Schmid

1.6

INSTRUCTIONS FOR THE USE OF

- ◇ ASSEMBLER GENERATOR
- ◇ GENERAL ASSEMBLER
- ◇ GENERAL LOADER

H. J. MYERS
8-624-3155

2/1/77

IBM INTERNAL USE ONLY

TAPE #3 ROESCH "ASSEMBLER GENERATOR"

)LIB

001	TEMP	06	030,27
002	ASMO	06,	021,001
003	GENASM	06	029,001
004	LOADO	06	014,001
005	SAMPLE	07	009,001
006	PALM	06	036,001

3 GENASM : Eingabe der Spezifikationen der TARGET MASCHINE
Erzeugt APL-Funktionen

OPERATING INSTRUCTIONS FOR THE ASSEMBLER GENERATOR

THE ASSEMBLER GENERATOR IN ¨3 GENASM¨ WILL ACCEPT SPECIFICATIONS FOR A TARGET MACHINE. IT THEN GENERATES APL FUNCTIONS AND VARIABLES WHICH, WHEN ADDED TO THE ASSEMBLER SKELETON (IN 2 ASM0) WILL PRODUCE A COMPLETE ASSEMBLER. CERTAIN CONSTANTS ARE ADDED TO THE LOADER SKELETON (IN 4 LOAD0) TO COMPLETE A COMPATIBLE RELOCATABLE LOADER.

THE FUNCTION ¨GENASM¨ CAN BE USED IN EITHER BATCH OR INTERACTIVE MODES. IF YOU ENTER

```
      GENASM ''
```

YOU WILL BE IN THE INTERACTIVE MODE. IF YOU ENTER

```
      GENASM α
```

WHERE ¨α¨ IS THE NAME OF AN APL VARIABLE, YOU WILL BE IN THE BATCH MODE. THE APL VARIABLE MUST BE A CHARACTER VECTOR WITH CARRIAGE RETURN CHARACTERS AS END-OF-LINE MARKERS. THIS DATA IS ENTERED AS THOUGH YOU HAD TYPED IT IN, ONE LINE AT A TIME. IN THE INTERACTIVE MODE, YOU ARE PROMPTED FOR EACH LINE. IN THE BATCH MODE, ALL PROMPTING AND THE BATCH RESPONSE IS PRINTED AS THOUGH IT HAD BEEN ENTERED INTERACTIVELY. THE VARIABLE ¨PALM¨ IN WORKSPACE ¨5 SAMPLE¨ CAN BE USED TO SEE HOW GENASM WORKS. COPY IT INTO THE GENASM WORKSPACE AND ENTER

```
      GENASM PALM
```

IT TAKES ABOUT 45 MINUTES ON A 5100 TO COMPLETELY PROCESS ALL OF ¨PALM¨ TO PRODUCE THE PALM ASSEMBLER WHICH IS IN ¨6 PALM¨.

THERE ARE TWO KINDS OF DEFINITION INFORMATION YOU MUST PROVIDE. THE FIRST KIND HAS TO DO WITH THE BASIC CHARACTERISTICS OF THE TARGET MACHINE SUCH AS WORD LENGTH, NUMBER BASE, AND SO ON. PROMPTING FOR THIS INFORMATION IS EXPLICIT. (EACH PARAMETER WILL BE FURTHER DESCRIBED BELOW.) THE SECOND KIND OF DEFINITION INFORMATION HAS TO DO WITH THE MACHINE INSTRUCTIONS. IN MOST COMPUTERS THE INSTRUCTIONS CAN BE GROUPED INTO CLASSES OF COMMON FORMAT. FOR EXAMPLE, IN THE IBM 370, THERE ARE ¨RR¨, ¨RX¨, ¨SS¨, ETC. FORMATS OF INSTRUCTIONS. FOR EACH OF THESE CLASSES YOU WILL DEFINE THE FORMAT. THEN FOR EACH CLASS YOU WILL DEFINE THE INSTRUCTION MNEMONIC AND OP-CODE VALUE(S) FOR EACH INSTRUCTION IN THAT CLASS. YOU SUPPLY INSTRUCTION FORMAT INFORMATION WHEN YOU ARE PROMPTED WITH A ¨→¨. ONCE YOU HAVE ENTERED ALL THIS INFORMATION, YOU ENTER ¨.G¨ TO CAUSE GENERATION OF THE DESIRED APL FUNCTIONS. IF YOU WANT TO EXIT GENASM BEFORE GENERATING, ENTER ONLY ¨.¨.

INFORMATION OF THE FIRST KIND MUST BE COMPLETELY ENTERED BEFORE YOU ARE PROMPTED TO ENTER INFORMATION OF THE SECOND KIND. AT ANY TIME YOU ARE PROMPTED (EXCEPT FOR ERROR CORRECTION) YOU CAN EXIT FROM GENASM BY ENTERING AN EMPTY LINE (CARRIAGE RETURN ONLY). HOWEVER, IF YOU DO THIS BEFORE BEING PROMPTED BY THE ¨→¨ YOU WILL HAVE TO REENTER ALL DATA OF THE FIRST KIND AGAIN. ONCE

YOU ARE PROMPTED BY "→" THE BASIC DATA CANNOT BE CHANGED. TO RESTART EITHER RELOAD THE WORKSPACE, OR ENTER

```
      GENASM 0
```

BASIC INFORMATION

EACH OF THE BASIC INFORMATION NEEDED IS EXPLAINED BELOW, ALONG WITH ITS ASSOCIATED PROMPTING MESSAGE. IF AN ERROR IS DETECTED IN YOUR INPUT, YOU ARE RE-PROMPTED. AN EMPTY INPUT TERMINATES GENASM. ERRORS ENCOUNTERED IN BATCH MODE ALSO TERMINATE GENASM IMMEDIATELY WHEN BASIC INFORMATION IS BEING SUPPLIED.

DO YOU WANT ERROR CHECKING?-------------------------------------
THE PRODUCED ASSEMBLER WILL RUN FASTER IF ERROR CHECKING IS OMITTED. ON THE OTHER HAND, A SOURCE PROGRAM ERROR COULD CAUSE AN APL ERROR MESSAGE WHICH THE USER COULD NOT EASILY RELATE TO HIS SOURCE PROGRAM. YOUR CHOICE, ANSWER YES OR NO.

MACHINE NUMBER BASE:---
"NUMBER BASE" OF A COMPUTER IS THE NUMBER OF UNIQUE STATES A SINGLE DIGIT CAN HAVE. MOST COMPUTERS ARE BINARY (BASE 2) ORIENTED. HOWEVER, SOME COMPUTERS ARE BASE-10 ORIENTED. THIS PARAMETER ALLOWS YOU TO SPECIFY WHICH. NUMBER BASES 2 TO 35 ARE POSSIBLE! THE SAMPLE "PALM" COMPUTER HAS A NUMBER BASE OF 2.

DIGITS/WORD:---
A "WORD" IS A CONVENIENT COLLECTION OF DIGITS. THE ASSEMBLER WILL ONLY ASSEMBLY MULTIPLES OF WORDS. OBJECT CODE LISTINGS AND MEMORY DUMPS WILL DISPLAY ALL DATA IN WORD GROUPINGS. THE TERM "WORD" AS USED HERE, DOES NOT REFLECT WHAT THE USER OF THE TARGET MACHINE CALLS A WORD. FOR EXAMPLE, THE PALM MACHINE HAS A MEMORY ORGANIZED IN 8-BIT BYTES. HOWEVER, MACHINE INSTRUCTIONS ARE ALWAYS MULTIPLES OF 2 BYTES AND MACHINE ADDRESSES REQUIRE 2 BYTES. IT IS THEREFORE CONVENIENT TO CONSIDER A WORD AS THOUGH IT WAS 16 BITS LONG.

