주제/단락	내용
논리적 시프트 정의	레지스터의 비트를 좌우로 한 칸 이동시키는 동작으로 데이터 배열을 바꾼다.
좌측 시프트 LSL	모든 비트를 왼쪽으로 한 칸 이동하고 최하위 비트에는 0을 넣으며 최상위 비트는 버린다.
우측 시프트 LSR	모든 비트를 오른쪽으로 한 칸 이동하고 최상위 비트에는 0을 넣으며 최하위 비트는 버린다.
시프트 레지스터 회로	D 플립플롭을 직렬로 연결해 클록마다 입력 D가 저장되고 출력 Q로 전달되며 이전 값은 다음 단계로 이동한다.
순환 시프트 정의	최상위(또는 최하위) 비트를 버리지 않고 반대편 끝으로 이동시키는 회전 동작이다.
순환 좌측 시프트	최상위 비트가 최하위 비트 위치로 이동한다.
순환 우측 시프트	각 비트가 오른쪽 이웃 자리로 이동하고 최하위 비트는 최상위 자리로 돌아간다.
직렬 데이터 전송	시프트를 연속 수행해 두 레지스터 사이를 한 선으로 전체 데이터를 전송한다.
직렬 전송 예시	A=1011을 4번의 시프트로 B로 이동시키고 A는 원래 값을 유지하며 B의 초기값은 0000으로 가정한다.
산술적 시프트 정의	부호 있는 수에 대해 부호 비트는 보존하고 크기 부분만 이동시키는 시프트이다.
산술 좌측 시프트	부호 비트는 유지하고 나머지 비트를 왼쪽으로 이동하며 최하위 비트에는 0을 넣는다.
산술 우측 시프트	부호 비트를 오른쪽으로 복사해 부호-비트 확장을 하며 크기 비트를 오른쪽으로 이동한다.
산술 시프트 예	−2의 4비트(1110)를 좌측 시프트하면 −4(1100), 이어서 우측 시프트 두 번은 −2, −1이 된다.
시프트의 산술 활용	2의 거듭제곱 곱셈·나눗셈을 빠르게 수행하며 LSL은 ×2^k, LSR은 ÷2^k 효과가 있다(오버플로·정보손실 없을 때).
시프트 간단 예	0010을 LSL→0100은 ×2, 0010을 LSR→0001은 ÷2에 해당한다.
24×n 시프트 구현	n을 4비트 LSL해 16n, 3비트 LSL해 8n을 만든 뒤 두 값을 더해 24n을 계산한다.
시프트+덧셈 장점	곱셈기보다 하드웨어가 단순하고 빠르므로 성능 최적화에 널리 사용된다.

주제/단락	내용
캐리 포함 시프트	SHLC, SHRC는 캐리를 포함한 좌·우 시프트, RLC, RRC는 캐리를 포함한 좌·우 회전이다.
산술 우측 시프트와 C	산술 우측 시프트에 C가 개입해 부호가 바뀌지 않도록 V 플래그로 알리거나 C가 레지스터로 유입되지 않게 처리한다.
2의 보수 덧셈 규칙	두 수를 더한 뒤 최상위 캐리가 발생하면 버린다.
2진·10진 덧셈 비교	(+3)+(+4), (−3)+(+3), (−6)+(+2), (−4)+(−1) 등 사례로 2진 덧셈과 10진 덧셈 결과를 대응시켜 이해한다.
반가산기 정의	1비트 두 입력의 합과 캐리를 출력하는 조합논리 회로이다.
전가산기 정의	A, B와 하위 자리 캐리 Cin을 포함해 1비트 세 값을 더하는 조합논리 회로이다.
덧셈 하드웨어 구성	비트 수만큼의 전가산기를 직렬 연결해 다비트 덧셈을 수행한다.
연산 플래그 설정	V는 오버플로, Z는 결과가 0, S는 부호, C는 최상위 자리 캐리를 나타내며 Z는 OR 후 반전으로 생성하고 C와 V를 구분한다.
덧셈 오버플로 검출	최상위·차상위 캐리의 XOR로 V=C4 XOR C3을 계산한다.
덧셈 오버플로 예	4비트에서 (+6)+(+3), (−7)+(−6) 등의 경우에 오버플로 발생 여부를 확인한다.
