**Phytium Diag操作手册**

图片1

**恒为科技（上海）股份有限公司**

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **描述** | **作者** |
| 2020/11/30 | 0.1 | 初稿 | 郑东雄 |
| 2021/04/25 | 0.2 | 1. Lcd脚本命令描述修改 2. 新增风扇数值读取脚本 3. 新增fpga版本读取 4. 新增run\_full.sh使用方法 | 郑东雄 |
| 2021/04/26 | 0.3 | 1. 增加run\_test脚本使用 2. 增加i2cdetect参数描述 3. 增加memtester工具使用说明 4. 增加hdparm工具使用 | 郑东雄 |
| 2021/05/26 | 0.4 | 1. 增加eth\_board.sh脚本使用说明3.4.1 2. 增加网讯芯片iom卡脚本使用说明3.4.2.1 3. 增加工具canconfig工具使用3.10.1 4. 增加网口自动发包说明3.4.4.2 | 郑东雄 |
| 2021/07/ | 0.5 | 1. 增加VGA测试工具 | 郑东雄 |

目 录

[修订记录 I](#_Toc76737368)

[目 录 III](#_Toc76737369)

[1. 简介 1](#_Toc76737370)

[2. Diag使用 1](#_Toc76737371)

[3. Diag工具 1](#_Toc76737372)

[3.1 I2C 1](#_Toc76737373)

[3.1.1 i2cdetect 1](#_Toc76737374)

[3.1.2 i2cget 2](#_Toc76737375)

[3.1.3 i2cset 2](#_Toc76737376)

[3.2 PCIE 2](#_Toc76737377)

[3.2.1 lspci 2](#_Toc76737378)

[3.3 SATA 4](#_Toc76737379)

[3.3.1 fio 4](#_Toc76737380)

[3.3.2 hdparm 6](#_Toc76737381)

[3.3.3 disk.sh 6](#_Toc76737382)

[3.4 网口 7](#_Toc76737383)

[3.4.1 eth\_board.sh 7](#_Toc76737384)

[3.4.2 get\_eth.sh 8](#_Toc76737385)

[3.4.2.1 网讯IOM卡 8](#_Toc76737386)

[3.4.3 eth\_counter.sh 8](#_Toc76737387)

[3.4.4 tx\_packet.sh 9](#_Toc76737388)

[3.4.4.1 测试IOM卡 10](#_Toc76737389)

[3.4.4.2 网口自动发包 11](#_Toc76737390)

[3.4.5 show\_rxtx.sh 12](#_Toc76737391)

[3.4.6 show\_error.sh 12](#_Toc76737392)

[3.4.7 iperf 12](#_Toc76737393)

[3.4.8 tcpdump 14](#_Toc76737394)

[3.5 USB 14](#_Toc76737395)

[3.5.1 lsusb 14](#_Toc76737396)

[3.5.2 fio 15](#_Toc76737397)

[3.5.3 disk.sh 15](#_Toc76737398)

[3.6 LCD 15](#_Toc76737399)

[3.6.1 lcd.sh 15](#_Toc76737400)

[3.7 LPC 16](#_Toc76737401)

[3.8 QSPI 16](#_Toc76737402)

[3.9 UART 16](#_Toc76737403)

[3.9.1 minicom 16](#_Toc76737404)

[3.9.2 stty 18](#_Toc76737405)

[3.10 CAN 18](#_Toc76737406)

[3.10.1 canconfig 18](#_Toc76737407)

[3.10.2 candump 19](#_Toc76737408)

[3.10.3 candsend 19](#_Toc76737409)

[3.11 GPIO 20](#_Toc76737410)

[3.12 FRU 20](#_Toc76737411)

[3.12.1 fru\_init.sh 20](#_Toc76737412)

[3.12.2 fru.sh 20](#_Toc76737413)

[3.13 TMP422 21](#_Toc76737414)

[3.13.1 tmp422.sh 21](#_Toc76737415)

[3.14 RTC 21](#_Toc76737416)

[3.14.1 hwclock 21](#_Toc76737417)

[3.15 内存 22](#_Toc76737418)

[3.15.1 memtester 22](#_Toc76737419)

[3.15.2 devmem 23](#_Toc76737420)

[3.16 Fan 23](#_Toc76737421)

[3.16.1 fan.py 23](#_Toc76737422)

[3.17 FPGA 24](#_Toc76737423)

[3.17.1 fpga\_version.sh 24](#_Toc76737424)

[3.18 拷机 25](#_Toc76737425)

[3.18.1 run\_full.sh 25](#_Toc76737426)

