Embedded FFT with dsPIC30F4013

Rachmad Setiawan
Teknik Biomedik, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia
e-mail: rachmad@ee.its.ac.id

Abstract- FFT-based digital spectral analyzer has become more widely used as a result of the development of Digital Signal Processing (DSP) techniques. Modern Analog-to-Digital Converters (ADC) and processors have made it possible to make fast measurements with a limited number of hardware.

In this paper, a design of a simple low-cost FFT-based digital spectrum analyzer was presented. The author discusses the design of each components of the system in qualitatively and quantitatively. The report presents the whole system design in detail which contains filter design, microcontroller design and UART transmission design. Some satisfying measurement result of the system were presented in the paper. The system can provide fast measurement with good accuracy but the measured result has a limited range and resolution of the display is not very high. At last, the advantages and disadvantages of the system was discussed which is considered as guidelines for further work.

Keywords— FFT, Digital Signal Processing, spectrum analyzer.

I. INTRODUCTION

A spectrum analyzer is basically supposed to measure a power of the signal versus its frequency. This job was done by using analog Swept Spectrum Analyzer (SSA) in the old times [1]. But nowadays, a modern FFT-based digital spectral analyzer and do the same with lesser requirement and provide better results [2]. This has made the spectral analyzing more convenient without hurting its accuracy [3] [4]. FFT-based spectrum analysis (FFTSA) has become a widely used method for many implementations [5].

As Figure 1 shows, the analyzed signal will be presented in frequency domain instead of time domain. Letting the horizontal axis illustrates the frequency and the vertical axis illustrates the amplitude of the signal [6].

Figure 2 presents the basic architecture of both SSA and Digital Spectral Analyzer. It explains theoretically how a basic spectral analyzer works [1].

In the analog module, the analog input signal was first sent into an analog down-converter. At the same time, the local oscillator will provide a sinusoidal signal as reference signal to the down-converter. After multiplying these two signals together, the down-converter will produce a signal with suitable frequency for subsequent processing [1].

This output signal is called Intermediate Frequency (IF) signal. The frequency of the reference signal generated by the LO is controlled by the frequency range of the FFT through a controller in a Digital Spectral Analyzer.

