

# CDMA 手机开发指南

Rev	Date	Description	Author
1.0	2009/05/18	Create	Liqing Huang



## 目录

1. CDMA 模块 AT 与 GSM AT 的联系与区别.....	4
2. 电话.....	4
3. 飞行模式、开关机.....	5
3.1 开机.....	5
3.2 关机.....	6
3.3 飞行模式.....	6
4. 拨打国际号码.....	6
5. 呼叫等待、呼叫转移和三方通话.....	6
5.1 呼叫等待与呼叫转移的设置.....	6
5.2 开启呼叫等待功能后第三方来电的接听与拒接.....	7
6. PIN 码管理.....	7
7. 短信(SMS).....	7
7.1 Comparing GSM SMS and CDMA SMS.....	7
7.2 Common GSM/CDMA message fields.....	9
7.3 CDMA-specific message fields.....	9
7.4 发送以+86 开头的电话号码.....	10
7.5 发送 PDU 格式.....	10
7.6 长短信的发送.....	10
7.7 短信状态的改变.....	10
7.8 短信中心.....	10
7.9 状态报告.....	10
8. 运营商名称的显示.....	10
9. 长短信.....	11
10. 批量复制、删除短信、电话本.....	11
11. 数据上网.....	11
9.1 WAP 方式上网.....	11
9.2 Modem 拨号上网: .....	12
12. 开启 MUX 功能说明.....	14
13. GSM 支持, CDMA 不支持的.....	14
14. 其他 CDMA 特性.....	14
15. VIA CDMA CP 编译.....	14
14.1 相关编译工具的安装: .....	14
14.2 cp 编译.....	15
14.3 模拟器 coyote 编译.....	15
16. VIA CDMA 代码目录结构.....	16
17. 升级模组软件.....	20
18. 参考文档.....	20
附录一 使用 Ruby 在 windows 下通过串口自动测试 AT.....	20



## 文档说明

文档撰写目的:

经过了很长一段时间,从事基于威盛 VIA CDMA AT 开发,在 Broncho 平台上写 CDMA 模组编解码,在 MTK 上部署 CDMA(双网 G+C),对 CDMA 与 GSM 有一定程度上的理解,故整理成文档,方便查阅,也当作团队离线交流,知识传递。文中所提到的 CDMA AT 均是广州信位(Simware)提供的,其他厂家 AT 请另行参考。现在 VIA CDMA 的很多 AT 已符合中国电信新规范,关于这方面的变化,再令整理文档。

### 1. CDMA 模块 AT 与 GSM AT 的联系与区别

就列举几个比较显眼的差别,很多小细节还是非常多的,具体请参阅厂家的 AT SPEC。

(1) Action 类型的 AT,比如信号查询,GSM 是 **AT+CSQ** 而 VIA CDMA 则是 **AT+CSQ?** 也就是在后面加上一个“?”号。

(2) CDMA 中"UIM"的字符串,这个主要出现在电话本与短信。GSM 是"SIM"对应 0,"ME"对应 1;CDMA 把"UIM"对应为 0 即可。

(3) 电话本,短信,GSM 记录索引从 1 开始,而 VIA CDMA 从 0 开始。

(4) TE 特征字符串设定

GSM: **AT+CSCS**

CDMA 没有这方面的设定,默认为 UNICODE,在电话本处理中,AT 请求和响应要特别指出。

以前开发过 GSM 相关产品,出于习惯,易将 VIA 模块 AT 命令集与 GSM 的做比较,指出 VIA 模块缺少某些命令的支持。相比 GSM,CDMA 标准化的 AT 命令不多,但 CDMA 与 GSM 网络毕竟存在很多差别,所以具体命令还是需要具体分析,不可盲目类比。发现问题时,最好拿参考机做对比测试,有很多问题其实是 CDMA 固有特性或是网络原因,同等环境下的参考机对比测试往往能省去不少开发时间。

### 2. 电话

在 CDMA 网络中,拨打电话时,网络端不会发送通话接通信令,只会发业务信道建立信令,所以终端是无法准确得知对方何时接通电话。

电话状态查询 CLCC,CDMA 提供的功能很弱,不管有没有来电,拨出电话,查询 CLCC 总有一个+CLCC:响应,GSM 则不然。CDMA 处于三方通话,呼叫保持,呼叫等待时,查询 CLCC 也是只有一路+CLCC:响应,这样,如果根据需要,要使得 CDMA 适配于 GSM,我们得自己构建适应于 GSM 的 CLCC,维护,更新 CLCC 列表。

CDMA (**AT+CLCC?**)状态表:

<state>	<mode>	<termination>	Notes
2	0	2	拨电话中,未连上.很快,看不到
1	0	2	拨电话中,已连上,然后有 +CCNT:3 上报
1	0	2	拨通后
1	0	2	呼叫保持,一直都是
1	0	2	三方通话,一直如此
0	9	0	正常状态(Ready)
3	0	1	来电时,未接听。有 RING +CLIP:xxx 上报

1	0	1	接听电话, 然后有 +CANS 上报
2	9	2	拨号上网,未接通前

注意: CDMA 只有一行 +CLCC: 无论时候有无来电。

GSM 与 CDMA CLCC 对比表:

Stat	GSM	CDMA
0	active	no call
1	held	traffic
2	dialing (MO call)	originate wait
3	alerting (MO call)	alert wait
4	Incoming (MT call)	no service
5	waiting (MT call)	state_num
<b>termination</b>		
<b>0</b>	originated (MO) call	unknown or not applicable
<b>1</b>	terminated (MT) call	mobile terminated (MT) call
<b>2</b>		mobile originated (MO) call

### 3. 飞行模式、开关机

#### 3.1 开机

一般情况下, 模块加电后会自行开启协议栈。不插 UIM 卡启动, 模块不会自行打开协议栈。需要输入 PIN 的情况下, 只有在用户输入正确的 PIN, 模块才会开启协议栈。在以上任何情形下开机, 模块都会上报 (**+VPUP**), 可以根据此上报来判断正常使用中的模块是否重启。

开机流程一般如下:

**+VPUP**

收到该命令后进行初始化设置。(模组有问题时, 这个上报是不可靠的.)

