DSP 应用技术作业

姓名：徐延宾
学号：9171040G0633

作业三：

根据“Example_ADC”中实验内容，打开程序 LAB11_main.c，DSP2833x_PieCtrl.c 以及相关头文件，阅读程序段落。

1.**摘录与中断设置相关的程序语句。**

（1）LAB11_main.c 的 void main(void)内相关设置语句：

```
DINT;                    //禁止 CPU 中断，禁止全局中断
InitPieCtrl();           //初始化 PIE 控制寄存器
IER=0x0000;              //禁用所有 CPU 中断并清除 CPU 中断标志位
IFR=0x0000;
InitPieVectTable();      //初始化 PIE 向量里面包含了 PieCtrlRegs.PIECTRL.bit.ENPIE=1
EALLOW;
PieVectTable.EPWM1_INT =&epwm1_timer_adc_isr;   //第三组第一中断
EDIS;
InitAdcParameters();
InitEPwm1Parameters();
PieCtrlRegs.PIEIER3.bit.INTx1 = 1;   //响应 EPWM1_INT 中断
PieCtrlRegs.PIECTRL.bit.ENPIE=1;   //打开 PIE 中断,使能 PIE
IER |= M_INT3;           //打开 CPU 第 3 组中断
EINT;                    //使能全局中断，允许中断响应
ERTM;
```

（2）LAB11_main.c 的 void InitAdcParameters(void) 函数内相关设置语句：

```
AdcRegs.ADCST.bit.INT_SEQ1_CLR=1;     //清除 SEQ1 中断标志位
AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1=0; //INT-SEQ1 对 CPU 的中断请求被禁用
AdcRegs.ADCTRL2.bit.INT_MOD_SEQ1=0; //每个 SEQ1 序列结束时，INT-SEQ1 置位
```

（3）LAB11_main.c 的 void InitEPwm1Parameters(void) 函数内相关设置语句：

```
EPwm1Regs.ETSEL.bit.INTEN = 1;                //使能 ePWMx_INT 产生
EPwm1Regs.ETPS.bit.INTPRD = ET_3RD;       //在第三个事件产生中断
```

（4）LAB11_main.c 的 interrupt void epwm1_timer_adc_isr(void) 函数内相关设置语句：

```
AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;          //复位 SEQ1
EPwm1Regs.ETCLR.bit.INT = 1;                //清除中断标志位
PieCtrlRegs.PIEACK.all = PIEACK_GROUP3; //中断应答
```

（5）DSP2833x_PieCtrl.c 的 void InitPieCtrl(void) 函数内相关设置语句：

```
DINT;
```

（6）DSP2833x_PieCtrl.c 的 void EnableInterrupts() 函数内相关设置语句：

```
EINT;
```

（7）DSP2833x_Device.h 的相关中断定义：

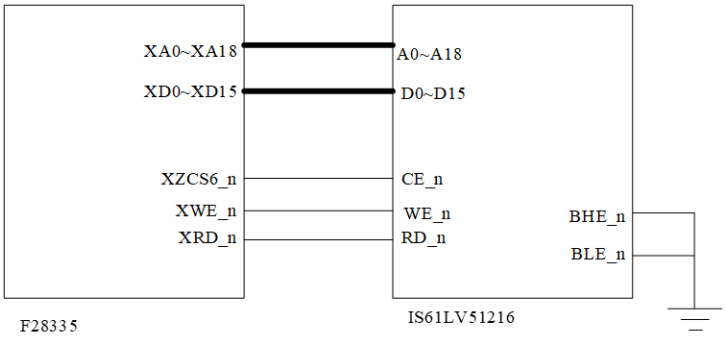
```
extern cregister volatile unsigned int IFR;
extern cregister volatile unsigned int IER;
```

#define EINT asm(" clrc INTM")
#define DINT asm(" setc INTM")

2. 函数 InitPieCtrl()实现的功能。
初始化 PIE 控制寄存器。
3. 函数 InitPieVectTable()实现的功能。
初始化 PIE 向量表。
4. 语句 PieVectTable.EPWM1_INT =&epwm1_timer_adc_isr; 实现的功能。
将中断函数映射到中断向量表的第三组第一中断。

