Xplicit Computing

XCOMPUTE: ADVANCED SYSTEMS FOR COMPLEX SCIENCE & ENGINEERING

NASA Advanced Modeling & Simulation Seminar

- NASA Advanced Supercomputing Division, Ames Research Center
- April 19, 2018, Graham J Orr, CEO & founder Xplicit Computing Inc
- info@xplicitcomputing.com
- www.xplicitcomputing.com

PROBLEM STATEMENT

As real-world systems increase in sophistication, conventional CAE is approaching complexity limits.

Teams have struggled with isolated analyses, resulting in lost-in-translation errors and higher-level integration oversights.

Most tools are slow, static, or inaccessible — impeding innovation.

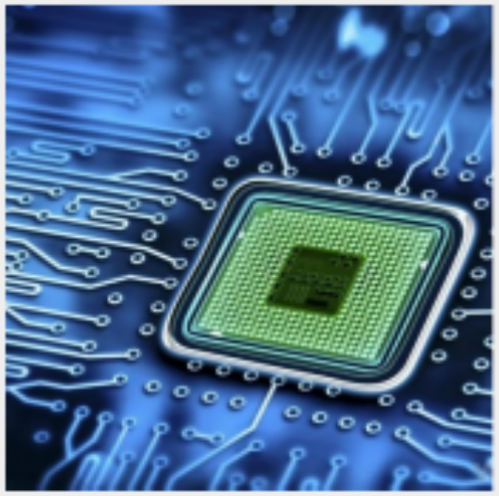
SOLUTION SUMMARY

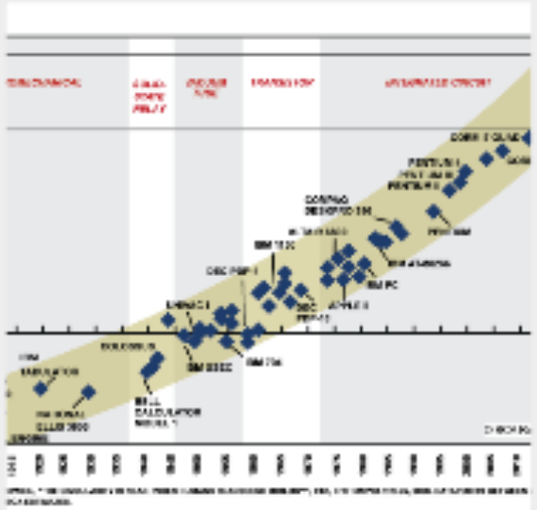
Integrate all CAE functionalities in a unified & dynamic numerical computing environment, leveraging progress in templated software and concurrent hardware.

Let engineers build and own their virtual systems using HPC building blocks.

Systems | Geometry | Physics | Graphics

Why Does Simulation Matter?





complexity is increasing exponentially in many sectors!







Leveraging design principles used for semiconductors and the resulting exponential growth of computational power we can accelerate innovation in other key industries!

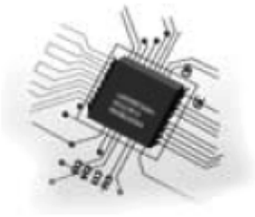
110 Years of Computing Virtualization
100,000,000,000,000,000x FLOPS/$

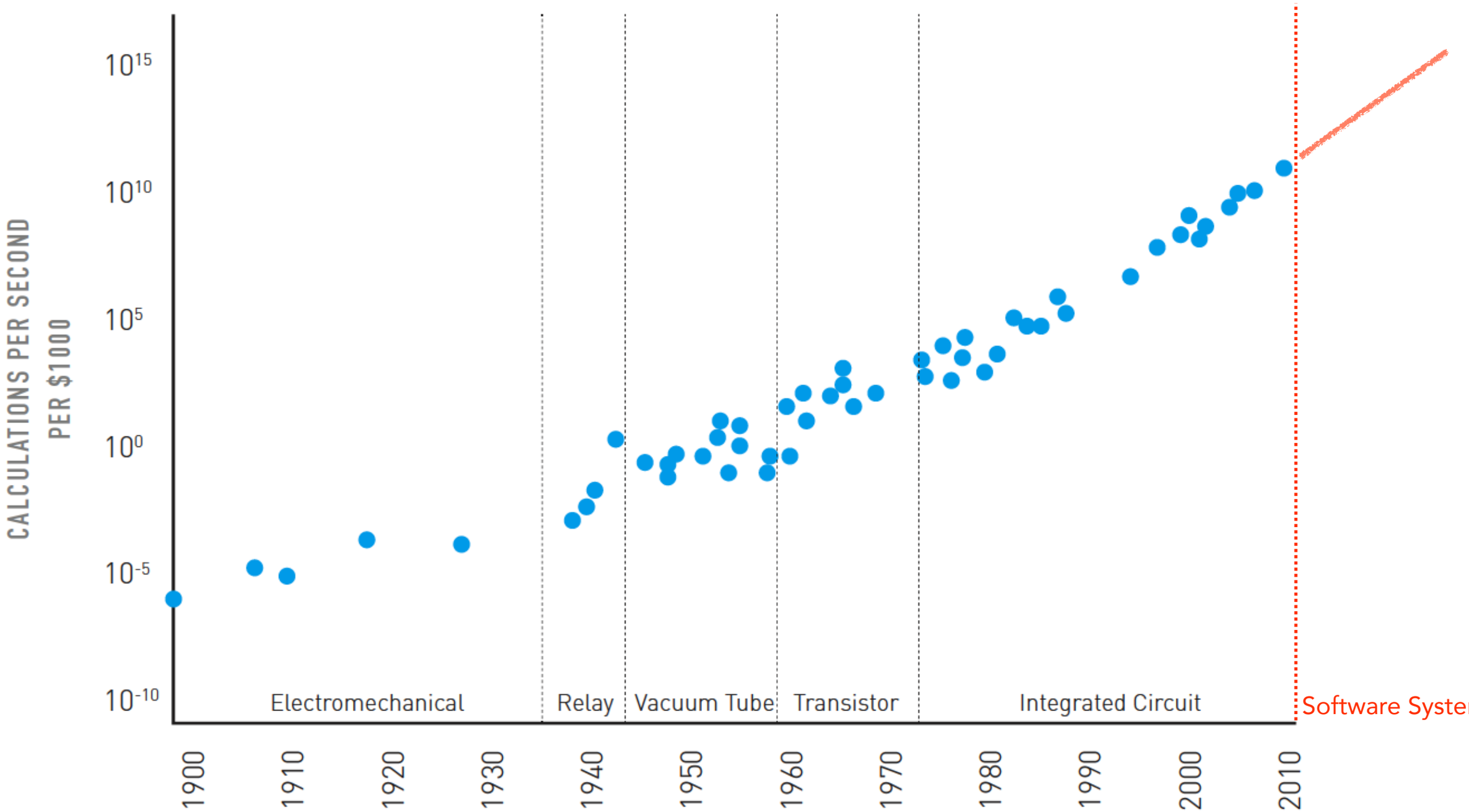








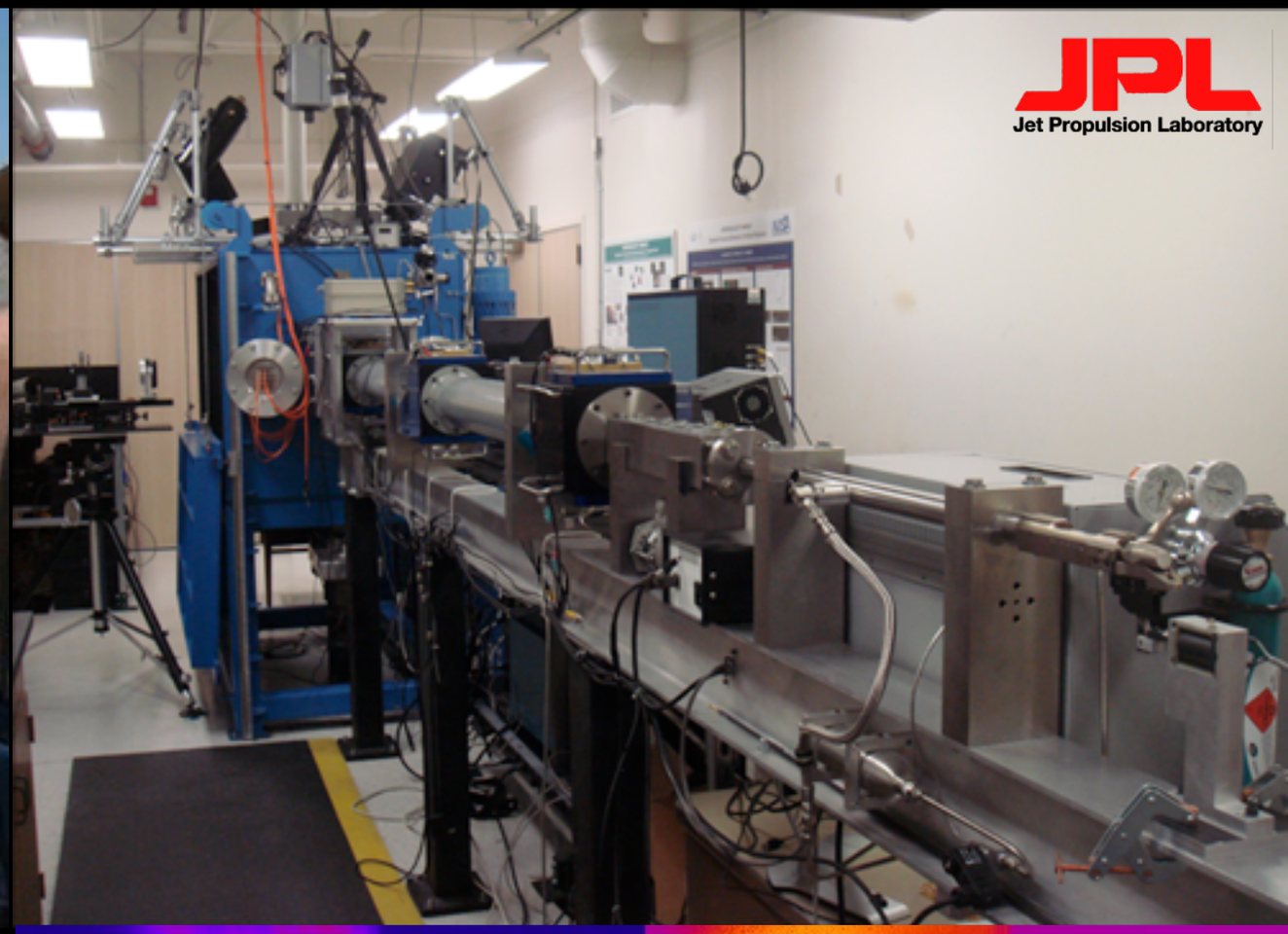




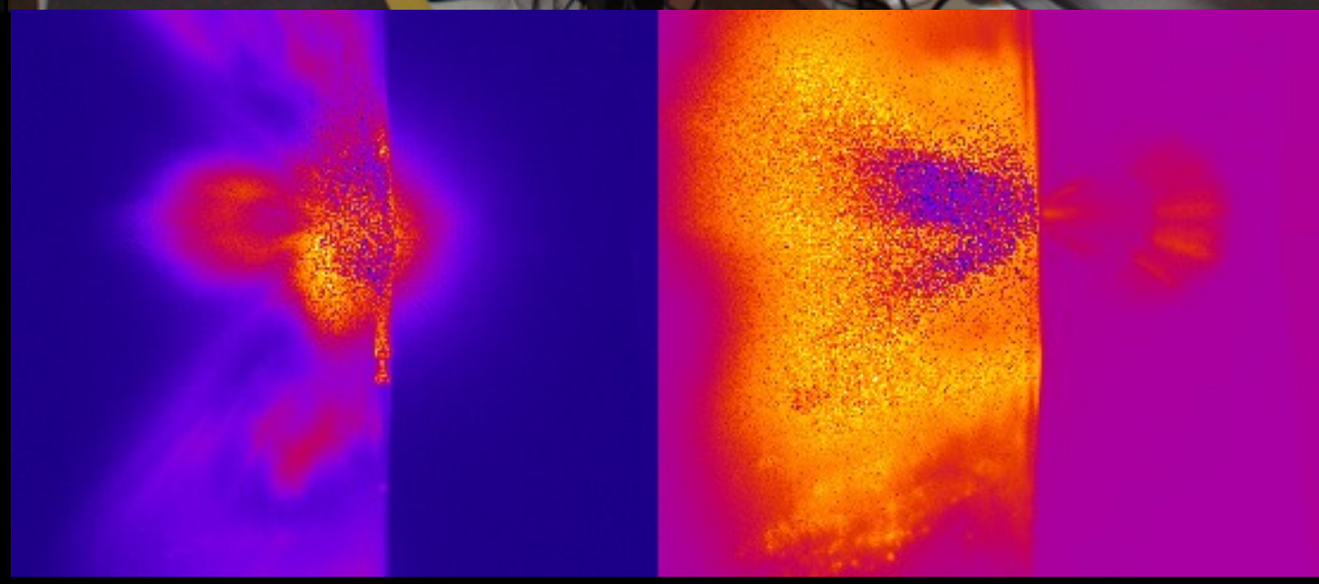
[7]

MY EARLY CAREER





- B.Sc. HMC General Engineering
    - Experimental Engineering rocket revamp
- JPL / GALCIT APT
    - Dawn and MSL Propulsion 353x I&C technician
    - Small Particle Hypervelocity Impact Range I&C