**+MSSStatus:0**

模块上报协议栈已打开。

**+VROM:1**

模块上报漫游状态 (1 为非漫游状态)。这个上报是不可靠的, 就是说, 并不是每次开机都会有这个上报。不过, 如果出现这个, 我们认为可以显示中国电信了, 其他手机已开机就显示运营商, 估计也是照这个来做。

**AT+ISF?**

查询模块初始化状态, 2s~3s 执行一次

**+ISF:1**

模块初始化完成。

接下来才可以对电话本, 短信, 通话记录初始化。否则操作AT都是ERROR。

由上, 在**+MSSStatus:0** 与 **+ISF:1** 之间, 可以操作的行为有:

- (0) **ATE0** 软件已经默认关闭回显, 该步骤可省。
- (1) 注册网络与信号上报. **AT+CREG=2 AT+ARSI=1**
- (2) 查询UIM是否插好? **AT+CPIN?**

- (3) 查询信号 *AT+CSQ?*
- (4) 查询国家码, 网络码 *AT+VMCC? AT+VMNC?*
- (5) 选择语音通道 *AT+SPEAKER=0*

开机时, 要初始化的AT有:

ATE0

ATV1

ATS0

AT+ARSI=1

AT+CREG=2

AT+CMGF=1

AT+CLIP=1

AT+CCWA=1

AT+SPEAKER=1

AT+SMICMODE=1

AT+VGR=8

AT+VGT=7

AT+CRSL=8

AT+MGH=1 (simware 长短信的支持)

AT+CNMI=1,1

AT+LCT

### 3.2 关机

由于关机需要在网络上进行一些登记操作, 因此正常的关机步骤是建议+CPOF 关闭协议栈后延迟一段时间断电。

### 3.3 飞行模式

模块实现飞行模式可以直接使用+CPOF 关闭协议栈, +CPON 打开协议栈返回正常模式。

查询当前是否在飞行模式: +VPON.

GSM 的, 则根据 AT+CFUN 来判断飞行模式, 协议栈的关闭, 打开。

## 4. 拨打国际号码

和 GSM 网络不一样, 全球的 CDMA 网络在这方面不太统一, 印度网络可以识别 “+” 号, 但中国电信网络不能识别。使用 *AT+CDV* 拨打国际号码前, UI 需将号码前的 “+” 转换或过滤掉, 比如可以将 “+86” 转换成 “0086”, 具体需要根据当地运营商的规定。

## 5. 呼叫等待、呼叫转移和三方通话

### 5.1 呼叫等待与呼叫转移的设置

CDMA 中呼叫等待与呼叫转移的设置类似于拨打某个特殊号码的语音电话, 其激活或取消操作是否成功也要根据网络端的语音提示方能知晓。

GSM 提供了 CCWA 和 CCFC 两个命令用于进行呼叫等待与呼叫转移相关设置。

中国电信网络中, 呼叫等待的号码是 \*74/\*740, 客户的 UI 显示可以类似拨打语音电话, 接通以后显示类似通话中的界面, 等用户自己挂断。(此处与 GSM 有差异)。与此类似, 呼叫转移对应号码如下:

- 无条件呼叫转移激活号码是\*72
- 无条件呼叫转移去激活号码是\*720
- 遇忙呼叫转移激活号码是\*90
- 遇忙呼叫转移去激活号码是\*900
- 无应答呼叫转移激活号码是\*92
- 无应答呼叫转移去激活号码是\*920
- 默认呼叫转移激活号码是\*68
- 默认呼叫转移去激活号码是\*680
- 取消所有呼叫转移设置号码是\*730

CDMA 呼叫限制/呼叫等待相关业务的开通/关闭需要拨特定的号码，是需要靠语音来提示相关操作是否成功，终端无法知道当前是否操作成功，也无法获得当前状态。

终端界面上：

CDMA：进入呼叫转移，设置“开启”或“取消”时为进入呼叫界面，给出语音提示；且不支持“查询”菜单；

GSM：进入呼叫转移，设置“开启”或“取消”时显示等待动画，给出文字提示；

## 5.2 开启呼叫等待功能后第三方来电的接听与拒接

CDMA 开启呼叫等待功能后，通话中，若接到第三方呼叫，则使用+CFSH 接听第三方来电，并可使用+CFSH 来回切换两路通话。另外，通话中有第三方来电，若用户不想接听，则只能忽视来电提示，无法拒接。

### (3) 三方通话

CDMA 首先确认模块上的 UIM 卡已经开通三方通话功能（由网络运营商开通）。在已经建立一路通话后，发送 AT+CFSH 将当前通话 hold，再发送 AT+CFSH=<第三方号码>进行呼叫。当第三方接听后，发送 AT+CFSH，即可实现三方通话。模块可发送 AT+CFSH 将第三方挂断，或 CHV 挂断所有通话。注意：CDMA 三方通话和呼叫转移，需要用 send 键开启或者切换，无法知道当前和哪一方在通话。

