文档版本	V1.0
发布日期	20210607

APT32S003 EPT 应用指南

目录

1 概述

本文介绍了在APT32S003中EPT应用

2. 适用的硬件

该例程使用于 APT32S003 系列学习板

3. 应用方案代码说明

3.1 EPT 配置

- **硬件配置：**

EPT 模块是一个增强型通用定时器，具有自动重载寄存器，有可编程的死区控制单元。支持捕获和波形发生器模式，有 7 个 TIMER 输出通道，支持 4 路独立输出或者 3 组互补输出。支持事件计数触发机制。

需要注意 EPT 中很多寄存器是由两个物理寄存器组成：活动寄存器（Active）和影子寄存器（Shadow），它们共享同一个物理访问地址。每个影子寄存器只有在特定条件满足时，才会更新到活动寄存器中。更新条件均可以独立设置。

- PWM 输出管脚：

管脚名称	突发计数模式	波形发生器： 单波形输出模式	波形发生器： 双波形输出模式
CHAX	时钟控制使能	输出波形	输出波形
CHAY	NA	输出波形	输出波形
CHBX	时钟控制使能	输出波形	输出波形
CHBY	NA	输出波形	输出波形
CHCX	NA	输出波形	输出波形
CHCY	NA	输出波形	输出波形
CHD	NA	输出波形	输出波形

