Report

HW#3



과 목 명 : 컴퓨터구조

담당교수 : 이귀상 교수님

팀 장 : 선윤태 130206

팀 원 : 신미주 160542

신형환 140407

심우주 170489

**1. 과제 내용**

**Programming assignment #1의 코드를 확장해 다음과 같이 프로그램을 작성**

**- 입력: 기계어 코드 또는 symbol 기반 assembly code**

**- 기능**

**1) 각 레지스터를 변수로 선언(예를 들어 DR 을 16비트 변수로 선언)**

**2) intruction cycle 에서의 각 timing signal 을 따라 intruction fetch, decode instruction, operand fetch, execution 을 수행하도록 함.**

**- 출력**

**1) 기계어는 symbol code 로, symbol code 는 기계어로 변환**

**2) 명령어 별로 각 타이밍 변수 마다 (t0, t1, t2, t3, ...) intruction cycle 수행과정을 보일 것. 각 레지스터 값 / 메모리값 의 변화를 일목요연하게 출력**

**2) 모든 프로그램 수행 후 최종적으로 레지스터 / 메모리 값을 모두 출력하여 어떤 변화가 있었는지 출력.**

**검증)**

**1) 먼저 기계어코드를 symbol 형태로 출력하시오. ( Programming Assignment #1)**

**2) 표 6-2 와 표 6-9 의 두 가지 프로그램을 입력으로 수행하여 결과를 출력함.**

**3) 각 조별로 한 가지 이상 씩의 프로그램을 작성하여 본 과제 코드를 검증할 것.**

**2. 과제 분석**

M[]배열에 저장된 값을 해석해서 무슨 명령인지 판단하고 실행하는 코드를 만들기로 함. finfuc() 함수를 실행하여 먼저FND()를 참조하고 그 후에 조건문을 통해 IO(), Register(), Memory() 함수 셋 중 하나를 실행하도록 한다.

**3. 표 6-2 전체 코드**

**package** ProgrammingAssignment2;

**public** **class** Main {

**static** String[] *M* = **new** String[1000];

**static** String *pc* = "000";// 16진 3비트(12비트)

**static** String *ar* = "---";// 16진 3비트(12비트)

**static** String *ir* = "----";// 16진 4비트(16비트)

**static** String *i* = "-";// 2진 1비트

**static** String *dr* = "----";// 16진 4비트(16비트)

**static** String *ac* = "----";// 16진 4비트(16비트)

**static** String *e* = "1";// 2진 1비트

**static** **int** *s* = 1;

**static** String *inpr* = "00000000";// 2진 8비트

**static** String *outr* = "00000000";// 2진 8비트

**static** **int** *fgi* = 0;

**static** **int** *fgo* = 1;

**static** **int** *ien* = 0;

**static** **int** *d0*, *d1*, *d2*, *d3*, *d4*, *d5*, *d6*, *d7*;

**public** **static** **void** main(String[] args) {

*M*[0] = "2004";// ac에 4번지값 로드

*M*[1] = "1005";// ac에 5번지값 더함

*M*[2] = "3006";// 6번지에 저장

*M*[3] = "7001";// 작업 중지

*M*[4] = "0053";// 첫번째 값

*M*[5] = "FFE9";// 두번째 값

*M*[6] = "0000";// 저장할 장소

**while** (*s* != 0)// s가 0이 될때까지 반복

*finfunc*();

System.***out***.println(*M*[6]);// M[6]에 저장된값 프린트

}

**public** **static** **void** decoder(String s) {// FND()에서 참조하는 함수

*d0* = 0;

*d1* = 0;

*d2* = 0;

*d3* = 0;

*d4* = 0;

*d5* = 0;

*d6* = 0;

*d7* = 0;

s = *Hextobin*(s);

**switch** (s.substring(1, 4)) {

**case** "000":

*d0* = 1;

**break**;

**case** "001":

*d1* = 1;

**break**;

**case** "010":

*d2* = 1;

**break**;

**case** "011":

*d3* = 1;

**break**;

**case** "100":

*d4* = 1;

**break**;

**case** "101":

*d5* = 1;

**break**;

**case** "110":

*d6* = 1;

