Realizing an Intelligent FFT-Analyzer using TMS320C40 DSPs

Frequency Ghost
University of Cassino, University of Salerno

ABSTRACT

“This document was an entry in the TI DSP Challenge 2000, an annual contest organized by TI to encourage students from around the world to find innovative ways to use DSPs. For more information on the TI DSP Challenge 2000, see TI’s World Wide Web site at www.ti.com/sc/dsp_challenge.”

Frequency domain signal analysis is continuously and increasingly spreading due to the increasing number of applications that require more in-depth information on the frequency content of given signals.

The reduced complexity and high accuracy of Fast Fourier Transform (FFT) algorithms [1]-[3] and the high processing capability of modern DSPs allow high-speed high-performance FFT analyzers to be set up. A number of instruments capable of executing an FFT-based spectral analysis are available on the market, with different hardware and software solutions and, consequently, different performance and application fields. However, their firmware is not easy modifiable (except for digitizer software) and this lack in flexibility makes them not easily employable in dedicated applications, such as on-line diagnosis or automatic process control. Moreover, whichever the type of FFT-Analyzer used, it is necessary to define some measurement parameters (e.g. point number, sampling frequency, window). The accuracy, or even the correctness, of the obtained measurements, strongly depends on the values assumed by these parameters.

As a consequence, the user has to be an expert to optimally fix these parameters and also to evaluate the effects of eventual setting mistakes. On the other hand, the optimum values of these parameters depend on the harmonic content of the signal to be analyzed and consequently they cannot be a-priori defined without any knowledge about the signal.

Here, the realization of a real-time “intelligent” FFT analyzer, based on a two TMS320C40 DSPs by Texas Instruments (TI™) architecture [4]-[6], is presented. It is able to analyze signals with an unknown and variable spectral content, without any user intervention in the configuration of the analysis parameters.

The acquisition and communication resources of ‘C40 chips are exploited to perform a continuous data acquisition and a parallel distribution of the computational load. A user-friendly interface on a PC displays measurement results with a continuous screen update, and lets user choose among some display options and some utilities. In the following, the realized FFT-Analyzer is described in detail, outlining the instrument autoconfiguration capabilities, the distribution of tasks among the processors, the time performances and the obtained measurement capability.

Keywords: DSP, DFT, frequency analysis, auto-configuration, parallel processing.
Contents

The Hardware Architecture Based on TMS320C40 ........................................................................................................ 3
The Software .................................................................................................................................................................. 4
  The DSP Procedure ............................................................................................................................................. 4
    Rough Analysis Module ............................................................................................................................... 4
    Signal Processing Module ............................................................................................................................ 5
    Post-processing Module .............................................................................................................................. 5
  The PC Procedure ............................................................................................................................................... 5
Implementing the Procedure on The TMS320C40-Based Architecture ................................................................. 7
  The Data Memory Management ....................................................................................................................... 7
  The Signal Data Acquisition ............................................................................................................................. 8
  The Algorithm coding .................................................................................................................................... 9
  The Software Sharing Between the two C40s ................................................................................................. 10
Experimental results ........................................................................................................................................... 11

Figures

Figure 1. Hardware architecture .......................................................................................................................... 3
Figure 2. Block diagram of the configuration procedure ..................................................................................... 4
Figure 3. The output panel ................................................................................................................................... 6
Figure 4. Memory concurrent access for acquisition (a(ti,ti+1)) and elaboration (e(ti)) .............................................. 7
Figure 5. The buffer updating strategy ................................................................................................................ 8
Figure 6. The DMA2 of the master DSP settings .............................................................................................. 8
Figure 7. The CPU tasks .................................................................................................................................... 10

Tables

Table I. Maximum elaboration times of DSP procedure ..................................................................................... 7
Table II. Some experimental results .................................................................................................................... 11
The Hardware Architecture Based on TMS320C40

The hardware architecture used is reported in Figure 1. It is based on a host PC (a Pentium 133 MHz processor in the set up considered) in which a carrier board (DPC/C40 by Loughborough Sound Image [6]) is plugged. It hosts two concurrent DSPs (TMS320C40 32 bit floating-point, 40 MHz, by TI) and a data acquisition system (Analog Daughter Module, ADM, by Burr-Brown) with two Delta-Sigma ADCs (16 bit resolution, 200 kHz maximum sampling frequency, +3 V full-scale, 20 kΩ input impedance) and two DACs (maximum generating frequency 500 kHz and output impedance 600 Ω, +3V full-scale).

