부동소수점 기반의 포맷 컨버터를 이용한 효율적인 지수 함수 근사화 알고리즘의 FPGA 구현

Implementation of Efficient Exponential Function Approximation Algorithm Using Format Converter Based on Floating Point Operation in FPGA

김 정 섭, 정 슬*
(Jeong-Seob Kim and Seul Jung)

Abstract: This paper presents the FPGA implementation of efficient algorithms for approximating exponential function based on floating point format data. The Taylor-Maclaurin expansion as a conventional approximation method becomes inefficient since high order expansion is required for the large number to satisfy the approximation error. A format converter is designed to convert fixed data format to floating data format, and then the real number is separated into two fields, an integer field and an exponent field to separately perform mathematic operations. A new assembly command is designed and added to previously developed command set to refer the math table. To test the proposed algorithm, assembly program has been developed. The program is downloaded into the Altera DSP KIT W/STRATIX II EP2S180N Board. Performances of the proposed method are compared with those of the Taylor-Maclaurin expansion.

Keywords: exponential function approximation, floating-point processor, FPGA

I. 서론

최근 지능형 로봇 시장의 규모가 점차 성장하고 있으며 이에 따라 지능형 로봇 시스템을 구동하기 위한 임베디드 프로세서의 수요가 급증하고 있다. 또한 임베디드 프로세서의 성능은 로봇의 성능에 직접적으로 영향을 미치기 때문에 향후에는 보다 고성능의 프로세서 사용이 요구될 것으로 예측할 수 있다. 특히, 로봇의 모션 제어에 있어서는 고전적인 PID 제어 알고리즘만으로는 외란 요소가 많은 환경에서 동작하는 지능형 로봇의 정확한 제어에 적합하지 않다고 할 수 있다. 따라서 그러한 외란에 강건한 제어를 위해서는 지능제어를 사용하는 것이 적합하다. 지능제어 알고리즘은 그 연산의 복잡성 때문에 비교적 고성능의 컴퓨팅 파워를 가지는 디바이스를 사용해야 하기 때문에 많은 시스템에서 고성능의 DSP를 사용하여 구현을 한다. 하지만 고성능의 DSP를 사용할 경우 비용이 증가하는 것과 함께 로봇 제어를 위한 충분한 부가적인 모듈이 부족하다는 단점이 존재한다. 이러한 점을 보완하기 위해 FPGA를 병행하여 사용하기도 하며[1,2], FPGA만을 사용하기도 한다[3].

고정밀도 연산을 위해서는 부동소수점 기반의 시스템 구현이 필요하다. IEEE 754 부동소수점 포맷 기반의 연산은 그 특성상 프로세서 타입의 디바이스로 순차적인 처리를 하는 것이 효율적이다. 따라서 부동소수점 기반의 산술 연산 모듈이 포함된 프로세서의 개발이 요구된다. 이러한 연구는 이미 많은 학자들 사이에서 논의된바 있으며[2], 부동소수점 기반에서의 지능 알고리즘이 구현되어 있으며[4,5], 본 논문에 있어서도 이미 지능 알고리즘을 FPGA에서 구현하기 위한 부동소수점 프로세서에 대한 선행 연구가 있었다[6,7]. RBF (Radial Basis Function)를 구현하기 위해 Taylor Series를 사용하여 근사화하였다[6]. 이런 근사화 기법을 사용할 경우 지수 함수의 입력 값이 크다면 많은 차수의 Taylor 전개를 할 지라도 근사화 오차는 상당히 커지게 된다. 따라서 이러한 약점을 보완하기 위한 새로운 접근방법이 필요하게 된다.

본 논문에서는 선행 연구[6-10]에서 구현된 부동 소수점 기반의 프로세서를 이용하여 제안된 지수함수 근사화 알고리즘에서 IEEE754 규약의 부동소수점 포맷의 특성상 구현이 어려운 부분에 대한 처리를 부동소수점-고정소수점 컨버터와 math table을 사용하여 지수 함수를 보다 효율적으로 근사화 시킬 수 있는 방법에 대해 소개하고자 한다.

II. Taylor-Maclaurin 확장

Taylor Series는 대표적인 비선형 함수 근사화 방법 중 하나이다. 지수 함수 근사화에 따른 Taylor-Maclaurin 전개는 아래와 같다.

$$e^x = \sum_{n=0}^{\infty} \frac{1}{n!} x^n \tag{1}$$

식 (1)을 전개시키면 다음과 같다.

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{3}x^3 + \frac{1}{4}x^4 + \frac{1}{5}x^5 + \frac{1}{6}x^6 + \frac{1}{7}x^7 + \frac{1}{8}x^8 + HOT \tag{2}$$

Taylor-Maclaurin 확장에 따른 오차 함수는 아래와 같다.

$$R_n(x) = \frac{f^{n+1}(\xi(x))}{(n+1)!} x^{n+1} \tag{3}$$

* 교신저자(Corresponding Author)
논문접수: 2009. 4. 30., 수정: 2009. 8. 12., 채택확정: 2009. 9. 9.
김정섭, 정슬: 충남대학교 메카트로닉스공학과
(insideasuram@hotmail.com/jungs@cnu.ac.kr)
※ 본 연구는 교육과학기술부 2008년 지역혁신인력양성사업의 지원으로 수행되었으며 이에 감사를 드립니다.

