高效利用 DSP 技术奖

三相交流参数智能测试仪

注册编号：MCHP16bitCDC0002

参赛队员：高尚勇
伍元彪　　李　超

1 项目意义

发电机电参量测试是出厂试验必不可少的一部分。在电机试验中，采用模拟表计进行发电机电参量测试仍然很普遍，使用模拟表在进行试验时需要较多的测量仪表和工作人员相互配合，且在测试过程中无法检测到波形畸变率及序分量。对于发电机来说，国际电工委员会(IEC)规定，发电机实际的端电压波形在任何瞬间与基波波形之差不得大于基波幅值的5%，同时对发电机端电压的对称性也有严格的要求。因此发出电压的波形畸变率及序分量是检测发电机性能的重要参数。本系统在对美国微芯公司的dsPIC30F系列数字信号控制器分析后，利用其强大的数据处理能力将谐波及序分量分析引入到发电机电参量测试系统中。

Microchip的16位数字信号控制器dsPIC30F系列芯片融合了高性能16位MCU的控制优势和完全实现的DSP高运算速度，形成了适合嵌入式系统设计的紧密结合的单芯片、单指令流解决方案。该系列芯片提供了强大的16位单片机所具备的所有功能。dsPIC30F系列芯片硬件资源丰富，硬件设计简单，可以对数字信号流执行快速的数学运算，包括简单的加减法、乘法、复杂滤波以及信号分析功能如快速傅立叶变换和离散余弦变换等。

我国发电系统的参数测试方面一直都在尝试，出现了很多智能测试装置，但国产的发电机专用测试装置在功能上都难以保证现场的需求，进口的装置一般以通用仪器为主，对需测试的某个参量讲是合适的，专用的装置很难见到。这是发电机试验项目、方法以及执行相关标准的差异所致，况且国外仪器普遍价格比较昂贵，给使用单位的购置造成了困难。次装置的研制成功对我国发电系统的参数测试有一定的推动作用。

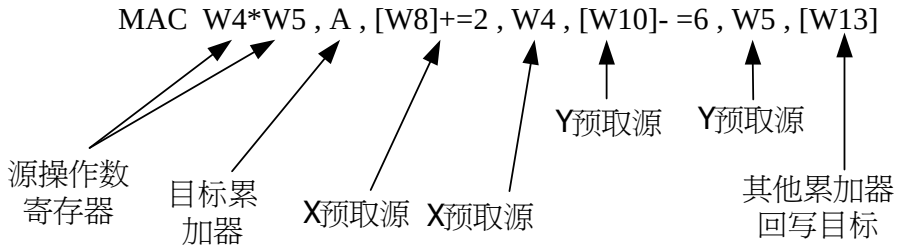
2 测试装置的硬件设计

本装置要实现发电机某些电量的测试，涉及的计算参数较多，既要计算电机的电压、电流、功率等参数，还要对各路电压或电流进行谐波分析，而且要在液晶显示屏上显示结果，因此采用何种处理器来解决大数据量的运算实时性、FFT运算的高效性等问题成为硬件设计一个关键的环节。

dsPIC30F 系列 DSC 分为通用系列、电机控制和电源变换系列及传感器系列三个系列，其中通用系列中的 dsPIC30F6014 芯片最适合用作测试系统的核心微处理器，针对本测试系统的实时性及精度要求，选择 dsPIC30F6014 的主要原因有以下几点：

执行速度高达 30MIPS(百万条指令每秒)，即一个指令周期可达近 33.33ns。有利于实现测试系统的实时性。

单指令周期乘加指令MAC，一个指令周期可以完成以下操作，在完成本次乘加操作的同时，还可以同时预取两个操作数用于下一条乘法指令：



③ 可灵活选择多种时钟模式。具有片上PLL时钟倍频器，dsPIC30F6014 可工作在一个很宽的频率范围内，可设置具有 4 倍、8 倍及 16 倍频，输出频率范围可达 64MHz~120MHz，大大提高了dsPIC30F6014 的处理速度。

④ 定时器：5 个 16 位定时器/计数器，其中有些可配对做 32 位定时器使用。

⑤ 模数转换器：16 通道 12 位A/D转换器模块，其精度可达 0.02%，最高转换速度可配置为 200ksps(Kilo Samples per Second)，并且有四种结果对齐方式。dsPIC30F系列的电机控制芯片dsPIC30F6010 虽然有四个采样保持器，可以进行四路同时采样，但是其A/D只有 10 位，精度只能到 0.1%。从测量精度方面考虑，本测试装置没有选择dsPIC30F6010 芯片作为主控芯片。

⑥ 程序空间可视性（PSV）：PSV是dsPIC30F系列DSC的一大优势，可以将 32KB的程序存储空间映射进数据存储空间的上部 32KB，允许任何指令像访问数据存储空间一样访问程序存储空间。像LCD图形和菜单、大型固定数据阵列、FFT旋转因子等许多数据常量都可以充分利用这一性能优势。由于本测试系统需

要对 6 路数据进行FFT运算，并且要显示大量数据及文字，需要很多数组来存储运算数据及点阵，所以对数据存储器的大小有较高的要求，而PSV正好扩展了数据存储器。因此，这也是本测试装置选择dsPIC30F系列DSC的一个重要原因。

