KPM204 系列用户手册

版本号：V2.1

河南康派

版本修订

版本号	修订日期	描述	审核
V1.0	20171107	创建文档	
V2.0	20180315	修改文档	
V2.1	20180428	修改文档	

特别说明

本公司保留在未通知用户的情况下，对产品、文档、服务等内容进行修改、更正等其他一切变更权利。

目录

一、产品概述

KPM204 是一款基于 RISC 架构五级流水线的芯片作为主处理器的嵌入式计算机。该 CPU 是以 ARM926EJS 为核心的系统级单芯片，内置 64MB DDR2，提供加解密软件，提供 AES、DES/3DES 加解密，内置看门狗功能，最高支持 300MHz 的频率。系统提供有线网络通讯，同时也提供无线 GPRS 通讯，具有体积小、功耗低、效率高等特点，适用于电力集中器、HMI、工业控制、网关等场合。

二、产品特性

1. 硬件特性

- NUC970 CPU：
  - 32bit ARM926EJ-S，主频 300MHz，1.1MIPS/MHz，最高支持 300MHz
  - 16KB I-cache，16KB D-cache
  - 支持 MMU，支持 JTAG Debug
- 内存：
  - 内置 64Mbyte DDR2，56Kbyte SRAM
  - 高达 150MHz 的 SDRAM 时钟
- FLASH：
  - 128Mbyte NANDFlash，最大支持 512Mbyte（可定制 128M/256M/512M）
  - 支持 SLC、MLC 等类型的 NAND FLASH
- 加密：
  - 支持 PRNG/DES/3DES/AES/SHA/HMAC 加密，最高 256 位加密模式（预留功能）
- 看门狗：
  - 内置 WDT，溢出时间小于 14 秒，支持空闲唤醒和掉电唤醒
- RTC:
  - 实时时钟，内置供电电池
- RS232：
  - 2 路 RS232 通讯端口，与 RS485 复用，内置 ESD 保护，全隔离保护设计
- RS485：
  - 2 路独立 RS485 通讯端口，内置 ESD 保护，全隔离保护设计

- 2 路复用 RS485 通讯端口，内置 ESD 保护，全隔离保护设计

- ADC：
  - 4 路模拟量输入，支持电压 0~5VDC、电流的输入 4~20mA
- 开关量：
  - 4 路开关量输出，最大驱动电流 200mA，内置隔离保护
  - 4 路开关量输入，干节点输入，内置隔离保护
- SD 卡、USB 接口：
  - 内置 SD 卡接口
  - 外置 USB 接口，支持 USB 存储设备
- 调试口：
  - 外置 RS232 调试口，波特率：115200，数据位：8，停止位：1，校验位：none，流控：无
- 网络：
  - 2 路 10M/100M 自适应工业以太网，标准 RJ45 接口
  - 内置 ESD 保护（8KV）、浪涌保护（4KV）
- 无线功能：
  - 射频波段 800/900/1800/1900MHz（可选 2/3/4G）
  - 可选 WIFI：可连接 AP，也可做 AP
  - 1 个 SIM 卡接口，1 个天线接口
  - 传输速度：达到相应功能的标准速度
- 电源：
  - 输入电压：9~35VDC，推荐使用 12VDC/3A 或 24VDC/1.5A
  - 单机功耗：< 3W
- 机械特性
  - 外壳金属材质
  - 尺寸：135mm * 110mm * 42mm
  - 防护等级：IP63
- 工作环境
  - 工作温度：-40℃~+85℃
  - 工作湿度：5% ~ 95%

2. 软件特性

2.1 系统特性

KPM204 预装基于 nuvoton NUV970 的 Linux 操作系统，版本为 3.10.10x。满足 POSIX 标准或类 UNIX 平台的应用程序。针对系统特有的硬件设备，内核提供了简单、易用的驱动接口，可加速用户的应用程序开发。

