



# Chip simulation of automotive ECUs

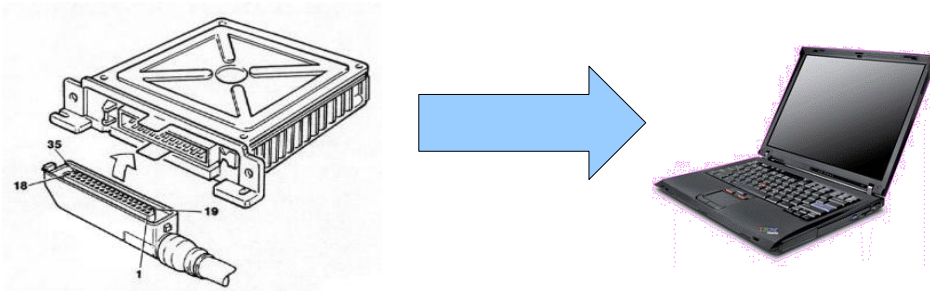
Jakob Mauss, QTronic GmbH

Matthias Simons, Daimler AG

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

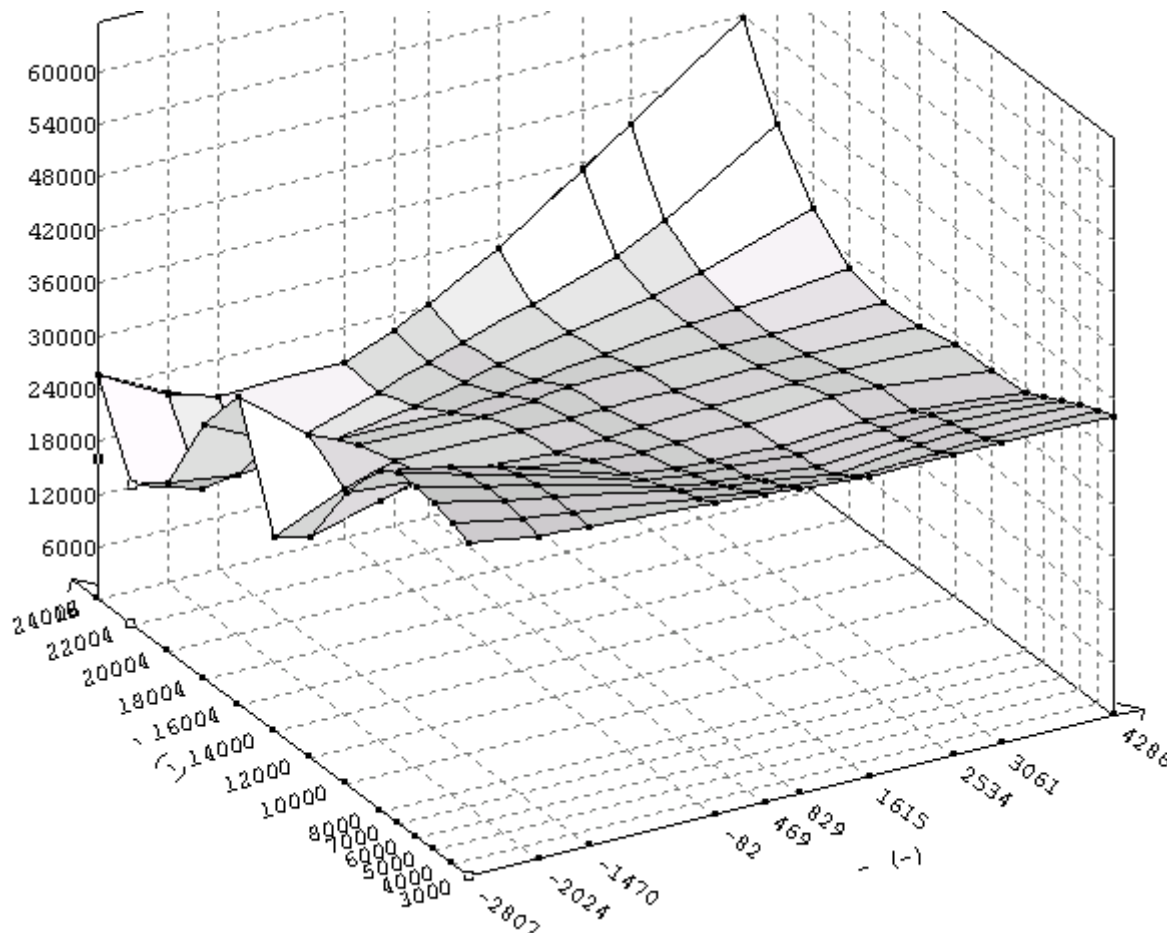
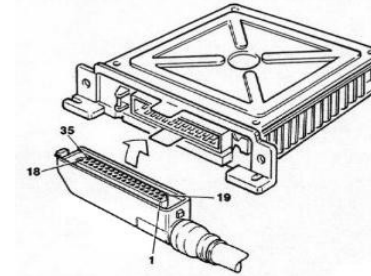
## Chip simulation of automotive ECUS

