Virtual Development for GTM with COSIDE
使用 COSIDE 进行 GTM 虚拟开发

COSEDA Technologies GmbH

Karsten Einwich

COSEDA Technologies

Company 公司

- Founded 2015 as spin-off from Fraunhofer
- More than 20 years experiences in Virtual Prototyping
- Contributed to several standardization activities (especially SystemC/SystemC AMS)
- We can also analog
- Tools to make complex modelling techniques easy usable
- www.coseda-tech.com

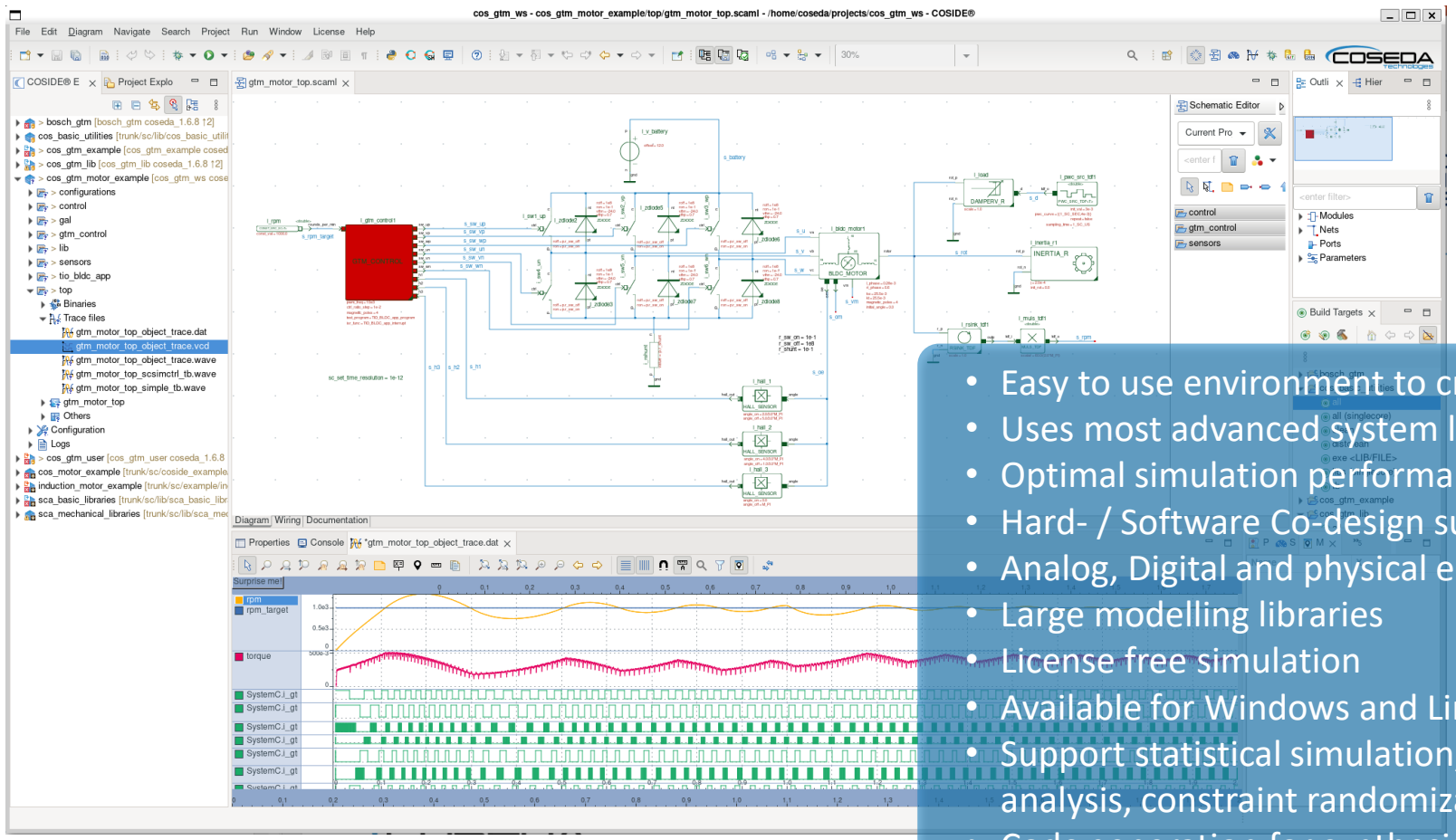
Motivation for Virtual Prototypes / Digital Twins

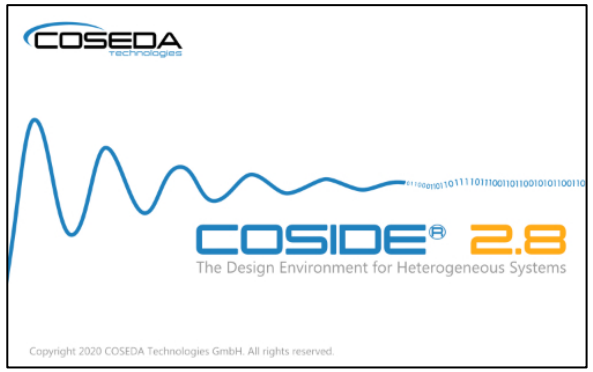
虚拟原型 / 数字孪生的动机

- Early available – early start of software development
- Significant cheaper than hardware prototypes
- Much better debugable – completely introspectable, reproducible
- Check of practical not realizable or expensive scenarios and parameter combinations
- Fast and cheap check of numerous realization variants
- Reference for implementation

COSIDE® for Virtual Prototyping

用于虚拟样机的 COSIDE®





- Easy to use environment to create virtual prototypes
- Uses most advanced system level modelling techniques
- Optimal simulation performance
- Hard- / Software Co-design support
- Analog, Digital and physical environment modelling support
- Large modelling libraries
- License free simulation
- Available for Windows and Linux computers
- Support statistical simulation, fault injection, regression testing, static analysis, constraint randomization, …
- Code generation for synthesis and high level synthesis

COSIDE Supported Hard- / Software Virtual Prototyping Techniques

COSIDE 支持的硬件/软件虚拟样机技术

- Host Compiled Software Modelling

  TLM utility libraries, examples, GTM Modelling Library throughput models

- Fast Models based on just in time compile techniques (Loosely timed)