## 6. PIN 码管理

UIM 卡启用 PIN 码后，模块每次开启，需要先输入 PIN 码校验，否则不会自动打开协议栈。

## 7. 短信

### 7.1 Comparing GSM SMS and CDMA SMS

Area	GSM	CDMA
Message types	GSM SMS defines six types (Protocol Data Unit (PDU) types) of SMS message: <ul style="list-style-type: none"> <li>• Deliver: a message sent from the service center (SC) to the phone.</li> </ul>	CDMA also defines six message types. Like GSM, it has Deliver and Submit types. The other types differ from GSM: <ul style="list-style-type: none"> <li>• Deliver: message sent to the phone</li> </ul>

	<ul style="list-style-type: none"> <li>• Submit: a message sent from the phone to the SC.</li> <li>• Deliver report: sent from the phone to inform the SC of a failure in a Delivered message.</li> <li>• Submit report: sent from the SC to inform the phone of a failure in a Submitted message.</li> <li>• Status report: sent from the SC to inform the phone of the status of a message previously submitted by the phone.</li> <li>• Command: sent from the phone to invoke an operation at the SC.</li> </ul>	<ul style="list-style-type: none"> <li>• Submit: message sent from the phone.</li> <li>• Cancellation: from the phone to the Message Center (MC) to cancel a submitted message.</li> <li>• User Acknowledgment: sent from a phone that has received a message to the submitter to inform it that the user has acknowledged the message.</li> <li>• Delivery Acknowledgment: sent from the MC to inform the phone of the status of a submitted message.</li> <li>• Read Acknowledgment Message: sent from a phone that has received a message to the submitter to inform it that the user has opened the message.</li> </ul>
Application layer	<p>A message can have a Protocol-Identifier field to specify a higher layer protocol being used. The standard does not define the operation of such protocols.</p>	<p>A message can have a Teleservice identifier to specify the application. The standard defines the operation of six teleservices.</p> <p>The Wireless Messaging Teleservice (WMT) is for short text messaging between users. The Wireless Enhanced Messaging Teleservice (WEMT) extends this to include EMS elements such as pictures.</p> <p>Other teleservices defined are: IS-91 Extended Protocol Enhanced Services, Wireless Paging Teleservice (WPT), Voice Mail Notification (VMN), and Service Category Programming Teleservice (SCPT).</p>

		Not all teleservices use all message types: for example, WPT does not support message cancellation, so does not use the Cancellation message type.
Enhanced messaging	The Enhanced Messaging Service (EMS) standard carries pictures, animations, sounds, and rich text in SMS messages.	Wireless Enhanced Messaging Teleservice (WEMT) carries EMS over CDMA SMS.
Character encoding	Alphanumeric text is encoded as 7-bit characters (GSM 03.38).	The encodings used can be specified in the message. Possible encoding types are listed in [3GPP2 C.R1001-C Table 9.1-1].
Point-to-Point\Broadcast	GSM SMS offers a one-to-one and one-to-a-few service.	In addition to a Point-to-Point service, CDMA SMS offers a broadcast service for sending a message to all phones in a defined area.

### **7.2 Common GSM/CDMA message fields**

The fields include:

- Character set for SMS messages
- Originating address
- Message Service Center Time Stamp
- Validity period

### **7.3 CDMA-specific message fields**

The fields are:

- Teleservice
- Originating Subaddress
- Deferred Delivery Time
- Priority
- Privacy
- Number of Messages
- Callback Number
- Message Display Mode
- Message Deposit Index
- User Response Code
- Alert On Message Delivery
- Language Indicator

由上面看到，GSM 与 CDMA 的 pdu 格式完全不一样，具体的类型含义，CDMA 请参阅 3GPP2 上的 C.S0015-A\_v1.0\_111403。

#### 7.4 发送以+86开头的电话号码

CDMA 网络不能识别“+”号，有些地方的网络也不支持“0086”的发送方式，发短信时请先去除“+86”再发送。

#### 7.5 发送PDU格式

(1) GSM:

```
AT+CMGS=<length>\r
PDU can be entered. <CTRL-Z>
```

(2) CDMA:

```
AT+CMGS=<number>,<PDU strings>\r
或: AT+CMGS=,<PDU strings>\r
```

因为 PDU 字符串中，已包含有号码项，故前面的号码可以省略。

#### 7.6 长短信的发送

CDMA 不能无缝的发送长短信，一般要等到 +CDS:上报才能发出下一条,一般情况下，发出一条短信过 4-5 秒后就有+CDS 上报，不过，如果发给自己的话，这个时间会更长，我们采用延时 30 秒。关于 CDMA 长短信的编解码，请参阅《中国电信 CDMA 终端需求规范-SMS 分册》。GSM 连续发多条短信时，需要用指令 AT+CMMS 设置。

#### 7.7 短信状态的改变

CDMA 用 CMGR 读取一条短信后，不会自动改变该属性为“已读”，需要用 CMSM 指令进行更改。

#### 7.8 短信中心

CDMA 不需要设置短信中心。GSM 用 *AT+CSCA* 设置。

#### 7.9 状态报告

CDMA 用+CMGS 发出一条短信后，有+CDS: 上报消息，与 GSM 不一样，+CDS 是网络回复报告，代表服务中心已经收到该短信，而不是对方的接收报告。

### 8. 运营商名称的显示

(1) CDMA 从 UIM 中获取运营商名称：利用 *AT+CSPN?* 获取，+CSPN 读取的是 UIM 卡中定义运营商名称的文件。AT 只返回 UIM 中的数据，并不对其进行解码，UI 可以根据需要显示相应内容。命令结果中的编码类型和语言类型可以参考 C.R1001 标准 (Administration of Parameter Value Assignments for cdma2000 Spread Spectrum Standards) 的 9.1 Data Field Encoding Assignments 与 9.2 Language Indicator Value Assignments。