LAST DECADE



BLUE ORIGIN

GEEx Superstructure

BE-3 Rapid Dev Support



VIRGIN GALACTIC

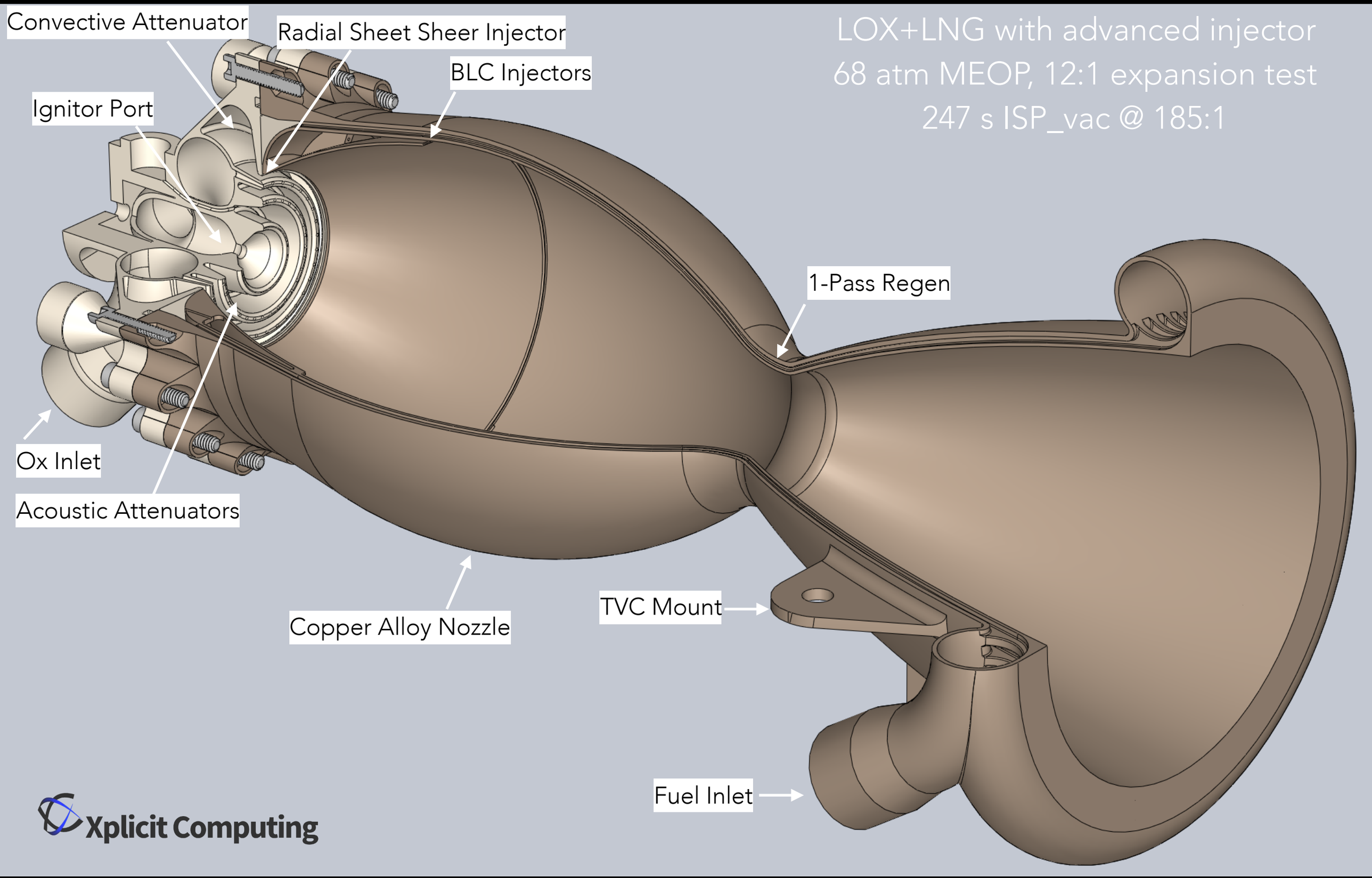
NS-1 Facility Systems CogE

Newton-1 Systems Support



Other R&D Programs

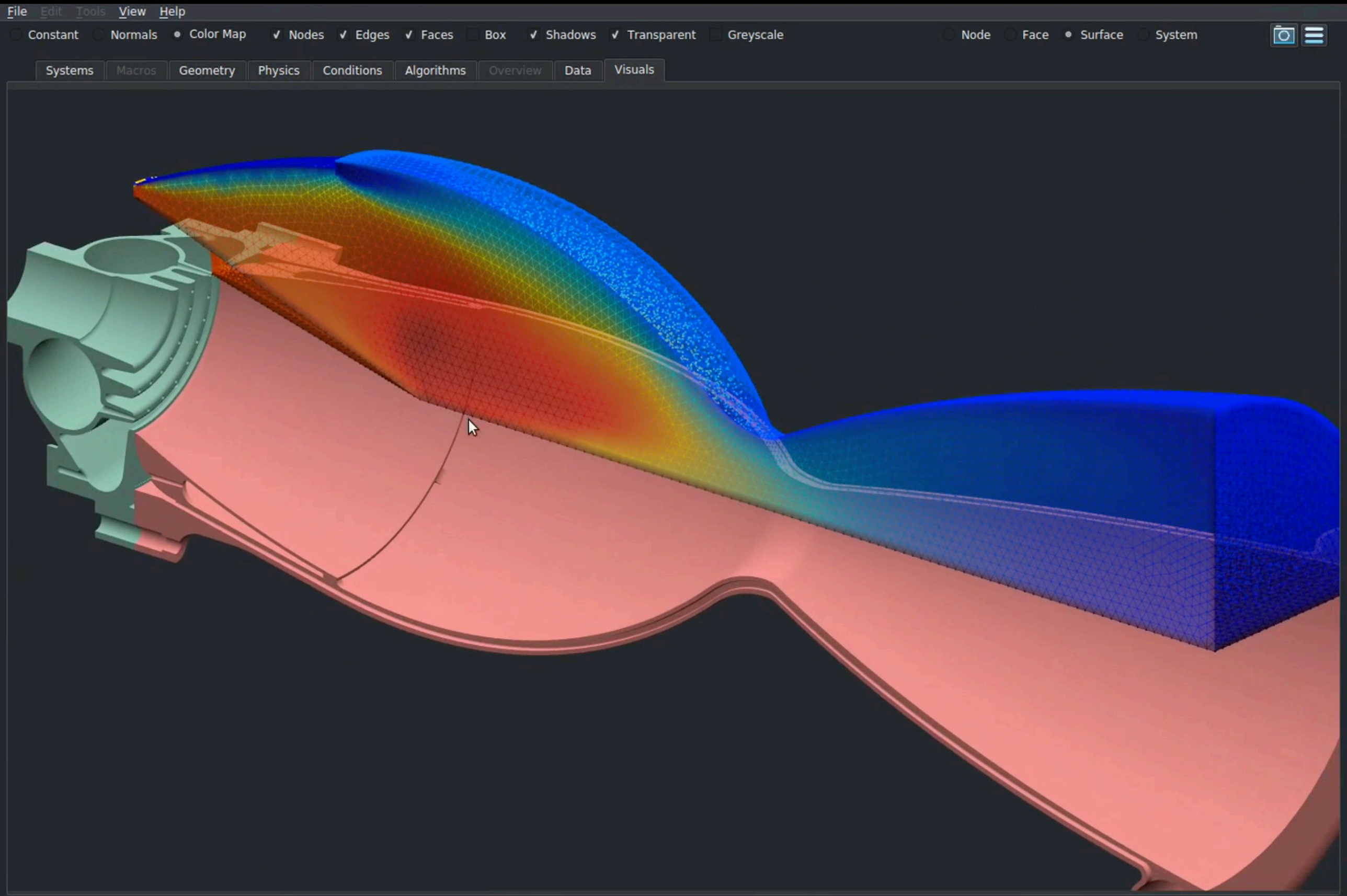
"LR1000" AM-COMBUSTOR TESTBED



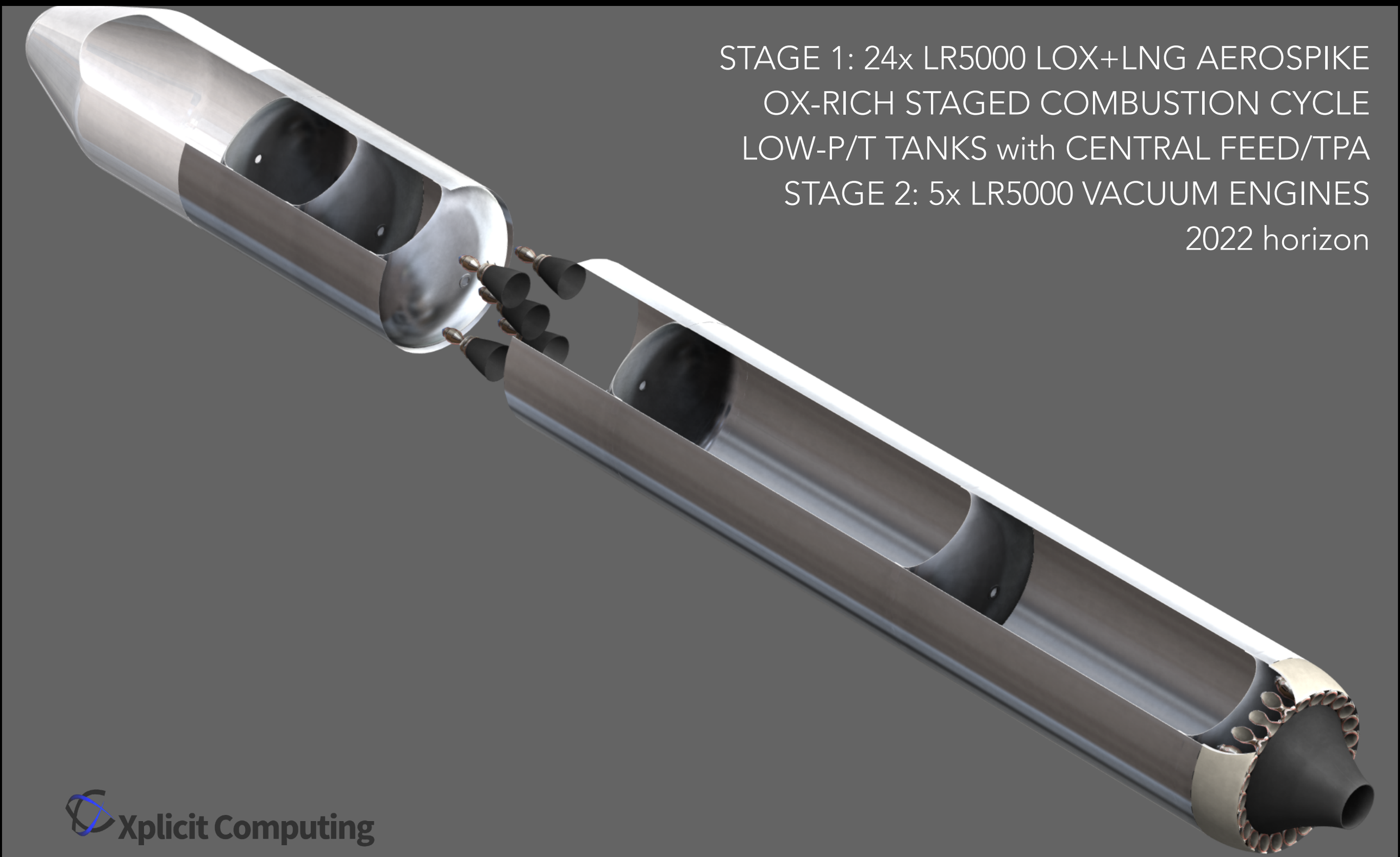
LOX+LNG with advanced injector
68 atm MEOP, 12:1 expansion test
247 s ISP_vac @ 185:1

Xplicit Computing

LR1000 PDR ~ COMPRESSIBLE VISICID FLOW
XCOMPUTE IN REAL-TIME (AUSM+MUSCL FVM, LOCAL DT)



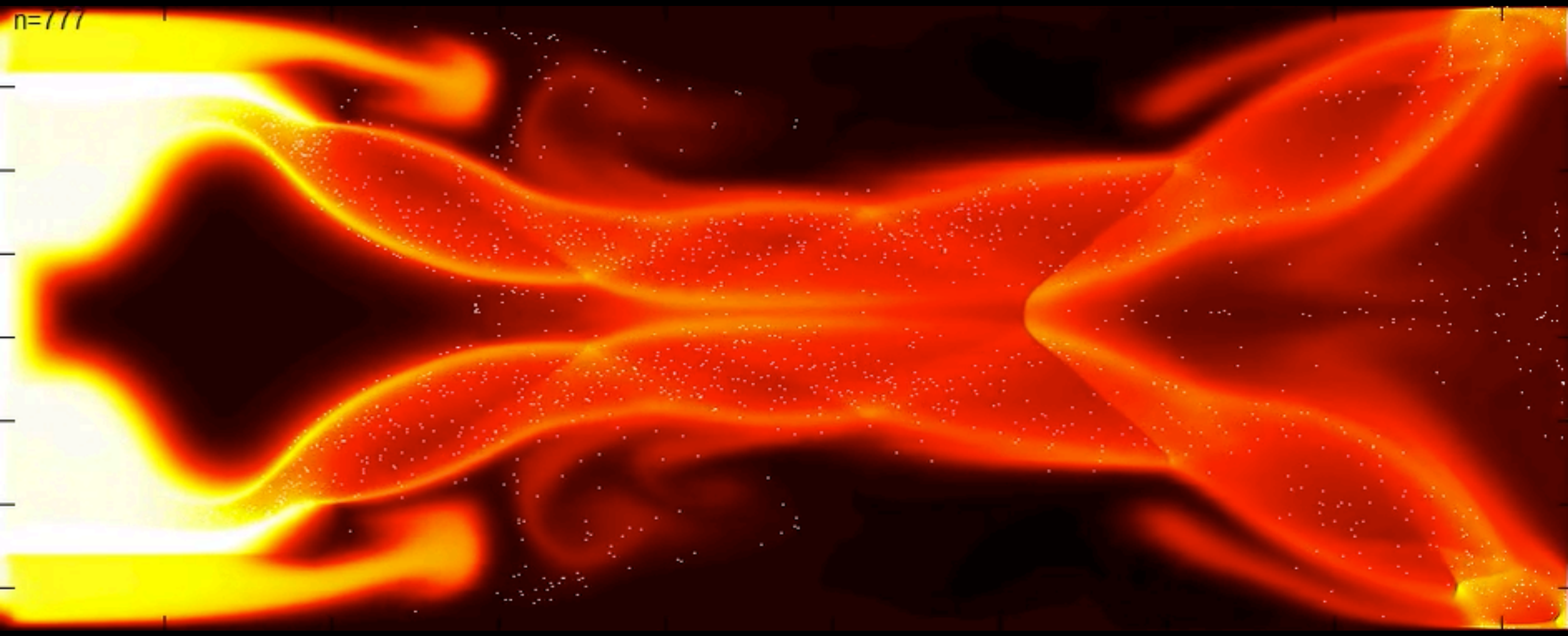
"ARMSTRONG" SMALL-SAT RLV CONCEPT



STAGE 1: 24x LR5000 LOX+LNG AEROSPIKE
OX-RICH STAGED COMBUSTION CYCLE
LOW-P/T TANKS with CENTRAL FEED/TPA
STAGE 2: 5x LR5000 VACUUM ENGINES
2022 horizon

Xplicit Computing

AEROSPIKE NOZZLE START-UP GAS TEMP (FDM, 2012)



mixed PDE solved with convolutional 2nd-order MacCormack method on a laptop

ISSUES WITH SETUP AND RESOLUTION - BUT STILL INSIGHTFUL!

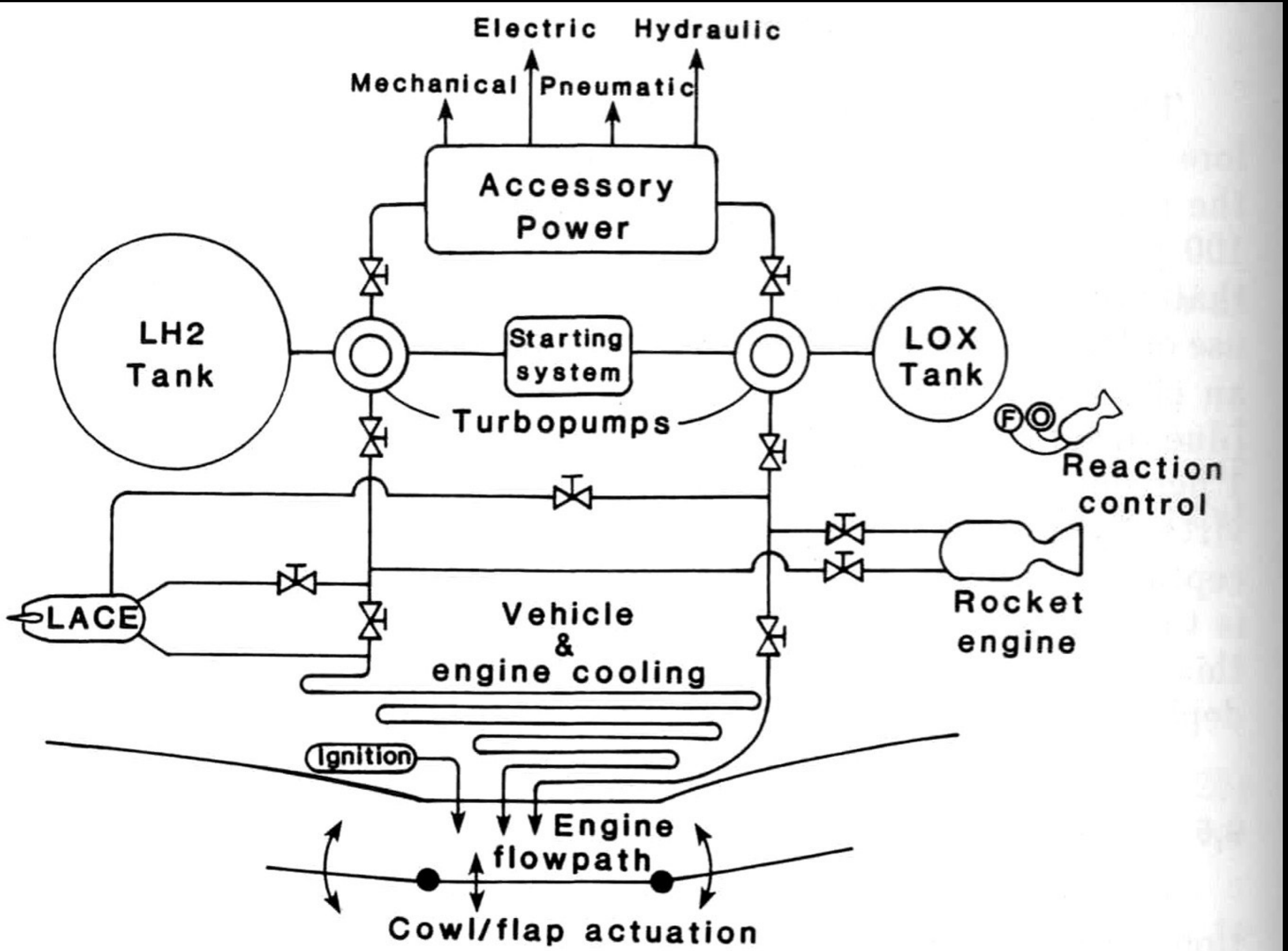
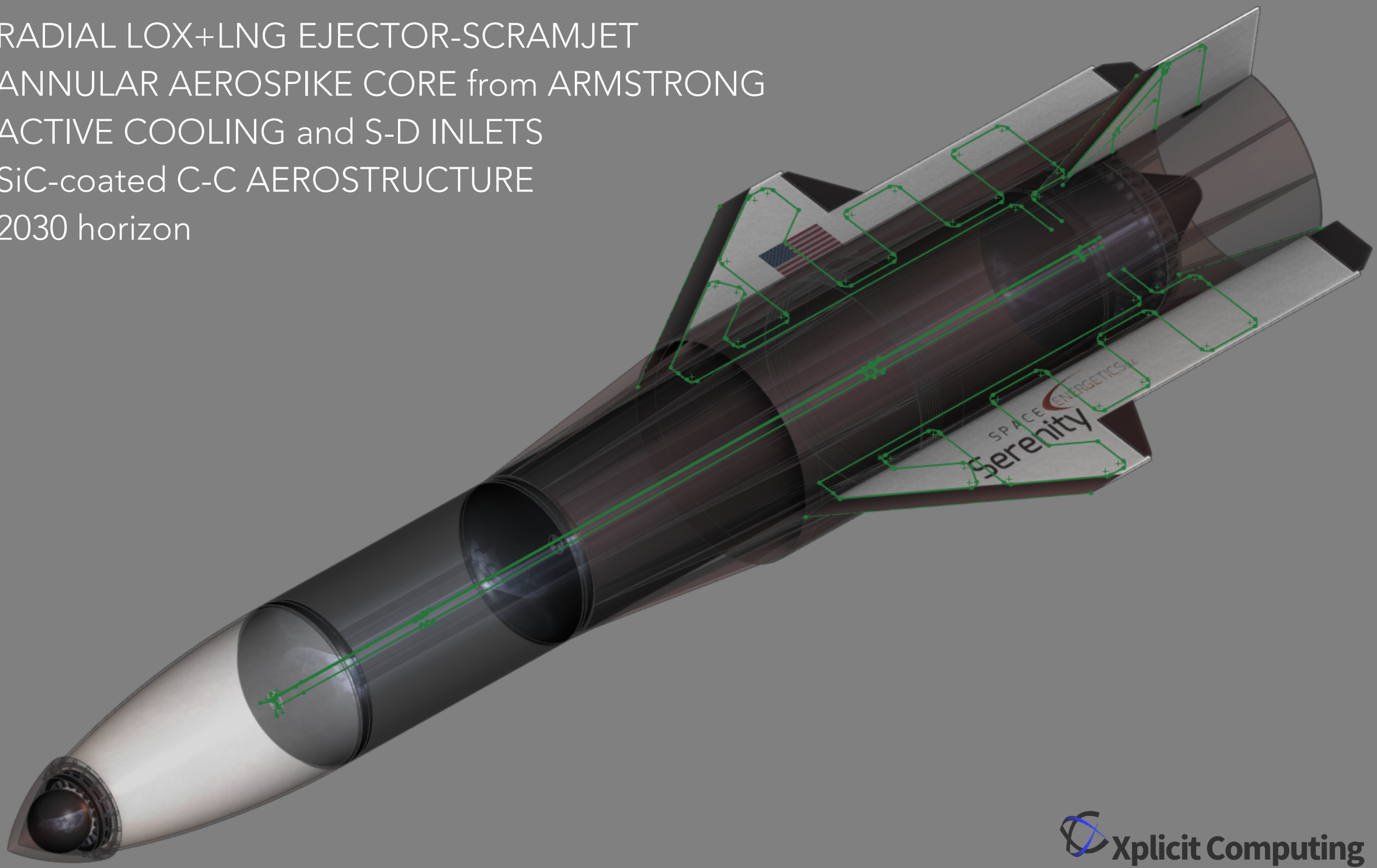


Fig. 9.15 Schematic diagram showing the subsystems of a typical hypersonic airbreathing engine. [6]

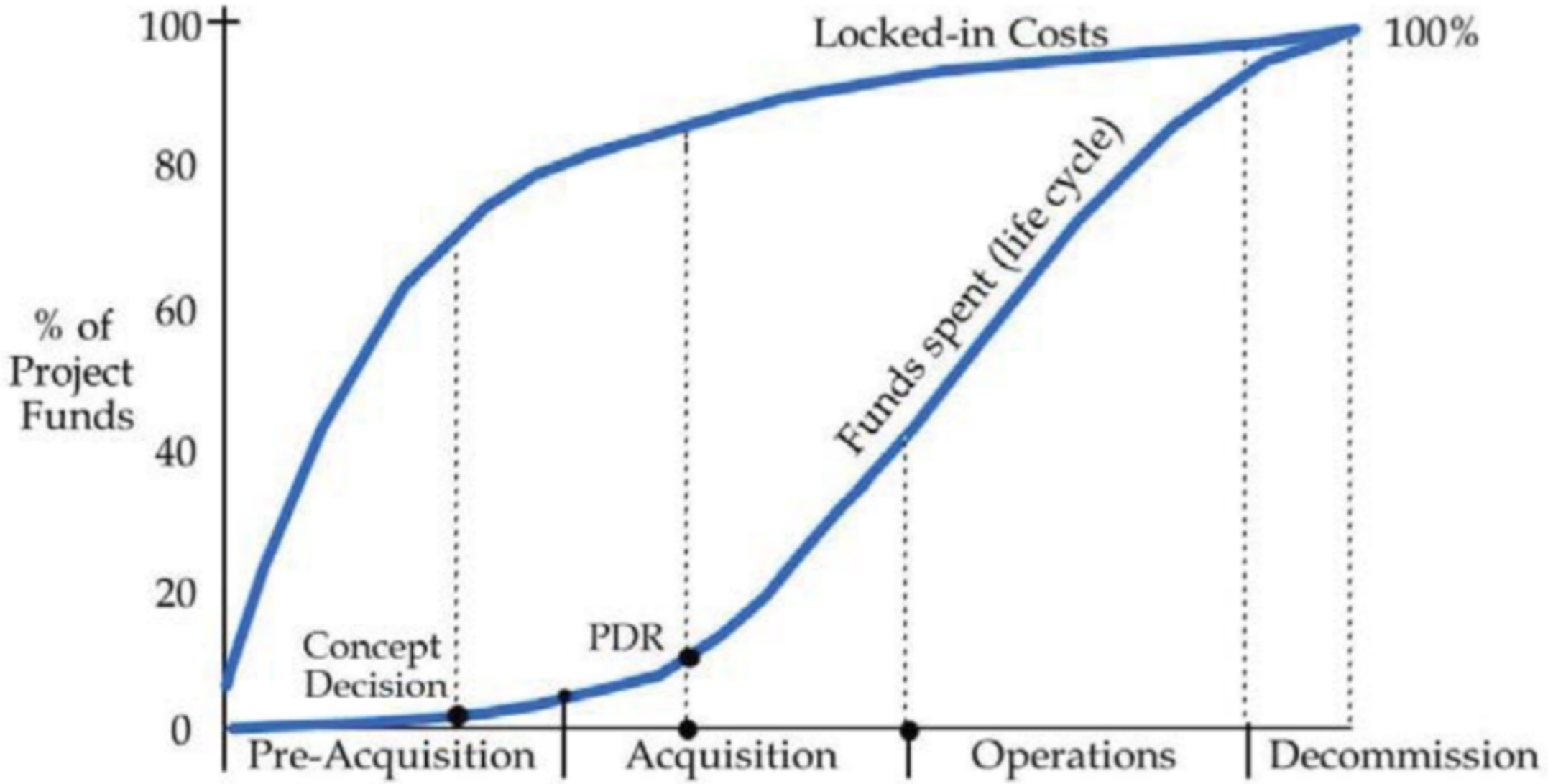
“SERENITY” REUSABLE SSTO CONCEPT

RADIAL LOX+LNG EJECTOR-SCRAMJET
ANNULAR AEROSPIKE CORE from ARMSTRONG
ACTIVE COOLING and S-D INLETS
SiC-coated C-C AEROSTRUCTURE
2030 horizon



