T860 系统

AWG 和 DIGITAL 功能

说明手册

绍兴宏邦电子科技有限公司

T860 系列板卡提供 AWG 和 DIGITAL 功能□并可实现不同板卡的同步功能□这提供给用户更为方便和灵活的使用□本章讲解如何实现不同 SLOT 之间的板卡同步功能□各板卡的 AWG 和 DIGITAL 功能详见相应的文档□

目前实现 AWG 和 DIGITAL 功能的板卡□

AWG	DVI3□OVI2
DIGITAL	DVI3□OVI2□OVI

注□后续所有板卡的 AWG 和 DIGITAL 功能都会陆续开放□

system_awg_create_sine_data

建立一个正弦波数据组□设置该正弦波的数据长度□峰峰值□直流偏置□相位以及周期数等信息□

格式

```
void system_awg_create_sine_data(double *awg_data,unsigned short size,double vpp=1,double voffset=0,double phase=0,unsigned short cycles=1);
```

有效参数

awg_data

double 型指针,存储 AWG 波形数据的首地址

size

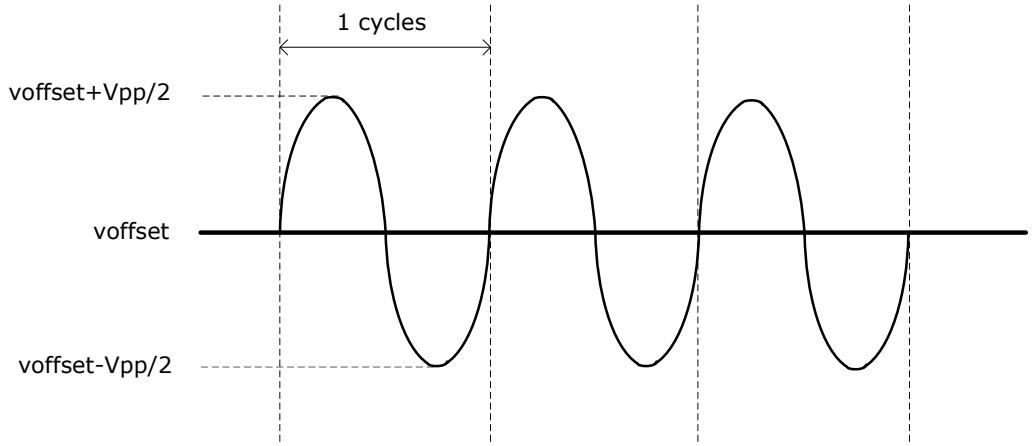
AWG 波形数据长度

vpp

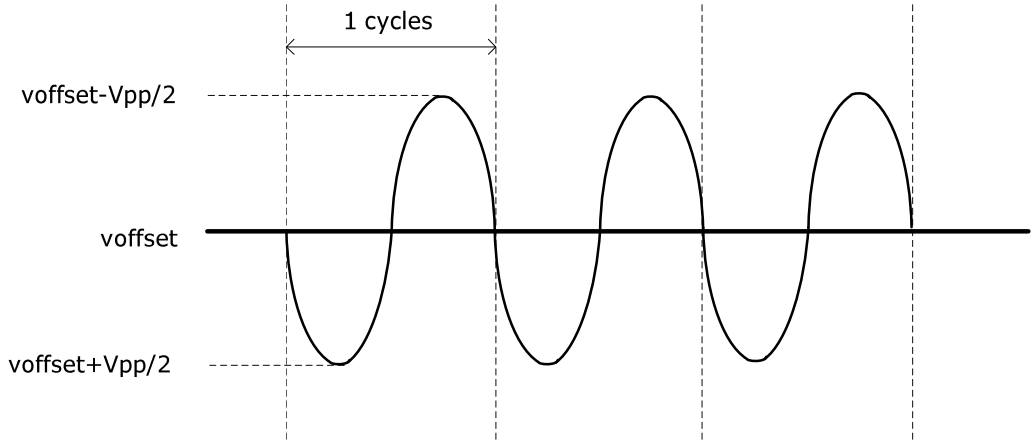
正弦波的峰峰值□该参数默认为 1□

该参数可正可负□下图为一个周期正弦波分别在 Vpp 为正值和负值时的波形图□二者相位相差 180 度□

当 Vpp 为正值时□个参数关系及波形如下□



当 Vpp 为负值时□个参数关系及波形如下□



voffset

正弦波的直流偏置□该参数默认为 0□

phase

正弦波的相位□单位□度□该参数默认为 0□

cycles

正弦波的周期数□该参数默认为 1□

应用范例

设置一个正弦波数据组□首地址从 0 开始□数据长度为 100□峰峰值为 4V□直流偏置为 1V□相位为 90°□周期数为 1□

