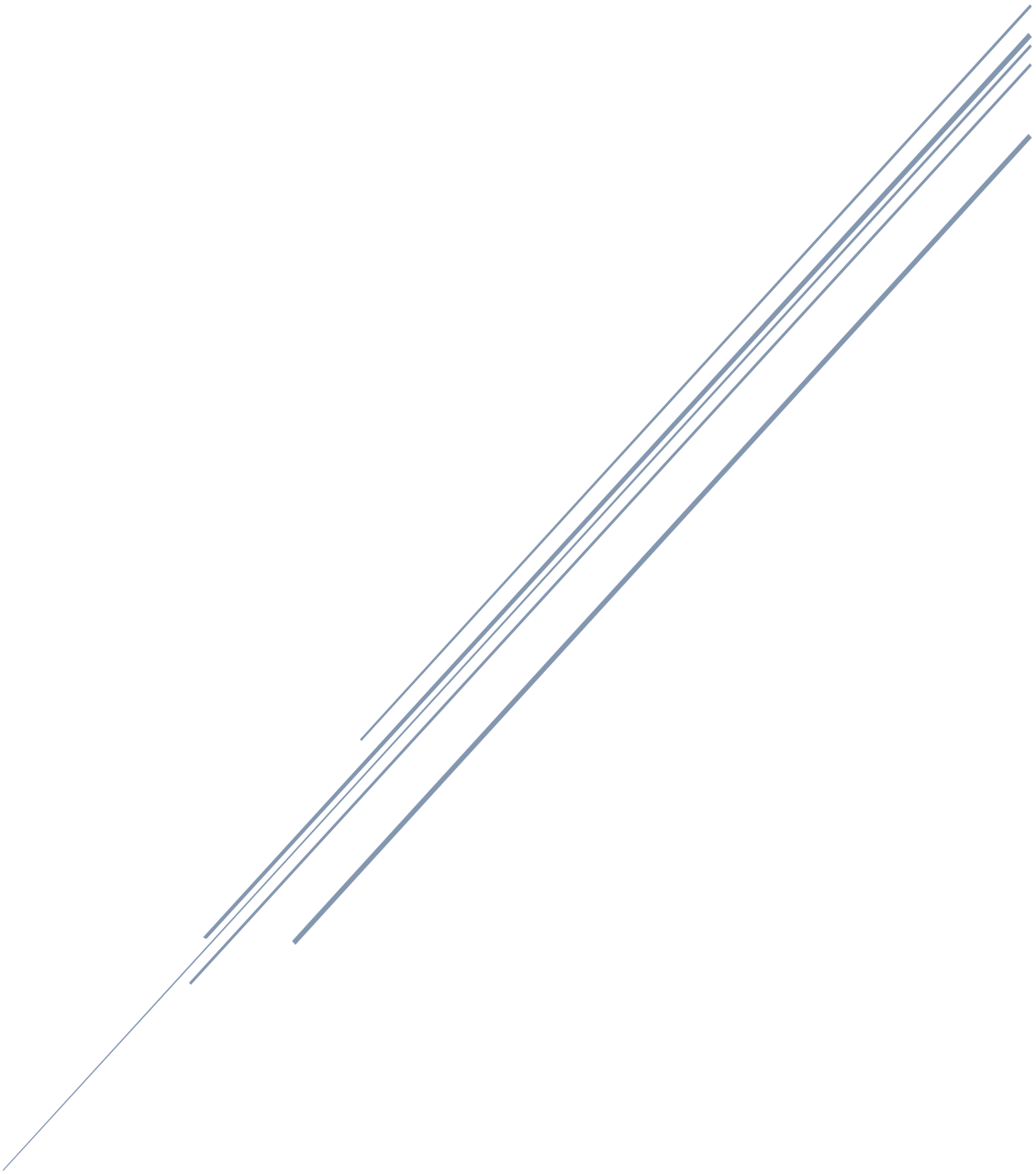
UPO2000CS&UPO7000Z 系列示波器

编程手册



优利德科技（中国）股份有限公司

首页

本文档适用于 UPO2000CS&UPO7000 系列示波器的二次开发，此文档只给出通信控制命令的说明，通信接口(UCI)请查阅《UCI 帮助文档》，本文档包括以下内如：

发布说明：

版本	修改项
V1.0	正式版
V1.0.1	新增 mea:all?; 接口，该接口为示波器参数测量的统一接口。
V1.1	去除多余接口，减少干扰。

通信接口：

- 写参数使用 uci_Write 、 uci_WriteX 或 uci_FormatWrite ；
- 读参数使用 uci_Read 或者 uci_ReadX；
- 读文件使用 uci_ReadToFile 或者 uci_ReadToFileX；
- 写文件使用 uci_WriteFromFile 或者 uci_WriteFromFileX.

命令字符串格式

“
命令名 1：命令参数@属性 1:属性值@属性 2:属性值 。。。@属性 n:属性值;
命令名 2：命令参数@属性 1:属性值@属性 2:属性值 。。。@属性 n:属性值;
。。。 。。。
命令名 n：命令参数@属性 1:属性值@属性 2:属性值 。。。@属性 n:属性值;
”

说明：

1、 对字符大小写不敏感；
2、 数值支持十六进制、八进制、十进制格式；
3、 支持多条语句（具体视机型而定）
如果多条语句或者多个属性执行失败，请尝试使用单条语句和单个属性重试；
4、 每个语句必须以';'结尾；
5、 名称、值和各标记间支持空格存在；

示例：

“CH:0@CP:D@Invert:10;”
“CH:0@EN:1;”
“MATH@SRC1:0;”