**break**;

**case** "111":

*d7* = 1;

**break**;

}

}

**public** **static** **void** FND() {// fetch&decode(t0~t2)

System.***out***.println("Tn PC AR IR I");

*ar* = *pc*;// t0:AR=PC;

System.***out***.println("t0" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

*ir* = *M*[Integer.*parseInt*(*ar*, 16)];// t1:IR=M[AR]

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;// t1:PC=PC+1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

System.***out***.println("t1" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

*decoder*(*ir*);// d0~d7=ir(12-14비트)

*ar* = *ir*.substring(1, 4);// ar=ir(0-11비트) 10진수 그대로

String binir = *Hextobin*(*ir*);

**if** (binir.substring(0, 1).equals("1"))// i=ir(15비트)

*i* = "1";

**else**

*i* = "0";

System.***out***.println("t2" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

}

**public** **static** **void** Memory() {// 메모리 참조 명령(t4~)

System.***out***.println("Tn PC AR DR AC");

**if** (*d0* == 1) {// AND

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

**int** in = Integer.*parseInt*(*ac*, 16) & Integer.*parseInt*(*dr*, 16);

*ac* = Integer.*toHexString*(in);

**while** (*ac*.length() != 4)

*ac* = "0" + *ac*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d1* == 1) {// ADD

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

**int** in = Integer.*parseInt*(*ac*, 16) + Integer.*parseInt*(*dr*, 16);

in = (**int**) (in % Math.*pow*(2, 16));

*ac* = Integer.*toHexString*(in);

**while** (*ac*.length() != 4)

*ac* = "0" + *ac*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d2* == 1) {// LDA

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];// t4

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

*ac* = *dr*;// t5

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d3* == 1) {// STA

*M*[Integer.*parseInt*(*ar*, 16)] = *ac*;

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d4* == 1) {// BUN

*pc* = *ar*;

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d5* == 1) {// BSA

*M*[Integer.*parseInt*(*ar*, 16)] = *pc*;

**int** iar = Integer.*parseInt*(*ar*, 16) + 1;

*ar* = Integer.*toHexString*(iar);

**while** (*ar*.length() != 3)

*ar* = "0" + *ar*;

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

*pc* = *ar*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d6* == 1) {// ISZ

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

**int** idr = Integer.*parseInt*(*dr*, 16) + 1;

*dr* = Integer.*toHexString*(idr);

**while** (*dr*.length() != 4)

*dr* = "0" + *dr*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

*M*[Integer.*parseInt*(*ar*, 16)] = *dr*;

**if** (*dr*.equals("0")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

System.***out***.println("t6" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

}

}

**public** **static** **void** Register() {// 레지스터 참조명령(t3)

System.***out***.println("Tn PC E AC S");

String s2 = *Hextobin*(*ir*).substring(4);// 0-11비트

**int** count = 0;// s2에 1의 갯수를 세기 위해 count 선언

**int** countup = -99;// count가 올라갈때 i의 값을 저장하기 위한 변수 countup 선언

**for** (**int** i = 0; i < s2.length(); i++) {

**if** (s2.charAt(i) == '1') {

count++;

countup = i;

}

}

**if** (count == 1) {// s2에서 1의 갯수가 하나일때

**switch** (countup) {// 몇번째 순서에서 1이 나왔나에 따라

**case** 0:// CLA

*ac* = "0000";

**break**;

**case** 1:// CLE

*e* = "0";

**break**;

**case** 2:// CMA

**int** iac = Integer.*parseInt*(*ac*, 16);

iac = ~iac;

*ac* = Integer.*toHexString*(iac);

*ac* = *ac*.substring(4);

**break**;

**case** 3:// CME

**int** ie = Integer.*parseInt*(*e*, 2);

ie = ~ie;

*e* = Integer.*toBinaryString*(ie);

*e* = *e*.substring(31);

**break**;

**case** 4:// "CIR"

*ac* = *Hextobin*(*ac*);

String tmp = *e*;

*e* = *ac*.substring(15);

*ac* = tmp + *ac*.substring(0, 15);

*ac* = *Bintohex*(*ac*);

**break**;

**case** 5:// "CIL"

*ac* = *Hextobin*(*ac*);

tmp = *e*;

