SOURCE ACCESS NAME= PPC2.P359.SRC.NUD
OBJECT ACCESS NAME= PPC2.P359.OBJ.NUD359
LISTING ACCESS NAME= PPC2.P359.LST.NUD359
ERROR ACCESS NAME=
OPTIONS= XREF
MACRO LIBRARY PATHNAME=

LINE	KEY	NAME
0002	A	VERSION

=>PPC2.P359.SRC.P359

```
0031                    IDT  'NUDS'
0032          ********************************************************************
0033          *
0034          *          N      N   U      U   DDDDDD      SSSSS              *
0035          *          NN     N   U      U   D     D     S     S            *
0036          *          N N    N   U      U   D     D     S                  *
0037          *          N  N   N   U      U   D     D      SSSSS             *
0038          *          N   N  N   U      U   D     D           S            *
0039          *          N    NN   U      U   D     D     S     S            *
0040          *          N     N    UUUUU    DDDDDD      SSSSS              *
0041          *                                                              *
0042          *          PPPP      3333     555555     9999                 *
0043          *          P   P    3    3    5         9    9                 *
0044          *          P   P         3    5         9    9                 *
0045          *          PPPP       3333    55555      99999                *
0046          *          P             3         5         9                 *
0047          *          P         3   3    5    5    9    9                 *
0048          *          P          3333     5555      9999                  *
0049          *                                                              *
0050          ********************************************************************
```

```
0055                        DEF   NABS,NATN,NCOS,NEXP,NINT,NLOG,NSGN,NSIN
0056                        DEF   NSQR,NTAN,LEXP
0057                        DEF   NLPR,NMINUS,NPLUS,O.AND,O.OR,O.XOR,O.NOT
0058                        DEF   CBH63
0059 0000
0060                        REF   ATN$$,COS$$,EXP$$,LOG$$,SIN$$
0061                        REF   SQR$$,TAN$$,GRINT,PWR$$
0062                        REF   LTRUE,LTST90,ERR,PARSE,CONT,WARN$$
0063                        REF   FAC,FAC2,FAC10,ARG,STREND,VSPTR,STND12
0064                        REF   VPSH23,COMPCT,ERR1,ERRT,PGMCHR
0065                        REF   CIF,CFI,PSHPRS,ARGTST,VPOP
0066                        REF   SAVREG,SETREG,ERRSO,GOTO90
0067                        REF   PAGE1,PAGE2
0068 0000
0069       00B6  RPAR$   EQU   >B6
0070       00B7  LPAR$   EQU   >B7
0071       00BA  OR$     EQU   >BA
0072       00BB  AND$    EQU   >BB
0073       00BC  XOR$    EQU   >BC
0074       00BD  NOT$    EQU   >BD
0075       00C1  PLUS$   EQU   >C1
0076       00C2  MINUS$  EQU   >C2
0077       00C5  EXPON$  EQU   >C5
0078       00CB  ABS$    EQU   >CB
0079       00D1  SGN$    EQU   >D1
```

```
0082                    *
0083 0000
0084 0000 9820  LEXP    CB    @FAC2,@CBH63      Must have a numeric
     0002 0000
     0004 0023'
0085 0006 1B39          JH    ERRSNM            Don't - so error
0086 0008 06A0          BL    @PSHPRS           Push 1st and parse 2nd
     000A 0000
0087 000C   C5          BYTE  EXPON$,0          Up to another expon or less
     000D   00
0088 000E 06A0          BL    @STKCHK           Make sure room on stack
     0010 00DE'
0089 0012 0202          LI    R2,PWR$$          Address of power routine
     0014 0000
0090 0016 1049          JMP   COMM05            Jump into common routine
```

```
0093                    *                       ABS
0094 0018
0095 0018-0288  NABS    CI    R8,LPAR$*256      Must have a left paren
     001A B700
0096 001C 1630          JNE   ERRSYN            If not - error
0097 001E 06A0          BL    @PARSE            Parse the argument
