CalmRISC : 효율적인 보조 프로세서 인터페이스를 가진 저전력 마이크로 컨트롤러

삼성전자 임경묵

1. 서 론

VLSI 기술이 발전하면서 MCU(마이크로 컨트롤러) 설계에 있어서 저전력, 고성능, SOC(System On a Chip)의 세 가지 경향이 나타나게 되었다.

최근에 PDA, 노트북, 핸드폰 등 배터리를 사용하는 휴대용 기기들이 널리 퍼졌다. 이러한 기기에 있어서 적은 소모전력은 배터리의 수명을 연장 시킬 뿐만 아니라 휴대하기 용이하도록 배터리의 무게나 크기를 줄일 수 있다. MCU에서 소모되는 전력은 정상 동작 시 전체 소모전력의 상당 부분을 차지하고 있으며 동작을 하고 있지 않을 때의 소모 전력의 대부분을 차지하고 있기 때문에 많은 주목을 받아왔다. 또한 시스템이 점차로 복잡해져 가면서 MCU는 좀 더 복잡한 작업을 수행하도록 요구되어졌다.

RISC(Reduced Instruction Set Computer)는 고성능에 적합하다고 알려져 있고 최근에 저전력의 특성으로 관심을 끌고있다. 때문에, 저전력과 고성능을 위한 아키텍처로 널리 받아들여지고 있다[10][12].

SOC 방식은 설계 비용, 전력 소모, 시스템 복잡도를 현저히 줄일 수 있기 때문에 ASIC 설계자에게 매력적인 방식이다. 대부분의 시스템에서 MCU는 필수 구성 요소이고 프로그래머블 코아를 중심으로 시스템을 구성하는 것이 다른 방식에 비해 용이하기 때문에 SOC는 MCU를 중심으로 이루어지고 있다. 특히 최근에 DSP 기술이 향상되면서 MCU와 DSP의 통합이 주목을 받아왔다[1].

MCU 코아와 DSP 코아를 합치는 가장 간단한 방법은 단일 칩에 컨트롤을 위한 MCU와 신호 처리를 위한 DSP를 독립적으로 놓는 방식이다. 이 두개의 코아 방식은 응용분야의 필요에 의해서 적합한 MCU와 DSP를 고를 수 있어 융통성이 뛰어나나 여러 가지 단점을 가지고 있다. 첫째로 각각의 코아는 자신만의 프로그램과 데이타 메모리를 가지고 있어야 하고 두개의 개발환경을 사용해야 하기 때문에 프로그램 개발이 힘들다. 둘째로 자원의 공유나 통신을 위해서 복잡한 프로토콜이 필요하다. 세째로는 각각의 코아가 어느 정도는 비슷한 기능을 하기 때문에 하드웨어 낭비가 발생한다.

또 다른 방식으로는 하나의 프로세서가 MCU와 DSP의 기능을 모두 가지고 있는 방식이 있다. 예로 MAC (Multiply and Accumulate) 명령을 가진 MCU나 비트 연산 명령이나 다양한 분기 명령을 가지고 있는 DSP를 들 수 있다[14][15][16]. 이 방식은 두 개의 코아 방식에 비해 하드웨어 낭비가 없고 통신을 위한 프로토콜이 필요 없으며 단일 개발 환경을 사용할 수 있는 장점이 있으나 융통성과 확장성이 떨어진다. 만일 좀 더 강력한 DSP 기능을 원한다면 전체 프로세서를 다시 설계해야 한다.

본 논문에서는 저전력 RISC MCU인 CalmRISC의 아키텍처와 CalmRISC에서 사용된 저전력 회로 설계 기법을 설명한다. 제2장에서는 적은 전력 소모를 위한 아키텍처 수준의 고려 사항에 대해 설명하고, 대표적인 저전력 회로 설계 기법을 제3장에서 설명한다. 제4장에서는 구현결과를 제시하고 결론을 맺는다.

2. CalmRISC의 저전력 아키텍처

MCU 코아의 아키텍처를 선정함에 있어서 소모전력, 성능, 크기, 호환성 등 여러 가지 기준이 있다. 특히 MCU 코아가 칩 안에 내장되는 응용의 경우 칩 안에는 MCU 외에 프로그램 ROM, 데이타 RAM, 주변 회로 등 다른 구성 요소들이 있기 때문에 MCU 코아 자체만이 아니라 전체 칩의 관점에서 아키텍처를 선정하여야 한다.