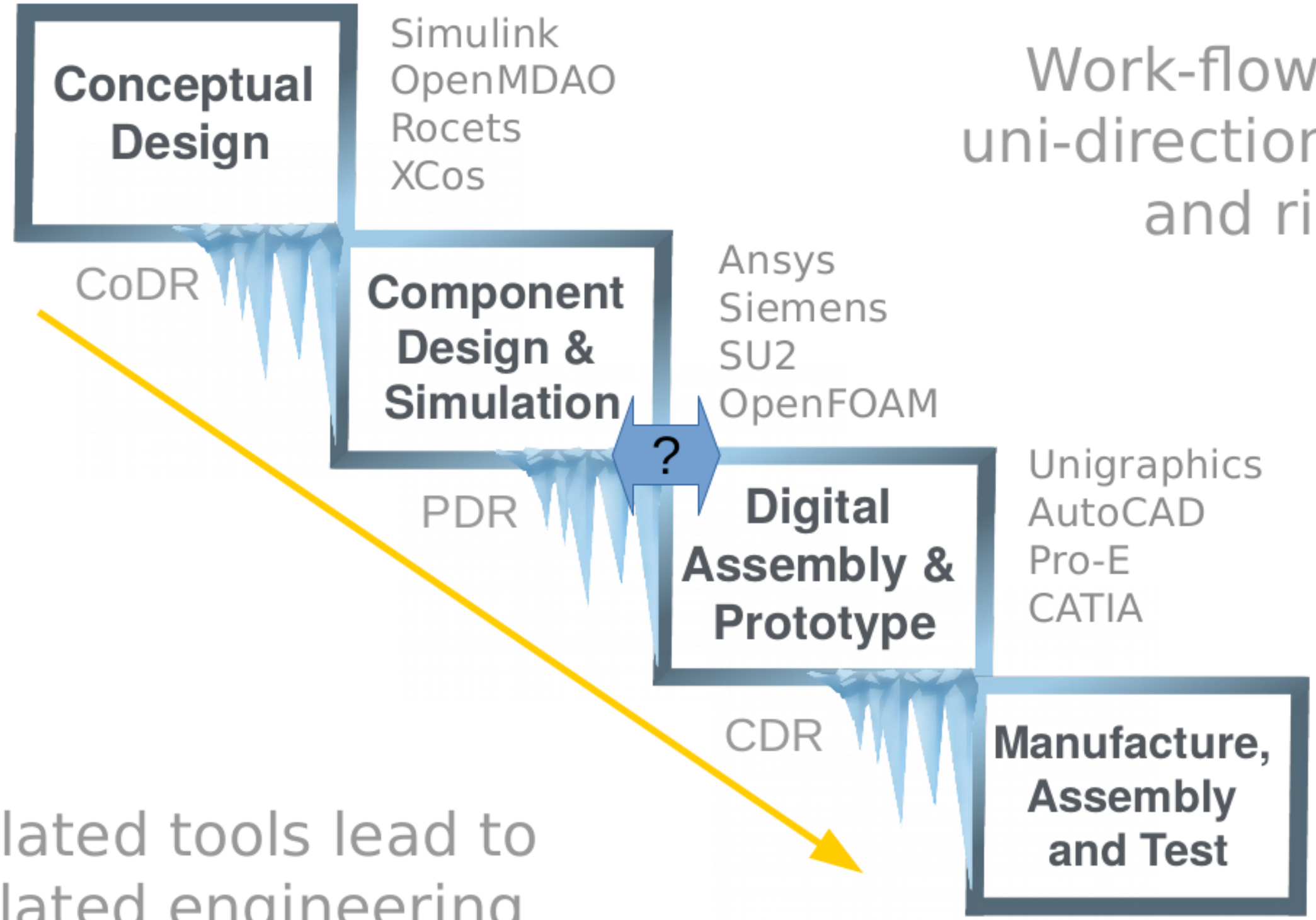
Xplicit Computing

BIGGEST OPPORTUNITY - AT EARLIER STAGES!



Di Pietro, David A. "Techniques for Conducting Effective Concept Design and Design-to-Cost Trade Studies." INCOSE International Symposium. Vol. 25. No. 1. 2015.

Traditional "Waterfall" Design Process



Work-flow is uni-directional and rigid

Isolated tools lead to isolated engineering

Jørgensen, Niels. "The Boeing 777: Development Life Cycle Follows Artifact." World Conference on Integrated Design and Process Technology (IDPT). 2006.

DEFINE

1-12 wks

ANALYZE

1-12 wks

BUILD

1-12 wks

TEST

1-12 wks

DESIGN CYCLE
1 MONTH - 1 YEAR

This approach is likely the best for simplicity and weight optimization

$$\sigma_{i\theta} = \begin{bmatrix} \frac{R_o^2 P}{R_o^2 - R_i^2} - \sigma_0 - \frac{M y(i,\theta)}{I_{yy}} & 0 & 0 \\ 0 & \frac{2 R_o^2 P}{R_o^2 - R_i^2} & 0 \\ 0 & 0 & -P\left[1 - \frac{r - R_i}{R_o - R_i}\right] \end{bmatrix}$$

$\sigma_p - \sigma_0 - \sigma_M$

$\sigma_r = P - \frac{r - R_i}{R_o - R_i} P$

$\tau = R_o - R_i$ near surface shear is small

$\lambda v_{i\theta} = \sigma_{i\theta} \cdot v_{i\theta} \rightarrow 0 = \sigma_{i\theta} \cdot v_{i\theta} - \lambda v_{i\theta} \rightarrow 0 = |\sigma_{i\theta} - \lambda I_{i\theta}|$

since $\nabla \times \sigma_{i\theta} = 0$ (by inspection) soln is simple

$$0 = \begin{vmatrix} \sigma_{ii} - \lambda & 0 & 0 \\ 0 & \sigma_{\theta\theta} - \lambda & 0 \\ 0 & 0 & \sigma_{rr} - \lambda \end{vmatrix} = |\sigma_{i\theta} - \lambda| \rightarrow 0 = (\sigma_{ii} - \lambda)(\sigma_{\theta\theta} - \lambda)(\sigma_{rr} - \lambda)$$

$\lambda = \{\sigma_{ii}, \sigma_{\theta\theta}, \sigma_{rr}\}$ as expected

Check Local Stresses:

Von Mises Stress (note: includes only principle components)

$$\sigma_{vm} = \sqrt{\frac{1}{2}\left[(\sigma_{ii} - \sigma_{\theta\theta})^2 + (\sigma_{\theta\theta} - \sigma_{rr})^2 + (\sigma_{ii} - \sigma_{rr})^2\right]}$$

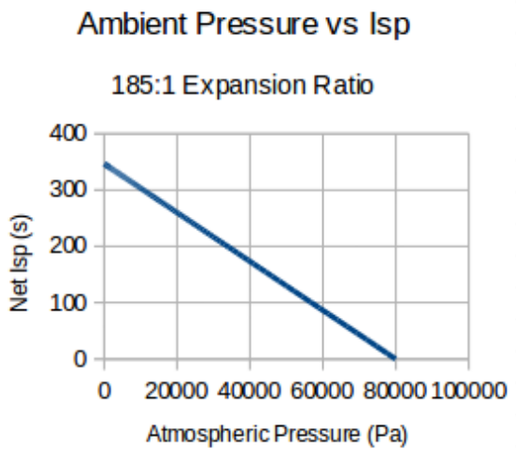
SEEM FAMILIAR? LR1000 - SYSTEM OVERVIEW

LR1000-calcs.ods - LibreOffice Calc

Liberation Sans | 10 | B I U

I30 | =(1-B30/B$17)*I$17*K$17/L$17

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	**LR1000-CoDR**		**(c) Xplicit Computing Inc, 2018**												last edit:	Graham Orr		17 April 2018
2	**1D NOZZLE ANALYSES**																	
3	**CHAMBER**		**LOX+CH4**															
4		F*	Pc	gamma_c	MW	T_adiabat	L*		R_c	cp	Tc	Dc	Lc	rho_c	uc_avg	ac	Mc	twl_min
5		N	N/m2	cp/cv	g/mol	K	m		J/kgK	J/kgK	K	m	m	kg/m3	m/s	m/s		m
6		**1620**	**6.91E+06**	1.149	20.69	**3348**	**1.14**		401.91	3090.3	**3248**	**0.0613**	**0.127**	5.3	**86**	1225	0.07	**6.8E-04**
7																		
8			Eta_chem		MR (O/F)		in					in	in					in
9			**0.97**		**2.6**		**45**					**2.4127**	**5.0000**					**0.0269**
10	**THROAT**																	
11		Eps_con		gamma*	MW				R_*	P*	T*	A*	D*	D*	u*	mdot*		
12		Ac/A*		cp/cv	g/mol				J/kgK	N/m2	K	m2	m	in	m/s	kg/s	lb/s	
13		**9**		1.153	20.90				389.19	4.29E+06	**3019**	3.28E-04	**0.0204**	**0.8042**	**1164**	**1.348**	2.965	
14	**EXIT**																	
15		P_exit		gamma_e	MW				R_e	Me	Te	ue	Ae	De		Eps_exp		
16		N/m2	Eta_noz	cp/cv	g/mol				J/kgK		K	m/s	m2	m	in			
17		**2000**	**0.985**	1.222	21.42				379.79	**5.53**	**739**	**3306**	6.05E-02	**0.278**	**10.928**	**184.63**		
18	**NET**																	
19		P_ambient							du_exit	u_eq	Fnet		Efficiency	Isp_net	Isp deviation		P_sep	Exit condition
20		N/m2							m/s	m/s	N	lbf	KE/Etot	s	s +/- design		Pa	
21		**0**	**vacuum**						85	**3391**	**4570**	**1026.92**	**57.3%**	**347**	8.68	2.50%	0	**attached**
22		414							67	3373	4546	1021.60	56.7%	345	6.88	1.99%	106	**attached**
23		539							62	3368	4539	1020.00	56.5%	344	6.34	1.84%	137	**attached**
24		700							55	3361	4530	1017.92	56.3%	344	5.64	1.64%	179	**attached**
25		910							46	3352	4518	1015.22	56.0%	343	4.73	1.38%	232	**attached**
26		1183							35	3341	4502	1011.71	55.6%	342	3.54	1.04%	302	**attached**
27		1538							20	3325	4482	1007.15	55.1%	340	2.00	0.59%	392	**attached**
28		**2000**	**design point**						**0**	**3306**	**4455**	**1001.22**	**54.4%**	**338**	0.00	0.00%	510	**attached**
29		4000							-85	3221	4341	975.53	51.7%	329	-8.68	-2.63%	1020	**attached**
30		**6000**	**Mars surface**						-170	3136	4227	949.83	**49.0%**	321	**-17.35**	**-5.41%**	**1530**	attached
31		12000							-424	2882	3884	872.74	41.4%	295	-43.38	-14.72%	3061	**Detached!**
32		24000							-933	2373	3198	718.55	28.0%	243	-95.43	-39.34%	6121	**Detached!**
33		48000							-1951	1354	1825	410.19	9.1%	138	-199.54	-144.09%	12242	**Detached!**
34		**101325**	**Earth sea level**						-4214	**-908**	**-1224**	**-274.96**	**4.1%**	**-93**	-430.85	464.14%	25843	**Detached!**
35																		
36																		



Nozzle | Injector | Thermal | Acoustics | Chemistry | data

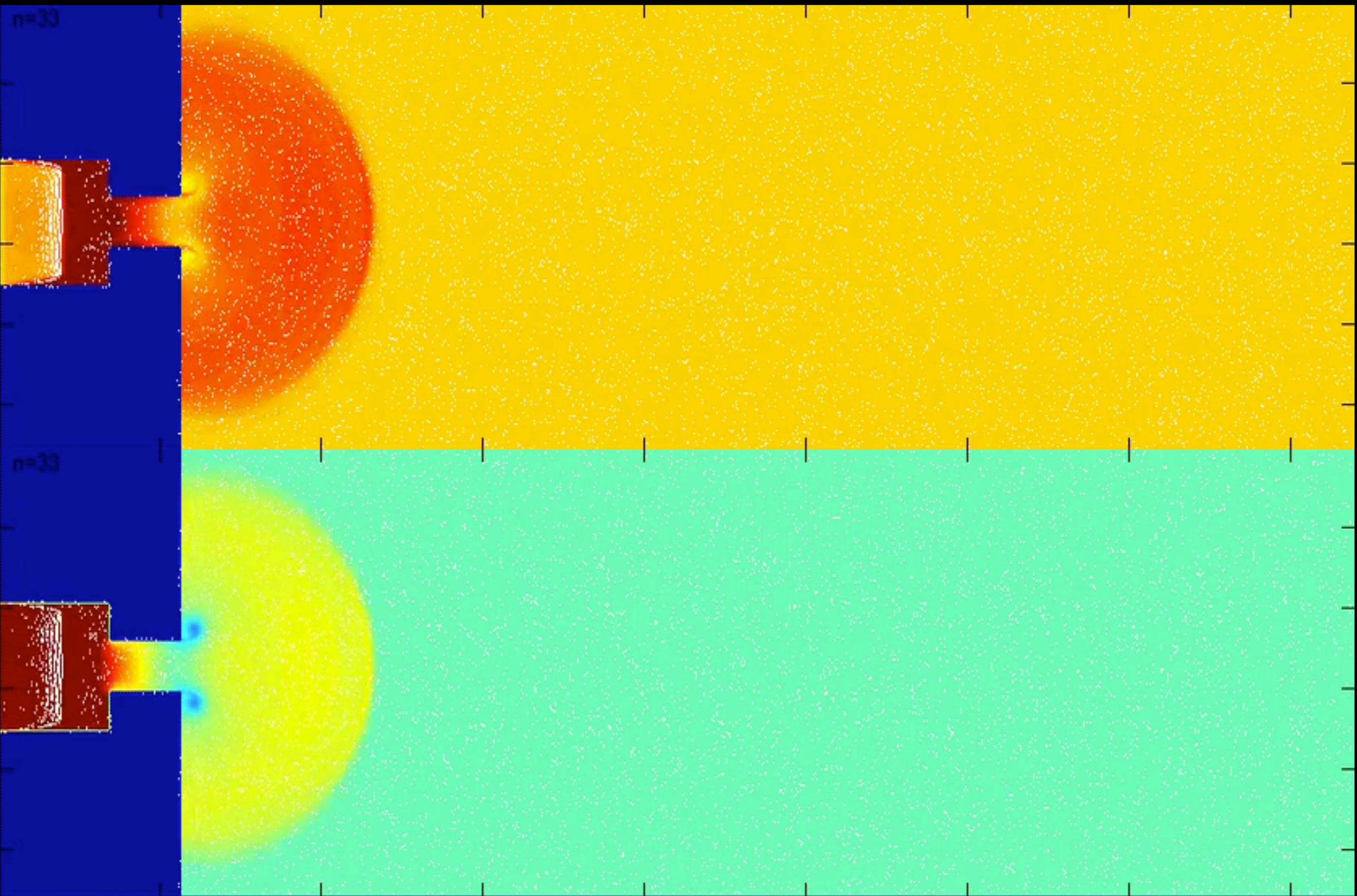
Sheet 1 of 6 | Default | Sum=15977 | 100%

tabulate design input / output parameters to assemble low-fidelity models

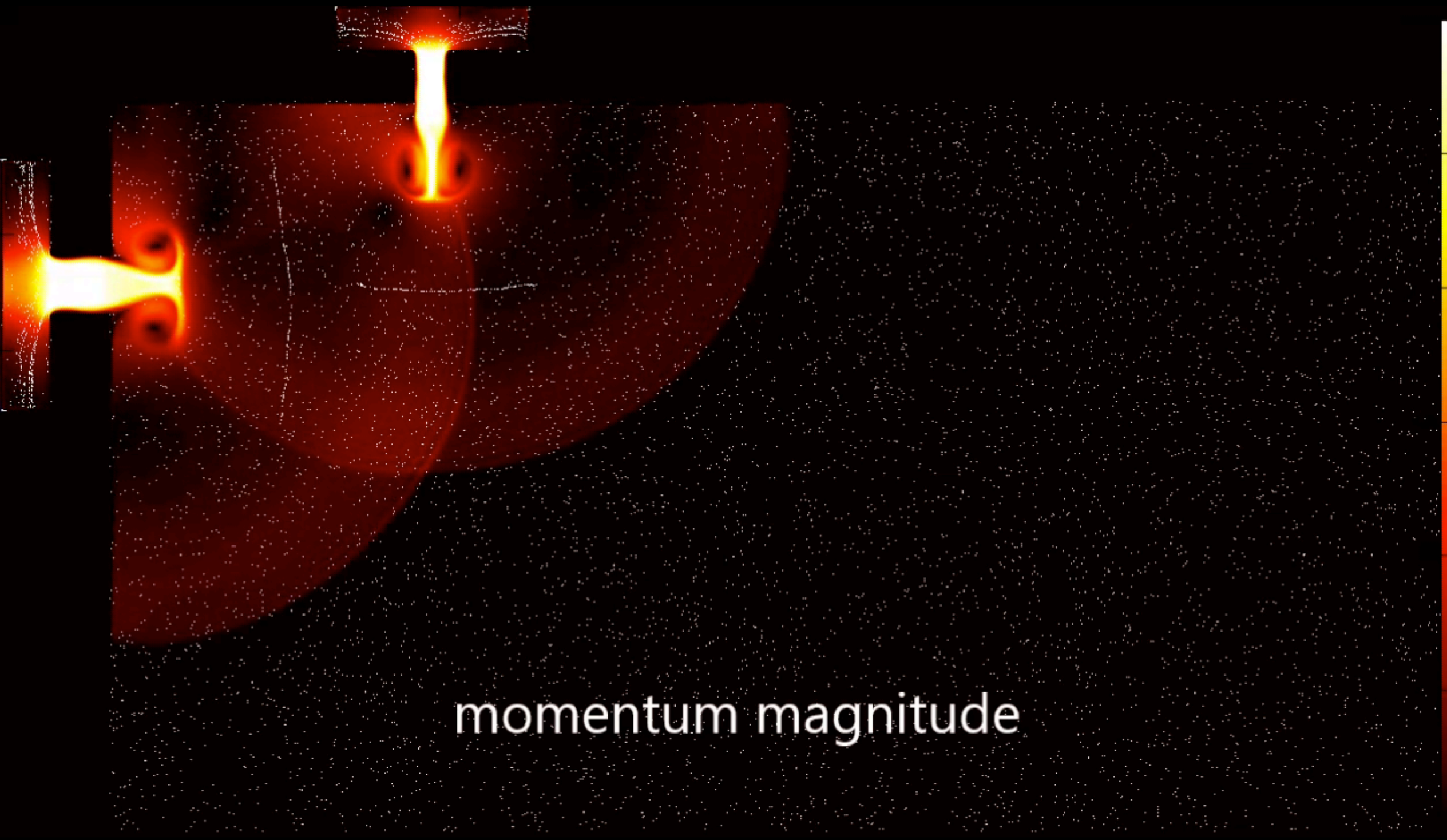
common tools: Excel, Matlab, Python, Julia, OpenMDAO, Fortran, C++

SUBSONIC GAS JET - TEMPERATURE AND DENSITY (FDM, 2012)

INJECTION SYSTEM ISOLATED?



