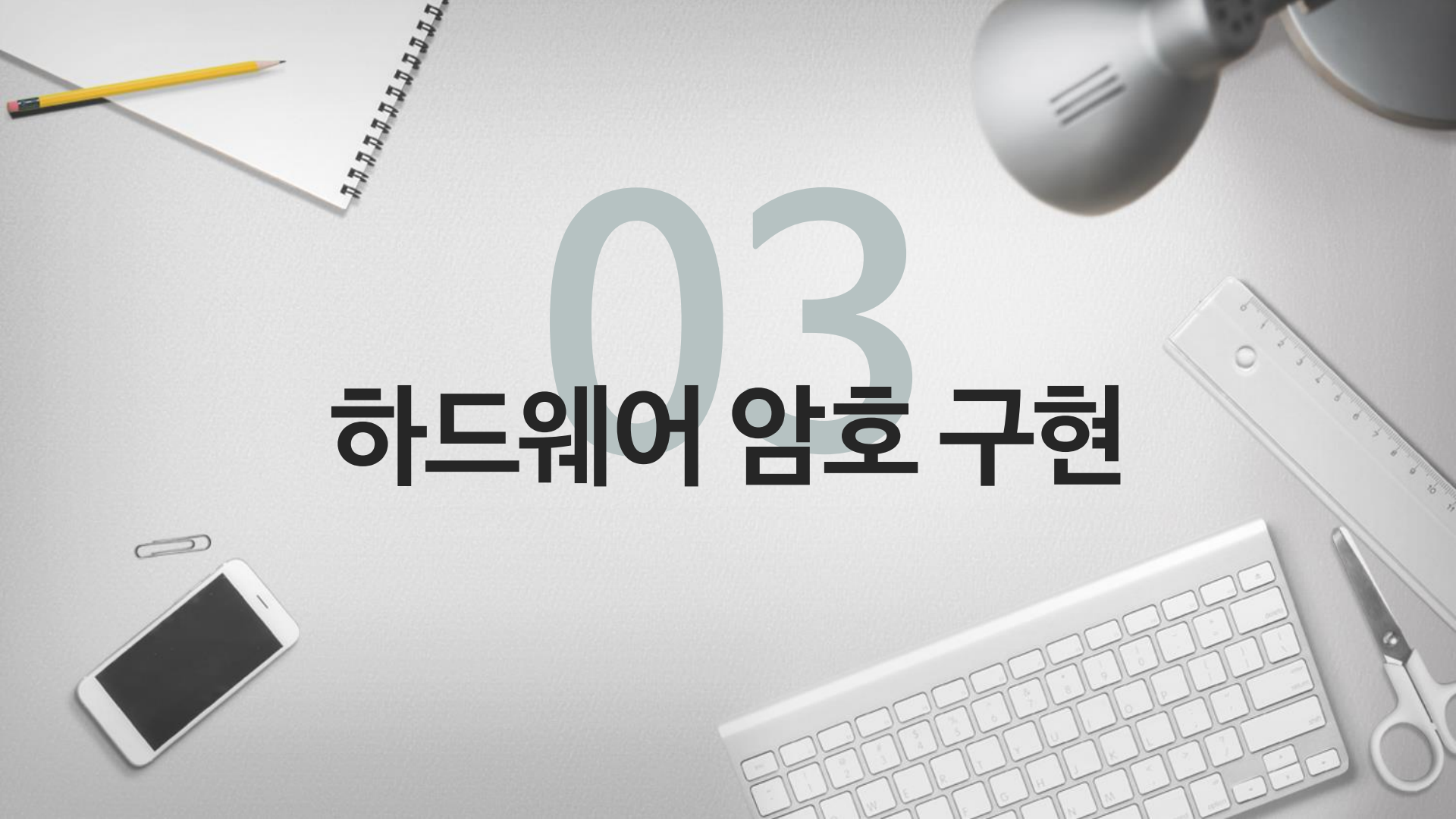

03
하드웨어 암호 구현

소프트웨어 vs 하드웨어

소프트웨어 프로그래밍 vs 하드웨어 프로그래밍

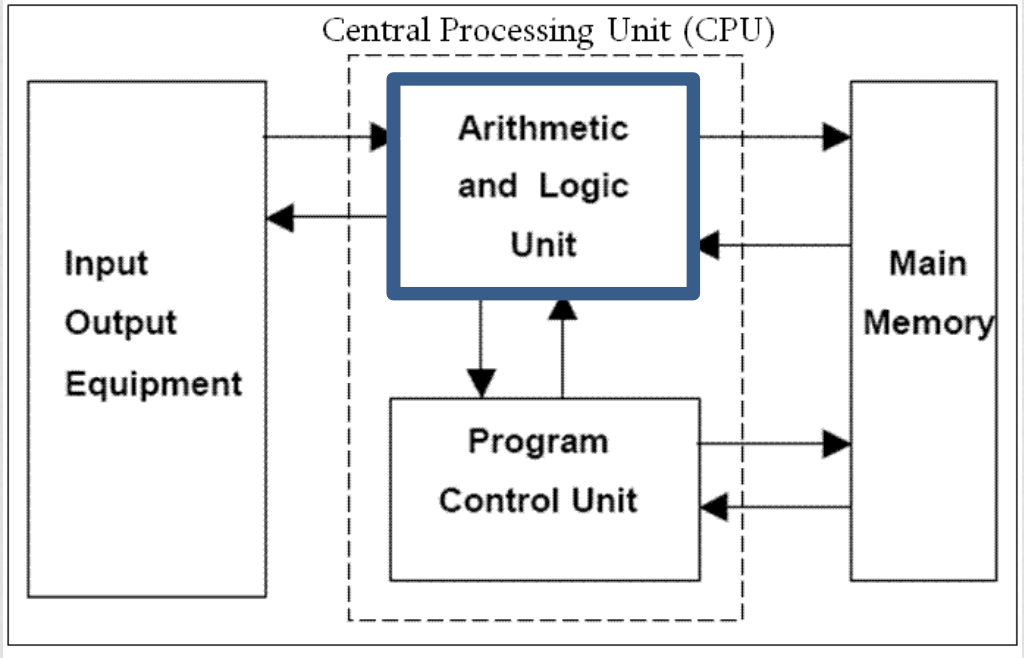
C 언어 (소프트웨어 프로그래밍)	Verilog (하드웨어 프로그래밍)
상위 레벨 프로그래밍 언어	하드웨어 정의 언어
순차적인 프로그램을 주로 수행	순차/병렬 프로그램 수행
논리적 명령어로 구성	하드웨어 회로에 대한 지식 필요

