

Multi Chip Simulation of Battery Management System with Synopsys Virtualizer Studio

Frank Poppen, Ralph Görden, Manfred Thanner

Frank Poppen, System Modelling Engineer
NXP Semiconductors Germany GmbH

Outline



- Motivation
- High Voltage Battery Management System
- Full System Simulation Setup
- Issue with TLM-SPI
- Complete SW Stack Execution
- Conclusions and Future Work



Motivation

Motivation



- Automotive Microelectronic Systems

- high complexity
- high degree of interaction inside them
- high degree of interaction between them

https://en.wikipedia.org/wiki/List_of_software_bugs

- Strong Requirements

- functional safety
- reliability
- cost
- time to market

Transportation [\[edit \]](#)

- By some accounts ██████████ electronic throttle control system (ETCS) had bugs that could cause sudden unintended acceleration.^[63]
- The ██████████ experienced an integer overflow bug which could shut down all electrical generators if the aircraft was on for more than 248 days.^[64] A similar problem was found in ██████████ which need to be powered down before reaching 149 hours of continuous power-on time, otherwise certain avionics systems or functions would partially or completely fail.^[65]
- In early 2019, the transportation-rental firm ██████████ discovered a firmware bug with its electric scooters that can cause them to brake very hard unexpectedly, which may hurl and injure riders.^[66]
- ██████████ had all cockpit displays go blank if a specific type of instrument approach to any one of seven specific airports was selected in the flight management computer.^[67]
- ██████████ equipped with flight management systems by ██████████ would make wrong turns during missed approach procedures executed by the autopilot in some specific cases when temperature compensation was activated in cold weather.^[68]
- In June 1996, ██████████ failed less than a minute after launch, because the horizontal bias value was too big for a 16 bit register.

- Modeling and Simulation-Based Systems Engineering

Motivation

Models are the Basis for Engineering

- Contain more detail than text specifications
 - no need to reread (reinterpret) text specifications repeatedly
 - less ambiguous (execute and observe)
- Can be shared
 - virtual integration of component models build larger systems (reuse and integrate)
 - early customer involvement
- Evaluate safety functions
 - fault injection
- Enable hardware/software co-design
 - improve time-to-market (shift left)



Introduction

Shift Left with HW/SW Co-Design



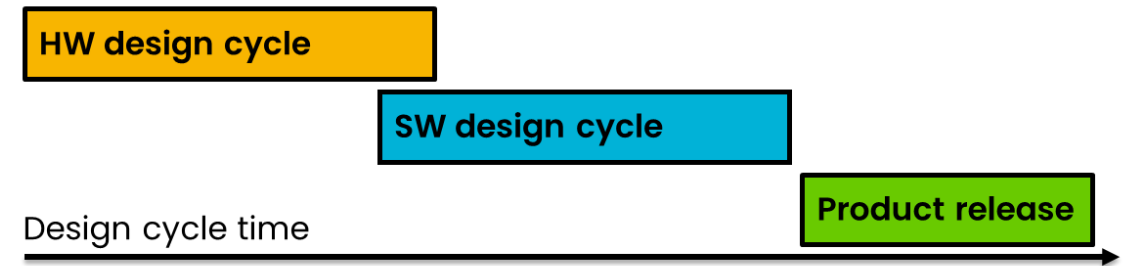
- **Developing HW and SW in parallel**

- traditional: first develop ECU then SW
- faster: start early SW development with virtual prototype
- supports collaboration and quality of results reducing time to production

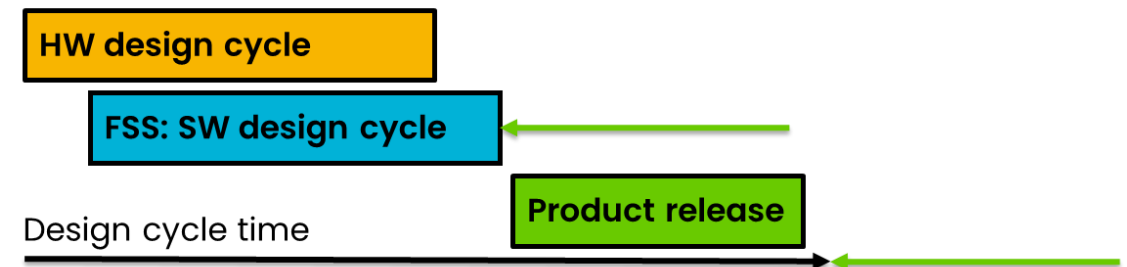
- **Early SW testing**

- HW and lab space limit test cycles
- fast and more test cycles enable increased coverage
- fault injection to test safety features