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



**ECU: more than 30.000 software parameter**

**Example: 16 x 10 map**



8039A3C0	00 80 00 80 00 80 00 80 00 80 00 80 00 80 00 80	
8039A3D0	00 80 00 80 00 80 00 80 00 80 00 80 00 80 00 80	
8039A3E0	00 80 00 80 00 80 00 80 00 80 00 80 00 80 00 80	
8039A3F0	09 F5 18 F8 42 FA AE FF D5 01 3D 03 4F 06 E6 09	
8039A400	F5 0B BE 10 B8 0B A0 0F 88 13 70 17 58 1B 40 1F	
8039A410	10 27 E0 2E B0 36 84 3E 54 46 24 4E F4 55 C8 5D	
8039A420	CC 5D D0 5D 7C 93 37 97 58 9A 0B 9C 69 9B A4 97	
8039A430	08 80 4B 62 A5 56 F5 88 EC 70 E0 4A 8D 3E A9 63	
8039A440	3C 63 CF 63 5D 8F D6 91 A9 93 89 94 35 94 35 92	
8039A450	34 7F D3 6A DF 64 A5 71 46 63 8F 47 22 36 4B 54	
8039A460	5B 54 6A 54 9E 8D 7D 8E 11 8F 0D 8F 11 8E 87 8B	
8039A470	2B 7D E5 6E CB 6A F2 66 18 5C 2D 47 EF 38 9A 4C	
8039A480	A4 4C AE 4C 24 8B A6 87 5E 84 1B 81 CE 7D EE 7A	
8039A490	56 74 AB 6C F4 6A 59 65 DC 5B CA 51 1B 4D 9E 4E	
8039A4A0	9E 4E 9F 4E D0 88 CD 84 1F 81 EC 7D 0B 7B 7C 78	
8039A4B0	63 73 C5 6D 16 6B 93 67 99 5E 1E 57 A7 54 6B 57	
8039A4C0	5C 57 6D 57 70 87 0D 83 1F 7F 1A 7C A9 79 76 77	
8039A4D0	1F 73 83 6E 43 6B F4 68 7D 60 94 5A 5F 59 2B 5E	
8039A4E0	2D 5E 30 5E 61 85 0F 80 65 7B 6B 79 26 78 EF 76	
8039A4F0	CA 73 07 70 37 6C E8 69 79 64 D9 61 F5 62 FE 6F	
8039A500	05 70 0B 70 9C 85 6D 81 1B 7E D7 7B F7 79 FA 77	
8039A510	5A 76 2C 71 DA 6C 2A 6A 12 68 94 68 89 6E 94 89	
8039A520	A2 89 B0 89 41 85 0C 82 51 7F 3B 7D 3F 7B 47 77	
8039A530	5B 75 3F 71 13 6D 73 6A 2C 6A AA 6D 95 79 1F 98	
8039A540	2E 98 3E 98 E3 84 0A 82 DC 7E 25 7B F9 76 E4 72	
8039A550	54 70 E8 6F 12 6D 91 6B D6 6F A4 7D AA 97 F6 BA	
8039A560	08 BB 1A BB DA 10 61 FC 76 FD F8 FD B9 FE D7 FF	
8039A570	46 00 0C 01 25 02 65 03 03 06 B8 0B A0 0F 88 13	
8039A580	70 17 58 1B 40 1F 10 27 E0 2E Bosch III 16/8: 16x10 (16 Bit)	
8039A590	24 4E F4 55 C8 5D CC 5D D0 5D 8A 92 52 96 65 99	
8039A5A0	FC 9A 2D 9A 2A 96 31 7F 72 6E AB 66 C7 6F 6E 87	