     0020 0000
0098 0022   CB          BYTE  ABS$              Up to another ABS
0099 0023   63  CBH63   BYTE  >63               Use the wasted byte
0100 0024 9820          CB    @FAC2,@CBH63      Must have numeric arg
     0026 0002'
     0028 0023'
0101 002A 1B27          JH    ERRSNM            If not - error
0102 002C 0760          ABS   @FAC              Take the absolute value
     002E 0000
0103 0030 0460  BCONT   B     @CONT             And continue
     0032 0000
0104 0034
0105                    *                       ATN
0106 0034
0107 0034 0202  NATN    LI    R2,ATN$$          Load up arctan address
     0036 0000
0108 0038 102C          JMP   COMMON            Jump into common routine
0109 003A
0110                    *                       COS
0111 003A
0112 003A 0202  NCOS    LI    R2,COS$$          Load up cosine address
     003C 0000
0113 003E 1029          JMP   COMMON            Jump into common routine
0114 0040
0115                    *                       EXP
0116 0040
0117 0040 0202  NEXP    LI    R2,EXP$$          Load up exponential address
     0042 0000
0118 0044 1026          JMP   COMMON            Jump into common routine
0119 0046
0120                    *                       INT
0121 0046
0122 0046 0202  NINT    LI    R2,GRINT          Load up greatest integer addr
     0048 0000
0123 004A 1023          JMP   COMMON            Jump to common routine
0124 004C
0125                    *                       LOG
0126 004C
0127 004C 0202  NLOG    LI    R2,LOG$$          Load up logarithm code
     004E 0000
0128 0050 1020          JMP   COMMON            Jump to common routine
0129 0052
0130                    *                       SGN
0131 0052
0132 0052-0288  NSGN    CI    R8,LPAR$*256      Must have left paren
     0054 B700
0133 0056 1613          JNE   ERRSYN            If not - error
0134 0058 06A0          BL    @PARSE            Parse the argument
     005A 0020'
0135 005C   D1          BYTE  SGN$,0            Up to another SGN
     005D   00
0136 005E 9820          CB    @FAC2,@CBH63      Must have a numeric arg
     0060 0026'
     0062 0023'
```

```
0137 0064 1B0A          JH    ERRSNM          If not - error
0138 0066 0204          LI    R4,>4001        Floating point one
     0068 4001
0139 006A C020          MOV   @FAC,R0         Check status
     006C 002E
0140 006E 13E0          JEQ   BCONT           If 0 - return 0
0141 0070 1502          JGT   BLTST9          If positive - return +1
0142 0072 0460          B     @LTRUE          If negative - return -1
     0074 0000
0143 0076 0460  BLTST9  B     @LTST90         Sets up the FAC w/R4 and 0s
     0078 0000
0144 007A
0145 007A 0460  ERRSNM  B     @ERRT           STRING-NUMBER MISMATCH
     007C 0000
0146 007E 0460  ERRSYN  B     @ERR1           SYNTAX ERROR
     0080 0000
0147 0082
0148            *                       SIN
0149 0082
0150 0082 0202  NSIN    LI    R2,SIN$$        Load up sine address
     0084 0000
0151 0086 1005          JMP   COMMON          Jump into common routine
0152 0088
0153            *                       SQR
0154 0088
0155 0088 0202  NSQR    LI    R2,SQR$$        Load up square-root address
     008A 0000
0156 008C 1002          JMP   COMMON          Jump into common routine
0157 008E
0158            *                       TAN
0159 008E
0160 008E 0202  NTAN    LI    R2,TAN$$        Load up tangent address
     0090 0000
```

```
0163 0092 06A0  COMMON BL   @STKCHK          Make sure room on stacks
     0094 00DE'
0164 0096-0288         CI   R8,LPAR$*256      Must have left paren
     0098 B700
0165 009A 16F1         JNE  ERRSYN            If not - error
0166 009C 05C9         INCT R9                Get space on subr stack