TRANSVERSE TRANSONIC GAS JETS (FDM, 2013)



elliptic PDE solved with convolutional 2nd-order MacCormack method on a desktop

COMPLEXITY FROM COUPLING

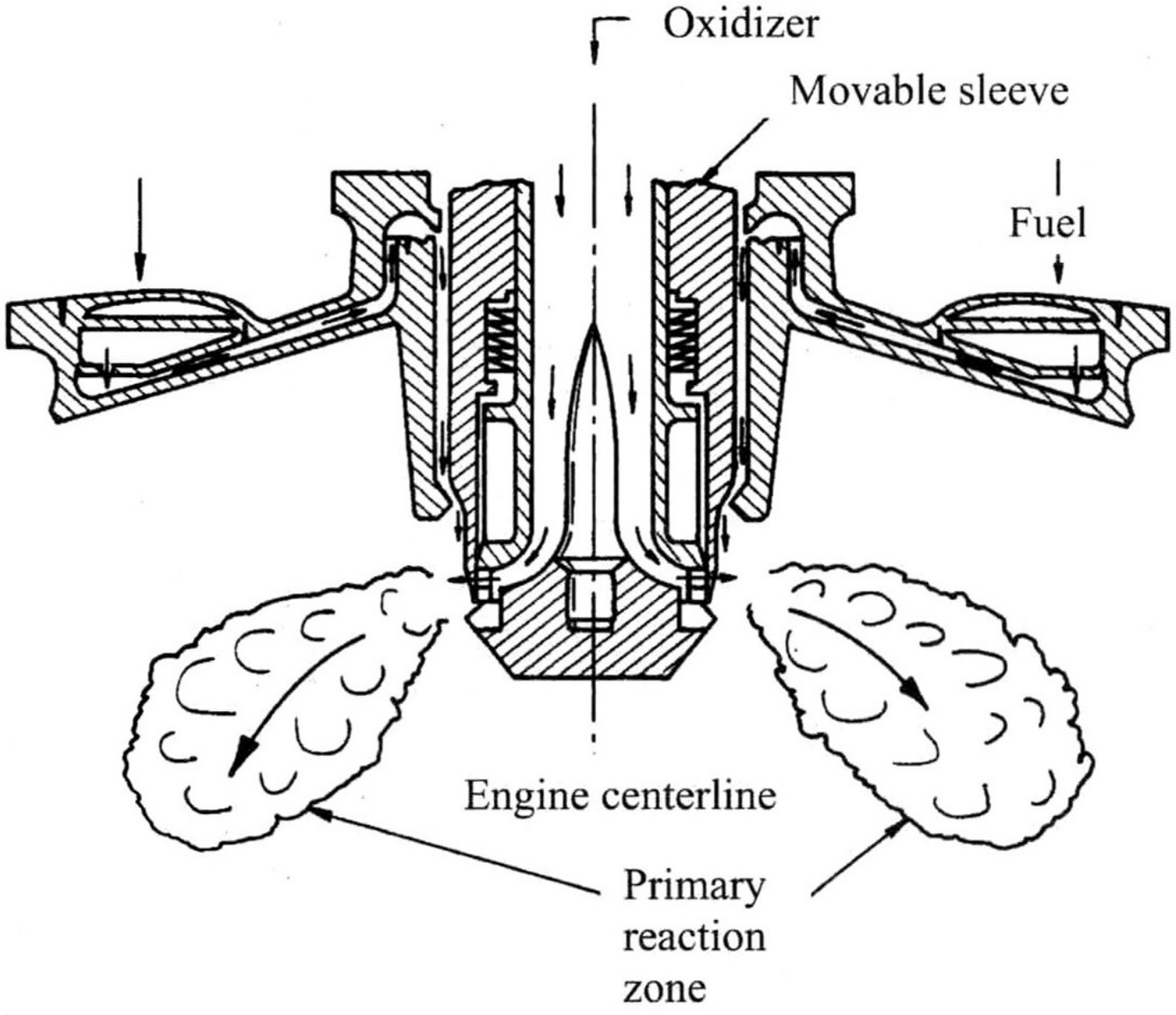


Fig. 4 Pintle injectors of throttable LRE.[3]

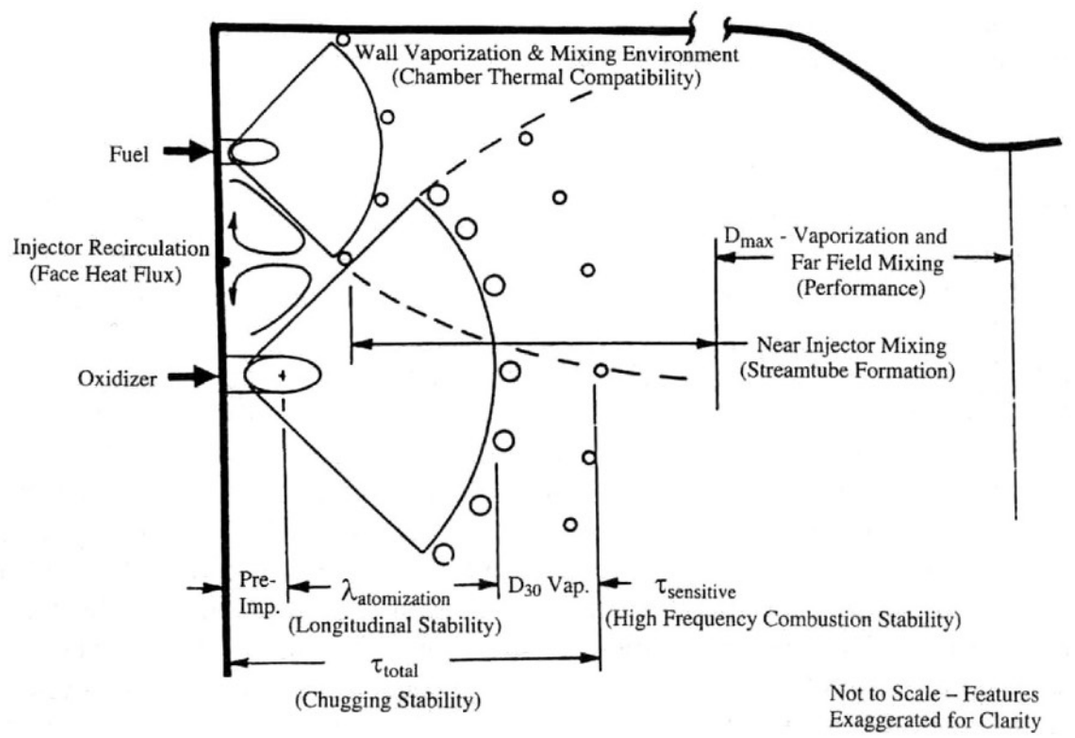


Fig. 2 Combustion processes involved in liquid rocket combustion devices.

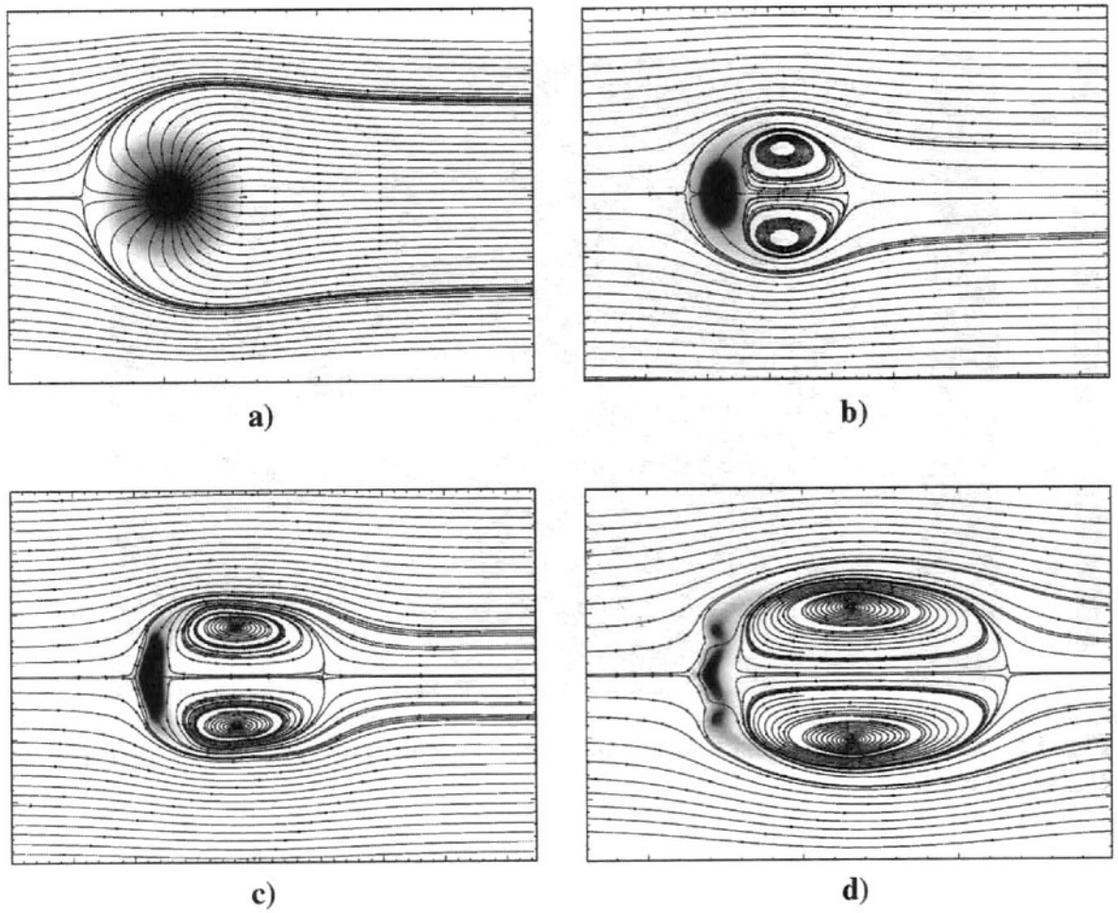


Fig. 13 LOX droplet gasification in supercritical hydrogen flow for $p = 100$ atm and $D_0 = 100$ μm: a) spherical mode, $U_\infty = 0.2$ m/s, $t = 0.61$ ms; b) deformation mode, $U_\infty = 1.5$ m/s, $t = 0.61$ ms; c) stripping mode, $U_\infty = 5$ m/s, $t = 0.17$ ms; d) breakup mode, $U_\infty = 15$ m/s, $t = 0.17$ ms.

Proposed Approach

octree for dynamic compressible LES+chem

poly mesh for thermo-elastic structural FEA

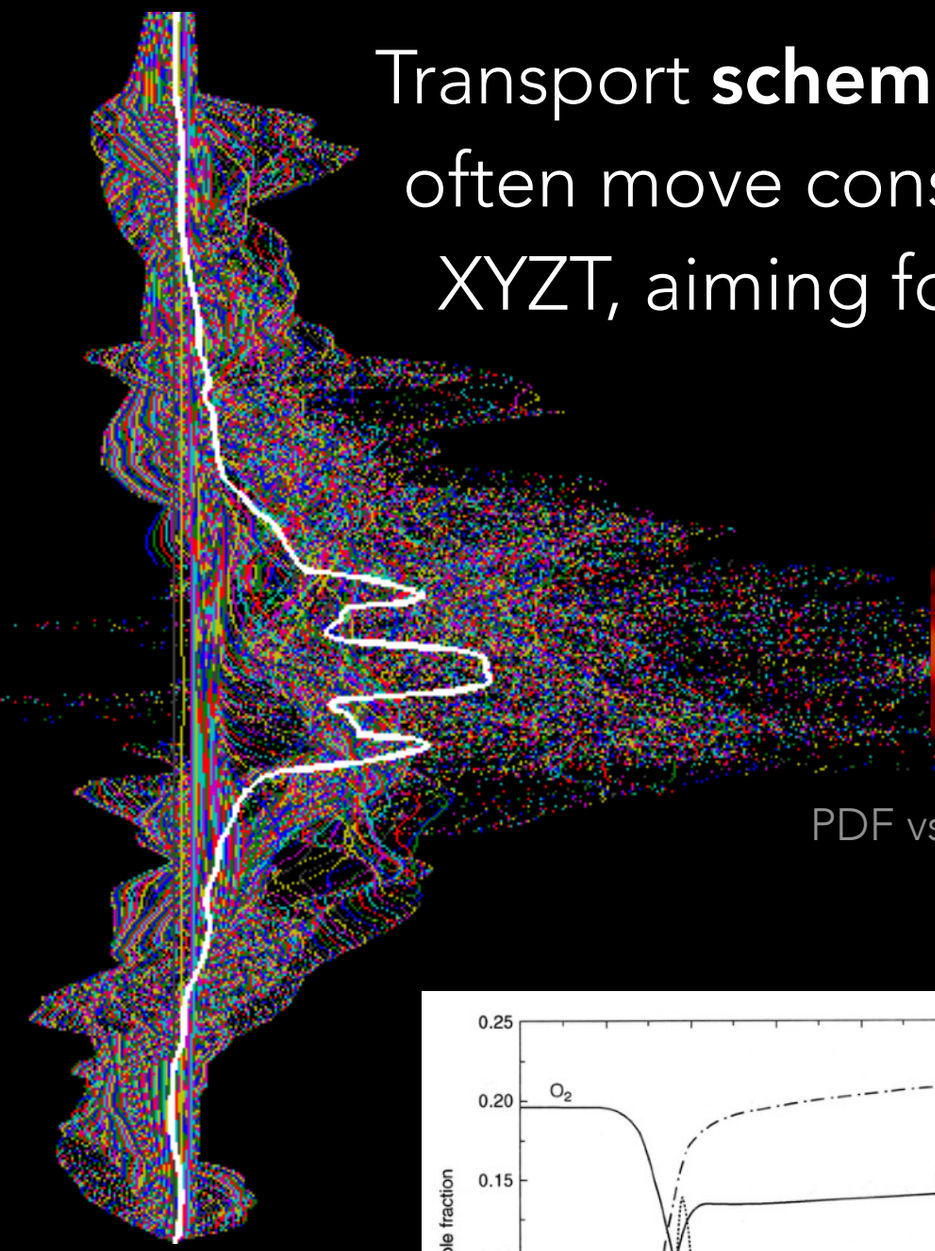
chimera prism boundary layer (FEA-FDM coupling)

multiphase LBM? Or unify FDM?

NUMERICAL PHYSICS
TRANSPORT <=> STATE

Xplicit Computing

Transport **schemes** (FDM, FEM, FVM, LBM) often move conserved quantities through XYZT, aiming for numerical consistency.



PDF vs LES?

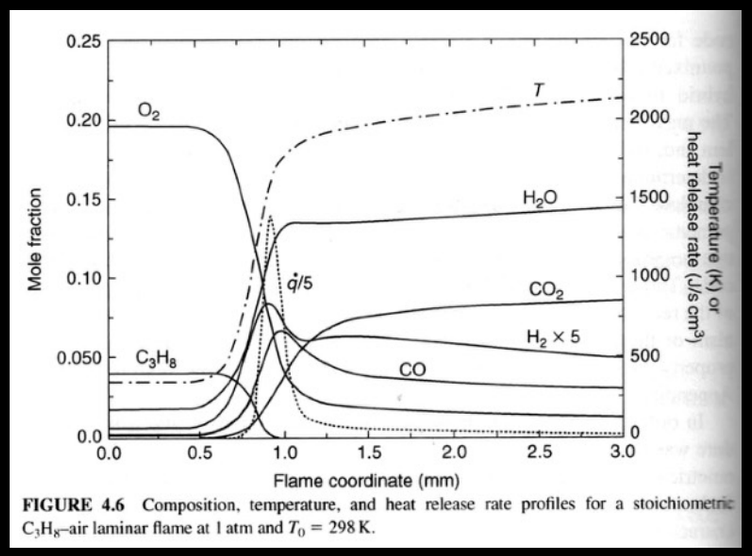
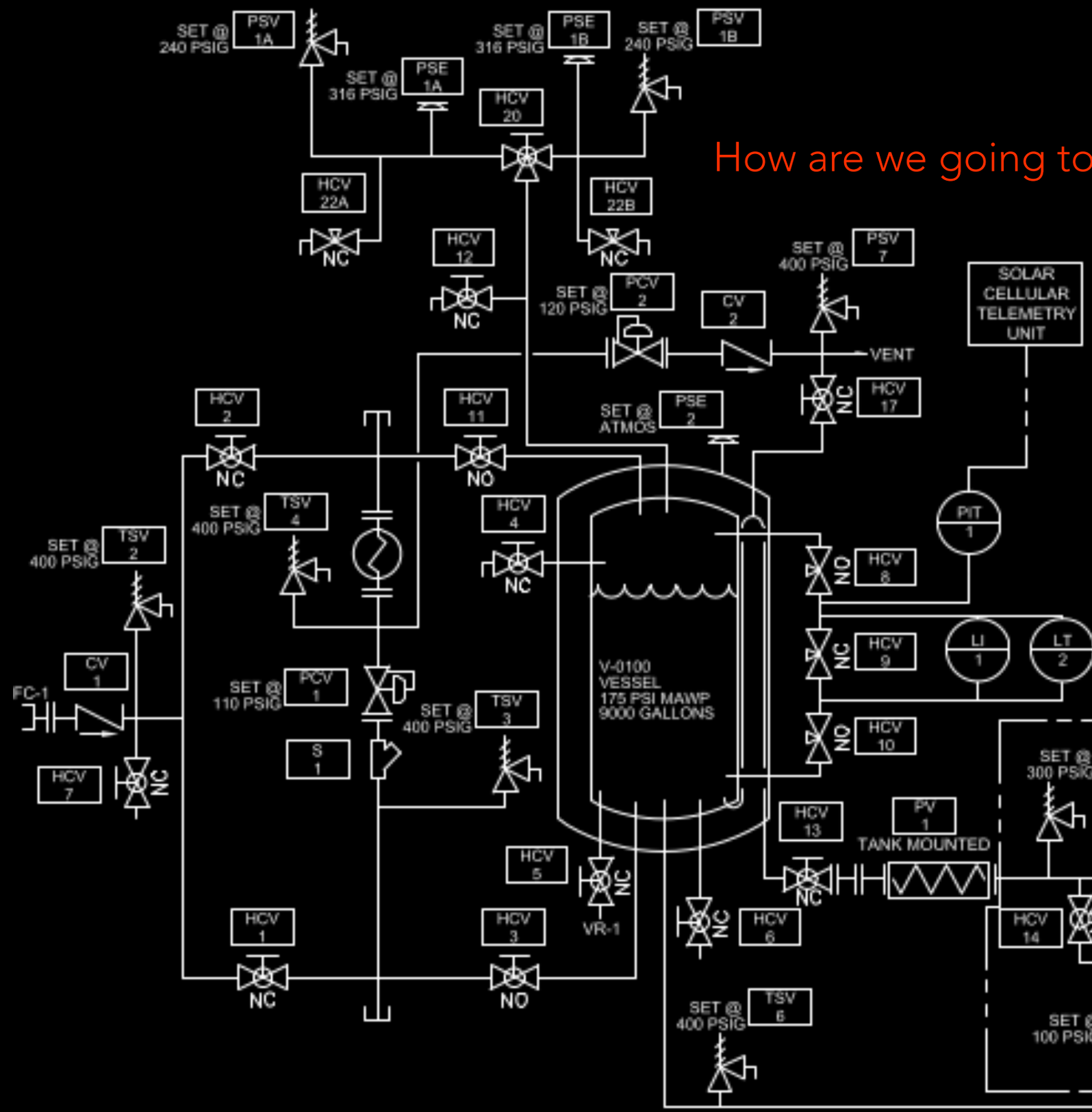


FIGURE 4.6 Composition, temperature, and heat release rate profiles for a stoichiometric C_3H_8–air laminar flame at 1 atm and $T_0 = 298$ K.

State **models** (solid, liquid, gas EOS) often describe the relations between primitive and conserved quantities.

All are built with modular **algorithms.**

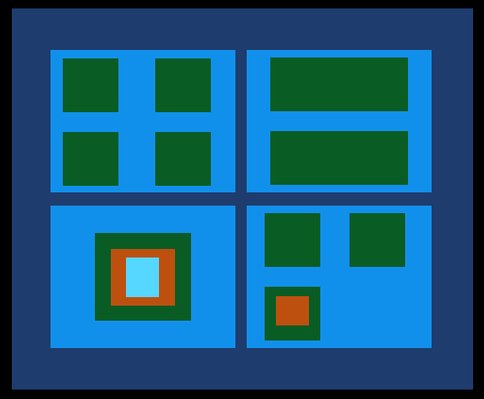
SYSTEM COMPLEXITY IS INCREASING



How are we going to keep up?

