# MITY SERVO

VEA 시리즈용

# QMCL 해설서

2002. 04. 25

# 목 차

# 1 QMCL 언어 설명

QMCL에는 표면 언어와 중간 언어가 있습니다. 컴파일러를 통해 사용할 때는 표면(고급) 언어로 프로그램을 작성할 수 있습니다. 컴파일러를 이용하는 일 없이 직접 MITY 서보에 생명
령을 주고 싶은, 혹은 프로그램을 입력하고 싶은 경우는 중간 언어를 사용하면(자) 편리합니다.

| 기계어 | 명령어 | 기능 | 설명 |
|---|---|---|---|
| C0 | C0. (OUT 0) | Out 신호 출력 | 8 bit의 출력 신호 |
| C1 | C1. (OUT 1) | ″ | ″ |
| C2 | C2. (OUT 2) | ″ | ″ |
| C3 | C3. (OUT 3) | ″ | ″ |
| C4 | C4. (IN 0) | 입력 신호 독해 | 8 bit의 입력 신호 |
| C5 | C5. (IN 1) | ″ | ″ |
| C6 | C6. (IN 2) | ″ | ″ |
| C7 | C7. (IN 3) | | ″ |
| C8 | C8 | 송신 명령 | MITY끼리의 송신 명령 |
| C9 | C9 | 수신 명령 | MITY끼리의 수신 명령 |
| CA | CA | 10진으로 표시 | |
| CB | PLS2 | 2 nd닫코타″ 카운트 | 2 nd엔코더의 카운트치가 격납. (4Byte) |
| CC | | | |
| CD | | | |
| CE | $ | 1바이트 HEX | 1바이트 16 진수 지정. |
| CF | $ | 2바이트 HEX | 2바이트 16 진수 지정. |

| 기계어 | 명령어 | 기능 | 설명 |
|---|---|---|---|
| D0 | = | 이콜 | 행선지두의 경우는 NOP. |
| D1 | + | 가산 기호 | |
| D2 | − | 감산 기호 | |
| D3 | × | 곱셈 기호 | |
| D4 | ÷ | 제산 기호 | |
| D5 | (쉬프트 L)×2n | 수치의 왼쪽 쉬프트 | 수치×2 n의 연산 |
| D6 | (쉬프트 R)/2n | 수치의 오른쪽 쉬프트 | 수치÷2 n의 연산 |
| D7 | AND | 논리적 | |
| D8 | OR | 논리합 | |
| D9 | EOR | 배타적 논리합 | |
| DA | NOT | 반전 | |
| DB | ABS | 절대치 | |
| DC | PEEK | 1 바이트 데이터 리드 | 특정 번지부터 1바이트 데이터 보기. |
| DD | POKE | 1 바이트 데이터 라이트 | 특정 번지에 1바이트 데이터 기록. |
| DE | DPEEK | 2 바이트 데이터 리드 | 특정 번지부터 2바이트 데이터 보기. |
| DF | DPOKE | 2 바이트 데이터 라이트 | 특정 번지에 2바이트 데이터 기록. |
| E0 | HZS | 현재 지령 주파수 | 현재 출력하고 있는 주파수가 격납. |
| E1 | HZP | 설정 지령 주파수 | 출력 주파수 지령 명령. |
| E2 | PLS | 1 st엔코더 카운트 | 1 st엔코더의 카운트치가 격납. (4Byte) |
| E3 | POS | 타켓 포지션 | 위치 결정 목표치 설정. |

| 기계어 | 명령어 | 기능 | 설명 |
|---|---|---|---|
| E4 | MAXHZ | 위치결정 최대 주파수 | 위치 결정시의 상한 주파수를 설정. |
| E5 | MINHZ | 위치결정 저속 주파수 | 위치 결정 저속시의 감속율. |
| E6 | VFA | 기준 토르크 | 벡터 제어시의 열매 토르크(READ ONLY) |
| | | 주파수 비례 전압 | 고속시의 토르크 컨트롤 (VF모드[PWM:2]시 유효) |
| E7 | VFB | 토르크 리밋트 | 벡터 제어시의 토르크의 상한치 |
| | | 바이어스 전압 | 저속시 토르크 컨트롤 (VF모드[PWM:2]시 유효) |
| E8 | SFT | 가감 속도도 | 설정 주파수에의 가감 속도도를 설정. |
| E9 | PSG | position gain | 목표치에의 기울기를 설정. 위치 결정 제어의 스타트·스톳후° 신호 도 겸용 |
| EA | TIC1 | 내부 타이머 1 설정 | 2. 44 msec의 타임 카운트 |
| EB | TIC2 | 내부 타이머 2 설정 | 2. 44 msec의 타임 카운트 |
| EC | HZF | 속도 검출 | 피드백 주파수 |
| ED | PLSI | Z상입력시 펄스 | Z상입력시로 설정치를 펄스치에 치환 |
| EE | KED | 키 코드 | 키 입력의 내용이 격납. |
| EF | SEVCC | 모터 전원제어 | =0 파워 OFF , =1 파워 ON |
| F0 | JSR | 서브루틴 실행 | 써브루틴 실행. |
| F1 | JMP | 무조건 점프 | 무조건 점프. |
| F2 | JMI | 조건 점프 | 결과가 <0이라면 점프. |
| F3 | JEQ | 조건 점프 | 결과가=0이라면 점프. |
| F4 | JPL | 조건 점프 | 결과가≥0이라면 점프. |
| F5 | JNE | 조건 점프 | 결과가≠0이라면 점프. |
| F6 | BRA | 상대 점프 | 설정 수치분 다음 행보다 점프. |

| 기계어 | 명령어 | 기능 | 설명 |
|---|---|---|---|
| F7 | CALL | 마신뚭사후˝ 루틴 | 기계어 써브루틴 실행. |
| F8 | ONTIM1 | 서브루틴 시작 명령 | 65 msec 마다 써브루틴 실행. |
| F9 | ONTIM2 | 서브루틴 시작 명령 | 가변 시간 마다의 써브루틴 실행. |
| FA | RTS | 귀환 명령 | 사후˝ 루틴보다 메인 루틴에 귀환. |
| FB | OFFRTS | 불귀환명령 | 메인 루틴에의 귀가를 캔슬. |
| FC | AOFRTS | 전불귀환명령 | 메인 루틴에의 귀가를 모두 캔슬. |
| FD | SCNO | 시리얼 채널 | 시리얼 통신의 자기 No.(을)를 설정 |
| FE | | | |
| FF | STOP | 스톱 | 프로그램 스톱 |

# 2 QMCL 상세

## 2-1. 변수

(1) 유저 변수 (2 바이트메모리 A0~A9, B0~B9) 20 종류

변수명 A0, A1, A2, A3, A4, A5, A6, A7, A8, A9, B0, B1, B2, B3, B4, B5, B6, B7, B8, B9

유저 변수와는 프로그램을 작성할 때에, 유저를 자유롭게 사용할 수 있는 2바이트(8 비트×2)의 메모리에리아로, 프로그램상, 16 비트(0~65535)의 데이터 격납에 사용합니다.

사용예 A1=A0×A9+B8

A0의 내용과 A9의 내용을 곱셈해, B8의 내용을 더해 결과를 A1에 격납합니다. (다만, 연산 결과가 2바이트를 넘어서는 안 된다) 만약 넘었을 경우는 오버플로우 한 수치가 결과가 됩니다.

(2바이트를 넘어 버릴 때는, 아래와 같은 특정 변수를 사용합니다.)

(2) 특별 변수 (AA~AF, BA~BF)···아래와 같이표 6 페어의 4바이트장의 메모리에리아가 됩니다.

변수명 AA, AB, AC, AD, AE, AF, BA, BB, BC, BD, BE, BF

이 변수는 4바이트의 곱셈과 나눗셈산에 이용하기 위한 변수입니다.

