FPGA를 이용한 TSK 퍼지 프로세서 설계

김 태 성, 이 원 창, 강 근 택
부경대학교 공과대학 전자공학과

Design of TSK-Fuzzy Processor Using FPGA

Taesung Kim, Wonchang Lee, Geuntaek Kang
Department of Electronic Engineering, Pukyong National University

Abstract - FPGA는 ASIC설계의 시험을 위한 테스트용으로 많이 사용되었으나 최근에는 비약적인 성능 향상으로 그 자체로 기능을 구현하고 있다. 퍼지 제어기의 구현은 일반적으로 범용 마이크로 프로세서를 이용하거나 DSP 프로세서를 이용하였다. 본 논문에서는 여러 퍼지 시스템 중에서 적은 규칙수로도 효과적인 성능을 나타내고 프로세서화가 용이한 TSK 퍼지 시스템을 구현한다. 대상 FPGA는 Xilinx사의 FPGA를 이용하고 Schematic과 VHDL을 혼용하여 설계한다. 또한 구현된 프로세서의 범용성을 유지하기 위해 외부 ROM에서 연산에 필요한 계수를 취하는 방식을 채택한다.

1. 서 론

FPGA는 주문형 반도체 제작의 활성화로 ASIC 제작의 전 단계로서 설계 및 검증 및 시스템의 성능개선과 소형화를 위해 다양한 분야에서 적용하고 있다. 현재는 FPGA자체의 구현 가능한 게이트수가 큰 폭으로 증가하였고, 고속화와 저전력화로 대규모 시스템이나 마이크로 프로세서의 설계에까지 이용되고 있다.

퍼지 시스템은 복잡한 비선형 시스템을 다루는 분야에서 인간의 언어적 논리를 표현하는 방법으로 사용되어져 왔다. 여러 형태의 퍼지 시스템 중에서 TSK 퍼지는 결론부가 선형식으로 표현되고 전제부의 퍼지집합 형태를 간단한 선형식으로 표현할 수 있어서 하드웨어적으로 구현이 유리하다. 본 논문에서는 FPGA를 이용하여 TSK 퍼지 프로세서를 구현하고 다양한 응용분야에 적용을 위해서 Flash ROM에 필요한 계수를 저장하여 범용성을 갖춘 프로세서를 설계한다.

본 논문은 다음과 같이 구성된다. 2.1절에서 TSK 퍼지 시스템과 FPGA구현을 위해 고려되어야 할 사항을 설명한다. 2.2절에서는 설계된 TSK 퍼지 프로세서에 대해서 설명하고 마지막으로 결론을 맺는다.

2. 본 론

2.1 TSK 퍼지 시스템

2.1.1 TSK 퍼지 시스템

복잡한 비선형 시스템을 다루는 지능시스템의 한 분야로서 퍼지시스템은 언어적 표현을 기술할 수 있는 형태로 많은 응용분야에 적용되었다. TSK 퍼지 추론법[3]은 복잡한 시스템에 대해서도 해석적인 지식이 필요없이 시스템의 입출력 데이터만으로 퍼지모델의 작성이 가능하다. TSK퍼지모델은 다음과 같이 결론부가 선형식인 퍼지 규칙들로 구성되어 있다.

$$M^i: \text{ if } z_1 is F_1^i,\ z_2 is F_2^i \cdots\ z_m is F_m^i \quad \text{then } y^i = a_0^i + a_1^i x_1 + \cdots + a_n^i x_n \tag{1}$$

여기서 M^i은 모델의 i번째 규칙임을 뜻하고, z_i는 전제부 변수, F_j^i는 x_i 위의 한 퍼지집합, x_j는 결론부 변수, a_j^i는 결론부 파라미터, y^i는 규칙 M^i로부터의 출력을 각각 뜻한다. 출력 y는 다음과 같이 구한다.

