FPGA를 이용한 확률논리회로 A/D 컨버터의 구현

이정원, 심덕선
중앙대학교 제어계측공학과
서울특별시 동작구 흑석동, Tel:820-5329, Fax:825-1858,
Email:jwlee@jupiter.cie.cau.ac.kr, dshim@jupiter.cie.cau.ac.kr

FPGA Implementation of A/D Converter using Stochastic Logic

Jung-won Lee, Duk-sun Shim
Department of Control & Instrumentation Engineering, Chung-ang University
Huksuk-dong Dongjak-gu Seoul, Tel:820-5329, Fax:825-1858
Email:jwlee@jupiter.cie.cau.ac.kr, dshim@jupiter.cie.cau.ac.kr

Abstract— One of the most difficult problem of designing VLSI is a mixed-circuit design, that is to design circuit containing both analog parts and digital parts. Digital to analog converter and analog to digital converter is a typical case. Especially it can be a serious problem when mixed circuit are put into a large digital circuit like microcontroller. However nowadays this problem is settled by separating analog circuit parts outside the IC. This technique is based on converting a digital signal into a pulse sequence. Then an analog signal is obtained by averaging this pulse sequence at the external low-pass filter. An analog to digital converter is designed using a stochastic logic instead of a traditional PWM(Pulse-Width Modulation) signal and is implemented using FPGA. Stochastic pulse sequence can be made as a simple circuits and moreover can be mathematically processed by simple circuits - AND gates. The spectral property of stochastic pulse sequence method is better than that of PWM method. So it make easy to design a external low-pass filter. This technique has important advantages, especially the reduction of the ADC cost.

I. 서 론

최근에 VLSI 기술이 발전함에 따라 기존에 여러 개의 칩으로 구성되어 있던 시스템을 하나의 칩으로 만드는 추세이다. 그러한 작업중 까다로우면서도 자주 직면하게 되는 작업이 바로 ADC(Analog-to-Digital Converter)를 마이크로 프로세서 또는 마이크로 콘트롤러에 내장시키는 작업이다. ADC는 기본적으로 아날로그 소자부분을 가지기 때문에 VLSI에 집적을 할 때 매우 복잡한 혼합 집적 기술과 작업을 요한다. 그래서 ADC의 디지털 집적화가 연구가 되었고 다양한 시도가 이루어졌다. 아나로그 소자부분이 대부분인 플래시(Flash) ADC나 비교적 아나로그 소자부분이 간소화된 Delta-sigma (Oversampling) ADC와 같이 고속의 변환 속도를 가지지는 못하나 디지털 회로만으로 꾸며지고 또한 간단한 외부의 첨가회로로 저가격을 실현한 ADC가 실용화 되기도 하였는데 PWM(Pulse Width Modulation) 방식을 이용한 ADC가 그 예라 할 수 있다[1].

본 연구에서는 확률논리회로(Stochastic logic)[2]를 이용한 DAC(Digital-to-Analog Converter) 및 ADC의 알고리즘[3]을 이용하여 FPGA로 ADC 칩을 제작하였다. 이 방법은 PWM 방식의 ADC에서 PWM 신호 발생기 대신 확률논리회로를 이용한 것으로, 확률논리회로를 이용하여 확률시퀀스(Stochastic sequence)를 생성하고 이 시퀀스를 칩의 외부에서 필터링(Low-pass filtering)을 하여 아날로그 값을 얻어내는 것이다. 간단한 회로로 구성할 수 있어 낮은 가격으로 생산이 가능하고 기존의 PWM 방법보다 스펙트럼 특성이 좋아[1] 외부의 필터회로의 선택이 자유로워진다. 그리고 PWM 방식이 마이크로 프로세서나 마이크로 콘트롤러에 많은 부분을 의존하는 것에 반해 완전히 독립적인 ADC로도 구현하기 쉬운 장점이 있다.

변환 알고리즘 성격상 빠른 변환 속도를 얻을 수는 없으나 구조가 완전히 디지털 회로로만 구성되므로 간단하여 저가격으로 구현이 가능한 장점이 있으며, 전지의 용량 검사, 온도 제어등에 쓰이는 마이크로 콘트롤러에 사용되는 ADC의 성능은 초당 수십에서 수백개의 변환 속도 정도면 충분하므로 여러 분야에 응용될 수 있다. VHDL 언어를 사용하여 설계하였으며, FPGA(Field Programmable Gate Array)를 이용하여 구현하고 테스트 보드를 제작하여 알고리즘과 성능을 테스트해 보았다. VHDL 컴파일러 및 시뮬레이션은 Synopsys 프로그램을 사용하였으며 Altera사의 FLEX8000 시리즈의 칩을 이용하여 제작을 하였다.