※OS－270 이후 4 byte 선언 불요

[4바이트 사용 플래그($FF02)를 1으로 하는 것으로 특별 변수가 됩니다.
례] POKE $FF02 1]

BE에 결과의 상위 2바이트, BF에 결과의 하위 2바이트가 격납됩니다.

**단, 4바이트를 넘는 수치끼리의 연산은 할 수 없습니다.**

**4바이트의 연산을 실행하는 경우, 대답은 반드시 4바이트 변수에 격납해 주세요.**

BE=A1×A0
BE=BE÷A1

사용예원주의 계산예
(소수점 이하의 계산)
BE=A0×314
BE=BE/100
A1=BF

| 상위 | 하위 |
|---|---|
| AA | AB |
| AC | AD |
| AE | AF |
| BA | BB |
| BC | BD |
| BE | BF |

(3) 시스템 변수

MITY 서보를 동작시키기 위해서(때문에) 시스템으로 사용하고 있는 변수명에 이하의 것이 있습니다.

a) HZS [중간 코드 E0]

서보 동작중, 자동적으로 현재의 주파수가 격납됩니다.

사용예

A0=HZS

(으)로 하면(자), 변수 A0에 현재의 주파수치가 격납됩니다. 수치는 분해가능 1/32 Hz의 정수배로 나타내집니다. (HZS)=160이라면, 실제의 지령 주파수는 5 Hz입니다.

※이 변수는 유저가 자유롭게 세트 해서는 안됩니다.

b) PLS [중간 코드 : E2]

엔코더로부터의 입력 펄스를 항상 카운트 한 수치가 격납됩니다.

속도 제어, 위치 제어든 엔코더를 접속하고 있는 한, 자동적으로 펄스를 카운트 해, 격납합니다.

AA=PLS

(으)로 하면(자) 변수 AA에 현재의 카운트치의 상위 2바이트 AB에 하위 2바이트가 격납됩니다. (4바이트 사용시)

PLS=1000

(으)로 하면(자), 지금까지의 카운트치 (와)는 관계없는 것으로 1000이 세트트 되고 이 후는 이 값으로부터 상/하 카운트 되게 됩니다.

c) HZF [중간 코드 : EC]

엔코더의 펄스를 읽어내, 주파수의 수치로서 연산한 수치가 격납됩니다. 수치는 분해가능 1/32 Hz의 정수배로 나타내집니다.

HZF가 1920이라면, 실제의 피드백 주파수는 60 Hz입니다.

d) KED [중간 코드 : EE]

키 입력을 실시했을 때의 키코드 내용이 격납됩니다. 이 내용은 65 msec마다 갱신됩니다. 키 입력되어 있지 않을 때는-1(FFFF)이 되고 있습니다.

4자릿수의 문자를 읽어내는 예

```
        ONTIM1  T00
            :
T00 JMI T04 KED     ; 키 입력되고 있는지를 판정
    JPL T10 KED-10  ; 0~9까지의 키인지를 판정
    JEQ T02 A1      ; 1회째인가 계속(형상)인지를 판정
    JPL T02 A0-1000     ; 4자리수 이내인지를 판정
    A0=A0*10+KED    ; 자리수 인상
    JMP T04
T02 A0=KED          ; 키 수치를 A0에 격납
    A1=1            ; 다음으로부터는  자리수 인상
T04 CA40=A0             ; 표시
    RTS             ; 메인 루틴에 귀환
T10     :
```

2-2. 입력, 출력 명령

(1) 출력 C0 [중간 코드 : C0] 외부에 ON/OFF 정보를 출력합니다.

```
C0=128          ; D7를 출력합니다.
C0=C0  0R  1    ; D7를 유지해 D0를 출력합니다.
```

(2) 입력 C4 [중간 코드 C4] 외부의 접점 정보를 입력하는 명령입니다.

사용예

B0=C4 AND 128

외부의 신호를 입력해 D7=1으로 논리적을 요구하고 그 값을 B0에 격납합니다.

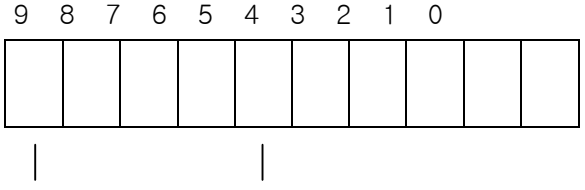
B0=128또는 0(B0=0 : OFF, B0≠0 : ON)

2-3. 표시

(1) 10 진수 표시 CAmn [중간 코드 CAχχ]

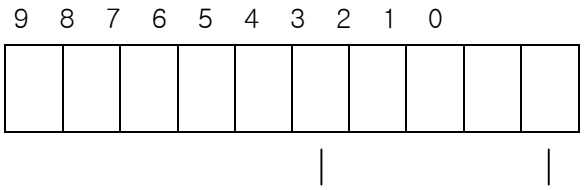
표시부의 m자리수로부터 n자리수에 변수명(혹은 정수)으로 지정한 내용이 표시됩니다.

CA95=A0



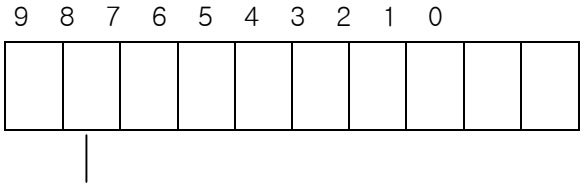

9~5에 A0의 내용이 10 진수로 표시되고る

CA40=PLS－1000




4~0에 시스템 변수 PLS(엔코더 카운트치)로부터 1000을 뺀 수치가 10 진수로 표시된다。

CA88=16




8의 자리수에 16(=공백)이 표시된다

1자리수의 특수 코드는 이하와 같습니다.

| | | |
|---|---|---|
| 10…A | 16…공백 | 22…h |
| 11…B | 17… － | 23…o |
| 12…C | 18… / | 24…P |
| 13…D | 19…H | 25…r |
| 14…E | 20…J | 26…U |
| 15…F | 21…L | 27…y |

**※ 정·부에 변화하는 수치를 표시하기 위한 프로그램예를 나타냅니다. HZS에는 정·부의 값이 들어가 있을 가능성이 있습니다.**

프로그램예

```
        JMI B00 HZS          ; (HZS)<0이라면 B00에 점프 합니다.
        CA40=HZS÷23          ; 0.125×(HZS)를 4~0자리수에 표시합니다.
        RTS
B00     CA30=ABS HZS÷23 ; 0.125×(HZS)의 절대치를 요구해 3~0에 표시합니다.
        CA44=17              ; 4자리수에(－)를 표시합니다.
        RTS
```

**********

***(2) 16 진수 표시 CBmn [중간 코드 CBχ χ ]

표시부의 m자리수로부터 n자리수에 변수명(or정수)으로 지정한 내용이 16진수로 표시됩니다. ***

**********

2－4. 수치 지정

(1) 10 진수

정수의 10 진수 지정은 통상의 수를 이용할 때와 같게 실시할 수가 있습니다.

A0=100　　A0에 100을 격납합니다.

A1=－3　　A1에-3을 격납합니다. (내용은 $FFFD)

(2) 16 진수

프로그램 중(안)에서 16 진수를 사용하고 싶을 때에는 그 수의 선두에 $를 붙이는 것으로 지정할 수가 있습니다.

$77 [중간 코드 : CE77]

$123 [중간 코드 : CF0123]

$(을)를 붙이면(자) 컴파일러에 의해, 표면 언어를 중간 언어로 변환할 때에 자동적으로, CE(1바이트), CF(2바이트)중 한쪽이 판단되어 수치 변환됩니다.

$77=119(10진)

$123=291(10진)

의 의미가 됩니다.

2－5. 사칙 및 논리 연산 명령

(1) 등호(=) [중간 코드 D0]

이 기호는 또 특수한 의미 [NOP : No Operation] 를 가지고 있어 명령문의 선두에 D0(=)가 있으면(자) 그 행의 명령은 실행되지 않습니다.

(2) 가산(＋) [중간 코드 : D1]

사용예 A0=B0＋B1　　(A0D0B0D1B1)…중간 언어

변수 B0에 B1를 가산해, A0에 격납됩니다.

