晟矽微电 应用笔记

MS32F031

Timer PWM out

AN22002

V1.0

目 录

1 适用范围

本文档适用于 MS32F031A6，Timer PWM 输出。

2 Timer PWM out 测试及分析

2.1 程序（Timer1、通道 1、PWM 模式 1、向上计数）

例程“MS32F0x1_Periph_Lib_Example\proj\MS32F031_EV\TIMER\TIM1_CH1_PWMwN”为基础进行测试；例程 Timer1、通道 1、PWM 模式 1、向上计数；在例程基础上修改：占空比初始化 25%，main 函数中不调节占空比。

TIMER_CFG.c

```
91    TIM_OC_InitStruct.CompareValue =  ((( MS32_TIM_GetAutoReload(TIM1) + 1 ) * 25 ) / 100);    // set duty cycle
```

TIMER_CFG.c | main.c

```
81
82        //MS32_TIM_OC_SetCompareCH1(TIM1, (MS32_TIM_GetAutoReload(TIM1)+1) * Timer1_PWM_Duty / 100 );    // update duty cycle
83        printf("\r\n--Inf:Timer1 CH1 PWM Duty %d",Timer1_PWM_Duty);
```

2.2 关键信号

2.2.1 OCxREF

寄存器 TIM1_CCMR1：

BIT[6:4]	OC1M	输出比较模式 1 (Output Compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的行为，而 OC1REF 决定了 OC1、OC1N 的电平。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用（此模式用于产生一个时基）； 001：匹配时，设置通道 1 为有效电平。当 TIMx_CNT = TIMx_CCR1 时，强制 OC1REF 为高。 010：匹配时，设置通道 1 为无效电平。当 TIMx_CNT = TIMx_CCR1 时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，当 TIMx_CNT<TIMx_CCR1 时，通道 1 为有效电平(OC1REF=1)，否则为无效电平(OC1REF=0)； - 在向下计数时，当 TIMx_CNT>TIMx_CCR1 时，通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，当 TIMx_CNT<TIMx_CCR1 时，通道 1 为无效电平，否则为有效电平； - 在向下计数时，当 TIMx_CNT>TIMx_CCR1 时，通道 1 为有效电平，否则为无效电平。 *注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位）并且 CC1S=00（该通道配置成输出），则该位不能被修改。* *注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。* *注 3:在通道有互补输出时,此位被预装载。如果 TIMx_CR2 中的 CCPC 位*

例程中 OCxREF 信号（PWM1，向上计数，MCU 内部）：

2.2.2 CCxP，CCxNP

寄存器 TIM1_CCER：

BIT[1]	CC1P	捕获/比较 1 输出极性（Capture/Compare 1 output polarity） **CC1 通道配置为输出：** 0：OC1 高电平有效； 1：OC1 低电平有效。

```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_HIGH;
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF（或 OCxREF 取反）有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_HIGH;

// 1，MS32_TIM_OCPOLARITY_LOW：OCxREF 有效时输出低电平，无效时输出高电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_LOW;
// 1，MS32_TIM_OCPOLARITY_LOW：OCxREF（或 OCxREF 取反）有效时输出低电平，无效时输出高电平
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_LOW;
```

2.2.3 CCxE，CCxNE

寄存器 TIM1_CCER：

BIT[0]	CC1E	捕获/比较 1 输出使能(Capture/Compare 1 output enable) **CC1 通道配置为输出：** 0：关闭 - OC1 未激活， OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 这些位的功能。 1：开启 - OC1 信号输出到对应的输出引脚，其输出依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 这些位的功能。

```
// 1：信号输出到引脚
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

2.3 MOE：1，OSSR：0

```
// MOE：1
MS32_TIM_EnableAllOutputs(TIM1);

