Chip simulation of automotive ECUs

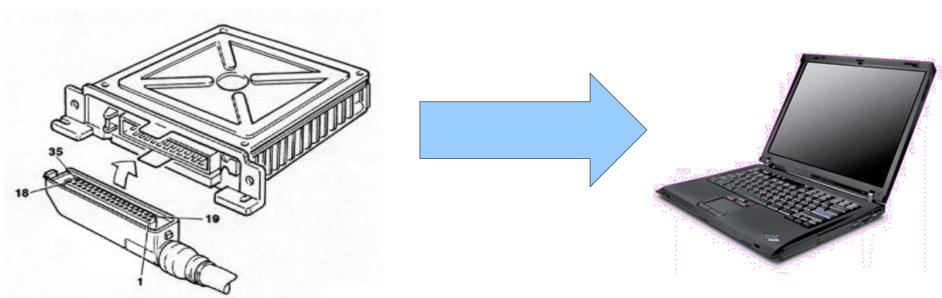
Jakob Mauss, QTronic GmbH

Matthias Simons, Daimler AG

9. Symposium
Steuerungssysteme für automobile Antriebe
Berlin-Tempelhof, 20.-21.09.2012

Chip simulation of automotive ECUS

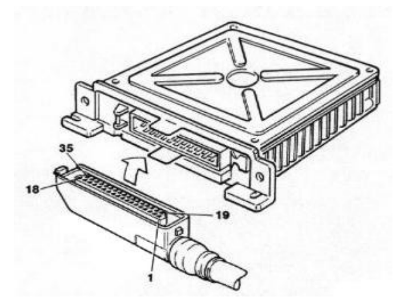
1. Motivation
2. Setting up a simulation
3. Performance
4. Limitations
5. Conclusion