(3) 감산(－) [중간 코드 : D2]

사용예 B0=B0－1　　(B0D0B0D201)…중간 언어

변수 B0의 내용으로부터 1을 감산한 값이 새롭게 B0에 격납됩니다.

(4) 곱셈(×) [중간 코드 : D3]

사용예 A1=A2×10　　(A1D0A2D310)…중간 언어

변수 A2의 내용에 10을 곱셈한 값이 변수 A1에 격납됩니다.

(5) 제산(÷) [중간 코드 : D4]

사용예 B2=B0÷5　　(B2D0B0D405)…중간 언어

B0의 내용을 5로 제산한 값이 B2에 격납됩니다.

(6) 왼쪽 쉬프트(×2 n) [중간 코드 : D5]

왼쪽에의 n비트 쉬프트 명령입니다. 이것은 있는 값을 2 n 배가 되는 것과 같습니다.

사용예

A0=B1×21(=B1×2)→(A0D0B1D501)

B1의 내용을 왼쪽에 1 비트 쉬프트 해 그 값을 A0에 격납합니다.

A0=B2×23(=B2×8)→(A0D0B2D508)

B2의 내용을 왼쪽에 3 비트 쉬프트 해 그 값을 A0에 격납합니다.

(7) 오른쪽 쉬프트(÷2 n) [중간 코드 : D6]

오른쪽에의 n비트 쉬프트 명령입니다. 이것은 있는 수를 2 n로 제산하는 것과 같습니다. 사용예 A0=B1÷23 (A0D0B1D603)

(A0=B1÷8) (A0D0B1D408)

B1의 내용을 오른쪽에 3 비트 쉬프트 해 그 내용이 A0에 격납됩니다. 이 때 B1의 내용은 변화하지 않습니다.

(8) 논리적(AND) [중간 코드 : D7]

2개의 입력(변수 혹은 정수)의 논리적인 적을 요구합니다. 진리치표는 아래 표대로입니다.

2 입력의 대응하는 비트가 함께 1때, 결과가 1이 됩니다.

| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

진리치표

예를 들면 $55 AND $33 →$11

| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | =$55 |
|---|---|---|---|---|---|---|---|---|

AND

| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | =$33 |
|---|---|---|---|---|---|---|---|---|

↓

| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | =$11 |
|---|---|---|---|---|---|---|---|---|

(메모리의 내용을 나타내고 있습니다.)

※QMCL에서는 C0=C0 AND 128 (와)과 같은 명령에 의해 D7비트만 ON 한 채로 외를 OFF로서 출력을 실시할 수도 있습니다.

(9) 논리합(OR) [중간 코드 : D8]

2개의 입력(변수 혹은 정수)의 논리적인 화를 요구합니다. 진리치표는 아래 표대로입니다. 2 입력의 대응하는 비트중 한쪽이 1때 결과는 1이 됩니다.

| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

진리치표

예를 들면 $55 OR $22→$77

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | =$55 |
| OR | | | | | | | | |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | =$22 |
| ↓ | | | | | | | | |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | =$77 |

※QMCL에서는 C0=C0 OR 64 명령에 의해 벌써 출력하고 있는 비트는 그대로

64( | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | )(와)과의 논리합을 요구하는 것으로

비트 6을 ON 한 출력을 낼 수가 있습니다.
(위는 C0=C0 OR $40과도 동일합니다)

(10) 배타적 논리합(EOR) [중간 코드 : D9]

2개의 입력(변수 혹은 정수)의 배타적인 논리합을 요구합니다. 그 진리치표는 아래 표대로입니다. 바꿔 말하면(자) 2 입력의 대응하는 비트가 같은 때는 0, 차이가 날 때는 1으로 합니다.

| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

진리치표

(11) 논리 반전(NOT) [중간 코드 : DA]

입력의 논리를 반전(Not 혹은 Invert라고도 한다)합니다.

사용예 A1=NOT A0 (A1D0DAA0)

A0의 내용을 반전해 A1에 격납합니다.

A0=1 때 A1=$FFFE가 됩니다.

A0=0 때 A1=$FFFF가 됩니다.

(12) 절대치(ABS) [중간 코드 : DB]

주어진 변수의 절대치를 요구합니다.

X=-3 때 Y=ABS X=3입니다.

사용예 A1=ABS A0 (A1D0DBA0)

A0의 내용의 절대치를 구해라 A1에 격납합니다.

A0=1 때 A1=1이 됩니다.

A0=$FFFF 때 A1=1이 됩니다.

A0=$FFFE 때 A1=2가 됩니다.

## 2-6. 메모리 내용에의 직접 액세스(읽어내 기입) 명령

(1) 내용(데이터)의 읽기

a) 1바이트 읽기 PEEK [중간 코드 : DC]

b) 2바이트 읽기 DPEEK [중간 코드 : DE]

명령으로 지정된 번지의 내용을 읽어내 지정된 변수에 격납합니다.

사용예 DPEEK A0 $FE50(DEA0CFFE50)

$FE50와 $FE51 번지의 내용을 읽어내 A0에 격납합니다.

PEEK B0 $FF0D(DCB0CFFF0D)

$FF0D 번지의 내용을 읽어내 B0에 격납합니다.

DPEEK A0 A2(DEA0A2)

A2의 내용(수치)을 번지로 해, 그 번지와 다음의 번지의 내용을 읽어내 와 A0에 격납합니다.

(2) 데이터의 기입

a) 1바이트 데이터의 기입 POKE [중간 코드 : DD]

b) 2바이트 데이터의 기입 DPOKE [중간 코드 : DF]

명령으로 지정한 번지에 변수의 내용을 기입합니다.

사용예 DPOKE $F200 A0(DFCFF200A0)

A0의 내용을 $F200 번지와 $F201 번지에 기입합니다.

POKE $F300 B0(DDCFF300B0)

B0의 하위 바이트의 내용을 $F300 번지에 기입합니다.

DPOKE A2 A0(DFA2A0)

A0의 내용을 A2의 내용 번지와 그 다음의 번지에 기입합니다.

2-7. 모터 제어를 위한 명령

(1) 주파수 비례 전압 설정(VFA) [중간 코드 : E6]

주파수 에 비례해 출력하고 싶은 전압의 기울기를 설정합니다.

아래와 같이(그림 2-1)에 참고도를 나타내고 있습니다. (VF모드시 유효)

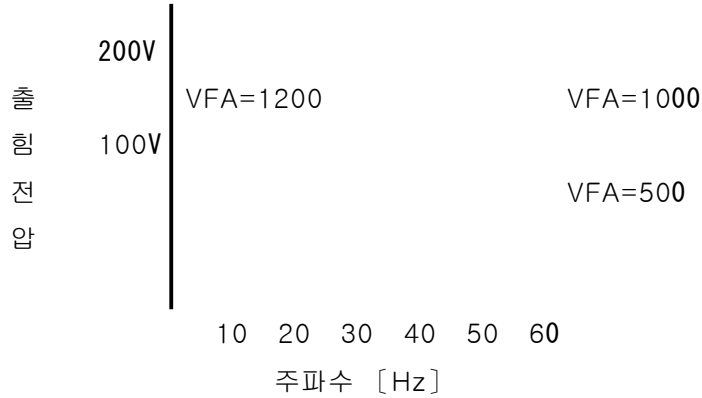



그림 2-1 VFA와 주파수

(2) 바이어스 전압 설정(VFB) [중간 코드 : E7]

VF모드 이외 [PWM Mode 0,1,3]

각 주파수에 있어서의 최대 토르크를 1000으로서 토르크 제어가 가능합니다. 기준으로서 일반의 60 Hz의 연속정격토르크의 약 3배를 1000으로 설정해 보세요.

VF모드시 [PWM Mode 2,4,5,6]

영주파수로 출력하고 싶은 전압을 설정합니다. 아래와 같이(그림 2-2)에 참고도를 나타냅니다.