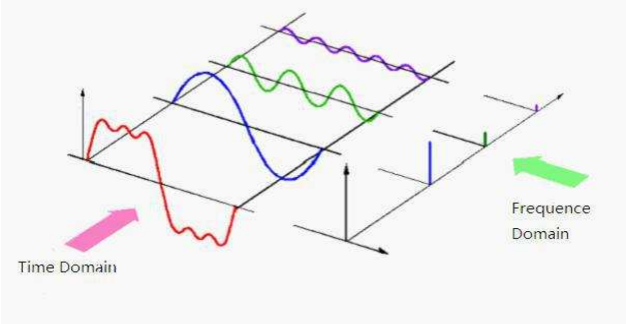

Figure 1. Imagination of spectrum for signal

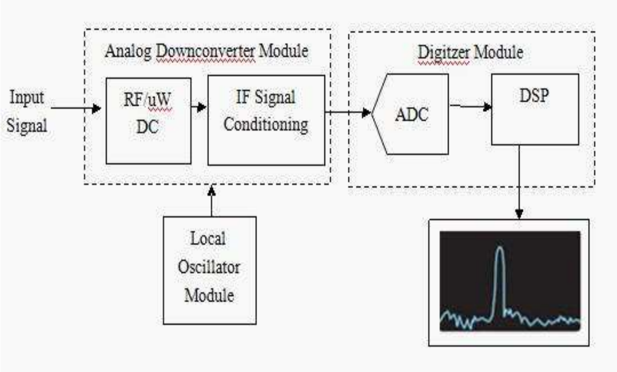

Figure 2. Simplified Spectrum Analyzer Block Diagram

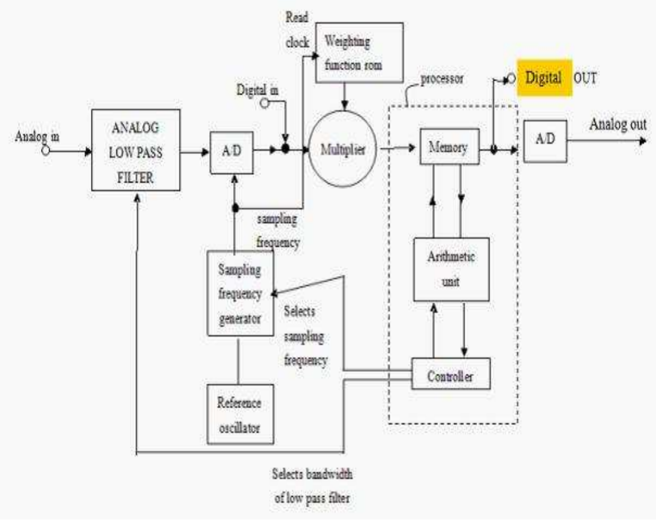

Figure 3. Detailed architecture of a Digital Spectral Analyzer

However, it is swept linearly over the frequency band or span to be measured [1].

In the digital module, the IF signals is sent into the ADC. The ADC will convert the analog signal into digital form. And the collected data is sent into DSP. The DSP will compute the FFT and output data to digital display. The processor should also control the LO with some algorithms. Most simple processors have the capability to do the job.

Figure 3 presents the detailed architecture of a simple Digital Spectral Analyzer which is studied in this paper [7]. This figure of Digital Spectral Analyzer shows more information about the digital module part than Figure 1 because it shows the FFT process.

In the analog module, the analog input signal is first applied to a sharp roll-off analog low-pass filter. The bandwidth of the low-pass filter is controlled by the controller in the processor. Then the output of the filter is applied to the digital module.

In the digital module, the analog signal is first applied to the ADC. The working frequency of the ADC is also controlled by the processor which can change the sampling frequency generated from the reference oscillator. The sampling rate of the ADC must be larger than twice the bandwidth of the low-pass filter to make sure no aliasing occurs [8]. At the same time, the reference signal with sampling frequency chosen is read by a Weighting Function ROM and converted to digital data. Then the multiplier will collect both data from the ADC and Weighting Function ROM to multiply them together to compute the FFT. Then data is transferred into the processor. The memory of the processor will add the numbers stored in the memory with coefficients (sin and cos) read from ROM and restore the result [7]. Then it will add and store again under the control of the arithmetic unit until the process is done. The arithmetic unit also decides how the controller should control the bandwidth of the analog low pass filter and the sampling frequency of the reference signal. At last the processor can output the results in memory to the display device.

The aim of the research is to design a simple Digital Spectral Analyzer with a limited number of hardware from HIG. The processor is a dsPIC30F4013 microcontroller. It will compute the analog to digital conversion and FFT. Then send the result to a PC which is considered to a display device. By building a GUI with Delphi, the PC will display the result graphically. Software used is MPLAB IDE with micro C30 compiler and Delphi 7.0

The system will analyze signal from 0 to 41 kHz. It should have some capabilities as the equipment used in laboratory works. The system could be powered by batteries and easy to carry, making it a convenient tool for simple signal analysis.

II. THEORITICAL BACKGROUND

2.1 FFT

The FFT refers to a "Fast Fourier Transform" which is a very efficient "Discrete Fourier Transform"(DFT). The reason of using FFT instead of DFT; to save time. For example, a straight DFT with N samples requires N^2 complex multiplications, while a FFT with the same samples requires only Nlog$_2$N complex multiplications [9].

The reason why FFT is fast can be explained by discussing its three characteristics: “Danielson-Lanczos Lemma” (D-L Lemma), “twiddle factors" and” Butterfly Diagram”.

2.2 D-L Lemma

The DFT process follows *equation 1 [7]*:

$$F(k) = \sum_{n=0}^{N-1} f(n) e^{\frac{-j2\pi kn}{N}}$$

$$k = 0,1,2, \ldots\ldots, N-1 \qquad (1)$$

Where F(k) is the power of the signal at frequency elements k, and f(n) are N samples of the input function.