0167 009E C642         MOV  R2,*R9            Put addr of routine on stack
0168 00A0 06A0         BL   @PARSE            Parse the argument
     00A2 005A'
0169 00A4   FF         BYTE >FF,0             To end of the arg
     00A5   00
0170 00A6 C099         MOV  *R9,R2            Get address of function back
0171 00A8 0649         DECT R9                Decrement subr stack
0172 00AA 9820  COMM05 CB   @FAC2,@CBH63      Must have a numeric arg
     00AC 0060'
     00AE 0023'
0173 00B0 1BE4         JH   ERRSNM            If not - error
0174 00B2 04E0         CLR  @FAC10            Assume no error or warning
     00B4 0000
0175 00B6 06A0         BL   @SAVREG           Save BASIC registers
     00B8 0000
0176 00BA C802         MOV  R2,@PAGE2         Select Page 2
     00BC 0000
0177 00BE 0692         BL   *R2               Evaluate the function
0178 00C0 C802         MOV  R2,@PAGE1         Reselect Page 1
     00C2 0000
0179 00C4 06A0         BL   @SETREG           Set registers up again
     00C6 0000
0180 00C8 D020         MOVB @FAC10,R0         Check for error or warning
     00CA 00B4'
0181 00CC 13B1         JEQ  BCONT             If not error - continue
0182 00CE 0990         SRL  R0,9              Check for warning
0183 00D0 1304         JEQ  WARN              Warning - issue it
0184 00D2 0200         LI   R0,>0803          BAD ARGUMENT code
     00D4 0803
0185 00D6 0460         B    @ERR              Issue the error message
     00D8 0000
0186 00DA
0187 00DA 0460  WARN   B    @WARN$$           Issue the warning message
     00DC 0000
```

```
0190 00DE 0289  STKCHK CI    R9,STND12          Enough room on the subr stack?
     00E0 0000
0191 00E2 1B18         JH    BSO                No - Mem Full error
0192 00E4 C020         MOV   @VSPTR,R0          Get the value stack pointer
     00E6 0000
0193 00E8 0220         AI    R0,48              Buffer-zone of 48 bytes
     00EA 0030
0194 00EC 8800         C     R0,@STREND         Room between stack & strings
     00EE 0000
0195 00F0 1A0E         JL    STKRTN             Yes - return
0196 00F2 05C9         INCT  R9                 Get space on subr stack
0197 00F4 CE4B         MOV   R11,*R9+           Save return address
0198 00F6 CE42         MOV   R2,*R9+            Save COMMON function code
0199 00F8 C640         MOV   R0,*R9             Save v-stack pointer+48
0200 00FA 06A0         BL    @COMPCT            Do a garbage collection
     00FC 0000
0201 00FE 8819         C     *R9,@STREND        Enough space now?
     0100 00EE'
0202 0102 1406         JHE   BMF                No - Mem Full error
0203 0104 0649         DECT  R9                 Decrement stack pointer
0204 0106 C099         MOV   *R9,R2             Restore COMMON function code
0205 0108 0649         DECT  R9                 Decrement stack pointer
0206 010A C2D9  RETURN MOV   *R9,R11            Restore return address
0207 010C 0649         DECT  R9                 Decrement stack pointer
0208 010E 045B  STKRTN RT                       and return to caller
0209 0110
0210 0110 0460  BMF    B     @VPSH23            * MEMORY FULL
     0112 0000
0211 0114 0460  BSO    B     @ERRSO             * STACK OVERFLOW
     0116 0000
```

```
0214                ********************************************************************
0215                *       LED routines for AND, OR ,NOT and XOR
0216                ********************************************************************
0217 0118
0218 0118 06A0  O.AND   BL    @PSHPRS          Push L.H. and PARSE R.H.
     011A 000A'
0219 011C   BB          BYTE  AND$,0           Stop on AND or less
     011D   00
0220 011E 06A0          BL    @CONVRT          Convert both to integers
     0120 01BC'
0221 0122 0560          INV   @FAC             Complement L.H.