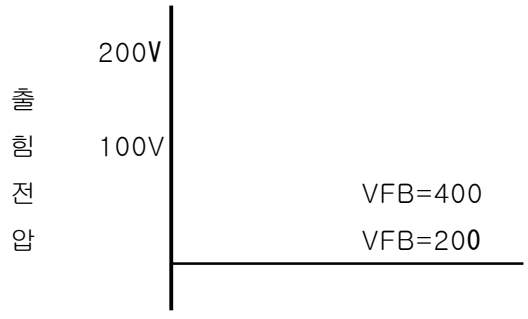



그림 2-2 VFB와 주파수

**(※설정치에 대해서는, 각 OS의 파라미터표의 설정 범위를 참조해 주세요.)**

(3) 가속도 설정(SFT) [중간 코드 : E8]

모터를 시동, 정지시킬 때의 가감 속도도의 설정을 실시합니다. 일반적인 소프트의 스타트를 의미하고 있습니다.

가감 속도도의 설정

$$\text{가감 속도도의 설정} = \frac{\text{가감 속도를 요구하는 주파수} \times 20}{\text{가감 속도 시간(초)}}$$

SFT=6000시약 0. 2초에 60 Hz

SFT=1200시약 1초에 60 Hz

SFT=120시약 10초에 60 Hz

(으)로 상승하게 됩니다.

(4) 주파수 설정(HZP) [중간 코드 : E1]

모터를 동작시키고 싶은 주파수의 설정을 실시합니다.

실제의 주파수 Hz(motor)는 차식에서 주어집니다.

Hz(motor)=1/32 ×(HZP)(Hz)

(HZP)＞0이라면 정회전(HZP)＜0이라면 역전이 됩니다.

(HZP)=0이라면 정지합니다.

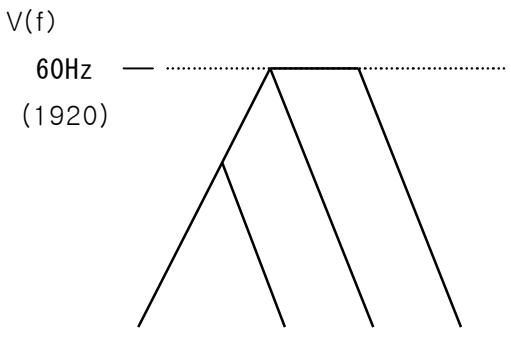
(5) 목표 위치 설정(POS) [중간 코드 : E3]

위치 결정 제어를 실시하고 싶을 때의 목표치를 설정하는 명령입니다. 엔코더 카운트치(시스템 변수 : PLS)가 POS의 내용과 동일하게 될 때까지 모터는 회전해 일치하면(자) 정지합니다.

(6) 최대 주파수 설정(MAXHZ) [중간 코드 : E4]

위치 결정의 제어에 대해 HZP를 설정하지 않고 최대 주파수만을 설정해 제어합니다.

예를 들면 MAXHZ=1920으로 해두면(자) 최대 60 Hz의 주파수를 이용해 위치 결정을 합니다. 아래와 같이(그림 2－3)에 참고도를 나타냅니다.



p1 p2 p3 t

그림 2-3 MAXHZ와 POS

(7) 최저시 저속 PSG(MINHZ) [중간 코드 : E5]

위치 결정의 제어에 대해 저속 제어시의 기울기를 설정할 수 있습니다. 이 수치가 클 수록 위치 결정 시간은 빨라집니다만, 위치 결정 정밀도는 나빠져 버리기 때문에 주의해 주세요.

설정치는 SQR(SFT/10×엔코더 보정치)로 계산합니다.

(8) 파워 컨트롤 명령(SEVCC) [중간 코드 : EF]

인버터에 공급하는 전원 ON/OFF를 위한 명령입니다.

SEVCC=0(파워를 OFF로 합니다.)

모터는 프리 런 상태가 됩니다.

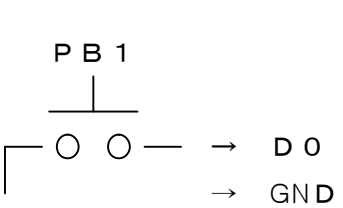
SEVCC=1(파워를 ON로 합니다.)

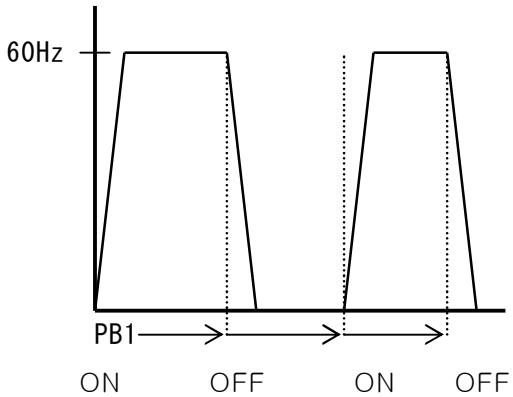
모터에 통전 됩니다.

[예 1] 입력을 사용한 모터 운전

입력 D0가 ON 되고 있는 동안, 주파수를 60 Hz로 설정해, 모터를 회전시킵니다.

```
QMCL명령어프로그램   행 NO.  QMCL 기계어
    CALL $460        000   F7CF0460FF  ; 파라미터 모드 설정
    VFB=1000         001   E7D01000FF  ; 저속 토르크 정수 설정
    SFT=2000         002   E8D02000FF  ; 가감 속도도 설정
    SEVCC=1          003   EFD001FF    ; 모터 통전
L00 JNE L01 C4 AND 1     004   F507C4D701FF    ; 입력 D0(ON? )
    HZP=0            005   E1D000FF    ; 모터(정지)
    JMP L00          006   F104FF      ; 입력 read에
L01 HZP=1920         007   E1D01920FF  ; 모터(60 Hz로 회전)
    JMP L00          008   F104FF      ; 입력 read에
```

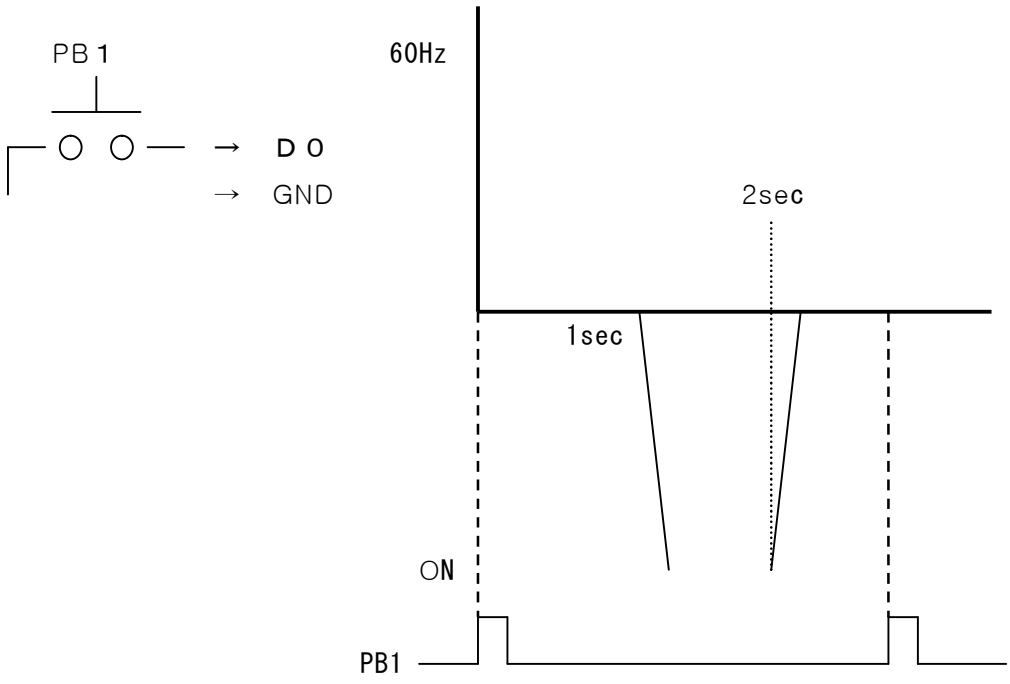


PB1를 ON/OFF 할 때마다 윗 그림과 같은 동작을 반복하기 위한 프로그램입니다.