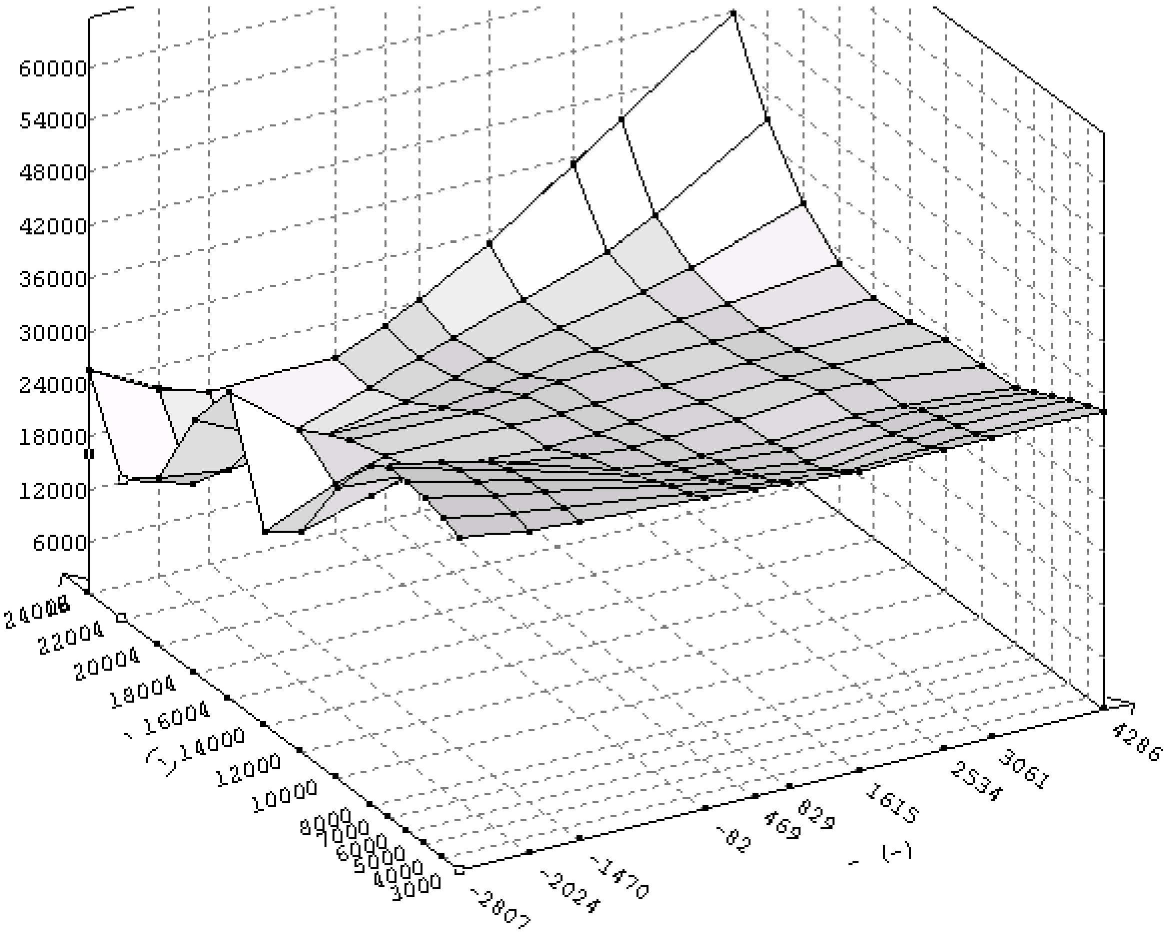


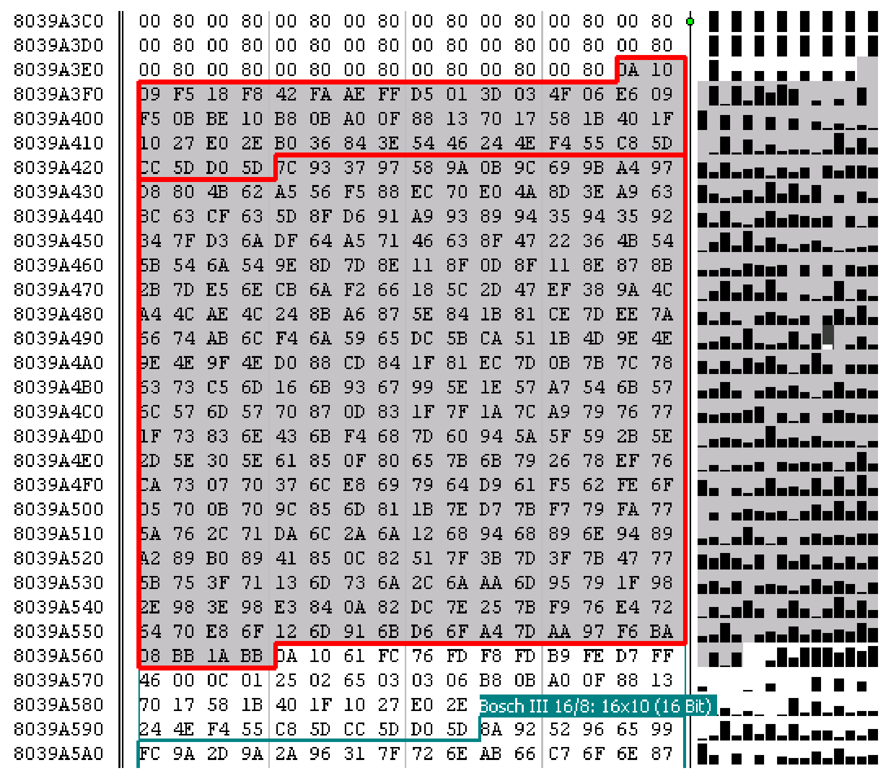
Motivation

ECU: more than 30.000 software parameter

Example: 16 x 10 map







ECU memory dump

Motivation

Engine calibration

- tune more than 30.000 ECU parameter
- done by the OEM, not by the supplier of the ECU