     0124 006C'
0222 0126 4820          SZC   @FAC,@ARG        Perform the AND
     0128 0124'
     012A 0000
0223 012C C820  O.AND1  MOV   @ARG,@FAC        Put back in FAC
     012E 012A'
     0130 0128'
0224 0132 06A0  O.AND2  BL    @CIF             Convert back to floating
     0134 0000
0225 0136 0460          B     @CONT            Continue
     0138 0032'
0226 013A
0227 013A
0228 013A 06A0  O.OR    BL    @PSHPRS          Push L.H. and PARSE R.H.
     013C 011A'
0229 013E   BA          BYTE  OR$,0            Stop on OR or less
     013F   00
0230 0140 06A0          BL    @CONVRT          Convert both to integers
     0142 01BC'
0231 0144 E820          SOC   @FAC,@ARG        Perform the OR
     0146 0130'
     0148 012E'
0232 014A 10F0          JMP   O.AND1           Convert to floating and done
0233 014C
0234 014C
0235 014C 06A0  O.NOT   BL    @PARSE           Parse the arg
     014E 00A2'
0236 0150   BD          BYTE  NOT$,0           Stop on NOT or less
     0151   00
0237 0152 9820          CB    @FAC2,@CBH63     Get a numeric back?
     0154 00AC'
     0156 0023'
0238 0158 1B49          JH    ERRSN1           No => error
0239 015A 04E0          CLR   @FAC10           Clear for CFI
     015C 00CA'
0240 015E 06A0          BL    @CFI             Convert to Integer
     0160 0000
0241 0162 D020          MOVB  @FAC10,R0        Check for an error
     0164 015C'
0242 0166 168B          JNE   ERRSYN           Error
0243 0168 0560          INV   @FAC             Perform the NOT
     016A 0146'
0244 016C 10E2          JMP   O.AND2           Convert to float and done
0245 016E
0246 016E
0247 016E 06A0  O.XOR   BL    @PSHPRS          Push L.H. and PARSE R.H.
     0170 013C'
0248 0172   BC          BYTE  XOR$,0           Stop on XOR or less
     0173   00
```

```
0249 0174 06A0          BL    @CONVRT           Convert both to integer
     0176 01BC'
0250 0178 C020          MOV   @ARG,RO           Get R.H. into a register
     017A 0148'
0251 017C 2820          XOR   @FAC,RO           Do the XOR
     017E 016A'
0252 0180 C800          MOV   RO,@FAC           Put result back in FAC
     0182 017E'
0253 0184 10D6          JMP   O.AND2            Convert and continue
```

```
0256                    ***************************************************************
0257                    *                  NUD for left parenthesis                   *
0258                    ***************************************************************
0259 0186-0288  NLPR    CI    R8,RPAR$*256        Have a right paren already?
     0188 B600
0260 018A 1332          JEQ   ERRSY1              If so - syntax error
0261 018C 06A0          BL    @PARSE              Parse insid the parentheses
     018E 014E'
0262 0190   B7          BYTE  LPAR$,0             Up to left paren or less
     0191   00
0263 0192-0288          CI    R8,RPAR$*256        Have a right paren now?