전개시키는 차수가 높아질수록 근사화 오차는 줄어들지만 그 만큼 연산은 늘어나게 된다. Taylor-Maclaurin 확장을 10차까지 전개시켰을 경우 최대 오차는 아래와 같이 계산할 수 있다.

$$R_{10}(1)=\frac{f^{11}(\xi(x))}{11!}x^{11}=\frac{(1)^{11}}{11!}<3\times10^{-8}$$

III. Exponential Function Approximation Algorithm

지수 함수는 아래와 같은 식으로 표현된다.

$$f(x)=e^{x} \tag{4}$$

여기서 x 는 실수이며, 다음과 같이 다시 쓸 수 있다.

$$x=a.b \tag{5}$$

여기서 a 는 x 값이 가질 수 있는 범위에서의 가용 정수 값이며, b 는 1 미만의 실수이다. 단정도 기반에서 표현할 수 있는 수의 범위를 고려할 때 a 가 기질 수 있는 범위는 -88 ~88이다. 식 (4)는 아래와 같이 다시 표현할 수 있다.

$$f(x)=e^{x}=e^{a+b}=e^{a}\cdot e^{b} \tag{6}$$

지수함수는 식(6)과 같이 e^{x} 는 e^{a} 와 e^{b} 의 곱으로 표현이 된다. 정수 a 의 범위가 제한되어 있기 때문에 e^{a} 값을 가질 수 있는 경우도 제한적이다. 따라서 e^{a} 에 대한 값을 math table로 미리 연산 결과값을 저장하여 사용하는 것이 가능하다. 정수 a 의 범위가 -88 ~ 88이기 때문에 math table을 만들기 위한 메모리 주소는 177개가 필요하게 된다. e^{b} 의 경우는 b 가 1 미만의 값을 가지므로 적당한 차수 전개의 Taylor-Maclaurin 확장에 의해 근사화가 가능하다.

e^{a} 에 대한 math table은 아래의 표와 같으며, 이에 대한 값은 32bit single precision 기반의 포맷 형태로 저장되어 프로세서 내부에 위치하게 된다.

e^{b} 에 대한 근사화는 Taylor-Maclaurin 확장의 오차 공식에 의해 단정도 기반에서의 표현범위를 고려하여 20차수에서 근사화 시키는 것으로 충분하다.

Exponential math table과 Taylor-Maclaurin 확장에 의해 e^{a} 와 e^{b} 의 값을 각각 참조 및 근사화 과정을 통해 구한 후, 이를 곱하여 최종 결과를 계산한다. 따라서, 입력 변수의 크기에 관계 없이 제한된 연산 횟수를 가지고 정확한 근사값을 구할 수 있다. 그림 1의 순서도에서 e^{a} 부분과 e^{b} 부분을 따로 연산하게 되는데, 새로운 명령어는 e^{a} 에 대한 값을 math table에서 참조하도록 하는 명령어이다.

표 1. 지수함수 테이블.
Table 1. Exponential math table.

Equation	Value
e^{-88}	6.054601895401186e-039
e^{-87}	1.645811431082274e-038
e^{-86}	4.473779306181121e-038
…	…
e^{86}	2.235246603734715e+037
e^{87}	6.076030225056873e+037
e^{88}	1.651636254994002e+038

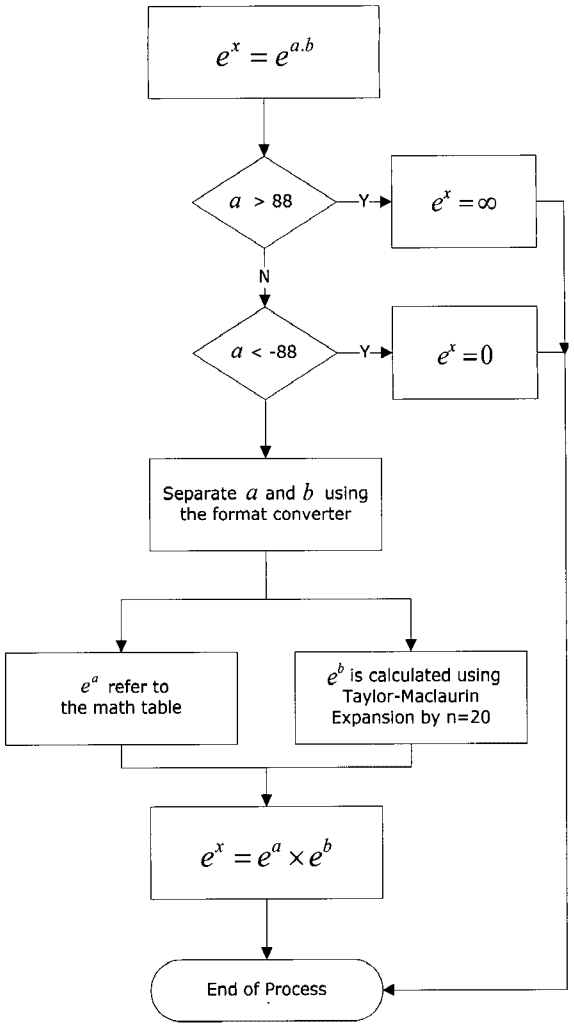


그림 1. 지수 연산 알고리즘 순서도.
Fig. 1. Flowchart of the exponential approximation algorithm.

위의 순서도는 Taylor-Maclaurin 확장을 이용한 단정도 표현 범위 안에서의 근사화 알고리즘을 나타내고 있다. IEEE754 부동소수점 포맷은 부호, 지수부, 그리고 가수부를 표현하는 영역이 분리가 되어있으며, 실제 값은 이들의 곱으로 표현이 되기 때문에 실제 값을 정수와 1보다 작은 소수로 나누기 어렵다. 제안된 알고리즘에서는 부동소수점 프로세서의 floating-point format to fixed-point format converter를 이용하여 이 부분에 대한 처리를 하드웨어로 구현한다.