通用命令

命令名	意义	IO	数据	备注
IDN?	查询设备名	R	char v[50];	
SN?	查询序列号	R	char v[50];	保留
SCN?	屏幕信息	R	Struct : ScreenInfo	
CHSel?	查询哪个通道被选中	R	Enum(Integer) : 0/1/2/3/0xff{ CH1/ CH2/CH3/CH4/无通道选中}	设置通道选中使用 “CH:1@SEL:1;”
CHNUM?	查询通道数	R	Integer : 4Bytes	
UDS?	查询 U 盘是否接入 (U Disk Status)	R	Enum(Integer): 0/1{NO/YES}	
Local	Lock Pad(锁键盘)	W	Enum:0/1{远程状态/本地状态}	远程状态时设备键盘锁。
Local?	查询键盘是否锁定	R	Enum:0/1{未锁定/已锁定}	
SYNC?	同步设备的所有状态。	R	SyncStates(详见定义头文件)	
CYMTR?	读取频率计测量结果	R	UValue	
Lock?	查询所有按键的锁定状态	R	64 有符号整数， 标记位	
Led?	查询所有按键的 LED 状态	R	KeyLedStat	

示例：

锁键盘 ： “Local:0;”

解锁键盘： “Local:1;”

运行状态

命令名	命令参数	数据
Proc	**设置**运行状态	Enum : STOP/RUN/AUTO
Proc?	**查询**运行状态	Enum : STOP/RUN/ARMD/READY/TRIGD/AUTO/SCAN/OVER/RESET { STOP/RUN/ARMED/READY/TRIGED/AUTO/SCAN/OVER/RESET }

示例：

查询：“Proc?;”

设置：“Proc:Stop;” “Proc:AUTO;”

注意：

关于“Proc:Stop”、 “Proc:Run” 和 按键“RUN/STOP”的区别：

与按键“RUN/STOP”的功能不同。这里“RUN”就是置 DSO 为 RUN 状态，无论目前处于何种状态。“STOP”类似。

截图

命令名	命令参数	命令参数类型
PrtScn	图像数据格式	String: 空/.bmp/zip{未压缩的像素数据/BMP 文件/压缩像素数据}

示例：

BMP 文件： “PrtScn:.bmp;” //注意 bmp 前有个‘.’

未压缩像素数据： “PrtScn;”

压缩像素数据： “PrtScn:zip;”

注意：

- “PrtScn:.bmp;” 使用 uci_ReadToFile 接口
- **“PrtScn;” ：使用 uci_Read 接口，**缓冲区大小可设置为 800 * 480 * 4 = 1536000 Bytes.
- **“PrtScn:zip;” 使用 uci_Read 接口，**缓冲区大小可设置为 1152000，实际有效数据量通过接口返回值获取。然后调用 alg_UnCompressPixels（uci_alg.h）解压像素数据。解压后的像素数据大小是 800 * 480 * 4 = 1536000 Bytes. 即是 32bit 颜色格式。

  得到的图像是：从下往上，按行扫描。示例代码：

```
for (int v = 480; v > 0; v--)
{
    for (int h = 0; h < 800; h++)
    {
        dc.SetPixelV(h, v, RGB(p[2], p[1], p[0]));
        p += 4;
    }
}
```

按键

命令名	命令参数	命令参数类型
KEY	键值	String ： 缩写的键名

按键名	意义
AT	AUTO
RS	RUN/STOP
TM	*MENU(触发菜单)
SG	SINGLE
FC	FORCE
HP	HELP
VM	*MENU(视窗设置菜单)

MT	MATH
RF	REF
C1	CH1
C2	CH2
C3	CH3
C4	CH4
F1	F1
F2	F2
F3	F3
F4	F4
F5	F5
MOF	Menu ON/OFF
PS	PrScrn(截屏)
MEA	Measure
CSR	Cursor
ACQ	Acquire
DISP	Display
STG	Storage
UTIL	Utility
CLR	Clear
DEC	Decode
DEF	Default
FKN	Function Knob
FKNL	Function Knob Left Rotation
FKNR	Function Knob Right Rotation
VKN	Vertical Position Knob
VKNL	Vertical Position Knob Left Rotation
VKNR	Vertical Position Knob Right Rotation
HKN	Horizontal Position Knob
HKNL	Horizontal Position Knob Left Rotation
HKNR	Horizontal Position Knob Right Rotation
TGKN	Trigger Position Knob
TGKNL	Trigger Position Knob Left Rotation
TGKNR	Trigger Position Knob Right Rotation
VBKN	Voltage Base Knob
VBKNL	Voltage Base Knob Left Rotation
VBKNR	Voltage Base Knob Right Rotation
TBKN	Time Base Knob
TBKNL	Time Base Knob Left Rotation
TBKNR	Time Base Knob Right Rotation

属性名	意义	IO	数据
Lock	锁定该按键	W	无数据
Unlock	解锁该按键	W	无数据
Lock?	查询该按键锁状态	R	Integer: 0 – 未锁; 1 – 已锁
Led?	查询 LED 状态	R	Integer: 0 – 不亮; 1 – 亮（绿灯）；2: 亮（红灯）

示例：

“KEY:AT;”
“KEY:C1;”
“KEY:MATH;”
“KEY:FKN;” -> 按下
“KEY:FKNL;” -> 功能旋钮增
“KEY:FKNR;” -> 功能旋钮减
“KEY:FKNL@lock;” -> 功能旋钮增-锁定

注意：

因为不同的设备对同一按键名会有不同，所以，采用缩写以规避这种差异！
键盘全锁和全解使用 local 命令。
带问号的指令需要用 uci_Read 读取。

抓取波形数据

命令名	命令参数	命令参数类型
Capture wave	波形格式	String:.dat /.csv {内置数据/CSV 文件}

属性名	意义	IO	数据
CH	Channel(导出的通道 ID)	--	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}

示例：

“capture wave:.dat;”
“capture wave:.csv;”
“capture wave:.dat@CH:0;”

说明：

1、 如果**不**添加“@CH:0”属性，表示抓取**当前选中**通道；
2、 如果添加了“@CH:0”属性，表示抓取**指定**的通道；
3、 如果命令指定的通道未被打开，则会返回错误；
4、 读文件使用 uci_ReadToFile 或者 uci_ReadToFileX 接口。

读写配置数据

命令名	命令参数	命令参数类型
dconfig	无	

示例：

“dconfig;”

