```
-- file: OpNames.mesa
-- edited by Johnsson, May 1, 1978  8:44 AM

DIRECTORY
  AltoDefs: FROM "altodefs",
  BinaryDefs: FROM "binarydefs",
  MiscDefs: FROM "miscdefs",
  OpTableDefs: FROM "optabledefs",
  SegmentDefs: FROM "segmentdefs",
  StringDefs: FROM "stringdefs";

OpNames: PROGRAM
  IMPORTS BinaryDefs, MiscDefs, SegmentDefs, StringDefs  EXPORTS OpTableDefs =
  BEGIN

  BYTE: TYPE = AltoDefs.BYTE;

  CompStringDesc: TYPE = RECORD [offset, length: CARDINAL];

  Names: POINTER TO RECORD [
    base: STRING,
    mnemonic: ARRAY BYTE OF CompStringDesc] ← NIL;

  s: STRING ← [16];
  ss: StringDefs.SubStringDescriptor;

  InstName: PUBLIC PROCEDURE [op: BYTE] RETURNS [STRING] =
    BEGIN
    SetupArray[];
    [ss.offset,ss.length] ← Names.mnemonic[op];
    s.length ← 0;
    StringDefs.AppendSubString[s,@ss];
    ReleaseArray[];
    RETURN[s]
    END;

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

  InstCode: PUBLIC PROCEDURE [name: STRING] RETURNS [i: BYTE] =
    BEGIN
    s: StringDefs.SubString ← @ss;
    ssname: StringDefs.SubStringDescriptor ← [name,0,name.length];
    sname: StringDefs.SubString ← @ssname;
    SetupArray[];
    FOR i IN BYTE DO
      [s.offset,s.length] ← Names.mnemonic[i];
      IF StringDefs.EquivalentSubStrings[s,sname] THEN EXIT;
      REPEAT FINISHED =>
        BEGIN i ← 0; SIGNAL UnknownInstruction[name] END;
      ENDLOOP;
    ReleaseArray[];
    RETURN;
    END;

  arraysegment: SegmentDefs.FileSegmentHandle ←
    MiscDefs.DestroyFakeModule[LOOPHOLE[BinaryDefs.MopcodeNames]].seg;

  SetupArray: PROCEDURE =
    BEGIN OPEN SegmentDefs;
    SwapIn[arraysegment];
    Names ← FileSegmentAddress[arraysegment];
    ss.base ← Names.base + LOOPHOLE[Names,CARDINAL];
    END;

  ReleaseArray: PROCEDURE =
    BEGIN OPEN SegmentDefs;
    Unlock[arraysegment];
    END;

  END.
```