(2) CDMA 根据网络下发的 MCC 和 MNC：从网络中获取 MCC 与 MNC 后，可在预置于终端中的 MCC/MNC 与运营商名称对应列表查找，并显示对应的运营商名称。VIA 模块提供了主动上报命令 *+VMCCMNC* 与查询命令 *+VMCCMNC?* 来获取网络中的 MCC 与 MNC。

(3) CDMA 搜索网络是根据 PRL 来搜索。

(4) GSM 可以根据 *AT+COPN* 获取网络运营商，*AT+COPS* 显示当前所有可用的网络。CDMA

则没有这方面的功能。

注意：如果开启UIM卡的Pin功能，该命令在输入PIN码前，读取的是模块NAM变参数值。pin码校验后，读取的是UIM卡的参数值。做开机初始化，当输入pin码后，马上查询CIMI号，查询的结果是：**+CIMI:11111111000**  
查询国家码，网络码也是如此。

**+VMCC:111**

**+VMNC:11**

解锁后 1-2 秒再查询，才能得到正确的值。

## 9. 长短信

### 10. 批量复制、删除短信、电话本

(1) 批量复制短信和电话本，若涉及到对 UIM 卡的操作，主控端程序在调用+CPBW 或+CMGW 等命令时要注意控制，以便让 UIM 有足够的时间执行操作（AT 命令返回的 OK 并不能表示 UIM 操作已经完成）。

从 UIM 卡到 FLASH 复制(主控方每次调用 AT 命令写，之间要间隔 300ms)

从 FLASH 到 UIM 卡复制(主控方每次调用 AT 命令写，之间要间隔 500ms)

(2) 批量删除短信与电话本时，只需要调用一次命令，但在某个时刻有没有删除完成，需要使用相关 AT 命令检测，统计当前还剩余的条数来确认是否删除完成。当全部删除时，就算是统计当前剩余的条数为 0，立刻关机，则是没有全部删除，开机后，还会有电话本记录没有删除掉，需要延时 1-2 分钟。

## 11. 数据上网

上网有两种方式，一是通过 AT 命令（此时 TCP/IP 运行在 CP 上）；一种是将 Module 当 Modem，拨号上网（此时 TCP/IP 运行在 PC 等）。

### 9.1 WAP方式上网

1) **AT+VACCNT="card","card"**

设置帐户名和密码（需从运营商处获取）

2) **AT+VPPPOPEN**

建立 PPP 连接，收到**+VPPPSTATUS:0**说明建立成功

3) **AT+VPPPCLOSE**

此命令用于关闭 PPP 连接

4) **AT+VTCPOPEN=1,"61.172.201.194",80**

创建一个 TCP 的 socket，1 是 socketnum，取值范围 1-8，后面是 IP 地址，例子是 www.sina.com.cn 的，80 是端口号

Socket 若创建成功会收到**+VTCPOPEN:1,0**

5) **AT+VUDPOPEN=2,"61.172.201.194",1600**

UDP 的，socketnum 和 TCP 是公用的，不能重复

创建 socket 之后，可以用 **+VTCSEND**、**+VUDPSEND** 命令发送 TCP，UDP 数据包。

## 9.2 Modem 拨号上网：

首先，确定模组是带数据业务的，UIM 卡要开通上网功能。

Windows 下测试：

- 1)正确的硬件连接，32CLK 等 pin 脚、usb vbus 拉低、等。
- 2)安装相应硬件驱动：如果是串口，安装“CbpSimpleIp”(实际上，可以用 windows 自带的 33.6K modem 驱动，在“advance”中输“**at+crm=1;+cmux=1;+cps=33;+cta=0**”初始化命令)，最后选择 SimpIp；若用 USB 口，安装“ViaUsb\_Setup1.x.x”。
- 3)此时应该可以在“设备管理器”中看到相关硬件：if 串口，找到“VIA Telecom CBP SimpIp”；若用 USB 口，找到“VIA Telecom CBP USB Modem”。
- 4)在 PC 上创建拨号连接：新建连接向导->工作场所->拨号连接。联通默认号码、用户名、密码分别是#777、card、card。
- 5)拨号前确保 UIM 卡开通数据业务、可以打电话。
- 6)相应的拨号 log，可以在“设备管理器”->属性->诊断 里看到。

此外，还可以安装Simware的《C200 CDMA2000无线商务终端》软件。

补充，如果提示692错误号，请在调职解调器 -> 属性 -> 硬件流控 前面的勾去掉。

AT Commands in Dialup network

+CRM 0=Async, 1=Relay Model, 2=Network Model

+CMUX 1=Rate set1,2=Rate set 2

+CPS 33=High Speed Packet Data 7 or 15=Low Speed Packet Data

+CTA 0 Traffic Channel not released during inactivity periods(default)

1-255 Release the Traffic Channel after <value> 1-second intervals have elapsed since last sending or receiving RLP data frames on the Um interface.