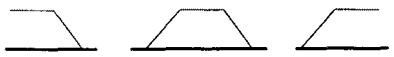
$$y = \sum_{i=1}^{m} w^i(z) y^i / \sum_{i=1}^{r} w^i(z) \tag{2}$$

여기서 r은 규칙의 갯수이며 $w^i(z)$는 i번째 규칙의 적합도를 나타내며 다음과 같이 구한다.

$$w^i(z) = \prod_{j=1}^{m} F_j^i(z_j) \tag{3}$$

여기서 $F_j^i(z_j)$는 퍼지집합 F_j^i에서 z_j의 멤버쉽 치이다.

TSK 퍼지 모델에서 전제부 퍼지집합의 멤버쉽 함수는 〈그림1〉과 같은 세가지 형태의 구분선형함수로 표현할 수 있다[4].



〈그림1〉 퍼지집합의 형태

입력 x에 대한 〈그림1〉의 퍼지집합의 멤버쉽값을 $A_1(x)$, $A_2(x)$, $A_3(x)$라고 하면, 아래와 같이 나타내어진다[4].

$$A_1(x)=0.5-(|x-p_1|-|x-p_2|)/(p_2-p_1) \quad (4a)$$

$$A_2(x) = (|x-p_1|-|x-p_2|)/(p_2-p_1) - (|x-p_3|-|x-p_4|)/(p_4-p_3) \quad (4b)$$

$$A_3(x)=0.5+(|x-p_3|-|x-p_4|)/(p_4-p_3) \quad (4c)$$

2.1.2 프로세서 구현의 고려 사항

TSK 퍼지 시스템에서 멤버쉽값의 범위는 0에서 1 사이의 실수를 사용한다. 실수연산은 연산자 자체의 구현뿐만 아니라 칩 상에서 많은 용적을 차지하므로 FPGA로 구현이 어렵다. 이러한 단점을 극복하기 위하여 본 논문에서는 부호없는 정수형 연산을 사용한다. 멤버쉽치는 8비트로 이산화하여 사용하고, 이산화된 나누기의 몫을 계산하는 정수형 나누기 연산자[5]를 구현하였다. 프로세서 전체적으로는 8비트의 데이터 버스를 사용하지만 연산의 정밀도를 유지하기 위해 각 연산 블록에서는 필요한 정도의 비트 확장을 하여 계산하며 부호없는 정수형 연산을 사용하고 있다. 앞에서 제시한 맴버쉽값 계산 (4a), (4b), (4c)는 이산화된 나누기 연산자를 이용하면 아래와 같이 된다.

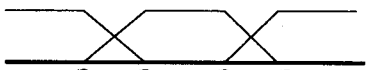
$$\overline{A_1}(x)=128-DIV((|x-p_1|-|x-p_2|),(p_2-p_1)) \quad (5a)$$

$$\overline{A_2}(x) = DIV((|x-p_1|-|x-p_2|),(p_2-p_1)) - DIV((|x-p_3|-|x-p_4|),(p_4-p_3)) \quad (5b)$$

$$\overline{A_3}(x)=128+DIV((|x-p_3|-|x-p_4|),(p_4-p_3)) \quad (5c)$$

여기서 DIV는 이산화 나누기 연산자이고 $\overline{A_1}(x)$, $\overline{A_2}(x)$, $\overline{A_3}(x)$는 멤버쉽값이다.

실수연산을 배제함으로써 사용해야 하는 나누기 연산자는 FPGA 상에서 많은 용적을 차지한다. 나누기 연산의 사용을 줄이고 계산 속도를 향상시키기 위해서 앞에서 서술한 퍼지집합의 형태를 전제부 변수의 퍼지집합이 〈그림3〉과 같이 제안된 형태의 퍼지집합의 끝점이 서로 일치하도록 그 형태을 제한한다.



〈그림2〉 제안된 퍼지집합

〈그림3〉과 같이 전제부 변수 형태를 제한하면 식(2)의 분모 부분이 항상 1이 되므로 출력 y는 다음과 같이 간략한 형태로 표현이 된다.

