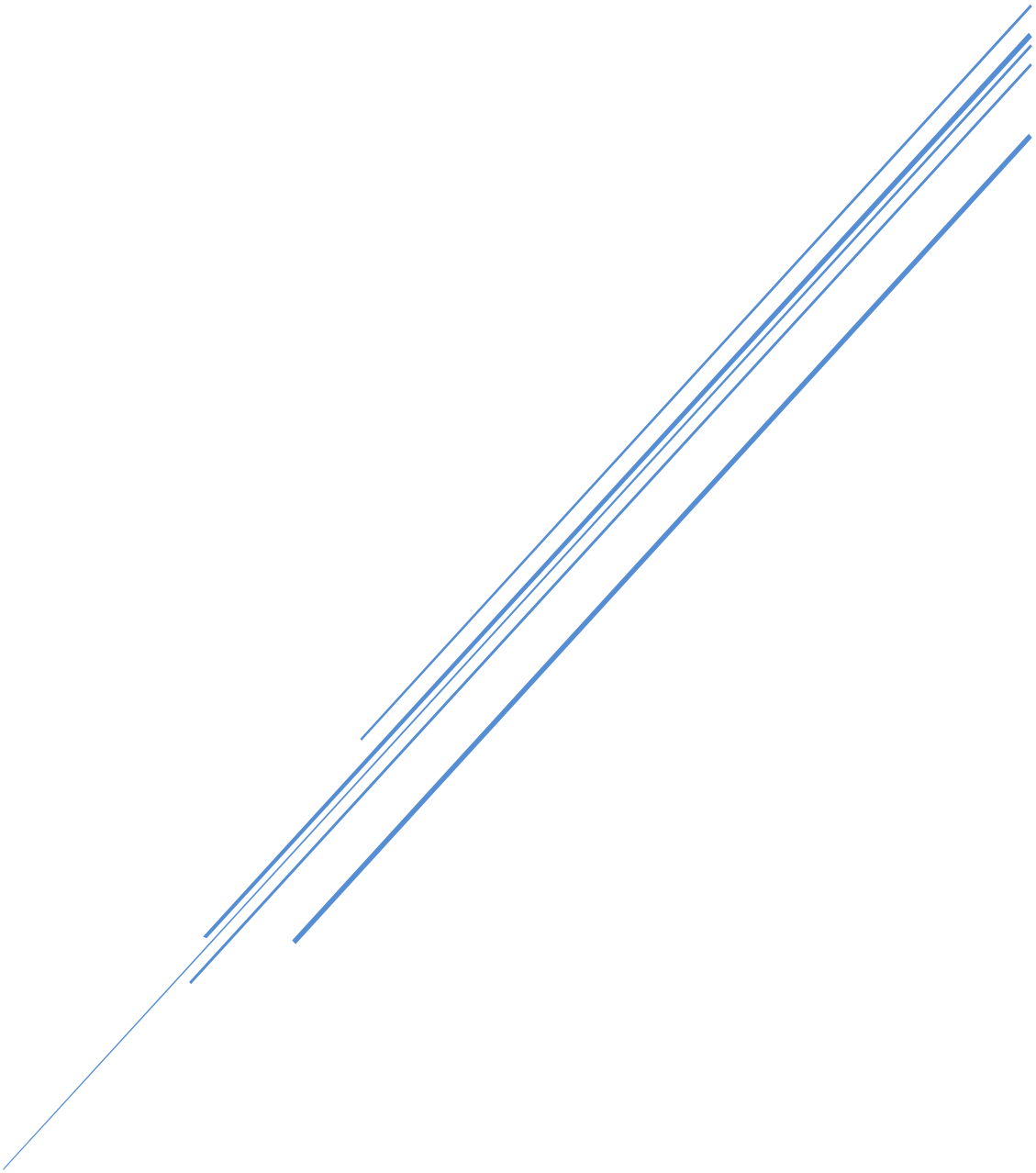
Programming Manual



Version

Version	Modified Item
V1.0	Official version
V1.0.1	Add interface mea:all?; this interface is the parameter measurement interface of oscilloscope.
V1.1	Delete additional interface to reduce interference

