Computer Organization and Design:
The Hardware/Software Interface
by Patterson and Hennessy

Processor Design with Pipelining
프로그래밍 개별 과제

Gi-Ho Park

Dept. of Computer Engineering

Sejong University

과제 설명

- 다음의 과제들 중 자신의 수준에 맞는 것을 골라 수행한 후 제출
  - 제출 기간 : 12/13 자정까지 (연장 제출 허용)
  - 제출파일형식: CA_학번_이름_난이도.zip
    - 소스 코드 파일.cpp (파일 하나에 구현)
    - 보고서.hwp

과제 설명

- Alternative (2점 만점)
  - 3가지 명령어 (lw, add, beq) 에 대하여 그림4.24 control 정보의 값을 적고,
  - 3가지 명령어 (lw, add, beq) 에 대하여 명령어 실행 단계를 설명하여 제출
    - lw)
      1. …
      2. …

      …
    - add)
      1. …
      2. …

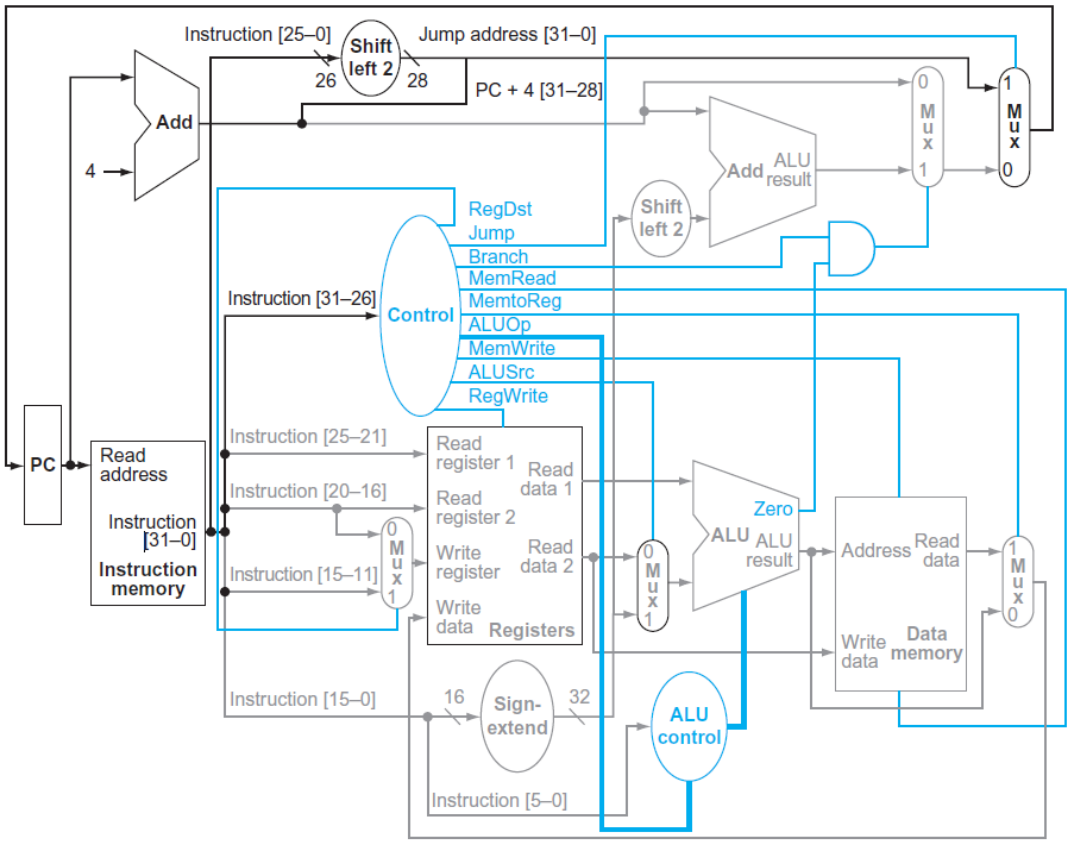
      …
    - beq)
      1. …
      2. …

      …

과제 설명

❖ Alternative (2점 만점)

- 0 / 1 / X



	Lw	add	beq
RegDst			
Jump			
Branch			
MemRead			
MemtoReg			
ALUOp1			
ALUOp2			
MemWrite			
ALUsrc			
RegWrite			

