특 집

차세대 Embedded 마이크로프로세서 기술 동향

이 희
(주)에이디칩스

요 약

1970년대에 개발된 마이크로 프로세서는 제어 기기 분야 및 소형 컴퓨터에서 주로 사용되어 오다가 1980년대에 이르러 RISC(Reduced Instruction Set Computer) 구조의 도입으로 중대형 컴퓨터에 이르기까지 광범위하게 사용되고 있다. 또한 반도체 기술의 급격한 발전으로 슈퍼스칼라 구조가 마이크로 프로세서에도 적용되고 있으며 동작 속도도 수백 MHz에 이르고 있다.

마이크로 프로세서는 프로그램을 수행하기 위해서 프로그램과 데이터를 메모리로부터 읽어 와야 한다. 그런데 메모리 용량은 빠른 속도로 증가하고 있지만 동작 속도는 마이크로 프로세서의 동작 속도에 크게 미치지 못하고 있다. 1980년에 DRAM의 접근 속도는 250nsec이었으나 1998년에 RDRAM의 동작속도는 300MHz로 70여 배 빨라졌다. 그러나 마이크로프로세서는 1980년에 8086의 동작 속도가 8MHz이던 것이 1998년에는 팬티엄-2가 500MHz에 이르고 있다. 더욱이 팬티엄-2는 슈퍼스칼라 구조이므로 이를 감안하면 1GHz 이상에 이르러 120여 배 빨라진 것을 알 수 있다. 이와 같은 메모리 속도와 마이크로 프로세서 속도 차이에 더하여, 메모리와 마이크로 프로세서를 인쇄 회로 기판에서 연결하는데 따른 물리적 특성은 변화하지 않으므로 데이터 전송 폭을 넓히는 것에는 한계가 있다.

따라서 향후 컴퓨터 성능 발달을 제한하는 주요 요소 중 하나는 마이크로 프로세서와 메모리 사이의 데이터 전송 폭이다. 프로그램과 데이터가 메모리에 저장되는 본 뉴먼 방식의 컴퓨터에서 데이터 전송 폭을 줄이기 위해서는 코드 밀도(Code Density)가 높은 컴퓨터 구조를 연구하는 것이 필요하다.

한편 마이크로 프로세서는 실장 제어용으로 거의 모든 전자 제품 및 자동화 기기에서 채용하고 있다. 특히 냉장고, 에어콘, 전축, TV, 세탁기 등 가전기기와 Fax, 복사기, 프린터 등 사무용기기와 자동차, 선박, 자동화기계 등 사무 및 산업용 기기와 PDA(휴대용 정보 기기), NC(Network Computer) 등 정보 기기 그리고 각종 오락기, 노래 반주기 등 정보 기기 등에서 사용하는 실장 제어용 마이크로 프로세서 시장은 매년 10% 이상씩 성장하고 있으며, 21세기 산업을 주도하는 핵심 기술로 자리 매김하고 있다.

이러한 실장 제어용 기기는 마이크로 프로세서와 메모리 및 입출력 장치가 하나의 반도체에 집적되는 경우가 많다. 그런데 반도체 가격은 반도체 크기에 따라 결정되며, 가장 넓은 면적을 차지하는 것은 메모리이다. 따라서 반도체 가격을 낮추기 위해서는 메모리 크기를 줄여야 하며, 이를 위해서 또한 코드 밀도가 높은 컴퓨터 구조에 대한 연구가 필요하다.

최근에는 32비트 RISC 명령어를 16비트 명령어로 축약한 구조가 연구되었다. ARM-7TDMI는 ARM-7의 16비트 축약 명령어 구조이며, TR4101은 MIPS-R3000의 16비트 축약 명령어

구조이다. 이들 16비트 축약 명령어 RISC는 종래 RISC와의 호환성을 위하여 2가지 모드로 동작하므로 구조가 복잡하고, 16비트 명령어에서는 8개의 레지스타만을 접근할 수 있으므로 성능이 크게 떨어지는 단점을 가진다.