II장에서는 확률 논리회로에 대해 살펴보고 III장에서는 이 논리회로를 이용한 DAC, ADC의 구조 및 구현을 IV장에서는 테스트 보드의 제작과 테스트 결과를 기술한다. 마지막으로 V장에서 결론을 내린다.

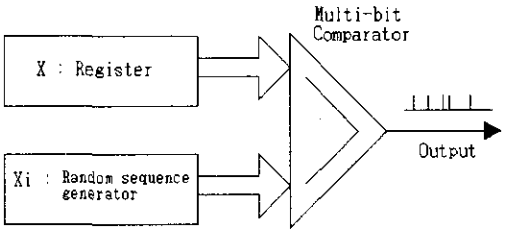

그림 1. 확률 논리회로의 기본 구조

Ⅱ. 확률논리회로

1. 확률논리회로의 구조

확률논리회로는 계산하고자 하는 값을 확률 값으로 변환된 펄스 시퀀스로 변환하고 이를 이용하여 덧셈, 곱셈, 적분 등의 아날로그 계산을 수행하는 디지털 회로이며 기본적인 구조는 그림 1과 같다.

그림1에서 변환하고자 하는 값 X는 균등 불규칙 시퀀스(Uniform random sequence) X_i ($< X_{max}$) 와 비교되고 X의 값이 클 경우만 비교기는 '1'값을 출력한다. 이때 '1'값이 출력될 확률(P_f)은 X/X_{max}이 된다. 따라서 n_a번 펄스를 더한 출력값 $\hat{X}$는 다음의 평균값과 분산을 가지는 이항 분포(binomial distribution)를 따른다[4].

$$E[\hat{X}] = (X_{max}/n_a)\, n_a P_f = X \quad (1)$$

$$V[\hat{X}] = (X_{max}/n_a)^2 n_a P_f (1-P_f) = \frac{1}{n_a} X (X_{max}-X) \quad (2)$$

확률논리회로의 큰 장점중에 하나는 각각의 출력값에 대한 수학적 연산을 매우 간단하게 구현할 수 있다는 점이다. 예로 두개의 확률논리회로의 출력에 AND 게이트를 달아 줌으로써 두 값의 곱을 연산할 수 있다[4].

2. 불규칙 시퀀스 발생기

불규칙 시퀀스 발생기는 LFSR(Linear Feedback Shift Register)을 사용한다. 정해진 클락마다 반복적인 값을 출력하게 때문에 완전히 불규칙한 시퀀스를 내지는 않지만 반복되는 주기가 충분히 큰 때는 불규칙 시퀀스 발생기로 사용이 가능하며 간단한 동기 회로로 구현이 가능하기 때문이 매우 유용하다. LFSR에 대해서는 많은 연구가 되어 있다[5,6]. 초기값이 0이 아닌 이상 0을 제외하고 2^n-1개의 서로 다른 값을 출력한다.

그림2는 이번 구현에 사용된 8비트 LFSR의 구조이며 2^8-1개, 즉 255개의 값을 주기적으로 발생시킨다.

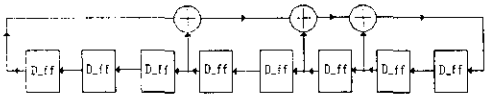

그림 2. 8비트 불규칙 시퀀스 발생기

Ⅲ. 확률논리회로를 이용한 ADC의 구현

1. 확률논리회로를 이용한 DAC

변환하고자 하는 디지털 값을 X라 할 때 확률논리회로의 출력이 X/X_{max}의 확률임에 착안하여 비교기의 출력 값을 필터링(Low-pass filtering)함으로써 원하는 아날로그 값을 얻는다. 확률논리회로의 출력에 간단한 RC 필터를 추가함으로써 구성할 수 있다[3]. 변환 속도는 칩 내부의 클락 속도와 외부 필터 회로의 RC값을 변환함으로써 조정할 수 있다. 칩은 자신이 동작할 수 있는 적정 클락 속도가 있을 것이고 따라서 변환 속도의 조절은 주로 외부의 RC값을 변환해야 하는데, 이 값은 칩에 가해지는 클락 속도와 변환하려는 디지털 값의 비트수에 따라 좌우된다. 변환할 비트수 즉 해상도(Resolution)가 n이라 하고 클락 주파수를 f_{clk}이라 하면 다음 식의 범위내의 RC값을 선택하도록 한다[3].