[3.18.2 stop\_run\_full.sh 25](#_Toc76737427)

[3.18.3 run\_test.sh 25](#_Toc76737428)

[3.19 VGA 26](#_Toc76737429)

[3.19.1 x11perf 26](#_Toc76737430)

[3.19.2 glxgears 26](#_Toc76737431)

[4. Linux环境工具 26](#_Toc76737432)

[4.1 更改进程优先级 26](#_Toc76737433)

[4.1.1 renice 26](#_Toc76737434)

[4.2 查看cpu使用 27](#_Toc76737435)

[4.2.1 top 27](#_Toc76737436)

1. 简介

本文描述Phytium项目中Diag工具包使用

1. Diag使用

解压工具包,解压出目录为tools:

tar –xf diag\_tools\_VX.X.tar.gz

进入目录进行初始化,初始化目的使用I2C工具与tx\_port\_packet.sh脚本和fru工具,解压后执行一次即可。

cd tools/

source init.sh

1. Diag工具

## I2C

### i2cdetect

测试工具:i2cdetect

使用说明:显示总线上所有I2C设备

使用方法:

./i2cdetect [opt]

opt:-l 显示所有I2C总线

-y <bus> 查看指定总线号上设备

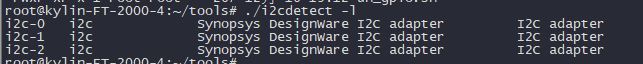
-r 使用SMBUS的"receive byte"扫描器件,SMBUS总线时使用

使用例子:

显示所有的总线:

./i2cdetect –l

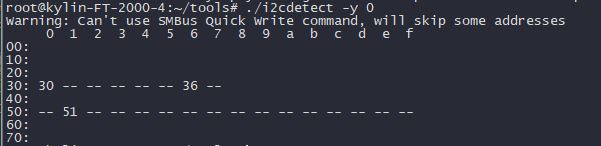
显示结果:



查看指定总线0上的所有设备地址:

./i2cdetect –y 0

显示结果:



总线号0上地址0x30 0x36 0x51有设备

### i2cget

测试工具:i2cget

使用说明:读取指定总线上设备寄存器的值

使用方法:

./i2cget –f –y <bus> <dev\_addr> <reg\_addr>

使用例子:

获取总线号为0 地址为0x51 设备中0x0寄存器的值

./i2cget –f –y 0 0x51 0x0

显示结果:

./i2cget -f -y 0 0x51 0x0

0x23

### i2cset

测试工具:i2cset

使用说明:设置指定总线上设备寄存器的值

使用方法:

./i2cset –f –y <bus> <dev\_addr> <reg\_addr> <date>

使用例子:

设置总线号0地址为0x30器件中0x10寄存器的值为0xFF

./i2cset –f –y 0 0x30 0x10 0xFF

## PCIE

### lspci

测试命令:lspci

使用说明:显示所有pci设备信息

使用方法:

lspci [opt]

opt:-vvv 显示设备详细信息,v越多显示越详细,最多三个,

-t 以树状形式显示

-n 以厂商号,设备号形式显示pci设备

-x 以十六进制显示PCI配置空间前64字节

选择特定设备选项:

-s <domain>:<bus>:<slot>.<func> 仅显示选中域号,总线号,插槽中的设备,func 区分相同设备不同位置

-d <vendor>:<device>:<class> 显示具有指定厂商号,设备号的设备

使用例子:

lspci

输出如下:

00:00.0 Host bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX Host bridge (rev 01)

02:00.0 USB controller: VMware USB1.1 UHCI Controller

00:00.0 表示该设备在总线号0上,设备号为0

02:00.0 表示该设备在总线号2上,设备号为0

使用例子:

lspci -t

输出如下:

-[0000:00]-+-00.0

+-01.0-[01]--+-00.0

| +-01.0

+-07.0

+-07.1

+-0f.0

+-15.0-[03]—-

[0000:00] 表示域号0,总线号0上的所有设备以树型打印

01.0 表示该设备在总线号0,设备号为1,该设备扩展出pci总线，且总线号为1

07.0 07.1 表示两个相同设备号,以.<func>,区分不同的设备

详细查看总线号0槽位1的设备,v越多显示越详细,最多三个:

lspci –s 00:01.0 –vvv

测试命令:setpci

使用说明:配置或查询PCI设备

使用方法:

setpci [opt] 设备 addr.[B|W|L]

opt:-v显示指令执行细节

-D 仅测试,不将配置写入寄存器

-d 选中指定厂商号,设备号

-s 选中域号,总线号,插槽中的设备

B|W|L:以8位/16位/32位读或写

使用用例:

从00:0f.0设备的3C地址中读取8位数据

setpci –s 00:0f.0 3C.B

将数据F8写入到00:0f.0设备的3C地址中

setpci –s 00:0f.0 3C.B=F8

## SATA

### fio

测试工具:fio

使用说明:运行fio,对系统硬盘进行读写测试,

使用方法:

[]中需要填入不同的值, 可以根据需要输入不同的值自由组合

./fio –filename=[disk] -direct=1 -iodepth 1 –thread –rw=[mode] -ioengine=psync -bs=4k –size=[size] -runtime=[time] -numjobs=[numjobs] -group\_reporting –name=[name]

[disk]:测试的硬盘名称(可用lsblk查看)

[mode]:randrw 随机混合读写

随机混合读写情况下默认读写各占50%,需要添加不同占比设置(选一即可),

-rwmixread=m 读占m%写占(100-m)%

-rwminwrite=n 写占n%读占(100-n)%

rw 顺序混合读写

randread 随机读

randwrite 随机写

write 顺序写

read 顺序读

[size]:根据需求设置测试硬盘的大小

[time]:本次运行的时间(测试设备1M 设置10h，测试完成就返回。设置测试设备20G 设置时间10s，10s后返回)测试时间可以不需要

[numjobs]:本次测试线程的数量

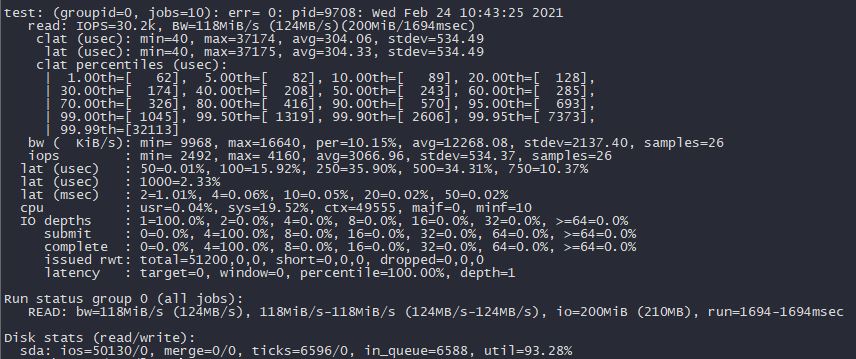
[name]:本次打印出报告的名称

使用用例:

使用10线程，测试硬盘/dev/sda，随机读20M空间的能力

./fio –filename=/dev/sda -direct=1 -iodepth 1 –thread –rw=randread -ioengine=psync -bs=4k –size=5G -numjobs=10 -group\_reporting –name=test

测试结果



报告中READ：

io：执行多少M的空间,10线程每次测试20M

bw:平均IO的带宽

报告中Disk stats

ios：所有group执行的IO数

merge：发生的IO合并数

io\_queue:花费在队列上的总共时间

util：磁盘利用率

### hdparm

测试工具:hdparm

使用说明:磁盘进行读测试

使用方法:

hdparm <opt> <disk>

opt : -t 评估磁盘读取效率

-T 评估磁盘快取的读取效率

使用方法:

hdparm –tT /dev/sda

使用用例:

对sda检测

./hdparm –Tt /dev/sda

### disk.sh

测试工具:disk.sh

使用说明:运行disk.sh,对系统硬盘进行读写测试,写入和读取一个随机文件测试其md5值,之后再到目录/tmp/log/disk.log 查看数据

使用方法:

./disk.sh <disk>

disk 对某个或多个硬盘检测

使用用例:

对sdb和sda1检测

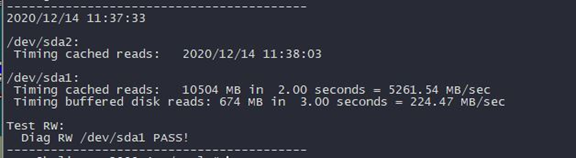
./disk.sh /dev/sdb /dev/sda1

对sda、sdb、sdc、sdd1检测

./disk.sh /dev/sda /dev/sdb /dev/sdb /dev/sdd1

查看结果:

cat /tmp/log/disk.log



## 网口

使用网口测试脚本前需要文件pktgen.ko,运行init.sh生成pktgen.ko,工具目录下有pktgen.ko则不需要运行

./init.sh

### eth\_board.sh

测试命令:eth\_board.sh

使用说明:只设置主板网口收发包,切换用户或重启设备要重新运行

使用方法:

source eth\_board.sh <board\_type>

type : 1 " FW1501 "

2 " FW1600 "