[예 2] 타이머정불운전

```
QMCL명령어프로그램   행 NO.  QMCL 기계어
    CALL $460        000    F7CF0460FF
    VFB=1000         001    E7D01000FF
    SFT=6000         002    E8D06000FF
    SEVCC=1          003    EFD001FF
L00 JEQ L00 C4 AND 1 004    F304C4D701FF    ; 스타트 대기
L01 HZP=960          005    E1D00960FF  ; 모터 30 Hz정회전
    TIC1=410         006    EAD00410FF  ; 타이머 1초에 세트
L02 JNE L02 TIC1     007    F507EAFF    ; 타이머 업 대기
    HZP=-960         008    E1D0D20960FF    ; 모터 30 Hz역전
    TIC1=410         009    EAD00410FF  ; 타이머 1초에 세트
L03 JNE L03 TIC1     010    F510EAFF    ; 타이머 업 대기
    JMP L00          011    F104FF      ; 프로그램 반복
```




PB1를 1번 ON 하면(자) 1초간 정회전해, 그 후 1초간 역전해 재차 PB1가 ON인가 아닌가를 보러 갑니다.

(9) 위치 결정 게인 설정(PSG) [중간 코드 : E9]

최대 주파수를 사용한 위치 결정때, 감속 곡선(직선)을 지정할 수가 있습니다. 구동되는 메카니즘(기계)의 관성 모멘트 혹은 마찰력에 응한 적절한 값을 설정할 수가 있습니다. 다음 페이지(그림 2-4)에 PSG의 값과 감속 직선의 개

략을 나타냅니다. 메카니즘의 움직임에 오버 슛이 발생하는 것 같은 경우에는 PSG를 작게 해 사용해 주세요.

위치 결정 제어를 실시할 때에는, PSG에 있는 값(영 이외의)을 설정합니다.
또, PSG 설정이 위치 결정 제어의 스타트 명령도 겸하고 있습니다.
이 PSG와 SFT에는 다음의 관계가 있습니다.

설정치=SQR(SFT×엔코더 보정치)

예를 들면

SFT=6000, 엔코더 보정치=800 때

(모터 : 4 pole, 엔코더 : 2500 펄스)

PSG= 6000×800 =2１９０

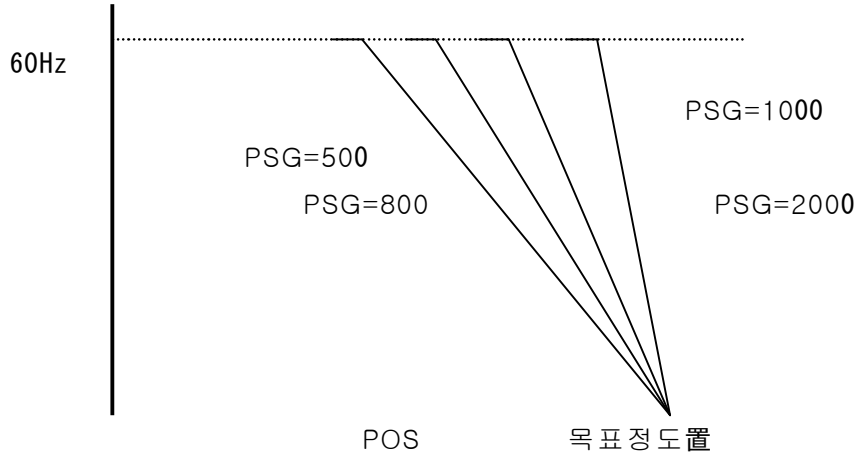



그림 2-4 POS와 주파수

POS=PLS가 되면(자) PSG=0(위치 결정 완료)이 되어 자동적으로 모터는 정지해, [HZP=0] (정지하고 있지만 부하의 반력에는 참고 정지 계속한다)을 유지합니다. 서보 락 상태로 보관 유지하는 경우는, 위치 결정 완료 후, 한번 더, PSG를 루프 시킵니다. (다음 페이지에서 사용예를 나타냅니다.)

사용예

```
        POS=2000
        PSG=1000
          :
L00     JNE  L00  PSG
```

PSG를 1000으로 주어, PSG가 0이 되는 것을 대기 0이 되면 위치 결정 완료라고 판단해, QMCL에서는 PSG=0으로서 L00의 다음의 명령의 실행에 들어갑니다. PSG가 클 수록, 감속 소요 시간은 짧아집니다.

서보 락 상태의 사용예(PSG를 루프 시킨다)

```
L01  PSG=5
       JEQ  L01  C4  AND  1
       PSG=0
       HZP=0
         :
```

2-8. 계시 명령

QMCL에는 모터 혹은 메카니즘의 움직임을 제어할 때에 필요한 계시의 카운트를 위한 타이머(시간 대기를 실시한다) 명령이 2케 준비되어 있습니다.

(1) 대기 시간(타이머) 설정(TIC1, TIC2) [중간 코드 : EA, EB]

TIC1는 타이머 1용, TIC2는 타이머 2용의 시간 설정 명령입니다.

TICn=xxx (n=1, 2)

(xxx=1, 2…65535)

xxx의 1은 2. 44 msec에 대응하고 있습니다.

TIC1=100은 타이머 1의 대기 시간을 0. 244초로 하는 것을 의미하고 있습니다.

사용예

```
       TIC2=100
L00  JNE  L00  TIC2
       HZP=240
         :
```

프로그램 위의 부분을 실행해 온 QMCL는 여기서 타이머 2를 0. 244초로 설정합니다. 이 때 모터의 움직임은 전 상태를 계속하고 있습니다. L00는 0. 244초의 대기 시간의 실행입니다. 0. 244초 지나면(자) QMCL는 다음의 명령 HZP=240(30 HZ주파수 설정)을 실행해, 이하의 프로그램의 실행으로 옮겨 갑니다.

2-9. 점프 명령

프로그램의 분기 명령을 해 이하의 것이 준비되어 있습니다.

(1) 써브루틴 점프(JSR) [중간 코드 : F0]

메인 프로그램으로부터 라벨 혹은 행 번호로 지정된 써브루틴에 점프 하는 경우 이용합니다.

사용예 JSR 130 (F0130)

QMCL 커멘드행의 130행째부터 쓰여져 있는 써브루틴에 점프 합니다.

(2) 무조건 점프(JMP) [중간 코드 : F1]

무조건 지정된 라벨 혹은 행 번호에 점프 하는 경우에 이용합니다.

사용예 JMP 50 (F150)

(3) 조건 점프

조건 점프로서 다음의 4개의 것이 준비되어 있습니다

| 무릎 관절 모닛크 | 중간 코드 | 의미 | 설명 |
|---|---|---|---|
| JMI | F2 | Jump MInus | 조건문이 부＜0때 점프 |
| JEQ | F3 | Jump Equal ZERO | 조건문이 영=0때 점프 |
| JPL | F4 | Jump PLus | 조건문이 정≥0때 점프 |
| JNE | F5 | Jump Not Equel ZERO | 조건문이≠0때 점프 |

사용법 JEQ L00 (조건문)

조건문으로서는 변수(유저 변수 및 시스템 변수) 혹은 직접 연산식을 쓸 수가 있습니다.

사용예 JEQ 50 A0－A1 (F350A0D2A1)

유저 변수 A0와 A1의 차이가 0이 되었을 때 50행째에 점프 합니다. 0이 아닐 때는, 바로 아래의 명령이 실행됩니다.

(4) 상대 점프 명령(BRA) [중간 코드 : F6]

무조건 점프의 일종입니다만, 지정된 변수의 내용에 따라 점프처를 바꿀 수가 있습니다.

