Final Benchmarks of Modelica Simulators vs. previous ones, based on other tools

Deliverable R6.18 of task T6.18

OPENPROD (08021)

Document version no.: v1.0
Edited by : H. Saafi, Z. Benjelloun, M. Ben Gaid, J. Bohbot, J. Brac
on : 29/10/2012

HISTORY

Version	Date	Remarks
1.0	29/10/2012	First release

TABLE OF CONTENTS

1. Goal of this document 4

2. Benchmarking methodology 4

2.1. Modelica environments 4

2.2. Benchmarking models overview 4

2.2.1. ModEngine combustion models 4

2.2.2. Engine idle speed model 4

2.3. Porting ModEngine to OpenModelica 5

2.4. Comparison criteria 5

3. Benchmarking results 6

3.1. Wiebe combustion model 6

3.1.1. Wiebe combustion model without valve and with simple injector 6

3.1.2. Wiebe with valve and simple injector 9

3.1.3. Wiebe with valve and complex injector model 10

3.2. Coherent flame Model (CFM) 10

3.2.1. CFM without valve and with simple injector 10

3.3. Barba combustion model 13

3.3.1. Barba without valve, with simple injector and without combustion 13

3.4. Valve 16

3.5. Test of poppet valve 18

3.6. Idle speed model benchmark 20

4. Conclusions 21

5. Acknowledgements 22

6. References 23

1. Goal of this document

This document is the deliverable of task 6.18 in the WP6 of OPENPROD. OPENPROD main objective is to generalize model-driven approaches to include most aspects of product development, thus significantly increasing the effectiveness of the process. Key aspects of the project are :

- A holistic whole-product model-driven rapid development and design environment for both software and hardware, also including support for product business processes.
- Open source tools and components for open reusable solutions.
- Standardized model representation of products primarily based on Modelica and UML.

In this process, WP6 demonstrators are aimed at validating the OPENPROD innovations developed in the other work packages. An important part of these innovations target to improve the OpenModelica tool, which is a central tool in the OPENPROD toolchain.

The task 6.18 aims at benchmarking OpenModelica by comparing it to other Modelica tools outside of OPENPROD, which are Dymola (from Dassault Systèmes) and ScicosLab (from INRIA).

2. Benchmarking methodology

2.1. Modelica environments

In this document, versions 1.5 of OpenModelica and 7.4 of Dymola were mainly compared. Unless stated otherwise, comparisons were undertaken using these versions. At the end of OPENPROD project, we revaluated some benchmarks using version 1.9 of OpenModelica.

2.2. Benchmarking models overview

To compare OpenModelica to other Modelica tools, two categories of benchmarks were selected.

2.2.1. ModEngine combustion models

ModEngine is a Modelica [1] library that allows the modeling of a complete engine with diesel and gasoline combustion models. Requirements for the ModEngine library were derived from the existing IFP-Engine AMESim library. ModEngine contains more than 250 sub models. It has been developed to allow the simulation of a complete virtual engine using a characteristic time-scale of the order of the crankshaft angle. A variety of elements are available to build representative models for engine components, such as turbocharger, wastegate, gasoline or Diesel injectors, valve, air path, EGR loop etc ...

The technological starting point of the work presented in this report was the ModEngine library from EuroSysLib project. During OPENPROD, a porting work was undertaken to allow the execution of this library in OpenModelica. Part of this porting work is briefly described in this document. Other extensions (flex and dual fuel compatibility, state events reduction ...) were performed, and are described in the deliverable of task T6.17. These new extensions were used by the demonstrator of task T6.13.

2.2.2. Engine idle speed model

In order to compare OpenModelica to ScicosLab, which are both free and open source simulation environments, an engine idle speed model was successfully ported to all these environments and evaluated.

2.3. Porting ModEngine to OpenModelica

ModEngine was originally developed in Dymola, and completely written in Modelica. Since both Dymola and OpenModelica have support for the same version of the Modelica language, one might think that a successful compilation and execution in Dymola will naturally lead to a successful compilation and execution in OpenModelica. But in reality, this was not the case. An important porting work was undertaken in order to allow the execution of the most important components of ModEngine in OpenModelica. Four main reasons explain these problems :

- ModEngine is a complex and industrial-size library
- OpenModelica does not fully support Modelica language
- Dymola and OpenModelica have sometimes different interpretations of the Modelica language specification
- Some bugs exist in OpenModelica Compiler (OMC), but were quickly and efficiently corrected by the OMC development team

In the following, some porting difficulties (with OpenModelica 1.5) and found workarounds are reported :

Problem : Some syntactic forms of the `import`-clause were not recognized by OpenModelica (for example, `import SI=Modelica.SIunits.*`)
Solution : Rewrite the import declarations following the 5 supported syntactic forms by OpenModelica :

```
import packagename; (qualified import)
import [packagename.]definitionname; (single definition import)
import packagename.*; (unqualified import)
import shortpackagename = packagename; (renaming import)
import shortpackagename = [packagename.]definitionname; (renaming single def. import)
```

Problem : Package `Modelica.Utilities` was not implemented in OpenModelica
Solution : We developed this package.

Problem : OpenModelica does not accept variables that have the same name as C language keywords (example `Boolean unsigned;`)
Solution : Modelica variables naming was modified.

Problem : Functions `edge(x)` and `change(x)` are not supported
Solution : These functions were replaced respectively by `(x and not pre(x))` and `(x <> pre(x))`

Problem : In a when loop, OpenModelica does not accept functions that return two vectors or two matrices
Solution : These functions were rewritten and split in parts to return only one vector or one matrix

Problem : Nested if loops lead to simulation errors
Solution : Same if loops were rewritten, but it was not possible to find workaround for all models from ModEngine. Details are given in the following.

2.4. Comparison criteria

The comparison between the tools was based on two major criteria :

- CPU time
- Simulation result accuracy

Modelica language support is another criterion that was also reported, and that prevented the evaluation of some sub-components.

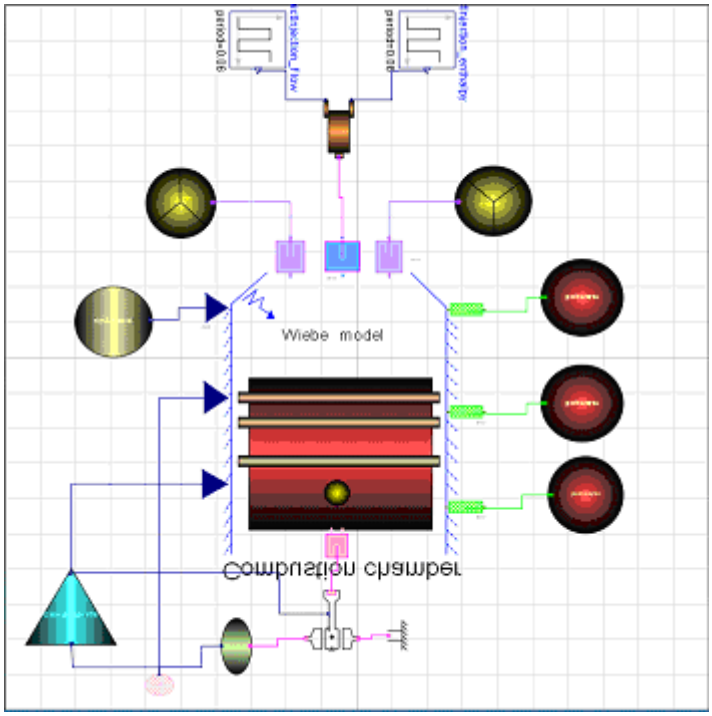
3. Benchmarking results

3.1. Wiebe combustion model

Wiebe combustion model is a semi-physical model, based on the Wiebe model for combustion heat release [2], and presenting much less complexity than phenomenological combustion models like CFM or Barba, which will be described later. It is based on a mix of physical approaches and identifications or learning processes applied on the results of an experimental or/and numerical combustion campaign performed with a more complex model. The main advantage of this model is to take into account the behavior of the engine with a crank angle degree timescale, which is not the case of look-up table models.

