마이크로프로그램

CPU의 명령어 세트 설계 과정에서

- 명령어들의 종류와 비트 패턴을 정의하고,
- 명령어들의 실행에 필요한 하드웨어를 설계하고,
- 각 명령어 실행을 위한 다양한 마이크로서브루틴을 작성한 후,
- 마이크로프로그램 코드들을 제어 기억장치에 저장한다.

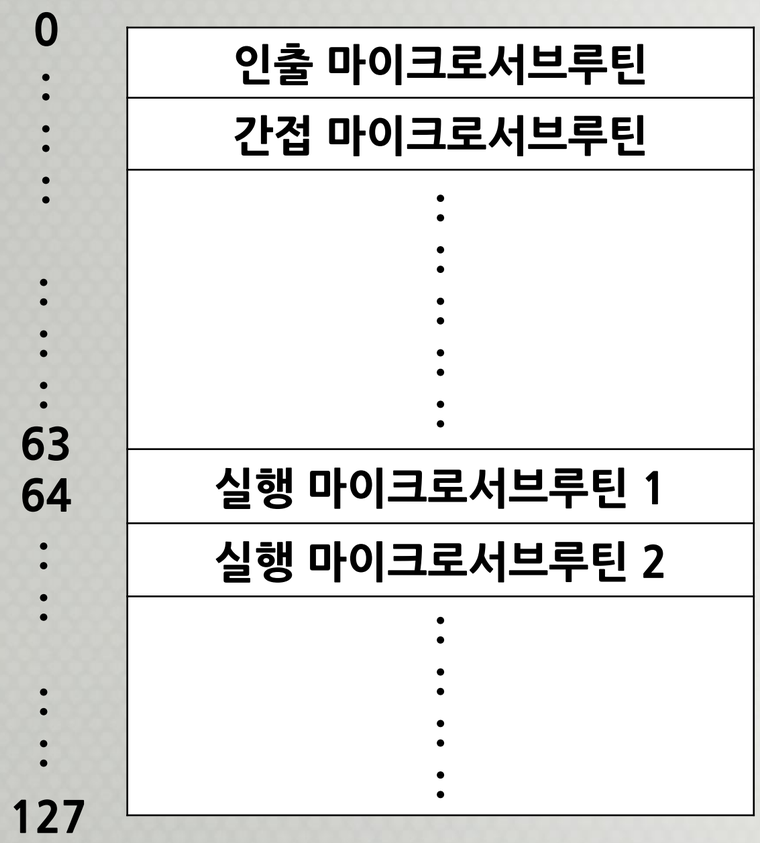
제어기억장치

- CPU 마다 종류가 매우 다양하다.
- ROM으로 구성된다.
- ROM의 사이즈는 마이크로명령어 형식에 따라 마이크로프로그램 크기로 결정된다.

마이크로프로그램

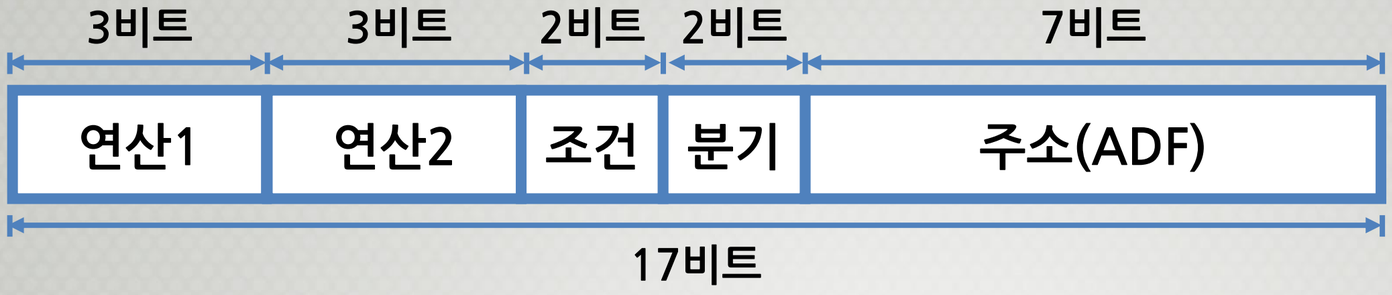
제어기억장치

예) 명령어 길이 : 16 bits ; op-code(4), 간접(1), 즉치 혹은 주소(11)
제어기억장치(ROM) : 2^7*17 bits
마이크로명령어 길이 : 17 bits ; 연산1(3), 연산2(3), 조건(2),
분기(2), ADF(7)



마이크로프로그램

1) 마이크로명령어 형식



연산1 필드 연산2 필드	● 이 필드들은 두 개의 마이크로 명령어들이 동시에 실행될 수 있다는 것을 의미한다.
조건 필드	● 분기 필드에서 사용될 조건을 명시한다.
분기 필드	● 분기의 종류를 명시하고, 조건에 따라 다음에 실행될 마이크로 명령어의 주소 결정방법을 명시한다.
주소 필드	● 조건에 따라 분기가 발생하는 경우에 분기의 목적지 주소로 사용된다.