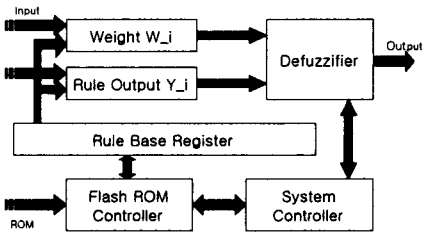
$$y=\sum_{i=1}^{m} w^i(z)y^i \quad (6)$$

2.2. TSK 퍼지 프로세서의 설계

2.2.1 전체 설계사양

본 논문에서 구현된 TSK 퍼지 프로세서는 최대 10개의 퍼지 규칙을 설정할 수 있고, 각 규칙에서 전제부 퍼지집합은 3개를 가질 수 있다. 5개의 입력과 1개의 출력을 가지고, 퍼지규칙을 표현하기 위해 사용되는 각 계수는 외부 ROM에 저장되며 모두 8비트의 값을 가진다.

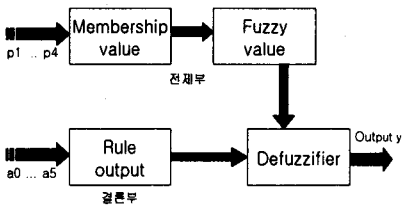
구현을 위하여 사용된 FPGA는 Xilinx사의 Virtex 시리즈중 XCV400-6HQ240C 이다[1]. 이 칩은 최대 130,000 Logic gate를 사용할 수 있는 고용량의 FPGA 칩이다. 또한 전체 설계를 Xilinx Foundation Series 2.1f에서 Schematic과 VHDL을 혼용하여 설계했다[2]. 설계 방식은 Top-down 방식을 취하여 Schematic 상에서 각 기능 블록을 표현하고 연결하였다. 각 블록의 설계는 VHDL을 사용하였다. 구현된 전체 블록 다이어그램은 아래의 〈그림3〉과 같다.



〈그림3〉 전체 블록 다이어그램

2.2.2 퍼지 연산부 설계

퍼지 연산부는 아래의 그림과 같이 연산의 특징과 순서에 따라 4개의 블록으로 나누어 설계하였다.



〈그림4〉 퍼지 연산부 블록 다이어그램

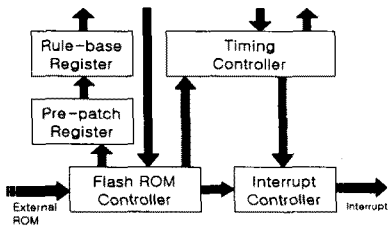
전제부의 멤버쉽값은 〈그림1〉에서 보여지는 세 가지 형태중 하나를 선택하고 p1, p2, p3, p4의 값과 프로세서로 입력되는 x_i값을 입력받는다. 식(5)를 계산하고 퍼지값 계산 블록으로 값을 넘겨준다. 퍼지값 계산 블록은 전제부 퍼지집합의 개

수만큼 멤버쉽값을 받아서 w^i를 계산한다.

결론부는 프로세서 입력값 x_i와 결론부 계수 를 입력받아 각 규칙의 출력 y^i를 계산하고, 마지막으로 비퍼지화 블록에서 식(6)을 이용하여 최종 프로세서 출력 y를 계산한다.

2.2.3 제어부와 레지스터 설계

제어부는 프로세서 전체의 동작이 원활하게 이루어지도록 하는 부분이다. 타이밍 제어부는 퍼지 연산부의 블록간의 연산 시간의 오차를 보정해 주기 위한 타이밍 문제를 해결하는 부분이다. Flash ROM 제어부[6][7]는 연산에 필요한 계수를 프로세서 외부의 ROM에서 읽어와서 레지스터에 저장하는 역할을 하며, 연산 동작이 ROM의 데이터를 읽는 과정에서 끊어지는 것을 방지하기 위하여 다음 연산의 계수를 미리 읽어 오도록 하는 부분이다. 인터럽트 제어부[7]는 연산에서의 오버플로우나 프로세서가 외부와 결합하여 동작 중에 일어나는 문제의 인터럽트를 체크하고 인터럽트 신호를 발생하는 부분이다. 제어부의 전체 블록 다이어그램은 〈그림5〉와 같다.