Statement

1、All the command of file must issue by interface UCI of uci.dll, the detailed description refer to《UCI Help Document.pdf》

2、All the character instruction is not case-sensitive；

3、the device address of UCI interface：

[C:DSO][D:DSO-C][T:USB][PID:0x834][VID:0x5656][EI:0x82][EO:0x4][CFG:1][I:0]

or

[C:DSO][D:DSO-C2][T:USB][PID:0x834][VID:0x5656][EI:0x82][EO:0x4][CFG:1][I:0]

use UCI interface to query device address

Keypad

Command Name	Command Parameter	Command Parameter Type
KEY	Key value	String : the abbreviation of keypad name

Keypad Name	Meaning
AT	AUTO
RS	RUN/STOP
TM	*MENU(trigger menu)
SG	SINGLE
FC	FORCE
HP	HELP
VM	*MENU(window setting menu)
MT	MATH
C1	CH1
C2	CH2
RF	REF

F1	F1
F2	F2
F3	F3
F4	F4
F5	F5
MOF	Menu ON/OFF
PS	PrScrn(screenshot)
COA	Coarse
MEA	Measure
CSR	Cursor
ACQ	Acquire
DISP	Display
STG	Storage
UTIL	Utility
FKN	Function Knob
FKNL	Function Knob Left Rotation
FKNR	Function Knob Right Rotation
VKN	CH1 Vertical Position Knob
VKNL	CH1 Vertical Position Knob Left Rotation
VKNR	CH1 Vertical Position Knob Right Rotation
VKN2	CH2 Vertical Position Knob
VKN2L	CH2 Vertical Position Knob Left Rotation
VKN2R	CH2 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
VBKNL	CH1 Voltage Base Knob Left Rotation
VBKNR	CH1 Voltage Base Knob Right Rotation
VBKN2L	CH2 Voltage Base Knob Left Rotation
VBKN2R	CH2 Voltage Base Knob Right Rotation
TBKNL	Time Base Knob Left Rotation
TBKNR	Time Base Knob Right Rotation

Attribute Name	Meaning	IO	Data
Lock	Lock the key	W	No Data
Unlock	Unlock the key	W	No Data
Lock?	Query key status	R	Integer(4B): 0 – unlock; 1 – lock

Example:

“KEY:AT;”
“KEY:C1;”
“KEY:MATH;”
“KEY:FKN;” -> push
“KEY:FKNL;” -> functional knob increase
“KEY:FKNR;” -> functional knob decrease
“KEY:FKNL@lock;” -> functional knob increase - locked

Notes:

Due to different device has different key name, the abbreviation is to avoid the difference!
Command with question mark should read by interface uci_Read.
Use local command to lock/unlock full qwerty:

Command Name	Meaning	IO	Data	Remark
Local	Lock Pad	W	Enum:0/1{remote status/local status}	Lock key when remote status

Screenshot

Command Name	Command Parameter	Command Parameter Type
PrtScn	Image data format	String: bmp { BMP file }

Example:

BMP file: “PrtScn:bmp;”

Notes:

1、 “PrtScn:bmp;” use interface uci_Read, **buffer area size can set to** 768512 bytes，read 16 bits BMP file.

uci_Read interface timeout can set as >=1s.

Configuration Data of Read and Write

Command Name	Command Parameter	Command Parameter Type
dconfig	none	

Example:

“dconfig;”

Description:

Please save the file by yourself. If time-out, it can set the long timing of overtime. It is suggested to read and write timeout set to 2000ms!

Interface:

Use interface uci_Read to read data, Use interface uci_Write to write data. Read data size can set to 1024,

the actual size get from returned value.

Automatic Reconnect

Command Name	Command Parameter	Command Parameter Type
Reconnect;	none	none

Example:

"Reconnect;"

Notes:

Use *uci_SendCommand* to send command.

If detected device is break in connect condition, it can send this command to reconnect.