In a D-L Lemma, the first step is to break down the expression into two parts: even terms and odd terms. It becomes *equation 2*:

$$F(k) = \sum_{n=0}^{N-1} f(n) e^{\frac{-j2\pi kn}{N}} = E + 0$$

$$F(k) = \sum_{k=0}^{\frac{N}{2}-1} f(2n)\, e^{\frac{-j2\pi kn}{N/2}} + \sum_{k=0}^{\frac{N}{2}-1} f(2n+1)\, e^{\frac{-j2\pi kn}{N/2}}\, e^{\frac{-j2\pi n}{N}} \qquad (2)$$

Where $e^{\frac{-j2\pi n}{N}} = W_N^n$ known as twiddle factor

Then F (k) is broken down in this order until the expression runs out of samples, which means

$\sum_{k=0}^{0} = e^0$[9]. This is the reason why the number of samples (N) in the FFT needs to be the power of 2.

Figure 4 presents the theoretical process of D-L Lemma [9]. After breaking down the expression, all units remains are $f(n)$ and $W_N{}^n$. For example, when N=4 the final expression is $F(n) = f(0) + W_2^n f(2) + W_4^n f(1) + W_4^n W_2^n f(3)$.

It is observable that the order of the input samples is not natural in the new expression. When N=4, the nature ordering of input samples should be "00 01 10 11”. While in the FFT calculation the ordering becomes " 00 10 01 11" which is the reverse of nature ordering. This is known as "bit reverse ordering" which is used in the "Butterfly Diagram" [9].

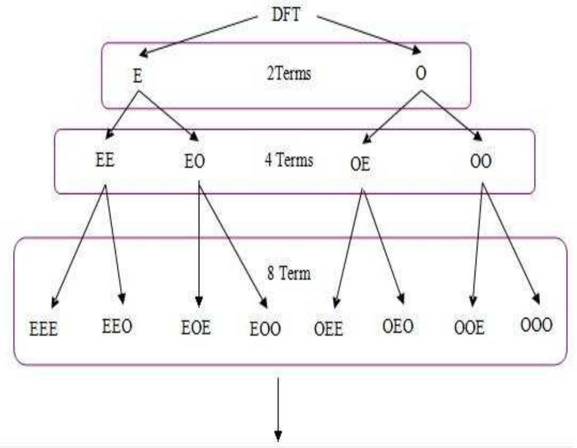

Figure 4. Diagram of D-L Lemma

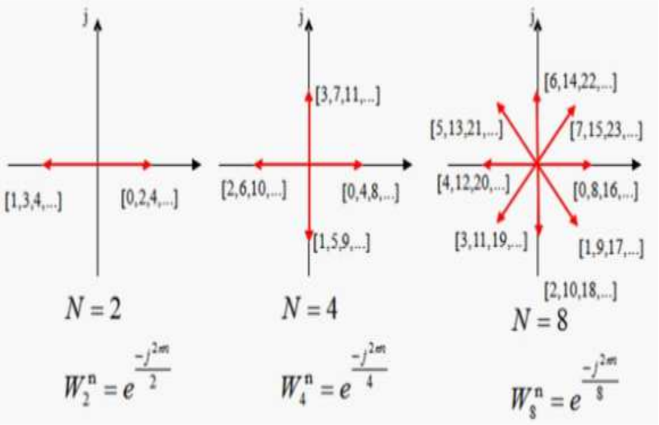

Figure 5. Diagram of Twiddle Factors

2.3 Twiddle Factors

The twiddle factor refers to a "rotating vector" which rotates in increments according to the number of samples. As mentioned in previous section, for calculating an FFT with N samples, twiddle factors $W_N{}^n$ are needed.
Figure 5 presents how to generate twiddle factors for a certain number of samples [9].
Figure 5 indicates that the large number of samples in FFT, the more twiddle factors needed. And the twiddle factor has redundancy in values as the vector rotates around [9]. This means $W_N^n = W_N^{n+N}$. Also, the values of twiddle factors with 180 degrees out of phase are the negative of each other which means $W_N^n = -W_N^{n+\frac{N}{2}}$. The "Butterfly diagram" takes advantage of these characteristic of the twiddle factor, making the FFT realizable.