**ECU memory dump**

## Engine calibration

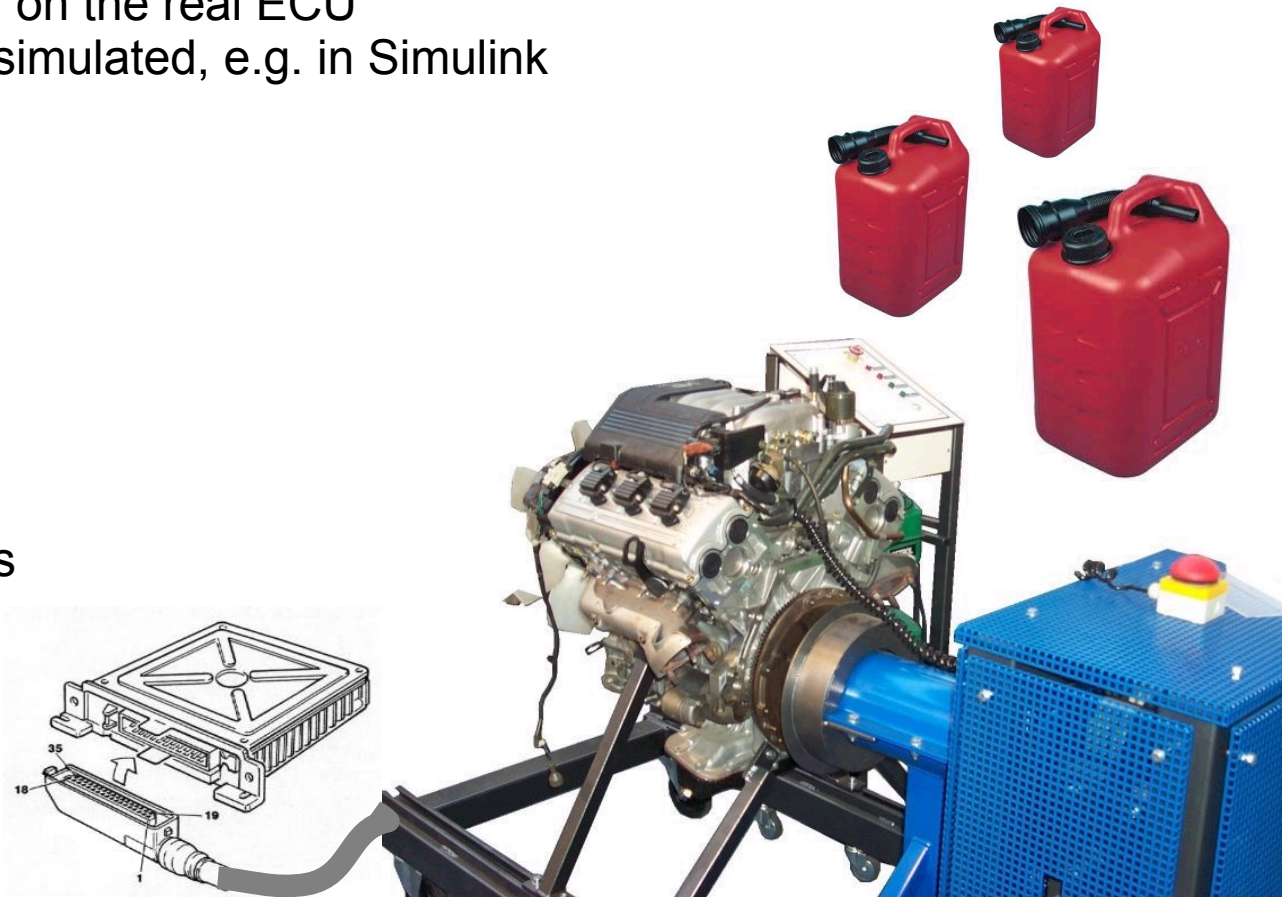
- tune more than 30.000 ECU parameter
- done by the OEM, not by the supplier of the ECU

