컴퓨터 연산

<3.4 나눗셈>

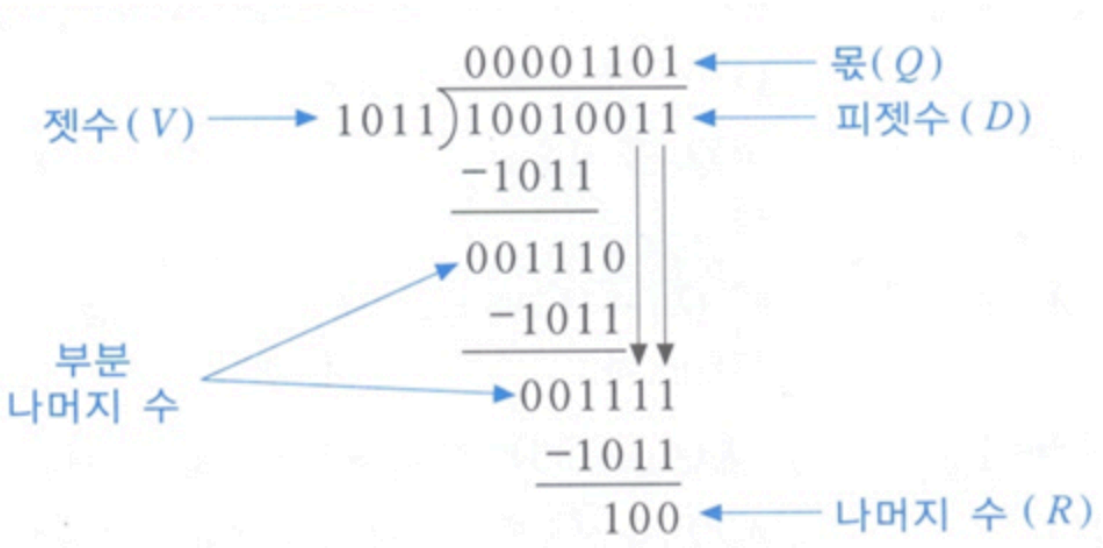
피제수(dividend), 제수(devisor) : 피연산자

몫(quotient)

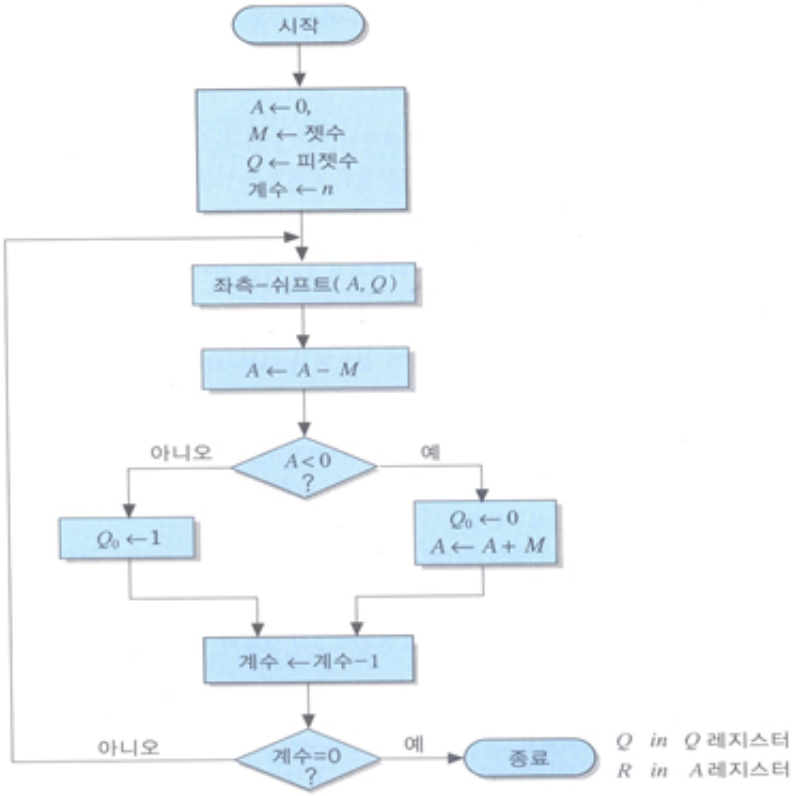
나머지(remainder)

피제수 = 몫 X 제수 + 나머지

<부호 없는 2진 나눗셈>



- 뺄셈과 동시에 피연산자와 몫을 자리이동시키면 성능 향상이 가능함.



- 부호 있는 정수 나눗셈을 할 때, 피제수와 제수의 부호가 다르면 몫을 음수로 바꿔 간단하게 연산할 수 있음.

  ex) +7 / -2 : 몫 = -3, 나머지 = +1

  -7 / -2 : 몫 = 3, 나머지 = -1

  => 몫 4, 나머지 +1이 성립은 하지만,

  피제수와 나머지의 부호는 항상 같아야 한다는 규칙을 따르고 있음.

  => 부호와 관계없이 몫과 나머지의 절대값은 같음.

- 나눗셈은 알고리즘의 다음 단계를 수행하기 전에 뺄셈한 결과의 부호를 알아야 하기 때문에 곱셈과 같이 여러 부분을 바로 계산할 수 없다. 따라서, 하드웨어를 추가하는 방법만으로는 성능을 향상시킬 수 없다.

- 더 빠른 나눗셈을 위해서 SRT 나눗셈을 사용하기도 한다.

  : 한 단계에서 몫을 두 비트 이상 만들 수 있는 기술

  : 각 단계에서 여러 개의 몫 비트를 예측하는 기법을 사용

  : 한 번에 4비트를 추측하는 것을 주로 사용함.

  : 뺄 값을 추측하고 틀리면 그 후 단계에서 수정함.