图 3.1.1 输出管脚

- **软件配置：**

可在 apt32s003_initial.c 文件中 EPT0_CONFIG()进行初始化的配置.（EPT0 代表第一个 EPT 模块）

```
/*****************************************************************/
//ETP0 Functions
//EntryParameter:NONE
//ReturnValue:NONE
/*****************************************************************/
void EPT0_CONFIG(void)
{
   //PWM
   EPT_Software_Prg();
   EPT_IO_SET(EPT_IO_CHAX,IO_NUM_PA10);            //设置GPIO 功能
   EPT_IO_SET(EPT_IO_CHAY,IO_NUM_PB03);
   //
   EPT_IO_SET(EPT_IO_CHBX,IO_NUM_PB02);
   EPT_IO_SET(EPT_IO_CHBY,IO_NUM_PB04);
   //
   EPT_IO_SET(EPT_IO_CHCX,IO_NUM_PB05);
   EPT_IO_SET(EPT_IO_CHCY,IO_NUM_PA04);
   //
  //PCLK 作为TCLK 时钟，递增模式，连续模式，TCLK=PCLK/(0+1)
   EPT_PWM_Config(EPT_Selecte_PCLK,EPT_CNTMD_increase,EPT_OPM_Continue,0);
  //T1 选择SYNCIN4 作为输入触发,T2 选择SYNCIN5 作为输入触发
   //EPT_Tevent_Selecte(0x00,0x00);
  //使能SYNCUSR4 SYNCUSR5,连续触发
     //EPT_SYNCR_Config(EPT_Triggle_Continue,EPT_SYNCUSR0_REARMTrig_DIS,EPT_TRGSRC0_ExtSync_SYNCUSR0,EPT_TRGSRC1_ExtSync_SYNCUSR4,0x30);
   EPT_PWMX_Output_Control(EPT_PWMA,EPT_CA_Selecte_CMPA,EPT_CB_Selecte_CMPA,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,

   EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
   EPT_PWMX_Output_Control(EPT_PWMB,EPT_CA_Selecte_CMPB,EPT_CB_Selecte_CMPB,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,

   EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
   EPT_PWMX_Output_Control(EPT_PWMC,EPT_CA_Selecte_CMPC,EPT_CB_Selecte_CMPC,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,

   EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM
```

```
_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
        EPT_PWMX_Output_Control(EPT_PWMD,EPT_CA_Selecte_CMPD,EPT_CB_Selecte_CMPD,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_
Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,


        EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM
_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
        EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(4800,2400,1200,600,0);//PRDR=2400,CMPA=1200,CMPB=600,CMPC=2400,CMPD=0
        EPT_DB_CLK_Config(0,24,24);//Fdbclk=Fhclk/(0+1)，DTR=24clk，DTF=24clk
        //PWMA 作为互补输入源,CHX 上升沿，CHY 下降沿
       EPT_DBCR_Config(EPT_CHA_Selecte,EPT_CHAINSEL_PWMA_RISE_FALL,EPT_CHA_OUTSEL_EnRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCHX
_PBtoCHY);
        //PWMB 作为互补输入源,CHX 上升沿，CHY 下降沿
       EPT_DBCR_Config(EPT_CHB_Selecte,EPT_CHBINSEL_PWMB_RISE_FALL,EPT_CHB_OUTSEL_EnRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCHX
_PBtoCHY);
        //PWMC 作为互补输入源,CHX 上升沿，CHY 下降沿
       EPT_DBCR_Config(EPT_CHC_Selecte,EPT_CHCINSEL_PWMC_RISE_FALL,EPT_CHC_OUTSEL_EnRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCH
X_PBtoCHY);
        //EPT_Int_Enable(EPT_CAP_LD0);              //CMPA load 中断
        //EPT_Int_Enable(EPT_CAP_LD1);              //CMPB load 中断
        //EPT_Int_Enable(EPT_CAP_LD2);              //CMPC load 中断
        //EPT_Int_Enable(EPT_CAP_LD3);              //CMPD load 中断
        //EPT_Int_Enable(EPT_CAU);                   //递增阶段 CNT=CMPA 作为中断
        //EPT_Int_Enable(EPT_CAD);                   //递减阶段 CNT=CMPA 作为中断
        //EPT_Int_Enable(EPT_CBU);                  //递增阶段 CNT=CMPB 作为中断
        //EPT_Int_Enable(EPT_CBD);                  //递减阶段 CNT=CMPB 作为中断
        //EPT_Int_Enable(EPT_CCU);                  //递增阶段 CNT=CMPC 作为中断
        //EPT_Int_Enable(EPT_CCD);                  //递减阶段 CNT=CMPC 作为中断
        //EPT_Int_Enable(EPT_CDU);                  //递增阶段 CNT=CMPD 作为中断
        //EPT_Int_Enable(EPT_CDD);                  //递减阶段 CNT=CMPD 作为中断
        //EPT_Int_Enable(EPT_PEND);                 //周期结束中断作为中断
        //EPT_Vector_Int_Enable();
        EPT_Start();
        //捕捉
        /*EPT_Software_Prg();
        EPT_Capture_Config(EPT_Selecte_PCLK,EPT_CNTMD_increase,EPT_CAPMD_Continue,EPT_CAP_EN,EPT_LDARST_EN,EPT_LDBRST_DIS,EPT_LD
CRST_DIS,EPT_LDDRST_DIS,1,0);//TCLK=pclk/(1+0),CMPAload CMPBload
        EPT_SYNCR_Config(EPT_Triggle_Continue,EPT_SYNCUSR0_REARMTrig_DIS,EPT_TRGSRC0_ExtSync_SYNCUSR0,EPT_TRGSRC1_ExtSync_SYNC
USR4,0x04);//使能 SYNCUSR2 ,连续触发
        EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(0XFFFF,0,0,0,0);
        EPT_Int_Enable(EPT_CAP_LD0);              //CMPA 载入中断
        EPT_Int_Enable(EPT_CAP_LD1);              //CMPB 载入中断
        EPT_Vector_Int_Enable();
        EPT_Start();*/
    }
```

