



GEC100 测试手册

Revision History

Draft Date	Revision No.	Description	Author
2024/05/22	V1.0	1.初始版本。	invoker_li
2024/06/26	V1.1	1.更新 HDMI 测试	



目录

1 GEC100 简单测试	1
1.1 GEC100 开机测试	1
1.2 GEC100 跨平台文件拷贝测试	3
1.2.1 通过 OpenSSH	3
1.2.2 通过 NFS	5
1.3 蜂鸣器测试	11
1.4 按键测试	11
1.5 DDR 读写测试	11
1.6 USB HOST 接口测试	12
1.7 GEC100 网口测试	14
1.8 AUDIO 音频测试	17
2 配置静态网络 IP	21
3 GEC100 接口测试	23
3.1 RS485 测试	23
3.1.1 接口功能	23
3.1.2 RS485 使用说明	24
3.1.3 代码截取	26
3.2 RS422 测试	27
3.2.1 接口功能	27
3.2.2 RS422 使用说明	27
3.2.3 代码截取	29
3.3 RS232 测试	30
3.3.1 接口功能	30
3.3.2 RS232 使用说明	30
3.3.3 代码截取	32
3.4 CAN 测试	32
3.4.1 接口功能	32
3.4.2 CAN 使用说明	33
3.4.3 代码截取	36
3.5 TCP_UDP 测试	36
3.5.1 接口功能	36
3.5.2 TCP_UDP 使用说明	37
3.5.3 代码截取	38
3.6 AO/AI 测试	39
3.6.1 接口功能	39
3.6.2 AO/AI 测试说明	39
3.6.3 代码截取	40
3.7 DO/DI 测试	41



3.7.1 接口功能.....	41
3.7.2 DO/DI 测试说明	41

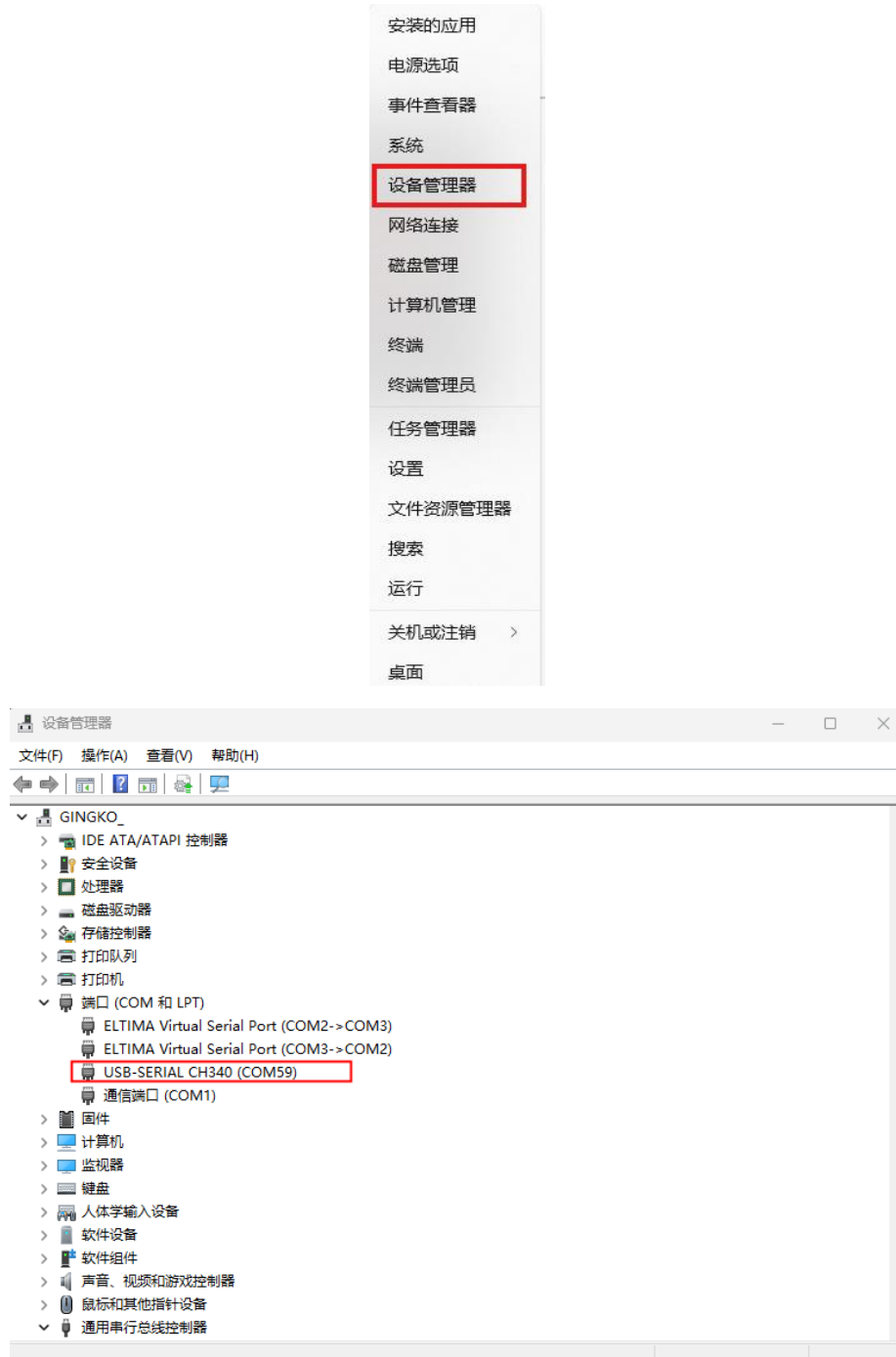
1 GEC100 简单测试

1.1 GEC100 开机测试

请将 GEC100 的 PA(正极)、PB(负极)分别接入电源正负极 (DC 13V~30V)，打开电源开关，使用 Type-C 线将 GEC100 UART 接口连接至 PC 机 USB 接口 (作为调试串口)，将 GEC100 上电启动。



打开 Windows 设备管理器，确认 GEC100 UART 调试串口对应的 COM 端口号。



打开串口调试终端 SecureCRT，选择对应的 COM 端口号，设置波特率为 115200，8N1，无校验位。建立串口连接，系统启动完成，将会自动登录 root 用户，串口终端会打印如下类似启动信息。



```
Serial-COM59 - SecureCRT
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)
Serial-COM59
Starting system message bus: [ 4.445423] urandom_read: 5 callbacks suppressed
[ 4.445435] random: dbus-uuidgen: uninitialized urandom read (12 bytes read)
[ 4.458724] random: dbus-uuidgen: uninitialized urandom read (8 bytes read)
done
Starting rpcbind: OK
Starting network: [ 4.849142] libphy: 4500000.eth: probed
[ 4.853803] sunxi-gmac 4500000.eth eth0: eth0: Type(7) PHY ID 02430c54 at 1 IRQ poll (4500000.eth-0:01)
Error: NextHop has invalid gateway.
FAIL
Starting ntpd: OK
Starting mosquitto: OK
[ 6.873481]
[ 6.873481] insmod_device_driver
[ 6.873481]
[ 6.880734] sunxi_usb_udec 4100000.udec-controller: 4100000.udec-controller supply udc not found, using dummy regulator
[ 10.133516] random: crng init done
Starting sshd: OK
Starting telnetd: OK
Starting dockerd: OK
Starting NFS statd: OK
Starting NFS services: OK
Starting NFS daemon: OK
Starting NFS mountd: OK
Starting vsftpd: OK
mount: /run/media/mmcblk0rmpb: /dev/mmcblk0rmpb is not a block device.
mount: /run/media/mmcblk0p7: wrong fs type, bad option, bad superblock on /dev/mmcblk0p7, missing codepage or helper program
, or other error.
mount: /run/media/mmcblk0p3: wrong fs type, bad option, bad superblock on /dev/mmcblk0p3, missing codepage or helper program
, or other error.
[ 11.368121] FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
mount: /run/media/mmcblk0p6: wrong fs type, bad option, bad superblock on /dev/mmcblk0p6, missing codepage or helper program
, or other error.
mount: /run/media/mmcblk0p4: wrong fs type, bad option, bad superblock on /dev/mmcblk0p4, missing codepage or helper program
, or other error.
mount: /run/media/mmcblk0p2: wrong fs type, bad option, bad superblock on /dev/mmcblk0p2, missing codepage or helper program
, or other error.
mount: /run/media/mmcblk0p8: wrong fs type, bad option, bad superblock on /dev/mmcblk0p8, missing codepage or helper program
, or other error.
/etc/init.d/rcs: line 46: Launcher: command not found

welcome to Allwinner kunoos Platform
TI13-GINGKO login: root (automatic login)

root@TI13-GINGKO:~# [ 14.726803] bridge: filtering via arp/ip/ip6tables is no longer available by default. Update your scr
ipts to load br_netfilter if you need this.
[ 14.747653] Bridge firewalling registered
[ 15.158566] Initializing XFRM netlink socket
[ 16.233731] sunxi-gmac 4500000.eth eth0: Link is Up - 100Mbps/Full - flow control off
[ 16.242673] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[ 33.113520] p10-18: disabling
[ 33.116859] p10-33: disabling
```

1.2 GEC100 跨平台文件拷贝测试

PC 机和 GEC100 之间传送文件的常见方式如下:

1. 通过 Linux 系统启动卡、U 盘等存储介质方式拷贝。
2. 通过 NFS、TFTP、OpenSSH 等网络方式拷贝。

本章节主要测试 NFS,OpenSSH 方式传送

1.2.1 通过 OpenSSH

OpenSSH 是 SSH(Secure Shell)协议的免费开源实现。SSH 协议族可用来进行远程控制,或在计算机之间传送文件,GEC100 文件系统默认已支持 SSH 库。

在 Ubuntu 系统中执行如下命令,查询是否已安装 OpenSSH。由下图可见系统已自带 OpenSSH。如未安装,请先自行正确安装 OpenSSH。

```
Host# ssh -v
```



Command 1

```
zh@zh-virtual-machine:~$ ssh -v
usage: ssh [-46AaCfGgKkMnNqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
[-D [bind_address:]port] [-E log_file] [-e escape_char]
[-F configfile] [-I pkcs11] [-i identity_file]
[-J [user@]host[:port]] [-L address] [-l login_name] [-m mac_spec]
[-O ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
[-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
[user@]hostname [command]
zh@zh-virtual-machine:~$ █
```

将 GEC100 以太网口通过路由器与 PC 机进行网络连接。执行如下命令可自动获取到 GEC100 IP 地址，“-i”用于指定网卡，eth0 为网卡名字，请根据实际情况修改。执行如下命令可查询 GEC100 网卡 IP 地址。本次查询到的 IP 地址是 192.168.1.110。