CalmRISC는 RISC 아키텍처를 선택하였다. 2단어

명령어를 제외하고는 모든 명령어는 1단어로 고정되어 있고 1사이클에 수행된다. 통상의 RISC 아키텍처의 명령어 집합은 적은 수의 명령어들을 포함하고 있고 각 명령어의 수행은 규칙적이고 간단하다. 따라서 컨트롤 회로와 데이타 처리부가 CISC(Complex Instruction Set Computer)에 비해 상대적으로 간단하고 전력 소모와 칩 크기를 이에 비례해 줄일 수 있다. 또 다른 RISC 아키텍처의 장점은 파이프라인 기법을 사용하기 쉽다는 점이다. 파이프라인이 잘 디자인 되어 있을 경우 각 명령어 수행 시의 시간적 여유, 1.0 근처의 낮은 CPI(clock cycles per instruction), 높은 성능을 얻을 수 있다. 낮은 CPI 혹은 높은 성능은 MCU로 하여금 주어진 작업을 더 작은 클럭 사이클만에 수행할 수 있게 해주고 높은 CPI를 가진 MCU에 비해서 더 적은 전력을 소모할 수 있는 가능성을 제시해 준다. 아키텍처의 관점에서 CPI를 낮추는데 많은 노력을 들이고 있으며 따라서 RISC 아키텍처는 자연스러운 선택이라고 할 수 있겠다.

앞에서 언급한대로 전체 칩 구성요소를 고려하지 않으면 MCU 자체의 전력 소모는 큰 의미를 가지지 않는다. 이러한 구성 요소를 프로그램 메모리를 들 수 있다. 만일 어떤 아키텍처의 명령어 세트가 효율적이지 못하여 큰 프로그램 메모리를 필요로 한다면 칩 크기를 증가시킬 뿐만 아니라 전력 소모 또한 증가시킨다. 지금부터는 MCU 분야에서 널리 받아들여지고 있는 레지스터/메모리, 메모리/메모리, 로드/스토어의 세가지 명령어 아키텍처를 고려하겠다[2]. 이 세가지 명령어 아키텍처의 공정한 비교를 위해서 모든 ALU 명령어들은 두개의 데이타, op1과 op2를 받아들여 계산을 수행한 후, 결과를 op1(즉 op1←op1⊕op2)에 저장하는 것으로 가정한다. CalmRISC는 레지스터-메모리 명령어 아키텍처를 가지고 있어 op2만 메모리가 될 수 있다. 메모리/메모리 명령어 아키텍처에서는 두개의 데이타 중 하나가 메모리가 될 수 있으며 로드/스토어 명령어 아키텍처에서는 메모리는 오직 로드 명령어나 스토어 명령어에서만 사용될 수 있다. 표 1에서는 모두 같은 주파수로 동작한다는 가정하에 전형적인 작업들에 대해 세 명령어 아키텍처에서 필요한 명령어의 수를 비교하였다.

위 표에서, M_i는 메모리 데이타, R_i는 내부 레지스터를 의미한다. 메모리/메모리 명령어 아키텍처가 가장 좋은 코드 밀도를 가지고 있으며 로드/스토어는 가장 나쁘다는 것을 알 수 있다. 하지만 단순한 평균 명령어 수의 비교는 명령어 당 평균 비트 수와 어드레싱 모드, 프로그램 및 데이타 메모리의 접근 시간, 그리고 레지스터의 수 등 다른 측면을 세심히 고려하지 않으면 잘못된 해석 결과를 낳을 수 있다.

표 1 여러 가지 명령어 아키텍처의 코드 크기 비교

	명령어 수		
작업	레지스터-메모리	메모리-메모리	로드/스토어
M1=M1+M2	3	2	4
M1=M2+M3	3	3	4
M1=M1+R1	2	1	3
M1=M2+R1	2	2	3
R1=M1+M2	2	2	3
M1=R1+R2	2	2	2
R1=R2+M1	2	2	2
R1=R2+R3	2	2	2
평 균	1.00	0.89	1.28

