Implementing FFT Algorithms on PSoC® System

AN42877

Authors: Nicola Sgambelluri, Gaetano Valenza
Associated Project: No
Associated Part Family: CY8C29x66
Software Version: PSoC Designer 4.2+SP3
Associated Application Notes: None

Application Note Abstract

This application note describes how to implement a Fast Fourier Transform (FFT) in a PSoC®. The application shows a simple embedded system on PSoC that evaluates the spectral analysis of an input analog signal in real time without any external components.

The FFT, written in C, computes the spectrum from a time domain analog signal. The PSoC processes the samples through an internal ADC converter before performing a 64-point FFT on the samples to obtain the spectrum in real time. The system calculates the magnitude and frequency of the spectrum and displays the data on a PC using the RS232 protocol. These features display the capability and flexibility of the PSoC system.

You can select and change the internal ADC based on a trade-off between the bandwidth and to resolve the input signal. The PSoCEval1 board is used to test this application.

Introduction

The Fast Fourier Transform (FFT) is an efficient algorithm to compute the Discrete Fourier Transform (DFT) and its inverse that reduces the number of calculations to be done.

The DFT is a numerical approximation of an analytically-defined Fourier Transform in a digital domain.

The DFT of a sequence x[n] of N samples can be denoted as X(k). The forward transform is defined as:

$$X[k]=\frac{1}{N}\sum_{n=0}^{N-1}x[n]e^{-\frac{j2\pi k n}{N}}$$

for k=0…N-1.

The inverse transform is defined as:

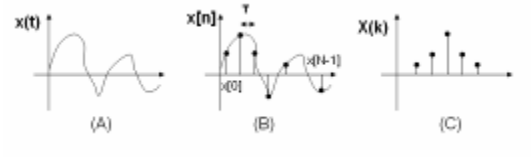
$$x[n]=\sum_{k=0}^{N-1}X[k]e^{\frac{j2\pi k n}{N}}$$

for n=0…N-1.

To evaluate the spectrum of a continuous signal $x(t)$ a sampling is performed every T seconds. The signal evaluated with t=nT is represented by a finite length sequence x[n]. The length of the temporal window and the sampling-interval T, introduces numerical errors and approximations. The complete process is shown in Figure 1.

The sampling rate must be greater than twice the highest frequency of the time record, ($f_c \geq 2f_{max}$) according to the Nyquist sampling criterion. If the sampling rate in the time domain is lower than the Nyquist rate, aliasing occurs.

Figure 1. DFT Evaluation Process



There are many FFT algorithms and different optimizations; the FFT algorithm used here is the standard Cooley-Tukey's algorithm. This algorithm decomposes the DFT into two smaller DFTs.

Typically for the FFT algorithm, N is an integer multiple of 2.

The FFT algorithm is implemented in the hardware to apply the DFT in real time to signals.

PSoC Implementation

When implementing the FFT algorithm in a simple µC system, you must avoid the limitations caused by floating point operations, memory, and sampling rate.

Floating Point Operations

For standard microcontrollers, it is best to implement without floating point operations. But with the PSoC we can use floating point capability because the fixed point computation and the time execution are satisfactory.

Memory

The algorithm requires memory space to store 3N variables (double), 2N to store the real and imaginary parts of the discrete Fourier transform, and N to evaluate the modules.

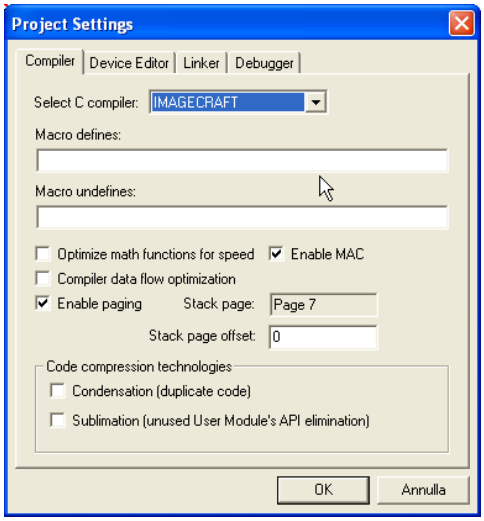
The 29 series has a 2 kB RAM. For optimizing memory usage, the paging memory maximizes the number of samples (N) and overcomes the memory limitation.

Using this method, you can set the number of points up to 64 (N_points=64).