*e* = *ac*.substring(0, 1);

*ac* = *ac*.substring(1) + tmp;

*ac* = *Bintohex*(*ac*);

**break**;

**case** 6:// "INC"

iac = Integer.*parseInt*(*ac*, 16) + 1;

*ac* = Integer.*toHexString*(iac);

**while** (*ac*.length() != 4)

*ac* = "0" + *ac*;

**break**;

**case** 7:// "SPA"

**if** (*Hextobin*(*ac*).substring(0, 1).equals("0")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 8:// "SNA"

**if** (*Hextobin*(*ac*).substring(0, 1).equals("1")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 9:// "SZA"

**if** (*ac*.equals("0000")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 10:// "SZE"

**if** (*e*.equals("0")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 11:// "HLT"

*s* = 0;

**break**;

}

System.***out***.println("t3" + " " + *pc* + " " + *e* + " " + *ac* + " " + *s*);

}

}

**public** **static** **void** IO() {// Input Output(t3)

System.***out***.println("Tn AC INPR FGI OUTR FGO IEN");

String s2 = *Hextobin*(*ir*).substring(4);// 0-11비트

**int** count = 0;// s2에 1의 갯수를 세기 위해 count 선언

**int** countup = -99;// count가 올라갈때 i의 값을 저장하기 위한 변수 countup 선언

**for** (**int** i = 0; i < s2.length(); i++) {

**if** (s2.charAt(i) == '1') {

count++;

countup = i;

}

}

**if** (count == 1) {// s2에서 1의 갯수가 하나일때

**switch** (countup) {// 몇번째 순서에서 1이 나왔나에 따라

**case** 0:// "INP"

*ac* = *Hextobin*(*ac*);

*ac* = *ac*.substring(0, 8) + *inpr*;

*fgi* = 0;

*ac* = *Bintohex*(*ac*);

**break**;

**case** 1:// "OUT"

*ac* = *Hextobin*(*ac*);

*outr* = *ac*.substring(8);

*fgo* = 0;

*ac* = *Bintohex*(*ac*);

**break**;

**case** 2:// "SKI"

**if** (*fgi* == 1) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 3:// "SKO"

**if** (*fgo* == 1) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 4:// "ION"

*ien* = 1;

**break**;

**case** 5:// "IOF"

*ien* = 0;

**break**;

}

System.***out***.println("t3" + " " + *ac* + " " + *inpr* + " " + *fgi* + " " + *outr* + " " + *fgo* + " " + *ien*);

}

}

**public** **static** **void** finfunc() {

*FND*();

**if** (*d7* == 1) {

**if** (*i*.equals("1")) {

*IO*();

} **else** {

*Register*();

}

} **else** {

**if** (*i*.equals("1")) {// Indirect(i=1) 조건부 AR=M[AR]

*ar* = *M*[Integer.*parseInt*(*ar*, 16)];

**if** (*ar*.length()>3)

*ar* = *ar*.substring(1,4);

**else** **if** (*ar*.length()<3)

while(*ar*.length()!=3)

*ar* ="0"+ *ar*;

System.***out***.println("t3" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

*Memory*();

}

*Memory*();

}

System.***out***.println();

}

**public** **static** String Hextobin(String s) {// 16진수를 2진수로

**int** iir = Integer.*parseInt*(s, 16);

String s2 = Integer.*toBinaryString*(iir);

**while** (s2.length() != 16) // 빈자리 0채우기

s2 = "0" + s2;

**return** s2;

}

**public** **static** String Bintohex(String s) {// 2진수를 16진수로

**int** is = Integer.*parseInt*(s, 2);

String s2 = Integer.*toHexString*(is);

**while** (s2.length() != 4)// 빈자리 0채우기

s2 = "0" + s2;