2.4 The Butterfly Diagram

The Butterfly Diagram is an efficient FFT algorithm based on D-L Lemma and the twiddle factors. For an FFT with N samples, the Butterfly Diagram will contain $\log_2 N$ stages. The first stage of the diagram is presented in *figure 6* [9].

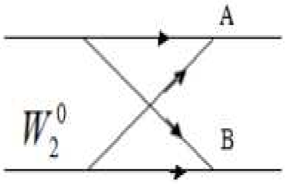

Figure 6. First stage of Butterfly Diagram

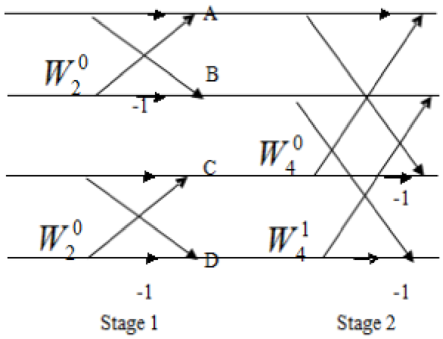

Figure 7. Butterfly Diagram

The -1 in the Figure 7 refers to $W_2^1 = -W_2^0$ as mentioned in previous section. Then the next stage will connect the upper and lower legs of the butterflies presented in *figure 7* [9].
The diagram will continue computing in this way until the $\log_2 N$ stage is done. Note that if the input samples are in bit reversed ordering, the output will be natural ordering.

2.5 Low Pass Filter

In signal processing, filter is a device that can remove these unwanted components in a signal. It can be either analog or

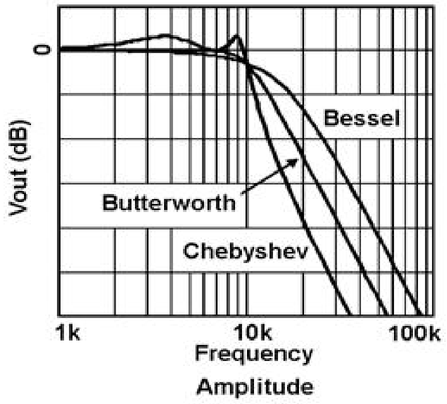

Figure 8. Different type of filter response

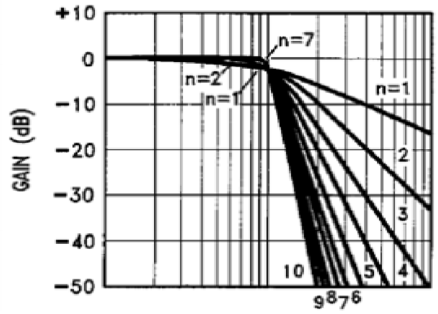

Figure 9. Frequency dependence of n: th order Butterworth filter.

digital, discrete-time or continuous-time, linear or non-linear, time-invariant or time-variant, passive or active. In this work, an analog continuous-time active low-pass filter is chosen. As it require lesser components and easy to be built. The purpose of using such a filter in this work is to remove signals that have frequency higher than half of the sampling frequency. So the analog low-pass filter is an anti-aliasing component of the system.
The Figure 9 are representative of a low pass filter has different type of *n th order Butterworth filter.*
The Butterworth low-pass filter provides maximum pass-band flatness. Hence, a Butterworth low-pass filter is often used as anti-aliasing filter in data converter applications where precise signal levels are across the entire pass-band [10].
Figure 9 illustrates that the higher order the filter has, the faster drop band drop off is [11]. And the actual value is -6dB per octave as it is first order.
The Sallen-key topology can provide high accuracy, unity gain, and low Q(Q<3).
The Multiple Feedback topology is commonly used in filters that have high Q and require a high gain [10]