// OSSR：0
  TIM_BDTRInitStruct.OSSRState = MS32_TIM_OSSR_DISABLE;
```

引脚信号输出：

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止（不由定时器驱动） OCx=0,	输出禁止（不由定时器驱动） OCxN=0
		0	0	1	输出禁止（不由定时器驱动） OCx=0	OCxREF + 极性 OCxN=OCxREF xor CCxNP
		0	1	0	OCxREF + 极性 OCx=OCxREF xor CCxP	输出禁止（不由定时器驱动） OCxN=0
		0	1	1	OCxREF + 极性 + 死区	OCxREF 互补(非 OCxREF) +极性 + 死区
		1	0	0	输出禁止（不由定时器驱动） OCx=CCxP	输出禁止（不由定时器驱动） OCxN=CCxNP
		1	0	1	Off-State (输出使能但无效态） OCx=CCxP	OCxREF + 极性 OCxN=OCxREF xor CCxNP,
		1	1	0	OCxREF + 极性 OCx=OCxREF xor CCxP	Off-State（输出使能但无效状态） OCxN=CCxNP
		1	1	1	OCxREF + 极性 + 死区	OCxREF 互补(非 OCxREF) +极性 + 死区

2.3.1 CCxE：0，CCxNE：0

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止（不由定时器驱动） OCx=0,	输出禁止（不由定时器驱动） OCxN=0

1） 输出理论分析：

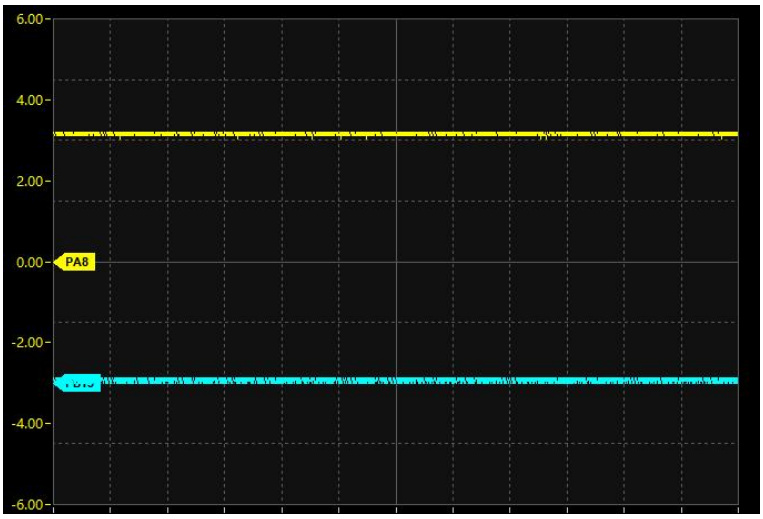
不由定时器驱动，输出取决于端口设置；例程代码 PA8 输出上拉，PB13 输出下拉；

2） 修改例程 CCxE：0、CCxNE：0；屏蔽使能语句。

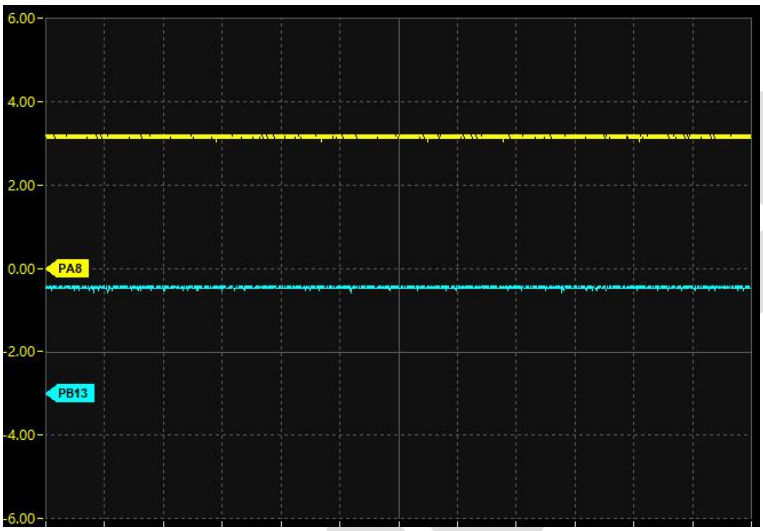
TIMER_CFG.c

```
111
112   Timer1_PWMwN_Pin_Init();
113
114 //  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
115 //  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

3） 测试输出：



4） 修改程序 PB13 上拉，测试输出



2.3.2 **CCxE：0，CCxNE：1**

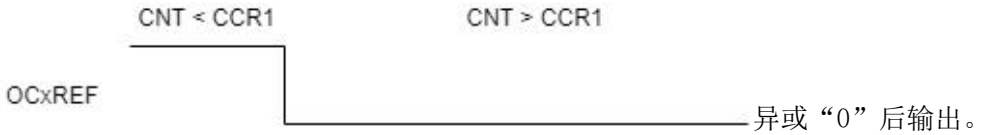
控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	1	输出禁止（不由定时器驱动） OCx=0	OCxREF + 极性 OCxN=OCxREF xor CCxNP

1） 输出理论分析：

PA8：不由定时器驱动，上下拉电阻决定（端口设置为输出）；应为高电平；

PB13：OC1REF 及 CC1NP 决定；程序中 CC1NP 为 0；（OCxREF 异或 CC1NP）应输出占空比 25% PWM 信号。

```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_HIGH;
```