Linux 下测试，以Fedora Core 7为例：

```
# gedit /etc/ppp/peers
user card
password card
115200
/dev/ttyUSB0
connect "chat -v -f /etc/ppp/cdma-connect"
debug
kdebug 4
ipcp-no-addresses
noipdefault
noauth
novj
nocc
```

```

# gedit /etc/ppp/cdma-connect
'ABORT' 'BUSY'
'ABORT' 'ERROR'
'ABORT' 'NO ANSWER'
'ABORT' 'NO DIALTONE'
'ABORT' 'Invalid Login'
'ABORT' 'Login incorrect'
" 'AT'
" 'AT'
#user: CARD
#pwd : CARD

'OK' 'AT'
'OK' 'AT+CRM=1'
'OK' 'AT+CPS=33'
'OK' 'AT+CMUX=1'
'OK' 'AT+CTA=0'
'OK' 'ATD#777'
'CONNECT' '

# gedit /etc/sysconfig/network-scripts/ifcfg-cdma
DEVICE=ppp0
NAME=20090522145624
MODEMPORT=/dev/ttyUSB0
LINESPEED=115200
USERCTL=true
ONBOOT=no
PERSIST=no
PEERDNS=yes
DEMAND=no
IDLETIMEOUT=600
TYPE=Modem
BOOTPROTO=dialup

# cd /etc/ppp/peers
# pppd call cdma
# tail -f /var/log/messages
看到以下类似的消息：
May 22 14:58:03 localhost chat[6176]: -- got it
May 22 14:58:03 localhost chat[6176]: send (ATD#777^M)
May 22 14:58:03 localhost chat[6176]: expect (CONNECT)
May 22 14:58:03 localhost chat[6176]: ^M
May 22 14:58:03 localhost chat[6176]: AT+CTA=0^M^M
May 22 14:58:03 localhost chat[6176]: OK^M

```

May 22 14:58:06 localhost chat[6176]: ATD#777^M^M  
May 22 14:58:06 localhost chat[6176]: CONNECT  
May 22 14:58:06 localhost chat[6176]: -- got it  
May 22 14:58:06 localhost chat[6176]: send (^M)  
May 22 14:58:06 localhost pppd[6148]: Serial connection established.  
May 22 14:58:06 localhost pppd[6148]: Using interface ppp0  
May 22 14:58:06 localhost pppd[6148]: Connect: ppp0 <--> /dev/ttyUSB0  
May 22 14:58:08 localhost pppd[6148]: CHAP authentication succeeded  
May 22 14:58:08 localhost pppd[6148]: CHAP authentication succeeded  
May 22 14:58:09 localhost pppd[6148]: local IP address 113.112.6.159  
May 22 14:58:09 localhost pppd[6148]: remote IP address 115.168.82.66  
May 22 14:58:09 localhost pppd[6148]: primary DNS address 202.96.128.86  
May 22 14:58:09 localhost pppd[6148]: secondary DNS address 220.192.32.103  
May 22 14:58:15 localhost ntpd[2476]: Listening on interface #12 ppp0, 113.112.6.159#123  
Enabled  
May 22 14:58:24 localhost smartd[2840]: Device: /dev/sda, 45 Currently unreadable (pending)  
sectors

## 12. 开启 MUX 功能说明

## 13. GSM 支持, CDMA 不支持的

本机号码、呼叫限制、线路切换、自动重拨、IP呼叫、自动限时、固定拨号、禁止拨号、网络设置、自动更新时间、小区广播、通话计费、GPRS流量、短信中心号码

## 14. 其他 CDMA 特性

- (1) CDMA 时间是从网络获得。
- (2) CDMA 一次呼叫结束后会再次重新搜索网络。
- (4) GSM 有 IMEI 号, 而 CDMA 对应的说法是 ESN/MEID; GSM 叫 SIM 卡, CDMA 叫做 UIM 卡; GSM 有 STK, CDMA 对应的是 UTK。
- (5) 呼叫限制(**AT+CLCK**), VIA 模组重启后设置无效, 需要记住设置状态, 开机时重新设置模组。
- (6) 语音加密
- (7) 短信优先级

## 15. VIA CDMA CP 编译

VIA 产品的编译流程

### 15.1 相关编译工具的安装:

- 1) ARM 的安装。目前 VIA 项目用到 ARM1.1 及 ARM1.2, 具体用到哪个版本要根据项目的需求。安装完后记得一定要将 ARM 的 update 包也安装完(一般直接 copy 到相应的安装目录下即可)。
- 2) Make 的安装。Make 工具不需要安装, 直接 copy 到 c: 目录下即可, make 文件包在 kingkong 模块的 tool 包里。为了不与现有 MTK 的编译环境冲突, 不要在 windows 系统属性的环境变量直接添加, 手动编辑 ads120.bat:  
set autopath=C:\PROGRA~1\ARM\ADSv1\_2\BIN;%path%;

```

set PATH=C:\make;C:\PROGRAM~1\ARM\ADSv1_2\BIN;%path%
set ARMLIB=C:\PROGRAM~1\ARM\ADSv1_2\LIB
set ARMINC=C:\PROGRAM~1\ARM\ADSv1_2\INCLUDE
set ARMHOME=C:\PROGRAM~1\ARM\ADSv1_2
set ADS_ACTIVE=TRUE

