```
-- file SymDefs.Mesa
-- last modified by Satterthwaite, October 3, 1977  2:04 PM

DIRECTORY
  AltoDefs: FROM "altodefs",
  BcdDefs: FROM "bcddefs",
  TableDefs: FROM "tabledefs";

SymDefs: DEFINITIONS =
  BEGIN

  VersionID: CARDINAL = 08287;

 -- symbol segment headers

  FileOffset: TYPE = CARDINAL;

  STHeader: TYPE = RECORD [
    versionIdent: CARDINAL,
    version: BcdDefs.VersionStamp,
    creator: BcdDefs.VersionStamp,
    definitionsFile: BOOLEAN,
    directoryCtx: CTXIndex,
    importCtx: CTXIndex,
    outerCtx: CTXIndex,
    hvOffset: FileOffset,
    hvSize: CARDINAL,
    htOffset: FileOffset,
    htSize: CARDINAL,
    ssOffset: FileOffset,
    ssSize: CARDINAL,
    seOffset: FileOffset,
    seSize: CARDINAL,
    ctxOffset: FileOffset,
    ctxSize: CARDINAL,
    mdOffset: FileOffset,
    mdSize: CARDINAL,
    bodyOffset: FileOffset,
    bodySize: CARDINAL,
    fgRelPgBase: CARDINAL,
    fgPgCount: AltoDefs.PageCount];


 -- hash table declarations

  HVLength: PRIVATE INTEGER = 71;
  HVIndex: TYPE = INTEGER [0..HVLength);


  HTRecord: TYPE = RECORD [
    anyInternal, anyPublic: BOOLEAN,
    link: HTIndex,
    ssIndex: CARDINAL];

  HTIndex: TYPE = INTEGER [0..TableDefs.TableLimit/2);

  HTNull: HTIndex = LAST[HTIndex];


 -- semantic entry table declarations

  TypeClass: TYPE = {
    mode,
    basic,
    enumerated,
    record,
    pointer,
    array,
    arraydesc,
    transfer,
    definition,
    union,
    subrange
    };

  TransferMode: TYPE = {procedure, port, signal, error, program, inline, none};
```

```
SERecord: TYPE = RECORD [
  mark3, mark4: BOOLEAN,
  sebody: SELECT setag: * FROM
    id => [
      public, readonly: BOOLEAN,
      ctxnum: CTXIndex,
      writeonce, constant: BOOLEAN,
      idtype: SEIndex,
      idinfo: UNSPECIFIED,
      idvalue: UNSPECIFIED,
      htptr: HTIndex,
      external: BOOLEAN,
      ctxlink: SELECT linktag: * FROM
        terminal => NULL,
        sequential => NULL,
        linked => [link: ISEIndex],
        ENDCASE],
    constructor => [
      typeinfo: SELECT typetag: TypeClass FROM
        mode => NULL,
        basic => [
          ordered: BOOLEAN,
          code: [0..16),
          length: CARDINAL],
        enumerated => [
          ordered: BOOLEAN,
          valuectx: CTXIndex,
          nvalues: CARDINAL],
        record => [
          machineDep: BOOLEAN,
          unifield: BOOLEAN,
          length: CARDINAL,
          fieldctx: CTXIndex,
          variant: BOOLEAN,
          linkpart: SELECT linktag: * FROM
            notlinked => NULL,
            linked => [linktype: SEIndex],
            ENDCASE],
        pointer => [
          ordered, readonly: BOOLEAN,
          pointedtotype: SEIndex],
        array, arraydesc => [
          packed: BOOLEAN,
          indextype: SEIndex,
          componenttype: SEIndex],
        transfer => [
          mode: TransferMode,
          inrecord: recordCSEIndex,
          outrecord: recordCSEIndex],
        definition => [
          nGfi: [1 .. 4],
          defCtx: CTXIndex],
        union => [
          equalLengths: BOOLEAN,
          casectx: CTXIndex,
          overlayed: BOOLEAN,
          controlled: BOOLEAN,
          tagsei: ISEIndex],
        subrange => [
          filled, empty, flexible: BOOLEAN,
          rangetype: SEIndex,
          origin: INTEGER,
          range: CARDINAL],
        ENDCASE],
    ENDCASE];

SEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO SERecord;

ISEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO id SERecord;
CSEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO constructor SERecord;
  recordCSEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO record constructor SERecord;
  arrayCSEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO array constructor SERecord;

