“Basics of microcontroller”

Lecture 12.

Organize and address the interface and input-output interface

Lecturer: Onon Otgonbaatar, MD.

МАКРО

Макро гэдэг нь хэсэг командуудыг агуулсан жижиг программ юм. Макрог хэд хэдэн эсвэл хэдэн зуун мөр командуудыг дахин дахин давтахаас зайлсхийн, нэг команд бичин энэ багц командуудаа ашиглахад хэрэглэнэ. Бичигдсэн команд бүрт Ассемблер нь нэг машины кодыг үүсгэнэ. Харин Pascal, C гэх мэт хөрвүүлэгчид нь команд тус бүрт нэг болон түүнээс дээш машины кодуудыг үүсгэдэг. Энэ талаас нь үзвэл эдгээр өндөр түвшний хэлүүд нь макро тодорхойлолтуудаас бүтсэн гэж хэлж болно. Эдгээртэй адил Ассемблерт “macro” гэсэн хэллэгийг ашиглан хөрвүүлэлтийг хялбарчилах боломжтой болно. MASM гэх мэт том Ассемблерийн программууд нь макрогийн хэрэглээг хангадаг боловч программист өөрөө макрог тодорхойлж өгч бас болно.

Доорх маягаар макрог үүсгэнэ.

```
MACRO
….
MEND
```

Макрог ашигласнаар дараах давуу талууд байна.

- Кодын хэмжээг багасгах, хялбарчлана.
- Уншихад илүү ойлгомжтой болно.
- Олон дахин кодууд давтагдсанаас үүсэх алдааг багасгана.

Макрог регистрүүдэд анхны утга олгох, тасалдлуудыг гүйцэтгэх, ASCII ба 16-тын өгөгдлүүдийг хөрвүүлэх, олон байт урттай арифметик үйлдлүүдийг гүйцэтгэх, тэмдэгтүүдтэй ажиллах гэх мэт оролт гаралтын үйлдлүүдэд хэрэглэж болно.

Жишээ болгон, сегмент регистрүүдэд анхны утга олгох программын эхлэлийн хэсгийг INIT1 нэртэй макро тодорхойлолт бичье.

```
TITLE MACRO1      Энгийн макро
.model small
;-------INIT1 гэсэн бичлэг программ дотор тааралдвал энэ
;-------  макрогийн үйлдлүүд хийгдэнэ.
INIT1       MACRO
            MOV  AX, @DATA
            MOV  DS, AX
            MOV  ES, AX
            ENDM
.stack  100H
.data
      MESSGE     DB     ‘Test of macro instruction’, 13
.code
      BEGIN      PROC FAR
            INIT1                        ; макро команд
            MOV        AH, 40H           ; дэлгэцэнд гаралт
            MOV        BX, 01            ; хүснэ.
            MOV        CX, 26            ; мэдээний урт
            LEA        DX, MESSGE        ; мэдээ
            INT        21H
```

```
        RET
    END BEGIN
```

Эндээс бид дэлгэцэнд мэдээлэл гаргах хэсгийг доорх байдлаар нэг макро тодорхойлолт болгон хялбарчилбал дэлгэцэнд мэдээлэл гаргах бүрт ашиглан, бичлэгийг хялбарчлах боломжтой болно. Энэ нь dummy гишүүнтэй макро тодорхойлолт болно.

```
PROMPT      MACRO       MESSGE
        ;-----дэлгэцэнд мэдээлэл гаргах макро
            MOV         AH, 09H             ; дэлгэцэнд гаралт
            LEA         DX, MESSGE          ; мэдээ
            INT         21H
            ENDM
```

Одоо манай үндсэн программ бүр хялбар болно.

```
TITLE MACRO1      макро агуулсан жишээ
.model small
;-----------------------------------
INIT2       MACRO
            MOV   AX, @DATA
            MOV   DS, AX
            MOV   ES, AX
            ENDM
;-----------------------------------
PROMPT      MACRO       MESSGE
        ;-----дэлгэцэнд мэдээлэл гаргах макро
        ;; Generates code that links to DOS
            MOV         AH, 09H             ; дэлгэцэнд гаралт
            LEA         DX, MESSGE          ; мэдээ
            INT         21H
            ENDM
;-----------------------------------
.stack  100H
.data
        MESSG1      DB      'Student name? ', '$'
        MESSG2      DB      'Student code? ', '$'
.code
        BEGIN       PROC FAR
            .SALL
            INIT2               ; макро команд
            PROMPT      MESSG1
            .LALL
            PROMPT      MESSG2
            RET
        BEGIN       ENDP
            END  BEGIN
```