IV. Hardware Implementation

1. 프로세서 설계 구조

이전 연구를 통해서 Floating-point Processor Architecture를 설계하였다[3,6,7]. 기존의 연구에서는 단순히 고정된 차수의 Taylor-Maclaurin 전개에 대한 수식을 어셈블리 코드로 구현하였다. Taylor-Maclaurin전개 20차수까지만 전개하면, 한정된 프로세서의 연산 횟수를 가지고 32비트 단정도 부동소수점 표현 영역에서의 근사화 오류를 최소화 시킬 수 있다. Math table은 이 알고리즘을 구현하기 위해 표 1의 데이터를 담아

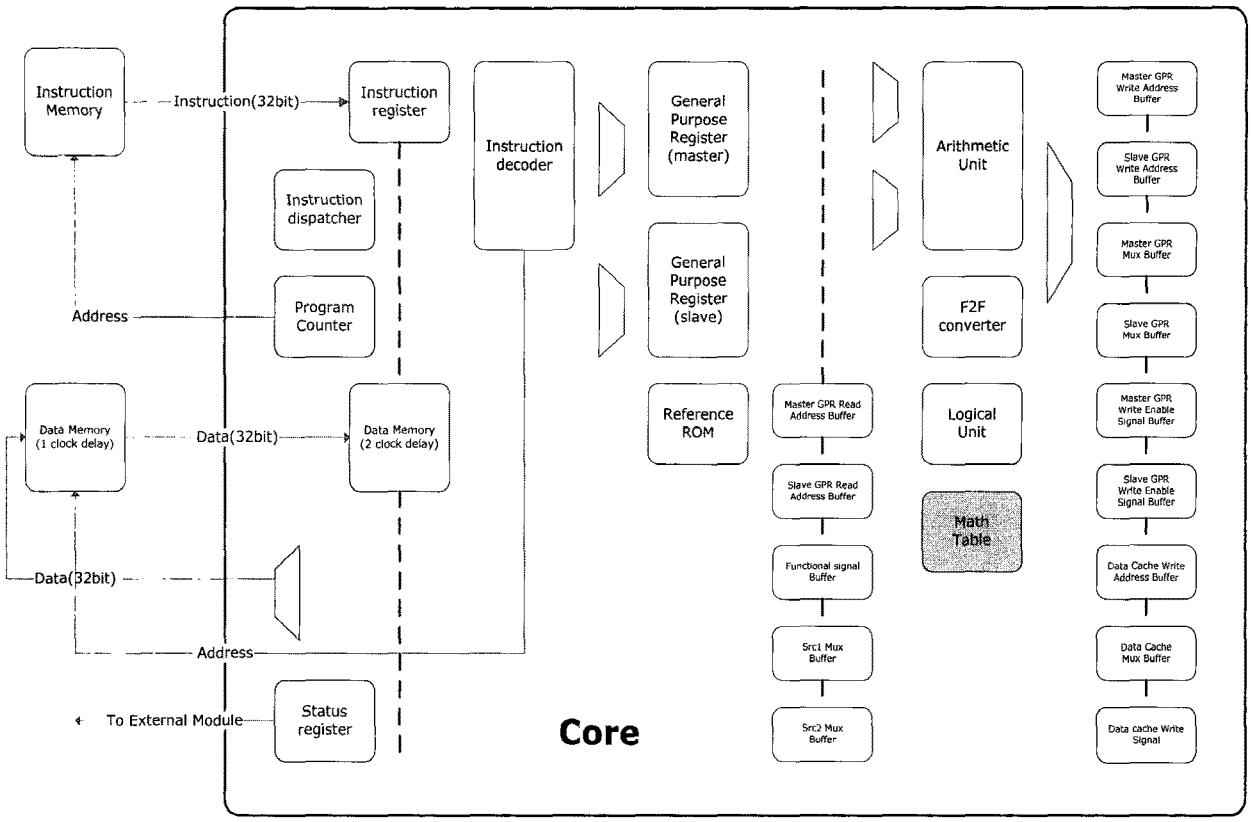


그림 2. 부동소수점 연산기의 구조.

Fig. 2. Architecture of floating-point processor core.

놓은 일종의 LUT (Look-Up Table) 이다. 따라서, 새로운 exponential function algorithm의 적용을 위해 기존의 구조에서 exponential math table module을 추가하고 새로운 명령어를 정의한다.

표 1에 나와있는 177개의 32bit 값의 math table을 VHDL로 코딩하여 exponential math table module을 만든다. 이 모듈은 입력 변수 x 에서 정수부에 대한 정수 형태의 값이 모듈에 입력되어 그 값에 해당하는 지수 함수 값을 출력시키는 역할을 수행한다. 이에 따라 참조 가능한 범위의 정수에 대한 지수 함수 연산 결과값을 출력시키는 동작을 수행할 수 있는 명령어를 정의하는 것이 필요하다. 따라서 LMD (Load Math-table Data)라는 명령어를 아래와 같이 정의한다.

전체적인 구조는 32bit 중에 상위 7bit의 op code와 하위 4bit의 master general purpose register 주소 번지로 이루어져있으며 해당 레지스터에 저장되어 있는 값은 exponential math table

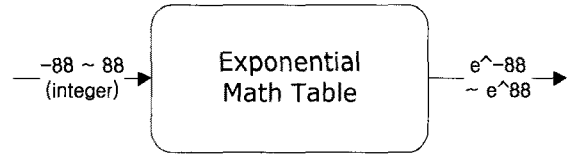


그림 3. 지수 함수 연산 테이블.

Fig. 3. Exponential math table module.

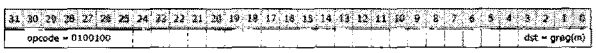


그림 4. LMD 명령어의 형태.

Fig. 4. Format of LMD instruction.

의 참조를 위해 -88~88 사이의 정수 값만이 유효하다.