Reduce the number of samples and use the same code for the 27 series.

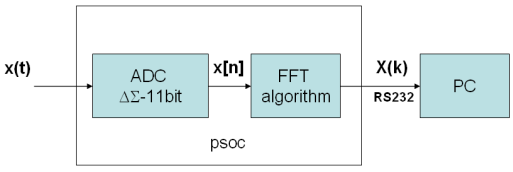
To enable paging, refer to the **Project Settings** dialog box, click on the **Compiler** tab and **Enable paging** in PSoC designer, as shown in Figure 2.

Figure 2. Page Setting



If you want to test an optimization of the code you can use the “Optimize math functions for speed” or “Compiler data flow optimization.” The block diagram of the system is shown in Figure 3.

Figure 3. Block Diagram of the System



To compute the DFT of a continuous signal, there must be a preliminary discretization.

The discretization that introduces periodicity in time and frequency domains is provided with the ADC block.

Sampling Rate

To overcome the sample time limitation, select a fast ADC converter Delta Sigma-11 bit to process the signal.

This resource allows a sampling rate up to 7.8K samples per second with an 11-bit resolution.

Sampling Parameters

The sampling parameters strictly depend on the assumed ADC.

To set the maximum sampling rate available (see ADC data sheet for details) the data clock is:

$$V_{C1} = \frac{SysClk}{3} = 8\,\text{Mhz}\ .$$

The sampling period is:

$$T = \frac{1}{N_{samples}} \approx 128\ \mu s$$

The time record is:

$$T_C = \frac{N_{points}}{N_{samples}} \approx 8.205\ \text{ms}$$

According to the Nyquist sampling criterion, the maximum frequency applicable for the FFT is:

$$f_{\max} = \frac{1}{2T} = 3900\,Hz$$

The frequency sampling interval for the FFT is:

Df=1/Tc=121.875 Hz

Note You can obtain the highest sample rate with the ADCIN which has a sample rate up to 46.8 ksps. It allows fast speed but low resolution.

The Firmware

The code is written in C using the example project, C_Example_ADC_UART_LCD.

The ‘fft.h’ file contains the FFT algorithm and the parameters for computation. A standard mathematical library is included.

‘N_points=64’ represents the number of the points. In the ‘main.c’, N_samples=7800’ is the number of samples and ‘time=N_points/N_samples’ the time record.

Three double array of N_points (*data_re[N_points]*, *data_im[N_points],* and *mod[N_points]*) are initialized to zero before computation.

When the samples read from the ADC are ready, they are stored in the *data_re* and *data_im* vectors. The *data_re* array contains the real values of the sampled sequence x[n]. *Data_im* contains the imaginary values.

When the FFT is complete, the code evaluates the absolute values of the spectral components of X(k) and stores the data in the vector *mod[i].* The data is sent to a PC using the RS232 protocol and then displays on the LCD.

The output is provided in the magnitude and frequency of the fundamental harmonic, except for the DC component and the actual frequency sampling interval df.

VC1 provides a sample clock of 3 MHz to the DELTA_SIGMA-11 bit, resulting in a sample rate of 7.8K samples per second. VC3 generates the baud clock of 19200 bps for the UART.

Any terminal utility, such as Hyper Terminal, can be used to view the results. Customized software in C++ is used here to plot the results according to the RS232 protocol.

In Figure 4, the graph in the oscilloscope shows a sine wave shifted upwards by a positive offset.

Figure 4. Experimental Setup

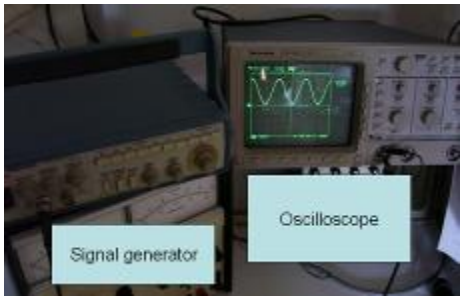
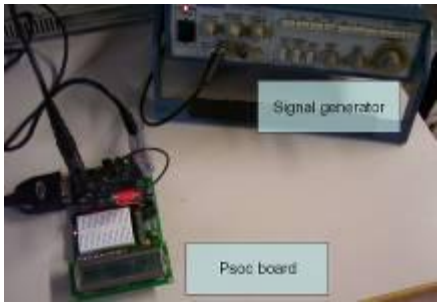


Figure 5 shows the experimental setup with a signal generator connected to the evaluation board through the analog pin input, port0_pin1 and gnd.