**return** s2;

}

}

**4. 표 6-2코드 메소드 별 설명**

**0) 전역변수 설정**

**static** String[] *M* = **new** String[1000];

**static** String *pc* = "000";// 16진 3비트(12비트)

**static** String *ar* = "---";// 16진 3비트(12비트)

**static** String *ir* = "----";// 16진 4비트(16비트)

**static** String *i* = "-";// 2진 1비트

**static** String *dr* = "----";// 16진 4비트(16비트)

**static** String *ac* = "----";// 16진 4비트(16비트)

**static** String *e* = "1";// 2진 1비트

**static** **int** *s* = 1;

**static** String *inpr* = "00000000";// 2진 8비트

**static** String *outr* = "00000000";// 2진 8비트

**static** **int** *fgi* = 0;

**static** **int** *fgo* = 1;

**static** **int** *ien* = 0;

**static** **int** *d0*, *d1*, *d2*, *d3*, *d4*, *d5*, *d6*, *d7*;

**1) main 함수**

**public** **static** **void** main(String[] args) {

*M*[0] = "2004";// ac에 4번지값 로드

*M*[1] = "1005";// ac에 5번지값 더함

*M*[2] = "3006";// 6번지에 저장

*M*[3] = "7001";// 작업 중지

*M*[4] = "0053";// 첫번째 값

*M*[5] = "FFE9";// 두번째 값

*M*[6] = "0000";// 저장할 장소

**while** (*s* != 0)// s가 0이 될때까지 반복

*finfunc*();

System.***out***.println(*M*[6]);// M[6]에 저장된값 프린트

}

**2) t0~t2 과정의 FND()함수**

**public** **static** **void** FND() {// fetch&decode(t0~t2)

System.***out***.println("Tn PC AR IR I");

*ar* = *pc*;// t0:AR=PC;

System.***out***.println("t0" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

*ir* = *M*[Integer.*parseInt*(*ar*, 16)];// t1:IR=M[AR]

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;// t1:PC=PC+1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

System.***out***.println("t1" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

*decoder*(*ir*);// d0~d7=ir(12-14비트)

*ar* = *ir*.substring(1, 4);// ar=ir(0-11비트) 10진수 그대로

String binir = *Hextobin*(*ir*);

**if** (binir.substring(0, 1).equals("1"))// i=ir(15비트)

*i* = "1";

**else**

*i* = "0";

System.***out***.println("t2" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

}

**3) d0~d7을 결정하는 decoder() 함수**

**public** **static** **void** decoder(String s) {// FND()에서 참조하는 함수

*d0* = 0;

*d1* = 0;

*d2* = 0;

*d3* = 0;

*d4* = 0;

*d5* = 0;

*d6* = 0;

*d7* = 0;

s = *Hextobin*(s); //16진수를 2진수로 변환

**switch** (s.substring(1, 4)) {//s의 2,3,4 번째 수를 뽑아서 case문으로 비교

**case** "000":

*d0* = 1;

**break**;

**case** "001":

*d1* = 1;

**break**;

**case** "010":

*d2* = 1;

**break**;

**case** "011":

*d3* = 1;

**break**;

**case** "100":

*d4* = 1;

**break**;

**case** "101":

*d5* = 1;

**break**;

**case** "110":

*d6* = 1;

**break**;

**case** "111":

*d7* = 1;

**break**;

}

}

**4) 메모리 참조 명령 함수 Memory()함수**

**public** **static** **void** Memory() {// 메모리 참조 명령(t4~)

System.***out***.println("Tn PC AR DR AC");

**if** (*d0* == 1) {// AND

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

**int** in = Integer.*parseInt*(*ac*, 16) & Integer.*parseInt*(*dr*, 16);

*ac* = Integer.*toHexString*(in);

**while** (*ac*.length() != 4)

*ac* = "0" + *ac*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d1* == 1) {// ADD

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

**int** in = Integer.*parseInt*(*ac*, 16) + Integer.*parseInt*(*dr*, 16);

in = (**int**) (in % Math.*pow*(2, 16));

*ac* = Integer.*toHexString*(in);

**while** (*ac*.length() != 4)

*ac* = "0" + *ac*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d2* == 1) {// LDA

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];// t4

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

*ac* = *dr*;// t5

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d3* == 1) {// STA

*M*[Integer.*parseInt*(*ar*, 16)] = *ac*;

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d4* == 1) {// BUN

*pc* = *ar*;

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d5* == 1) {// BSA

*M*[Integer.*parseInt*(*ar*, 16)] = *pc*;