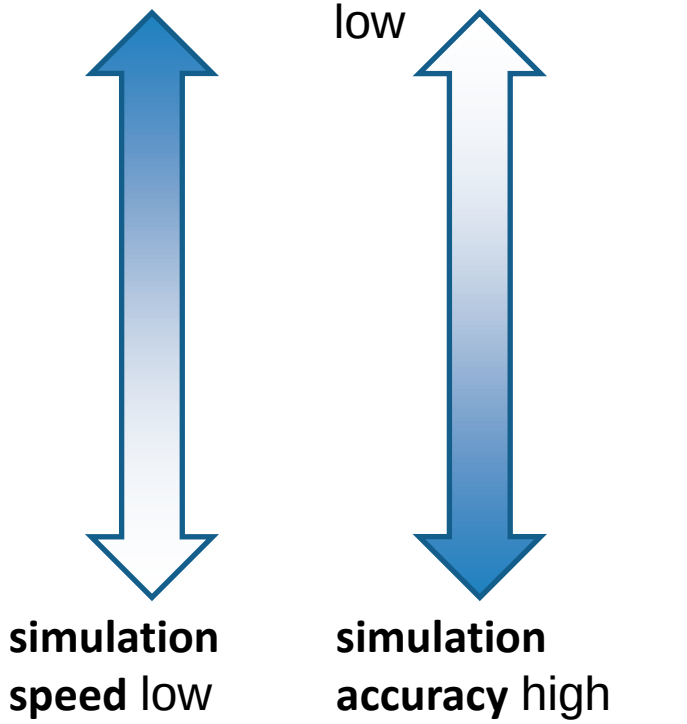
  MachineWare, ARM fast models

- Instruction timing accurate models (Approximately timed)

  Gem5, Infineon Aurix Integration, ARM performance models

- Cycle accurate models

  Verilator, Cadence, Synopsys, Siemens, Xilinx Integration



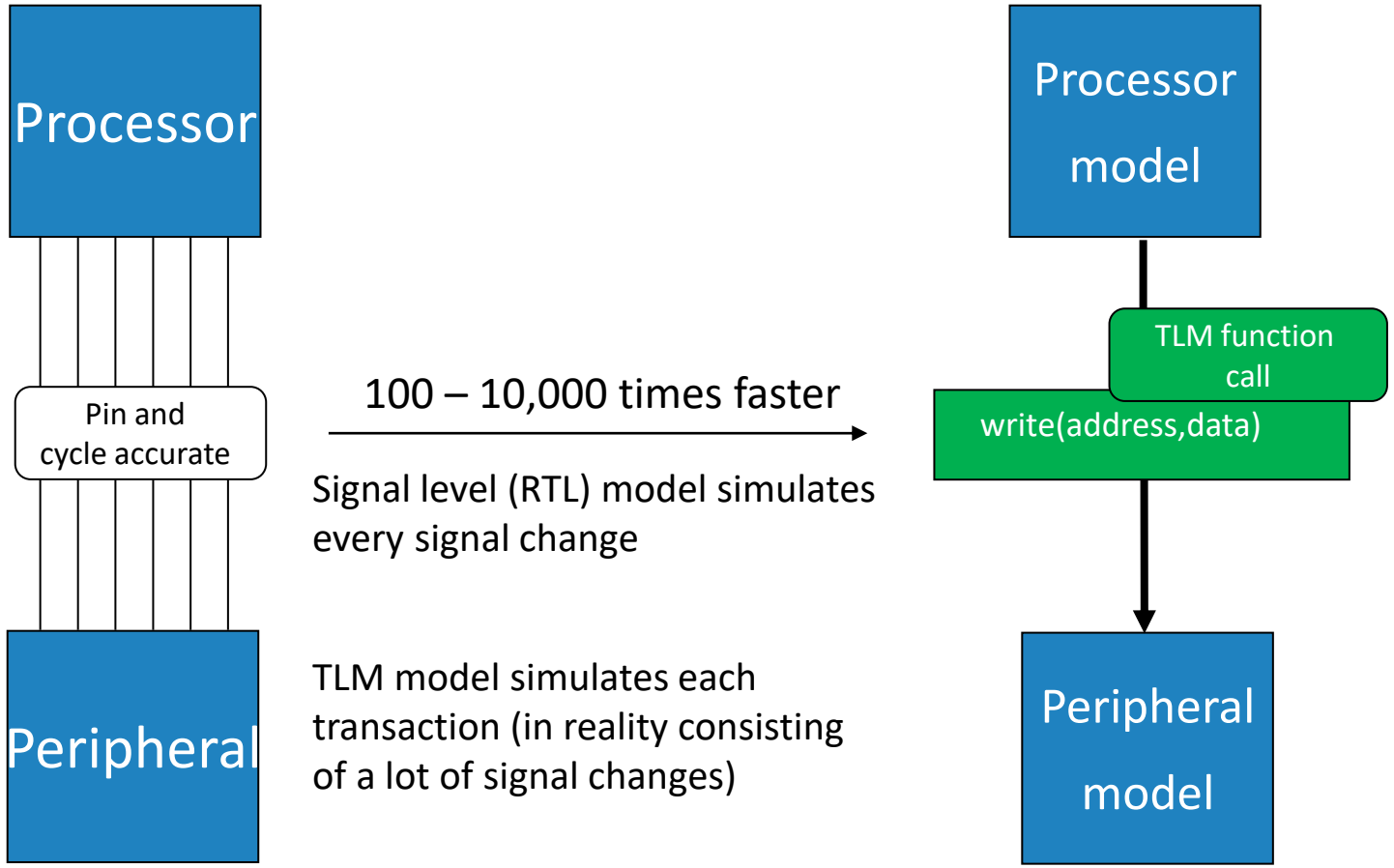
Virtual BOSCH GTM

虚拟博世 GTM

- Virtual GTM core model is provided by BOSCH
- Approximately timed SystemC model with a high accuracy
- Uses advanced modelling techniques like TLM to achieve optimal simulation performance
- COSIDE integrates the BOSCH provided model and provides add-on libraries to simplify the usage
- Highly configurable Interrupt modelling library provided
- Different configurations are provided
- Processor / Controller independent

Principle of used Modelling Techniques – Transaction Level Modelling (TLM)