⑦ dsPIC30F的DSP库函数提供了 49 个函数，其中就包括进行FFT运算的矢量函数和变换函数，为进行FFT运算节约了大量的时间，这是本测试装置选择dsPIC30F系列DSC的另一个重要的原因。

综上所述，由于dsPIC30F6014兼具16位单片机的控制优势和DSP的高速运算优势，以及针对本测试系统所具有的突出特点，因此，选用dsPIC30F6014作为测试装置的主控芯片。

2.1 测试装置的总体设计

本系统所设计的测试系统的硬件框图如图 1 所示，主要有以下几个模块：频率检测模块、A/D 采样模块、人机接口模块、通信接口等。

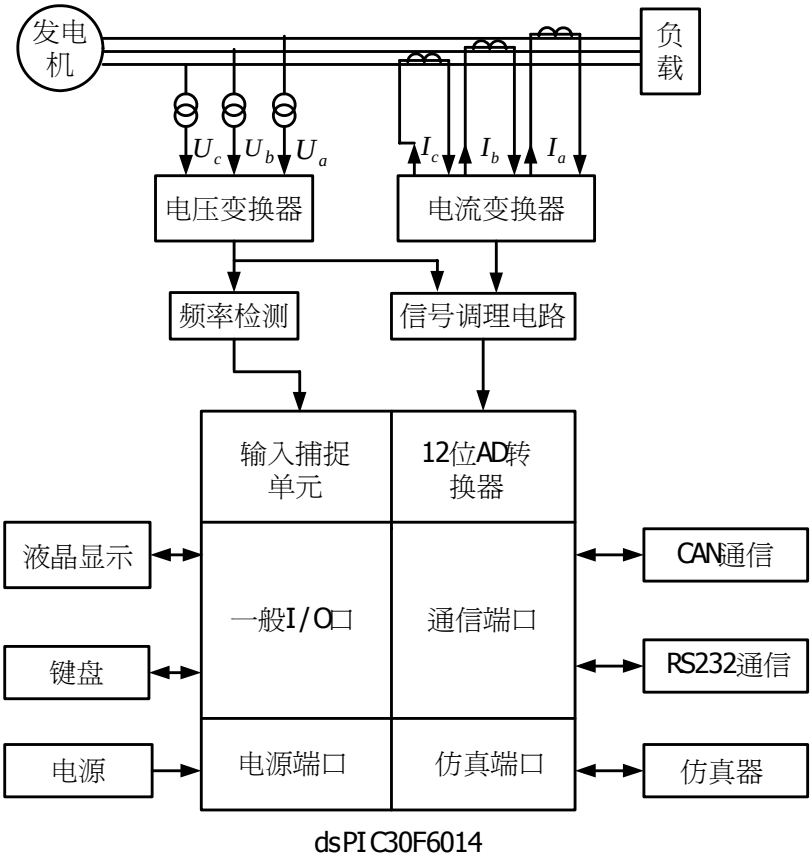


图 1 测试装置的总体硬件框图

发电机的定子出线端三相交流电压、电流分别经过电压、电流互感器变换成有效值为 100V、5A 交流电压、电流，再通过电压(电流)变换器转换成为 $\pm 5V$ 的

交流电压，经过调理电路转换成 0～5V 的电压后，接入 dsPIC30F6014 的 A/D 采样通道。为了实现对调理后的信号进行自适应调整采样时间间隔以实现等间隔采样，特别设计了频率检测电路。dsPIC30F6014 实时处理采样值，并将结果显示在液晶屏上，并通过触摸屏来实现翻屏的功能。

2.2 模拟量采样及调理电路设计

在工程实际中，为了防止各类干扰通常需要对装置中的输入信号通道采取必要的隔离，以保证装置的可靠运行。本测试装置采用湖北天门天瑞电子有限公司生产的120V/3.53V的精密电压变换器和5A/3.53V的电流变换器，在电路中起到变压、变流和电气隔离的作用。它们的优点是价格便宜、无需工作电源；缺点是只能测量交流信号。本装置没有对直流信号进行测量，因此选用这类变换器可以满足要求。电压及电流变换器的参数如表1所示：

表1 电压及电流变换器参数

规格	最大测量电压（V）	输入电压额定值（V）	输出电压额定值（V）	非线性度		空载电流（mA）	工频耐压（V）
				比差（±%）	角差（±分）		
120V/3.53V	140	120	3.53	<0.1	≤5	<0.5	>2500
5A/3.53V	6	5	3.53	≤0.1	≤5	<0.5	>2500

由于dsPIC30F6014的A/D模块具有16个12位的A/D转换通道，其最高转换速度可达200ksps，并有四种转换结果对齐方式。为了简化硬件电路，同时dsPIC30F6014内部的A/D模块又能够满足测试精度要求，因此本装置采用dsPIC30F6014内部的A/D模块来完成模拟量采样电路的设计。dsPIC30F6014的A/D模块输入引脚的电压范围为0～5V，而电压变换器和电流变换器的输出电压都为-5V～＋5V，故需要将电压变换器和电流变换器输出的电压信号经调理电路后转换成0～5V的电压信号后再接入dsPIC30F6014芯片的A/D引脚。