```
double awg_pattern[100]={0.0};
double  Vpp=4;
system_awg_create_sine_data(&awg_pattern[0],100,Vpp,1,90,1);
```

system_awg_create_triangle_data

建立一个三角波数据组□设置该三角波的数据长度□峰峰值□直流偏置□相位以及周期数等信息□

格式

```
void system_awg_create_triangle_data(double *awg_data,unsigned short size,double vpp=1,double voffset=0,double phase=0,unsigned short cycles=1);
```

有效参数

awg_data

double 型指针,存储 AWG 波形数据的首地址

size

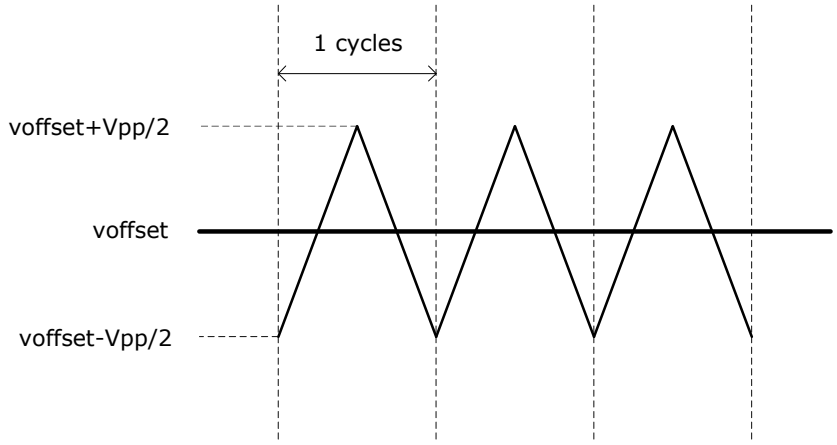
AWG 波形数据长度

vpp

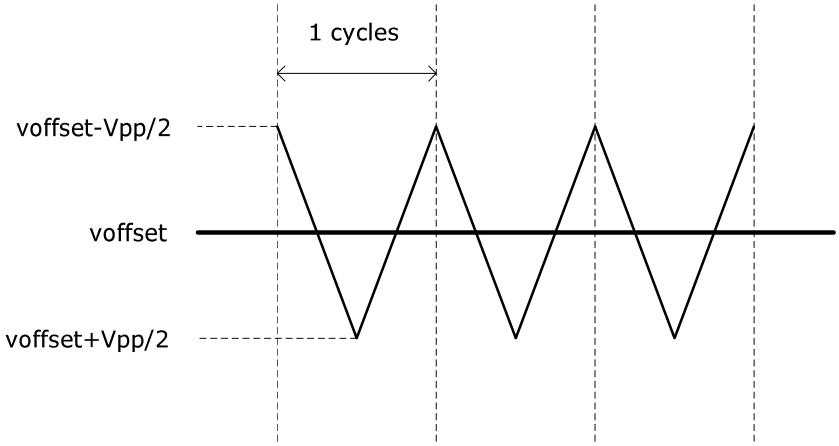
三角波的峰峰值□该参数默认为 1□

该参数可正可负□下图为一个周期正弦波分别在 Vpp 为正值和负值时的波形图□二者相位相差 180 度□

当 Vpp 为正值时□个参数关系及波形如下□



当 Vpp 为负值时□个参数关系及波形如下□



voffset

三角波的直流偏置□该参数默认为 0□

phase

三角波的相位□单位□度□该参数默认为 0□

cycles

三角波的周期数□该参数默认为 1□

应用范例

设置一个三角波数据组□首地址从 0 开始□数据长度为 100□峰峰值为-5V□直流偏置为 1V□相位为 0°□周期数为 2□

```
double awg_pattern[100]={0.0};
double  Vpp=-5;
system_awg_create_triangle_data(&awg_pattern[0],100,Vpp,1,0,2);
```