III. DESIGN IMPLEMENTATION

3.1 System set up

First of all, a system clock needs to be configured. Adjusting the clock of the microcontroller is done by setting up the oscillator configuration register. In this work can be easily done by using a macro: _FOSC. Then Oscillator Control Register (OSCCON) and FRC Oscillator Tuning Register (OSCTUN) were not written to as the FRC oscillator for this work do not need specification. So the system clock is configured as below:

_FOSC (CSW_FSCM_OFF & FRC)

Set up Internal Fast RC Oscillator as source oscillator. No PLL mode enabled and disable clock switch. The oscillator is working at 7.37MHz so the frequency of the instruction cycle is 1.84MHz.

Then some other registers were written to provide necessary configuration.

_FWDT (WDT_OFF)

Watch-Dog Timer is disabled.

_FBORPOR (MCLR_EN & PWRT_OFF)

Enable MCLR reset pin is enabled and the power-up timers is disabled.

_FGS (CODE_PROT_OFF)

Code protection is disabled.

3.2 ADC configuration

The analog to digital converter used in this work in the internal ADC of dsPIC30F4013. It can run up to 200k samples per second without using external reference voltage. The output format is 12-bit fractional or integer with high accuracy (0.02%). [12]

The block diagram of the ADC is shown in Figure 12 [12].

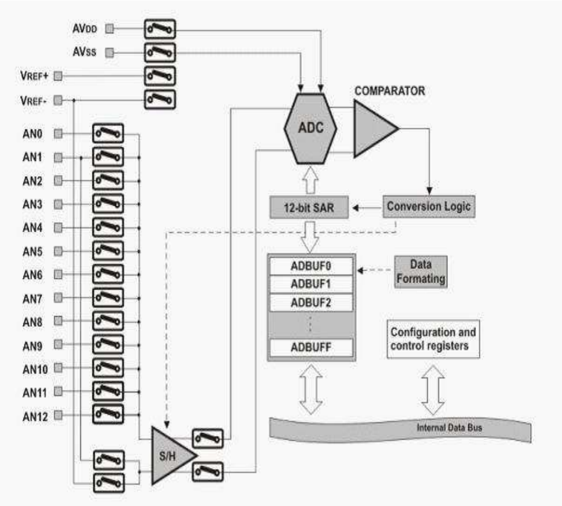

Figure 12. Functional block diagram of the 12-bit A/D converter of dsPIC30F4013

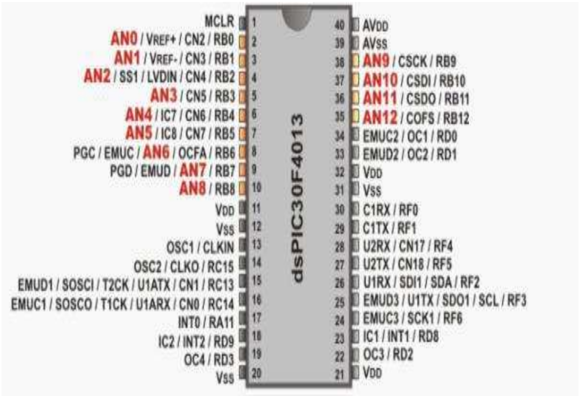

Figure 13. Pin out of dsPIC30F4013

AN0 to AN12 are input channels. Only one of them is used in this work. Vref+ and Vref- refers to external reference voltage which is not used in this work. ADBUF0 to ADBUFF refers are output buffers of the ADC. Only ADBUF0 is used in this work for easier data collecting.

The pin out of dsPIC30F4013 is presented in Figure 13 [12]. To setup a ADC that satisfy the system, the following status registers in the dsPIC30F4013 need to be written: ADCON1, ADCON2, ADCON3, ADPCFG, TRISB, ADCHS and ADCSSL. The ADC was set to work with following characteristics.