◆ 명령어 당 평균 비트 수와 어드레싱 모드 : 세가지 명령어 아키텍처는 서로 다른 명령어 당 평균 비트 수와 어드레싱 모드를 가지고 있다. 일반적으로 메모리의 한 장소를 고르는 것은 레지스터 중 하나를 고르는 것에 비해 더 많은 명령어 공간을 차지한다. 메모리/메모리 명령어 아키텍처는 두 데이타의 메모리 어드레스가 필요한 반면 레지스터/메모리 명령어 아키텍처는 한 데이타의 어드레스가 필요하고 로드/스토어 명령어 아키텍처의 경우는 로드나 스토어 명령어에만 어드레스가 필요하다. 만일 어드레싱 모드가 모두 같다면 메모리/메모리 명령어 아키텍처가 명령어 당 평균 비트 수가 가장 크고 로드/스토어 명령어 아키텍처가 가장 작으며 레지스터/메모리 아키텍처는 중간 정도이다. 만일 명령어 당 평균 비트수가 같다면 로드/스토어 명령어 아키텍처가 가장 강력한 어드레싱 모드를 가지고 있고 메모리/메모리 명령어 아키텍처가 가장 약한 어드레싱 모드를 가지며 레지스터/메모리 명령어 아키텍처는 그 중간이다. 만일 응용 프로그램이 크고 강력한 어드레싱 모드를 요구한다면 메모리/메모리 명령어 아키텍처를 가진 MCU는 약한 어드레싱 모드를 보완하기 위해서 좀 더 많은 수의 명령어를 필요로 하게 되고 따라서 코드 크기가 빠르게 증가할 것이다. 우리는 메모리/메모리 명령어 아키텍처는 작은 크기의 응용 프로그램

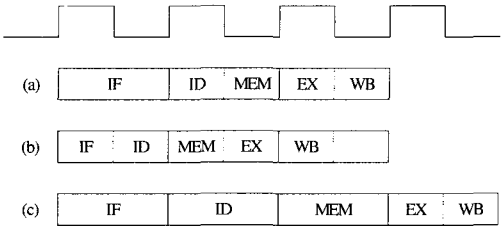
에 대해 코드 밀도가 가장 좋고 로드/스토어 명령어 아키텍처는 큰 크기의 프로그램에 좋으며 레지스터/메모리 명령어 아키텍처는 작거나 중간 정도 크기의 프로그램에서 좋은 코드 밀도를 보인다고 결론지을 수 있다.

◆ 레지스터 수 : 레지스터 수는 전력 소모나 성능에 매우 중요한 요소이다. 시간적인 국부성이 있어서 가까운 미래에 사용될 변수들을 레지스터에 저장함으로써 메모리 접근 횟수를 줄일 수 있다. 이것은 큰 레지스터 화일을 사용함으로써 전력소모나 코드 크기를 줄일 수 있는 확률이 커진다는 것을 의미한다. 명령어 공간의 부족으로 인하여 메모리/메모리 명령어 아키텍처에서는 레지스터를 늘리기가 용이하지 않다. 로드/스토어 명령어 아키텍처는 상대적으로 레지스터를 늘리기 쉬우며 레지스터/메모리 명령어 아키텍처는 그 중간 정도이다. 따라서 많은 변수를 사용하는 복잡한 프로그램에는 로드/스토어 명령어 아키텍처가 적합하며 간단한 프로그램을 위해서는 레지스터/메모리 명령어 아키텍처가 적당하다.

◆ 프로그램/데이타 메모리 접근 시간 : 필요로 하는 프로그램 메모리 (통상 ROM)나 데이타 메모리의 접근 시간은 전체 칩의 전력 소모나 크기에 지대한 영향을 미치는 요소이다. 일반적으로 짧은 메모리 접근 시간은 좀 더 큰 크기와 더 많은 전력 소모를 동반하게 된다[3][4]. 메모리/메모리 명령어 아키텍처의 CPI는 약 4에서 20정도 되기 때문에 RISC와 같은 성능을 유지하기 위해서는 이에 비례해서 빠른 동작 주파수가 필요하다. 따라서 메모리/메모리 명령어 아키텍처는 다른 아키텍처에 비해서 좀 더 빠른 메모리를 필요로 하고 이것은 칩 크기와 소모 전력을 증가시킨다.

지금까지 살펴본 바와 같이 레지스터/메모리 RISC 아키텍처가 프로그램 메모리가 상대적으로 적은 저전력 8 비트 내장 응용에 가장 적합하다고 할 수 있다. 따라서 가장 중요한 설계 기준이 저전력인 CalmRISC에서는 레지스터/메모리 RISC 아키텍처를 채택하였다.

CalmRISC는 간단한 3단계 파이프라인 아키텍처를 가지고 있다. 그림 1은 CalmRISC의 파이프라인 구조와 레지스터/메모리 명령어 아키텍처를 가진 MCU가 채택 가능한 다른 파이프라인 구조를 보이고 있다. CalmRISC의 가장 중요한 설계 기준은 저전력이기 때문에 4단계보다 더 깊은 파이프라인은 고려 대상에서 제외하였다.



