序号 | 帧长度 | 帧校验 | 帧类型 | 数据区 |

帧 头：2个字节，区分每一帧数据，帧头约定为“0xAAAA”。

帧序号：4个字节，低3位字节发送方每一帧自动加1，应答方应答时使用接收到发送方使用的帧序号。

帧长度：2个字节，表示整个帧的数据长度。

帧校验：1个字节，对帧数据中的帧命令和数据区的数据进行异或计算。

帧类型：2个字节，帧的命令类型代码，表示该帧数据的功能，发送命令最高位为0，应答方使用的应答命令为接收到的命令最高位或1。

数据区：每条命令的具体数据，详见命令说明。

**注**：通信协议帧中的数据全部为小端格式。

## 1.3通讯规则

采用UDP通讯协议的模块之间，通过“一问一答”的方式，发送命令方发送命令后，必须等待接收方的应答后，才能确定对方已经收到，并修改相应的小车的状态，如果对方没有接收到，不影响后续消息的发送，需继续发送后面的消息。

与相机之间的连接采用TCP通讯协议，WCS模块作为TCP客服端，主动连接相机，相机把小车号和其他信息发送出来，两者之间保持长连接状态。

# 车载系统与WCS通讯命令详解

## 2.1旧版车载协议

### 2.1.1车载上线命令

（1）发送时间

车载系统与WCS每次建立连接成功后发送，发送方为车载系统，WCS接收到后，要把车载系统的版本号写在日志中，用于系统维护。

（2）发送帧命令：0x1101

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned int16 car\_version; //车载系统的版本号

} sCar2WCS\_online \_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned int16 | car\_version | 车载系统的版本号 |

（4）应答帧命令：0x9101

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sCar2WCS\_online\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

### 2.1.2车载运行命令信息（速度/上料/下料/IO四包数据合一）

（1）发送时间

WCS接收到供包信息后发送，发送方为WCS，车载应答命令中加载车载系统的故障信息，如果WCS在80ms内没收到车载的反馈，则会在80ms后重发一次。

（2）发送帧命令：0x1301

（3）数据格式

发送实时速度信息、上包信息、下料信息以及下料口状态信息。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cmd[11] | 实时速度信息区 | 上包信息区 | 下料信息区 | 下料口IO状态信息区 |

cmd[11]：详见1.2 通讯协议格式

**实时速度信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned int32 | Speed | WCS接收到地面系统发送上来的实时速度(实际速度（米/秒）\*100)后，转发给车载系统。 |

**上料信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | car\_load\_num | 上料的小车的数量。 |
| sCar\_Load\_Info | scar\_Load\_Info[car\_load\_num] | 需上料的小车数据。 |

上料信息数据结构体格式如下：

通用版格式：

typedef struct

{

unsigned int16 car\_load\_index; //每个上包的小车编号

unsigned char load\_no; //上料的供包台编号

unsigned int16 car\_load\_lenght; //上料货物的长度，单位mm

unsigned int16 car\_load\_offset; //上料货物偏离中心位置，单位mm

signed int16 car\_load\_delay;//上料延时参数（毫秒）

} sCar\_Load\_Info;

顺丰供包机特殊版：

typedef struct

{

unsigned int16 car\_load\_index; //每个上包的小车编号

unsigned char load\_no; //上料的供包台编号

unsigned int16 Car\_run\_time;//小车电机运行持续时间1~255，单位 10ms

unsigned int16 car\_load\_offset; //上料货物偏离中心位置，单位mm

unsigned int16 Car\_wait\_time;//小车电机运转等待时间1~127，单位 10ms

} sCar\_Load\_Info;

**下料信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | car\_ unload \_num | 单次控制小车下料的数量。 |
| sCar\_Unload\_Data | scar\_Unload\_Data[car\_ unload \_num] | 需下料的小车数据。 |

下料信息数据结构体格式如下：

typedef struct

{

unsigned int16 car\_unload\_index; //每个下料的小车编号

unsigned int16 car\_unload\_exitno; //每个下料口的位置（以车载光电开关为序）

unsigned char car\_unload\_direction; //每个下料口的方向

signed int16 car\_unload\_delay; //下料时小车启动的延时时间（毫秒），是根据包裹在小车的位置，获取参数设置中九宫格里设置的延时时间。

unsigned int16 car\_rotate\_length; //每个小车转动的皮带长度

} sCar\_Unload\_Data;

**下料口IO状态信息**区数据格式如下：

该信息把所有关闭状态的下料口发送到车载，若没有关闭状态的下料口，则信息区无数据。格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned int16 | car\_ io\_num | IO关闭的个数。 |
| sIO\_Info | sio\_Info[car\_ io\_num] | 下料IO口的数据。 |

下料I/O口信息数据结构体格式如下：

typedef struct

{

unsigned int16 io\_addr; //下料口信息表的序号

} sIO\_Info;

（4）应答帧命令：0X9301

（5）应答数据区

typedef struct

{

unsigned char cmd[11];

unsigned int32 Receive\_count;

unsigned int16 statue\_guangdian\_num;

unsigned char car\_state\_ack\_num;

unsigned int16 car\_serial\_num;

unsigned char car\_state;

}  sCar2WCS\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned int32 | Receive\_count | 车载累计接收到WCS的命令个数 |
| unsigned int16 | statue\_guangdian\_num | 车载错误信息：  高8位为故障类型：  BIT15—头车光电计数多一个；  BIT14—头车光电计数少一个；  BIT13—头车光电触发时间波动较大  （头车发生任何故障，全线均禁止分拣）；  BIT12—从车丢了一个光电（整组仅颜色显示）；  BIT11—从车连续丢掉至少两个光电（整组不允许分拣，且颜色显示）；  BIT10—从车丢了4包合一数据（整组仅颜色显示）；  BIT9—备用；  BIT8—备用；  低8位：  出问题的小车组号 |
| unsigned char | car\_state\_ack\_num | 有状态返回的小车的个数 |
| unsigned int16 | car\_serial\_num | 有状态返回的小车的序号 |
| unsigned char | car\_state | 状态信息用一个字节表示，BIT7：0表示小车485通讯正常，1表示小车485通讯异常； BIT3：0表示电滚筒无过压，1表示电滚筒过压；BIT2：0表示电滚筒皮带正常，1表示电滚筒皮带异常；BIT1：0表示电滚筒无过流，1表示电滚筒过流； |
| …… | …… | 根据car\_state\_ack\_num，每组数据按car\_serial\_num和car\_state依次存放在数据包中 |

