TRON 프로젝트중 마이크로 프로세서

최 윤 석
(서울대 대학원 전자계산기공학과 석사과정)

1. 서 론

일본의 독자적인 운영 체제 개발 계획인 TRON(The Real time Operating system Necleus) 프로젝트는 지금껏 미국에 의존해 온 일본의 컴퓨터 업계가 독자적인 운영 체제의 개발과 아울러 유저가 퍼스컴을 기종에 관계없이 통일 규격에 의해 쉽게 사용할 수 있는 90년대에 요구되는 새로운 컴퓨터를 탄생시키기 위한 계획이다.

TRON 프로젝트는 크게 5가지의 보조 프로젝트로 그중 4개는 여러 응용에 적합한 운영 체제의 개발을 위한 것이고 나머지는 이러한 운영 체제가 효과적으로 동작할 수 있는 마이크로 프로세서의 개발을 위한 것이다.[1]

본고에서는 TRON 프로젝트중 마이크로 프로세서의 개발 계획에 관하여 소개한다.

2. TRON 프로젝트의 동기와 구성

메모리 집적도의 향상은 퍼스널 컴퓨터에서도 수 메가 바이트 이상의 메모리를 장착할 수 있게 되었다. 그러나 기존의 CPU 구조상으로는 이런 넓은 메모리 공간을 충분히 이용할 수 없다. 예를 들어 1024x1024 bit map의 화상을 처리하는 데는 128K 바이트의 메모리가 필요하나 기존의 몇몇 CPU로는 64K 바이트 이상의 데이타를 처리하기 힘들다. 그 이유는 CPU의 설계자들이 호환성만을 고려하여 기존의 CPU에 새로운 특징을 부가시키는 방향으로 설계를 하기 때문이다.

시스템 설계자의 입장에서 새로운 워크스테이션을 설계할 때는 통상 UNIX를 염두에 둔다. 물론 UNIX는 많은 응용에서 좋은 결과를 보이고 있으나 프로그래머가 아닌 일반 유저들이 사용하기에는 어렵다는 단점이 있다. 이런 문제는 각각의 사용자 쉘에 UNIX 사용 환경을 제공함으로써 해결하려고 한다. 그러나 다양한 사용자에게 각각 적합한 UNIX 사용 인터페이스를 제공하기는 힘들뿐만 아니라 UNIX는 쉽게 익히기가 힘들고 실시간 처리를 요하는 응용의 경우에는 부적합하다.

1971년 인텔사에서 최초의 마이크로 프로세서를 설계한 이후 최근의 32비트 마이크로 프로세서는 1970년대 후반의 메인 프레임의 성능에 필적하게 되었다. 즉 불과 16년의 마이크로 프로세서의 역사는 40여년의 메인 프레임의 발전 과정을 그대로 따르고 있다. 현재의 프로세서는 많은 문제점을 갖고 있으며 이의 근본적인 해결을 위해서는 과거의 아키텍춰상에 새로운 특징을 부가함으로써만은 극복할 수 없다. 90년대의 만족스런 컴퓨터 시스템의 설계를 위해서는 보다 근본적인 새로운 아키텍춰의 도입이 필요하게 되었고 현재의 기술 수준은 과거의 아키텍춰를 능가하는 구조를 가능하게 한다.[1), 2)]

TRON 프로젝트는 이러한 기존의 문제점들을 분석하여 차세대 컴퓨터 시스템을 위한 통합적인 사양을 제안하는 데 목적이 있으며 다양한 응용 분야를 충족시키기 위해 몇개의 보조 프로젝트로 나누어 구성된다. 각 보조 프로젝트는 다음과 같은 4개의 계층과 각 계층간의 인터페이스를 제공한다.

