打印单元管理系统说明

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 更新日期 | 更新内容 | 操作者 |
| V1 | 2016/8/12 | 创建本文档 | 黄冠恒 |
|  |  |  |  |

目录

[打印单元管理系统 1](#_Toc458806426)

[1.1 打印单元结构的定义 1](#_Toc458806427)

[1.1.1 概要 1](#_Toc458806428)

[1.1.2 代码文件位置 1](#_Toc458806429)

[1.2 打印单元管理表的定义 2](#_Toc458806430)

[1.2.1 概要 2](#_Toc458806431)

[1.2.2 代码文件位置 2](#_Toc458806432)

[1.3 打印单元管理表的初始化 2](#_Toc458806433)

[1.3.1 概要 2](#_Toc458806434)

[1.3.2 代码文件位置 2](#_Toc458806435)

[1.3.3 流程图 2](#_Toc458806436)

[1.4 待打印订单的获取 3](#_Toc458806437)

[1.4.1 概要 3](#_Toc458806438)

[1.4.2 代码文件位置 3](#_Toc458806439)

[1.4.3 流程图 4](#_Toc458806440)

[1.5 打印订单的分发 4](#_Toc458806441)

[1.5.1 概要 4](#_Toc458806442)

[1.5.2 代码文件位置 4](#_Toc458806443)

[1.5.3 流程图 4](#_Toc458806444)

[1.6 多数据类型订单的打印 5](#_Toc458806445)

[1.6.1 概要 5](#_Toc458806446)

[1.6.2 代码文件位置 6](#_Toc458806447)

[1.6.3 流程图 6](#_Toc458806448)

[1.7 打印单元的管理 7](#_Toc458806449)

[1.7.1 概要 7](#_Toc458806450)

[1.7.2 代码文件位置 7](#_Toc458806451)

[1.7.3 流程图 7](#_Toc458806452)

# 打印单元管理系统

## 打印单元结构的定义

### 概要

为维护打印单元，需要为打印单元建立一个结构，用于保存其编号、状态、属性等信息。其定义如下所示：

typedef u8\_t PrintCellNum; //打印单元编号类型

typedef s8\_t OrderEntry; //订单信息结构入口

typedef enum {

PRINT\_CELL\_STATUS\_IDLE, //空闲

PRINT\_CELL\_STATUS\_BUSY, //忙碌

PRINT\_CELL\_STATUS\_ERR //不可用

}PrintCellStatus; /\* 打印单元状态类型 \*/

typedef struct {

PrintCellNum no; /\* 单元编号 \*/

OrderEntry entryIndex; /\* 回指订单 \*/

PrintCellStatus status; /\* 指示打印单元的状态 \*/

u32\_t totalTime; // 打印总时长，单位为0.1s

u32\_t cutCnt; // 总切刀次数

u32\_t totalLength; // 总打印长度

u8\_t exceptCnt[EXCEPTION\_NUM+1]; // 异常与故障发生次数

u32\_t workedTime; // 连续打印时长，单位为0.1s

u32\_t beginTick; // 一次打印开始时的滴答

u32\_t endTick; // 一次打印结束时的滴答

}PrintCellInfo; /\* 打印单元数据结构 \*/

### 代码文件位置

代码位于[USER/print\_cells.h](../USER/netconf.c)。

## 打印单元管理表的定义

### 概要

为管理多个打印单元，需要建立一个管理结构，用于分配和回收打印单元。其定义见下：

typedef struct {

OS\_EVENT \*resrcSem; //标记可用的打印单元资源数

PrintCellInfo cells[MAX\_CELL\_NUM]; //可使用的打印单元表

}PrintCellsMgrInfo; /\* 打印单元管理结构 \*/

### 代码文件位置

代码位于[USER/print\_cells.h](../USER/netconf.c)。

## 打印单元管理表的初始化

### 概要

系统上电后，会初始化管理表。进行的工作有：分配用于标记可用的打印单元数的信号量资源（初始值为0）、将各个打印单元的成员设置为无效、将打印单元的状态设置为错误，然后从ROM中读取各个打印单元的长期属性，如工作总时长、切刀次数以及异常故障次数等。