$$f_{max} < 2^{(-2(n-1))} f_{clk} \quad (3)$$

$$RC > \frac{1}{2\pi f_{max}} \quad (4)$$

예를 들어 100khz의 클락과 8비트의 해상도를 가진다고 했을 때 f_{max}는 약 6.1Hz 정도가 되고 따라서 RC값은 2.61×10^{-2} 보다 큰 값을 선택하도록 한다. 물론 RC 값을 작게 잡을수록 빠른 속도를 얻을 수 있으나 너무 작게 잡으면 리플(Ripple)의 영향이 커서 정확한 변환값을 얻지 못한다. 이에 대한 자세한 실험 결과는 4장에서 기술하도록 하겠다.

2. 확률논리회로를 이용한 ADC

확률논리회로를 이용한 DAC를 이용하여 축차 비교형 ADC(Successive approximation (sampling) ADC)를 구성하였다[3]. 축차 비교형 ADC에 사용되는 SAR(Successive approximation register)을 디지털 값 X를 가지는 레지스터로 사용하고 이 SAR은 외부의 변환하고자 하는 아날로그 값 Vin 과 DAC에서 변환된 아날로그 값 Va 를 비교하는 비교기의 출력에 의해 값을 결정한다. 이렇게 설계된 ADC는 그림3과 같다.

변환 시간 T_{conv}는 ADC에 사용되는 클락을 f_{clk}라고 하고 해상도를 n이라 할 때 다음과 같은 식에 의해 구할 수 있다.

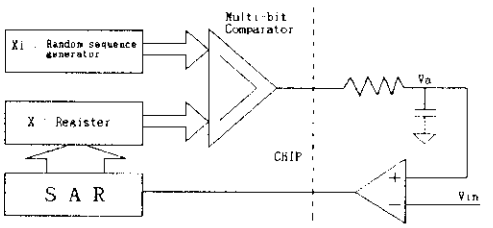

그림 3. 확률논리회로를 이용한 축차 비교형 ADC

$$T_{conv} = n(2^n - 1)\frac{1}{f_{clk}} \quad (5)$$

식(5)에서 (2^n-1)은 불규칙 시퀀스 발생기에서 한 번의 불규칙 시퀀스가 나가는 클락 수이고 n은 SAR에서 n비트의 비교를 하기 위한 클락 수이다. 예를 들어 100kHz의 클락에 8비트의 해상도라고 한다면 T_{conv}는 20.4ms가 된다.

3. FPGA를 이용한 ADC의 구현

ADC의 기본구조를 크게 다음과 같이 5개의 블록으로 나누어 설계를 하였다.

(1) 불규칙 시퀀스 발생기
(2) 비교기(Multi-bit comparator)
(3) SAR(Successive Approximation Register)
(4) 3상 버퍼(Three-state buffer)
(5) 제어기(Control unit)

확률논리회로에서 변환된 신호는 Aout을 통하여 나가고 변환하고자 하는 아날로그 값과 비교기에서 비교되어 COMP로 입력된다. 이 신호가 SAR의 상태를 변화시키도록 하였다. 제어기는 SOC(Start of Conversion) 신호로 변환을 시작하고 한 값이 변환될 때마다 EOC(End of Conversion)로 펄스의 형태로 신호를 내도록 설계하였다. 또한 이 제어기는 불규칙 시퀀스 발생기와 SAR의 동작 타이밍을 결정하고 3상 버퍼의 출력을 제어한다. 설계는 VHDL 언어를 이용하였으며 VHDL 컴파일, 시뮬레이션은 Synopsys사의 컴파일러와 시뮬레이터를, 합성(Synthesis)은 Synopsys사의 FPGA 합성기를 이용하였다. 이러한 구조로 설계된 ADC를 블록 다이어그램으로 나타내면 그림4와 같다.

칩으로 만들기 위해 FLEX8000(EPF8282ALC84-3 (A9613))칩에 맞게 프로그래밍을 하였다. 합성 결과 1009개의 게이트가 사용되었으며 구성(Configuration)을 위해 16핀이 사용되었다. 그림5는 프로그래밍 된 핀배치도이다. Max2plus 프로그램에 의해서 배열되었고 핀배치의 수정은 가하지 않았다.