과제 설명

- Alternative (2점 만점)
  - CA_학번_이름_난이도.hwp
    - 수행 난이도
    - 과제 4페이지 그림 및 표
    - 3가지 명령어(lw, add, beq)에 대한 실행 단계 설명
    - 느낀점
  - 채점 기준
    - 3가지 명령어 (lw, add, beq) 에 대하여 명령어 실행 단계 설명 혹은 신호 값이 하나라도 틀리면 1점 감점

과제 설명

- ADD, BEQ, LW, SW, J 다섯 가지 명령어에 대해 구현
- 제공되는 3개의 입력 파일을 사용할 것.
- C++, C 이용. 파일 하나에 구현할 것.

과제 설명

- Basic (5점 만점) – a simple implementation
  - 단일 사이클 구현(single-cycle implementation)
  - ADD, BEQ, LW, SW, J 다섯 가지 명령어로 이루어진 32bit 2진수 MIPS Machine Code를 입력 파일로부터 읽어 들여 1 Cycle마다 수행
  - 파일에는 16진수로 명령어가 저장되어 있음

과제 설명

- Basic (5점 만점)
  - 1 cycle 마다 pc, total cycle, register, memory 출력
  - pc가 64이상 이면, 프로그램 종료
  - Basic.cpp 참고
    - Basic.cpp 를 반드시 사용할 필요는 없음
    - Basic.cpp 의 출력 형식(result)은 그대로 사용할 것
  - CA_학번_이름_난이도.zip
    - Code.cpp
    - 보고서.hwp
      - 수행 난이도
      - 3개의 입력 파일에 대한 수행 결과 – input_1, 2, 3 (최종 pc, total cycle, register, memory 만)
      - 느낀점

과제 설명

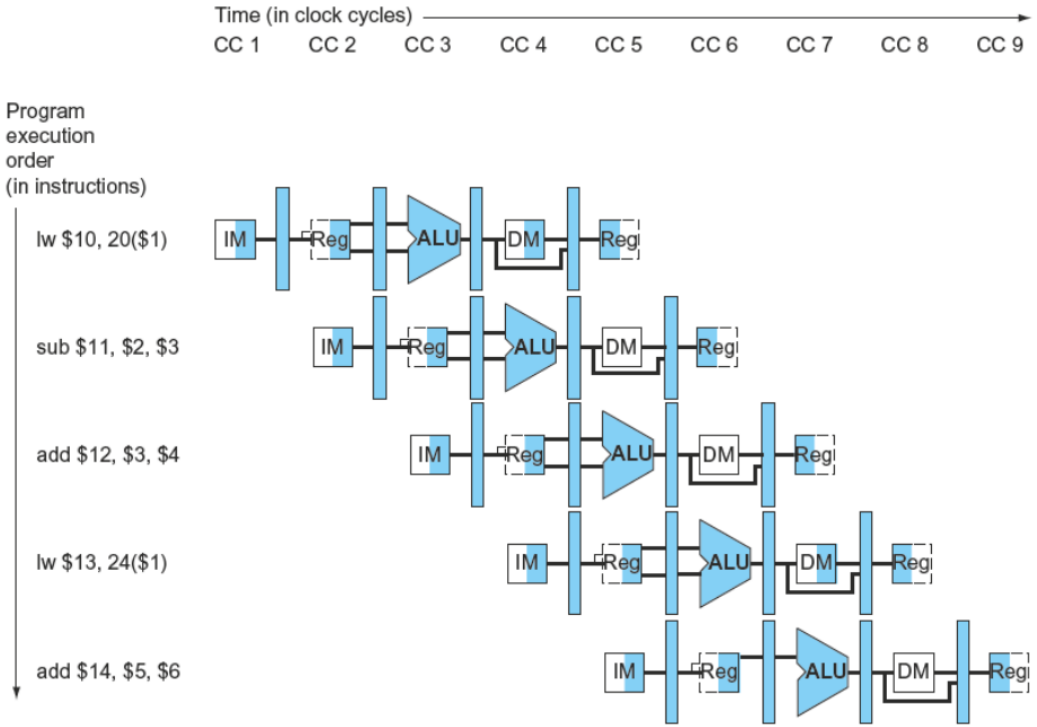
- Advanced (10점 만점) – a simple implementation
  - 단일 사이클 구현(single-cycle implementation)
  - Basic 과제에서, 각 구현에 사용된 소자 (각component, MUX, register, PC, ALU 등) 각각을 함수로 정의하여 구현한다.
  - 함수로 구현된 component들을 실제 처리 순서와 동일하게 구현한다.
  - 파일에는 16진수로 명령어가 저장되어 있음