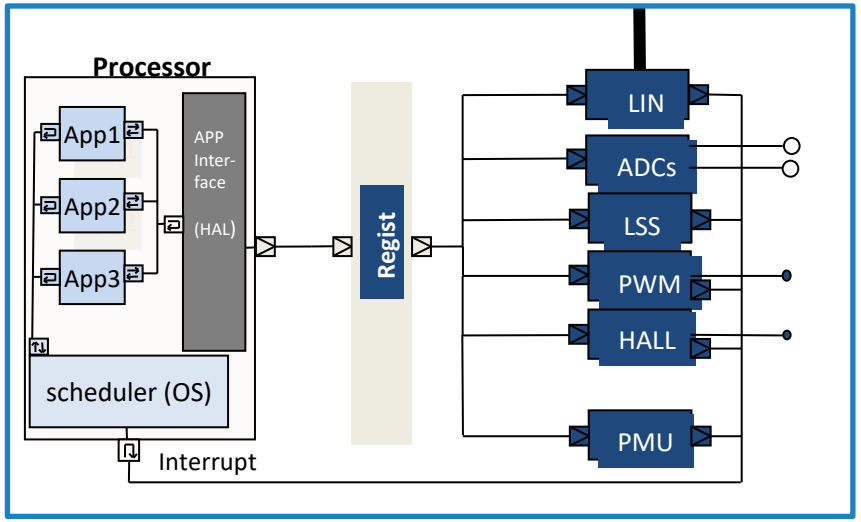
所用建模技术原理——事务级建模（TLM）

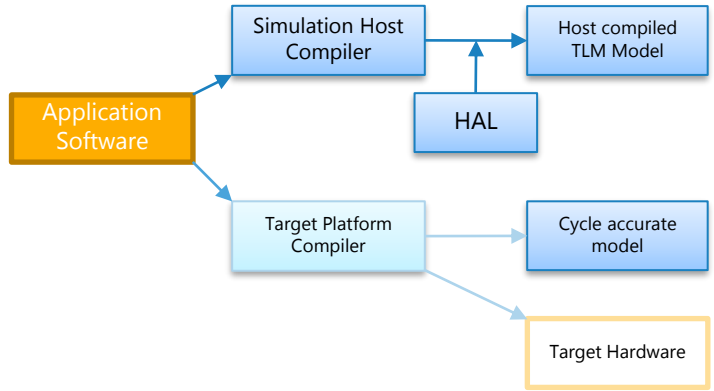


Principle of used Modelling Techniques – Host compiled software modelling

使用建模技术的原理 – 主机编译的软件建模

- The **behavior** (not the timing) of software written in C/C++ is to the greatest extent **independent from** the **processor**
- **Software is layered** – the lowest layer manages the communication (hardware abstraction layer – HAL)
- The application software is **compiled for the simulation host** and the HAL is re-implemented to manage the communication with the model