<3.6 병렬성과 산술연산: 서브워드 병렬성>

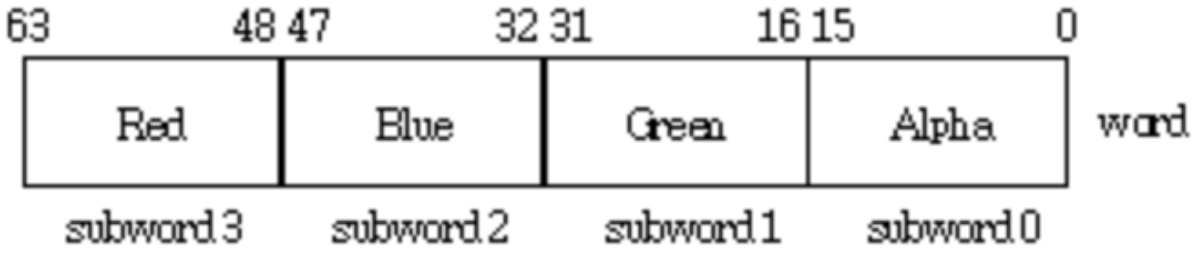
=> 모든 마이크로프로세서는 바이트나 하프워드가 메모리에 저장될 때, 공간을 덜 차지하도록 지원함.

=> 일반적인 정수 프로그램은 데이터의 크기가 보통 작아, 데이터 전송 이상의 지원은 하지 않았음.

=> 그래픽과 오디오 응용 프로그램이 데이터의 벡터에 같은 연산을 반복 수행한다는 것을 알게됨.

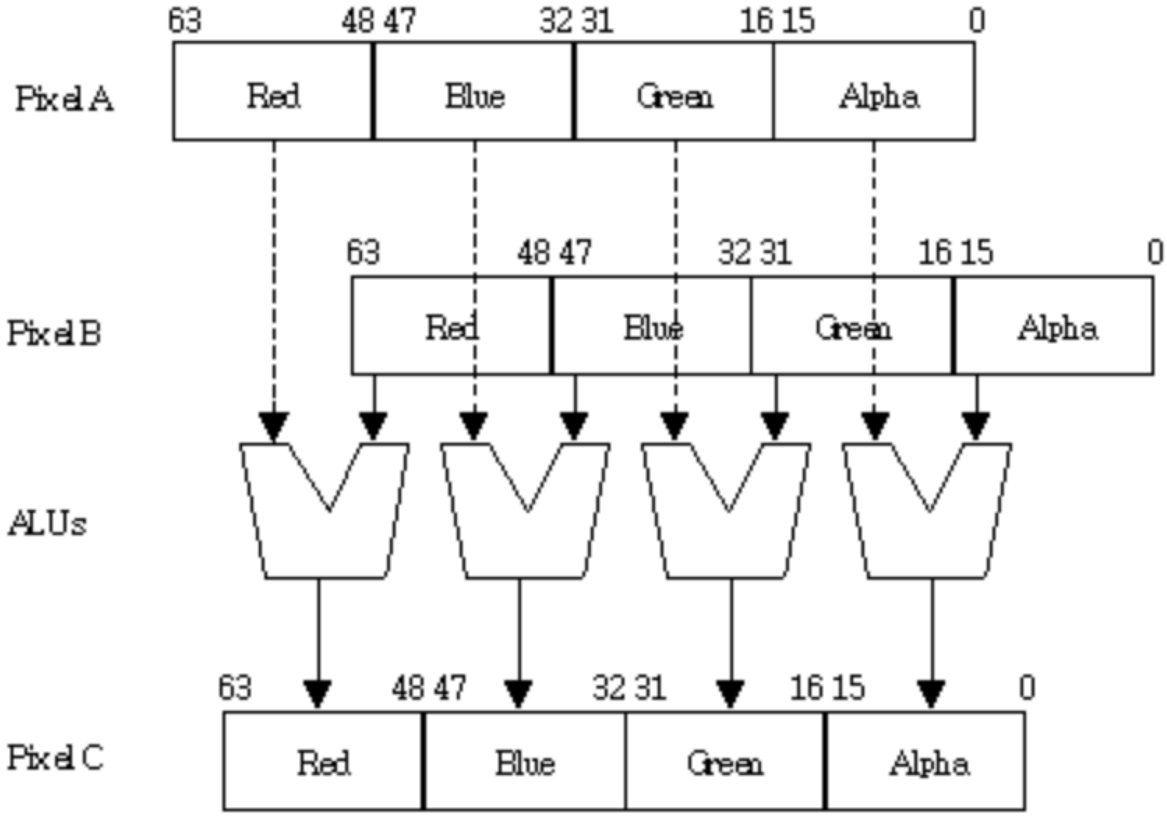
=> 멀티미디어 응용이 많아지면서 작은 데이터에 대한 병렬 연산을 지원하는 산술 명령어가 등장하게 됨.

- 비트 덧셈기 내부의 올림수 체인을 분할하면 프로세서가 병렬성을 활용하여 동시에 연산할 수 있다.
- 서브워드 병렬성, 데이터 수준 병렬성, 벡터, SIMD(single instruction, multiple data)
  : 워드 내부의 병렬성



(그림 1) 워드와 서브워드(픽셀 데이터의 경우)

** **서브워드** : 하나의 워드 안에 들어가는 각각의 다른 데이터



(그림 2) 서브워드 병렬 연산(픽셀 데이터 연산의 경우)

** **서브워드 병렬처리** : 한 워드를 복수 개의 서브워드로 나누고, 각 서브워드별로 복수 개의 연산을 한 번에 처리하는 방식. 기본적으로 동일한 크기의 서브워드를 동시에 처리함.