어셈블프로그램 이해(2)

6.6 산술 및 논리 연산의 프로그래밍

- 명령어의 수
  - 대형 : 수백개
  - 소형 : 수십개
- 곱셈 프로그램
  - 8비트 두 개의 이진수의 곱셈
  - 승수 Y의 비트를 검사하여 1이면 피승수 x를 더한다. 한번 수행시마다 왼쪽으로 한 비트씩 시프트.

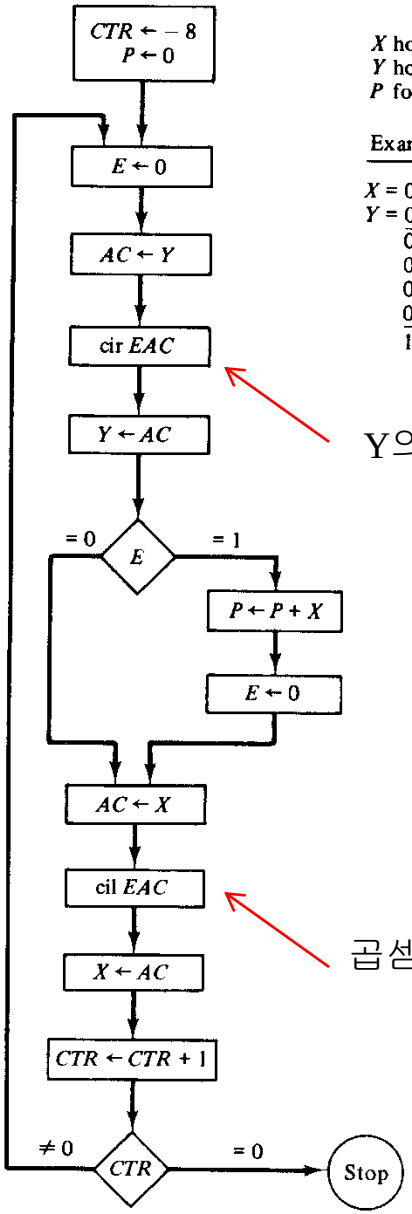


그림 6-3 곱셈 프로그램을 위한 흐름도

X holds the multiplicand
Y holds the multiplier
P forms the product

Example with four significant digits

		P
X =	0000 1111	
Y =	0000 1011	0000 0000
	0000 1111	0000 1111
	0001 1110	0010 1101
	0000 0000	0010 1101
	0111 1000	1010 0101
	1010 0101	

Y의 LSB를 E 레지스터에 넣어 검사하기 위해

곱셈 동작의 프로그래밍 단계적 과정
8번의 loop
X는 피승수 Y는 승수
CTR은 –8에 세트되고 P는 0에 세트
CTR이 0이 될 때까지 루프 반복
결과는 P에 저장

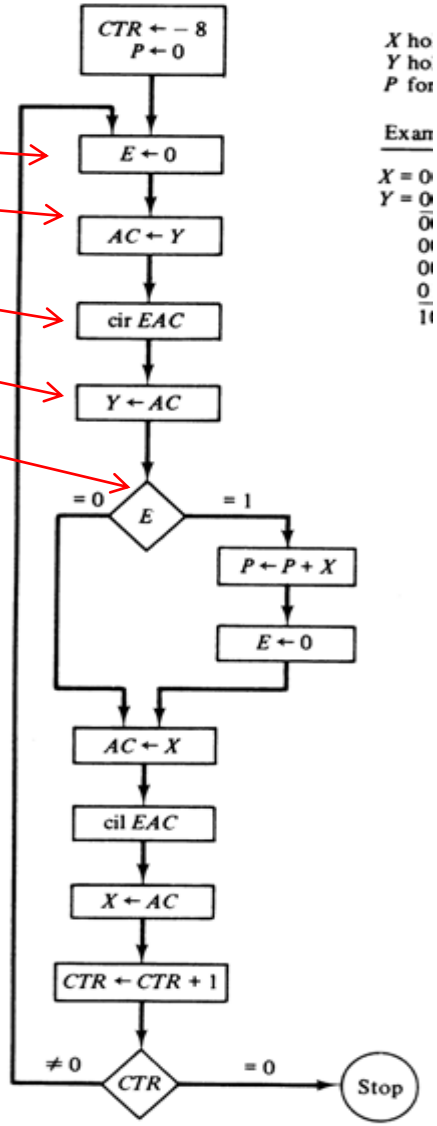
곱셈의 다음 자리가 1인 경우에 1비트 자리 올려 더하기위해

두 개의 양수를 곱하는 프로그램

```
        ORG 100
LOP,    CLE
        LDA Y
        CIR
        STA Y
        SZE
        BUN ONE
        BUN ZRO
ONE,    LDA X
        ADD P
        STA P
        CLE
ZRO,    LDA X
        CIL
        STA X
        ISZ CTR
        BUN LOP
        HLT
CTR,    DEC -8
X,      HEX 000F
Y,      HEX 000B
P,      HEX 0
        END
```