(a) CalmRISC 파이프라인
(b) 다른 3단계 파이프라인
(c) 다른 4단계 파이프라인

그림 1 파이프라인 구조

CalmRISC 파이프라인(그림 1 (a))의 첫번째 단계인IF (Instruction Fetch) 단계에서는 프로그램 메모리로부터 명령들이 읽혀지며 몇몇 명령어들에 대해서 사전 디코딩이 수행된다. 두번째 단계인 ID/MEM (Instruction Decode/Memory) 단계에서는 읽혀진 명령어가 디코딩되고 데이타가 메모리나 레지스터 화일로부터 읽혀진다. 만일 필요 데이타가 메모리로부터 오거나 결과 데이타가 메모리에 써져야 하는 경우에는 간단한 사전 디코딩에 의해서 첫번째 단계에서 이 사실을 알아내고 필요로 하는 메모리 주소값이 두번째 단계의 앞부분에서 계산된다. 세번째 단계인 EX (Execution) 단계에서는 ALU 동작이 수행된 후 그 결과가 레지스터 화일에 기록된다. 파이프라인 구조가 간단하기 때문에 데이타 상호 의존에 의한 파이프라인 휴식은 없으나 조건부 분기 명령에 의한 파이프라인 휴식이 발생한다. 이것 때문에 CalmRISC의 CPI는 1.0보다 조금 큰 약 1.1에서 1.2정도가 되나 보통의 CISC 보다는 훨씬 작은 값이다.

그림 1 (b)는 CoolRISC™가 채택한 다른 3 단계 파이프라인을 나타낸다[5]. 이 파이프라인 구조는 IF 단계를 제외한 다른 단계를 CalmRISC에 비해 반 사이클 먼저 수행한다. 이 구조의 장점은 분기 명령에 의한 파이프라인 휴식이 없기 때문에 CPI가 이상적인 값인 1.0을 가진다는 것이다. 하지만 이 구조는 좀 더 빠른 프로그램 메모리를 요구한다. 파이프라인의 첫번째 사이클에서 프로그램 메모리 읽기와 명령어 디코딩이 모두 일어나야 하기 때문에 CalmRISC에 비해 약 2배 빠른 프로그램 메모리를 요구하게 된다. 일반적으로 프로그램 메모리가 소모하는 전력은 MCU가 소모하는 것보다 비슷하거나 많은 편이고 2배 빠른 메모리는 2내지 4배의 전력을 소모한다[3].

프로그램 메모리는 명령어 당 한번씩 수행되기 때문에 2배 빠른 메모리 속도는 2내지 4배의 더 많은 전력 소모를 의미한다. 만일 어떤 이유에서 더 빠른 프로그램 메모리가 사용되지 못한다면 수행 속도를 느리게 하는 수 밖에 없고 이는 성능의 저하를 의미한다.

그림 1 (c)는 4단계의 다른 파이프라인을 나타내고 있고 이 파이프라인에서는 CalmRISC 파이프라인의 ID/MEM 단계를 별도의 두 단계로 분리를 하였다. 일견에는 CalmRISC에 비해서 데이타 메모리 계산과 데이타 메모리 접근 시간에 여유가 있는 것처럼 보이지만 연속한 명령어간에 존재하는 레지스터 상호 의존성 때문에 현실적으로 CalmRISC 파이프라인보다 개선된 점이 없게 된다. 명령어 디코딩 시간에는 여유가 생기나 CalmRISC 파이프라인에서 명령어 디코딩은 임계경로가 아니기 때문에 4단계로 파이프라인을 늘려서 생기는 수행 시간의 여유는 거의 없다. 이 파이프라인은 분기 명령에 의한 파이프라인 휴식이 2 사이클이기 때문에 유효 CPI는 1.2에서 1.5 정도가 되고 이는 성능의 저하를 의미한다.

CalmRISC는 잘못된 예측으로부터 야기되는 전력 소모를 방지하기 위해서 분기예측 방법을 사용하지 않는다. 이 방식과 분기되지 않는 것으로 예측하는 방식과 비교를 해보자. SPEC 벤치마크 프로그램에서 전체 수행 명령어 중 약 20%를 분기 명령이 차지하고 분기 명령 중 60% 정도가 분기를 수행한다. 분기가 일어나면 분기 예측의 오류가 발생하고 이미 읽혀진 명령어는 사용되지 않고 버려지게 된다. 프로그램 메모리는 MCU와 거의 비슷한 전력을 소모하기 때문에 예측 오류 확률에 비례해서 전력 낭비가 발생한다. 아래의 표는 100개의 명령어 수행 시 CalmRISC가 소모하는 전력과 분기하지 않는다고 예측하는 방식에서 소모하는 전력을 비교한 것이다. 분석을 간단히 하기 위해서 프로그램 메모리는 MCU와 같은 전력을 소모한다고 가정을 하였고 MCU가 1 사이클 당 소모하는 전력을 1로 하였다.