说明：

如果要保存到文件，请自行保存。 读数据超时一般 2s 足够；写超时设置为 8s；如果超时，请延长超时时间。

接口：

- 用 uci_Read；读取数据，用 uci_Write 写数据。 读取数据量设置为 500000Bytes， 实际大小通过 uci_Read 的返回值得到。

通道控制

基本（共有）属性：

命令名	命令参数	命令参数类型	备注
CH	通道号	以 0 开始的通道编码，整数值。	**通道号**指明后续的指令针对的是哪个通道！

属性名	意义	IO	数据
EN	ON/OFF	WR	Enum(Integer): 0/1{ON/OFF}
SEL	Select(设置或者查询当前通道的选中状态)	WR	W：无{命令类型} R：Enum(Integer): 0/1{ON/OFF}
VP	VPOS	WR	Integer ： 位置值
HP	HPOS	WR	Integer ： 位置值
TB	Time Base（时基挡）	WR	W ： Enum(String) ： +/- {加一档/减一档}或 Integer ： Rang code R ： Integer ： Rang code
VB	Voltage Base（幅格挡）	WR	W ： Enum(String) ： +/- {加一档/减一档}或 Integer ：

			Rang code. R : Integer : Rang code
TBV	Time Base Value（时基挡）	R	CellBaseValue ：读取时基档数据。
VBV	Voltage Base Value(幅格挡值)	R	CellBaseValue ：读取伏格档数据。
STZ	VPOS Set to zero	W	无{命令类型}
HSTZ	Set HPos to zero	W	无{命令类型}

示例：

假设通道号分别是：0/1/2/3/4/5/6/7/8 {CH1/ CH2/ CH3/ CH4/ MATH/ REF-A/ REF-B/ REF-C/ REF-D}

设置：

“CH:0@EN:1@VP:128@HP:350;”

“CH:4@EN:1;” --- 打开 MATH 通道

“CH:4@VP:128;” --- 调节 MATH 通道的垂直位置为 128

“CH:4@SEL;” --- 选中 MATH 通道

查询：

“CH:0@VP;”

“CH:0@TB;”

注意：

1、 关于基础命令和物理通道的命令协议：
基础命令和物理通道的命令名都使用“CH”，格式为“CH: channel id@Attribute:Value;”。基础命令是适用于物理通道、MATH、REF 的，通过 channel id 进行区分；

2、 CH 命令必须跟通道号作为命令参数，否则为错误的命令格式；

3、 执行“@SEL;”指令时，如果将要选中的通道未打开，则会打开通道；

4、 如果是物理通道（CH1/ CH2/ CH3/ CH4），“@VB” 在细调时（通过“@ VD”设置）只能通过“Enum(String) : +/- {加一档/减一档}”的方式设置，然后通过“@VBV”指令获取实际的伏格挡信息。

物理通道：

命令名	命令参数	命令参数类型	备注
CH	通道号	以 0 开始的通道编码，整数值。	**通道号**指明后续的指令针对的是哪个通道！

属性名	意义	IO	数据
CP	Coupling(通道耦合)	WR	Enum(String): D/A/G{DC/AC/GND}
BW	BW limit	WR	Enum(String): F/20M{Full/20MHz} / Enum(Integer): 0/1{ON/OFF}
VD	Volts/Div	WR	Enum(String): C/F{Coarse/Fine}
Probe	Probe	WR	Enum(Decimals): 0.001/0.01/0.1/1/10/100/1000
Invert	Invert	WR	Enum(Integer): 0/1{ON/OFF}
BSEN	Bias-ON/OFF	WR	Enum(Integer): 0/1{ON/OFF}
BSV	Bias-VolValue	WR	W:Integer:步进差值(如，1，-2) R: CellBaseValue

BSZ	Bias-Set to zero (归零)	W	无{命令类型}
Unit	Unit(单位)	WR	Enum(String): V/A/W/U{V/A/W/U }

示例：

“CH:0@CP:G@BW:F@VD:C@ Probe:1@Invert:1@BSEN:1@BSV:2;”

注意：

1、“CH:0;”这条是无效的命令，通道选中应该使用“CH:0@SEL;”

数学：

MATH：

命令名	命令参数	命令参数类型
MATH	无	无

属性名	意义	IO	数据
T	Type(类型)	WR	Enum(String) : M/F/L/Fi/AD{Math/FFT/Logic/Filter/Advance}
SRC1	Source1	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
Op	Operator (算子)	WR	Enum(String) : +,-,*,/
SRC2	Source2	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}

示例：

“MATH@T:F;” ---切换类型

“MATH@SRC1:0@OP:+@SRC2:1;”

FFT：

命令名	命令参数	命令参数类型
FFT	无	无

属性名	意义	IO	数据
SRC	Source	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
WND	Windows	WR	Enum(String): Hm/B/R/Hn{Hamming/BlackMan/Rectangle/Hanning}
Unit	Vertical Scale	WR	Enum(String): r/d {Linear RMS/dBV RMS}
Freq	Frequency	R	CellBaseValue : 读取 FFT 波形的频率数据。

示例：

“FFT@SRC:0@WND:hm@ Unit:d;”

注意：

“@Unit”切换到“dBV”时，FFT 对应的伏格挡信息“@VBV”就变成 dB 值了。

Logic:

命令名	命令参数	命令参数类型
lg	无	无

属性名	意义	IO	数据
Exp	Expression	WR	Enum(String): A/O/N/X { AND/OR/NOT/XOR}
Src1	Source1	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
Src2	Source1	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
Invert	Invert	WR	Enum(Integer): 0/1{ON/OFF}

示例：

“lg@src1:0@exp:a@ src2:2;”

Filter:

命令名	命令参数	命令参数类型
Filter	无	无

属性名	意义	IO	数据
SRC	Source	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
T	Filter Type	WR	Enum (String): L/H/B/BS {LowPass/HighPass/BandPass/BandStop}
FH	Freq High	WR	Integer (Step)，设置步进(**step**)。
FL	Freq Low	WR	Integer (Step)，设置步进(**step**)。

示例：

“Filter@SRC:0;”

“Filter@T:L;” --- 切换滤波类型为低通滤波。

“Filter@ FH:1@ FL:2;”