Comment буюу тайлбаруудыг зохион байгуулахдаа доорх директивүүдийг ашиглана.
.LALL – List all
2 ширхэг ;; тэмдэгийн ард бичнэ.
.XALL – Зөвхөн командуудыг бичих ба обьект кодуудыг үүсгэнэ.
.SALL – Suppress all
Программын листийг бага болгохын тулд макрог дахин дахин бичихгүй, гэхдээ обьект кодуудын хэмжээнд нөлөөлөхгүй.

DOS21 нэртэй макро нь DOS-ын тасалдлын функцын дугаарын AH регистрт олгоод, DOS INT 21H тасалдлыг дууддаг байхаар хийе.

```
DOS21   MACRO   CONT_AH
        MOV     AH, CONT_AH
        INT     21H
        ENDM
```

Энэ макрог программ дотроос ашиглахдаа:

```
        LEA     DX, NAMEPAR
        DOS21   0AH
```

гэж бичвэл гарнаас өгөгдөл авахад макрог ашиглана.

Тэмдэгтийг дэлгэцэнд гаргах макрог бичвэл доор байдалтай болно.

```
DISP    MACRO   CHAR
        MOV     AH, 02
        MOV     DL, CHAR
        INT     21H
        ENDM
```

Макрог ашиглан өөр тодорхойлогдсон макрод хандаж болно. Жишээ нь:

```
DISP    MACRO   CHAR
        MOV     DL, CHAR
        DOS21   02
        ENDM
```

Дээрх дотроо макро ашигласан макро нь доорх үйлдэлтэй ижил болно.

```
        MOV     DL, '*'
        MOV     AH, 02
        INT     21H
```

LOCAL директив

LOCAL нь үүсгэсэн нэртэй хувьсагч нь уг программд цорын ганц байгаа гэдгийг заана. Өөрөөр хэлбэл, уг нэр программын өөр хэсэгт хэрэглэгдэхгүй. Жишээ нь:

```
TITLE MACRO3    LOCAL-ийн хэрэглээ
;--------------------
DIVIDE      MACRO   DIVIDEND, DIVISOR, QUOTIENT
        LOCAL   COMP
        LOCAL   OUT
        ;--- AX=DIVIDEND, BX=DIVISOR, CX=QUOTIENT
```

```
        MOV         AX, DIVIDEND
        MOV         BX, DIVISOR,
        SUB         CX, CX          ; коэффициентийг цэвэрлэнэ.
COMP:
        CMP         AX, BX
        JB          OUT
        SUB         AX, BX
        INC         CX
        JMP         COMP
OUT:
        MOV         QUOTIENT, CX    ; коэффициентийг хадгална.
        ENDM
;-----------------
.stack  100h
.code
        MAIN PROC NEAR
                .LALL
                DIVIDE      DVDND, DIVSOR, QUOTNT
                RET
        MAIN ENDP
                END   MAIN
```

САН (LIBRARY)-д макрог оруулах

Макрог тодорхойлоод зөвхөн программ дотроосоо хэрэглэх нь учир дутагдалтай юм. Илүү сайн арга нь ямар нэг нэр, жишээ нь MACRO.LIB нэртэй файлд бүх зохиосон макронуудаа оруулаад хэрэглэх юм. Ингэснээр бусад бүх ассемблер программууд LIB файл доторх макронуудыг ашиглах боломжтой болно.

```
INIT            MACRO
                …
                ENDM
;---------------------------------
PROMPT      MACRO       MESSGE
                …
                ENDM
```

гэх мэтээр бичсэн макронуудаа macro.lib нэртэй файлд хадгална. Эдгээр LIB файлд агуулагдсан макронуудыг хэрэглэхийн тулд программынхаа эхэнд INCLUDE директивийг хэрэглэнэ.

```
INCLUDE     C:MACRO.LIB
                …
                INIT
                …
```

Ер нь макрог программд ашиглах үед (list) LST программ нь мөрийнхөө өмнө + тэмдэгтэйгээр макрог программд байх ёстой газарт нь оруулж харуулна.