Exponential math table module은 그림 2와 같이 execute cycle 이 이루어지는 곳에 위치하며, LMD 명령어에서 지정된 레지스터에 저장되어있는 정수 형태의 값으로 참조된 지수 함수의 연산 값이 해당 모듈로부터 나와 다시 LMD 명령어에서의 레지스터 주소 번지에 저장된다. 따라서 전체적인 동작은 format converting instruction들과 비슷하다고 할 수 있다.

2. FPGA Implementation

새로 정의된 명령어를 적용시키기 위해 processor core 내부에 존재하는 instruction decoder를 수정했으며, 파이프라인 단계에서 execute 단계가 실행되는 위치에 exponential math table module을 설계하여 위치시켰다. 또한 명령어 해석과의 동기화를 위해 multiplexer 또한 수정하였다.

그림 5는 새로운 지수함수 근사화 알고리즘을 구현하기 위해 설계된 시스템의 전체적인 모습을 나타내고 있다. 이

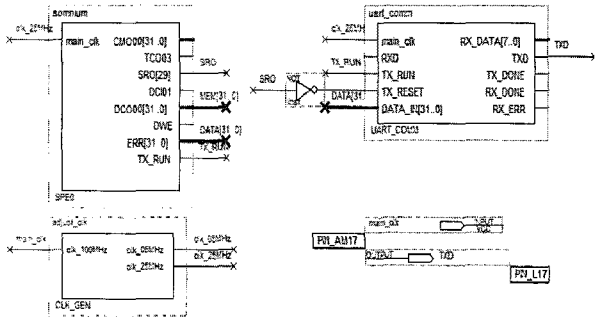


그림 5. 시스템의 전체 구조.

Fig. 5. Overall structure of the system.

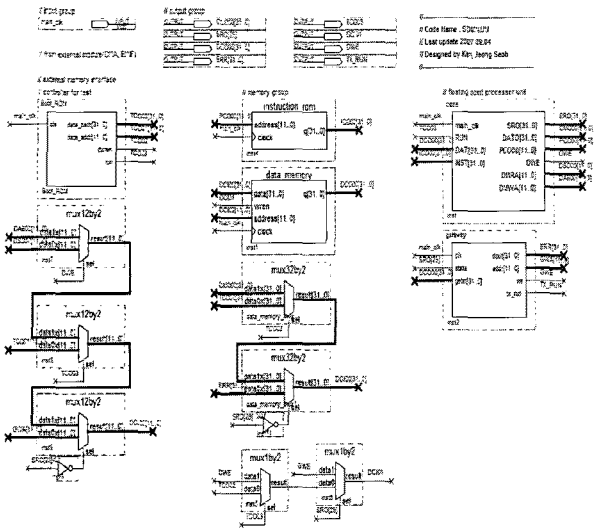


그림 6. SOMMUM 프로세서의 FPGA 구현.
Fig. 6. FPGA implementation of SOMNIUM processor unit.

시스템의 핵심 유닛인 SOMNIUM 프로세서 유닛과 PC와의 데이터 통신을 위한 UART 모듈, 그리고 기본 클럭을 조정해서 내보내도록 하는 모듈로 구성되어 있다.

부동소수점기반의 실제 연산은 SOMNIUM 프로세서 유닛에서 행해지며 유닛 내부의 status register의 상태에 따라 연산 결과 데이터를 UART 모듈로 전송하게 된다.

그림 6은 VHDL로 구현된 SOMNIUM 프로세서 유닛의 내부 구조를 나타내고 있다. 4단계 파이프라인 기반의 프로세서 코어를 기본으로 명령어를 담고 있는 program memory, 연산에 필요한 데이터를 담고 있는 data memory, 프로세서 시스템의 초기화를 담당하는 boot ROM 그리고 전체적인 연산의 흐름을 제어하기 위한 제어 모듈로 구성이 되어 있다.

그림 7은 SOMNIUM 프로세서 유닛의 코어 모듈의 내부 구조를 나타내고 있다. 크게 instruction register, program counter, instruction decoder, register group, reference math rom, arithmetic unit, logical unit, format converter 그리고 지수 함수 근사화 알고리즘의 하드웨어 구현을 위한 exponential math table module로 구성되어있다.

3. Code Implementation

지수 함수 근사화 알고리즘의 성능을 검증하기 위해 미리 정의된 명령어를 이용하여 수식을 전개하였다. 지수 함수 근사화 알고리즘의 구현을 위한 어셈블리 코드를 전개하였으며, e^b 의 근사화를 위해 Taylor-Maclaurin 확장을 20차까지 전개시켰다. e^a 의 연산을 위해 LMD 명령어를 사용하여 해당되는 값을 참조하였으며, 이 둘의 값을 곱하여 data memory에