	CalmRISC	비분기 예측 방식	차 이
수행된 명령수	100	100	0%
프로그램 메모리 접근 수	100	112	10.7%
총 에너지 소모	200	212	5.7%
수행 시간	120	112	7.1%
사이클 당 소모전력	1.67	1.89	11.9%

위의 결과에 의하면 CalmRISC의 분기 명령어 처리 방식은 7.1%의 성능 저하를 야기하며 5.7%의 전력을 절약한다. 만일 응용 프로그램이 최대 성능을 요구하지 않는다면 7.1%의 성능저하는 큰 의미가 없는 수치이다.

3. CalmRISC의 저전력 회로 설계 기법

CMOS 회로가 소모하는 전력은 다음 식으로 표현된다.

$$P=\sum_i \alpha_i C_i V_{dd}^2 f$$

여기서 i는 전이 확률, C_i는 커패시턴스, V_{dd}는 동작 전압, f는 동작 주파수를 의미한다. 성능을 그대로 유지하면서 동작 전압 Vdd를 줄이기 위해서는 임계 전압이 더 낮은 공정을 사용하거나 만일 공정이 이미 주어진 상태라면 임계 경로를 줄여야만 한다 [6][13][14]. 동작 주파수를 줄이기 위해서는 CPI를 낮추어야 한다. 앞에서 설명한 바와 같이 CalmRISC는 하바드 RISC 아키텍처를 채택하여 시간적인 여유가 많고 CPI가 낮다. 아키텍처와 공정이 주어졌을 때 저전력 회로를 설계하기 위해서는 전이 확률 i와 커패시턴스 C_i를 줄여야한다. C_i를 줄이기 위해서 배치 커패시턴스와 가능하다면 최소 크기의 셀을 사용하여 입력 커패시턴스도 줄여야 한다. 예를 들어 래치는 플립플랍보다 보통의 경우 전력 소모가 30%정도 적고 25%정도 작기 때문에 가능하다면 래치를 사용하는 것이 전력 소모가 적다.

각 노드의 전이 확률을 줄이기 위해서는 신호 전이 방지를 통하여 불필요한 전이를 피하여야 한다. 특히, 클락 신호는 커패시턴스가 매우 크고 동작 주파수 또한 다른 신호에 비해 크기 때문에 특별한 주의가 필요하다. 래치/플립플랍의 주요 전력 소모 요소는 클락 신호이다. 만일 어떤 셀에 클락이 인가 되고 새로운 데이타가 전체 동작에 아무런 영향을 끼치지 않는다면 해당 클락의 전이는 불필요한 것이 된다. A[n-1:0]를 입력으로, Y[n-1:0]를 출력으로, CLK를 클락으로 하는 n 비트의 플립플랍을 생각해 보자. 만일 새롭게 입력된 A[n-1:0]가 주변 회로에 아무런 영향을 끼치지 않는다면 우리는 CLK의 전이를 막아도 된다. 만일 이러한 조건이 쉽게 검출 가능하다면 이 조건을 이용하여 CLK의 전이를 막아 원하는 기능을 바꾸지 않고 CLK의 전이로 인한 전력 소모를 방지할 수 있다[8].

클락의 전이를 막아 전력 소모를 줄인 예로 그림 7에 나와있는 PAGU(Program Address Generation Unit)의 12 비트 가산기를 들 수 있다. 명령어가 분기하지 않고 연속적으로 수행될 때 (대부분의 경우 분기를 하지 않는다) 프로그램 주소는 각 명령어마다 1씩 증가한다. 아래쪽 비트에 비해서 위쪽 비트의 전이 확률을 현저히 떨어지는 사실로부터 그림 2에 나와있는 구조가 구현 가능하다.

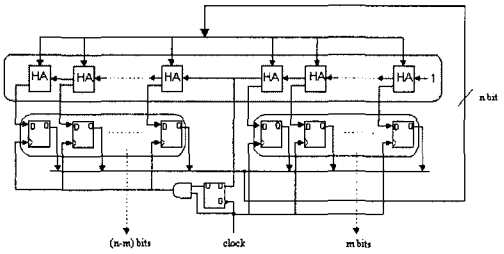


그림 2 저전력 가산기를 위한 클락 전이 방지 방식

아래쪽 m 비트는 매 사이클마다 클락을 인가하고 나머지 (n-m) 비트에 대해서는 m 번째의 캐리가 발생되었을 때만 클락을 인가한다. 한 비트당 사이클당 소모하는 전력을 1이라고 둘 때 클락 신호가 소모하는 전체 전력 P는 다음 공식으로 표현할 수 있다.

$$P=(n-m)(1/2)^m+m, \text{ where } n=12$$

간단한 계산에 의해서 m이 3일 때 전력 소모 P는 4.1의 최소값을 가진다. 클락의 전이를 막는 방식으로 PAGU에서 사용하는 클락 전력 소모의 70% 가량을 줄일 수 있었다.