Target# ifconfig

Command 2

```
^oot@T113-GINGKO:/# ifconfig
docker0  Link encap:Ethernet  Hwaddr 02:42:77:A4:85:79
         inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
         UP BROADCAST MULTICAST  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:0
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0     Link encap:Ethernet  Hwaddr A6:8A:6F:53:98:DC
         inet addr:192.168.1.110  Bcast:192.168.1.255  Mask:255.255.255.0
         inet6 addr: fe80::a48a:6fff:fe53:98dc/64 Scope:Link
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:1631 errors:0 dropped:0 overruns:0 frame:0
         TX packets:83 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:228423 (223.0 KiB)  TX bytes:13660 (13.3 KiB)
         Interrupt:38

lo       Link encap:Local Loopback
         inet addr:127.0.0.1  Mask:255.0.0.0
         inet6 addr: ::1/128 Scope:Host
         UP LOOPBACK RUNNING  MTU:65536  Metric:1
         RX packets:15634 errors:0 dropped:0 overruns:0 frame:0
         TX packets:15634 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:1156908 (1.1 MiB)  TX bytes:1156908 (1.1 MiB)

^oot@T113-GINGKO:/#
```

(1) 使用 OpenSSH 从 PC 机传送文件至 GEC100 执行如下命令在 Ubuntu 系统中

创建文件 test1，并使用 OpenSSH 命令将 test1 文件拷贝至 GEC100 文件系统根录。

Host# touch test1

Command 3

```
zh@zh-virtual-machine:~/doc/nfsroot$ touch test1
zh@zh-virtual-machine:~/doc/nfsroot$ ls test1
test1
zh@zh-virtual-machine:~/doc/nfsroot$ scp test1 root@192.168.1.110:/
```

若出现提示“Are you sure you want to continue connecting (yes/no)?”，请输入：yes。在 GEC100 文件系统执行如下命令，可看到从 Ubuntu 拷贝过来的文件，如下图所示。



```

root@T113-GINGKO:/# ls
THIS_IS_NOT_YOUR_ROOT_FILESYSTEM  lib32  libexec  linuxrc  lost+found  media  mnt  opt  proc  root  run  sbin  sys  test1  tmp  usr  var
bin  dev  etc  helloworld.c  home  init  lib
root@T113-GINGKO:/#

```

(2) 使用 OpenSSH 从 GEC100 传送文件至 PC 机执行如下命令，在 GEC100 文件系统根目录创建文件 test2

Target# cd /

Target# touch test2

Command 4

```

root@T113-GINGKO:/# touch test2
root@T113-GINGKO:/# ls
THIS_IS_NOT_YOUR_ROOT_FILESYSTEM  lib32  libexec  linuxrc  lost+found  media  mnt  opt  proc  root  run  sbin  sys  test1  test2  tmp  usr  var
bin  dev  etc  helloworld.c  home  init  lib

```

在 Ubuntu 系统执行如下命令，将 GEC100 文件系统根目录下的 test2 文件拷贝至 Ubuntu 系统"/home/zh/"目录下。若传输内容为文件夹，请在 scp 后面添加参数"-r"。

Host# sudo scp root@192.168.1.110:/test2 /home/zh/

Host# ls

Command 5

```

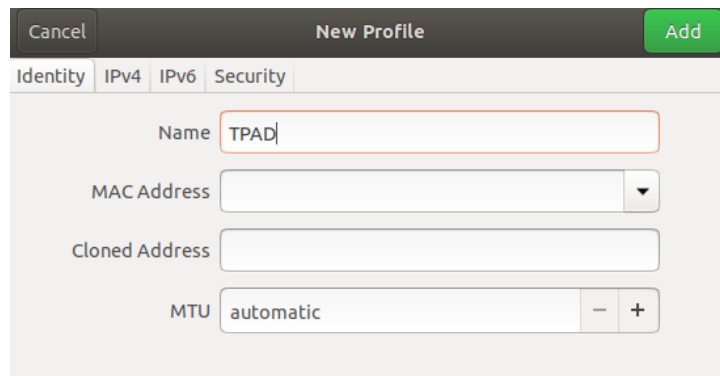
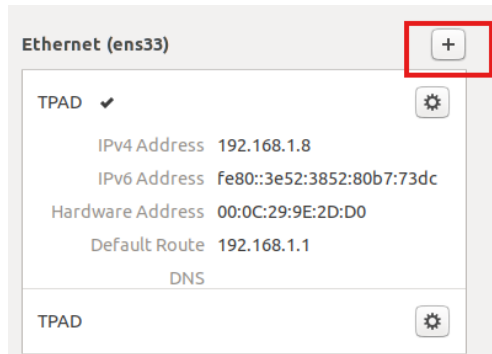
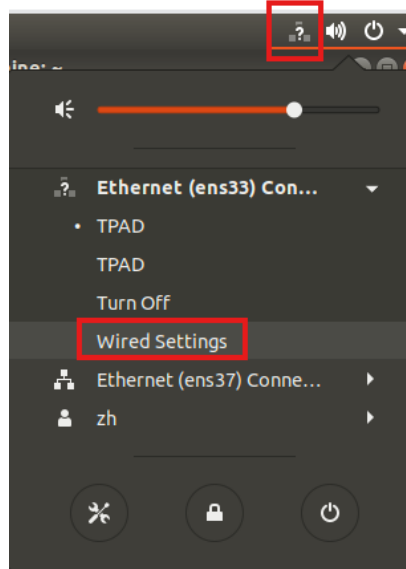
zh@zh-virtual-machine:~$ sudo scp root@192.168.1.110:/test2 /home/zh/
zh@zh-virtual-machine:~$ ls
doc  examples.desktop  mnt  nfsroot  T113-i  test2  公共的  模板  视频  图片  文档  下载  音乐  桌面
zh@zh-virtual-machine:~$ █

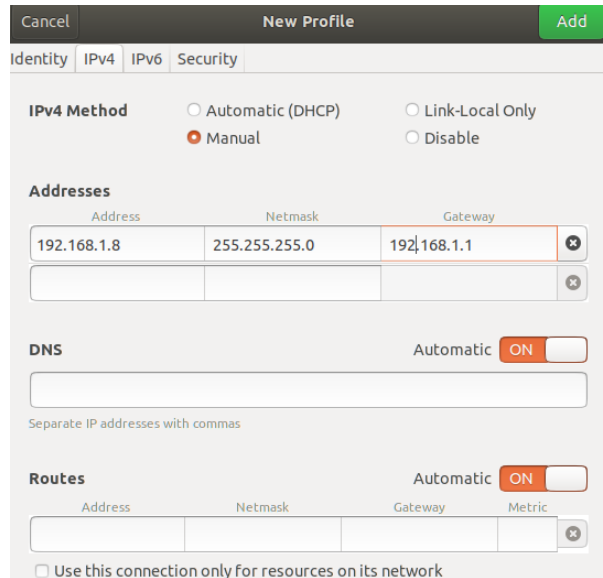
```

1.2.2 通过 NFS

(1) 给上位机的 linux 系统配置一个静态 IP 地址为 192.168.1.8

具体实施步骤如下:右键点击 ubuntu 右上角的网络连接图标(上下箭头或者扇形)->再点击"Wired Settings"->点击"右上角+号"->Identity->Name 填"TPAD"->选择"IPv4"->"Manual"-> >Address 为: 192.168.1.8,Netmask: 255.255.255.0,Gateway: 192.168.1.1->最后点击"Add"退出即可





Address	Netmask	Gateway
192.168.1.8	255.255.255.0	192.168.1.1

Address	Netmask	Gateway	Metric

配置好 IP 地址，输入以下命令，确保 IP 地址为 192.168.1.8

Host# ifconfig

Command 6

```
zh@zh-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.8 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::3e52:3852:80b7:73dc prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:9e:2d:d0 txqueuelen 1000 (Ethernet)
    RX packets 1370 bytes 123690 (123.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 265 bytes 42173 (42.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

(2) 安装 NFS 网络服务，输入以下命令：

Host# sudo apt-get install nfs-kernel-server

Command 7

```
zh@zh-virtual-machine:~$ sudo apt-get install nfs-kernel-server
[sudo] password for zh:
```

注：如果是 Ubuntu18.04 系统之后需要添加以下配置：

Host# sudo vim /etc/default/nfs-kernel-server

RPCNFSDOPTS="--nfs-version 2,3,4 --debug --syslog"

Command 8

```
zh@zh-virtual-machine:~$ sudo vim /etc/default/nfs-kernel-server
```

```
zh@zh-virtual-machine: ~  
File Edit View Search Terminal Help  
# Number of servers to start up  
RPCNFSDCOUNT=8  
  
# Runtime priority of server (see nice(1))  
RPCNFSDPRIORITY=0  
  
# Options for rpc.mountd.  
# If you have a port-based firewall, you might want to set up  
# a fixed port here using the --port option. For more information,  
# see rpc.mountd(8) or http://wiki.debian.org/SecuringNFS  
# To disable NFSv4 on the server, specify '--no-nfs-version 4' here  
RPCMOUNTDOPTS="--manage-gids"  
  
# Do you want to start the svcgssd daemon? It is only required for Kerberos  
# exports. Valid alternatives are "yes" and "no"; the default is "no".  
NEED_SVCGSSD=""  
  
# Options for rpc.svcgssd.  
RPCSVCSSDOPTS="--nfs-version2,3,4 --debug --syslog"  
~  
~  
~  
~  
~
```

- (3) 在 Ubuntu 系统中指定 NFS 网络服务的共享目录，在/home/zh 目录下创建 nfsroot 文件作为，NFS 共享目录

Host# sudo mkdir /home/zh/nfsroot

Command 9

```
zh@zh-virtual-machine:~$ mkdir nfsroot  
zh@zh-virtual-machine:~$ ls  
examples.desktop  nfsroot  test1  公共的  视频  文档  音乐  
mnt               T113-i  test3  模板  图片  下载  桌面
```

Host# sudo vim /etc/exports

Command 10

```
zh@zh-virtual-machine:~$ sudo vim /etc/exports
```



```
zh@zh-virtual-machine: ~  
File Edit View Search Terminal Help  
# /etc/exports: the access control list for filesystems which may be exported  
# to NFS clients. See exports(5).  
#  
# Example for NFSv2 and NFSv3:  
# /srv/homes hostname1(rw, sync, no_subtree_check) hostname2(ro, sync, no_sub  
tree_check)  
#  
# Example for NFSv4:  
# /srv/nfs4 gss/krb5i(rw, sync, fsid=0, crossmnt, no_subtree_check)  
# /srv/nfs4/homes gss/krb5i(rw, sync, no_subtree_check)  
#  
/home/zh/nfsroot *(rw, sync, no_root_squash)
```

(4) 重启 Ubuntu 的 NFS 网络服务

Host# sudo service nfs-kernel-server restart

Command 11