I. 차세대 컴퓨터 아키텍쳐 요구사항

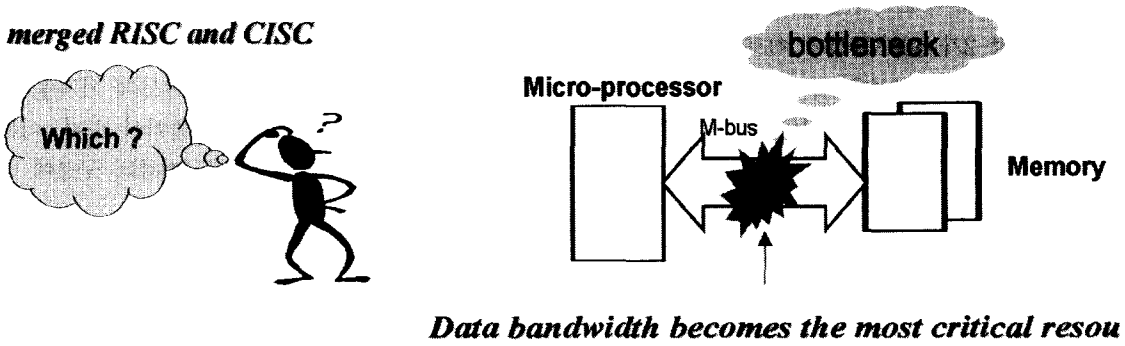
반도체 기술의 급속한 발전으로 1GHz CPU가 본격적으로 시장에 출시되고 있다. 예를 들어 Superscalar MIPS-R4000의 평균적인 병렬처리 명령어의 수는 2.5개로 보고되고 있다. 그러므로 1GHz superscalar MIPS-R4000은 명령어 수행을 위하여 평균 10GByte/sec의 memory bandwidth가 필요하다. 한편 MIPS-R4000에서 load/store 명령어가 차지하는 빈도가 30%이며, 이를 감안하면 13GByte/sec의 memory bandwidth가 필요하다. L1 cache hit ratio를 80%로 가정하면 2.6GByte/sec의 off chip memory bandwidth가 필요하다. 따라서 64 bit data bus, 50% bus utilization을 가정하면 650MHz bus clock이 필요하다.

이러한 고속 버스는 PCB(Printed Circuit Board)의 제작을 어렵게 하며, CPU clock이 높아질수록 이러한 문제는 더욱 심각해지게 된다. 이에 더하여 Sync DRAM, RAMBUS DRAM 등 high speed burst transfer 기능을 가지는 memory 발전 속도는 CPU clock 발전 속도에 항상 미치지 못하므로 차세대 컴퓨터에서 성능을 저해하는 중요한 요소 중의 하나가 memory bandwidth bottle neck이 되고 있다.

Memory bandwidth bottle neck을 해결하기 위해서는 code density가 높은 명령어 구조를 개발하는 것이 중요하다. Code density가 높으면 program 길이가 짧아지므로 data bus traffic이 줄어 들어 memory bandwidth를 보다 효율적으로 사용할 수 있다.

한편 microprocessor는 종래 복잡한 hardware를 가지는 CISC에서 간단한 hardware를 가지는 RISC로 발전하고 있다. RISC는 hardware가 간단하므로 동작 clock을 높일 수 있고, 따라서 performance를 크게 높일 수 있지만 CISC에 비하여 code density가 낮으므로 memory bandwidth bottle neck에 더욱 취약하다.

따라서 차세대 컴퓨터는 RISC처럼 간단한 hardware를 가지며 CISC보다 높은 code density를 가지는 새로운 형태의 architecture가 요구된다.