클락의 전이를 막는 또 다른 예로 ICU(Interrupt Control Unit)을 들 수 있다. 저전력을 위한 ICU의 구조가 그림 3에 나타나 있다.

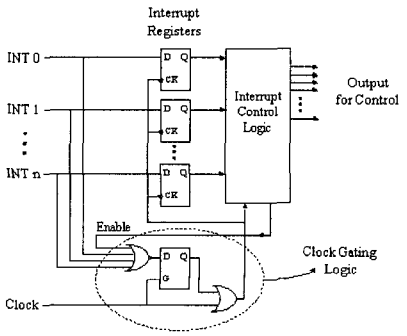


그림 3 저전력 인터럽트 처리를 위한 클락 전이 방식

인터럽트 신호(INT0~INTn)들은 양의 값을 가질 때 인터럽트의 발생을 의미하며 이것들은 인터럽트 레지스터들에 기억된다. 클락 전이 방지 회로는 새로운 인터럽트가 발생되거나 발생된 인터럽트를 처리할 때만 클락의 전이를 허용하고 다른 경우에는 클락의 전이를 막는다. 그림 3에서 사용한 방식으로 ICU에서 소모하는 전력의 90% 이상을 줄일 수 있다.

파이프라인 컨트롤에도 클락 전이 방지 방식이 사용된다. 어떤 단계가 수행해야 할 일이 없을 때 해당 단계에는 클락이 공급되지 않는다. 예를 들어 메모리에 데이타를 저장하는 명령어를 수행할 때는 EX 단계에 클락이 공급되지 않으며 많이 파이프라인 휴식이 발생되었을 때에는 모든 단계에 클락이 공급되지 않는다.

위에서 설명한 클락 전이 방지 방식 외에 선택적 입력이라 불리는 방식이 CalmRISC 디자인 시 많은 곳에서 사용되었다. 선택적 입력은 어떤 조합회로의 출력이 사용되지 않을 때에는 해당 조합회로의 입력을 래치로 변하지 않도록 막아서 불필요한 신호의 전이를 막는 방식이다. CalmRISC에서는 선택적 입력 방식은 특히 파이프라인 레지스터들간에 존재하는 조합회로의 소모 전력을 줄이는데 효과적으로 사용되었다.

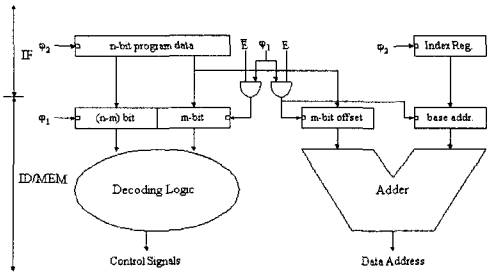


그림 4 DAGU에서의 선택적 입력의 예

그림 4는 DAGU(Data Address Generation Unit)에서 선택적 입력 방식을 이용하여 전력 소모를 줄인 예를 보여주고 있다. 프로그램 메모리로부터 오는 데이타의 일부는 명령어 디코딩이나 메모리 어드레스 계산에 쓰이게 되는데 만일 데이타 메모리를 접근하는 명령어이면 어드레스를 계산하는 가산기의 입력으로 프로그램 데이타가 저장되고 그렇지 않은 경우는 명령어 디코딩 쪽으로 저장된다. 이 방식으로 데이타 메모리 접근이 필요하지 않은 명령을 수행할 때

어드레스 가산기의 불필요한 전이를 방지하였다. 비슷한 구조가 PAGU에도 적용되었다.

표 2는 앞에서 언급한 여러가지 방식이 최종 소모 전력인 34A에 대비하여 얼마나 전력 소모를 절감했나를 나타내고 있다.

표 2 여러 방식의 전력 절감

사용 방식	절감량(A)	절감 비율(%)
PAGUincremental logic	1.94	5.7
Interrupt Control Unit	2.72	8.0
Pipeline clock control	7.39	21.7
Selective input latching	9.27	27.2

4. 구현결과 및 결론

그림 5에는 CalmRISC의 블럭도가 나타나있다.

