学生学号	0122120450310	**实验课成绩**	

武汉理工大学

学 生 实 验 报 告 书

实验课程名称	DSP 芯片原理与开发
开 课 学 院	信息工程学院
指导教师姓名	王克浩
学 生 姓 名	胡肖安
学生专业班级	电信 2101

2023 -- 2024 学年 第 二 学期

实验教学管理基本规范

实验是培养学生动手能力、分析解决问题能力的重要环节；实验报告是反映实验教学水平与质量的重要依据。为加强实验过程管理，改革实验成绩考核方法，改善实验教学效果，提高学生质量，特制定实验教学管理基本规范。

1、 本规范适用于理工科类专业实验课程，文、经、管、计算机类实验课程可根据具体情况参照执行或暂不执行。

2、 每门实验课程一般会包括许多实验项目，除非常简单的验证演示性实验项目可以不写实验报告外，其他实验项目均应按本格式完成实验报告。

3、 实验报告应由实验预习、实验过程、结果分析三大部分组成。每部分均在实验成绩中占一定比例。各部分成绩的观测点、考核目标、所占比例可参考附表执行。各专业也可以根据具体情况，调整考核内容和评分标准。

4、 学生必须在完成实验预习内容的前提下进行实验。教师要在实验过程中抽查学生预习情况，在学生离开实验室前，检查学生实验操作和记录情况，并在实验报告第二部分教师签字栏签名，以确保实验记录的真实性。

5、 教师应及时评阅学生的实验报告并给出各实验项目成绩，完整保存实验报告。在完成所有实验项目后，教师应按学生姓名将批改好的各实验项目实验报告装订成册，构成该实验课程总报告，按班级交课程承担单位（实验中心或实验室）保管存档。

6、 实验课程成绩按其类型采取百分制或优、良、中、及格和不及格五级评定。

附表：实验考核参考内容及标准

	观测点	考核目标	成绩组成
实验预习	1. 预习报告 2. 提问 3. 对于设计型实验，着重考查设计方案的科学性、可行性和创新性	对实验目的和基本原理的认识程度，对实验方案的设计能力	20%
实验过程	1. 是否按时参加实验 2. 对实验过程的熟悉程度 3. 对基本操作的规范程度 4. 对突发事件的应急处理能力 5. 实验原始记录的完整程度 6. 同学之间的团结协作精神	着重考查学生的实验态度、基本操作技能；严谨的治学态度、团结协作精神	30%
结果分析	1. 所分析结果是否用原始记录数据 2. 计算结果是否正确 3. 实验结果分析是否合理 4. 对于综合实验，各项内容之间是否有分析、比较与判断等	考查学生对实验数据处理和现象分析的能力；对专业知识的综合应用能力；事实求实的精神	50%

实验课程名称：DSP 芯片原理与开发

实验项目名称	数字振荡器设计实验			实验成绩	
实 验 者	胡肖安	专业班级	电信 2101	组　　别	
同 组 者				实验日期	2024.4.15

第一部分：实验预习报告（包括实验目的、意义，实验基本原理与方法，主要仪器设备及耗材，实验方案与技术路线等）

一、实验目的

1．学习数字振荡器的原理；

2．学习 C5402 定时器使用；

3．学习中断服务程序编写；

4．实现数字振荡器的设计。

二、实验内容

1．设计数字振荡器的算法；

2．综合运用各种知识在 C5402 芯片上实现数字振荡器的算法；

3．通过 CCS 提供的图形显示窗口观察输出信号波形以及频谱并分析实验结果。

三、实验原理与方法

1．数字振荡器原理

设正弦序列 x[k] = sin(kωT)，其 z 变换为

$$H(z)=\frac{Cz^{-1}}{1-Az^{-1}-Bz^{-2}}$$

其中，A=2cos(ωT)，B=-1，C=sin(ωT)。设初始条件为 0，求出上式的反 Z 变换得：

y[k]=Ay[k-1]+By[k-2]+Cx[k-1]

这是一个二阶差分方程，其单位冲击响应即为 sinkωT。利用单位冲击函数 x[k-1]的性质，即仅当 k=1 时，x[k-1]=1，代入上式得：

k=0　　y[0] = Ay[-1] + By[-2] + 0 = 0

k=1　　y[1] = Ay[0] + By[-2] + c = c

k=2　　y[2] = Ay[1] + By[0] + 0 = Ay[1]

k=3　　y[3] = Ay[2] + By[1]　　.