调理电路采用加提升电压的二阶巴特沃思有源低通滤波电路。理想滤波电路的频响在通带内应具有最大幅值和线性相移，而在阻带内其幅值应为零。实际的滤波电路往往难以达到理想的要求。如要同时在幅频和相频响应两方面都满足要求就更为困难。因此，只有根据不同的实际需要，寻求最佳的近似理想特性。巴特沃思滤波电路对幅频响应的要求是：在小于截止频率的范围内具有最平幅度的响应，而在大于截止频率的范围内，幅频响应迅速下降。

图2就是相应的调理电路，其截止频率为8.5kHz，增益为0.43。ADCA0为电压（电流）变换器的输出，ADCINA0接dsPIC30F6014的A/D输入引脚。

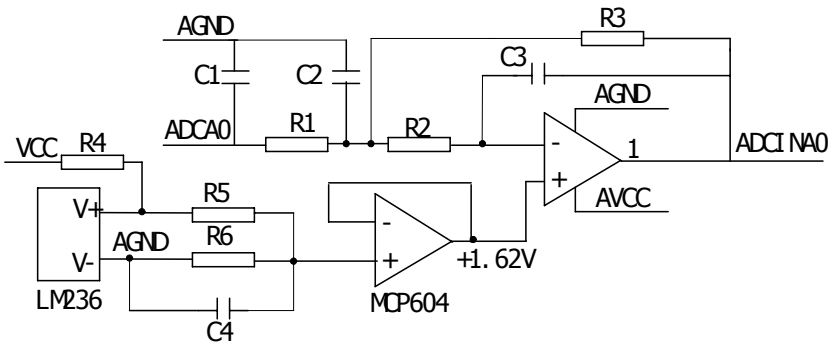


图 2 信号调理及电压提升电路

以信号

$$u = 5\sin(2\pi f_1 t) + 0.5\sin(2\pi f_3 t) + 0.2\sin(2\pi f_5 t) + 0.2\sin(2\pi f_{11} t) +$$

$$0.2\sin(2\pi f_{22} t) + 0.2\sin(2\pi f_{33} t) + 0.2\sin(2\pi f_{44} t) + 0.2\sin(2\pi f_{55} t) + 0.2\sin(2\pi f_{64} t)$$

$$+ 0.2\sin(2\pi f_{80} t) + 0.2\sin(2\pi f_{128} t)$$ 作为输入信号进行调理电路的 saber 仿真。

图 3 为输入电压信号波形，图 4 为图 2 电路输出电压波形，可以看出经过滤波电路以后，信号的毛刺明显减少，且输出信号幅值也满足 dsPIC30F6014 的 A/D 模块要求的 0～5V 范围内。

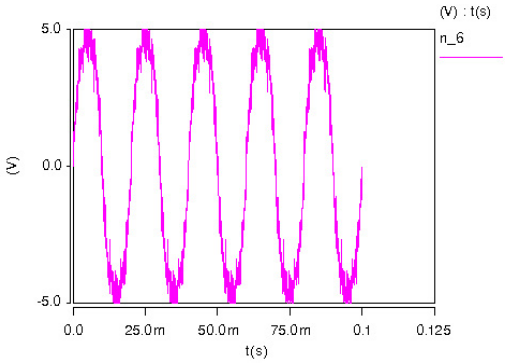


图 3 图 2 中输入电压信号

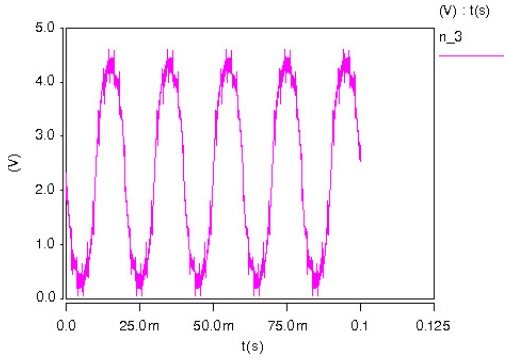


图 4 图 2 中输出电压信号

2.3 频率检测电路设计

频率检测电路如图 5 示，输入信号经过前级处理电路后，输出信号 ADCINA0，ADCINA0 再经过一个电压比较电路输出方波，并接到 dsPIC30F6014 的输入捕捉引脚。在频率测量时，为了提高测量的分辨率以保证测量精度，用 16 倍频的 6MHz 系统主时钟作为定时器 3 捕捉模块的时钟源，这时指令周期时钟为 24MHz，响应中断的时间约为 4 个指令周期，即 0.167μs，对约 20ms 的基频周期来说，产生的误差是非常小的，对测量精度的影响很小，可以忽略不计。