하드웨어 기반 암호 구현

특징	내용
속도	고속 구현, 주파수, 에너지
하드웨어 면적	초소형 칩 (RFID, 스마트 카드), 에너지
병렬 처리	파이프 라이닝, 멀티 코어 (ALU)

컴퓨터 구조 (ALU)

ALU는 기본적인 사칙연산부터
복잡한 머신러닝 연산까지 수행



고전 컴퓨터의 기본 요소 (게이트)

기본 게이트 (NAND)

- 모든 게이트는 NAND 게이트로 구성 가능

Q = **A** NAND **B**

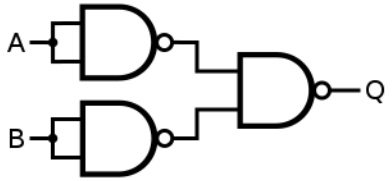
Truth Table

Input A	Input B	Output Q
0	0	1
0	1	1
1	0	1
1	1	0

Desired OR Gate

Q = **A** OR **B**

NAND Construction

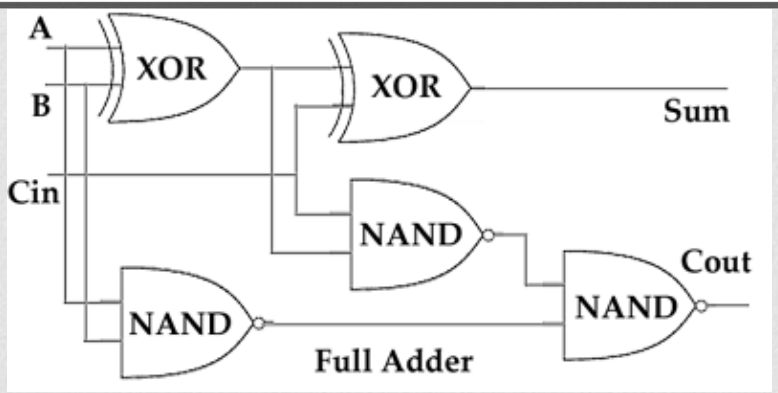


= (**A** NAND **A**) NAND (**B** NAND **B**)

Truth Table

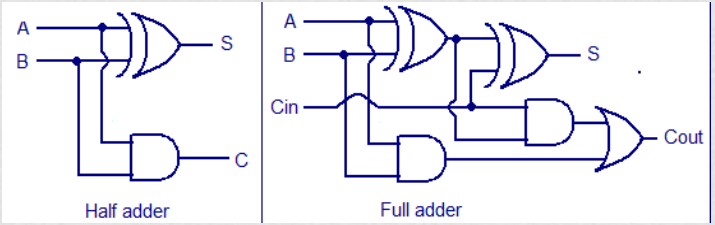
Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

기본 게이트를 활용한 연산자 구현 (Full Adder)



Input			Output	
A	B	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

기본 게이트를 활용한 연산자 구현 (Multiplier)