(1) ISP(instruction set processer) layer

(2) operating system kernel layer

(3) OS kernel layer

(4) application and man-machine interface layer

TRON의 ISP 계층에서는 궁극적으로 모든 응용 분야에서 사용하게 될 CPU 칩을 설계 제작하기 위한 사양을 제시한다. 모든 응용 분야에서 같은 CPU를 사용하게 되면 그 CPU에 대한 수요가 증가될 것이므로 현재 많은 메이커에서 TRON CPU의 생산을 검토 혹은 개시하고 있다. TRON CPU는 32, 48, 64비트 버젼인 chip32, chip48, chip64의 3가지로 구성되며 그중 현재 chip32가 개발되고 있다.

다양한 응용을 모두 만족시키는 단일의 운영체제 커널을 구성하기는 힘드므로 여러 요구 조건을 분석하여 아래와 같이 크게 4분야의 응용을 지원하는 운영체에 커널을 분류 설계하며 〈그림 1〉은 그들간의 상호 연관을 보여 준다.[3]

(1) 공업용의 ITRON(Industrial TRON)

(2) 사무용의 BTRON(Business TRON)

(3) 통신 제어용의 CRON(Communication TRON)

(4) 생활환경을 제어하는 MTRON(Macro TRON)

3. TRON CPU의 특징

TRON 프로젝트는 90년대에도 계속 확대될 마이크로 프로세서 응용 분야에 부응하기 위하여 새로운 프로세서의 구조를 제안한다.

많은 32비트 마이크로 프로세서는 기본적으로 16비트 프로세서의 구조에 근거를 두고 있으며 이러한 구조적 호환성은 시장성의 측면에서는 매우 중요하다. 그러나 하드웨어와 소프트웨어가 복잡해짐에 따라 응용 분야 역시 그러해지는 경향이 있으므로 기존의 구조에 대한 호환성만을 중요시한 구조에서는 이러한 새로운 응용에 충분히 적합하지 못하다. 또한 마이크로 프로세서는 콘트롤러나 시퀀서등의 용도로 사용하기 위해 개발되었으므로 고성능 웍스테이션의 응용에 적합하기 위해서는 새로운 구조로의 접근이 필요하다. 따라서 90년대의 마이크로 프로세서는 고도로 다양한 응용에 적합한 새로운 구조를 채택해야

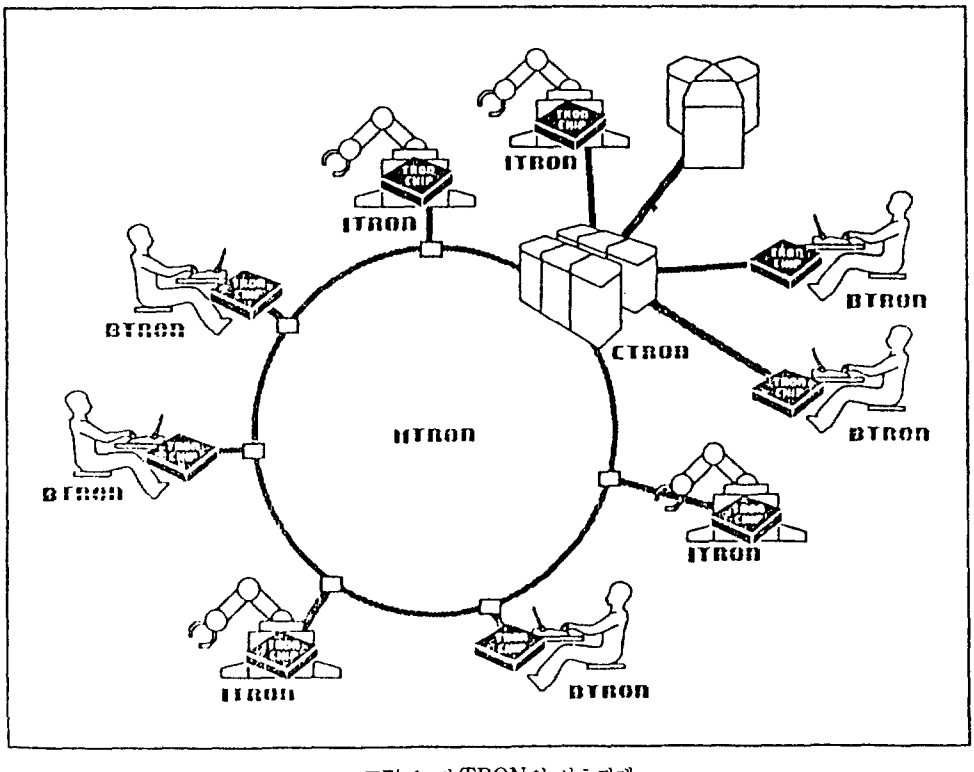

그림 1 각TRON의 상호관계

만 한다.