The 'C40 DSPs are in a master/slave configuration; only the master DSP can directly access the motherboard resources. The ADM communicates only with the master DSP by a serial non-standard interface (AMELIA). Data exchange between DSPs is carried out using parallel communication ports (six ports each). The DSPs have six DMA channels, that perform transfers to and from anywhere in the processor’s memory map. A DMA coprocessor can automatically initialize its registers via linked list stored in memory, without any intervention by the CPU. Furthermore, the data transfer can be synchronized via internal and external interrupts, allowing data exchange between each DSP and the external resources to be carried out as a task parallel to the C40 CPU activity [4].

The communication between the master DSP and the host PC takes place through a dual port RAM (DPRAM) of 4K x 32 bit.

Figure 1. Hardware architecture
The Software

The instrument requires software both for measurement procedure execution, running on the DSPs, and for measurement command sending, and user interface, running on the PC.

The DSP Procedure

Different algorithms that can be grouped in three main modules compose the implemented procedure (see Figure 2): auto-configuration, signal processing, and post-processing. The auto-configuration module deals with the choice of the techniques to be used in the signal analysis and with the determination of the optimum values of measurement parameters. The signal processing module executes the algorithms identified by the previous module and provides the spectrum samples. The post-processing module computes amplitude and phase spectrum and extracts the required information on the signal under analysis.

Rough Analysis Module

Since optimum choices can be taken only on a known signal, a rough spectrum estimate is the first activity of this module. A preliminary FFT is carried out on samples acquired at the maximum sampling frequency allowed by the hardware (f_s = 100 kHz, in the considered setup), considering a limited point number N_0 = 512, and using a window function that privileges detectability over resolvability (Blackman-Harris, -67 dB) [7]. The obtained amplitude spectrum is then analyzed to determine the signal minimum and maximum frequencies (f_min and f_max), the bandwidth (B_w), and the eventual dc value (V_{dc}) [8].

Figure 2. Block diagram of the configuration procedure

![Figure 2. Block diagram of the configuration procedure](image-url)
On the basis of the values assumed by $f_{\text{min}}$, $f_{\text{max}}$, and $B_w$, the possibility of improving the rough resolution by applying suitable techniques is evaluated. If $B_w$ is greater than one fourth of $f_s$, the only possibility of improving the resolution is to perform the final FFT on a higher point number (algorithm $\alpha$). On the contrary ($B_w < f_s/4$), there are two possibilities: decimation (algorithm $\beta$); signal translation at low frequencies followed by a decimation (algorithm $\gamma$).

**Algorithm $\alpha$.** The final FFT is executed on a higher point number $N_\alpha = 2048$.

**Algorithm $\beta$.** The decimation factor ($D_\beta$) is computed as the integer part of the ratio between the sampling frequency and twice the maximum signal frequency ($\text{int}\left[f_s/(2 \cdot f_{\text{max}})\right]$). The number ($N_\beta$) of processed points is 1024 or 512, in function of the resulting $D_\beta$ (if $D_\beta \leq 32$ $N_\beta = 1024$ else $N_\beta = 512$).

**Algorithm $\gamma$.** The signal is translated of $f_0 = f_{\text{min}}$ in order to obtain a new maximum frequency as low as possible, thus increasing the decimation factor value ($D_\gamma = \text{int}\left[f_s/(2 \cdot B_w)\right]$). This algorithm is applied only if the signal will result shifted on frequency values close to the upper frequency limit and if its resolution will be better than the $\beta$ one, namely if $D_\gamma > N_\beta/N_\gamma$. The number of final FFT is fixed to $N_\gamma = 512$.