KPM204 系统的软件系统共分为 3 部分，分别为 Bootloader、linux 内核和 rootfs。Bootloader 是遵循 GPL 条款的开放源码项目，UBoot 主要是引导内核的启动，支持 NFS 挂载、NAND Flash 启动；linux 内核是整个操作系统的最底层，负责整个硬件的驱动，以及提供各种系统所需的核心功能；rootfs 是用于明确磁盘或分区上的文件的方法和数据结构，即在磁盘上组织文件的方法。

2.2 环境配置

编译环境：本公司提供的虚拟机系统 ubuntu，可直接编译使用

编译命令：arm-linux-gcc -o filename filename.c

编译链：本公司提供的 arm_linux_4.8-20150831.tar.gz

非本公司提供的编译环境下，把编译链拷贝到 PC 的 LINUX 系统下，解压编译链后，把根目录下的 bin 目录添加到系统的环境变量即可。

添加环境变量：export　PATH=/xxx/bin:$PATH

编译链下载地址：

https://pan.baidu.com/s/1nv19D1Z#list/path=%2FWORK%2F%E6%96%B0%E5%A1%98ARM9&parentPath=%2F

本公司提供的编译环境：

操作系统：ubuntu 10.04

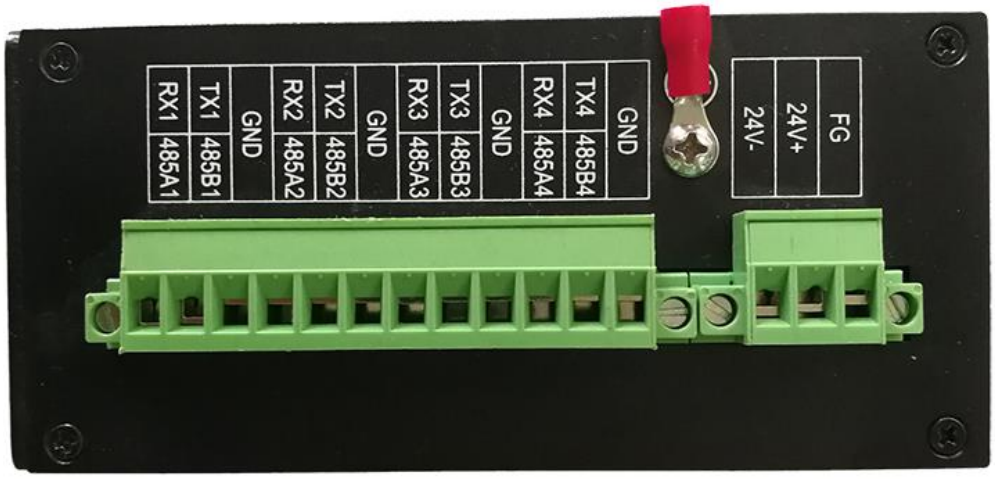
用户名：work　　密码：123456

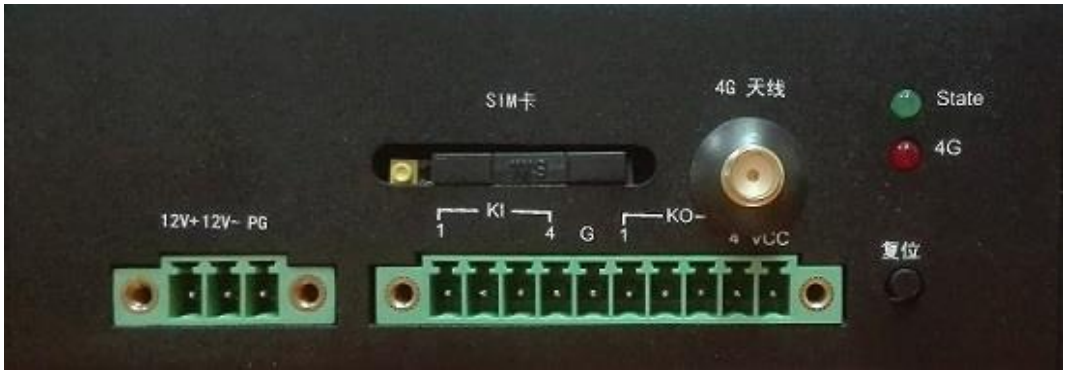
管理机用户名：root　　密码：root

管理机 IP：eth0：192.168.1.177

　　eth1：192.168.2.177

三、接口定义





1. 电源接口（VIN）

编号	标识符	功能说明
1	12V+	12V/24V 电源正极
2	12V—	输入电源地或负极
3	PG	保护地

2. SIM 卡接口（SIM CARD）