- 代码说明：

EPT_Software_Prg(); ----用于软件复位控制

EPT_IO_SET(); ----用于配置 GPIO 作为 PWM 功能

EPT_PWM_Config(); ----用于配置 PWM 模式及时钟

EPT_PWMX_Output_Control(); ----用于 PWM 输出配置

EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(); ----用于周期设置

EPT_DB_CLK_Config(); ----用于配置死区时钟分频

EPT_DBCR_Config(); ----用于选择死区通道及使能

EPT_Int_Enable(); ----用于配置中断使能

EPT_Vector_Int_Enable(); ----用于启动中断

EPT_Start(); ----用于启动 EPT 模块

EPT_Capture_Config(); ----用于配置捕获

EPT_SYNCR_Config(); ----用于配置同步控制

3.2 PWM 互补输出

系统时钟选择内部 48Mhz，输出周期为 100us，占空比为 50us,死区为 0.5us。

PA0.10->CHAX 周期:CLKS*4800=100us,占空比:CLKS*2400=50us

PB0.3->CHAY

PB0.2->CHBX 周期:CLKS*4800=100us,占空比:CLKS*1200=25us

PA0.8->CHBY

PB0.0->CHCX 周期:CLKS*4800=100us,占空比:CLKS*600=12.5us

PA0.4->CHCY

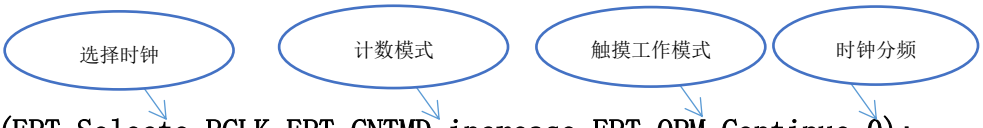
```
    /*****************************************************************/
    //ETP0 Functions
    //EntryParameter:NONE
    //ReturnValue:NONE
    /*****************************************************************/
    void EPT0_CONFIG(void)
    {
        //PWM
        EPT_Software_Prg();
        EPT_IO_SET(EPT_IO_CHAX,IO_NUM_PA10);              //设置GPIO 功能
        EPT_IO_SET(EPT_IO_CHAY,IO_NUM_PB03);
        //
        EPT_IO_SET(EPT_IO_CHBX,IO_NUM_PB02);
        EPT_IO_SET(EPT_IO_CHBY,IO_NUM_PA08);
        //
        EPT_IO_SET(EPT_IO_CHCX,IO_NUM_PB00);
        EPT_IO_SET(EPT_IO_CHCY,IO_NUM_PA04);
        //
       //PCLK 作为TCLK 时钟，递增模式，连续模式，TCLK=PCLK/(0+1)
        EPT_PWM_Config(EPT_Selecte_PCLK,EPT_CNTMD_increase,EPT_OPM_Continue,0);
        EPT_PWMX_Output_Control(EPT_PWMA,EPT_CA_Selecte_CMPA,EPT_CB_Selecte_CMPA,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_N
ochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T
1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
        EPT_PWMX_Output_Control(EPT_PWMB,EPT_CA_Selecte_CMPB,EPT_CB_Selecte_CMPB,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_
Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_
T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);

        EPT_PWMX_Output_Control(EPT_PWMC,EPT_CA_Selecte_CMPC,EPT_CB_Selecte_CMPC,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_
Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_
T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
        EPT_PWMX_Output_Control(EPT_PWMD,EPT_CA_Selecte_CMPD,EPT_CB_Selecte_CMPD,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_
Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_
T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
       //PRDR=2400,CMPA=1200,CMPB=600,CMPC=2400,CMPD=0
        EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(4800,2400,1200,600,0);
        EPT_DB_CLK_Config(0,24,24);//Fdbclk=Fhclk/(0+1)，DTR=24clk，DTF=24clk
        //PWMA 作为互补输入源,CHX 上升沿，CHY 下降沿
       EPT_DBCR_Config(EPT_CHA_Selecte,EPT_CHAINSEL_PWMA_RISE_FALL,EPT_CHA_OUTSEL_EnRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCHX
_PBtoCHY);
        //PWMB 作为互补输入源,CHX 上升沿，CHY 下降沿
       EPT_DBCR_Config(EPT_CHB_Selecte,EPT_CHBINSEL_PWMB_RISE_FALL,EPT_CHB_OUTSEL_EnRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCHX
_PBtoCHY);
        //PWMC 作为互补输入源,CHX 上升沿，CHY 下降沿
       EPT_DBCR_Config(EPT_CHC_Selecte,EPT_CHCINSEL_PWMC_RISE_FALL,EPT_CHC_OUTSEL_EnRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCH
```

```
X_PBtoCHY);
        EPT_Start();
    }
```

- 参数说明：


```
EPT_IO_SET(EPT_IO_CHAX,IO_NUM_PA10);
```




```
EPT_PWM_Config(EPT_Selecte_PCLK,EPT_CNTMD_increase,EPT_OPM_Continue,0);
```

```
EPT_PWMX_Output_Control(EPT_PWMA,EPT_CA_Selecte_CMPA,EPT_CB_Selecte_CMPA,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochange,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);
```