**int** iar = Integer.*parseInt*(*ar*, 16) + 1;

*ar* = Integer.*toHexString*(iar);

**while** (*ar*.length() != 3)

*ar* = "0" + *ar*;

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

*pc* = *ar*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

} **else** **if** (*d6* == 1) {// ISZ

*dr* = *M*[Integer.*parseInt*(*ar*, 16)];

System.***out***.println("t4" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

**int** idr = Integer.*parseInt*(*dr*, 16) + 1;

*dr* = Integer.*toHexString*(idr);

**while** (*dr*.length() != 4)

*dr* = "0" + *dr*;

System.***out***.println("t5" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

*M*[Integer.*parseInt*(*ar*, 16)] = *dr*;

**if** (*dr*.equals("0")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

System.***out***.println("t6" + " " + *pc* + " " + *ar* + " " + *dr* + " " + *ac*);

}

}

**5) 레지스터 참조 명령 함수 Register()함수**

**public** **static** **void** Register() {// 레지스터 참조명령(t3)

System.***out***.println("Tn PC E AC S");

String s2 = *Hextobin*(*ir*).substring(4);// 0-11비트

**int** count = 0;// s2에 1의 갯수를 세기 위해 count 선언

**int** countup = -99;// count가 올라갈때 i의 값을 저장하기 위한 변수 countup 선언

**for** (**int** i = 0; i < s2.length(); i++) {

**if** (s2.charAt(i) == '1') {

count++;

countup = i;

}

}

**if** (count == 1) {// s2에서 1의 갯수가 하나일때

**switch** (countup) {// 몇번째 순서에서 1이 나왔나에 따라

**case** 0:// CLA

*ac* = "0000";

**break**;

**case** 1:// CLE

*e* = "0";

**break**;

**case** 2:// CMA

**int** iac = Integer.*parseInt*(*ac*, 16);

iac = ~iac;

*ac* = Integer.*toHexString*(iac);

*ac* = *ac*.substring(4);

**break**;

**case** 3:// CME

**int** ie = Integer.*parseInt*(*e*, 2);

ie = ~ie;

*e* = Integer.*toBinaryString*(ie);

*e* = *e*.substring(31);

**break**;

**case** 4:// "CIR"

*ac* = *Hextobin*(*ac*);

String tmp = *e*;

*e* = *ac*.substring(15);

*ac* = tmp + *ac*.substring(0, 15);

*ac* = *Bintohex*(*ac*);

**break**;

**case** 5:// "CIL"

*ac* = *Hextobin*(*ac*);

tmp = *e*;

*e* = *ac*.substring(0, 1);

*ac* = *ac*.substring(1) + tmp;

*ac* = *Bintohex*(*ac*);

**break**;

**case** 6:// "INC"

iac = Integer.*parseInt*(*ac*, 16) + 1;

*ac* = Integer.*toHexString*(iac);

**while** (*ac*.length() != 4)

*ac* = "0" + *ac*;

**break**;

**case** 7:// "SPA"

**if** (*Hextobin*(*ac*).substring(0, 1).equals("0")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 8:// "SNA"

**if** (*Hextobin*(*ac*).substring(0, 1).equals("1")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 9:// "SZA"

**if** (*ac*.equals("0000")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 10:// "SZE"

**if** (*e*.equals("0")) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 11:// "HLT"

*s* = 0;

**break**;

}

System.***out***.println("t3" + " " + *pc* + " " + *e* + " " + *ac* + " " + *s*);

}

}

**6) 입출력 함수 IO()함수**

**public** **static** **void** IO() {// Input Output(t3)

System.***out***.println("Tn AC INPR FGI OUTR FGO IEN");

String s2 = *Hextobin*(*ir*).substring(4);// 0-11비트

**int** count = 0;// s2에 1의 갯수를 세기 위해 count 선언

**int** countup = -99;// count가 올라갈때 i의 값을 저장하기 위한 변수 countup 선언

**for** (**int** i = 0; i < s2.length(); i++) {

**if** (s2.charAt(i) == '1') {

count++;

countup = i;

}

}

**if** (count == 1) {// s2에서 1의 갯수가 하나일때

**switch** (countup) {// 몇번째 순서에서 1이 나왔나에 따라

**case** 0:// "INP"

*ac* = *Hextobin*(*ac*);