system_awg_create_square_data

建立一个方波数据组□设置该方波的数据长度□峰峰值□直流偏置□占空比以及周期数等信息□

格式

void system_awg_create_square_data(double *awg_data,unsigned short size,double vpp=1,double

```
voffset=0,double duty_cycle=0.5,unsigned short cycles=1);
```

有效参数

awg_data

double 型指针,存储 AWG 波形数据的首地址

size

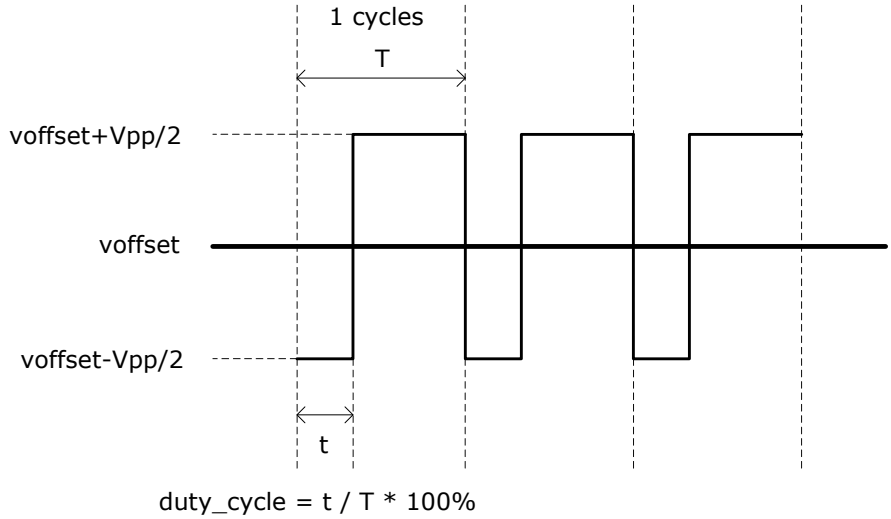
AWG 波形数据长度

vpp

方波的峰峰值□该参数默认为 1□

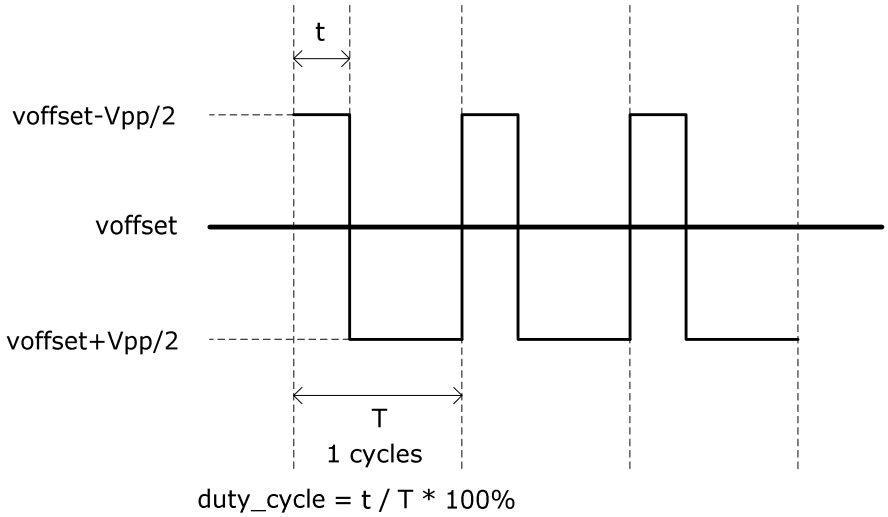
该参数可正可负□下图为一个周期正弦波分别在 Vpp 为正值和负值时的波形图□二者相位相差 180 度□