사용법 BRA (유저 변수)

사용예 (중간 언어)

```
BRA   B0   (F6B0)
JMP   L00  (F1nn)   ※(nn, mm, ee)
JMP   L01  (F1mm)에는, 점프처의 행번
JMP   L02  (F2ee) 호가 들어갑니다.
```

유저 변수 B0에 들어가 있는 수치(0, 1혹은 2)에 따라 0의 경우 L00, 1의 경우 L01, 2의 경우 L02에 점프 합니다.

(5) 실시간 시계 이용 명령(약칭 : ONTIM 명령)

QMCL는 65 msec 단위와(0. 244msec×n) 단위의 계시를 실시하고 있는 시계를 각 1케 가지고 있습니다.

(n : ONTIM2 제어 시간($F01C) 설정)

65 msec마다 혹은(0. 244msec×n) 마다 실시하고 싶은 처리를 실행시키기

위해서(때문에) 이용할 수가 있습니다.

시계 1 이용 선언  ONTIM1(라벨명 혹은 행 번호) [중간 코드 : F8]
시계 2 이용 선언  ONTIM2(라벨명 혹은 행 번호) [중간 코드 : F9]

사용법

```
          ONTIM1  100      (F80100)
  L00     A0=A0+A1
```

65 msec마다행 번호 100보다 시작되는 써브루틴에 점프 해, 그 처리가 종료하는 곳의 루틴에 돌아와 즉시 L00 이하의 명령을 실행해 갈 것입니다.

```
  ONTIM2  50      (F950)
  L01  B0=A0×B1
```

(0. 244msec×n) 마다행 번호 50보다 시작되는 써브루틴 1에 점프 해 그 처리가 종료하는 곳의 루틴에 돌아와 즉시 L01로부터의 명령을 실행해 갈 것입니다. 또, ONTIM1보다 우선해 실행합니다.
ONTIM 명령을 여러 차례 계속해 사용했을 경우에는 마지막 것이 유효가 됩니다.
시계의 이용 종료 선언은, OFTIM 명령에 의해 행해집니다.

사용예  OFTIM1 [중간 코드 ; F800] : 시계 1 미사용

(6) 특수 점프(CALL) [중간 코드 : F7]
기계어로 쓰여진 써브루틴에 점프 할 때에 이용합니다. QMCL에는 실행 시간의 고속화를 목적으로 한 기계어의 써브루틴을 몇개인가 가지고 있습니다.
또, 기계어로 작성한 써브루틴을 실행하는 일도 할 수 있습니다.
이것들을 사용하고 싶을 때에 이용하는 명령입니다.

사용예  CALL  $420      (F7CF0420)
디스플레이의 표시를 전소등합니다.

2-10. 귀가 명령

(1) 귀가 명령(RTS) [중간 코드 : FA]
주루틴으로부터 점프 해 온 해당 써브루틴의 종료 선언과 원래의 주루틴의 귀가를 지령하는 명령입니다. ONTIM 명령, JSR 명령으로 실행하는 써브루틴의 종료행에 필요합니다.

(2) 귀가 금지 명령(OFFRTS) [중간 코드 : FB]
써브루틴에서의 처리의 결과에 따라서는 앞의 주루틴에 돌아올 필요가 없는 경우가 생깁니다. 이러한 때의 써브루틴·프로그램의 종료 선언 명령으로

서 이용하면(자) 편리합니다.

(3) 귀가처 테이블 클리어 명령(AOFRTS) [중간 코드 : FC]

프로그램의 실행에 의해 차례차례 자동적으로 작성되고 있는 써브루틴으로부터의 귀가처를 기억한 테이블을 모두 크리야(소거) 하는 명령입니다.

어떠한 이유로써 프로그램이 정상적으로 동작하고 있지 않는 것이 검출되었을 때 등에 본명령을 이용해, 정상적인 움직임에 되돌릴 수가 있습니다.

2－11. 실행 정지 명령(STOP) [중간 코드 : FF]

모터를 정지시키고 파워부에의 전원을 차단해(모터는 프리 런) QMCL를 에디터 모드에 세트 하는 명령입니다.

※QMCL로 클리어 한다고는 메모리의 내용을" F" 라고 하는 문자로 한다고 하는 것으로 동의입니다. 1행의 명령이 FFF…가 되고 있는 것은 아무것도 프로그램이 쓰여지지 않은 것을 의미하고 있습니다. 이것은 즉" STOP" 명령이라고 하게도 됩니다.

# 3　프로그래밍상의 주의

## 3-1. 연산 명령의 실행 순서

사칙 및 논리 연산과도 좌측 우선으로 실행됩니다. 따라서(1) 식의 의미는 통상의 수학에서는(2) 식과 같은 연산이 됩니다.

A0=10＋A1×5·····(1)

A0=(10＋A1)×5···(2)

## 3-2. 음수의 표현

계산기의 내부에서의 음수 표현은 1번 왼쪽의 비트(MSB)가 1이 되고 있습니다.

메모리의 내용을 볼 때에는 주의해 주세요. 이하에 약간의 예를 나타냅니다.

메모리 내용 (HEX)

A0=1···0001

| 내용 형식 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

A0=2···0002

| 내용 형식 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

A0=63···003F

| 내용 형식 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

A0=－1···FFFF

| 내용 형식 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

A0=－7···FFF9

| 내용 형식 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

A0=32767···7FFF

| 내용 형식 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

A0=32768(－32768)···8000

| 내용 형식 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

3－3. 소수점 연산

QMCL에서는 소수점 연산이 용서되고 있지 않습니다. 정수 연산만 이용할 수 있습니다. 소수 연산을 실시하고 싶은 경우의 참조예를 아래와 같이 합니다.

＜예＞A1=A0/6을 실시해, 그 결과를 소수점 이하 1자리수의 수치까지 표시한다.

프로그램예

```
A1=A0×10/6                ; 원의 수를 10배로 해, 6으로 제산합니다.
CA40=A1                   ; 0~4의 부분에 A1의 내용을 표시합니다.
                            (2－3. 표시 참조)
POKE  $F0B1  $80          ; 1의 부분( 오른쪽으로부터 2번째 )에
```

닷 포인트($80)를 점등.

**※7 SEG(디스플레이)의 닷 번지 : $F0B0~$F0B9**

3－4. 제산

QMCL에서의 제산은 정의 수끼리의 연산 밖에 용서되고 있지 않습니다. 부의 수의 연산을 실시하는 경우에는 먼저 주어진 수의 절대치를 요구한 후, 연산을 실시해 주세요.

사용예 A0=(－9)/3

프로그램예

```
        A1=－9
        A2=ABS  A1  /3
        JPL  L00  A2
        A0=－A2
L00       :
```

3－5. 곱셈

QMCL에서는 2바이트의 정수(4바이트 특수 변수는 제외하다) 밖에 취급하고 있지 않으므로 2개의 곱셈 결과가 32767을 넘을 때는 부의 수가 되어 버리기 때문에 주의해 주세요.(3－3. 음수의 표현 참조)

3－6. PLS의 초기치

시스템 변수 PLS에는 엔코더로부터의 펄스수의 카운트치가 항상 보존되고 있습니다.

원점 위치 설정으로서 PLS=0으로 하면(자) 원점 근방에서 그 내용은 1으로 65535(=－1)를 교대로 취하거나 해 수치의 연결과 움직임의 실감이 잡기 어려워집니다. 이러한 일을 피하고 싶은 경우, 초기치로서 예를 들어 PLS=1000으로 해두면(자) 편리합니다.

원점에 되돌리는 것은 1000에의 설정이 되어 999, 1000, 1001이라고 하는 값을 받게 되어, 그 연결이 용이하게 잡을 수 있습니다. 다만 표시하는 경우는, 1000을 감산하는 것을 추천합니다.

프로그램예

```
        PLS=1000
        POS=2000
        PSG=10
L00     CA40=PLS－1000
                    (1000을 감산해 PLS를 디스프레 4~0에 표시)
JNE     L00  PSG
          :
```

3－7. 프로그램 작성

(1) 10 진수의 내부 표현

10 진수는 내부에서는 2 문자 단위로 표현되고 있습니다.