과제 설명

- Advanced (10점 만점)
  - 1 cycle 마다 pc, total cycle, register, memory 출력
  - pc가 64이상 이면, 프로그램 종료
  - Advanced.cpp 참고
    - Advanced.cpp 를 반드시 사용할 필요는 없음
    - Advanced.cpp 의 출력 형식(result)은 그대로 사용할 것
  - CA_학번_이름_난이도.zip
    - Code.cpp
    - 보고서.hwp
      - 수행 난이도
      - 3개의 입력 파일에 대한 수행 결과 – input_1, 2, 3
        (최종 pc, total cycle, register, memory 만)
      - 느낀점

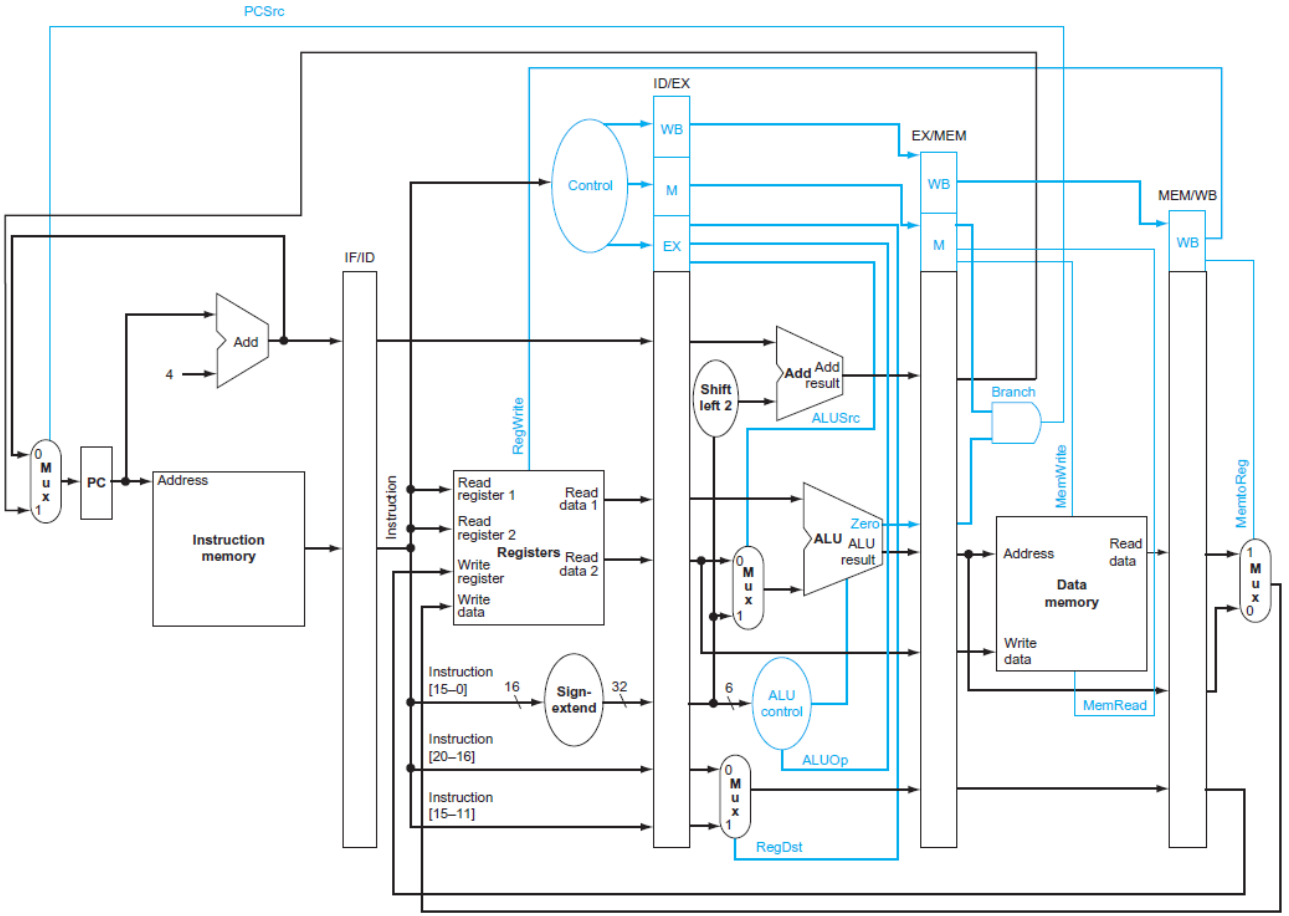
과제 설명

- **Challenge (10점 + 10점)**
  - Advanced를 포함하고, 다수 명령어간의 파이프 라인 단계를 고려하여 cycle별로 관련 메모리 소자의 값(register값, PC값 등)을 출력하는 프로그램을 작성하라.
  - Hazard는 고려하지 않음



과제 설명

- **Challenge (10점 + 10점)**
  - 그림 4.51 참고



과제 설명

- Challenge (10점 + 10점)
  - 1 cycle 마다 pipeline register, pc, total cycle, register, memory 출력
  - pc가 64이상 이면, 수행 중이던 명령어 끝까지 수행 후 프로그램 종료
  - Challenge.cpp 참고
    - Challenge.cpp 를 반드시 사용할 필요는 없음
    - Challenge.cpp 의 출력 형식(result)은 그대로 사용할 것
  - CA_학번_이름_난이도.zip
    - Advanced.cpp
    - Challenge.cpp
    - 보고서.hwp
      - 수행 난이도
      - 3개의 입력 파일에 대한 수행 결과 – input_4, 5, 6
        (최종 pipeline register(signal), pc, total cycle, register, memory)
      - 느낀점

과제 설명

- Register 초기값