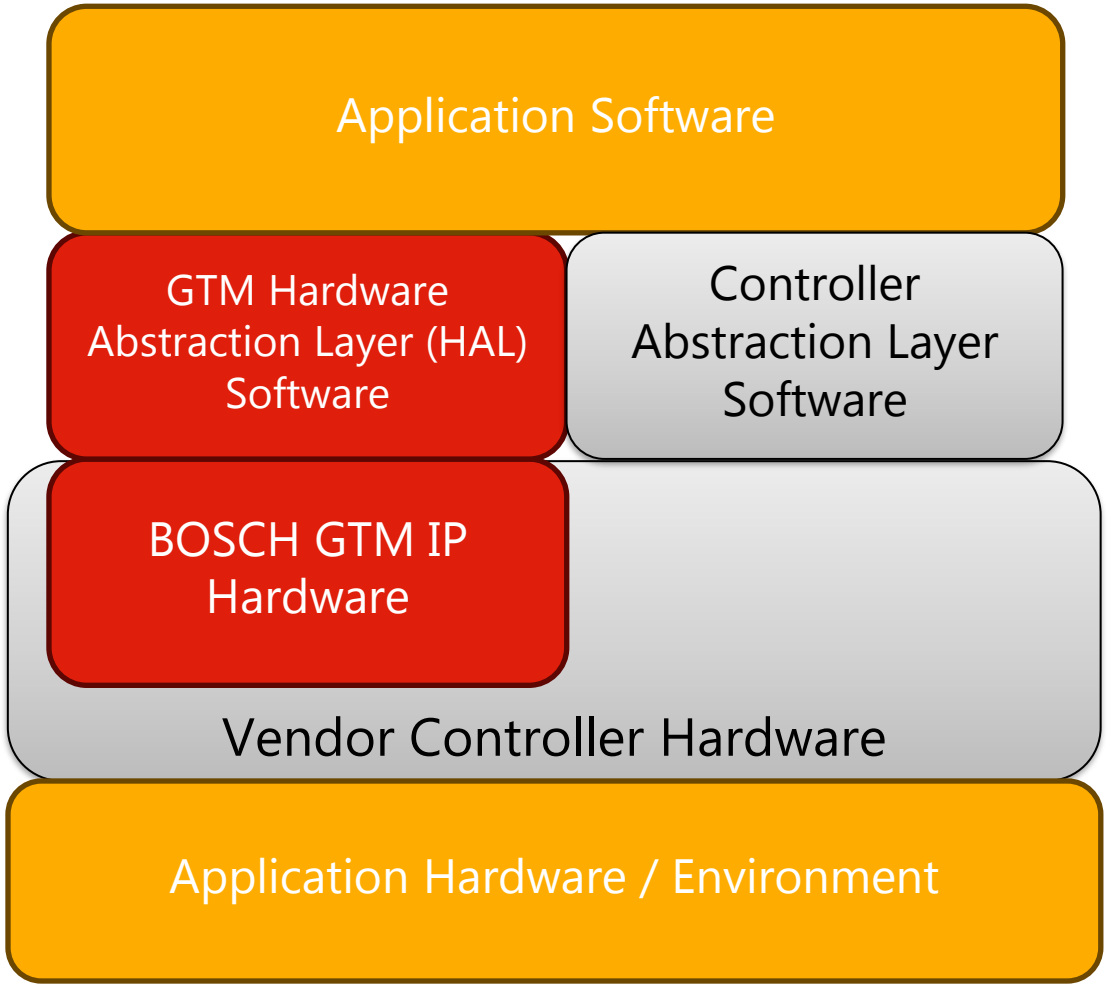




Abstract view to a system which integrates the BOSCH GTM

集成 BOSCH GTM 的系统的抽象视图

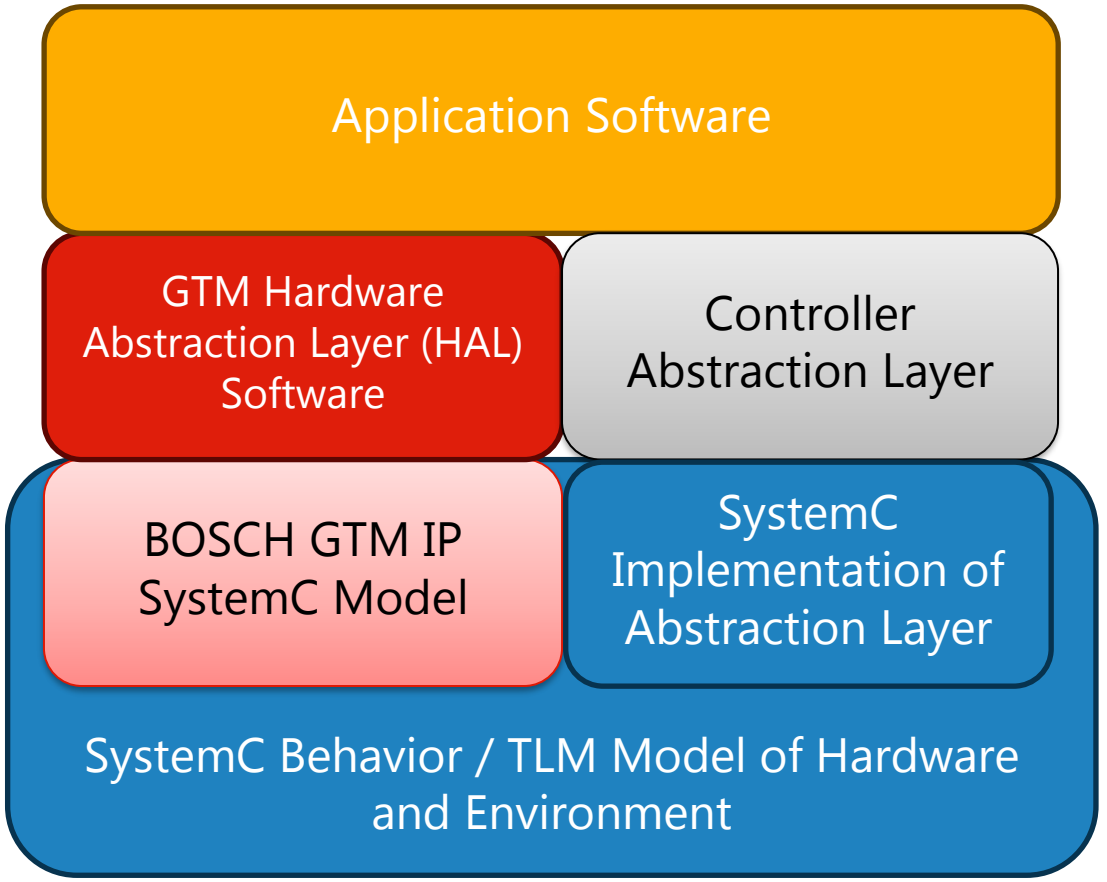
- Timing critical behavior is realized within the BOSCH GTM
- Application software becomes to the greatest extent timing and processor independent
- BOSCH provides a HAL to program the GTM
- The HAL is processor independent



Principle of the BOSCH GTM Model Integration

BOSCH GTM模型集成原理

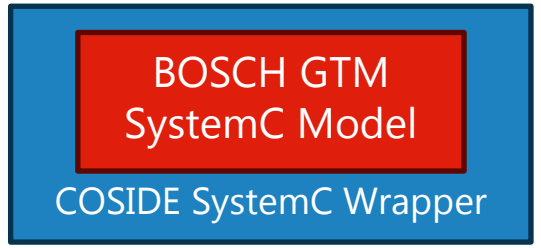
- BOSCH GTM SystemC Model is highly timing accurate
- Controlled by the same HAL like the vendor integrated hardware
- GTM integrated controller (MCS) can be loaded via the HAL with the original binary code
- Model runs fast enough to simulate application scenarios

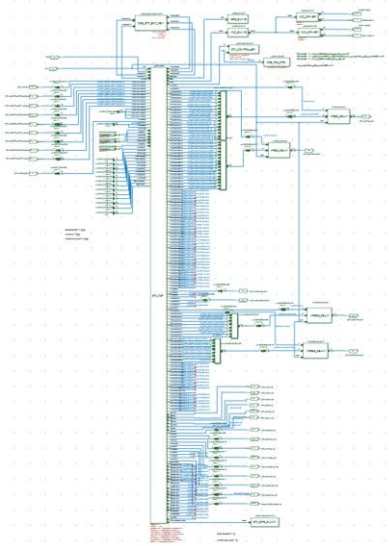


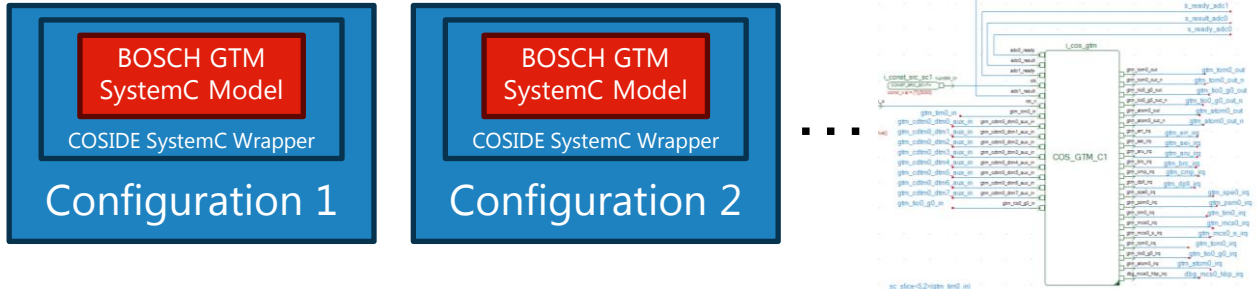
BOSCH GTM Integration in COSIDE

COSIDE 中的 BOSCH GTM 集成

- The model provided by BOSCH supports all possible GTM configurations
- In cooperation with BOSCH different configuration (may on customer request) are provided
- A configuration reduces the GTM complexity to the application and/or customer needs



