

# Center for Simulation of Advanced Rockets University of Illinois at Urbana-Champaign

## Summary

The primary goal of CSAR is detailed whole-system simulation of solid propellant rockets under normal and abnormal operating conditions. The GEN2 code developed at CSAR is a highly-scalable general-purpose solver for 3-D, tightly coupled, fluid/structure/combustion problems. GEN2 has been applied to a variety of complex rocket problems, including a failure mechanism that destroyed an early Titan IV booster. A newer version, GEN2.5, improves modularity, adds load balancing and more physics.

## GEN2 code

GEN2 includes several physics applications and a set of GEN2 service modules that comprise a plug-and-play software integration framework. The finite volume fluid dynamics solver *Rocflo* and finite element structural dynamics solver *Rocfrac* both take explicit time steps and adopt an Arbitrary Lagrangian-Eulerian (ALE) formulation to handle boundary motion due to deformation and burning. *Rocburn* applies a 1-D ignition and dynamic burn rate model at the face of each element on the burning propellant surface. Data transfer between physics applications with non-matching meshes at material interfaces is performed by *Rocface*, which ensures that conservation laws are exactly satisfied and that interpolation errors are minimal.

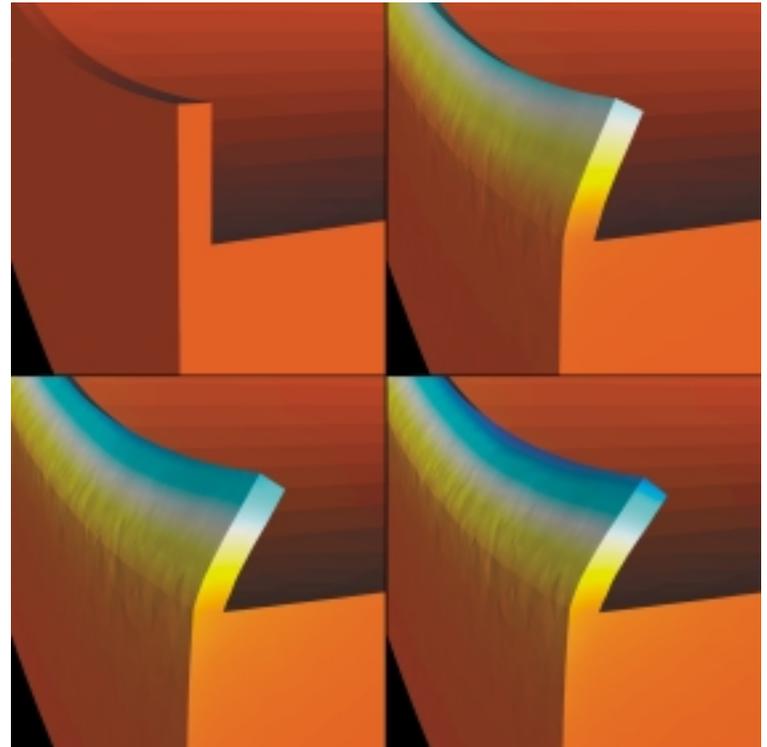


Figure 1. Flexible inhibitor deformation time series (clockwise from upper left).

The GEN2 code runs as a single executable program and uses MPI for message passing. The physics applications and a few service modules are written in Fortran 90, while most service modules are written in C++. All of the components of GEN2 are highly scalable, and the coupled code has achieved speedups of approximately 820 for 1024 processors on IBM SP systems at LLNL and SDSC. *Rocpanda* (<http://cdr.cs.uiuc.edu/panda/rocpanda/>) decreases I/O latency for large simulations.

GEN2 undergoes nightly automated building and testing on multiple target platforms, using a growing suite of coupled verification and validation problems. Each physics application is also tested individually on problems with known solutions and against industry-standard solvers.



## GEN2 Results

Several 3-D fully-coupled rocket simulations have been performed using GEN2. In one computation, we determined the motion of a flexible inhibitor at a joint slot in a section of a large booster such as the Space Shuttle RSRM. Our results demonstrate that such an inhibitor oscillates with a period of  $\sim 10$  ms about an equilibrium position with an angle of deflection of approximately 35 degrees (see Figure 1).

We also used GEN2 to perform 3-D simulations of propellant slumping in the Titan IV

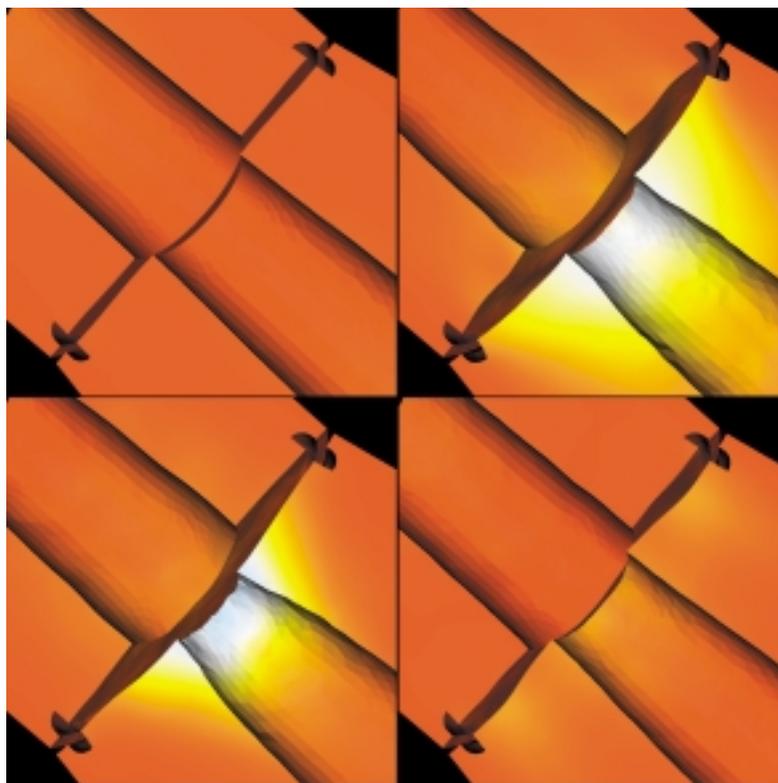


Figure 2. Titan IV propellant slumping time series (clockwise from upper left).

Prequalification Motor #1. In each panel of Figure 2, hot gas flows from the cylindrical section of burning propellant at the upper left, past a joint slot, and into the cylindrical propellant sec-

tion with a smaller bore at lower right. The gas pressure drops significantly as it speeds up in the narrower section, and the resulting net pressure force on the propellant pulls it further toward the axis of the rocket. The aft section of propellant oscillates with growing amplitude until the gas flow is choked and the pressure in the forward section bursts the case.

## GEN2.5

GEN2.5, a more modular, versatile, and accurate version of GEN2, has recently been implemented. The most important improvement is that time stepping, interface data exchange, surface propagation, and evaluation of jump conditions at the combustion interface are now performed by a new service module *Rocman*. It is much easier to add a new general-purpose discipline-specific physics module to GEN2.5 compared to GEN2, because *Rocman* reduces significantly the need to modify new modules.

GEN2.5 is fully compatible with the Charm++ runtime library (<http://charm.cs.uiuc.edu>), in which the job's processes are implemented as lightweight threads. When there are several threads running per processor, the threads can be migrated from one CPU to another with low overhead. This enables automatic load balancing, which can be triggered by an initially unbalanced load, or at any time by automatic

mesh refinement.

**Contact Information:**

<http://www.csar.uiuc.edu> or [rfiedler@uiuc.edu](mailto:rfiedler@uiuc.edu)