DESIGN PRINCIPLES

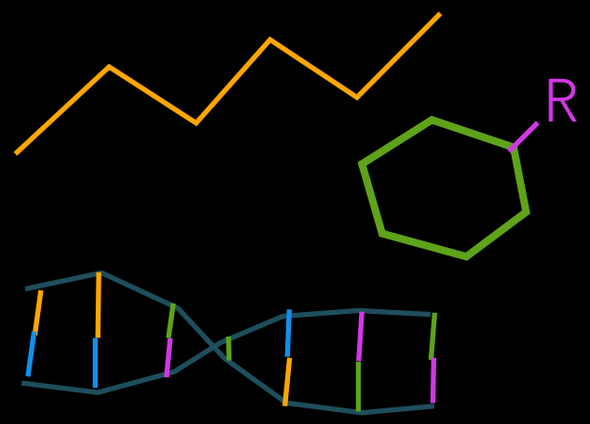
Hierarchy



Scale

Organization
Recursive Definition

Modularity



Reuse

Amplification
Functional Inheritance

Regularity



Pattern

Performance
Heterogeneous Parallelism

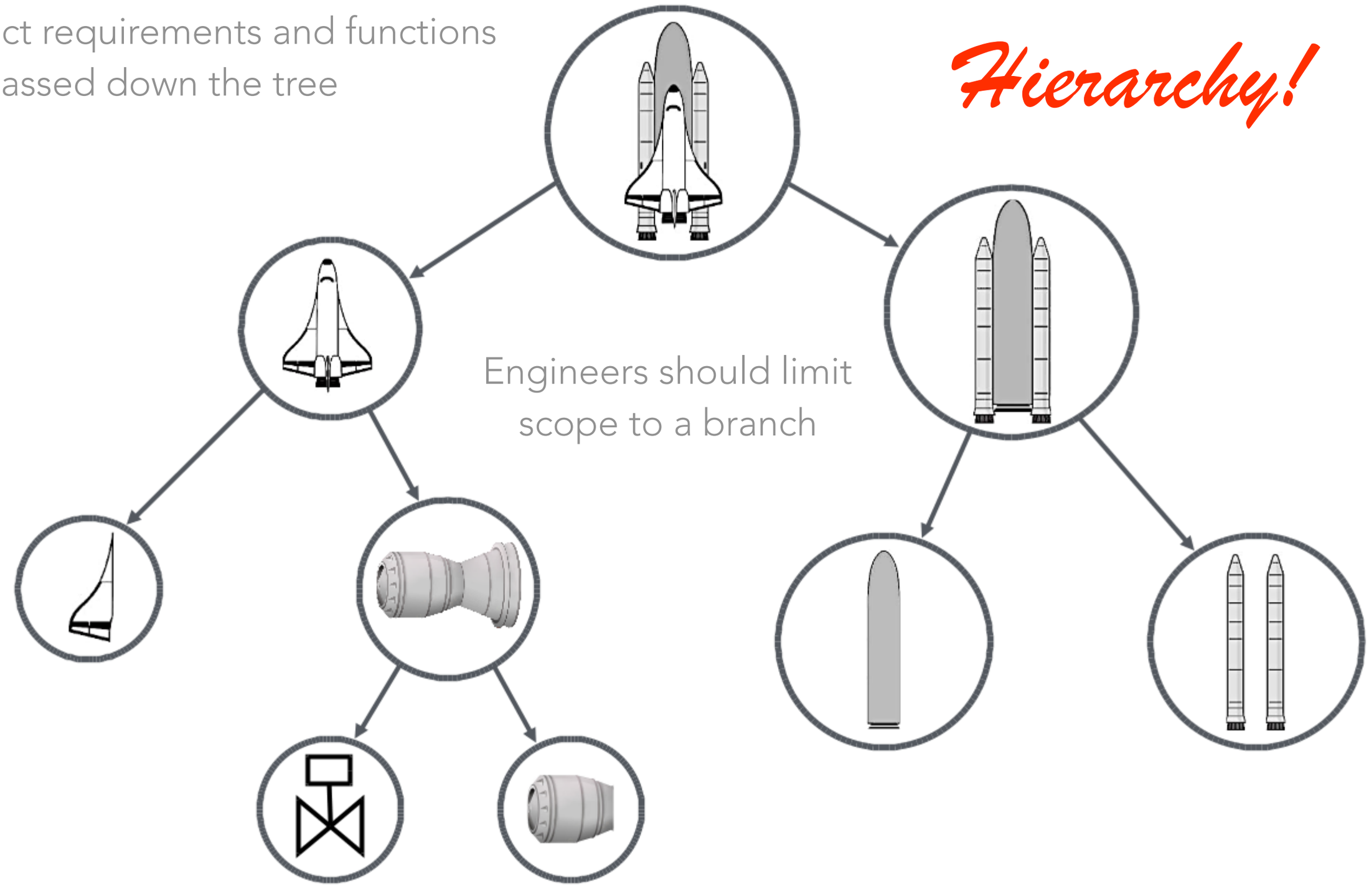
ADVANCED SYSTEMS

HIGH DEGREE OF COHERENCE WITH MODERN C++

SYSTEMS-OF-SYSTEMS

Project requirements and functions are passed down the tree

Hierarchy!



Engineers should limit scope to a branch

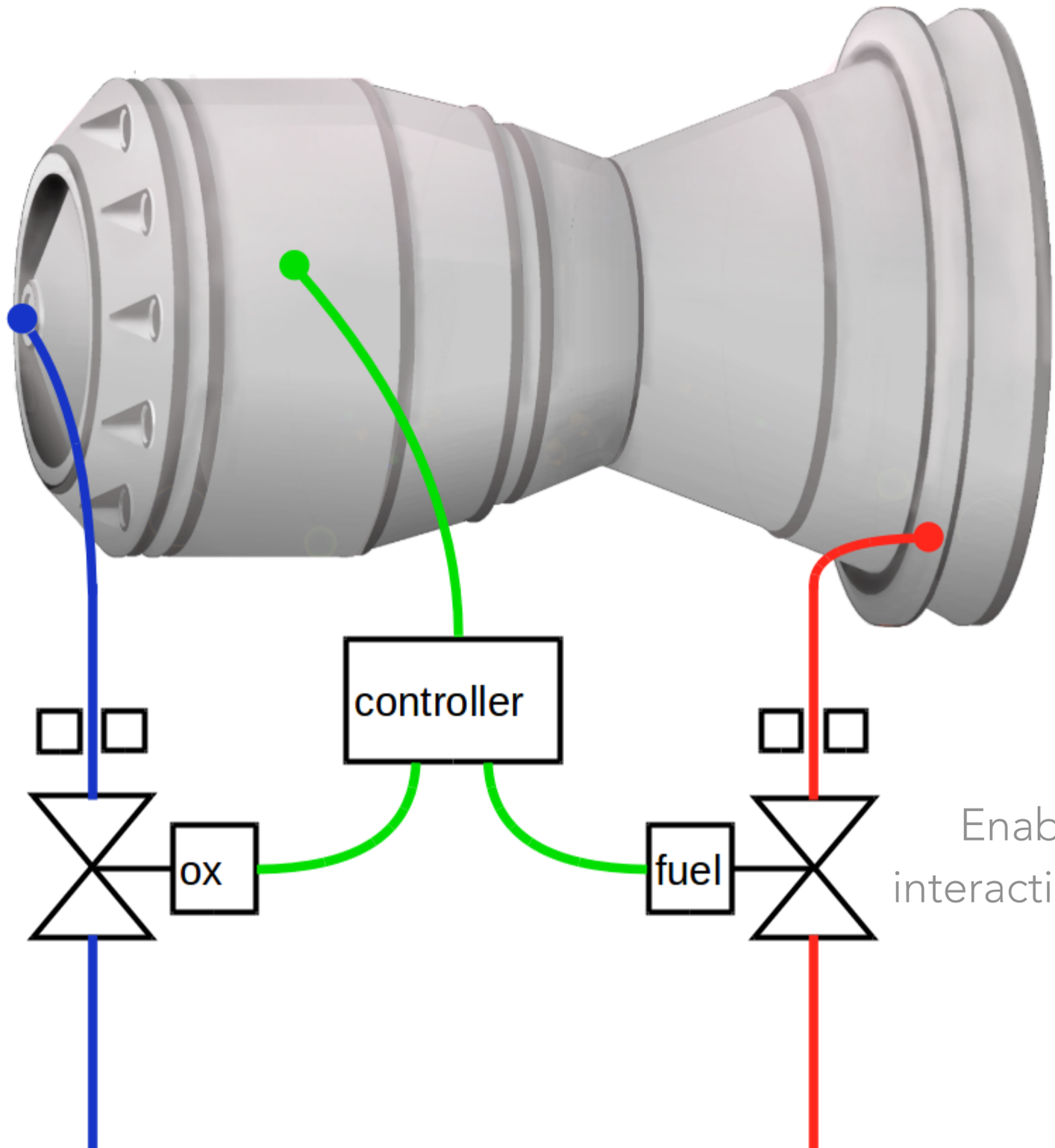
systems + algorithms = complex machines

MULTI-FIDELITY SIMULATION

WYSIWYG

in varying formats

- spatial render
- schematic
- tree
- CLI
- param editors



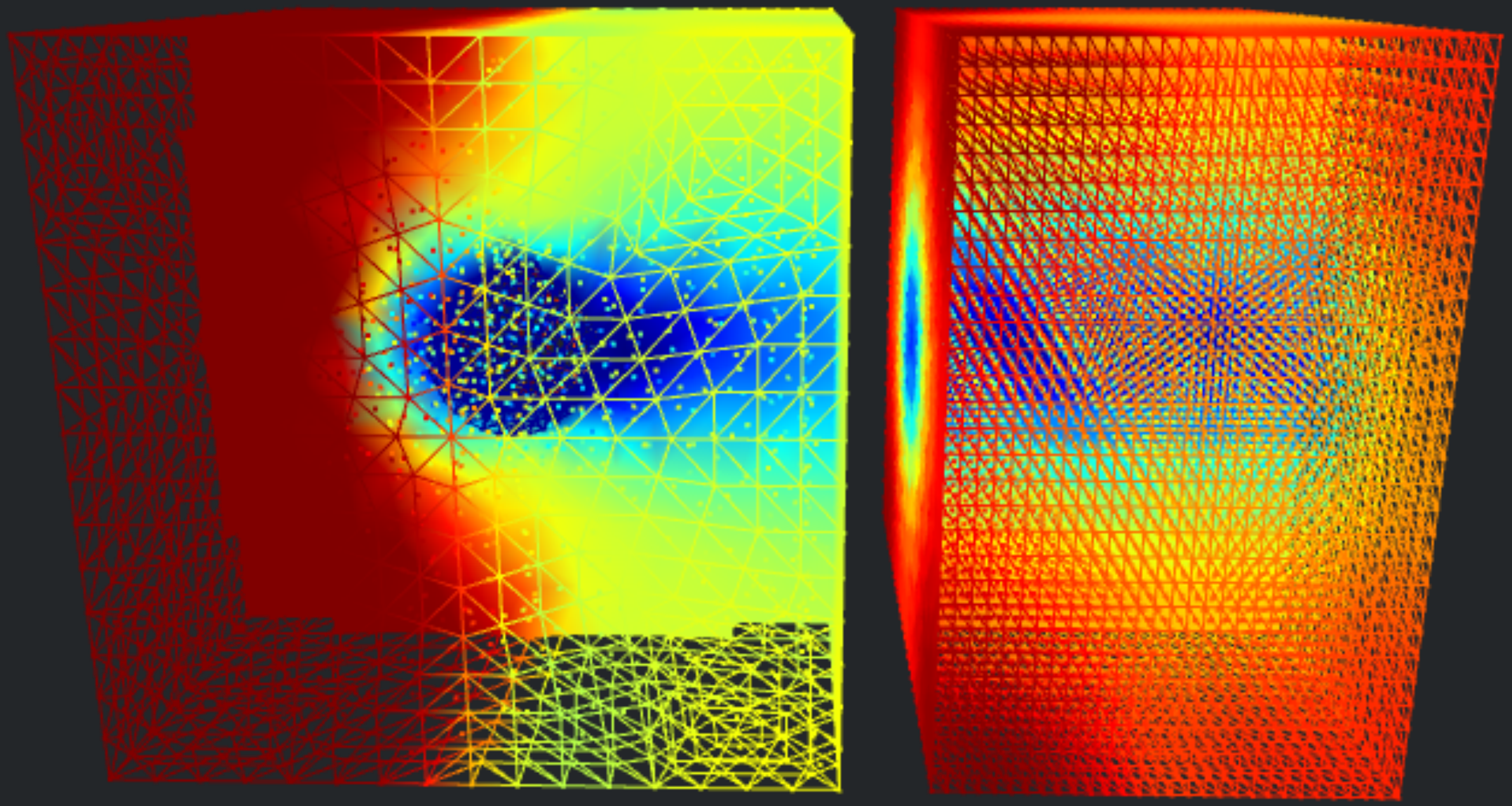
Integrated

varying systems

- 0D (surrogate)
- 1D (routing, etc)
- 2D (revolve, etc)
- 3D (full CAD)

Enable higher-level interactions and feedback

MUTUAL LINKING (FVM, NOV 2016)



unstructured mesh region <=> structured grid region

DIABLOS IN THE DETAILS

Structured

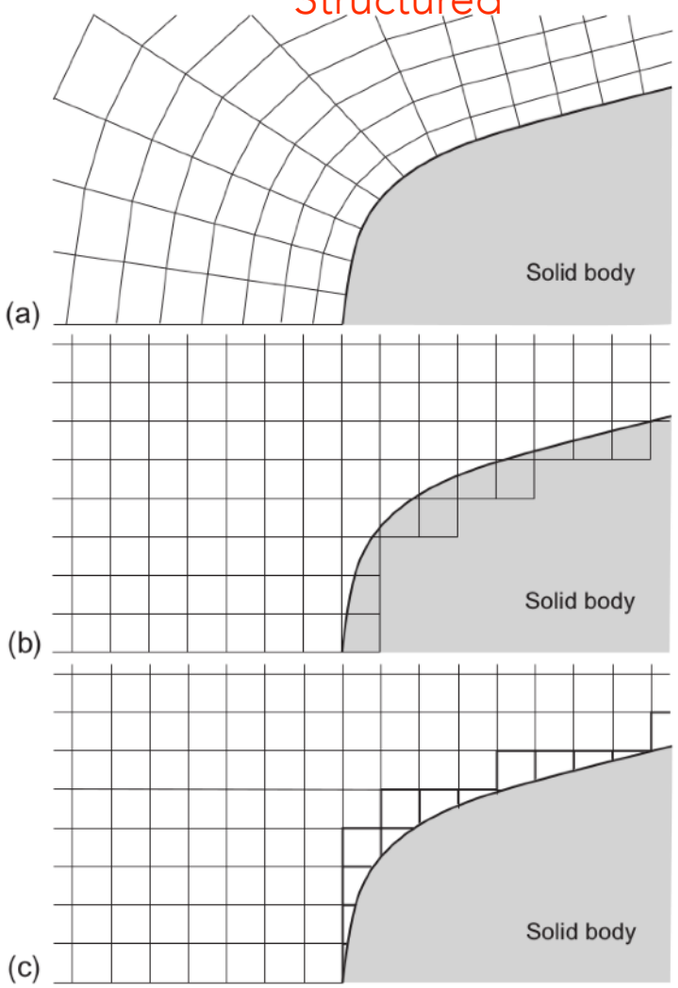
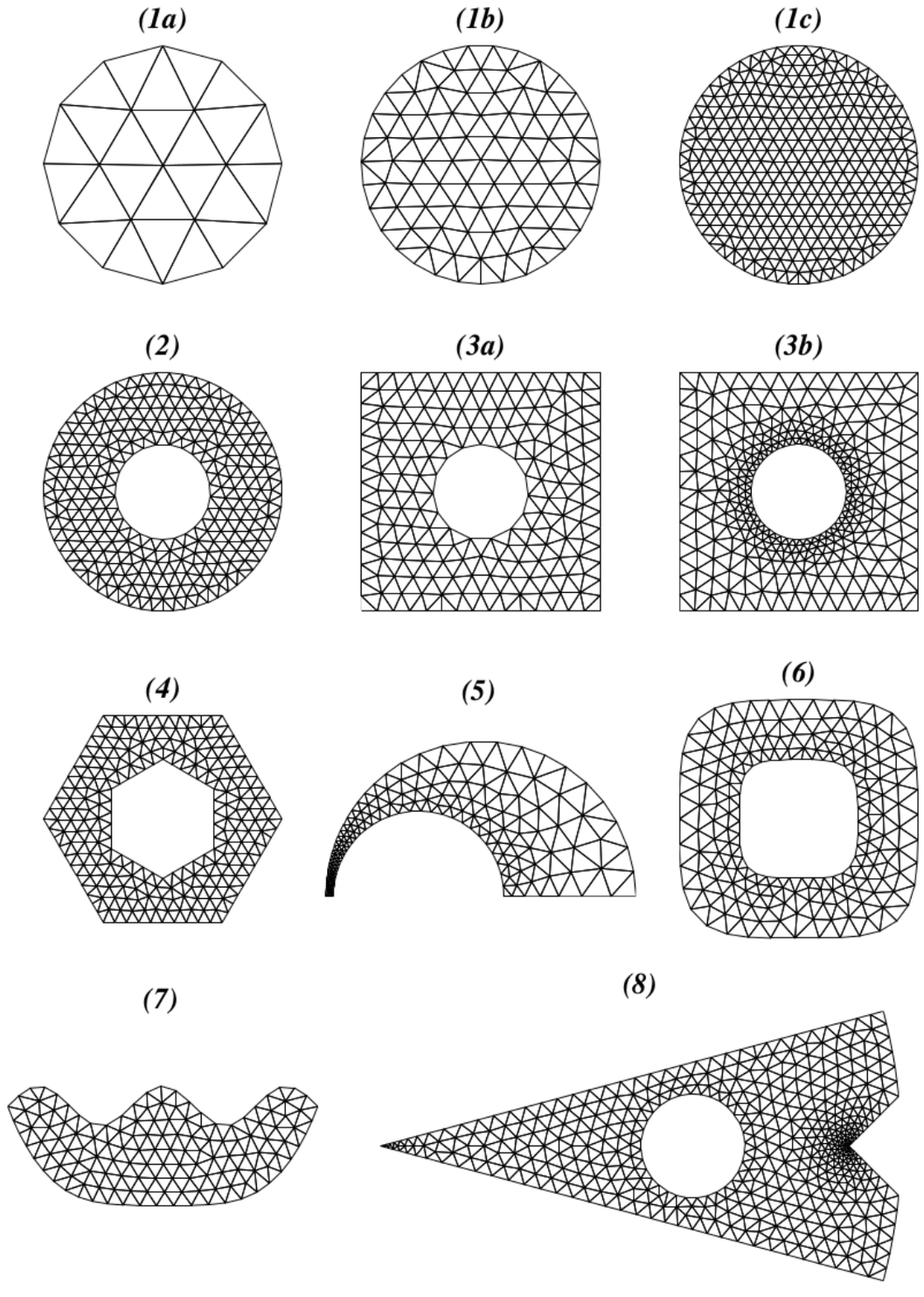


Figure 3.1 2-D body-fitted grid (a), Cartesian grid with immersed boundary (b), Cartesian grid with polygonal boundary cells (c) near a solid body. Boundary cells are marked by a thick line in (c).