EPT_PWMA：选择 EPT_PWM 通道

EPT_CA_Selecte_CMPA：CA 比较值选择

EPT_CB_Selecte_CMPA：CB 比较值选择

EPT_PWM_ZRO_Event_OutHigh：CNT 值等于零时，PWMX 管脚输出状态配置

EPT_PWM_PRD_Event_Nochange：CNT 值等于 PRDR 时，PWMX 管脚输出状态配置

EPT_PWM_CAU_Event_OutLow：CNT 值等于 CAU(计数方向递增)时，PWMX 管脚输出状态配置

EPT_PWM_CAD_Event_OutLow：CNT 值等于 CAD(计数方向递减)时，PWMX 管脚输出状态配置

EPT_PWM_CBU_Event_Nochange：CNT 值等于 CBU(计数方向递增)时，PWMX 管脚输出状态配置

EPT_PWM_CBD_Event_Nochange：CNT 值等于 CBD(计数方向递减)时，PWMX 管脚输出状态配置

EPT_PWM_T1U_Event_Nochange：T1 事件发生(计数方向递增)时，PWMX 管脚输出状态配置

EPT_PWM_T1D_Event_Nochange：T1 事件发生(计数方向递减)时，PWMX 管脚输出状态配置

EPT_PWM_T2U_Event_Nochange：T2 事件发生(计数方向递增)时，PWMX 管脚输出状态配置

EPT_PWM_T2D_Event_Nochange：T2 事件发生(计数方向递减)时，PWMX 管脚输出状态配置

0h：不动作（过滤该处理事件）

1h：清除输出（低电平）

2h：置位输出（高电平）

3h：反向（翻转）

周期设置　比较值 A,B,C,D

```
EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(4800,2400,1200,600,0);
```

死区时钟分频　上升/下降沿延时

```
EPT_DB_CLK_Config(0,24,24);
```

通道选择　延时输入选择　死区输出配置

```
EPT_DBCR_Config(EPT_CHA_Selecte,EPT_CHAINSEL_PWMA_RISE_FALL,EPT_CHA_OUTSEL_E
```