异或“0”后输出。

2） 修改例程还原 PB13 下拉、CCxE：0、CCxNE：1

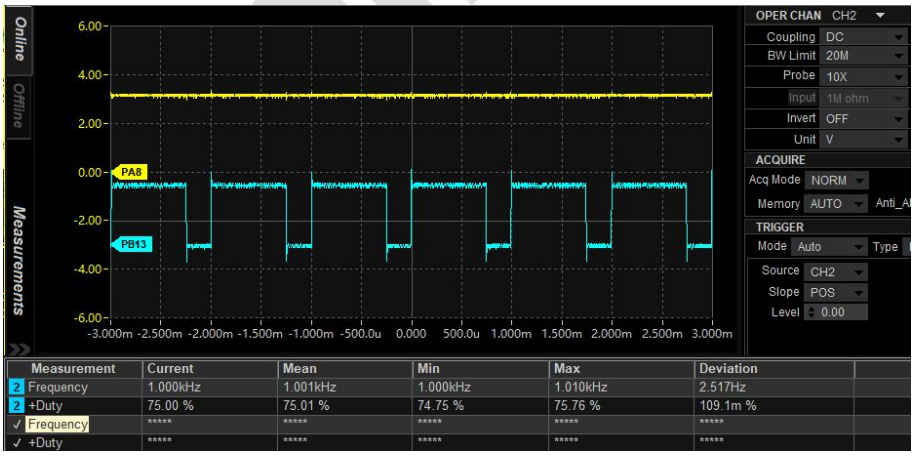
```
TIMER_CFG.c
108   TIM_BDTRInitStruct.BreakPolarity = MS32_TIM_BREAK_POLARITY_HIGH;
109   TIM_BDTRInitStruct.AutomaticOutput = MS32_TIM_AUTOMATICOUTPUT_DISABLE;
110   MS32_TIM_BDTR_Init(TIM1, &TIM_BDTRInitStruct);
111
112   Timer1_PWMwN_Pin_Init();
113
114 //  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
115   MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
116   MS32_TIM_EnableAllOutputs(TIM1);
```

3） 测试输出：

Measurement	Current	Mean	Min	Max	Deviation
2 Frequency	1.000kHz	1.000kHz	1.000kHz	1.000kHz	0.000Hz
2 +Duty	25.00 %	25.00 %	25.00 %	25.00 %	0.000 %
√ Frequency	*****	*****	*****	*****	*****
√ +Duty	*****	*****	*****	*****	*****

4） 修改程序，CC1NP 为 1；（OCxREF 异或 CC1NP）应输出占空比 75% PWM 信号。

```
// 1，MS32_TIM_OCPOLARITY_LOW：OCxREF 有效时输出低电平，无效时输出高电平
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_LOW;
```



Measurement	Current	Mean	Min	Max	Deviation
2 Frequency	1.000kHz	1.001kHz	1.000kHz	1.010kHz	2.517Hz
2 +Duty	75.00 %	75.01 %	74.75 %	75.76 %	109.1m %
√ Frequency	*****	*****	*****	*****	*****
√ +Duty	*****	*****	*****	*****	*****

2.3.3 CCxE：1，CCxNE：0

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	1	0	OCxREF + 极性 OCx=OCxREF xor CCxP	输出禁止（不由定时器驱动） OCxN=0

1） 输出理论分析：

PA8：OC1REF 及 CC1P 决定；程序中 CC1P 为 0；（OCxREF 异或 CC1P）应输出占空比 25% PWM 信号。

```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_HIGH;
```

异或“0”后输出。

PB13：不由定时器驱动，上下拉电阻决定（端口设置为输出）；应为低电平；

2） 修改例程 CCxE：1、CCxNE：0

```
114   MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
115   //MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

3） 测试输出：

2.3.4 CCxE：1，CCxNE：1

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	1	1	OCxREF + 极性 + 死区	OCxREF 互补（非 OCxREF）+极性 + 死区

1） 输出理论分析：

PA8：OC1REF 及 CC1P 决定；程序中 CC1P 为 0；应输出占空比 25% PWM 信号。

```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_HIGH;
```

有效时输出高电平，无效时输出低电平。

PB13：OC1REF 取反及 CC1NP 决定并插入死区；程序中 CC1NP 为 0；应输出占空比 75% PWM 信号并插入死区。

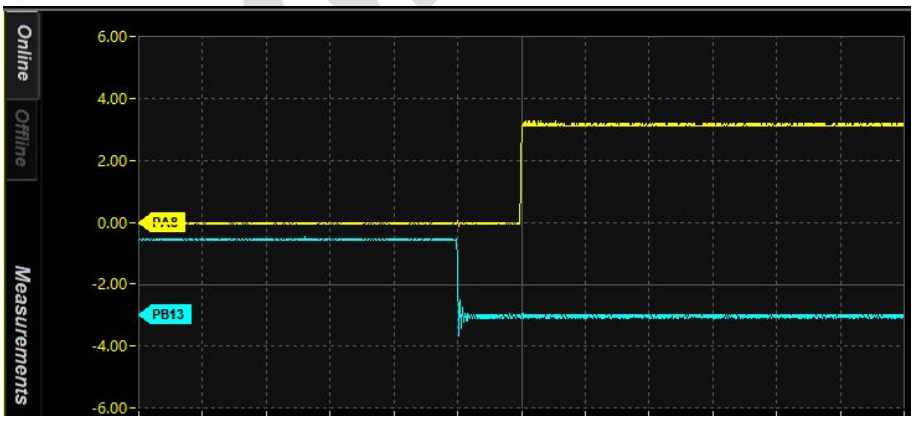
// 0，MS32_TIM_OCPOLARITY_HIGH：**OCxREF 互补**（取反）有效时输出高电平，无效时输出低电平

TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_HIGH;

蓝色为互补（取反），（蓝色）有效时输出高电平，（蓝色）无效时输出低电平

2） 例程初始设置 CCxE：1、CCxNE：1

3） 输出测试：



注：1uS 每格

2.3.5 CCxE：1，CCxNE：1，PWM 模式 1→OCxREF 强制有效

TIM_OC_InitStruct.OCMode = MS32_TIM_OCMODE_ACTIVE;

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	1	1	OCxREF + 极性 + 死区	OCxREF 互补(非 OCxREF) +极性 + 死区

1） 输出理论分析：

PA8：OC1REF 及 CC1P 决定；OC1REF 有效，程序中 CC1P 为 0 即 OC1REF 有效时输出高电平；应输出**高**电平。

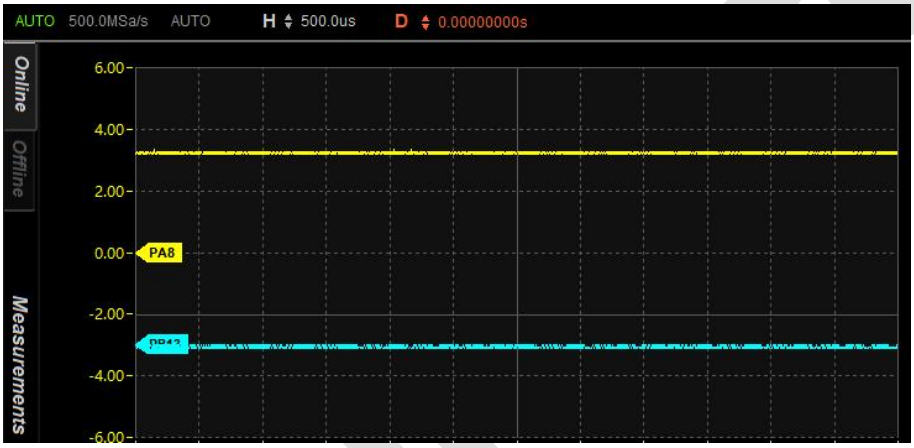
```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_HIGH;
```

PB13：OC1REF 取反及 CC1NP 决定并插入死区；OC1REF 有效互补后为无效，程序中 CC1NP 为 0 即 OC1REF 互补后无效输出低电平，应输出**低**电平；