A. Generate input analog signal from port B 10, channel AN10.
B. The ADC runs with MUX A multiplexer. Positive and negative input of MUX A were set to be AN10 and ground.
C. The sampling process takes 1 T_{AD} and the conversion takes 14 T_{AD}.
D. The sampling frequency is of ADC is 89 kHz as the ADCS bits on ADCON3 were set to be two. From $AD = \frac{T_{CY}(2+1)}{2}$, knowing that TCY=1/FCY=0.54us, TAD is 0.81 us. So F_{AD}=1/15*T_{AD}=82 kHz.
E. No scan on the ADC input
F. Reference voltage of ADC is AVdd (5V) and AVss(ground).
G. An interrupt will occur upon completion of each conversion. This will make all the output data of ADC stored in ADCBUF0.
H. The output data format is fractional. And the ADC will auto start sampling and conversion after the ADC is on.

3.3 DSP engine configuration

The core of DSP used in this work is the multiplier. It computes the FFT arithmetic. The first step to set up the DSP is to declare the array of twiddle factors in program memory. Then declare the array of ADC output array in the Y data bus of the multiplier.

As mentioned in theory, an FFT process with N samples needs only N/2 twiddle factors. As each element in twiddle factors has one "real" part and one "image" part. So totally N fractional numbers written in 2N bytes were contained in the Twiddle Factor array. This array is considered as constants and stored in program memory. The PSV mode will generate these constant to the X data bus of multiplier when computing the FFT which will reduce the processing time. The PSVPAG register which is used to translate 24-bit data in program memory to 16-bit data in multiplier can be automatically written by a macro. The final code of this step is as follows.

constfractcomplex twiddleFactors[]__attribute__((space(auto_psv), aligned(FFT_N*2)));[13]

The array on input sample is the array of ADC output data. For an FFT with N samples, the input array contain N elements which is 2N fractional numbers written in 4N bytes. To collect data from ADC output buffer , the "real" part of the array needs to be declared equal to ADCBUF0. Programmed as ADCoutput[i].real=ADCBUF0, where i is an integer which increase from zero to N-1. Then declare the array to be transferred to y data bus of the multiplier. The code for this step is:

fractcomplex ADCoutput[FFT_N]
__attribute__((space(ymemory),far,aligned(FFT_N*2*2)))[13]

As the output data range is [-1,1] and the multiplier requires data range to be [-0.5,0.5][11]. The "real" parts of the array need to be scale by 0.5 for further processing. This is done by shift the ADC output data one bit to the right. The "imag" part of the array also needs to be declared to be 0 as the input signal does not contain an imaginary part.

ADCoutput[i].real = ADCoutput[i].real>>1;
ADCoutput[i].imag = 0x0000;

where i is an integer which increase from zero to N-1.

The FFT process is done with the macro

FFTComplexIP (LOG2_N, &sourcevector, (fractcomplex *) __builtin_psvoffset(&twiddleFactors[0]), (int)__builtin_psvpage(&twiddleFactors[0]));[13]

With this macro, the multiplier will generate input samples from the "sourcevector" and twiddle factors from program memory. LOG2_N refers to the number of stages the Butterfly will process. The output result with frequency component is in bit reversed ordering and stored back into the "sourcevector". As the real frequency of the signal is calculated in further processing, the "sourcevector" needs to be translated into natural ordering. This is done by the macro **BitReverseComplex(LOG2_N, &sourcevector)**[12]. As the "sourcevector" still contains a "real" part and an "imag" part , and the final output is the power of the signal in "real" only. The power is calculated as$P = \sqrt{R^2 + I^2}$. This is done by the macro

SquareMagnitudeCplx(FFT_N, (fractcomplex *) &sourcevector[0], (fractional*) &sourcevector[0].real);[13]

The value of the power is stored in "real" part of the "sourcevector" after this step. And the FFT process is considered to be over.

3.4 UART configuration

The UART part of the system is designed to connect the dsPIC30F4013 to the PC. Output data of DSP model is transferred through the transmitter.