……　　.

k=n　　y[n]= Ay[n-1] + By[n-2]

在 k>2 以后，y[k]能用 y[k-1]和 y[k-2]算出，这是一个递归的差分方程。

根据上面的说明，我们可以开始数字振荡器的设计。设该振荡器的频率为 2kHz，采样率为 40kHz（通过定时器设置，每隔 25us 中断一次，即产生一个 y[n]），则递归的差分方程系数为：

A=2cos(ωT)=2cos (2 x PI x 2000 / 40000)=2 x 0.95105652

B=-1

C=sin(ωT)=sin (2 x PI x 2000 / 40000)=0.30901699

为了便于定点 DSP 处理，我们将所有的系数除以 2，然后用 16 位定点格式表示为：

$\frac{A}{2}\times 2^{15}=79BC$　　$\frac{B}{2}\times 2^{15}=C000$　　$\frac{C}{2}\times 2^{15}=13C7$

这便是本实验中产生 2kHz 正弦信号的三个系数。在本实验中，主程序在初始化时先计算出 y[1]和 y[2]，然后开放定时器中断。以后每次进入定时器中断服务程序时，利用前面的 y[1]和 y[2]，计算出新的有 y[0]，通过 CCS 提供的图形显示工具，我们将在图形窗口中看到一个正弦信号波形。下面是初始化和中断服务程序代码片段：

初始化 y[1]和 y[2]：

```
ssbx    FRCT          ；置 FRCT=1，准备进行小数乘法运算
st      #INIT_A,AA    ；将常数 A 装入变量 AA
st      #INIT_B,BB    ；将常数 B 装入变量 BB
st      #INIT_C,CC    ；将常数 C 装入变量 CC
pshd    CC            ；将变量 CC 压入堆栈
popd    y2            ；初始化 y2=CC
ld      AA, T         ；装 AA 到 T 寄存器
mpy     y2,a          ；y2 乘系数 A，结果放入 A 寄存器
sth     a,y1          ；将 A 寄存器的高 16 位存入变量 Y1
```

中断服务程序片段：

```
ld      BB,T          ；将系数 B 装入 T 寄存器
mpy     y2,a          ；y2 乘系数 B，结果放入 A 寄存器
ltd     y1            ；将 y1 装入 T 寄存器，同时复制到 y2
mac     AA,a          ；完成新正弦数据的计算，a 寄存器中为
                      ；y1*AA+y2*BB
sth     a,1,y1        ；将新数据存入 y1，因所有系数都除过 2，所以在保
                      ；存结果时转移一位，恢复数据正常大小。
sth     a,1,y0        ；将新正弦数据存入 y0
```

2．C54X 的定时器操作

C54X 的片内定时器利用 CLKOUT 时钟计数， 定时器主要由 3 个寄存器所组成：定时器寄存器(TIM，Timer Registers)、定时器周期寄存器(PRD，Timer Period Registers)和定时器控制寄存器(TCR，Timer Control Registers)。这 3 个寄存器都是 16 位存储器映像寄存器，在数据存储器中的地址分别为 0024H、0025H 和 0026H。TIM 是一个减 1 计数器；PRD 中存放时间常数；TCR 中包含有定时器的控制位和状态位。定时器的功能框图如图 1 所示。

图 1 中含一个 16 位的主计数器(TIM)和一个 4 位预定标计数器(PSC)。TIM 从周期寄存器 PRD 加载，PSC 从周期寄存器 TDDR 加载。

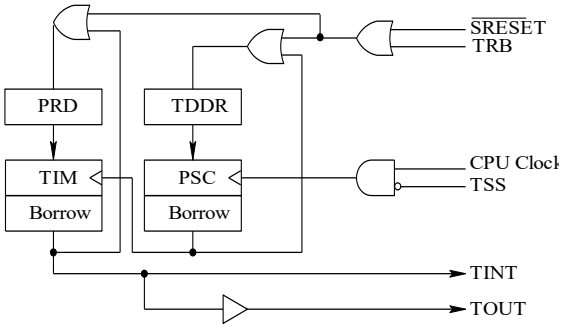


图 1 定时器的功能框图

定时器的典型操作顺序为

(1) 在每个 CLKOUT 脉冲后 PSC 减 1，直到它变为 0。

(2) 在下一个 CLKOUT 周期，TDDR 加载新的除计数值到 PSC，并使 TIM 减 1。

(3) 以同样方式，PSC 和 TIM 连续进行减操作，直到 TIM 减为 0。

(4) 在下一个 CLKOUT 周期，将定时器中断信号(TINT)送到 CPU，同时又用另一脉冲送到 TOUT 引脚，把新定时器计数值从 PRD 加载到 TIM，并使 PSC 再次减 1。