ADDRESS UNITS/WORD:--
THE MEMORY OF A COMPUTER IS COMPOSED OF ADDRESSABLE CELLS. EACH CELL CONTAINS AN INTEGRAL NUMBER OF DIGITS. A WORD MUST BE SOME MULTIPLE OF THESE CELLS. THIS PARAMETER INDICATES THE NUMBER OF CELLS PER WORD. THE "PALM" MACHINE HAS TWO BYTES PER WORD.

DIGITS IN MAXIMUM ADDRESS FIELD:---------------------------------
THE ADDRESS RANGE OF A COMPUTER REQUIRES SOME NUMBER OF DIGITS TO EXPRESS. FOR EXAMPLE, THE PALM COMPUTER CAN ADDRESS 65536 CELLS. THIS REQUIRES 16 DIGITS (BASE-2) TO EXPRESS. THE ASSEMBLER AND LOADER NEED THIS INFORMATION TO HANDLE ADDRESS RELOCATION CALCULATIONS.

DISPLACEMENT FIELD (SIGN AND BITS, IF ANY):----------------------
SOME COMPUTERS FORM ADDRESSES BY ADDING A DISPLACEMENT VALUE TO THE CURRENT CONTENTS OF THE PROGRAM COUNTER. IF YOUR MACHINE HAS THIS FEATURE THE ASSEMBLER WILL NEED THIS INFORMATION TO

GENERATE THE CORRECT DISPLACEMENT FIELD VALUES. SOME DISPLACEMENT FIELDS ARE SIGNED, SOME ARE NOT. (IN THIS LATTER CASE, THE SIGN IS INDICATED ANOTHER WAY. THIS WILL BE DISCUSSED LATER.) YOU INDICATE THAT A SIGN IS PRESENT IN THE DISPLACEMENT FIELD BY PREFIXING THE FIELD SIZE WITH A "+". FOR EXAMPLE, THE PALM HAS AN UNSIGNED 8-BIT DISPLACEMENT FIELD. IF IT WERE SIGNED, THE PARAMETER WOULD BE "+8". THIS PARAMETER IS OPTIONAL. AN EMPTY INPUT INDICATES NO DISPLACEMENT FIELD.

THE FOLLOWING DEFINES THE LISTING FORMAT-----------------------
DISPLAY NUMBER BASE:---
THE DISPLAY NUMBER BASE IS THE NUMBER SYSTEM IN WHICH ALL OBJECT CODE IS PRESENTED TO THE USER BY THE ASSEMBLER AND DUMP PROGRAMS. IT MUST BE A MULTIPLE OF THE MACHINE NUMBER BASE. FOR DECIMAL MACHINES IT IS TYPICALLY 10. FOR BINARY MACHINES IT IS USUALLY OCTAL (8) OR HEXADECIMAL (16). THE PALM ASSEMBLER USES 16.

NUMBER OF WORDS/LINE LISTED:-------------------------------------
SPACE WILL BE PROVIDED IN THE LISTING FOR THIS NUMBER OF WORDS TO APPEAR ON ONE LINE. IF ONE LINE OF ASSEMBLY CODE EXCEEDS THIS AMOUNT, IT WILL BE FOLDED TO THE NEXT LINE. TOO MANY WORDS PER LINE CAUSES TOO WIDE A LISTING (AND EXCESS TYPING ON AN IMPACT PRINTER). TOO FEW WORDS PER LINE CAUSES EXCESSIVE FOLDING OF LINES. THE PALM ASSEMBLER USES 4 WORDS PER LINE.

STANDARD HEADING IS: × × × × × × × × ×------------------------
YOUR HEADING IF DIFFERENT:-------------------------------------
FROM THE FOREGOING LISTING PARAMETERS A SAMPLE HEADING WAS CONSTRUCTED. YOU MAY ENTER ANY OTHER HEADING OF THE SAME LENGTH (OR SHORTER) IF YOU DESIRE. IF YOU ENTER AN EMPTY LINE THE SAMPLE HEADING WILL BE USED. THIS PARAMETER IS THE LAST OF THE BASIC INFORMATION. ONCE IT IS ENTERED YOU WILL BE PROMPTED BY "→" FOR MACHINE INSTRUCTION FORMAT INFORMATION. YOU WILL NOT BE PROMPTED FOR ANY OF THE ABOVE INFORMATION AGAIN UNLESS YOU RESET GENASM.

MACHINE FORMAT INFORMATION

TO ILLUSTRATE THE KIND OF MACHINE FORMAT INFORMATION NEEDED, WE SHOW A PALM MACHINE INSTRUCTION, IN BOTH ASSEMBLER AND OBJECT CODE FORMATS. ALSO SHOWN IS THE FORMAT DESCRIPTION GIVEN TO GENASM THAT SHOWS THE CONNECTION BETWEEN THE TWO FORMATS.

```
          GETA 2,3

    o----o----o----o----o
    | 0  | 2  | 3  | F  |
    o----o----o----o----o
      4    4    4    4

ALU 4,0; 4,ND1; 4,ND2; 4,OP
ALU(GETA F; GETR E; ...  )
```

IN THE ABOVE EXAMPLE ALL ¨ALU¨ TYPE INSTRUCTIONS ARE DIVIDED INTO 4 4-BIT FIELDS. THE FIRST FIELD ALWAYS CONTAINS A ZERO. THE SECOND FIELD HOLDS THE VALUE FOR OPERAND 1 OF THE SOURCE PROGRAM STATEMENT. FIELD 3 HOLDS THE SECOND OPERAND FROM THE SOURCE STATEMENT. AND FIELD 4 HOLDS THE OPCODE, WHICH IS 15 (HEXADECIMAL ¨F¨) FOR A GETA INSTRUCTION, ¨E¨ FOR A GETR INSTRUCTION, ETC.

THE FIRST LINE STARTING ¨ALU¨ DEFINES THE FORMAT. THE LINE BELOW IT INDICATES WHICH INSTRUCTIONS ARE OF THAT FORMAT, AND WHAT THEIR MNEMONICS AND OP-CODE VALUES ARE.

THERE ARE, THEN, FORMAT DEFINITIONS, AND OP-CODE DEFINITIONS. THE FORMAT DESCRIPTION FIRST NAMES A FORMAT CLASS, AND THEN DEFINES THE FIELD FORMATS OF THE INSTRUCTIONS WHICH MAKE UP THAT CLASS. THE FIELD DEFINITIONS (WHICH ARE SEPARATED BY SEMICOLONS) EACH CONSIST OF TWO COMPONENTS. THE COMPONENTS (WHICH ARE SEPARATED BY COMMAS) DEFINE THE WIDTH AND CONTENT OF A FIELD. THE WIDTH IS THE NUMBER OF DIGITS THAT ARE CONTAINED IN THE FIELD. THE CONTENTS OF A FIELD MAY BE A CONSTANT NUMBER (EXPRESSED AS A DECIMAL NUMBER, OR IN THE DISPLAY NUMBER BASE). THE CONTENTS MAY ALSO BE AN OPERAND FROM THE SOURCE STATEMENT. ¨ND1¨, ¨ND2¨, ETC. REFER TO OPERANDS 1, 2, ETC. IN THE STATEMENT. ¨ND¨ IS AN ABBREVIATION FOR ¨ND1¨ (EXCEPT AS WILL BE EXPLAINED LATER). USUALLY, ONE OF THE FIELDS CONTAINS THE OP-CODE VALUE. THIS IS REFERENCED AS ¨OP¨. IT IS POSSIBLE FOR AN INSTRUCTION TO HAVE MORE THAN ONE OP-CODE VALUE. ¨OP1¨, ¨OP2¨, ETC. STAND FOR THE APPROPRIATE ONE. THESE VALUES WILL BE DEFINED IN THE OP-CODE DEFINITION TO BE DESCRIBED LATER. IN ADDITION TO CONSTANTS, OPERANDS, AND OP-CODES, THE CONTENTS OF A FIELD CAN BE DEFINED AS AN APL EXPRESSION. THIS EXPRESSION CAN OPERATE ON ANY OF THE FOREGOING EXCEPT THE DISPLAY-BASE NUMBERS. FOR EXAMPLE, WE MIGHT SEE THE FOLLOWING FIELD DEFINITION:

```
     ...; 8,⌊.5×ND2; ...
```