사용예 HZP=960의 내부 표현은

(E1 D0 09 60)과 같이 됩니다.

직접 키보다 입력할 때는 09 60 (으)로서 주세요.

(2) 1행의 명령은 16 문자 이내

QMCL의 커멘드행은 16 문자 이내가 되고 있습니다. 긴 명령문을 작성하고 싶은 경우에는 2개로 나누어 작성해 주세요.

사용예 A0=BA＋1000＋A2＋A1 이 명령문은

(A0 D0 BA D1 10 00 D1 A2 D1 A1)의 20 문자가 되어 버립니다. 이러한 경우는 다음과 같이 명령을 나누어 기술해 주세요.

```
A0=BA＋1000
A0=A0＋A2＋A1
```

(3) 조건 점프문의 연산식의 처음은 문자로 JNE, JMI등의 조건 점프문의 조건부분에 연산식을 이용하고 싶은 경우, 그 선두에는 반드시 문자 변수를 사용해 주세요.

사용예 JMI 20 10－A0…오

JMI 20 A0－10…정

(4) 점프문의 나는 일처 지정

점프문의 날아 먼저 연산식을 이용할 수 없습니다. 반드시 지정의 행 번호 혹은 라벨을 사용해 주세요.

JMP 20＋A0…오
JMP 100·····정
JMP L01·····정

(5) NOP 커멘드

1개의 프로그램이 있는 행을 무시하고 싶은 경우에 NOP [노 오퍼레이션의 의미, 중간 코드 D0(=와 동일)] 를 사용합니다.

지금 있는 행의 명령으로서 A0=A1＋2
(중간 언어 표현 : A0 D0 A1 D1 02)가 있었다고 합니다.
상기 명령문의 선두에 D0를 추가합니다.
(중간 언어 표현 : D0 A0 D0 A1 D1 02)
D0를 추가한 것에 의해, 상기 명령문은 무시됩니다. D0를 삭제하면(자) 재차 실행됩니다.
디버그의 도중등으로 NOP를 사용하면(자) 효율 좋게 버그해를 진행시킬 수가 있습니다.

(6) ONTIM1, 2의 시간 제약

ONTIM1, 2에 의해 기동되어 실행되는 써브루틴은, 각각 65 msec, (0. 244msec×n) 이내에 처리가 종료되지 않으면 안됩니다.

QMCL의 커멘드행 1행은 약 0. 1 msec로 처리된다고 해, 써브루틴의 총처리 시간을 산출해, 그 값이 각각의 지적 시간내에 들어가 있는 것을 확인해 주세요.

**※TIC1, 2 명령을 ONTIM1, 2 중(안)에서 사용하는 것은 금지되고 있습니다.**

(7) 프로그램 입력

QMCL의 중간 언어로 직접 MITY 서보에 커멘드를 입력해 프로그램을 작성하는 경우, 1행의 커멘드 입력에 계속해“ FF”를 삽입해 주세요.

사용예 HZP=32→E1 D0 32 FF
A4=A0*2 → A4 D0 A0 D3 02 FF

**※직접 프로그램 입력하고 말이야 있고 하행의 마지막의 자리수에는,“ FF”를 기입해 주세요.**

# 4 명령 실행의 원리

## 4-1. 프로그램 구성

MITY 서보의 프로그램은 행 단위로 구성됩니다.

| QMCL명령어프로그램 | 행 NO. | QMCL 기계어 |
|---|---|---|
| CALL $460 | 000 | F7CF0460FF |
| VFB=1000 | 001 | E7D01000FF |
| SFT=2000 | 002 | E8D02000FF |
| SEVCC=1 | 003 | EFD001FF |
| L00 JNE L01 C4 AND 1 | 004 | F507C4D701FF |
| HZP=0 | 005 | E1D000FF |
| JMP L00 | 006 | F104FF |
| L01 HZP=1920 | 007 | E1D01920FF |
| JMP L00 | 008 | F104FF |

L00, L01는, 점프처의 기준을 위한 라벨로, 커멘드가 아닙니다. QMCL 중간 언어로 FF란, 행선지두에 있을 때는 STOP 커멘드이며, 그 이외때는 행 엔드입니다. 이후 설명에서는 FF는 생략합니다.

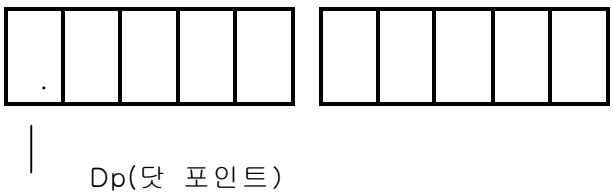
명령어의 나타내는 방법은, 영숫자의 편성 , 1 커멘드 2 문자로 구성되어 1행 최대 8 커멘드 16 문자로 구성되게 됩니다. 따라서, 1 명령은 1행에 납입할 필요가 있습니다. 또, 동일 은행내에 2 명령의 프로그램은 할 수 없습니다.

프로그램의 실행 순서는, 행 No. 의 젊은 순서로부터 실행됩니다.

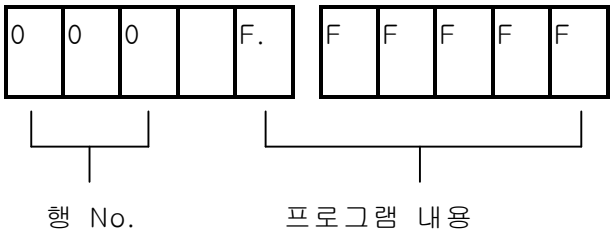
## 4-2. 에디터 및 프로그램에 대해

에디터와는 수정 편집이라고 하는 의미로, 즉 프로그램의 수정 편집에 필요한 것입니다. 플래쉬 메모리의 프로그램을 선택하는 것처럼 합선 핀이 세트되어 있지 않을 때에 전원을 투입하면(자) 에디터가 되어, 디스플레이의 좌단에 Dp(닷 포인트)가 점등 해 커멘드 대기가 됩니다.

에디터에 있어서 프로그램을 기입할 수 있는 상태가 된 일을 프로그램 모드라고 해, 작성한 프로그램을 MITY 서보에 입력할 수가 있습니다. 에디터때의 표시는 아래와 같이 됩니다. 또, 프로그램 스톱시의 에디터는 디스플레이왼쪽으로 스톱 한 행수를 표시합니다.

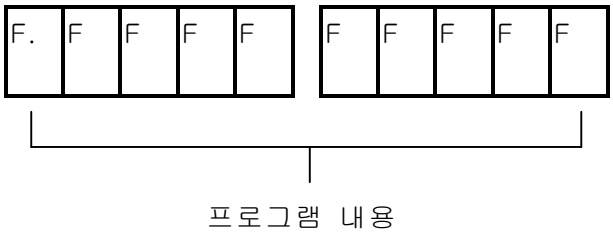


프로그램 모드( F CR)가 되었을 때의 표시를 아래와 같이에 나타냅니다.




**0행 눈앞반을 표시**

여기서 → 키를 6회 누른다고 표시가 아래와 같은님이든지, 그 행의 후반의 프로그램을 표시합니다.




**0행째의 후반을 표시**

프로그램 모드에서는 Dp(닷 포인트)의 표시하고 있는 자리수가 입력중의 자리수로 커서가 됩니다. 이후 커서라고 부릅니다.

# 5 키 조작

## 5-1. 키 조작

① [C L R], [L.CLR]

키 조작의 캔슬을 할 수 있어 CLR는 1 캐릭터의 캔슬, L. CLR는 모든 캔슬이 됩니다.

② [N1], [N2]

프로그램 에리어의 클리어가 생깁니다.

(아래와 같은 X, Y는 행수 및 수치를 입력합니다. )

[N1] [X] [CR] 그리고 X행부터 프로그램이 클리어 됩니다.