```

## 15.2 cp 编译

- 1) 点击 PC 上的开始 (start) 按钮, 选择运行 (run...), 键入 cmd 进入命令输入窗口
- 2) 编译环境设置 (如果是 ARM1.2, 运行 c:\make\ads120.bat ; 如果是 ARM1.1, 运行 c:\make\armads.bat)
- 3) 进入 code 所在的文件目录下, 进行 code 编译, 编译命令为: build (相应得参数设置)。模组版本 build\_kingkong.bat 的编译选项说明:

```

:beginmake
:: %1. romc4,romc4_clean_all,ramc2,ramc2_clean_all,romc3,romc3_dean_all----芯片型号, 最近的芯片都是romc4。
:: %2. TRUE, FALSE (RUIM,UTK)-----有无uim卡
:: %3. TRUE, FALSE (DATASVC_FCH_ONLY)-----是否支持高速数据业务, 5.1的cpb选false , 5.0的芯片选true。
:: %4. TRUE, FALSE(FEATURE_CHINATELECOM)----- 是否支持中国电信的一些特性;
:: %5. TRUE, FALSE(USE_BANDCLASS5_450M)----- 是否支持450m频段
echo on

```

C218我们用到的是:

```
build_kingkong.bat romc4 TRUE TRUE TRUE FALSE
```

cpb5.0 与 cpb5.1 的区别是前者数据业务比较弱, 一般是低端产品; 后者支持高速数据业务。如果需要使用高速数据业务务必使用 NO\_DATASVC=FALSE 的库。如果要支持数据业务和 AT 命令, 请务必检查 cp.srp 中有 ETS,Id=CP Set AT cmds default。

4) 编译完后会生成 boot.axf、cp.axf、boot.rom 和 cp.rom , 具体目录会根据 CBP 硬件版本的不同目录文件名会有不同 (如 ram\_rev2 或 rom\_rev0 等)。boot.axf、cp.axf 用于 audio 等一些调试, boot.rom 和 cp.rom 用于手机或模块。

## 15.3 模拟器 coyote 编译

模拟器压缩文件解压后, 把 coyote 目录放在与 cp 处于同一个目录下。

In VC, using menu “File\Open Workspace...” to open “coyote\all\all.dsw”.

BuildAll project contains four projects: CoyoteBin, CoyoteUI, CoyoteVAL and CoyoteVTUI.

CoyoteBin project contains source files of windows version of system low layer, including Accessory, Battery, Display, FileSystem, etc.

CoyoteUI project contains the Framework of Coyote as a windows program.

CoyoteVAL project contains VAL source code of the project.

CoyoteVTUI project contains UI/APP parts of the project.

Right click “BuildAll files” and select “Set as Active Project”, click menu “Build\Build” to build the project.

After building the project, select CoyoteVTUI as active project.

Clicking menu “Build\Go” or F5 to run the program. If not having been set, selecting the executable file of the project, normally in coyote\Bin\Debug\,

## 16. VIA CDMA 代码目录结构

Abbreviations and Acronyms:

MON Monitor and Diagnostics  
IOP Input/Output processing  
DBM Data Base Manager  
Boot Bootloader  
FILE ATI Nucleus File Manager  
EXE Real Time Executive  
MMI Man-Machine Interface  
LMD Multiplex Sub Layer Drivers  
LTD Layer One Drivers - CDMA  
L1A Layer One Drivers - AMPS  
HWD Hardware Drivers  
SYS System Level Service  
HAL Hardware Abstraction Layer  
VAL VIA Abstraction Layer  
VTUI VIA Telecom UI  
ASL Application Support Layer  
ResEditor Resource Editor  
Resgen Resource Generator  
MS Mobile Station

via cdma 代码目录结构(版本号:0.21)

cp:各种底层文件的集合, 包括协议栈, DSPM/DSPV, L1D/L1A 软件.

cp 下目录及文件:

- | - ai
- | - app
- | - bm
- | - boot
- | - boot.lnk
- | - build\_201c.bat
- | - build\_grace.bat
- | - build\_muse.bat
- | - build\_ourea.bat
- | - build\_selle.bat
- | - cat.exe
- | - cp2
- | - cp3
- | - CP.LN

- | - cp2. lnk
- | - cp3. lnk
- | - cpflash. LNK
- | - cpflash\_16\_4. LNK
- | - cpflash\_16\_4\_NoDS. LNK
- | - cpflash\_16\_8. LNK
- | - cpflash\_16\_8\_G. LNK
- | - cpflash\_16\_8\_NoDS. LNK
- | - cpflash\_16\_8\_NoDS\_G. LNK
- | - cpflash\_4\_2\_NoDS. LNK
- | - cpflash\_4\_2\_NoDS\_0. LNK
- | - cpflash\_8\_4. LNK
- | - cpflash\_8\_4\_NoDS. LNK
- | - cpflash\_NoDS. LNK
- | - cpsim. lnk
- | - cpver. c
- | - cust
- | - dbm
- | - exe
- | - fsm
- | - hl
- | - hwd
- | - image
- | - inc
- | - iop
- | - ipc
- | - jt
- | - lld
- | - lmd
- | - CP\_Targets. mak
- | - Makefile
- | - makefile\_end
- | - makefile\_feature
- | - makefile\_feature\_def
- | - makefile\_hdr
- | - media
- | - mkdate. exe
- | - mon
- | - nu
- | - nucleus
- | - obj
- | - pde
- | - ps
- | - pst

- | - res
- | - resource.bat
- | - rlp
- | - rom
- | - runit.exe
- | - shared
- | - sys
- | - tst
- | - ui
- | - uim
- | - val

下面对代码进行分类:

CP 代码主要有下面一些模块(目录),

1. 协议相关(ai/dbm/h1/11a/11d/lmd/ps/rlp/ipc)
2. 应用相关(ui/app/res/val/brew/vaawab)
3. 驱动相关(bm/boot/hwd/media/uim)
4. 文件系统相关(fsm)
5. 操作系统相关(nu/nucleus/sys/exe)
6. 公共头文件(inc)
7. 测试调试相关(pst/tst/jt/mon/iop)
8. 客户化相关(cust)