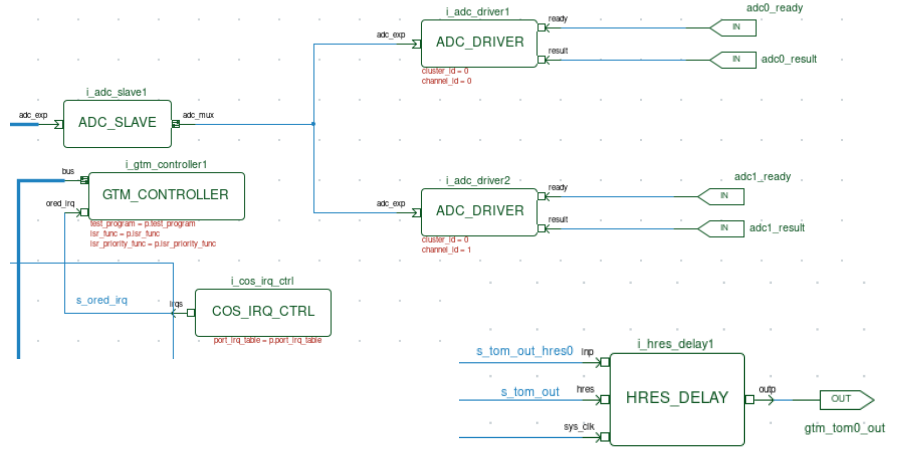


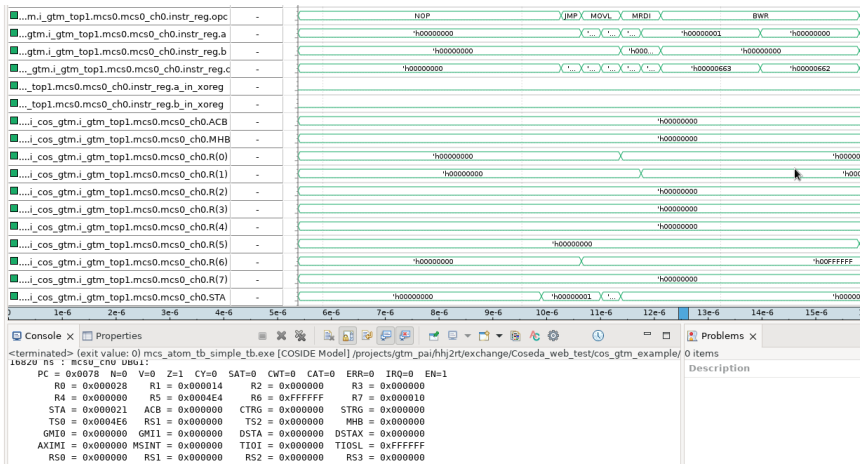


BOSCH GTM COSIDE Integration and Library Feature

BOSCH GTM COSIDE 集成和库功能

- Configurable Interrupt handling
- Generic ADC‘s with control logic
- HRES handling and logic
- Generic controller Model
- Controller program and Interrupt service routine (ISR) provided as parameter (lambda function)
- Debug capabilities and Pre-defined Traces
- Hardware Debugger (MCD) Interface (in development)

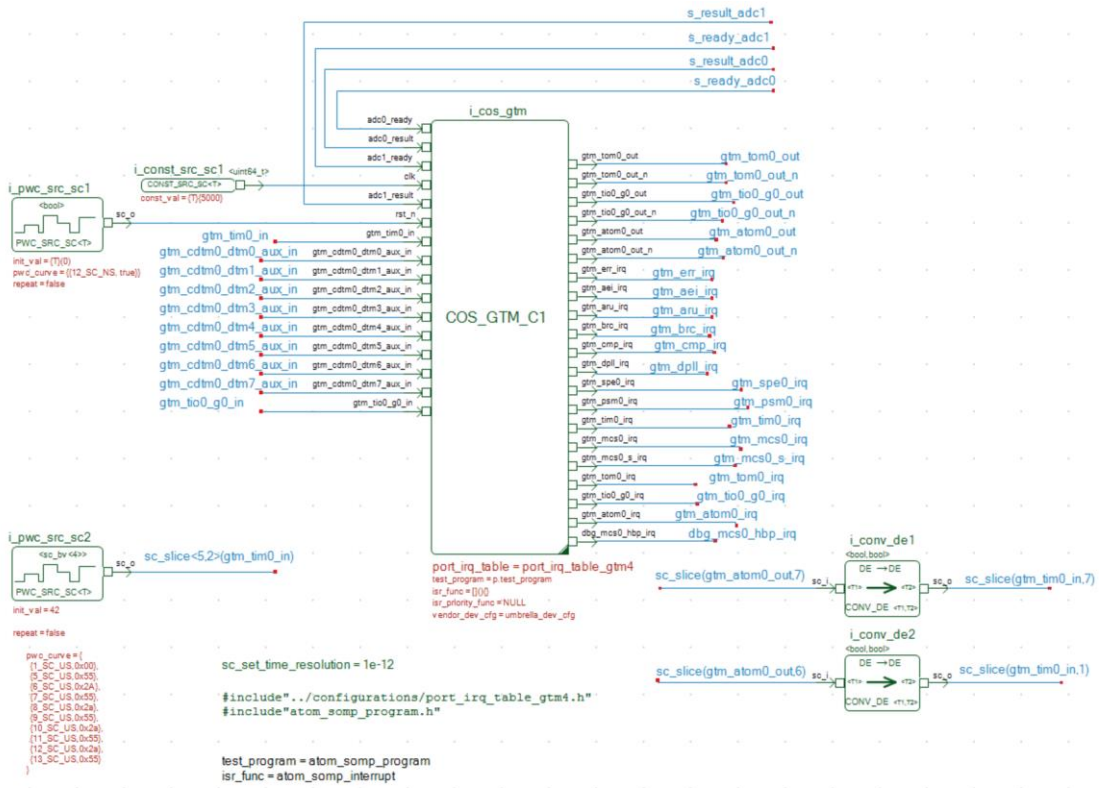




User Instance of Virtual BOSCH GTM Configuration

虚拟 BOSCH GTM 配置的用户实例

- Moderate number of ports – mostly of type bit vector
- Bit values can be extracted by the COSIDE slice utilities
- Controller program and interrupt service routine will be provided as C++ lambda function



Software Development with the Virtual BOSCH GTM Model

使用虚拟 BOSCH GTM 模型进行软件开发

Controller Program

- Programmed in C(++) using the HAL functions and definitions
- Has one entry function void function_name(), which is assigned to the lambda function parameter
- This function is started in a SystemC thread process
- The function can also spawn new SystemC thread processes to e.g. model the operating system

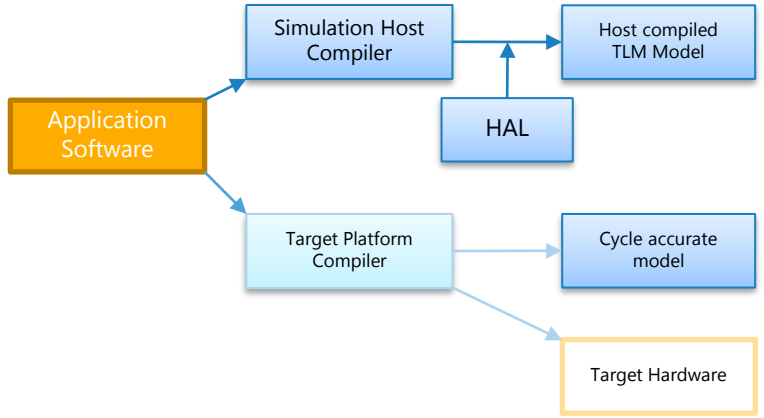
MCS program

- Programmed in Assembler
- Assembler and build scripts will be available
- The assembler creates a C/C++ File which defines the program memory content of the MCS
- This definition can be used in the controller program to load the MCS program into the GTM

Micro Controller Program

微控制器程序

- Original C-code – GTM access via BOSCH provided HAL
- Compiled with compiler of simulation host
- Macros for debug messages – will be empty in case the controller target compiler is used