[N2] [X] [CR] 그리고 0행부터 X행까지 프로그램이 클리어 됩니다.

[N1] [X] [N2] [Y] [CR] 그리고 X행부터 Y행까지 클리어가 됩니다.

③ [JOB] [X] [CR]

X행부터 프로그램을 실행합니다.

④ [F] [X] [CR]

X행부터 프로그램 모드가 됩니다.

⑤ [M O NI TOR] [CR]

기계어의 조작을 할 수 있는 모드가 됩니다. 이 모드로부터 에디터에 돌아올 때는 END 키를 눌러 주세요.

⑥ [A] [X] [CR]

메모리의 내용을 표시합니다.

A 0 CR 그렇다면, A0의 변수의 내용을 10 진수로 표시합니다.

A0~AF, B0~BF, C0~C7가 가능합니다.

⑦ [A] [X] [Y] [CR]

메모리의 내용을 10 진수로 써 바꿀 수가 있습니다.

A 2 3 0 CR 그렇다면, A2의 변수의 내용이 30이 됩니다.

A0~AF, B0~BF, C0~C1가 가능합니다.

CLR Dp의 점등 하고 있는 1 문자가 삭제되어 그 이후의 문자가 융통 오릅니다.

L.CLR 1행 모두, ”F”가 됩니다.(1행 클리어)

1CHR INS Dp의 점등 하고 있는 곳이, ”F”가 되어 자리수가 융통 오릅니다.

INS 표시하고 있는 행도 포함해 그 이후의 프로그램이 1행 올라, 새로운 행이 추가됩니다.

LINE DEL 표시행이 삭제되어 그 이후의 행이 융통 올라, 표시행에는 다음 행의 프로그램이 표시됩니다.

END 에디터에 돌아옵니다.

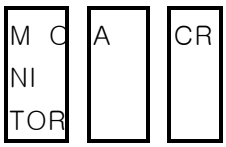
CR 표시하고 있는 행의 프로그램을 기억해, 다음 행을 표시합니다. 이 키를 누르지 않으면 프로그램은 기억되지 않습니다.

↑ ↓ 표시하고 있는 행의 진행되어, 반환을 합니다.

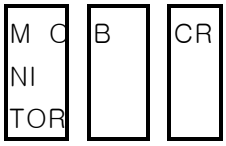
← → 은행내에서의 커서의 이동을 합니다.

⑨ MONITOR 1 CR QMCL 파라미터 모드가 됩니다. 이 모드로부터 에디터 모드에 돌아오려면 END 키를 눌러 주세요.

⑩ MONITOR 2 CR 유저 모드가 됩니다. 이 모드로부터 에디타모드에 돌아오려면 END 키를 눌러 주세요.

⑪ 

QMCL 파라미터 설정치를 초기화합니다.

⑫ 

유저 파라미터의 설정치를 0으로 합니다.

# 6 프로그램의 기입 방법

타이머에 치수동 운전하는 프로그램을 만들어, MITY 서보에 기입해 움직여 보겠습니다.

## 6-1. 프로그램예

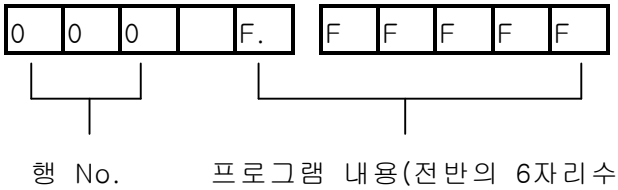
```
        CALL   $460
        VFB=1000
        SFT=6000
        SEVCC=1
L00     HZP=960                 ; 30 HZ지령
        TIC1=410                ; 1초 타이머
L01     JNE  L01  TIC1
        HZP=0                   ; 정지
        TIC1=205                ; 0. 5초 타이머
        L02  JNE  L02  TIC1
        JMP  L00
```

## 6-2. 중간 언어로 번역 (커멘드표로 번역, 또는, 컴파일러 소프트로 자동번역)

| 행 No. | QMCL 중간 언어 | | | | |
|---|---|---|---|---|---|
| 000 | F7 | CF | 04 | 60 | FF |
| 001 | E7 | D0 | 10 | 00 | FF |
| 002 | E8 | D0 | 60 | 00 | FF |
| 003 | EF | D0 | 01 | FF | |
| 004 | E1 | D0 | 09 | 60 | FF |
| 005 | EA | D0 | 04 | 10 | FF |
| 006 | F5 | 06 | EA | FF | |
| 007 | E1 | D0 | 00 | FF | |
| 008 | EA | D0 | 02 | 05 | |
| 009 | F5 | 09 | EA | | |
| 010 | F1 | 04 | | | |

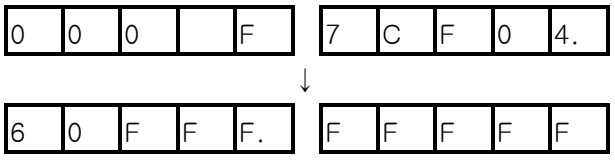
6－3. 프로그램 입력

(1) MITY 서보의 전원을 ON 합니다. 에디터가 됩니다. 우선, 나카에 들어가 있는 프로그램을 일단, 클리어 합니다. N1 CR 그리고 클리어 됩니다. 프로그램 모드로 합니다. F 0 CR 그리고, 0행째부터 프로그램 모드가 됩니다.
(0행째만 0 (을)를 생략 해, F CR 그렇지만 가능)



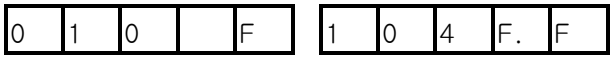

(2) 2. 그리고 중간 언어로 번역한 프로그램을 키보드보다 입력해 갈 것입니다.

0행째 F 7 C F 0 4 6 0 (이)라고 입력한다.




(3) 0행째를 입력하면(자), CR 키를 눌러 주세요. 프로그램은 0행째로서 기억되어 표시는 다음의 행에 진행됩니다.


(4) 이후, 10행째까지와 같이 입력해 갈 것입니다.




(5) 10행째의 입력이 종료하면(자) 잊지 않고 CR 키를 누릅니다.


(6) 이상으로 프로그램의 입력은 끝났습니다. 여기서, END 키를 눌러 에디터에 되돌립니다.

6-4. 프로그램 실행

입력된 프로그램을 실행합니다. JOB CR 키를 눌러 주세요.

0행째부터 프로그램은 실행되어 운전합니다. 이 프로그램은 정지하는 프로그램이 아니기 때문에 정지하는 것은 전원 OFF로 합니다.

또, 100행째부터의 프로그램의 경우

```
F 1 0 0 CR          ; 프로그램 100행째를 지정
JOB 1 0 0 CR        ; 100행째부터 프로그램을 실행
```

# 7 디스플레이·키 배치도

## 7-1. 7 SEG 배치

| 9 | 8 | 7 | 6 | 5 |
|---|---|---|---|---|

| 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|

## 7-2. KEY 코드 배치

(1) 내부 KEY

| 7 | 8 | 9 | 13 |
|---|---|---|---|
| 4 | 5 | 6 | 12 |
| 1 | 2 | 3 | 11 |
| 0 | 16 | 17 | 10 |

| 19 | 23 | 27 | 31 |
|---|---|---|---|
| 18 | 22 | 26 | 30 |
| 15 | 21 | 25 | 29 |
| 14 | 20 | 24 | 28 |

(2) 에디터 KEY

| 7 | 8 | 9 | D |
|---|---|---|---|
| 4 | 5 | 6 | C |
| 1 | 2 | 3 | B |
| 0 | STOR<br>L.CLR | LOAD<br>CLR | A |

| INC<br>N1 | 1CHR<br>INS | END | JOB |
|---|---|---|---|
| DEC<br>N2 | RUN<br>INS | LINE<br>DEL | MONI<br>TOR |
| F | DATA<br>← | ↓ | OPT<br>ION |
| E | ADR<br>→ | ↑ | CR |

초판 1999.09. 07

제2판 2000.02. 28