The UART model within dsPIC30F4013 is designed with following characteristics [12]:

A . Use U1TX as transmit channel.
B. 8-bit data communication with no parity.
C. A transmission interrupt is generated when a character is transferred to the transmit shift register and the transmit buffer becomes empty.
D. UART model continue operation in IDLE mode.
E. UART loop back mode is disabled.
F. UART baud rate is 2300 bytes per second.

IV. Experimental Results

In this section, measured FFT result of different kinds of signals with the theoretical results . Figure 17-20 present the test result of different signals

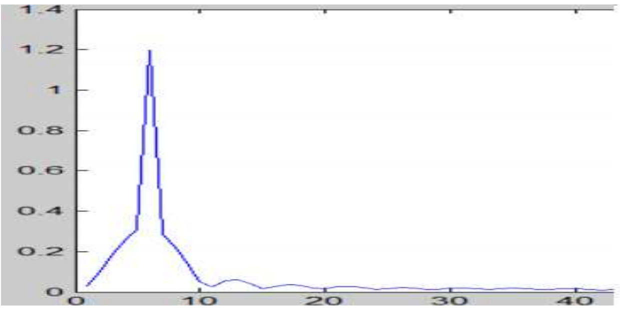

Figure 17. Measurement result of a 5kHz sine wave with 1.2V amplitude.

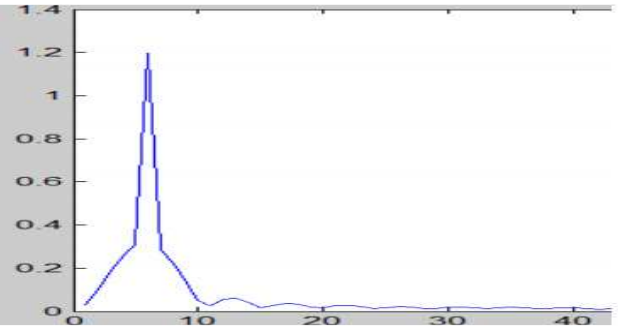

Figure 18. Simulation result of a 5kHz sine wave with 1.2V amplitude on MATLAB.

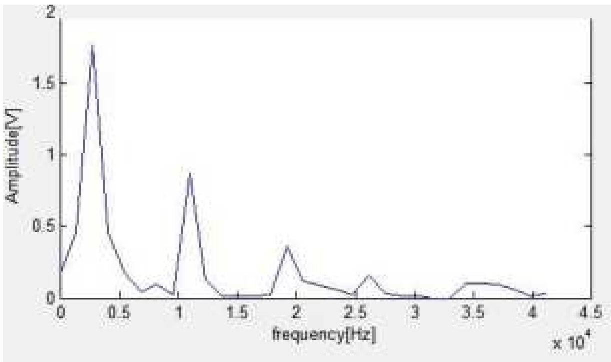

Figure 19. Measurement result of a 4kHz square wave with 1.8V amplitude.

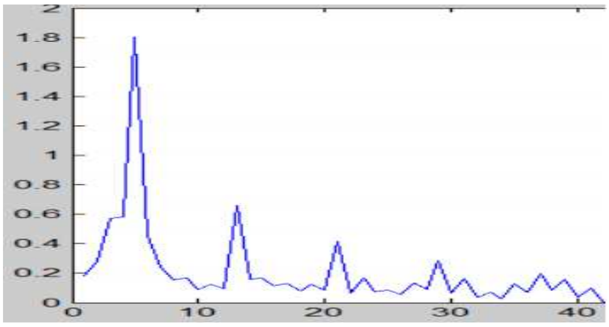

Figure 20.Simulation result of a 4kHz square wave with 1.8V amplitude on MATLAB

Figure 17 is the measured result of a 5kHz sine wave while Figure 18 is the simulation result of the same signal. The simulation was made by doing a 64 samples FFT to a 5kHz sine wave with 82kHz sampling frequency on MATLAB. So the simulation has similar circumstances as the real measurement. Comparing both figures, it is indictable that, there is a noise at the first element of the output. But it does not affect the measurement a lot as it is display on 0Hz. This error is produced by the A/D converter. When the A/D converter is just turned on, the first output data has low accuracy. Also the measured amplitude of the signal which is 1.25V is a little bit higher than the theoretical result which is 1.2V.

