COBHAM

TSIM3 – Emulating the CAN interface of the GR716 microcontroller

Cobham Gaisler

Julius Barendt

14 June 2019

CAN in space 2019

Overview

- Introduction
- Why use a simulator?
- Emulating the CAN interface of GR716 in TSIM3
    - The process
    - The problems
    - The solutions

Introduction

Cobham Gaisler and CAN

- CAN expertise
  - All of our processor devices has CAN
  - New devices will have the new CAN cores with DMA (GRCAN/GRCAN-FD)

Introduction

Software support

- RTEMS, VxWorks, Linux and Bare metal
- All CAN interfaces are supported, but support varies by operating system
- Tool-chains and examples

Introduction

Simulator and debugger

- TSIM
  - Simulator for LEON and ERC32
  - Instruction level simulator
  - Highly accurate and extensible
  - Standard simulator for LEON
- GRMON

Why use a simulator?

Why use a simulator?

Common applications

- No hardware
  - Not yet available
  - Too expensive
  - No space
  - Many developers few devices
- Simple to get going
  - No setup times

Why use a simulator?

Validation and verification

- Non intrusive code coverage and profiling
  - No need for custom binaries, test unmodified flight software
- Extended debug capabilities
- Automated testing
  - Scripting
- Deterministic execution

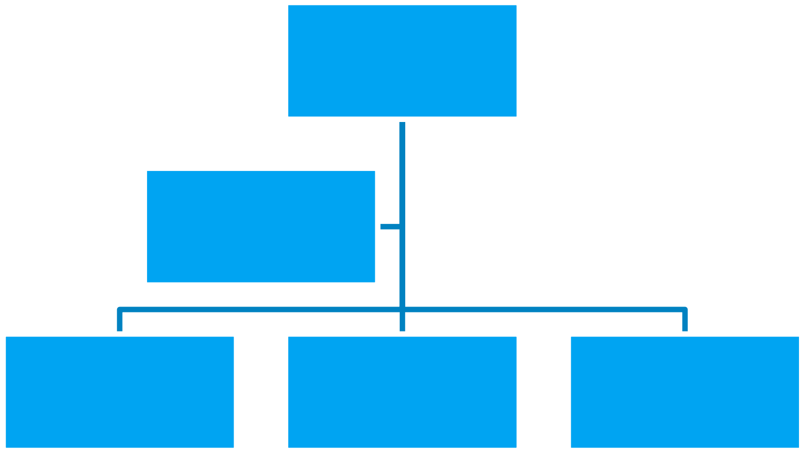
TSIM

- The goto LEON simulator
  - Highly accurate
- Deterministic behavior
- Highly optimized and easy to use
- Can be used in larger simulation systems

TSIM

Extensible

- User extensible
  - IO bus and AMBA bus device models
  - Models for connected peripherals
  - FPU and coprocessor models
  - Custom instructions
  - Interrupt controllers