Unstructured



Hardware & software advances have enabled key HPC++ benefits

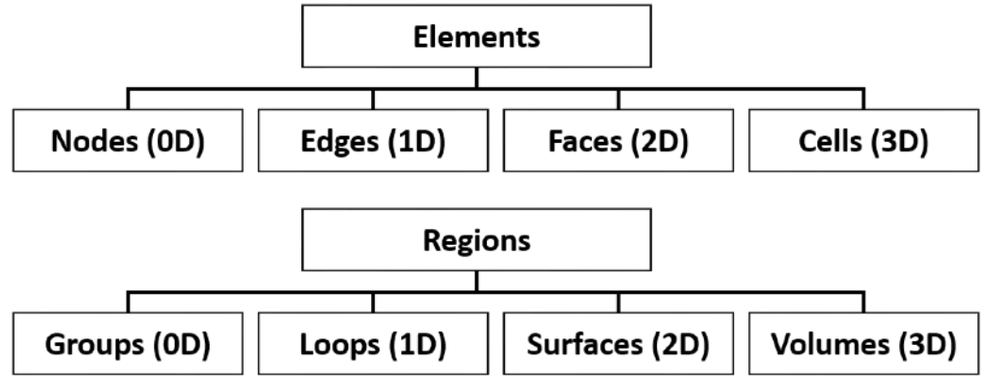
Appreciate C++14 and its STL, but take machinery approach to data parallelism/IO

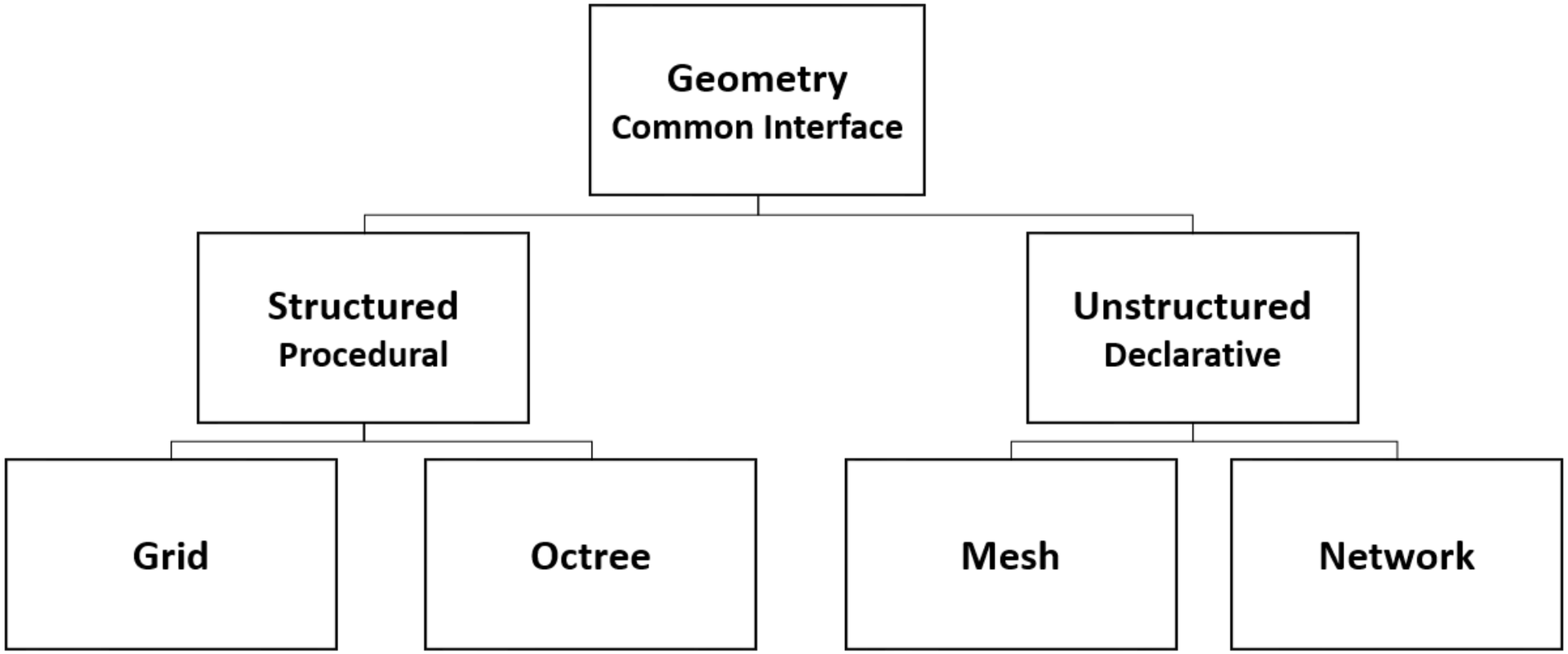
[5]

COMPUTABLE GEOMETRY INHERITANCE

- **CPU: HIGH-LEVEL C++ POLYMORPHISM** (EASY, BUT STATIC!)
- **GPU: EMULATE C++ USING OPENCL JIT COMPILER** (HARD, BUT DYNAMIC!)

Modularity!





EXAMPLE C++ GEOMETRY SPECIALIZATION

```
//base class provides unified programming interface to common functions
class Geometry{
  Elements nodes;//perhaps we loaded a mesh that defined vertex positions
  public:
  //base function can return declared element memory (e.g. unstructured)
  virtual vec3 position(int n){return nodes.center[n]};}
}

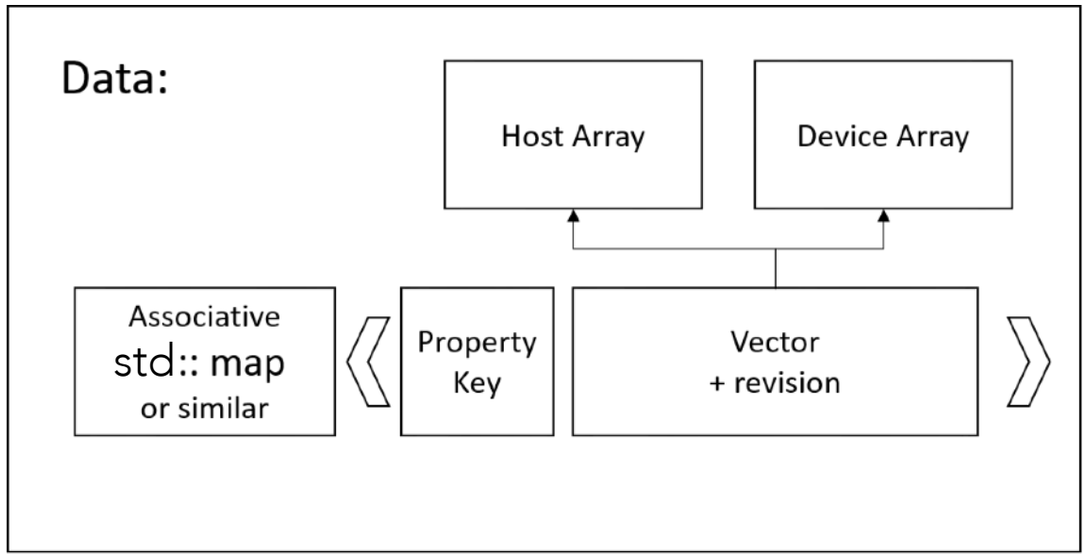
//specialized derived class updates v-table with new function pointers
class Grid : public Geometry{
  ivec3 size;
  //private utility functions specific to this specialized class (and its derived types)
  vec3 position(ivec3 loc){return Extent::min + Extent::delta()*loc/vec3(size-1);};//equal spacing
  ivec3 location(int n){
    int Ak = size.x*size.y;//find z-base
    int k = n/Ak;//fold z
    int r = n - k*Ak;//find remainder
    int j = r/size.x;//fold y
    return ivec3(r - j*size.x, j, k);}//fold x and return ijk coord
  public:
  Grid(ivec3 sz) : Geometry(), size(sz){;};//constructor assigns any additional input params
  //function override returns procedural data (e.g. structured)
  virtual vec3 position(int n){return position(location(n));};
}
```

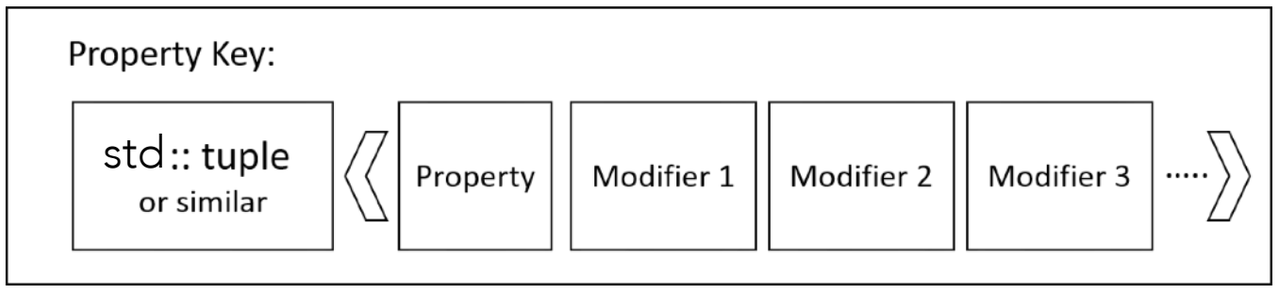
XCOMPUTE HETEROGENEOUS DATA

example Properties:

AnyProperty
Position
Displacement
SignedDistance
FeatureSize
Curvature
MeshQuality
CellArea
Tolerance
Energy
KineticEnergy
TurbulentKineticEnergy
Enthalpy
InternalEnergy
GibbsEnergy
Pressure
Temperature
SpeedOfSound
Gamma
CFL
TimeStep
Density
MomentumX
MomentumY
…

CPU thread traverses the data map in log time
returns coalesced heterogeneous vector ref for a PK





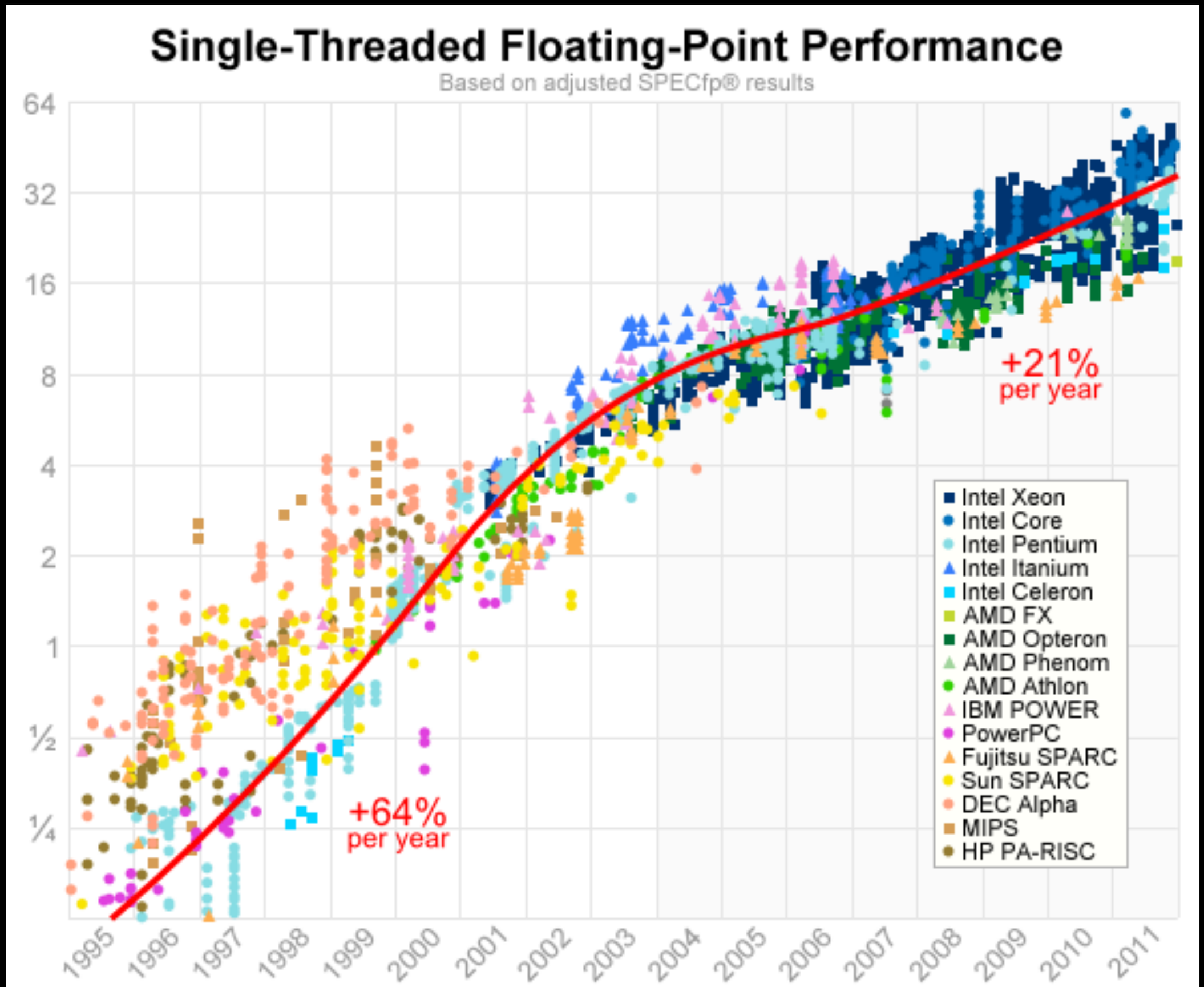
Property and Modifier are evaluated using
global pointer comparison for speed and uniqueness

Regularity!

example Modifiers:

AnyModifier
Residual
Max
Min
Mean
Median
Mode
StDev
RMS
Reference
Characteristic
Standard
Ratio
Flux
Divergence
Curl
Dt
Dx
Dy
Dz
Delta
Neutral
Minus
Plus
…

PROCESSOR SCALING IN TIME — AND SPACE



2018:
~5GHz @ 9nm

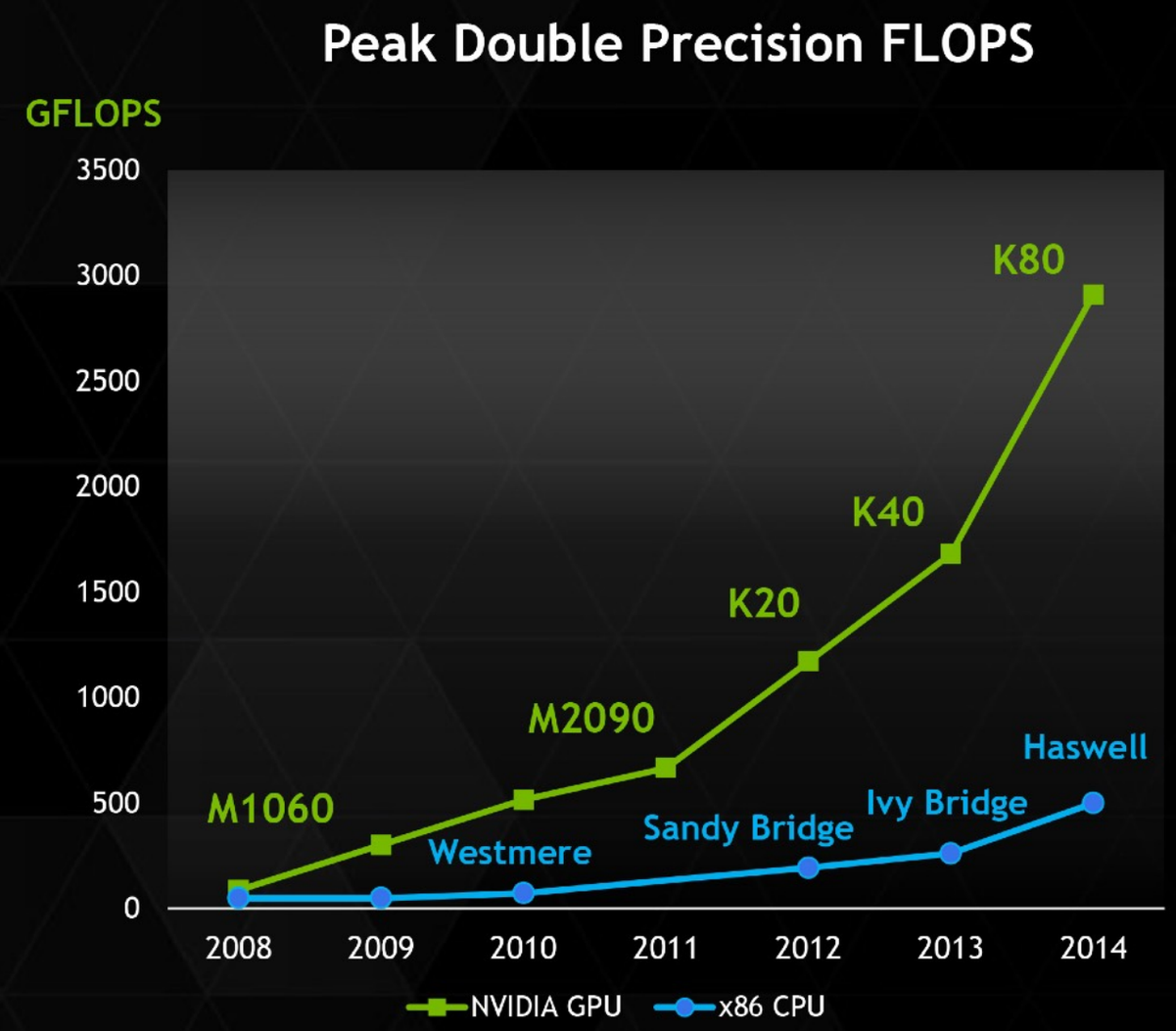
5x faster
100x smaller

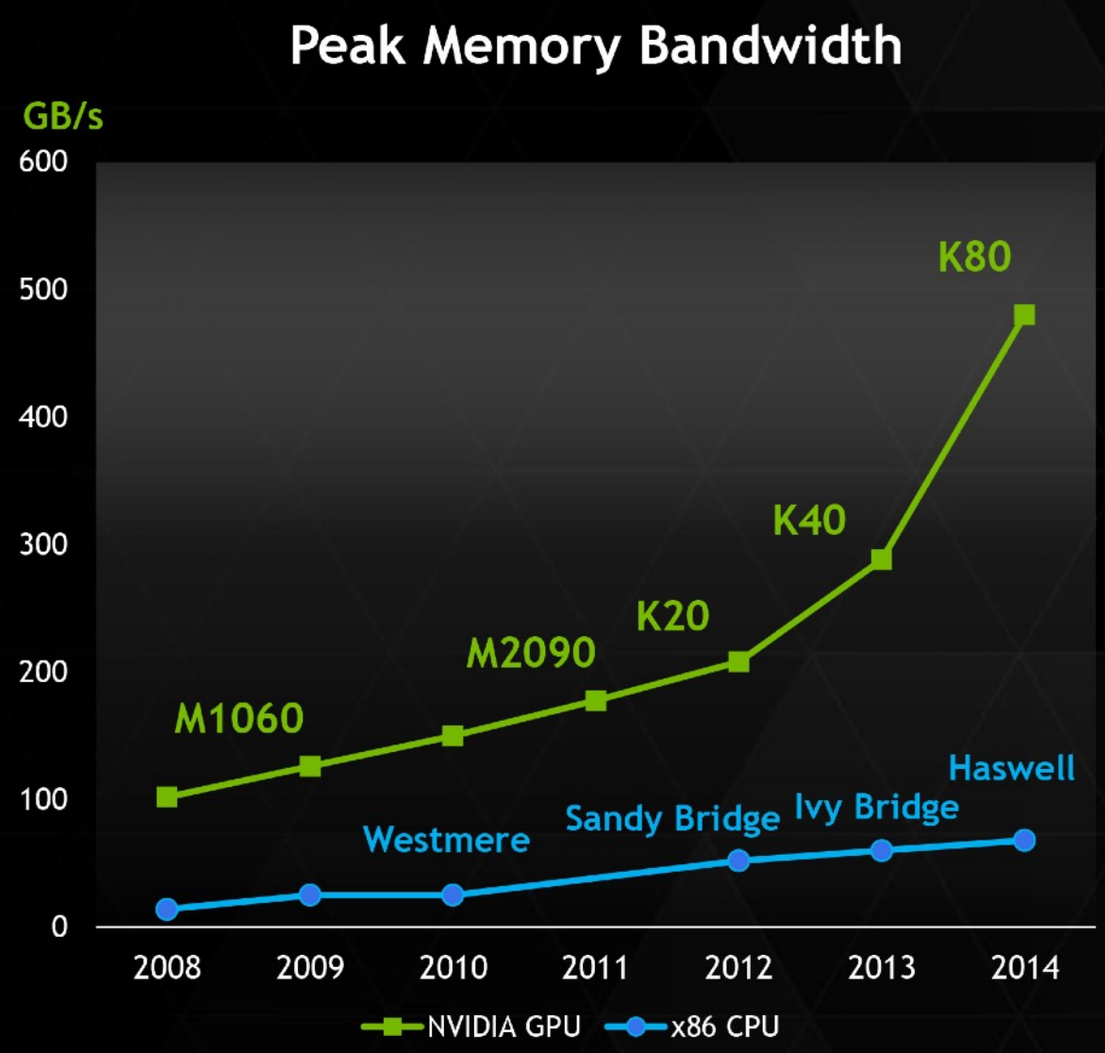
2004:
~1GHz @ 90nm

PARADIGM SHIFT TO MASSIVE **PARALLELISM**

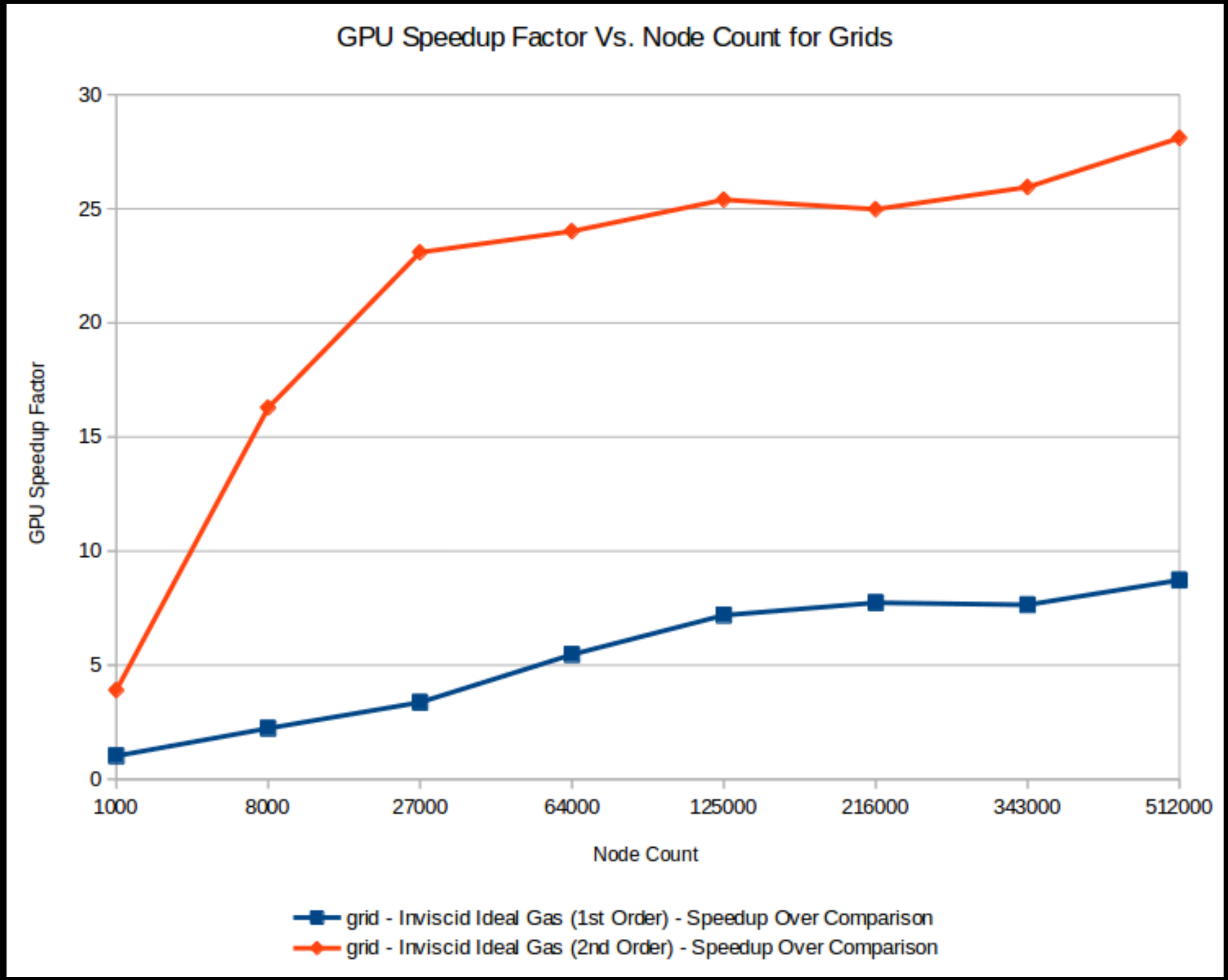
MULTI-CPU, GPU, TCU, FPGA, ASIC...