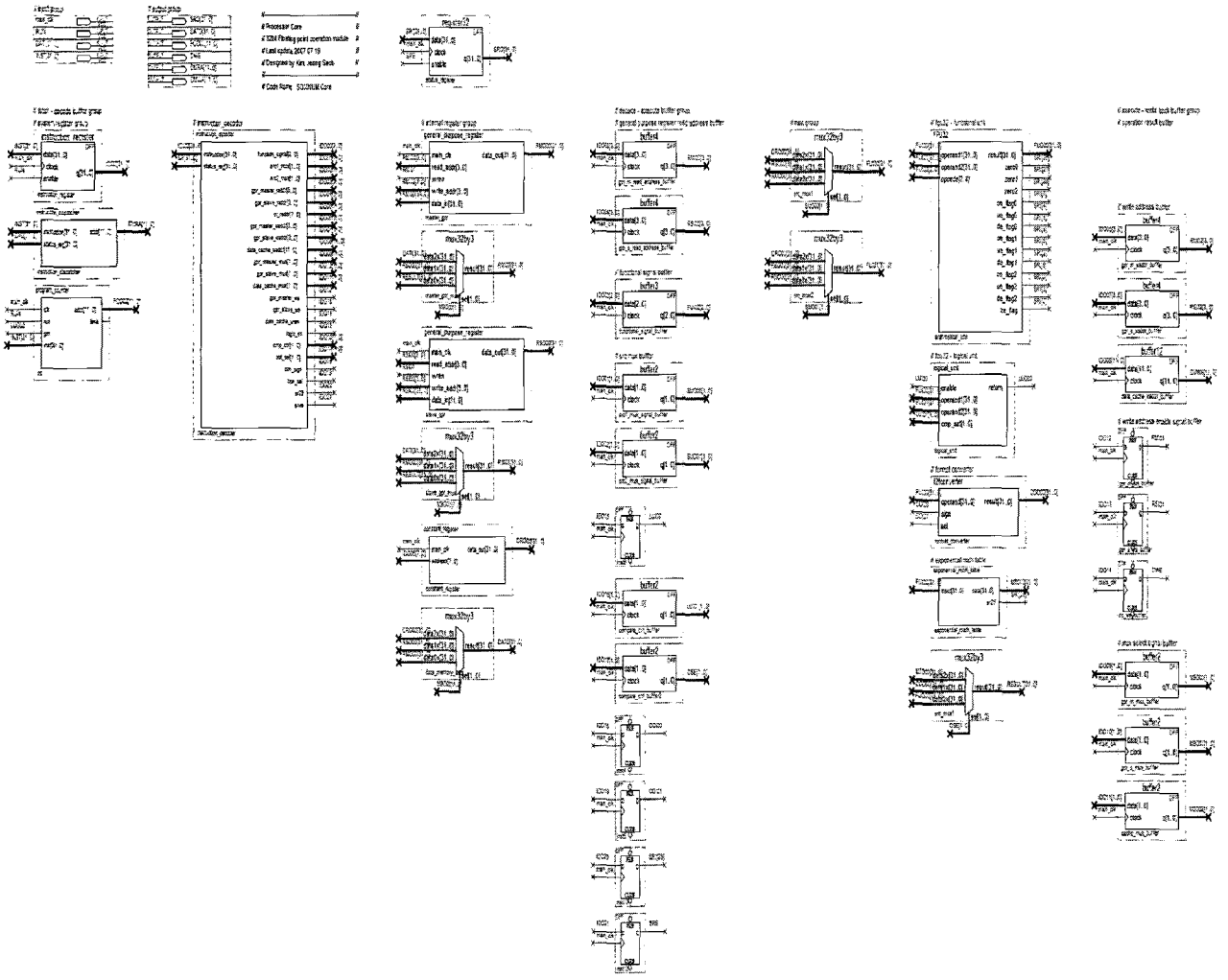


그림 7. 코어모듈의 전체 구조.
Fig. 7. Overall architecture of core module.

```
ZERO   mr0
MOV    sr15    mr0
NOP
MOV    sr0     mr0
SPINT  mr0
INTSP  mr0
SUB    mr1     sr0     mr0
DIV    sr2     mr1     cr21
ADD    sr2     sr2     cr2
DIV    sr2     sr2     cr20
 :      :       :       :
MUL    sr2     sr2     mr1
ADD    sr2     sr2     cr2
DIV    sr2     sr2     cr3
MUL    sr2     sr2     mr1
ADD    sr2     sr2     cr2
MUL    sr2     sr2     mr1
ADD    mr1     sr2     cr2
MOV    sr1     mr1
SPINT  mr0
LMD    mr0
MUL    mr0     mr0     sr1
ST     mem891  mr0
NOP
NOP
```

Taylor-Maclaurin Expansion (N=20)

그림 8. 지수 근사화 알고리즘의 코드.

Fig. 8. Code implementation of exponential approximation algorithm.

저장하게 된다.

입력 변수 x 에 0부터 88.9까지 0.1씩 증가시켜 총 890번의 입력을 주고 하드웨어로 구현된 알고리즘을 테스트하였다. Exponential math table에 저장되어 있는 값은 단정도 기반에서 근사화하여 저장한 값이며 제한된 유효숫자 자릿수를 가진다. Single precision에서 1ulp(unit in last place) 이내의 오차를 가지기 때문에 실제 값에 비해 $(1/2^{23})\times 100\,\%$ 이내의 오차를 가진다.

새로운 지수함수 근사화 알고리즘의 성능을 확인하기 위해 기존에 사용하던 근사화 방법인 단순한 형태의 Taylor-Macluarin 확장을 50차까지 전개시켰으며, 이를 같은 방법으로 코드화시켰다.

V. Experiment

실험은 Altera의 Stratix II 디바이스를 사용한 DSP Development Kit를 사용하였으며 Quartus II를 사용하여 VHDL로 작성한 코드를 컴파일 하여 합성시켰다.

그림 10는 지수함수 근사화 알고리즘의 성능을 테스트하기 위한 목적으로 SOMNIUM 프로세서 유닛을 탑재한 시스템의 Stratix II에서의 합성된 결과를 나타내고 있다.

실험은 그림 11에 나타난 것처럼 입력 변수인 x 값을 0으로 초기화 시키고 이를 본 논문에서 제안하는 방식의 알고리즘을 통해 지수함수를 근사화하고 역시 기존에 사용했던 알고리즘인 단순한 형태의 Taylor-Maclaurin 확장으로 근사화시킨다. 연산이 완료되었으면 일정 시간 동안 연산을 중지하고 status register를 변화시켜 외부의 UART 모듈에서 연산이 완료되었음을 확인하고 연산 결과를 PC에 전송한다. 전송이 완료되면 입력 값 x 를 0.1만큼 증가시켜 다시 같은 패턴으로 연산을 수행하면서 연산 결과를 전송한다. x 값이 89이상이 되면 이는 지수함수 연산 결과가 단정도 부동소수점 표현범위를 넘어선다고 판단하여 연산을 종료하게 된다.