TSIM

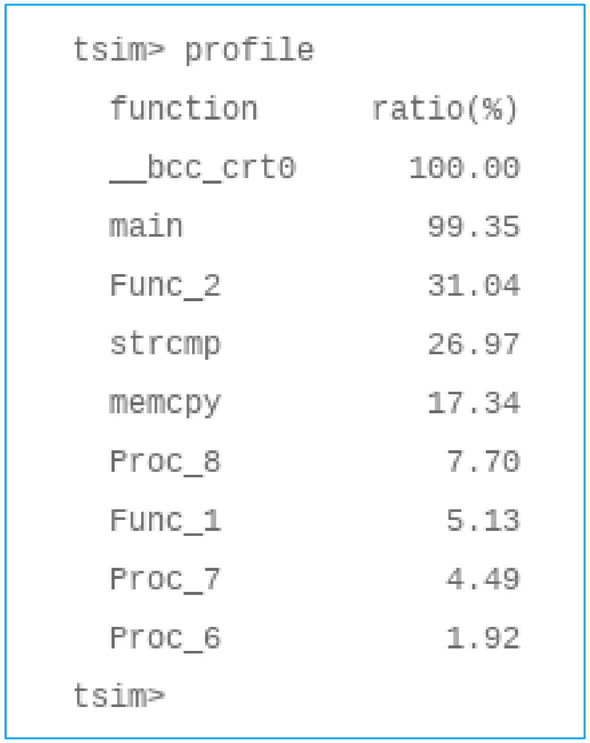
Modes of operation

- Stand alone
  - Controlled by command line and/or scripts
- Library interface
  - Controlled via C/C++ language API specific for TSIM
  - Parts of TSIM as object files that can be linked into overall system/satellite simulators.
- Remote GDB interface
  - Controlled via GDB command line or via GUI such as Eclipse IDE
  - Thread interface support
  - GDB built for SPARC/LEON distributed with all our tool chains

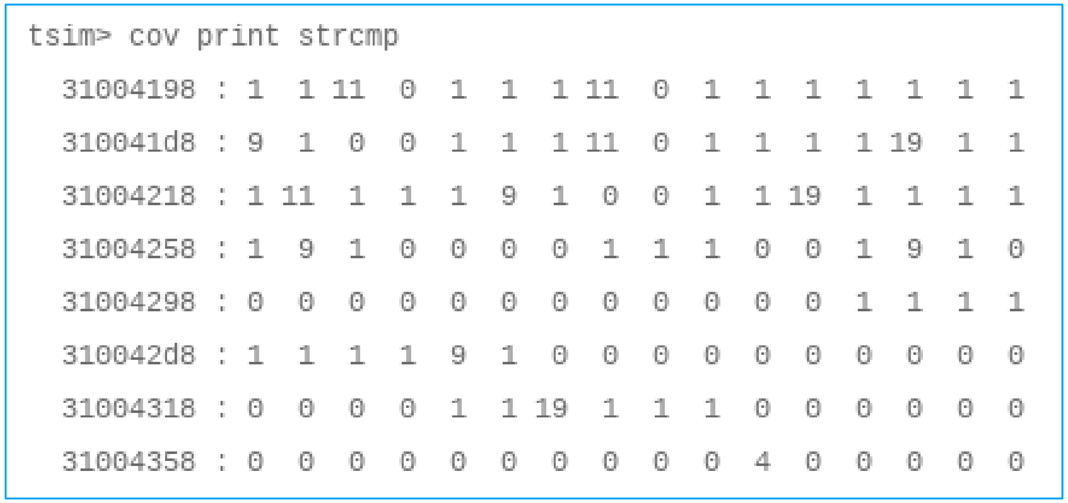
TSIM

Non intrusive execution statistics

- Instruction, cache and bus traces
- Execution profiling
- Code, decision and data coverage



```
tsim> profile
  function          ratio(%)
  __bcc_crt0         100.00
  main                99.35
  Func_2              31.04
  strcmp              26.97
  memcpy              17.34
  Proc_8               7.70
  Func_1               5.13
  Proc_7               4.49
  Proc_6               1.92
tsim>
```



```
tsim> cov print strcmp
  31004198 : 1  1 11  0  1  1  1 11  0  1  1  1  1  1  1  1
  310041d8 : 9  1  0  0  1  1  1 11  0  1  1  1  1 19  1  1
  31004218 : 1 11  1  1  1  9  1  0  0  1  1 19  1  1  1  1
  31004258 : 1  9  1  0  0  0  0  1  1  1  0  0  1  9  1  0
  31004298 : 0  0  0  0  0  0  0  0  0  0  0  0  1  1  1  1
  310042d8 : 1  1  1  1  9  1  0  0  0  0  0  0  0  0  0  0
  31004318 : 0  0  0  0  1  1 19  1  1  1  0  0  0  0  0  0
  31004358 : 0  0  0  0  0  0  0  0  0  0  4  0  0  0  0  0
```

TSIM

Debugging capabilities

- Unlimited breakpoints
- Unlimited watchpoints
- Instruction traces
- Stack back traces with symbolic information
- Check-pointing capabilities, save and restore simulator state
- I/O core event tracing
- RTEMS thread support
- Source level debugging using GDB remote connection