1.) HW based development of embedded SW



2.) Shift left: HW/SW co-design



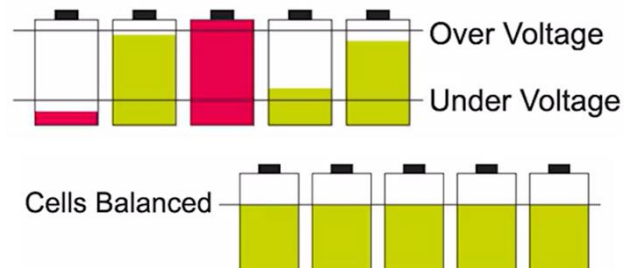


High Voltage Battery Management System

High Voltage Battery Management System



- No one size fits all BMS solution
 - scooters, motorcycles, buses, trucks, trains, boats or airplanes
 - battery packs as low as 14V or high voltages of 800V and more
- BCC directly connected to cells
 - no means of “turning off and on again”
 - low power consumption to avoid discharge of the cells
- Fire in an electric vehicle is an (ASIL) D hazard
- Supervises voltage and temperature of cells
- Cell balancing
 - lowest voltage cell limits discharge
 - highest voltage cell limits charging

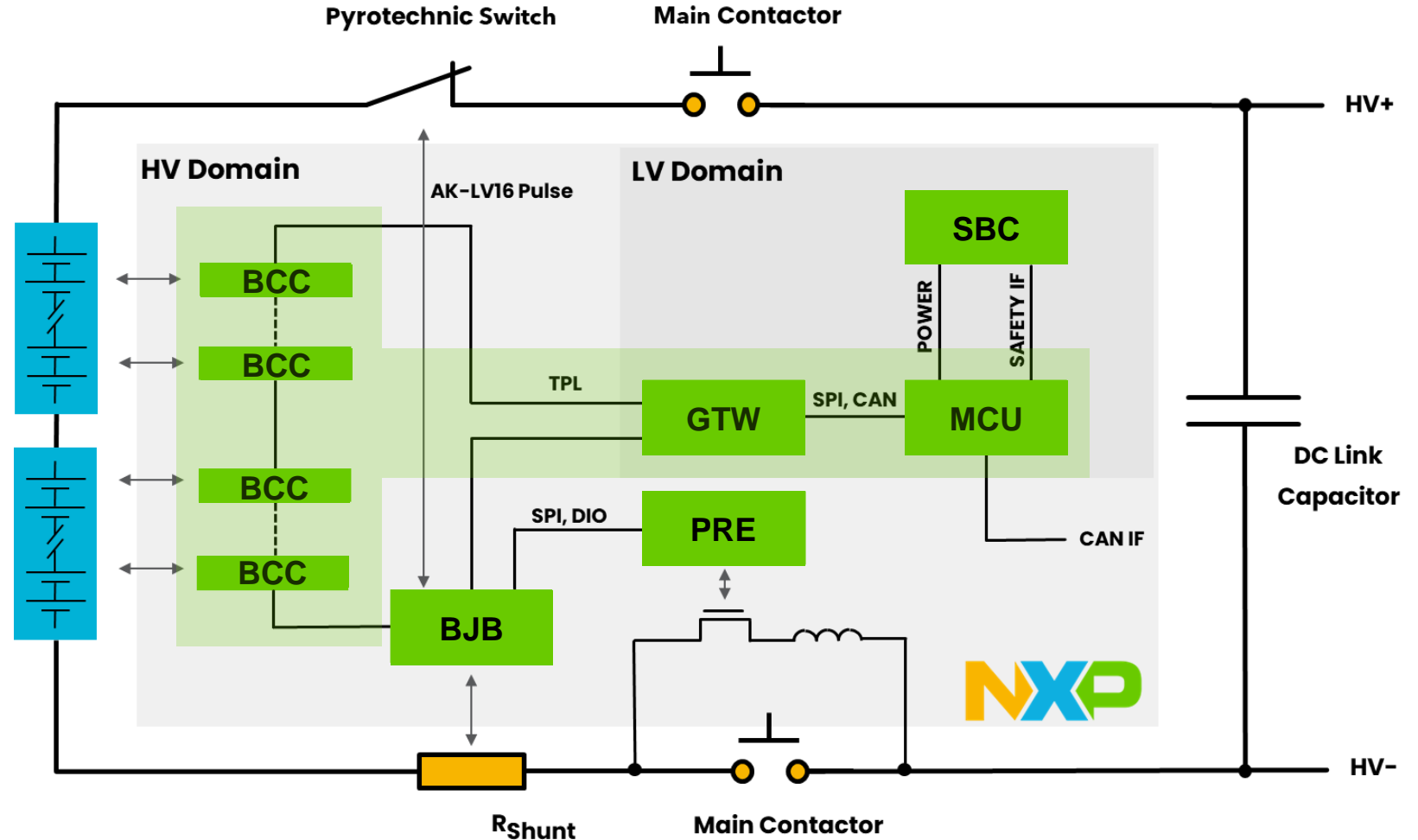


High Voltage Battery Management System



System Solution of a HV BMS

- **MCU**
Microcontroller Unit
- **GTW**
Gateway
- **BCC**
Battery Cell Controller
- **TPL**
Transport Protocol Link
- **BJB**
Battery Junction Box
- **PRE**
Precharge
- **SBC**
Safety System Basis Chip