3 " FW1700 "

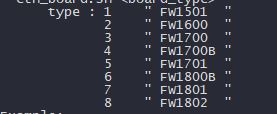
4 " FW1700B "

使用用例:

查看载板类型

source eth\_board.sh

显示结果:



设置FW1801载板网口

source eth\_board.sh 7

查看结果:

tx\_list : enaphyt4i0 enaphyt4i1 enp3s0f0 enp3s0f1 enp3s0f2 enp3s0f3

rx\_list : enaphyt4i1 enaphyt4i0 enp3s0f1 enp3s0f0 enp3s0f3 enp3s0f2

上述结果为enaphyt4i0和enaphyt4i4互环, enp3s0f0和enp3s0f1互环, enp3s0f2和enp3s0f3互环, tx发包与rx收包每一项一一对应

### get\_eth.sh

测试命令:get\_eth.sh

使用说明:设置设备上所有网口为收发包,除网讯芯片的子卡(设置网讯子卡见3.4.2.1小节),切换用户或重启设备要重新运行.

使用方法:

source get\_eth.sh

结果:



结果此为enp3s0f0和enp3s0f1互环, enp3s0f2和enp3s0f3互环:

发包:enp3s0f0 enp3s0f1 enp3s0f2 enp3s0f3

收包:enp3s0f1 enp3s0f0 enp3s0f3 enp3s0f2

#### 网讯子卡

网讯芯片子卡ethtool识别4光4电卡时,光口识别为Twisted,原因是使用PHY电口转光口,ethtool不能识别为Fiber,因此需要另外选择子卡类型

测试命令:iom\_wangxun.sh

使用说明:设置网讯子卡为发包口,使用前运行get\_eth.sh

使用方法:

source iom\_wangxun.sh <type>

type : 1 " 4 Fiber & 4 Twisted "

2 " 8 Twisted "

3 " 8 Fiber "

使用用例：

所使用IOM卡全部为4光4电

source iom\_wangxun.sh 1

### eth\_counter.sh

测试命令:eth\_counter.sh

使用说明:清理数据包,清理数据包前先设置网口

使用方法:

./eth\_counter.sh

使用用例：

./eth\_counter.sh

### tx\_packet.sh

测试命令:tx\_packet.sh

使用说明:对应网口进行发包,使用前先运行清理数据包脚本eth\_counter.sh

使用方法:

./tx\_packet.sh <delay> <count>

delay:发包中间延时(ns)

count:发包个数

0(无尽发包)使用无尽发包时,网口数量要对应使用CPU核数 或 延时使用1ns

使用网口测试脚本前需要文件pktgen.ko,运行init.sh生成pktgen.ko,工具目录下有pktgen.ko则不需要运行

./init.sh

确保网口处于RUNNING状态,使用ifconfig命令查看测试的网口是否处于RUNNING状态

ifconfig

enp3s0f0 Link encap:以太网 硬件地址 00:25:a9:01:72:c8

UP BROADCAST RUNNING MULTICAST MTU:1500 跃点数:1

接收数据包:44 错误:0 丢弃:0 过载:0 帧数:0

发送数据包:110 错误:0 丢弃:0 过载:0 载波:0

碰撞:0 发送队列长度:1000

接收字节:15048 (15.0 KB) 发送字节:20724 (20.7 KB)

用命令export查看发包网口tx\_list：

export

使用用例：

延时1ns发送10000个数据包

./tx\_packet.sh 1 10000

1ns延时无尽发送数据包,需要手动暂停(Ctrl+C)

./tx\_packet.sh 1 0

查看发包结果:

详见[3.4.5](#_show_rxtx.sh)小节

#### 测试IOM卡

使用网口测试脚本前需要文件pktgen.ko,运行init.sh生成pktgen.ko,工具目录下有pktgen.ko则不需要运行

./init.sh

确保网口处于RUNNING状态

ifconfig

enp3s0f0 Link encap:以太网 硬件地址 00:25:a9:01:72:c8

UP BROADCAST RUNNING MULTICAST MTU:1500 跃点数:1

接收数据包:44 错误:0 丢弃:0 过载:0 帧数:0

发送数据包:110 错误:0 丢弃:0 过载:0 载波:0

碰撞:0 发送队列长度:1000

接收字节:15048 (15.0 KB) 发送字节:20724 (20.7 KB)

需要IOM子卡的网口名,设置相应网口名发包

export tx\_list="enp3s0f0 enp3s0f1 enp3s0f2 enp3s0f3 enp4s0f0 enp4s0f1 enp4s0f2 enp4s0f3"