注意：

1、 频率设置：

得到以 s 为单位的时基档时间值 **ts**，然后计算一个设置步进对应的频率步进为：

```
FilterFreqStep = 1 / (ts * 4) ;
```

再根据下面的步进转换频率的公式转换：

```
freq = FilterFreqStep * step;
```

Advance:

命令名	命令参数	命令参数类型
adv	无	无

属性名	意义	IO	数据
exp	Expression	WR	String : 50 chars

示例：

“adv@exp:ch1+ch2;”

注意：

如果表达式错误，接口会返回错误码。

Ref:

命令名	命令参数	命令参数类型
Ref	无	无

属性名	意义	IO	数据
SRC	REF Source	WR	Enum(String) : A/B/C/D
Disk	Storage Medium(存储媒介)	WR	Enum(String) : F/U{Flash/U-Disk }
Pos	Storage Position(存储位置)	WR	Integer : 位置索引，范围[0,255]，以 1 为步进。
Load	回调	W	空（执行命令）
Clear	清除	W	空（执行命令）

示例：

“REF@Disk:F@ POS:1@Load;”

“REF@Clear;”

注意：

1、 REF 波形的保存是通过“STGE”命令，这里只是加载。

参数测量(Measure)

命令名	命令参数	命令参数类型
Mea	类别	字符串

命令参数	意义	IO	数据
All?	打包读取测量参数	R	读取的是所有示波器统一的数据结构。数据见参数测量统一数据包：
user	定制 5 组测量参数	W	MeaUserParams params[5]; 10Bytes
User?	读取 5 组测量参数数据	R	MeaValue mp[5] （40Bytes）
Clear	执行清除功能	W	无

属性名	意义	IO	数据
Src	Source(信源)	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
ALL	开关所有参数测量	W	Enum(Integer) : 0/1{ON/OFF}

示例：

“Mea@ALL:1;” --- 打开“所有参数测量”

“Mea@SRC:1;”

“mea:all?;” --- 打包读取所有参数值（通用）.

注意：

- 使用“Mea:All?;”首先要使用“@ALL;”属性开启测量，否则会读取失败。

光标测量(Cursor)

命令名	命令参数	命令参数类型
Cursor	无	无

类别	属性名	意义	IO	数据
共有命令	T	Type(类型/关闭)	WR	Enum(String) : T/A/C{Time/Amp/Close}
	Mode	Mode(模式)	WR	Enum(String) : Ind/Tck{Independent/Tracking }
	AP	A Line Pos(A 光标线位置)	WR	Integer : 位置值，竖线范围[0,699]，横线范围[28,227].
	BP	B Line Pos(B 光标线位置)	WR	Integer : 位置值，竖线范围[0,699]，横线范围[28,227].
	Src	Source(信源)	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
	Result	读取测量结果	R	CursorMeasurePack （50Bytes）

示例：

关闭：“Cursor@T:T;”

切换到幅度测量：“Cursor@T:A;”

读取参数测量结果： “Cursor@Result;”

注意：

1、 光标测量的结果是通过“数据帧”读取的；

2、 使用“Cursor@Result;”首先要使用“@T;”属性开启光标测量，否则会读取失败。

频率计：

命令名	命令参数	命令参数类型	备注
cmeter	--	--	--

属性名	意义	IO	数据
EN	打开/关闭	WR	Enum(Integer<4Bytes>): 0/1
Freq?	频率值	R	Double<8Bytes>

示例：

“cmeter@en:1;”

“cmeter@freq?;” -- 读取频率值

采集配置(Acquire)

命令名	命令参数	命令参数类型
ACQ	无	无

类别	属性名	意义	IO	数据
共有命令	T	Type(类型)	WR	Enum(String) : N/P/A/E/H {Normal/Peak/Average/Envelope/High Res }
	Depth	Memory Depth(存储深度)	WR	~~Enum (String): A/32K/320K/3.2M/32M {Auto/32K/320K/3.2M/32M}~~ Enum (String): A/28K/280K/2.8M/28M {Auto/28K/280K/2.8M/28M}
平均	AVGS	Averages(平均次数)	WR	Integer(Step) : 步进值（N）。范围[0,12]，计算公式见注1.

示例：

切换类型：“ACQ@T:N;”

“ACQ @Depth:A;”

“ACQ@AVGS:2;”

注意：

1、2^(N + 1), ie: {(0: 2), (1: 4), (2: 8), (3: 16)... (12: 8192)}

显示配置(Display)

命令名	命令参数	命令参数类型
Disp	无	无

属性名	意义	数据
VT	View Type(视图类型)	Enum(String): YT/XY12/XY34
WT	Wave Type(波形显示类型)	Enum(String) : V/D{Vector /Dots}
PST	Persist Time (持续时间)	Enum(String): Min/0.1/0.2/0.5/1/5/10/X {Min/0.1s/0.2s /0.5s /1s /5s /10s /Infinite}
GLum	Grid Luminance (栅格亮度)	Integer(Step): 步进值，范围为 [1,100]，以 1 为步进。
WLum	Wave Luminance (波形亮度)	Integer(Step): 步进值，范围为 [1,100]，以 1 为步进。

示例：

“Disp @VT:XY@WT:V @PST:1@Bri:100;”

存储配置(Storage)

命令名	命令参数	命令参数类型
STGE	无	无

属性名	意义	数据
T	Type(类型)	Enum(String) : S/R{Setup/Ref Wave }
Src	Source(信源)	Enum(Integer) : 0/1/2/3{CH1/CH2/CH3/CH4}
Disk	Storage Medium(存储媒介)	Enum(String) : F/U/UC{Flash/U-Disk/USB-CSV }
Pos	Storage Position(存储位置)	Integer : 位置索引，范围[0,255]，以 1 为步进。
Save	保存	空（执行命令）
Load	装载	空（执行命令）

示例：

REF:

保存：

“STGE@T:R@SRC:0@DISK:F@POS:1@SAVE;”

回调：

“REF@DISK:U@POS:1@LOAD;”

Setting:

保存：

“STGE@T:S@DISK:F@POS:1@SAVE;”