Full System Simulation Setup

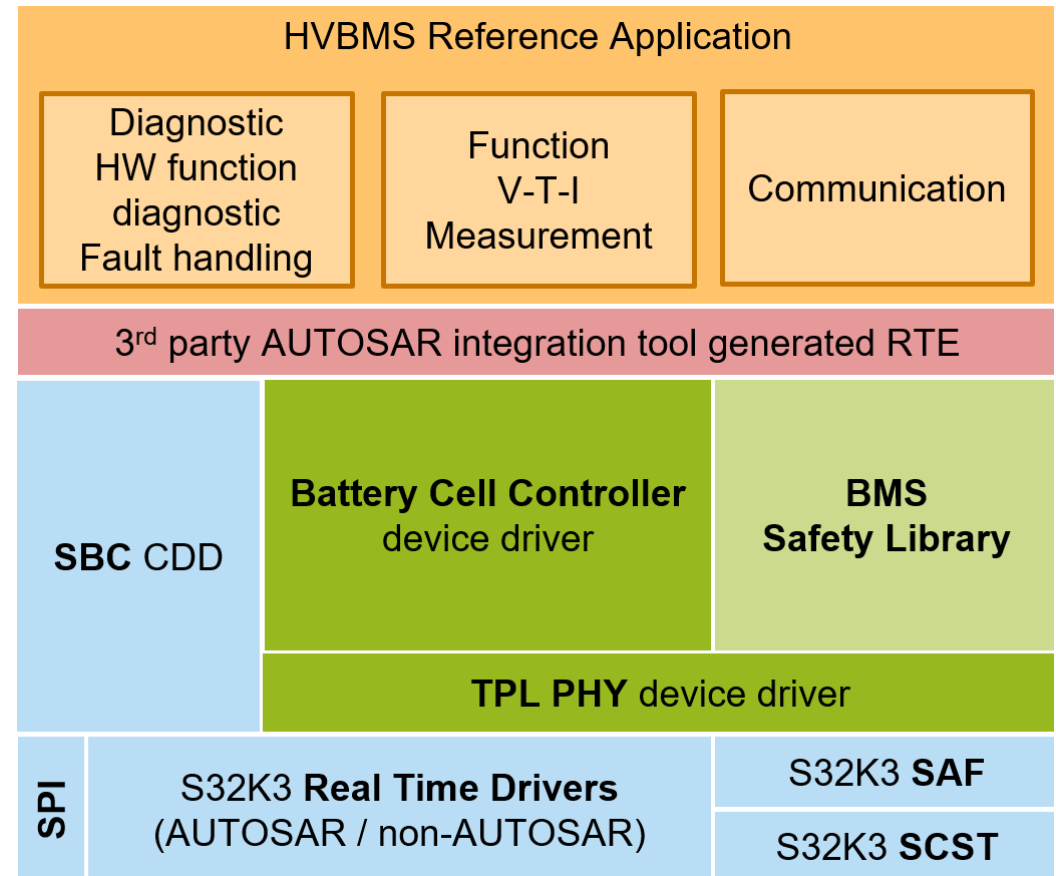
Full System Simulation Setup

Software Stack



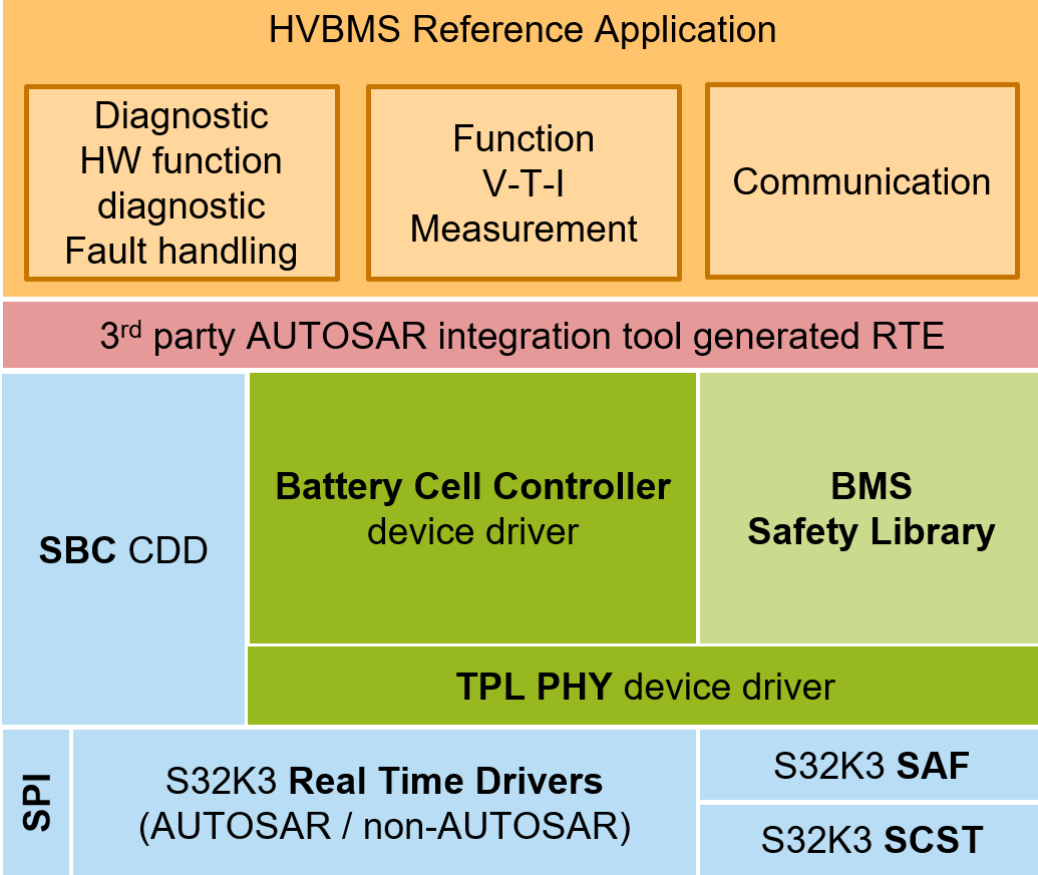