编号	标识符	功能说明

1	SIM 卡	SIM 卡接口，支持移动、联通卡、电信卡

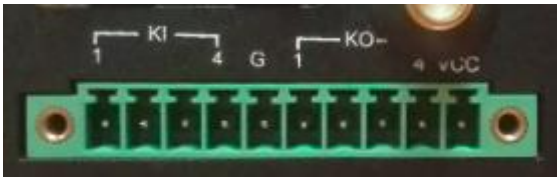
3. 天线接口（ANT）

编号	标识符	功能说明
1	4G 天线	2G、3G、4G 的 SMA 天线接口

4. 4G 指示灯

编号	标识符	功能说明
1	State	4G 网络状态指示灯，工作时处于闪动状态
2	4G	4G 工作电源指示灯

5. 开关量输入（KI）



编号	标识符	功能说明
1	KIn	第 n 路开关量输入(n=1~4)
2	G	开关量输入 GND，即公共端

6. 开关量输出（KO）

编号	标识符	功能说明
1	KOn	第 n 路开关量输出(n=1~4)
2	VCC	开关量电源输出，+5VDC

7. 复位

编号	标识符	功能说明
1	复位	预留按键，用户可自定义使用

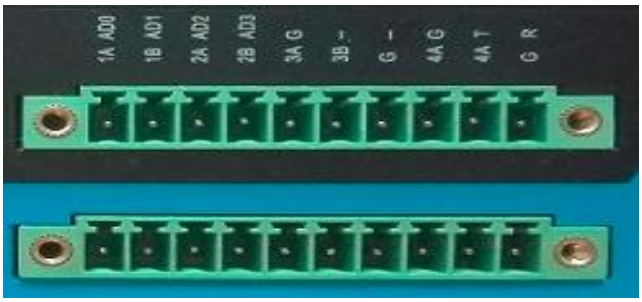
8. 网络接口（ETH）

编号	标识符	功能说明
1	LAN1	以太网接口 1
2	LAN2	以太网接口 2

9. USB 接口（USB）

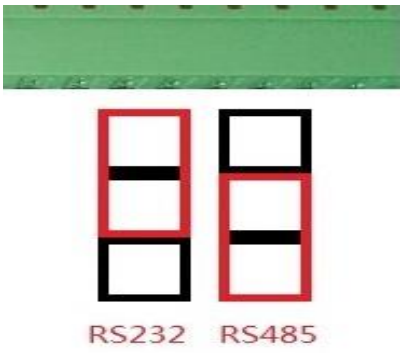
编号	标识符	功能说明
1	USB	USB 接口

10. RS485 接口



编号	标识符	功能说明
1	nA	第 n 通道 RS485 通讯 A 端口(n=1~2)
2	nB	第 n 通道 RS485 通讯 B 端口(n=1~2)
3	G	通讯系统地
4	nA	第 n 通道 RS485/RS232 通讯 A/TX 端口(n=3~4)
5	nB	第 n 通道 RS485/RS232 通讯 B/RX 端口(n=3~4)

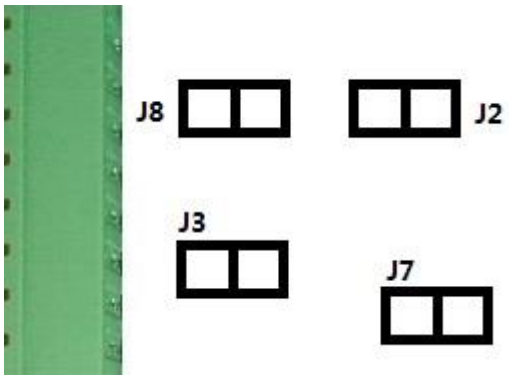
注：第 3 通道和第 4 通道为 RS485 与 RS232 可选通道，通过内部跳选择。红色表示选中。跳线选择功能如下图：