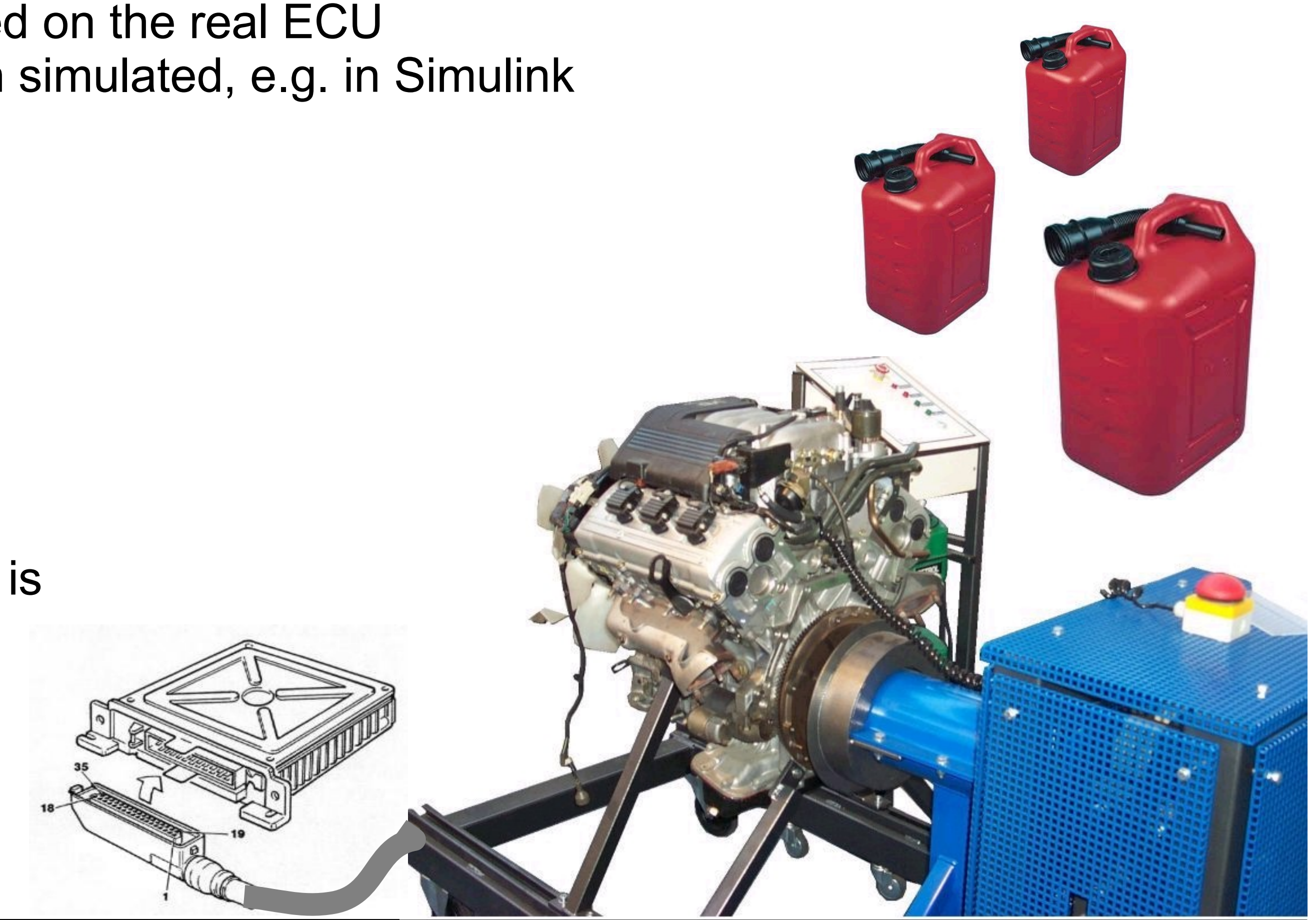
Process today

- automated optimization of stationary states
- real-time test rig or vehicle: based on the real ECU
- PC based: engine and ECU both simulated, e.g. in Simulink

Problems

- real-time test rig:
  - limited reproducibility
  - expensive (invest, operation)
  - slow (real time)

- PC: reverse engineering of ECU is
  - time consuming
  - complex
  - error prone