Ⅳ. 테스트 및 결과

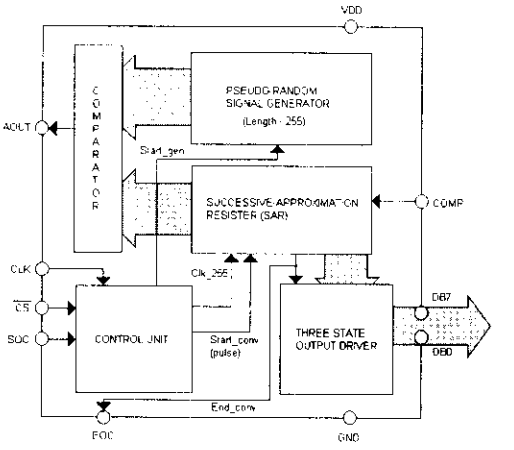

그림 4. 설계된 ADC의 구조

1. 테스트 보드의 제작

FLEX8000은 SRAM방식으로 설계되었기 때문에 구성이 된 뒤에도 전원이 없어지면 그 구성을 유지하지 못한다. 따라서 외부에 따로 롬(ROM)을 달거나 또는 컴퓨터와 인터페이스(Interface) 해주는 장치를 이용하여 사용할 때마다 컴퓨터에서 프로그램을 다운 로딩(Down loading)해야 한다. 테스트 보드에서는 Altera사의 비트 블래스터(Bit Blaster)로 직렬 인터페이스 하여 칩을 구성하도록 하였다.

외부 필터는 여러 가지 RC값을 테스트하기 위해 저항은 1KΩ, 커패시턴스는 0.1㎌, 1㎌, 2.2㎌, 22㎌ 용량의 전해 콘덴서를 사용하였고 비교기는 LM339 칩을 사용하였다. 그리고, 변환된 디지털 값, EOC의 신호 동작 확인을 위하여 컴퓨터와 8255칩을 이용하여 병렬 인터페이스 하였고 그 값은 컴퓨터에서 C언어로 작성된 프로그램에서 살펴 볼 수 있도록 하였다.

2. 테스트 결과

여러가지 클락과 여러가지 RC값을 조절하며 최적화된 변환 속도를 찾는 것에 중점을 두었다. 또한 최적화 되지 않았을 때의 문제점과 해결법에 대한 고찰을 하였다.

변환 속도를 높이기 위해서는 클락 속도를 늘리거나 RC값을 작게 함으로써 얻을 수 있었다. 그러나 칩의 구조 문제상 클락 속도를 높이는 것이 한계가 있고 설사 높이더라도 그에 맞는 RC값을 찾기는 쉬운 일이 아니었다.

큰 RC값은 안정된 필터링을 하지만 상대적으로 큰 시상수로 빠르게 작동하지는 못한다. 따라서 고속의 클락의 경우 RC필터가 원하는 값에 다다르기 전에 비교기의 입력으로 사용됨에 따라 SAR이 오동작하여 옳바른 추정을 할 수 없었다.

RC값을 매우 작게 하면 빠른 변환 속도를 얻을수는 있으나 작게 하면 작게 할 수록 리플이 커져 심각

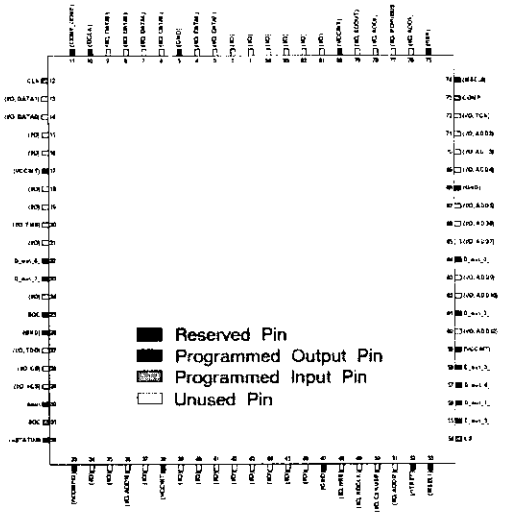

그림 5. 프로그래밍 된 칩의 핀 배치도

한 오차를 나타낸다. 리플이 큰 순간의 값과 아날로그 입력이 비교될 때 비교기는 올바른 출력을 할 수 없다. 따라서 적정한 클락 속도와 그에 맞는 RC값을 찾는 것이 중요하다.