Running Status

Command Name	Command Parameter	Data
Proc	**Running status setup**	Enum<string> : STOP/RUN/AUTO
Proc?	**Query running status**	Enum<string> : STOP/RUN/ARMD/READY/TRIGD/AUTO/SCAN/OVER/RESET { STOP/RUN/ARMED/READY/TRIGED/AUTO/SCAN/OVER/RESET }

Example:

Query: "Proc?;"

Setup: "Proc:Stop;" "Proc:AUTO;"

Notes:

The difference of "Proc:Stop", "Proc:Run" and button "RUN/STOP":

It is different from button function of "RUN/STOP". Hereon "RUN" put DSO at RUN status whatever the current condition is. "STOP" is the same.

Channel Control

The Basic (Common) Attribute:

Command Name	Command Parameter	Command Parameter Type	Remark
CH	Channel number	Start from 0 to number	**Channel number is for the command to**

		channel, integer value. 0:CH1;1:CH2	**point out which channel to process**

Attribute Name	Meaning	IO	Data
EN	ON/OFF	WR	Enum(Integer): 0/1{ON/OFF}
SEL	Select(set or query select status of the current channel)	WR	W： none{command type} R： Enum(Integer): 0/1{ON/OFF}
VP	Vertical position of channel	WR	Integer : position value， the middle of screen is 0, screen top is -100, the bottom is 100, each grid is 25.
HP	Pre-trigger position （horizontal adjustment）	WR	Integer : position value， screen left is 0， middle is 300， left is 600， left to minus right to plus, one grid is 50.
TB	Time Base（time base）	W	W: Enum(String) : Time value。
VB	Voltage Base	WR	W: Enum(String) : +/- { plus/minus one grid } or voltage value R : Double(8Bytes), power source in V unit
STZ	VPOS and HPos Set to zero	W	none{command type}

Support Voltage Base

--	10mV	100mV	1V	10V
2mV	20mV	200mV	2V	--
5mV	50mV	500mV	5V	--

Support Time Base

--	10ns	100ns	1us	10us	100us	1ms	10ms	100ms	1s	10s
2ns	20ns	200ns	2us	20us	200us	2ms	20ms	200ms	2s	20s
5ns	50ns	500ns	5us	50us	500us	5ms	50ms	500ms	5s	50s

Voltage base and the current base command written numeric value and unit directly, for example 100mv：“CH:0@VB:100MV”，“CH:0@TB:500US;”， unit is not case-sensitive.

Example：

Channel number：0/1/2/3/4{CH1/ CH2/ MATH/ REF-A/ REF-B }

Setup：

“CH:0@EN:1@VP:128@HP:350@VB:100MV@TB:500US;”

“CH:2@EN:1;” --- open MATH channel

“CH:2@VP:128;” --- adjust the vertical position of MATH channel to 128

"CH:2@SEL;" --- select MATH channel

Query：

"CH:0@VP;"

"CH:0@TB;"

"CH:0@TBV;"

Notes：

1、 The command protocol of basic command and physical channel:
The command name of basic command and physical channel both use "CH", format is "CH: channel id@Attribute:Value;" the basic command is suitable for physical channel, MATH,REF, distinguish by channel ID；

2、 CH command and channel number should be command parameter, otherwise it be the wrong command format;

3、 If selected channel does not open when executing command "@SEL;"， return error "channel doesn't open";

4、 If it is a physical channel（CH1/ CH2），"@VB" in fine tuning（set by "@ VD"） can only set by the way of "Enum(String) : +/- {plus/minus one grid }", get the actual voltage base information from command "@VB".

Physical Channel

Command Name	Command Parameter	Command Parameter Type	Remark
CH	Channel number	Start from 0 to number channel, integer value	**Channel number is for the command to point out which channel to process**

Attribute Name	Meaning	IO	Data
CP	Coupling(channel coupling)	WR	Enum(String): D/A/G{DC/AC/GND}
BW	BW limit(bandwidth limit)	WR	Enum(String): Enum(Integer): 0/1{ON/OFF}
VD	Volts/Div(coarse/fine tuning of amplitude)	WR	Enum(String): C/F{Coarse/Fine}
Probe	Probe（probe ratio）	WR	Enum(Decimals): 1/10/100/1000
Invert	Invert（inverse）	WR	Enum(Integer): 0/1{ON/OFF}

Example：

"CH:0@CP:D@BW:0@VD:C@Probe:1@invert:0;"

Notes：

1、"CH:0;" this is an invalid command， selected channel should use command "CH:0@SEL;"

2、 Read attribute, buffer area is 10 bytes at least, return character string, coding format of character string is based on compiler. If it is UNICODE, return character string of UNICODE coding.