其他文件

Makefile 文件

CP\_Targets.mak

Makefile

makefile\_end

makefile\_feature

makefile\_feature\_def

makefile\_hdr

编译批处理命令

build\_201c.bat

build\_grace.bat

build\_muse.bat

build\_ourea.bat

build\_selle.bat

201c, grace, muse, ourea, selle 是不同的平台, 他们的 UI 配置, Flash 类型, LCD 尺寸等不同。

FLASH

cpflash.LNK

cpflash\_16\_4.LNK

cpflash\_16\_4\_NoDS.LNK  
cpflash\_16\_8.LNK  
cpflash\_16\_8\_G.LNK  
cpflash\_16\_8\_NoDS.LNK  
cpflash\_16\_8\_NoDS\_G.LNK  
cpflash\_4\_2\_NoDS.LNK  
cpflash\_4\_2\_NoDS\_0.LNK  
cpflash\_8\_4.LNK  
cpflash\_8\_4\_NoDS.LNK  
cpflash\_NoDS.LNK

lnk 文件命名规则 cpflash\_大小\_无数据业务.lnk

三个 EXE 文件，协助编译

cat.exe  
mkdate.exe  
runit.exe

前两个的用处

```
./mkdate cp dsrom_rev0/bldtime.c
```

```
cat dsrom_rev0/FLASH dsrom_rev0/SRAM dsrom_rev0/IRAM dsrom_rev0/FLASHV  
> dsrom_rev0/cp.rom
```

runit.exe RunIt Executable with Cat parameter

资源批处理文件

resource.bat

添加新的运用程序，添加一些资源后，要先运行这个脚本。

单一的 C 文件

cpver.c

不言自明了，定义软件版本号

最后，简单讲下与 AT 相关的内容。

```
ai |           |- aic  
  - isotel -|- aie  
           |- aiw
```

ai, AT interpret. isotel 是 ISOTEL Corp. 大概代码被 VIA 拿用就是了。

isotel 分为三个目录：

aic 是给客户可定制的 AT,

aie Ai engine for kerkel process, 提供解析 AT, 数据结构, 宏等头文件。

aiw Ai wrapper to interactive with other module.

与 AT 相关的文件还有 val\ValATCmd.c, 常用上报命令默认开关和常用命令的默认设置都在这文件里设置, 里面还提供了短信编码相关的函数等。

## 17. 升级模组软件

用数据线接 CDMA 模块的 Debug 串口, 按顺序用 ETS 下载 boot.rom cp.rom cp.srp。

**注意:** 在下载 boot.rom 时切勿断电而导致返厂 JT 维修处理。假使模块引导 Boot 区被冲掉, 只能通过 JTAG 工具进行升级或将 FLASH 取下, 用烧录器写入程序文件。

## 18. 参考文档

1. Zjiang, VIA-Telecom 《模块开发指南 20090303.doc》
2. Simware 《G+C 网开发注意的问题》
3. 中国电信《CDMA 终端需求规范》

## 附录一 使用 Ruby 在 windows 下通过串口自动测试 AT

编译生成新的软件版本, 烧入 Flash, 要对模组的 AT 进行一次测试, 验证是否正常, 比较与上个版本是否有差别。手工一个个验证, 乏味, 容易对工作失去兴趣。

从 <http://rubyinstaller.rubyforge.org> 可以下载 Ruby One-Click, 现在最新版本安装软件为 ruby186-27\_rc2.exe。安装过程中, 把 RubyGems 也勾上。在 windows 上用 ruby 访问串口, 可以用 win32ole 模块, 也可以用 ruby-serialport, 后者是跨平台的。安装 ruby-serialport 前要做一些准备工作, 确认有装 Microsoft's Visual C++ 或 Borland's C++ 编译器, 这里以 VC6.0 为例。

(1)设置环境变量:

`PATH=C:\Program Files\Microsoft Visual Studio\VC98\Bin`

`INCLUDE=C:\Program Files\Microsoft Visual Studio\VC98\Include`

`LIB=C:\Program Files\Microsoft Visual Studio\VC98\Lib`

(2)安装 ruby-serialport:

开始 -> 程序 -> Ruby-186-27 -> RubyGems -> RubyGems Package Manager, 输入:  
`gem install ruby-serialport`

如果提示找不到 mspdb60.dll, 请到 <http://www.dll-files.com> 找, 把它放到

`C:\Program Files\Microsoft Visual Studio\VC98\Bin`

接下来, 就可以写一个测试脚本了。

```
# filename: test.rb
require 'rubygems'
require 'serialport'
```

```
# 0 is mapped to "COM1" on Windows, and 5 is COM6, 115200 is baud rate
sp = SerialPort.new(5, 115200)
```

```
sp.write "AT\r\n"  
sleep(0.2)  
puts sp.read # hopefully "OK" ;-
```

注意：如果没有 sleep,可能会收不到"OK"响应。

如果你弄了老半天, ruby-serialport 也没装得上, 那就用用 win32ole,美其名曰: Windows Automation。win32ole 是 Masaki Suketa 编写的 Ruby 扩展, 是标准 ruby 发行版本的一部分。现在写一个 mscomm.rb 文件

```
#!/usr/bin/env ruby  
#  
# filename: mscomm.rb  
# @date 2009.5.16  
#  
  
require 'win32ole'  
  
class MSCOMM  
  def initialize(port)  
    @serial = WIN32OLE.new("MSCOMMLib.MSComm")  
  
    @serial.CommPort = port  
    @serial.Settings = "115200,N,8,1"  
    @serial.InputLen = 0  
    @serial.PortOpen = true  
  end  
  def write(str)  
    @serial.Output = str  
  end  
  def read  
    str = @serial.Input  
    str  
  end  
  def close  
    @serial.PortOpen = false  
  end  
  def serial  
    @serial  
  end  
end
```