WHICH SPECIFIES AN 8-DIGIT FIELD, CONTAINING HALF OF OPERAND 2.

```
     ...; 16,⍴ND; ...
```

SPECIFIES A FIELD OF 16 DIGITS CONTAINING THE NUMBER OF OPERANDS IN THE INSTRUCTION. (SUCH A FIELD MIGHT BE PART OF A ¨CALL¨ INSTRUCTION WHICH CAN HAVE A VARIABLE NUMBER OF OPERANDS.) NOTE, THAT IN THIS CASE, ¨ND¨ STANDS FOR THE ENTIRE OPERAND LIST, NOT OPERAND 1. THIS IS DETERMINED BY THE PRESENCE OF ¨⍴¨ IN FRONT OF ¨ND¨. IN GENERAL, ¨ND¨ STANDS FOR OPERAND 1 ONLY WHEN IT IS NOT OPERATED UPON BY AN APL FUNCTION.

THERE ARE TWO SPECIAL CASES OF FIELD-WIDTH DEFINITION. THESE ARE USED WHEN DEFINING A DISPLACEMENT FIELD, OR AN ADDRESS FIELD. USE ¨D¨ AND ¨A¨ RESPECTIVELY FOR THESE FIELDS. AN ADDRESS FIELD IS AUTOMATICALLY RELOCATED. DISPLACEMENT CALCULATIONS WILL BE PERFORMED AUTOMATICALLY ON A DISPLACEMENT FIELD. (THE DISPLACEMENT ADDRESS VALUE IS THE TARGET ADDRESS

MINUS THE LOCATION OF THE INSTRUCTION FOLLOWING THE ONE BEING PROCESSED.)

SOMETIMES IT IS DESIRABLE TO HAVE A DEFAULT VALUE FOR OMITTED OPERANDS. DEFAULT VALUES CAN BE SPECIFIED FOR A FIELD BY SUFFIXING IT WITH A COLON FOLLOWED BY THE DEFAULT VALUE. FOR EXAMPLE,

```
...; 4,ND3:15; ...
```

THE ABOVE FIELD CONTAINS THE VALUE OF OPERAND 3. HOWEVER, IF OPERAND 3 IS OMITTED, A DEFAULT VALUE OF "15" WILL BE SUPPLIED.

SO FAR ONLY SIMPLE FORMATS HAVE BEEN DISCUSSED. THE ASSEMBLER CAN HANDLE TWO KINDS OF COMPLEX INSTRUCTION FORMATS: SHORT/LONG INSTRUCTIONS, AND VARIABLE-LENGTH INSTRUCTIONS.

WHEN THE TARGET COMPUTER CONTAINS DISPLACEMENT ADDRESSING, IT IS FREQUENTLY POSSIBLE FOR AN INSTRUCTION TO HAVE TWO ADDRESSING MODES. THE SHORT MODE USES DISPLACEMENT ADDRESSING, WHILE THE LONG MODE USES A FULL ADDRESS FIELD. FOR INSTRUCTIONS OF THIS KIND, THE ASSEMBLER CAN MAKE THE PROPER CHOICE OF ADDRESSING MODE FOR THE PROGRAMMER. ALL THAT HAS TO BE SPECIFIED IS THE OPERAND WHICH CONTAINS THE ADDRESS, AND THE FORMATS FOR THE SHORT AND LONG VERSIONS OF THE INSTRUCTION. THE FOLLOWING DEFINES A BRANCH INSTRUCTION FOR THE SAMPLE "PALM" COMPUTER.

```
→ BR /ND=1/ 4,10 15 FWB ND1; 4,0; D,ND-1
CLS-S/L: 16,D008; A,ND1
```

IN THE PALM COMPUTER, THE PROGRAM COUNTER IS REGISTER 0. SHORT FORM BRANCHES CONSIST OF EXECUTING AN ADD- OR SUBTRACT-IMMEDIATE TO REGISTER 0. THE FIRST FIELD CONTAINS THE OP-CODE WHICH MUST BE SELECTED BASED ON THE DIRECTION (FORWARD OR BACKWARD) OF THE BRANCH. THE UTILITY FUNCTION, FWB, IS PROVIDED IN THE SYSTEM TO AID IN THIS SELECTION.

NOTE SEVERAL THINGS ABOUT THE EXAMPLE. THERE IS A "VARIATION" SPECIFIED OF THE FORM "/ND=1/" FOR THE BRANCH. THIS INDICATES THAT THIS IS A SHORT/LONG FORMAT, AND THAT THE ADDRESS FIELD IS OPERAND 1. THERE IS NO OP-CODE EXPLICITLY GIVEN. SINCE THERE IS ONLY ONE INSTRUCTION IN THE FORMAT CLASS, NO OP-CODE IS NEEDED, AND HENCE THE CLASS-NAME BECOMES THE INSTRUCTION MNEMONIC. ALTHOUGH THE SHORT FORM IS FIRST IN THIS EXAMPLE, THIS ORDERING IS NOT REQUIRED. THE NAME OF THE FORMAT-CLASS IS NOT SPECIFIED ON THE SECOND LINE. THE SECOND LINE IS CONSIDERED TO BE PART OF THE FIRST LINE. HOWEVER, IT MUST BE ENTERED AS AS SEPARATE LINE. (YOU WILL BE PROMPTED BY "CLS-S/L:" INSTEAD OF "→" TO ENTER THE SECOND LINE.+- THE FIRST FIELD OF THE SECOND (LONG) FORM IS "16,D008". THIS IS AN EXAMPLE OF A CONSTANT THAT IS A DISPLAY-BASE (HEXADECIMAL) NUMBER. THE LAST FIELDS OF THE FIRST AND SECOND FORMS ARE DISPLACEMENT AND ADDRESS FIELDS, RESPECTIVELY. (THE ADJUSTMENT "ND1-1" IS NECESSARY BECAUSE OF A PECULIARITY OF THE PALM COMPUTER.)

THE VARIABLE-LENGTH FORMAT VARIATION IS INDICATED BY THE VARIATION EXPRESSION "/α=ρND/". THIS INDICATES THAT THE FIELD DEFINITIONS THAT FOLLOW ARE TO BE USED WHEN THERE ARE α OPERANDS IN THE SOURCE STATEMENT. ONCE YOU HAVE ENTERED THIS FOR THE FIRST FORMAT, YOU ARE PROMPTED WITH "CLS-VAR:" TO ENTER THE NEXT LENGTH VARIATION. FOR EXAMPLE,

```
→ MULTI /2=ρND/ 8,OP1; 4,ND1; 4,ND2
CLS-VAR: /3=ρND/ 8,OP2; 4,ND1; 4,ND3; A,ND2
CLS-VAR:
```

IN THIS EXAMPLE WHEN THE INSTRUCTION HAS TWO OPERANDS, THE FIRST FORM IS USED WHERE OPERAND 2 IS A REGISTER REFERENCE. IN THE SECOND FORM (3 OPERANDS) OPERAND 2 IS AN ADDRESS, AND OPERAND 3 IS A REGISTER REFERENCE (PRESUMABLY AN INDEX REGISTER). NOTE THAT DIFFERENT OP-CODE VALUES ARE USED FOR EACH FORM. AN EMPTY LINE SIGNALS THAT YOU HAVE ENTERED ALL OF THE VARIATIONS.