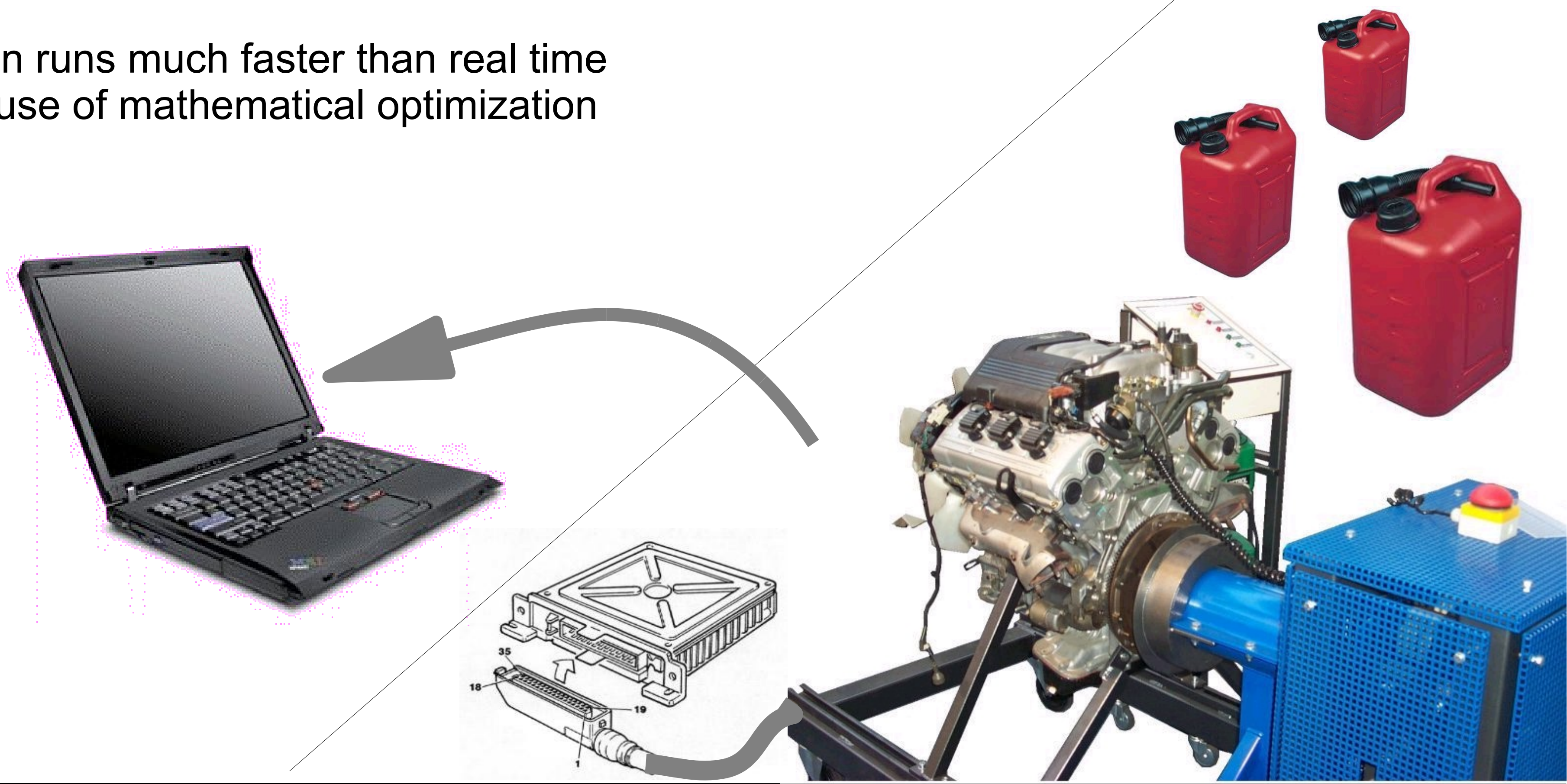
Motivation

Idea

move engine calibration (and other development tasks)
from test rig to PC

Benefit

- simulation runs much faster than real time
- enables use of mathematical optimization

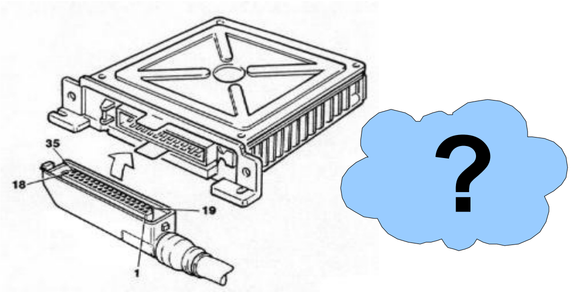


Motivation

Simulation of ECUs on PC:

Problem:

How to simulate ECU if **no C source or model** is available ?