Performance Lead Continues to Grow



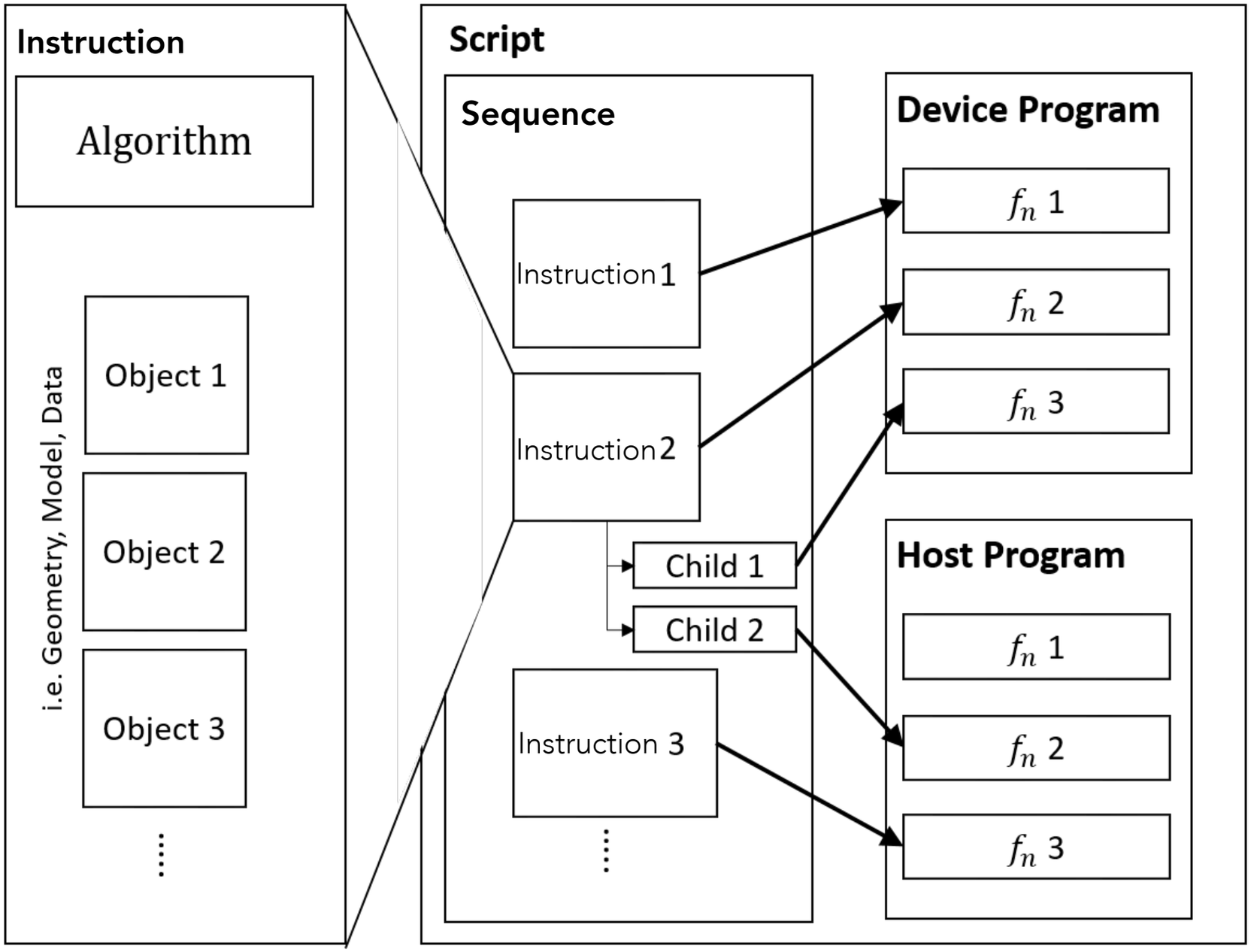


RESULTING XCOMPUTE GPU SPEED-UP (FVM JAN 2017)



[8]

KERNEL GENERATION



XCOMPUTE ALGORITHM EXAMPLE - SPEED OF SOUND

```cpp
IdealGasSpeedOfSound::IdealGasSpeedOfSound()  : Algorithm("Ideal Gas Speed Of Sound Calculation",
                                                          {Temperature},
                                                          {SpeedOfSound},
                                                          {Gamma, GasConstant}){
    requirements.insert<std::reference_wrapper<DATA>>("Data for the algorithm to operate on");
    requirements.insert<DATA>("Data to get the constant gamma");
}
```

Constructor:

- req'd data params
- req'd arg types

```cpp
void IdealGasSpeedOfSound::operator()(const heterogeneous_map<Arguments>& args) {
    auto& data = args.get<std::reference_wrapper<DATA>>().get();
    auto& constants = args.get<DATA>();

    VECTOR& a = data(SpeedOfSound);
    VECTOR& T = data(Temperature);
    double gamma = constants(Gamma)(0);
    double R = constants(GasConstant)(0);

#pragma omp parallel for
    for (INDEX i = 0; i < a.rows(); i++)
        a(i) = std::sqrt(gamma * R * T(i));

}
```

CPU Function:

- get data params
- define C++ code

GPU Code:

```cpp
string IdealGasSpeedOfSound::gpuCode(const heterogeneous_map<Arguments>& args) const {
    string mass_name = kernel_builder::arrayArgName(Mass);
    string T = kernel_builder::arrayArgName(Temperature);
    string sos_name = kernel_builder::arrayArgName(SpeedOfSound);
    string gamma_name = kernel_builder::constArgName(Gamma);
    string R = kernel_builder::constArgName(GasConstant);

    string code = "\tconst int nodeIndex = get_global_id(0);\n"
                  "\t" + sos_name + "[nodeIndex] = sqrt(" + gamma_name + " * " + T + "[nodeIndex] * " + R + ");\n";


    return code;
}
```

- get data params
- define OpenCL code

XCOMPUTE ALGORITHM EXAMPLE - GREEN GAUSS GRADIENT
1. SPECIFY REQUIREMENTS

```cpp
#include "green_gauss_gradient.hpp"

using namespace Properties;
using namespace Modifiers;

GreenGaussGradient::GreenGaussGradient() : Algorithm("Green-Gauss Gradient",
                                {AnyProperty},
                                {{AnyProperty, Gradient}}){
    requirements.insert<std::reference_wrapper<DATA>>("Data for the algorithm to operate on");
    requirements.insert<std::reference_wrapper<const GEOMETRY>>("Geometry to compute upon");
    requirements.insert<PropertyKey>("Property key indicating what this gradient is acting on");
}


bool GreenGaussGradient::add(Instruction& instruction, std::reference_wrapper<const GEOMETRY> geometry, bool overwrite
    Instruction boundaryCall(make_shared<GreenGaussBoundary>());
    boundaryCall.matchState(instruction);
    Algorithm::add(instruction, geometry, overwrite);
    boundaryCall.algorithm->add(boundaryCall, geometry.get().surfaces, overwrite);
    instruction.post->push_back(boundaryCall);
    return true;
}

```

XCOMPUTE ALGORITHM EXAMPLE - GREEN GAUSS GRADIENT

2. DEFINE C++ CODE

```cpp

void GreenGaussGradient::operator()(const heterogeneous_map<Arguments>& args) {
    auto& data = args.get<std::reference_wrapper<DATA>>().get();
    auto& geometry = args.get<std::reference_wrapper<const GEOMETRY>>().get();

    auto val_pk = PropertyKey(args.get<PropertyKey>().property());
    auto grad_pk = PropertyKey(val_pk.property(), Gradient);

    VECTOR& values = data(val_pk);
    VECTOR& gradient = data(grad_pk);

    #pragma omp parallel for
    for (int n = 0; n < geometry.nNode(); n++) {
        bool zeroOut = true;
        for (int i = 0; i < geometry.nNeighbor(n); i++) {
            FACE face = geometry.getInteriorFace(n, i);
            if(face.empty())
                continue;

            int nt = face.indices[0];
            int ns = face.indices[1];
            double volume = geometry.volume(nt);
            if (approxEqual(volume,0.))
                continue;

            // Compute the interface value by taking an average
            double interfaceValue = .5 * (values(ns) + values(nt));

            // If this is the first connection, zero out the gradient vector
            if (zeroOut) {
                gradient.row(nt) = VECTOR::Zero(1, geometry.nDim);
                zeroOut = false;
            }
            // Compute the gradient using Gauss' theorem
            gradient(nt,0) += interfaceValue * face.area.x / volume;
            gradient(nt,1) += interfaceValue * face.area.y / volume;
            gradient(nt,2) += interfaceValue * face.area.z / volume;
        }
    }

}
```

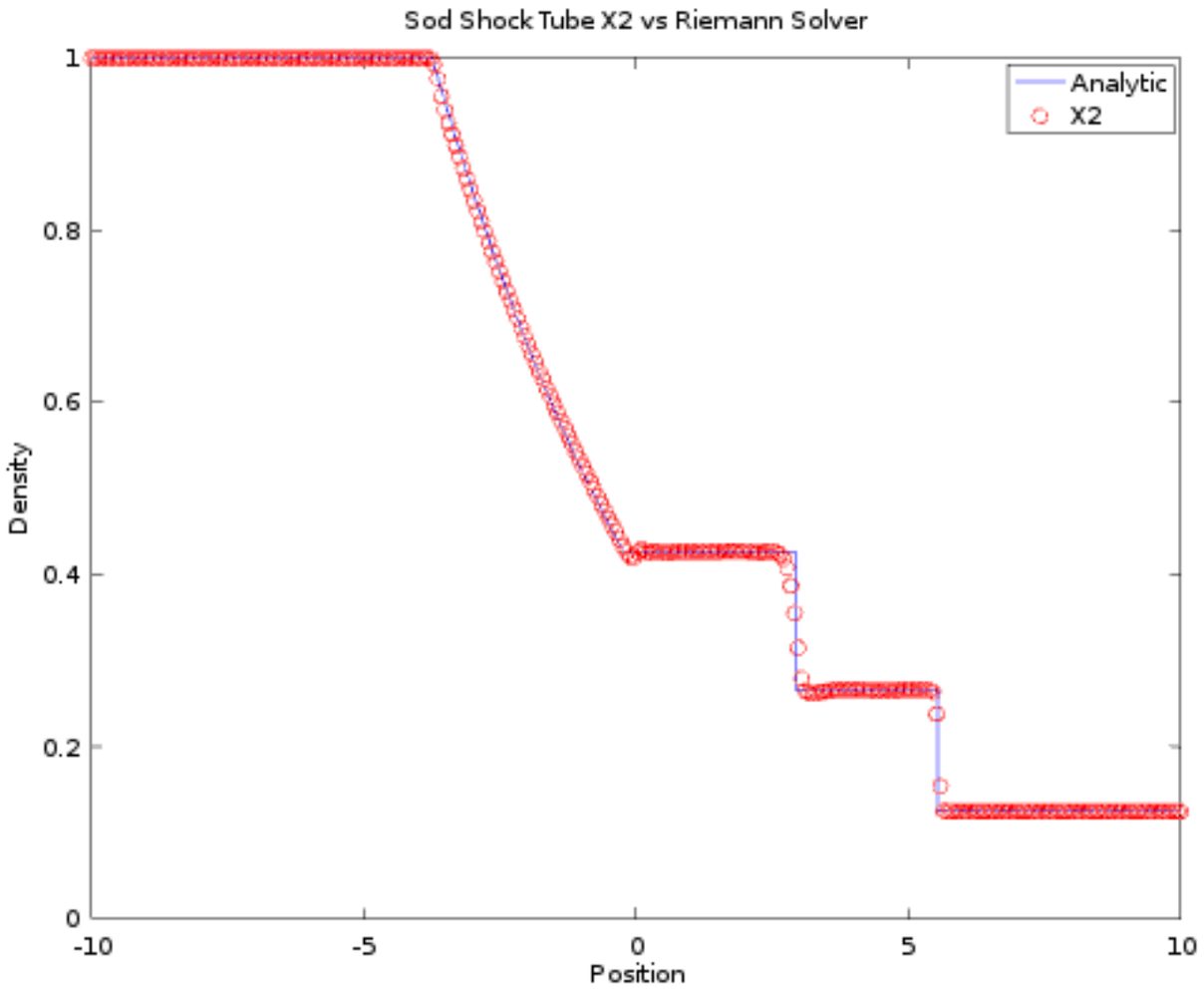
XCOMPUTE ALGORITHM EXAMPLE - GREEN GAUSS GRADIENT

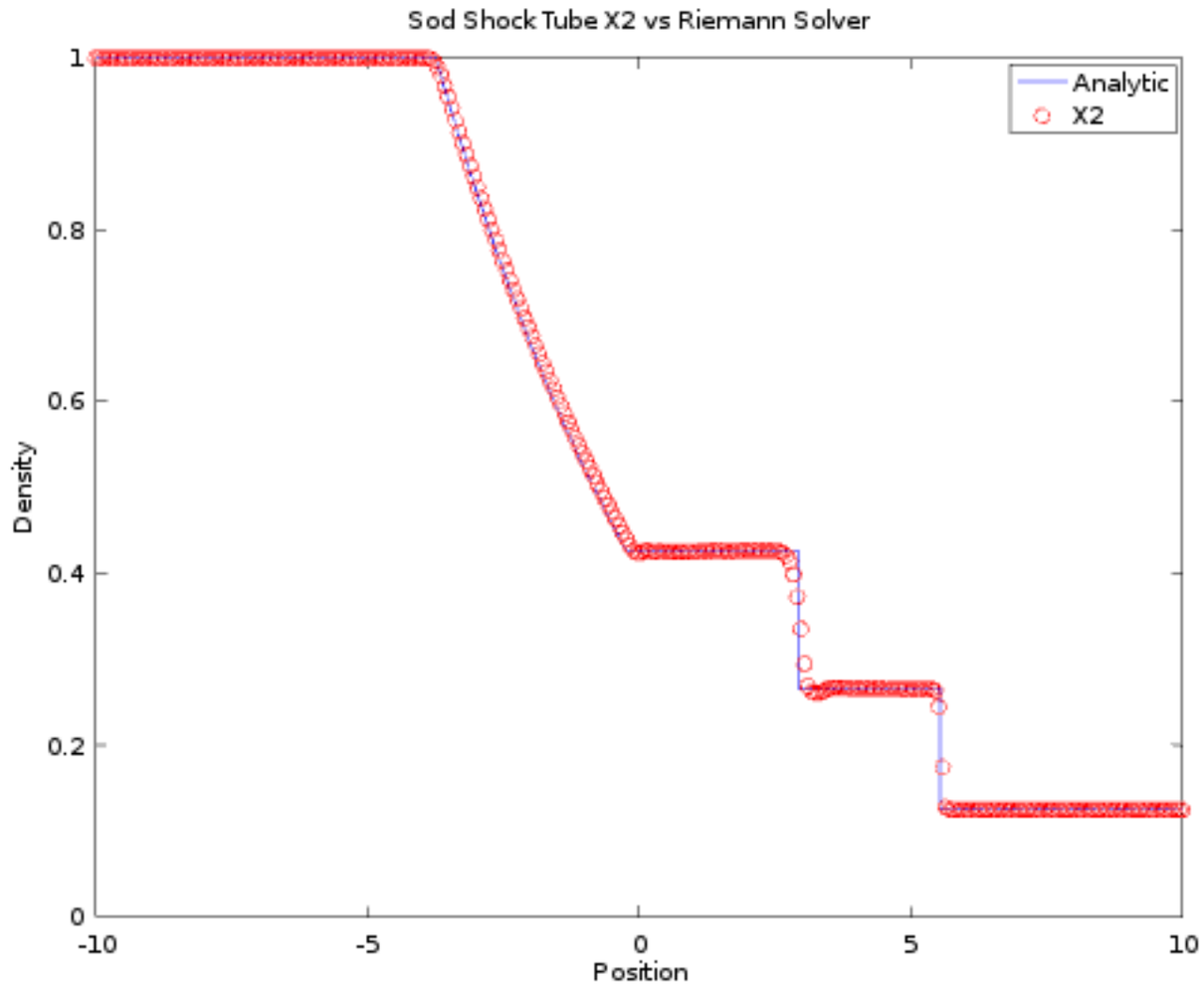
3. DEFINE OPENCL CODE

```
string GreenGaussGradient::gpuCode(const heterogeneous_map<Arguments>& args) const {
    string value_name = kernel_builder::arrayArgName(PropertyKey(AnyProperty));
    string grad_name = kernel_builder::arrayArgName(PropertyKey(AnyProperty, Gradient));

    string code = "\tconst int nodeIndex = get_global_id(0);\n"
            "\tbool zeroOut = true;\n"
            "\tfor (int i = 0; i < nNeighbor(geometry, nodeIndex); i++) {\n"
            "\t\tstruct Face face = getInteriorFace(geometry, nodeIndex, i);\n"
            "\t\tif(isEmpty(face))\n"
            "\t\t\tcontinue;\n"
            "\n"
            "\t\tint nt = face.indices[0];\n"
            "\t\tint ns = face.indices[1];\n"
            "\t\tdouble vol = volume(geometry, nt);\n"
            "\t\tif (vol<1.e-15) continue;\n"
            "\n"
            "\t\t// Compute the interface value by taking an average\n"
            "\t\tdouble interfaceValue = .5 * (" + value_name + "[ns] + " + value_name + "[nt]);\n"
            "\n"
            "\t\t// If this is the first connection, zero out the gradient vector\n"
            "\t\tif (zeroOut) {\n"
            "\t\t\t" + grad_name + "[3*nt] = 0;\n"
            "\t\t\t" + grad_name + "[3*nt + 1] = 0;\n"
            "\t\t\t" + grad_name + "[3*nt + 2] = 0;\n"
            "\t\t\tzeroOut = false;\n"
            "\t\t}\n"
            "\t\t// Compute the gradient using Gauss' theorem\n"
            "\t\t" + grad_name + "[3*nt] += interfaceValue * face.area.x / vol;\n"
            "\t\t" + grad_name + "[3*nt + 1] += interfaceValue * face.area.y / vol;\n"
            "\t\t" + grad_name + "[3*nt + 2] += interfaceValue * face.area.z / vol;\n"
            "\t}";
    return code;
}
```