Different variants with increased complexity of this model were benchmarked.

3.1.1. Wiebe combustion model without valve and with simple injector



The evaluation parameters are the following :

Solver	**: DASSL**
Simulation stop time	**: 1s**
Tolerance	**: 0.00001**
Number of intervals	**: 500**

The evaluation results are the following :

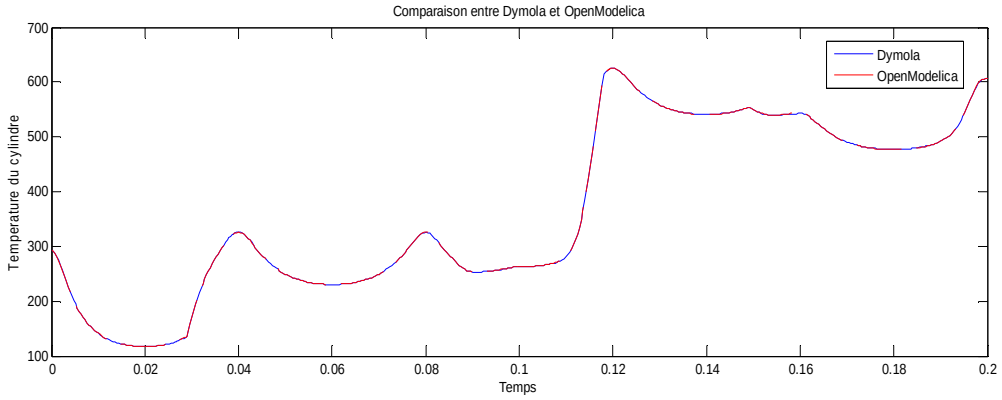
Solver	DASSL	
Platform	Dymola	OpenModelica
CPU Time	0.361	5.639
Number of numerical Jacobians	0	0
Number of zero crossings	12	12
Number of equations	683	683

This model can be configured to use different heat transfer models. In the following, for each transfer model, simulation results are compared between Dymola and OpenModelica.

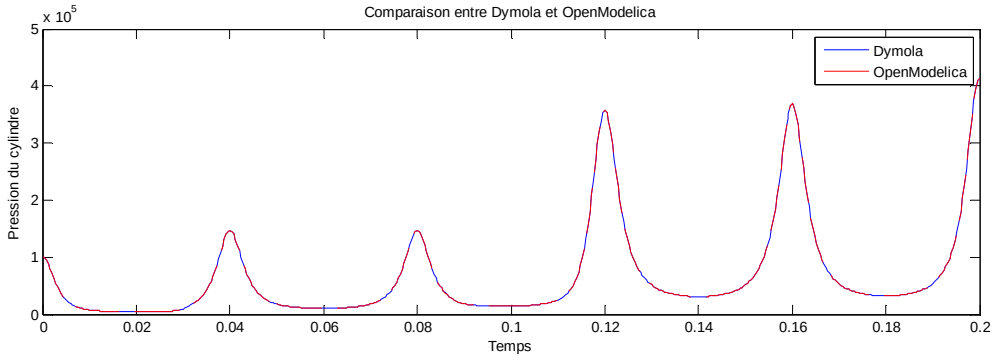
3.1.1.1. Adiabatic heat transfer model

Simulation results are depicted in the following figures:

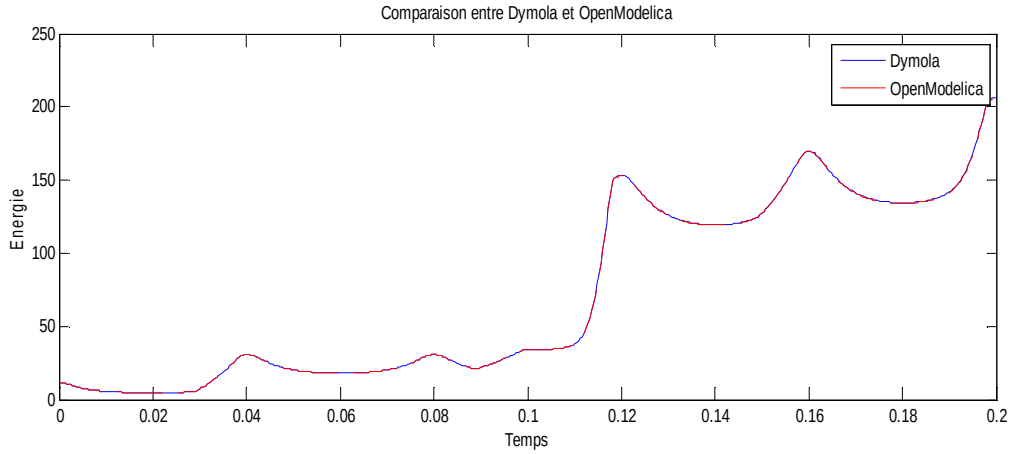
Cylinder temperature



Cylinder pressure

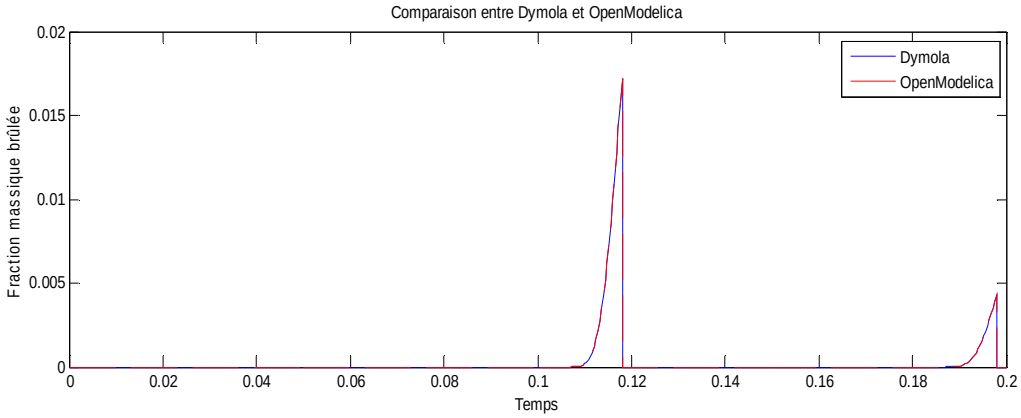


Energy

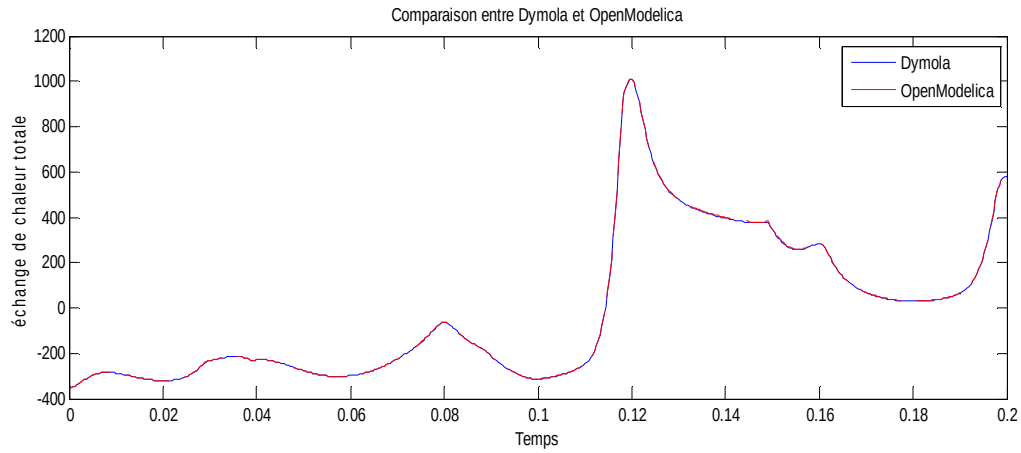


3.1.1.2. "Annand" heat transfer model

Burnt mass fraction

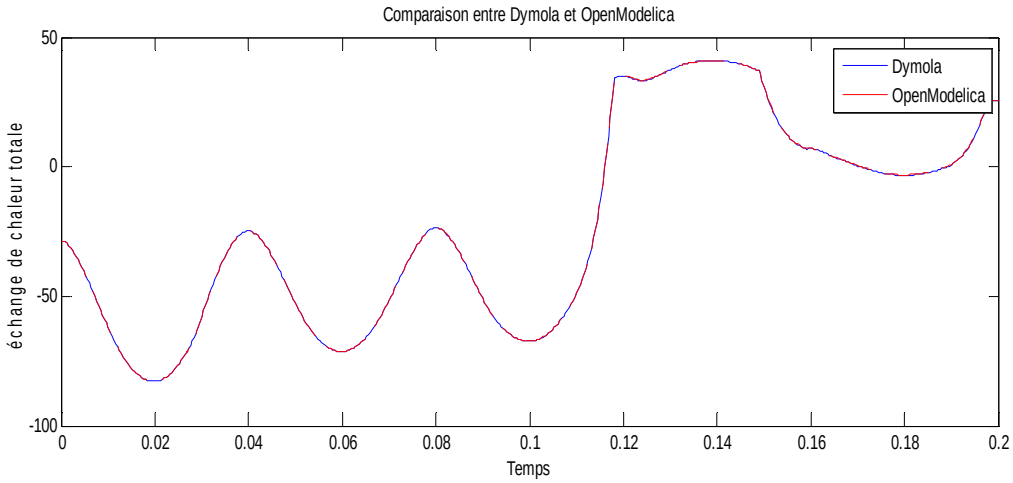


Total exchanged heat



3.1.1.3. "Eicheberg" heat transfer model

Total heat release



3.1.1.4. "Woschni" heat transfer model

OpenModelica was not able to translate the following loop to C code :

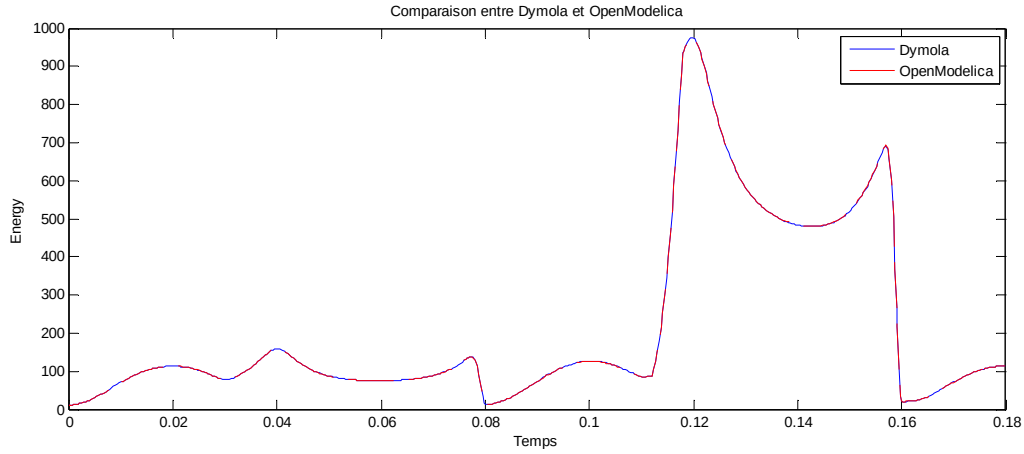
```
if not initial() and bWoschniExchangeHeatModel then
  if abs(intakeMassFlow) > Modelica.Constants.eps then
     woschniMotorCycle = 2;  // Intake Period
   elseif abs(exhaustMassFlow) > Modelica.Constants.eps then
     woschniMotorCycle = 1;  // Exhaust Period
   else
     if pre(woschniMotorCycle) == 2 then
       woschniMotorCycle = if icalCombustion == 1 then pre(woschniMotorCycle)-2 else
                           (if abs(exhaustMassFlow) <= Modelica.Constants.eps and
abs(intakeMassFlow) <= Modelica.Constants.eps then
                             pre(woschniMotorCycle)+1 else pre(woschniMotorCycle));
     elseif pre(woschniMotorCycle) == 3 then
       woschniMotorCycle = if icalCombustion == 1 then
                          pre(woschniMotorCycle)-3 else pre(woschniMotorCycle);
     else
       woschniMotorCycle = pre(woschniMotorCycle);
     end if;
   end if;
 else
   woschniMotorCycle = initMotorPhase;
 end if
```

We were not able to find a workaround to this problem.

3.1.2. Wiebe with valve and simple injector

Simulation results are illustrated in the following figures.

Energy



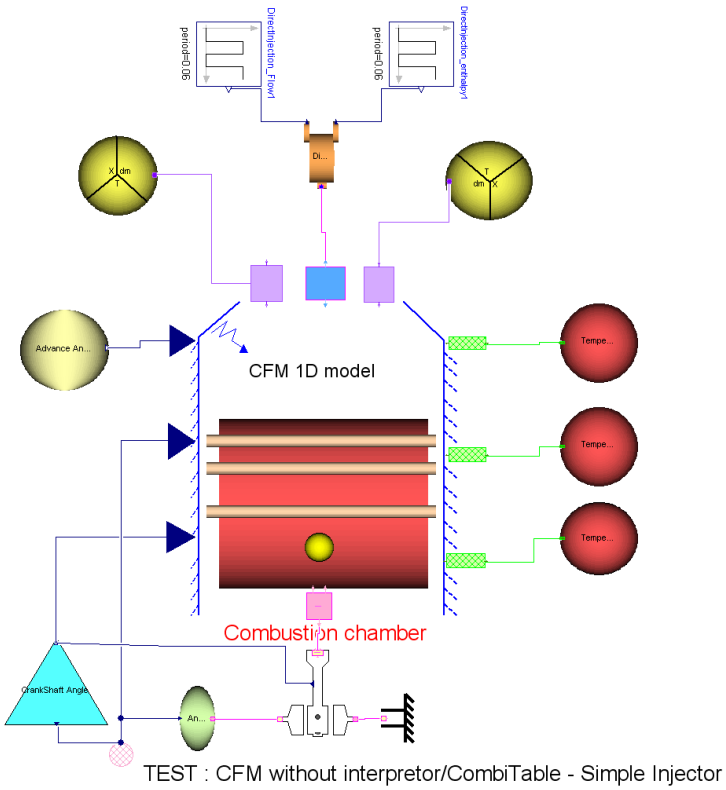
3.1.3. Wiebe with valve and complex injector model

The complex injector contains the Modelica component `CombiTable`, which was not supported by OpenModelica. Consequently, it was not possible to test the execution of this model in OpenModelica.