OP-CODE DEFINITIONS ARE SIMPLY GROUPED IN PARENTHESES FOLLOWING THE NAME OF THE FORMAT CLASS TO WHICH THEY BELONG. FOR EXAMPLE,

```
→BRCN(BRALLM 5 D; BREQ 2 A; BRHAM 7 F; BRHE 9 1; BRHI 8 0;
→BRHL A 2; BRSM E 6; BRHSNM F 7; BRLE 0 8; BRLO 1 9;
→BRNOM 6 E; BRSNM D 5)
```

THE ABOVE DEFINES OPCODES OF THE FORMAT CLASS "BRCN" (BRANCH ON CONDITION). THE MNEMONIC "BREQ", WHICH STANDS FOR "BRANCH ON EQUAL", HAS TWO OP-CODE VALUES -- "2" AND "A" (HEXADECIMAL FOR 10). ONE OF THESE IS USED FOR THE SHORT FORM, THE OTHER USED FOR THE LONG FORM OF THE INSTRUCTION. NOTE THAT THE DEFINITION FOR EACH MNEMONIC IS SEPARATED FROM THE NEXT BY A SEMICOLON. NOTE ALSO THAT THIS DEFINITION TAKES SEVERAL LINES. ANY DEFINITION LINE THAT ENDS IN A SEMICOLON IMPLIES A CONTINUATION TO THE NEXT LINE. THIS IS ALSO TRUE FOR FORMAT DEFINITIONS. A SINGLE FIELD OR MNEMONIC DEFINITION MAY NOT BE SPLIT ACROSS LINES.

WHEN ALL OF YOUR DEFINITIONS ARE ENTERED, YOU CAN ENTER ".G" TO CAUSE CODE GENERATION. YOU WILL THEN BE PROMPTED WITH "ARE YOU FINISHED DEFINING?" ANSWER YES OR NO. IF YOU HAVE MORE DEFINITIONS TO DO LATER, SAVE THE WORKSPACE AS "1 TEMP" FOR LATER USE. YOU CAN REENTER GENASM TO ADD MORE DEFINITIONS LATER. IF YOU ANSWER YES, ALL GENASM FUNCTIONS AND VARIABLES (-EXCEPT GENASM) WILL BE EXPUNGED FROM THE WORKSPACE. IF YOU ANSWER NO YOU CAN USE YOU SHOULD ERASE GENASM AND ANYTHING YOU HAVE ADDED TO THE WORKSPACE WHICH YOU DON'T WANT INCLUDED IN YOUR ASSEMBLER WORKSPACE. THEN SAVE THE REMANANTS AS "1 TEMP". THEN LOAD "2 ASMO" AND COPY "1 TEMP" INTO IT. AFTER YOU RENAME YOUR NEW ASSEMBLER WORKSPACE AND SAVE IT, LOAD THE "4 LOADO" WORKSPACE ~~AND ENTER "GETCONS"~~. A)COPY COMMAND WILL BE DISPLAYED ON THE SCREEN TO COPY VALUES FROM "1 TEMP". YOU THEN RENAME AND SAVE YOUR NEW LOADER WORKSPACE.

THE ABOVE DESCRIPTION SHOULD HAVE FAMILIARIZED YOU WITH GENASM.

SO THAT YOU CAN READILY UNDERSTAND THE SYNTAX RULES FOR THE FORMAT AND OP-CODE DEFINITIONS THAT ARE SHOWN BELOW. WE WILL USE THE FOLLOWING NOTATION:

NOTATION	MEANING
α => ω	α IS DEFINED AS THE SEQUENCE ω
[α]	α IS OPTIONALLY PRESENT
α...	ONE OR MORE α
[α]...	0 OR MORE α
α1|α2|α3	USE EITHER α1 OR α2 OR α3
[α\ϵ]	SAME AS α [ϵ α]...

```
FORMAT-DEFINITION => CLASS-NAME [/ VARIATION /] FIELD-DEFS
CLASS-NAME => LETTER LETTER [ALFANUMERIC]...
VARIATION => NUMBER = ρND | ND = NUMBER
FIELD-DEFS => [FIELD-DEFINITION\;]
FIELD-DEFINITION => WIDTH , SOURCE
WIDTH => NUMBER | "A" | "D"
SOURCE => DISPLAY-BASE-NUMBER | ND [NUMBER] | OP [NUMBER] |
          AN APL EXPRESSION
OP-DEFINITION => CLASS-NAME ( [OP-FIELD\;] )
OP-FIELD => OP-MNEMONIC OP-VALUE...
OP-MNEMONIC => LETTER LETTER [ALFANUMERIC]...
OP-VALUE => NUMBER | DISPLAY-BASE-NUMBER
NUMBER => A BASE-10 (DECIMAL) NUMBER
```

(NOTE: CONSECUTIVE NAMES OR MNEMONICS NOT SHOWN WITH AN INTERVENING DELIMITER MUST BE SEPARATED BY BLANKS. LEADING AND TRAILING BLANKS IN A FIELD-CONTENT DEFINITION ARE IGNORED.)

LISTA - LISTS GENERATED APL FUNCTIONS.

1) FORMAT FUNCTIONS

KEYBOARD UNLOCKS FOR YOU TO SPACE PAPER AND LIST

2) OP-MNEMONICS

OPERATING INSTRUCTIONS FOR THE GENERAL ASSEMBLER

NOTATIONAL CONVENTIONS: ----------

IN DESCRIBING THE USE OF THE ASSEMBLE A META-NOTATION IS NEEDED TO SPECIFY THE SYNTAX. WE WILL USE THE FOLLOWING NOTATION:

NOTATION	MEANING
[α]	α IS OPTIONALLY PRESENT
α...	ONE OR MORE α
[α]...	0 OR MORE α
[α\ϵ]	SAME AS α [ϵ α]...

UNDERSCORED LETTERS INDICATE EXACT SPELLING OF A SYNTACTIC COMPONENT, WHILE OTHER LETTERS INDICATE VARIABLE COMPONENTS. (FOR EXAMPLE, "XYZ" MEANS USE EXACTLY THE LETTERS "XYZ". "LABEL" MEANS USE SOME LABEL FROM SOME SET OF PROPER LABELS.)

INVOKING THE ASSEMBLER: ----------

A SOURCE PROGRAM TO BE ASSEMBLED IS STORED IN YOUR APL WORKSPACE AS AN APL FUNCTION. THE FIRST LINE OF THE SOURCE PROGRAM (WHICH IS ALSO THE FUNCTION HEADER) CONTAINS ITS NAME. IN ORDER TO PERFORM ASSEMBLY, YOU ENTER

```
ASM 'NAME[,OPTIONS]'
```

WHERE "NAME" IS THE NAME OF THE PROGRAM TO BE ASSEMBLED. THE OPTIONS THAT MAY BE SPECIFIED ARE SINGLE LETTERS. THE OPTIONS CURRENTLY RECOGNIZED ARE "L", WHICH CAUSES A LISTING TO BE PRODUCED; AND "N", WHICH FORCES SHORT/LONG INSTRUCTION SELECTION TO SELECT LONG INSTRUCTIONS (AND SPEEDS UP THE ASSEMBLY). WHEN THIS METHOD OF ASSEMBLY IS USED, THE RESULTS OF THE ASSEMBLY ARE LEFT IN AN APL NUMBER VECTOR CALLED MEM. THIS VECTOR CAN BE TRANSFORMED INTO ONE MEMORY LOAD FOR THE TARGET MACHINE BY AN OFF-LOADING ROUTINE (TO BE IMPLEMENTED LATER).

IF YOU WISH TO PRODUCE A RELOCATABLE OBJECT MODULE ENTER

```
OBJ←ASMR 'NAME[,OPTIONS]'
```

AND THE APL VARIABLE "OBJ" WILL RECEIVE THE RESULTING CODE. THIS CODE CAN BE COMBINED WITH OTHER RELOCATABLE MODULES USING THE RELOCATING LOADER.