0번-$zero	0	8번-$t0	41621	16번-$s0	40	24번-$t8	0
1번-$at	0	9번-$t1	41621	17번-$s1	0	25번-$t9	0
2번-$v0	0	10번-$t2	0	18번-$s2	0	26번-$k0	0
3번-$v1	0	11번-$t3	0	19번-$s3	0	27번-$k1	0
4번-$a0	0	12번-$t4	0	20번-$s4	0	28번-$gp	0
5번-$a1	0	13번-$t5	0	21번-$s5	0	29번-$sp	0
6번-$a2	0	14번-$t6	0	22번-$s6	0	30번-$fp	0
7번-$a3	0	15번-$t7	0	23번-$s7	0	31번-$ra	0

- PC, Control Signal, Instruction Register, Memory Data Register, A, B, ALUOut 등의 저장 소자들은 모두 초기값으로 0을 가짐
- 모든 pipeline register도 초기값으로 0을 가짐

과제 설명

- Memory 범위와 초기값
  - Address 0 ~ 16 : 입력 파일에서 읽은 Instruction 5개가 저장
  - Address 16 ~ 64 : Instruction이 사용할 Data가 저장

Address	Data	Address	Data
0	입력 파일의 첫번째 Instruction	32	0
4	입력 파일의 두번째 Instruction	36	0
8	입력 파일의 세번째 Instruction	40	3578
12	입력 파일의 네번째 Instruction	44	0
16	입력 파일의 다섯번째 Instruction	48	0
20	0	52	0
24	0	56	0
28	0	60	0

입력 파일 설명

- 1번 입력 파일(input_1.txt)
  - MIPS Machine Code

```
00010001000010010000000000000010
10001110000010000000000000000000
00000001000010010101000000100000
00001000000000000000000000010000
10101110000010100000000000000100
```

  - MIPS Assembly Code

```
BEQ $t0, $t1, 2
LW $t0, 0($s0)
ADD $t2, $t0, $t1
J 16
SW $t2, 4($s0)
```