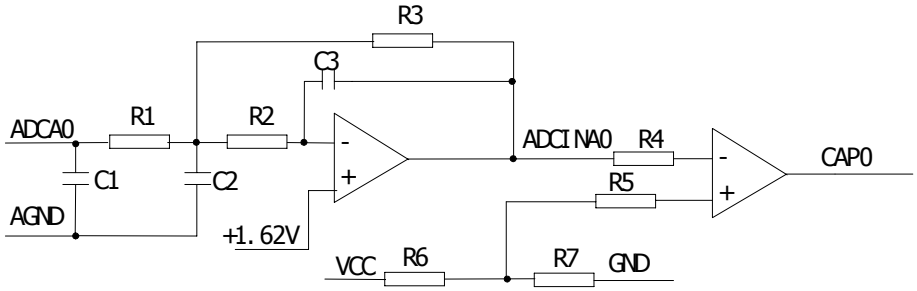


图 5 频率检测电路

2.4 通信接口电路设计

dsPIC30F6014 片内集成了两个完整的 CAN 控制器，支持 CAN1.2、CAN2.0A、CAN2.0B Passive 和 CAN2.0B Active 协议。本测试装置充分利用 dsPIC30F6014 的片上资源，扩展出 2 路 CAN 通信接口，其接口电路结构如图 6 所示，图中给出了第一路 CAN 通信接口电路。

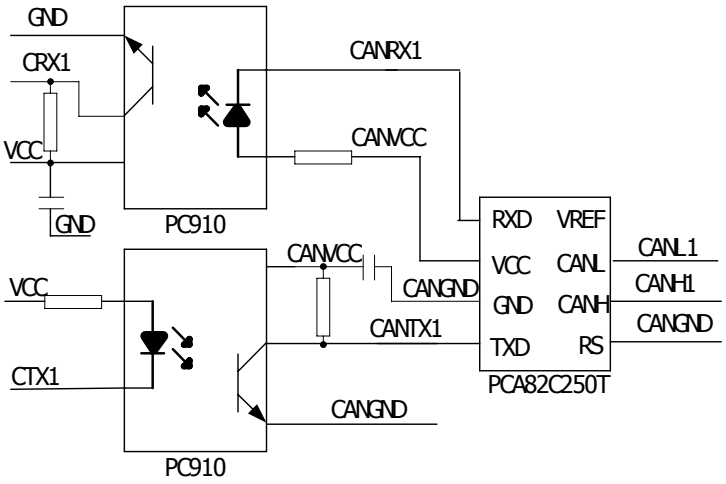


图 6 CAN 通信模块接口电路

2.5 触摸屏显示器控制电路

液晶显示模块为 CA320240，控制芯片为 RA8803, RA8803 是一个双图层(Two Pages)中英文文字与绘图模式的点矩阵液晶显示(LCD)控制器，内建 7602 个常用简体字库，国家标准 GB 码字库；内建对比度调节电路，可软件设置对比度；内建多组半宽字符(ASCII 码)，方便编程；内建粗体字型和行距设定；提供显示屏幕水平卷动和垂直拖动功能；提供单个字符反白显示和 N 行反白显示；提供简单 4 级灰度显示功能；提供中英文对齐/不对齐功能；提供触摸屏控制功能；全屏幕点阵，可支持最大点阵数为 320(列)×240(行),可显示 20(列)×15(行)个(16×16 点阵)汉字，也可完成图形，字符的显示。与 CPU 接口采用 5 条位控制总线和 8 位并行数据总线输入输出，可适配 Intel8080 时序或者 M6800 时序；内部有显示数据锁存器；简单的操作指令，每个指令为一个寄存器，写入数值即相当于指令输入。

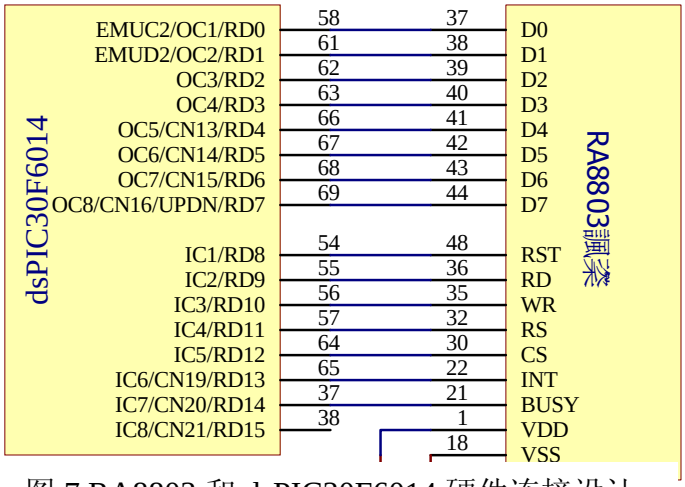


图 7 RA8803 和 dsPIC30F6014 硬件连接设计

2.6 硬件装置实物图

装置的实物图如下图所示：（注：详细的操作说明请参考附件《电参量测试装置使用说明书 V1.0》）

图 8 硬件装置实物图

3 测试装置的软件设计

对于一个测试系统，软件设计占有举足轻重的地位。在硬件的基础上，好的软件设计能够充分发挥微控制器的运算和逻辑控制功能，从而提高测试的精度和应用的方便性。

3.1 软件总体设计思路

测试系统总流程图如图 9 所示。其工作过程是：系统上电复位后，dsPIC30F6014 首先对自身的工作条件、环境进行设置，依次对片内的外设(包括 I/O 端口、A/D 模块、CAN 控制器和“看门狗”等)进行初始化，利用 dsPIC30F6014 的捕获单元来实现频率跟踪。而后采用 FFT 算法完成对发电机的电压、电流信号的谐波分析，并进一步计算发电机电压电流的有效值、功率、功率因数、序分量和波形畸变率等参数。液晶用于显示发电机电压电流有效值、功率、功率因数、序分量结果以及谐波分析的结果，显示界面的切换由触摸屏控制。报文发送既可以通过按触摸控制实现，也可以通过上位机进行控制，将测试结果上传到 PC 机上的上位机软件上进行数据的管理。