**Signal Processing Module**

This module executes the previously defined algorithms. While execution of algorithms $\alpha$ and $\beta$ is not particularly complex (see Figure 2), the algorithm $\gamma$ includes: antialiasing filter synthesis, elimination of the dc component, frequency translation, filtering, decimation, windowing, final FFT execution. The dc component has to be subtracted from the sampled signals before translation to avoid its image at $f_s - f_0$. As far as the filter synthesis is concerned, the cut-off frequency ($f_{\text{cut}}$) is equal to the signal frequency bandwidth while the type and the order ($M = 30$) are fixed as a compromise between the obtainable transition bandwidth and the computation complexity.

**Post-processing Module**

At first it computes the amplitude spectrum. Then a suitable algorithm searches for the detectable tones. For each one of these, an interpolation technique locates the tone and estimates its frequency, amplitude and phase [9]-[11]. Finally, the master DSP sends via DPRAM to the PC the following: the tone characteristics (frequency and amplitude), the frequency resolution, the minimum analyzed frequency, and the square value of spectrum samples.

**The PC Procedure**

A hardware interrupt on the host PC is generated when the master DSP writes into a particular DPRAM address. In the corresponding service routine the elaboration results are read from DPRAM. The PC elaborates the obtained data giving both the spectrum visualization and the characteristics of signal tones. The indicators of frequency, amplitude and phase of detected tones on the output panel are updated at each elaboration step, while the plot of the amplitude (with a chosen scale) or phase spectra is updated only when significant variations occur.
As the PC programming language is concerned, a general-purpose development environment was chosen (LabView™) since it allows both a very friendly user-interface to be easily set-up and the DSPs to be visible. In particular, the main LabView™ program directly run some executable programs, that allow resetting of the DSPs and program loading. The routines for data exchange between the master DSP and PC via DPRAM, instead, are managed by calls to external routines. These, in the form of C functions, are collected in a Dynamic Link Library (DLL) suitably implemented.

In Figure 3 the user interface is shown. As you can see, there is the amplitude or phase spectrum, and tone information. A number of different utilities are also available to the user (Save Spectrum, Save Points, More Measurement). For example, by pushing the “More Measurement” button, a number of additional post-processing activities (THD, Floor, Zoom, Power Spectrum) can also be requested by the user and carried out by the PC.

Figure 3. The output panel
Implementing the Procedure on The TMS320C40-Based Architecture

In order to optimally exploit the power of ‘C40 chips and to achieve real-time performances, some solutions were adopted. They concern:

1. the data management;
2. the signal data acquisition;
3. the algorithms coding;
4. the software sharing between the two DSPs.

By means of all these solutions, detailed in the following, the maximum elaboration times reported in Table I are obtained.

Table I. Maximum elaboration times of DSP procedure

<table>
<thead>
<tr>
<th>Kind of Processing</th>
<th>Algorithm α</th>
<th>Algorithm β (Nβ=512)</th>
<th>Algorithm β (Nβ=1024)</th>
<th>Algorithm γ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elaboration time</td>
<td>7.09 ms</td>
<td>3.23 ms</td>
<td>4.48 ms</td>
<td>21.29 ms</td>
</tr>
</tbody>
</table>

The Data Memory Management

The two DSPs have identical data structures: signal samples are memorized in a 64 K point buffer composed of two 32 K points arrays, handled in a circular way. The access to the buffer is concurrently performed by DMA for acquisition and by CPU for elaboration as is evidenced in Figure 4 for three different time instant t₁, t₂, t₃. Buffer dimension, elaboration times, and maximum decimation factor are chosen to allow the concurrent access without the use of semaphores. Namely, thanks to these design choices any memory location is never used contemporaneously for writing (by the DMA) and elaboration (by the CPU).

Figure 4. Memory concurrent access for acquisition (a(t₁,t₁+1)) and elaboration (e(t₁))
The Signal Data Acquisition

As previously shown, the signal samples are all memorized on both DSPs, but the data acquisition buffer updating, in order to obtain the best time performance, has to be completely performed by DMAs allowing the DSP CPUs to do all the elaboration continuously.

