```
-- file: OpNames.mesa
-- edited by Sandman, July 17, 1977  8:49 PM

DIRECTORY
  AltoDefs: FROM "altodefs",
  InlineDefs: FROM "inlinedefs",
  ListerDefs: FROM "listerdefs",
  SegmentDefs: FROM "segmentdefs",
  StreamDefs: FROM "streamdefs",
  StringDefs: FROM "stringdefs";

DEFINITIONS FROM AltoDefs;

OpNames: PROGRAM
  IMPORTS SegmentDefs, StreamDefs, StringDefs  EXPORTS ListerDefs =
BEGIN

--  mnemonic: ARRAY BYTE OF STRING ← [
--    "NOOP", "LABEL", "LFC", "NOOP", "NOOP", "NOOP", "NOOP",
--    "LG0", "LG1", "LG2", "LG3", "LG4", "LG5", "LG6", "LG7",
--    "LGB", "LGS", "SG0", "SG1", "SG2", "SG3", "SGB", "SGS",
--    "LL0", "LL1", "LL2", "LL3", "LL4", "LL5", "LL6", "LL7",
--    "LLB", "LLS", "SL0", "SL1", "SL2", "SL3", "SL4", "SL5",
--    "SL6", "SL7", "SLB", "SLS", "LI0", "LI1", "LI2", "LI3",
--    "LI4", "LI5", "LI6", "LIN1", "LIB", "LIW", "LINB", "NOOP",
--    "LGDB", "LGDS", "SGDB", "SGDS", "LLDB", "LLDS", "SLDB", "SLDS",
--    "R0", "R1", "R2", "R3", "R4", "RB", "W0", "W1",
--    "W2", "WB", "RF", "WF", "RDB", "RD0", "WDB", "WD0",
--    "RSTR", "WSTR", "RXL0", "RXL1", "RXL2", "RXL3", "WXL0", "RIG0",
--    "RIG1", "RIG2", "RIG3", "WIG0", "RIL0", "RIL1", "RIL2", "RIL3",
--    "WIL0", "WS0", "WSB", "WSF", "WSDB", "WSDS", "NOOP", "NOOP",
--    "J1E", "J2E", "J3E", "J4E", "J1O", "J2O", "J3O", "J4O",
--    "JBE", "JBO", "JWE", "JWO", "NJBE", "NJBO", "JEQ1E", "JEQ2E",
--    "JEQ3E", "JEQ4E", "JEQ1O", "JEQ2O", "JEQ3O", "JEQ4O", "JEQBE", "JEQBO",
--    "JNE1E", "JNE2E", "JNE3E", "JNE4E", "JNE1O", "JNE2O", "JNE3O", "JNE4O",
--    "JNEBE", "JNEBO", "JLBE", "JLBO", "JGEBE", "JGEBO", "JGBE", "JGBO",
--    "JLEBE", "JLEBO", "JULBE", "JULBO", "JUGEBE", "JUGEBO", "JUGBE", "JUGBO",
--    "JULEBE", "JULEBO", "JZEQBE", "JZEQBO", "JZNEBE", "JZNEBO", "JDEQBE", "JDEQBO",
--    "JDNEBE", "JDNEBO", "JIB", "JIW", "NOOP", "NOOP", "NOOP", "NOOP",
--    "ADD", "SUB", "MUL", "DBL", "DIV", "LDIV", "NEG", "INC",
--    "AND", "OR", "XOR", "SHIFT", "UCOMP", "NOOP", "NOOP", "NOOP",
--    "PUSH", "POP", "EXCH", "PUSHX", "DUP", "NOOP", "NOOP", "NOOP",
--    "GFC0", "GFC1", "GFC2", "GFC3", "GFC4", "GFC5", "GFC6", "GFC7",
--    "GFC8", "GFC9", "GFC10", "GFC11", "GFC12", "GFC13", "GFC14", "GFC15",
--    "GFCB", "LFC1", "LFC2", "LFC3", "LFC4", "LFC5", "LFC6", "LFC7",
--    "LFC8", "LFC9", "LFC10", "LFC11", "LFC12", "LFC13", "LFC14", "LFC15",
--    "LFC16", "LFCB", SFC", "RET", "NOOP", "PORTO", "PORTI", "KFCB",
--    "ADDL", "ADDG", "BLT", "ALLOC", "FREE", "IWDC", "DWDC", "NOOP",
--    "STOP", "CATCH", "CONVERT", "BLOCK", "BITBLT", "STARTIO", "NOOP", "NOOP",
--    "DST", "LST", "LSTF", "NOOP", "WR", "RR", "BRK", "NOOP"];

mnemonic: DESCRIPTOR FOR ARRAY OF STRING ← DESCRIPTOR[NIL,0];
numberofinstructions: CARDINAL = 256;

instname: PUBLIC PROCEDURE [op: BYTE] RETURNS [STRING] =
  BEGIN
  IF BASE[mnemonic] = NIL THEN SetupArray[];