최근에는 병렬 프로세서및 데이타 플로우 구조등 폰 노이만 구조가 아닌 새로운 구조에 대한 연구가 활발히 진행되고 있으나 일반적인 사용 목적이나 기술의 향상 속도등을 고려할 때 90년대에도 역시 폰 노이만 구조의 컴퓨터가 주종을 이룰 것으로 예상되므로 이 구조의 장점을 최대한 이용하는 방향으로 개발이 진행될 것이다.

기존의 많은 마이크로 프로세서는 주소 공간을 분할해서 사용하며 수행 속도 역시 빠르지 않다. 따라서 차세대의 폰 노이만 방식의 컴퓨터는 매우 큰 주소 공간을 제공하고 고성능이어야 한다는 요구 사항을 만족해야 한다. 분할된 주소 기법은 커다란 크기의 프로그램에서는 많은 어려움을 야기시키며 수행 속도는 아이크로 프로세서의 명령어 세트의 복잡성에 의해 저하된다. 이러한 문제점은 기존의 아키텍춰를 벗어나지 않는 한 극복해 낼수 없다.[3)]

새로운 아키텍춰를 제안하는 또다른 이유는 표준적인 명령어 세트를 설계하는 데 있다. 시장성의 측면에서 볼 때 이미 인텔사나 모토롤라사등에서는 자사의 표준적인 명령어 세트를 갖고 있으나 이들은 모두 자사 제품에만 종속되므로 다른 제작자들은 이것을 사용할 수 없다. 명령어 세트는 소프트웨어에 절대적인 영향을 미치므로 프로세서에 무관하게 설계된 명령어 세트가 있다면 이식성이 우수한 소프트웨어의 개발및 낮은 가격의 어셈블러와 컴파일러의 개발을 촉진시킬 것이며 프로그래머의 교육 역시 쉬워질 것이다.

에이다(Ada)와 파스칼(pascal)같은 언어에 대해서는 이미 표준화 작업이 이루어져 왔으며 그결과 어느 단체이건 이들 표준화된 언어의 컴파일러와 인터프리터를 쉽게 설계할 수 있다. 그러나 프로세서의 명령어 세트에 관해서는 이러한 표준화 작업이 전혀 이루어지지 않고 있으며 법적으로 다른 제작자의 명령어 세트를 자신의 프로세서에서 구현하는 것을 막고 있다. 마이크로 프로세서의 보급이 늘어나면서 표준 명령어 세트를 구현하는 오픈 아키텍춰의 도입이 필요해짐에 따라 이러한 시도가 학계를 중심으로 이루어지고 있다.

TRON 프로젝트에서는 위에서 열거한 여러 문제를 해결하는 32비트 마이크로 프로세서의 설계를 제안하며 TRON 프로젝트 CPU의 특징을 요약하면 다음과 같다.

(1) 고속의 수행 속도

1990년대의 응용 분야를 분석한 결과 실시간 처리를 위한 시스템 내장 운영체제인 ITRON, 워스테이션에 근거한 운영체제인 BTRON등의 응용이 확대될 것이라는 판단하에 TRON CPU는 이러한 운영체제와 기타 응용 프로그램이 효과적으로 동작할 수 있는 공통의 아키텍춰를 설계함으로써 고속의 실행이 가능하게 한다.