输出极性控制　死区输出交换控制

```
nRise_EnFall,EPT_PB_OUT_Reverse,EPT_PAtoCHX_PBtoCHY);
```

- **输出波形：**

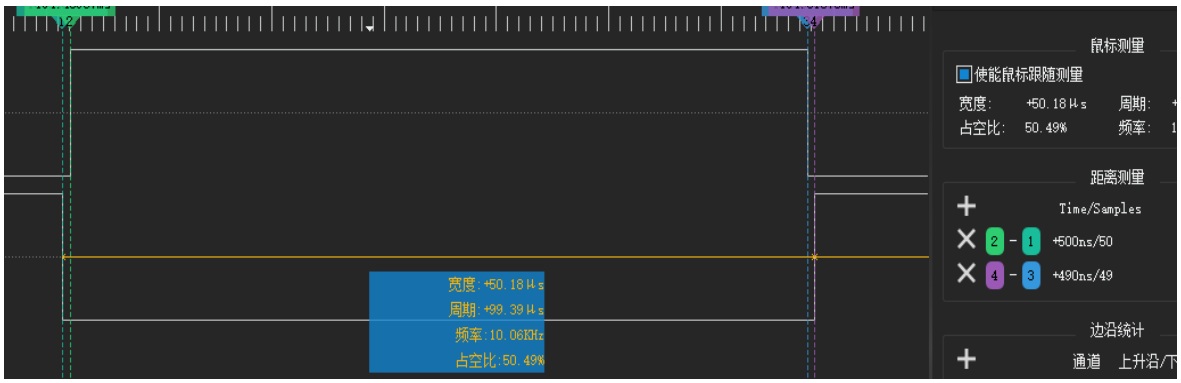


图 3.2.1 PWM 互补波形

- **计算死区延时：**

  Fdbclk=Fhclk/(0+1)，DTR=24clk，DTF=24clk

  Fdbclk = (1/48Mhz) /1

  上升沿延时 TRED = DTR x TDBCLK = (1/48Mhz)*24 = 0.5us

  下降沿延时数值 TRED = DTF x TDBCLK = (1/48Mhz)*24 = 0.5us

3.3 输入捕获

设置 PA0.0 的下降沿外部中断事件，通过 ET 触发 TIMER 的捕获操作，R_CMPA_BUF 存储低电平计数值，R_CMPB_BUF 存储周期计数值

- **编程要点：**

  1. 配置 GPIO 的外部触发
  2. 设置 ETCB 触发
  3. 配置 EPT 捕获模式
  4. 配置 EPT 中断
  5. 配置 SYSCON 中的事件触发选择

```
void GPIO_CONFIG(void)
{
//EXI0_INT= EXI0/EXI16,EXI1_INT= EXI1/EXI17, EXI2_INT=EXI2~EXI3/EXI18/EXI19, EXI3_INT=EXI4~EXI9, EXI4_INT=EXI10~EXI15
   GPIO_IntGroup_Set(PA0,0,Selete_EXI_PIN0);                    //EXI0 set PA.0
   GPIOA0_EXI_Init(EXI0);                                       //PA0.0 as input
   EXTI_trigger_CMD(ENABLE,EXI_PIN0,_EXIFT);                    //ENABLE falling edge
   //EXTI_trigger_CMD(ENABLE,EXI_PIN0,_EXIRT);                  //ENABLE rising edge
     EXTI_interrupt_CMD(ENABLE,EXI_PIN0);                       //enable EXI
     GPIO_EXTI_interrupt(GPIOA0,0b0000000000000001);            //enable GPIOA00 as EXI
}
/*************************************************************/
//ETP0 Functions
//EntryParameter:NONE
//ReturnValue:NONE
/*************************************************************/
void EPT0_CONFIG(void)
```

```
    {
        //捕捉
        EPT_Software_Prg();
        EPT_Capture_Config(EPT_Selecte_PCLK,EPT_CNTMD_increase,EPT_CAPMD_Continue,EPT_CAP_EN,EPT_LDARST_EN,EPT_LDBRST_DIS,EPT_LD
CRST_DIS,EPT_LDDRST_DIS,1,0);                    //TCLK=pclk/(1+0),CMPAload CMPBload
        EPT_SYNCR_Config(EPT_Triggle_Continue,EPT_SYNCUSR0_REARMTrig_DIS,EPT_TRGSRC0_ExtSync_SYNCUSR0,EPT_TRGSRC1_ExtSync_SYNC
USR4,0x04);                                      //使能 SYNCUSR2 ,连续触发
        EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(0XFFFF,0,0,0,0);
        EPT_Int_Enable(EPT_CAP_LD0);             //CMPA 载入中断
        EPT_Int_Enable(EPT_CAP_LD1);             //CMPB 载入中断
        EPT_Vector_Int_Enable();
        EPT_Start();
    }

    /*****************************************************************/
    //ET Initial
    //EntryParameter:NONE
    //ReturnValue:NONE
    /*****************************************************************/
    void ET_CONFIG(void)
    {
         ET_DeInit();
         ET_CH0_SRCSEL(ET_SRC0,ENABLE,ET_EXI_SYNC0);
        ET_CH0_CONTROL(ENABLE,TRG_HW,ET_EPT0_TRGSRC2);      //SYNCIN2：Capture 触发事件

        ET_ENABLE();
    }
    /*****************************************************************/
    //syscon Functions
    //EntryParameter:NONE
    //ReturnValue:NONE
    /*****************************************************************/
    void SYSCON_CONFIG(void)
    {
    //------SYSTEM CLK AND PCLK FUNTION--------------------------/
        SYSCON_RST_VALUE();                                          //SYSCON all register clr