오버플로 발생 조건	피연산자 부호가 다르면 발생하지 않고 둘 다 양수인데 차상위 캐리가 1이면, 둘 다 음수인데 차상위 캐리가 0이면 발생한다.
뺄셈의 덧셈화	A−(+B)=A+(-B), A−(−B)=A+(+B)로 감수의 2의 보수를 취해 덧셈으로 수행한다.
뺄셈 예시	2의 보수를 이용해 (+2)−(+6), (7)−(+4) 등의 계산을 수행한다.
뺄셈 오버플로	덧셈과 동일하게 V=C4 XOR C3으로 검출하며 4비트에서 (+6)−(−4), (−7)−(+6) 등을 점검한다.
4비트 가감산기	M=0이면 A+B, M=1이면 A−B를 수행하며 B를 반전하고 최하위 캐리 C0에 M을 입력한다.
곱셈 원리	각 비트에 대한 부분 적을 생성해 모두 더해 결과를 얻는다.
곱셈 예시	1101×1011에서 부분 적을 더해 최종 결과가 13×11=143과 일치함을 확인한다.
곱셈기 하드웨어	M에 피승수, Q에 승수, 결과는 A와 Q에 저장하며 Q0=1이면 A←A+M 후 C-A-Q를 우측 시프트, Q0=0이면 시프트만 한다.
부스 알고리즘 구성	M과 가산기 사이에 보수기를 추가하고 Q의 오른쪽에 1비트 레지스터 Q−1을 둔다.

주제/단락	내용
부스 알고리즘 규칙	Q0Q−1이 00·11이면 A,Q,Q−1 산술 우측 시프트, 01이면 A←A+M 후 시프트, 10이면 A←A−M 후 시프트한다.
부스 예시 개요	−7×3에서 초기 A=0000, Q=0011, Q−1=0, 계수 4로 시작해 Q0Q−1 패턴에 따라 연산·시프트를 반복한다.
블록 최적화 개념	0으로 둘러싸인 1의 블록을 덧셈·뺄셈 두 번으로 대체해 연산 횟수를 줄이며 (...01...10...)는 (...10...00...)−(...00...10...)로 표현된다.
블록 최적화 예	M×00111110은 M×(2^6−2^1)=M×62, M×00111010은 M×(2^6−2^3+2^2−2^1)=M×58로 계산한다.
블록 처리 규칙	1의 블록 시작(01)에서 덧셈, 끝(10)에서 뺄셈을 수행할수록 연산 횟수가 감소한다.
나눗셈 식 표현	A÷B=q ... r로 A는 피제수, B는 제수, q는 몫, r은 나머지이다.
부호 없는 나눗셈 절차	좌측 시프트 후 A가 M보다 크거나 같으면 A←A−M, Q0←1을 수행한다.
나눗셈 단계 예시	A,Q를 반복 좌측 시프트하고 필요 시 제수를 빼서 Q0를 1로 세트하는 과정을 표로 진행한다.
7÷(−3) 예시	제수 −3의 2의 보수 1101을 M에 저장하고 규칙에 따라 연산을 순차 수행한다.
부호 없는 곱셈 규칙	Q0=1이면 A←A+M 후 우측 시프트, Q0=0이면 시프트만 수행한다.
부호 있는 곱셈 규칙	부스 알고리즘으로 Q0Q−1=00·11은 시프트만, 01은 A←A+M, 10은 A←A−M 후 산술 우측 시프트한다.
결과 저장 폭	2N비트 결과를 A와 Q 레지스터에 나눠 저장한다.
부호 없는 나눗셈 규칙	좌측 시프트를 수행하고 A≥M이면 A←A−M, Q0←1로 설정한다.
부호 있는 나눗셈 규칙	좌측 시프트 후 A와 M의 부호가 같으면 A←A−M, 다르면 A←A+M을 수행하고 연산 전후 부호가 다르고 0이 아니면 A를 복구하며 그렇지 않으면 Q0←1로 세트한다.
몫·나머지 저장	A 레지스터에는 나머지, Q 레지스터에는 몫을 저장하며 부호가 다른 나눗셈은 Q에 2의 보수를 취해 부호를 보정한다.