### 2.1.3下料口及供包台初始化配置信息

（1）发送时间

接收到下料口上线命令（0x1403）后，再接收到车载上线命令（0x1101）后发送，发送方为WCS，如果WCS没有收到车载的反馈，WCS不会重发。

（2）发送帧命令：0x1130

（3）数据格式

该命令帧发送下料口参数以及供包参数，每帧数据长度不超过200字节，可以分包发送。帧格式如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cmd[11] | 数据类型 | 起始地址 | 数据个数 | **数据区** |

信息格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned char | data\_type | 数据类型：1字节。 0：表示数据区数据为下料口信息；  1：表示数据区数据为供包台信息 |
| unsigned int16 | start\_address | 起始地址：数据区里的信息分别在下料口信息表或者供包信息表里的序号 |
| unsigned int16 | actual\_num | 数据个数：数据区里的信息分别有几组下料口信息或者供包信息 |
| 数据区 | …… | …… |

数据区：下料口信息或者供包信息，按各自的信息格式排列。

**数据区-下料口**信息格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned int16 | exit\_position | 每个下料口的位置（以车载光电开关为序） |
| unsigned char | exit\_direction | 每个下料口的方向（里侧、外侧） |
| unsigned char | exit\_speed | 每个下料口的卸载速度 |
| unsigned char | exit\_stop | 每个下料口的关闭状态（0：打开，1：关闭） |

**数据区-供包台**信息格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned int16 | load\_position | 供包台的位置（以车载光电开关为序） |
| unsigned char | load\_direction | 供包台上包的方向 |
| unsigned int16 | a\_u16 | 乘法因子 a (16bit)采用定点模式,如实际值是 0.8 ,这里 a\_u16 == 800; |
| unsigned int16 | b\_u16 | 加法算子 b（16bit）,采用定点方式，如实际值是 100，这里 b\_u16 == 100; 单位： mm |
| signed int16 | c\_s16 | 延迟启动参数,可正可负，16bit, 单位：ms |

（4）应答帧命令：0x9130

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

unsigned long Receive\_count;

} sCar2WCS\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned int32 | Receive\_count | 车载累计接收到WCS的命令个数 |

### 2.1.4 车载心跳命令

（1）发送时间

心跳命令，WCS与车载连续5秒未发生数据交互时发送，发送方为车载系统。

（2）发送帧命令：0x1150

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

} sCar2WCS\_heart\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

（4）应答帧命令：0x9150

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sCar2WCS\_heart\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

### 2.1.5分拣许可命令

（1）发送时间

当分拣系统开启或者关闭的时候WCS发送给车载系统，用于同步车载系统与WCS系统车辆运行状态的同步。WCS系统根据主线速度是否处于合法速度范围决定分拣系统的打开关闭状态。发送方为WCS。

（2）发送帧命令：0x1121

（3）数据格式定义

表示该控制器后面的小车数量以及小车是否正常，具体格式如下：

typedef struct

{

unsigned char cmd[11];

unsigned char sys\_en;

} sWCS2Car\_SYS\_EN\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned char | sys\_en | 0：关闭 1：开启。 |

（4）应答帧命令：0X9121

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

unsigned long Receive\_count;

} sWCS2Car\_SYS\_EN\_Data \_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned int32 | Receive\_count | 车载累计接收到WCS的命令个数 |

### 2.1.6下料上包配置参数命令

（1）发送时间

车载上线后以及参数更改后发送。发送方为WCS。

（2）发送帧命令：0x1135

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned int16 car\_num; //分拣线小车的数量

unsigned int16 unload\_moto\_run\_lenght; //下料小车运行长度mm

unsigned int16 mainline\_speed; //主线运行速度（乘以1000表示，如1.35表示成1350）

} sWCS2Car\_unload\_load\_run\_Data;

### 2.1.7 WCS上线命令

（1）发送时间

WCS启动成功，在收到车载的心跳包后，判断是否已发送初始化消息给车载，如果没有发送，则发送WCS上线命令。

（2）发送帧命令：0x1120

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

} sWCS2Car\_online\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

（4）应答帧命令：0x9120

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sWCS2Car\_online\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

### 2.1.8上料运行时参数

（1）发送时间

系统运行时对供包台上料参数进行修改时发送。

（2）发送帧命令：0x1132

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned char adjust\_count;

sLoad\_Platform\_Paras paras\_t[adjust\_count];

} sLoadRuntime\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned char | adjust\_count | 发送adjust\_count个供包台对应的上料运行时参数 |
| sLoad\_platform\_paras | paras\_t[adjust\_count] | 具体的上料运行时参数 |

上料系统的包裹信息数据结构体格式如下：

typedef struct

{

unsigned char platform\_index\_u8; //要调整上料运行时参数的供包台索引（现实中我们定义的//供包台号为1---24，对应到程序中为0---23， 这里的//platform\_index\_u8的值应采用程序规则）

unsigned int16 a\_u16; //乘法因子a 16bit, 采用定点模式，如实际值是0.8，这里//a\_u16 == 800。

unsigned int16 b\_u16; //加法因子b 16bit, 采用定点模式，如实际值是100，这里

//b\_u16 == 100, 单位: mm.

signed int16 c\_s16; //延迟启动参数，可正可负， 单位：ms。

} sLoad\_platform\_paras

（4）应答帧命令：0x9132

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sLoadRuntime\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

### 2.1.9测试小车正转/反转

（1）发送时间

WCS停止时，发送该消息到车载系统，用于检修小车是否正常。

（2）发送帧命令：0x1133

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned int16 car\_no;

unsigned char turn\_direction;

unsigned char speed;

unsigned int16 length;

} sCheckCar\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned char | car\_no | 需要测试转动的小车号 |
| unsigned char | turn\_direction | 小车转动的方向（与下料口的方向定义一致） |

（4）应答帧命令：0x9133

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sCheckCar\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

### 2.1.10 设置强排口参数

（1）发送时间

系统初始化时发送。

（2）发送帧命令：0x1136

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned char fixed\_port\_num;

sFixed\_Port\_Data fixed\_port\_data\_t [fixed\_port\_num];

} sFixed\_Port\_Info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |
| unsigned char | fixed\_port\_num | 强排口的数量 |
| sFixed\_Port\_Data | fixed\_port\_data\_t [fixed\_port\_num] | 具体的强排口位置参数 |