Emulating CAN in TSIM3

CAN in TSIM3

Development process – A new bus model

- Simulation models needed for GRCAN and GRCAN-FD
- Enabling users to develop their own CAN nodes
  - Needs to be easy to use
- Ease the development of CAN applications
  - Diagnostics
  - Error injections

CAN in TSIM3

C API

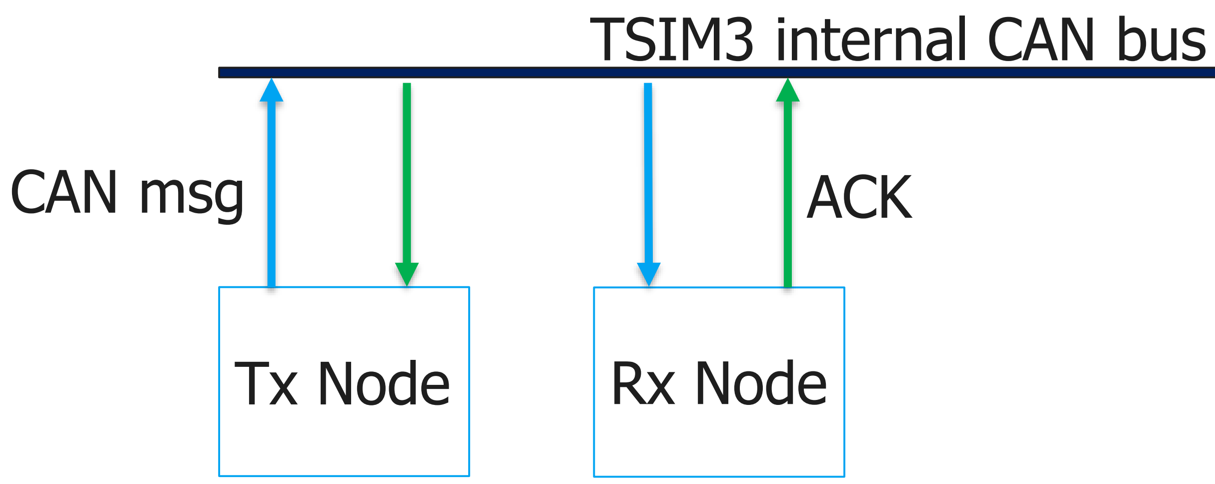
- Register CAN nodes the TSIM3 internal CAN bus model
- Send messages on the bus, represented by a C struct
- Frame lengths of up to 64 bytes of data, enabeling CAN-FD
- Each messages decides the bit-rate to be sent at
    - Data at package level
- Flags allowing for custom behaviour, such as error injection

```
struct can_msg {
    unsigned int *data;

    unsigned int flags;
    unsigned int nominal_bitrate;
    unsigned int fd_bitrate;
};
```