        SYSCON_General_CMD(ENABLE,ENDIS_ISOSC);
        SYSCON_HFOSC_SELECTE(HFOSC_SELECTE_48M);                     //HFOSC selected 48MHz
        //system clock set, Hclk div ,Pclk div    set system clock=SystemCLK/Hclk div/Pclk div
       SystemCLK_HCLKDIV_PCLKDIV_Config(SYSCLK_HFOSC,HCLK_DIV_1,PCLK_DIV_1,HFOSC_48M);
       //------------    WDT FUNTION    -------------------------------/
       //WDT TIME 1s,WDT alarm interrupt time=1s-1s*1/8=0.875S
      SYSCON_IWDCNT_Config(IWDT_TIME_2S,IWDT_INTW_DIV_7);
      SYSCON_WDT_CMD(ENABLE);                                        //enable WDT
```

```
    SYSCON_IWDCNT_Reload();                                                    //reload WDT
    //IWDT_Int_Enable();
//------------  WWDT FUNTION  ---------------------------------/
    WWDT_CNT_Load(0xFF);
    WWDT_CONFIG(PCLK_4096_DIV0,0xFF,WWDT_DBGDIS);
    WWDT_Int_Config(ENABLE);
    //WWDT_CMD(ENABLE);
//------------  CLO  ---------------------------------/
    //SYSCON_CLO_CONFIG(CLO_PA02);
    //SYSCON->OPT1=(SYSCON->OPT1&0XFFFF8000)|(0X01<<12)|(0X04<<8)|(0x00<<4);
//------------  LVD FUNTION  ---------------------------------/ 、
  //LVD LVR Enable/Disable
  SYSCON_LVD_Config(DISABLE_LVDEN,INTDET_LVL_3_3V,RSTDET_LVL_1_9V,DISABLE_LVD_INT,INTDET_POL_fall);
  //LVD_Int_Enable();
//------------ EVTRG function ---------------------------------/
    SYSCON->EVTRG=0X00|0x01<<20;                                               //选择 EXI0 事件作为当前触发通道事件 SYSCON_trgsrc0
    SYSCON->EVPS=0X00;
    //SYSCON->IMER =EM_EVTRG0_ST;
//------------  SYSCON Vector  ---------------------------------/
    //SYSCON_Int_Enable();                                                     //SYSCON VECTOR
    //SYSCON_WakeUp_Enable();                                                  //Enable WDT wakeup INT
}

/*****************************************************************/
//EPT0 Interrupt
//EntryParameter:NONE
//ReturnValue:NONE
/*****************************************************************/
void EPT0IntHandler(void)
{
     // ISR content ...
    if((EPT0->MISR&EPT_TRGEV0_INT)==EPT_TRGEV0_INT)
    {
        EPT0->ICR=EPT_TRGEV0_INT;
    }
    else if((EPT0->MISR&EPT_TRGEV1_INT)==EPT_TRGEV1_INT)
    {
        EPT0->ICR=EPT_TRGEV1_INT;
    }
    else if((EPT0->MISR&EPT_TRGEV2_INT)==EPT_TRGEV2_INT)
    {
        EPT0->ICR=EPT_TRGEV2_INT;
    }
    else if((EPT0->MISR&EPT_TRGEV3_INT)==EPT_TRGEV3_INT)
```

```
    {
        EPT0->ICR=EPT_TRGEV3_INT;
    }
    else if((EPT0->MISR&EPT_CAP_LD0)==EPT_CAP_LD0)
    {
        EPT0->ICR=EPT_CAP_LD0;
        EXTI_trigger_CMD(DISABLE,EXI_PIN0,_EXIRT);
        EXTI_trigger_CMD(ENABLE,EXI_PIN0,_EXIFT);
        R_CMPA_BUF=EPT0->CMPA;                          //低电平
    }
    else if((EPT0->MISR&EPT_CAP_LD1)==EPT_CAP_LD1)
    {
        EPT0->ICR=EPT_CAP_LD1;
        EXTI_trigger_CMD(ENABLE,EXI_PIN0,_EXIRT);
        EXTI_trigger_CMD(DISABLE,EXI_PIN0,_EXIFT);
        R_CMPB_BUF=EPT0->CMPB;                          //周期计数值
    }
}
```