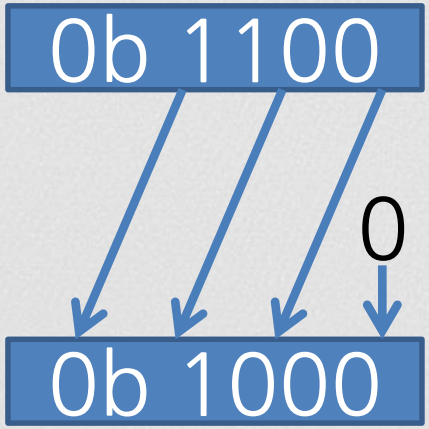
Carry가 없는 경우

- Half Adder
- Full Adder

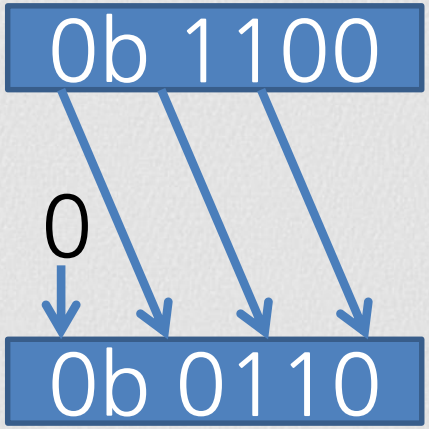
Carry가 있는 경우

2의 승수에 대한 곱하기 및 나누기

곱하기 2



나누기 2

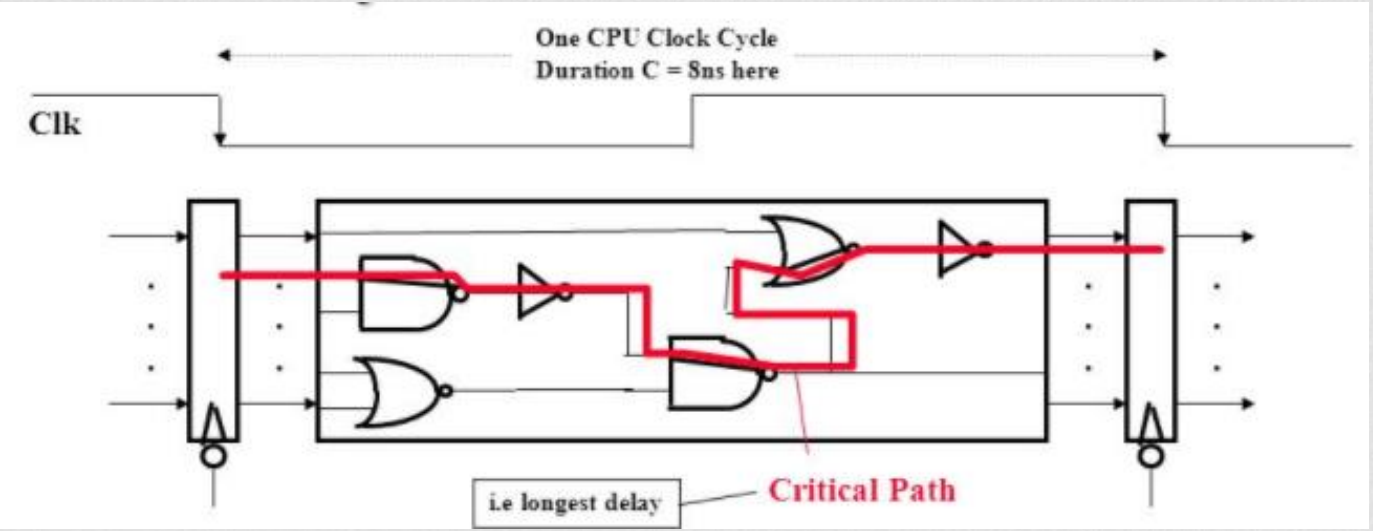


와이어링 (회로 연결)으로
곱하기 및 나누기 구현 가능

Critical Path

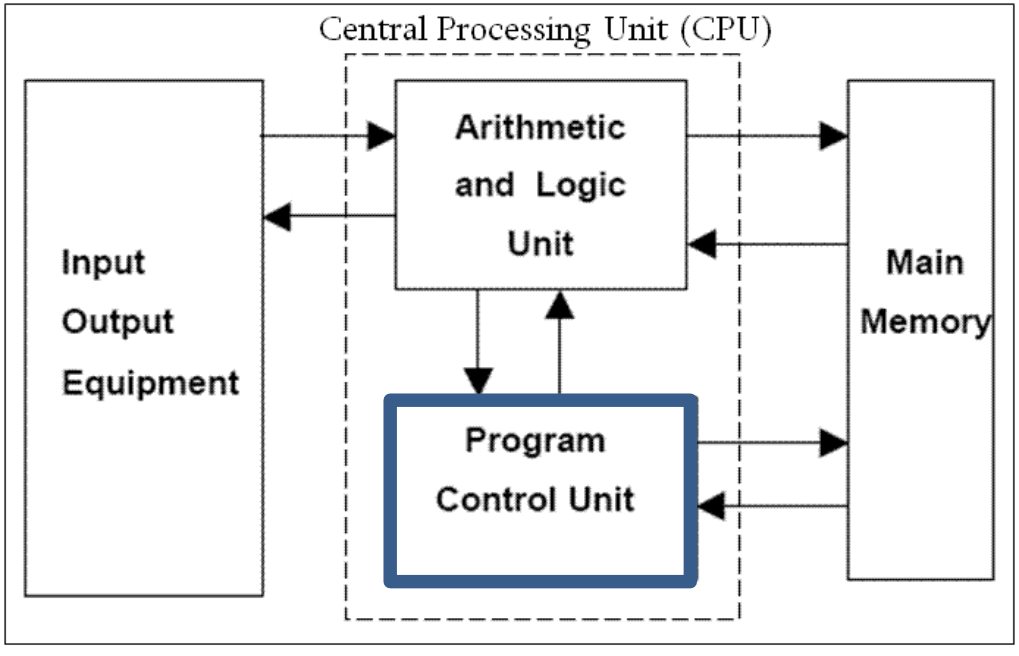
Flipflop을 중간에 삽입하여 Path를 줄임

- Flipflop은 정보가 잠시 쉬어가는 (저장되어 있는) 휴게소 (저장공간)



컴퓨터 구조 (Control Unit)