(2) 광대한 주소 공간

TRON CPU는 넓은 주소 공간을 제공하기 위하여 다음과 3가지 모델로 구성된다. 즉 각각 32, 48, 64비트인 chip32, chip48, chip64의 3가지이며 32비트 비전인 chip32의 개발이 진행되고 있다. 물론 각 상위 모델은 하위모델에 대하여 호환성이 있다.

(3) 표준 명령어 세트

TRON CPU의 명령어 세트는 동경 대학의 사까무라 교수팀에 의해 설계되었고 구현은 여러 반도체 메이커에서 개별적으로 진행되고 있다. 이렇게 명령어 세트의 설계와 하드웨어의 구현이 공동으로 이루어질 수 있는 것은 TRON CPU가 오픈 아키텍춰로써 여러 컴퓨터 메이커에게 표준의 명령어 세트로 제안되었기 때문이다.

4. TRON CPU의 구조

최초의 TRON CPU인 chip32는 16개의 32비트 레지스터를 갖는 일반 용도의 마이크로 프로세서이다. 64비트로의 명령어 세트및 포맷에 주목하여 chip32의 특징을 요약하면 다음과 같다.

4. 1 64비트로의 확장성

폰 노이만 컴퓨터의 중요한 장점중의 하나는 네은 주소 공간으로의 직접적이고 자유로운 접근이다. 제한된 주소 공간의 사용은 소프트웨어 이식성에 어려움을 주고 수행 속도면에서도 불리하다. 특히 미래에는 인공지능 분야에서의 응용이 확대될 것이므로 네은 주소 공간의 사용 가능은 인공지능 분야의 소프트웨어 생산성 향상에 필수적이다. 현재의 디바이스 기술 개발과 가격 저하속도에 비추어 볼때 메모리의 집적도는 매 2년마다 2배로 높아지고 있으므로 1997년 경에는 32비트 메모리 칩이 등장할 것이며 결국 4G 바이트의 메모리 공간을 갖는 32비트 chip32는 32M 메모리 1024개 정도로 구현이 가능해질 것이다.

그러나 인공지능 응용 분야에서의 요구가 계속되면 이것도 충분한 주소 공간을 제공한다고 볼수 없다. 따라서

chip48, chip64로의 상향 호환성이 가능케 함으로써 이에 대한 해결을 도모한다. chip32에서 chip64로의 상향호환은 chip32에서도 64비트 데이타 조작과 어드레싱을 지원하는 명령어를 설계함으로써 가능하게 한다. 이를 위해 어드레싱 모드 사양에 64비트로의 확장을 위한 비트(P bit)를 둔다. 한편 모든 명령어의 오퍼랜드 필드를 6비트로 하여 64비트 데이타를 처리할 수 있도록 하며 현재 예측할 수 없는 미래의 기능으로의 확장을 위하여 사용하지 않는 비트를 확보해 둔다.

4. 2 컴파일러의 구성을 고려한 명령어 세트.

소프트웨어 생산성의 중요성은 날로 증가하고 있다. 따라서 모든 새로운 프로세서는 우수한 컴파일러가 구성될 수 있는 명령어 세트를 갖추어야 한다. 이러한 컴파일러는 소프트웨어에 대한 고려없이 설계된 프로세서상에서는 구현될 수 없다.

효과적인 목적 코드를 생산하는 컴파일러가 가능하기 위해서는 프로세서내의 모든 레지스터는 같은 기능을 갖는 일반 목적의 것이어야 한다. 즉 컴파일러를 고려한 프로세서의 설계를 위해서는 명령어의 대칭성이 중요하다. 명령어가 비대칭적일 때는 특정 함수를 가능하게 하기 위해 특별한 레지스터만을 사용해야 하는 경우로써 이때는 변수의 할당이나 작업 영역의 설정 과정이 매우 복잡해지기 때문이다.