입력 파일 설명

- 1번 입력 파일(input_1.txt)
  - 최종 결과

```
Instruction = 08000010
>> JUMP
PC : 64
Total cycle : 2

==================== REGISTER ====================
reg[00] = 00000000        reg[08] = 00041621        reg[16] = 00000040        reg[24] = 00000000
reg[01] = 00000000        reg[09] = 00041621        reg[17] = 00000000        reg[25] = 00000000
reg[02] = 00000000        reg[10] = 00000000        reg[18] = 00000000        reg[26] = 00000000
reg[03] = 00000000        reg[11] = 00000000        reg[19] = 00000000        reg[27] = 00000000
reg[04] = 00000000        reg[12] = 00000000        reg[20] = 00000000        reg[28] = 00000000
reg[05] = 00000000        reg[13] = 00000000        reg[21] = 00000000        reg[29] = 00000000
reg[06] = 00000000        reg[14] = 00000000        reg[22] = 00000000        reg[30] = 00000000
reg[07] = 00000000        reg[15] = 00000000        reg[23] = 00000000        reg[31] = 00000000
==================== REGISTER ====================

==================== MEMORY ====================
mem[00] = 000285802498        mem[32] = 000000000000
mem[04] = -01912078336        mem[36] = 000000000000
mem[08] = 000017387552        mem[40] = 000000003578
mem[12] = 000134217744        mem[44] = 000000000000
mem[16] = -01375076348        mem[48] = 000000000000
mem[20] = 000000000000        mem[52] = 000000000000
mem[24] = 000000000000        mem[56] = 000000000000
mem[28] = 000000000000        mem[60] = 000000000000
==================== MEMORY ====================
```

입력 파일 설명

- 2번 입력 파일(input_2.txt)
  - MIPS Machine Code

```
10001110000010000000000000000000
00010001000010010000000000001110
00000001000010010101000000100000
10101110000010100000000000000100
00001000000000000000000000010000
```

  - MIPS Assembly Code

```
LW $t0, 0($s0)
BEQ $t0, $t1, 14
ADD $t2, $t0, $t1
SW   $t2, 4($s0)
J 16
```

입력 파일 설명

- 2번 입력 파일(input_2.txt)
  - 최종 결과

```
Instruction = 08000010
>> JUMP
PC : 64
Total cycle : 5

==================== REGISTER ====================
reg[00] = 00000000        reg[08] = 00003578        reg[16] = 00000040        reg[24] = 00000000
reg[01] = 00000000        reg[09] = 00041621        reg[17] = 00000000        reg[25] = 00000000
reg[02] = 00000000        reg[10] = 00045199        reg[18] = 00000000        reg[26] = 00000000
reg[03] = 00000000        reg[11] = 00000000        reg[19] = 00000000        reg[27] = 00000000
reg[04] = 00000000        reg[12] = 00000000        reg[20] = 00000000        reg[28] = 00000000
reg[05] = 00000000        reg[13] = 00000000        reg[21] = 00000000        reg[29] = 00000000
reg[06] = 00000000        reg[14] = 00000000        reg[22] = 00000000        reg[30] = 00000000
reg[07] = 00000000        reg[15] = 00000000        reg[23] = 00000000        reg[31] = 00000000
==================== REGISTER ====================

==================== MEMORY ====================
mem[00] = -01912078336        mem[32] = 000000000000
mem[04] = 000285802510        mem[36] = 000000000000
mem[08] = 000017387552        mem[40] = 000000003578
mem[12] = -01375076348        mem[44] = 000000045199
mem[16] = 000134217744        mem[48] = 000000000000
mem[20] = 000000000000        mem[52] = 000000000000
mem[24] = 000000000000        mem[56] = 000000000000
mem[28] = 000000000000        mem[60] = 000000000000
==================== MEMORY ====================
```

입력 파일 설명

- 3번 입력 파일(input_3.txt)
  - MIPS Machine Code

```
00000001000010010101000000100000
10001110000010000000000000000000
10101110000010110000000000000100
00001000000000000000000000010000
00000001000010010111000000100000
```

  - MIPS Assembly Code

```
ADD $t2, $t0, $t1
LW $t0, 0($s0)
SW $t3, 4($s0)
J 16
ADD $t6, $t0, $t1
```