초기값을 **X**와 **Y**에 로드
카운터에 **8**을 **P**에 **0**을 세트

//skip if E is zero



논리연산 및 시프트 연산

```
LDA A          /A의 값을 AC에 로드한다.
CMA            /AC를 1의 보수를 취한다.
STA TMP        /AC의 값을 TMP에 저장
LDA B          /B의 값을 AC에 로드한다.
CMA            /AC를 1의 보수를 취한다.
AND TMP        /AC와 TMP를 AND 연산한다.
CMA            /AC를 1의 보수를 취한다.
```

$A \rightarrow \overline{A}$

$B \rightarrow \overline{B}$

$\overline{A} \wedge \overline{B}$

$\overline{\overline{A} \wedge \overline{B}} = A \vee B$

순환명령을 이용한 논리 시프트(오른쪽 논리 시프트)

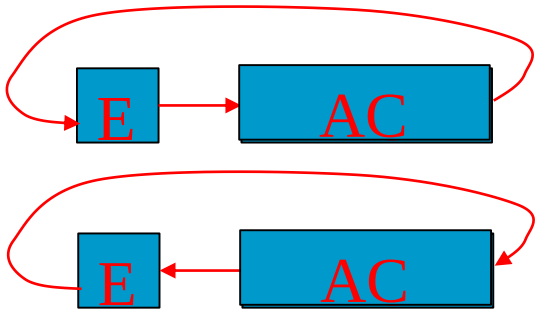
CLE

CIR

순환명령을 이용한 논리 시프트(왼쪽 논리 시프트)

CLE

CIL




```
CLE     /E를 0으로 리셋
SPA     /AC가 양수이면 다음 문장 skip
CME     /AC가 음수일 때 수행문장, E=1로
CIR     /E와 AC를 circulate
```

오른쪽 산술시프트

: AC의 MSB에 1을 복사하는 기능

6.7 서브루틴

BSA(Branch and save return address) 명령어

$D_5T_4: M[AR] \leftarrow PC,\ AR \leftarrow AR+1$

$D_5T_5: PC \leftarrow AR,\ SC \leftarrow 0$

서브루틴 : 프로그램내에서 여러 번 사용되는 공통된 명령어

Location			
		ORG 100	
		LDA X	
100		BSA SH4	Subroutine call
101		STA X	
102		LDA Y	
103		BSA SH4	Subroutine call
104		STA Y	
105		HLT	
106	X,	HEX 1234	
107	Y,	HEX 4321	
108			
	SH4,	HEX 0 ← 102	
109		CIL	
10A		CIL	
10B		CIL	Subroutine
10C		CIL	
10D		AND MSK	
10E		BUN SH4 I	
10F	MSK,	HEX FFF0	
110		END	

주소 **X**의 데이터를 **AC**에 로드하고 **SH4**에 **102**를 저장하고 **SH4+1**의 위치를 실행하고 **10E**의 **BUN** 명령에 의해 간접주소 지정명령이므로 **SH4**의 **102** 번지로 부터 다음 명령을 시작.

Subroutine linkage : 서브루틴으로 분기하고 주 프로그램으로 리턴하는 과정.