그림 11은 전체적인 실험 내용의 흐름을 나타내고 있다. 지수함수의 연산은 그 결과가 입력 변수의 선형적 증가에 따라 지수함수적으로 증가하기 때문에 적당한 구간으로 나누어 근사화가 잘 이루어지고 있는지 확인하였다.

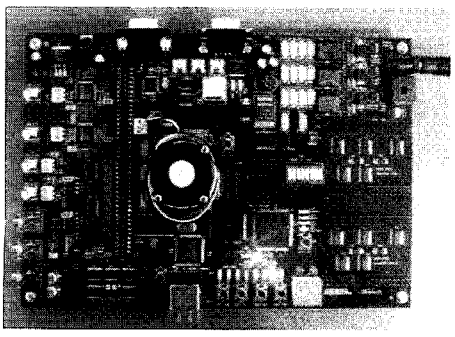
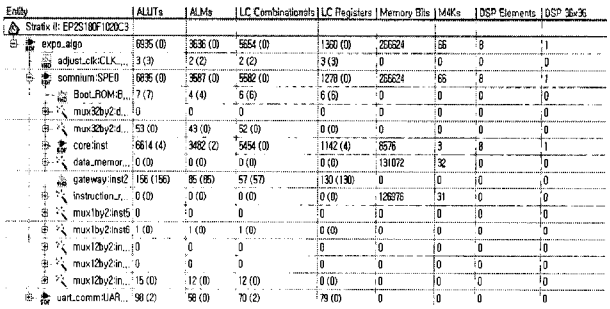


그림 9. Altera DSP KIT W/STRATIX II EP2S180N 보드.

Fig. 9. Altera DSP KIT W/STRATIX II EP2S180N board.



Entity	ALUTs	ALMs	LC Combinationals	LC Registers	Memory Bits	M4Ks	DSP Elements	DSP 36x36
Stratix II: EP2S180F1020C3								
expo_algo	6935 (0)	3636 (0)	5654 (0)	1360 (0)	266624	66	8	1
adjust_clk:CLK_...	3 (3)	2 (2)	2 (2)	3 (3)	0	0	0	0
somnium:SPE0	6896 (0)	3587 (0)	5582 (0)	1278 (0)	266624	66	8	1
Boot_ROM:B...	7 (7)	4 (4)	6 (6)	6 (6)	0	0	0	0
mux32by2:d...	0	0	0	0	0	0	0	0
mux32by2:d...	53 (0)	43 (0)	52 (0)	0 (0)	0	0	0	0
core1inst	6814 (4)	3492 (2)	5454 (0)	1142 (4)	8576	3	8	1
data_memor...	0 (0)	0 (0)	0 (0)	0 (0)	131072	32	0	0
gateway:inst2	156 (156)	85 (85)	57 (57)	130 (130)	0	0	0	0
instruction_r...	0 (0)	0 (0)	0 (0)	0 (0)	126976	31	0	0
mux1by2:inst5	0	0	0	0	0	0	0	0
mux1by2:inst6	1 (0)	1 (0)	1 (0)	0 (0)	0	0	0	0
mux12by2:in...	0	0	0	0	0	0	0	0
mux12by2:in...	0	0	0	0	0	0	0	0
mux12by2:in...	15 (0)	12 (0)	12 (0)	0 (0)	0	0	0	0
uart_comm:UAR...	98 (2)	58 (0)	70 (2)	79 (0)	0	0	0	0

그림 10. 합성의 결과.

Fig. 10. Result of synthesis.

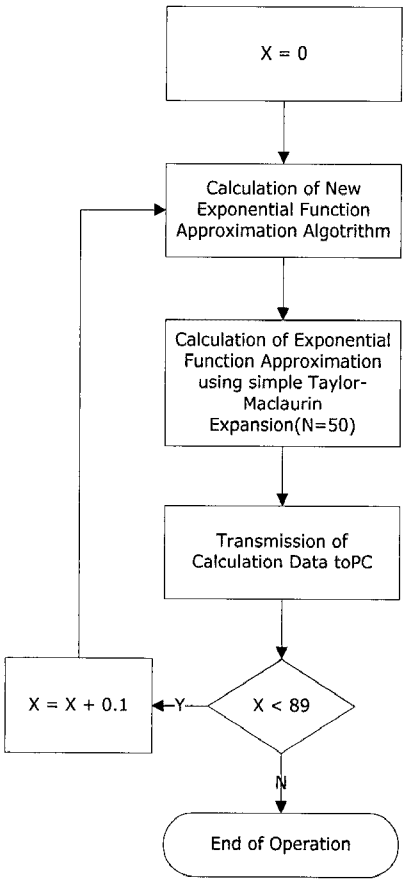


그림 11. 실험 순서.

Fig. 11. Experiment flow.

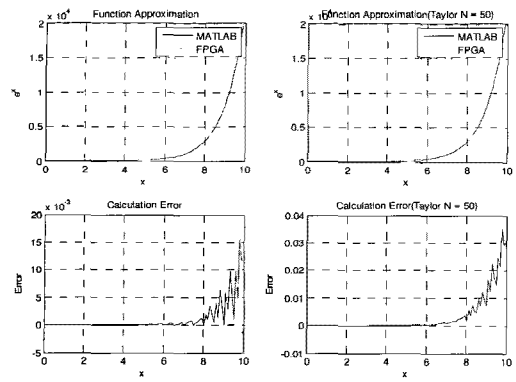


그림 12. x = 0~9.9에서 지수 알고리즘의 결과.

