文档版本	V1.0
发布日期	20191108

APT32F172 GTC 应用开发指南

目录

1 概述

本文介绍了在APT32F172中使用GTC的应用范例。

2. 适用的硬件

该例程使用于 APT32F172 开发板 APT-DB172

3. 应用方案代码说明

基于 APT32F172 完整的库文件系统，可以很方便的对 GTC 进行配置。

3.1 GTC 定时配置

软件配置：

开启内部主频 20MHz,并作为系统时钟。

计数器单周期时间：TCCLK=sysclock/2^1/10=1us

PB0.0 输出周期 100us,占空比 50us 方波

```
/*************************************************************/
//GTC Functions
//EntryParameter:NONE
//ReturnValue:NONE
/*************************************************************/
void GTC_CONFIG(void)
{
    GTC_RESET_VALUE();              //GTC 所有寄存器复位赋值
    GTC_SoftwareReset();            //GTC 软件复位
    GTC_Configure(GTC_FIN_IMOSC , 1 , 9 , Counter_Size_32BIT  , 50 , 0);
    //TCCLK=sysclock/2^1/10=1us,周期 100*1US
    GTC_ControlSet_Configure(GTC_ControlSet_REPEAT,ENABLE);
    //使能该模式后，计数器在连续计数模式溢出或者周期模式周期结束后，会自动重新开始计数
    GTC_ConfigInterrupt_CMD(GTC_PSTARTI);//GTC 周期开始中断使能
    GTC_Int_Enable();                   //GTC 中断向量使能
    GTC_Start();                        //start counter
}
```

代码说明：

GTC_Configure(GTC_FIN_IMOSC , 1 , 9 , Counter_Size_32BIT , 50 , 0);

GTC_FIN_IMOSC----选择内部 IMOSC

GTCclk 计算公式：TCCLK=FIN/2^DIVN/(DINM+1)

1 DIVN=1

9---- DINM=9

Counter_Size_32BIT----选择 32BIT 计数器

50----周期=50

0----比较值=0

```
/*******************************************************************************/
//APT32F172_init
//EntryParameter:NONE
//ReturnValue:NONE
/*******************************************************************************/
void APT32F172_init(void)
{
SYSCON_WDT_CMD(DISABLE);                    //关闭 WDT

SYSCON->PCER0=0xFFFFFFFF;                   //使能 IP
SYSCON->PCER1=0xFFFFFFFF;                   //使能 IP
while(!(SYSCON->PCSR0&0x1));                //判断 IP 是否使能

SYSCON_Int_Enable();                        //使能 SYSCON 中断向量
SYSCON->IECR=ISOSC_ST|IMOSC_ST|EMOSC_ST|SYSCLK_ST;
//使能 ISOSC 时钟稳定中断,使能 IMOSC 时钟稳定中断,使能 EMOSC 时钟稳定中断

CK_CPU_EnAllNormalIrq();                    //打开全局中断
SYSCON_CONFIG();                            //syscon 参数 初始化

GPIO_CONFIG();                              //GPIO 初始化
GTC_CONFIG ();                              //GTC 初始化
}

volatile U32_T f_io_toggle;
/*************************************************************/
//GTC Interrupt
//EntryParameter:NONE
//ReturnValue:NONE
/*************************************************************/
void GTCIntHandler(void)
{
    if((GTC->MISR&GTC_PSTARTI)==GTC_PSTARTI)
    {
```

```
            GTC->ICR = GTC_PSTARTI;
            if(!f_io_toggle)
            {
                f_io_toggle=1;
                GPIO_Write_High(GPIOB0,0);
            }
            else
            {
                f_io_toggle=0;
                GPIO_Write_Low(GPIOB0,0);
            }
        }
    }
```

3.2 GTC PWM 输出配置

开启内部主频 20MHz，并作为系统时钟。

PB0.0 输出周期 100us，占空比 50us 方波

```
/*******************************************************************************/
//APT32F172_init
//EntryParameter:NONE
//ReturnValue:NONE
/*******************************************************************************/
void GTC_CONFIG(void)
{
    GTC_RESET_VALUE();              //GTC 所有寄存器复位赋值
    GTC_SoftwareReset();            //GTC 软件复位
    GTC_IO_Init(GTC_IO_TXOUT , 0 ); //PB0.0 做 PWM 输出口
    GTC_Configure(GTC_FIN_IMOSC , 1 , 9 , Counter_Size_32BIT   , 100 , 50);
    //TCCLK=sysclock/2^1/10=1us,周期 100*1US
    GTC_ControlSet_Configure(GTC_ControlSet_REPEAT,ENABLE);
    //使能该模式后，计数器在连续计数模式溢出或者周期模式周期结束后，会自动重新开始计数
    /GTC_ControlSet_Configure(GTC_ControlSet_PWMEN,ENABLE);//使能 PWM 模式
    GTC_Start();                    //start counter
}
```

代码说明：

GTC_ControlSet_Configure(GTC_ControlSet_OUTST,ENABLE); ----

使能计数开始输出高电平，计数到比较值翻转

GTC_ControlSet_Configure(GTC_ControlSet_IDLEST,ENABLE); ----

使能 Idle 状态下输出高电平，禁止输出低电平

```
/*******************************************************************************/
//APT32F172_init
//EntryParameter:NONE
//ReturnValue:NONE
/*******************************************************************************/
void APT32F172_init(void)
{
    SYSCON_WDT_CMD(DISABLE);                    //关闭 WDT

    SYSCON->PCER0=0xFFFFFFF;                    //使能 IP
    SYSCON->PCER1=0xFFFFFFF;                    //使能 IP
    while(!(SYSCON->PCSR0&0x1));                //判断 IP 是否使能

    SYSCON_Int_Enable();                        //使能 SYSCON 中断向量
    SYSCON->IECR=ISOSC_ST|IMOSC_ST|EMOSC_ST|SYSCLK_ST;
    //使能 ISOSC 时钟稳定中断,使能 IMOSC 时钟稳定中断,使能 EMOSC 时钟稳定中断

    CK_CPU_EnAllNormalIrq();                    //打开全局中断
    SYSCON_CONFIG();                            //syscon 参数 初始化

    GTC_CONFIG ();                          //GTC 初始化
}
```

