```
-- file bootnewcompiler.mesa
-- last modified by Satterthwaite, April 5, 1978  3:33 PM

DIRECTORY
  AltoFileDefs: FROM "altofiledefs",
  BFSDefs: FROM "bfsdefs",
  BinaryDefs: FROM "binarydefs",
  CompilerDefs: FROM "compilerdefs",
  ControlDefs: FROM "controldefs",
  DiskDefs: FROM "diskdefs",
  ImageDefs: FROM "imagedefs",
  MiscDefs: FROM "miscdefs",
  SegmentDefs: FROM "segmentdefs",
  SystemDefs: FROM "systemdefs",
  TimeDefs: FROM "timedefs";

BootNewCompiler: PROGRAM
  IMPORTS BinaryDefs, BFSDefs, CompilerDefs, ImageDefs, MiscDefs, SegmentDefs, SystemDefs, TimeDefs
  SHARES SegmentDefs =
  BEGIN

  FileSegmentHandle: TYPE = SegmentDefs.FileSegmentHandle;

  CollectDiskAddresses: PROCEDURE =
    BEGIN OPEN SystemDefs, SegmentDefs, AltoFileDefs;
    ImageFile: FileHandle =
      LOOPHOLE[REGISTER[ControlDefs.Greg],
        ControlDefs.GlobalFrameHandle].codesegment.file;
    DAs: DESCRIPTOR FOR ARRAY OF vDA;
    maxunknown, maxknown: CARDINAL ← FIRST[CARDINAL];
    minunknown: CARDINAL ← LAST[CARDINAL];
    maxknownDA: vDA;
    DisplayHead: POINTER TO WORD = LOOPHOLE[420B];
    DisplayInterruptWord: POINTER TO WORD = LOOPHOLE[421B];
    saveDisplay, saveDiw: WORD;
    diskrequest: DiskDefs.DiskRequest;
    bufseg, DAseg: DataSegmentHandle;
    FindEnds: PROCEDURE [seg: FileSegmentHandle] RETURNS [BOOLEAN] =
      BEGIN
      WITH s: seg SELECT FROM
        disk =>
          IF s.file = ImageFile AND s.hint.da = eofDA THEN
            BEGIN
            maxunknown ← MAX[maxunknown,s.base];
            minunknown ← MIN[minunknown,s.base];
            END;
        ENDCASE;
      RETURN[FALSE];
      END;
    FindKnown: PROCEDURE [seg: FileSegmentHandle] RETURNS [BOOLEAN] =
      BEGIN
      WITH s: seg SELECT FROM
        disk =>
          IF s.file = ImageFile AND s.hint.da # eofDA AND s.base < minunknown
            AND s.base > maxknown THEN
            BEGIN maxknown ← s.base; maxknownDA ← s.hint.da END;
        ENDCASE;
      RETURN[FALSE];
      END;
    PlugDA: PROCEDURE [seg: FileSegmentHandle] RETURNS [BOOLEAN] =
      BEGIN
      WITH s: seg SELECT FROM
        disk =>
          IF s.file = ImageFile AND s.hint.da = eofDA AND
            s.base IN (maxknown..maxunknown] THEN
            SegmentDefs.SetFileSegmentDA[@s,DAs[s.base]];
        ENDCASE;
      RETURN[FALSE];
      END;

    saveDisplay ← DisplayHead↑;
    saveDiw ← DisplayInterruptWord↑;
    DisplayHead↑ ← DisplayInterruptWord↑ ← 0;
    [] ← EnumerateFileSegments[FindEnds];
    [] ← EnumerateFileSegments[FindKnown];
    bufseg ← NewDataSegment[DefaultBase, 1];
```

```
    DAseg ← NewDataSegment[
      DefaultBase, PagesForWords[maxunknown-maxknown+3]];
    DAs ← DESCRIPTOR[DataSegmentAddress[DAseg]-(maxknown-1),maxunknown+2];
    diskrequest ← DiskDefs.DiskRequest [
      ca: DataSegmentAddress[bufseg],
      fixedCA: TRUE,
      da: @DAs[0],
      fp: @ImageFile.fp,
      firstPage: maxknown,
      lastPage: maxunknown,
      action: ReadD,
      lastAction: ReadD,
      signalCheckError: FALSE,
      option: update[cleanup: BFSDefs.GetNextDA]];
    MiscDefs.SetBlock[@DAs[maxknown-1],fillinDA,maxunknown-maxknown+3];
    DAs[maxknown] ← maxknownDA;
    [] ← BFSDefs.ActOnPages[LOOPHOLE[@diskrequest]];  -- we know it is an Update diskrequest
    [] ← EnumerateFileSegments[PlugDA];
    DeleteDataSegment[DAseg];
    DeleteDataSegment[bufseg];
    DisplayHead↑ ← saveDisplay;
    DisplayInterruptWord↑ ← saveDiw;
    RETURN;
    END;

  parseseg, errorseg: SegmentDefs.FileSegmentHandle;

  time: STRING ← SystemDefs.AllocateHeapString[18];

  TimeDefs.AppendDayTime[time, TimeDefs.UnpackDT[ImageDefs.ImageVersion[].time]];
  time.length ← time.length - 3;

  [parseseg, ] ← MiscDefs.DestroyFakeModule[LOOPHOLE[BinaryDefs.MesaTab]];
  [errorseg, ] ← MiscDefs.DestroyFakeModule[LOOPHOLE[BinaryDefs.ErrorTab]];

  START CompilerDefs.Control[time, REGISTER[ControlDefs.Greg], parseseg, errorseg];

  STOP;

  CollectDiskAddresses[];

  RESTART CompilerDefs.Control;

  END.
```