〈그림 1〉 차세대 컴퓨터 아키텍쳐의 critical resource

II. 임베디드 마이크로프로세서 요구사항

실장제어용 microprocessor에서 필요로 하는 기능들 을 나열하면 다음과 같다.

1. Simple hardware

실장제어용 microprocessor는 30-50 MIPS 이하의 성능을 가지는 것이 전체 시장의 대부분을 점유하고 있다. 이들 low end 시장에서는 microprocessor와 memory, IO 등을 하나의 ASIC에 집적하는 것이 중요하다. 또한 FPGA의 급속한 발전으로 집적도가 높아지면서 microprocessor를 FPGA로 구현하여, program 개발 및 hardware 개발을 용이하게 하는 것이 요구되고 있다.

따라서 이들 시장의 요구를 충족시켜 주기 위해서는 RISC처럼 간단한 hardware로 구현 가능한 microprocessor가 필요하다.

2. Efficient high level language

제품의 개발주기 및 생명주기가 짧아지면서 새로운 제품 개발에 소요되는 기간을 줄여야 한다. 종래 8/16 bit microprocessor는 많은 부분을 assembler를 사용하여 개발하였는데, 이러한 환경은 향후 실장제어 시장에 적합하지 않다.

한편 80년대 중반까지만 해도 많이 사용되었던 PASCAL, PL/1, APL 등 언어는 향후 사용빈도가 극히 적을 것으로 판단된다. 종래 microprocessor 명령어 구조를 선정할 때 이들 언어에 적합성을 많이 고려하였지만, 향후 이러한 고려는 불필요하다.

그러므로 향후 주도적으로 사용될 것이 확실시되는 C/C++/Java로 대변되는 고급언어에 적합한 구조를 가져야 한다.

3. High code density

향후 실장제어기기는 하나의 반도체에 microprocessor, RAM, ROM, IO가 모두 집적되는 구조로 된다. 그런데 이중에서 program을 저장하는 ROM이 점유하는 silicon 면적이 제일 크다. 따라서 반도체 가격을 낮추려면 silicon 면적을 줄여야 하며, 이를 위해서는 가장 많은 면적을 차지하는 program ROM의 크기를 줄여야 하고, 그러기 위해서는 code density가 높은 microprocessor의 개발이 필수적이다.

4. 16 bit microprocessor will have the largest volume market

현재 물량면에서 가장 큰 시장을 점유하는 것이 8 bit microprocessor이다. 그리고 16 bit microprocessor는 몇 종류가 시장에 나와 있지만 이들은 모두 CISC 구조를 가지고 있으므로 hardware가 복잡하여, IP 형태로 제공되지 않으므로 ASIC에서 사용하는 것이 어렵다. 또한 CISC 구조이므로 C/C++/Java 등 고급언어 지원 기능이 미약하다. 따라서 8 bit microprocessor보다 높은 성능을 얻으려면 부득이 32 bit microprocessor를 사용해야 하는 불편함이 있다.

따라서 RISC처럼 간단한 hardware 구조를 가져서 IP 제공이 가능하며, C/C++/Java 등 고급언어 지원 기능이 좋은 16 bit microprocessor의 개발이 필요하다. 이러한 16 bit microprocessor가 개발되면 물량적으로 가장 큰 시장을 형성할 것으로 전망된다.

5. 32 bit microprocessor

16 bit microprocessor는 address 영역에 제약이 있다. 따라서 대규모 program이나 data를 요구하는 분야에서는 32 bit microprocessor의 사용이 필요하다.

현재 실장제어 시장에서 많이 사용되는 32 bit microprocessor는 CISC 계열로 Motorola의 MC680x0이 있으며 RISC 계열로는 MIPS/IDT사의 MIPS-R3000/R4000 계열이 있다. 또한 RISC와 CISC의 중간 형태로 ARM사의 ARM-7/8, Hitachi사의 SH series, NEC사의 V8x0 등이 있다.