Frequency Meter

Command Name	Command Parameter	Command Parameter Type	Remark
cmeter	--	--	--

Attribute Name	Meaning	IO	Data
EN	Enable(on/off)	WR	Enum(Integer<2Bytes>): 0/1
Freq?	Frequency value	R	Double<8Bytes>，in Hz unit. If value is -1，it presents <2Hz.

Example：

“cmeter@en:1;”

“cmeter@freq?;” -- read frequency value

Notes：

1、 Frequency meter measure the channel frequency that is corresponding to trigger source.

2、 Frequency meter is hardware measurement; the accuracy is higher than frequency parameter measurement in parameter measurement.

Parameter Measurement

Command Name	Command Parameter	Command Parameter Type	Remark
Mea	See the follow table	Character string	

Command Parameter	Meaning	IO	Data
All?	Packaging read measured parameter	R	Read oscilloscope united data structure. Data see ParaMeasureDataPacket：
freq	Freq	R	Double<8Bytes>
period	Period	R	Double<8Bytes>
rtime	Rise	R	Double<8Bytes>
ftime	Fall	R	Double<8Bytes>
pwidth	+Width	R	Double<8Bytes>
nwidth	-Width	R	Double<8Bytes>
oshoot	Overshoot	R	Double<8Bytes>
pshoot	Preshoot	R	Double<8Bytes>
pduty	+Duty	R	Double<8Bytes>
nduty	-Duty	R	Double<8Bytes>
avg	Average	R	Double<8Bytes>
vpp	Peak	R	Double<8Bytes>
rms	RMS	R	Double<8Bytes>
high	High	R	Double<8Bytes>
low	Low	R	Double<8Bytes>
mid	Middle	R	Double<8Bytes>
max	Max	R	Double<8Bytes>
min	Min	R	Double<8Bytes>
amp	Amplitude	R	Double<8Bytes>

Attribute Name	Meaning	IO	Data
src	Measurement source	WR	Enum(Integer<2Bytes>): 0/1 {CH1/CH2}

Example：

“mea:freq;” --- read frequency value；

“mea:all;” -- packaging read measured parameter.

“mea:all?;” -- packaging read all measured parameters（common）.

“mea@src:0;” --- set measurement source as CH1；

Notes：

When read parameter alone if return：

```
#define FLT_MAX         3.402823466e+38F        /* max value */
```

It presents this parameter is invalid.

Code Example：

Read parameter alone：

```
double dv = 0.0;
r = uci_ReadX(m_session, _T("mea:freq"), 1000, (byte*)&dv, sizeof(dv));
if (UCISUCCESS(r)) {
    printf("Freq = %f", dv);
}
```