## 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



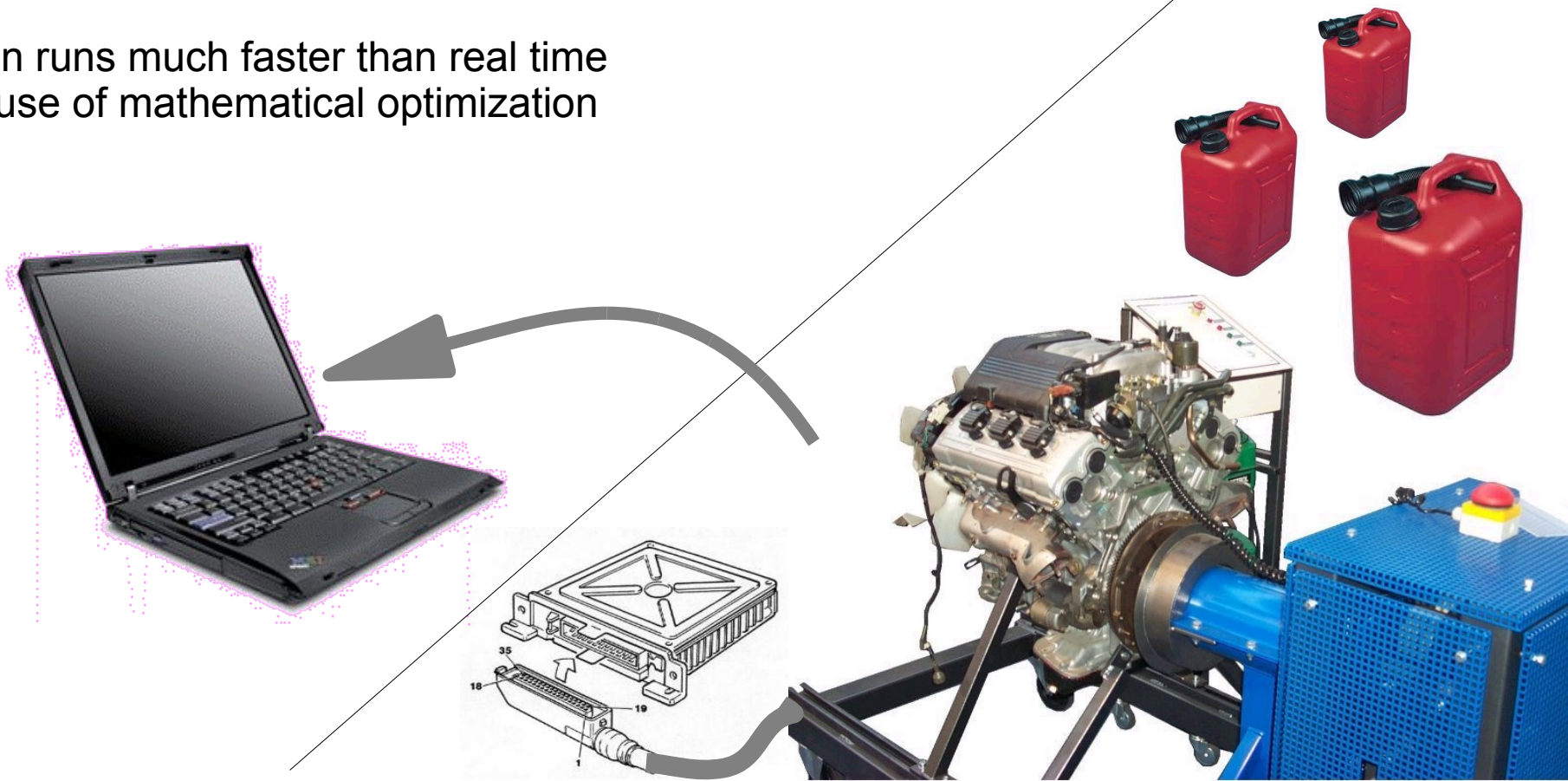


## Idea

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

## Benefit

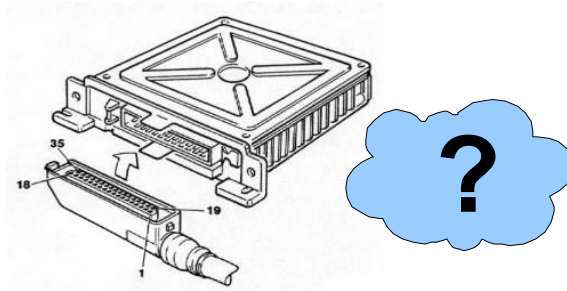
- simulation runs much faster than real time
- enables use of mathematical optimization