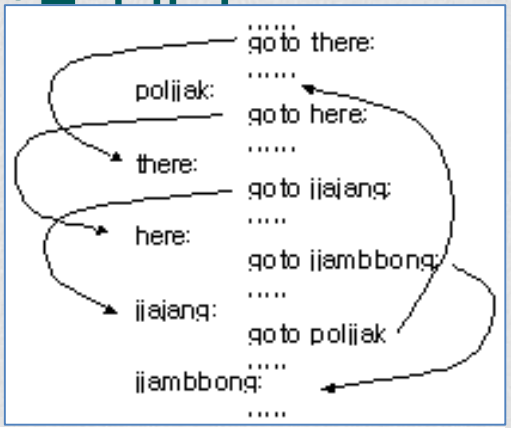
- Control Unit에서는 전체 하드웨어를 스캐쥴링하는 역할 수행

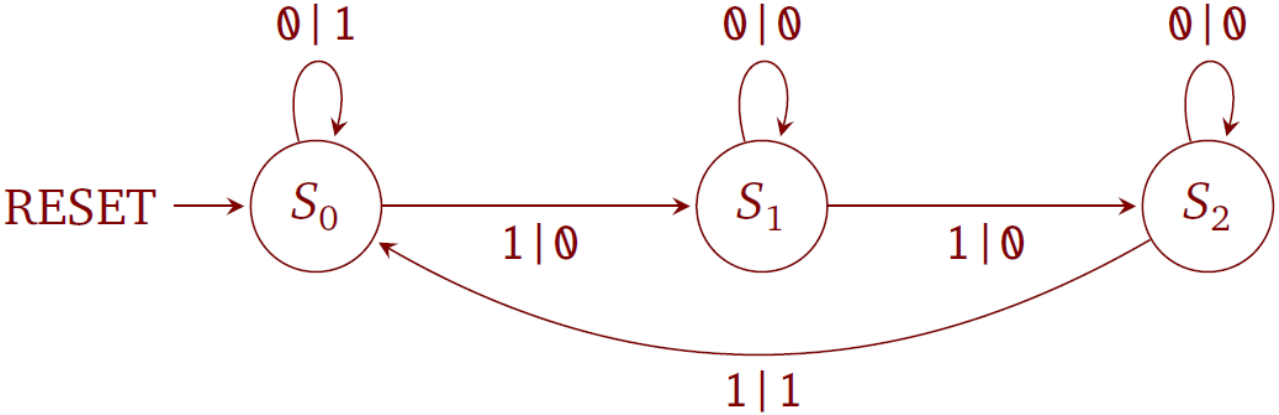


Finite State Machine (오토마타)

하드웨어 구현은 소프트웨어의 goto문과 유사

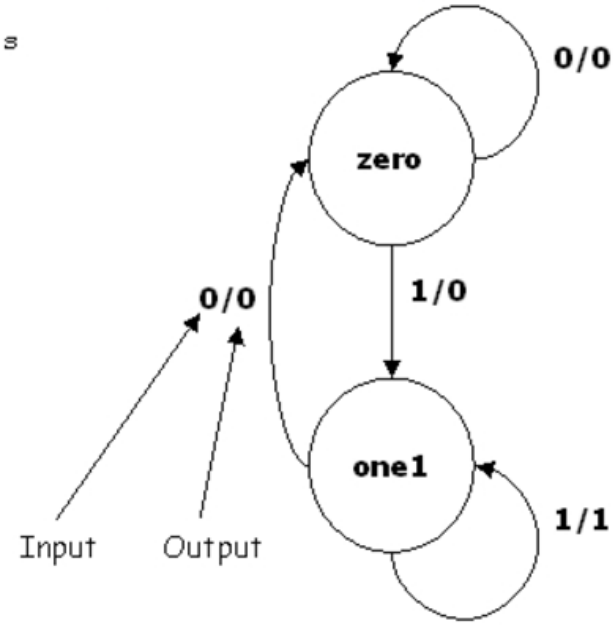
- 하드웨어 상태 (label)에 따라 수행되는 연산 상이
- 상태 전이는 개발자에 의해 자유롭게 설정 가능





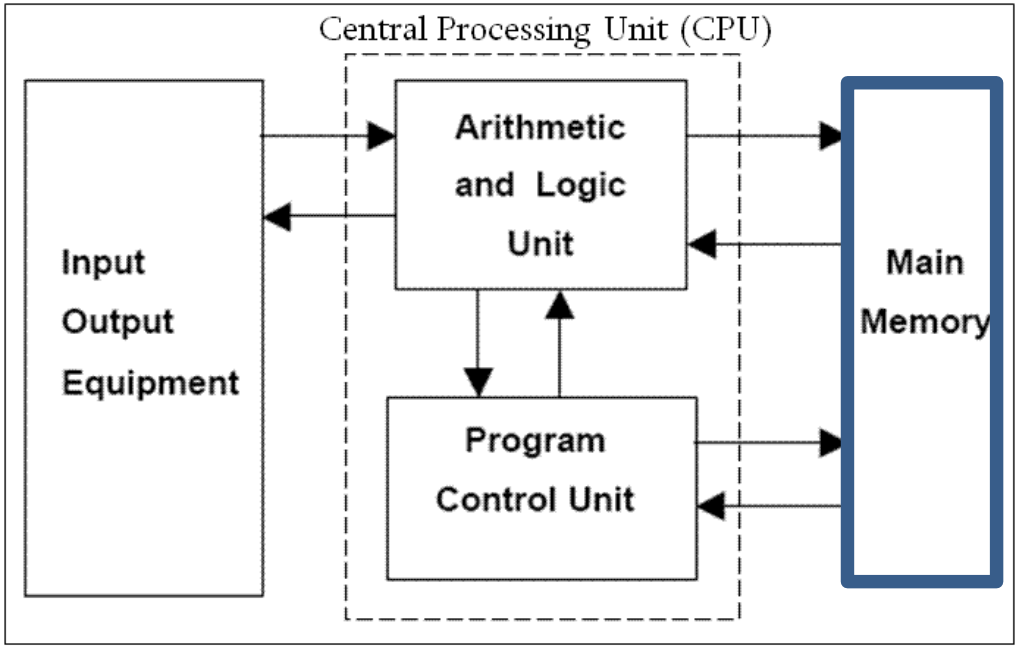
Finite State Machine과 하드웨어 매핑

```
module reduce (CLK, reset, in, out);
  input CLK, reset, in;
  output out;
  reg out;
  reg state;              // state variables
  reg next_state;

  always @(posedge CLK)
    if (reset) state = `zero;
    else       state = next_state;

  always @(in or state)
    case (state)
      `zero: // last input was a zero
      begin
        out = 0;
        if (in) next_state = `one;
        else    next_state = `zero;
      end
      `one: // we've seen one 1
        if (in) begin
          next_state = `one; out = 1;
        end else begin
          next_state = `zero; out = 0;
        end
    endcase
endmodule
```



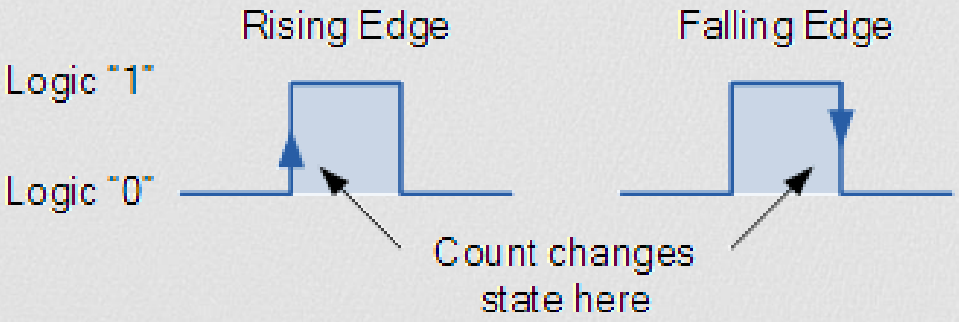
컴퓨터 구조 (Main Memory)