```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 互补（取反）有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_HIGH;
```

2） 例程初始设置 CCxE：1、CCxNE：1

3） 输出测试：

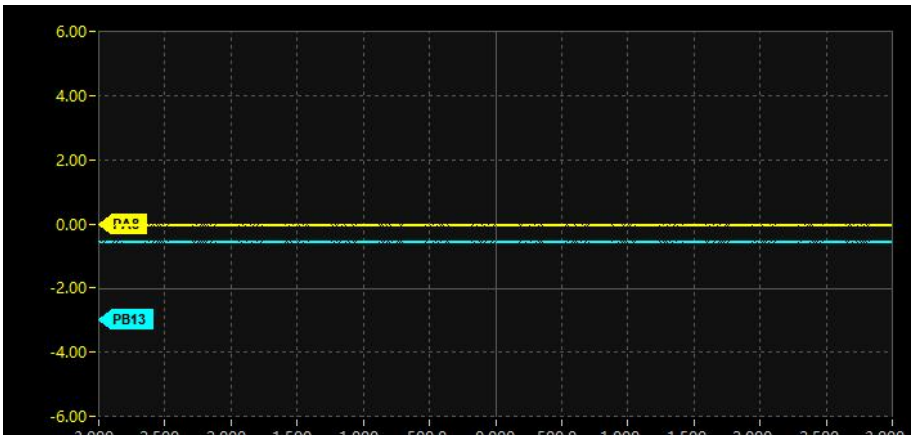


4） 修改 CC1P，CC1NP 极性

```
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_LOW;
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_LOW;
```

PA8：OC1REF 及 CC1P 决定；OC1REF 有效，程序中 CC1P 为 1 即 OC1REF 有效时输出低电平；应输出**低**电平。

PB13：OC1REF 取反及 CC1NP 决定并插入死区；OC1REF 有效互补后为无效，程序中 CC1NP 为 1 即 OC1REF 互补后无效，无效时输出**高**电平；



2.4 MOE：1，OSSR：1