3.2. Coherent flame Model (CFM)

The phenomenological CFM-1D model was developed by IFPEN [3] and is based on the reduction of the 3D ECFM model [4]. In this model, the rate of fuel consumption depends on the flame surface, computed thanks to the laminar flame speed and the turbulent kinetic energy. Only one parameter related to turbulent kinetic energy is tuned for combustion calibration. The other ones remain constant for the whole operating conditions. The CFM-1D model is the typical modeling level able to combine a good representation of physical phenomena with reasonable CPU performances. Thanks to these characteristics, this model can be embedded in a full engine simulator and used for architecture design or control strategy development issues [5].

3.2.1. CFM without valve and with simple injector



The evaluation parameters are the following :

Solver : Runge-Kutta
Simulation stop time : 0.065s
Tolerance : 0.00001

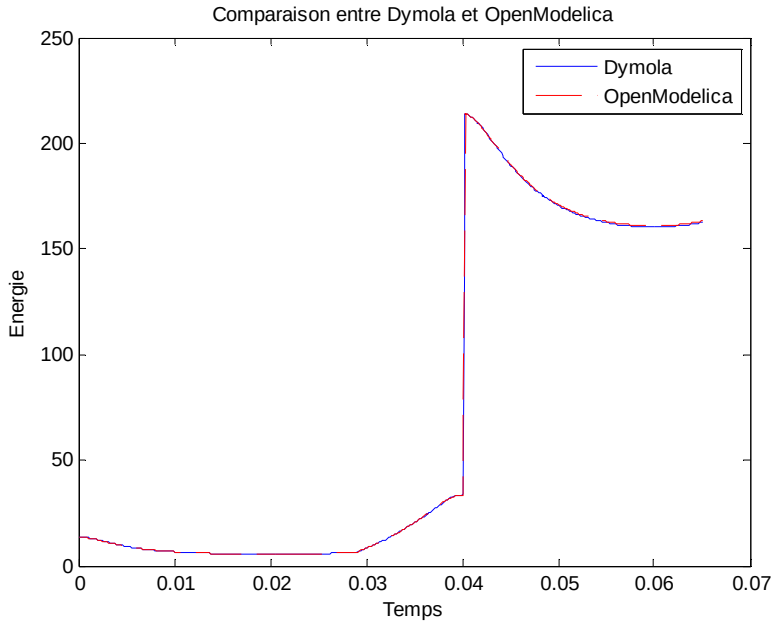
The evaluation results are the following :

Solver	Runge-Kutta	
Platform	Dymola (step=10-5)	OpenModelica (step=10-4)
CPU Time	2.44	32.133
Number of numerical Jacobians	1	
Number of zero crossings	19	
Number of equations	711	

This model can be configured to use different heat transfer models. In the following, for each transfer model, simulation results are compared between Dymola and OpenModelica.

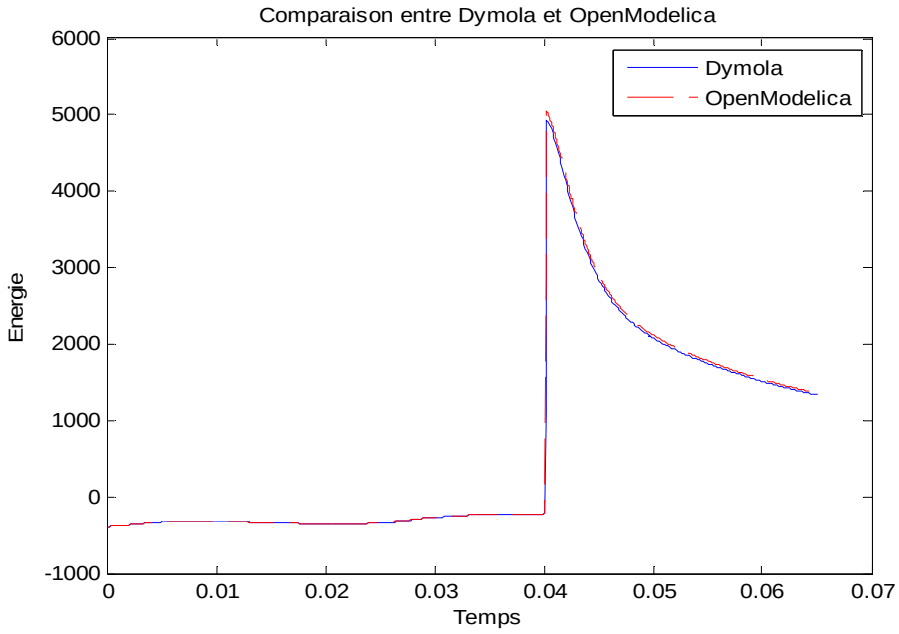
3.2.1.1. Adiabatic heat transfer model

Energy



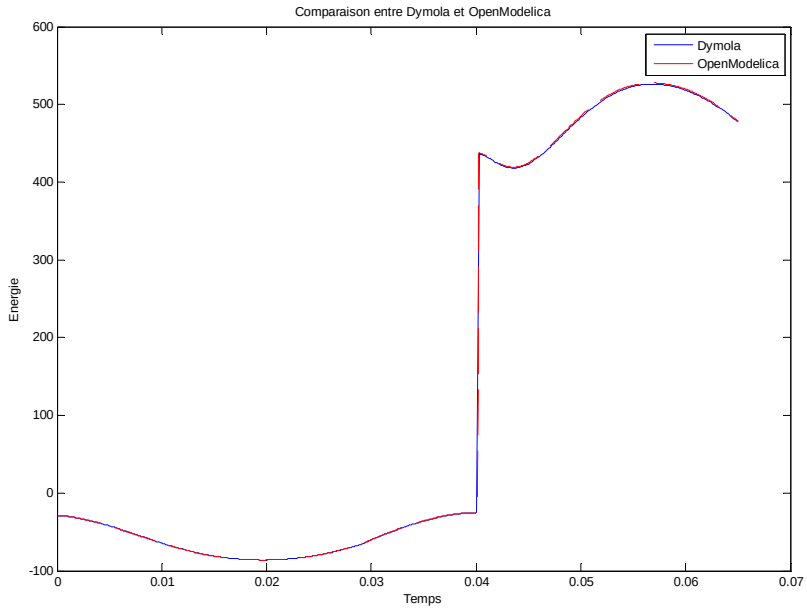
3.2.1.2. "Annand" heat transfer model

Total exchanged heat



3.2.1.3. "Eicheberg" Heat transfer model

Total exchanged heat



3.2.1.4. "Woschni" heat transfer model

The same problem as with Wiebe model was encountered.

3.2.1.5. CFM with Valve and simple injector

Simulation stops at ~0.00016 seconds with convergence error, probably due to an insufficient used step-size.

