ACT-208DL 系列用户手册

版本号：V2.0

瑞强科技

版本修订

版本号	修订日期	描述	审核
V1.0	20171017	创建文档	
V2.0	20180301	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

ACT-208DL 是一款基于精简指令集（RISC）架构高性能的 32 位 MPU 的嵌入式计算机。该 CPU 是以 ARM Cortex-A8 为核心的系统级单芯片，内置 NEON 单指令流多数流（SIMD）协处理，带有错误校正码（ECC）的 256KB L2 缓存，最高支持 1GHz 的频率。系统提供 RS458/RS232 通讯，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

- AM355x CPU：
  - 32bit ARM Corte-A8 架构，主频 800MHz，1.6MIPS/MHz，最高主频 1GHz
  - 32KB I-cache，32KB D-cache，NeonSIMD 协处理器
- 内存：
  - 512Mbyte DDR3、64KB 专用 RAM
- FLASH：
  - 256Mbyte NANDFlash，最大支持 8Gbyte
  - 支持 NAND、NOR、SRAM 等 FLASH
- 加密：
  - 支持 PRNG/DES/3DES/AES/SHA/HMAC 加密，最高 256 位加密模式
- 看门狗：
  - 内置 WDT，溢出时间小于 60 秒，支持空闲唤醒和掉电唤醒
- RTC:
  - 高精度实时时钟，内置供电电池
- 调试口：
  - 1 路串口为系统 console 口，波特率：115200，数据位：8，停止位：1，校验位：none，流控：无
- RS485/RS232：
  - 8 路 RS485 通讯端口，内部全隔离保护设计

- 2 路 RS232 通讯端口，内部全隔离保护设计

- 网络：
  - 2 路 10M/100M/1000M 自适应工业以太网，标准 RJ45 接口
  - 15KV TVS 保护，内部全隔离保护设计
- 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，2 个天线接口
  - 传输速度：达到相应功能的标准速度
- SD CARD：
  - 内置一个 SD/MMC 卡接口
- 电源：
  - 输入电压：6~35VDC，推荐使用 24VDC/1.5A
  - 单机功耗：< 4W
- 机械特性
  - 外壳：铝型材
  - 尺寸（mm）：173*126*50
  - 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

ACT-208DL 预装基于 TI AM335x 的 Linux 操作系统，版本为 3.2.0。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。

ACT-208DL 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

本公司提供的虚拟机系统：

用户名：work　　密码　：123456

编译环境：本公司提供的虚拟机系统 ubuntu 10.04，可直接编译使用

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

编译链：本公司提供的 arm-linux-gnueabihf-4.7.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=$PATH:/xxx/bin

如解压到/opt/arm-linux-gnu 目录下，则添加环境变量为：

export　PATH=$PATH:/ opt/arm-linux-gnu /bin

2.3 管理机登录

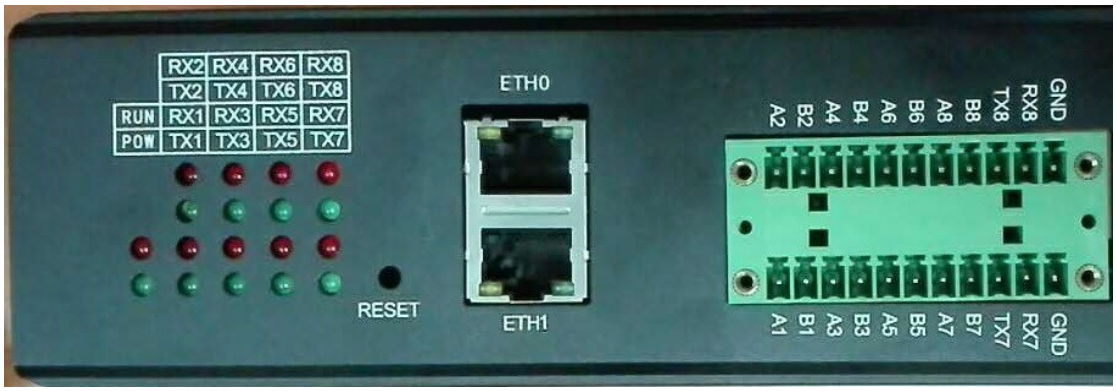
IP：　eth0：192.168.1.177

eth1：192.168.2.177

用户名：root

密码　：root

三、接口定义



1. 电源接口

编号	标识符	功能说明
1	24V-	系统电源地
2	0V	系统电源，输入电压范围 DC12~38V，推荐使用 DC24V
3	FG	屏蔽地、保护地，可不接

2. RS485 接口

编号	标识符	功能说明
1	An	第 n 通道 RS485 端口 A（n=1~8）
2	Bn	第 n 通道 RS485 端口 B（n=1~8）
3	TXn	第 n 通道 RS232 端口 TX（与 An 复用，n=7~8）
4	RXn	第 n 通道 RS232 端口 RX（与 Bn 复用，n=7~8）

3. SIM 卡接口

编号	标识符	功能说明
1	SIM 卡	2G/3G/4G 的 SIM 卡接口，支持移动、联通卡

4. 天线接口及指示灯

编号	标识符	功能说明
1	Wireless	WIFI 指示灯
2	State	GPRS 状态指示灯，指示网络状态
3	POW	GPRS 电源指示灯
4	WIFI	WIFI 天线接口
5	GSM 4G	GPRS 天线接口/4G 天线接口