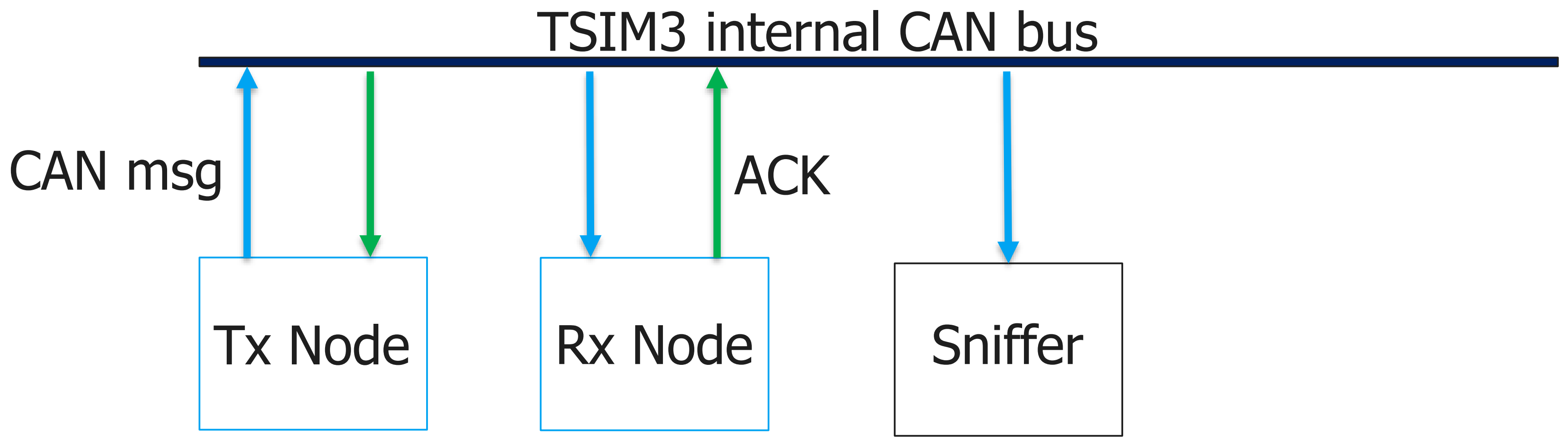
CAN in TSIM3

Normal node



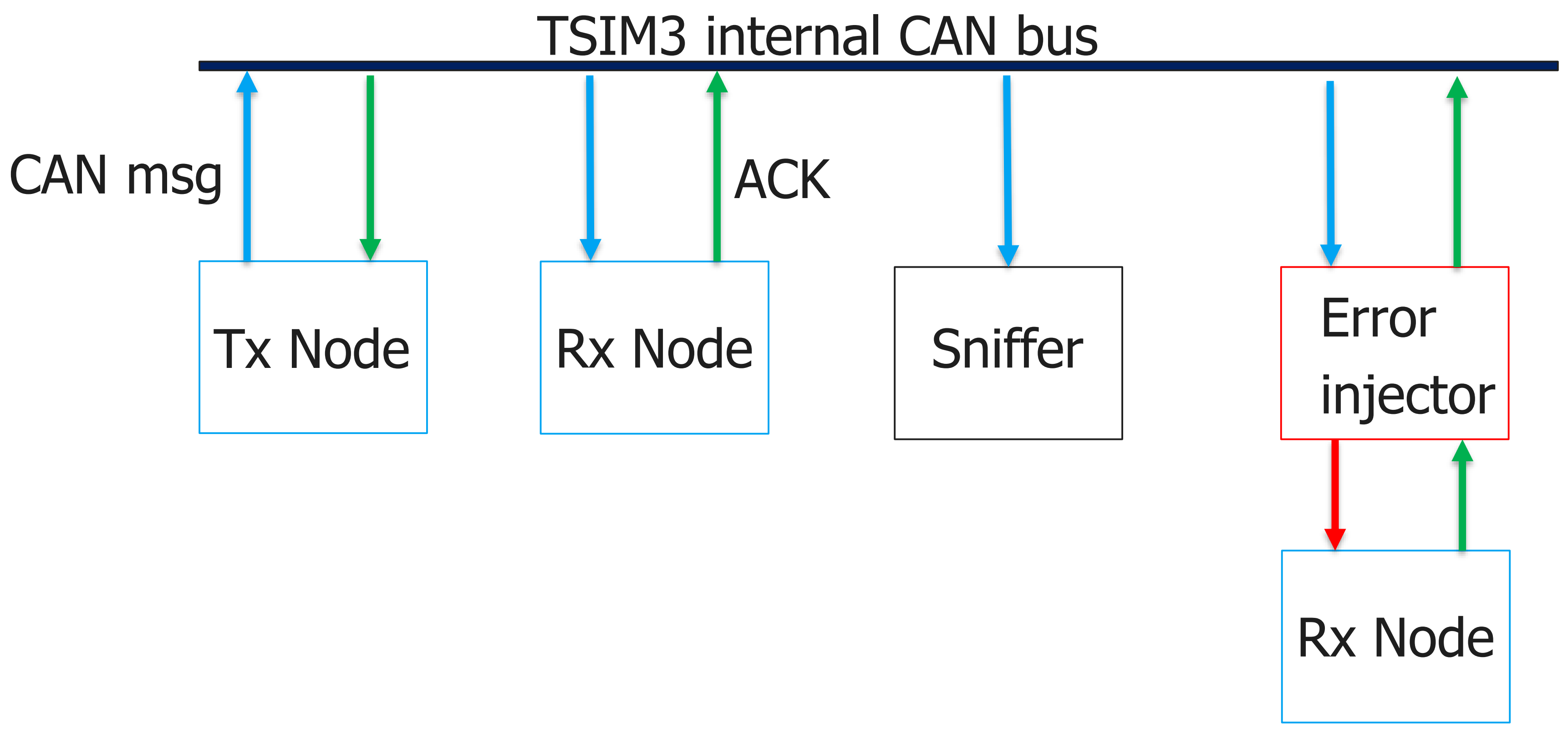
CAN in TSIM3

Sniffer node



CAN in TSIM3

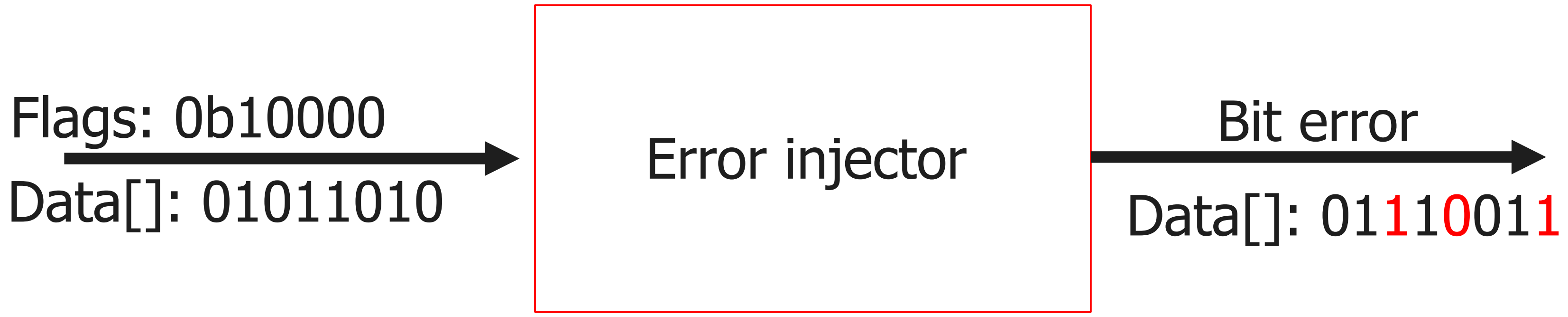
Sniffer node



