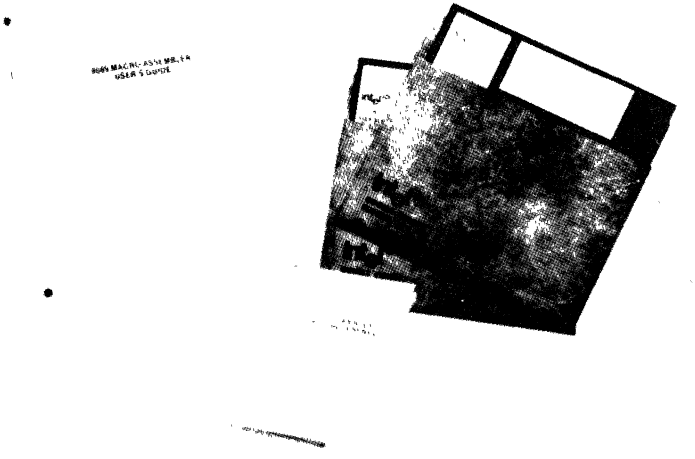
8089 IOP
SOFTWARE SUPPORT PACKAGE
#407200

- **Program Generation for the 8089 I/O Processor on the Intellec® Microcomputer Development System**
- **Contains 8089 Macro Assembler, plus Relocation and Linkage Utilities**
- **Relocatable Object Module Compatible with All iAPX 86 and iAPX 88 Object Modules**
- **Fully Supports Symbolic Debugging with the RBF-89 Software Debugger**
- **Supports 8089-Based Addressing Modes with a Structure Facility that Enables Easy Access to Based Data**
- **Powerful Macro Capabilities**
- **Provides Timing Information in Assembly Listing**
- **Fully Detailed Set of Error Messages**

The IOP Software Support Package extends Intellec Microcomputer Development System support to the 8089 I/O Processor. The macro assembler translates symbolic 8089 macro assembly language instructions into relocatable machine code. The relocation and linkage utilities provide compatibility with iAPX 86, iAPX 88, and 8089 modules, and make structured, modular programming easier.

The macro assembler also provides symbolic debugging capability when used with the RBF-89 software debugger. 8089 program modularity is supported with inter-segment jumps and calls. The macro assembler also provides instruction cycle counts in the listing file, for giving the programmer execution timing information. The programs in the 8089 Software Support Package run on any Intellec Series II or Model 800 with 64K bytes of memory.



MAY 1983

ORDER NUMBER:210853-002

FUNCTIONAL DESCRIPTION

The IOP Software Support Package contains:

ASM89 —The 8089 Macro Assembler.

LINK86 —Resolves control transfer references between 8089 object modules, and data references in 8086, 8088, and 8089 modules.

LOC86 —Assigns absolute memory addresses to 8089 object modules.

OH86 —Converts absolute object modules to hexadecimal format.

UPM —The Universal PROM Mapper, which supports PROM programming in all iAPX 86/11 and iAPX 88/11 applications.

ASM89 translates symbolic 8089 macro assembly language instructions into the appropriate machine codes. The ability to refer to both program and data addresses with symbolic names makes it easier to develop and modify programs, and avoids the errors of hand translation.

The powerful macro facility allows frequently used code sequences to be referred to by a single name, so that any changes to that sequence need to be made in only one place in the program. Common code sequences that differ only slightly can also be referred to with a macro call, and the differences can be substituted with macro parameters.

ASM89 provides symbolic debugging information in the object file. The RBF-89 debugger makes use of this information, so the programmer can symbolically debug 8089 programs. ASM89 also provides cycle counts for each instruction in the assembly listing file (see Table 1). These cycle counts help the programmer determine how long a particular routine or code sequence will take to execute on the 8089.

ASM89 provides relocatable object module compatibility with the 8086 and 8088 microprocessors. This object module compatibility, along with the 8086/8088 relocation and linkage utilities, facilitates the designing of iAPX 86/11 and iAPX 88/11 systems.

ASM89 fully supports the based addressing modes of the 8089. A structure facility allows the user to define a template that enables accessing of based data symbolically.

