Lab Exercise 1- Structure of a Modelica Model & Acausal vs Causal Modeling

---

1. Aim

To understand:

1. Structure of a Modelica model
    - Parameters
    - Variables
    - Equations
2. Difference between acausal and causal modeling
3. How equation-based modeling works in OpenModelica

---

2. Learning Outcomes

After completing this lab, students will be able to:

- Create a Modelica model
- Differentiate parameters and variables
- Write physical equations
- Understand acausal modeling
- Compare causal vs acausal approach

---

3. Software Required

- OpenModelica (OMEdit)

---

4. PART A – Structure of a Modelica Model

We will create a **Mass-Spring-Damper System**, commonly used in aerospace and defense vibration systems.

STEP 1: Open OpenModelica

1. Launch OMEdit
2. Click File → New Model
3. Name the model: **MassSpringSystem**

Click OK.

STEP 2: Understand Basic Model Structure

Basic Modelica structure:

```
model ModelName

  // Parameters
  parameter Real p;

  // Variables
  Real x;

equation

  // Equations
```

```
  x = p;

end ModelName;
```

Modelica has 3 main parts:

1. Parameters
2. Variables
3. Equations

STEP 3: Add Parameters

Parameters are constants defined before simulation.

Add:

```
parameter Real m = 10;     // mass (kg)

parameter Real k = 200;    // spring constant (N/m)

parameter Real c = 20;     // damping coefficient
```

Explanation:

- m → mass of object
- k → spring stiffness
- c → damping factor

These values remain constant during simulation.

STEP 4: Add Variables

Variables change during simulation.

Add:

```
Real x(start=0.1);     // displacement

Real v(start=0);       // velocity
```

Explanation:

- x → position
- v → velocity
- start value helps initialize simulation

STEP 5: Add Equations

Add physics equations:

```
equation

  der(x) = v;

  m*der(v) + c*v + k*x = 0;
```

Explanation:

- der(x) = v → velocity is derivative of position
- m*der(v)* + *c*v + k*x = 0 → Newton's law

This describes vibration motion.

COMPLETE CODE

```
model MassSpringSystem

  // Parameters
  parameter Real m = 10;
  parameter Real k = 200;
  parameter Real c = 20;

  // Variables
  Real x(start=0.1);
  Real v(start=0);

equation

  der(x) = v;

  m*der(v) + c*v + k*x = 0;

end MassSpringSystem;
```

STEP 6: Simulate

1. Click Check Model
2. Click Simulate
3. Plot:
   - x
   - v

You will observe oscillation with damping.

PART B – Understanding Each Section in Detail

---

1. Parameters

Defined using:

```
parameter Real name = value;
```

Properties:

- Constant during simulation
- Used for physical constants
- Improves model flexibility

If you change m from 10 to 20, behavior changes.

---

2. Variables

Defined using:

```
Real variableName;
```

Properties:

- Solved by equation solver
- Change over time
- Represent system states

---

3. Equations

Defined under:

equation

Important:

This is NOT assignment like C or MATLAB.

It is a mathematical relationship.

Example:

```
m*der(v) + c*v + k*x = 0;
```

Means:

All variables must satisfy this equation simultaneously.

PART C – Acausal vs Causal Modeling

STEP 7: Understand Causal Modeling

Causal modeling defines direction.

Example in algorithm style:

```
acceleration = (-c*v - k*x)/m;

v = v + acceleration*dt;

x = x + v*dt;
```

Here:

- Input/output defined
- Order matters

- Step-by-step execution

This is used in MATLAB or C.

STEP 8: Acausal Modeling (Modelica)

Modelica version:

```
m*der(v) + c*v + k*x = 0;
```

No input/output specified.

The solver:

- Determines unknowns
- Solves equations simultaneously

You define physics, not computation order.

KEY DIFFERENCE

Causal → Algorithm

Acausal → Physical law

PART D – Defense-Based Example

Consider a **Tank Suspension System**.

Physics:

```
mder(v) + cv + k*x = TerrainForce
```

You write:

```
model TankSuspension

  parameter Real m = 20000;
  parameter Real k = 500000;
  parameter Real c = 5000;

  Real x(start=0.2);
  Real v(start=0);
  Real terrainForce;

equation

  terrainForce = 10000*sin(time);

  der(x) = v;

  m*der(v) + c*v + k*x = terrainForce;

end TankSuspension;
```

Simulation shows vibration under rough terrain.

Defense Use:

- Improve shock absorption
- Increase crew safety
- Optimize suspension design