向量名称	PIE向量地址	功能	内容
INT1.1	0x0000 0D40	SEQ1INT	0xXXXXXX
……	……	……	……
INT3.1	0x0000 0D60	EPWM1_INT	&epwm1_timer_adc_isr
……	……	……	……
INT12.8	0x0000 0DFE	PIEINT12.8	0xXXXXXX

作业四：XINTF 模块设置



1.指出存储器地址范围；

地址范围：0x00000-0x7FFFF

2.在程序 Example_2833xDMA_xintf_to_ram.c 指出相关的配置代码。指出 XTIMING6、XINTCNF2 寄存器各字段的数值及含义。

(1) 程序 Example_2833xDMA_xintf_to_ram.c 相关的配置代码：

在函数 void init_zone6(void)中配置：

```
void init_zone6(void)
{
    EALLOW;
    // Make sure the XINTF clock is enabled
    SysCtrlRegs.PCLKCR3.bit.XINTFENCLK = 1;
    EDIS;
    // Configure the GPIO for XINTF with a 16-bit data bus
    // This function is in DSP2833x_Xintf.c
    InitXintf16Gpio();
    // All Zones---------------------------------
    // Timing for all zones based on XTIMCLK = SYSCLKOUT
    EALLOW;
    XintfRegs.XINTCNF2.bit.XTIMCLK = 0;
    // Buffer up to 3 writes
    XintfRegs.XINTCNF2.bit.WRBUFF = 3;
    // XCLKOUT is enabled
    XintfRegs.XINTCNF2.bit.CLKOFF = 0;
    // XCLKOUT = XTIMCLK
    XintfRegs.XINTCNF2.bit.CLKMODE = 0;
    // Zone 6------------------------------------
    // When using ready, ACTIVE must be 1 or greater
    // Lead must always be 1 or greater
    // Zone write timing
    XintfRegs.XTIMING6.bit.XWRLEAD = 1;
    XintfRegs.XTIMING6.bit.XWRACTIVE = 2;
    XintfRegs.XTIMING6.bit.XWRTRAIL = 1;
    // Zone read timing
    XintfRegs.XTIMING6.bit.XRDLEAD = 1;
```

```
    XintfRegs.XTIMING6.bit.XRDACTIVE = 3;
    XintfRegs.XTIMING6.bit.XRDTRAIL = 0;
    // don't double all Zone read/write lead/active/trail timing
    XintfRegs.XTIMING6.bit.X2TIMING = 0;
    // Zone will not sample XREADY signal
    XintfRegs.XTIMING6.bit.USEREADY = 0;
    XintfRegs.XTIMING6.bit.READYMODE = 0;
    // 1,1 = x16 data bus
    // 0,1 = x32 data bus
    // other values are reserved
    XintfRegs.XTIMING6.bit.XSIZE = 3;
    EDIS;
  //Force a pipeline flush to ensure that the write to
  //the last register configured occurs before returning.
  asm(" RPT #7 || NOP");
}
```

（2）指出 XTIMING6、XINTCNF2 寄存器各字段的数值及含义：

表格 1 XTIMING6 Register

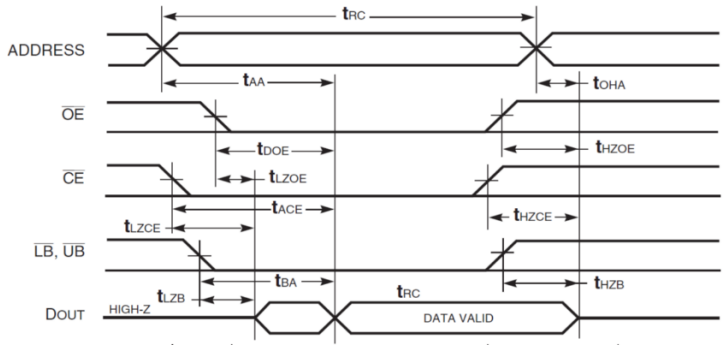
字段	数值	含义
XWRLEAD	1	Write Lead Period=1 XTIMCLK cycle
XWRACTIVE	2	Write Active Period Waitstates =2 XTIMCLK cycles
XWRTRAIL	1	Write Trail Period =1 XTIMCLK cycle
XRDLEAD	1	Read Lead Period =1 XTIMCLK cycle
XRDACTIVE	3	Read Active Period Waitstates =3 XTIMCLK cycles
XRDTRAIL	0	Read Trail Period =0
X2TIMING	0	The values are scaled 1:1
USEREADY	0	The XREADY signal is ignored when accesses are made to the zone.
READYMODE	0	XREADY input is synchronous for the zone.
XSIZE	3	In this mode the zone will only use 16 data lines. The XA0/WE1 signal will behave as XA0.