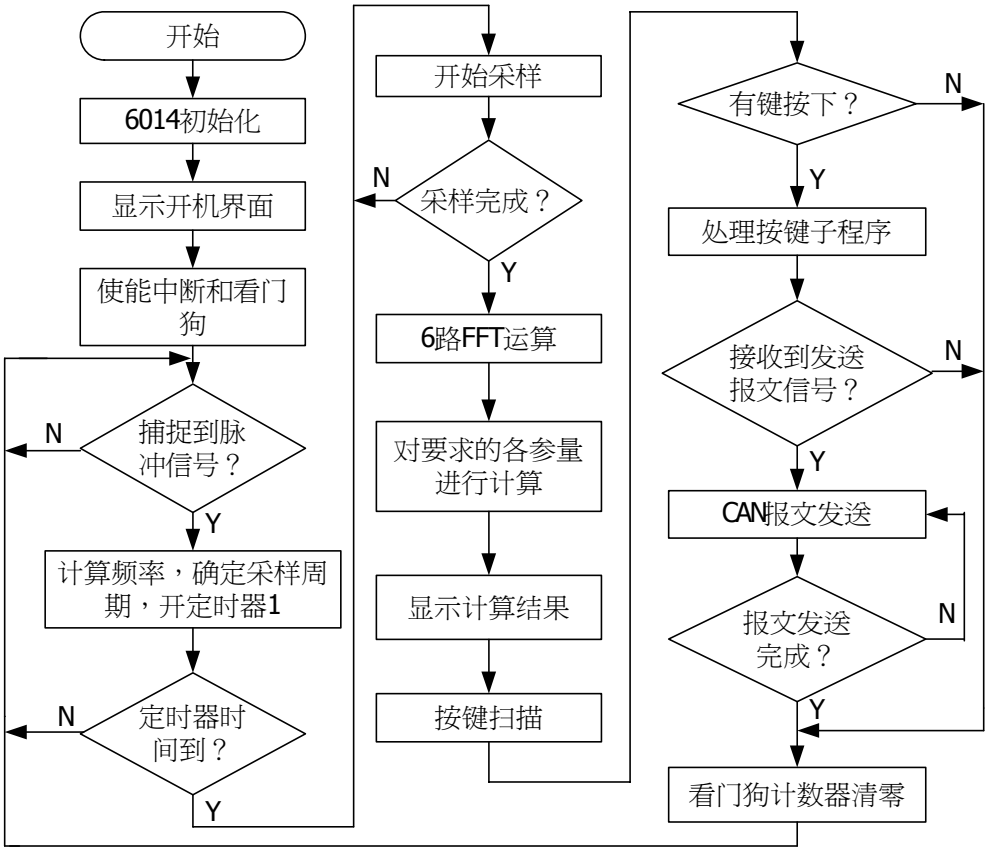


图 9 系统软件总流程图

3.2 FFT 算法的实现

本系统的 FFT 运算使用 DSC 的 DSP 函数库中的 FFT 函数。DSP 函数库共支持 49 个函数。DSP 函数库主要由优化的汇编语言编写。与 ANSI C 相比，在代码大小相同的情况下，应用 DSP 函数库可以大大地提高执行速度。另外，由于 DSP 函数库已经经过严格的测试，应用 DSP 函数库将缩短应用程序的开发时间。

DSP 函数库提供的变换函数中的 FFT 函数使用的是复数数据集合，复数矢量通过数据集合来表示，在该数据集合中，矢量中每个元素由两个值组成。其中第一个值是元素的实部，第二个值是元素的虚部。用存储器的一个字（2 个字节）来存储实部和虚部，且必须都表示为“1.15”小数格式。与小数矢量一样，小数型复数矢量将其元素连续地存放在存储器中。

小数型复数矢量的数据结构如下：

```
#ifdef  fractional
#ifndef  fractcomplex
typedef  struct {
                fractional  real;
                fractional  imag;
                } fractcomplex;
#endif
```

就是说，测试装置采样进来的数据需要转化为 1.15 格式小数，旋转因子需要转换为 1.15 格式小数放在一个数组中。

FFT 计算需要用到以下函数：

TwidFactorInit：生成离散傅里叶变换(DFT)所需旋转因子集合的前半部分，并将结果存放在复数目标矢量中。返回值为指向旋转因子基地址的指针。在程序中，将因子存放在可视化的程序空间中，可以节省 RAM 空间。

