

SAND#5241130

11/29/01 Victor R. Yarberry, Sandia National Labs  
04/01/04 VRY - updated to reflect use of ENDBLK  
05/04/06 VRY - Corrected 3D CROSSING\_AREA definition.  
                  Corrected end of block (ENDBLK) definition.  
                  Added implied POLYLINE closure to definition.  
07/20/06 VRY - Added End of Entity (EOE) explanation.  
11/15/10 VRY - Corrected OBJECTS\_CROSSED typo.

The MEM File Format is a hierarchical structure where the primitives are blocks, closed polylines, and circles. It is used as the input file for the 2D Process Visualizer, the 3D Visualizer, and the 3D Modeler.

All block definitions are ordered such that there are no forward references.

Example input file:

```
CS_DUMP_FILE 1.01
CROSSING_LINE: (-10000 0 0) (10000 0 0)
OBJECTS_CROSSED: 5
BLOCKDEF TOP_BLOCK
POLYLINE LAYER MMPOLY3
VERTEX (-2491.5 -751.5 0.0) 0.0
VERTEX (-2491.5 3908.5 0.0) 0.0
VERTEX (6168.5 3908.5 0.0) 0.0
VERTEX (6168.5 -751.5 0.0) 0.0
EOE
POLYLINE LAYER MMPOLY3
VERTEX (-1491.5 -751.5 0.0) 0.0
VERTEX (-1491.5 3908.5 0.0) 0.0
VERTEX (3168.5 3908.5 0.0) 0.0
VERTEX (3168.5 -751.5 0.0) 0.0
EOE
ENDBLK
POLYLINE LAYER DIMPLE1_CUT
VERTEX (2272.81 506.728 0.0) -0.209759
VERTEX (2221.95 419.166 0.0) 3.20817
VERTEX (2737.12 432.928 0.0) -0.209759
VERTEX (2681.66 517.65 0.0) 0.869459
EOE
POLYLINE LAYER MODULE_BOUNDARY
VERTEX (-1491.5 -751.5 0.0) 0.0
VERTEX (-1491.5 3908.5 0.0) 0.0
VERTEX (3168.5 3908.5 0.0) 0.0
VERTEX (3168.5 -751.5 0.0) 0.0
EOE
CIRCLE LAYER MMPOLY2
CENTER (0.0 500.0 0.0) RADIUS 500.0
POLYLINE LAYER MMPOLY0
```

```

VERTEX (-932.5 1155.0 0.0) 0.0
VERTEX (1737.0 1155.0 0.0) 0.0
VERTEX (1737.0 -73.0 0.0) 0.0
VERTEX (-1058.5 -73.0 0.0) 0.0
VERTEX (-1058.5 1029.0 0.0) -0.414214
EOE
BLOCK LAYER PIN_JOINT_CUT
NAME TOP_BLOCK @ (4500.0 -70.0 0.0) XSCALE 1.0 YSCALE 1.0 ROT 1.5708

```

In the example above, the first line specifies the file type and version. File type 'CS\_DUMP\_FILE' is used for the 2D Process Visualizer. File type 'AS\_DUMP\_FILE' is used for the 3D Visualizer and 3D Modeler.

The second line specifies the the cross-section line for the for the 2D Process Visualizer. The format specifies two 3D points:

```
CROSSING_LINE: (xxx.x yyy.y zzz.z) (xxx.x yyy.y zzz.z)
```

The second line specifies the the selected retangular mask area for the for the 3D Visualizer and the 3D Modeler. The format specifies two 3D points:

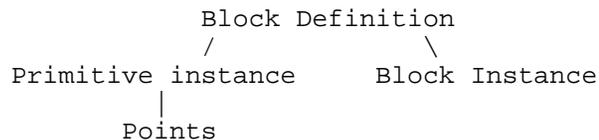
```
CROSSING_AREA: (xxx.x yyy.y zzz.z) (xxx.x yyy.y zzz.z)
```

The third line specifies the number of objects selected by the crossing line or crossing area. This parameter is not used by any of the tools.

Essentially, both file formats are interchangeable except that the 3D versions specify the keyword CROSSING\_AREA instead of CROSSING\_LINE and OBJECTS\_CROSSED becomes OBJECTS\_SELECTED.

The remainder of the file contains the geometry of interest. It consists of two sections: The first section contains block definitions. The second section contains block and primitive instances.

A block definition may be hierarchical and may contain block instances and/or primitive entries. The form is shown below:



Block definitions start with a header instance. It may contain primitive instances or block instances, and is finished with an End Of Block (ENDBLK) instance.

A header instance defines the block name:

```
BLOCKDEF block_name
```

Any combination of block or primitive instances may be contained in a block definition. This allows nested blocks. All block instances must be previously defined in a block definition (no forward references). The block definition is finished when a matching End of Block (ENDBLK) instance is encountered.

A block instance references a block definition:

```
BLOCK LAYER layername  
NAME block_name @ (xxx.x yyy.y zzz.z) XSCALE ff.f YSCALE ff.f ROT ff.f
```

The 'layername' is an ASCII string defining the layer where the block is placed. The 'block\_name' is the block to be placed at the double precision floating point 3D coordinate specified. An X and Y scale is specified in double precision floating point values along with a CW rotation in radians (also double precision).

Primitive instances consist of polylines and circles. A polyline instance consists of a polyline header and a list of vertices. The polyline header:

```
POLYLINE LAYER layername
```

defines the layer on which the closed polyline exists. It is followed by a list of VERTEX instances with the format:

```
VERTEX (xxx.x yyy.y zzz.z) bbb.b
```

where x,y,z are double precision floating point coordinates and b is the 'bulge' factor used to define arcs in the AutoCAD DXF format (see the AutoDESK web site for more information about bulges). Each polyline instance is finished when a matching EOE instance is encountered. All polylines are closed by implication that the last point connects to the first point.

The first entity instance encountered that is not contained inside a block definition signals the start of geometry placed in the world coordinate system, assumed to be in microns, centered about [0,0,0].