表格 2 XINTCNF2 Register

字段	数值	含义
XTIMCLK	0	XTIMCLK = SYSCLKOUT
WRBUFF	3	Buffer up to 3 writes（此处不明白，用户指南未找到说明）
CLKOFF	0	XCLKOUT is enabled
CLKMODE	0	XCLKOUT = XTIMCLK

3.根据存储器的读写时序，能否优化 DSP 的 XINTF 配置？给出具体配置方案。

IS61LV51216 存储器读时序图

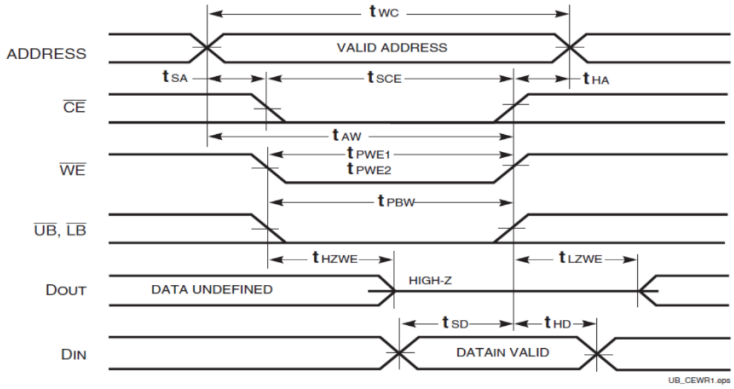


IS61LV51216 存储器读访问参数表

Symbol	Parameter	-8 Min.	-8 Max.	-10 Min.	-10 Max.	-12 Min.	-12 Max.	Unit
t_{RC}	Read Cycle Time	8	—	10	—	12	—	ns
t_{AA}	Address Access Time	—	8	—	10	—	12	ns
t_{OHA}	Output Hold Time	3	—	3	—	3	—	ns
t_{ACE}	$\overline{CE}$ Access Time	—	8	—	10	—	12	ns
t_{DOE}	$\overline{OE}$ Access Time	—	3.5	—	4	—	5	ns
t_{HZOE}[2]	$\overline{OE}$ to High-Z Output	—	3	—	4	0	5	ns
t_{LZOE}[2]	$\overline{OE}$ to Low-Z Output	0	—	0	—	0	—	ns
t_{HZCE}[2]	$\overline{CE}$ to High-Z Output	0	3	0	4	0	6	ns
t_{LZCE}[2]	$\overline{CE}$ to Low-Z Output	3	—	3	—	3	—	ns
t_{BA}	$\overline{LB}$, $\overline{UB}$ Access Time	—	3.5	—	4	—	5	ns
t_{HZB}[2]	$\overline{LB}$, $\overline{UB}$ to High-Z Output	0	3	0	3	0	4	ns
t_{LZB}[2]	$\overline{LB}$, $\overline{UB}$ to Low-Z Output	0	—	0	—	0	—	ns
t_{PU}	Power Up Time	0	—	0	—	0	—	ns
t_{PD}	Power Down Time	—	8	—	10	—	12	ns

Notes:
1. Test conditions assume signal transition times of 3 ns or less, timing reference levels of 1.5V, input pulse levels of 0V to 3.0V and output loading specified in Figure 1.
2. Tested with the load in Figure 2. Transition is measured ±500 mV from steady-state voltage.