11. ADC 接口（ADC）

编号	标识符	功能说明
1	ADn	第 n 通道 ADC 输入(n=0~3)
2	G	模拟信号 GND

注：电压输入或者电流输入通过内部跳线选择，如下图



说明：

编号	标识符	功能说明
1	J2	ADC0 功能选择，悬空为电压输入，短接为电流输入
2	J8	ADC1 功能选择，悬空为电压输入，短接为电流输入
3	J3	ADC2 功能选择，悬空为电压输入，短接为电流输入
4	J7	ADC3 功能选择，悬空为电压输入，短接为电流输入

注：电压输入或者电流输入通过内部跳线选择，如下图

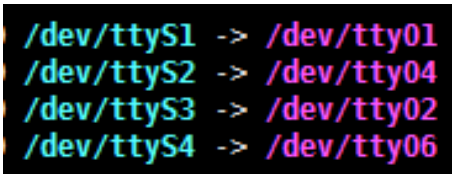
12. 调试口

编号	标识符	功能说明
1	T	调试打印口 RS232-TX
2	R	调试打印口 RS232-RX
3	G	调试打印口 RS232-GND

13. 指示灯

编号	标识符	功能说明
1	POW	系统电源指示灯
2	RX1/TX1	系统运行指示灯
3	RXn/TXn	第 n 通道 RS485/RS232 通讯指示灯

14. 驱动接口映射



编号	标识符	功能说明
1	ttyS1	第一通道 RS485 通讯
2	ttyS2	第二通道 RS485 通讯
3	ttyS3	第三通道 RS485/RS232 通讯
4	ttyS4	第四通道 RS485/RS232 通讯

四、驱动实例

在系统的/program 目录下有相应的脚本文件，可以进行一些简单的测试。其中要确保 startup.sh 文件里，端口映射的正确的。文件内容见附录。

1. RS485/RS232 接口驱动的使用

编号	通讯接口	设备文件
1	RS485	/dev/ttyS1
2	RS485	/dev/ttyS2
3	RS485/RS232	/dev/ttyS3
4	RS485/RS232	/dev/ttyS4

1.1 RS485/RS232 发送、接收数据

附录例程 2

2. GPRS 操作

GPRS 拨号操作的相应文件，具体操作参考 GPRS 使用说明。

4G：对应的设备文件为 ttyUSB3。

2.1 GPRS 的 AT 指令使用

使用例子：4G：at_u　2　AT

3. 开关量输入操作

见附录 3

4. 开关量输出操作

见附录 4

5. ADC 操作

见附录 5

附录：

1、startup.sh 文件内容：

```
#!/bin/sh
ln -sf /dev/ttyO1 /dev/ttyS1
ln -sf /dev/ttyO4 /dev/ttyS2
ln -sf /dev/ttyO2 /dev/ttyS3
ln -sf /dev/ttyO6 /dev/ttyS4
```