이들 32 bit microprocessor는 RISC 계열

은 hardware가 간단하지만 code density가 낮아서 program 크기가 증가하며 따라서 system cost를 높이는 문제가 있으며, CISC 계열은 IP가 제공되지 않으므로 ASIC화가 사실상 불가능하다.

최근에는 ARM-TDMI(Thumb)과 MIPS-16이 개발되어서 많이 보급되고 있다. 이들은 16 bit code 체계를 가지므로 code density가 높아지는 장점이 있지만, 완전한 명령어 체계를 가지고 있지 않아 ARM-7/8, MIPS-R3000/R4000 core를 가지고 있어야 하므로 hardware가 필요 이상으로 복잡해지며, 16 bit code상에서는 8개의 레지스타만을 사용할 수 있으므로 load/store 빈도가 증가하여 performance가 저하되는 단점이 있다.

따라서 RISC처럼 hardware가 간단하고 CISC처럼 code density가 높고 풍부한 레지스타를 사용하여 load/store 빈도를 RISC 수준으로 낮추는 새로운 개념의 microprocessor에 대한 연구가 필요하다.

6. 64 bit microprocessor

Microprocessor는 개발하는 데 수년이 소요되고 시장 형성에 또 다시 수년이 걸리는 기술분야이다. 따라서 향후 3-5년 후부터 10-15년 후를 내다보고 연구, 개발하는 것이 필요하다.

현재 DRAM의 집적도와 HDD 등 보조기억장치의 용량 등 발전 속도와 급속한 multimedia 환경으로의 변화는 32 bit microprocessor가 가지는 4GByte address 범위를 짧은 기간내에 초과할 것으로 전망된다. 이러한 변화에 맞추어서 64 bit microprocessor에 대한 연구, 개발이 요구된다.

그러나 종래 CISC는 명령어 구조가 너무 복잡하여 64 bit microprocessor 구현이 사실상 어렵다. 따라서 현재 CISC 64 bit microprocessor는 연구되지 않으며, 현재 64 bit microprocessor는 모두 RISC 기반이다. 그런데 RISC에서는 offset 및 immediate data field는 16 bit로 제한되므로 64 bit microprocessor에 적합하지 않다.

따라서 RISC처럼 간단한 hardware를 가지면서 64 bit addressing에 적합한 새로운 개념의 64 bit microprocessor에 대한 연구, 개발이 요구된다.

Ⅲ. CISC(Complex Instruction Set Computer)와 RISC(Reduced Instruction Set Computer)

오늘날 microprocessor architecture는 RISC와 CISC 그리고 이들을 혼합한 형태로 크게 구분할 수 있다. 이들 architecture는 명령어 구성에서 큰 차이를 보이고 있다. 모든 명령어는 작용을 나타내는 Op-Code와 작용을 받는 Operand로 구분된다. 예를 들면

```
ADD  Rs, immd, Rd ; Rd=Rs+immd
```

라는 명령어에서 작용을 나타내는 Op-Code는 'ADD'이고, Operand는 'Rs', 'Immd', 'Rd'가 된다. 이중에서 'Rs'와 'Rd'는 register를 지정하는 것으로 사용 가능한 register 수에 따라서 고정길이이다. 이에 반하여 immd는 immediate data 가변길이 Operand로 architecture에 따라서 표현 방법이 다르다. 이러한 가변길이 Operand는 immediate data 이외에도 load/store 명령문에서 offset, branch 명령문에서 offset 등이 있다.

CISC는 가변길이 Operand에 따라서 여러 가지의 Op-Code를 가진다. 즉,

```
ADD.B  %R0,  #N
ADD.W  %R0,  #NN
ADD.L  %R0,  #NNNN
```

과 같이 된다. 이러한 CISC는 가변길이 Operand를 필요한 길이 만큼만 표현하므로 code density가 높아지는 장점을 가지나 많은 종류의 Op-Code를 가져야 하므로 hardware가 복잡해지고

따라서 CPU clock이 낮아지므로 performance 또한 저하되는 단점이 있다.