입력 파일 설명

- 3번 입력 파일(input_3.txt)
  - 최종 결과

```
Instruction = 08000010
>> JUMP
PC : 64
Total cycle : 4

==================== REGISTER ====================
reg[00] = 00000000        reg[08] = 00003578        reg[16] = 00000040        reg[24] = 00000000
reg[01] = 00000000        reg[09] = 00041621        reg[17] = 00000000        reg[25] = 00000000
reg[02] = 00000000        reg[10] = 00083242        reg[18] = 00000000        reg[26] = 00000000
reg[03] = 00000000        reg[11] = 00000000        reg[19] = 00000000        reg[27] = 00000000
reg[04] = 00000000        reg[12] = 00000000        reg[20] = 00000000        reg[28] = 00000000
reg[05] = 00000000        reg[13] = 00000000        reg[21] = 00000000        reg[29] = 00000000
reg[06] = 00000000        reg[14] = 00000000        reg[22] = 00000000        reg[30] = 00000000
reg[07] = 00000000        reg[15] = 00000000        reg[23] = 00000000        reg[31] = 00000000
==================== REGISTER ====================

==================== MEMORY ====================
mem[00] = 000017387552        mem[32] = 000000000000
mem[04] = -01912078336        mem[36] = 000000000000
mem[08] = -01375010812        mem[40] = 000000003578
mem[12] = 000134217744        mem[44] = 000000000000
mem[16] = 000017395744        mem[48] = 000000000000
mem[20] = 000000000000        mem[52] = 000000000000
mem[24] = 000000000000        mem[56] = 000000000000
mem[28] = 000000000000        mem[60] = 000000000000
==================== MEMORY ====================
```

입력 파일 설명

- 4번 입력 파일(input_4.txt) - Challenge용
  - MIPS Machine Code

```
00000001001010100110000000100000
10001110000010000000000000000000
10101110000010110000000000000100
00000001001011010111000000100000
00001000000000000000000000010000
```

  - MIPS Assembly Code

```
ADD $t4, $t1, $t2
LW $t0, 0($s0)
SW $t3, 4($s0)
ADD $t6, $t1, $t5
J 16
```

입력 파일 설명

- 4번 입력 파일(input_4.txt)
  - 최종 결과

```
==================== REGISTER ====================
reg[00] = 00000000        reg[08] = 00003578        reg[16] = 00000040        reg[24] = 00000000
reg[01] = 00000000        reg[09] = 00041621        reg[17] = 00000000        reg[25] = 00000000
reg[02] = 00000000        reg[10] = 00000000        reg[18] = 00000000        reg[26] = 00000000
reg[03] = 00000000        reg[11] = 00000000        reg[19] = 00000000        reg[27] = 00000000
reg[04] = 00000000        reg[12] = 00041621        reg[20] = 00000000        reg[28] = 00000000
reg[05] = 00000000        reg[13] = 00000000        reg[21] = 00000000        reg[29] = 00000000
reg[06] = 00000000        reg[14] = 00041621        reg[22] = 00000000        reg[30] = 00000000
reg[07] = 00000000        reg[15] = 00000000        reg[23] = 00000000        reg[31] = 00000000
==================== REGISTER ====================

==================== MEMORY ====================
mem[00] = 000019554336        mem[32] = 000000000000
mem[04] = -01912078336        mem[36] = 000000000000
mem[08] = -01375010812        mem[40] = 000000003578
mem[12] = 000019755040        mem[44] = 000000000000
mem[16] = 000134217744        mem[48] = 000000000000
mem[20] = 000000000000        mem[52] = 000000000000
mem[24] = 000000000000        mem[56] = 000000000000
mem[28] = 000000000000        mem[60] = 000000000000
==================== MEMORY ====================
```

입력 파일 설명

- 5번 입력 파일(input_5.txt) - Challenge용
  - MIPS Machine Code

```
00010001001010110000000000000010
00000001001010100110000000100000
00000001001011010111000000100000
10101110000010010000000000000100
00001000000000000000000000010000
```

  - MIPS Assembly Code

```
BEQ $t1, $t3, 2
ADD $t4, $t1, $t2
ADD $t6, $t1, $t5
SW $t1, 4($s0)
J 16
```