```
TIM_BDTRInitStruct.OSSRState = MS32_TIM_OSSR_ENABLE;
```

2.4.1 CCxE：0，CCxNE：0

控制位					输出状态(注1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	1	0	0	输出禁止（不由定时器驱动） OCx=CCxP	输出禁止（不由定时器驱动） OCxN=CCxNP

结果应和 OSSR：0 一致。

2.4.2 CCxE：0，CCxNE：1

控制位					输出状态(注1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	1	0	1	Off-State (输出使能但无效态） OCx=CCxP	OCxREF + 极性 OCxN=OCxREF xor CCxNP,

1） 输出理论分析：

PA8：**输出无效态**；程序中 CC1P 为 0；应输出**低**；

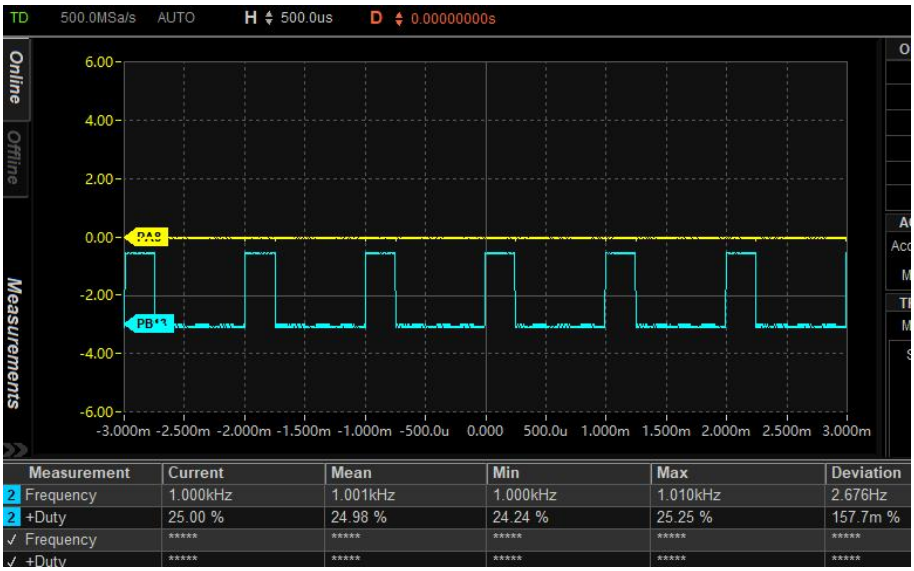
```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_HIGH;
```

PB13：（同 OSSR：0）OC1REF 及 CC1NP 决定；程序中 CC1NP 为 0；（OCxREF 异或 CC1NP）应输出占空比 25% PWM 信号。

```
// 0，MS32_TIM_OCPOLARITY_HIGH：OCxREF 有效时输出高电平，无效时输出低电平
  TIM_OC_InitStruct.OCNPolarity = MS32_TIM_OCPOLARITY_HIGH;
```

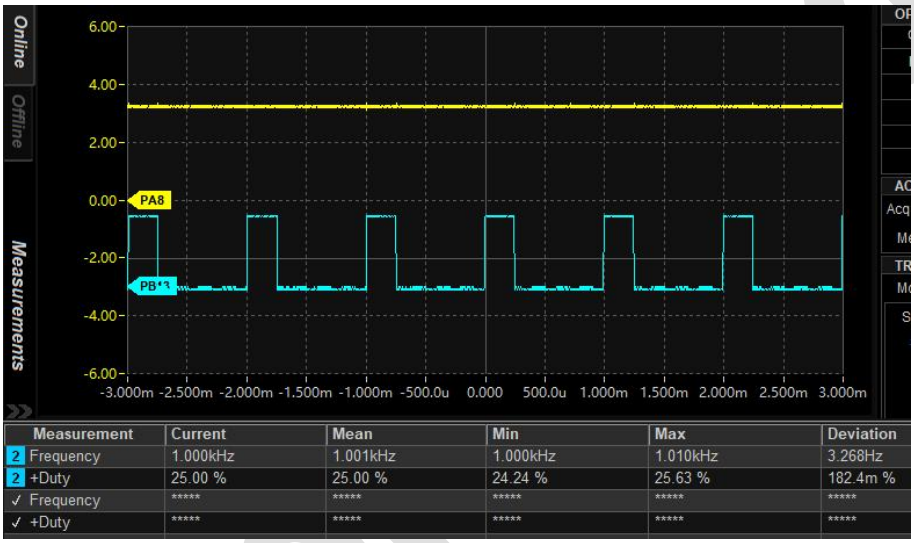
异或“0”后输出。

2）测试输出：



3）修改程序 CC1P 为 1；（无效态）PA8 应输出**高**；

```
// 1，MS32_TIM_OCPOLARITY_LOW：OCxREF 有效时输出低电平，无效时输出高电平
  TIM_OC_InitStruct.OCPolarity = MS32_TIM_OCPOLARITY_LOW;
```



2.4.3 CCxE：1，CCxNE：x

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	1	1	0	OCxREF + 极性 OCx=OCxREF xor CCxP	Off-State（输出使能但无效状态） OCxN=CCxNP
		1	1	1	OCxREF + 极性 + 死区	OCxREF 互补（非 OCxREF）+极性 + 死区

略。

2.5 MOE：1 输出总结

输出禁止（不由定时器驱动）：端口设置为输出时，上下拉电阻决定输出电平；
端口电平：OCxREF 异或 CCxP（CCxNP）；CCxP（CCxNP）为寄存器值（0 或 1）；

端口电平：OCxREF +极性；依据 OCxREF 及极性设置分析输出，如 CCxP 寄存器值 0（MS32_TIM_OCPOLARITY_HIGH），OCxREF 有效时输出高，无效输出低；

2.6 MOE：0，OSSI：0