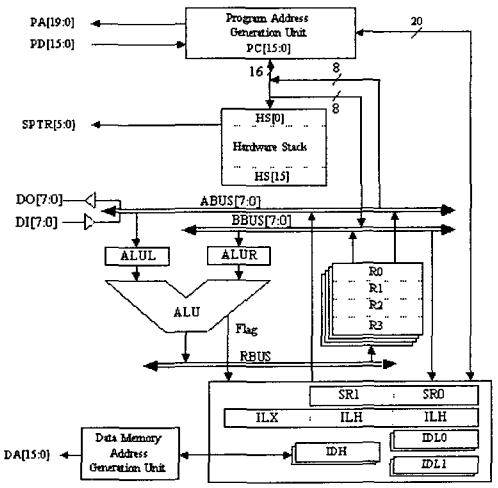
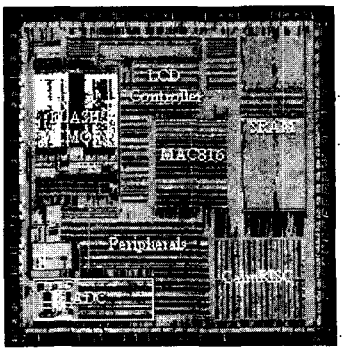
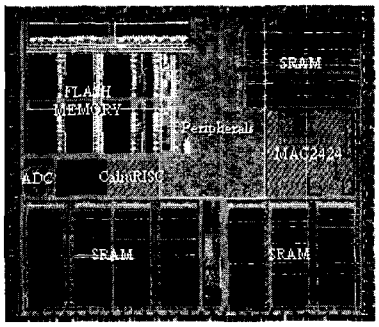


그림 5 CalmRISC의 전체 블럭도

그림 6에는 CalmRISC와 보조 프로세서를 사용한 응용칩의 사진이 나와있다. 그림 6 (a)는 CalmRISC의 보조 프로세서 중의 하나인 MAC816을 사용한 발신자 표시(caller-id) 전화기용 칩이고[23], 그림 6 (b)는 24 비트 DSP 보조 프로세서인 MAC2424를 사용한 MP3 플레이어용 응용칩이다. CalmRISC는 모든 주변 회로의 컨트롤을 담당하고 DSP 보조 프로세서는 신호처리를 담당한다. 그림 6에서 보는 바와 같이 CalmRISC의 보조 프로세서 인터페이스를 통하여 MCU와 DSP의 자연스러운 통합을 이룰 수 있다.



(a) MAC816을 이용한 Caller-ID 칩



(b) MAC2424를 이용한 MP3 응용칩

그림 6 CalmRISC와 DSP 보조 프로세서의 사진

CalmRISC는 레지스터/메모리 타입의 명령어를 가진 RISC 아키텍처 MCU이다. 내장된 메모리를 포함한 전체 칩의 소모전력을 고려하여 CalmRISC의 3단계 파이프라인이 가장 적절한 파이프라인 구조라는 것을 보였다.

적은 전력 소모를 이루기 위해서 아키텍처 단계뿐만 아니라 회로설계 단계에서도 세심한 고려가 이루어졌다. CalmRISC에서 저전력을 위해서 사용된 몇몇 회로설계 방식을 보였다. 제안된 방식의 기본 원리는 불필요한 신호의 전이를 없애는 것으로 클락 전이 방지 및 선택적 입력 방식이 이 범주에 속한다.

CalmRISC는 0.5μm 표준 셀 라이브러리를 기반으로 설계되었으며 자동화 P&R(Place and Route) 프로그램을 사용하였다. 코아의 크기는 약 0.84 mm^2 정도이고 소모전력은 0.10mW(34μA per MIPS)이며 30MHz까지 동작 가능하다.

CalmRISC는 효율적이고 강력한 보조 프로세서 인터페이스를 가지고 있어 DSP와 쉽게 통합이 가능하다. 보조 프로세서와의 통신 부담을 줄이기 위해서 공유 데이타 메모리 구조와 파이프라인 동기식 방식

을 사용했으며 보조 프로세서의 상태를 주 프로세서에 효과적으로 전달 가능하다.

참고문헌

[1] M. Levy, DSP Architecture Directory, EDN magazine, May 1997, pp.43-107.

[2] M. Flynn, Computer Architecture, Jones and Bartlett publishers, 1995.

[3] 0.5m 5V/3.3V Standard Cell Library Data Book, Samsung Electronics Co., Ltd., 1996.

[4] Betty Prince, Semiconductor Memories: A Handbook of Design, Manufacture and Application, Wiley, 1997.