Figure 5. The buffer updating strategy

To this aim the DMA 2 of the master DSP serves the interrupt associated with the sampling; each sample being transferred to the slave DSP via the communication port. The DMA 5 of the slave DSP reads the new sample and updates its buffer. At the same time, a suitably created slave-path updates the master acquisition buffer via DMA. Figure 5 describes this slave-path: DMA 2 of the master DSP puts each new sample in its port 1, too; DMA 4 of the slave DSP reads the sample and puts it in its port 2; finally, DMA 5 of the master updates the buffer. As an example, Figure 6 shows the code section for configuration of the DMA 2 and of the DMA 5 of the master DSP, pointing out the linked lists used to re-initialize the transfer with different settings.

Figure 6. The DMA2 of the master DSP settings

```c
DMA[2][SRC_ADD]=(UINT) IMR;   /*clear AMELIA interrupt*/
DMA[2][S_INDX]=0;
DMA[2][COUNT]=1;
DMA[2][DES_ADD]=(UINT) &fit; /* dummy destination */
DMA[2][D_INDX]=0;
DMA[2][LINK_PNT]=(UINT) tab2; /*address of the linked configuration list */
DMA[2][CONTROL]=0x00c00049; /* start, source synchronization,
same priority as CPU, autoinitialization */

tab2[CONTROL]=0x00c00009; /*start,same priority as CPU,autoinitialization*/
tab2[SRC_ADD]=(UINT) CHANA; /* source = ADC buffer */
tab2[S_INDX]=0;
tab2[COUNT]=2; /* 2 transfers (port 1, port 2) */
tab2[DES_ADD]=(UINT) C_PORT1_OUT; /* first destination = PORT 1*/
tab2[D_INDX]=16; /* second destination = PORT 2, since
index port2 = index port1 + 16. */
tab2 [LINK_PNT]=(UINT) tab20; /* address of the linked configuration list */
```
The Algorithm coding

All the algorithms were coded paying particular attention to the execution time.

In particular, as far as the algorithm $\gamma$ is concerned, translation and filtering of 32 K points with 30th order filter require about $2 \times 10^6$ multiplication. The number of computations to perform is drastically reduced considering that the subsequent decimation will select a time-window with $N_\gamma = 512$. The sample sequence that will be windowed and Fourier transformed is obtained in an unique step (which thus include dc component elimination, frequency translation, filtering, and decimation) as:

$$x_{D_n}(n) = x_t(nD_\gamma) = \sum_{k=0}^{M} b(k) \cdot [x_t(nD_\gamma - k) - V_{dc}] = \sum_{k=0}^{M} b(k) \cdot [x(nD_\gamma - k) - V_{dc}] \cdot \cos\left[2\pi(nD_\gamma - k) \frac{f_{\text{min}}}{f_s}\right] + \sum_{k=0}^{M} b(k) \cdot [x(nD_\gamma - k) - V_{dc}] \cdot \sin\left[2\pi(nD_\gamma - k) \frac{f_{\text{min}}}{f_s}\right] \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ (1)$$

By this way, only 46080 $(3N_\gamma^*M)$ multiplication are required as maximum, instead of 2031616 $(2(M+1)*L)$.

Furthermore, trigonometric functions were tabled thus obtaining a noticeable execution time reduction, especially for the algorithm $\gamma$, since each call to a trigonometric function requires 5.9 $\mu s$ in spite of 0.3 $\mu s$ taken by the table-based approach.
The Software Sharing Between the two C40s

At the instrument start up, it is necessary to wait for the acquisition of \( L \) samples (\( L = 32 \, \text{K} \)). Then the CPUs start with the continuous execution of the measurement procedure (see Figure 7).

At first, the master DSP gives to the slave one via Port0 the index of the current position on the circular buffer so that both DSPs have available the same data set.

Then, they start with the rough FFT: the master windows the even ones of the last 512 acquired samples and executes the FFT, the slave does the same but on the samples with odd index. At the end, they transmit each other the FFT results and combine them (the master calculates the first \( N_0/2 \) points and the slave the others) to obtain the correct FFT samples. Now, the spectrum is available to both DSPs. The master finds its minimum frequency and analyzes the first two bins in order to detect an eventual dc component, while the slave computes the maximum frequency and sends it to the master. If the \( \alpha \) algorithm has to be executed, the FFT is directly applied, while in the case of the \( \beta \) algorithm it has to be run on the decimated sequence. This elaboration is shared by the two DSPs as for the rough FFT.