IS61LV51216 存储器写时序图



IS61LV51216 存储器写访问参数表

Symbol	Parameter	-8 Min.	-8 Max.	-10 Min.	-10 Max.	-12 Min.	-12 Max.	Unit
t_{WC}	Write Cycle Time	8	—	10	—	12	—	ns
t_{SCE}	$\overline{CE}$ to Write End	6.5	—	8	—	8	—	ns
t_{AW}	Address Setup Time to Write End	6.5	—	8	—	8	—	ns
t_{HA}	Address Hold from Write End	0	—	0	—	0	—	ns
t_{SA}	Address Setup Time	0	—	0	—	0	—	ns
t_{PWB}	$\overline{LB}$, $\overline{UB}$ Valid to End of Write	6.5	—	8	—	8	—	ns
t_{PWE1}	$\overline{WE}$ Pulse Width	6.5	—	8	—	8	—	ns
t_{PWE2}	$\overline{WE}$ Pulse Width ($\overline{OE}$ = LOW)	8.0	—	10	—	12	—	ns
t_{SD}	Data Setup to Write End	5	—	6	—	6	—	ns
t_{HD}	Data Hold from Write End	0	—	0	—	0	—	ns
t_{HZWE}[2]	$\overline{WE}$ LOW to High-Z Output	—	3.5	—	5	—	6	ns
t_{LZWE}[2]	$\overline{WE}$ HIGH to Low-Z Output	2	—	2	—	2	—	ns

XINTF 的配置主要是读写时序：

读时序：

$t_{XRDLEAD} = t_{ACE} - t_{DOE}$

$t_{XRDACTIVE} = t_{RC} + t_{OHA} - t_{HZOE} - (t_{AA} - t_{DOE})$

$t_{XRDTRAIL} = t_{HZCE} - t_{HZOE}$

```
// Zone read timing（tc=6.67ns）
    XintfRegs.XTIMING6.bit.XRDLEAD = ;
    XintfRegs.XTIMING6.bit.XRDACTIVE = ;
    XintfRegs.XTIMING6.bit.XRDTRAIL = ;
```

写时序：

$t_{XWRDLEAD}$

$t_{XWRACTIVE} = t_{PBW1}$

$t_{XWRTRAIL}$

```
// Zone write timing（tc=6.67ns）
    XintfRegs.XTIMING6.bit.XWRLEAD =;
    XintfRegs.XTIMING6.bit.XWRACTIVE =;
    XintfRegs.XTIMING6.bit.XWRTRAIL =;
```