3.3 GTC 捕捉输入配置

开启内部主频 20MHz,并作为系统时钟。

计数器单周期时间：TCCLK=sysclock/2^1/10=1us

PA1.1 捕捉高电平 50us，低电平 50us 方波。

R_Capture_buf1 存储周期时间。

R_Capture_buf2 存储低电平时间。

R_Capture_buf3 存储高电平时间。

```
/*******************************************************************************/
//APT32F172_init
//EntryParameter:NONE
//ReturnValue:NONE
/*******************************************************************************/
void GTC_CONFIG(void)
{
    GTC_RESET_VALUE();          //GTC 所有寄存器复位赋值
    GTC_SoftwareReset();        //GTC 软件复位
    GTC_IO_Init(GTC_IO_TCAPX , 1 ); //PA1.1 做捕捉输入口
    GTC_Configure(GTC_FIN_PCLK, 1 , 9 , Counter_Size_32BIT   , 0 , 0); //TCCLK=sysclock/2^1/10=1us
    GTC_ControlSet_Configure(GTC_ControlSet_STOPCLEAR,ENABLE); //计数器停止后清除计数值
    GTC_ControlSet_Configure(GTC_ControlSet_CNTM,ENABLE);
    //工作在连续计数模式，计数值自增直到溢出
    GTC_ControlSet_Configure(GTC_ControlSet_REPEAT,ENABLE); //使能循环重复模式
    GTC_ControlSet_Configure(GTC_ControlSet_CAPT_F,ENABLE);          //下降沿捕捉使能
    GTC_ControlSet_Configure(GTC_ControlSet_CAPT_TCAP,ENABLE);       //捕捉输入使能
    GTC_ConfigInterrupt_CMD(GTC_CAPTI, ENABLE);                      //捕捉中断使能
    GTC_Start();                                                     //start counter
    GTC_Int_Enable();                                                //使能 GTC 中断向量
}

/*******************************************************************************/
```

```
//APT32F172_init
//EntryParameter:NONE
//ReturnValue:NONE
/*******************************************************************************/
void APT32F172_init(void)
{
    SYSCON_WDT_CMD(DISABLE);                    //关闭 WDT

    SYSCON->PCER0=0xFFFFFFF;                    //使能 IP
    SYSCON->PCER1=0xFFFFFFF;                    //使能 IP
    while(!(SYSCON->PCSR0&0x1));                //判断 IP 是否使能

    SYSCON_Int_Enable();                        //使能 SYSCON 中断向量
    SYSCON->IECR=ISOSC_ST|IMOSC_ST|EMOSC_ST|SYSCLK_ST;
    //使能 ISOSC 时钟稳定中断,使能 IMOSC 时钟稳定中断,使能 EMOSC 时钟稳定中断

    CK_CPU_EnAllNormalIrq();                    //打开全局中断
    SYSCON_CONFIG();                            //syscon 参数 初始化

    GTC_CONFIG ();                          //GTC 初始化
}

volatile U32_T   R_Capture_buf1,R_Capture_buf2,R_Capture_buf3,f_GTC_CaptTrigg;
/*************************************************************/
//GTC Interrupt
//EntryParameter:NONE
//ReturnValue:NONE
/*************************************************************/
void GTCIntHandler(void)
{
```

```
        if((GTC->MISR&GTC_CAPTI)==GTC_CAPTI)
        {
            GTC->ICR = GTC_CAPTI;
            if(!f_GTC_CaptTrigg)
            {
                    f_GTC_CaptTrigg=1;
                    R_Capture_buf1=GTC->CDCR;                    //R_Capture_buf1  周期时间
                    GTC_ControlSet_Configure(GTC_ControlSet_CAPT_R,ENABLE);     //capture down enable
                    GTC_ControlSet_Configure(GTC_ControlSet_CAPT_F,DISABLE);
                    GTC_Stop();
                    GTC_Start();
            }
            else
            {
                    f_GTC_CaptTrigg=0;
                    R_Capture_buf2=GTC->CUCR;                    // R_Capture_buf2 低电平时间
                    GTC_ControlSet_Configure(GTC_ControlSet_CAPT_R,DISABLE); //捕捉上升沿禁止
                    GTC_ControlSet_Configure(GTC_ControlSet_CAPT_F,ENABLE);//捕捉下降沿使能
            }
            if(R_Capture_buf1>R_Capture_buf2)                    //R_Capture_buf3  高电平时间
            {
                    R_Capture_buf3=R_Capture_buf1-R_Capture_buf2;
            }
            else
            {
                    R_Capture_buf3=R_Capture_buf2-R_Capture_buf1;
            }
        }
}
```

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. 定时测试将示波器挂在对应 TOGGLE IO 上
3. PWM 输出将示波器挂在对应 PWM 输出口上
4. 将需检测波形输出接到捕捉口上
5. 程序编译后仿真运行
6. 定时和 PWM 观察示波器波形，捕捉功能观察 R_Capture_buf1~R_Capture_buf3 变量值是否与输入波形的周期和占空比匹配

5. 改版历史

版本	修改日期	修改概要
V1.0	2019-11-08	初版