从上面的介绍可以看到定时器实际上可以有 20 个比特的周期寄存器。它对 CLKOUT 信号计数，先将 PSC 减 1，直到 PSC 为 0，然后用 TDDR 重新装入 PSC，同时将 TIM 减 1，直到 TIM 减为 0。这时 CPU 发出 TINT 中断，同时在 TOUT 引脚输出一个脉冲信号，脉冲宽度为 CLKOUT 一致。然后用 PRD 重新装入 TIM，重复下去直到系统或定时器复位。因而定时器中断的频率由下面的公式决定：

$$f_{TINT} = \frac{1}{t_c \times (TDDR+1) \times (PRD+1)}$$

其中 t_c 表示 CLKOUT 的周期。定时器当前的值可以通过读取 TIM 寄存器和 TCR 寄存器的 PSC 比特位得到。下面是本实验中初始化定时器的程序片段：

```
stm    #10h,TCR        ；停止定时器
stm #2499,PRD          ；设置 PRD 寄存器值为 2499，TINT 中断频率为
                       ；  Foutclk /（2499+1）= 100MHz/2500 = 40 KHz
stm #20h,TCR           ；  重新装入 TIM 和 PSC，然后启动定时器
```

3．C54X 中断的使用

在 C54X 中用户可以通过中断屏蔽寄存器 IMR 来决定开放或关闭一个中断请求。其中， TINT0 为定时器 0 中断。在中断屏蔽寄存器 IMR 中，1 表示允许 CPU 响应对应的中断，0 表示禁止。当然要 CPU 响应中断，ST1 寄存器中的 INTM 还应该为 0（允许所有的中断）。

当 DSP 响应中断时，PC 指针指向中断向量表中对应中断的地址，进入中断服务子程序。中断向量表是 C5402 存放中断服务程序的一段内存区域，大小为 80H。在中断向量表中，每一个中断占用 4 个字的空间，一般情况是将一条跳转或延时跳转指令存放于此。当然，如果中断服务程序很短（小于或等于 4 个字），可以直接放入该向量表。中断向量表的位置可以通过修改基地址来改变，其基地址由 PMST 寄存器中的 IPTR（15-7 bits）决定。例如 C54x 复位后其 IPTR 全为 1，所以中断向量表起始位置在 0FF80H，因而复位后程序从 0FF80H 开始运行。响应中断之后，CPU 将执行下列操作：