2、串口测试例程

```
/******************串口测试程序******************/
#include <stdio.h>
#include <string.h>
#include <malloc.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <termios.h>
#define max_buffer_size     100      /*bufffer size*/
/*****************************************/
int fd;
int flag_close;
int open_serial(int k,int *fd)
{
      int sfd = -1;
      char str[100];
      sprintf(str,"/dev/ttyS%d",k);
      printf("open %s\n",str);
      sfd = open(str,O_RDWR|O_NOCTTY| O_NONBLOCK);
       if(sfd == -1)
```

```
    {
        perror(str);
        return -1;
    }
    else
    {
        *fd = sfd;
        return 0;
    }
}

/*****************************************************************************/
int main(int argc, char *argv[ ] )
{
    time_t tNow,tOld;
    int port;
    char sbuf[]={"12345678901234567890123456789012345678901234567890\n"};
    char sbufrec[256]={0};
    int sfd,retv,i,ncount=0,mcount = 0;
    struct termios opt;
    int length=sizeof(sbuf);
    /*************************************************************/
    if(argc<2)
    {
        printf("input erro :serial <no> \n");
        return 0;
    }
    //open first serial
    port = atoi(argv[1]);
    open_serial(port,&fd);
```

```
/*****************************************************************/
printf("ready for sending data...\n");
tcgetattr(fd,&opt);
cfmakeraw(&opt);
/*****************************************************************/
cfsetospeed(&opt,B9600); /*set speed 9600bps*/
cfsetispeed(&opt,B9600);
/*****************************************************************/
tcsetattr(fd,TCSANOW,&opt);

while(mcount < 5)
{
    retv=write(fd,sbuf,length);    /*send data*/
    if(retv==-1)
    {
        printf("write error .....\n");
    }
    else
    {
        printf("the number of char sent is %d\n",retv);
    }

    ncount=0;
    printf("ready for receiving data...\n");

    time(&tOld);
    tNow=tOld;
    ncount = 0;
    while((tNow-tOld) < 2)          /*set timeout*/
    {
```

```
            time(&tNow);
            retv=read(fd,&sbufrec[0],1); /*read data*/
            if(retv==-1)
            {
                //perror("read");
            }
            else
            {
                printf("%02x ",sbufrec[0]);
                ncount+=1;
            }
        }
        mcount+=1;
        printf("\n");

    }
    flag_close = close(fd);
    if(flag_close == -1)      /*close */
        printf("Close the Device1 failur!\n");
    return 0;
}
```

3、开关量输入测试例程

kin.sh 脚本内容：

```
#!/bin/sh
start_num=0
while true;
do
    cat /sys/class/gpio/gpio205/value
    cat /sys/class/gpio/gpio206/value
```

```
    cat /sys/class/gpio/gpio194/value
    cat /sys/class/gpio/gpio195/value
    cat /sys/class/gpio/gpio37/value
    echo "the next test"
    sleep 2;
done;
```

4、开关量输出测试例程

kout.sh 文件内容：

```
#!/bin/sh
start_num=0
while true;
do
    let "start_num+=1"
    if [ "$start_num" = "2" ];
    then
    sync;
   echo 1 > /sys/class/leds/KOUT1/brightness;
    echo 1 > /sys/class/leds/KOUT2/brightness;
    echo 1 > /sys/class/leds/KOUT3/brightness;
    echo 1 > /sys/class/leds/KOUT4/brightness;
    start_num=0
    else
    echo 0 > /sys/class/leds/KOUT1/brightness;
    echo 0 > /sys/class/leds/KOUT2/brightness;
    echo 0 > /sys/class/leds/KOUT3/brightness;
    echo 0 > /sys/class/leds/KOUT4/brightness;
   fi
    sleep 2;
done;
```

5、ADC 测试例程

adc.c 文件内容：

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/ioctl.h>

#define ADS_CHN0    _IO('g', 0x01)

int main(int argc, char *argv[ ] )
{
    int fd,ret;
    unsigned long val;
    if(argc<2)
    {
        printf("input erro :ads port(port = 0,1,...)\n");
        return 0;
    }
    int port = atoi(argv[1]);
    if ((fd=open("/dev/ads",O_RDWR)) < 0)
    {
        perror("open");
        printf("Error opening /dev/ads\n");
        return -1;
    }
    ret = ioctl(fd,(ADS_CHN0 + port),&val);
    if(ret < 0){
```

```
            printf("AD read port(%d) Error\n",port);
            return -1;
    }
    printf("val = %d\n",val);
    printf("vol = %8.4fV\n",((float)val * 2.5)/4096.*2.);
    printf("amp = %8.4fmA\n",((float)val * 2.5)/4096.*2. / 0.25);
    close(fd);
}
```