TO CREATE OR MODIFY YOUR SOURCE PROGRAM, USE THE FUNCTION EDITING FACILITIES THAT ARE AUTOMATICALLY INCLUDED IN YOUR APL SYSTEM. (OPERATION OF THE APL FUNCTION EDITOR IS DESCRIBED IN THE APL MANUAL APPROPRIATE TO YOUR APL SYSTEM. THIS IS THE ONLY ASPECT OF APL THAT YOU NEED TO KNOW IN ORDER TO USE THE ASSEMBLER.)

LABELS, EXPRESSIONS AND SPECIAL FUNCTIONS: ----------

BELOW THERE WILL BE REFERENCES TO THE SYNTACTIC COMPONENTS "LABEL", "EXPRESSION", AND "ABSOLUTE-EXPRESSION". A "LABEL" HAS THE SYNTAX OF AN APL NAME. (A LETTER FOLLOWED BY ONE OR MORE

ALFANUMERIC CHARACTERS). LABELS ARE SYMBOLIC NAMES FOR VALUES. A LABEL RECEIVES A VALUE IN A MANNER THAT DEPENDS ON THE INSTRUCTION IN WHICH IT APPEARS. IF A LABEL RECIEVES A "LOCATION COUNTER" VALUE, ITS VALUE IS RELOCATABLE. IF A LABEL RECEIVES ITS VALUE FROM AN EXTRN IT IS "EXTERNAL". OTHERWISE ITS VALUE IS "ABSOLUTE".

AN "EXPRESSION" IS AN APL ARITHMETIC EXPRESSION INVOLVING LABELS, CONSTANTS, OR FUNCTIONS. A SIMPLE EXPRESSION CONTAINS ONLY A LABEL OR CONSTANT. COMPOUND EXPRESSIONS INVOLVE FUNCTIONS OR ARITHMETIC OPERATORS. COMPOUND EXPRESSIONS THAT ARE FOLLOWED BY OPERANDS MUST BE ENCLOSED IN PARENTHSES. ALL EXPRESSIONS FOLLOW APL SYNTAX. THAT IS, THERE IS NO OPERATOR PRECEDENCE, AND EACH SEQUENCE OF OPERATIONS WITHIN ONE PARENTHESIS LEVEL IS PREFORMED FROM RIGHT TO LEFT. THE VALUE OF AN EXPRESSION IS RELOCATABLE, EXTERNAL OR ABSOLUTE DEPENDING UPON HOW ITS COMPONENTS ARE COMBINED. IF "R" STANDS FOR A RELOCATABLE VALUE, "X" STANDS FOR AN EXTERNAL VALUE, AND A STANDS FOR AN ABSOLUTE VALUE, THEN

R=R+A, R=R-A, A=R-R, X=X+A, X=X-A

THE ONLY ARITHMETIC PERMISSIBLE ON R OR X IS AS SHOWN ABOVE. ALL ARITHMETIC CAN BE PERFORMED ON A TO PRODUCE AN A. THERE ARE SEVERAL FUNCTIONS PROVIDED WITH THE ASSEMBLER THAT CAN BE USED AS AN INSTRUCTION OPERAND. THEY ARE

```
X 'HEX'
O 'OCTAL'
L EXPRESSION
Δ
```

(NOTE: THESE EXACT NAMES, X, O AND L, ARE UNDERSCORED WHEN YOU WRITE THEM.) "X" TAKES A QUOTED ARGUMENT THAT IS A HEXADECIMAL NUMBER. "O" TAKES A QUOTED ARGUMENT THAT IS AN OCTAL (BASE-8) NUMBER. "L" TAKES ANY EXPRESSION, PLACES ITS VALUE IN A LITERAL POOL, AND RETURNS ITS LOCATION IN THE POOL. (SEE LTORG.) "Δ" GIVES THE LOCATION COUNTER VALUE OF THE FIRST WORD OF THE CURRENT INSTRUCTION.

AN ABSOLUTE-EXPRESSION) IS AN EXPRESSION THAT IS NOT RELOCATABLE OR EXTERNAL.

ASSEMBLER INSTRUCTIONS: ----------

[LABEL:] ORG EXPRESSION

SET THE LOCATION COUNTER TO THE VALUE OF "EXPRESSION". IF "LABEL" IS PRESENT, ASSIGN IT THE VALUE OF THE LOCATION COUNTER AS IT WAS BEFORE THE ORG WAS ENCOUNTERED.+

LABEL EQU EXPRESSION

ASSIGN TO THE "LABEL" THE VALUE OF THE "EXPRESSION".

EXTRN '[LABEL\,]'

DEFINE EACH OF THE LABELS APPEARING IN THE OPERAND LIST TO BE EXTERNAL. AN EXTERNAL LABEL IS ONE WHICH WILL HAVE ITS VALUE DEFINED IN SOME OTHER PROGRAM MODULE. (SEE ENTRY.) IF ANY OF THE LABELS THAT APPEAR IN AN EXTRN OPERAND ARE DEFINED IN THE CURRENT PROGRAM, THEIR PRESENCE IN THE EXTRN WILL BE IGNORED. EXTERNAL LABELS MUST NOT BE LONGER THAN 6 CHARACTERS. (NOTE THAT THE OPERAND OF AN EXTRN IS QUOTED.)

ENTRY [LABEL\,]

ALLOW THE LABELS IN THE OPERAND LIST TO BE KNOWN BY OTHER PROGRAM MODULES. (SEE EXTRN.) THE LABELS IN THE OPERAND LIST MUST BE DEFINED IN THE CURRENT PROGRAM.

[LABEL:] DS ABSOLUTE-EXPRESSION

DEFINE SPACE. THE LOCATION COUNTER IS ADVANCED BY THE NUMBER OF WORDS CALCULATED BY "ABSOLUTE-EXPRESSION". IF A "LABEL" IS PRESENT ITS VALUE IS ASSIGNED AS THE LOCATION OF THE FIRST WORD.

[LABEL:] DC [EXPRESSION\,]

DEFINE CONSTANT(S). EACH EXPRESSION IN THE OPERAND IS EVALUATED AND ITS VALUE IS PLACED IN ONE WORD. VALUES MAY BE RELOCATABLE OR ABSOLUTE. IF A LABEL IS PRESENT, ITS VALUE IS THE ADDRESS OF THE FIRST CONSTANT.

LTORG

THIS ASSEMBLER INSTRUCTION FORMS A LITERAL POOL WHERE IT IS ENCOUNTERED. LITERALS APPEAR IN THE OPERANDS OF MACHINE AND DC INSTRUCTIONS. THEY TAKE THE FORM

L EXPRESSION

AS EACH LITERAL APPEARS, ITS EXPRESSION IS EVALUATED. THE VALUE IS PLACED INTO A LITERAL POOL. DUPLICATE VALUES OCCUPY ONE SPACE. THE LTORG INSTRUCTION INDICATES WHERE THE LITERAL POOL IS TO BE LOCATED. LITERALS APPEARING AFTER A LTORG ARE PLACED IN THE NEXT POOL. THE END INSTRUCTION (SEE BELOW) ACTS AS A FINAL LTORG INSTRUCTION.

TITLE 'ANY CHARACTER STRING'

THE OPERAND IS USED AS A PAGE TITLE ON SUBSEQUENT PAGES OF THE OUTPUT LISTING. (THE FIRST TITLE WILL BE IN EFFECT ON THE FIRST PAGE, NO MATTER WHERE IT APPEARS.)

EJECT NUMBER

IF THERE ARE LESS THAN "NUMBER" OF LINES LEFT ON THE CURRENT

PAGE OF THE LISTING, EJECT TO THE NEXT PAGE.

SPACE NUMBER

SKIP "NUMBER" OF BLANK LINES ON THE CURRENT PAGE, OR TO THE END OF THE PAGE, WHICHEVER OCCURS FIRST.

