**Computer Architecture**

**~Lab 10~**

**2013210111 남세현**

1. **목표**

Pipe-lined Processer를 만들어야 합니다.

1. **구현**
2. **와이어링**

PT에서 제공되는 KuStar의 배선도에 맞춰서 와이어링 작업을 해줍니다.

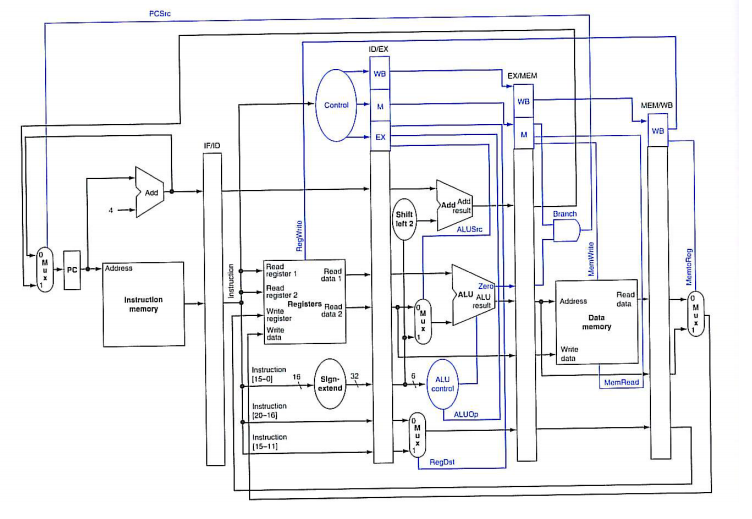


Figure 1 : Kustar Diagram

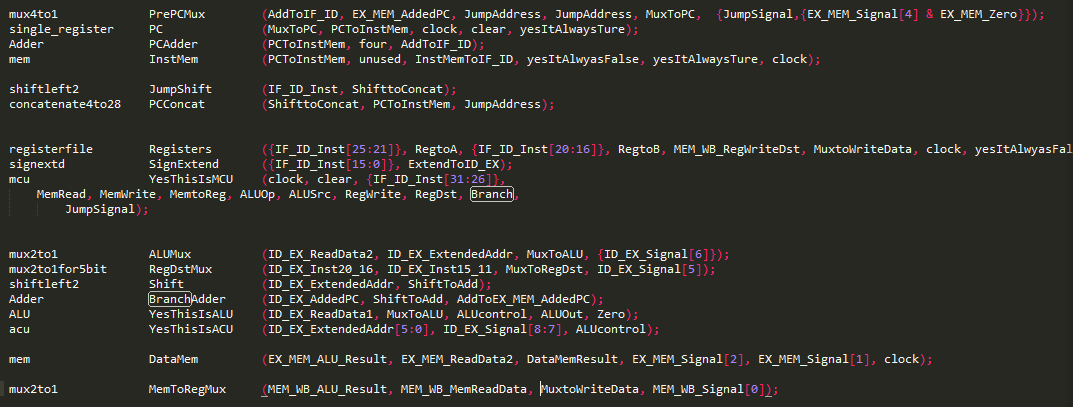


Figure 2 : Kustar Diagram -> Verilog

1. **Control Unit**

Lab09와는 다르게, FSM이 특별히 필요하지 않습니다.

위 A.에서의 배선도처럼 ID 구간에서 Opcode를 Control Unit으로 전송합니다.

전송받은 Opcode를 토대로 Signal을 생성하고, 그것을 버퍼에 저장시킵니다.

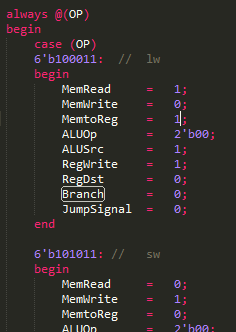


Figure 3 : MCU Signal 생성

1. **버퍼**

구간 A와 그 다음 구간 B 사이에 Pipe-lined에 필요한 데이터를 옮겨주기 위한 버퍼가 존재합니다.

Control Unit이 만들어놓은 시그널의 경우 한 뭉텅이의 reg에 같이 저장시킵니다.

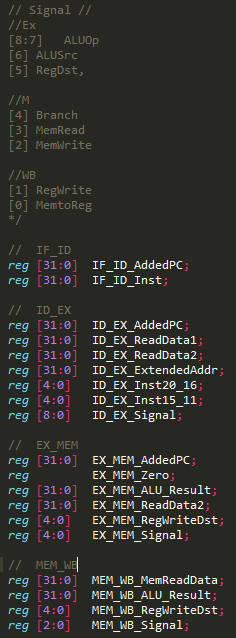


Figure 4 : 구간 사이의 Buffer

1. **Nop 추가**

Hazard Detection이나 Forwarding을 구현하지 않았기 때문에, Dependency가 있는 Instruction에 Nop들을 추가해줍니다.

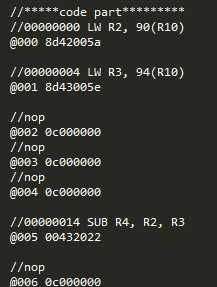


Figure 5 : Dependency == Nop

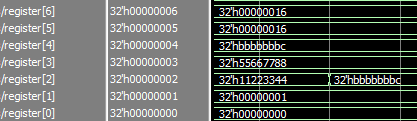
1. **결과 분석**
   1. **PC와 Instruction**

Multi-Cycle과는 다르게 PC가 계속 증가하면서 Instruction Fetch가 이루어짐을 볼 수 있습니다.

Branch와 Jump도 정상적으로 작동함을 볼 수 있습니다.

* 1. **Register**

$2와 $3은 lw의 결과물들이, $4에는 두 값의 차, 그리고 다음번 $2의 결과물은 sw로 저장된 $4의 값이 그대로 들어옴을 볼 수 있습니다. $5와 $6도 항상 24가 나오는 것을 볼 수 있습니다.



* 1. **Memory**

Sw의 결과물로 $2의 값이 계속 저장됨을 볼 수 있습니다.