Packaging read all parameters：

```
namespace cb = comAPICommon;//comAPICommon alias

    void Test_MeasureParams_DSO() {
        comAPICommon::MeaValue params[50];

        auto r = uci_ReadX(m_session, _T("mea:all?;"), 2000, (byte*)params, sizeof(params));
        ASSERT(r >= 0);

        PrintMeasureParams(params);
    };

    void PrintMeasureParams(comAPICommon::MeaValue* _p) {
        printf("\n++++++++++++++++++++++++++++++++++++++++++++++++++\n");
        PrintMeaParam(_p[cb::MP_FREQ], "freq");
        PrintMeaParam(_p[cb::MP_PERIOD], "period");

        PrintMeaParam(_p[cb::MP_NDUTY], "NDUTY");
        PrintMeaParam(_p[cb::MP_PDUTY], "PDUTY");
```

```
        PrintMeaParam(_p[cb::MP_MAX], "MAX");
        PrintMeaParam(_p[cb::MP_MIN], "MIN");

        PrintMeaParam(_p[cb::MP_PKPK], "VPP");
        PrintMeaParam(_p[cb::MP_RMS], "RMS");

        PrintMeaParam(_p[cb::MP_OVERSHOOT], "OVERSHOOT");
        PrintMeaParam(_p[cb::MP_PRESHOOT], "PRESHOOT");

        PrintMeaParam(_p[cb::MP_AMP], "AMP");

        PrintMeaParam(_p[cb::MP_RISE_TIME], "RISE_TIME");
        PrintMeaParam(_p[cb::MP_FALL_TIME], "FALL_TIME");
        printf("\n--------------------------------------------------");
    }

void PrintMeaParam(const comAPICommon::MeaValue& _p, const char * _name) {
    printf("%s = ", _name);
    if (_p.IsExist) {
        if (_p.IsValid) {
            printf("%f ", _p.Value);
        } else {
            printf("--");
        }
    } else {
        printf("NotExit");
    }

    printf(" %s%s\n", unit::uci_UnitFindScaleName(_p.Unit.Scale),
        unit::uci_UnitFindTypeName(_p.Unit.Type));
}
```

The structure and description of `MeaValue` see：ParaMeasureDataPacket

Capture Waveform Data

Command Name	Command Parameter	Command Parameter Type
Capture wave	Waveform format	String:.bin /.csv/.sav { binary system data /CSV file/built-in waveform file in oscilloscope }

Attribute Name	Meaning	IO	Data
CH	Channel(output channel ID)	W	Enum(Integer) : 0/1/2/3{ CH1/ CH2/CH3/CH4}
DT	Data Type(data format)	W	Enum(String) : ad/vol{ ADC initial data / voltage value }

Example：

"capture wave:.bin@CH:0@DT:AD;" -- read waveform data of CH1, AD value， keep in internal buffer area

"capture wave:.bin@CH:0@DT:vol;" -- read waveform data of CH1, voltage value， keep in internal buffer area

"capture wave:.csv@CH:0@DT:vol;" -- read waveform data of CH1, voltage value, keep in CSV file

Description：

1、 ".bin" and ".csv" file must include：@CH:0" and "@DT:" attribute， and ".sav" file must include @CH:0 attribute；；

2、 This waveform data is the initial data, not display data(（there are only a few hundred points)， it can use in waveform analysis task；

3、 It can use read parameter interface{ uci_Read or uci_ReadX }or read file interface{ uci_ReadToFile or uci_ReadToFileX }, the former keep data in buffer are of in-memory, the latter keep data in hard disk file（if it is UCI_DEMO.EXE program，please correct the corresponding suffix name.）

4、Waveform data format：

a) AD value ： 16 bits short type, a waveform point of waveform data；

b) VOL value，that is voltage value, zero point based on channel base line.

Trigger System

Command Name	Command Parameter	Command Parameter Type	Remark
trig	--	--	--

Attribute Name	Meaning	IO	Data
t	Type	W	Enum(String) : E/V /P{ EDGE/VIDEO/PLUSE WIDTH }
src	Trigger source	W	Enum(String) : c1/c2 /ext/ac/alt{ CH1/CH2/EXT/AC LINE/ALTER }
mode	Trigger mode	W	Enum(String) : A/ N /S{ AUTO/NORMAL/SINGLE }
cp	Trigger coupling	W	Enum(String) : D/A/H/L { DCl / AC /H Restrain /L H Restrain }
pos	Trigger mode	W	Ingeger<2Bytes> ： zero point based on channel base line，up to positive down to negative, each grid is 25；
st	Edge trigger slope type	W	Enum(String) : F / R /A{ Fall / Rise/Rise and Fall }

Example：

"trig@t:e;" -- set trigger type as edge trigger；

"trig@pos:25;" -- set trigger level higher one grid than base line, if voltage base is 1V, then trigger voltage is 1V；

【Basic】 Write Parameter

Parameter address in this command is digital coding; it need refer to the command coding definition in file include\CMD_COMVer2.h.
This command is to compatible the old protocol.