但是具体的配置方案暂时没有更好地想法，这里的更加优化指的是读写时间更短吗？还是什么其他的指标呢？这里不甚清楚。

作业五：

根据“Example_ADC”中实验内容，打开程序 LAB11_main.c 及相关头文件，阅读程序段落。

1.摘录与 ePWM 模块设置相关的程序语句；

（1）程序 LAB11_main.c 的 void InitEPwm1Parameters(void)相关的配置代码：

```
void InitEPwm1Parameters(void)
{
// InitEPwm1Gpio();
// Disable TBCLK within the ePWM
    EALLOW;
    SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 0; //停止 epwm 模块内部的时间基准时钟
    EDIS;
// TBCLK = SYSCLKOUT / (HSPCLKDIV*CLKDIV)=150/(6*1)=25
    EPwm1Regs.TBCTL.bit.HSPCLKDIV =0x03; //高速时间基准时钟预分频为两倍
    EPwm1Regs.TBCTL.bit.CLKDIV = 0x00;     //时间基准时钟预分频位 等于 0 即 1 分频
// Set Period for EPWM1
    EPwm1Regs.TBPRD = 208;
//设定时间基准器计数器的周期 208-fs 20kHz,139-fs 30kHz    149--27.9kHz
T(PWM1)=TBCLK/(TBPRD*2*3)=25/(208*3*2) = 0.02MHz , 20KHz
    EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;    //增减计数模式
// Setup Compare A = 2 TBCLK counts
    EPwm1Regs.CMPA.half.CMPA = 2; //计数比较寄存器 A CMPA 当前工作的 CMPA 的值不
断和时间基准计数器 TBCTR 比较
// Phase is 0 for Synchronization Event
    EPwm1Regs.TBPHS.half.TBPHS = 0x0000; //TBCTR 不装载相位寄存器 TBPHS 的值
// Clear    TB counter
    EPwm1Regs.TBCTR = 0x0000;//事件基准计数寄存器 TBCTR 读取写到其中的 TBCTR 的
值清除
// Phase loading disabled
    EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;//禁止 TBCTR 对 TBPHS 的装载
// Enable the TBCTL Shadow
    EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;//TBCTR 装载其映射寄存器的值
// Disable EPWMxSYNCO signal
    EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE; //禁用 EPWMxSYNCO signal
// CMPA Register operating mode, 0 means operates as a double buffer, all writes via the CUP access
the shadow register
    EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;//映射模式，双缓冲模式，所有
CPU 写操作将访问映射寄存器
// Active CMPA Load From Shadow Select Mode when CTR=0
    EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;    // load on CTR = Zero
// Set actions
// Force EPWMA output high when the counter equals the active CMPA register and the counter is
incrementing
    EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;//计数递增       强制 ePWMxA 输出高
```

```
// Force EPWMA output low Action when the counter equals the active CMPA register and the counter is decrementing
    EPwm1Regs.AQCTLA.bit.CAD = AQ_CLEAR;//计数递减       强制 ePWMxA 输出低
// Dead-Band Generator Rising Edge Delay Count Register=0
//     EPwm1Regs.DBRED=0;
// Dead-Band Generator Falling Edge Delay Count Register=0
//     EPwm1Regs.DBFED=0;
// Enable ADC Start of SOCA Pulse
    EPwm1Regs.ETSEL.bit.SOCAEN = 1;        //使能 ePWMxSOCA 脉冲
// Select SOC from CPMA on upcount
    EPwm1Regs.ETSEL.bit.SOCASEL = 2;        //TBCTR=TBPRD 时产生 ePWMxSOCA
// Select how many selected ETSEL events need to occur before an EPWMxSOCA pulse is generated;//在第三个事件产生 ePWMxSOCA 脉冲
    EPwm1Regs.ETPS.bit.SOCAPRD = 3;
// Enable event time-base counter equal to period (TBCTR = TBPRD)
    EPwm1Regs.ETSEL.bit.INTSEL = ET_CTR_PRD; // TBCTR=TBPRD 时产生 ePWMxSOCA
// Enable EPWMx_INT generation
    EPwm1Regs.ETSEL.bit.INTEN = 1;                //使能 ePWMx_INT 产生
// These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated.
    EPwm1Regs.ETPS.bit.INTPRD = ET_3RD;          //在第三个事件产生中断
// Enable TBCLK within the ePWM
    EALLOW;
    SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 1;
    EDIS;
}
```

（2）程序 LAB11_main.c 的 interrupt void epwm1_timer_adc_isr(void)函数相关的配置代码:

```
interrupt void epwm1_timer_adc_isr(void)       //中断函数
{
  xn=AdcRegs.ADCRESULT1;
 *Da_out=xn;
// Reinitialize for the next ADC Sequence
// Reset SEQ1
 AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;        //复位 SEQ1
// Clear INT SEQ1 bit
 EPwm1Regs.ETCLR.bit.INT = 1;               //清除中断标志位
// Acknowledge interrupt to PIE
 PieCtrlRegs.PIEACK.all = PIEACK_GROUP3; //PIEACK-PIE ackonwledge register //中断应答
 return;
}
```

