*ALE_STRUCTURED_MESH

Purpose: This keyword generates a structured 3D mesh and invokes the Structured ALE (S-ALE) solver. Spacing parameters are input through one or more of the *ALE_STRUCTURED_MESH_CONTROL_POINTS cards. The local coordinate system is defined using the *ALE_STRUCTURED_MESH card.

In certain contexts it is advantageous to use structured meshes. With structured meshes the element and node connectivity are straightforward and the searching algorithm used for ALE coupling is greatly simplified. Also numerous checks are avoided because these meshes include only HEX elements.

The S-ALE solver supports SMP, MPP and MPP hybrid configurations. All three implementations require less simulation time and memory usage than the regular ALE solver. We, therefore, encourage using the S-ALE solver when the ALE mesh is structured.

The S-ALE solver uses the same set of keyword cards as the regular ALE solver with the only exception being this keyword. Once an ALE mesh is generated using *ALE_STRUCTURED_MESH card, this card invokes the S-ALE and performs the ALE advection time step. For fluid-structure interaction using the *CONSTRAINED_LAGRANGE_IN_SOLID card, S-ALE uses a much faster searching algorithm that takes advantage of the mesh structure.

This keyword can be used multiple times. In each occurrence an independent, separate mesh is constructed. These meshes can occupy different spatial domains, or same spatial domain. Simulations are done in those meshes independently.

There are two related keywords: *ALE_STRUCTURED_MESH_TRIM to trim/untrim the generated mesh and *ALE_STRUCTURED_MESH_MOTION to control the mesh motion during the simulation.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	DPID	NBID	EBID				TDEATH
Type	I	I	I	I				F
Default	0	none	0	0				1.e+16

Card 2	1	2	3	4	5	6	7	8
Variable	CPIDX	CPIDY	CPIDZ	NID0	LCSID			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. A unique number must be specified.
DPID	Default Part ID. The elements generated are assigned to DPID. This part definition is automatically generated during the input phase and contains neither material nor element formulation information. Please see Remark 1. It also serves as an indicator to merge to adjacent mesh to form a mesh consists of several box-shaped sub-meshes. Please see remark 5.
NBID	Nodes are generated and assigned with node IDs starting from NBID.
EBID	Elements are generated and assigned with element IDs starting from EBID.
TDEATH	Death time for this mesh. Please see Remark 3.
CPIDX, CPIDY, CPIDZ	Control point IDs defining node ID/value pairs along each local axis. See *ALE_STRUCTURED_MESH_CONTROL_POINTS. Setting CPIDX to 0 or -1 invokes ALE to S-ALE converter. Please see remark 4.
NID0	NID0 sets the mesh's origin node. During the simulation, prescribed motion applied to this node applies to the entire structure S-ALE mesh.
LCSID	Local coordinate system ID. Please see Remark 2.

Remarks:

1. **DPID.** DPID only consists of elements and nodes. It does not include material properties or integration rules. The requirement that a part ID be specified for these automatically

generated S-ALE solid elements exists only to satisfy the legacy rule that every element must be associated with a part. Users do not need to set up the *PART card for DPID. All PART definitions used in this card only refer to the mesh, not material.

2. **LCSID.** The local coordinate system is defined on the data cards associated with the *DEFINE_COORDINATE keyword. This local coordinate system specifies the three cardinal directions used for generating the structured ALE mesh. The structured mesh can be made to rotate by specifying a rotating local coordinate system. To define a rotating local coordinate system, use the *DEFINE_COORDINATE_NODES keyword with FLAG = 1 and then apply prescribed motion to the three coordinate nodes.

3. **TDEATH.** This option provides a way to terminate ALE calculation while Lagrangian parts still evolve. Once this death time is reached, all elements belonging to this mesh are deleted. In addition to that, *CONSTRAINED_LAGRANGE_IN_SOLID and *ALE_COUPLING_NODAL_?, are checked to see if related ALE elements were deleted. If no ALE elements are left on the fluid side for certain coupling cards, this coupling card is disabled from further calculation too.

4. **ALE to S-ALE Converter**. For existing ALE models with rectilinear mesh, *ALE_STRUCTURED_MESH can convert invoke the ALE mesh into an S-ALE mesh. To use this feature, add a *ALE_STRUCTURED_MESH card in the model input with CPIDX = -1 or 0 and all other fields blank. LS-DYNA will then convert all ALE keywords to the S-ALE format and write the modified input to a file named "saleconvrt.inc". The solver used to perform the analysis depends on the value of CPIDX. If -1, the S-ALE solver is used; if 0, the ALE solver is used.

5. **Mesh merging.** Two or more separated, but adjacent sub-meshes can be merged to form a mesh of complex shape. To do that, first set up the two or more sub-meshes; make sure their mesh spacing are the same at their interfaces so that these interface nodes collides with each other (a tolerance of 1/100 element size is allowed); and finally use the SAME DPID in those *ALE_STRUCTURED_MESH cards. The code automatically checks and merges sub-meshes with the same DPID.

Example:

1. The following example generates a regular evenly distributed 0.2 by 0.2 by 0.2 box mesh that has 22 nodes along each direction. The generated mesh is aligned to the local coordinate system specified by nodes 2, 3, and 4 with an origin at node 1.

All the elements inside the mesh are assigned to part 1. Note that part 1 is not explicitly defined in the input. The necessary part definition is automatically generated and contains neither material definitions nor integration rules.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid                                tdeath
         1         1    200001    200001                                 0.010
$    cpidx     cpidy     cpidz      nid0     lcsid
```

```
      1001      1001      1001         1       234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3      flag
       234         2         3         4         1
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2
                   1                  .0
                  22                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```