FFTComplexIP：计算源复数矢量的离散傅里叶变换，计算结果丢回“原址”(In Place)，“原址”运算指输入序列(在物理上)被变换后的序列替代，这样可以节约存储空间。参数 N 必须是 2 的整数次幂。要求源复数矢量中的元素按照自然顺序存储。变换结果以位反转顺序存放。为了避免在计算程中出现饱和(溢出)，源复数矢量值的幅值应该在[-0.5,0.5] ([0xC000,0x3FFF])范围内。输出应乘以因子 1/N 进行定标。注意源复数矢量应分配到 Y 数据空间。

BitReverseComplex：以位反转顺序重新组织复数矢量元素，即码位倒置。该函数为“原址”运算。

以下为微芯公司网站提供的 FFT 程序段代码，其中加深的字段为错误段：

```
for ( i = 0; i < FFT_BLOCK_LENGTH; i++ )/* The FFT function requires input data
{                                /* to be in the fractional fixed-point range [-0.5, +0.5]*/
    *p_real = *p_real >>1 ;      /* So, we shift all data samples by 1 bit to the right. */
    *p_real++;                   /* Should you desire to optimize this process, perform */
}                                /* data scaling when first obtaining the time samples */
p_real = &sigCmpx[(FFT_BLOCK_LENGTH/2)-1].real ;/* Set up pointers to convert real array */
p_cmpx = &sigCmpx[FFT_BLOCK_LENGTH-1] ; /* to a complex array. */
for ( i = FFT_BLOCK_LENGTH; i > 0; i-- ) /* Convert the Real input sample array */
{                            /* to a Complex input sample array   */
    (*p_cmpx).real = (*p_real--);   /* We will simpy zero out the imaginary   */
    (*p_cmpx--).imag = 0x0000;   /* part of each data sample */
}/* the real input samples followed by a series of zeros */
/* Perform FFT operation */
#ifndef FFTTWIDCOEFFS_IN_PROGMEM
    FFTComplexIP (LOG2_BLOCK_LENGTH, &sigCmpx[0], &twiddleFactors[0], COEFFS_IN_DATA);
#else
    FFTComplexIP (LOG2_BLOCK_LENGTH, &sigCmpx[0], (fractcomplex *) __builtin_psvoffset(&twiddleFactors[0]), (int) __builtin_psvpage(&twiddleFactors[0]));
#endif
BitReverseComplex (LOG2_BLOCK_LENGTH, &sigCmpx[0]); /* Store output samples in bit-reversed order of their addresses */
SquareMagnitudeCplx(FFT_BLOCK_LENGTH, &sigCmpx[0], &sigCmpx[0].real); /* Compute the square magnitude of the complex FFT output array so we have a Real
```

output vetor */

调试中发现此段程序代码不能正确实现 FFT 运算，经过仔细研究发现错误段主要产生在复数数组变换以及复数矢量的幅值计算中，对上面程序段复数数组变换部分进行更正，其中斜体并加深字段为修改过的程序段：

for (i =0 ; i < FFT_BLOCK_LENGTH; i++)

{ (*p_cmpx).real = *p_real >>1 ; /* fractional fixed-point range [-0.5, +0.5]*/

****p_real++;***

(*p_cmpx++).imag = 0x0000; } /* the imaginary was writed s series of zero*/

p_real = &sigCmpx[0].real ;

p_cmpx = &sigCmpx[0] ;

SquareMagnitudeCplx()函数为一段计算幅值的汇编代码，目的是为快速实现乘加计算，SquareMagnitudeCplx()函数中对累加器的操作错误，产生了错误的结果，修改后 SquareMagnitudeCplx()函数的程序代码如下，其中 W10 寄存器通过 Y 地址发生单元进行操作并寻址 Y 数据空间：

```
.global _SquareMagnitudeCplx          ; provide global scope to routine
_SquareMagnitudeCplx:
    push    w2                        ; save return value (dstV)
    push.d  w4
    push    w10
    push    CORCON                    ; Prepare CORCON for fractional computation.
    mov     w1, w10                   ; w10 points to beginning of complex input array
    clr     a, [w10]+=2, w4           ; Fetch Real(srcV[0]) and increment pointer to array
    dec2    w0, w0                    ; w0 = numElems-1
    do      w0, L0
    mpy     w4*w4, a, [w10]+=2, w5    ; accA = Real(srcV[n])^2, fetch Imag(srcV[n])
    mac     w5*w5, a, [w10]+=2, w4    ; accA = Real(srcV[n])^2 + Imag(srcV[n])^2
    SFTAC   a, #-15                   ; left shift 15 bits
    sac.r   a, #0, [w2++]             ; round, saturate and store result in dstV[n] array,
    ; then add to 2
    L0: add     #0x02, w2
```

```
; Last iteration outside the DO loop
    mpy     w4*w4, a, [w10]+=2, w5   ; accA = Real(srcV[n])^2, fetch Imag(srcV[n])
    mac     w5*w5, a                 ; accA = Real(srcV[n])^2 + Imag(srcV[n])^2
    SFTAC   a, #-15                  ;left shift 15 bits
    sac.r   a, #0, [w2]              ; round, saturate and store result in dstV[n] array
    pop     CORCON                   ; restore CORCON.
    pop     w10                      ; Restore working registers.
    pop.d   w4
    pop     w0                       ; restore return value.
    return
    .end
```