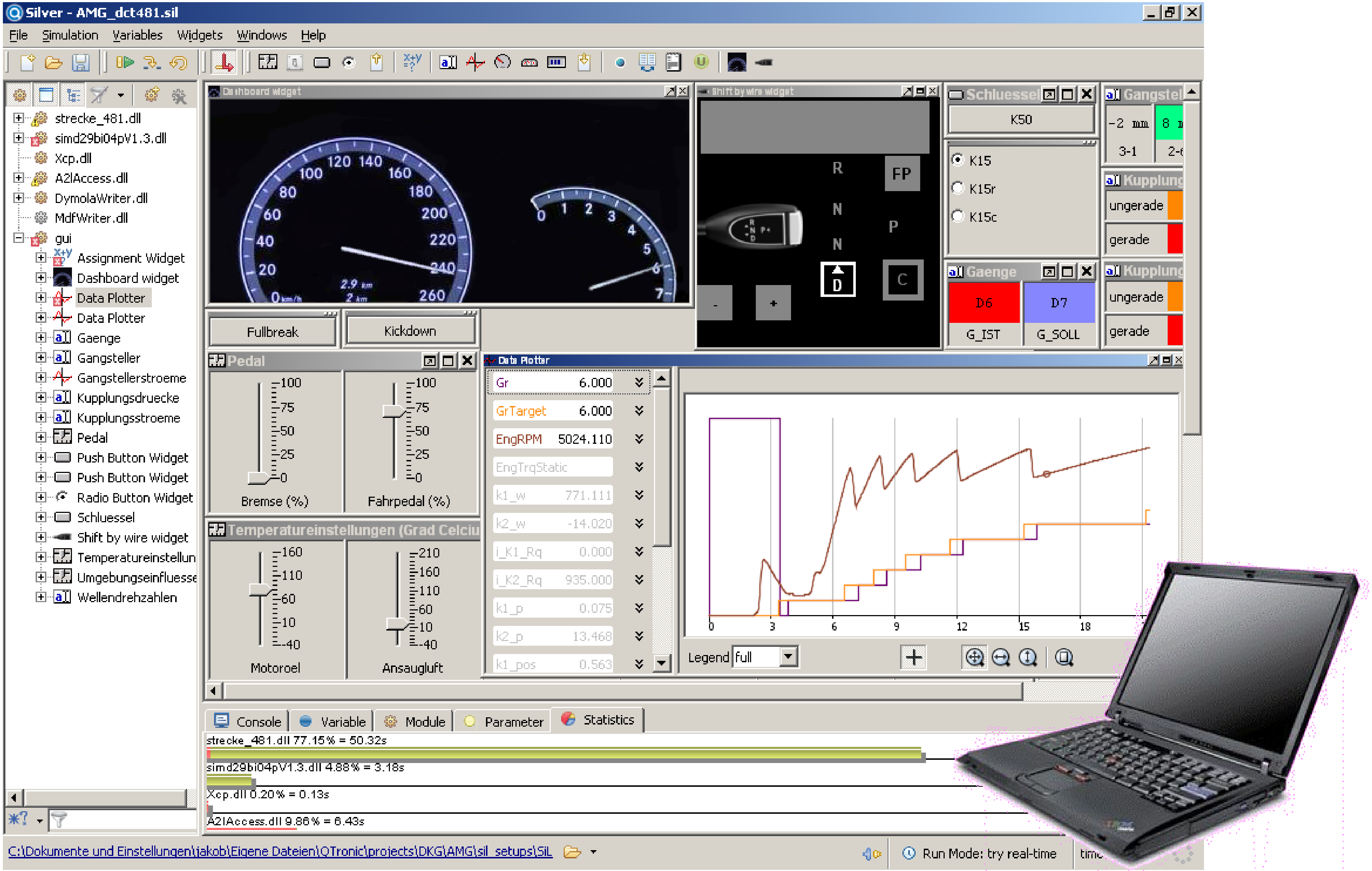


Ideas:

- Simulate the CPU based on the hex file
- Integrate this feature into MATLAB and QTronic Silver

Example - TCU Control Software in Silver

QTronic SIMULATION FOR ENGINEERING



Setting up a TriCore simulation

1. write spec.txt to specify what functions to run
2. step and debug the simulation in Silver debug mode
3. generate fast running SFunction or Silver module: runs without a2l and hex

Setting up a TriCore simulation

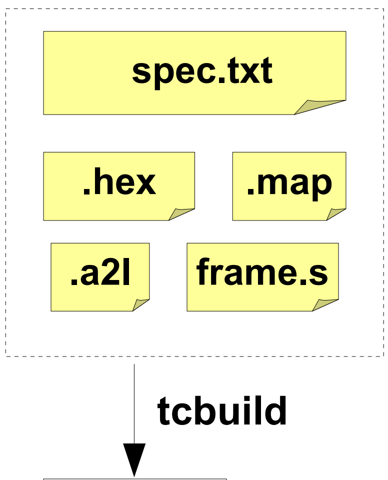
QTronic
SIMULATION FOR ENGINEERING

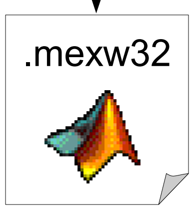
1. write spec.txt to specify what functions to run
2. step and debug the simulation in Silver debug mode
3. generate fast running SFunction or Silver module: runs without a2l and hex

```
# specification of sfunction or Silver module
hex_file(m12345.hex, TriCore_1.3.1)
a2l_file(m12345.a2l)
map_file(m12345.map)      # a TASKING or GNU map file
frame_file(frame.s)       # assembler code to emulate RTOS
frame_set(STEP_SIZE, 10) # Silver step size in ms
frame_set(TEXT_START, 0xa0000000) # location of frame code

# functions to be simulated, in order of execution
task_initial(ABCDE_ini)
task_initial(ABCDE_inisyn)
task_triggered(ABCDE_syn, trigger_ABCDE_syn)
task_periodic(ABCDE_20ms, 20, 0)
task_periodic(ABCDE_200ms, 200, 0)

# interface of the generated sfunction or Silver module
a2l_function_inputs(ABCDE)
a2l_function_outputs(ABCDE)
a2l_function_parameters_defined(ABCDE)
```