CAN in TSIM3

Error node for GRCAN

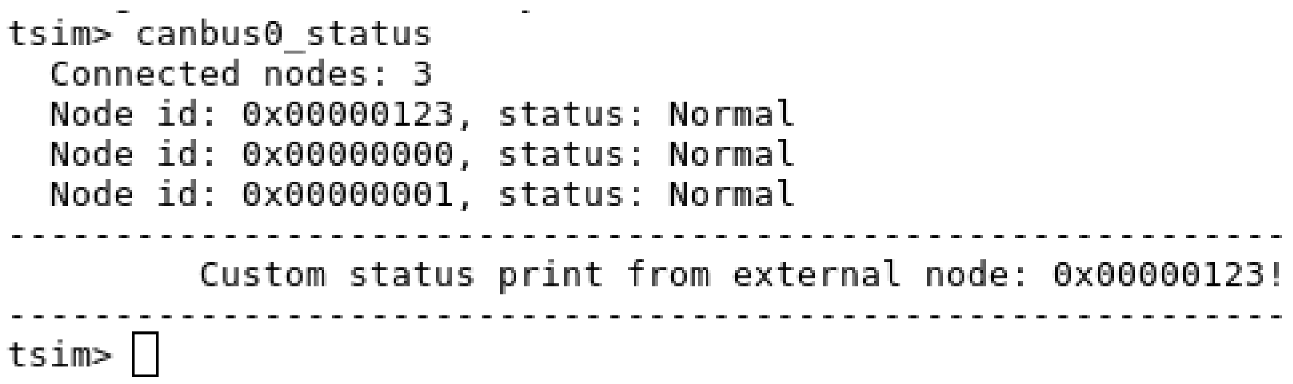
Bit index	Error type
0	Ack error
1	Form error
2	CRC error
3	Stuff error
4	Bit error
31	Error flagged by error passive node