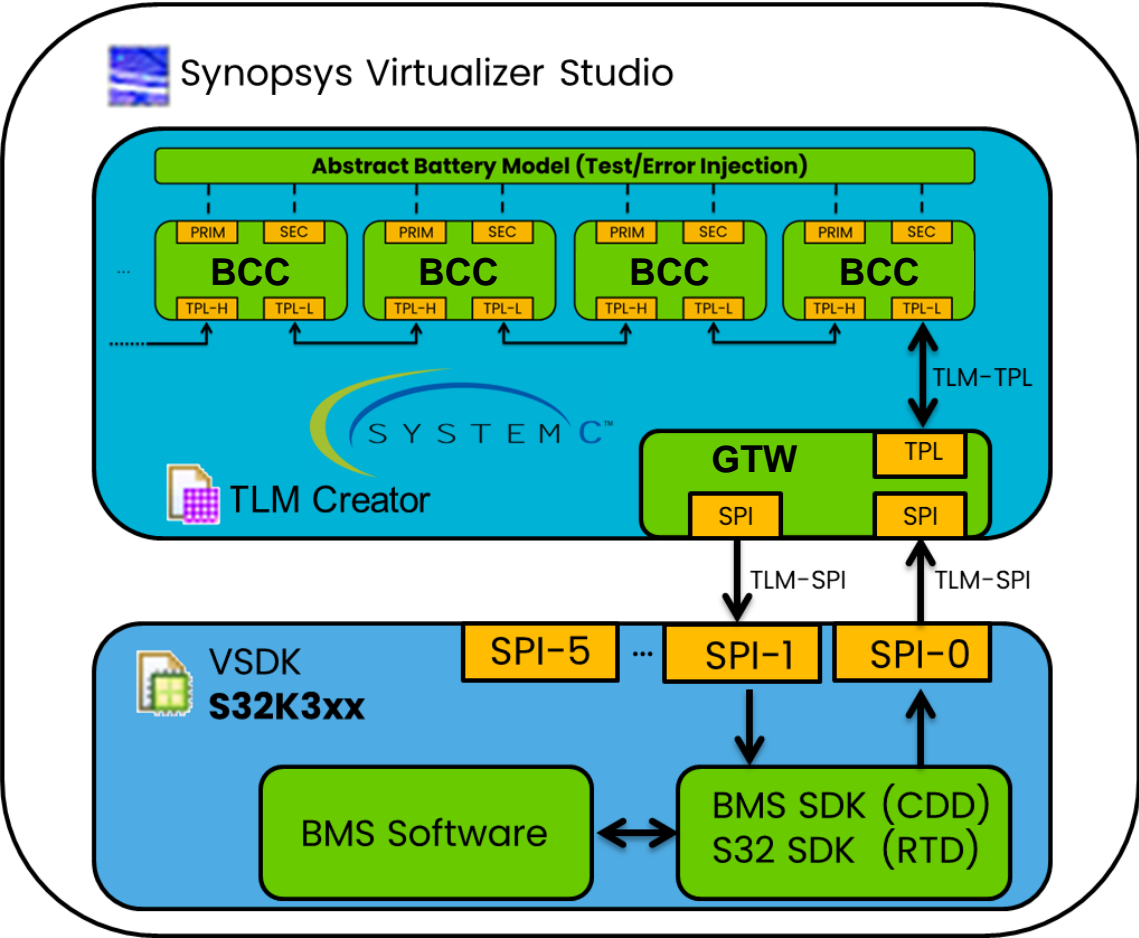
- Real time drivers
 - real time communication
- Complex device drivers
 - system basis chips
 - battery cell controller
 - ...
- Safety library
- Application