当 Vpp 为正值时□个参数关系及波形如下□



duty_cycle = t / T * 100%

当 Vpp 为负值时□个参数关系及波形如下□



duty_cycle = t / T * 100%

voffset

方波的直流偏置□该参数默认为 0□

duty_cycle

方波的占空比□单位□%□该参数默认为 50□

cycles

方波的周期数□该参数默认为 1□

应用范例

设置一个方波数据组□首地址从 0 开始□数据长度为 100□峰峰值为 4V□直流偏置为 0V□占空比为 25□周期数为 1□

```
double awg_pattern[100]={0.0};
double  Vpp=4;
system_awg_create_square_data(&awg_pattern[0],100,Vpp,0,25,1);
```

system_awg_create_ramp_data

建立一个斜波数据组□设置该斜波的数据长度□起始值□结束值以及周期数等信息□

格式

```
void system_awg_create_ramp_data(double *awg_data,unsigned short size,double start_value=1,double stop_value=0,unsigned short cycles=1);
```

有效参数

awg_data

double 型指针,存储 AWG 波形数据的首地址

size

AWG 波形数据长度

start_value

斜波的起始值□该参数默认为 1□

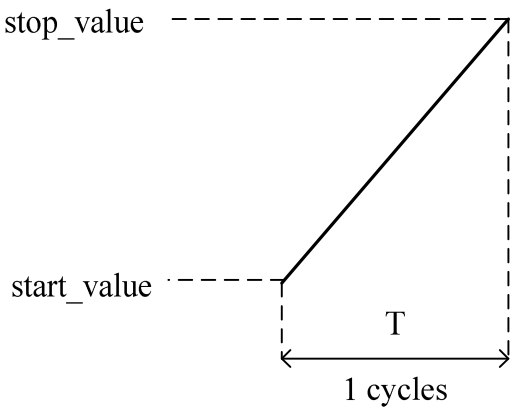
stop_value

斜波的结束值□该参数默认为 0□

cycles

斜波的周期数□该参数默认为 1□

下图表示了该函数各个参数的关系□该图 start_value < stop_value□因此为一个上升的斜波□



设置一个斜波数据组□首地址从 0 开始□数据长度为 100□周期数为 1□起始值为 1V□结束值为 5V□使用 SoftView 得到的波形请参考 2.14 或 3.14 的例 1□

应用范例

设置一个斜波数据组□首地址从 0 开始□数据长度为 100□起始值为-1□结束值为 3□周期数为 1□

```
double awg_pattern[1000]={0.0};
system_awg_create_ramp_data(&awg_pattern[0],100,-1,3,1);
```

system_awg_create_trapezoid_data

建立一个梯形波数据组□设置该梯形波的数据长度□下底值□上底值□腰的数据长度□上底的数据长度□腰的数据长度及周期数等信息□

格式

```
void system_awg_create_trapezoid_data(double *awg_data,unsigned short size,double bottom_value=0,double high_value=1,unsigned short size1=0,unsigned short size2=0,unsigned short size3=0,unsigned short cycles=1);
```

有效参数

awg_data
double 型指针,存储 AWG 波形数据的首地址

size
AWG 波形数据长度

bottom_value
梯形波下底值□该参数默认为 0□

high_value
梯形波上底值□该参数默认为 1□

size1
梯形波腰的数据长度□该参数默认为 0□

size2
梯形波上底的数据长度□该参数默认为 0□

size3
梯形波腰的数据长度□该参数默认为 0□

cycles
梯形波的周期数□该参数默认为 1□

应用范例

设置一个脉冲波数据组□首地址从 0 开始□数据长度为 100□周期数为 1□下底值为 0V□上底值为 5V□上升及下降各设 20 个数据□

```
    double awg_pattern[100]={0};
    system_awg_create_trapezoid_data(&awg_pattern[0],100,0,5,20,
60,20);
```

system_awg_sync_enable

使能同步AWG□

格式

```
    void system_awg_sync_enable(unsigned short
board_no_1=0xffff,unsigned short
board_no_2=0xffff,unsigned short
board_no_3=0xffff,.....unsigned short
board_no_21=0xffff,);
```