[5] C. Piguet et al., Low-Power Design of 8-b Embedded CoolRisc Microcontroller Cores, IEEE J. of Solid-State Circuits, Vol. 32, No 7, July 1997, pp. 1067-1078.

[6] D. Liu, C. Svensson, Trading Speed for Low Power by Choice of Supply and Threshold Voltages, JSSC-28, No 1, Jan. 1993, pp.10-17.

[7] J. Bunda, D. Fussell, R. Jenevein, 16-Bit vs. 32-Bit Instructions for Pipelined Microprocessors, Proc. 20th Annual Symposium on Computer Architecture, May 1993, pp. 237-246.

[8] M. Alidina, J. Monteiro, S. Devadas, Precomputation-Based Sequential Logic Optimization for Low Power, Proc. IEEE/ACM International Conf. On CAD-94, Nov 1994, pp. 74-81.

[9] Zilog Z89138/Z89139 Voice Processing Controllers Manual, Zilog Inc.(http://www.zilog.com).

[10] CalmRISC Technical Reference Manual, Samsung Electronics Co., Ltd., 1999.

[11] MAC816 Technical Reference Manual, Samsung Electronics Co., Ltd., 1999.

[12] AVR Enhanced RISC microcontroller, data book, May 1996, Atmel Corporation.

[13] N. Weste, K. Eshraghian , Principles of CMOS VLSI Design, 2nd Ed., Addison Wesley, 1993.

[14] A. Bellaouar, M. Elmasry, Low-Power Digital VLSI Design Circuits & Systems, Kluwer Academic Publishers, 1995.

[15] Jennifer Eyre, Jeff Bier, The Evolution of DSP Processors From Early Architectures to the Latest Developments, IEEE Signal Processing Magazine, March 2000.

[16] ARM9E-S Technical Reference Manual, ARM Limited, 1999.

[17] K.M.Lim, H.K.Kim, Y.C.Kim, S.W.Jeong, B.Y.Chung, H.L.Roh An Efficient Coprocessor Interface Scheme in CalmRISC, International Symposium on Low-Power and High-Speed Chips (Cool Chips II) , Kyoto, Japan, 1999.

[18] H.K.Kim, K.M.Lim, Y.C.Kim, S.W.Jeong, B.Y.Chung, H.L.Roh, A Customizable DSP Coprocessor Extension in CalmRISC Microcontroller, International Conference on Signal Processing and Application Technology, Orlando, Florida, USA, 1999.

[19] K.M.Lim, S.W.Jeong, etc., CalmRISC: A Low Power Microcontroller with Efficient Coprocessor Interface, International Conference on Computer Design, Austin, Texas, USA, 1999.

[20] D.-H Kim, Advanced Compiler Optimization for CalmRISC8 Low-End Embedded Processor, the 9th Int. Conference on Compiler Construction (CC 2000), pp 173-188, Berlin, Germany, Springer-Verlag, 2000.

[21] A.V. Aho, R.S. Sethi, J.D. Ullman, Compilers: Principles, Techniques and Tools, Addison-Wesley, 1985.

[22] C. Fraser and D. Hanson, A Retargetable C Compiler: Design and Implementation, the Benjamin/Cummings, Redwood City, CA, 1995

[23] S3FB42F 8-Bit CMOS Microcontroller Users Manual, Samsung Electronics Co., Ltd., 2001.

[24] S3FB41D(KS85F40113) Microcontroller Users Manual, Samsung Electronics Co., Ltd., 2000.

임 경 묵

1989 서울대학교 물리학과
1991 KAIST 전산학과 석사
2002 KAIST 전산학과 박사
2002~현재 삼성전자 SOC 연구소 책임 연구원으로 재직중
관심분야:MCU, DSP를 포함한 프로세서 구조와 MPEG, DVD, Multimedia 등
E-mail:lracer@samsung.co.kr

• 2002 컴퓨터비젼및패턴인식 워크샵 •

- 개최일자 : 2002년 11월 9일(토)
- 개최장소 : 숙명여자대학교
- 논문접수마감 : 2002년 10월 12일(토)
- 상세정보 : http://cs.sookmyung.ac.kr/~cvpr02f
- 논문제출문의 : 전북대 오일석 교수
  E-mail:isoh@moak.chonbuk.ac.kr
  Tel. 063-270-3401
- 기타문의 : 숙명여대 최영우 교수
  E-mail:ywchoi@sookmyung.ac.kr
  Tel. 02-710-9763

• SIGPL 2002 학술발표회 및 정기총회 •

- 개최일자 : 2002년 11월 9일(토)
- 개최장소 : 한양대학교(안산)
- 상세정보 : www.sigpl.or.kr