Software Stack Based on AUTOSAR Standard



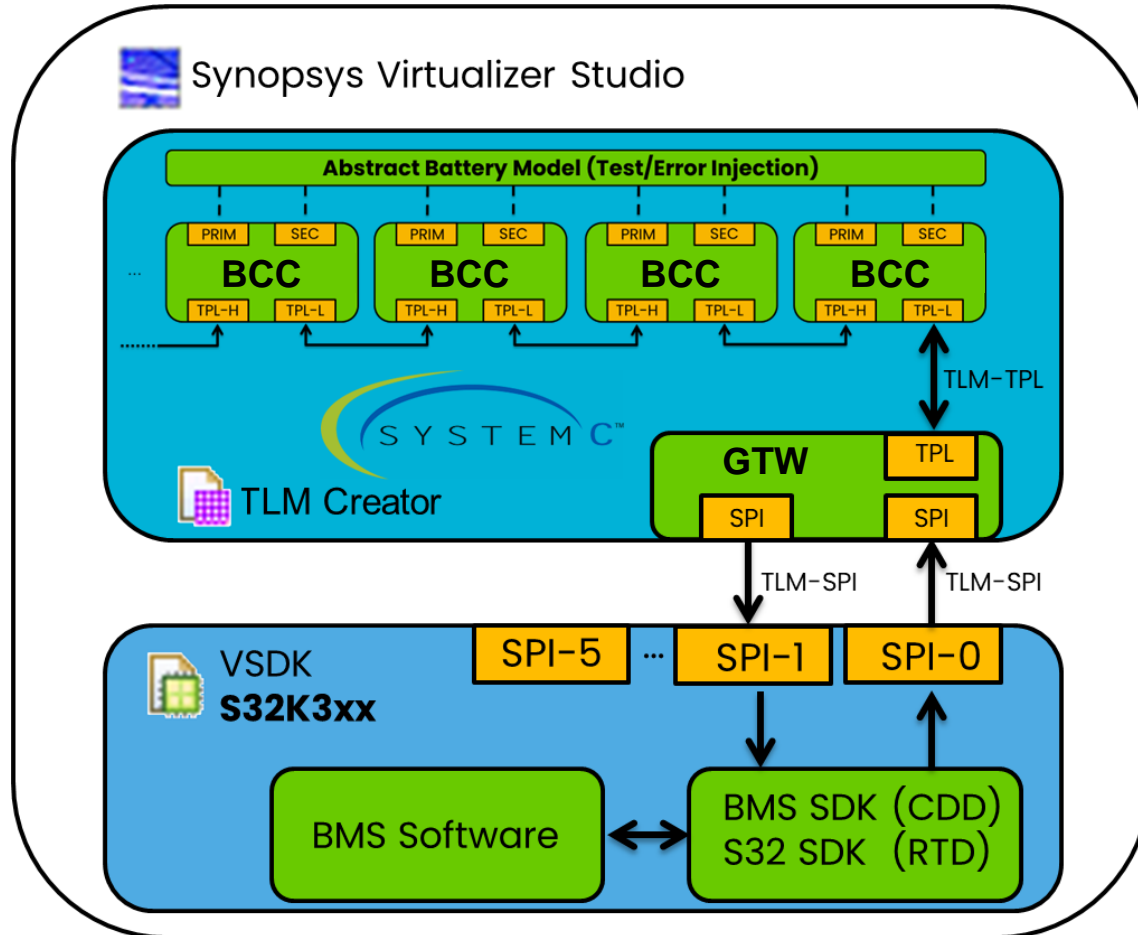
Full System Simulation Setup

Hardware



Full System Simulation Setup

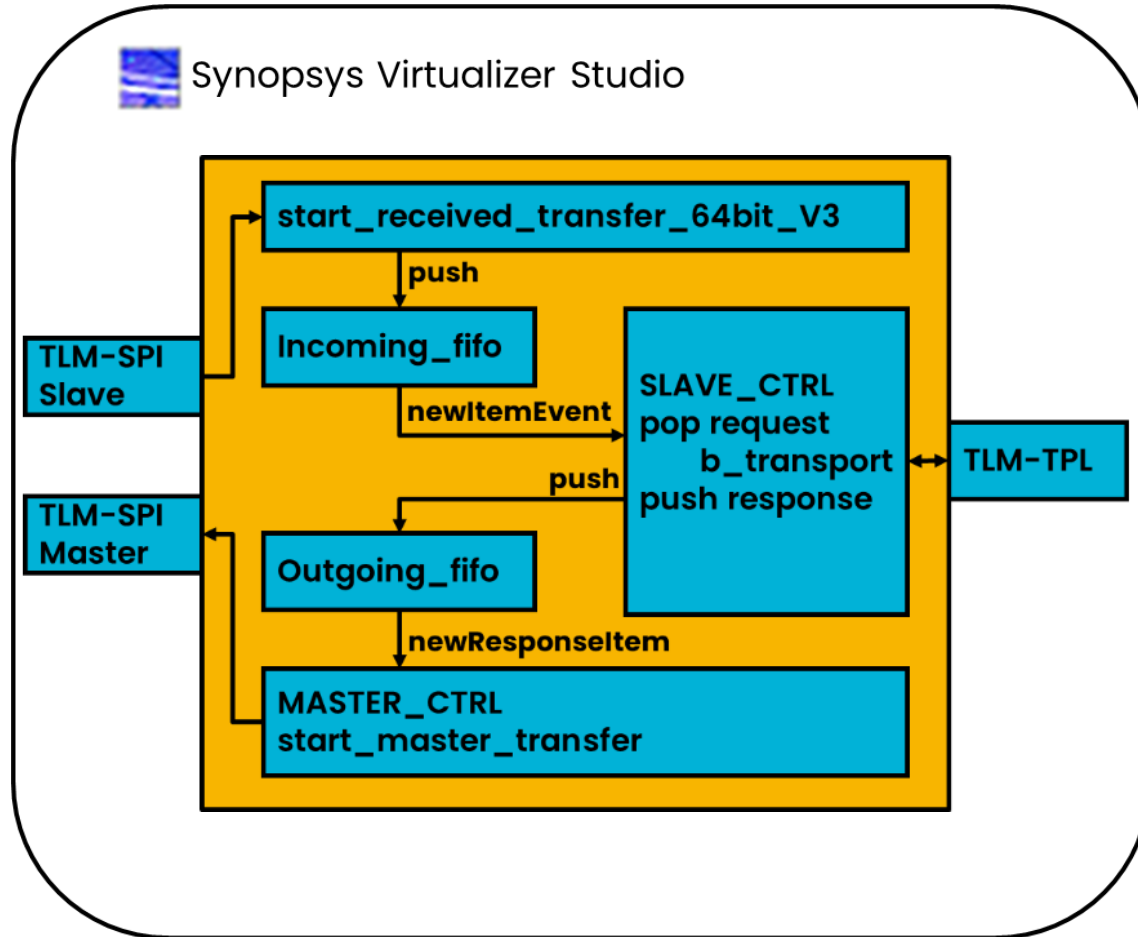
Hardware



- Synopsys VSDK of NXP's MCU
 - complete hardware perspective with registers and peripherals (GPT, UART, CAN, SPI)
- BCC functional SystemC model
 - SystemC architecture generated from SysML
 - functional behavior added manually
 - importing SystemC into TLM creator using python script (300 path- and filenames)
- Gateway
 - helpful Synopsys tutorial on SPI controller
 - translates TLM-SPI to TLM-TPL