```
zh@zh-virtual-machine:~$ sudo service nfs-kernel-server restart  
zh@zh-virtual-machine:~$
```

(5) 在 Ubuntu 系统的 NFS 网络服务的共享目录/home/zh/nfsroot 中添加一个测试文件

Host# cd /home/zh

Host# mkdir nfsroot

Command 12

```
zh@zh-virtual-machine:~$ cd /home/zh/  
zh@zh-virtual-machine:~$ mkdir nfsroot
```

Host# cd /home/zh/nfsroot

Host# touch test



Command 13

```
zh@zh-virtual-machine:~$ cd /home/zh/nfsroot/  
zh@zh-virtual-machine:~/nfsroot$ touch test  
zh@zh-virtual-machine:~/nfsroot$ ls  
test  
zh@zh-virtual-machine:~/nfsroot$ █
```

(6) GEC100 利用 NFS 网络服务, 将 Ubuntu 系统中/home/zh/nfsroot 共享目录挂接到 GEC100 的/mnt 目录下, 执行以下命令:

```
Target# ifconfig eth0 192.168.1.110
```

/注: 给 GEC100 的网卡配置 IP 地址,此 IP 地址必须和 Ubuntu 系统的 IP 地址 192.168.1.8 在同一个网段。

```
Target# ping 192.168.1.8
```

Command 14

```
root@T113-GINGKO:/# ifconfig eth0 192.168.1.110  
root@T113-GINGKO:/# ping 192.168.1.8  
PING 192.168.1.8 (192.168.1.8): 56 data bytes  
64 bytes from 192.168.1.8: seq=0 ttl=64 time=0.534 ms  
64 bytes from 192.168.1.8: seq=1 ttl=64 time=0.480 ms  
64 bytes from 192.168.1.8: seq=2 ttl=64 time=0.521 ms  
64 bytes from 192.168.1.8: seq=3 ttl=64 time=0.453 ms
```

出现类似: 64 bytes from 192.168.1.8: seq=0 ttl=64 time=3.598 ms 信息表示网络联通了, 按 ctrl+c 退出 ping 命令。

继续执行:

```
Target# mount -t nfs -o nolock 192.168.1.8:/home/zh/nfsroot /mnt
```

```
Target# ls /mnt
```

Command 15

```
root@T113-GINGKO:7# mount -t nfs -o nolock 192.168.1.8:/home/zh/nfsroot /mnt  
root@T113-GINGKO:7# ls /mnt/  
test  
root@T113-GINGKO:7# █
```

可以看到在 GEC100/mnt 目录下有 Ubuntu 系统 NFS 共享目录的 test 文件。



1.3 蜂鸣器测试

进入 GEC100 文件系统，执行如下命令开启、关闭蜂鸣器。

```
Target# echo 0 > /sys/class/leds/user-beep/brightness //开启 BEEP
```

```
Target# echo 1 > /sys/class/leds/user-beep/brightness //关闭 BEEP
```

1.4 按键测试

GEC100 包含，1 个 DOWNLOAD 按键。参考《Linux 系统使用》文档步骤，可固化 Linux 系统至 eMMC

1.5 DDR 读写测试

本小节使用文件系统自带内存读写工具测试 DDR 读写性能。DDR 读写速度受测试方法和实际情况影响，如下测试数据仅供参考。

(1) DDR 读速度测试

进入 GEC100 系统，执行如下命令对 DDR 进行读速度测试。"-P"参数指定 CPU 核心数量。

```
Target# bw_mem -P 2 30M frd
```

Command 16

```
root@T113-GINGKO:~# bw_mem -P 2 30M frd
30.00 2053.96
root@T113-GINGKO:~#
```

测试从 DDR 中读取 30MByte 数据，可看到本次测试的读速度为 2205.96MB/s。

(2) DDR 写速度测试



进入 GEC100 系统，执行如下命令对 DDR 进行写速度测试。

Target# bw_mem -P 2 30M fwr

Command 17

```
root@T113-GINGKO:~# bw_mem -P 2 30M fwr
30.00 612.21
root@T113-GINGKO:~#
```

测试写入 30MByte 数据至 DDR，可看到本次测试的写速度为 612.21MB/s。

(3) DDR 拷贝速度测试

进入 GEC100 系统，执行如下命令对 DDR 进行拷贝速度测试。

Target# stream -M 30M -P 2

Command 18

```
root@T113-GINGKO:~# bw_mem -P 2 30M fwr
30.00 612.21
root@T113-GINGKO:~# stream -M 30M -P 2
STREAM copy latency: 14.72 nanoseconds
STREAM copy bandwidth: 2173.37 MB/sec
STREAM scale latency: 22.15 nanoseconds
STREAM scale bandwidth: 1444.89 MB/sec
STREAM add latency: 33.78 nanoseconds
STREAM add bandwidth: 1421.07 MB/sec
STREAM triad latency: 36.62 nanoseconds
STREAM triad bandwidth: 1310.62 MB/sec
root@T113-GINGKO:~#
```

测试拷贝 30MByte 数据至 DDR 中，可看到本次测试的拷贝速度为 2173.37MB/s。

1.6 USB HOST 接口测试

本章节使用 U 盘进行测试。不同品牌、格式或容量的硬盘，测试结果会有所差异。

```
root@T113-GINGKO:~# [19916.285082] usb 1-1.1: USB disconnect, device number 4
[19916.292605] blk_update_request: I/O error, dev sda, sector 1 op 0x1:(WRITE) flags 0x0 phys_seg 1 prio class 0
[19916.303882] Buffer I/O error on dev sda, logical block 1, lost async page write
[19918.629595] usb 1-1.1: new high-speed USB device number 5 using sunxi-ehci
[19918.784235] usb-storage 1-1.1:1.0: USB Mass Storage device detected
[19918.792076] scsi host0: usb-storage 1-1.1:1.0
[19919.850660] scsi 0:0:0:0: Direct-Access VendorCo ProductCode 2.00 PQ: 0 ANSI: 4
[19919.866946] sd 0:0:0:0: [sda] 15728640 512-byte logical blocks: (8.05 GB/7.50 GiB)
[19919.882941] sd 0:0:0:0: [sda] write Protect is off
[19919.888340] sd 0:0:0:0: [sda] Mode Sense: 03 00 00 00
[19919.895013] sd 0:0:0:0: [sda] No Caching mode page found
[19919.901047] sd 0:0:0:0: [sda] Assuming drive cache: write through
[19919.954125] sda:
[19919.961889] sd 0:0:0:0: [sda] Attached SCSI removable disk
[19920.089904] FAT-fs (sda): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
root@T113-GINGKO:~# █
```

执行如下命令可查询挂载路径。挂载名和挂载路径请以实际查询为准，如下图所示。



Target# df -h

Command 19

```
root@T113-GINGKO:~# echo 3 > /proc/sys/vm/drop_caches
[19962.863225] sh (1534): drop_caches: 3
root@T113-GINGKO:~# time dd if=/dev/zero of=/dev/sda bs=1024k count=500
500+0 records in
500+0 records out

real    0m35.283s
user    0m0.000s
sys     0m4.237s
root@T113-GINGKO:~#
```

- (1) 写速度测试执行如下命令，测试 USB1 HOST 接口写速度。

Target# echo 3 > /proc/sys/vm/drop_caches

Target# time dd if=/dev/zero of=/dev/sda bs=1024K count=500

Command 20

```
root@T113-GINGKO:~# echo 3 > /proc/sys/vm/drop_caches
[19962.863225] sh (1534): drop_caches: 3
root@T113-GINGKO:~# time dd if=/dev/zero of=/dev/sda bs=1024k count=500
500+0 records in
500+0 records out

real    0m35.283s
user    0m0.000s
sys     0m4.237s
```

此处通过 USB1 HOST 接口一共写 500MByte 测试数据至固态硬盘，可看到本次测试的 USB1 HOST 接口写速度约为： $500\text{MByte} / 35.28\text{s} \approx 14.17\text{MB/s}$ 。

- (2) 读速度测试执行如下命令，测试 USB1 HOST 接口读速度。

Target# echo 3 > /proc/sys/vm/drop_caches

Target# time dd if=/dev/sda of=/dev/null bs=1024K count=500



Command 21

```
root@T113-GINGKO:~#  
root@T113-GINGKO:~# echo 3 > /proc/sys/vm/drop_caches  
[20446.963587] sh (1534): drop_caches: 3  
root@T113-GINGKO:~# time dd if=/dev/sda of=/dev/null bs=1024K count=500  
500+0 records in  
500+0 records out  
  
real    0m23.022s  
user    0m0.000s  
sys     0m2.887s  
root@T113-GINGKO:~#
```

此处通过 USB1 HOST 接口从固态硬盘读取 500MByte 数据，可看到本次测试的 USB1 HOST 接口读速度约为 $500\text{MByte} / 23.022\text{s} \approx 27.718\text{MB/s}$ 。

1.7 GEC100 网口测试

GEC100 配备一个百兆以太网口 LAN，请使用网线将 GEC100 网口、PC 机连接然后启动 GEC100。

(1) 网络连通测试

进入 GEC100 文件系统，执行如下命令获取 IP 以及查看网口配置信息，其中 eth0 为 LAN 以太网驱动名称。

Target# ifconfig

Command 22

```
root@T113-GINGKO:~# ifconfig  
docker0  Link encap:Ethernet  Hwaddr 02:42:1b:E5:EE:DD  
         inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0  
         UP BROADCAST MULTICAST  MTU:1500  Metric:1  
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
         collisions:0 txqueue:0  
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)  
  
eth0     Link encap:Ethernet  Hwaddr 7E:D8:6E:1D:2B:18  
         inet6 addr: fe80::7cd8:6eff:fe1d:2b1b/64 Scope:Link  
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
         RX packets:72 errors:0 dropped:0 overruns:0 frame:0  
         TX packets:14 errors:0 dropped:0 overruns:0 carrier:0  
         collisions:0 txqueue:1000  
         RX bytes:8520 (8.3 KiB)  TX bytes:1620 (1.5 KiB)  
         Interrupt:38  
  
lo       Link encap:Local Loopback  
         inet addr:127.0.0.1  Mask:255.0.0.0  
         inet6 addr: ::1/128 Scope:Host  
         UP LOOPBACK RUNNING  MTU:65536  Metric:1  
         RX packets:834 errors:0 dropped:0 overruns:0 frame:0  
         TX packets:834 errors:0 dropped:0 overruns:0 carrier:0  
         collisions:0 txqueue:1000  
         RX bytes:61708 (60.2 KiB)  TX bytes:61708 (60.2 KiB)  
  
root@T113-GINGKO:~#
```

添加与虚拟机同一网段下的 IP 例如请执行如下命令，可按"Ctrl + C"终止 ping 命令。



Target# ifconfig eth0 192.168.1.110

Command 23