有效参数

填写需要使能的board_no□一共可以填21路源□

board_no_1~board_no_21	BOARD_1
	BOARD_2
	.....
	BOARD_21

例□使能 11,14 号槽两块板卡的 AWG 波形□如下□

```
    system_awg_sync_enable(BOARD_11,BOARD_14);
```

system_awg_start

AWG同步启动□运行完这个函数之后□AWG扫描和测量才开始运行□每次运行前都必需配合awg_select□system_awg_sync_enable和所有同步板卡的awg_enable一起使用□

格式

```
void system_awg_start(UNIT32 delay_time=0xffffffff);
```

有效参数

delay_time

设置延时时间□范围□0~4294967294□单位 1us□AWG 输出 delay_time 之后□系统允许 system_awg_start 之后的操作开始运行□delay_time 不填的时候□delay_time 默认为 AWG 扫描时间或同步测量时间中较长的时间为准□只有在 AWG 扫描完毕□同时测量也完毕之后□系统才允许 ov2_awg_start 之后的操作开

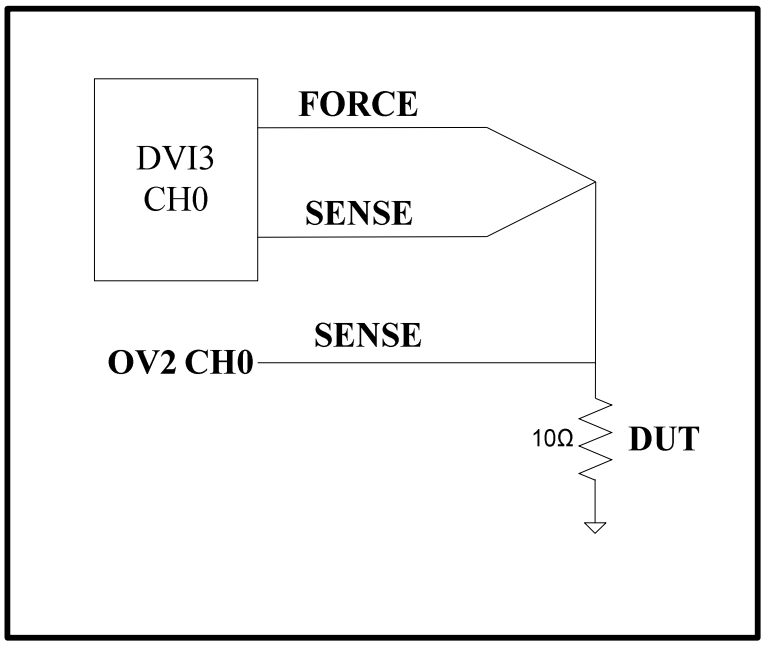
应用范例

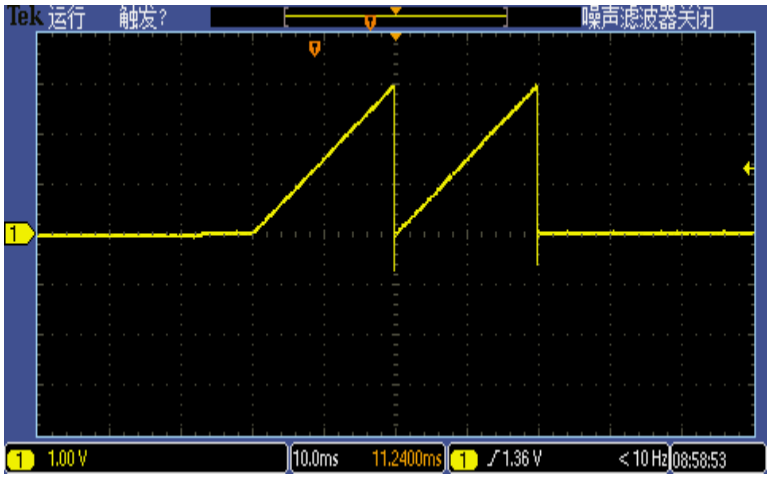
AWG 波形持续 5mS□如下□

```
dvi_11->dvi_awg_enable();
ov2_14->ov2_awg_enable();
system_awg_sync_enable(BOARD_11,BOARD_14);
system_awg_start(5000);
```