强排口位置参数数据结构体格式如下：

typedef struct

{

u16 fixed\_position; //强排口位置(格式同普通下料口)

u8 fixed\_direction; //方向

u8 fixed\_speed; //速度

u16 fixed\_length; //转动长度

u16 fixed\_delay; //延迟时间

} sFixed\_Port\_Data;

（4）应答帧命令：0x9136

（5）应答数据区

具体格式如下

typedef struct{

unsigned char cmd[11];

} sFixed\_Port\_Info\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](#_1.2通讯协议格式) |

### 2.1.11车载can通讯报警（0x1138）针对旧车载协议

（1）发送时间

每10s发送一次。

（2）发送帧命令：0x1138

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned char num;

unsigned char index[num];

} sFixed\_Port\_Info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned char | num | 从板卡can通讯报警的数量 |
| Unsigned char | index[num] | 具体的can通讯异常的板卡号 |

（4）应答帧命令：无需应答

## 2.2 新版车载协议

### 2.2.1车载上线命令(0x1101)

（1）发送时间

车载系统与WCS每次建立连接成功后发送，发送方为车载系统，WCS接收到后，要把车载系统的版本号写在日志中，用于系统维护。

（2）发送帧命令：0x1101

（3）数据格式定义

typedef struct{

unsigned char cmd[11];

unsigned int16 car\_version; //车载系统的版本号

} sCar2WCS\_online \_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned int16 | car\_version | 车载系统的版本号 |

（4）应答帧命令：0x9101

（5）应答数据区

具体格式如下

typedef struct{

unsigned char cmd[11];

} sCar2WCS\_online\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式)[讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.2车载运行命令信息（上料/下料/IO三包数据合一）(0x1110)

（1）发送时间

WCS接收到供包信息后发送，发送方为WCS，车载应答命令中加载车载系统的故障信息，如果WCS在80ms内没收到车载的反馈，则会在80ms后重发一次。

（2）发送帧命令：0x1110

（3）数据格式

发送上包信息、下料信息以及下料口状态信息。

|  |  |  |  |
| --- | --- | --- | --- |
| cmd[11] | 上包信息区 | 下料信息区 | 下料口IO状态信息区 |

cmd[11]：详见1.2 通讯协议格式

**上料信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | car\_load\_num | 上料的小车的数量 |
| sCar\_Load\_Info | scar\_Load\_Info[car\_load\_num] | 需上料的小车数据 |

上料信息数据结构体格式如下：

typedef struct{

unsigned int16 car\_load\_index; //每个上包的小车编号

unsigned int16 car\_load\_position; //供包台光电位置

unsigned int16 car\_load\_length; //小车皮带转动长度

unsigned int16 car\_load\_delay;//小车转动延时(ms)

unsigned char car\_load\_direction; //小车转动方向

unsigned char car\_load\_speed;//小车转动速度

} sCar\_Load\_Info;

**下料信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | car\_ unload \_num | 单次控制小车下料的数量。 |
| sCar\_Unload\_Data | scar\_Unload\_Data[car\_ unload \_num] | 需下料的小车数据。 |

下料信息数据结构体格式如下：

typedef struct{

unsigned int16 car\_unload\_index; //每个下料的小车编号

unsigned int16 car\_unload\_exitno; //每个下料口的位置（下料口光电取整，不再\*100）

unsigned int16 car\_unload \_length; //小车转动的皮带长度(软件下发保持不变，根据配置来，如120)

unsigned int16 car\_unload\_delay; //小车启动的延时时间(ms)（格口位置的小数部分换算时间+九宫格最终延时）

unsigned char car\_unload\_direction; //小车转动方向

unsigned char car\_unload\_speed; //卸载速度

unsigned int16 chute\_index; //格口编号

} sCar\_Unload\_Data;

**下料口IO状态信息**区数据格式如下：

该信息把所有关闭状态的下料口发送到车载，若没有关闭状态的下料口，则信息区无数据。格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned int16 | car\_ io\_num | IO关闭的个数。 |
| sIO\_Info | sio\_Info[car\_ io\_num] | 下料IO口的数据。 |

下料I/O口信息数据结构体格式如下：

typedef struct{

unsigned int16 chute\_index; //关闭的格口编号

} sIO\_Info;

由WCS决定是发送所有关闭的格口号，还是只发送当前有分拣任务的格口号。

（4）应答帧命令：0X9110

（5）应答数据区

typedef struct{

unsigned char cmd[11];

}  sCar2WCS\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.3 车载故障反馈命令(0x1111)

（1）发送时间

车载有故障就发，每秒发送一次，发送方为车载系统，无异常 则不发。

（2）发送帧命令：0x1111

（3）数据格式定义

typedef struct{

unsigned char cmd[11];

car\_group\_error\_num; //以组为单位的故障组数;

car\_group\_error\_Info [car\_group\_error\_num]; //组故障信息数据;

car\_error\_num； //以单个车为单位的故障车数;

car\_type; // 驱动器类型（应等同于软件的配置）;

car\_error\_Info[car\_error\_num]； //小车故障信息数据;

} Car\_error\_Data;

以组为单位的故障，数据区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | car\_group\_error\_num | 故障组数 |
| car\_group\_error\_Info | car\_group\_error\_Info [car\_group\_error\_num] | 故障信息 |

故障信息数据结构体格式如下：

typedef struct{

unsigned char car\_group\_index; //组号

unsigned char group\_error\_type; //组故障类型

} car\_group\_error\_Info;

组故障类型说明：

BIT0：头车光电计数多一个；

BIT1：头车光电计数少一个；

BIT2：从车丢了一个光电（整组仅颜色显示）；

BIT3：从车连续丢掉至少两个光电（整组不允许分拣，且颜色显示）；

BIT4：从车丢了分拣指令数据（整组仅颜色显示）；

BIT5：从站未连接

以单个小车为单位的故障，数据区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | car\_error\_num | 故障车数 |
| Unsigned char | Car\_type | 驱动器类型 1、2 |
| Car\_error\_Info | car\_error\_Info[car\_error\_num] | 故障信息 |

故障信息数据结构体格式如下：

typedef struct{

unsigned int16 car\_index; //车号

unsigned int8 error\_type; //故障类型

} Car\_error\_Info;