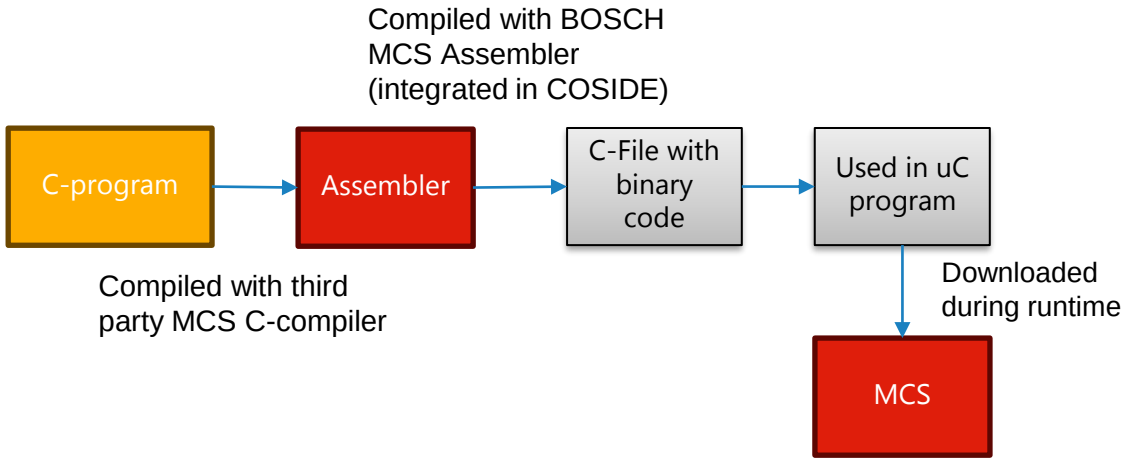



```
int TIO_BLDC_app()
{
  // setup TIO BLDC master PWM
  tio_bldc_master1.period = 0;
  tio_bldc_master1.update_source = 0;
  tio_bldc_master1.cls = 0;
  tio_bldc_master1.quad = 1;

   init_tio_bldc_master_period(&tio_bldc_master1);
   ...
  return 0;
}


void TIO_BLDC_app_isr() {


  GAL_INFO("Interrupt service routine");
  GTM.CLS[tio_bldc1.cls].TIO.G[tio_bldc1.tio_group].CH[0 +
       tio_bldc1.ch_start].IRQ_NOTIFY = 0x3; //S rise/fall edge
            :

  int hall_step= get_hall_step(&tio_bldc1);
   ...
  update_bldc_output(hall_step,&tio_bldc1.spwm);


  asynchron_update_pwm(&tio_bldc1.spwm);
}
```

MCS Program

MCS计划

- Compiled from C-code or assembler
- Binary is embedded if a C-file which will be loaded via the microcontroller

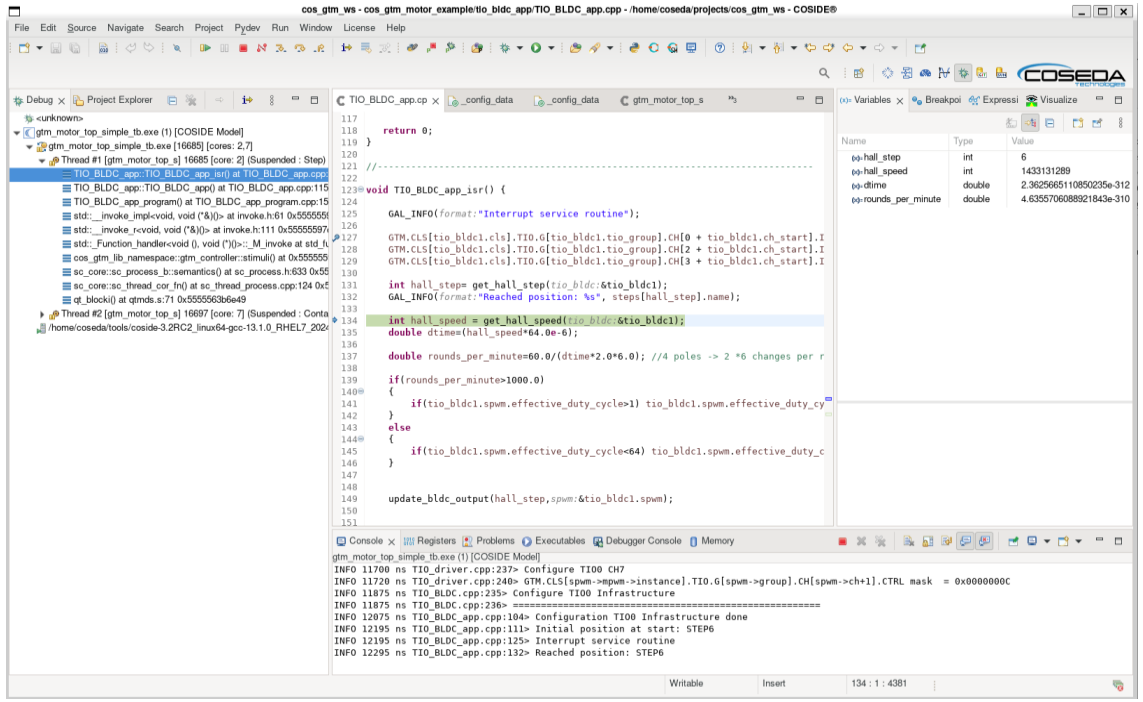



```
l_loop:
    ## read PWM parameter from memory
    mrdi R0, R7, mem_sr0 ; R0 = SR0
    mrdi R1, R7, mem_sr1 ; R1 = SR1


    ## write new period and duty cycle to ATOM Ch 3
    mrd R3, agc_upen_off
    bwr R3, ATOM_AGC_GLB_CTRL
    bwr R1, ATOM_CH3_SR1
    bwr R0, ATOM_CH3_SR0
    mrd R3, agc_upen_on
    bwr R3, ATOM_AGC_GLB_CTRL
    ## wait for time to change PWM-parameters
    mov R5, TBU_TS0
    add R5, R4
    wurmx R5, TBU_TS0
    ## increment loop-variable
    addl R7, 4
    atul R7, 0x10 ; loop four times
    jbs STA, CY, l_loop ; CY is set if R7 < 0x10
    jmp repeat_for_ever
```