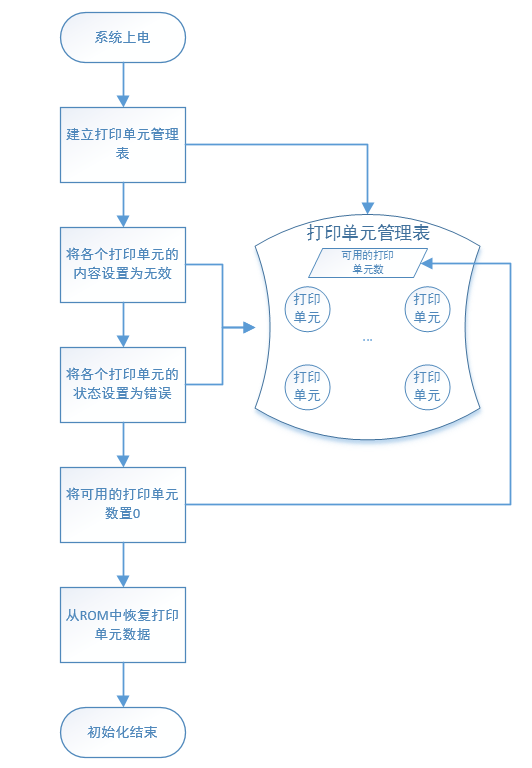
实现接口为：

* void InitPrintCellsMgr(void)
* void ReadPrintCellsInfo (void)

### 代码文件位置

代码位于[USER/print\_cells.c](../USER/netconf.c)。

### 流程图



流程图 1

## 待打印订单的获取

### 概要

打印单元的职责是打印订单，而订单的来源有两个：异常订单表和打印队列。其中，异常订单表中的订单由故障的打印单元放置，而打印队列中的订单由打印队列管理线程进行放置。打印单元只取第一份异常订单或者位于打印队列头的订单。

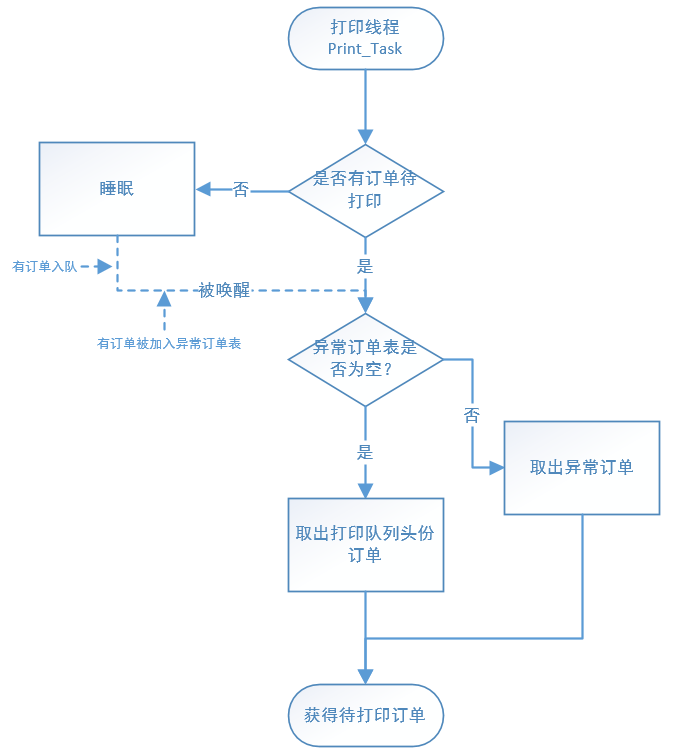
实现接口为：

* int RestoreOrder(u8\_t \*entryp)
* s8\_t GetOrderFromQueue(u8\_t \*entryp)

### 代码文件位置

代码位于USER/print\_cells.c以及USER/print\_queue.c。

### 流程图



流程图 2

## 打印订单的分发

### 概要

由于存在打印单元的管理结构，因此按照约定，我们应该通过打印单元管理表来请求获取打印单元，并将一份待打印订单交付给该打印单元。

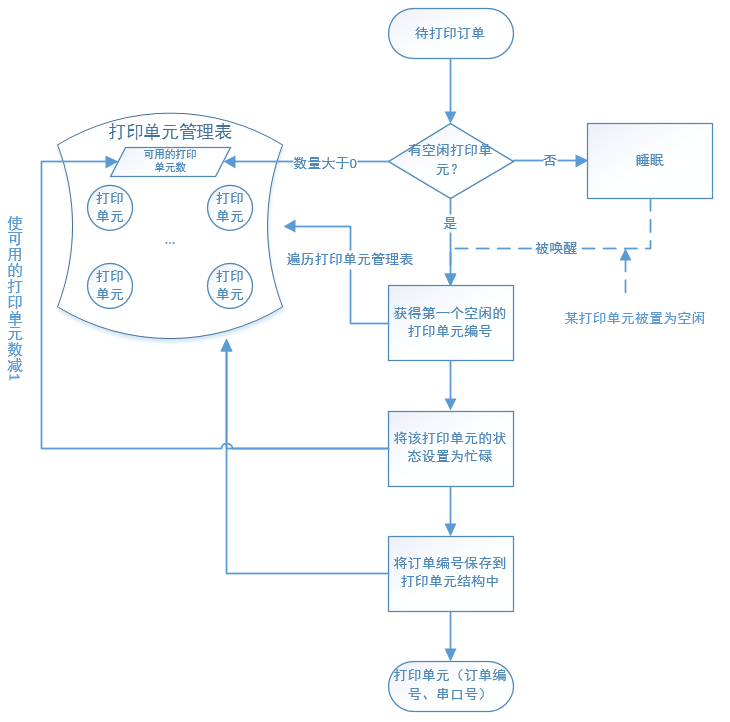
实现接口为：

* void DispensePrintJob(u8\_t entryIndex)