generated SFunction in MATLAB/Simulink

QTronic
SIMULATION FOR ENGINEERING

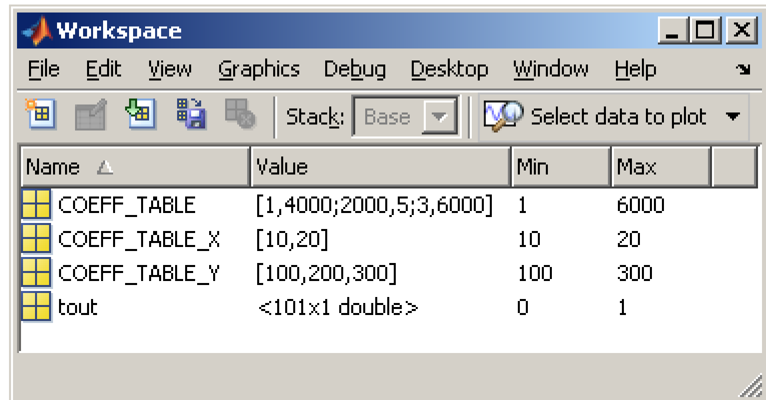




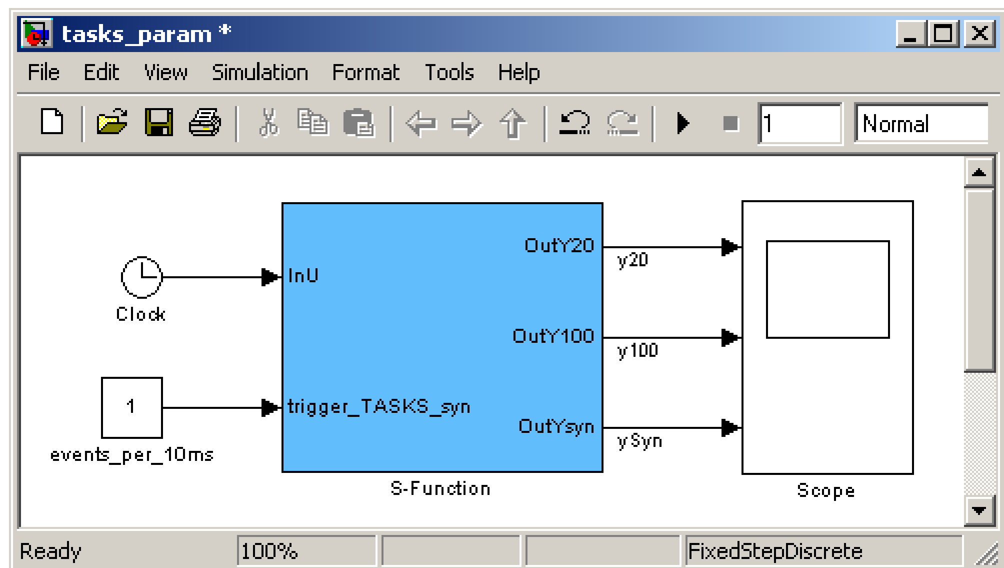
MATLAB/Simulink S-function
40 MIPS

default values for characteristics from HEX file as m script,