```
root@TI13-GINGKO:~# ping 192.168.1.110
PING 192.168.1.110 (192.168.1.110): 56 data bytes
0
--- 192.168.1.110 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss
root@TI13-GINGKO:~# ifconfig eth0 192.168.1.110
root@TI13-GINGKO:~# ping 192.168.1.8
PING 192.168.1.8 (192.168.1.8): 56 data bytes
64 bytes from 192.168.1.8: seq=0 ttl=64 time=0.812 ms
64 bytes from 192.168.1.8: seq=1 ttl=64 time=0.464 ms
64 bytes from 192.168.1.8: seq=2 ttl=64 time=0.444 ms
64 bytes from 192.168.1.8: seq=3 ttl=64 time=0.513 ms
64 bytes from 192.168.1.8: seq=4 ttl=64 time=0.474 ms
```

(1) TCP 带宽测试如下

使用 Iperf3 工具测试 GEC100 与 PC 机的网络通信带宽。若未安装，可在 Ubuntu 系统中执行"sudo apt-get install iperf3"安装。在 Ubuntu 系统执行如下命令查看 PC 机 IP 地址并等待 GEC100 连接。

Host# ifconfig

Host# iperf3 -s

Command 24

```
zh@zh-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.137.128 netmask 255.255.255.0 broadcast 192.168.137.255
    inet6 fe80::6d49:37b7:b168:8e2a prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:9e:2d:d0 txqueuelen 1000 (Ethernet)
    RX packets 1060295 bytes 1508133436 (1.5 GB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 174070 bytes 11381584 (11.3 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 68710 bytes 5071431 (5.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 68710 bytes 5071431 (5.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

zh@zh-virtual-machine:~$ iperf3 -s
-----
Server listening on 5201
-----
```

在 GEC100 文件系统执行如下命令测试网络通信带宽，命令中 192.168.31.43 为 PC 机 IP 地址。测试完成后，Ubuntu 系统和 GEC100 均会打印测试结果。不同测试环境，测试结果将会有所差异。



Target# iperf3 -c 192.168.31.43 -i 1 -P 4

Command 25

```

root@T113-GINGKO:~# iperf3 -c 192.168.137.128 -i 1 -P 4
iperf3: error - unable to connect to server: Network is unreachable
root@T113-GINGKO:~# iperf3 -c 192.168.1.8 -i 1 -P 4
Connecting to host 192.168.1.8, port 5201
[ 5] local 192.168.1.110 port 60096 connected to 192.168.1.8 port 5201
[ 7] local 192.168.1.110 port 60098 connected to 192.168.1.8 port 5201
[ 9] local 192.168.1.110 port 60100 connected to 192.168.1.8 port 5201
[11] local 192.168.1.110 port 60102 connected to 192.168.1.8 port 5201
[ ID] Interval           Transfer             Bitrate             Retrr  Cwnd
[ 5] 0.00-1.00      sec  2.70 MBytes        22.7 Mbits/sec      0    26.9 KBytes
[ 7] 0.00-1.00      sec  3.24 MBytes        27.2 Mbits/sec      0    31.1 KBytes
[ 9] 0.00-1.00      sec  2.86 MBytes        24.0 Mbits/sec      0    28.3 KBytes
[11] 0.00-1.00      sec  2.99 MBytes        25.1 Mbits/sec      0    29.7 KBytes
[SUM] 0.00-1.00      sec  11.8 MBytes        98.9 Mbits/sec      0
-----
[ 5] 1.00-2.00      sec  2.82 MBytes        23.6 Mbits/sec      0    32.5 KBytes
[ 7] 1.00-2.00      sec  3.11 MBytes        26.1 Mbits/sec      0    32.5 KBytes
[ 9] 1.00-2.00      sec  2.35 MBytes        19.7 Mbits/sec      0    28.3 KBytes
[11] 1.00-2.00      sec  3.11 MBytes        26.1 Mbits/sec      0    32.5 KBytes
[SUM] 1.00-2.00      sec  11.4 MBytes        95.5 Mbits/sec      0
-----
[ 5] 2.00-3.00      sec  2.91 MBytes        24.4 Mbits/sec      0    32.5 KBytes
[ 7] 2.00-3.00      sec  2.98 MBytes        25.0 Mbits/sec      0    32.5 KBytes
[ 9] 2.00-3.00      sec  2.35 MBytes        19.7 Mbits/sec      0    35.4 KBytes
[11] 2.00-3.00      sec  2.98 MBytes        25.0 Mbits/sec      0    33.9 KBytes
[SUM] 2.00-3.00      sec  11.2 MBytes        94.2 Mbits/sec      0
-----
[ 5] 3.00-4.00      sec  2.54 MBytes        21.3 Mbits/sec      0    32.5 KBytes
[ 7] 3.00-4.00      sec  2.98 MBytes        25.0 Mbits/sec      0    36.8 KBytes
[ 9] 3.00-4.00      sec  2.91 MBytes        24.4 Mbits/sec      0    35.4 KBytes
[11] 3.00-4.00      sec  2.73 MBytes        22.9 Mbits/sec      0    33.9 KBytes
[SUM] 3.00-4.00      sec  11.2 MBytes        93.6 Mbits/sec      0
-----
[ 5] 4.00-5.00      sec  2.63 MBytes        22.1 Mbits/sec      0    32.5 KBytes
[ 7] 4.00-5.00      sec  2.98 MBytes        25.0 Mbits/sec      0    36.8 KBytes
[ 9] 4.00-5.00      sec  2.91 MBytes        24.4 Mbits/sec      0    35.4 KBytes
[11] 4.00-5.00      sec  2.86 MBytes        24.0 Mbits/sec      0    33.9 KBytes
[SUM] 4.00-5.00      sec  11.4 MBytes        95.5 Mbits/sec      0
-----
[ 5] 5.00-6.00      sec  2.54 MBytes        21.3 Mbits/sec      0    32.5 KBytes
[ 7] 5.00-6.00      sec  3.11 MBytes        26.1 Mbits/sec      0    36.8 KBytes
[ 9] 5.00-6.00      sec  2.82 MBytes        23.6 Mbits/sec      0    35.4 KBytes
[11] 5.00-6.00      sec  2.73 MBytes        22.9 Mbits/sec      0    33.9 KBytes
[SUM] 5.00-6.00      sec  11.2 MBytes        93.9 Mbits/sec      0
-----
[ 5] 6.00-7.00      sec  2.54 MBytes        21.3 Mbits/sec      0    32.5 KBytes
[ 7] 6.00-7.00      sec  2.98 MBytes        25.0 Mbits/sec      0    36.8 KBytes
[ 9] 6.00-7.00      sec  3.00 MBytes        25.2 Mbits/sec      0    35.4 KBytes
[11] 6.00-7.00      sec  2.61 MBytes        21.9 Mbits/sec      0    33.9 KBytes
[SUM] 6.00-7.00      sec  11.1 MBytes        93.4 Mbits/sec      0
-----
[ 5] 7.00-8.00      sec  2.63 MBytes        22.1 Mbits/sec      0    32.5 KBytes
[ 7] 7.00-8.00      sec  3.11 MBytes        26.1 Mbits/sec      0    36.8 KBytes
[ 9] 7.00-8.00      sec  2.82 MBytes        23.6 Mbits/sec      0    35.4 KBytes
[11] 7.00-8.00      sec  2.86 MBytes        24.0 Mbits/sec      0    33.9 KBytes
[SUM] 7.00-8.00      sec  11.4 MBytes        95.7 Mbits/sec      0
-----
[ 5] 8.00-9.00      sec  2.63 MBytes        22.1 Mbits/sec      0    32.5 KBytes
[ 7] 8.00-9.00      sec  2.73 MBytes        22.9 Mbits/sec      0    36.8 KBytes
[ 9] 8.00-9.00      sec  2.82 MBytes        23.6 Mbits/sec      0    35.4 KBytes
[11] 8.00-9.00      sec  2.86 MBytes        24.0 Mbits/sec      0    33.9 KBytes
[SUM] 8.00-9.00      sec  11.0 MBytes        92.6 Mbits/sec      0
-----
[ 5] 9.00-10.00     sec  2.16 MBytes        18.1 Mbits/sec      0    32.5 KBytes
[ 7] 9.00-10.00     sec  3.23 MBytes        27.1 Mbits/sec      0    48.1 KBytes
[ 9] 9.00-10.00     sec  2.91 MBytes        24.4 Mbits/sec      0    35.4 KBytes
[11] 9.00-10.00     sec  3.11 MBytes        26.1 Mbits/sec      0    50.9 KBytes
[SUM] 9.00-10.00     sec  11.4 MBytes        95.7 Mbits/sec      0
-----
[ ID] Interval           Transfer             Bitrate             Retrr  Cwnd
[ 5] 0.00-10.00     sec  26.1 MBytes        21.9 Mbits/sec      0
[ 5] 0.00-10.00     sec  25.9 MBytes        21.7 Mbits/sec      0
[ 7] 0.00-10.00     sec  30.5 MBytes        25.6 Mbits/sec      0
[ 7] 0.00-10.00     sec  30.2 MBytes        25.4 Mbits/sec      0
[ 9] 0.00-10.00     sec  27.7 MBytes        23.3 Mbits/sec      0
[ 9] 0.00-10.00     sec  27.5 MBytes        23.1 Mbits/sec      0
[11] 0.00-10.00     sec  28.8 MBytes        24.2 Mbits/sec      0
[11] 0.00-10.00     sec  28.6 MBytes        24.0 Mbits/sec      0
[SUM] 0.00-10.00     sec  113 MBytes        94.9 Mbits/sec      0
[SUM] 0.00-10.00     sec  112 MBytes        94.2 Mbits/sec      0
iperf Done.
root@T113-GINGKO:~#

```



1.8 AUDIO 音频测试

请使用带麦克风的耳机连接至 GEC100 HP OUT/MIC IN 接口。将产品资料“软件资料\Demo\base-demos\mic_in_test\”目录下的 mic_in_test.sh 脚本程序拷贝至 GEC100 文件系统“/opt/”目录下。进入 GEC100 文件系统，执行如下命令，使用耳机的麦克风进行录音。按“Ctrl + C”终止 录音后，会将录音文件保存为“test.wav”。

Target# /opt/mic_in_test.sh

Command 26