- Drag and drop into VSDK
 - components in Virtualizer Studio Library

Full System Simulation Setup

Hardware

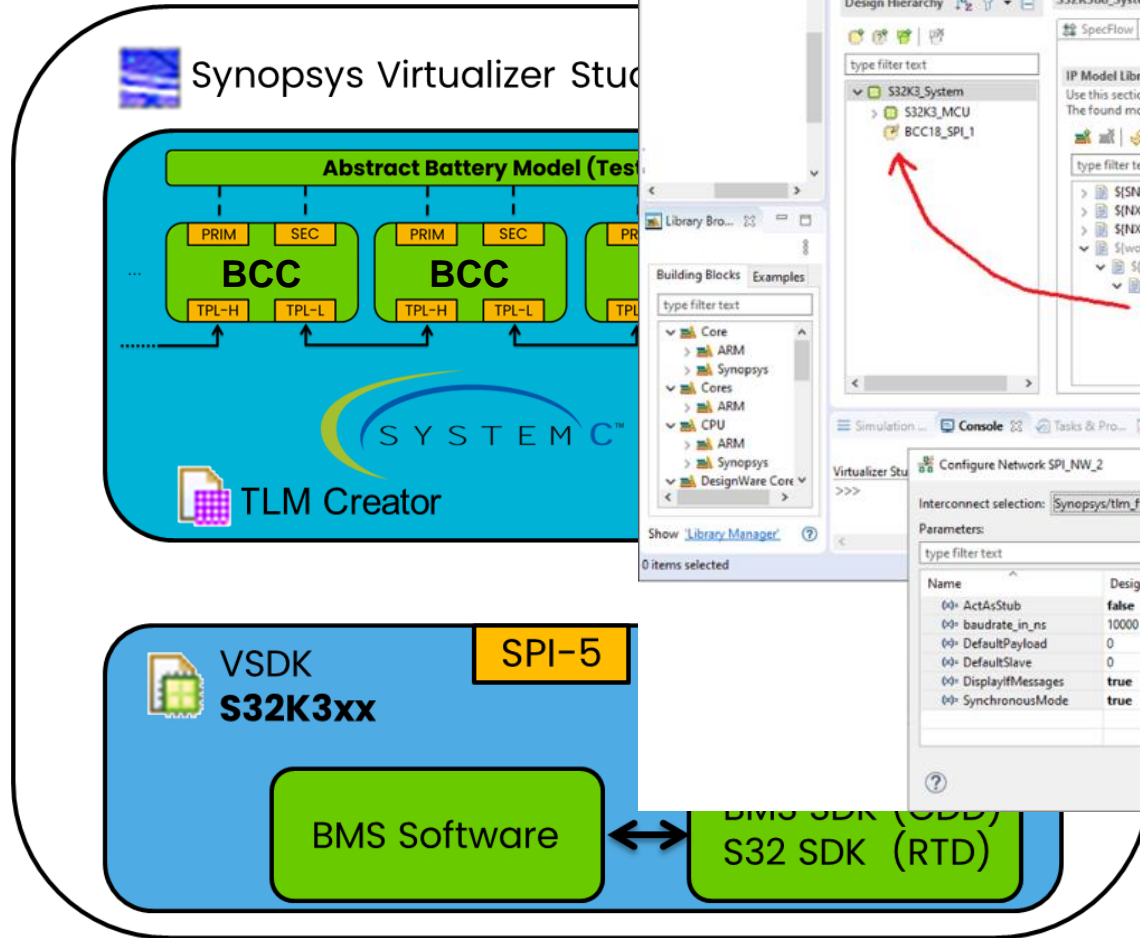


- Synopsys VSDK of NXP's MCU
 - complete hardware perspective with registers and peripherals (GPT, UART, CAN, SPI)
- BCC functional SystemC model
 - SystemC architecture generated from SysML
 - functional behavior added manually
 - importing SystemC into TLM creator using python script (300 path- and filenames)
- Gateway
 - helpful Synopsys tutorial on SPI controller
 - translates TLM-SPI to TLM-TPL
- Drag and drop into VSDK
 - components in Virtualizer Studio Library