驱动器类型1小车故障类型：

BIT3：电滚筒过压；

BIT2：电滚筒皮带异常；

BIT1：电滚筒过流；

BIT0：小车485通讯异常；

驱动器类型2小车故障类型：

1： 母线过流 2：母线过压 3：母线欠压 5：电机过流 6：电机过载 7： 电机堵转

8： 电机过速 9：电机HALL故障 10：电机故障 12：无编码器信号 13： 系统软件故障 14： 看门狗复位-保护复位 15 电机转过的位置未达到指令要求 16 编码器零位

故障 0xFE：回复消息的格式不正确 0xFF: 未连接

（4）应答帧命令：0x9111

（5）应答数据区

具体格式如下

typedef struct{

unsigned char cmd[11];

} Car\_error\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.4 车载执行异常反馈命令（0x1112）

（1）发送时间

发送方为车载系统，每一个车次发送一次，无异常 则不发。

（2）发送帧命令：0x1112

（3）数据格式定义

typedef struct{

unsigned char cmd[11];

unsigned int16 carnum; //异常下料小车数量

struct car\_state[carnum]; //具体异常信息

} sCar2WCS\_heart\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned int16 | carnum | 异常下料小车数量 |
| struct | car\_state[carnum]; | 具体异常信息 |
| … |  |  |

STRUCT car\_state{

U16 car\_num; //小车编号

U8 state; // 1、车载未发运行指令 2、关闭格口时，小车已下发运行指令

}

（4）应答帧命令：0x9112

（5）应答数据区

具体格式如下

typedef struct{

unsigned char cmd[11];

} sCar2WCS\_heart\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.5 车载心跳命令(0x1150)

（1）发送时间

心跳命令，WCS与车载连续5秒未发生数据交互时发送，发送方为车载系统。

（2）发送帧命令：0x1150

（3）数据格式定义

typedef struct{

unsigned char cmd[11];

} sCar2WCS\_heart\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

（4）应答帧命令：0x9150

（5）应答数据区

具体格式如下

typedef struct{

unsigned char cmd[11];

} sCar2WCS\_heart\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.6分拣许可命令(0x1121)

（1）发送时间

当分拣系统开启或者关闭的时候WCS发送给车载系统，用于同步车载系统与WCS系统车辆运行状态的同步。WCS系统根据主线速度是否处于合法速度范围决定分拣系统的打开关闭状态。发送方为WCS。

（2）发送帧命令：0x1121

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned char sys\_en;

} sWCS2Car\_SYS\_EN\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned char | sys\_en | 0：关闭 1：开启。 |

（4）应答帧命令：0X9121

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

unsigned long Receive\_count;

} sWCS2Car\_SYS\_EN\_Data \_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned int32 | Receive\_count | 车载累计接收到WCS的命令个数 |

### 2.2.7配置参数命令(0x1135)

（1）发送时间

车载上线后以及参数更改后发送。发送方为WCS。

（2）发送帧命令：0x1135

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned int16 total\_car\_num; //分拣线小车的数量

unsigned int16 mainline\_speed; //主线运行速度（乘以1000表示，如1.35表示成1350）

unsigned int8 car\_mode; // 1：单车; 2：一车双带;

unsigned int8 driver\_mode; // 1表示类型1驱动器，2表示类型2驱动器（软件默认配置1，如有需要现场修改配置）

unsigned int8 onegroup\_carnum; // 一个从控制器控制多少个小车 （软件新增参数，默认16，新线应该配置为20，需交代技术支持）

} sWCS2Car\_unload\_load\_run\_Data;

### 2.2.8 WCS上线命令(0x1120)

（1）发送时间

WCS启动成功，在收到车载的心跳包后，判断是否已发送初始化消息给车载，如果没有发送，则发送WCS上线命令。

（2）发送帧命令：0x1120

（3）数据格式定义

typedef struct{

unsigned char cmd[11];

} sWCS2Car\_online\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

（4）应答帧命令：0x9120

（5）应答数据区

具体格式如下

typedef struct{

unsigned char cmd[11];

} sWCS2Car\_online\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.9测试小车正转/反转 (0x1133)

（1）发送时间

WCS停止时，发送该消息到车载系统，用于检修小车是否正常。

（2）发送帧命令：0x1133

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned int16 car\_no;

unsigned char turn\_direction;

unsigned char speed;

unsigned int16 length;

} sCheckCar\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned char | car\_no | 需要测试转动的小车号 |
| unsigned char | turn\_direction | 小车转动的方向（与下料口的方向定义一致） |

（4）应答帧命令：0x9133

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sCheckCar\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.10 设置强排口参数(0x1136)

（1）发送时间

系统初始化时发送。

（2）发送帧命令：0x1136

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned char fixed\_port\_num;

sFixed\_Port\_Data fixed\_port\_data\_t [fixed\_port\_num];

} sFixed\_Port\_Info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned char | fixed\_port\_num | 强排口的数量 |
| sFixed\_Port\_Data | fixed\_port\_data\_t [fixed\_port\_num] | 具体的强排口位置参数 |

强排口位置参数数据结构体格式如下：

typedef struct

{

u16 fixed\_position; //强排口位置(格式同普通下料口)

u8 fixed\_direction; //方向

u8 fixed\_speed; //速度

u16 fixed\_length; //转动长度

u16 fixed\_delay; //延迟时间

} sFixed\_Port\_Data;

（4）应答帧命令：0x9136

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sFixed\_Port\_Info\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

### 2.2.11 头车位置信息(0x1137)

（1）发送时间

车载头车位置改变时，主动发送（WCS只负责打印记录）

（2）发送帧命令：0x1137

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned 16 position;

unsigned 16 time;

} sCar\_position;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned 16 | position | 头车位置 |
| unsigned 16 | time | 头车位置发生变化的时间间隔 |

1. 应答帧命令：无需应答

### 2.2.12 车载控制器光电位置信息(0x1139)

（1）发送时间

车载控制器位置改变时，主动发送（WCS只负责打印记录）

（2）发送帧命令：0x1139

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned 16 ctrlindex;

unsigned 16 position;

unsigned 16 time;

} sCar\_position;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned 16 | ctrlindex | 控制器编号 |
| unsigned 16 | position | 光电位置 |
| unsigned 16 | time | 头车位置发生变化的时间间隔 |