- **代码说明：**

  EPT_Software_Prg(); ----用于软件复位寄存器

  EPT_Capture_Config(); ----用于配置捕获

  EPT_SYNCR_Config(); ----用于配置触发模式

  EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config();----用于配置周期值

  EPT_Int_Enable(); ----用于使能中断

  EPT_Vector_Int_Enable(); ----用于开启 EPT 中断

  EPT_Start(); ----用于启动 EPT

- **函数参数说明：**

- **PA0 口输入波形：**

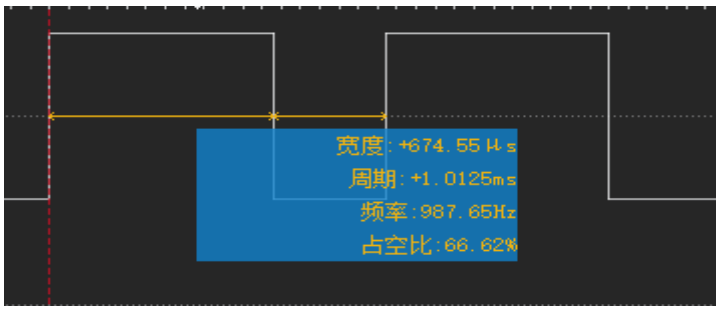


图 3.3.1 PA0 触发波形

- **计数器单周期时间：**

T(低电平) = (0x9a91-0x5834)*(1/48Mhz) = 339us

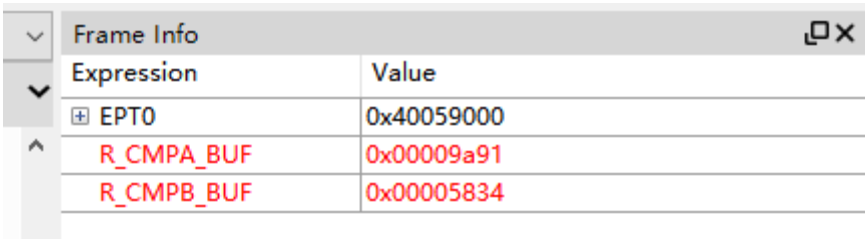


图 3.3.2 CDK 中单次触发

3.4 四路独立 PWM

系统时钟选择内部 48Mhz，输出周期为 100us,占空比不同。

- ETP 输出引脚选择：

  PA0.10->CHAX / PB0.2->CHBX / PB0.3->CHCX / PA0.8->CHD

```
/*************************************************************/
//ETP0 Functions
//EntryParameter:NONE
//ReturnValue:NONE
/*************************************************************/
void EPT0_CONFIG(void)
{
    EPT_Software_Prg();
    EPT_IO_SET(EPT_IO_CHAX,IO_NUM_PA10);
    EPT_IO_SET(EPT_IO_CHBX,IO_NUM_PB02);
    EPT_IO_SET(EPT_IO_CHCX,IO_NUM_PB03);
    EPT_IO_SET(EPT_IO_CHD,IO_NUM_PA08);
    //PCLK 作为 TCLK 时钟，递增模式，连续模式，TCLK=PCLK/(0+1)
    EPT_PWM_Config(EPT_Selecte_PCLK,EPT_CNTMD_increase,EPT_OPM_Continue,0);

EPT_PWMX_Output_Control(EPT_PWMA,EPT_CA_Selecte_CMPA,EPT_CB_Selecte_CMPA,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochang
e,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1U_Ev
ent_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);

EPT_PWMX_Output_Control(EPT_PWMB,EPT_CA_Selecte_CMPB,EPT_CB_Selecte_CMPB,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochan
    ge,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1
    U_Event_Nochange, EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);

EPT_PWMX_Output_Control(EPT_PWMC,EPT_CA_Selecte_CMPC,EPT_CB_Selecte_CMPC,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochan
    ge,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,EPT_PWM_T1
    U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);

EPT_PWMX_Output_Control(EPT_PWMD,EPT_CA_Selecte_CMPD,EPT_CB_Selecte_CMPD,EPT_PWM_ZRO_Event_OutHigh,EPT_PWM_PRD_Event_Nochan
ge,EPT_PWM_CAU_Event_OutLow,EPT_PWM_CAD_Event_OutLow,EPT_PWM_CBU_Event_Nochange,EPT_PWM_CBD_Event_Nochange,
  EPT_PWM_T1U_Event_Nochange,EPT_PWM_T1D_Event_Nochange,EPT_PWM_T2U_Event_Nochange,EPT_PWM_T2D_Event_Nochange);

   //PRDR=4800,CMPA=2400,CMPB=1200,CMPC=600,CMPD=300
  EPT_PRDR_CMPA_CMPB_CMPC_CMPD_Config(4800,2400,1200,600,300);

EPT_DBCR_Config(EPT_CHA_Selecte,EPT_CHAINSEL_PWMA_RISE_FALL,EPT_CHA_OUTSEL_PWMA_PWMB_Bypass,EPT_PA_PB_OUT_Direct,EPT_PAto
```

```
CHX_PBtoCHY)                                                    ;//PWMA 输作为 CHAX 输出源

EPT_DBCR_Config(EPT_CHB_Selecte,EPT_CHBINSEL_PWMB_RISE_FALL,EPT_CHB_OUTSEL_PWMB_PWMC_Bypass,EPT_PA_PB_OUT_Direct,EPT_PAto
CHX_PBtoCHY);                                                   //PWMB 输作为 CHBX 输出源

EPT_DBCR_Config(EPT_CHC_Selecte,EPT_CHCINSEL_PWMC_RISE_FALL,EPT_CHC_OUTSEL_PWMC_PWMD_Bypass,EPT_PA_PB_OUT_Direct,EPT_PAto
CHX_PBtoCHY);                                                   //PWMC 输作为 CHCX 输出源
        EPT_Start();
}
```