TSIM

CAN diagnostics

- Displays number of connected nodes and their status
- Optional status callback to get node specific information directly in TSIM



```
tsim> canbus0_status
  Connected nodes: 3
  Node id: 0x00000123, status: Normal
  Node id: 0x00000000, status: Normal
  Node id: 0x00000001, status: Normal
------------------------------------------------------------
         Custom status print from external node: 0x00000123!
------------------------------------------------------------
tsim> 
```

TSIM3 for GR716

GR716 BETA

- TSIM3 GR716 BETA was released in April 2019
  - Some TSIM3 functionality temporarily disabled in BETA
- New and improved simulation models:
  - LEON3 CPU model updated with:
    - Local I/D RAM with dual-port (DMA and CPU)
    - 31 SPARC register windows
    - Cycle-counter
    - Floating Point Unit (FPU)
    - Register window partitioning
    - Memory controllers with access timings:
    - PROM, MRAM/SRAM interface
    - SPI-Memory controller
  - Atomic and bit-operations on local RAM and APB registers
  - SpaceWire, CAN, UART, SPI master, DAC
  - Timers, GPIO, AHB Status register, IRQMP with time-stamping and 64 remappable IRQs
  - AHBROM – including ROM Boot loader & Boot strapping registers

www.gaisler.com/tsim3-gr716

TSIM3 for GR716

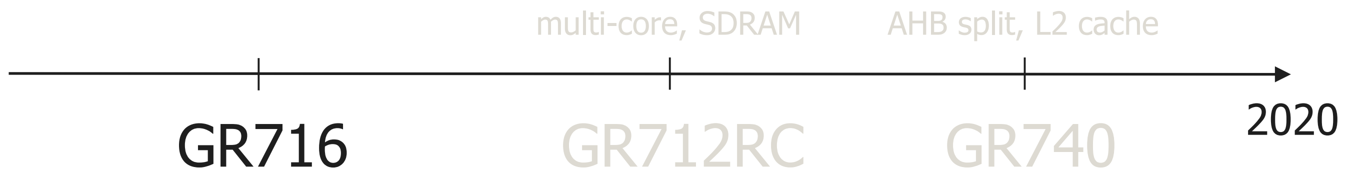
GR716 BETA

- Supports application load & start from different memory areas
  - Load and run directly from local or external memory
  - Boot using all boot configurations apart from I^{2}C and remote access
  - Allows testing of GR716 Application SW Image format loading and booting
- AHB bus trace now available
- GDB 8.2 source level debugging support
- Faster than real-time, benchmarks show 100-150% of GR716@50MHz

TSIM3 development on-going

TSIM2 and GRSIM successor

- Multiprocessor support
- Support for additional systems
  - GR712RC extended with multi CPU support
  - Quad core GR740
- Support for general system configurations (SoC designs)
- New internal architecture to handle more diverse system architectures
- Tcl scripting support for better automation of tests
- Support for additional I/O cores
- User extension possibilities for custom models
- Continuing the accuracy profile of TSIM2



TSIM3 and GR716

- TSIM3 has been used internally at Gaisler to successfully:
  - Execute RTEMS UP/SMP test-suites in GR712RC single-core and GR740 multi-core GR740 BSPs configurations during ESA LLVM development 2018
  - Used to run VxWorks 7 test-suites in single/multi-core configuration in GR712RC/GR740 controlled by Jenkins
  - Ported Zephyr Operating System to TSIM3-GR716 and used the new Register Window Partitioning successfully
  - Used to develop CAN remote boot demo for GR716

COBHAM