Micro Controller Software Debugging

单片机软件调试

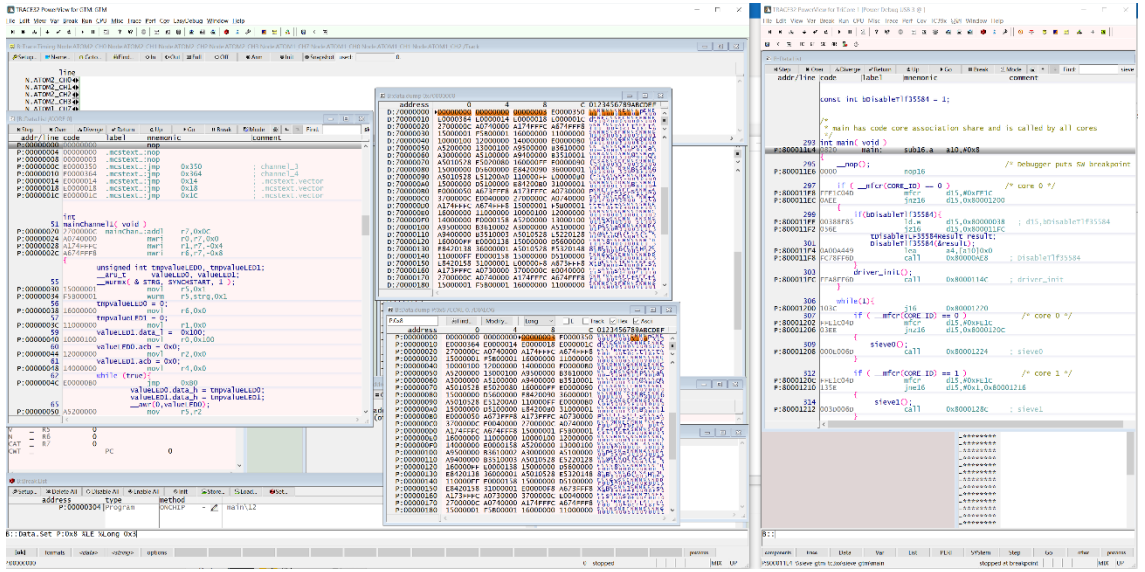
- Host compiled software can be easily debugged with standard debugger
- Arbitrary debug messages / logfiles can be printed / generated
- Time will stop if a breakpoint is hit -> whole system state holds at the current time



MCS Software Debugging

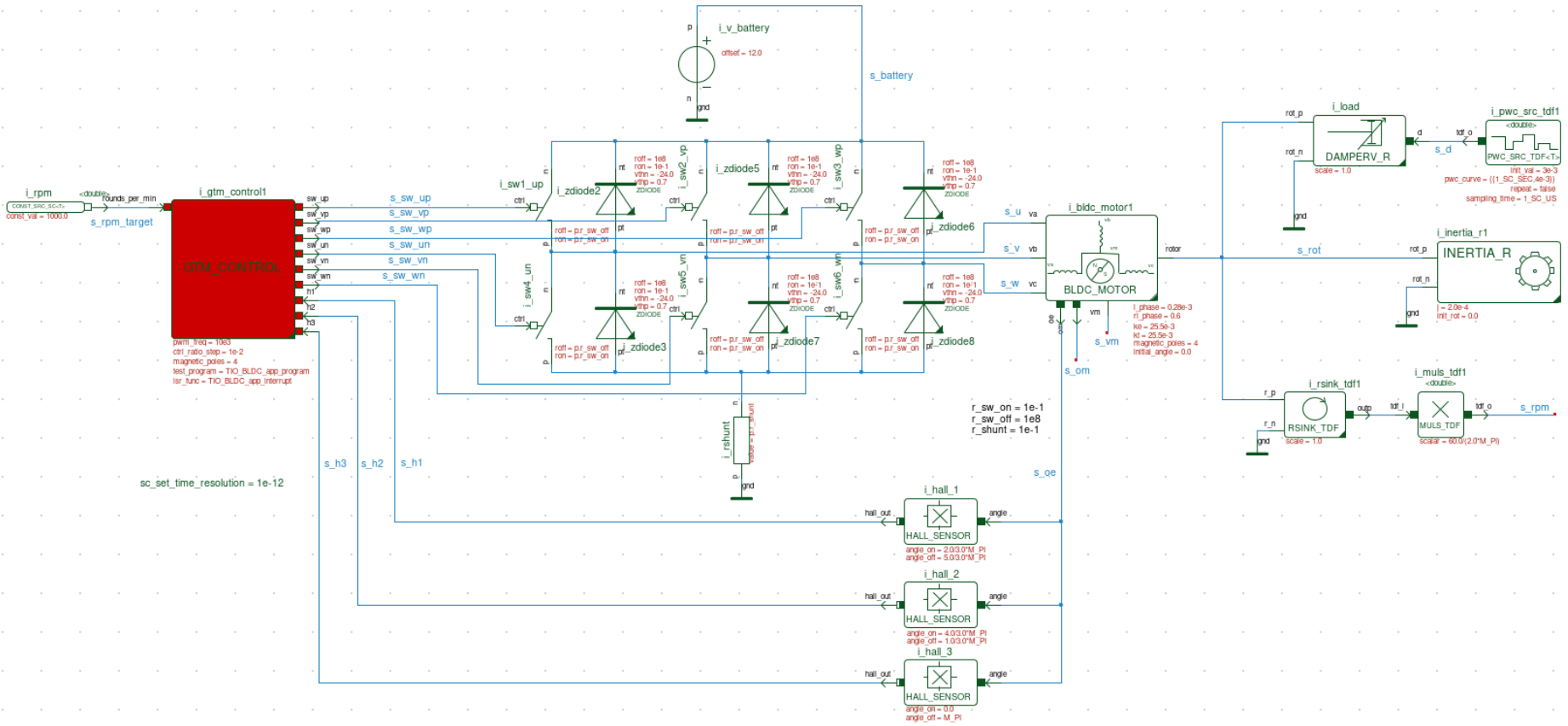
MCS软件调试

- Tracing and logging of executed opcodes and registers possible
- Third party debugger e.g. Lauterbach Trace 32 can be used (in development)
- Arbitrary number of MCS cores can be debugged in parallel
- Same GUI like for real hardware
- Breakpoint can stop the time for the whole model – state at this time is freezed and can be continued