通过在提供的函数基础上修正后，可以进行FFT运算，但计算结果误差很大，以 $y = 0.2 + 0.65\sin(\omega t + 30^\circ) + 0.473\sin(3\omega t - 38^\circ) + 0.717\cos(7\omega t - 44^\circ)$ 进行FFT分析，计算结果如表2所示。

表2修正后FFT运算结果

Tab2 The FFT arithmetic results after amendment

	256点	相对误差	128点	相对误差	64点	相对误差
直流幅值	0.2031	1.55%	0.2031	1.55%	0.1875	-6.20%
基波幅值	0.6562	0.95%	0.6435	-1.00%	0.6435	-1.00%
基波相角	30.26°	0.86%	29.71°	-0.97%	29.71°	-0.97%
3次谐波幅值	0.4688	-0.89%	0.4688	-0.89%	0.4880	3.49%
3次谐波相角	-38.37°	0.97%	-37.64°	-0.94%	-38.81°	2.13%
7次谐波幅值	0.7188	0.25%	0.7188	0.25%	0.7071	-1.38%
7次谐波相角	45.89°	-0.26%	45.68°	-0.69%	45.68°	-0.69%

从表中可以看出不论是 64 点还是 256 点，均不满足测试精度要求。因此，在修正后的 FFT 函数的基础上又进一步做了改进，显著提高了计算精度。基本 FFT 程序段代码如下：

```
for (k=0; k<FFT_BLOCK_LENGTH-1; k++)
{    p_real++;
     if(*p_real > Real_max)          Real_max = *p_real;
      else if(*p_real < Real_min)    Real_min = *p_real;   }
(Real_min) = -(Real_min);
```

```
if ((Real_max) < (Real_min))          (Real_max)= (Real_min);
if((Real_max) > 16383)                {ShiftCnt = -1; }
else if((Real_max) < 8192)
{     if((Real_max) > 512)
      { while((Real_max) < 8192)
            {(Real_max) = (Real_max << 1);
                  ShiftCnt++;}  } }
for (k=0; k<FFT_BLOCK_LENGTH; k++)
{     if(ShiftCnt < 0)
      {   (*p_cmpx).real = *pp_real >> 1; }
      else
      {   (*p_cmpx).real = *pp_real << ShiftCnt; }
      *pp_real++;
       (*p_cmpx++).imag = 0;     }
TwidFactorInit(LOG2_BLOCK_LENGTH, &twiddleFactors[0], 0);
p_real = &sigCmpx[0].real ;
p_cmpx = &sigCmpx[0] ;
FFTComplexIP(LOG2_BLOCK_LENGTH, p_cmpx, &twiddleFactors[0],
COEFFS_IN_DATA);
BitReverseComplex(LOG2_BLOCK_LENGTH, p_cmpx);
  for (k=0; k<FFT_BLOCK_LENGTH; k++)
  {     if(ShiftCnt < 0)
    {   sigCmpx[k].real= sigCmpx[k].real;
        sigCmpx[k].imag= sigCmpx[k].imag;   }
    else   {   sigCmpx[k].real=(sigCmpx[k].real>>(ShiftCnt+1));
        sigCmpx[k].imag=(sigCmpx[k].imag>>(ShiftCnt+1));     }   }
  p_cmpx = &sigCmpx[0] ;
  pp_real =parameter_sigReal ;
ComputeMag (p_cmpx, pp_real, FFT_BLOCK_LENGTH/2);
```

本文利用以上函数编写了更通用的 FFT 函数，函数声明为 void FFTFun(fractional *parameter_sigReal,float *Arg,float *THD)，通过调用本函数，不但能一次计算出各次谐波幅值，同时还可以得出谐波相角及各通道波形畸变率。

结合本装置要求及应用，综合精度和时间的因素，本文采用 128 点进行 FFT 分析，即在一个周波内采样 128 点分析，以尽量同时满足实时性和计算精度的要求。

3.3 A/D 模块程序设计

确定了采样点数，并完成硬件设计和检验，就可以开始进行 A/D 模块的编程了。要达到 200ksps 采样/转换速率，参考电压源必须使用外部电压，但在硬件设计初，并没有提供外部电压源，而是使用 AV_{DD} 和 AV_{SS}，因此每通道的采样转换速率可能不能达到 5μs，因此，对 6 路通道 A/D 采样/转换时间通过定时器 3 进行总的计时：A/D 采样开始打开定时器 3，6 路采样/转换结束后关闭定时器 3，此时 TMR3=0X02DA=730，并且未进入 T3 中断。则可以计算出 6 路 A/D 采样/转换总共需要时间为 730/24μs=30.4167μs。因此，可以认为每路 A/D 采样/转换时间近似 5μs。

A/D转换由定时器1周期启动，完成所有通道的A/D转换后产生中断，图4.2为A/D中断流程。进入中断以后，如果变量Zu=0，则将采样数据处理并存入Sig1[6][128]，二维数组Sig1[6][128]是1.15格式小数，故需要将采样数据转换成小数后存入Sig1[6][128]。当Sig1[6][128]数据存满，即K1>127以后，此时主程序便可以调用FFT函数及其它电参量计算函数，这些计算都是针对Sig1[6][128]内的数据进行的。同时K1>127以后，置Zu=1，将采样数据处理并存入Sig2[6][128]，当Sig2[6][128]数据存满，即K2>127以后，此时主程序便可以调用FFT函数及其他电参量计算函数，这些计算是针对Sig2[6][128]内的数据进行的。