대용량의 정보를 저장하는 하드웨어 공간



Rising Edge / Falling Edge

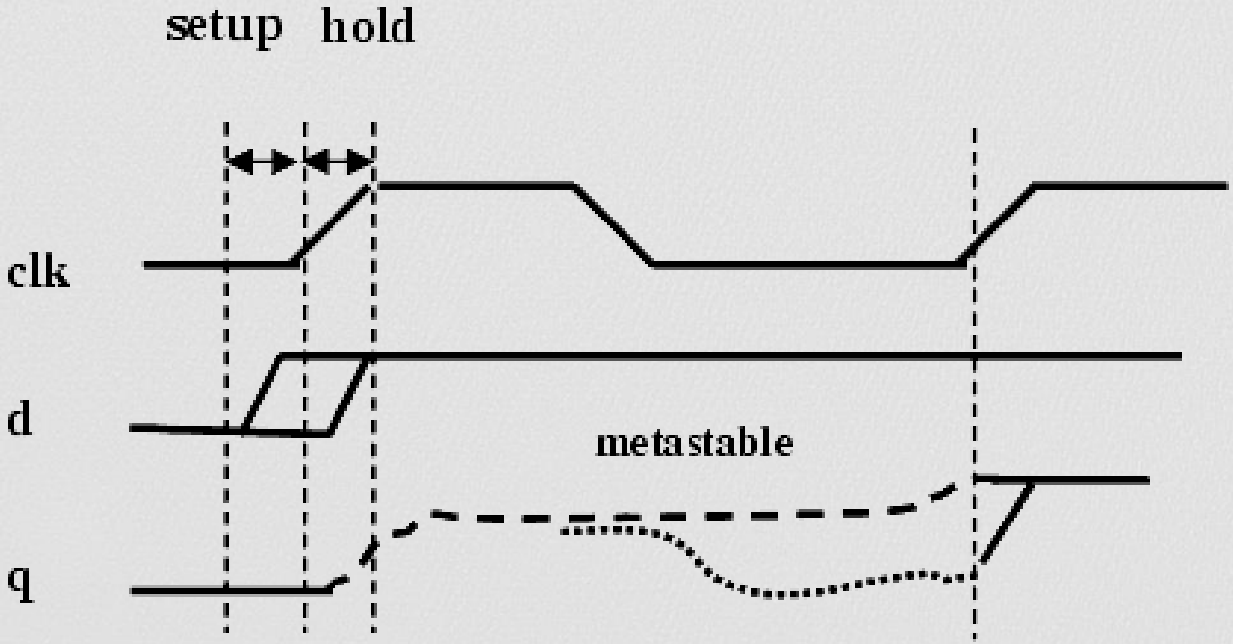
메모리에 정보 접근은 매우 짧은 그리고 특수한 경우에만 가능



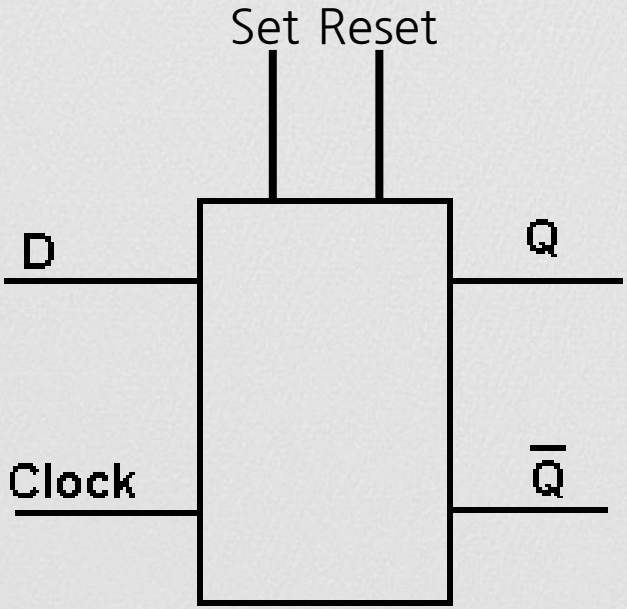
Metastable