TRON CPU에서는 모든 명령어는 가능한한 대칭적이 되도록 설계하였다. 16개의 레지스터는 컴파일러가 직접 생산할 수 없는 소수의 고급 명령어의 경우를 제외하고는 모두 같은 기능을 수행한다. TRON CPU의 명령어 포맷은 동작 모드와 어드레싱 모드 그리고 오퍼랜드의 크기를 지정하는 각각의 필드와 아울러 다른 프로세서는 갖추지 못한 다음의 두가지 모드를 컴파일러가 제공한다.

additional addressing mode는 기본적으로 덧셈과 간접 주소 지정 방식을 사용한다. 즉, displacement 덧셈, scaling(x1, x2, x4, or x8), 인덱스 레지스터와의 덧셈, 기억장소에 대한 간접 참조등으로 구현되며 임의의 n에 대하여$(n+1)$번의 간접 지정은 n번 어드레싱 확장에 의해 가능하다.

〈그림2〉는 어드레싱 확장이 일어나는 방법을 보여 준다.

서로 다른 크기의 데이타에 대한 연사의 가능은 피연산

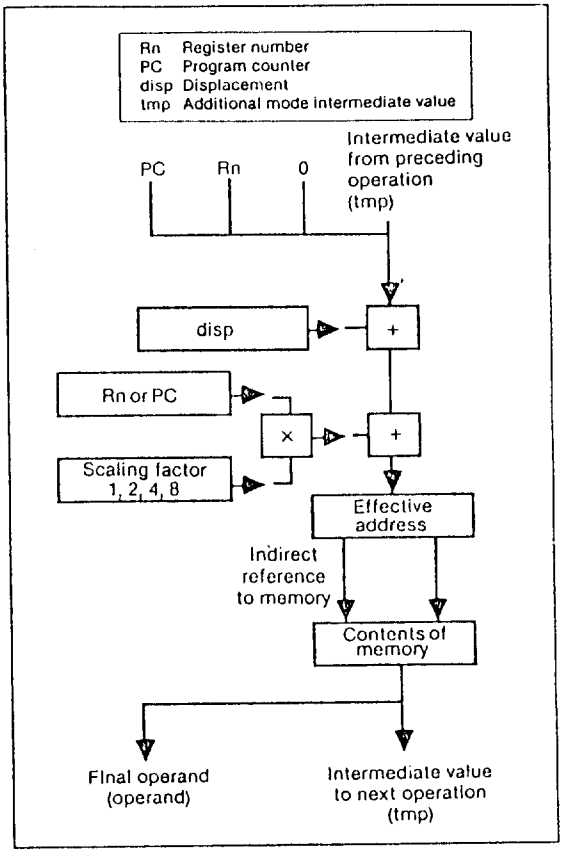

그림 2 추가적인 어드레싱 확장의 예

자의 zero 및 sign확장 그리고 연산을 하나의 명령어로 가능하게 한다는 것을 의미한다. 예를 들어 32비트 데이타와 64비트 데이타와의 덧셈이 하나의 명령어를 사용함으로써 가능하다. 이러한 가능은 서로 다른 데이타 타입간의 전환(conversion)이 빈번히 일어나는 C와 같은 언어의 컴파일러의 구현에 매우 용이하다. 또한 주소의 계산및 작은 immediate값을 오퍼랜드로 갖는 명령의 길이를 짧게 할때 유용하다. 이러한 가능이 없는 프로세서에서는 32비트 데이타를 64비트 데이타로 확장한 후 64비트 데이타끼리 연산하여야 하므로 임시값을 저장하기 위한 레지스터를 필요로 하며, 결국 레지스터 할당에 영향을 미치게 되어 전체 컴파일러 구성을 복잡하게 한다.

4. 3 운영 체제를 고려한 명령어 세트

TRON 프로젝트에서는 ITRON 및 BTRON 운영체제의 구조를 고려하여 CPU를 설계함으로써 운영체제가

CPU상에서 가장 효율적으로 동작할 수 있는 환경을 제공한다. 시스템 소프트웨어가 고속으로 동작하기 위해서는 빈번히 사용하는 간단한 명령어의 속도 향상이 필요하다. 그러나 간단한 명령의 속도 향상은 전체적인 성능의 향상에는 기여하나 몇몇의 미묘한 경우에는 좀더 고급의 명령어가 필요하다.