END

THE END INSTRUCTION MUST APPEAR AT THE END OF EACH SOURCE PROGRAM. (IT IS AUTOMATICALLY ADDED BY THE ASSEMBLER IF YOU OMIT IT.) IT ACTS AS A LTORG INSTRUCTION BY DEFAULT.+

MACHINE INSTRUCTIONS: ----------

MACHINE INSTRUCTIONS HAVE ONE OF THE TWO FOLLOWING FORMATS.

```
     [LABEL:] MACHINE-MNEMONIC
OR
     [LABEL:] MACHINE-MNEMONIC [EXPRESSION\,]
```

THE SPECIFIC MACHINE INSTRUCTIONS VARY FROM ASSEMBLER TO ASSEMBLER, BUT THEY ALL HAVE THE SAME GENERAL FORMAT SHOWN ABOVE. THE NUMBER OF OPERANDS REQUIRED BY EACH DEPENDS ON THE PARAMETERS PROVIDED DURING ASSEMBLER GENERATION. SOME INSTRUCTIONS MAY HAVE NO OPERAND, SOME MAY HAVE A FIXED NUMBER, AND SOME MAY HAVE A VARIABLE NUMBER OF OPERAND EXPESSIONS. INSTRUCTIONS THAT HAVE A VARIABLE NUMBER OF OPERANDS MUST HAVE AT LEAST ONE.

OPERATING INSTRUCTIONS FOR THE GENERAL LOADER

THE PURPOSE OF THE LOADER IS TO TAKE RELOCATABLE OBJECT PROGRAMS AS PRODUCED BY ASMR AND COMBINE THEM INTO A SINGLE STORAGE LOAD. TO INVOKE THE LOADER ENTER

```
          LOAD α
```

WHERE "α" STANDS FOR SOME DECIMAL NUMBER THAT WILL BE THE LOCATION INTO WHICH THE FIRST PROGRAM IS LOADED. (YOU CAN USE THE FUNCTION "X" TO SUPPLY A HEXADECIMAL NUMBER. -- VIZ. "LOAD X'3C4'".)

ONCE YOU CALL "LOAD" YOU WILL BE PROMPTED TO ENTER THE NAMES OF THE PROGRAMS TO BE LOADED. YOU ENTER ONE OR MORE NAMES, SEPARATED BY BLANKS. IF ALL OF THE NAMES WILL NOT FIT IN A SINGLE LINE, END THE LINE WITH A BLANK AND YOU WILL BE PROMPTED TO ENTER MORE NAMES. WHEN ALL OF THE PROGRAMS ARE LOADED, THE NUMBER OF THE NEXT AVAILABLE LOCATION WILL BE DISPLAYED. IT IS NOT NECESSARY TO LOAD ALL OF THE PROGRAMS AT ONE TIME. YOU CAN ADD ADDITIONAL PROGRAMS LATER.

LOADING AT LOCATION ZERO (0) AUTOMATICALLY RESETS THE STORAGE AND EXTERNAL SYMBOL TABLE TO EMPTY. IF YOU WANT TO LOAD AT ZERO WITHOUT THIS RESET, USE THE NUMBER ¯1 (NEGATIVE ONE).

THE OBJECT STORAGE IS THE APL NUMBER VECTOR "MEM". IT WILL BECOME AS LONG AS NECESSARY TO CONTAIN THE LOADED DATA. THE FIRST ELEMENT OF MEM IS LOCATION ZERO (0) OF THE TARGET MACHINE. EACH ELEMENT OF MEM WILL CONTAIN ONE "WORD" (AS DEFINED BY THE "GENASM" PROCESS).

THE RELOCATABLE LOADER HAS THE TASK OF RESOLVING ALL EXTERNAL LABELS. WHEN ALL OF THE PROGRAMS ARE LOADED, UNRESOLVED LABELS ARE DISPLAYED IF THERE ARE ANY. YOU CAN GET A LIST OF ALL EXTERNAL LABELS AND THEIR VALUES BY ENTERING "SYMBOLS". YOU CAN ALSO GET A DUMP OF THE LOADED STORAGE BY ENTERING

```
          DUMP α ω
```

WHERE LOCATIONS α THROUGH ω ARE TO BE DUMPED. THE DATA DUMPED WILL BE DISPLAYED IN THE SAME FORMAT (NUMBER BASE) AS IN THE ASSEMBLY LISTING.