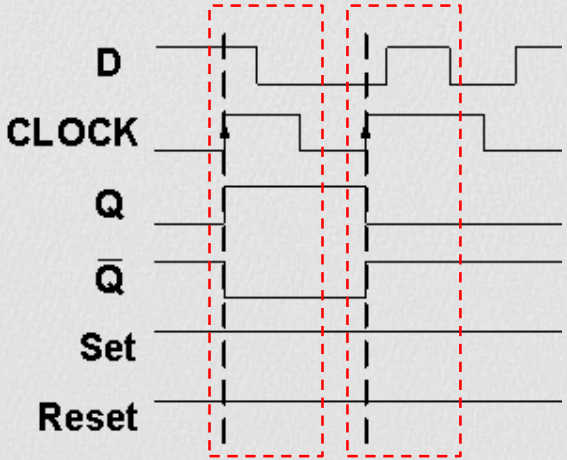
메모리의 특정 조건에서 예측 불가능한 값 도출

- Setup과 Hold 기간에서 일정한 시간 동안 값 안정화 필요

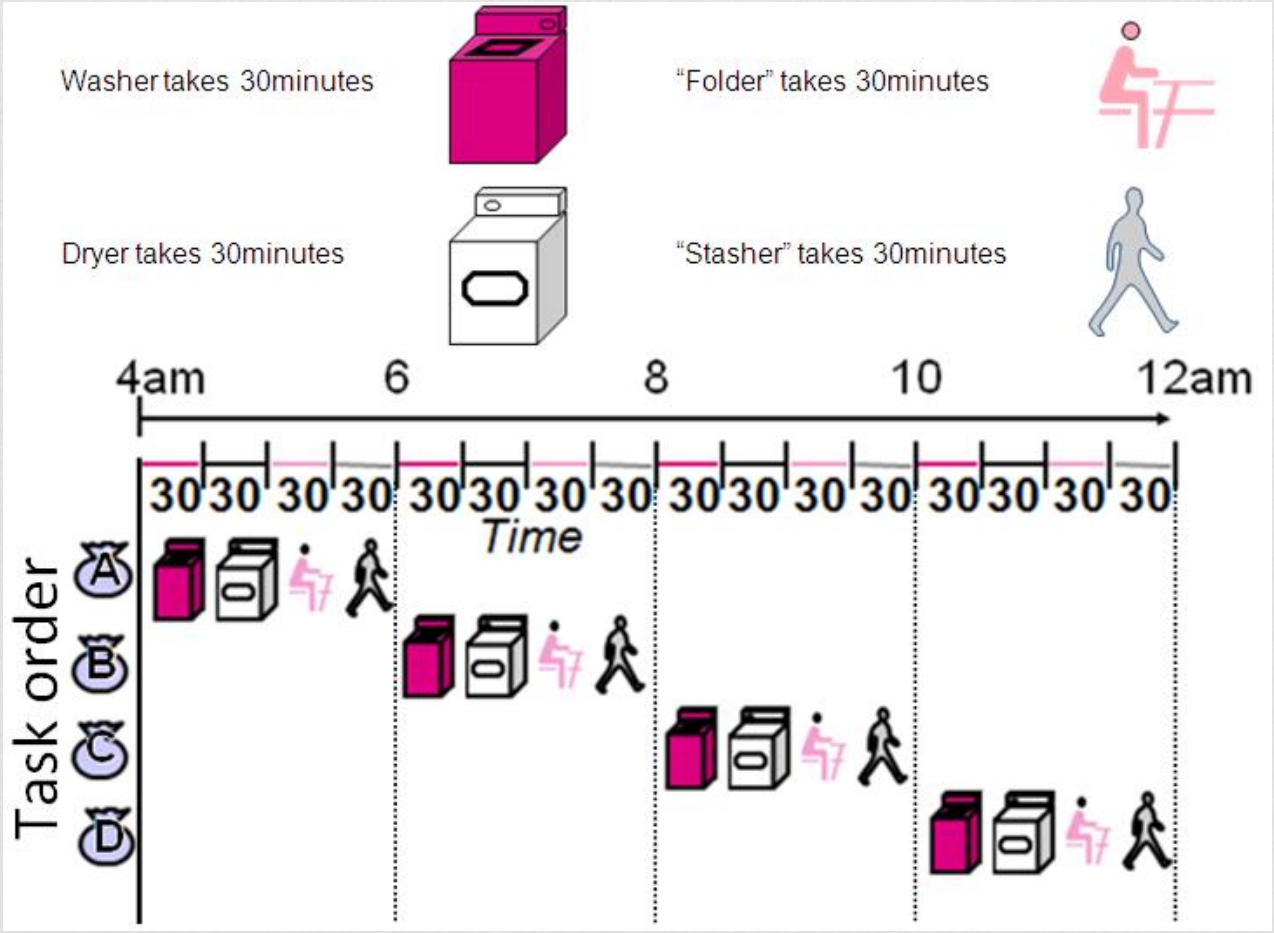


저장 장치 (D Flip Flop)