再写一个简单的测试用例 test\_comm.rb:

```
#!/usr/bin/env ruby
```

```

#
# filename: test_comm.rb
# @date 2009.5.16
#

require 'mscomm.rb'

comm = MSCOMM.new(6)
comm.write("AT\r\n")
sleep(0.2)
puts comm.read
comm.close

```

和上面的 `test.rb` 用法一样，没什么神秘感，是不是很简单呢 :-)

再加个查询信号的 AT，在 `test_comm.rb` 倒数第二行加上

```

comm.write("AT+CSQ?\r\n")
sleep(0.2)
puts comm.read
要是很多很多...想想，还是写个函数吧.
def exec_cmd(comm, cmd)
  comm.write("#{cmd}\r\n")
  sleep(0.2)

  begin
    result = comm.read
    result = "Command not support\n" if result.include?("ERROR\n")
  rescue
    result = "Writing serial port error\n"
  end

  puts result
end

```

于是, `test_comm.rb` 中间部分的可以可以写成:

```

exec_cmd(comm, "AT")
exec_cmd(comm, "AT+CSQ?")

```

可是每添加一个 AT... 还是得写一串的 `exec_cmd(comm...`。Dave Thomas 大师说过: "Don't Repeat Yourself!"，这是 ruby 的设计理念。假设所有的 AT 都放在另外一个文件呢? 我们也可以一个个读取出来。不过现在考虑的是暂时放在同个文件，那就定义一个字符串数组:

```

atcmd = {
  'AT',
  'AT+CSQ?',

```

```
'AT+CREG?'  
}
```

也可以这样写:

```
Atcmd = %w{  
AT  
AT+CSQ?  
AT+CREG?  
}
```

第二种对于添加 AT 比较方便, 但缺点是 AT 不能出现空格。那么 test\_comm.rb 现在又可以写为:

```
atcmd.each {|at| exec_cmd(comm, at) }
```

可是, 有的 AT 需要花一些时间才有响应。之前都默认是 0.2 秒, 好吧, 重新定义:

```
def exec_cmd(comm, cmd, timeout = 0.2)  
  ...  
  sleep(timeout)  
  ...  
end
```

于是, 就可以这样使用

```
exec_cmd(comm, "AT+CDV=10000", 3)  
exec_cmd(comm, "AT+CLCC?")  
exec_cmd(comm, "AT+CHV")  
exec_cmd(comm, "AT+CPOF", 2)  
exec_cmd(comm, "AT+CPON", 5)
```

像这样的一组 AT, 具有依赖顺序而每个 AT 的响应时间又不一样, 我们只能根据不同的情况对 AT 做分类, 写不同的测试脚本。

再回头看看 test\_commm.rb, 如果我们想把输入与输出都放在同一个 Excel 表格, 那该如何写呢? 原理一样, 依旧用 win32ole:

```
#!/usr/bin/env ruby  
#  
# filename: excel.rb  
# @date 2009.5.16  
#
```

```
require 'win32ole'
```

```
class Excel  
  def initialize(filename = nil)  
    @excel = WIN32OLE.new("excel.Application") # create Excel object
```

```

    @excel.Visible = TRUE
    if (filename == nil)
        @workbook = @excel.Workbooks.Add() # create new file
    else
        @workbook = @excel.Workbooks.Open(filename) # open exist file
    end
end

def setvalue(pos, data)
    @excel.Range(pos).Value = data
end

def save
    @excel.Save()
end

def close
    @excel.Quit()
end

def excelobj
    @excel
end

end # end of class

```

我们重新写个 test\_cdma\_at.rb

```

#!/usr/bin/env ruby
#
# test_cdma_at.rb
# @date 2009.5.16
#

require 'mscomm.rb'
require 'excel.rb'

def exec_cmd(comm, cmd)
    comm.write("#{cmd}\r\n")
    sleep(0.2)

    begin
        result = comm.read
        result = "Command not support\n" if result.include?("ERROR\n")
    rescue
        result = "Writing serial port error\n"
    end
end

```

```

    return result
end

atcmd = %w{
AT
AT+CPIN?
AT+CPINC?
AT+CSQ?
AT+CREG=2
AT+CREG?
AT+VMCC?;+VMNC?
AT+CPBS=?
AT+CPBS?
AT+CPBS="ME"
AT+CPBS?
AT+CPBW=3,13544049382,"violet",0
AT+CPBR=3
AT+CMGS=13544049382,"Hello!"
AT+CMGW=,13544049382,"Hi!"
}

comm = MSCOMM.new(6)
excel = Excel.new("d:\\Book1.xls")

i = 1

atcmd.each {|at|
  excel.setvalue("a#{i}", at)
  excel.excelobj.Range("b#{i}").Value = exec_cmd(comm, at)
  i += 1
}

comm.close
excel.save
excel.close

```

代码不难理解，就是把 AT 输入放在 Excel 表格某行的 A 列，响应写入某行的 B 列，仅此而已。

脚本测试也不是万能的，AT 本身具有随意性，写脚本意在减轻编码工作量，避免重复机械的劳动。