### 代码文件位置

代码位于USER/print\_cells.c。

### 流程图



流程图 3

## 多数据类型订单的打印

### 概要

由于一份订单可能同时含有文字数据、图像数据以及二维码链接，而这些数据对应的打印机指令又不相同。因此，根据与底层、服务器的约定，需要根据订单数据的类型进行不同打印接口的调用。由于数据存在错误的可能性，因此也对订单的数据进行了相应的处理。错误的数据将导致订单被丢弃，并反馈一个订单数据错误的状态。

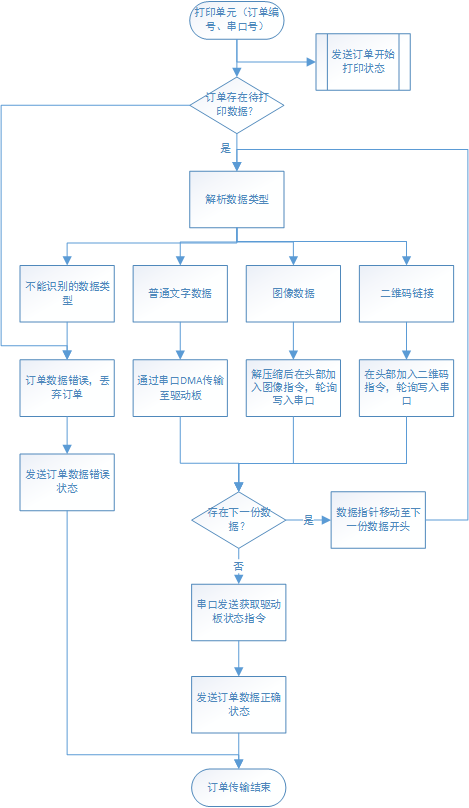
实现接口为：

* s8\_t Print\_Order(u8\_t entry\_index, u8\_t cellno)
* s8\_t Delete\_Order(s8\_t entry\_index)

### 代码文件位置

代码位于USER/print\_queue.c。

### 流程图



流程图 4

* 1. 打印单元的管理
     1. 概要

打印订单的成功与否依赖于打印结束后打印单元的状态，因此，每一次订单传输结束后都会发送获取打印单元状态的指令。根据接收到的状态进行相应的判断。

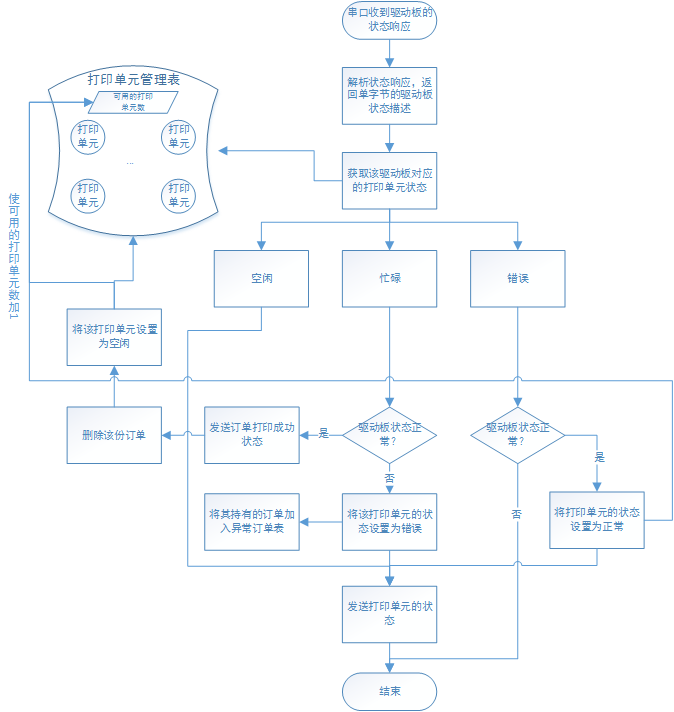
打印单元的状态决定了打印单元是否可供调度，也决定了设备的健康度。因此，根据打印单元上一次的状态和当次获取到的状态，可得出打印单元工作的情况。

实现接口为：

* static PrintCellInfo \*GetIdlePrintCell(void)
* static void PutPrintCell(PrintCellNum no, PrintCellStatus status)
* static void DealwithOrder(PrintCellNum cellno)
* static void DealwithDMA(PrintCellNum cellno)
  + 1. 代码文件位置

代码位于USER/print\_cells.c。

* + 1. 流程图



流程图 5