- 代码说明：

  EPT_Software_Prg(); ----用于软件复位 EPT 模块

  EPT_IO_SET(); ----用于配置 GPIO 作为 PWM 功能

  EPT_PWM_Config(); ----用于配置时钟

  EPT_PWMX_Output_Control();----用于配置 PWM

  EPT_DBCR_Config(); ----用于 PWM 输出及极性配置

- 输出波形：

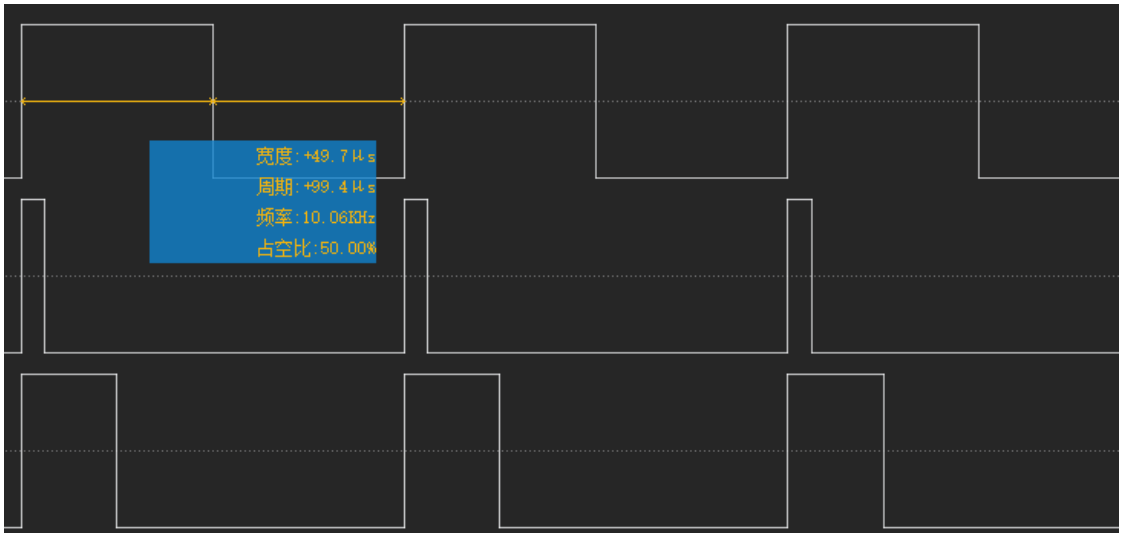


图 3.4.1 4 路独立输出波形

4. 程序下载和运行

1. 将目标板与仿真器连接，分别为 VDD SCLK SWIO GND

2. 程序编译后仿真运行

3. 通过示波器或逻辑分析仪查看输出波形。如上图 3.2.1、图 3.2.4 所示