*ac* = *ac*.substring(0, 8) + *inpr*;

*fgi* = 0;

*ac* = *Bintohex*(*ac*);

**break**;

**case** 1:// "OUT"

*ac* = *Hextobin*(*ac*);

*outr* = *ac*.substring(8);

*fgo* = 0;

*ac* = *Bintohex*(*ac*);

**break**;

**case** 2:// "SKI"

**if** (*fgi* == 1) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 3:// "SKO"

**if** (*fgo* == 1) {

**int** ipc = Integer.*parseInt*(*pc*, 16) + 1;

*pc* = Integer.*toHexString*(ipc);

**while** (*pc*.length() != 3)

*pc* = "0" + *pc*;

}

**break**;

**case** 4:// "ION"

*ien* = 1;

**break**;

**case** 5:// "IOF"

*ien* = 0;

**break**;

}

System.***out***.println("t3" + " " + *ac* + " " + *inpr* + " " + *fgi* + " " + *outr* + " " + *fgo* + " " + *ien*);

}

}

**7) FND() 실행 후 조건에 따라 메모리 참조,레지스터 참조, 입출력 명령인지 정하고 실행하는 함수 finfunc()**

**public** **static** **void** finfunc() {

*FND*();

**if** (*d7* == 1) {

**if** (*i*.equals("1")) {

*IO*();

} **else** {

*Register*();

}

} **else** {

**if** (*i*.equals("1")) {// Indirect(i=1) 조건부 AR=M[AR]

*ar* = *M*[Integer.*parseInt*(*ar*, 16)];

**if** (*ar*.length() > 3)

*ar* = *ar*.substring(1,4);

**elst if** (*ar*.length() < 3)

**while** (*ar*.length() != 3)

*ar* = "0" + *ar*;

System.***out***.println("t3" + " " + *pc* + " " + *ar* + " " + *ir* + " " + *i*);

*Memory*();

}

*Memory*();

}

System.***out***.println();

}

**8) 16진수를 2진수로 변환하는 Hextobin() 함수**

**public** **static** String Hextobin(String s) {// 16진수를 2진수로

**int** iir = Integer.*parseInt*(s, 16);

String s2 = Integer.*toBinaryString*(iir);

**while** (s2.length() != 16) // 빈자리 0채우기

s2 = "0" + s2;

**return** s2;

}

**9) 2진수를 16진수로 변환하는 Bintohex() 함수**

**public** **static** String Bintohex(String s) {// 2진수를 16진수로

**int** is = Integer.*parseInt*(s, 2);

String s2 = Integer.*toHexString*(is);

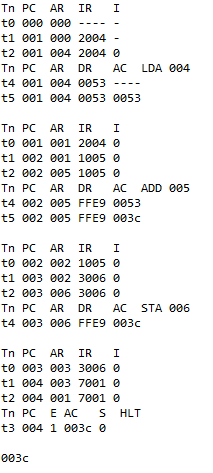
**while** (s2.length() != 4)// 빈자리 0채우기

s2 = "0" + s2;

**return** s2;

}

**5. 표 6-2 실행 결과**

****

M[0] = "2004";// ac에 4번지값 로드

M[1] = "1005";// ac에 5번지값 더함

M[2] = "3006";// 6번지에 저장

M[3] = "7001";// 작업 중지

M[4] = "0053";// 첫번째 값

M[5] = "FFE9";// 두번째 값

M[6] = "0000";// 저장할 장소

**해석)**

**(pc=0) M[0]의 값(2004)을 참조한다. ac에 004 번지의 값인 0053을 로드한다. //LDA 004**

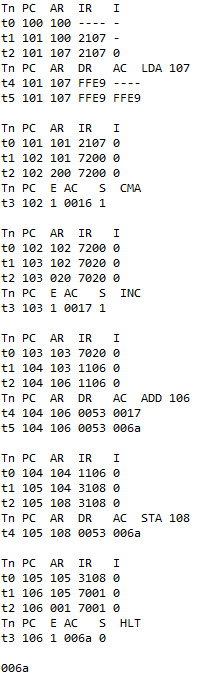
**(pc=1) M[1]의 값(1005)을 참조한다. 005번지의 값을 더해준다. //ADD 005**