也就是指，第二周波的数据采样与上一周波数据处理同时进行，当第二周波的数据采样完成开始处理时，同时进行下一周波的采样，并放在另一数组内。这样，使用两个数组就避免了数据丢失，保证了实时性。

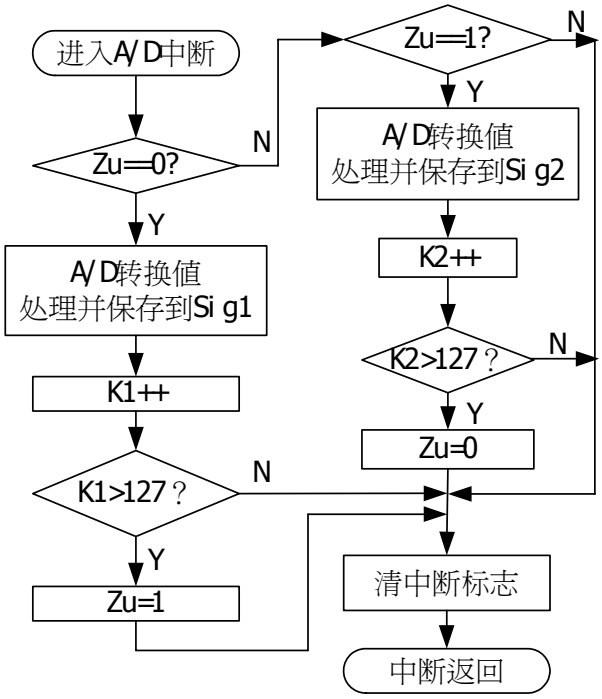


图10 A/D中断流程图

3.4 CAN 模块配置

本系统根据传输数据量及传输距离，配置通信速率为 500kbps；CAN 标识符使用标准数据帧；每个节点应当有地址和名称；每个 CAN 节点在初始化时，不管节点的地址是否可配置，都必须进行地址声明，当节点出现地址冲突时，必须根据名称进行地址仲裁。

在 CAN 控制模块中，通过屏蔽器和过滤器的配置，实现节点的地址配置。过滤器和屏蔽器用于决定报文合成缓冲器的一条报文是否应该被装入接收缓冲器中的一个。

本测试系统在应用中需要三个发送报文缓冲器，分别为 TXB0、TXB1 和 TXB2。其发送顺序为第一个正在发送报文时，第二个则准备等到第一个发送完就立即开始发送，而第三个被 CPU 重新载入。这样减轻了用软件维持总线同步的负担。同时可以为三个发送缓冲器配置发送优先级，确保比较重要，或者需要及时到达的报文优先发送。

4 总结

本系统开发了一种基于数字信号控制器dsPIC30F6014的发电机电参量测试装置。该测试装置能够实现以下电参量的测试，包括：三相电压、三相电流、功率、功率因数、三相电压和电流的各次谐波幅值、序分量(正、负、零序)和波形畸变率。完成了测试装置的硬件和软件设计，通过实验验证，证明其可以成功实现预期功能，主要成果有一下几点：

① 根据现有发电机试验中存在测试精度不高，浪费大量人力资源及无法进行某些重要参数，如序分量和波形畸变率的分析等问题，分析了针对发电机电参量的综合性测试装置研究的必要性和重要性。

② 对交流采样有关算法进行介绍及选择。介绍FFT基本思想，本测试装置采用基2－DIT FFT算法进行谐波分析。根据FFT分析结果可以计算序分量、波形畸变率及谐波功率。

③ 完成了测试装置的硬件设计。通过分析数字信号控制器dsPIC30F系列芯片，及其通用系列的dsPIC30F6014芯片适合本测试系统的特点，决定以dsPIC30F6014作为测试装置的微处理器。

④ 在dsPIC30系列DSP函数库的基础上，利用其变换函数及矢量函数(fractional数据类型)，在微芯公司网站提供的FFT函数的基础上进行修正及改进，并进行验证，验证结果表明改进后FFT运算结果精度明显提高。在编程中，在空间和时间上要充分利用dsPIC30F6014的优势，如程序空间可视化(PSV)、单周期MAC指令及DSP函数等。

⑤ 对测试装置误差来源进行分析，包括硬件部分和软件部分，并简单介绍降低误差的方法。对测试装置的各部分软件进行实验验证，包括谐波分析功能验证、顺序采样相位验证，以及序分量和波形畸变率计算功能的验证等。在实验室搭建实验平台，模拟现场，对电网电参量进行测试。通过大量的实验数据验证，证明本测试系统可以实现发电机基本电参量的测试、序分量及波形畸变率计算等功能。

⑥对于显示装置我们采用了带触摸控制的液晶显示器（CA320240），这种方案具有完善的汉字显示和图形显示功能，占用I/O口少，程序可移植性好，可以减少应用系统中的键盘模块，具有良好的人机交换功能。