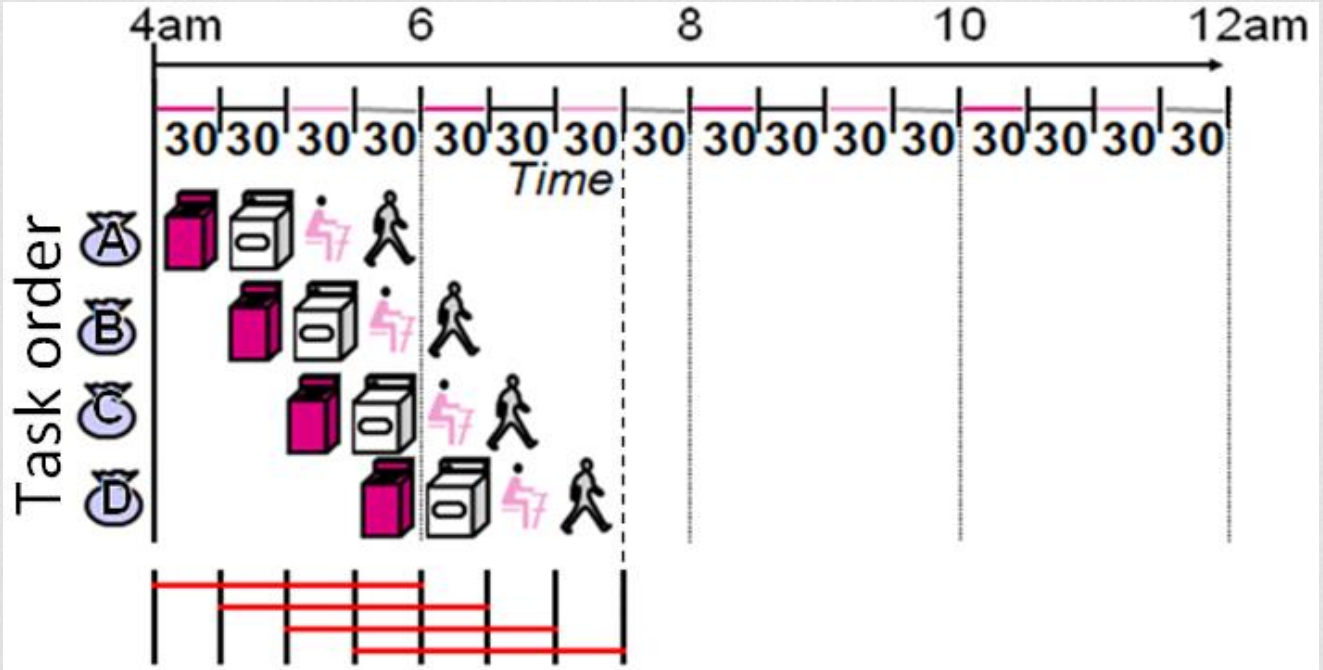


파이프라이닝이 적용되지 않은 경우



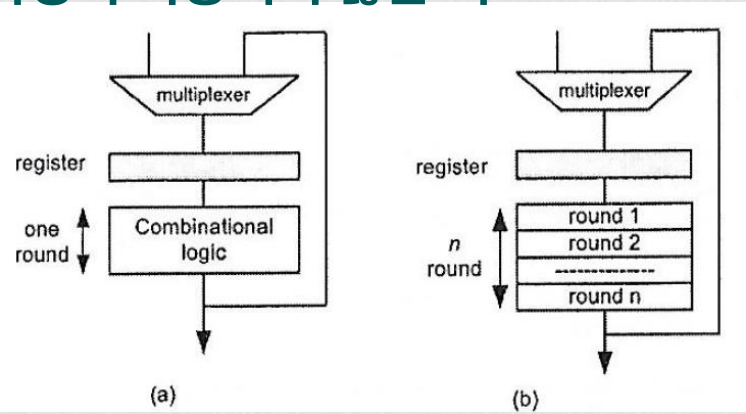
파이프라이닝이 적용된 경우

- 세탁, 건조, 정리, 수납과 같은 업무 병렬화 가능

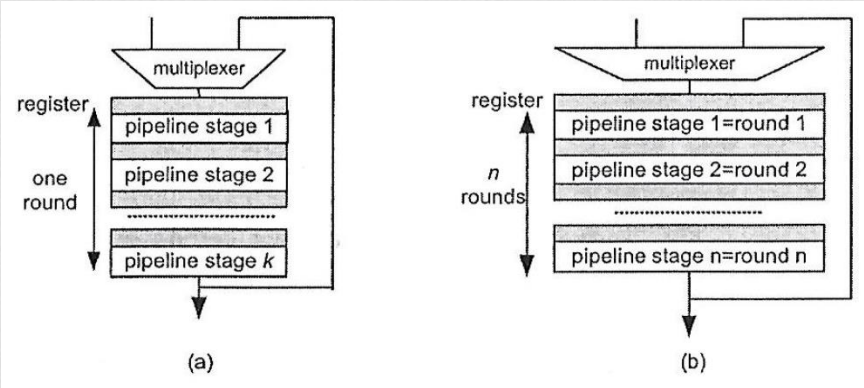


파이프라이닝과 회로

파이프라이닝이 적용되지 않은 회로



파이프라이닝이 적용된 회로



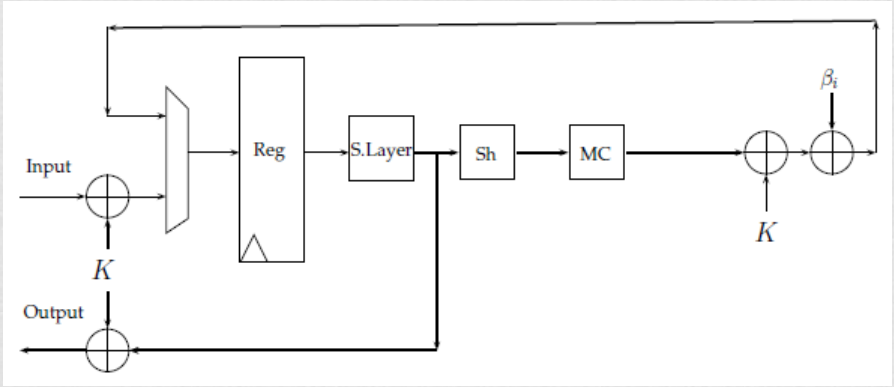
칩 면적 최소화

하드웨어 자원의 공유는 칩 면적 축소에 효과적임

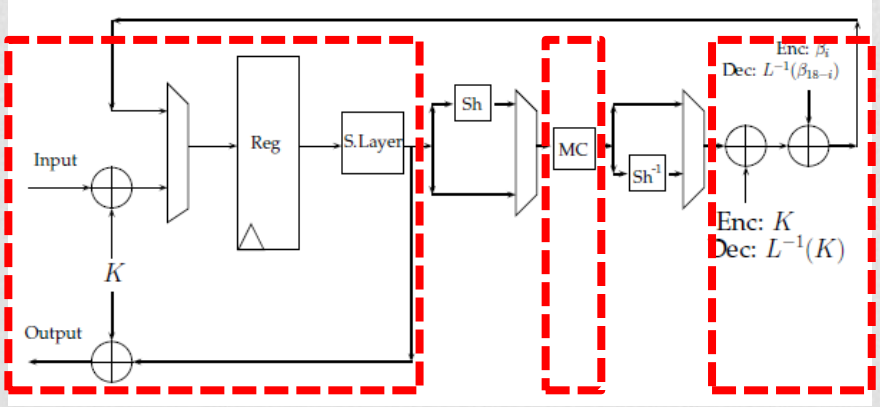
- 특정한 자원에 대한 스캐쥴링을 통해 자원 공유 최적화 가능
- 다만 자원 재사용 시 항상 칩크기가 줄어드는 것은 아님
- 자원 재활용 (스캐쥴링)을 위해서는
  MUX가 필요한데 해당 MUX가 공유하는 연산자보다
  복잡한 경우 칩크기가 오히려 증가
- RFID 태그의 경우 2,000GE 이하가 암호화 칩에 적당한 크기