EARLY XCOMPUTE CFD VALIDATION
SOD SHOCK TUBE - AUSM AND ROE





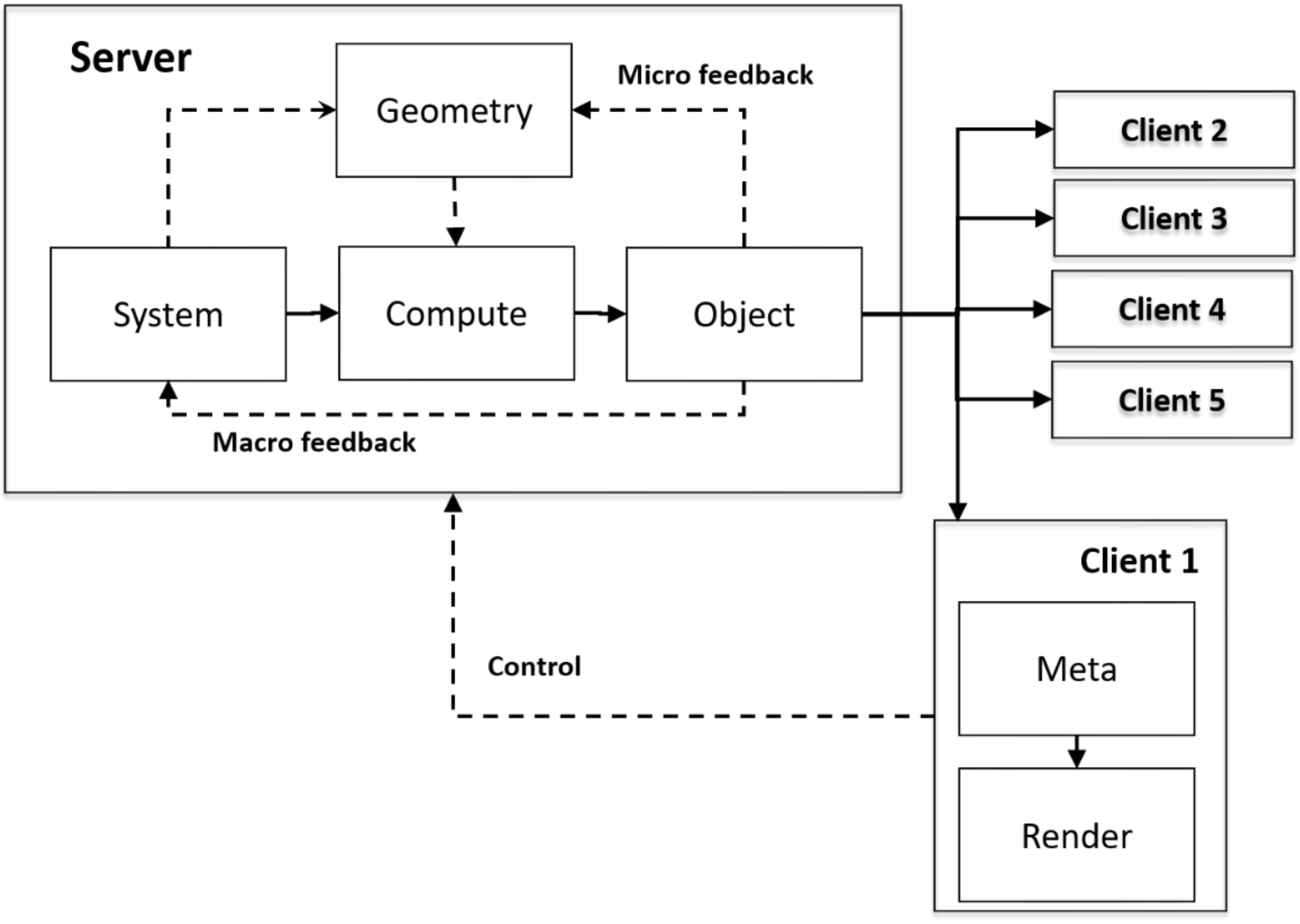
Other CFD Tests Include:

- Couette Flow
- Laminar Flat Plate
- Turbulent Flat Plate
- Backwards Step

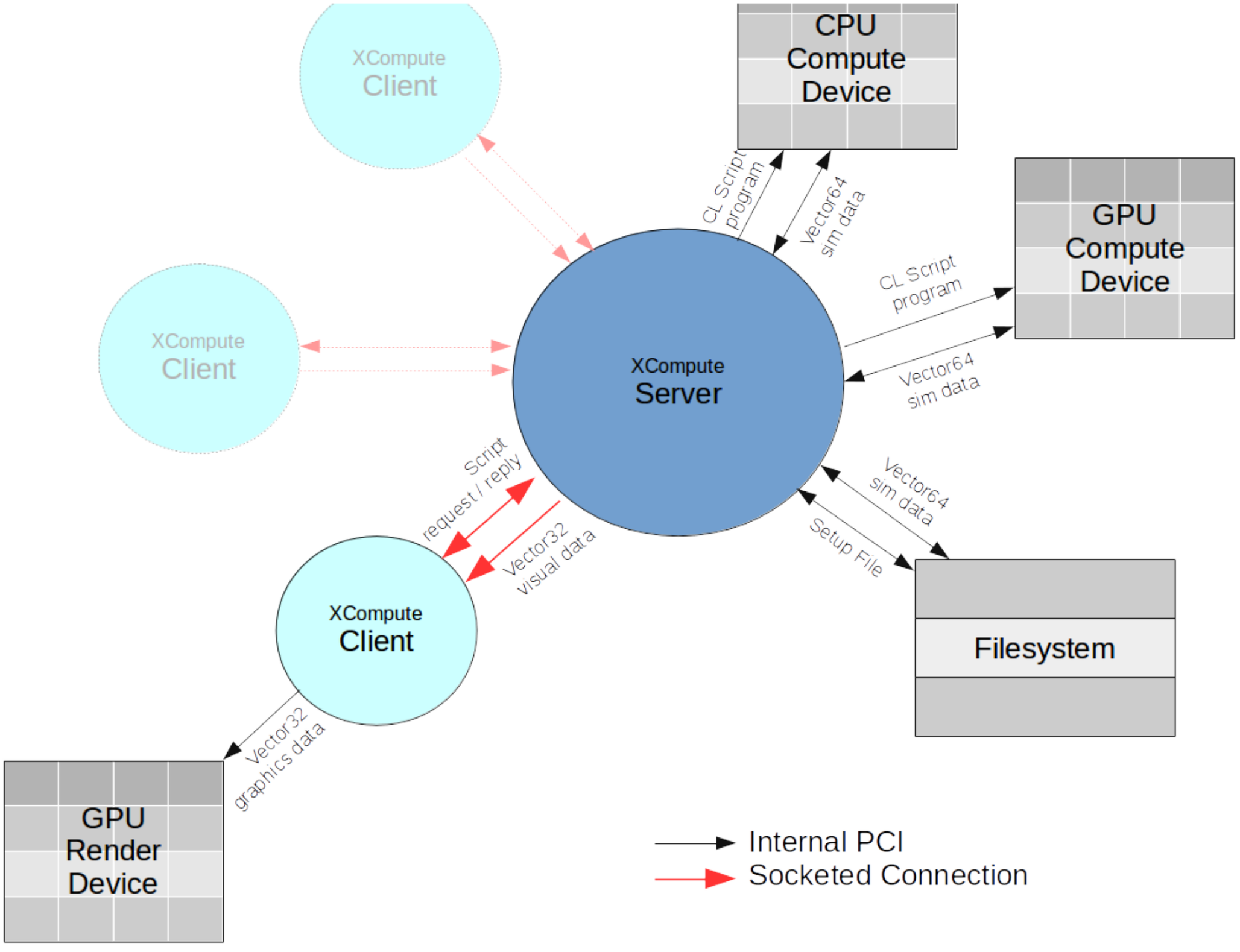
Lab Tests (external contracted):

- Laminar Pipe Flow
- Turbulent Pipe Flow

XCOMPUTE MESSAGE FLOW



XCOMPUTE PLATFORM OVERVIEW



CURRENT WORKFLOW

XCOMPUTE 2018a

- **Systems** Management
  - Complex Organization *
  - Data Interoperability

- Numerical **Modeling**
  - Solids (Equilibrium)
    - FEM Linear-Elastic, Thermo-Elastic
    - 2nd-order spatial, steady-state
  - Liquids (Incompressible Fluids)
    - LBM Pull (Zou He) **
    - 2nd-order spatial, 1st-order temporal
  - Gases (Compressible Fluids)
    - FVM AUSM+MUSCL
    - 2nd-order spatial, 2nd-order temporal

- **Geometry** Preparation
  - Structured (Procedural)
    - Cartesian Grid
    - Octree **
  - Unstructured (Declarative)
    - Median-Dual Mesh
      - Import/Export MSH
      - Persson-Delaunay Mesh Generator *
    - Lagrangian Network **

- **Graphics** and **Integration**
  - Modern OpenGL Shader Pipeline
  - Universal Protocol Buffers

* beta
** in dev

LR1000 SETUP IN XCOMPUTE

Demo4 | graham@xplicit-offic - XCompute Client v0.9

File Edit Tools View Help

Constant Normals Color Map Nodes Edges Faces Extents Shadows Transparent Greyscale Node Face Surface System

Systems Macros Geometry Physics Conditions Algorithms Overview Data Visuals

System Tree
- Demo4
  - Fuel Press
    - Fuel Pump
      - Fuel Compressor 2
      - Fuel Compressor 1
      - Fuel Diffuser 3
      - Induction Motor
      - Fuel Compressor 3
      - Fuel Diffuser 2
      - Fuel Diffuser 1
    - Fuel Bleed
      - Fuel Bleed Check V
      - Fuel Bleed Solenoic
    - MFV
  - Control
    - Pre-Burner
    - Ignitor
    - Throttle Valve
    - Ignition Coil
  - Powerhead
    - Induction Generator
    - Mixer
    - Post-Burner
    - Stator Nozzle
    - Turbine Expander 1
    - Turbine Expander 2
  - Nozzle
    - Extension
    - Regen Cooling
    - BLC Cooling
  - Ox Press
    - Ox Pump
      - Ox Impeller 2
      - Ox Diffuser 2
      - Ox Impeller 1
      - Ox Diffuser 1
    - Ox Bleed
      - Ox Bleed Solenoid
      - Ox Bleed Check Va
    - MOV
    - Ox Bypass

Schematic view
Demo4

Ox Press
Ox Blee
Ox B
Ox Pum
MOV

Control
Throttle
Pre-Bur
Ignitio
Ignitor

Powerhead
Mi
Indu
Turbine
Stator N
Turbine
Post-Burn

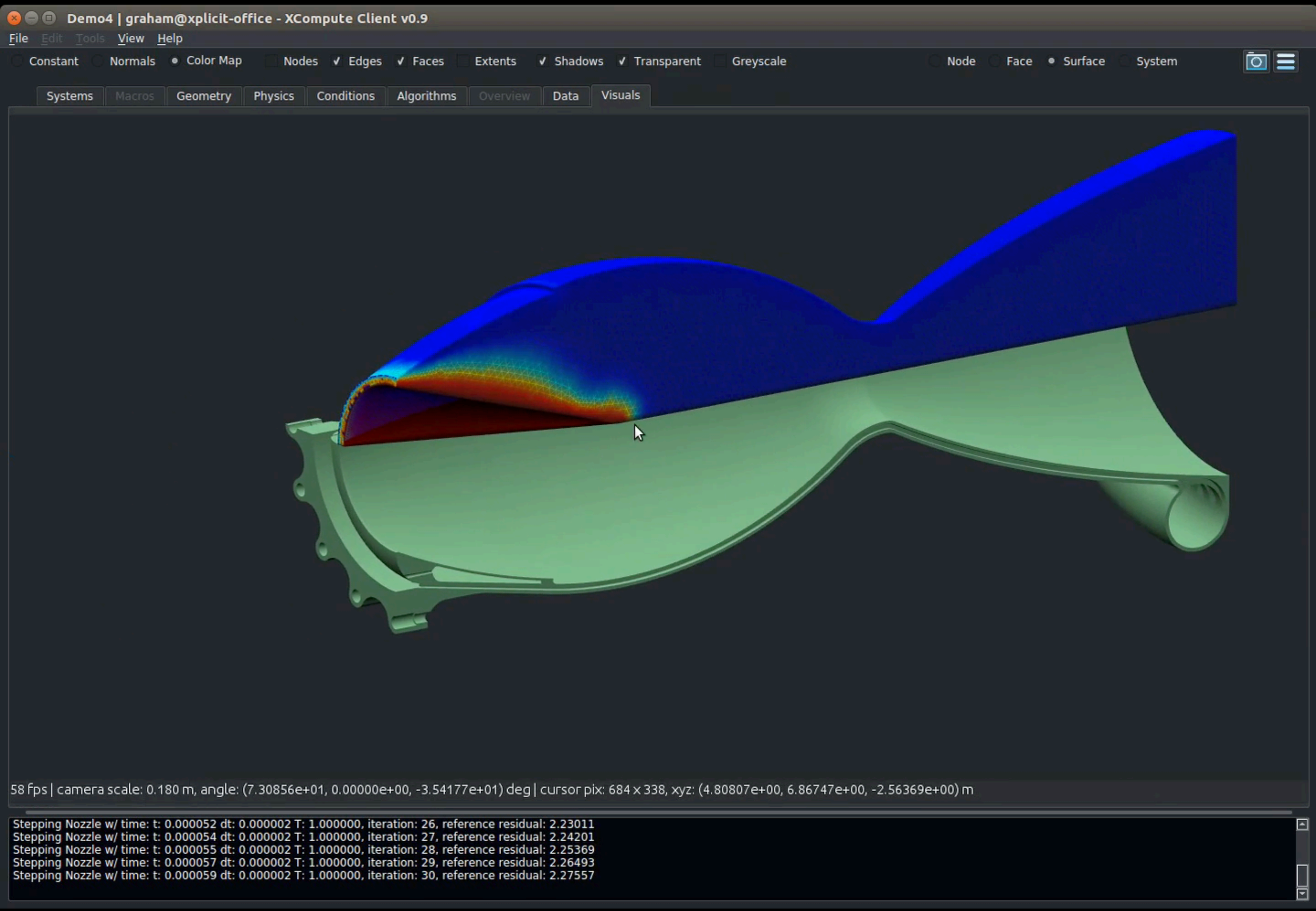
Fuel Press
Fuel Pu
MFV
Fuel Ble

Nozzle
BLC
Rege
Extension

30 fps | camera scale: 0.236 m, angle: (5.91425e+01, 0.00000e+00, -4.67239e+01) deg | cursor pix: 238 x 42, xyz: (7.04276e+00, 6.94550e+00, -5.79555e+00) m

```
refreshing systems from 'Demo4'...complete.
open complete.
Changing active system to: 'Nozzle'
seeking objects in /home/graham/Desktop/XCOMPUTE/User/Demo4/Nozzle
loaded iteration: 1574
```

LR1000 CFD (2M ELEMENTS, REALTIME ON A TITAN-Z GPU)



LR1000 SCHEMATIC (IN WORK)

Demo4 | graham@xplicit-office - XCompute Client v0.9

File Edit Tools View Help

Constant Normals Color Map Nodes Edges Faces Extents Shadows Transparent Greyscale Node Face Surface System

Systems Macros Geometry Physics Conditions Algorithms Overview Data Visuals

System Tree

- Demo4
  - Fuel Press
    - Fuel Pump
      - Induction Motor
      - Fuel Diffuser 1
      - Fuel Compressor 1
      - Fuel Diffuser 3
      - Fuel Compressor 2
      - Fuel Compressor 3
      - Fuel Diffuser 2
    - Fuel Bleed
      - Fuel Bleed Check V
      - Fuel Bleed Solenoic
    - MFV
  - Control
    - Pre-Burner
    - Throttle Valve
    - Ignition Coil
    - Ignitor
  - Powerhead
    - Induction Generator
    - Mixer
    - Post-Burner
    - Stator Nozzle
    - Turbine Expander 2
    - Turbine Expander 1
  - Nozzle
    - Extension
    - Regen Cooling
    - BLC Cooling
  - Ox Press
    - Ox Bypass
    - MOV
    - Ox Pump
      - Ox Impeller 2
      - Ox Diffuser 2
      - Ox Impeller 1
      - Ox Diffuser 1
    - Ox Bleed
      - Ox Bleed Solenoid
      - Ox Bleed Check Va

Schematic view

Ox Press
Ox Bleed
Ox Bl
Ox Bl
Ox Bypass
S
Ox Pump
Ox Im
Ox Di
Ox Im
Ox Di
MOV
S
P

Control
Throttle Valve
M
Pre-Burner
Ignition Coil
Ignitor

Powerhead
Mixer
Inductio
Turbine Expande
Stator Nozzle
Turbine Expande
Post-Burner

Fuel Press
Fuel Pump
Fu
Fu
Fu
MFV
S
P
Fuel Bleed
Fuel
Fuel

Nozzle
BLC Coo
Regen Co
Extension

58 fps | camera scale: 0.150 m, angle: (7.05713e+01, 0.00000e+00, -6.32163e+01) deg | cursor pix: 963 x 5, xyz: (6.15839e+00, 3.10958e+00, -2.37033e+00) m

```
Stepping Nozzle w/ time: t: 0.000565 dt: 0.000001 T: 1.000000, iteration: 367, reference residual: 1.41547
Stepping Nozzle w/ time: t: 0.000566 dt: 0.000001 T: 1.000000, iteration: 368, reference residual: 1.41725
Stepping Nozzle w/ time: t: 0.000567 dt: 0.000001 T: 1.000000, iteration: 369, reference residual: 1.4192
Stop signal received. Stopping...
job: System Tree Solver - Nozzle stopped.
```

META-MACHINES

TRANSCENDING SOFTWARE & HARDWARE

CORE MODULE ROADMAP

- Server-Client Networking
    - Local Team
    - INET SaaS
- Systems Schematic
    - MDAO
    - Multi-Fidelity Coupling
- Octree Geometry
    - Advanced Meshing
    - Advanced FVM
- Convolutional FDM
    - Compressible LES
    - FDTD E&M
- Advanced Modeling
    - Multi-Phase LBM
    - Nonlinear FEM
- Dynamic Modeling
    - Chemical Reactions
    - Adjoint Optimization
- XCM Presentation Format
- XCS and CL Marketplace

THIRD-PARTY MODULE ROADMAP

- Spatial/File Interoperability
- XLS/X Spreadsheet Integration
- STEP Integration
- Complex Internal Topology
- Additive Manufacture
- Hydraulic Optimization
- On-Demand SaaS
- Free Web Client
- your app?

UPCOMING RELEASE - *XCompute 2018b*

- Ubuntu, SciLinux, RHEL (gcc, clang)
- Mac OSX (clang)
- Win64 (mingw)

References:
[0] Xplicit Computing, Inc. USPTO Filings 62540917, 62540919, 62540922, 62540927, 62540933
[1] Milton Van Dyke. "An Album of Fluid Motion". 1982 Parabolic Press
[2] Jiri Blazek. " Computational Fluid Dynamics: Principles and Applications."
[3] Vigor Yang, M. Habiballah, J. Hulka, M. Popp. "Liquid Rocket Thrust Chambers: Aspects of Modeling, Analysis, and Design." AIAA Vol 200, 2004.
[4] Stephen B Pope. "Turbulent Flows." Cambridge University Press, 2000.
[5] Per-Olof Persson. "Mesh Generation for Implicit Geometries." MIT Department of Mathematics, 2005.
[6] William Hieser, David T Pratt. "Hypersonic Airbreathing Propulsion." AIAA Education Series, 1994.
[7] Ray Kurzweil. "The Singularity Is Near". Penguin Book, 2005.
[8] Anthony Corso. "X2 Benchmarking Tool". Xplicit Computing Report, 2017.

Media Courtesy:
[9] Space Energetics LLC
[10] Virgin Galactic LLC
[11] Blue Origin LLC
[12] NVidia Corporation
[13] Harvey Mudd College
[14] University of Southern California
[15] California Institute of Technology
[16] The Verge (www.theverge.com)
[17] CFD Israel Blog

Special Thanks to:
David Lortscher
Innovation Accelerators Partnership
Mike Jaffe and Family
Amador Family
Ivan Wong

(c) 2018 Xplicit Computing, Inc
www.xplicitcomputing.com
info@xplicitcomputing.com