(1) 将 PC 值(即返回地址)压入堆栈。

(2) 将中断向量的地址装入 PC；将程序引导至中断服务程序 ISR。

(3) 现场保护，将某些要保护的寄存器和变量压入堆栈。

(4) 执行中断服务程序 ISR。

(5) 恢复现场，以逆序将所保护的寄存器和变量弹出堆栈。

(6) 中断返回，从堆栈弹出返回地址加载到 PC。

(7) 继续执行被中断的程序。

中断操作流程图如下：

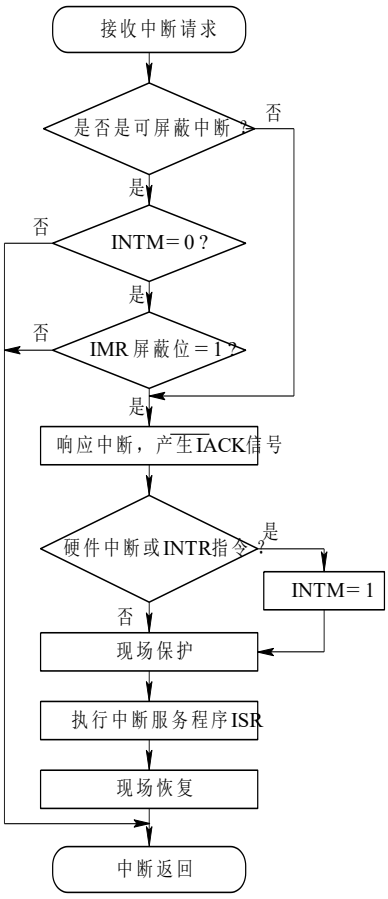


图 2 中断操作流程图

本实验的初始化程序读取中断向量表的起始地址，然后设置 PMST 的高 9 位，以便 DSP 能正确响应中断，代码如下：

```
    ld    #0,dp          ；设置 DP 页指针
   ssbx  intm            ；关闭所有中断
    ld    #vector, a     ；读出中断向量（地址 vector 在中断向量表程序中定义）
    and   #0FF80h, a     ；保留高 9 位（IPTR）
    andm  #007Fh, pmst   ；保留 PMST 的低 7 位
    or    pmst, a        ；
   stlm a, pmst          ；设置 PMST（其中包括 IPTR）
```

四、 实验仪器设备

1．PC 机一台；

2．CCS 开发软件一套；

3．DSP 教学实验系统一套。

第二部分：实验过程记录（可加页）（包括实验原始数据记录，实验现象记录，实验过程发现的问题等）

1．根据确定数字振荡器的频率，确定系数。数字振荡器系数的确定在前面已经说明，这里不再赘述。

2．启动 CCS，新建工程文件，如文件名为 nco.pjt。选择 Project 菜单中的 Add File to Project 选项，将汇编源程序 nco.asm、vec_table.asm 和连接定位 nco.cmd 文件依次添加到工程文件中。注意，你可以在添加文件对话框中选择显示不同的文件类型来加快文件选择速度。你也可以使用鼠标右键单击工程文件名（如 nco.pjt）并选择 Add Files 项来添加需要的文件。其中，nco.asm 包括初始化代码和中断服务程序，而 vec_table.asm 包含中断向量表。

3．选择 Project 菜单中的 Options 选项，或使用鼠标右键单击工程文件名（如 nco.pjt）并选择 build options 项来修改或添加编译、连接中使用的参数。选择 Linker 窗口，在“Output Filename”栏中写入输出 OUT 文件的名字，如 nco.out，你还可以设置生成的 MAP 文件名。

4．完成编译、连接，正确生成 OUT 文件。然后使用 File 菜单的“Load Program”选项，将生成的 OUT 文件（如 nco.out）装入 DSP 的片内存储器。这时 CCS 将显示程序的启始地址_c_int00。

5．选 View→Graph→Time/Frequency…打开图形显示设置窗口。在弹出的对话框中按下图设置，主要修改“Start Address”为 y0（y0 为生成的正弦波输出变量）；“Acquisition Buffer Size”为 1，“DSP Data Type”为“16-bit signed integer”。

6．在汇编源程序的中断服务程序（_tint）中的“nop”语句处设置断点。选择 Debug→Animate，运行程序，观察输出波形。数一数一个周期的正弦波有多少个点？算算频率是否是 2kHz？另外，想想 Run 和 Animate 两种运行方式的区别？

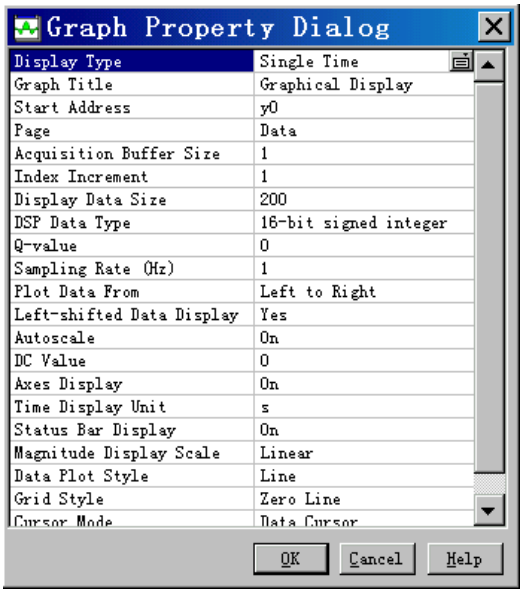


图 3 时域波形观察对话框

7．用右键单击图形显示窗口，并选择“Properties”项以便修改显示属性。将“Display Type”项改为“FFT Magnitude”以便显示信号频谱。修改“Sampling Rate(Hz)”项为 40000，然后退出。注意观察生成的正弦波频率。