**(pc=2) M[2]의 값(3006)을 참조한다. AC값을 006번지에 저장하는 것이다. //STA 006**

**(pc=3) M[3]의 값(7001)을 참조한다. S 값이 0이 된다. //HLT**

**while(s!=0)구문에 걸려서 반복문이 끝나게 된다.**

**6. 표 6-9 실행 결과**

****

M[Integer.parseInt("100", 16)] = "2107";// ac에 107번지값 로드

M[Integer.parseInt("101", 16)] = "7200";// ac의 보수를 취함

M[Integer.parseInt("102", 16)] = "7020";// ac값 1증가

M[Integer.parseInt("103", 16)] = "1106";// ac에 106번지값 더함

M[Integer.parseInt("104", 16)] = "3108";// 108번지에 저장

M[Integer.parseInt("105", 16)] = "7001";// 작업 중지

M[Integer.parseInt("106", 16)] = "0053";// 두번째 값

M[Integer.parseInt("107", 16)] = "FFE9";// 첫번째 값

M[Integer.parseInt("108", 16)] = "0000";// 저장할 장소

pc = "100";

**해석)**

**처음 pc 값은 100으로 100번지에서 부터 프로그램이 실행된다.**

**(pc=100) 2107대로 107번지에 있는 값을 load해 온다. //LDA 107**

**(pc=101)이 값을 보수를 취하고 //CMA**

**(pc=102) increment를 해준뒤 //INC**

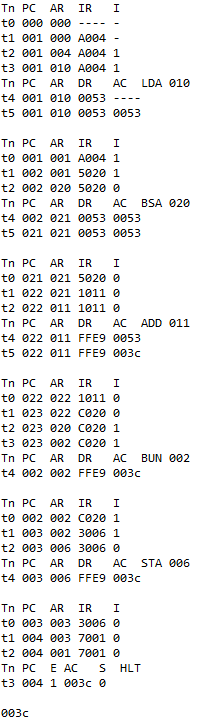
**(pc=103) 106번지에 있는 값과 더해준다. //ADD 106**

**(pc=104)그리고 이 값을 108번지에 저장한다 //STA 108**

**(pc=105)에서 S값이 0이 된다. //HLT**

**while(s!=0)구문에 걸려서 반복문이 끝나게 된다.**

**7. 직접 작성한 프로그램 실행 결과**

****

M[0] = "A004";// ac에 4번지값 간접참조(LDA 004 I)

M[4] = "0010";// 010번지 직접참조

M[Integer.parseInt("010",16)] = "0053";// 첫번째 값

M[1] = "5020";// 020번지로 점프(BSA 020)

M[Integer.parseInt("021", 16)] = "1011";// ac에 11번지값 더함(ADD 011)

M[Integer.parseInt("011", 16)] = "FFE9";// 두번째 값

M[Integer.parseInt("022", 16)] = "C020";//020번지 값 간접참조 점프(BUN 020 I)

M[2] = "3006";// 6번지에 저장

M[3] = "7001";// 작업 중지

M[6] = "0000";// 저장할 장소

**해석)**

**(pc=0) M[0]의 값(A004)을 참조한다. 간접참조이므로 M[4]의 값(0010)을 직접참조한다. M[10]의 값인 0053을 로드한다. //LDA 004 I**

**(pc=1) M[1]의 값(5020)을 참조한다. 20번지로 점프한다. //BSA 020**

**(pc=21) M[21]의 값(1011)을 참조한다. AC값에 M[6]값인 FFE9를 더해준다. //ADD 011**

**(pc=22) M[22]의 값(C020)을 참조한다. 20번지의 값을 간접참조한다. 위의 BSA 020과정에서 M[20]에 2가 저장되서 2번지로 점프한다. //BUN 020 I**

**(pc=2) M[2]의 값(3006)을 참조한다. AC 006번지에 저장한다. //STA 006**

**(pc=3) M[3]의 값(7001)을 참조한다. S 값이 0이 된다. //HLT**

**while(s!=0)구문에 걸려서 반복문이 끝나게 된다.**