Үгүйсгэх директив: PURGE

IF1 ба PURGE гэсэн директивүүдийг макротай ажиллахдаа ашиглаж болно. Бид INIT, PROMPT, DIVIDE гэсэн нэртэй макронуудыг макро сан дотор агуулсан. Харин эдгээрээс зөвхөн INIT –ийг л программдаа ашиглана гэвэл доорх байдлаар бичиж болно.

```
IF1
        INCLUDE      MACRO.LIB  ; бүхэл санг агуулна.
ENDIF
PURGE          PROMPT, DIVIDE  ; хэрэггүй макронуудыг үгүйсгэнэ.
…
INIT                           ; үлдсэн макрог хэрэглэнэ.
```

PURGE-ийг хэрэглэхэд хэрэггүй макронуудыг программаас хасах боловч LIB файл дотроо хадгалагдсан хэвээрээ байна.

Давталтын директивүүд (Repetition): REPT, IRP, IRPC

Блок өгөгдлийг давтах боломжийг эдгээр директивүүд олгох ба ENDM директивээр дуусгагдана. Эдгээр директивүүд нь заавал макро дотор байх албагүй.

REPT (repeat) үйлдэл нь ENDM директив тааралдтал блок өгөгдлүүдийг заасан тоогоор давтана.

```
N=0
REPT  5
N=N+1
DB      N
ENDM
```

N утгыг 0-тэй тэнцүүлээд, DB N үйлдлийг 5 давтана. DB 1-ээс DB 5 хүртэл DB командуудыг 5 удаа хийнэ. Иймэрхүү байдлаар матрицад хэрэглэж болно.
Өөр нэг жишээ бичвэл, доорх кодуудаар 5 ширхэг MOVSB командуудыг үүсгэх ба энэ нь REP MOVSB үйлдлийг CH=5 үед хийсэнтэй ижил юм.

```
REPT 5
MOVSB
ENDM
```

IRP (indefinite repeat) үйлдэл нь ENDM хүртэлх блок командуудыг давтана. Формат:
IRP гишүүн, <аргумент>
Жишээ нь:

```
IRP N, <3, 9, 17, 25, 28>
DB N
ENDM
```

үйлдлүүдийн үр дүнд DB 3, DB 9, DB 17, DB 25, DB 28 гэсэн үйлдлүүд хийгдэнэ.

IRPC (indefinite repeat charachter) үйлдэл нь блок хэллэгүүдийг давтана. Формат нь:
IRPC гишүүн, тэмдэгт
Жишээ нь:
IRPC N, 345678
DW N
ENDM
үйлдлүүдийн үр дүнд DW 3, DW 4, DW 5, DW 6, DW 7, DW 8 гэсэн кодууд үүснэ.

Нөхцөлт директивүүд

Нөхцөлд директивүүд нь зөвхөн макрод хэрэглэгддэггүй боловч макрод их чухал үүрэг гүйцэтгэдэг.
IF директив бүрд ENDIF гэсэн шалгагдаж буй нөхцөлийг зогсоох директив байна. Эсрэг нөхцөлийг нь шалгах ELSE директив хэрэглэгдэнэ. Нөхцөлт блок нь доорх байдалтай байна.

IFxx (нөхцөл)
…
ELSE (туслах үйлдлүүд)
…
ENDIF (төгсгөл)

ENDIF директивийг бичээгүй бол “Undeterminated conditional” гэсэн алдааны мэдээлэл гарна. Хэрэв шалгаж буй нөхцөл (true) үнэн бол ассемблер нь IFxx –ээс ELSE хоорондох үйлдлүүдийг гүйцэтгээд, хойшхийг нь орхино. Нөхцөл (false) худал бол ELSE-ээс ENDIF хоорондох үйлдлүүдийг гүйцэтгээд, өмнөх хэсгийг орхино.
Нөхцөлт директивүүдийг тоочвол:

IF нөхцөл
Хариу нь 0 биш бол нөхцөлт блок доторх үйлдлүүдийг хийнэ.