입력 파일 설명

- 5번 입력 파일(input_5.txt)
  - 최종 결과

```
==================== REGISTER ====================
reg[00] = 00000000        reg[08] = 00041621        reg[16] = 00000040        reg[24] = 00000000
reg[01] = 00000000        reg[09] = 00041621        reg[17] = 00000000        reg[25] = 00000000
reg[02] = 00000000        reg[10] = 00000000        reg[18] = 00000000        reg[26] = 00000000
reg[03] = 00000000        reg[11] = 00000000        reg[19] = 00000000        reg[27] = 00000000
reg[04] = 00000000        reg[12] = 00041621        reg[20] = 00000000        reg[28] = 00000000
reg[05] = 00000000        reg[13] = 00000000        reg[21] = 00000000        reg[29] = 00000000
reg[06] = 00000000        reg[14] = 00041621        reg[22] = 00000000        reg[30] = 00000000
reg[07] = 00000000        reg[15] = 00000000        reg[23] = 00000000        reg[31] = 00000000
==================== REGISTER ====================

==================== MEMORY ====================
mem[00] = 000288030722        mem[32] = 000000000000
mem[04] = 000019554336        mem[36] = 000000000000
mem[08] = 000019755040        mem[40] = 000000003578
mem[12] = -01375141884        mem[44] = 000000041621
mem[16] = 000134217744        mem[48] = 000000000000
mem[20] = 000000000000        mem[52] = 000000000000
mem[24] = 000000000000        mem[56] = 000000000000
mem[28] = 000000000000        mem[60] = 000000000000
==================== MEMORY ====================
```

입력 파일 설명

- 6번 입력 파일(input_6.txt) - Challenge용
  - MIPS Machine Code

```
00000001001010100110000000100000
10101110000010010000000000000100
00010001001010110000000000000010
10001110000010000000000000000000
00001000000000000000000000010000
```

  - MIPS Assembly Code

```
ADD $t4, $t1, $t2
SW $t1, 4($s0)
BEQ $t1, $t3, 2
LW $t0, 0($s0)
J 16
```

입력 파일 설명

- 6번 입력 파일(input_6.txt)
  - 최종 결과

```
==================== REGISTER ====================
reg[00] = 00000000        reg[08] = 00003578        reg[16] = 00000040        reg[24] = 00000000
reg[01] = 00000000        reg[09] = 00041621        reg[17] = 00000000        reg[25] = 00000000
reg[02] = 00000000        reg[10] = 00000000        reg[18] = 00000000        reg[26] = 00000000
reg[03] = 00000000        reg[11] = 00000000        reg[19] = 00000000        reg[27] = 00000000
reg[04] = 00000000        reg[12] = 00041621        reg[20] = 00000000        reg[28] = 00000000
reg[05] = 00000000        reg[13] = 00000000        reg[21] = 00000000        reg[29] = 00000000
reg[06] = 00000000        reg[14] = 00000000        reg[22] = 00000000        reg[30] = 00000000
reg[07] = 00000000        reg[15] = 00000000        reg[23] = 00000000        reg[31] = 00000000
==================== REGISTER ====================

==================== MEMORY ====================
mem[00] = 000019554336        mem[32] = 000000000000
mem[04] = -01375141884        mem[36] = 000000000000
mem[08] = 000288030722        mem[40] = 000000003578
mem[12] = -01912078336        mem[44] = 000000041621
mem[16] = 000134217744        mem[48] = 000000000000
mem[20] = 000000000000        mem[52] = 000000000000
mem[24] = 000000000000        mem[56] = 000000000000
mem[28] = 000000000000        mem[60] = 000000000000
==================== MEMORY ====================
```

입력 파일 설명

- **Challenge – pipeline register 출력**
  - **매 cycle 마다 출력**

```
=== ID/EX ===
WB - RegWrite: 0, MemtoReg: 0
M  - Branch  : 0, MemRead : 0, MemWrite: 0
EX - RegDst  : 0, ALUOp   : 0, ALUSrc  : 0
=== EX/MEM ===
WB - RegWrite: 0, MemtoReg: 0
M  - Branch  : 0, MemRead : 0, MemWrite: 0
=== MEM/WB ===
WB - RegWrite: 0, MemtoReg: 0
```

과제 설명

- 채점기준
  - 미제출: 0점
  - 프로그램 실행 안됨(단, 소스파일확인): 1점
  - 일부 test 시 정확한 동작 안함: 2점
  - 모든 test에서 정확한 동작: 해당 난이도에 따른 만점
  - 제출내용부족: 항목당 1점씩 감점
  - 연장 제출시 감점