（4）应答帧命令：无需应答

# 地面主控系统与WCS通讯命令详解

## 3.1设置速度控制参数命令

（1）发送时间

WCS点击系统启动或者系统停止以及设置速度控制参数时发送。发送方为WCS。

（2）发送帧命令：0x1220

（3）数据格式

Typedef struct

{

uint8 cmd[11];

float max\_speed;

float min\_speed;

uint32 fwKp;

uint32 fwKi;

uint32 fwKd;

floats peed\_target;

uint32 speed\_factor;

uint32 pe\_distance;

uint16 motorRate

}sWCS2GndCtrl\_Para\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| float | max\_speed | 最大速度 |
| float | min\_speed | 最小速度 |
| Uint32 | fwKp | Pid参数 |
| Uint32 | fwKi | Pid参数 |
| Uint32 | fwKd | Pid参数 |
| float | speed\_target | 目标速度 |
| Uint32 | speed\_factor | 目标速度与变频器频率的系数 |
| Uint32 | pe\_distance | 两个光电开关距离 |
| Uint16 | motorRate | 电机的参考频率 |

（4）应答帧命令：0x9220

（5）应答数据区

具体格式如下：

typedef struct

{

unsigned char cmd[11];

} sWCS2GndCtrl\_Para\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 3.2地面主控系统实时信息命令

（1）发送时间

主线运行一个小车车位发送。发送方为地面系统。

（2）发送帧命令：0x1201

（3）数据格式

typedef struct

{

uint8 cmd[11];

uint32 speed;

uint16 carNo;

uint16 carState;

uint16 motorRate;

}sGndCtrl2WCS\_CMD\_FAULT\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint32 | speed | 主线速度（需要WCS换算） |
| Uint16 | carNo | 当前第一个光电挡板上的小车号 |
| Uint16 | carState | 小车状态(有无包裹)（0：无；1：有）  bit0：第一个检测位  bit1：第二个检测位 |
| Uint16 | motorRate | 直线电机的频率，地面控制系统发送4位10进制的数据过来，例如地控发送2050过来，WCS接收到后，要换算成20.5，保留小数后一位。 |

（4）应答帧命令：0x9201

（5）应答数据区

具体格式如下：

typedef struct

{

unsigned char cmd[11];

} sGndCtrl2WCS\_CMD\_FAULT\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 3.3把指定小车开到维修区

（1）发送时间

WCS在分拣线运行或停止时，发送该消息到地面控制系统。

（2）发送帧命令：0x1221

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned int16 car\_no;

unsigned int16 repair\_locate;

unsigned int16 adjust\_para;

unsigned int16 motorRate

} sRepairCar\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned char | car\_no | 需要定位到维修区的小车号 |
| unsigned int16 | repair\_locate | 维修区的起始光电位置 |
| unsigned int16 | car\_count | 小车总数 |
| unsigned int16 | adjust\_para | 可调节系数（小车数，提前几个车开始减速） |
| unsigned int16 | motorRate | 固定的电机频率 |
| unsigned float 32 | target\_speed | 目标速度 |

（4）应答帧命令：0x9221

（5）应答数据区

具体格式如下

typedef struct

{

unsigned char cmd[11];

} sRepairCar \_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 3.4地面主控系统版本信息

（1）发送时间

WCS给地控系统发送启停命令后，地控系统主动发送，与9220不冲突。

（2）发送帧命令：0x1222

（3）数据格式

typedef struct

{

uint8 cmd[11];

Uint8 version[4];版本号小端格式

}sGndCtrl2WCS\_CMD\_version\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | version[4] | 版本号小端格式（比如实际版本号 2.0.0.8 则对应 0x08,0x00,0x00,0x02） |

1. 应答帧命令：无需应答，WCS打印解析即可

## 3.5地面主控系统位置确认信息

（1）发送时间

地控系统位置发生变化时主动发送，WCS接收解析打印日志即可。

（2）发送帧命令：0x1223

（3）数据格式

typedef struct

{

uint8 cmd[11];

Uint16 PreCarNum;//上一次的位置

Uint16 CurCarNum;//当次的位置

Uint16 interval;//位置发生变化的时间间隔

}sGndCtrl2WCS\_CMD\_interval\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | PreCarNum | 上一次的车头位置 |
| Uint16 | CurCarNum | 当次的车头位置 |
| Uint16 | interval | 位置发生变化的时间间隔 |

（4）应答帧命令：无需应答，WCS打印解析即可

# 上料系统与WCS通讯命令详解

## 4.1待上料小车命令

（1）发送时间

主线运行一个小车车位发送。发送方为WCS。

（2）发送帧命令：0x1320

（3）数据格式

typedef struct

{

Uint8 head[11];

Uint8 load\_system\_num;

sLoad\_System\_Car\_Info sLoad\_System\_Car\_Info;

} sWCS2LoadSysem\_CMD\_Data;

（4）命令说明

**改成支持多个上料板卡，每个上料板卡支持12个供包台，WCS发送待上料小车命令时，需根据小车位置计算该命令发送到哪个上料板卡。**

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | load\_system\_num | 上包台个数（0~12个，有几个供包台就有几个小车的数据） |
| sLoad\_System\_Car\_Info | sLoad\_System\_Car\_Info[load\_system \_num] | 需上料的小车数据，按照供包台内部编号顺序排序 |

上料小车信息数据结构体格式如下：

typedef struct

{

Uint16 load\_car\_num //待上料小车号。

Uint8 load\_car\_state //0：不可上料1：可以上料。

}sLoad\_System\_Car\_Info;

（5）应答帧命令：0x9320

（6）应答数据区

具体格式如下：

typedef struct

{

unsigned char cmd[11];

}  sWCS2LoadSysem\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 4.2供包系统匹配上包信息命令

（1）发送时间

供包匹配上包信息。发送方为供包系统。

（2）发送帧命令：0x1301

（3）数据格式

typedef struct

{

Uint8 head[11];

Uint8 load\_system\_num;

sLoad\_System\_Pack\_Info sLoad\_System\_Pack\_Info;

} sLoadSysem2WCS\_CMD\_Data;

（4）命令说明

**改成支持多个上料板卡，每个上料板卡支持12个供包台，WCS要支持同时接收多个上料板卡的消息。**

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | load\_system\_num | 上包台个数（0~12个） |
| sLoad\_System\_Pack\_Info | sLoad\_System\_Pack\_Info [load\_system \_num] | 需上料的包裹数据。 |