回调：

“STGE@T:S@DISK:F@POS:1@Load;”

系统配置(Utility)

命令名	命令参数	数据
sys	无	

属性名	意义	IO	数据与备注
CAL	Self Calibration(自校准)	W	自校正期间，不能正常通信。
Clear	Clear info(清除参考波，与面板 Clear 功能一致)	W	无{命令类型}
CLRD	Clear Data(清除自定义设置)	W	无{命令类型}
Reset	Reset(恢复出厂设置)	W	无{命令类型}
Ver?	Version (版本)	R	**String(128Byts):** **HW:1.0;SW:1.0;PD:2014-11-20;ICV:1.4.0;** 其中 HW 为硬件版本号，SW 为软件版本号，PD 为生成日期，ICV 为协议版本号；

Time	RTC		数据类型： Time
CYMTR	Cymometer(频率计)	WR	Enum(Integer): 0/1{ON/OFF}
SWO	Square wave output (方波输出)	WR	Enum(String): 10/100/1K/10K{10Hz/100Hz /1KHz /10KHz }
OSEL	Output Select(输出选择)	WR	Enum(String): T/PF{Trgger/Pass&Fail}
LANG	Language	WR	Enum(String): en/chs{English/Simple Chinese}
MDisp	Menu Display (界面显示时间)	WR	Enum(String) : 1/2/5/10/20/M {1s/2s/5s/10s/20s/Manual}
BLUM	Backlight Luminance (背光亮度)	WR	Integer(Step): 步进值，范围为 [1,100]，以 1 为步进。

示例：

“SYS@CAL;”

“SYS@CYMTR:1;”

“SYS@LANG:CHS;”

“SYS@GBri:90;”

触发系统(Trigger)

命令名	命令参数	数据
Trig	无	无

类别	属性名	意义	IO	数据
基础	T	Trigger Type (类别)	WR	Enum(String):E/P/V/S/R/W/D/T/DU/SH/N/PN {Edge/Pulse/Video/Slope/RUNT/Window/Delay/Timeout /Duration/SetupHold/NthEdge/Pattern}
共用	SRC	Trigger Source (触发源)	WR	Enum(**Integer**):0/1/2/3/4/5/6 {CH1`CH2`CH3`CH4`AC Line` EXT ` EXT/5} 注：只有 Edge/Pulse/Video 有 ： AC Line ` EXT ` EXT/5
	CP	Trigger Coupling (触发耦合)	WR	Enum(String) : D/A/L/H/N {DC/AC/LF Reject/HF Reject/Noise Reject} 注：只有 Video 没有
	Mode	Trigger Mode (触发模式)	WR	Enum(String): A/N{Auto/Normal}
触发电平	POS	Trigger Pos (触发位置)	WR	Integer ： 位置值(以通道基线为 0 点的上正下负值) W ： 当前触发电平位置的偏移量，高低电平同时存在时： 1、 高低电平同步移动时，为低电平的移动量； 2、 其它情况下为当前移动的电平的偏移量。 R ：{低 16 位为正常的触发位置（或斜率触发低电平位置），高 16 位为斜率触发的高电平位置}

	STZ	Set to zero (触发电平位置置零)	W	无{命令类型}
	TV	Trigger Voltage (触发电压)	WR	W: 写触发电压 {写电压值，但数值必须根据幅格挡、屏幕信息换算} R: 读取当前触发电压信息 CellBaseValue v[2]，详见注①
Pulse &Slope & Runt & Delay	COND	Condition(条件)\When	WR	**Pulse&Slope** : Enum(String): > / < / = **Runt** : Enum(String): N / > / < / ={N: None} **Delay** : Enum(String): > / < / < > / > < **Duration** : Enum(String): > / < / < >
Pulse &Runt	POL	Polarity(极性)	WR	Enum(String): N/P { Negative (负脉宽/负极性)/Positive(正脉宽/正极性) }
Edge &Slope &Window& NthEdge& SetupHold	ST	Slope Type (斜率类型/边沿类型)	WR	**Slope& NthEdge & SetupHold :** Enum(String): F/R {Fall/Rise } **Edge&Window&Timeout :** Enum(String): F/R/A {Fall/Rise/Any}
Runt &Window	TL	Trigger Level (触发电平)	WR	Enum(String): L/H {Low/High }
Delay& Duration	TS	Time Sel(时间)	WR	Enum(String): N/U/L{Normal/Upperlimit/Lowlimit}
Duration& SetupHold &Pattern	CODE	Code(码型)		**Duration:** Enum(String): H/L/X {H/L/X} **SetupHold:** Enum(String): H/L {H/L} **Pattern:** Enum(String): H/L/X/R/F {H/L/X/Rise/Fall}
Pulse& Slope& RUNT& Window& Timeout& SetupHold & Nth Edge	TSET	时间设置	WR	Integer(Step) : 步进值，范围[2,2500000000]，以 1 为步进， 一个步进对应的时间是 4ns，所以对应的时间范围为：[8ns, 10s]， 也说明设置的时间是 4ns 的整数倍。
Video	STD	Standard(视频制式)	WR	Enum(String): N/P/S{NTSC/PAL/SECAM}
	SYNC	Synchronization(同步)	WR	Enum(String): E/O/A/L { Even Field/Odd Field/All Lines/Line Num}
	LN	Line Number(行数)	WR	Integer : 行数{设置方式视机型而定}
	SR	Slew Rate(压摆率)	R	只读属性。数据：*char **Data**[16]*
Slope	TH	Threshold(阈值)	WR	Enum(String): L/H/LH {Low/High/Low&High}
Window	WP	Window Position (位置)	WR	Enum(String): E/EX/T {Enter/Exit/Time}
Delay	SRC1	Source1(信源 1)	WR	Enum(Integer) : 0/1/2/3{CH1/CH2/CH3/CH4}
	SRC2	Source1(信源 2)	WR	Enum(Integer) : 0/1/2/3{CH1/CH2/CH3/CH4}