## Simulation of ECUs on PC:

### Problem:

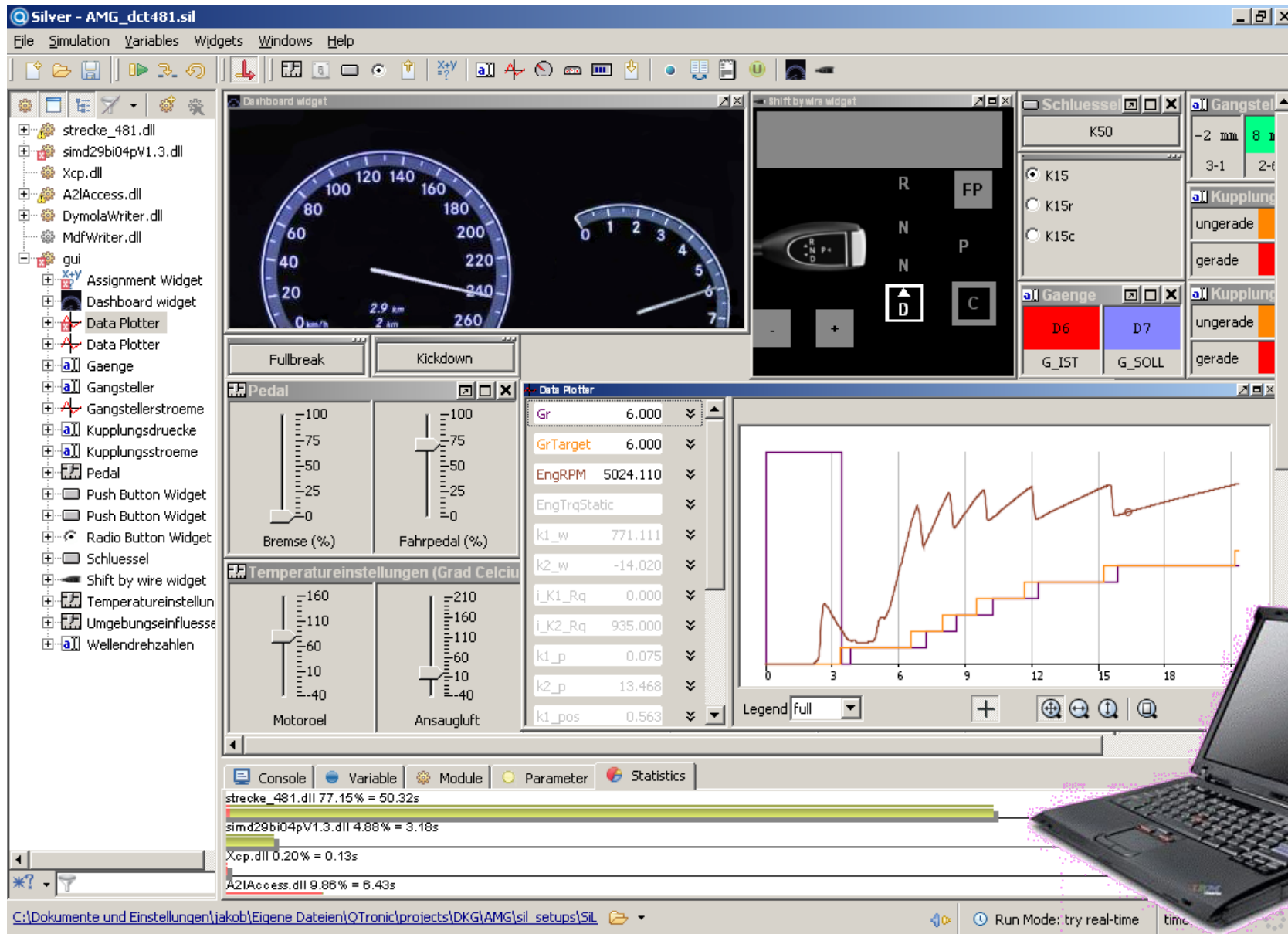
How to simulate ECU if no C source or model is available ?



### Ideas:

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

# Example - TCU Control Software in Silver

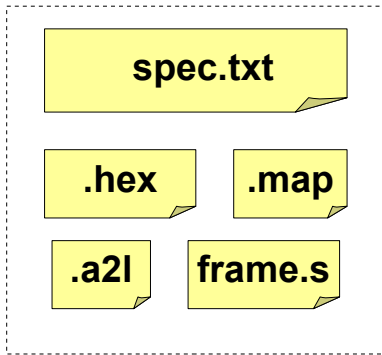


1. write spec.txt to specify what functions to run
2. step and debug the simulation in Silver debug mode
3. generate fast running SFunction or Silver module: runs without a2l and hex