运行脚本

1ns延时无尽发送数据包

./tx\_packet.sh 1 0

运行出问题情况:

[ 76.698674] igb 0000:03:00.1 enp3s0f1: PCIe link lost, device now detached

[ 76.854666] igb 0000:03:00.3 enp3s0f3: PCIe link lost, device now detached

[ 77.758667] igb 0000:03:00.0 enp3s0f0: PCIe link lost, device now detached

[ 77.830802] igb 0000:03:00.2 enp3s0f2: PCIe link lost, device now detached

[ 78.494660] igb 0000:04:00.3 enp4s0f3: PCIe link lost, device now detached

[ 80.146680] igb 0000:04:00.1 enp4s0f1: PCIe link lost, device now detached

### show\_rxtx.sh

测试命令: show\_rxtx.sh

使用说明:查看对应网口的发包数

使用方法:

./show\_rxtx.sh

用命令export查看收包网口rx\_list,发包网口tx\_list：

export

使用用例：

./show\_rxtx.sh

显示结果：

enp3s0 10000 10000 ETH\_PASS

enp4s0 10000 10000 ETH\_PASS

### show\_error.sh

测试命令: show\_error.sh

使用说明:查看对应网口收到的错包数,设置收包网口rx\_list后则不需要设置,直接运行show\_error.sh查看

使用方法:

./show\_error.sh

使用用例：

./show\_error.sh

显示结果：

enp3s0 0 PASS

enp4s0 0 PASS

### iperf

测试命令: iperf

使用说明:测试网络性能,分为客户端与服务器,需要两台机器网口互相连接,两个网口需要配置相同网段,。

通用选项：服务器端与客户端都可以使用的选项

-i 以秒为单位显示报告间隔

-p 指定使用的端口

-u 使用udp数据包,使用udp数据包发送时,默认带宽为1Mbps,客户端需要额外指定发送带宽。

-o <filename> 输出为日志文件

-B

服务器端使用方法：

./iperf –s 以服务器形式,等待接收信息

客户端选项:

-c <IP> 以客户端形式,发送数据包到指定IP中

-t <time> 发送持续时间 默认为10秒

-u 发送UDP数据包

-b 仅在发送udp数据包时有效,指定发送带宽

-d 同时进行双向测试,即为服务器与客户端互相发送

-P 并行客户端的数量

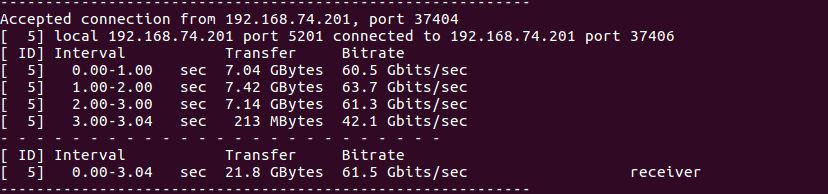
客户端使用用例:

向192.168.74.201 发送3秒数据包,客户端处每隔1s打印报告

./iperf –c 192.168.74.201 –t 3 –i 1

显示结果：

服务器端:



Interval表示时间间隔内情况,Transfer在间隔内传输数据量,Bitrate每秒传输位数

最后一行为总共统计。

向192.168.74.201 发送2小时的udp数据包,客户端处每隔1s打印报告

./iperf –c 192.168.74.201 –t 7200 –i 1 –u –b 1G

### tcpdump

测试命令:tcpdump

使用说明:截获网口传输数据,可以将截获数据放在windows下Wireshark查看

使用方法:

./tcpdump [opt]

-i <eth\_name> 监视指定网口数据

-w <filename> 将数据保存文件中,可在Wireshark下查看

-w eth0.pcap 将数据保存在eth0.pcap文件中,文件后缀为pcap

-c <num> 抓取指定数量包后退出

过滤数据包关键字

类型(缺省为host)

host(指定主机名或IP地址):host [www.baidu.com](http://www.baidu.com)监听发送到指定网址的数据包

net(指定网络段):net 192.168 监听网段为192.168的数据包

port(指明端口):port 80 监听80端口的数据包

portrange(指明端口范围):portrange 80-90 监听80-90端口的数据包

方向(缺省为dst or src)

src(监听数据包中的源地址):src 192.168.10.1 监听源地址为192.168.10.1的数据包

dst(监听数据包中的目的地址):dst 192.168.10.2 监听目的地址为192.168.10.2的数据包

dst or src(指明数据包中的目的地址)

dst and src

协议(缺省为所有协议): arp tcp udp等协议

使用用例:

监听网口enp3s0,并保存文件为enp3s0.pcap 存放在后台执行

./tcpdump –i enp3s0 –w enp3s0.pcap &

## USB

### lsusb

测试命令:lsusb

使用说明:显示所有usb设备信息

使用方法:

lsusb [opt]

opt:-v 显示设备详细信息

-t 以树状形式显示

-s <bus>:<devnum> 以指定总线号和设备号查看设备

-d <vendor>:<product> 以指定厂商号和产品号查看设备

使用用例：

lsusb

输出结果：

Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

查看指定设备并详细显示:

lsusb –s 001:001 -v

输出结果：

Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

Device Descriptor:

bLength 18

bDescriptorType 1

………

### fio

详见[3.3.1](#_disk.sh)小节

### disk.sh

详见[3.3.3](#_disk.sh_1)小节

## LCD

### lcd.sh

测试命令:lcd.sh

使用说明:设置lcd屏显示的字符

使用方法:

./lcd.sh <opt>

opt : reset 重置LCD设备

clear 清屏

send "msg" 屏幕显示字符

save "msg" 保存字符至开机/重置显示

使用用例：

重置LCD设备

./lcd.sh reset

LCD设备清屏

./lcd.sh clear

LCD屏幕显示FW1801

./lcd.sh send "FW1801"

设置LCD开机显示FW1801 test…

./lcd.sh save "FW1801 test …"

## LPC

## QSPI

## UART

### minicom

测试命令:minicom

使用说明:串口控制工具

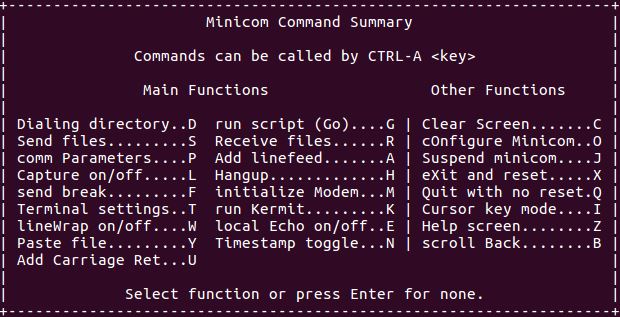
使用方法:

./minicom

进入界面后:

进入帮助界面

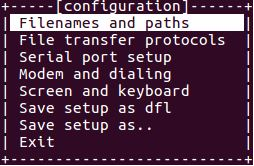
ctrl+a z



选择E打开local Echo on/off

输入 O 进入设置

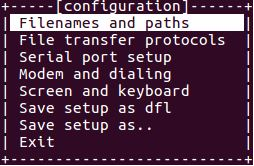
选择Serial port setup





输入对应字母，对串口设置,关闭所有流控(FG)

退出后回到configuration界面,并选择Save setup as dfl



回到帮助界面 输入x 保存并退出

重新进入程序后串口即为所设置串口

### stty

测试工具:stty

使用说明:设置串口参数

使用方法：

stty –F <devices> [opt] <speed>

opt:-echo 禁止回显

raw 原始输入

<speed> :波特率

使用用例：

设置串口/dev/ttyS0,波特率115200

stty –F /dev/ttyS0 –echo raw 115200

在系统后台不断读取串口ttyS0

cat /dev/ttyS0 &

往串口/dev/ttyS0写入123456,串口自环,则在终端看见输入的123456

echo 123456 > /dev/ttyS0

关闭后台不断读取串口/dev/ttyS0信息,不同串口修改

ps -elf | grep "cat /dev/ttyS0" | grep -v grep | awk '{print $4}' | xargs kill -9

## CAN

### canconfig

测试工具:candconfig

使用说明:设置对应can口属性,设置属性需要can口down状态

使用方法：

./canconfig <dev> <opt>

dev : 对应can口名

opt : bitrate 波特率

start

stop

restart

ctrlmode [type]

type : loopback on/off (需要can控制器支持)

berr-reporting on/off

使用用例:

设置can0 波特率

./canconfig can0 bitrate 2000000

开启can0

./canconfig can0 start

### candump

测试工具:candump

使用说明:接收对应can口的数据

使用方法：

./candump <can>

使用用例:

./candump can0

### candsend

测试工具:candsend

使用说明:给对应can口发送命令

使用方法：

./candsend <can> [opt] <can-msg>

opt:-i <hex> 指定帧编号默认为0x001

-e 使用扩展帧,帧编号4字节

--loop=<count> 循环发送count次

<can-msg>:最大长度8字节

使用用例:

./candsend can0 0x54 0x30

使用candump端显示结果:

<0x001> [2] 54 30

## GPIO

## FRU

使用网口测试脚本前需要文件at24.ko,运行init.sh生成at24.ko,工具目录下有at24.ko则不需要运行

./init.sh

FRU模板：

Diag工具目录下fru\_doc/fru\_fb.txt

创建文件fru\_file,写入fru信息

vi /tmp/fru\_file

### fru\_init.sh

测试工具:fru\_init.sh

使用说明:查找FRU设备所在I2C总线与地址,使用fru\_init.sh初始化fru设备,初始化后方可使用,fru读写测试,已经初始化则不用初始化

使用方法：

./fru\_init.sh <bus> <addr>

使用用例:

fru设备在,在总线号1,地址0x53上

./fru\_init.sh 1 0x53

fru设备在,在总线号1,地址0x53上

### fru.sh

测试工具:fru.sh

使用说明:对fru设备进行读写验证,初始化对应总线号上设备才可使用

使用方法：

./fru.sh <opt> <bus> <addr>

opt:read

write

使用用例写fru:

./fru.sh write 0 0x50

使用用例读fru:

./fru.sh read 0 0x50

## TMP422

### tmp422.sh

测试工具:tmp422.sh

使用说明：查找tmp422模块所在I2C总线与地址,运行tmp422.sh,之后查看/tmp/log/tmp422.log

使用方法:

./tmp422.sh <bus> <addr>

使用用例:

查看总线号0上,0x30和查看总线号1上,0x4d的tmp422器件

./tmp422.sh 0 0x30

./tmp422.sh 1 0x4d

查看结果模块检测温度结果,读取16进制转化为10进制:

Tmp422 bus: 1 addr:0x4d

Local Temperature :26

Remote Temperature 1:29

Remote Temperature 2:25

## RTC

### hwclock

测试工具:hwclock

使用说明:读取或设置RTC

使用方法:

hwclock [opt]

opt:-s 将系统时间写入RTC时间

-w 将RTC时间写入系统时间

使用用例:

hwclock

测试结果:

屏幕打印

Thu 15 Oct 2020 03:21:30 PM CST .823055 seconds

将系统时间写入RTC中

hwclock -s

## 内存

### memtester

测试工具:memtester

使用说明:简单内存测试工具。

使用方法:

./memtester <mem>[B|K|M|G] [loops]

mem:输入可用剩余内存 单位可以选择B,K,M,G 默认为M

loop:测试次数

使用用例:

查看系统剩余可用内存

free –m

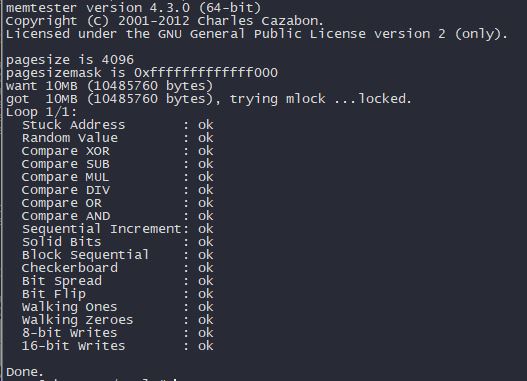
C:\Users\zheng.dongxiong.EMBEDWAY\AppData\Local\Microsoft\Windows\INetCache\Content.Word\free.jpg

可用剩余内存7404M

测试1次1G内存

./memtester 1G 1

测试结果: 屏幕打印



### devmem

测试工具:devmem

使用说明:对物理内存地址进行读或写。

使用方法:

./devmem <addr> <type> [date]

type : b 8位读取

h 16位读取

d 32位读取

date:有date时为写操作,无则为读操作

使用用例:

以16位读取物理内存中0xFF地址中的值

./devmem 0xFF h

显示结果：

/dev/mem opened.

Memory mapped at address 0x7fd002f8a000.

Value at address 0xFF (0x7fd002f8a0ff): 0xAAF0

## Fan

### fan.py

测试工具:fan.py

使用说明:读取当前风扇的转速,适用风扇转速数值在2个16位寄存器情况,填写reg\_addr时,只需填写低16位寄存器地址

使用方法:

./fan.py <bus> <dev\_addr> <reg\_addr>

使用用例:

读取总线1上, 0x30器件(fpga)中fan1的转速(fan1低16位0x22 高16位0x23)

./fan.py 1 0x30 0x22

显示结果：

当前fan1转速约为357RPM

fan\_speed 0x000fffff

fan\_speed 357 RPM

使用用例:

读取总线1上, 0x30器件(fpga)中fan2的转速(fan2低16位0x24 高16位0x25)