SENull: SEIndex = LAST[SEIndex];
  ISENull: ISEIndex = LOOPHOLE[SENull];
```

```
    CSENull: CSEIndex = LOOPHOLE[SENull];
      recordCSENull: recordCSEIndex = LOOPHOLE[SENull];
      arrayCSENull: arrayCSEIndex = LOOPHOLE[SENull];

  -- the following two values are guaranteed by the compiler
   typeTYPE: CSEIndex = FIRST[CSEIndex];
   typeANY: CSEIndex = typeTYPE + SIZE[mode constructor SERecord];

  -- codes identifying the basic types (extensible)
   codeANY: CARDINAL = 0;
   codeINTEGER: CARDINAL = 1;
   codeBOOLEAN: CARDINAL = 2;
   codeCHARACTER: CARDINAL = 3;

   bitaddress: TYPE = RECORD[
     wd: [0..AltoDefs.VMLimit/AltoDefs.wordlength],      -- word displacement
     bd: [0..AltoDefs.wordlength)];                      -- bit displacement


  -- context table declarations

   maxcontextlevel: INTEGER = 7;

   ContextLevel: TYPE = [0..maxcontextlevel];
     1Z: ContextLevel = 0;        -- context level of non-frame records
     1G: ContextLevel = 1;        -- context level of global frame
     1L: ContextLevel = 1G+1;     -- context level of outer procedures

   CTXRecord: TYPE = RECORD [
     sn: Sn,      -- for DeSoto
     selist: ISEIndex,
     ctxlevel: ContextLevel,
     extension: SELECT ctxType: * FROM
       simple => [ctxNew: CTXIndex],     -- for DeSoto
       included => [
         ctxchain: includedCTXIndex,
         ctxmodule: MDIndex,
         ctxmap: CTXIndex,
         ctxclosed, ctxcomplete, ctxreset: BOOLEAN],
       imported => [includeLink: includedCTXIndex],
       ENDCASE];

   CTXIndex: TYPE = ORDERED POINTER [0..3777B] TO CTXRecord;
    includedCTXIndex: TYPE = ORDERED POINTER [0..3777B] TO included CTXRecord;

   CTXNull: CTXIndex = LAST[CTXIndex];
     includedCTXNull: includedCTXIndex = LOOPHOLE[CTXNull];


  -- module table declarations

   FileIndex: TYPE = [0..77777B];        -- internal file handle
   nullFileIndex: FileIndex = LAST[FileIndex];

   MDRecord: TYPE = RECORD [
     mdhti: HTIndex,              -- hash entry for file name
     mdctx: includedCTXIndex,     -- context of copied entries
     mdshared: BOOLEAN,           -- overrides PRIVATE, etc.
     mdExported: BOOLEAN,
     mdStamp: BcdDefs.VersionStamp,
     mdFile: FileIndex];          -- associated file

   MDIndex: TYPE = ORDERED POINTER [0..TableDefs.TableLimit) TO MDRecord;
   MDNull: MDIndex = LAST[MDIndex];

   OwnMdi: MDIndex = FIRST[MDIndex];

  -- body table declarations

   BodyRecord: TYPE = RECORD [
     id: ISEIndex,
     iotype: SEIndex,
     entryindex: CARDINAL,
     fileindex: CARDINAL,
     info: bodyinfo,
     localCtx: CTXIndex,
```

```
    extension: SELECT nested: * FROM
      outer => NULL,
      inner => [frameoffset: [0..AltoDefs.VMLimit]],
      ENDCASE];

    bodyinfo: TYPE = RECORD [
      SELECT mark: * FROM
        internal => [
          nestLink: BTIndex,
          bodylink: BTIndex,
          bodytree: --TreeLink-- UNSPECIFIED,
          framesize: CARDINAL],
        external => [
          origin: [0..AltoDefs.VMLimit/2],
          bytes: CARDINAL,
          startindex, indexlength: CARDINAL],
        ENDCASE];

  BTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO BodyRecord;
    innerBTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO inner BodyRecord;
    outerBTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO outer BodyRecord;
  BTNull: BTIndex = LAST[BTIndex];


  -- fine grain table header

  fgHeader: TYPE = RECORD [
    fgoffset: FileOffset,
    fglength: CARDINAL,
    sourcefile: StringBody -- text follows --];

  -- fine grain table declarations

  ByteIndex: TYPE = CARDINAL;

  FGTEntry: TYPE =  RECORD [
    findex: ByteIndex,
    cindex: ByteIndex];


  -- definitions for use by DeSoto

  Sn: TYPE = {snNil, snValid, snInvalid, snIndirect};

  END.
```