上料系统的包裹信息数据结构体格式如下：

typedef struct

{

uint16 load\_car\_num; //待上料小车号1

uint16 load\_car\_num; //待上料小车号2

uint16 pack\_length; //包裹的尺寸（宽度，与旧协议一样）(单位：mm)，0表示无包裹

uint16 pack\_position; //包裹中心离短边侧的距离(单位：mm)

uint16 pack\_width; //包裹宽度(单位：mm)

uint16 pack\_height; //包裹高度(单位：mm)

uint32 pack\_weight; //包裹重量(单位：g)

uint16 pack\_length //包裹长度(单位：mm)

uint32 sortcode; //包裹分拣编码

uint32 back; //预留

随机长度 barcode//条码

} sLoad\_System\_Pack\_Info

（5）应答帧命令：0x9301

（6）应答数据区

具体格式如下：

typedef struct

{

unsigned char cmd[11];

} sLoadSysem2WCS\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 4.3供包通信转换系统与供包台485通信信息命令

（1）发送时间

PTC接收到WCS发送的待上料命令（0x1320）后立即转发。发送方为PTC，接收方为供包台。

（2）数据帧格式

typedef struct

{

Uint8 head; //帧头

Uint16 load\_car\_num; //待上料小车号

Uint8 load\_car\_state //0：不可上料1：可以上料。

Uint8 sum\_check //异或校验

} sPTC2LoadSystem\_CMD\_Data;

（3）命令说明

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | head | 0xAA |
| Uint16 | load\_car\_num | 待上料小车号 |
| Uint8 | load\_car\_state | 0：不可上料1：可以上料 |
| Uint8 | sum\_check | 异或校验(sum\_check之前所有字节异或) |

（4）应答帧格式

typedef struct

{

uint8 head; //帧头

uint16 load\_car\_num; //待上料小车号

uint16 pack\_position; //包裹中心离短边侧的距离(单位：mm)

uint16 pack\_length; //包裹长度(单位：mm)，0表示无包裹

uint16 pack\_width; //包裹宽度(单位：mm)

uint16 pack\_height; //包裹高度(单位：mm)

uint32 pack\_weight; //包裹重量(单位：g)

uint8 sum\_check //异或校验

} sLoadSystem2PTC\_CMD\_Data;

（5）命令说明

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | head | 0x55 |
| Uint16 | load\_car\_num | 待上料小车号 |
| Uint16 | pack\_position | 包裹中心离短边侧的距离(单位：mm) |
| Uint16 | pack\_length | 包裹长度(单位：mm)，0表示无包裹 |
| Uint16 | pack\_width | 包裹宽度(单位：mm) |
| Uint16 | pack\_height | 包裹高度(单位：mm) |
| Uint32 | pack\_weight | 包裹重量(单位：g) |
| Uint8 | sum\_check | 异或校验(sum\_check之前所有字节异或) |

# 5下料口IO控制系统与WCS通讯命令详解

## 5.1下料口按键实时状态

（1）发送时间

下料口系统每隔500毫秒发送所有下料口的状态信息，发送方为下料口系统。

（2）发送帧命令：0x1402

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 close\_addr\_num;

uint16 close\_addr[close\_addr\_num];

uint16 full\_addr\_num;

uint16 full\_addr[full\_addr\_num];

uint16 block\_addr\_num;

uint16 block\_addr[block\_addr\_num];

uint16 door\_addr\_num;

uint16 door\_addr[door\_addr\_num];

uint16 Pause\_addr\_num；

uint16 Pause\_addr[Pause\_addr\_num];

}sIOCtrl2WCS\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | close\_addr\_num | 关闭的格口数量 |
| Uint16 | close\_addr | 关闭的格口地址 |
| Uint16 | full\_addr\_num | 满袋的格口数量 |
| Uint16 | full\_addr | 满袋的格口地址 |
| Uint16 | block\_addr\_num | 卡包的组的数量 |
| Uint16 | block\_addr | 卡包组的地址 |
| Uint16 | door\_addr\_num | 门关上的格口数量 |
| Uint16 | door\_addr | 门关上的格口地址 |
| Uint16 | Pause\_addr\_num | 暂停的格口数量 |
| Uint16 | Pause\_addr | 暂停的格口地址 |

（4）应答帧命令：0x9402

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sIOCtrl2WCS\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.2下料口上线命令

（1）发送时间

下料口系统与WCS每次建立连接成功后发送，发送方为下料口系统，WCS接收到后，要把下料口控制器的数量发送给下料口系统。

（2）发送帧命令：0x1403

（3）数据格式定义

typedef struct

{

uint8 cmd[11];

} sIoCtrl2WCS\_online \_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

（4）应答帧命令：0x9403

（5）应答数据区

具体格式如下

typedef struct

{

uint8 char cmd[11];

uint16 ioCtrl\_num;

} sIoCtrl2WCS\_online \_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | ioCtrl\_num | 下料口控制板的个数 |

## 5.3 下料口打印命令

（1）发送时间

打印按钮触发时由下料口系统主动发送。

（2）发送帧命令：0x1404

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 print\_addr\_num;

uint16 print\_addr[print\_addr\_num];

}sIOPrt2WCS\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | print\_addr\_num | 需打印的格口数量 |
| Uint16 | print\_addr | 需打印的格口地址 |

（4）应答帧命令：0x9404

（5）应答数据区

typedefstruct

{

uint8 cmd[11];

uint16 print\_addr\_num;

uint16 print\_addr[print\_addr\_num];

}sIOPrt2WCS\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | print\_addr\_num | 打印完成的格口数量 |
| Uint16 | print\_addr | 打印完成的格口地址 |

## 5.4 SCADA中关闭滑槽口

（1）发送时间

用户在SCADA中关闭下料口后，WCS马上发送关闭命令给下料口系统。

（2）发送帧命令：0x1405

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint8 close\_cmd;

uint16 close\_addr\_num;

uint16 close\_addr[close\_addr\_num];

}sWCS2IOCtrl\_Close\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | close\_cmd | 关闭命令(0:关闭 1:打开 2：仅按钮灯闪烁 3：取消闪烁 4:集包失败) |
| Uint16 | close\_addr\_num | 需变化的下料口地址数量 |
| Uint16 | close\_addr | 需变化的下料口地址 |

（4）应答帧命令：0x9405

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_Close\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.5 清空下料口命令

（1）发送时间

在用户选择清空模式时有WCS发送给下料口系统。

（2）发送帧命令：0x1406

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 clear\_addr\_num;

uint16 clear\_addr[clear\_addr\_num];

}sWCS2IOCtrl\_Clear\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | clear\_addr\_num | 需清空的下料口地址数量 |
| Uint16 | clear\_addr | 需清空的下料口地址 |

（4）应答帧命令：0x9406

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_Clear\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.6 下料口包裹数量实时状态（20220707-燕聪修订）