Beispiel

```
      GENASM PALM
DO YOU WANT ERROR CHECKING? Y
MACHINE NUMBER BASE: 2
DIGITS/WORD: 16
ADDRESS UNITS/WORD: 2
DIGITS IN MAXIMUM ADDRESS FIELD: 16
DISPLACEMENT FIELD (SIGN AND DIGITS, IF ANY): 8
THE FOLLOWING DEFINES THE LISTING FORMAT
DISPLAY NUMBER BASE: 16
NUMBER OF WORDS/LINE LISTED: 4
LINES/PAGE IN LISTING: 66
STANDARD HEADING IS: LOC  WD1  WD2  WD3  WD4
YOUR HEADING IF DIFFERENT:
→ ALU 4,0; 4,ND1; 4,ND2; 4,OP
→ MOVE 4,0; 4,ND2; 4,ND1; 4,MVEX ND3:0
→ SHF 8,E0; 4,ND; 4,OP
→ IND 4,OP; 4,ND1; 4,ND2; 4,MMEX ND3:0
→ DIR 4,OP; 4,ND1; 8,⌊.5×ND2
→ IOR 4,OP; 4,1⌈ND1; 4,ND2; 4,MMEX ND3:0
→ JMP 4,C; 4,ND1; 4,ND2; 4,OP
→ JMP1 4,C; 4,ND; 4,0; 4,OP
→ ALU(GETA F; GETR E; ADDS1 A; ADDS2 B; SUB 9; ADDB 8;
→  XOR 7; OR 6; AND 5; LTH D; HTL C)
→ SHF(SHFTR C; ROTR D; SRR3 E; SRR4 F)
→ IND(LDHI D; STHI 5; LDBI 6; STBI 7)
→ DIR(LDHD 2; STHD 3; EMIT 8; CLRI 9; ADDI A; SETI B;
→  SUBI F; CTL 1)
→ IOR(PUTB 4; GETB E)
→ JMP(JALLM 5; JEQ 2; JHAM 7; JHE 9; JHI 8; JHL A; JSM E;
→  JHSNM F; JLE 0; JLO 1; JNOM 6; JSNM D)
→ JMP1(JALL 4; JNO 3; JSB B; JSN C)
→ BR /ND=1/ 4,10 15 FWB ND; 4,0; D,ND-1
CLS-L/S: 16,D008; A,ND
→ BRCN /ND=3/ 4,C; 4,ND1; 4,ND2; 4,OP2; 4,10 15 FWB ND3; 4,0; D,ND3-1
CLS-L/S: 4,C; 4,ND1; 4,ND2; 4,OP1; 16,A004; 16,D008; A,ND3
→ BRCN(BRALLM 5 D; BREQ 2 A; BRHAM 7 F; BRHE 9 1; BRHI 8 0;
→  BRHL A 2; BRSM E 6; BRHSNM F 7; BRLE 0 8; BRLO 1 9;
→  BRNOM 6 E; BRSNM D 5)
→ BRCN1 /ND=2/ 4,C; 4,ND1; 4,0; 4,OP2; 4,10 15 FWB ND2; 4,0; D,ND2-1
CLS-L/S: 4,C; 4,ND1; 4,0; 4,OP1; 16,A004; 16,D008; A,ND2
→ BRCN1(BRALL 4 C;  BRNO 3 B;  BRSB B 3;  BRSN C 4)
→ .G
NAME FINAL ASM WS:



  01/20/77 11:31:55
```

```
     ∇ ALU OP
[1]    →PASS2/A1
[2]    LCX[''ρLV↑⎕LC;]←2
[3]    →0
[4]   A1:→(2=ρND←,ND)/A2
[5]    ND←LER ERR(2↑ND),1
[6]   A2:EMIT POK PACK 0,ND[ 0 1 ],OP
     ∇

     ∇ BR ND;LC
[1]    LC←''ρLV↑⎕LC
[2]    →PASS2/A2
[3]    →(0≠ρρPASS2)/A1
[4]    LCX[LC;]← 2 4
[5]    →~SL[LC]←1
[6]   A1:255 CKREACH 0
[7]    →0
[8]   A2:→(1=ρND←,ND)/A3
[9]    ND←LER ERR(1↑ND),1
[10]  A3:→(2≠LCX[LC])/A4
[11]   EMIT P2K PACK( 10 15 FWB ND),0,|DISP ND-1
[12]   →0
[13]  A4:EMIT 53256,1 RELOC ND
     ∇

     ∇ BRCN OP;LC
[1]    LC←''ρLV↑⎕LC
[2]    →PASS2/A2
[3]    →(0≠ρρPASS2)/A1
[4]    LCX[LC;]← 4 8
[5]    →~SL[LC]←1
[6]   A1:255 CKREACH 2
[7]    →0
[8]   A2:→(3=ρND←,ND)/A3
[9]    ND←LER ERR(3↑ND),1
[10]  A3:→(4≠LCX[LC])/A4
[11]   EMIT(POK PACK 12,ND[ 0 1 ],OP[1]),P2K PACK( 10 15 FWB ND[2]
       ),0,|DISP ND[2]-1
[12]   →0
[13]  A4:EMIT(POK PACK 12,ND[ 0 1 ],OP[0]),40964,53256,3 RELOC ND[
       2]
     ∇

     ∇ BRCN1 OP;LC
[1]    LC←''ρLV↑⎕LC
[2]    →PASS2/A2
[3]    →(0≠ρρPASS2)/A1
[4]    LCX[LC;]← 4 8
[5]    →~SL[LC]←1
[6]   A1:255 CKREACH 1
[7]    →0
[8]   A2:→(2=ρND←,ND)/A3
[9]    ND←LER ERR(2↑ND),1
[10]  A3:→(4≠LCX[LC])/A4
[11]   EMIT(POK PACK 12,ND[0],0,OP[1]),P2K PACK( 10 15 FWB ND[1]),
       0,|DISP ND[1]-1
[12]   →0
[13]  A4:EMIT(POK PACK 12,ND[0],0,OP[0]),40964,53256,3 RELOC ND[1]
     ∇
```

```
      ∇ DIR OP
[1]     →PASS2/A1                                          01/20/77 11:31:55
[2]     LCX[''ρLV↑⎕LC;]←2
[3]     →0                                                 IBM INTERNAL USE ONLY
[4]    A1:→(2=ρND←,ND)/A2
[5]     ND←LER ERR(2↑ND),1
[6]    A2:EMIT P2K PACK OP,ND[0],⌊0.5×ND[1]
      ∇

      ∇ IND OP
[1]     →PASS2/A1
[2]     LCX[''ρLV↑⎕LC;]←2
[3]     →0
[4]    A1:→((3≥ρND)∧2≤ρND←,ND)/A2
[5]     ND←LER ERR( 3 2 [2>ρND]↑ND),1
[6]    A2:ND←3↑ND
[7]     EMIT POK PACK OP,ND[ 0 1 ],MMEX ND[2]
      ∇

      ∇ IOR OP
[1]     →PASS2/A1
[2]     LCX[''ρLV↑⎕LC;]←2
[3]     →0
[4]    A1:→((3≥ρND)∧2≤ρND←,ND)/A2
[5]     ND←LER ERR( 3 2 [2>ρND]↑ND),1
[6]    A2:ND←3↑ND
[7]     EMIT POK PACK OP,(1⌈ND[0]),ND[1],MMEX ND[2]
      ∇

      ∇ JMP OP
[1]     →PASS2/A1
[2]     LCX[''ρLV↑⎕LC;]←2
[3]     →0
[4]    A1:→(2=ρND←,ND)/A2
[5]     ND←LER ERR(2↑ND),1
[6]    A2:EMIT POK PACK 12,ND[ 0 1 ],OP
      ∇

      ∇ JMP1 OP
[1]     →PASS2/A1
[2]     LCX[''ρLV↑⎕LC;]←2
[3]     →0
[4]    A1:→(1=ρND←,ND)/A2
[5]     ND←LER ERR(1↑ND),1
[6]    A2:EMIT POK PACK 12,ND,0,OP
      ∇

      ∇ MOVE ND
[1]     →PASS2/A1
[2]     LCX[''ρLV↑⎕LC;]←2
[3]     →0
[4]    A1:→((3≥ρND)∧2≤ρND←,ND)/A2
[5]     ND←LER ERR( 3 2 [2>ρND]↑ND),1
[6]    A2:ND←3↑ND
[7]     EMIT POK PACK 0,ND[ 1 0 ],MVEX ND[2]
      ∇
      ∇ SHF OP
[1]     →PASS2/A1
[2]     LCX['⍴LV↑⎕LC;]←2
[3]     →0
[4]    A1:→(1=⍴ND←,ND)/A2
[5]     ND←LER ERR(1↑ND),1
[6]    A2:EMIT P1K PACK 224,ND,OP
      ∇
```

```
      P0K
16 16 16 16

      P1K
256 16 16

      P2K
16 16 256
```

OPCODES (...)

```
      ∇ ADDB ND
[1]     ALU 8

      ∇ ADDI ND   (A)
[1]     DIR 10

      ∇ ADDS1 ND   (0)
[1]     ALU 10  + modifier

      ∇ ADDS2 ND   (0)
[1]     ALU 11

      ∇ AND ND   (0)
[1]     ALU 5

      ∇ BRALL ND
[1]     BRCN1 4 12

      ∇ BRALLM ND
[1]     BRCN 5 13

      ∇ BREQ ND
[1]     BRCN 2 10

      ∇ BRHAM ND
[1]     BRCN 7 15

      ∇ BRHE ND
[1]     BRCN 9 1

      ∇ BRHI ND
[1]     BRCN 8 0

      ∇ BRHL ND
[1]     BRCN 10 2

      ∇ BRHSNM ND
[1]     BRCN 15 7

      ∇ BRLE ND
[1]     BRCN 0 8

      ∇ BRLO ND
[1]     BRCN 1 9

      ∇ BRNO ND
[1]     BRCN1 3 11
```

```
      ∇ BRNOM ND
[1]     BRCN 6 14

      ∇ BRSB ND
[1]     BRCN1 11 3

      ∇ BRSM ND
[1]     BRCN 14 6

      ∇ BRSN ND
[1]     BRCN1 12 4

      ∇ BRSNM ND
[1]     BRCN 13 5

      ∇ CLRI ND   (9)
[1]     DIR 9

      ∇ CTL ND   (1)
[1]     DIR 1

      ∇ EMIT ND   (8)
[1]     DIR 8

      ∇ GETA ND   (0)
[1]     ALU 15

      ∇ GETB ND   (E)
[1]     IOR 14

      ∇ GETR ND   (0)
[1]     ALU 14

      ∇ HTL ND   (0)
[1]     ALU 12

      ∇ JALL ND   (C)
[1]     JMP1 4

      ∇ JALLM ND
[1]     JMP 5

      ∇ JEQ ND
[1]     JMP 2

      ∇ JHAM ND
[1]     JMP 7
```

```
      ∇ JHE ND
[1]     JMP 9

      ∇ JHI ND
[1]     JMP 8

      ∇ JHL ND
[1]     JMP 10

      ∇ JHSNM ND
[1]     JMP 15

      ∇ JLE ND
[1]     JMP 0

      ∇ JLO ND
[1]     JMP 1

      ∇ JNO ND
[1]     JMP1 3

      ∇ JNOM ND
[1]     JMP 6

      ∇ JSB ND
[1]     JMP1 11

      ∇ JSM ND
[1]     JMP 14

      ∇ JSN ND
[1]     JMP1 12

      ∇ JSNM ND
[1]     JMP 13

      ∇ LDBI ND
[1]     IND 6

      ∇ LDHD ND
[1]     DIR 2

      ∇ LDHI ND
[1]     IND 13

      ∇ LTH ND
[1]     ALU 13
```

```
                 ORB
      ∇ OR ND          0/6
[1]     ALU 6

      ∇ PUTB ND   4/...
[1]     IOR 4

      ∇ ROTR ND   E/...
[1]     SHF 13

      ∇ SETI ND   B/-
[1]     DIR 11

      ∇ SHFTR ND   E/C
[1]     SHF 12

      ∇ SRR3 ND
[1]     SHF 14

      ∇ SRR4 ND
[1]     SHF 15

      ∇ STBI ND   7/...
[1]     IND 7

      ∇ STHD ND   3/-
[1]     DIR 3

      ∇ STHI ND   5/...
[1]     IND 5

      ∇ SUB ND   0/9
[1]     ALU 9

      ∇ SUBI ND   F/-
[1]     DIR 15

      ∇ XOR ND   0/7
[1]     ALU 7
```

bei allen OP-Codes mit OP-Code "0" wird Alu + Modifier aufgerufen!