Example – Brush Less DC (BLDC) Motor Control

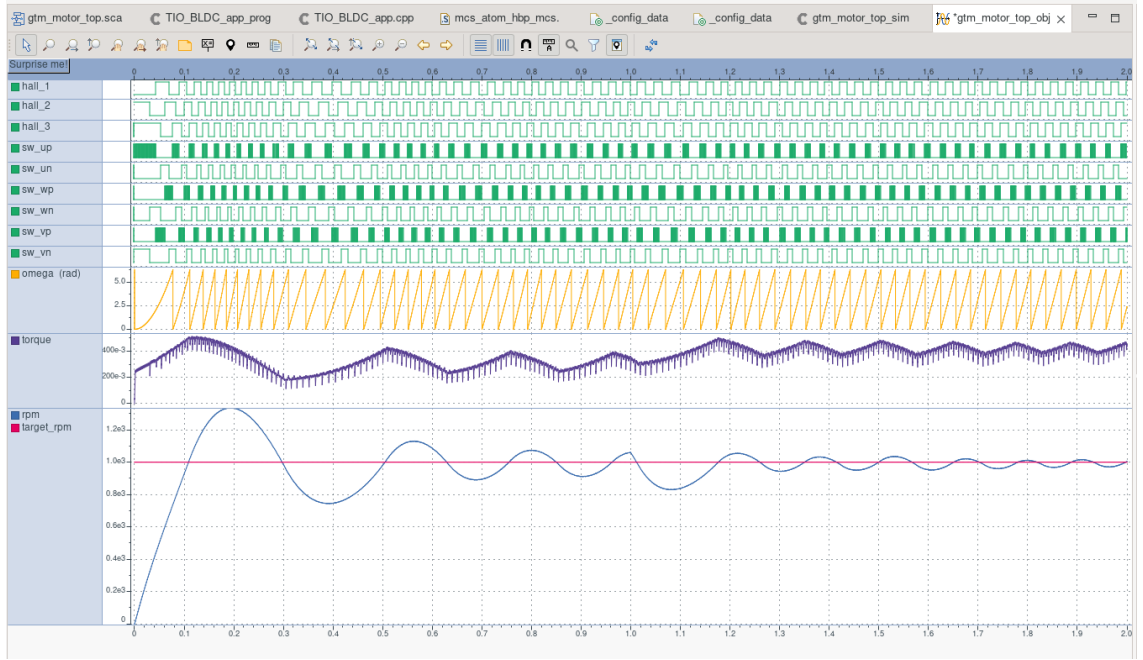
示例 – 无刷直流 (BLDC) 电机控制

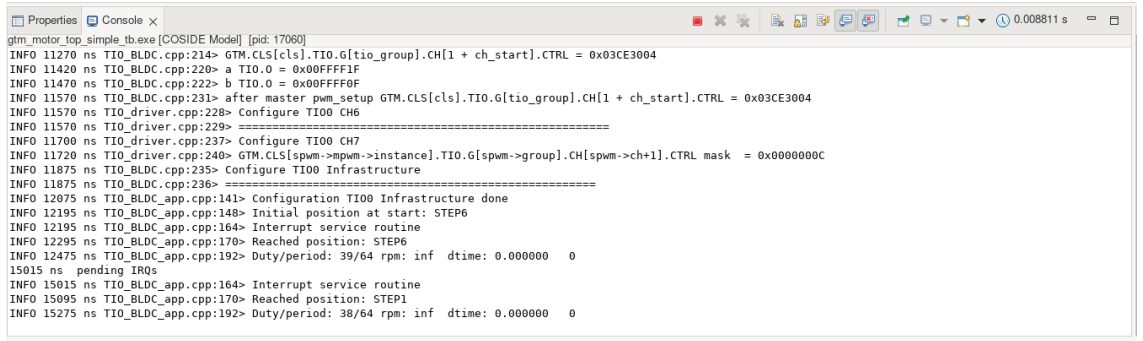


BLDC Motor Simulation Results

BLDC 电机仿真结果

- All signals and GTM interna traceable
- Arbitrary log messages can be generated
- Simulation reproducible
- Regression testing and continuous integration possible
- Simulation performance depends on the activity and pwm frequencies – in this example ca. 1h/sec

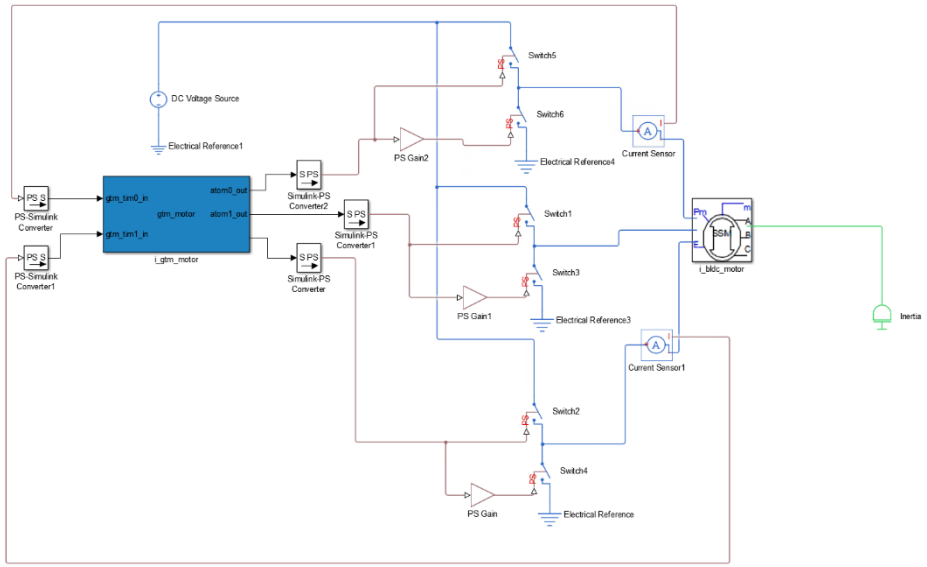




GTM Model Export (Simulink)

GTM 模型导出 (Simulink)

- Models can be exported to many other simulation tools
- Exported model for Matlab Simulink will be provided
- Program has to be compiled with the provided flow
- The generated dll/so will be loaded dynamically



Summary

概括

- The Virtual BOSCH GTM permits algorithm and software development without hardware availability
- A virtual prototype allows more powerful debugging and introspection compared to hardware debugging
- Virtual prototypes enable regression testing and continuous integration
- COSIDE simplifies the development and debugging of virtual prototypes

- Virtual BOSCH GTM 允许在没有硬件可用的情况下进行算法和软件开发
- 与硬件调试相比，虚拟原型允许更强大的调试和自省
- 虚拟原型支持回归测试和持续集成
- COSIDE 简化虚拟样机的开发和调试

Download for Virtual BOSCH GTM Model

下载虚拟 BOSCH GTM 模型

- The BOSCH GTM model can be downloaded for COSIDE 3.1.1
- https://www.coseda-tech.com/bosch-gtm-systemc-model
- For more information contact:
  - Jürgen Hanisch Robert Bosch GmbH - juergen.hanisch@de.bosch.com
  - Karsten Einwich COSEDA Technologies GmbH - karsten.einwich@coseda-tech.com

Your Contact

您的联系方式

Karsten Einwich

- CEO
- karsten.einwich@coseda-tech.com
- +49 351 321490-11

Thomas Hartung

- Marketing & Sales
- thomas.hartung@coseda-tech.com
- +49 351 321490-31

COSEDA Technologies GmbH
Koenigsbruecker Str. 124
01099 Dresden
Germany

www.coseda-tech.com