Fig. 12. Result of exponential function approximation at x = 0~9.9.

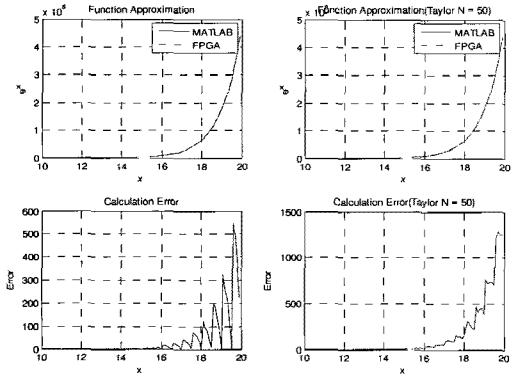


그림 13. x = 10~19.9에서 지수 알고리즘의 결과.

Fig. 13. Result of exponential function approximation at x = 10~19.9.

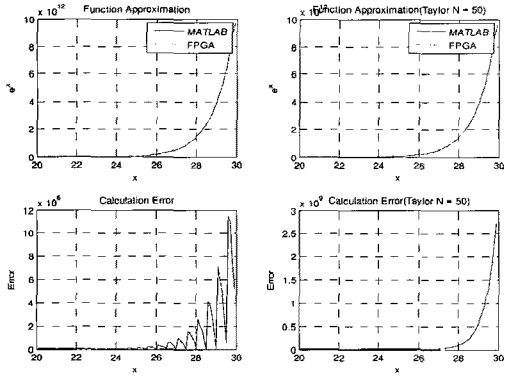


그림 14. x = 20~29.9에서 지수 알고리즘의 결과.

Fig. 14. Result of exponential function approximation at x = 20~29.9.

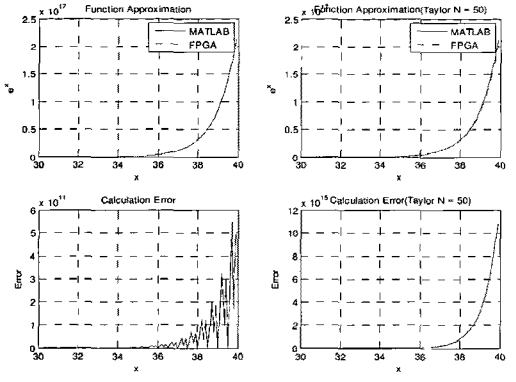


그림 15. x = 30~39.9에서 지수 알고리즘의 결과.

Fig. 15. Result of exponential function approximation at x = 30~39.9.

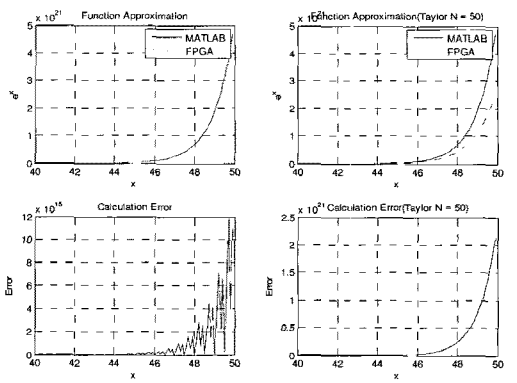


그림 16. x = 40~49.9에서 지수 알고리즘의 결과.

Fig. 16. Result of exponential function approximation at x = 40~49.9.

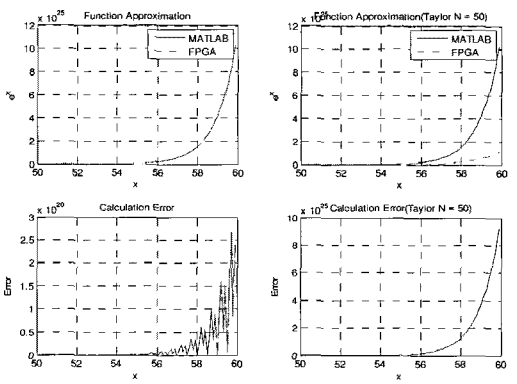


그림 17. x = 50~59.9에서 지수 알고리즘의 결과.

Fig. 17. Result of exponential function approximation at x = 50~59.9.

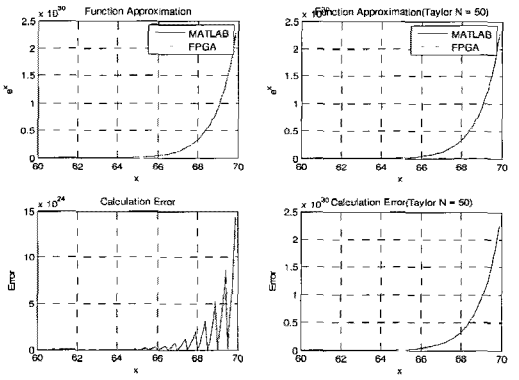


그림 18. x = 60~69.9에서 지수 알고리즘의 결과.

Fig. 18. Result of exponential function approximation at x = 60~69.9.

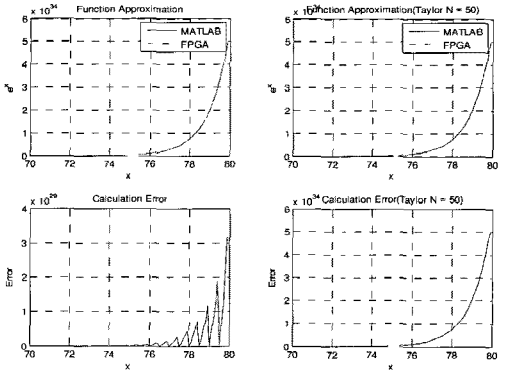


그림 19. x = 70~79.9에서 지수 알고리즘의 결과.