	DSel	Select Focused Source	WR	Enum(Integer) : 1/2{SRC1/SRC2}
	EG1	Edge1(边沿 1)	WR	Enum(String): R/F{Rise/Fall}
	EG2	Edge1(边沿 2)	WR	Enum(String): R/F{Rise/Fall}
SetupHold	DSRC	Data Source (数据源)	WR	Enum(Integer) : 0/1/2/3{CH1/CH2/CH3/CH4}
	CSRC	Clock Source (时钟源)	WR	Enum(Integer) : 0/1/2/3{CH1/CH2/CH3/CH4}
	SSel	Select Focused Source	WR	Enum(Integer): 1/2{Data Source/Clock Source}
	SH	Setup/Hold	WR	Enum(String): S/H/SH{Setup/Hold/Setup&Hold}
	ENUM	Edge Num(边沿值)	WR	Integer : 步进值 范围:

示例：

切换类型：“Trig@T:E;”

“Trig@T:E@ET:RF@SRC:0@CP:D@MODE:A;”

“Trig@T:P@POL:N@ TSET:12;”

“Trig@T:V@STD:N@SYNC:LN@LN:20;”

“Trig@T:S@ST:F@SR:12 @TH:L;”

注意：

① “@TV”读取到的数据是 CellBaseValues TrigVolt[2],
普通触发只有一个触发电平 数据为 TrigVolt[0];
斜率触发有两个触发电平， 数据为 TrigVolt[1];

解码系统(Decode)

命令名	命令参数	命令参数类型
Decode	无	无

类别		属性名	意义	IO	数据
共有	普通	T	Type(类型)	WR	Enum(String) : C/R/I/S {Close/RS232/I2C/SPI}
		When	When(条件)	WR	**RS232**: Enum(String):FB/EF/EE/D {FrameBegin/ErrorFrame/ECCError/Data} **I2C** : Enum(String): S/RS/STP/L/A/D/AD {Start/ReStart/Stop/Loss/Addr/Data/AddrData} **SPI**: Enum(String): C/I/CD/ID{CS/Idle/CS&Data/Idle&Data }
	总线设置	BS	Bus Status (总线状态)	WR	Enum(Integer): 0/1{OFF/ON}
		FMT	Display Format	WR	Enum(String): H/D/B/A{Hex/Dec/Binary/ASCII }

			(显示格式)		
		ET	Event Table (事件列表)	WR	Enum(Integer): 0/1{OFF/ON}
		PSW	Pseudo square wave(伪方波)	WR	Enum(Integer): 0/1{OFF/ON}
		VPOS	Vertical Pos	WR	Integer (Step)，以 6 为步进，范围：[-160,160]，以屏幕中心为零点，上正下负。
	触发	TGMode	Trigger Mode (触发模式)	WR	Enum(String): A/N{Auto/Normal}
		TGCP	Trigger Coupling (触发耦合)	WR	Enum(String) : D/A/H/L/N {DC/AC/HF Reject/LF Reject/Noise Reject}
I2C& SPI		DBV	Data : Bit Value Set	WR	64 有符号整数
RS232& SPI		**BSeq**	Bit Sequence (位/字节顺序)	WR	Enum(String): L/M{LSB/MSB}
RS232		Src	Source(信源)	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
		POL	Polarity(极性)	WR	Enum(String): N/P { Negative (负极性)/Positive(正极性) }
		BR	Baudrate(波特率)	WR	Integer : [120~5*10^6] 需要确定限制合理性
	DataSet	**DDB**	Data : Data Bit (数据位)	WR	Enum(Integer): 5/6/7/8 {5/6/7/8 bits}
		DV	Data : Value	WR	Integer (Step)，设置步进. 范围: [0, 2^ DDB - 1] DDB (5) -- [0,31]; DDB (6) -- [0,63]; DDB (7) -- [0,127]; DDB (8) -- [0,255];
		DSB	Data : Stop bits (停止位)	WR	Enum(Integer): 1/2 {1/2 bits}
		DP	Data : Parity (奇偶性)	WR	Enum(String): N/E/O{None/Even/Odd }
I2C		SCL	SCL	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
		SDA	SDA	WR	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
		ABW	Addr Bit Wide	WR	Enum(Integer): 7/10 {7/10 bits}
		Addr	AddrSet : Addr	WR	Integer (Step) : 范围: [0, 2^ ABW - 1] ABW (7) -- [0,127(0x7F)]; ABW (10) -- [0, 1023(0x3FF)];
		AIO	AddrSet : Direction/IO (读写)	WR	Enum(String): R/W {Read/Write }
		DWhen	Data:When	WR	Enum(String): =,>,< {=,>,< }

		(数据条件)		
SPI	**CSS**	CS Source	WR	Enum(Integer): 0/1/2/3{ CH1/ CH2/CH3/CH4}
	CSP	CS Polarity	WR	Enum(String): N/P { Negative (负极性)/Positive(正极性) }
	CLKS	SCLK Source	WR	Enum(Integer): 0/1/2/3{ CH1/ CH2/CH3/CH4}
	CLKE	SCLK Edge	WR	Enum(String) : R/F{ Rise/Fall }
	MOS	MOSI Source	WR	Enum(Integer): -1/0/1/2/3{ OFF/CH1/ CH2/CH3/CH4}
	MOP	MOSI Polarity	WR	Enum(String): N/P { Negative (负极性)/Positive(正极性) }
	MIS	MISO Source	WR	Enum(Integer): -1/0/1/2/3{ OFF/CH1/ CH2/CH3/CH4}
	MIP	MISO Polarity	WR	Enum(String): N/P { Negative (负极性)/Positive(正极性) }
	TGCH	Trigger Channel	WR	Enum(String): MO/MI/AO{ MOSI/MISO/Anyone }
	FL	Frame Length	WR	Enum(Integer): 4/8/16{ 4/8/16}
	STGT	SPI Trigger CS Timeout	WR	Integer (Step) : n ∈ [25,25*10^8] ; t = n * 4ns. Time ∈ [100ns,1s)

示例：

切换类型： “Decode @T:S;”

注意：

数据结构

1、数值调节型参数的数据结构：

```
typedef struct _cell_base_value{
    float Value;
    short Unit;
    short IsLimit;  // == 1,yes; =0, no!
}CellBaseValue; // 这样福格档时基档都可以使用该结构，FFT 切换 db 或者 v 都可以使用该结构读数!
```