./fan.py 1 0x30 0x24

显示结果：

当前fan2转速约为10047RPM

fan\_speed 0x000091ce

fan\_speed 10047 RPM

## FPGA

### fpga\_version.sh

测试工具:fpga\_version.sh

使用说明:读取fpga版本信息, bus默认为1 addr默认为0x30, 若需读取其他总线上fpga信息需要填写对应参数

使用方法:

./fpga\_version.sh <bus> <addr>

使用用例:

读取总线1上, 0x30器件(fpga)中信息

./fpga\_version.sh

显示结果：

FPGA INFO:

FPGA version : 0001

build year : 2020

build date : 0309

build time : 1058

使用用例:

读取总线2上, 0x35器件(fpga)中信息

./fpga\_version.sh 2 0x35

显示结果：

FPGA INFO:

FPGA version : 0001

build year : 2020

build date : 0309

build time : 1058

## 拷机

### run\_full.sh

测试工具:run\_full.sh

使用说明:简单压力测试脚本,运行无尽发包,memtest,md5sum测试

使用方法:

./run\_full.sh

使用用例:

./run\_full.sh

查看结果：

log文件在当前路径下log\_[]/目录中, 其中[]指代为开始测试时间如开始测试时间20210425 18:58,则目录为log\_20210425\_1858

### stop\_run\_full.sh

测试工具:stop\_run\_full.sh

使用说明:暂停run\_full.sh脚本

使用方法:

./stop\_run\_full.sh

使用用例:

./stop\_run\_full.sh

### run\_test.sh

测试工具:run\_test.sh

使用说明:拷机脚本,指定执行run\_full.sh的时间

使用方法:

./run\_test.sh <hours>

hours : 单位为小时

使用用例:

拷机4小时

./run\_test.sh 4

查看结果:

见[3.18.1](#_run_full.sh)小节

## VGA

### x11perf

测试工具:x11perf

使用说明:需要接显示器进行2D图形测试

使用方法:

./x11perf

使用用例:

./x11perf [opt]

-all : 测试所有项(花费时间可能过长)

-time 测试项时间(s 默认5s)

-help 查看可以测试项的参数(可供需求挑选,不需要-all全部测试)

使用方法:

测试所有项

./x11perf –all

### glxgears

测试工具:glxgear

使用说明:需要接显示器进行2D, 3D测试

使用方法:

./glxgears

使用用例:

./glxgears

1. Linux环境工具

## 更改进程优先级

### renice

命令:renice

使用说明:重新设置进程的优先级-20-19,数字越小优先级越高,只有root权限可以设置负数。运行程序时默认为0

使用方法:

查找进程ID

ps –elf | grep "进程名称" | awk '{print $4}'

更改优先级

renice [priority] 进程ID

priority: -20~19

使用用例:

后台运行md5sum程序,查找进程ID

ps –elf | grep "md5sum" | awk '{print $4}'

进程ID：7932

更改md5sum的优先级为10

renice 10 7932

后台运行脚本test.sh,查找进程ID

pe –elf | grep "test.sh" | awk '{print $4}'

进程ID：7851

更改test.sh的优先级为-10

renice -10 7851

## 查看cpu使用

### top

命令:top

使用说明:可以查看当前cpu使用情况，输入不同字符显示不同情况

使用方法:

top

输入top之后单独输入字符

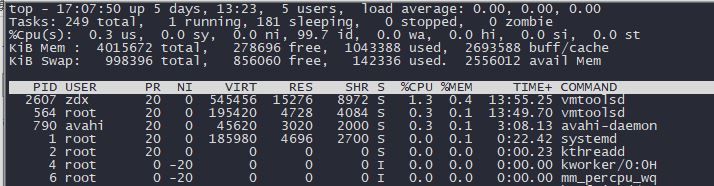
1 显示每个cpu的使用

f 增加显示功能

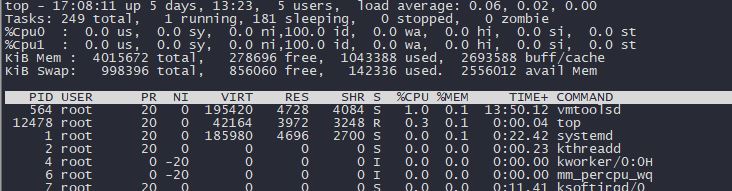
使用用例:

top

显示情况：



输入1：



us:用户空间占用CPU百分比

sy：内核空间占用CPU百分比

ni：用户进程空间内改变过优先级的进程占用CPU百分比

id：空闲CPU百分比

wa：等待输入输出的CPU时间百分比

hi：硬件中断占CPU百分比

si：软件中断占CPU百分比

st：实时