（3）程序 DSP2833x_ePwm_defines.h 中对某些寄存器的位的定义：

```
// TBCTL (Time-Base Control)
//=============================
// CTRMODE bits
```

```
#define    TB_COUNT_UP          0x0
#define    TB_COUNT_DOWN        0x1
#define    TB_COUNT_UPDOWN      0x2
#define    TB_FREEZE            0x3
// PHSEN bit
#define    TB_DISABLE           0x0
#define    TB_ENABLE            0x1
// PRDLD bit
#define    TB_SHADOW            0x0
#define    TB_IMMEDIATE         0x1
// SYNCOSEL bits
#define    TB_SYNC_IN           0x0
#define    TB_CTR_ZERO          0x1
#define    TB_CTR_CMPB          0x2
#define    TB_SYNC_DISABLE      0x3
// HSPCLKDIV and CLKDIV bits
#define    TB_DIV1              0x0
#define    TB_DIV2              0x1
#define    TB_DIV4              0x2
// PHSDIR bit
#define    TB_DOWN              0x0
#define    TB_UP                0x1

// ETSEL (Event Trigger Select)
//=============================
#define    ET_CTR_ZERO          0x1
#define    ET_CTR_PRD           0x2
#define    ET_CTRU_CMPA         0x4
#define    ET_CTRD_CMPA         0x5
#define    ET_CTRU_CMPB         0x6
#define    ET_CTRD_CMPB         0x7

// ETPS (Event Trigger Pre-scale)
//=============================
// INTPRD, SOCAPRD, SOCBPRD bits
#define    ET_DISABLE           0x0
#define    ET_1ST               0x1
#define    ET_2ND               0x2
#define    ET_3RD               0x3
```

2.指出寄存器 TBCTL 与 TBPRD 各字段的数值及其含义；

表格 3 TBCTL Register

字段	数值	含义
HSPCLKDIV	3	二者共同决定 time-base clock 预分频 TBCLK = SYSCLKOUT / (HSPCLKDIV × CLKDIV)
CLKDIV	0	
CTRMODE	2	time-base counter mode：Up-down-count mode.
PHSEN	0	Phase loading disabled.
PRDLD	0	TBPRD is loaded from its shadow register when the time-base counter , TBCTR, is equal to zero.
SYNCOSEL	3	Disable EPWMxSYNCO signal.

表格 4 TBPRD Register

字段	数值	含义
TBPRD	208	TB counter 的计数最大值 208（增减计数下:0-208，208-0）

3.指出时间基准模块 TB 产生事件的频率；

首先 CPU 复位默认频率为 150MHz, TBCTL Register 中 HSPCLKDIV=3，CLKDIV=0，根据 TBCLK 频率的计算表达式计算可得：

$$TBCLK = \frac{SYSCLKOUT}{HSPCLKDIV \times CLKDIV} = \frac{150MHz}{6\times 1} = 25MHz$$

又因为计数器采用增减计数模式且 TBPRD Register 中的计数值为 208，故时间频率为：

$$f_1 = \frac{TBCLK}{208\times 2} = \frac{25MHz}{416} = 60.096kHz$$

故：**时间基准模块 TB 产生事件的频率为** 60.096kHz

4.指出寄存器 ETSEL 和 ETPS 各字段的数值及其含义；

表格 5 ETSEL Register

字段	数值	含义
SOCAEN	1	Enable EPWMxSOCA pulse.
SOCASEL	2	Enable event time-base counter equal to period (TBCTR = TBPRD)
INTSEL	2	Enable event time-base counter equal to period (TBCTR = TBPRD)
INTEN	1	Enable EPWMx_INT generation

表格 6 ETPS Register

字段	数值	含义
SOCAPRD	3	Generate the EPWMxSOCA pulse on the third event: ETPS[SOCACNT] = 1,1
INTPRD	3	Generate interrupt on ETPS[INTCNT] = 1,1 (third event)

5.指出 ADCSOC 信号的产生频率；

根据 ETPS Register 中相关位的设置可知，事件触发子模块在第三个事件产生一个 EPWMxSOCA pulse 信号，因此 ADCSOC 信号的产生频率：

$$f = \frac{f_1}{3} = \frac{60.096kHz}{3} = 20.032kHz$$

[注]，f_1 为第 3 问中的时间基准模块 TB 产生事件的频率。

6.概括此程序运行后所产生的效果。

我认为此段程序可实现的功能是：产生一个频率约为 20kHz 的脉冲信号，并附有中断控制功能，因此可利用此程序进一步做 AD 采样的相关工作。