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

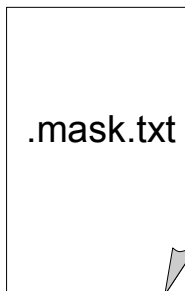
```
01 # specification of sfunction or Silver module
02 hex_file(m12345.hex, TriCore_1.3.1)
03 a2l_file(m12345.a2l)
04 map_file(m12345.map)          # a TASKING or GNU map file
05 frame_file(frame.s)          # assembler code to emulate RTOS
06 frame_set(STEP_SIZE, 10)     # Silver step size in ms
07 frame_set(TEXT_START, 0xa0000000) # location of frame code
08
09 # functions to be simulated, in order of execution
10 task_initial(ABCDE_ini)
11 task_initial(ABCDE_inisyn)
12 task_triggered(ABCDE_syn, trigger_ABCDE_syn)
13 task_periodic(ABCDE_20ms, 20, 0)
14 task_periodic(ABCDE_200ms, 200, 0)
15
16 # interface of the generated sfunction or Silver module
17 a2l_function_inputs(ABCDE)
18 a2l_function_outputs(ABCDE)
19 a2l_function_parameters_defined(ABCDE)
```



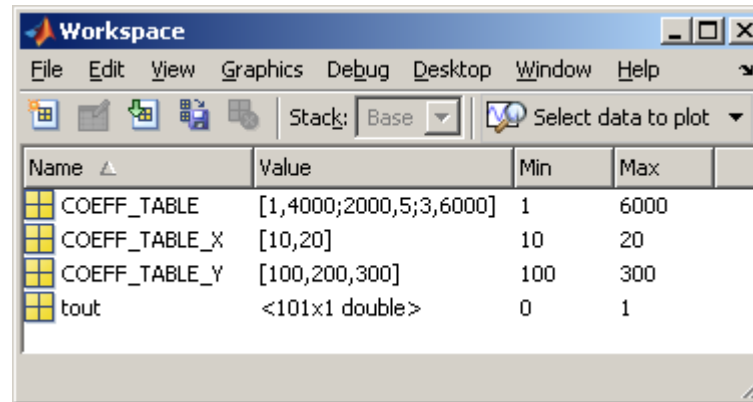
tcbuild



MATLAB/Simulink  
S-function  
**40 MIPS**

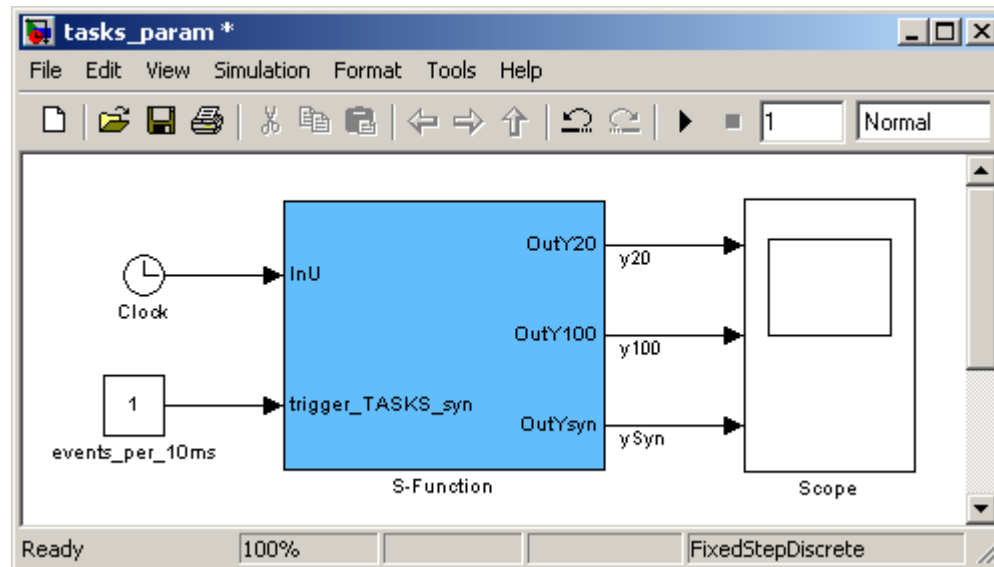


default values for  
characteristics from  
HEX file as m script,  
mask for S-function  
block and similar  
Simulink snippets