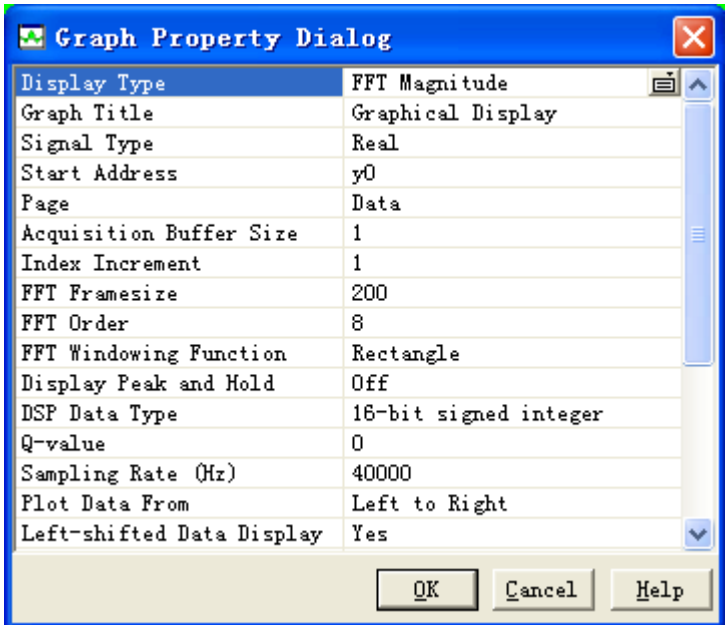


图 4 频谱观察对话框

1）主文件 noc.asm：

```
   .title  "for test INT service program  ...(25 us) "
     .mmregs
     .global _c_int00,_tint,vector
OFF_INTIMER     .set          04Ch              ; vector of INTtimer at
VECTOR+OFF_INTIMER
```

```
INIT_A          .set      079bch          ; A/2=0.9510498
INIT_B          .set      0c000h          ; B/2=-0.5
INIT_C          .set      013c7h          ; C/2=0.1545105
     .bss y0,1
     .bss y1,1
     .bss y2,1
     .bss temp,1
     .bss AA,1
     .bss BB,1
     .bss CC,1
 .text
_c_int00:
 ld   #0,dp
 ssbx intm
 ld   #vector, a              ; 获取中断向量表地址!
 and  #0FF80h, a
 andm #007Fh, pmst            ;
 or pmst, a
 stlm a, pmst                    ; 设置 IPTR

;***********定时器设置*************
 stm  #10h,TCR                ; 初始化定时器，不分频，停止定时器
 stm  #2499,PRD               ; f=100M/(2499+1)=40kHz
 stm  #20h,TCR                ;复位定时器 重新加载初值
 stm     #0008h,IMR           ;开中断

 STM    #0000H,IFR         ;通过写 IFR，取消所悬挂的中断
 nop
 stm  #0400h,TCR              ; 定时器使能
 nop
 rsbx   intm                  ; 状态寄存器 ST0 的 INTM 位，允许中断
;*************************************************************

 ld   #temp,dp; set DP
```

```
 ssbx FRCT                    ; 准备小数乘法
 st   #INIT_A,AA              ; 初始化 AA,BB,CC
 st   #INIT_B,BB              ;
 st   #INIT_C,CC              ;
 pshd CC
 popd y2                      ; 初始化 y2,y2=CC
 ld   AA,T                    ; T=AA
 mpy  y2,a                    ; y2*AA -> a
 sth  a,y1; y2*AA -> y1

again:
  nop
 b    again

 nop
 nop
 nop
 nop
 nop
 nop

;-------------------------------------------------------------
;  定时器中断处理程序
;-------------------------------------------------------------
_tint:
   ld   #BB,DP
   ld   BB,T              ; T=BB
   mpy  y2,a              ; a=y2*BB
   ltd  y1                ; T=y1,y2=y1
   mac  AA,a             ; a=a+y1*AA
   sth  a,1,y1; new cos data -> y1
   sth  a,1,y0            ; new cos data -> y0
   nop
   nop
```

```
   nop                  ; CCS 中设置断点!!!
int1_end:
    nop
    rete

    .end
```