Figure 5. Experimental Setup with PSoC Board



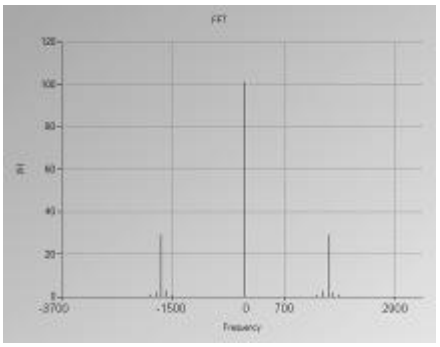
The LCD shows in real time magnitude and phase of the estimated fundamental harmonic in Figure 6.

Some examples of signals in the frequency domain are obtained using a simple data logger plot.

Figure 6. FFT Algorithm Running on PSoCEval1

In Figure 7, the estimated spectrum of a sine wave at a frequency of approximately 1828 Hz (1828.125 Hz) is displayed.

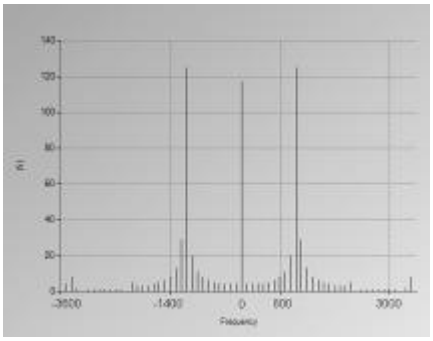
Figure 7. Estimated Spectrum of a Sinusoidal Waveform in the Frequency Domain with a DC Offset



The spectrum consists of a fundamental at approximately 1800 Hz due to the sine wave and one more at zero frequency. The width of the spectrum is proportional to the amplitude of the sine wave. The zero frequency component is due to the DC level (positive), which was used to maximize the signal within the dynamic range of the ADC.

Figure 8 shows the estimated spectrum of a triangular waveform at approximately 1218 Hz comprising a DC offset.

Figure 8. Estimated Spectrum of a Triangular Waveform in the Frequency Domain



A brief analysis of the complexity and speed is provided. Table 1 reports the memory occupation (RAM) of the code with and without paging in terms of N samples. By changing other parameters such as sampling rate, ADC performance, power factors, different results are obtained. For sake of completeness, the values 1, 3 (* N is typically an integer multiple of 2) are included.

Table 1. RAM Occupation

Samples N.	RAM bytes (%)	
	Paging	No Paging
1*	74 bytes (3%)	74 (28%)
2	86 (4%)	86 (33%)
3*	98 (5%)	98 (38%)
4	110 (5%)	110 (42%)
8	158 (8%)	158 (61%)
16	254 (14%)	254 (99%)
32	446 (24%)	Overflow
64	830 (46%)	Overflow
>64	Overflow	Overflow

The real time effectiveness of the algorithm cannot be easily estimated and evaluated. A simple benchmark test based on a timer-interrupt routine is used to measure the response time (24 MHz CPU core frequency). Table 2 contains average results obtained using this test in terms of real time computation. The timing implemented internally perturbs partially the numerical computation.

Table 2. FFT Computation (CPU Frequency 24 MHz)

Samples N.	FFT approx. time (msec)
4	10
16	15
32	20
64	40

Summary

This application note describes the implementation of a Fast Fourier Transform (FFT) algorithm in a PSoC system. It also illustrates the possibility of evaluating the spectrum of an analog input in real time without any external components. A simple embedded FFT implementation is a useful solution in various applications for data acquisition and signal conditioning; for example, biomedical signal analysis such as ECG and EEG.

About the Authors

Name: Nicola Sgambelluri and Gaetano Valenza

Title: PhD, Electronic Engineer and Electronic Engineer, respectively.

Background: Nicola received his MSc degree in Electronic Engineering from the University of Pisa in 2002, and the PhD degree in Automation, Robotics and Bioengineering in 2006.

His main research interests concern embedded control, microprocessors, haptic devices, hardware software systems, real-time applications, domotics, robotics and mechatronics.

Gaetano Valenza is a Biomedical engineer with expertise in bioengineering and electronic applications.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com/