```
root@T113-GINGKO:/opt# cp /mnt/mic_in_test.sh /opt/
root@T113-GINGKO:/opt# cd ~
root@T113-GINGKO:~# ls /opt/mic_in_test.sh
/opt/mic_in_test.sh
root@T113-GINGKO:~# /opt/mic_in_test.sh
Recording WAVE 'test.wav' : [ 2784.419485] [SNDCODEC][sunxi_card_hw_params][620]:stream_flag: 1
Signed 24 bit Little Endian, Rate 16000 Hz, Mono
Hardware PCM card 0 'audiocodec' device 0 subdevice 0
It's setup is:
  stream      : CAPTURE
  access     : RW_INTERLEAVED
  format     : S24_LE
  subformat  : STD
  channels   : 1
  rate       : 16000
  exact_rate : 16000 (16000/1)
  msbits     : 32
  buffer_size : 8000
  period_size : 2000
  period_time : 125000
  tstamp_mode : NONE
  tstamp_type : MONOTONIC
  period_step : 1
  avail_min  : 2000
  period_event : 0
  start_threshold : 1
  stop_threshold : 8000
  silence_threshold: 0
  silence_size : 0
  boundary   : 2097152000
  appl_ptr   : 0
  hw_ptr     : 0
Max peak (2000 samples): 0x01b9f000 ##### 345%
Max peak (2000 samples): 0x01bb0000 ##### 346%
Max peak (2000 samples): 0x01c23000 ##### 351%
Max peak (2000 samples): 0x01c19000 ##### 351%
```

执行如下命令，耳机将播放“test.wav”录音文件，不存在杂音、失真现象则说明功能正常，按“Ctrl + C”停止播放。

Target# aplay -Dhw:audiocodec -vv test.wav

Command 27

```
root@T113-GINGKO:~# aplay -Dhw:audiocodec -vv test.wav
Playing WAVE 'test.wav': [ 2815.900290] [SNDCODEC][sunxi_card_hw_params][620]:stream_flag: 0
Signed 24 bit Little Endian, Rate 16000 Hz, Mono
Hardware PCM card 0 'audiocodec' device 0 subdevice 0
Its setup is:
  stream      : PLAYBACK
  access      : RW_INTERLEAVED
  format      : S24_LE
  subformat   : STD
  channels    : 1
  rate       : 16000
  exact_rate  : 16000 (16000/1)
  msbits     : 32
  buffer_size : 8000
  period_size : 2000
  period_time : 125000
  tstamp_mode : NONE
  tstamp_type : MONOTONIC
  period_step : 1
  avail_min   : 2000
  period_event : 0
  start_threshold : 8000
  stop_threshold : 8000
  silence_threshold: 0
  silence_size : 0
  boundary    : 2097152000
  appl_ptr    : 0
  hw_ptr      : 0
#####| MAX !clip
root@T113-GINGKO:~#
```

1.9 HDMI 测试

1.9.1 显示功能测试

GEC100 Linux 系统默认支持 HDMI OUT 显示，请将 GEC100 HDMI OUT 接口连接至 HDMI 显示屏，GEC100 上电启动，可在 HDMI 显示屏观察到如下显示界面。



1.9.2 显示分辨率修改

如需修改 HDMI 显示分辨率，请参考如下方法修改 U-Boot 和内核设备树，系统目前支持的 HDMI 显示分辨率为：1920x1080@60fps、1366x768@60fps、1280x720@60fps、720x480@ 60fps，其中默认支持的分辨率为 1920x1080@60fps，本次操作将分辨率修改为 1280x720@60fps。



- (1) 请参考“参考资料\其他参考文档”目录下《DMTr1 v13》文档，根据 VESA 标准修改 1280x720@60fps 分辨率对应的时序。

VESA MONITOR TIMING STANDARD

Adopted: 11/17/08
Resolution: 1280 x 720 at 60 Hz (non-interlaced)
EDID ID: DMT ID: 55h; Std. 2 Byte Code: 81h, C0h; CVT 3 Byte Code: n/a
Method: ***** NOT CVT COMPLIANT *****
Per CEA-861 --- 720p (Code 4) Timing Definitions

Detailed Timing Parameters

Timing Name	= 1280 x 720 @ 60Hz;		
Hor Pixels	= 1280;	// Pixels	
Ver Pixels	= 720;	// Lines	
Hor Frequency	= 45.000;	// KHz	= 22.2 usec / line
Ver Frequency	= 60.000;	// Hz	= 16.7 msec / frame
Pixel Clock	= 74.250;	// MHz	= 13.5 nsec ± 0.5%
Character Width	= 1;	// Pixels	= 13.5 nsec
Scan Type	= NONINTERLACED; // H Phase = 3.3 %		
Hor Sync Polarity	= POSITIVE;	// HBlank	= 22.4% of HTotal
Ver Sync Polarity	= POSITIVE;	// VBlank	= 4.0% of VTotal
Hor Total Time	= 22.222;	// (usec)	= 1650 chars = 1650 Pixels
Hor Addr Time	= 17.239;	// (usec)	= 1280 chars = 1280 Pixels
Hor Blank Start	= 17.239;	// (usec)	= 1280 chars = 1280 Pixels
Hor Blank Time	= 4.983;	// (usec)	= 370 chars = 370 Pixels
Hor Sync Start	= 18.721;	// (usec)	= 1390 chars = 1390 Pixels
// H Right Border	= 0.000;	// (usec)	= 0 chars = 0 Pixels
// H Front Porch	= 1.481;	// (usec)	= 110 chars = 110 Pixels
Hor Sync Time	= 0.539;	// (usec)	= 40 chars = 40 Pixels
// H Back Porch	= 2.963;	// (usec)	= 220 chars = 220 Pixels
// H Left Border	= 0.000;	// (usec)	= 0 chars = 0 Pixels
Ver Total Time	= 16.667;	// (msec)	= 750 lines HT - (1.06xHA)
Ver Addr Time	= 16.000;	// (msec)	= 720 lines = 3.95
Ver Blank Start	= 16.000;	// (msec)	= 720 lines
Ver Blank Time	= 0.667;	// (msec)	= 30 lines
Ver Sync Start	= 16.111;	// (msec)	= 725 lines
// V Bottom Border	= 0.000;	// (msec)	= 0 lines
// V Front Porch	= 0.111;	// (msec)	= 5 lines
Ver Sync Time	= 0.111;	// (msec)	= 5 lines
// V Back Porch	= 0.444;	// (msec)	= 20 lines
// V Top Border	= 0.000;	// (msec)	= 0 lines

在 T113-i_v1.0 目录执行如下命令，根据时序要求修改 uboot-board.dts 设备树文件。

Target# vi device/config/chips/t113_i/configs/gec100-emmc/uboot-board.dts

Command 28

```
zh@zh-virtual-machine:~/T113-i/T113-i_v1.0$ vi device/config/chips/t113_i/configs/gec100_emmc/uboot-board.dts
zh@zh-virtual-machine:~/T113-i/T113-i_v1.0$
```

修改内容如下:

```
lcd_x = <1280>;
```

```
lcd_y = <720>;
```

```
lcd_dclk_freq = <74>;
```

```
lcd_hbp = <260>;
```



```
lcd_ht = <1650>;
```

```
lcd_hspw = <40>;
```

```
lcd_vbp = <25>;
```

```
lcd_vt = <750>;
```

```
lcd_vspw = <5>;
```

(2) 修改内核，在 T113-i_v1.0 目录执行如下命令，根据时序要求修改 board.dts 设备树文件。

Target# vi device/config/chips/t113_i/configs/gec100-emmc/board.dts

```
zh@zh-virtual-machine:~/T113-i/T113-i_v1.0$ vi device/config/chips/t113_i/configs/gec100_emmc/board.dts
zh@zh-virtual-machine:~/T113-i/T113-i_v1.0$
```

修改内容如下：

```
lcd_x = <1280>;
```

```
lcd_y = <720>;
```

```
lcd_dclk_freq = <74>;
```

```
lcd_hbp = <260>;
```

```
lcd_ht = <1650>;
```

```
lcd_hspw = <40>;
```

```
lcd_vbp = <25>;
```

```
lcd_vt = <750>;
```

```
lcd_vspw = <5>;
```

(3) 分辨率测试

内核与 uboot-board.dts 设备树文件修改完成后，请参考《LinuxSDK 安装及编译》文档重新编译生成新的内核镜像。打包新镜像并烧固到 GEC100 中，查看 HDMI 显示分辨率如下图所示。

Target# fbset

Command 29

```
root@T113-Gingko:~# fbset
mode "1280x720-60"
# D: 74.003 MHz, H: 44.850 kHz, V: 59.800 Hz
geometry 1280 720 1280 720 32
timings 13513 220 110 20 5 40 5
rgba 8/16,8/8,8/0,8/24
endmode
root@T113-Gingko:~# █
```

为更直观看出分辨率不同，本次测试 bootlogo 做反色处理



1280X720-60



1920X1080-60

2 配置静态网络 IP

GEC100 配备 1 个百兆网口。本小节主要演示如何将网口配置为静态 IP 地址。进入 GEC100 文件系统，执行如下命令打开"/etc/network/interfaces"文件，按照如下方法修改网络配置。



Target# vi /etc/network/interfaces

Command 30

```
root@T113-GINGKO:~# vi /etc/network/interfaces
root@T113-GINGKO:~#
```

修改内容如下:

//根据实际网络环境修改如下内容

```
# interface file auto-generated by buildroot auto lo iface lo inet loopback
```

```
auto eth0
```

```
iface eth0 inet static
```

```
address 192.168.1.110
```

```
netmask 255.255.255.0
```

```
gateway 192.168.1.255
```

```
# interface file auto-generated by buildroot
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.1.110
netmask 255.255.255.0
gateway 192.168.1.255
```

执行如下命令，删除"/etc/resolv.conf"软链接，重新创建 resolv.conf 文件，并在文件中添加如下内容。

Target# rm /etc/resolv.conf

Target# touch /etc/resolv.conf

Target# vi /etc/resolv.conf

Command 31

```
root@T113-GINGKO:~# rm /etc/resolv.conf
root@T113-GINGKO:~# touch /etc/resolv.conf
root@T113-GINGKO:~# vi /etc/resolv.conf
```

添加内容如下:

```
nameserver 8.8.8.8 //DNS 请根据实际情况修改
```



配置完成后，请保存退出，并将 GEC100 断电重启，网络静态 IP 即可生效。

Target# ifconfig

Command 32

```
root@T113-GINGKO:~# ifconfig
docker0  Link encap:Ethernet  Hwaddr 02:42:0A:0C:D7:ED
         inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
         UP BROADCAST MULTICAST  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:0
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0     Link encap:Ethernet  Hwaddr EE:A4:79:72:2F:6F
         inet addr:192.168.1.110  Bcast:0.0.0.0  Mask:255.255.255.0
         inet6 addr: fe80::eca4:79ff:fe72:2f6f/64  Scope:Link
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:33 errors:0 dropped:0 overruns:0 frame:0
         TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:8838 (8.6 KiB)  TX bytes:726 (726.0 B)
         Interrupt:38

lo       Link encap:Local Loopback
         inet addr:127.0.0.1  Mask:255.0.0.0
         inet6 addr: ::1/128  Scope:Host
         UP LOOPBACK RUNNING  MTU:65536  Metric:1
         RX packets:514 errors:0 dropped:0 overruns:0 frame:0
         TX packets:514 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:38028 (37.1 KiB)  TX bytes:38028 (37.1 KiB)
```

3 GEC100 接口测试

3.1 RS485 测试

3.1.1 接口功能

本次测试主要实现 PC 主机通过 RS485 向 GEC100 发送数据，GEC100 再将收到的数据通过 RS485 发送至 PC 主机。本测试包含一个程序：rs485_test

在 GEC100 系统中 RS485 设备节点为"/dev/ttyAS3"目录