칩 면적 최소화

암호화와 복호화 연산에서 공유되는 부분 최대화



암호화 회로



암호화와
복호화 회로

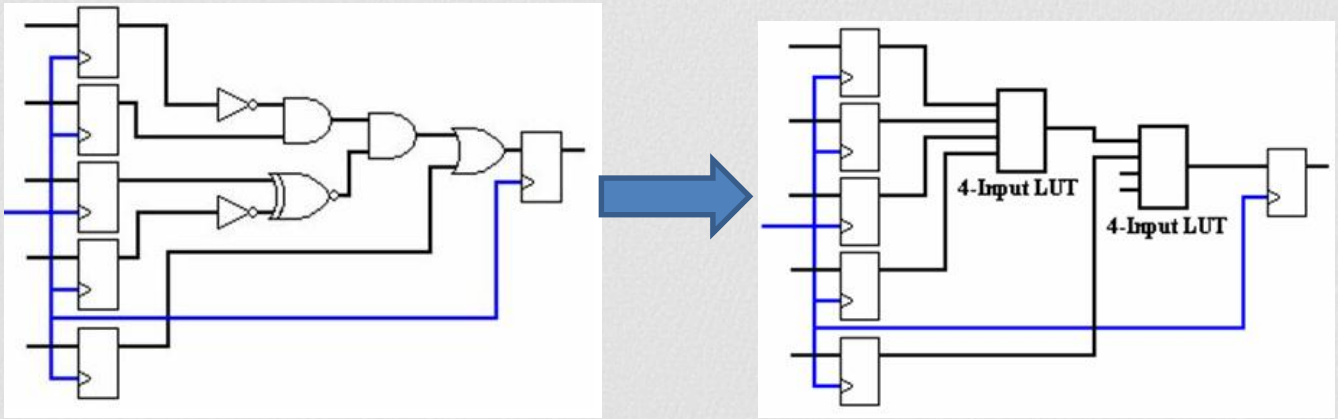
구현 플랫폼

FPGA	ASIC
저렴한 비용	비싼 가격
빠른 구현 및 테스트 가능	연산속도가 FPGA보다 빠름
내부 하드웨어 디자인이 간단함	전력소모가 적음
간단한 프로토타입으로 활용 가능	아날로그 회로와 혼합가능

FPGA 회로 구성

연산자를 사전테이블로 변환

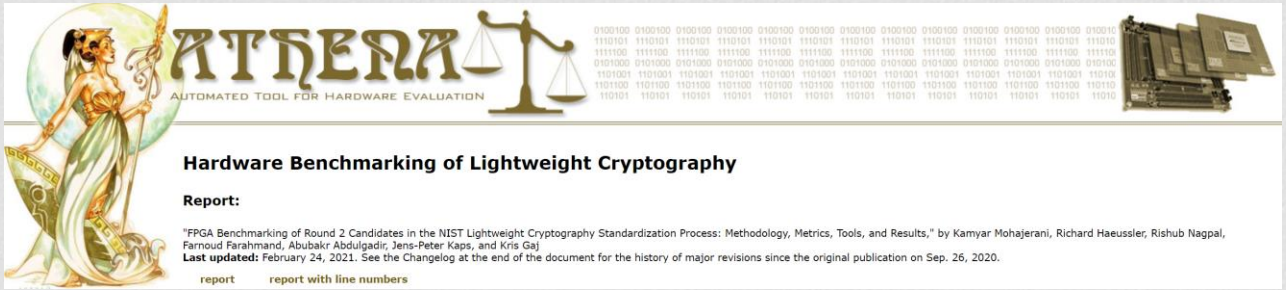
- 모든 함수는 조건문 (사전테이블)을 통해 표현 가능



하드웨어 벤치마크

NIST 경량암호공모전 후보군에 대한 FPGA 벤치마크 플랫폼

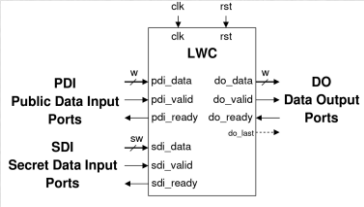
- FPGA 상에서 암호 구현을 공평하게 비교하기 위한 용도로 개발
- FPGA 상에서 LWC 암호 군들의 순위 확인 용이



하드웨어 벤치마크

해당 벤치마크를 위한 LWC 하드웨어 API

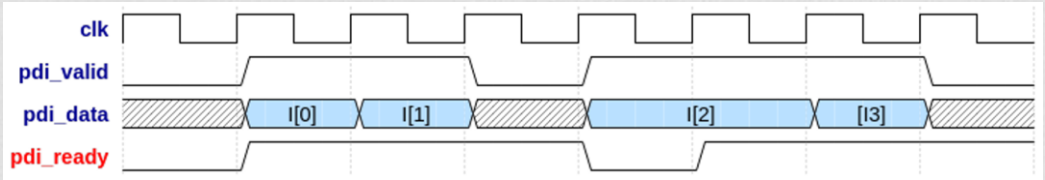
- 최소한의 준수 요건: 제공되는 연산자, 허용되는 입력길이
- 인터페이스



- 통신 프로토콜

instruction = ACTKEY
instruction = ENC
seg_0_header
seg_0 = Npub
seg_1_header
seg_1 = AD
seg_2_header
seg_2 = Plaintext

- 타이밍 특성



하드웨어 벤치마크

해당 벤치마크 기준

- 자원 활용성: LUT, 플립플롭, 특수 하드웨어는 사용하지 않음 (BRAM, DSP)
  - 최대 2,000 LUT, 4,000 플립플롭 사용 (Artix-7 FPGA 기준)

- 최대 클락 주파수 (최대 throughput 계산에 활용)
  - 긴 데이터, 1,536 바이트, 64 바이트, 16 바이트에 대한 throughput