예를 들어 ITRON 사양에 기초한 운영체제에서는 인터럽트의 발생과 처리를 위한 타스크의 액티베이션간의 시간 간격을 줄이는 것이 매우 중요하다. 이러한 타스크 액티베이션이 고급의 명령어에 의해 하드웨어상에서 수행된다면 수행속도는 훨씬 빨라질 것이며, 인터럽트에서 액티베이션으로의 응답속도 역시 빠를 것이다. 실시간 처리 응용의 경우에 응답 속도는 평균 성능의 향상보다 매우 중요하므로 응답속도 향상을 위한 이러한 고급 명령어의 도입이 요구된다.

ITRON 운영체제를 위해 구현되는 고급 명령어에는 문맥교환과 큐의 조작을 위한 것이 있다. 문맥교환을 위한 명령어 LDCTX와 STCTX는 타스크간의 빠른 교체를 위해 사용되며 미래의 TRON CPU는 이러한 문맥을 저장하기 위한 고속의 기억장소가 포함될 것이다. 큐 조작 명령어로는 QINS, QDEL및 QSCH가 있으며 이것들은 ITRON에서 사용되는 ready큐와 wait큐의 효과적인 조작을 위해 사용된다.

BTRON 운영체제상에서 구현되는 고급 명령으로는 bitmap 처리를 지원하는 명령어를 들수 있다. 예를 들어 BVMAP, BVCPY 및 BVPAT가 있으며 이들은 BTRON 사양에서 윈도우의 처리를 위해 필요한 bitmap 데이타의 계산과 이동을 효율적으로 수행한다. 이를 위해서는 그래픽 프로세서와 같은 전용의 칩을 사용할 수도 있으나 이 경우는 메인 프로세서와의 통신 부담으로 인한 병목 현상이 일어날 수도 있다. 또한 두개의 프로세서간의 데이타 통로는 bitmap 조작을 충분히 지원할 정도로 넓어야만 한다. 이러한 이유로 전체 성능을 고려할 때 전용의 프로세서를 두는 것보다는 메인 프로세서에 bitmap 처리 기능을 부여하는 방안이 더욱 좋다.

4. 4 명령어 포맷

전형적인 명령어의 포맷은 가변 길이 포맷과 고정 길이 포맷의 두가지가 있다.

가변 길이 포맷은 VAX11 컴퓨터에서 사용되며 동작 모드와 어드레싱 모드및 1개 혹은 그 이상의 오퍼랜드의 지정을 위한 8비트의 필드로 구성된다. 이것의 단점은 레지스터와 레지스터간의 산술연산과 같은 경우 레지스터를 지정하기 위해서는 4비트이면 충분한데도 필요 이상의 크기로 명령어가 길어지게 된다는 점이다.

고정 길이 포맷은 명령어 디코더의 복잡도를 줄이고 디코딩 속도를 빠르게 하며 RISC 컴퓨터에서 채택되고 있으나 단점으로는 명령어에서 제공하는 기능이 제한되므로 프로그램밍이 어려워진다는 것을 들 수 있다.

우리는 빈번하게 사용되는 명령어의 크기를 줄이고 수행시간을 단축시키는데 주목한다. TRON CPU에서 사용되는 포맷은 가변 길이의 포맷과 고정 길이의 짧은 포맷이 모두 동시에 사용된다. 가변 길이 포맷에서 주소 확장 부분을 제외한 부분은 4바이트로 되어 있고 고정 길이 포맷은 2바이트이다.

피연산자가 2개인 명령어의 경우에는 모두 가변 길이 포맷을 사용한다. 그러나 빈번하게 사용되는 2개의 피연산자를 갖는 명령어에서는 예를 들어 데이타 전송(MOV)과 데이타 비교(CMP)의 경우에는 2가지의 포맷이 모두 가능하다. 데이타 전송 명령(MOV)에서 MOV : G는 메모리와 메모리와 메모리간의 데이타 전송및 부호 확장을 수행하며, 고정 길이 포맷인 MOV : Q는 3비트 상수값을 메모리로 전송한다. MOV : L은 메모리에서 레지스터로의 전송을, MOV : S는 레지스터에서 메모리로의 전송을 수행한다. 만일 어드레싱 모드 확장을 고려하지 않는다면 고정 길이 포맷의 데이타 전송 명령어는 2바이트로 충분할 것이다.