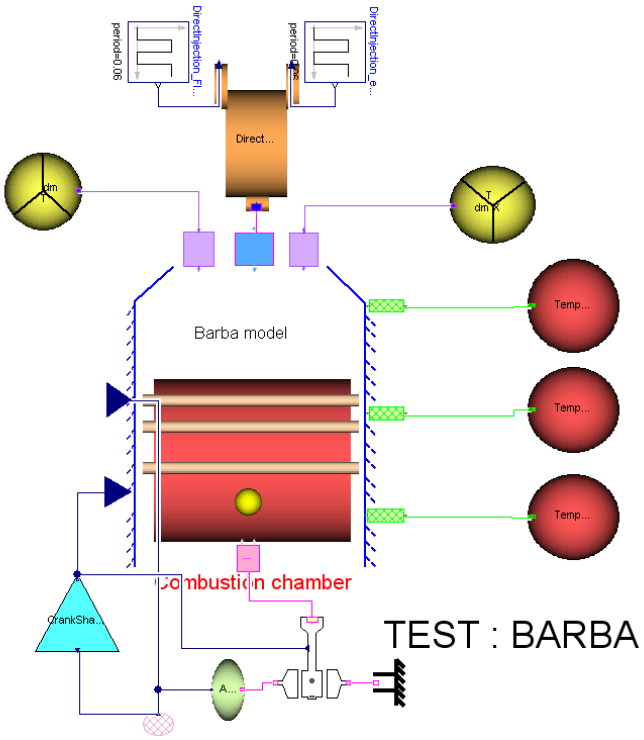
3.2.1.6. CFM with valve and complex injector

The complex injector contains the Modelica component `CombiTable`, which was not supported by OpenModelica. Consequently, it was not possible to test the execution of this model in OpenModelica.

3.3. Barba combustion model

For Diesel engine, an advanced Barba [6] model developed at IFPEN was implemented in ModEngine. The Barba's model can reproduce the conventional Diesel combustion process, using only 2 zones (a first zone for the description of the pre-mixed combustion and a second one for the diffusion mode). With a reduced number of parameters, it can be used for a wide range of operating points. In this model, the combustion process is divided in 2 steps. In a first step, the fuel is burnt using a premixed model with the hypothesis of flame propagation in the premixed zone. In a second step, when the pre-mixed zone is burnt, the remaining fuel is oxidized using a mixing controlled combustion model. The different hypothesis and equations of the Barba's combustion model are presented in [6].

3.3.1. Barba without valve, with simple injector and without combustion



The evaluation parameters are the following :

Solver : DASSL
Simulation stop time : 0.2s
Tolerance : 0.0001
Number of intervals : 500

The evaluation results are the following :

Solver	DASSL	
Platform	Dymola	OpenModelica
CPU Time	0.156	3.969
Number of numerical Jacobians	0	0
Number of zero crossings	48	48
Number of equations	831	831

Two problems needed to be solved to allow combustion modeling :

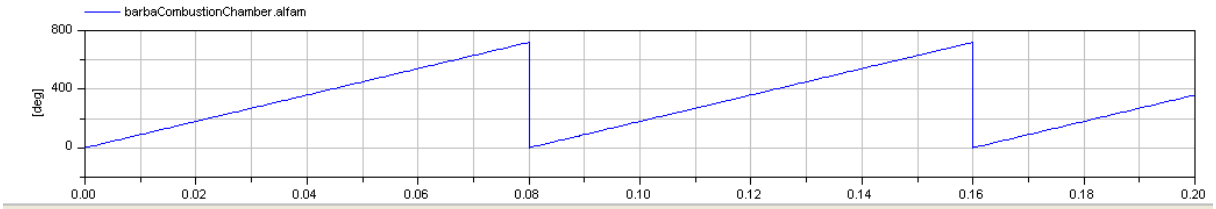
Problem : OpenModelica was not able to detect cycle change which was determined by instant change of integer `iAngleCylcle`.
Solution : was the creation of a Boolean `bNewCycle4T` that takes the 1 state when `alfam` approaches with an error of 10^{-8} rad. We then look at the rising front of this Boolean.

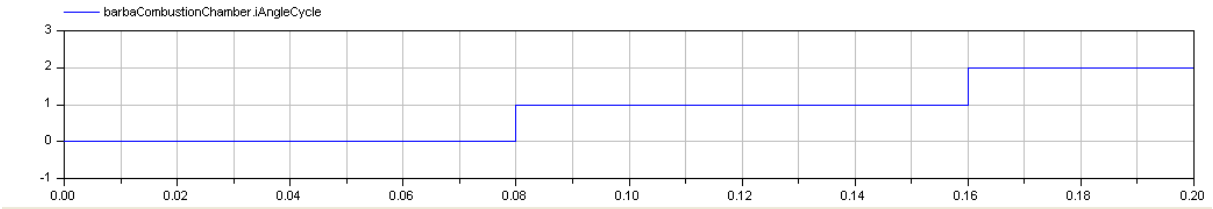
Loop with Boolean

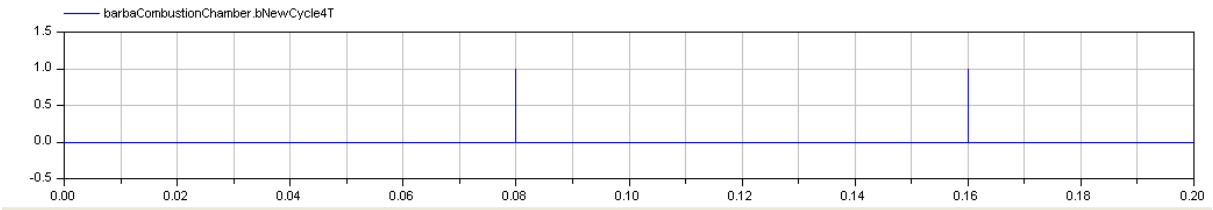
```
when (edge(bNewCycle4T)) then
 reinit(autoIgnitionDelay[1],0.);
 reinit(autoIgnitionDelay[2],0.);
 reinit(autoIgnitionDelay[3],0.);

 for injectionIndex in 1:NumberMaxInjection loop
  reinit(massFuelInj[injectionIndex],0.);
  reinit(massFuelDifZone[injectionIndex],0.);
  reinit(massZone1[1,injectionIndex],0.);
  reinit(massZone1[2,injectionIndex],0.);
  reinit(massZone1[3,injectionIndex],0.);
  reinit(massFreshGasZone1[1,injectionIndex],0.);
  reinit(massFreshGasZone1[2,injectionIndex],0.);
  reinit(massFreshGasZone1[3,injectionIndex],0.);
 end for;

end when;
```