2、时基档编码：

```
    typedef enum{
```

```
    TLEVEL_1NS = 0,
    TLEVEL_2NS,
    TLEVEL_5NS,
    TLEVEL_10NS,
    TLEVEL_20NS,
    TLEVEL_50NS,
    TLEVEL_100NS,
    TLEVEL_200NS,
    TLEVEL_500NS,
    TLEVEL_1US,
    TLEVEL_2US,
    TLEVEL_5US,
    TLEVEL_10US,
    TLEVEL_20US,
    TLEVEL_50US,
    TLEVEL_100US,
    TLEVEL_200US,
    TLEVEL_500US,
    TLEVEL_1MS,
    TLEVEL_2MS,
    TLEVEL_5MS,
    TLEVEL_10MS,
    TLEVEL_20MS,
    TLEVEL_50MS,
    TLEVEL_100MS,
    TLEVEL_200MS,
    TLEVEL_500MS,
    TLEVEL_1S,
    TLEVEL_2S,
    TLEVEL_5S,
    TLEVEL_10S,
    TLEVEL_20S,
    TLEVEL_50S,
    TLEVEL_100S,
    TLEVEL_200S,
}E_TIMEBASE_LEVEL;
```

编码表：

档位	编码
1ns	0
2ns	1
5ns	2
10ns	3
20ns	4
50ns	5

100ns	6
200ns	7
500ns	8
1μs	9
2μs	10
5μs	11
10μs	12
20μs	13
50μs	14
100μs	15
200μs	16
500μs	17
1ms	18
2ms	19
5ms	20
10ms	21
20ms	22
50ms	23
100ms	24
200ms	25
500ms	26
1s	27
2s	28
5s	29
10s	30
20s	31
50s	32
100s	33
200s	34

3、伏格档编码：

```
typedef enum {
    VLEVEL_1MV = 0,
    VLEVEL_2MV,
    VLEVEL_5MV,

    VLEVEL_10MV,
    VLEVEL_20MV,
    VLEVEL_50MV,

    VLEVEL_100MV,
```

```
    VLEVEL_200MV,
    VLEVEL_500MV,

    VLEVEL_1V,
    VLEVEL_2V,
    VLEVEL_5V,

    VLEVEL_10V,
    VLEVEL_20V,
} E_VOLTAGEBASE_LEVEL;
```

编码表：

档位	编码
1mV	0
2mV	1
5mV	2
10mV	3
20mV	4
50mV	5
100mV	6
200mV	7
500mV	8
1V	9
2V	10
5V	11
10V	12
20V	13

4、Integer 型参数数据：

即 int 类型，4 个字节。

选项型参数都使用 int 类型；

5、Integer(Step)型参数数据：

即 unsigned int 类型，占用 4 个字节；

6、Enum(String)型参数数据

选项型参数，以字符串类型表示，长度为 10Bytes,即读写的数据量是：

```
char v[10];
```

7、SCN? 的数据：

```
typedef struct _screen_info{
    short screen_width;       //屏幕宽度，像素值
    short screen_height;      //屏幕高度，像素值
    short x_grid_count;       //网格格数： 横向
    short y_grid_count;       //网格格数： 纵向
    short x_grid_pixels;      //单元格横向像素数
    short y_grid_pixels;      //单元格纵向像素数
    short y_min;              //数据坐标系，y坐标范围[y_min， y_max]
    short t_min;              //数据坐标系，x坐标范围[t_min， t_max]，以0起始
    short y_max;              //数据坐标系，y坐标范围[y_min， y_max]
    short t_max;              //数据坐标系，x坐标范围[t_min， t_max]，以0起始
}ScreenInfo;
```

8、参数测量统一数据包：

代码定义：

数据包为 MeaValue mp[50] （400Bytes），即固定50个测量参数的一维序列。各参数在序列中的位置由EMeaParam定义。

```
struct UnitParam {
    char Type;      //单位类型，比如Time、Freq等，由EType定义。
    char Scale;     //量级，比如k、n，p，M等，由EScale定义。
};

//@brief : 物理量数值
//@remark: 4Byte align -> 8Bytes
struct MeaValue {
    float      Value;
    UnitParam  Unit;
    char       IsValid;   //是否有效。  0 表示无效； 1表示有效。
    char       IsExist;   //是否存在。  0 表示存在； 1表示不存在。
};
```

内存表定义：

每个参数占用 8 个字节，一共 50 个参数空间，共 400 个字节，每个参数的内存模型是：

字段	Value	Unit.Type	Unit.Scale	IsValid	IsExist
占用	4Bytes	1Byte	1Byte	1Byte	1Byte
意义	物理值（小数）	单位类型编码	单位量级编码	是否有效	是否存在
数据		取值	取值	0:无效;1:有效	0:不存在；1:存在

物理单位编码

代码定义：

```
enum EScale : char {
    SCALE_p = -4,
    SCALE_n,
    SCALE_μ,
    SCALE_m,
    SCALE_STD = 0,
    SCALE_K,
    SCALE_M,
    SCALE_G,
    SCALE_T,
};

enum EType : char {
    TYPE_INVALID = -1,
    TYPE_FREQ,
    TYPE_TIME,
    TYPE_AREA,   //面积(Vs)
    TYPE_SAMPLERATE,//采样率（Sa/s）
    TYPE_POINT,  //点数（Sa）
    TYPE_VPP,    //峰峰值
    TYPE_VOLTAGE,//电压
    TYPE_CURRENT,//电流
    TYPE_DB,     //DB
    TYPE_VV,     //
    TYPE_PERCENT,//百分比
    TYPE_DEGREE, //度数
    TYPE_WATT,   //瓦特，功率
    TYPE_UNKNOWN,// 未知单位
};
```