Fig. 19. Result of exponential function approximation at x = 70~79.9.

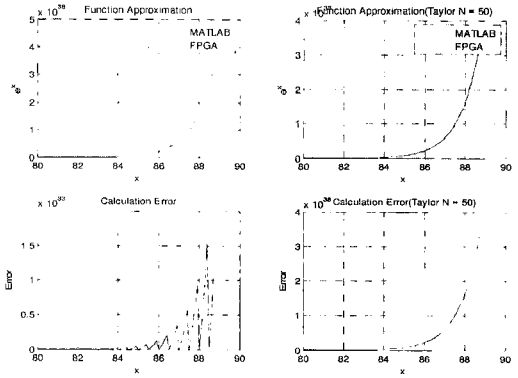


그림 20. x=80~88.9에서 지수 알고리즘의 결과.

Fig. 20. Result of exponential function approximation at x=80~88.9.

그림 12~20은 새로운 지수함수 근사화 알고리즘의 연산 결과와 기존의 Taylor-Maclaurin 확장의 50차 전개에 대한 근사화 결과를 구간별로 비교하고 있다. PC기반의 MATLAB에서 지수함수 연산 결과를 기준으로 FPGA에서의 연산에 얼마만큼의 오차가 있는지 비교하였다. 입력 변수의 범위가 10 이내일 경우 새로운 근사화 알고리즘과 기존의 근사화 알고리즘의 차이는 그다지 크지 않지만 입력 변수의 크기가 증가할 수록 기존의 근사화 알고리즘의 오차는 점차 커져서 결국에는 오차비율이 최대 99.99945%에 이르고 있는 것을 알 수 있다. 이에 반해 새로운 알고리즘의 경우 최대 약 0.00064% 이내의 오차를 보였다. 따라서 기존의 단순한 형태의 50차 전개의 Taylor-Maclaurin 확장 근사화 방법보다 좋은 성능을 확인할 수 있다.

VI. Conclusion

본 논문은 선행 연구를 통해 설계된 부동소수점 기반의 프로세서 유닛을 이용하여 보다 효율적인 지수 함수 연산 알고리즘을 개발 및 하드웨어를 구현하였으며, 실험을 통해 성능을 검증하였다. PC 기반에서의 연산과 비교했을 때, 약간의 오차는 존재했지만 이는 지능형 로봇의 제어용으로 설계된 단정도 기반 프로세서 연산에 있어 허용할 수 있는 범위라고 할 수 있다.

참고문헌

[1] S. Jung and S. S. Kim, "Hardware implementation of real-time neural network controller with a DSP and an FPGA for nonlinear system," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 1, pp. 265-271, 2007.

[2] H. HU, T. Jin, X. Zhang, Z. LU, and Z. Qian, "A floating-point coprocessor configured by a fpga in a digital platform based on fixed-point DSP for power electronics," *International Power Electronics and Motion Control Conference*. 2006.

[3] J.-S. Kim, H.-W. Jeon, and S. Jung, "Embedded hardware Implementation of an FPGA based nonlinear PID controller for the ROBOKER arm," *Journal of Control, Automation, and Systems Engineering*, vol. 13, no. 12, pp. 1153-1159, Dec. 2007.

[4] K. Basterretxea, J. M. Tarela, Ines del Campo, and G. Bosque, "An experimental study on nonlinear function computation for neural/fuzzy hardware design," *IEEE Trans. on Neural Networks*, vol. 18, no. 1, pp. 266-283, 2007.

[5] L.-K. Wang and M. J. Schulte, "Decimal floating-point square root using Newton-raphson iteration," *Proc. of the 16th International Conference on ASAP'05*. 2005.

[6] J.-S. Kim and S. Jung, "Implementation of neural network hardware based on a floating point operation in an FPGA," *ICMIT*, pp. 679451-6, 2007.

[7] J.-S. Kim and S. Jung, "Design of floating point processor for nonlinear functions based on FPGA," *Conference on Information and Control Systems*, Korea, pp. 74-76, 2007.

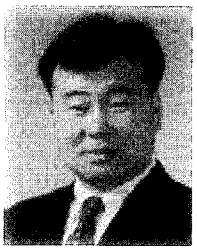
[8] J.-S. Kim and S. Jung, "Implementation of the RBF neural chip with the on-line learning back-propagation," *IEEE WCCI*, pp. 378-384, 2008.

[9] J.-S. Kim and S. Jung, "Hardware implementation of a neural network controller on FPGA for a humanoid robot arm," *IEEE AIM*, pp. 1164-1169, 2008.

[10] 김정섭, 정슬, "비선형 함수 연산을 위한 FPGA 기반의 부동 소수점 프로세서의 설계," 대한 임베디드공학논문지, 3권 4호, pp. 251-259, 2008.

김 정 섭

2006년 충남대학교 메카트로닉스공학과 졸업. 2009년 충남대학교 대학원 메카트로닉스공학과 지능로봇시스템 졸업. 현재 도담 시스템스 연구소 근무. 관심분야는 지능 시스템의 SoC 구현, 영상처리 및 인식 시스템.



정 슬

1988년 미국 웨인 주립대 전기 및 컴퓨터 공학과 졸업. 1991년 미국 캘리포니아대 데이비스 전기 및 컴퓨터 공학과 석사. 동 대학 박사. 1997년~현재 충남대학교 메카트로닉스공학과 교수. 관심분야는 지능 제어 및 지능 로봇 시스템, 임베디드 제어기 설계, 로봇과 인간의 상호작용, 가정용 로봇 응용.