```
root@T113-GINGKO:~# ls /dev/ttyAS3
/dev/ttyAS3
root@T113-GINGKO:~#
```

3.1.2 RS485 使用说明

将 GEC100 上端子口 T+/T-，分别连接到 EVC8013(USB 转 RS-232/485/422)的 A+/A- 端口，EVC8013 的 USB 接到 PC 主机上，设备管理器中会分配相应的 COM 端口号。

备注：为方便测试案例本章采用 EVC8013 为本司 USB 转换模块，可自行选择其他转换模块。



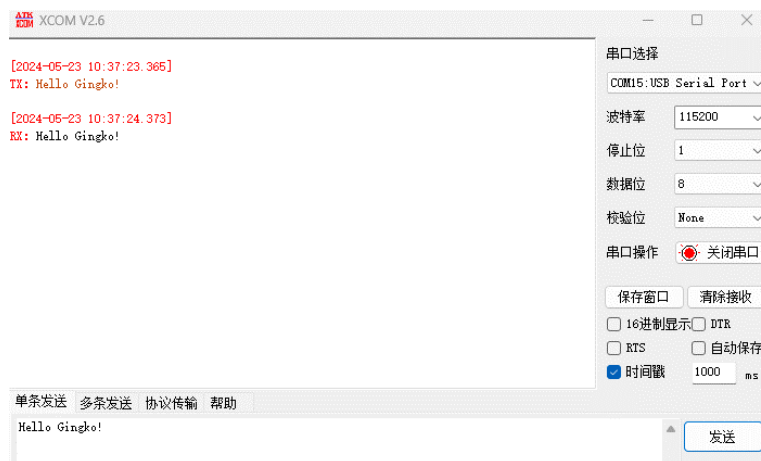
解压“软件资料\XCOM V2.6.rar”，打开 XCOM V2.6.exe 串口调试工具，根据设备管理器中 COM 端口号进行串口选择，并设置波特率 115200、停止位 1、数据位 8、校验位 None，打开串口 如下图所示：



将本案例 bin 目录下将本案例 bin 目录下的可执行程序 rs485_test 拷贝至 GEC100 文件系统，并在可执行程序所在目录执行如下命令运行程序：

```
Target# ./rs485_test
```

程序执行后，在串口调试工具输入字符串“Hello Ginkgo! ”，按下发送即可在 GEC100 显示对应内容，串口调试工具并且能接受到来自 GEC100 发送的“Hello Ginkgo! ”字符串，如下图所示。



```
root@T113-GINGKO:/mnt# ./rs485_test  
Hello Ginkgo!
```

3.1.3 代码截取

- (1) 由于 rs485 为半双工工作模式，所以需要添加收发模式转换开关

```
27 int dir_fd;
28 char dir_off = 0;
29 char dir_on = 1;
30 int u_delay;
31 dir_fd = open("/dev/ged100_485_dir",O_RDWR);
32 if(dir_fd < 0){
33     printf("RS485 DIR init failed %d\n",dir_fd);
34 }
35
36 write(dir_fd,&dir_off,1);
```

- (2) 设置串口波特率、数据位、校验位、停止位。

```
27 //set baudrate
28 tcgetattr(ser_fd, &options);
29 cfsetispeed(&options, B115200);
30 cfsetospeed(&options, B115200);
31 options.c_cflag |= (CLOCAL | CREAD);
32
33 //set databits
34 options.c_cflag &= ~CSIZE;
35 options.c_cflag |= CS8;
36
37 //set crc
38 options.c_cflag &= ~PARENB;
39 options.c_cflag &= ~CSTOPB;
40
41 //set stopbits
42 options.c_cflag &= ~CSTOPB;
43
44 //set fc
45 options.c_cflag &= ~PARENB;
46
47 options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG); /*Input*/
48 options.c_oflag &= ~OPOST; /*Output*/
49
50 options.c_iflag &= ~(BRKINT | ICRNL | INPCK | ISTRIP | IXON);
51
52 options.c_cc[VTIME] = 0;
53 options.c_cc[VMIN] = 0;
54
55 tcflush(ser_fd,TCIFLUSH);
```

- (3) 轮询读取接受到的数据，休眠一段时间转换为写模式，将读取取到的数据发送到 PC 端上。

```
72 while(1){
73     /* rs485 receive*/
74     ret = read(ser_fd,(char *)receive_buffer,(size_t)100);
75
76     if(ret > 0){
77         printf("%s\n",(const char *)receive_buffer);
78         write(dir_fd,&dir_on,1);
79         /* rs485 send */
80         write(ser_fd,receive_buffer,strlen((const char *)receive_buffer));
81         u_delay = 100*strlen((const char *)receive_buffer) + 300;
82         usleep(u_delay);
83         write(dir_fd,&dir_off,1);
84         memset(receive_buffer,0,sizeof(receive_buffer));
85     }
86 }
```

3.2 RS422 测试

3.2.1 接口功能

本次测试主要实现 PC 主机通过 RS422 向 GEC100 发送数据，GEC100 再将收到的数据通过 RS422 发送至 PC 主机。本测试包含一个程序：rs422_test

在 GEC100 系统中 RS422 设备节点为 "/dev/ttyAS3" 目录

```
root@T113-GINGKO:~# ls /dev/ttyAS3  
/dev/ttyAS3  
root@T113-GINGKO:~#
```

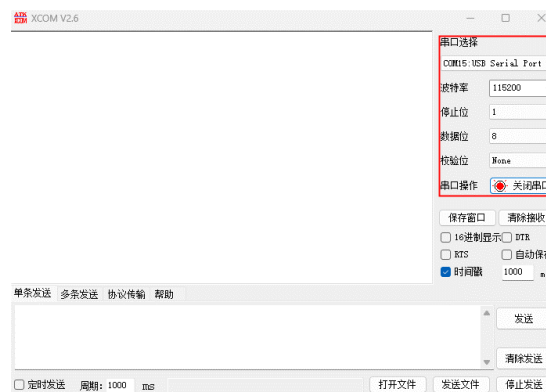
3.2.2 RS422 使用说明

将 GEC100 上端子口 R+/R-、T+/T- 分别连接到 EVC8013(USB 转 RS-232/485/422)的 T+/T-、R+/R- 端口，EVC8013 的 USB 接到 PC 主机上，设备管理器中会分配相应的 COM 端口号。





打开 XCOM V2.6.exe 串口调试工具, 根据设备管理器中 COM 端口号进行串口选择, 并设置波特率 115200、停止位 1、数据位 8、校验位 None, 打开串口 如下图所示:



将本案例 bin 目录下将本案例 bin 目录下的可执行程序 rs422_test 拷贝至 GEC100 文件系统, 并在可执行程序所在目录执行如下命令运行程序:

```
Target# ./rs422_test
```

程序执行后, 在串口调试工具输入字符串"Hello Ginkgo! ", 按下发送即可在 GEC100 显示对应内容, 串口调试工具并且能接受到来自 GEC100 发送的"Hello Ginkgo! "字符串, 如下图所示。



```
root@T113-GINGKO:~# cd /mnt/  
root@T113-GINGKO:/mnt# ./rs422_test  
Hello Ginkgo!
```

3.2.3 代码截取

- (1) 设置串口波特率、数据位、校验位、停止位。

```
27 //set baudrate  
28 tcgetattr(ser_fd, &options);  
29 cfsetispeed(&options, B115200);  
30 cfsetospeed(&options, B115200);  
31 options.c_cflag |= (CLOCAL | CREAD);  
32  
33 //set databits  
34 options.c_cflag &= ~CSIZE;  
35 options.c_cflag |= CS8;  
36  
37 //set crc  
38 options.c_cflag &= ~PARENB;  
39 options.c_cflag &= ~CSTOPB;  
40  
41 //set stopbits  
42 options.c_cflag &= ~CSTOPB;  
43  
44 //set fc  
45 options.c_cflag &= ~PARENB;  
46  
47 options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG); /*Input*/  
48 options.c_oflag &= ~OPOST; /*Output*/  
49  
50 options.c_iflag &= ~(BRKINT | ICRNL | INPCK | ISTRIP | IXON);  
51  
52 options.c_cc[VTIME] = 0;  
53 options.c_cc[VMIN] = 0;  
54  
55 tcflush(ser_fd,TCIFLUSH);
```

- (2) 由于 rs422 为全双工工作模式，接收数据同时也能发送数据，不需要在休眠转换发送模式。

```
61 while(1){
62     /* rs422 receive*/
63     ret = read(ser_fd,(char *)receive_buffer,(size_t)100);
64
65     if(ret > 0){
66         printf("%s\n",(const char *)receive_buffer);
67         /* rs422 send */
68         write(ser_fd,receive_buffer,strlen((const char *)receive_buffer));
69         memset(receive_buffer,0,sizeof(receive_buffer));
70     }
71 }
```

3.3 RS232 测试

3.3.1 接口功能

本次测试主要实现 PC 主机通过 RS232 向 GEC100 发送数据，GEC100 再将收到的数据通过 RS232 发送至 PC 主机。本测试包含一个程序：rs232_test

在 GEC100 系统中 RS232 设备节点为 "/dev/ttyAS4" 目录

```
root@T113-GINGKO:/mnt# ls /dev/ttyAS4
/dev/ttyAS4
root@T113-GINGKO:/mnt#
```

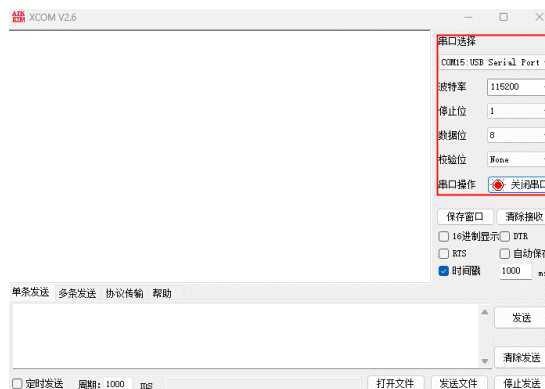
3.3.2 RS232 使用说明

将 GEC100 上 RS232 接口用转接线连接到 PC 主机的 USB 接口，设备管理器中会分配相应的 COM 端口号。



打开 XCOM V2.6.exe 串口调试工具，根据设备管理器中 COM 端口号进行串口选择，并设

置波特率 115200、停止位 1、数据位 8、校验位 None，打开串口 如下图所示：



将本测试用例 bin 目录下将本案例 bin 目录下的可执行程序 rs422_test 拷贝至 GEC100 文件系统，并在可执行程序所在目录执行如下命令运行程序：