마이크로프로그램

2) 마이크로명령어의 2진 코드와 기호

- 연산1 필드에 위치할 마이크로-연산

코드	마이크로-연산	기호
000	None	NONE
001	MAR ← PC	PCTAR
010	MAR ← IR(addr)	IRTAR
011	AC ← AC+MDR	ADD
100	MDR ← M[MAR]	READ
101	AC ← MDR	DRTAC
110	IR ← MDR	DRTIR
111	M[MAR] ← MDR	WRITE

마이크로프로그램

2) 마이크로명령어의 2진 코드와 기호

- 연산2 필드에 위치할 마이크로-연산

코드	마이크로-연산	기호
000	None	NONE
001	PC ← PC+1	INCPC
010	MDR ← AC	ACTDR
011	MDR ← PC	PCTDR
100	PC ← MDR	DRTPC
101	MAR ← SP	SPTAR
110	AC ← AC-MDR	SUB
111	PC ← IR(addr)	IRTPC

마이크로프로그램

2) 마이크로명령어의 2진 코드와 기호

- 조건 필드에 위치할 마이크로-연산
  - U : 무조건 분기
  - I : ‘I=1’이면 간접 사이클 루틴을 호출
  - S : 누산기에 저장된 데이터의 부호가 ‘1’이면 분기
  - Z : 누산기에 저장된 데이터가 ‘0’이면 분기

코드	조건	기호	설명
00	1	U	무조건 분기
01	I 비트	I	간접 주소 지정
10	AC(S)	S	누산기(AC)에 저장된 데이터의 부호
11	AC=0	Z	AC에 저장된 데이터=0

마이크로프로그램

2) 마이크로명령어의 2진 코드와 기호

- 분기 필드에 위치할 마이크로-연산

JUMP	• 조건에 따라 CAR의 내용을 주소필드(ADF)의 값 혹은 CAR +1 의 값으로 저장한다.
CALL	• 조건에 따라 CAR의 내용을 주소필드(ADF)의 값 혹은 CAR +1 의 값으로 저장한다. 여기서, 주소필드의 값으로 저장할 때는 복귀 할 주소 값을 SBR에 저장한다.
RET	• 마이크로서브루틴으로부터 복귀 (SBR에 저장된 내용을 CAR로 적재)
MAP	• 사상방식(mapping)에 의하여 분기 목적지 주소 결정

마이크로프로그램

2) 마이크로명령어의 2진 코드와 기호

- 분기 필드에 위치할 마이크로-연산

코드	기호	설명
00	JMP	‘조건=1’이면 CAR ← ADF ‘조건=0’이면 CAR ← CAR+1
01	CALL	‘조건=1’이면 CAR ← ADF, SBR ← CAR+1 ‘조건=0’이면 CAR ← CAR+1
10	RET	CAR ← SBR(서브루틴으로부터의 복귀)
11	MAP	CAR_6 ←“1”, $CAR_{5..2}$ ← IR(op-code), $CAR_{1,0}$ ← “0”

마이크로프로그램

3) 마이크로프로그램

- 인출 사이클의 마이크로서브루틴

```
        ORG 0
FETCH : PCTAR  NONE   U   JMP   NEXT   ; MAR ← PC
        READ   INCPC  U   JMP   NEXT   ; MDR ← M[MAR], PC← PC+1
        DRTIR  NONE   U   MAP          ; IR ← MDR
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
0000000	001	000	00	00	0000001
0000001	100	001	00	00	0000010
0000010	110	000	00	11	0000000

마이크로프로그램

3) 마이크로프로그램

- 간접 사이클의 마이크로서브루틴

```
            ORG 4
INDRT :  IRTAR   NONE   U   JMP   NEXT   ; MAR ← IR(addr)
         READ    NONE   U   JMP   NEXT   ; MDR ← M[MAR]
         DRTIR   NONE   U   RET          ; IR(addr) ← MDR
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
0000100	010	000	00	00	0000101
0000101	100	000	00	00	0000110
0000110	110	000	00	10	0000000

마이크로프로그램

3) 마이크로프로그램

- 실행 사이클의 마이크로서브루틴을 찾기 위한 사상(Mapping)방법

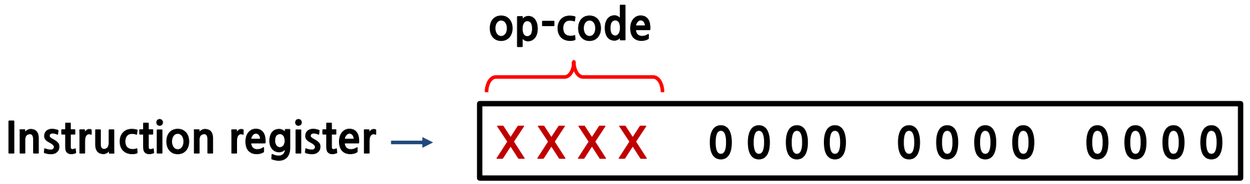
  - 명령어 내의 op-code 가 지정하는 연산을 실행하기 위하여 제어기억장치 내에 실행 사이클의 마이크로서브루틴이 프로그램 되어있는 **시작 주소를 찾아가는 방법**이다.
  - 명령어 op-code의 비트를 **사상함수의 특정 비트 패턴과 조합**하는 방법이다.