The simulation in Figure 20 was done similar to simulations before, changing the simulated signal to a 4kHz square wave. In Figure 19, the noise at 0Hz is smaller in this measurement as the input signal is a square wave and the noise of the first output of A/D converter did not affect the measurement a lot. The aliasing signals in the theoretical result were not contained in the measurement result because of the analog low-pass filter.

V. CONCLUSION

The final result of the work almost follows the research aim. So generally the system is satisfying. Considering the errors in the measurement and limitation of the system as mentioned in the discussion part, some further work can be done to improve the system.

A. Add an MAX232 chip to the system as communication tool between UART and the PC. The MAX232 mode must be made stable for good data transmission. This can avoid using two software for data analyze and simplify the system.

B. Add a window function before the FFT process to improve the result. This will also add more multiplication to the system and greatly increase the measurement time.

C. Configure the A/D converter to sample at a higher sampling rate. This can increase the range of the measurement .

D. Use a graphic LCD as display device. This makes the system work without a PC. But also increase the cost of the system greatly.

E. Increase the number of samples of the FFT to improve the result which will also increase the measurement time.

REFERENCES

[1] M. T. Hunter , A. G. Kourtellis, C. D. Ziomek and W. B. Mikhael, "Fundamentals of Modern Spectral Analysis," in AUTOTESTCON, 2010 IEEE, 13-16 2010, pp. 1-5.

[2] M. T. Hunter, W. B. Mikheal and A. G. Kourtellis, "Wideband digital down converters for Synthetic instrumentation," in IEEE Transaction on Instrumentation and Measurement, vol. 58, no.2, pp. 263-269, Feb. 2009.

[3] M. Hunter, "Efficient fft-based spectral analysis using polynomial-based filters for next generation test systems," in AUTOTESTCON, 2007 IEEE, 17-20 2007, pp. 677-686.

[4] W. Lowdermilk and F. Harris, "Cost effective, versatile, high performance, spectral analysis in a synthetic instrument," in AUTOTESTCON, 2008 IEEE, 08-11 2008, pp. 148-153.

[5] W. Lowdermilk and F. Harris, "Wide spectral span spectrum analysis with an anlog step and dwell translation pre-processor to a high dynamic range fft-based spectrum analyzer," in AUTOTESTCON, 2009 IEEE, 14-17 2009, pp. 365-368.

[6] RF, Electronics and Wireless Test and Measurement Blog(2010).'RF Spectrum Analyzer Tutorial and Basics'[online]. Available at: http://testrf.com/2010/spectrum-analyzertutorial/ last accessed: 23th March 2011.

[7] J. H. Flink , J. Bertrand and V. Cottage, "Spectrum analyzer using digital filters" in United States Patent, Jun. 1978, pp. 54-75

[8] A. M. Chwastyk, "A fast digital spectral analyzer," in IEEE Transaction on Instrumentation and Measurement, vo. 20, no. 4, pp. 198-202, Nov. 1971.

[9] "A DFT and FFT tutorial " (2011), available at : http://www.alwayslearn.com/DFT%20and%20FFT%20Tutorial/DFTand FFT_FFT_Overview .html

[10] R. Manchini. "Op amps for everyone" available at :http://focus.ti.com/lit/an/slod006b/slod006b.pdf. Last acceded on: 12th June 2011

[11] “Butterworth Filters" (2011), available at :http://wwwk.ext.ti.com/SRVS/Data/ti/KnowledgeBases/analog/document/faqs/bu.htm [12] Microchip Technology Inc. (2007), "dsPIC30F3014, dsPIC30F4013 Data Sheet". [13] Microchip Technology Inc. (2004), "dsPIC language tools libraries".