The actions to be performed in the case of the \( \gamma \) algorithm are more complex. First of all, the 31 coefficients of a low-pass, \( 30^{\text{th}} \) order filter have to be set up. The master and slave DSPs calculate, respectively, the first 16 and the last 15 coefficients and exchange them each other. The frequency shifting yields a complex sequence, so a 512-point complex FFT is needed. This task can be distributed in two 512-point real FFT, applied, respectively, on the real (imaginary) part of the sequence by the master (slave) DSP. Then, the slave DSP sends its result to the master, and viceversa, and both DSPs finally put all together, evaluating the amplitude spectrum.

At this point, the elaboration module is completed and both DSP have available the samples of the amplitude spectrum of the signal, whose number depends on the performed algorithm.

Figure 7. The CPU tasks
The last step of DSP procedure consists of the interpolation of spectral samples, that yields the frequencies and the amplitudes of the tones found in the spectrum. This task, too, is carried out in a concurrent way: each DSP processes half spectrum.

### Experimental results

A large number of tests on real signals are carried out in order to prove the correctness and the advantages of the proposed FFT-Analyzer. The correctness of the choices carried out by the configuration algorithm was verified by an extensive experimental analysis. Table II reports for some signals: the tone composition, the main signal processing characteristics (algorithm used; \(D_\alpha\), decimation factor; \(T\), analyzed time window; \(\Delta f\), frequency resolution; \(T_{elab}\), elaboration time), the memory handling and the elaboration results.

#### Table. II. Some experimental results

<table>
<thead>
<tr>
<th>Tone composition</th>
<th>Signal processing characteristics</th>
<th>Memory handling</th>
<th>Measurement results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>(f \pm U_f) [Hz]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(A \pm U_A) [mV]</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>(\text{Algoritmo} = \alpha)</td>
<td>(20000.05 \pm 0.49)</td>
<td>(75.23 \pm 0.57)</td>
</tr>
<tr>
<td></td>
<td>(N_\alpha = 2048)</td>
<td>(24999.76 \pm 0.49)</td>
<td>(382.6 \pm 2.1)</td>
</tr>
<tr>
<td></td>
<td>(D_\alpha = 1)</td>
<td>(29999.96 \pm 0.49)</td>
<td>(33.65 \pm 0.37)</td>
</tr>
<tr>
<td></td>
<td>(T = 20.48) ms</td>
<td>(35000.05 \pm 0.49)</td>
<td>(1000.4 \pm 5.2)</td>
</tr>
<tr>
<td></td>
<td>(\Delta f = 49) Hz</td>
<td>(39999.60 \pm 0.49)</td>
<td>(10.13 \pm 0.25)</td>
</tr>
<tr>
<td></td>
<td>(T_{elab} = 8.4) ms</td>
<td>(44999.81 \pm 0.49)</td>
<td>(33.58 \pm 0.37)</td>
</tr>
</tbody>
</table>

Other tests are carried out to point out the auto-configuration capability in case of signals with suddenly changing spectrum. In any test, the FFT-analyzer proves to be able to optimally configure itself. As an example, dealing with a signal with period \(T = 240\) s, defined as:

\[
x(t) = \begin{cases} 
\sin(2\pi 10000 t) & t = [0, 120] \text{ s} \\
\text{triangle}(2\pi 10000 t) & t = [120, 240] \text{ s}
\end{cases}
\]

the instrument operates as detailed in the following. At first, it chooses and runs the algorithm \(\gamma\) achieving a no-interpolated resolution of 4.9 Hz (9408.2 Hz, 10662.6 Hz minimum and maximum investigated frequencies), and measures by interpolation a tone at 10000.156 Hz.
The spectrum and all the related information are obtained each 22 ms. As soon as the waveform becomes triangular, the spectrum stabilizes again in less than 1 s.

In this second phase, the algorithm $\alpha$ is run each 5 ms obtaining a frequency resolution of 49 Hz. When the signal becomes sinusoidal, the instrument behavior is analogous, but with a longer delay (1.2 s) due to the longer duration of the time window required by the analysis of the sinusoidal waveform.

References