```
-- File: CoreSwapDefs.mesa
-- Last edited by Johnsson; July 18, 1978  10:35 AM

DIRECTORY
  AltoFileDefs: FROM "altofiledefs" USING [CFA, CFP],
  ControlDefs: FROM "controldefs" USING [
    GlobalFrameHandle, StateVector, SVPointer, WordPC];

CoreSwapDefs: DEFINITIONS =

  BEGIN

  SVPointer: TYPE = ControlDefs.SVPointer;

  ExternalStateVector: TYPE = MACHINE DEPENDENT RECORD [
    state: SVPointer,
    reason: CoreSwapDefs.SwapReason, level: [0..7777B],
    tables: POINTER,
    drumFile: POINTER,
    parameter: POINTER TO CoreSwapDefs.DebugParameter,
    extension: ESVExtension,
    loadstateCFA: AltoFileDefs.CFA,
    lspages: CARDINAL,
    mapLog: LONG POINTER,
    mds: CARDINAL,
    fill: ARRAY [16..19) OF WORD];

  ESVExtension: TYPE = MACHINE DEPENDENT RECORD [
    SELECT OVERLAID * FROM
      loadstate => [
        loadstate: POINTER],
      either => [
        fill: [0..7777B],
        type: DebuggeeType,
        notMesa40: BOOLEAN],
      ENDCASE];

  DebuggeeType: TYPE =
    {mesa40, wideBodyMesa, longMesa, nonSwappingPilot, pilot};

  SwapReason: TYPE = {
    -- handled by debugnub
    proceed,    -- THIS MUST BE FIRST !!
    start,
    call,
    resume,
    quit,
    showscreen,
    kill,

    -- handled by external debugger
    install,
    breakpoint,
    worrybreak,
    uncaughtsignal,
    explicitcall,
    return,
    punt,
    interrupt,
    cleanmaplog
    };

  DebugParameter: TYPE = MACHINE DEPENDENT RECORD [
    string: STRING,
    body: SELECT OVERLAID SwapReason FROM
      uncaughtsignal => [
        msg: UNSPECIFIED,
        signal: UNSPECIFIED],
      return => [
        value: UNSPECIFIED],
      start => [
        frame: ControlDefs.GlobalFrameHandle],
      call => [
        sv: ControlDefs.StateVector],
      ENDCASE];
```

```
  uncaughtsignalDP: TYPE = POINTER TO uncaughtsignal DebugParameter;
  returnDP: TYPE = POINTER TO return DebugParameter;
  startDP: TYPE = POINTER TO start DebugParameter;
  callDP: TYPE = POINTER TO call DebugParameter;

  GetLevel: PROCEDURE RETURNS [INTEGER];
  SetLevel: PROCEDURE [l: INTEGER];
  CoreSwap: PROCEDURE [why: SwapReason, sp: SVPointer];
  CantSwap: SIGNAL;
  CAbort: SIGNAL;

  -- Conditional Breakpoint Stuff

  BBArray: TYPE = RECORD [
    length: CARDINAL,
    blocks: ARRAY [0..0) OF UserBreakBlock];

  BBHandle: TYPE = POINTER TO BBArray;

  UserBreakBlock: TYPE = RECORD [
    frame: ControlDefs.GlobalFrameHandle,
    pc: ControlDefs.WordPC,
    ptrL: POINTER,
    ptrR: POINTER,
    posnL: [0..16),
    posnR: [0..16),
    sizeL: [1..16],
    sizeR: [1..16],
    inst: [0..377B],
    relation: Relations,
    immediateR: BOOLEAN,
    counterL: BOOLEAN,
    localL: BOOLEAN,
    localR: BOOLEAN];

  UBBPointer: TYPE = POINTER TO UserBreakBlock;

  Relations: TYPE = {eq, ne, lt, ge, gt, le};

  -- Punt and Swapping Information

  CFP: TYPE = AltoFileDefs.CFP;

  PuntTable: TYPE = MACHINE DEPENDENT RECORD [
    pDebuggerFP: POINTER TO CFP,
    pCoreFP: POINTER TO CFP,
    puntESV: ExternalStateVector,
    debuggerFP: CFP,
    coreFP: CFP,
    other: UNSPECIFIED];

  PuntInfo: POINTER TO POINTER TO PuntTable = LOOPHOLE[456B];

  END ...
```