（1）发送时间

下料口下包数量发生变化时，下料口主动发送对应下料格口的已下包数量, 可由下料口清空命令清除下料口下包数量信息为0。

（2）发送帧命令：0x1407

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 Date\_type;

uint16 package\_addr\_num;

uint16 package\_addr;

uint16 package\_num;

}sIOCtrl2WCSl\_Package\_Num\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](E:/1-工作/4-内部接口协议/2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | Date\_type | 0表示下层，1表示上层，单层线为0，2表示中间层即第二层 |
| Uint16 | package\_addr\_num | 下包数量发生变化的下料格口总数 |
| Uint16 | package\_addr | 下包数量发生变化的一条下料口地址信息 |
| Uint16 | package\_num | 当前下料地址的已下包数量 |
| ...... | ...... | 根据package\_addr\_num，每组数据按package\_addr和package\_num依次存放在数据包中 |

4）应答帧命令：0x9407

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_Package\_Num\_\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](E:/1-工作/4-内部接口协议/2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | package\_errnum\_ack | 下料口实际下包数量与wcs 的计数不符的格口总数 |
| Uint16 | Package\_addr\_ack | 下料有问题的下料口地址 |
| Uint16 | Package\_num\_ack | 下料口实际下包数量与wcs包裹计数的差值 |
|  |  | 根据package\_errnum\_ack，每组数据依次按Package\_addr\_ack和Package\_num\_ack  依次存放在数据包中 |

## 5.7 一键开关命令（20200620-燕聪修订）

（1）发送时间

在用户选择一键开关下料格口时由WCS发送给下料口系统。

（2）发送帧命令：0x1408

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 io\_status;

uint16 special\_ctrlIo\_num;

uint16 special\_ctrlIo\_status;

uint16 io\_sddr[special\_ctrlIo\_num];

}sWCS2IOCtrl\_closeall\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | io\_status; | 0为一键打开下料口，1为一键关闭下料口 |
| Uint16 | special\_ctrlIo\_num | 在一键开关下料口状态下，需要区别控制的下料格口的数量 |
| uint16 | special\_ctrlIo\_status | 需要被区别控制的下料口的状态，0打开，1关闭 |
| uint16 | io\_sddr[special\_ctrlIo\_num] | 需要被区别控制的下料口的地址，依次放在数据包中 |

（4）应答帧命令：0x9408

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_closeall\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.8 下料口操作无效命令（20200624-燕聪修订）

（1）发送时间

在选择是否可以由现场手动操作下料口时由WCS发送给下料口系统。

（2）发送帧命令：0x1409

（3）数据格式

typedefstruct

{

uint8 cmd[11];

Uint16 manual\_mode;

}sWCS2IOCtrl\_manual\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | manual\_mode | 0手动操作下料口有效，1手动操作下料口无效 |

（4）应答帧命令：0x9409

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_manual\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

5.9 下料口大小格口合并信息

（1）发送时间

在下料口大小格口合并或拆分时由WCS发送给下料口系统。

（2）发送帧命令：0x140A

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 io\_num;

Port\_info[io\_num];

}sWCS2IOCtrl\_IO\_\_Data;

结构体数据如下：

Typedef struct

{

u16 index; // 序号（区分第几组信息）

u16 addr1; // 一个大格口其中一个小格口的地址

u16 addr2; // 另一个小格口的地址

}Port\_info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | Io\_num | 大格口的数量 |
| Port\_info | Port\_info[io\_num] | 格口信息 |

（4）应答帧命令：0x940A

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_manual\_IO\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.10下料口系统版本信息

（1）发送时间

下料口系统每隔一段时间主动发送版本信息。

（2）发送帧命令：0x140B

（3）数据格式

typedef struct

{

uint8 cmd[11];

Uint8 version[4];版本号小端格式

}sIOCtrl2WCS\_CMD\_version \_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | version[4] | 版本号小端格式（比如实际版本号 2.0.0.8 则对应 0x08,0x00,0x00,0x02） |

（4）应答帧命令：无需应答，WCS打印解析即可

## 5.11 WCS发送摆臂位置命令（0x140C）

（1）发送时间

WCS认为需要调整摆臂位置时，由WCS发送给“摆臂系统”，每次发送该指令最多发送400个摆臂信息，多于400个摆臂信息则分包发送。

（2）发送帧命令：0x140C

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 num;

baibi\_info[num];

}sWCS2IOCtrl\_IO\_\_Data;

结构体数据如下：

Typedef struct

{

uint16 io\_baibi; // 需要调整摆臂位置的摆臂的编号

uint8 direction; // 摆臂方向，0是中间，1是左边，2是右边

}baibi\_info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | Io\_num | 需要调整摆臂的滑槽数量 |
| baibi\_info | baibi\_info[num] | 摆臂信息 |

（4）应答帧命令：0x940C

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sIO\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.12 摆臂实时位置（0x140D）

（1）发送时间

“摆臂系统”实时上传所有摆臂的当前位置，每包最多发400个摆臂的信息，多于400个则分包发送，间隔暂定50ms。

（2）发送帧命令：0x140D

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 num;

baibi\_info[num];

}sWCS2IOCtrl\_IO\_\_Data;

结构体数据如下：

Typedef struct

{

uint16 io\_baibi; // 摆臂的IO编号

uint8 direction; // 摆臂方向，0是中间，1是左边，2是右边

}baibi\_info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | Io\_num | 摆臂数量 |
| baibi\_info | baibi\_info[num] | 摆臂信息 |

（4）应答帧命令：0x940D

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sIO\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.13 摆臂实时状态（0x140E）

（1）发送时间

“摆臂系统”检测出某个摆臂状态异常时发送给WCS,每次最多上传400个摆臂的故障，多于400个则分包发送。

（2）发送帧命令：0x140E

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint16 num;

baibi\_info[num];

}sWCS2IOCtrl\_IO\_\_Data;

结构体数据如下：

Typedef struct

{

uint16 io\_baibi; // 摆臂故障了的IO编号

uint16 baibi\_status; // 补充故障信息

}baibi\_info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint16 | Io\_num | 故障摆臂的数量 |
| baibi\_info | baibi\_info[io\_num] | 摆臂的故障信息 |

（4）应答帧命令：0x940E

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sIO\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.14 格口特殊操作方式命令