Full System Simulation Setup



Hardware



Name	Design Value	Type
ActAsStub	false	Boolean
baudrate_in_ns	10000	Integer
DefaultPayload	0	Integer
DefaultSlave	0	Integer [0..2]
DisplayIfMessages	true	Boolean
SynchronousMode	true	Boolean

- Drag and drop into VSDK
 - components in Virtualizer Studio Library

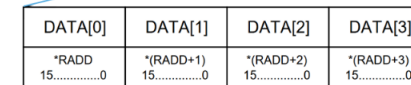
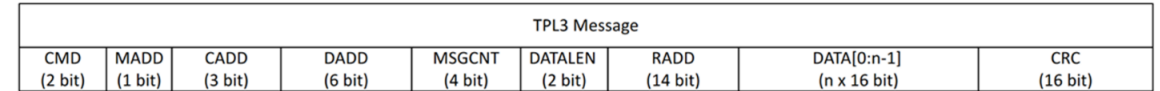
Issue with TLM-SPI

Issue with TLM-SPI

Transport Protocol Link



- Copy frame bits from SPI to TPL
- TPL can have 64, 80, 96 or 112 bits
 - Length encoded inside the frame itself
- TLM-SPI can carry max of 64 bit
 - decided to send 16 bit TLM-SPI messages
 - Four to Seven TLM-SPI for one TLM-TPL
- How to know start of new TPL-frame?



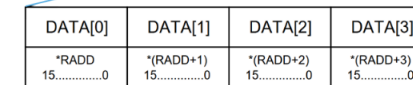
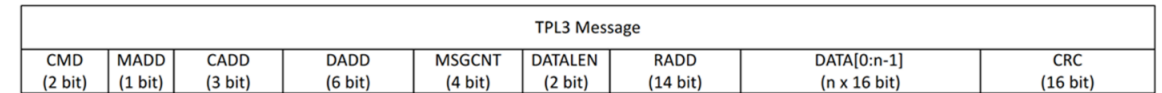
```
received_transfer_64bit_v3(  
    unsigned long long data,  
    unsigned bit_length,  
    bool continuous_select  
)
```

Issue with TLM-SPI



How to know start of new TPL-frame?

- Decode the frame and look for DATALEN
 - bitwise interpretation breaks level of abstraction
 - adds function to GTW that is not there
- Collect until continuous select flag becomes '0'
- We would see cases with faulty flag
 - SW RTD? HW GTW? TLM-SPI!
 - Fix for VSDK by Synopsys within three days



```
received_transfer_64bit_V3(  
    unsigned long long data,  
    unsigned bit_length,  
    bool continuous_select  
)
```

Figure 2-3 Continuous Select Flag Sending 64 Bits of Data



To send 64 bits of data in a single job of SPI, when only 16 bits of data can be sent by the SPI device at a time (that is, in a single payload), the continuous select flag is set as follows:

- 1st Tx payload: Continuous Select = 1 (start of Tx)
- 2nd Tx payload: Continuous Select = 1
- 3rd Tx payload: Continuous Select = 1
- 4th Tx payload: Continuous Select = 0 (end of Tx)

Complete SW Stack Execution

Complete SW Stack Execution

Configure and Compile



Parameter	File	Offset	Load Type	Ad.
(*)= image_path				
CPU_SS				
ImageInfo				
(*)= initial_image				
main	/out/main.elf	...	0x0	Image+Symbols
Security_Device_GPRs				
HSE_B				
HSE_CM7_0				

- Simulation and real device operate on same executable
- Compiler setup is used for both cases
 - Elektrobit (EB) Tresos as microcontroller abstraction layer (MCAL) AUTOSAR workflow
 - design and compilation environment S32 Design Studio (S32DS)

Complete SW Stack Execution



Execute and Debug

```

for devNum=1 to 5 do (
  nopTPL(DADD=0);
  writeTPL(DADD=0, RADD=0x0001, val=devNum);
  devNumCheck=readTPL(DADD=devNum, RADD=0x0001);
  if devNumCheck == devNum -> OK
  else NOK
)
    
```

Design Browser | Memory Map | Interrupt Table

modules filter, * for wild cards on full path

/S32K3_System/BCC18_SPI_1/bcc18_dc/bcc_1/impl/system_ctrl

Item	Start ...	End...	Value
m_System_Control_Registers	-	-	-
m_SYS_CFG_CRC	-	-	0x0000
m_SYS_COM_CFG	-	-	0x0202
m_SYS_COM_CFG_DADD	-	-	0x2
m_SYS_COM_CFG_CADD	-	-	0x0
m_SYS_COM_CFG_BUSFW	-	-	0x1
m_SYS_COM_CFG_NUMNODES	-	-	0x0

Timing diagram showing memory accesses for system_ctrl.m_System_Control_Registers. The diagram shows four transactions where the address is 0x200 and the data is 0x201, 0x202, 0x203, and 0x204 respectively. The DADD register value is also shown as 0x1, 0x2, 0x3, and 0x4.