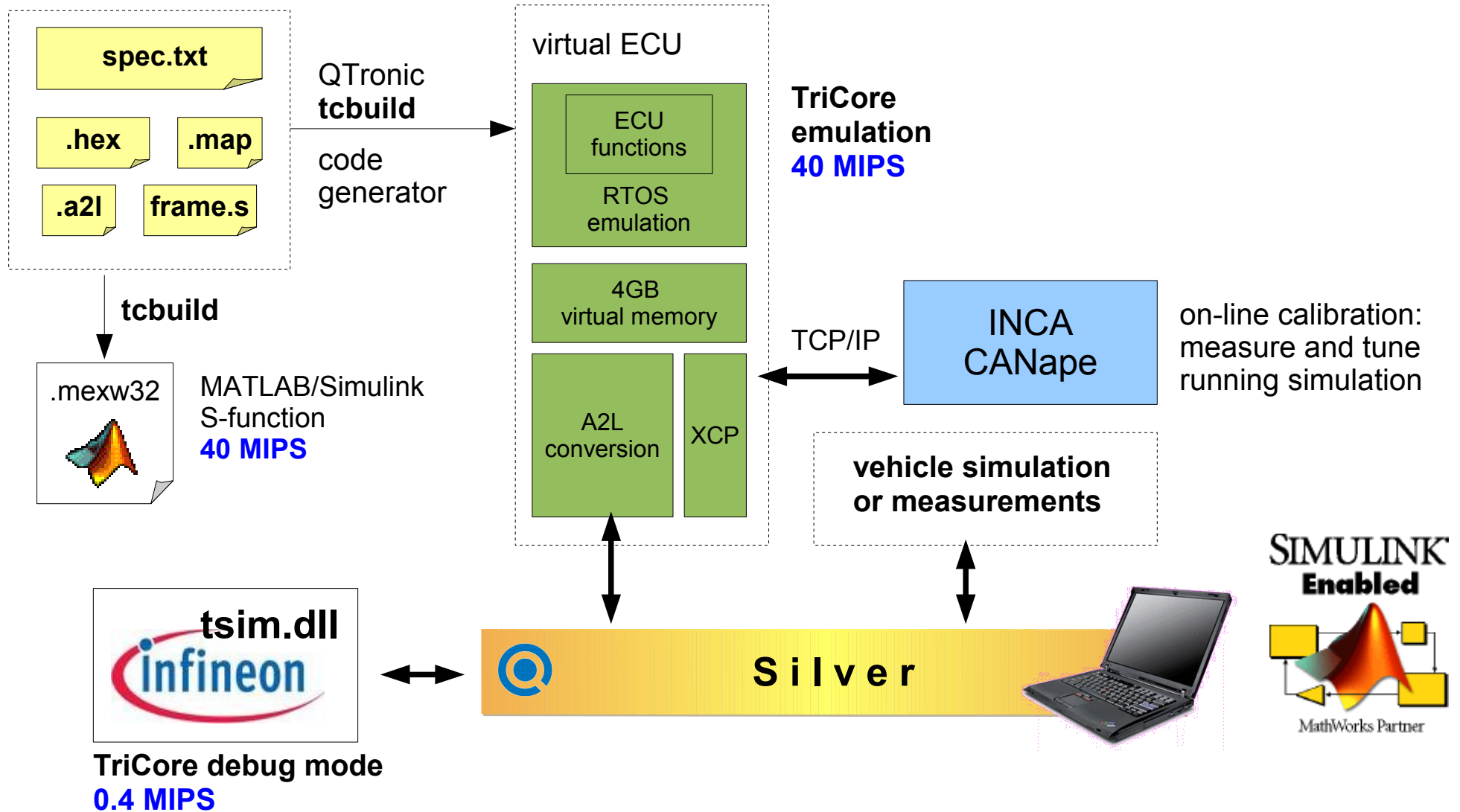


Name	Value	Min	Max
COEFF_TABLE	[1,4000;2000,5;3,6000]	1	6000
COEFF_TABLE_X	[10,20]	10	20
COEFF_TABLE_Y	[100,200,300]	100	300
tout	<101x1 double>	0	1