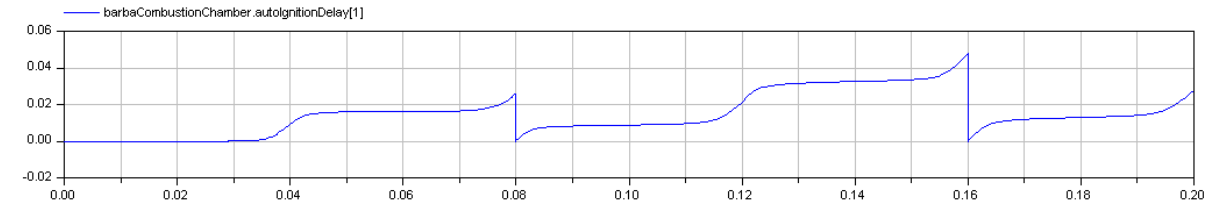
Loop with integer

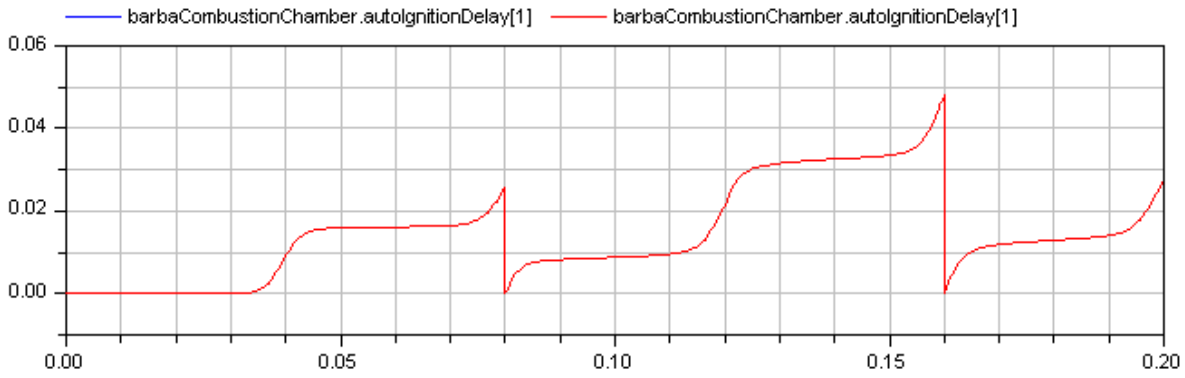
```
when change(iAngleCycle) then
 for injectionIndex in 1:NumberMaxInjection
loop
 reinit(autoIgnitionDelay[injectionIndex],0.);
 reinit(massFuelInj[injectionIndex],0.);
 reinit(massFuelDifZone[injectionIndex],0.);
 reinit(massZone1[:,injectionIndex],
zeros(NumberofGas));
 reinit(massFreshGasZone1[:,injectionIndex],
zeros(NumberofGas));
 end for;
end when;
```







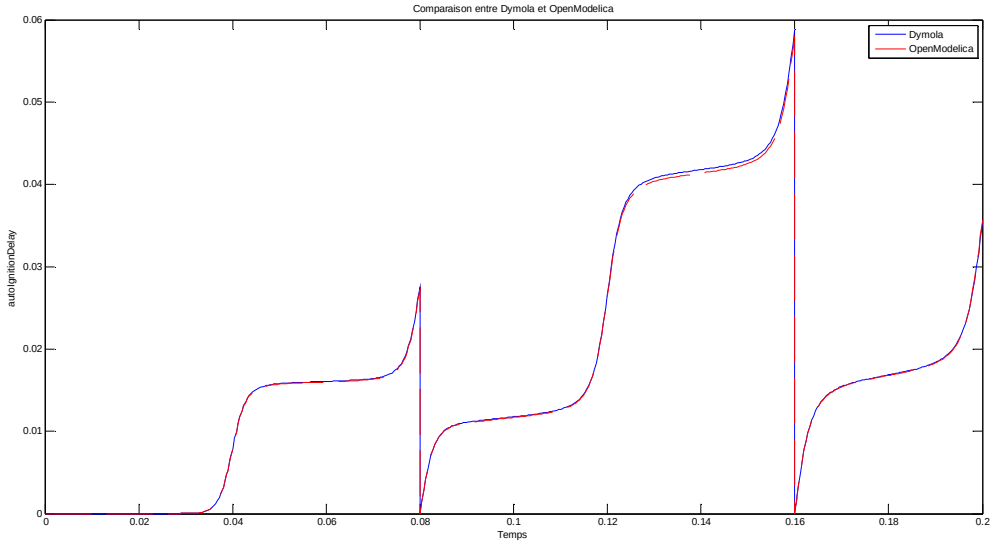




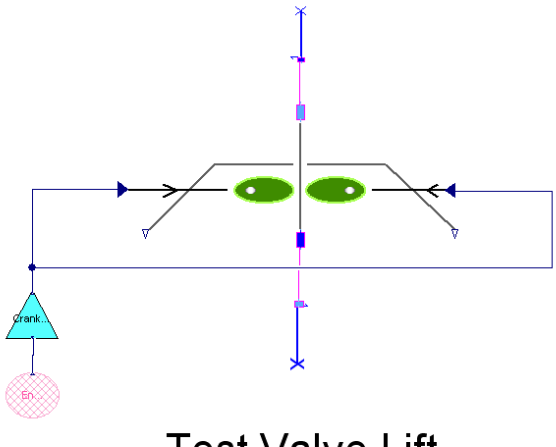
The above figure shows how this programming trick solves the problem (red curve wrt. blue curve).

This allows the comparison of the auto ignition delay variable.

autoIgnitionDelay



3.4. Valve



Test Valve Lift

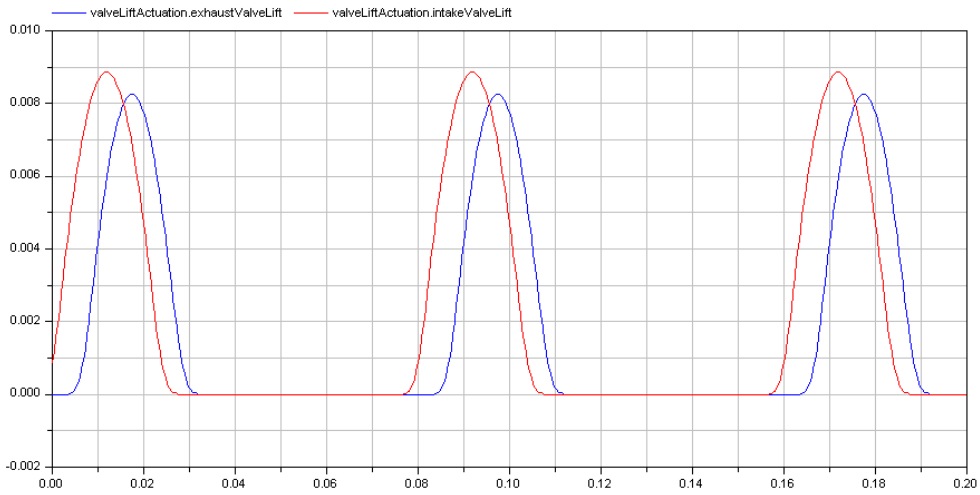
The evaluation parameters are the following :

Solver : DASSL
Simulation stop time : 0.2s
Tolerance : 0.0001
Number of intervals : 500

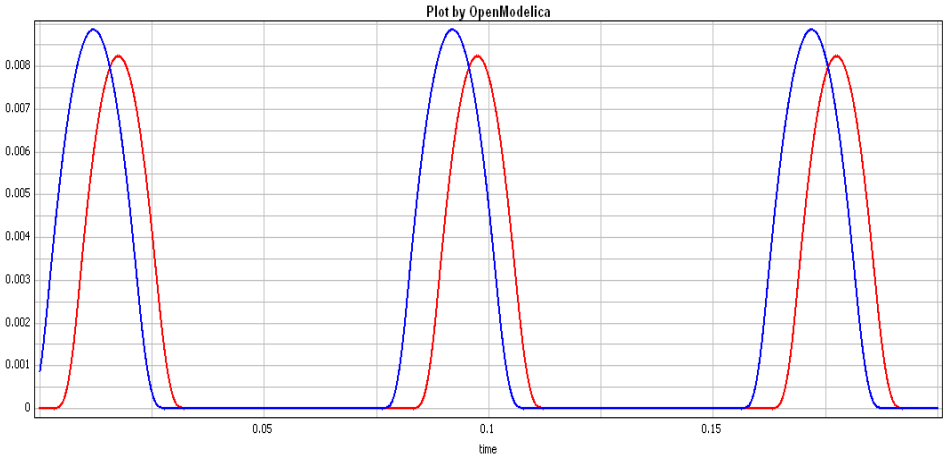
The evaluation results are the following :