2）中断向量表文件 vec_table.asm：

```
  .mmregs
;   .ref _ret
    .ref _c_int00
    .ref _tint
    .global vector
    .sect  ".int_table"
;-----------------------------------------------------------------------
; 中断向量表 !
;-----------------------------------------------------------------------
vector:
rs      b _c_int00
 nop
 nop
nmi     b __ret
 nop
 nop
sint17  b __ret
 nop
 nop
sint18  b __ret
 nop
 nop
sint19  b __ret
 nop
 nop
sint20  b __ret
```

```
        .word   0,0
sint21  b __ret
        .word   0,0
sint22  .word   01000h
        .word   0,0,0
sint23  .word   0ff80h
        .word   0,0,0
sint24  .word   01000h
        .word   0,0,0
sint25  .word   0ff80h
        .word   0,0,0
sint26  .word   01000h
        .word   0,0,0
sint27  .word   0ff80h
        .word   0,0,0
sint28  .word   01000h
        .word   0,0,0
sint29  .word   0ff80h
        .word   0,0,0
sint30  .word   01000h
        .word   0,0,0
int0    b __ret
 nop
 nop
int1    b __ret
 nop
 nop
int2    b __ret
 nop
 nop
tint    b _tint
       nop
       nop
       nop
```

```
brint0  b __ret
 nop
 nop
bxint0  b __ret
 nop
 nop
trint    b  __ret
 nop
 nop
dmac1   b __ret
 nop
 nop
int3    b __ret
 nop
 nop
hpint   b __ret
 nop
 nop
q26     .word   0ff80h
        .word   0,0,0
q27     .word   01000h
        .word   0,0,0
dmac4   b __ret
 nop
 nop
dmac5   b __ret
 nop
 nop
q30     .word   0ff80h
        .word   0,0,0
q31     .word   01000h
        .word   0,0,0
;-------------------------------------------------------------------------
----
```

```
; end of interrupte vector table !
;-------------------------------------------------------------------------
---


__ret  rete
```