```
//  MS32_TIM_EnableAllOutputs(TIM1);
  TIM_BDTRInitStruct.OSSIState = MS32_TIM_OSSI_DISABLE;
```

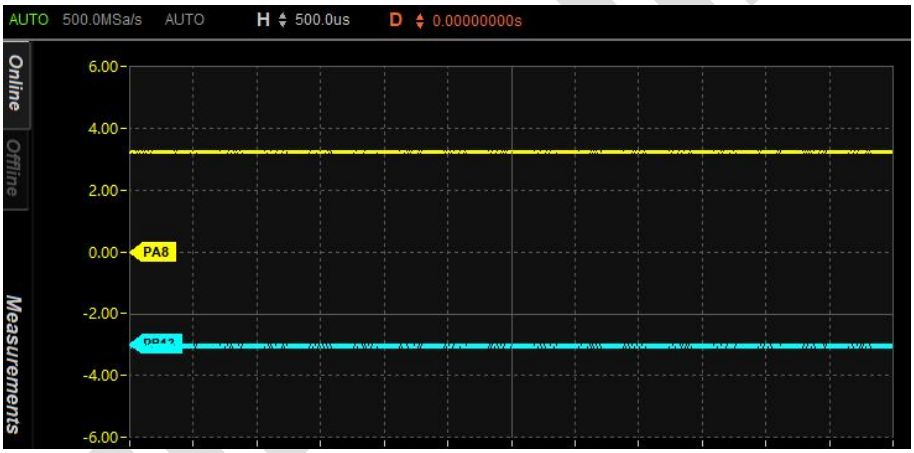
2.6.1 CCxE：0，CCxNE：0

略。

2.6.2 CCxE：0，CCxNE：1

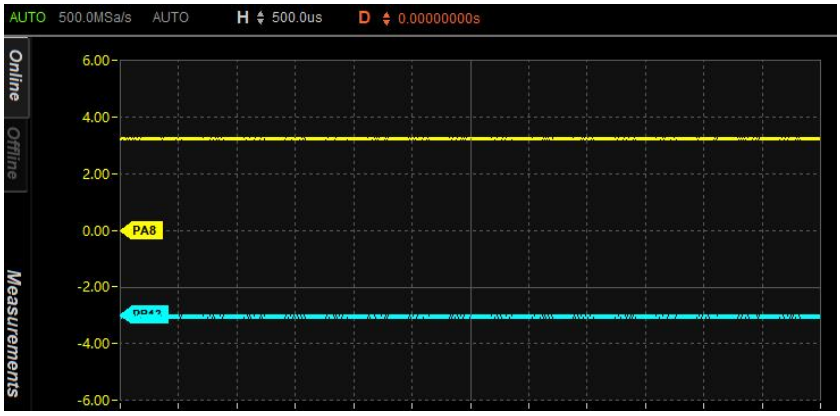
<table>
<tr><th colspan="5">控制位</th><th colspan="2">输出状态(注 1)</th></tr>
<tr><th>MOE bit</th><th>OSSI bit</th><th>OSSR bit</th><th>CCxE bit</th><th>CCxNE bit</th><th>OCx 输出状态</th><th>OCxN 输出状态</th></tr>
<tr><td rowspan="4">0</td><td>0</td><td rowspan="4">X</td><td>0</td><td>0</td><td>输出禁止（不由定时器驱动）
OCx=CCxP
OCx_EN=0</td><td>输出禁止（不由定时器驱动）
OCxN=CCxNP
OCxN_EN=0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td colspan="2">输出禁止（不由定时器驱动）</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td colspan="2">异步：OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td colspan="2">若时钟存在:经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平。</td></tr>
</table>

```
//  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

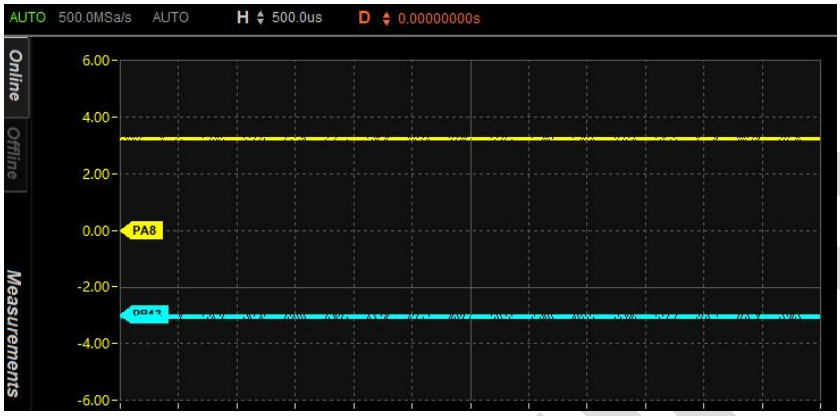


2.6.3 CCxE：1，CCxNE：0

```
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
//  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```



2.6.4 CCxE：1，CCxNE：1



2.7 MOE：0,OSSI：1