수행결과
X=2340
Y=3210

BSA : Save return address in m and branch to m+1

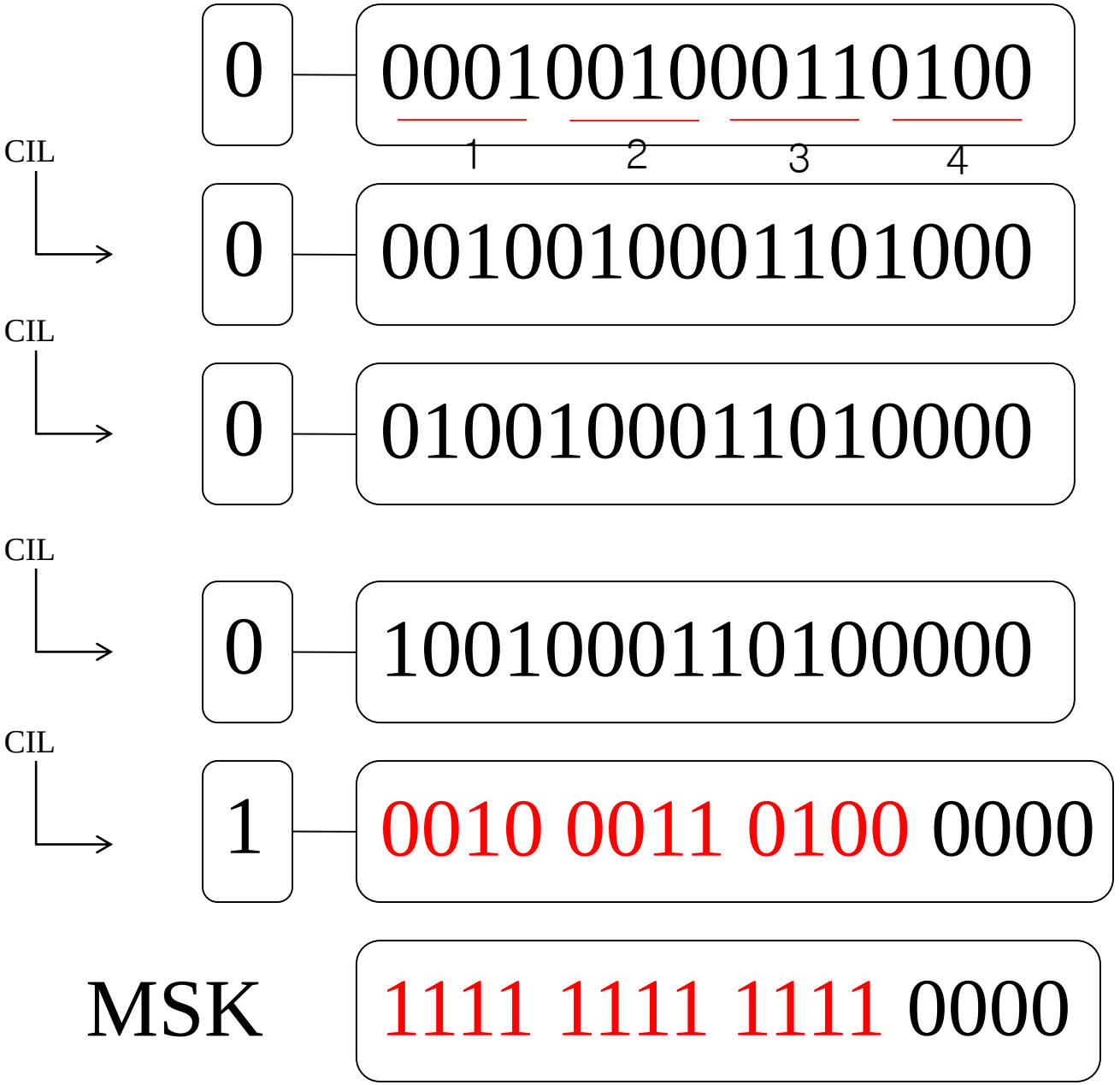

0
0001001000110100
1
2
3
4
CIL
0
0010010001101000
CIL
0
0100100011010000
CIL
0
1001000110100000
CIL
1
0010 0011 0100 0000
MSK
1111 1111 1111 0000

표 6-17 파라미터 링키지를 보이기 위한 프로그램

```
Location
                    ORG 200
200                 LDA X
201                 BSA OR
202                 HEX 3AF6
203                 STA Y
204                 HLT
205         X,      HEX 7B95
206         Y,      HEX 0
207         OR,     HEX 0       ← 202
208                 CMA
209                 STA TMP
20A                 LDA OR I
20B                 CMA
20C                 AND TMP
20D                 CMA
20E                 ISZ OR
20F                 BUN OR I
210         TMP,    HEX 0
                    END
```

두개 이상의 파라미터를 어떻게 서브루틴에 전달하여 처리 할 것인가의 문제.....

BSA : Save return address in m and branch to m+1

Y에 최종의 결과가 저장되고 프로그램 종료됨. 결과값은?

ISZ : Increment M and skip if zero
; OR에 202번지가 203번지로 변경됨

7B95 : 0111 1011 1001 0101

3AF6 : 0011 1010 1111 0110

$\overline{7B95}$ = 1000 0100 0110 1010

$\overline{3AF6}$ = 1100 0101 0000 1001

$\overline{7B95} \wedge \overline{3AF6}$ = 1000 0100 0000 1000

= 8408

$\overline{\overline{7B95} \wedge \overline{3AF6}}$ = 0111 1011 1111 0111

= 7BF7

Location		
		ORG 200
200		LDA X
201		BSA OR
202		HEX 3AF6
203		STA Y
204		HLT
205	X,	HEX 7B95
206	Y,	HEX 0
207	OR,	HEX 0
208		CMA
209		STA TMP
20A		LDA OR I
20B		CMA
20C		AND TMP
20D		CMA
20E		ISZ OR
20F		BUN OR I
210	TMP,	HEX 0
		END

표 6-18 데이터의 블록을 이동시키는 서브루틴

```
300             BSA MVE
301             HEX 100
302             HEX 200
303             DEC -16
304             HLT
305     MVE,    HEX 0
306             LDA MVE I
307             STA PT1
308             ISZ MVE
309             LDA MVE I
30A             STA PT2
30B             ISZ MVE
30C             LDA MVE I
30D             STA CTR
30E             ISZ MVE
30F     LOP,    LDA PT1 I
310             STA PT2 I
311             ISZ PT1
312             ISZ PT2
313             ISZ CTR
314             BUN LOP
315             BUN MVE I
316     PT1,    -
317     PT2,    -
318     CTR,    -
```

BSA : Save return address in m and branch to m+1

어떤 값이 채워지는가?