mask for S-function block and similar Simulink snippets

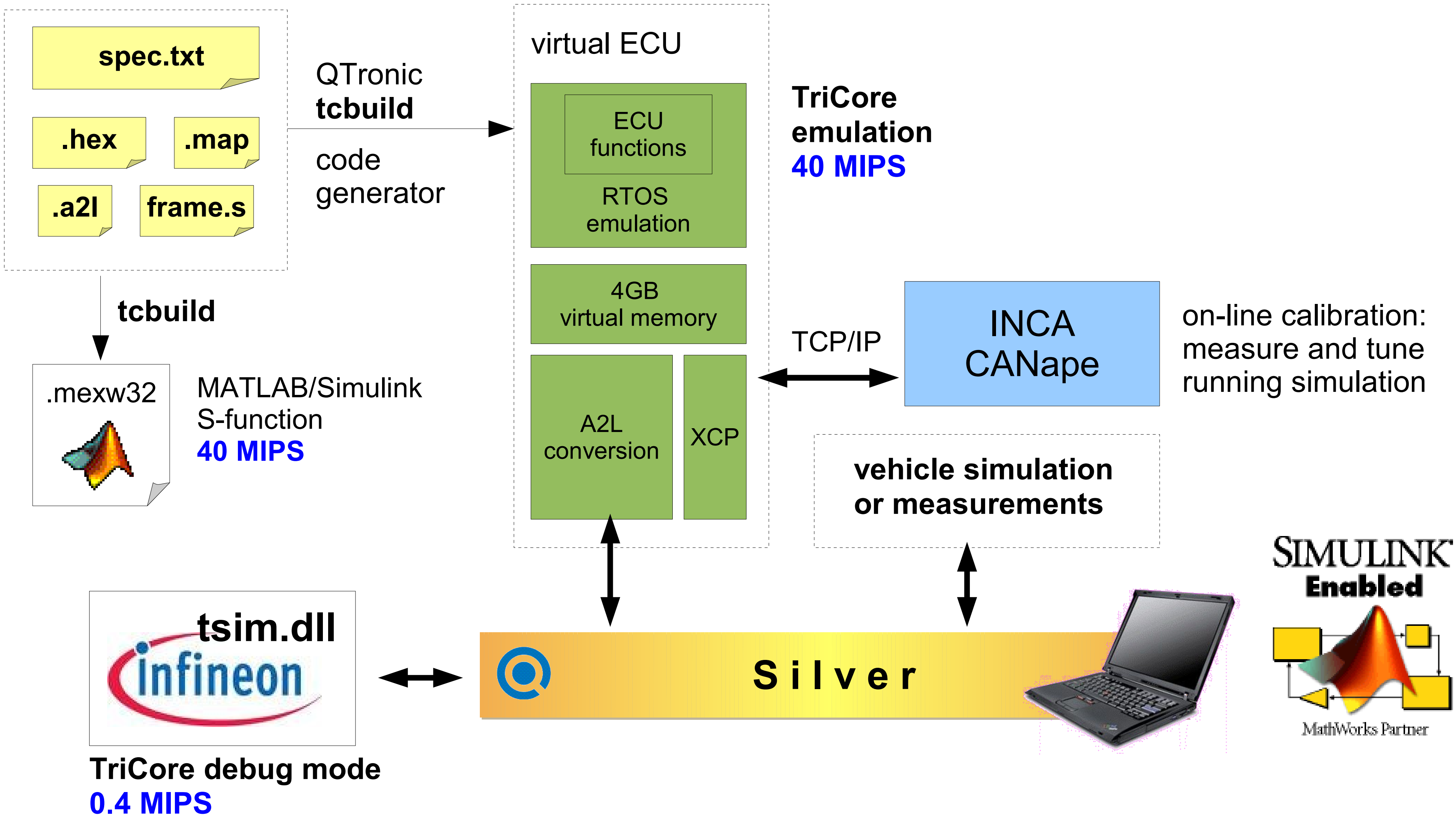


characteristics turned into MATLAB workspace variables
- read by S-function
- may be modified by script