```
Target# ./rs232_test
```

程序执行后，在串口调试工具输入字符串"Hello Gingko! "，按下发送即可在 GEC100 显示对应内容，串口调试工具并且能接受到来自 GEC100 发送的"Hello Gingko! "字符串，如下图所示。



```
root@T113-GINGKO:/mnt# ./rs232_test  
Hello Gingko!
```

3.3.3 代码截取

- (1) 设置串口波特率、数据位、校验位、停止位。

```
27 //set baudrate
28 tcgetattr(&ser_fd, &options);
29 cfsetispeed(&options, B115200);
30 cfsetospeed(&options, B115200);
31 options.c_cflag |= (CLOCAL | CREAD);
32
33 //set databits
34 options.c_cflag &= ~CSIZE;
35 options.c_cflag |= CS8;
36
37 //set crc
38 options.c_cflag &= ~PARENB;
39 options.c_cflag &= ~CSTOPB;
40
41 //set stopbits
42 options.c_cflag &= ~CSTOPB;
43
44 //set fc
45 options.c_cflag &= ~PARENB;
46
47 options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG); /*Input*/
48 options.c_oflag &= ~OPOST; /*Output*/
49
50 options.c_iflag &= ~(BRKINT | ICRNL | INPCK | ISTRIP | IXON);
51
52 options.c_cc[VTIME] = 0;
53 options.c_cc[VMIN] = 0;
54
55 tcflush(&ser_fd,TCIFLUSH);
--
```

- (2) rs232 为全双工工作模式，接收数据同时也能发送数据，不需要在休眠转换发送模式。

```
61 while(1){
62     /* rs232 receive*/
63     ret = read(&ser_fd,(char *)receive_buffer,(size_t)100);
64
65     if(ret > 0){
66         printf("%s\n",(const char *)receive_buffer);
67         /* rs232 send */
68         write(&ser_fd,receive_buffer,strlen((const char *)receive_buffer));
69         memset(receive_buffer,0,sizeof(receive_buffer));
70     }
71 }
72
```

3.4 CAN 测试

3.4.1 接口功能

本次测试主要实现 PC 主机通过 USBCAN 适配器向 GEC100 发送数据，GEC100 再将收到的数据通过 CAN 发送至 PC 主机。在“软件资料\USB_CAN TOOL”中有 CAN 调试工具

安装包，安装到自定义目录下，使用调试工具以便获取接收与发送的数据。本测试包含一个程序：can_test

Linux 系统将 CAN 设备作为网络设备进行管理，因此在 CAN 总线应用开发方面，Linux 提供了 SocketCAN 应用编程接口，使得 CAN 总线通信近似于和以太网的通信。

3.4.2 CAN 使用说明

将 GEC100 上 CANH/CANL 接口分别接到 USBCAN 适配器的 H/L 接口，PC 主机的 USB 接口连接适配器的 USB 接口。



在 GEC100 上执行命令：

```
Target# ifconfig -a
```

可以看到，CAN 设备名称为 can0，如下图所示

```
root@T113-GINGKO:/mnt# ifconfig -a
can0    Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
        NOARP MTU:16 Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:10
        RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
        Interrupt:253
```

在 GEC100 上先设置比特率在启动 CAN 设备，执行以下命令：

Target# ip link set can0 type can bitrate 1000000

Target# ip link set can0 up

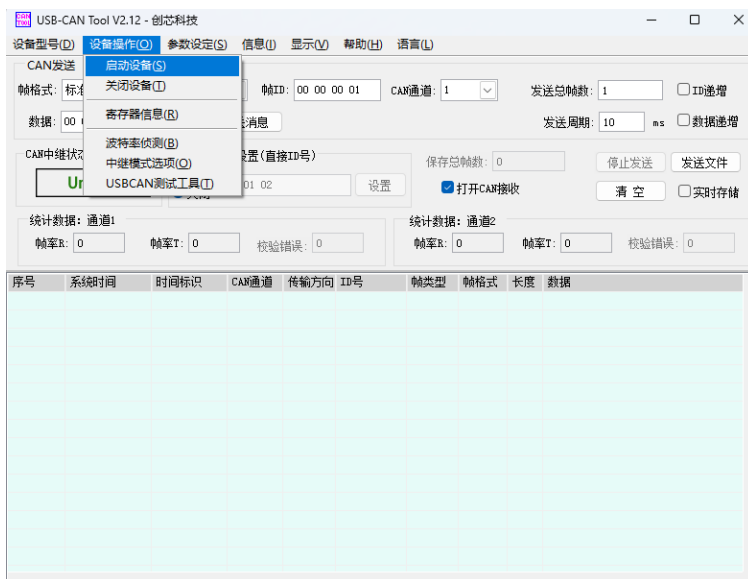
```
root@T113-GINGKO:/mnt# ip link set can0 type can bitrate 1000000
root@T113-GINGKO:/mnt# ip link set can0 up
[ 6814.110243] IPV6: ADDRCONF(NETDEV_CHANGE): can0: link becomes ready
```

如上图所示，CAN0 已启动成功，将本案例 bin 目录下将本案例 bin 目录下的可执行程序 can_test 拷贝至 GEC100 文件系统，并在可执行程序所在目录执行如下命令运行程序：

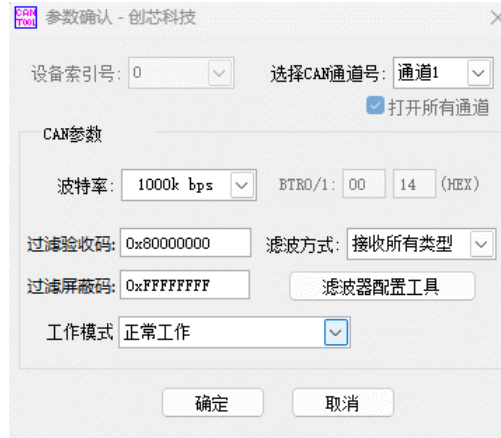
Target# ./can_test

```
root@T113-GINGKO:/mnt# ./can
can_test
```

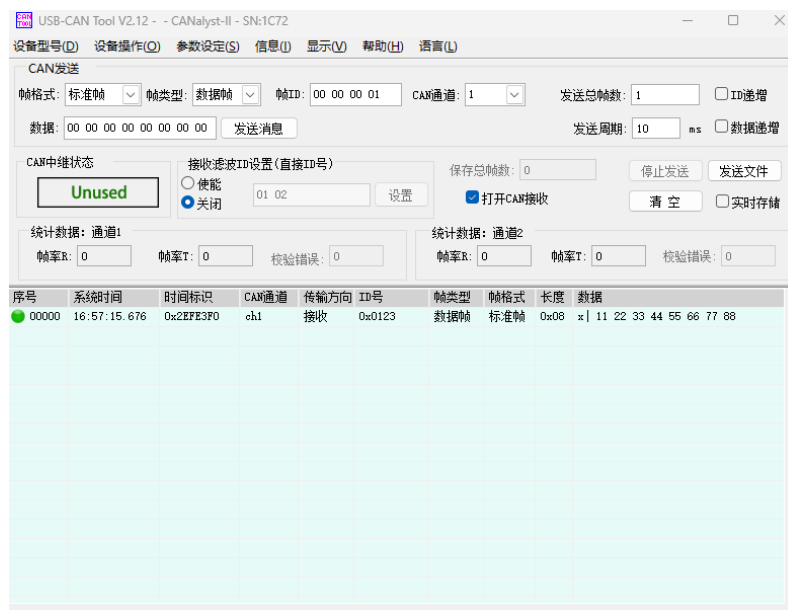
打开 USB-CAN Tool 调试工具，启动设备



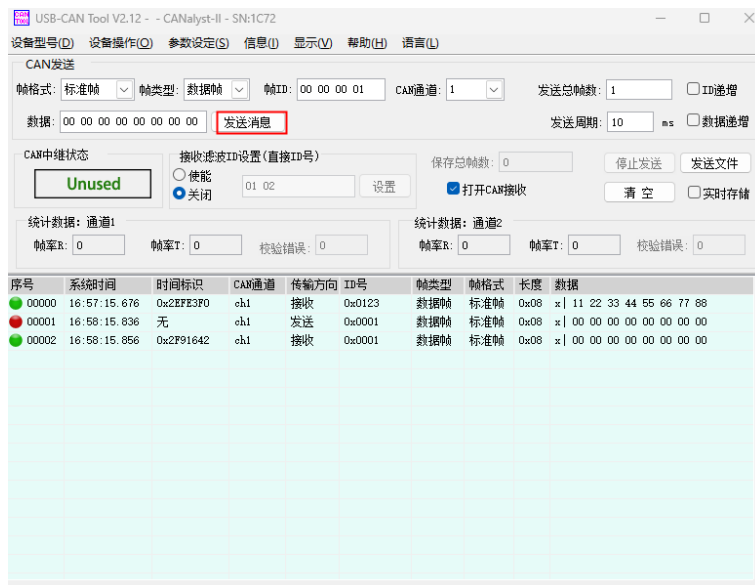
设置波特率为 1000k bps，其他默认选项



点击确定后获取，GEC100CAN 发来的数据



点击发送消息，可以获取 GEC100CAN 读到的消息。



3.4.3 代码截取

注意 struct ifreq 和 struct sockaddr_can 结构体的使用

```
31 //open socket
32 sockfd = socket(PF_CAN,SOCK_RAW,CAN_RAW);
33 if(sockfd < 0){
34     printf("open socket error!\n");
35     goto out;
36 }
37
38 strcpy(ifr.ifr_name,"can0");
39 ioctl(sockfd,SIOCGIFINDEX,&ifr);
40 can_addr.can_family = AF_CAN;
41 can_addr.can_ifindex = ifr.ifr_ifindex;
42
43 //bind
44 ret = bind(sockfd,(struct sockaddr *)&can_addr,sizeof(can_addr));
45 if(ret < 0){
46     printf("bind error!\n");
47     goto out;
48 }
```

3.5 TCP_UDP 测试

3.5.1 接口功能

本次测试主要实现客户端(client)与服务端(server)的文本数据相互收发功能。本测试包含 4 个程序:

- (1) tcp_server: TCP 服务端测试程序。



- (2) tcp_client: TCP 客户端测试程序。
- (3) udp_server: UDP 服务端测试程序。
- (4) udp_client: UDP 客户端测试程序。

服务端和客户端程序均可在 GEC100、PC 机 Ubuntu 系统上运行。本章采用 Ubuntu 系统作为服务端 IP 为 192.168.1.8, GEC100 作为客户端当服务端进行测试, bin 目录下的 4 个文件均为 ARM 端可执行程序。

3.5.2 TCP_UDP 使用说明

请将本测试 src 目录下的 4 个.c 文件拷贝至 GEC100 文件系统任意目录下, GCC 编译服务器文件生成可执行文件, arm-linux-gnueabi-gcc 编译生成可执行文件拷贝到 NFS 共享目录/home/zh/nfsroot 下。在 GEC100 中/mnt 目录下可以看到可执行文件。