（1）发送时间

当指定某些格口需要特殊操作时，WCS发送给下料口系统，收到下料口的反馈，视为成功，否则重发。

（2）发送帧命令：0x1410

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint8 status\_cmd;//需要切换的状态

uint16 status\_addr\_num;//需要切换格口控制状态的格口数量

uint16 status\_addr[status\_addr\_num];//需要切换控制状态的格口IO编号

}sWCS2IOCtrl\_Status\_CMD\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | close\_cmd | 状态命令(0:普通模式，备用 1:长按两秒才能关闭) |
| Uint16 | close\_addr\_num | 需切换的下料口地址数量 |
| Uint16 | close\_addr | 需切换的下料口IO编号 |

（4）应答帧命令：0x9410

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_Status\_CMD\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.15 摆臂类型配置信息

（1）发送时间

配置摆臂的驱动类型，以及该类型下所有的摆臂编码。目前有俩种摆臂类型，伺服摆臂、电滚筒摆臂，每次最多配置400个摆臂，多于400个分包发送。

（2）发送帧命令：0x1411

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint8 swing\_type; //摆臂类型 0 正弦驱动器 /1实验室生产的控 制器

uint16 num; //摆臂编码数量

uint16 swing\_index[num];//具体的摆臂编码

}sWCS2IOCtrl\_TYPE\_CONFIG\_Data;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| Uint8 | swing\_type | 摆臂类型 |
| Uint16 | num | //摆臂编码数量 |
| Uint16 | swing\_index[num] | //摆臂的编码 |

（4）应答帧命令：0x9411

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_TYPE\_CONFIG\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.16 摆臂基础参数配置

（1）发送时间

配臂基础参数配置，每次板卡上电，发送一次，如果不配置，则按下料口控制器默认参数执行(目前只有实验室生产的电滚筒需要配置)。

（2）发送帧命令：0x1412

（3）数据格式

typedefstruct

{

uint8 cmd[11];

uint 8 port\_inside; // 0 内侧格口 1外侧格口

uint16 swing\_spd; //摆臂转动速度 滚筒转速 （1-700）

uint16 left\_angle; //左边摆动的角度；

uint16 right\_angle; //右边摆动的角度；

uint16 zero\_offset; //中间位置相对于原点的偏移角度

}sWCS2IOCtrl\_BASIC\_CONFIG\_Data;

（4）应答帧命令：0x9412

（5）应答数据区

具体格式如下：

typedef struct

{

uint8 cmd[11];

} sWCS2IOCtrl\_BASIC\_CONFIG\_Data\_ACK;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| Uint8 | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |

## 5.17 下料口从板卡can通讯报警（0x1413）

（1）发送时间

每5s发送一次报警信息，没有报警则不发送。

（2）发送帧命令：0x1413

（3）数据格式定义

typedef struct

{

unsigned char cmd[11];

unsigned char num;

unsigned char index[num];

} sIO\_CANERR\_Port\_Info;

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | cmd[11] | [详见1.2 通讯协议格式](file:///E:\\1-工作\\4-内部接口协议\\2_交叉带分拣系统通讯协议V2.0.04_T20220530.doc" \l "_1.2通讯协议格式) |
| unsigned char | num | 从板卡can通讯报警的数量 |
| Unsigned char | index[num] | 具体的can通讯异常的板卡号 |

（4）应答帧命令：无需应答

# 6 PLC控制系统与WCS通讯命令详解

## 6.1 IOB信息、急停按钮、报警灯状态、电机状态、防碰撞、48伏故障

（1）发送时间

PLC每过一辆小车发送一次所有以上信息到WCS。

（2）发送帧命令：0x1501

（3）数据格式

发送IOB信息、急停按钮状态信息、报警灯状态信息以及电机状态信息。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| cmd[11] | IOB信息区 | 急停按钮状态信息区 | 报警灯状态信息区 | 电机状态信息区 | 防碰撞信息区 | 48V开关信息区 |

cmd[11]：详见1.2 通讯协议格式

**IOB信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned int16 | car\_first\_no | 车头号 |
| unsigned char | iob\_num | IOB检测位个数 |
| sIob\_Status\_Info | iob\_status\_info[iob\_num] | IOB的信息， |

IOB信息数据结构体格式如下：

typedef struct

{

unsigned char iob\_no; //IOB编号

unsigned char iob\_status; //0-正常，1-超出小车边界

unsigned char iob\_direc; //内侧0 、外侧1，与下料口的方向一致

} sIob\_Status\_Info;

**急停按钮信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | emerg\_num | 急停开关个数 |
| sEmert\_Status\_Info | emerg\_status\_info[emerg\_num] | 急停开关的信息， |

急停按钮信息数据结构体格式如下：

typedef struct

{

unsigned char emerg\_no; //急停按钮编号

unsigned char emerg\_status; //0-正常，1-按下

} sEmert\_Status\_Info;

**报警灯信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | warn\_num | 急停开关个数 |
| sWarn\_Status\_Info | Warn\_status\_info[warn\_num] | 急停开关的信息， |

急停按钮信息数据结构体格式如下：

typedef struct

{

unsigned char warn\_no; //报警灯编号

unsigned char warn\_status; //0-正常，1-报警

} sWarn\_Status\_Info;

**电机信息**区数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | motor\_num | 电机个数 |
| sMotor\_Status\_Info | motor\_status\_info[motor\_num] | 电机的信息， |

急停按钮信息数据结构体格式如下：

typedef struct

{

unsigned char motor\_no; //电机编号

unsigned char motor\_status; //0-正常，1-故障

} sMotor\_Status\_Info;

**防碰撞**设备状态数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | coll\_num | 防碰撞设备个数 |
| sColl\_Status\_Info | coll\_status\_info[coll\_num] | 防碰撞设备的状态信息， |

急停按钮信息数据结构体格式如下：

typedef struct

{

unsigned char coll\_no; //电机编号

unsigned char coll\_status; //0-正常，1-故障

} sColl\_Status\_Info;

**48V开关**状态数据格式如下：

|  |  |  |
| --- | --- | --- |
| 变量类型 | 变量名 | 含义 |
| unsigned char | power\_num | 48V开关个数 |
| sPower\_Status\_Info | power\_status\_info[power\_num] | 48V开关的状态信息， |

急停按钮信息数据结构体格式如下：

typedef struct

{

unsigned char power\_no; //电机编号

unsigned char power\_status; //0-正常，1-故障

} sPower\_Status\_Info;

（4）应答帧命令：0x9501

（5）应答数据区

具体格式如下

typedef struct

{

uint8 char cmd[11];

} sWcs2Plc\_Info\_Data\_ACK;