     0194 B600
0264 0196 162C          JNE   ERRSY1              No - so error
0265 0198 06A0          BL    @PGMCHR             Get next token
     019A 0000
0266 019C 0460  BCON1   B     @CONT               And continue
     019E 0138'
0267 01A0
0268 01A0
0269                    ***************************************************************
0270                    *                    NUD for unary minus                      *
0271                    ***************************************************************
0272 01A0 06A0  NMINUS  BL    @PARSE              Parse the expression
     01A2 018E'
0273 01A4   C2          BYTE  MINUS$,0            Up to another minus
     01A5   00
0274 01A6 0520          NEG   @FAC                Make it negative
     01A8 0182'
0275 01AA 9820  NMIN10  CB    @FAC2,@CBH63        Must have a numeric
     01AC 0154'
     01AE 0023'
0276 01B0 1B1D          JH    ERRSN1              If not - error
0277 01B2 10F4          JMP   BCON1               Continue
0278 01B4
0279 01B4
0280                    ***************************************************************
0281                    *                     NUD for unary plus                      *
0282                    ***************************************************************
0283 01B4 06A0  NPLUS   BL    @PARSE              Parse the expression
     01B6 01A2'
0284 01B8   C1          BYTE  PLUS$,0
     01B9   00
0285 01BA 10F7          JMP   NMIN10              Use common code
```

```
0288                    *
0289                    ***************************************************************
0290                    *
0291                    *       CONVRT - Takes two arguments, 1 from FAC and 1 from
0292                    *            the top of the stack and converts them to
0293                    *            integer from floating point, issuing appropriate
0294                    *            errors
0295                    *
0296                    ***************************************************************
0297 01BC              CONVRT
0298 01BC 05C9                 INCT R9
0299 01BE C64B                 MOV  R11,*R9            SAVE RTN ADDR
0300 01C0 06A0                 BL   @ARGTST            ARGS MUST BE SAME TYPE
     01C2 0000
0301 01C4 1313                 JEQ  ERRSN1             AND NON-STRING
0302 01C6 04E0                 CLR  @FAC10             FOR CFI ERROR CODE
     01C8 0164'
0303 01CA 06A0                 BL   @CFI               CONVERT R.H. ARG
     01CC 0160'
0304 01CE D020                 MOVB @FAC10,R0          ANY ERROR OR WARNING?
     01D0 01C8'
0305 01D2 160A                 JNE  ERRBV              YES
0306 01D4 C820                 MOV  @FAC,@ARG          MOVE TO GET L.H. ARG
     01D6 01A8'
     01D8 017A'
0307 01DA 06A0                 BL   @VPOP              GET L.H. BACK
     01DC 0000
0308 01DE 06A0                 BL   @CFI               CONVERT L.H.
     01E0 01CC'
0309 01E2 D020                 MOVB @FAC10,R0          ANY ERROR OR WARNING?
     01E4 01D0'
0310 01E6 1391                 JEQ  RETURN             No-get rtn off stack and rtn
0311                    *                              Yes-issue error
0312 01E8 0460         ERRBV   B    @GOTO90            BAD VALUE
     01EA 0000
0313 01EC 0460         ERRSN1  B    @ERRT              STRING NUMBER MISMATCH
     01EE 007C'
0314 01F0 0460         ERRSY1  B    @ERR1              SYNTAX ERROR
     01F2 0080'
0315                           END
NO ERRORS,    0005 WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES							