01/20/77 11:31:55

IBM INTERNAL USE ONLY

```
      OBJ1←ASMR 'TESTP,L'
LOC  WD1  WD2  WD3  WD4      | SAMPLE ASSEMBLY                PAGE 1
X--                         2|   EXTRN 'X1,X2'
E--                         3|   ENTRY A1,A3
0000=      0:A              4|A  EQU 0
0000=      0:B              5|B  EQU ∆
0200:0000                   6|   ORG X '200'
0200:023F                   7|A1 GETA 2,3
0202:0234                   8|   MOVE 3,2
0204:0233                   9|   MOVE ¯3,2,2
0206:E05C                  10|   SHFTR 5
0208:D234                  11|   LDHI 2,3,¯1
020A:3201                  12|   STHD 2,3
020C:2303                  13|   LDHD 3,B+6
020E:2303                  14|   LDHD 3,A+6
0210:46F8                  15|   PUTB 6,15
0212:E7B1                  16|   GETB 7,11,2
0214:C904                  17|   JALL 9

C--                        19|⍝LONG/SHORT BRANCHES
0216:F019                  20|   BR A1
0218:A017                  21|   BR A2
021A:D008 043C             22|   BR A3
021E:D008 0000             23|   BR X1
0222:C40C F027             24|   BRALL 4,A1
0226:C50C A007             25|   BRALL 5,A2
022A:C804 A004 D008 043C   26|   BRALL 8,A3


C--                        28|⍝GAP TO TEST BRANCH RANGE SELECTION
0232:                      29|A2 DS X '200'
0432:0001 0002 0003 043C   30|A4 DC 1 2 3 ,A3,L 9
     045E
043C:D008 0200             31|A3 BR A1
0440:D008 0232             32|   BR A2
0444:F00B                  33|   BR A3
0446:D008 0000             34|   BR X2
044A:C245 A004 D008 0200   35|   BRALLM 2,4,A1
0452:C355 A004 D008 0232   36|   BRALLM 3,5,A2
045A:C68D F023             37|   BRALLM 6,8,A3
045E:0009                  38|   LTORG
0460:0001 0002 0003        39|A5 DC 1 2 3
0466      1126             40|   END

SYMBOL TABLE
 A       4    0:A 0000
 A1      7  512:E 0200
 A2     29  562:R 0232
 A3     31 1084:E 043C
 A4     30 1074:R 0432                           01/20/77 11:31:55
 A5     39 1120:R 0460
 B       5    0:R 0000                           IBM INTERNAL USE ONLY
 X1      2    0:X 0000
 X2      2    1:X 0001

ERRORS:
  13: FIELD OVERFLOW
```

```
      OBJ2←ASMR 'TEST2,L'
LOC  WD1  WD2  WD3  WD4       | SECOND PROGRAM     PAGE 1
E--                          2|   ENTRY X1,X2
X--                          3|   EXTRN 'A1,A3'
0000:D008 0000               4|X1 BR A3
0004:D008 0000               5|X2 BR A1
0008         8               6|   END

SYMBOL TABLE
 A1  3 0:X 0000
 A3  3 1:X 0001
 X1  4 0:E 0000
 X2  5 4:E 0004

      LOAD 0                                SYMBOLS
ENTER OBJECT DECK NAMES             A1     OBJ1  512 0200
→OBJ1 OBJ2                          A3     OBJ1 1084 043C
NEXT LOCATION: 1134 (046E)          X1     OBJ2  563 0233
                                    X2     OBJ2  567 0237
      DUMP 0 1134

   0000 THRU 01FE CONTAIN 0000
0200: 023F 0234 0233 E05C D234 3201 2303 2303
0210: 46F8 E7B1 C904 F019 A017 D008 043C D008
0220: 0233 C40C F027 C50C A007 C804 A004 D008
0230: 043C 0000 0000 0000 0000 0000 0000 0000
   0240 THRU 042E CONTAIN 0000
0430: 0000 0001 0002 0003 043C 045E D008 0200
0440: D008 0232 F00B D008 0237 C245 A004 D008
0450: 0200 C355 A004 D008 0232 C68D F023 0009
0460: 0001 0002 0003 D008 043C D008 0200

      LOAD 0                                SYMBOLS
ENTER OBJECT DECK NAMES             X1     OBJ2    0 0000
→OBJ2 OBJ1                          X2     OBJ2    4 0004
NEXT LOCATION: 1134 (046E)          A1     OBJ1  516 0204
                                    A3     OBJ1 1088 0440
      DUMP 0 1134

0000: D008 0440 D008 0204 0000 0000 0000 0000
   0010 THRU 01FE CONTAIN 0000
0200: 0000 0000 0000 0000 023F 0234 0233 E05C
0210: D234 3201 2303 2303 46F8 E7B1 C904 F019
0220: A017 D008 0440 D008 0000 C40C F027 C50C
0230: A007 C804 A004 D008 0440 0000 0000 0000
   0240 THRU 042E CONTAIN 0000
0430: 0000 0000 0000 0000 0000 0001 0002 0003
0440: 0440 0462 D008 0204 D008 0236 F00B D008
0450: 0004 C245 A004 D008 0204 C355 A004 D008
0460: 0236 C68D F023 0009 0001 0002 0003
```

```
THE FOLLOWING CHANGES HAVE BEEN MADE TO THE 2/1/77 VERSION
OF THE ASSEMBLER GENERATION SYSTEM
NOTATION:
[∆N]          DELETE LINE N
[N+]          INSERT FOLLOWING LINE N
[N/] /α/ω/    CHANGE ALL α TO ω IN LINE N
[N→M+]        MOVE LINE N TO FOLLOW LINE M

*** 2/11/77
DODFLT[9/] /⍕/GENPKS/
DOFIELD[17/] /⍕/GENPKS/
DOSOURCE[10/] /¯2/¯1/∆DR++/+/
GENER[2/] /NOCK/NOCK∨J[0]=0/
LISTA[0/] /⎕IO/⎕IO;⎕PW/
LISTA[1+] ⎕PW←130
LISTA[13→9+]
DUMP[11 16/] /CVA/,CVA/
END[14/] /CVA/,CVA/PRT I/PRT ¯1/
LOAD[16/] /CVH/,CVA/
SYMBOLS[3/] /CVH/0 ¯1↓CVA/
∧[4/] /0∊/~0∊/
CVA[1/] /NABτ''ρ/⍉NABτ/
DOENL[18/] /MEM/MEM/
PRSYM[8/] /CVH/0 ¯1↓CVA/
X[1/] /WL|//
LOAD[14+] J←LC+⍳N[3]
LOAD[15+] ∆3:→⍳0=ρJ←(MEM[J]≥WL)/J
          MEM[J]←(2ρWL)⊤WL⊥MEM[J← ¯1 0 ∘.+J]
          J←J[0;]
          →∆3
```