AWG 程序设计范例

两块板卡 DVI3 和 OV2 间的 AWG 同步□DVI3 作为源□OV2 进行扫描测量□

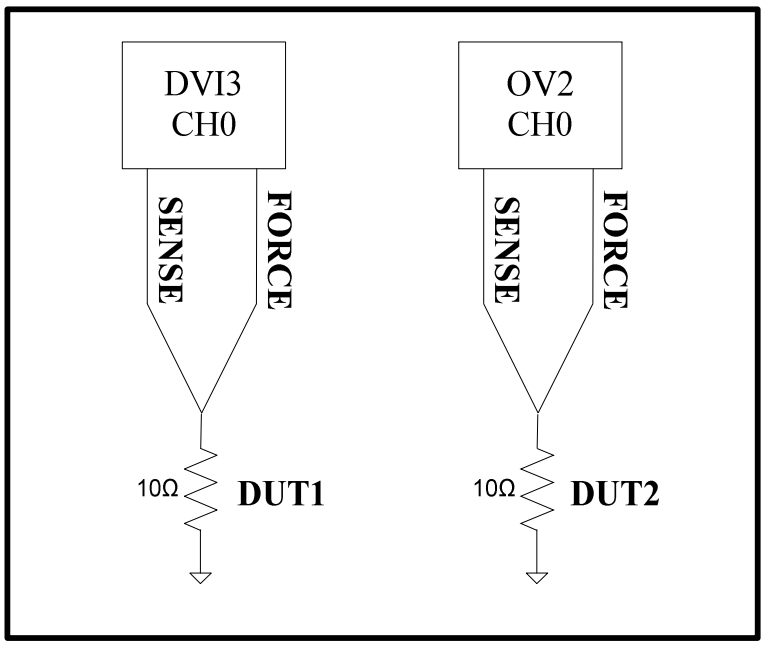




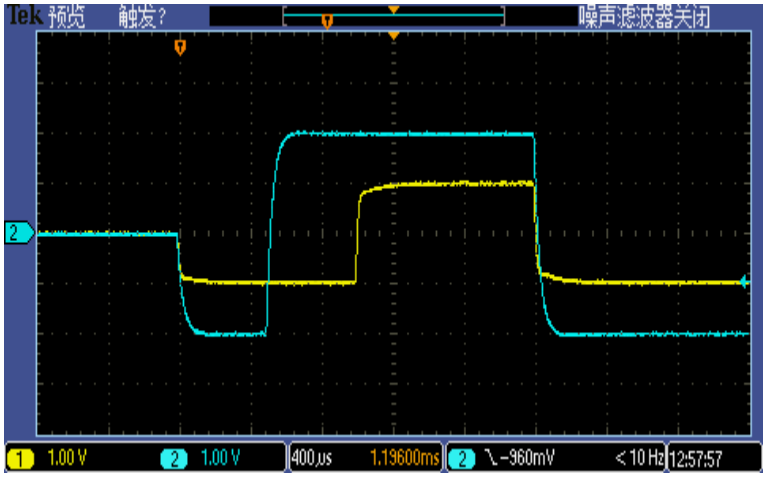

```
void NEW_FUNC(test_function& func)
{
    // The two lines below must be the first two in the function.
    NEW_FUNC_params *ours;
    ours = (NEW_FUNC_params *)func.params;

    double dvi_meas[100]={0},ov2_meas[100]={0};
    float result[2]={0};
    int size = 100;//AWG 波形数据的长度
```

```
    double dvi_awg_pattern[100]={0};

    dvi_11->init();
    ov2_14->init();
    dvi_11->set_meas_mode(DVI_CHANNEL_0,DVI_MEASURE_CURRENT);
    dvi_11->set_voltage(DVI_CHANNEL_0,4.5,RANGE_5_V);
    dvi_11->set_current(DVI_CHANNEL_0,1.0e-6f,AMP_2);
    ov2_14->set_meas_mode(OV2_CHANNEL_0,DVI_MEASURE_VOLTAGE);
    ov2_14->set_voltage(OV2_CHANNEL_0,0,RANGE_5_V);
    delay(2);
    dvi_11->dvi_measure(100,200,dvi_meas,DVI_MEAS_AWG,100);
    ov2_14->ov2_measure(OV2_MEAS_0,100,200,ov2_meas,OV2_MEAS_AW
G,100);

    system_awg_create_ramp_data(&dvi_awg_pattern[0],100,0,0.3,1);
    dvi_11->dvi_awg_load_current(DVI_CH_0,AMP_2,FALSE,0,size-1,dvi_aw
g_pattern);
    dvi_11->dvi_awg_select_current(DVI_CH_0,AMP_2,FALSE,0,size-1,size-1,
200,DVI_AWG_LOOP);

    dvi_11->dvi_awg_enable();
    ov2_14->ov2_awg_enable();
    system_awg_sync_enable(BOARD_11,BOARD_14);
    system_awg_start();
    result[0]=dvi_11->dvi_get_measure_result(DVI_MEAS_MAX,0,size-1);
    result[1]=ov2_14->ov2_get_measure_result(OV2_MEAS_0,OV2_MEAS_M
AX,0,size-1);
    delay(5);