Solver	DASSL	
Platform	Dymola	OpenModelica
CPU Time	0.015	2.093
Number of numerical Jacobians	0	0
Number of zero crossings	0	0
Number of equations	1254	1254

The obtained `exhaustValveLift` and `intakeValveLift` simulation results are depicted in the following figures, for both Dymola and OpenModelica

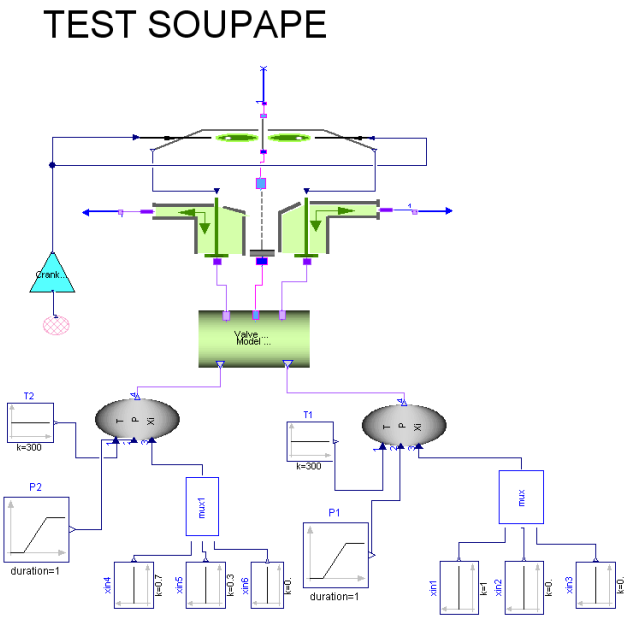


Dymola



OpenModelica

3.5. Test of poppet valve



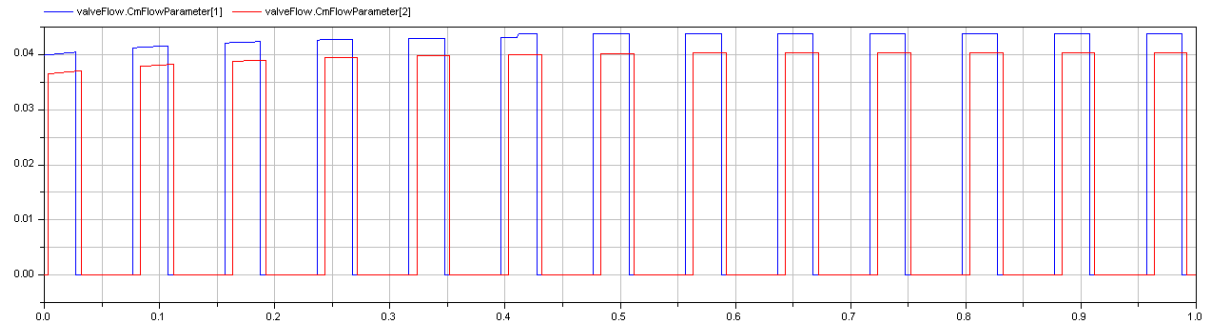
The evaluation parameters are the following :

Solver : DASSL
Simulation stop time : 1s
Tolerance : 0.0001
Number of intervals : 500

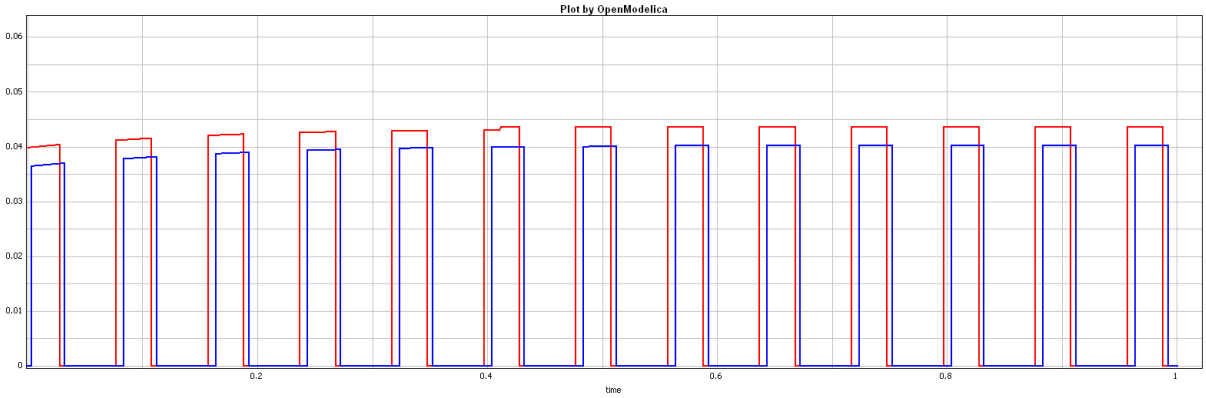
The evaluation results are the following :

Solver	DASSL	
Platform	Dymola	OpenModelica
CPU Time	0.094	4.795
Number of numerical Jacobians	0	0
Number of zero crossings	12	12
Number of equations	2215	2215

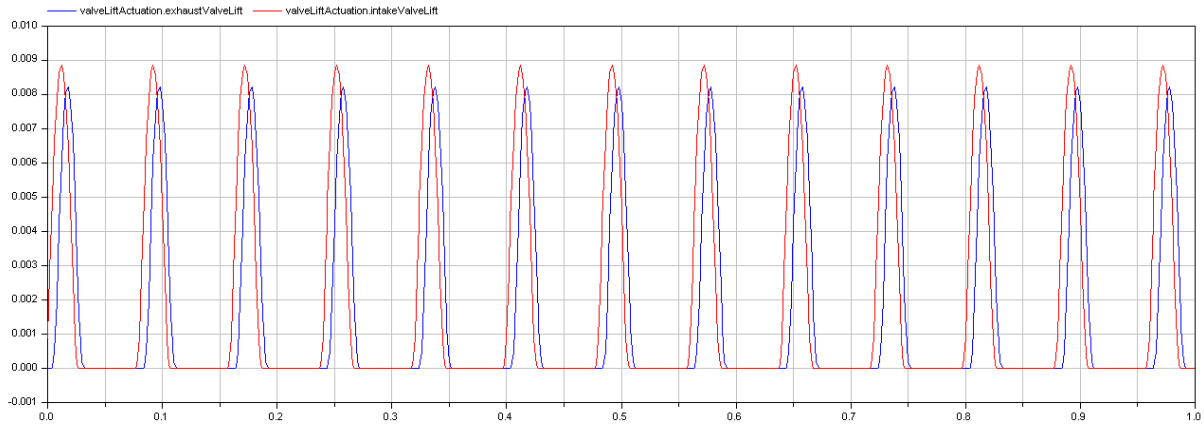
Simulation results are depicted in the following figures :