마이크로프로그램

3) 마이크로프로그램

- 실행 사이클의 마이크로서브루틴을 찾기 위한 사상(Mapping)방법

사상방법



Mapping function →

$f(X\,X\,X\,X) = (1\,X\,X\,X\,X\,0\,0)_2$

$= (1\times2^6 + X\times2^5 + X\times2^4 + X\times2^3 + X\times2^2 + 0\times2^1 + 0\times2^0)_{10}$

$= (64 + ? + ? + ? + ? + 0 + 0)_{10}$

마이크로프로그램

3) 마이크로프로그램

- 사상(Mapping)방법에 따른 실행 사이클의 마이크로서브루틴 주소

Instruction	op-code	마이크로서브루틴의 시작 주소
NOP	0 0 0 0	f (0 0 0 0) = 1 0 0 0 0 0 0 = 64_{10}
LOAD(I)	0 0 0 1	f (0 0 0 1) = 1 0 0 0 1 0 0 = 68_{10}
STORE(I)	0 0 1 0	f (0 0 1 0) = 1 0 0 1 0 0 0 = 72_{10}
ADD	0 0 1 1	f (0 0 1 1) = 1 0 0 1 1 0 0 = 76_{10}
SUB	0 1 0 0	f (0 1 0 0) = 1 0 1 0 0 0 0 = 80_{10}
JUMP	0 1 0 1	f (0 1 0 1) = 1 0 1 0 1 0 0 = 84_{10}

마이크로프로그램

3) 마이크로프로그램

- NOP 명령어 - 실행 사이클의 마이크로서브루틴

```
        ORG 64
NOP :   NONE   INCPC   U  JMP    FETCH ; PC ← PC+1
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
1000000	000	001	00	00	0000000

마이크로프로그램

3) 마이크로프로그램

- LOAD 명령어 - 실행 사이클의 마이크로서브루틴

```
        ORG 68
LOAD :  NONE  NONE  I  CALL  INDRT ;
        IRTAR NONE  U  JMP   NEXT  ; MAR ← IR(addr)
        READ  NONE  U  JMP   NEXT  ; MDR ← M[MAR]
        DRTAC NONE  U  JMP   FETCH ; AC ← MDR
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
1000100	000	000	01	01	0000100
1000101	010	000	00	00	1000110
1000110	100	000	00	00	1000111
1000111	101	000	00	00	0000000

마이크로프로그램

3) 마이크로프로그램

- STORE 명령어 - 실행 사이클의 마이크로서브루틴

```
        ORG 72
STORE : NONE  NONE   I   CALL  INDRT ;
        IRTAR NONE   U   JMP   NEXT  ; MAR ← IR(addr)
        NONE  ACTDR  U   JMP   NEXT  ; MDR ← AC
        WRITE NONE   U   JMP   FETCH ; M[MAR] ← MDR
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
1001000	000	000	01	01	0000100
1001001	010	000	00	00	1001010
1001010	000	010	00	00	1001011
1001011	111	000	00	00	0000000

마이크로프로그램

3) 마이크로프로그램

- ADD 명령어 - 실행 사이클의 마이크로서브루틴

```
        ORG 76
ADD :   IRTAR   NONE    U   JMP   NEXT   ; MAR ← IR(addr)
        READ    NONE    U   JMP   NEXT   ; MDR ← M[MAR]
        ADD     NONE    U   JMP   FETCH  ; AC ← AC + MDR
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
1001100	010	000	00	00	1001101
1001101	100	000	00	00	1001110
1001110	011	000	00	00	0000000

마이크로프로그램

3) 마이크로프로그램

- SUB 명령어 - 실행 사이클의 마이크로서브루틴

```
        ORG 80
SUB :   IRTAR   NONE    U   JMP   NEXT   ; MAR ← IR(addr)
        READ    NONE    U   JMP   NEXT   ; MDR ← M[MAR]
        NONE    SUB     U   JMP   FETCH  ; AC ← AC - MDR
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
1010000	010	000	00	00	1010001
1010001	100	000	00	00	1010010
1010010	000	110	00	00	0000000

마이크로프로그램

3) 마이크로프로그램

- JUMP 명령어 - 실행 사이클의 마이크로서브루틴

```
        ORG 84
JUMP :  NONE    IRTPC    U  JMP     FETCH ; PC ← IR(addr)
```

Addr	μ-OP1	μ-OP2	CD	BR	ADF
1010100	000	111	00	00	0000000