IFE нөхцөл
Хариу нь 0 бол нөхцөлт блок доторх үйлдлүүдийг хийнэ.

IF1 (хоосон)
Ассемблер нь 1 гэсэн утга авч,нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IF2 (хоосон)
Ассемблер нь 1 гэсэн утга авч, нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IFDEF Symbol
Symbol буюу тэмдэгт нь программд тодорхойлогдсон эсвэл EXTRN директивээр агуулагдсан бол нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IFNDEF Symbol
Symbol буюу тэмдэгт нь программд тодорхойлогдсон эсвэл EXTRN директивээр агуулагдсан бол нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IFB <аргумент>
Аргумент нь хоосон бол нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IFNB <аргумент>
Аргумент нь хоосон биш бол нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IFIDN <аргумент1>, <аргумент2>

Хэрэв хоёр аргумент ижил бол нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

IFDIF <аргумент1>, <аргумент2>

Хэрэв хоёр аргумент ижил биш бол нөхцөлт блок дахь командуудыг гүйцэтгэнэ.

Одоо IFNB (if not blank) директивийн хамгийн энгийн жишээг үзье. DOS INT 21H тасалдлыг хэрэглэх үед AH регистрт утга өгч, функцүүдийг сонгодог ба харин зарим тохиолдолд л DX регитсрт утга шаарддаг. Энэ шаардлагуудыг хангах макрог бичвэл:

```
DOS21   MACRO   DOSFUNC, DXADDRESS
        MOV     AH, DOSFUNC
        IFNB    <DXADDRESS>
        MOV     DX, AFFSET DXADDRESS
        ENDIF
        INT     21H
        ENDM
```

DOS21 01 гэж программд бичвэл AH регистрт 01 гэсэн утга өгч, DOS-ын 1 дүгээр тасалдлыг ашиглан гарнаас утга авна.

DOS21 0AH, IPFIELD гэж программд бичвэл AH регистрт 0AH гэсэн утга өгч, DX регистрт оруулах хаягийг авч, тэмдэгтийг оруулна.

Нэгтгэл (concatenation) ‘**&**’

Ассемблер программд &(ampersand) тэмдэгт нь текст ба тэмдэгтүүдийг нэгтгэхэд хэрэглэгдэнэ. Доорх MOVE нэртэй макро нь MOVSB болон MOVSW командуудыг гүйцэтгэнэ.

```
MOVEMACRO   TAG
REP MOVS&TAG
ENDM
```

Хэрэглэгч нь TAG-ийн оронд B, W 2 үсгийн нэгийг өгснөөр

REP MOVSB, REP MOVSW гэсэн командуудын аль нэг нь сонгогдоно.

EXITM директив

Макро дахь нөхцөлт директивийн нөхцөл нь үнэн бол макрогаас гарна.

```
IFxx  [нөхцөл]
…
EXITM
…
ENDIF
```

EXITM мөр дээр очмогц макрог үргэлжлүүлэхээ больж, шууд ENDM директивийн дараагийн мөрөнд очно. Мөн REPT, IRP, IRPC зэрэг үйлдлүүдийг таслахдаа хэрэглэж болно.

Textbook

1. Peter Abel, “IBM PC Assembler language and programming”, USA, 1987
2. Jim Mischel, "Macro magic with Turbo Assember ”, USA, 1993
3. William C.Runnion, “Structured programming in Assembly Language for the IBM PC”, Boston, 1988
4. Thomas A.Wadlow, “Memory resident programming on the IBM PC”, USA, 1987
5. Robert S.Lai, “Writing MS-DOS device drivers”, USA, 1987
6. E. Majigsuren, “IBM assembly language”, MGL, 2003
7. Muhammad Ali Mazidi, Janice Gillispie Mazidi, Rolin D. McKinlay, “The 8051 microcontroller and Embedded Systems Using Assembly and C”,USA, 2007