Command Name	Command Parameter	Command Parameter Type	Remark
wp	--	--	--

Attribute Name	Meaning	IO	Data
CH	Channel number	W	Enum(Integer<2Bytes>): 0/1 {CH1/CH2}
addr	Parameter address （command number）	W	see the definition of *COMMON_CMD* and *CHAN_CMD* in CMD_COMVer2.h

Example：

“WP@CH:0@ADDR:400@v:2;” -- set sampling rate as average mean sampling

```
enum E_ACQ_MODE
{
    ACQ_MODE_NORMAL = 0,
    ACQ_MODE_PEAK,
    ACQ_MODE_AVERAGE,
};
```

Notes：

“@CH” and “@addr” must use at the same time. As long as it can be format as character string, it is recommend to use “@v:” to set parameter.

Write parameter use uci_Write, uci_WriteX or uci_FormatWrite ;

【Basic】 Read Parameter

Parameter address in this command is digital coding; it need refer to the command coding definition in file include\CMD_COMVer2.h.
This command is to compatible the old protocol.

Command Name	Command Parameter	Command Parameter Type	Remark
rp	--	--	--

Attribute Name	Meaning	IO	Data
CH	Channel number	R	Enum(Integer<2Bytes>): 0/1 {CH1/CH2}
addr	Parameter address （command number）	R	see the definition of *COMMON_CMD* and *CHAN_CMD* in CMD_COMVer2.h

Example：

“RP@CH:0@ADDR:400:;” -- read sampling mode

Notes：

"@CH" and "@addr" must use at the same time.

Read parameter use interface uci_Read or uci_ReadX；

Read data size, see the annotation in CMD_COMVer2.h.

Case

1、Measure signal frequency and amplitude parameter

Method

1. Use parameter measure function to test signal, including frequency, amplitude, cycle etc. The detailed see Measure function: Parameter measurement
2. Use UTILITY->frequency meter to test signal frequency：Frequency Meter

Difference：

1. MEASURE is software measurement, frequency meter is hardware measurement, frequency meter test accuracy is higher；
2. Frequency meter test the signal source that is the corresponding signal of trigger source; parameter measurement test signal source that is source only can set in parameter measurement.

2、 After the trigger channel is triggered, capture waveform data of the channel that is to be tested, and then deep analysis

For example：CH1 is the signal test channel which to connect with test source. CH2 is the trigger channel which to connect with trigger signal, it is usually user-defined. The aim is to find trigger timing, to capture waveform block in CH1 in corresponding time, subtract data and then analyze it.

Common model：

1、 trigger mode set as single trigger, command："trig@mode:s;"(use write parameter interface);

2、 running status as RUN，command："proc:run;" (use write parameter interface);

3、 get running status, check running status whether is STOP, if it is STOP, that is means it's been triggered, command "proc？" (use read and write interface)

4、 If it has been triggered, subtract waveform data, command："capture wave:.bin@CH:0@DT:vol;"

Repeat 2-4 steps to complete this model.

Appendix

ParaMeasureDataPacket:

Code Definition

Data packet is `MeaValue mp[50] （400Bytes）`, that is fixed 50 test parameter in one dimensional sequence. `EMeaParam` defines the position of each parameter in the sequence.

```
struct UnitParam {
    char Type;      // unit type，like Time,Freq etc.，define by EType
    char Scale;     // magnitude，like k、n，p，M etc.，define by EScale
};

//@brief : physical quantity value
//@remark: 4Byte align -> 8Bytes
struct MeaValue {
    float     Value;
    UnitParam  Unit;
    char      IsValid;   // whether is valid. 0 presents invalid；  1 presents valid.
    char      IsExist;   // whether is exist. 0 presents existence；  1 presents absent.
};
```

Definition of In-memory Table

Each parameter takes 8 bits, 50 parameter space, and 400 bytes in total.