여러번의 실험 결과 설계한 ADC는 클락은 100~400kHz, R은 1kΩ, C는 1㎌에서 오차없이 정확한 결과를 나타내었다. 따라서 초당 20~40샘플의 변환 속도를 가지는 것이다. 이러한 결과는 그림6에 나타나 있고 2.5V와 0V의 입력시 커패시터의 전압 변화를 보여주고 있다. 상단의 파형은 ADC의 추정을 나타내고 하단의 파형은 한 번의 변환이 있은 후 EOC에서 변환의 끝을 나타내는 펄스이다.

확률논리회로를 이용한 ADC 알고리즘이 위의 성능으로 잘 동작하는 것을 확인하였다. 변환 속도가 느리기 때문에 이러한 변환 속도의 향상을 위해서는 다음의 방법이 있을 수 있다. RC값을 작게 하면 리플의 영향으로 비교기에서 올바른 결과가 출력되지 않는 것이 문제라면 한가지 방법으로 비교기의 출력부에 추가의 디지털 필터를 달아 올바른 값을 추정해 내는 것이다. SAR의 입력단 앞에 교류성분을 제거하는 디지털 필터를 칩에 내장시킴으로써 RC값을 작게 선택할 수 있다. 다른 한가지 방법으로는 1차 RC필터를 쓰는 대신 고차의 필터를 사용하는 것이다.

SAR의 초기값인 2.5V를 추정할 때의 시간은 다른 추정시간의 2배 가량이 걸린다. 따라서 이 값의 정확한 추정을 원한다면 SAR의 초기치를 2회 연속 출력함으로써 보정할 수 있다. 그러나 이렇게 한다면 변환 속도가 느려지게 된다.

V. 결 론

본 논문에서는 확률논리회로를 이용한 ADC를 실제 FPGA로 구현하고 테스트 보드를 꾸며 알고리즘을 검증하고 그 성능을 테스트해 보았다. 테스트 결과 순수 알고리즘의 구현만으로 100kHz의 클락, 1kΩ, 1㎌의

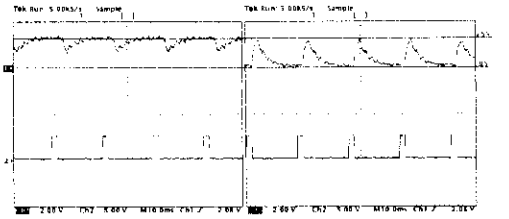

그림6. 2.5V와 0V 입력시 커패시터의 전압변화
(R=1kΩ,C=1㎌,Clk=100kHz)

RC값에서 초당 20~40샘플의 변환 속도로 오차없이 정확히 변환이 되었다.

고속의 ADC에 비하면 매우 느린편이나 확률논리회로를 이용한 ADC는 완전히 디지털로 구현이 되어 있기 때문에 다른 칩에 내장시키기 용이하고 저가격으로 구현이 가능하다. 또한 기존의 PWM보다 구성 및 외부회로의 설계도 간단하다. 변환 속도는 칩에 교류성분을 효과적으로 추출해내는 디지털 필터를 내장시키고 외부의 필터를 고차의 필터로 꾸밈으로써 크게 향상될 것으로 기대된다.

감사의 글

이 연구에서 사용된 Synopsys, Max2plus 프로그램과 Flex8000 FPGA 칩은 반도체설계교육센터(IDEC)에서 지원 받은 것입니다.

참고 문헌

[1] Kevin M. Daugherty, Analog-to-Digital Conversion :A practical approach, McGraw-Hill, 1995.

[2] B.R.Gaines, "Advances in Information Systems Science," Advances in Information Systems Science, 2, 1969.

[3] J.G.Ortega, C.L.Janer, J.M.Quero, L.G.Franquelo, J.Pinilla and J.Serrano, "Analog to Digital and Digital to Analog Conversion Based on Stochastic Logic," IEEE. 21st. International Conference on Industrial Electronics, Control and Instrumentation (IECON'95), Orlando, Fl, U.S.A., pp. 995-999, Nov. 6-10, 1995

[4] Y.Kondo and Y.Sawada, "Functional Abilities of a Stochastic Logic Neural Networks," IEEE Trans. Neural Networks, Vol. 3, pp. 434-443, 1992.

[5] W.Peterson, Error Correcting Codes, MIT Press, 1992.

[6] Miron Arbramovic ,Melvin A. Breuer and Arthur D. Friedman, Digital Systems Testing And Testable Design, IEEE Press, 1990.