编码表：

单位量级编码：

单位	编码
p	-4
n	-3
μ	-2
m	-1
标准单位	0
K	1
M	2
G	3
T	4

单位类型编码：

单位	编码
无效的类型	-1
频率（Hz）	0
时间（s）	1
面积（Vs）	2
采样率（Sa/s）	3
点数（Sa ， pkts）	4
峰峰值（VPP）	5
电压（V）	6
电流（A）	7
功率（DB）	8
VV	9
百分比（%）	10
度数（°）	11
瓦特（W）	12
未知单位（U）	13

参数编码

代码定义：

```
//@brief : 参数测量数据包的通用定义。
//@remark:
enum EMeaParam {
    MP_MAX = 0, //最大值
    MP_MIN,     //最小值
    MP_HIGH,    //High(Top)-高电平(顶端值)
    MP_MIDDLE,  //Middle-中间值
    MP_LOW,     //Low(Bottom) - 低电平(底端值)

    MP_PKPK,    //VPP-峰峰值
    MP_AMP,     //幅度
    MP_MEAN,    //平均值
    MP_CYCMEAN, //
    MP_RMS,     //均方根

    MP_CYCRMS,   //周期均方值
    MP_AREA,     //面积
    MP_CYCAREA,  //周期面积
    MP_OVERSHOOT,//过冲
    MP_PRESHOOT, //预冲

    MP_PERIOD,   //周期
    MP_FREQ,     //频率
    MP_RISE_TIME,//上升时间
    MP_FALL_TIME,//下降时间
    MP_PWIDTH,   //正脉宽

    MP_NWIDTH,   //负脉宽
    MP_PDUTY,    //正占空比
    MP_NDUTY,    //负占空比
    MP_RISEDELAY,//上升延时
    MP_FALLDELAY,//下升延时

    MP_PHASE,    //相位
    MP_FRR,      //
    MP_FRF,
    MP_FFR,
    MP_FFF,
```

```
        MP_LRF,
        MP_LRR,
        MP_LFR,
        MP_LFF,

        MP_BURST_WIDTH, //突发脉冲

        //
        //reserve section
        //
        //固定位50个参数
        MP_MAX_COUNT = 50,
    };
```

编码表：

参数	编码
最大值（MAX）	0
最小值（MIN）	1
高电平/顶端值(High/Top)	2
中间值（Middle）	3
低电平/底端值（Low）	4
峰峰值（PKPK）	5
幅度（AMP）	6
平均值（MEAN）	7
周期平均（Cycmean）	8
均方根（RMS）	9
周期均方根（Cycrms）	10
面积（AREA）	11
周期面积（Cycarea）	12
过冲（Overshoot）	13
预冲（Preshoot）	14
周期（Period）	15
频率（Freq）	16
上升时间（Rise Time）	17
下降时间（Fall Time）	18
正脉宽（PWidth）	19
负脉宽（NWidth）	20
正占空比（PDuty）	21
负占空比（NDuty）	22
上升延时（Rise Delay）	23
下降延时（Fall Delay）	24
相位（Phase）	25

FRR	26
FRF	27
FFR	28
FFF	29
LRF	30
LRR	31
LFR	32
LFF	33
突发脉冲（Burst Width）	34
预留	35~39

注意：

1、 本机型只支持上述表格中的部分参数；

2、 C#接口中的定义名称有所不同，放在 ucics.unit 和 ucics.mea 两个命名空间中。

9、光标测量(CursorMeasurePack)

```
typedef struct {
    UValue Ax;
    UValue Bx;
    UValue Bx_Ax;
    UValue Hz_Bx_Ax;
    UValue Ay;
    UValue By;
    UValue By_Ay;
    char   Reserver[8];
}CursorMeasurePack; //50bytes
```

10、时间数据

```
typedef struct _time {
    unsigned short Year;
    unsigned short Month;
    unsigned short Day;
    unsigned short Hour;
    unsigned short Minute;
    unsigned short Second;
}Time;
```

11、按键锁数据

```
typedef struct _KeyLedStat {
```

```
        char CH1, CH2, CH3, CH4, MATH, REF, RUN_STOP, Single;
    }KeyLedStat;
```

12、按键锁数据标记位：

```
enum KeyFlagsPos {
        IKEY_F1,
        IKEY_F2,
        IKEY_F3,
        IKEY_F4,
        IKEY_F5,

        IKEY_CH1,
        IKEY_CH2,
        IKEY_CH3,
        IKEY_CH4,

        IKEY_MATH,
        IKEY_REF,
        IKEY_HORIZON,
        IKEY_TRIGGER,
        IKEY_FORCE,
        IKEY_HELP,
        IKEY_MEASURE,
        IKEY_CURSOR,
        IKEY_ACQUIRE,
        IKEY_DISPLAY,
        IKEY_STORAGE,
        IKEY_UTILITY,
        IKEY_DECODE,
        IKEY_DEFAULT,
        IKEY_AUTOSET,
        IKEY_RUNSTOP,
        IKEY_SINGLE,
        IKEY_CLEAR,
        IKEY_PRINTSCREEN,
        IKEY_MENU,

        IKEY_OFFSET_LEFT,
        IKEY_OFFSET_RIGHT,
        IKEY_OFFSET_OK,

        IKEY_PRETRIG_LEFT,
```

```
        IKEY_PRETRIG_RIGHT,
        IKEY_PRETRIG_OK,

        IKEY_TRIGLEVEL_LEFT,
        IKEY_TRIGLEVEL_RIGHT,
        IKEY_TRIGLEVEL_OK,

        IKEY_VOLTS_LEFT,
        IKEY_VOLTS_RIGHT,
        IKEY_VOLTS_OK,

        IKEY_TIMEBASE_LEFT,
        IKEY_TIMEBASE_RIGHT,
        IKEY_TIMEBASE_OK,

        IKEY_SELECT_LEFT,
        IKEY_SELECT_RIGHT,
        IKEY_SELECT,
    };
```

13、定制参数测量数据结构：

```
struct MeaUserParams{
    char ID;   //-1 : 代表无测量参数；
    char CH;   //0 起始的通道号；   ID == -1 ：  CH无效。
}//2Bytes
```

引用文件：

文档中出现的相关数据结构定义，可以查看下面的文件：

Dso_base.h

Upo.h