In-memory model of each parameter：

Fields	Value	Unit.Type	Unit.Scale	Is Valid	Is Exist
Byte	4Bytes	1Byte	1Byte	1Byte	1Byte
Meaning	Physical value （decimal）	Unit type coding	Unit quantity coding	Whether is valid	Whether is existence
Data		Value	Value	0:invalid;1:valid	0:absent; 1: existence

Encoding Physical Unit

Code Definition

```
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,      // area (Vs)
        TYPE_SAMPLERATE,// sampling rate（Sa/s）
        TYPE_POINT,     // count（Sa）
        TYPE_VPP,       // peak-to-peak value
        TYPE_VOLTAGE,// voltage
        TYPE_CURRENT,// current
        TYPE_DB,        //DB
        TYPE_VV,        //
        TYPE_PERCENT,// percentage
        TYPE_DEGREE, // degree
        TYPE_WATT,      // watt，power
        TYPE_UNKNOWN,// unknown unit
};
```

Coding table

Unit Coding：

Unit	Coding
p	-4
n	-3
μ	-2
m	-1
Standard unit	0
K	1
M	2
G	3
T	4

Unit Type Coding：

Unit	Coding
Invalid type	-1
frequency（Hz）	0
time（s）	1
area（Vs）	2
sampling rate（Sa/s）	3
count（Sa , pkts）	4
peak-to-peak value（VPP）	5
voltage（V）	6
current（A）	7
power（DB）	8
VV	9
percentage（%）	10
degree（° ）	11
Watt（W）	12
unknown unit（U）	13

Parameter Coding

Coding Definition

```
//@brief : the common definition of parameter measurement data packet
//@remark:
enum EMeaParam {
    MP_MAX = 0, // maximum value
    MP_MIN,     // minimum value
    MP_HIGH,    //High(Top)- high level(top value)
    MP_MIDDLE,  // Middle value
    MP_LOW,     //Low(Bottom) - lower level(bottom value)

    MP_PKPK,    //VPP--peak-to-peak value
    MP_AMP,     // amplitude
    MP_MEAN,    // mean value
    MP_CYCMEAN, //
    MP_RMS,     // root mean square

    MP_CYCRMS,   // cycle root mean square
    MP_AREA,     // area
    MP_CYCAREA,  // cycle area
    MP_OVERSHOOT,// overshoot
    MP_PRESHOOT, // preshoot

    MP_PERIOD,   // period
    MP_FREQ,     // frequency
    MP_RISE_TIME,// rise time
    MP_FALL_TIME,// fall time
    MP_PWIDTH,   // positive pulse width

    MP_NWIDTH,   // negative pulse width
    MP_PDUTY,    // positive duty ratio
    MP_NDUTY,    // negative duty ratio
    MP_RISEDELAY,// rise delay
    MP_FALLDELAY,// fall delay

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

```
    MP_LRF,
    MP_LRR,
    MP_LFR,
    MP_LFF,

    MP_BURST_WIDTH, // burst

    //
    //reserve section
    //
    // fixed 50 parameters
    MP_MAX_COUNT = 50,
};
```

Coding Table

Parameter	Coding
Maximum value	0
Minimum value	1
High level/ top value (High/Top)	2
Middle value	3
High level/bottom value（Low）	4
Peak-to-peak value（PKPK）	5
Amplitude（AMP）	6
Mean value（MEAN）	7
Cycle mean（Cycmean）	8
Root mean square（RMS）	9
Cycle root mean square（Cycrms）	10
Area（AREA）	11
Cycle area（Cycarea）	12
Overshoot	13
Preshoot	14
Cycle	15
Frequency（Freq）	16
Rise Time	17
Fall Time	18
Positive pulse width（PWidth）	19
Negative pulse width（NWidth）	20
Positive duty ratio（PDuty）	21
Negative duty ratio（NDuty）	22
Rise delay	23

Fall delay	24
Phase	25
FRR	26
FRF	27
FFR	28
FFF	29
LRF	30
LRR	31
LFR	32
LFF	33
Burst（Burst Width）	34
Reserved	35~39

Notes：

1、 This model only support a part of parameter in the above table；

2、 Different definition name in C#Interface， put it in the name space `ucics.unit` and `ucics.mea`.