이에 반하여 RISC는 모든 명령어의 길이를 32 bit로 통일시키고 일정 길이의 Operand만을 가지는 구조이다. 대부분의 RISC에서는 16 bit 길이 Operand만을 가지고 있다. 만일 이보다 긴 길이의 Operand를 필요로 하면 여러 개의 명령어를 사용해서 조합해야 한다. 따라서 RISC는 code density가 낮아지는 문제점을 가지게 된다.

한편 RISC와 CISC를 혼합한 형태의 micro-processor는 가변길이 Operand를 표현하는 방법에 따라서 크게 2가지로 구분할 수 있다. 첫째 ARM-7/8, SH-3, M-Core, ARM-7/8TDMI (Thumb) 등은 PC relative immediate data load 명령어를 가지고 있다. 즉,

```
        LDI   Label, Ri
                    ; Label=PC+offset
        ADD   Rs, Ri, Rd
              ......
Label : INT   immd
```

의 형태를 취한다. ARM-7/8은 32 bit 고정길이 명령어이고 SH-3, M-Core, ARM-7/8 TDMI는 16 bit 고정길이 명령어이다. 이러한 PC relative immediate data load 명령어의 사용은 IBM-360에서부터 사용되기 시작한 방식으로 고정길이 명령어 구조를 이룰 수는 있지만, load/store의 빈도가 증가하며, cache 성능을 저하시키는 등 문제점이 있다.

또 다른 방법은 V800과 MN10300 등에서 사용하는 방식으로 가변길이 명령어 방식이다. V800은 16/32 bit 2가지 길이 명령어만을 가지고 있으며, MN10300은 8/16/24/32/40/48/56 bit의 가변길이 명령어이다. 이러한 가변길이 명령어는 CISC에 보다 가까운 구조로 hardware가 복잡해지는 등 CISC의 문제점을 그대로 가지고 있다.

IV. 차세대 CISC/RISC merged architecture

다음과 같은 CISC와 RISC의 장점만을 취합한 새로운 architecture에 대한 연구가 필요합니다.

1. High code density

차세대 컴퓨터와 실장제어용에서 모두 high code density가 가장 중요한 요구 사항입니다. 따라서 code density가 CISC 보다 높은 구조가 필요합니다.

2. 16/32/64 bit microprocessor have same architecture(Scalable Architecture)

CISC는 32/64 bit 계열에서 명령어의 수가 급격하게 늘어나므로 비효율적이 됩니다. 반면에 RISC는 32 bit에서는 적합하지만 16/64 bit 계열에서 비효율적입니다. Micro-processor를 요구하는 분야는 대단히 광범위하므로 16/32/64 bit 계열이 모두 필요하며, 따라서 16/32/64 bit 계열이 모두 동일한 architecture를 가진다면 연구, 개발이 용이해지는 등 많은 장점을 가지게 됩니다.

3. High level language support (고급 언어에 적합)

RISC는 대부분 32개의 범용 레지스타를 가지고 있다. 이것은 RISC 연구가 진행되던 1980년대 초에 주로 사용하였던 고급 언어가 C/PASCAL/APL 등이었는데, PASCAL과 APL은 nested procedure를 허용하는 언어이다. 따라서 RISC는 nested procedure를 지원하기에 적합한 구조를 가지게 되었다. 그러나 현재는 C/C++/Java가 주로 사용되며 이들 언어는 nested procedure를 허용하지 않는다. 따라서 RISC 연구 당시와는 다른 환경을 가지게 되었다.

고급언어에 적합한 아키텍처 개발이 필요하다.

4. Simple instruction set, simple hard ware

FPGA/ASIC procesor에 적합할 정도의 간단한 hardware를 가져야 하며, 또한 high performance 구현이 가능해야 합니다.