〈그림5〉 제어부 블록 다이어그램

Rule-base 레지스터는 연산에 필요한 계수를 저장하는 레지스트이다. Pre-patch 레지스트는 한 동작 앞서 미리 데이터를 입력받는 레지스터로서 Rule-base 레지스터와 같은 형태를 가진다. 레지스터의 구성은 〈표1〉과 같다.

〈표1〉 Rule-base 레지스터

MSB							LSB
Total_set	Current_set	Type					
P1							
P2							
P3							
P4							

첫 번째 레지스터의 Total_set은 전제부 변수에서 사용되는 퍼지집합의 총 개수를 나타내고 Current_set은 아래에 있는 네 개의 값이 사용되는 퍼지집합의 번호를 나타낸다. 이와 같은 레지스터 그룹이 세 개가 있다. 그리고 마지막에는 결론부 변수를 나타내는 A0에서 A5까지의 8비트 레지스터가 6개 있다. 위의 전제부와 결론부 계수를 저장하는 레지스터의 앞쪽에 퍼지 규칙의 수와 현재 연산하고 있는 규칙의 번호를 알려주는 〈표2〉와 같은 레지스터를 둔다.

〈표2〉 Rule Number 레지스터

MSB							LSB
Total_Rule				Current_Rule			

위에서 나타낸 22개의 레지스터의 구조는 외부 ROM에 똑같은 형태로 저장되고 한번에 22 바이트씩 읽어 온다. 퍼지집합의 형태에 따라서 값이 없는 레지스터가 생기고 이곳은 모두 0으로 채워진다.

3. 결 론

본 논문에서는 FPGA를 이용하여 TSK 퍼지 프로세서를 설계하였다. 설계의 용이를 위하여 실수값으로 표현되던 멤버쉽값 연산을 8비트로 이산화시키고 이산화된 연산을 위한 나누기 연산자를 설계하였다. 전제부 퍼지집합의 형태를 제한하여 비퍼지화 연산을 간단히 하여 FPGA 합성시의 과다한 용적 문제를 고려하였다.

구현된 TSK 퍼지 프로세서가 특정한 영역에서만 사용되는 것을 방지하고 여러 영역에서 범용성을 유지하도록 범용 마이크로 프로세서의 특징인 외부 ROM을 사용하였다. Flash ROM을 채택함으로써 PC에서 필요한 데이터를 다운로드하여 쓸 수 있도록 고려하였다.

[참 고 문 헌]

[1]Xilinx, "The Programmable Logic Data Book", versin1.03, PP.4,1-4,49. 1996

[2]Xilinx, "XACTstep Foundation", version1.03, PP.1-128, 1996

[3]Li-Xin Wang, "A Course in Fuzzy Systems and Control", Prentice-Hall International, Inc., PP.265-276

[4]고경천, "퍼지이론을 이용한 Robot Manipulator의 역기구학 제어에 관한 연구", 부산수산대학교, PP.9-10, 1992. 11

[5]K. C Chang, "Digital Systems Design with VHDL and Synthesis", Computer Society, PP.445-465

[6]박세현, "VHDL에 의한 디지털 컴퓨터 설계와 구현", 도서출판 그린, PP.385-404, 1999. 11

[7]김태성, 이진희, 강근택, 이원창, "회전체 신호처리 시스템의 ASIC 설계", 대한전자공학회 부산·경남지부 춘계 학술대회 논문집, 1999. 6.