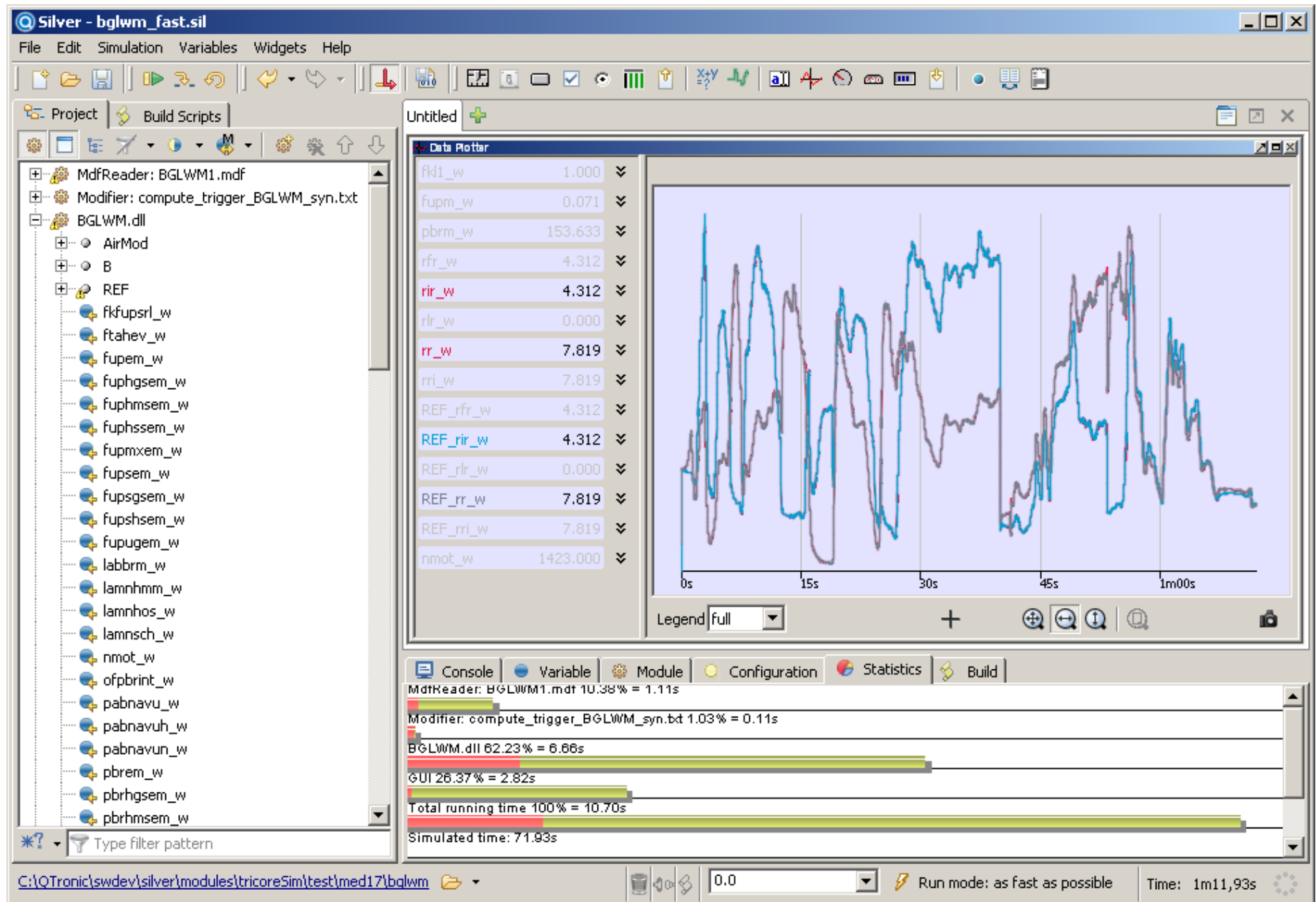
characteristics turned into  
MATLAB workspace variables  
- read by S-function  
- may be modified by script



# generated virtual ECU in Silver



# Virtual ECU running in Silver: MED17



**Run complex function for a measured scenario, 3.5 minutes**

<b>target</b>	<b>execution time</b>	<b>MIPS</b>
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

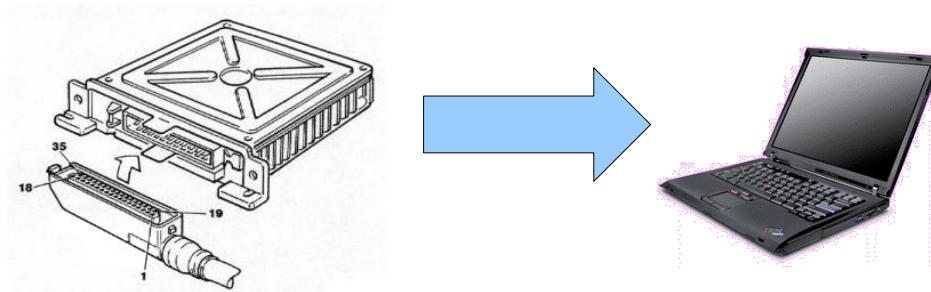
## **Limitations:**

- 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



## 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