```
//  MS32_TIM_EnableAllOutputs(TIM1);
  TIM_BDTRInitStruct.OSSIState = MS32_TIM_OSSI_ENABLE;
```

2.7.1 CCxE：0，CCxNE：0

略。

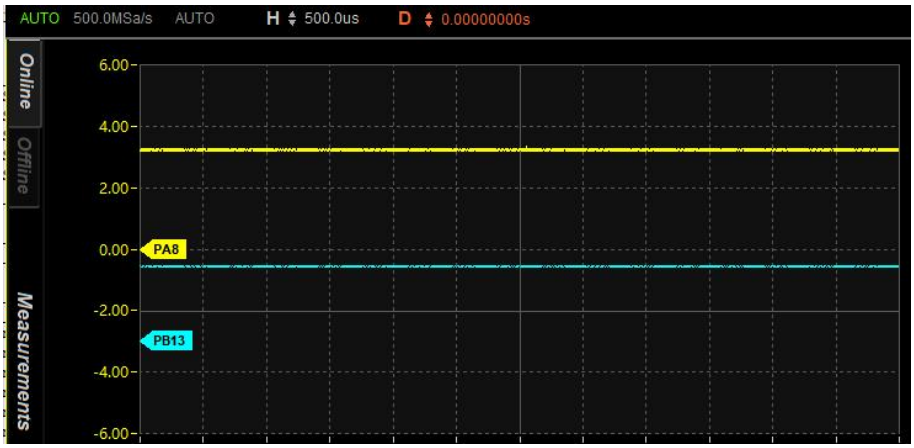
2.7.2 CCxE：0，CCxNE：1

控制位					输出状态(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
0	1	X	0	0	输出禁止（不由定时器驱动） OCx=CCxP	输出禁止（不由定时器驱动） OCxN=CCxNP
	1		0	1	Off-State（输出使能但无效状态）	
	1		1	0	异步：OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1	
	1		1	1	若时钟存在：经过一个死区时间后 OCx=OISx, OCxN=OISxN，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	

```
//  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

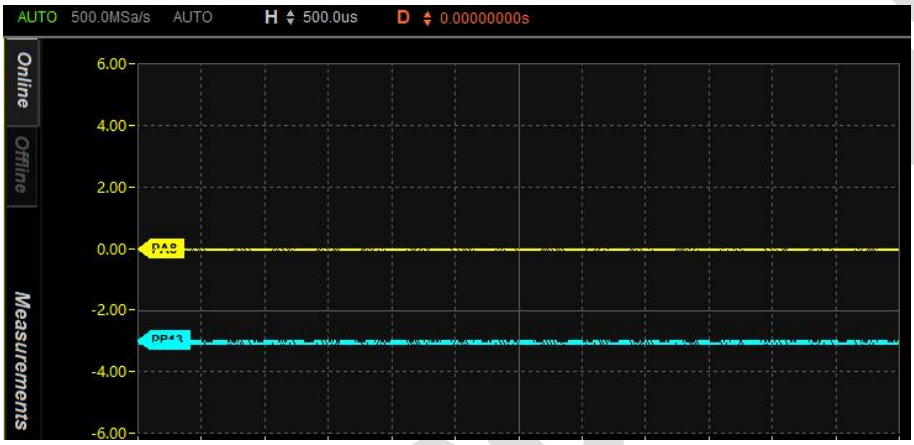
1) OISx：1，OISxN：1

```
  TIM_OC_InitStruct.OCIdleState = MS32_TIM_OCIDLESTATE_HIGH;
  TIM_OC_InitStruct.OCNIdleState = MS32_TIM_OCIDLESTATE_HIGH;
```



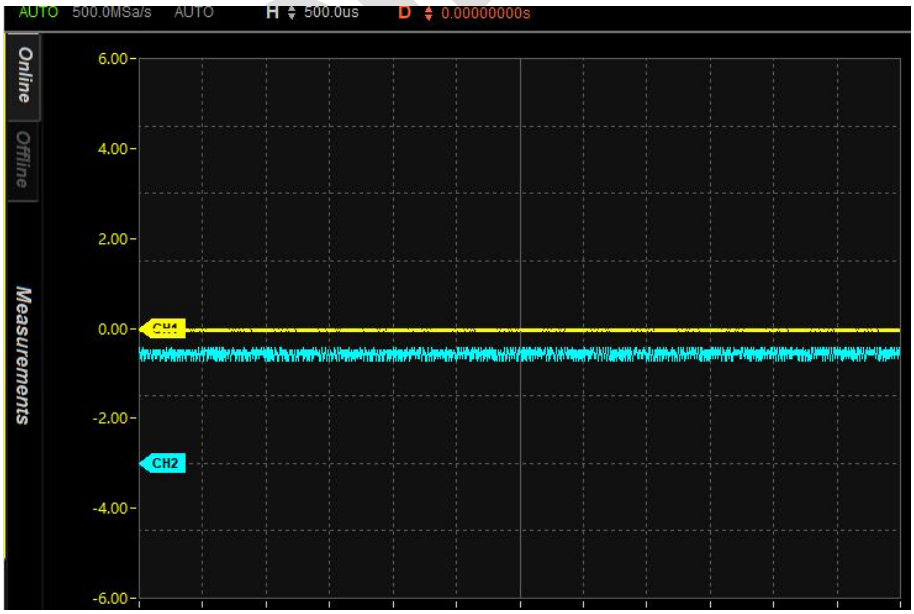
2) OISx：0，OISxN：0

```
TIM_OC_InitStruct.OCIdleState = MS32_TIM_OCIDLESTATE_LOW;
TIM_OC_InitStruct.OCNIdleState = MS32_TIM_OCIDLESTATE_LOW;
```