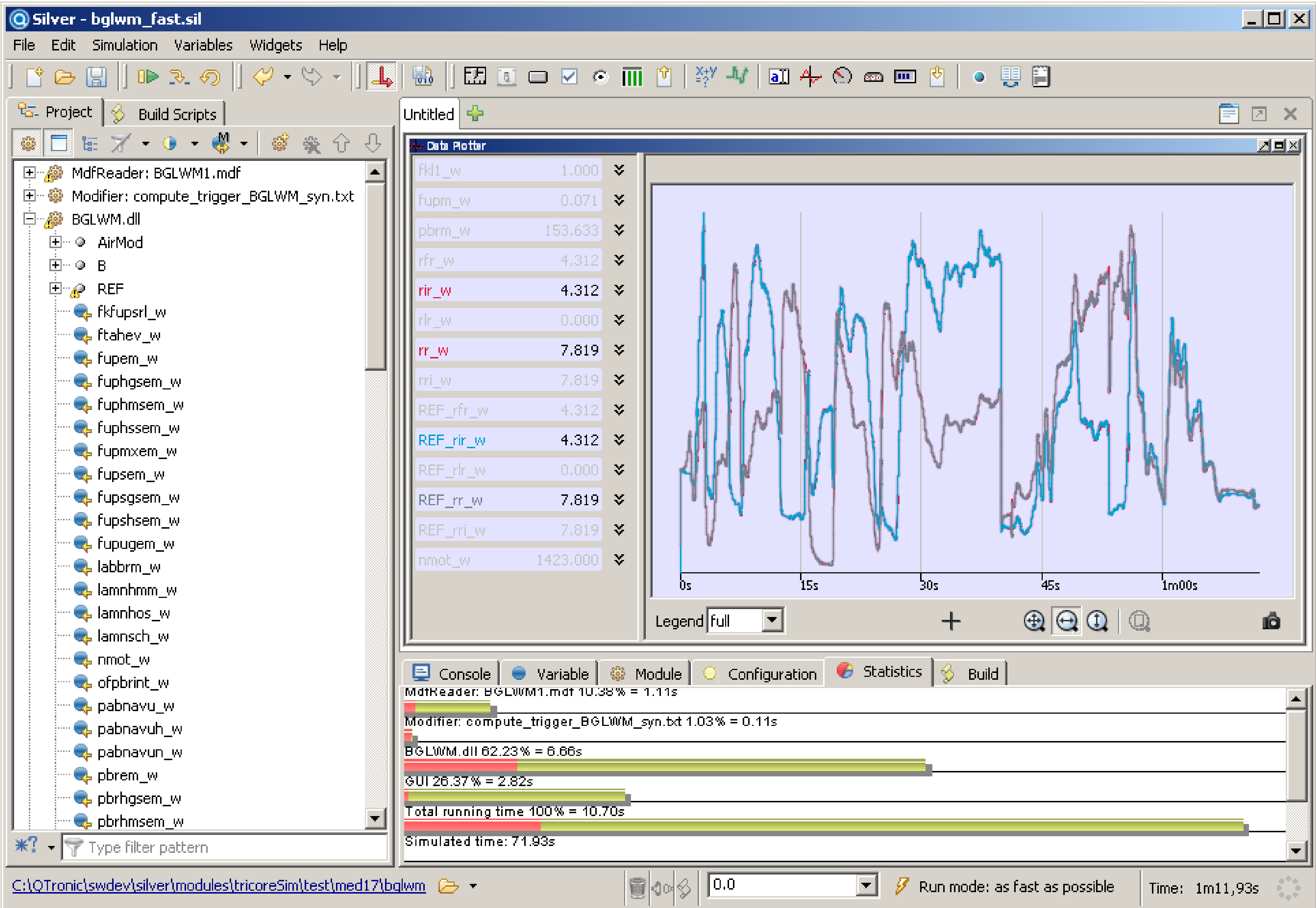


generated virtual ECU in Silver

QTronic
SIMULATION FOR ENGINEERING



Virtual ECU running in Silver: MED17



Performance and Limitations

QTronic SIMULATION FOR ENGINEERING

Run complex function for a measured scenario, 3.5 minutes

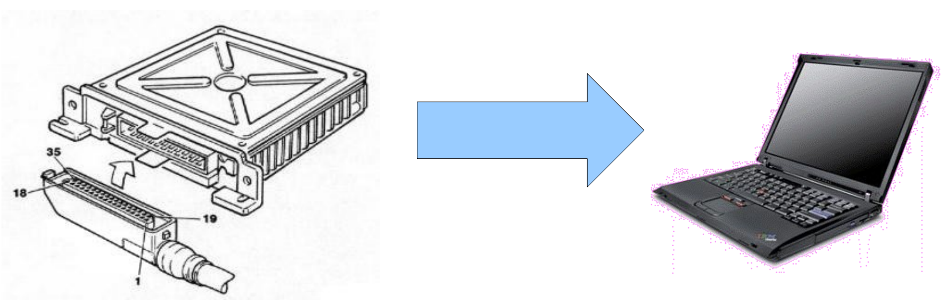
target	**execution time**	**MIPS**
Silver in debug mode	919.15 sec	0.41
generated Silver module or MATLAB/Simulink SFunction	9.30 sec	40.80
MED17 with TC1797, 180 Mhz	210.00 sec	270

Limitations:
- instruction accurate, but not cycle accurate
- based on TriCore specification: 'silicon bugs' are not simulated
- PCP, CAN controllers and other on chip devices not modeled

Summary

ECU simulation on Windows PC

- without expensive reverse engineering
- without access to ECU source files
- based on HEX, MAP and A2L file
- low work effort for modeling
- high accuracy of model
- application example: automated calibration



- works for TriCore processors: TC1796, TC1797, TC1798, ...
- performance: 40 MIPS