VDK Debug | Window | Help

- Launch Simulation using VP Config (Ctrl+6)
- Launch Simulation using Executable
- Detach from Simulation
- Restart Simulation
- Resume Simulation (F8)
- Configure Analysis... (Ctrl+Shift+A)
- Display Charts... (Ctrl+Shift+D)
- Show Parameters... (Ctrl+Shift+P)
- Launch Source Code Debugger
- Stop Simulation (Ctrl+F2)
- Kill Simulation
- Clear Debug Session State
- Open Simulation Results...
- Manage Search Paths for Results...
- Create Coverage Report
- Step into Source Code
- Create Checkpoint
- Restore Checkpoint
- Launch Debugger
- File Systems

Images

Filter text, filters can be cascaded with '&&'

S32K3_System

- S32K3_MCU
 - Peripherals
 - VP9D
 - config
 - root_path
 - VIRTIO
 - config
 - image_out
 - image_path
 - CPU_SS
 - ImageInfo
 - initial_image
 - Security_Device_GPRs
 - HSE_B

Launch Debugger

- Arm Development Studio
- MULTI
- TRACE32
 - Debug S32K3_System.S32K3_MCU.CPU_SS.CLUSTER_0
 - Debug S32K3_System.S32K3_MCU.CPU_SS.CLUSTER_1
 - Debug S32K3_System.S32K3_MCU.CPU_SS.CLUSTER_2
 - Debug S32K3_System.S32K3_MCU.CPU_SS.CLUSTER_3
 - Debug S32K3_System.S32K3_MCU.Security_Device_GPRs.HSE_B.HSE_CM7_0
- UDE
- winIDEA
- Standalone Start-Up

Memory Map

modules filter, * for wild cards on full path

/S32K3_System/BCC18_SPI_1/bcc18_dc/bcc_1/impl/system_ctrl

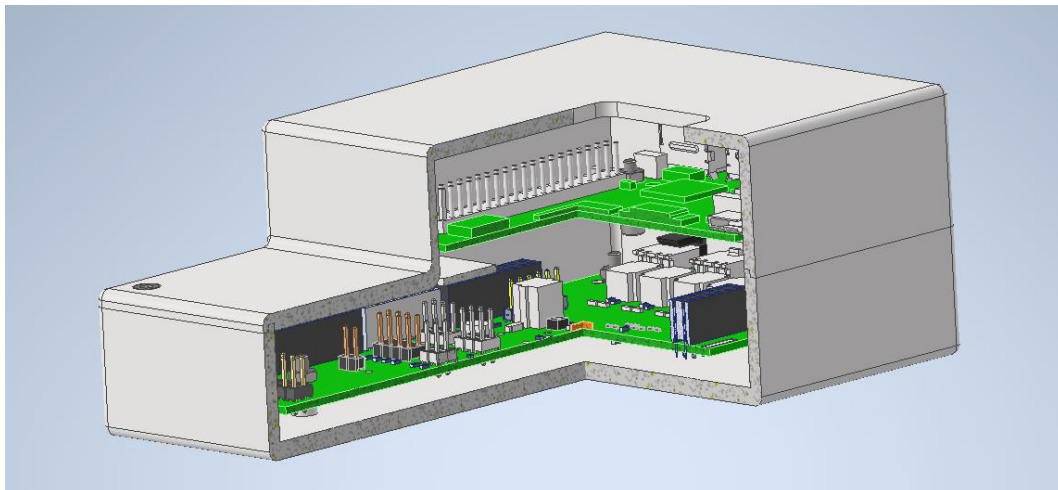
Item	Start ...	End...	Value
m_SYS_SYNC_CTRL	-	-	0x0000
m_SYS_R2R_MEAS	-	-	0x0000
m_SYS_SYNC_DEBUG0	-	-	0x0000
m_SYS_SYNC_DEBUG1	-	-	0x0000
reg_bus_slv_pi	-	-	<N/A>
clk_pi	-	-	<N/A>
init_ack_pi	-	-	0x00000000
mcu_if_select_pi	-	-	0
por_pi	-	-	0x00000000
prim_meas_rdy_pi	-	-	0x00000000
sleep_ack_pi	-	-	0x00000000
supply_status_pi	-	-	0
wake_up_pi	-	-	0x00000000
clk_ctrl_po	-	-	0
com_cfg_po	-	-	<N/A>
init_req_po	-	-	0x00000000
logic_fault_po	-	-	0x00000000
meas_invalidate_po	-	-	0x00000000
operation_mode_po	-	-	<N/A>
rst_ctrl_po	-	-	0
clean_req_po	-	-	0x00000000

Conclusions and Future Work

Conclusion and Future Work



- Experience and results are very convincing
- Investment of resources is justified
- Involved RTD/CDD developers:
 - “when to make use of this in our projects?”
 - “would help to clean out laboratory space!”



- Functionality behind registers must be coded manually and is not complete
- Product variants to be modelled
 - reuse potential expected between models
- Accurate battery cell models
 - simple model does not respond to load
 - possibly connect to Matlab/Simulink or similar
- Ideally, Synopsys would open up SCML as open-source standard
- Emulation with MIL (Model In the Loop)
 - first experiments with Raspberry Pis 4 and 5



THANK YOU

Our
Technology,
Your
Innovation™