3) OISx：0，OISxN：1

```
TIM_OC_InitStruct.OCIdleState = MS32_TIM_OCIDLESTATE_LOW;
TIM_OC_InitStruct.OCNIdleState = MS32_TIM_OCIDLESTATE_HIGH;
```

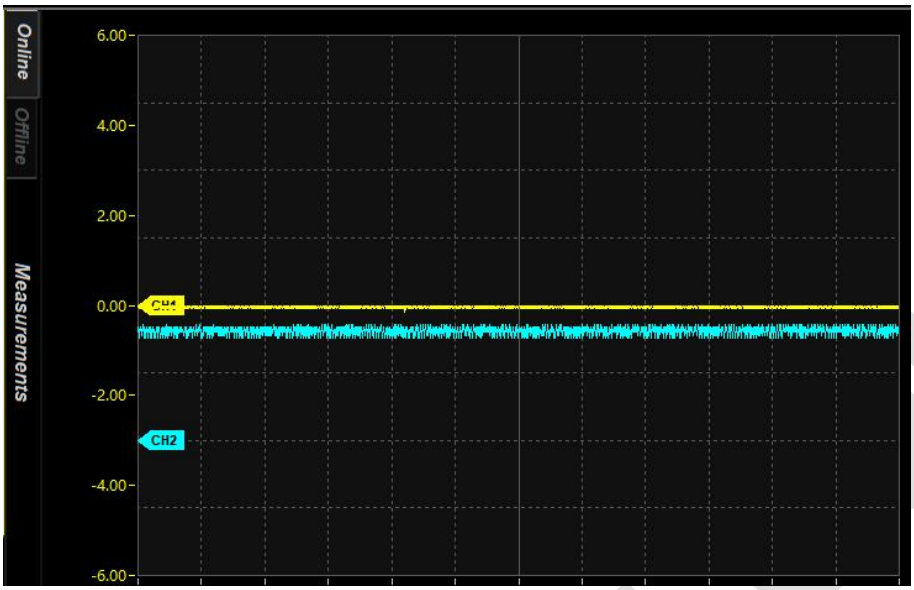


2.7.3 CCxE：1，CCxNE：0

```
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
//  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

OISx：0，OISxN：1

```
  TIM_OC_InitStruct.OCIdleState = MS32_TIM_OCIDLESTATE_LOW;
  TIM_OC_InitStruct.OCNIdleState = MS32_TIM_OCIDLESTATE_HIGH;
```

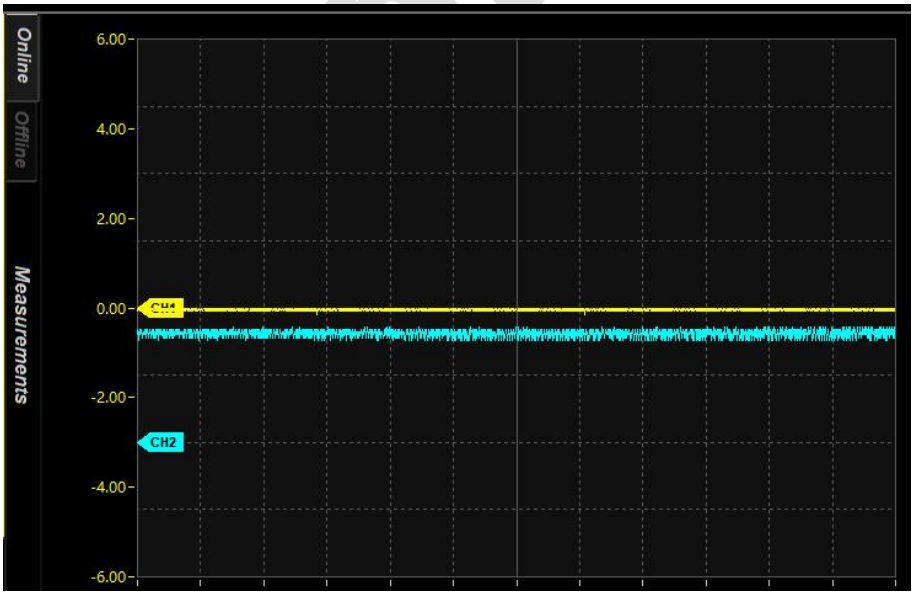


2.7.4 CCxE：1，CCxNE：1

```
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1);
  MS32_TIM_CC_EnableChannel(TIM1, MS32_TIM_CHANNEL_CH1N);
```

OISx：0，OISxN：1

```
  TIM_OC_InitStruct.OCIdleState = MS32_TIM_OCIDLESTATE_LOW;
  TIM_OC_InitStruct.OCNIdleState = MS32_TIM_OCIDLESTATE_HIGH;
```



2.8 其它模式

PWM2，OCxREF 强制有效等未测试；可借鉴文中方式进行测试；

使用互补 PWM 控制 H 桥等电机驱动电路时，建议按文中方式在 EV 板上测试输出信号，避免上、下管同时导通。特别是未使用带死区保护的预驱芯片驱动 MOS 的应用。

3 修订记录

版本	修订日期	修订内容
V1.0	2022-07-29	1359，初版。