PT1 = 100

ISZ : Increment M and skip if zero

PT2 = 200

CTR = -16

100번지에서 시작되는 데이터블럭을 200번지로 시작하는 블록으로 이동 16번 반복

6.8 입출력 프로그래밍

```
(a) input a character
CIF,    SKI             /입력 플래그 검사
        BUN CIF         /flag=0이면 CIF로
        INP             /flag=1, 문자 입력
        OUT             /출력을 프린트
        STA CHR         /문자를 CHR에 저장
        HLT
CHR,    --
(b) output one character
        LDA CHR         /CHR의 문자를 AC로 로드
COF,    SKO             /출력 flag 검사
        BUN COF         /flag=0이면 COF로
        OUT             /flag=1이면 문자 출력
        HLT
CHR,    HEX 0057        /문자 "W"
```

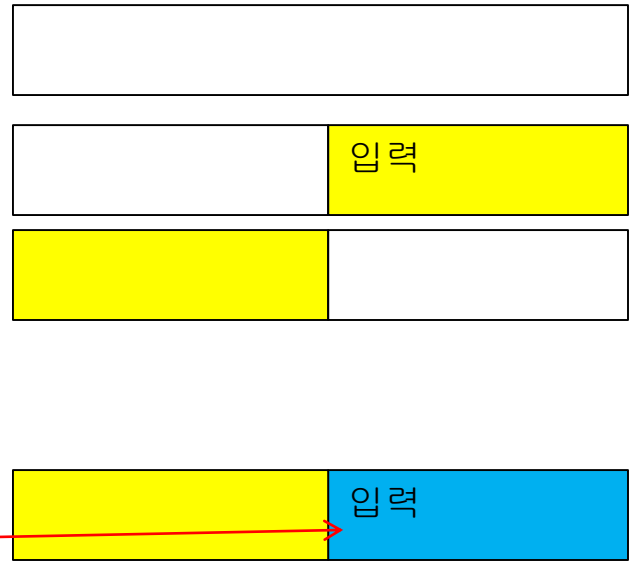
SKI : Skip if Input flag is on
SKO : Skip if Output flag is on

ski에서 입력이 되는지 체크 데이터가 입력되면 **flag=1**이 되어 **INP** 명령으로 분기 이 명령은 받은 문자를 **AC**에 전송

두 문자를 입력하여 **packing** 하는 서브루틴

SKI : Skip if Input flag is on
SKO : Skip if Output flag is on

```
IN2,    -
FST,    SKI
        BUN  FST
        INP
        OUT
        BSA  SH4
        BSA  SH4
SCD,    SKI
        BUN  SCD
        INP
        OUT
        BUN  IN2  I
```



버퍼(Buffer)에 입력문자를 저장

```
        LDA  ADS
        STA  PTR
LOP,    BSA  IN2
        STA  PTR I
        ISZ  PTR
        BUN  LOP
        HLT
ADS,    HEX  500
PTR,    HEX  0
```

AC ← 500

PTR ← AC

AC에 두문자 받음

500번지 ← AC

PTR ← PTR+1

500번지부터 16비트(2문자)의 데이터를 저장한다. PTR이 0이 될때까지

IN2 : 두 문자를 입력받아 packing 하는 서브루틴(앞페이지)

두 워드를 비교하는 프로그램

```
        LDA  WD1
        CMA
        INC
        ADD  WD2
        SZA
        BUN  UEQ
        BUN  EQL
WD1,    -
WD2,    -
```

WD2-WD1=0 이면 Skip, 즉 WD1과 WD2가 같으면 Skip

프로그램 인터럽트

```
Locat ion
0           ZRO,    —
1                   BUN SRV
100                 CLA
101                 ION
102                 LDA X
103                 ADD Y
104                 STA Z
.                   .
.                   .
200         SRV,    STA SAC
                    CIR
                    STA SE
                    SKI
                    BUN NXT
                    INP
                    OUT
                    STA PT1 I
                    ISZ PT1
            NXT,    SKO
                    BUN EXT
                    LDA PT2 I
                    OUT
                    ISZ PT2
            EXT,    LDA SE
                    CIL
                    LDA SAC
                    ION
                    BUN ZRO I
            SAC,    —
            SE,     —
            PT1,    —
            PT2,    —
```

Return address 여기에 저장

인터럽트 서비스 루틴

수고하셨습니다!