  RETURN [mnemonic[op]]
  END;

UnknownInstruction: PUBLIC SIGNAL[name: STRING] = CODE;

instcode: PUBLIC PROCEDURE [name: STRING] RETURNS [i: BYTE] =
  BEGIN
  IF BASE[mnemonic] = NIL THEN SetupArray[];
  FOR i IN BYTE DO
    IF StringDefs.EquivalentString[name,mnemonic[i]] THEN RETURN;
    ENDLOOP;
  SIGNAL UnknownInstruction[name];
  RETURN[0];
  END;

BadFormat: ERROR = CODE;
Semicolon: SIGNAL = CODE;
binaryfile: STRING ← "OpNames.binary";
arraysegment: SegmentDefs.FileSegmentHandle ← NIL;
```

```
gennamefile: PROCEDURE =
  BEGIN OPEN SegmentDefs, InlineDefs, StreamDefs, StringDefs;
  a: ARRAY [0..numberofinstructions) OF CARDINAL;
  i: CARDINAL;
  pos: CARDINAL ← numberofinstructions;
  s: STRING ← [40];
  l: CARDINAL;
  in, out: StreamHandle;
  end: StreamIndex;
  getid: PROCEDURE =
    BEGIN
    leading: BOOLEAN ← TRUE;
    c: CHARACTER;
    s.length ← 0;
    DO
      IF in.endof[in] THEN ERROR BadFormat;
      SELECT c←in.get[in] FROM
        IN ['A..'Z], IN ['a..'z] =>
          BEGIN AppendChar[s,c]; leading ← FALSE END;
        IN ['0..'9] =>
          IF ~leading THEN AppendChar[s,c];
        '; => SIGNAL Semicolon;
        ENDCASE => IF ~leading THEN RETURN;
      ENDLOOP;
    END;

  in ← CreateByteStream[NewFile["OpNames.mesa",Read,DefaultVersion],Read];
  out ← CreateWordStream[NewFile[binaryfile,Write+Append,DefaultVersion],Write+Append];
  SetIndex[out,StreamIndex[0,LENGTH[a]*2]];
  UNTIL EqualString[s,"mnemonic"] DO getid[!Semicolon=>RESUME] ENDLOOP;
  WHILE in.get[in] # '[ DO NULL ENDLOOP;
  FOR i IN[0..numberofinstructions) DO
    getid[!Semicolon=>ERROR BadFormat];
    IF BITAND[l←s.length,1] # 0 THEN AppendChar[s,0C];
    a[i] ← pos;
    out.put[out,l]; out.put[out,l];
    pos ← pos + WriteBlock[out,s+2,s.length/2] + 2;
    ENDLOOP;
  in.destroy[in];
  end ← GetIndex[out];
  out.reset[out];
  [] ← WriteBlock[out,BASE[a],numberofinstructions];
  SetIndex[out,end];
  out.destroy[out];
  END;

SetupArray: PUBLIC PROCEDURE =
  BEGIN OPEN SegmentDefs;
  i: BYTE;
  base: POINTER;
  IF arraysegment = NIL THEN
    BEGIN
    arraysegment ←
      NewFileSegment[NewFile[binaryfile,Read,DefaultVersion],
        DefaultBase, DefaultPages, Read];
    SwapIn[arraysegment];
    SP.proc ← CantSwap;
    -- AddSwapStrategy[@SP];
    END
  ELSE SwapIn[arraysegment];
  base ← FileSegmentAddress[arraysegment];
  mnemonic ← DESCRIPTOR[base, numberofinstructions];
  FOR i IN[0..LENGTH[mnemonic]) DO
    mnemonic[i] ← mnemonic[i]+LOOPHOLE[base];
    ENDLOOP;
  SP.proc ← DeleteArray;
  END;

SP: SegmentDefs.SwapStrategy;

DeleteArray: PUBLIC SegmentDefs.SwappingProcedure =
  BEGIN
  SP.proc ← SegmentDefs.CantSwap;
  IF BASE[mnemonic] = NIL THEN RETURN[FALSE];
  SegmentDefs.Unlock[arraysegment];
```

```
  SegmentDefs.SwapOut[arraysegment];
  mnemonic ← DESCRIPTOR[NIL,0];
  RETURN [TRUE];
  END;

gennamefile[];

END.
```