16 bit fixed length instruction set를 정의하며, 한 가지 Op-Code에 한 개의 instruction만을 가지고 있으므로 instruction의 수가 적을 수 있으며, 즉, Operand length에 따른 여러 개의 instruction을 만들 필요가 없으므로 instruction 수가 기존 microprocessor 보다 적어진다. 따라서 hardware가 간단하고, operating frequency가 높아지므로 performance 높일 수 있다.

5. low power Architecture

Microprocessor의 power consumption은 implementation과 realization에 주로 의존한다. 예를 들면 pentium chip이 desk top용이 있고 note book 용이 있는데 이들은 동일한 architecture이다. 이와 같이 microprocessor의 전력 소모는 implementation과 realization에 주로 영향을 받는다.

한편 architecture level에서도 low power consumption에 대한 고려가 필요하다.

CMOS에서 전력 소모는 logic status가 변화하여서 발생한다. 즉, CMOS에서 전력 소모 Pd는 $Pd=C\times V2\times F$로 주어진다. C는 CMOS 제작 공정 중에서 발생하는 capacitance이고, V는 공급 전압, 그리고 F는 동작 주파수이다.

여기서 C와 V를 낮추는 것은 implementation과 realization에 직접적으로 관련되어 있으며 architecture와는 무관하다. F는 동작 주파수인데 CMOS 출력의 변화 주파수이다.

따라서 architecture에서 전력 소모를 낮추기 위해서는 첫째로 전력을 소모하는 logic gate count를 줄이도록 하여야 하는데 이를 위해서는 hardware가 간단하게 되는 구조가 되어야 한다. 둘째로는 logic gate 입출력의 상태가 자주 변화하지 않도록 하여야 하는데, 이를 위해서는 data bus traffic을 낮추어야 한다. 따라서 program size가 작은 architecture 즉 high code density architecture가 요구된다.

6. High Performance Architecture

16 bit fixed length instruction set를 정의하며, 한 가지 Op-Code에 한 개의 instruction만을 가지고 있으므로 instruction의 수가 적을 수 있으며, 즉, Operand length에 따른 여러 개의 instruction을 만들 필요가 없으므로 instruction 수가 기존 microprocessor 보다 적어진다. 따라서 hardware가 간단하고, operating frequency가 높아지므로 performance 높일 수 있다.

V. 결 론

마이크로프로세서는 실장 제어 기기 분야에서 중대 형 및 소형 컴퓨터에 이르기까지 광범위하게 사용되고 있으며, 반도체 기술의 급격한 발전으로 동작 속도도 2GHz에 조만간 이르게 될 전망이다. 이들 마이크로프로세서에서 프로그램을 수행하기 위해서는 프로그램과 데이터를 메모리로부터 읽어 와야 한다. 그런데 메모리 동작 속도는 마이크로 프로세서 동작 속도에 크게 미치 지 못하기 때문에 마이크로 프로세서와 메모리사이의 데이터 전송 폭이 시스템 성능을 제한하는 중요한 요인으로 작용하고 있다.

또한 실장 제어용 기기는 마이크로 프로세서와 메모리 및 입출력 장치가 하나의 반도체에 집적되는 경우가 많다. 반도체 가격을 낮추기 위해서는 메모리 크기를 줄여야 하며, 이를 위해서 또한 프로그램 크기가 작은 컴퓨터 구조에 대한 연구가 필요하다.

저 자 소 개

李 熙

1958년 7월 1일생. 1977~1981 인하대학교 전자공학과 졸업. 1982~1984 인하대학교 대학원 졸업(전자 전공), 1983. 10~98. 10 : 삼성전자 반도체 연구소 수석 연구원, 1998. 10~현재 : (주)에이디칩스 연구소장, 〈주관심 분야 : embedded MCU (16/32/64 bit 마이크로프로세서), Videp/Graphic (2D/3D rendering), network (VoIP, VoDSL...)〉