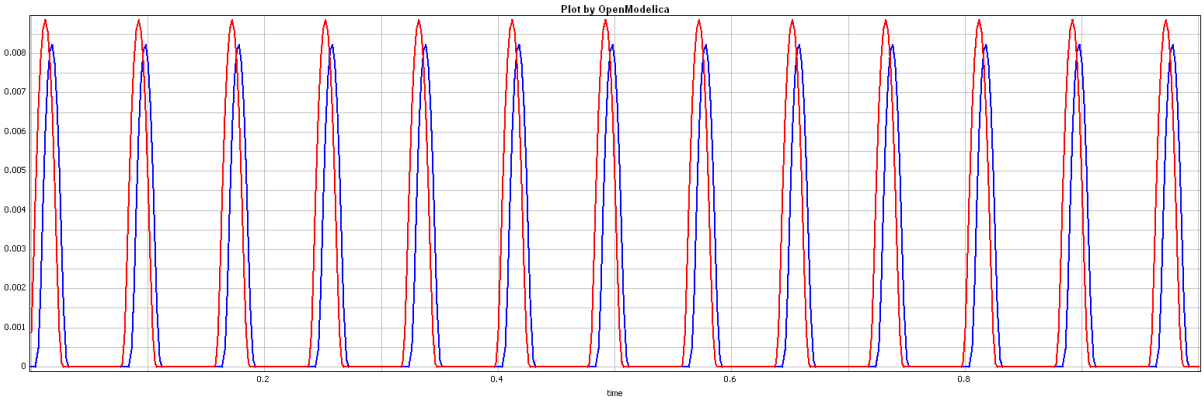
Dymola



OpenModelica



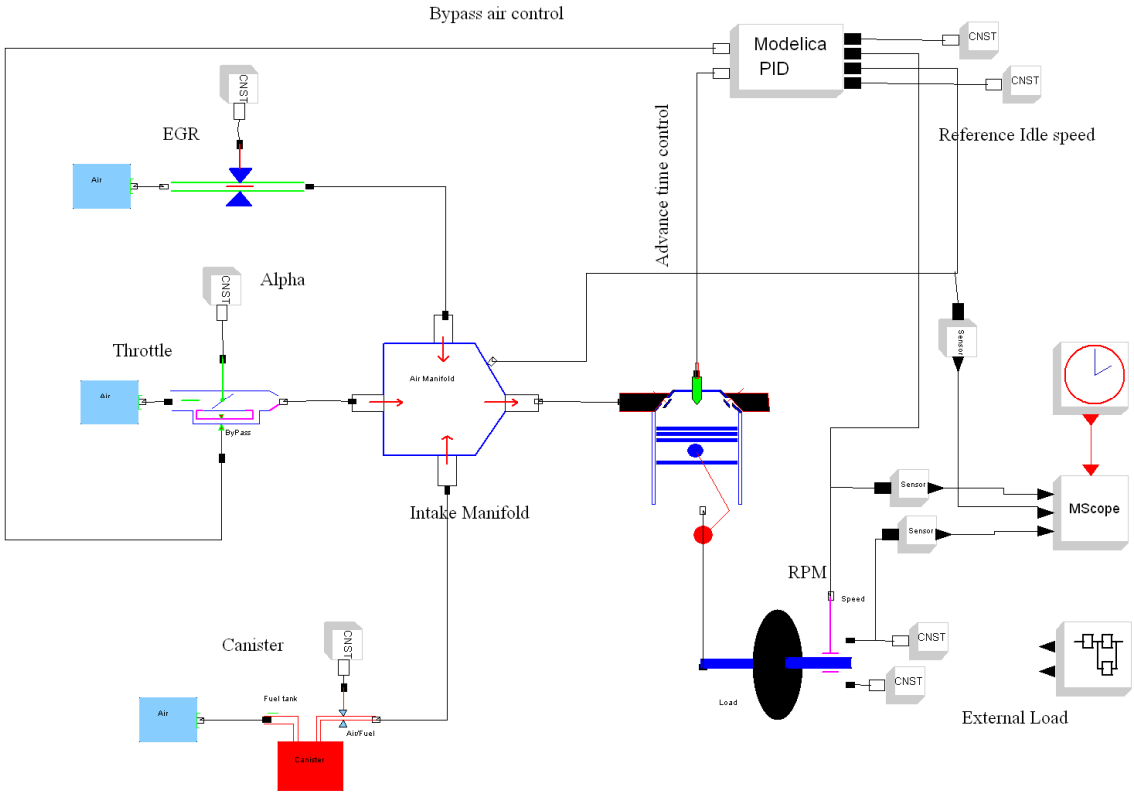
Dymola



OpenModelica

3.6. Idle speed model benchmark

An idle speed internal combustion engine model was developed in Modelica, and successfully executed in Dymola, ScicosLab and OpenModelica. The model sketch in ScicosLab is shown in the figure below:



The evaluation parameters are the following :

Integrator : DASSL
Simulation stop time : 80s
Tolerance : 10^{-6}
Number of intervals : 10^{5}

The evaluation results are the following :

Solver	DASSL		
Platform	Dymola	Scicos	OpenModelica
CPU time	1,31	5,2	57,875

Number of numercial jacobian	0	0	0
Number of zero crossing	10	10	10
Number of equations	148	148	148

The evaluation parameters are the following :

Integrator : Runge-Kutta
Simulation stop time : 80s
Tolerance : 10^{-6}
Number of intervals : 50 10^{6}.

The evaluation results are the following :

Solver	Runge-Kutta	
Platform	Dymola	OpenModelica
CPU time	15,7	60,259
Number of numerical jacobian	0	0
Number of zero crossing	10	10
Number of equations	148	148

4. Conclusions

Obtained simulation results are very close between the compared environments. This is a good point, which is due to the standardization operated by Modelica.

OpenModelica is much slower than Dymola and Scicos, for the case of the studied examples.
One reason, in our opinion, is the generation of .mat or .csv result files, that could not be disabled, and that leads to huge result file (hundreds of megabytes or even gigabytes).

One possibility to test this hypotheses is to use FMI generation wizard from OpenModelica, which can allow disabling result files generation. The execution of the generated FMU in another FMI-master can allow verifying this assumption.

5. Acknowledgements

We would like to acknowledge OpenModelica developers for their efficient and quick support. Their quick responses to our questions and bug reports were very helpful.

6. References

[1] Fritzson, P. Principles of Object-Oriented Modeling and Simulation with Modelica. Wiley-IEEE Computer Society. 2003.

[2] Wiebe, I. Brennverlauf und Kreisprozess von Verbrennungsmotoren. VEB Verlag Technik, Berlin. 1970.

[3] Richard, S., Font, G., Berr, F.L., Grasset, O., and Fremovici, M. On the use of system simulation to explore the potential of innovative combustion systems: Methodology and application to highly downsized SI engines running with ethanol-gasoline blends. JSAE Paper, 2011-04. 2011

[4] Colin, O., Benkenida, A., and Angelberger, C. A 3D modeling of mixing, ignition and combustion phenomena in highly stratified gasoline engines. Oil & Gas Science and Technology, 58, 47–62. 2003.

[5] Richard, S., Bougrine, S., Font, G., Lafossas, F.-A., and Le Berr, F. On the reduction of a 3D CFD combustion model to build a physical 0D model for simulating heat release, knock and pollutants in SI engines. Oil & Gas Science and Technology, 64, 223–242. 2009

[6] Barba C., Burkhardt C., Boulouchos K., Bargende M. (2000) A phenomenological combustion model for heat release rate prediction in high speed DI Diesel engines with common rail injection, SAE Technical Paper 2000-01-2933.