TRON CPU 어셈블리 프로그래밍을 용이하게 하기 위해서 일반 뉴모닉(general mnemonic)과 포맷 의존 뉴모닉(format dependent mnemonic)의 2가지가 사용된다.

일반 뉴모닉은 명령어의 동작은 표시하지만 형태를 보여주지는 못한다. 예를 들어 MOV, CMP, ADD등은 일반 뉴모닉의 대표적인 예이다. 반면 포맷 의존의 뉴모닉은 명령어의 동작뿐 아니라 형태까지도 명시하고 있다. 예컨데 MOV : G, MOV : Q, MOV : L등이 이에 해당한다.

명령어의 형태에 구애받지 않고 프로그램을 작성시에는 일반 포맷의 뉴모닉을 사용하고 이때 어셈블러는 자동적으로 각 명령어에 대하여 최적의 포맷을 갖도록 한다. 그러나 명령어의 길이와 수행시간에 밀접한 관련을 갖는 프로그램을 작성시에는 포맷 의존의 뉴모닉을 사용하여

야만 한다.

가변 길이의 포맷을 갖는 명령어로는 TRON 사양이 제공하는 모든 어드레싱 모드와 사이즈가 다른 데이타간의 연산이 자유롭다. 따라서 가변 길이 포맷만을 사용하는 명령어로 구성된 목적코드를 생산하는 컴파일러의 구성은 매우 쉽고 간단하다. 그러나 고속의 수행이 가능케 하기 위해서는 고정 길이 포맷으로 된 목적 코드를 생산해야 하므로 컴파일러는 두가지 포맷을 자유롭게 변환시킬 수 있는 기능을 제공해야 한다.

이러한 최적화 과정이 각 명령어 단계에서 일어난다면 이것은 비교적 간단한 알고리즘으로 구현이 가능하다. TRON CPU에서 동작하는 컴파일러는 최적화 과정을 거친 목적코드를 생산해낸다.

5. 결 론

TRON CPU는 64비트로의 확장및 컴파일러와 운영체제의 요구조건을 고려하여 설계되었다. TRON CPU의 구조는 오픈 아키텍춰로써 서양은 학교에서 설계하고 실제 구현은 여러 메이커에서 개별적으로 진행되고 있다. 이러한 오픈 아키텍춰는 특정 메이커에 종속되지 않으므로 차후의 표준 아키텍춰로써 자리잡게 될것이다.

현재 일본에서는 히다찌와 후지쓰및 미쯔비시 3사가 G마이크로 패밀리라고 불리는 그룹을 형성하였으며 이 3사는 1980년대말의 반도체 기술(0.1 미크론의 설계 규칙과 1백만개 이하의 트렌지스터)로 개발할 수 있는 TRON 양식의 마이크로 프로세서를 미니컴급의 처리 능력을 대상으로 하는 상위(G마이크로/300)와 웍스테이션급을 대상으로 하는 중위(G마이크로/200), ASIC 코아용의 하위(G마이크로/100)로 나누어 3사가 부담, 개발하기로 하는등 활발한 개발 붐을 이루고 있다.

참 고 문 헌

1). K. Sakamura, "Looking into the future with TRON", IEEE Micro, 1987.

2). K. Sakamura, "The TRON project", IEEE Micro, 1987

3). K. Sakamura, "Architecture of the TRON VLSI CPU", IEEE Micro, 1987.

4). K. Sakamura, "Configuration of the CTRON Kernel", IEEE Micro, 1987.

5). K. Sakamura, "Introduction to ITRON", IEEE Micro, 1987

6). K. Sakamura, "BTRON", IEEE Micro, 1987