SPECIFICATIONS

Operating Environment

Intel Microcomputer Development Systems (Model 800, Series II, Series III, Series IV)

Support

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

Documentation Package

8089 Macro Assembler User's Guide (9800938)

8089 Macro Assembler Pocket Reference (9800936)

MCS-86 Software Development Utilities Operating Instructions for ISIS-II Users (9800639)

Universal PROM Programmer User's Manual (9800819)

Shipping Media

—Single and Double Density Diskettes

ORDERING INFORMATION

Part Number	Description
MDS*-312	8089 IOP Software Support Package

Requires Software License

*MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation

Table 1. Sample Program Listing

```
8089 MACRO ASSEMBLER


ISIS-II 8089 MACRO ASSEMBLER X105 ASSEMBLY OF MODULE TASK
OBJECT MODULE PLACED IN :F1:TASK.OBJ
ASSEMBLER INVOKED BY:  asm89 :f1:task.a89 gen macro debug pagewidth(132) print(:f1:taskx.lst)


LOC    OBJECT CODE              TIMING    INC MAC  LINE SOURCE

                                                    1  ;*******************************************
                                                    2  ;*                                         *
                                                    3  ;*              8089 TASK PROGRAM          *
                                                    4  ;*                                         *
                                                    5  ;*******************************************
                                                    6
                                                    7  name TASK
0000                                                8  TASK segment
                                                    9
                                                   10          In the first part of this   mple program data is moved from
                                                   11  ;       8086 system RAM to memory   cal to the 8089 IOP   In the second
                                                   12  ;       part, the data is moved f    the local memory to a data port
                                                   13  ;       also in the 8089 I/O spac
                                                   14
  C888                                             15  data@port@8251  equ     0c888h          ;8251 DP on 8089 local bus
  C881                                             16  command@port@8251 equ   0c881h          ;8251 CP on 8089 local bus
  0200                                             17  buffer@8089     equ     0200h           ;RAM buffer in 8089 I/O space
                                                   18
                                                   19  extrn   buffer@8086                     ;RAM buffer in 8086 system memory
                                                   20  extrn   y                               ;location of the buffer count
                                                   21
                                                   22  %*define (macro_1)                      *
                                                   23  (       mov     gb,buffer@8089          ; Move buffer address into GB
                                                   24          lpd     gc,y                    ; Load pointer to count into GC
                                                   25          movb    bc,[gc]                 ; Move byte count into BC
                                                   26  )
                                                   27  %*define (macro_2(param_1, param_2)) local loop
                                                   28  (       inc     %param_1                ; Increment pointer into source
                                                   29          dec     %param_2                ; Decrement byte count
                                                   30          jnz     %param_2,%loop          ; Loop back if byte count > 0
                                                   31  )
0000  1100 00000000              38    46          32  ONE:    lpd     ga,buffer@8086          ; Load register GA with address
                                                   33                    /                     ; of 8086 buffer
                                                   34  %macro_1
0006  3130 0002                  47   171    +1    35          mov     gb,buffer@8089          ; Move buffer address into GB
000A  5100 00000000              77   117    +1    36          lpd     gc,y                    ; Load pointer to count into GC
0010  6002                       92   139    +1    37          movb    bc,[gc]                 ; Move byte count into BC
                                               +1  38
0012  0098 00CD                 124   185          39  loop@8: movb    [gb],[ga]               ; Move byte from 8086 to 8089 buffer
0016  0030                      134   202          40          inc     ga                      ; Increment pointer into 8086 buffer
                                                   41  %macro_2(gb,gc)
0018  2030                      144   219    +1    42          inc     %PARAM_1
                                               +2  43      gb              ; Increment pointer into source
001A  403C                      154   236    +1    44          dec     %PARAM_2
                                               +2  45      gc              ; Decrement byte count
001C  4048 F3                   173   259    +1    46          jnz     %PAPAM_2
                                               +2  47      gc,%LOOP
                                               +2  48          LOOP@8          ; Loop back if byte count > 0
                                               +1  49
001F  1130 00C8                 191*  294          50  TWO:    mov     ga,data@port@8251       ; load GA with address of 8251 DP
0023  1130 01C8                 209   309          51          mov     ga,command@port@8251    ; load GC with address of 8251 CP
                                                   52  %macro_1
0027  3130 0002                 227   334    +1    53          mov     gb,buffer@8089          ; Move buffer address into GB
002B  5100 00000000             242   388    +1    54          lpd     gc,y                    ; Load pointer to count into GC
0031  6002                      261   402    +1    55          movb    bc,[gc]                 ; Move byte count into BC
                                               +1  56
0033  008A FD                  286*  434          57  loop@1: int     [gc],8,loop@1           ; loop until 8251 transmit ready
0036  0091 00CC                 338.  488          58          movb    [ga],[gb]               ; move message into buffer
                                                   59  %macro_2(gb,gc)
003A  2030                      348   497    +1    60          inc     %PARAM_1
                                               +2  61      gb              ; Increment pointer into source
003C  403C                      359   514    +1    62          dec     %PARAM_2
                                               +2  63      gc              ; Decrement byte count
003E  4048 F2                   377   537    +1    64          jnz     %PARAM_2
                                               +2  65      gc,%LOOP
                                               +2  66          LOOP@1          ; Loop back if byte count > 0
                                               +1  67
0041  2048                      349   562          68          hlt
0043                                               69  TASK    ends
                                                   70  END


ASSEMBLY COMPLETE, NO ERRORS FOUND
```