    dvi_11->dvi_awg_stop(DVI_CH_0);//停止 fovi0 的 AWG 循环运行
    ov2_14->ov2_awg_stop(OV2_CH_0);//停止 fovi0 的 AWG 循环运行

    delay(50);
    dvi_11->init();
    ov2_14->init();
}
```

两块板卡 DVI3 和 OV2 间的 AWG 同步输出□



黄色□DVI_CH0□蓝色□OV2_CH0




```
void NEW_FUNC(test_function& func)
{
    // The two lines below must be the first two in the function.
    NEW_FUNC_params *ours;
    ours = (NEW_FUNC_params *)func.params;

    dvi_11->init();
    ov2_14->init();
    dvi_11->set_voltage(DVI_CHANNEL_0,0,RANGE_5_V);
    dvi_11->set_current(DVI_CHANNEL_0,0.5f,AMP_2);
    ov2_14->set_voltage(OV2_CHANNEL_0,0,RANGE_5_V);
```

```
    ov2_14->set_current(OV2_CHANNEL_0,0.5f,RANGE_500_MA);
    delay(2);

    system_awg_create_square_data(&dvi_awg_pattern[0],100,2,0,50);
    system_awg_create_square_data(&ov2_awg_pattern[0],100,4,0,25);

    dvi_11->dvi_awg_load_voltage(DVI_CH_0,RANGE_5_V,POSITIVE_V_OUT,
0,size-1,dvi_awg_pattern);
    dvi_11->dvi_awg_select_voltage(DVI_CH_0,RANGE_5_V,POSITIVE_V_O
UT,0,size-1,0,20,DVI_AWG_LOOP);
    ov2_14->ov2_awg_load_voltage(OV2_CH_0,RANGE_5_V,0,size-1,ov2_a
wg_pattern);
    ov2_14->ov2_awg_select_voltage(OV2_CH_0,RANGE_5_V,0,size-1,0,20,
OV2_AWG_LOOP);

    dvi_11->dvi_awg_enable();
    ov2_14->ov2_awg_enable();
    system_awg_sync_enable(BOARD_11,BOARD_14);
    system_awg_start();
    dvi_11->dvi_awg_stop(DVI_CH_0);//停止 fovi0 的 AWG 循环运行
    ov2_14->ov2_awg_stop(OV2_CH_0);//停止 fovi0 的 AWG 循环运行

    delay(50);
    dvi_11->init();
    ov2_14->init();
}
```