5. 网络接口

编号	标识符	功能说明
1	ETH0	网络接口 0
2	ETH1	网络接口 1

6. 指示灯

编号	标识符	功能说明
1	RUN/POW	运行状态指示灯/电源指示灯
2	RXn/TXn	第 n 通道 RS485 通讯指示灯（n=1~8）

7. 复归

编号	标识符	功能说明
1	RESET	备用，用户可自行编程使用

8. SD 卡接口

编号	标识符	功能说明
1	SD Card	SD 存储卡接口（内置接口，如需使用，需打开设备）

9. 调试口

调试口（内置接口，如需使用，需打开设备），TTL 电平

编号	标识符	功能说明
1	VCC	3.3V 电源输出
2	R	TTL 电平 RDX 端
3	T	TTL 电平 TDX 端
4	GND	电源地

10. 驱动接口

编号	标识符	功能说明
1	ttySn	RS485 驱动接口（n=1~6）
2	ttySn	RS485/RS232 驱动接口（n=7~8）
3	ttyS9	GPRS 通讯接口，用于 AT 指令与数据通道

```
/dev/ttyS1 -> /dev/tty02
/dev/ttyS2 -> /dev/tty03
/dev/ttyS3 -> /dev/tty04
/dev/ttyS4 -> /dev/tty05
/dev/ttyS5 -> /dev/ttySU0
/dev/ttyS6 -> /dev/ttySU1
/dev/ttyS7 -> /dev/ttySU2
/dev/ttyS8 -> /dev/ttySU3
/dev/ttyS9 -> /dev/tty01
```

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。

脚本及说明：

4G-connect-chat：4G 拨号需要的脚本

at_u：AT 指令使用脚本，可用于查看信号、中心号码等信息

dial-on.sh：拨号脚本，在给 gprs 上电后，即可使用此脚本进行拨号

gprs-on.sh：GPRS 上电开机

gprs-off.sh：GPRS 关机断电

serial：串口测试脚本。

watchdog：看门狗测试

keytest：RESET 键测试

startup.sh：开机启动脚本，如果有需要开机启动的程序，可添加到此文件内，注意程序路径的问题。

1. RS485/RS232 接口驱动的使用

RS485 为固定功能，RS485/RS232 为两种接口，同一个驱动接口，使用时同时都会有数据输出，但同一时刻，只能接受一种接口的数据。

编号	通讯接口	设备文件
1	RS485-1	/dev/ttyS1
2	RS485-2	/dev/ttyS2
3	RS485-3	/dev/ttyS3
4	RS485-4	/dev/ttyS4
5	RS485-5	/dev/ttyS5
6	RS485-6	/dev/ttyS6
7	RS485-7/RS232-7	/dev/ttyS7
8	RS485-8/RS232-7	/dev/ttyS8

1.1 RS485/RS232 发送、接收数据

使用方法：

```
cd program
```

./serial n // n = 1 ,2 ,3

此时串口 n 会有数据输出，同时也能接收外面传的进来的数据

附录：serial.c 文件内容。

2. GPRS 操作

GPRS 操作的相应文件，请查看附件文件。

2G：对应的设备文件为/dev/ttyS9

3G/4G：对应的设备文件为/dev/ttyUSB3

2.1 GPRS 的 AT 指令使用

使用例子： 2G：at_c 9 AT

3G/4G：at_u 2 AT

2.2 GPRS 开机拨号

参考附件：GPRS 使用说明.txt

附录：

1. startup.sh 文件内容：

```
#!/bin/sh
ln -sf /dev/ttyO2 /dev/ttyS1
ln -sf /dev/ttyO3 /dev/ttyS2
ln -sf /dev/ttyO4 /dev/ttyS3
ln -sf /dev/ttyO5 /dev/ttyS4
ln -sf /dev/ttySU0 /dev/ttyS5
ln -sf /dev/ttySU1 /dev/ttyS6
ln -sf /dev/ttySU2 /dev/ttyS7
ln -sf /dev/ttySU3 /dev/ttyS8
ln -sf /dev/ttyO1 /dev/ttyS9
```

2. serial.c 文件内容：

```
#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     /* buffer size */
/*****************************************/
int fd1;
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"};/*
固定发送的数据 */
    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 <1~4> \n");
        return 0;
    }
    port = atoi(argv[1]);
    open_serial(port,&fd1);
    /*********************************************************************/
    printf("ready for sending data...\n");
    tcgetattr(fd1,&opt);
    cfmakeraw(&opt);
    /*********************************************************************/
    cfsetispeed(&opt,B9600); /*设置波特率为 9600bps*/
    cfsetospeed(&opt,B9600);
    /*********************************************************************/
```

```
tcsetattr(fd1,TCSANOW,&opt);

while(mcount < 5)
{
    retv=write(fd1,sbuf,length);   /* 发送数据 */
    if(retv==-1){
        //perror("write");
        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))          /* 设置接收超时 */
    {
        time(&tNow);
        retv=read(fd2,&sbufrec[0],1);
        if(retv==-1){
            //perror("read");
            //printf("error read \n");
            //printf("tOld=%d;tNow=%d\n",tOld,tNow);
        }
        else{
            printf("%02x ",sbufrec[0]);
            ncount+=1;
```

```
                }
            }
            mcount+=1;
            printf("\n");
        }
        flag_close = close(fd1);
        if(flag_close == -1)     /*关闭口端口*/
            printf("Close the Device1 failur!\n");
        return 0;
}
```