```
root@T113-GINGKO:/mnt# ls
tcp_client tcp_server udp_client udp_server
root@T113-GINGKO:/mnt#
```

(1) TCP 通信测试

在可执行程序所在目录执行如下命令运行 TCP 服务端和客户端程序。5000 为服务端程序指定的端口号, 客户端程序端口号参数需与服务端程序一致。

Target#	<code>./tcp_server 5000</code>	//TCP 服务端命令
Host#	<code>./tcp_client 192.168.1.8 5000</code>	//TCP 客户端命令

程序执行后, 客户端将会连接服务端。在服务端 (GEC100) 输入字符串"Gingko", 按下回车键即可在客户端显示对应内容。在客户端(Ubuntu)输入字符串"Hello Gingko! ", 按下回车键即可在服务端显示对应内容, 如下图所示。



```
root@T113-GINGKO:/mnt# ./tcp_client 192.168.1.8 5000
connect server(IP:192.168.1.8 Port: 5000).
Gingko
[Recv from server 192.168.1.8:5000] : hello gingko!
```

```
zh@zh-virtual-machine:~/doc/tcp_udp/src$ ./tcp_server 5000
Waiting for client connection...
Client accepted : 192.168.1.110:36522
[Recv from client 0.0.0.0:5000] : Gingko
hello Gingko!
```

(2) UDP 通信测试

在可执行程序所在目录执行如下命令运行 TCP 服务端和客户端程序。5000 为服务端程序指定的端口号，客户端程序端口号参数需与服务端程序一致。

Target# ./udp_server 5000 //UDP 服务端命令

Host# ./udp_client 192.168.1.8 5000 //UDP 客户端命令

程序执行后，客户端将会连接服务端。在服务端（GEC100）输入字符串"Gingko"，按下回车键即可在客户端显示对应内容。在客户端(Ubuntu)输入字符串"Hello Gingko! "，按下回车键即可在服务端显示对应内容，如下图所示。

```
zh@zh-virtual-machine:~/doc/tcp_udp/src$ ./udp_server 5000
Waiting for client connection
[Recv from client 192.168.1.110:36487] : hello Gingko!
```

```
root@T113-GINGKO:/mnt# ./udp_client 192.168.1.8 5000
hello Gingko!
```

3.5.3 代码截取

以 TCP 通信程序为例。

(1) tcp_client.c

注意源码中的 struct sockaddr_in、socket、connect、fgets、send、recv 等数据结构和系统调用的使用。

(2) tcp_server.c

注意源码中的 struct sockaddr_in、socket、connect、fgets、send、recv、bind、listen、accept 等数据结构和系统调用的使用。

3.6 AO/AI 测试

3.6.1 接口功能

本次测试主要实现 GEC100 模拟信号的输入与输出, 通过测试程序 `dac_test` 控制 GEC100 产生一个 2.5v (默认) 的模拟信号, 由 AI 接口接收并打印出电压值。本测试包含一个程序:
`dac_test`

3.6.2 AO/AI 测试说明

将 GEC100 的 AO0、AO1、AO2、AO3 与 AI0、AI1、AI2、AI3 连接如下图所示:



在 GEC100 系统中 AI 设备节点为" /sys/bus/iio/devices/iio\:device0/in_voltageX_raw"目录, AO 设备节点为"/dev/dacdevX"

```
root@T113-GINGKO:~# ls /sys/bus/iio/devices/iio\:device0
dev                in_voltage3_raw   in_voltage7_raw
in_voltage0_raw    in_voltage4_raw   in_voltage_scale
in_voltage1_raw    in_voltage5_raw   name
in_voltage2_raw    in_voltage6_raw   of_node
power              subsystem
uevent
waiting_for_supplier
```

```
root@T113-GINGKO:~# ls /dev/dac*  
/dev/dacdev0 /dev/dacdev1  
root@T113-GINGKO:~#
```

将本案例 bin 目录下将本案例 bin 目录下的可执行程序 dac_test 拷贝至 GEC100 文件系统，并在可执行程序所在目录执行如下命令运行程序：

```
Host# ./dac_test  
  
root@T113-GINGKO:/mnt# ./dac_test  
[ 7017.002196] dac_open -- 36.  
[ 7017.005399] dac_open -- 36.  
[ 7017.008542] dac_write -- 61.  
[ 7017.012021] dac_write -- 61.  
[ 7017.015413] dac_write -- 61.  
[ 7017.018790] dac_write -- 61.  
[ 7017.022150] dac_write -- 61.  
[ 7017.025608] dac_write -- 61.  
dac1_fd: 4  
2.44V  
2.59V  
2.49V  
[ 7017.030548] dac_release -- 43.  
2.42V  
[ 7017.035834] dac_release -- 43.
```

如上图所示，AI 四个接口的模拟信号电压在 2.5V 左右，与默认输出 AO 模拟信号电压一致。

3.6.3 代码截取

- (1) 设置 AO 值，vout0、vout1

```
17 int dac0_fd;  
18 int dac1_fd;  
19 int ad_fd[3];  
20 float vol;  
21 unsigned char write_buffer[20] = {0};  
22 unsigned char read_buffer[20] = {0};  
23 float vout0 = 2.5;  
24 float vout1 = 2.5;  
25 unsigned short int data = 0;
```

- (2) 打印 AI 值

```
74 ad_fd[0] = open("/sys/bus/iio/devices/iio:device0/in_voltage0_raw", O_RDONLY);  
75 read(ad_fd[0], read_buffer, 10);  
76 close(ad_fd[0]);  
77 vol = atoi(read_buffer);  
78 vol = vol/4096.*5.0;  
79 vol = vol*2.502487;  
80 printf("%.2fV\n", vol);
```

3.7 DO/DI 测试

3.7.1 接口功能

本次测试主要实现 GEC100 数字信号的输入与输出，通过应用空间控制 GPIO。在 GEC100 系统目录/sys/class/gpio/下有 export 文件，向 export 文件写入要操作的 GPIO 号，使得该 GPIO 的操作接口从内核空间暴露到用户空间，GPIO 的操作接口包括 direction 和 value 等，direction 控制 GPIO 输入或者输出模式，而 value 可控制 GPIO 的状态或者读取状态。

3.7.2 DO/DI 测试说明

(1) DO 接口供电

将 GEC100 端子口 DOPWR/DOGND 接一个外部电源,本次测试采用 12V 电源。如下图所示：





(2) 测试 DO 接口

说明：由原理图可知 GEC100 四个 DO 接口对应的 GPIO 编号如下所示

DO0-PWM4: PG5-197
DO1-PWM5: PF6-166
DO2-PWM6: PG1-193
DO3-PWM7: PD22-118

注：GEC100 的 GPIO 编号计算方式，例 PB13 = 32 x 1(PA) + 13 = 45

在 GEC100 系统目录“/sys/class/gpio”下注册 DO0 的 GPIO 编号，如下所示：

Target# cd /sys/class/gpio

Target# echo 197 > export

```
root@T113-GINGKO:~# cd /sys/class/gpio/  
root@T113-GINGKO:/sys/class/gpio# echo 197 > export  
root@T113-GINGKO:/sys/class/gpio# ls  
export      gpio197     gpiochip0   unexport  
root@T113-GINGKO:/sys/class/gpio#
```

进入到 gpio197 目录下，将 GPIO197 设置为输出模式，并将 value 设置为 1 高电平模式，DO0 的 LED 等就会亮起，测量 DO0 接口电压为 12V。

Target# cd /sys/class/gpio/gpio197

Target# echo out > direction

Target# echo 1 > value

```
root@T113-GINGKO:/sys/class/gpio# cd gpio197  
root@T113-GINGKO:/sys/class/gpio/gpio197# echo out > direction  
root@T113-GINGKO:/sys/class/gpio/gpio197# echo 1 > value  
root@T113-GINGKO:/sys/class/gpio/gpio197#
```

向 value 输入 0，LED 灭掉，输出电压为 0。

```
root@T113-GINGKO:/sys/class/gpio/gpio197# echo 0 > value  
root@T113-GINGKO:/sys/class/gpio/gpio197#
```

同理，DO1、DO2、DO3 与 DO0 步骤一致，注意修改 GPIO 编号。

(3) DI 接口接线

将外部电源正极接到 DI 接口，负极接到 DIGND，如下图所示



接入电源后，对应的 DI 的 LED 灯会亮起。

(4) 测试 DI 接口

说明：由原理图可知 GEC100 四个 DI 接口对应的 GPIO 编号如下所示

DI0-PWM0: PG12-204

DI1-PWM1: PG6-198



DI2-PWM2: PG13-205

DI3-PWM3: PG10-202

注: GEC100 的 GPIO 编号计算方式, 例 PB13 = 32 x 1(PA) + 13 = 45

在 GEC100 系统目录“/sys/class/gpio”下注册 DI0 的 GPIO 编号, 如下所示:

Target# cd /sys/class/gpio

Target# echo 204 > export

```
root@T113-GINGKO:~# cd /sys/class/gpio
root@T113-GINGKO:/sys/class/gpio# echo 204 > export
root@T113-GINGKO:/sys/class/gpio# ls
export      gpio204    gpiochip0  unexport
root@T113-GINGKO:/sys/class/gpio#
```

进入到 gpio197 目录下, 将 GPIO204 设置为输入模式, 并用 cat 命令查看 value 状态, value 的 0/1 即反应了 DI0 的高低电平。

Target# cd /sys/class/gpio/gpio204

Target# echo in > direction

Target# cat value

```
root@T113-GINGKO:/sys/class/gpio# cd gpio204
root@T113-GINGKO:/sys/class/gpio/gpio204# echo in > direction
root@T113-GINGKO:/sys/class/gpio/gpio204# cat value
1
root@T113-GINGKO:/sys/class/gpio/gpio204#
```

将电源拔掉, DI0 的 LED 灯灭掉, value 值变为 0

Target# cat value

```
root@T113-GINGKO:/sys/class/gpio# echo 204 > export
root@T113-GINGKO:/sys/class/gpio# cd gpio204
root@T113-GINGKO:/sys/class/gpio/gpio204# echo in > direction
root@T113-GINGKO:/sys/class/gpio/gpio204# cat value
1
root@T113-GINGKO:/sys/class/gpio/gpio204# cat value
0
root@T113-GINGKO:/sys/class/gpio/gpio204#
```

同理, DI1、DI2、DI3 接口测试步骤与 DI0 相同。