第三部分　结果与讨论（可加页）

一、实验结果分析（包括数据处理、实验现象分析、影响因素讨论、综合分析和结论等）

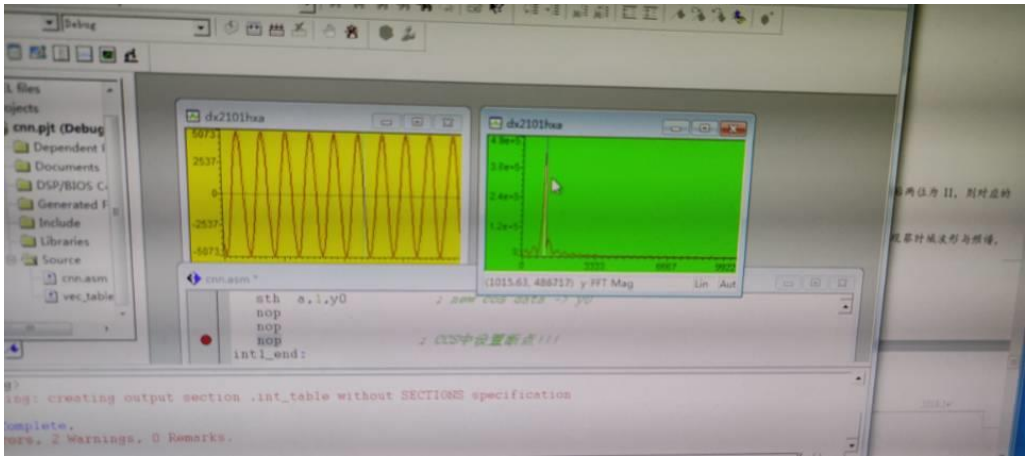
1. 实验结果分析

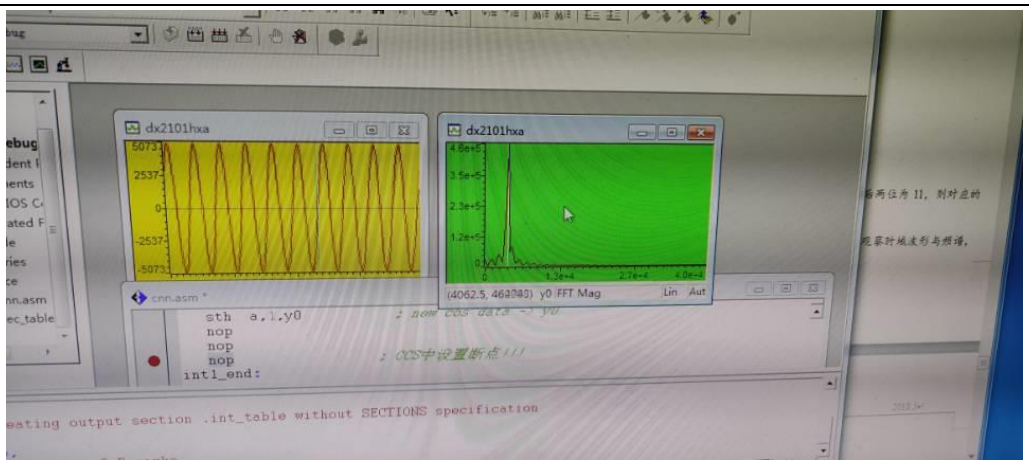
在本次实验中，我们学习了数字振荡器的原理并成功实现了一个基于 C5402 芯片的数字振荡器。我们采用了以下算法来设计数字振荡器：

1. 振幅大小（A）的确定：A 值即为生成正弦波的最大值。
2. 采样点数（B）的计算：B = 目标频率 / 采样频率。
3. 相位值（C）的计算：C = (相移大小 / 360) * B。

通过设置不同的参数，我们观察了输出信号的时域波形和频谱，并进行了分析。

由于我的学号后位为 10 我们将振荡器频率设置为 1.0kHz，并修改 PRD 值为 71428，观察到相应的输出频率变化。同时，我们设置了两种不同的采样频率（20kHz 和 80kHz），并对比了它们在时域和频域上的表现。结果显示，当信号频率为 40kHz 时，采样频率为 80kHz 时所得图像更加完整和清晰，这反映了采样频率对输出信号的影响。





2. 实验现象分析

在实验过程中，我们观察到了几个重要的实验现象：

1. **不同参数设置下的输出变化**：我们通过调整振幅大小、采样点数和相位等参数，观察到了输出信号的频率、幅值和相位随参数变化而变化的情况。例如，增大振幅大小会使输出信号的幅值增大；增大采样点数会使输出信号的频率增加；调整相位会改变输出信号的相位延迟。

2. **采样频率对输出信号的影响**：我们设置了不同的采样频率（20kHz 和 80kHz），并观察了输出信号在时域和频域上的表现。在采样频率为 80kHz 时，输出信号的时域波形更加清晰且频谱更加完整，与采样频率为 20kHz 时相比有明显的改善。

3. 影响因素讨论

通过分析实验现象，我们可以推导出以下结论和影响因素：

1. **振幅大小对输出信号的影响**：增大振幅大小会使输出信号的幅值增大，从而改变信号的强度和清晰度。

2. **采样点数对输出信号的影响**：增大采样点数会使输出信号的频率增加，从而改变信号的周期性和分辨率。

3. **采样频率对输出信号的影响**：采样频率决定了信号的采样率和频谱分辨率，较高的采样频率可以更准确地捕捉信号的变化。

4. 综合分析与结论

在本次实验中，我们着重学习了数字振荡器的原理，并通过在 C5402 芯片上实现了一个数字振荡器。在实验中，我们调整了振幅大小、采样点数和相位设置等参数，观察并分析了它们对输出信号的影响。我们发现，不同参数设置下，输出信号的频率、幅值和相位都有所变化。例如，增大振幅大小会使输出信号的幅值增加，而增大采样点数则会增加输出信号的频率。通过调整这些参数，我们可以优化数字振荡器的性能和输出效果。

在观察输出信号时，我们特别关注了采样频率对信号清晰度和完整性的影响。通过比较不同采样频率下的输出结果，我们发现较高的采样频率能够更准确地捕捉信号的细节和变化，使得输出信号的时域波形和频谱更加清晰和完整。这一发现对于实际应用中数字信号处理的精度和质量至关重要。

总的来说，本次实验加深了我对 C5402 芯片和 DSP 原理的理解还提升了我对 CCS 工具的使用能力，提升比较大