ABS$		00CB	0078	0098							
AND$		00BB	0072	0219							
ARG	R	01D8'	0063	0222	0223	0231	0250	0306			
ARGTST	R	01C2'	0065	0300							
ATN$$	R	0036'	0060	0107							
BCON1		019C'	0266	0277							
BCONT		0030'	0103	0140	0181						
BLTST9		0076'	0143	0141							
BMF		0110'	0210	0202							
BSO		0114'	0211	0191							
CBH63	D	0023'	0099	0058	0084	0100	0136	0172	0237	0275	
CFI	R	01E0'	0065	0240	0303	0308					
CIF	R	0134'	0065	0224							
COMMO5		00AA'	0172	0090							
COMMON		0092'	0163	0108	0113	0118	0123	0128	0151	0156	
COMPCT	R	00FC'	0064	0200							
CONT	R	019E'	0062	0103	0225	0266					
CONVRT		01BC'	0297	0220	0230	0249					
COS$$	R	003C'	0060	0112							
DX10		0001	0003	0004							
ERR	R	00D8'	0062	0185							
ERR1	R	01F2'	0064	0146	0314						
ERRBV		01E8'	0312	0305							
ERRSN1		01EC'	0313	0238	0276	0301					
ERRSNM		007A'	0145	0085	0101	0137	0173				
ERRSO	R	0116'	0066	0211							
ERRSY1		01F0'	0314	0260	0264						
ERRSYN		007E'	0146	0096	0133	0165	0242				
ERRT	R	01EE'	0064	0145	0313						
EXP$$	R	0042'	0060	0117							
EXPON$		00C5	0077	0087							
FAC	R	01D6'	0063	0102	0139	0221	0222	0223	0231	0243	0251	0252
				0274	0306							
FAC10	R	01E4'	0063	0174	0180	0239	0241	0302	0304	0309		
FAC2	R	01AC'	0063	0084	0100	0136	0172	0237	0275			
GOTO90	R	01EA'	0066	0312								
GRINT	R	0048'	0061	0122								
LEXP	D	0000'	0084	0056								
LOG$$	R	004E'	0060	0127								
LPAR$		00B7	0070	0095	0132	0164	0262					
LTRUE	R	0074'	0062	0142								
LTST90	R	0078'	0062	0143								
MINUS$		00C2	0076	0273								
NABS	D	0018'	0095	0055								
NATN	D	0034'	0107	0055								
NCOS	D	003A'	0112	0055								
NEXP	D	0040'	0117	0055								
NINT	D	0046'	0122	0055								
NLOG	D	004C'	0127	0055								
NLPR	D	0186'	0259	0057								
NMIN10		01AA'	0275	0285								
NMINUS	D	01A0'	0272	0057								
NOT$		00BD	0074	0236								
NPLUS	D	01B4'	0283	0057								
NSGN	D	0052'	0132	0055								
NSIN	D	0082'	0150	0055								
NSQR	D	0088'	0155	0056								
NTAN	D	008E'	0160	0056								
O.AND	D	0118'	0218	0057								
O.AND1		012C'	0223	0232								

LABEL		VALUE	DEFN	REFERENCES								
O.AND2		0132'	0224	0244	0253							
O.NOT	D	014C'	0235	0057								
O.OR	D	013A'	0228	0057								
O.XOR	D	016E'	0247	0057								
OR$		00BA	0071	0229								
P359		0000	0003	0003								
PAGE1	R	00C2'	0067	0178								
PAGE2	R	00BC'	0067	0176								
PARSE	R	01B6'	0062	0097	0134	0168	0235	0261	0272	0283		
PGMCHR	R	019A'	0064	0265								
PLUS$		00C1	0075	0284								
PSHPRS	R	0170'	0065	0086	0218	0228	0247					
PWR$$	R	0014'	0061	0089								
R0		0000		0139	0180	0182	0184	0192	0193	0194	0199	0241
				0250	0251	0252	0304	0309				
R11		000B		0197	0206	0299						
R2		0002		0089	0107	0112	0117	0122	0127	0150	0155	0160
				0167	0170	0176	0177	0178	0198	0204		
R4		0004		0138								
R8		0008		0095	0132	0164	0259	0263				
R9		0009		0166	0167	0170	0171	0190	0196	0197	0198	0199
				0201	0203	0204	0205	0206	0207	0298	0299	
RETURN		010A'	0206	0310								
RPAR$		00B6	0069	0259	0263							
SAVREG	R	00B8'	0066	0175								
SETREG	R	00C6'	0066	0179								
SGN$		00D1	0079	0135								
SIN$$	R	0084'	0060	0150								
SQR$$	R	008A'	0061	0155								
STKCHK		00DE'	0190	0088	0163							
STKRTN		010E'	0208	0195								
STND12	R	00E0'	0063	0190								
STREND	R	0100'	0063	0194	0201							
TAN$$	R	0090'	0061	0160								
VERMAC	M		A0001	0003								
VERS		0000	0003	0004								
VPOP	R	01DC'	0065	0307								
VPSH23	R	0112'	0064	0210								
VSPTR	R	00E6'	0063	0192								
WARN		00DA'	0187	0183								
WARN$$	R	00DC'	0062	0187								
XOR$		00BC	0073	0248								