GreenBus

Wolfgang Klingauf

w.klingauf@tu-braunschweig.de

14th ESCUG Meeting

FDL’06 Darmstadt

GreenSocs

THE Open Source community based SystemC *infrastructure* project.

- Open to all to contribute / join / USE
- It is maintained and funded by subscriptions and contracts with GreenSocs Ltd.
- GreenSocs Ltd's "product" is

  co-ordination and delivery of open source infrastructure and construction of the open source community.
- GreenSocs uses a SourceForge repository to guarantee longevity.

GreenSocs SystemC Sandpit

Contributions

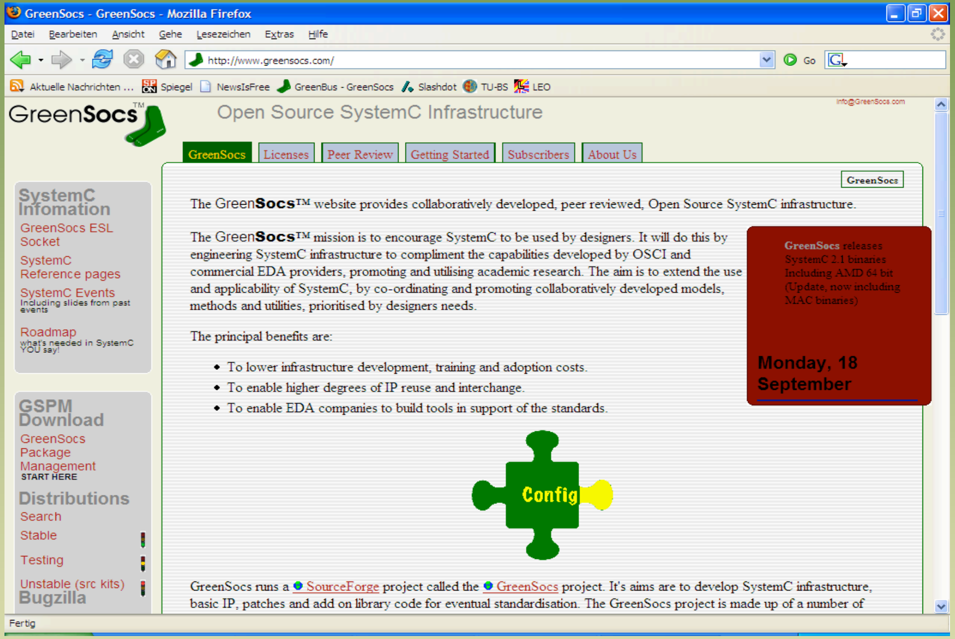
GreenSocs Engineering

De-Facto Standards

Passed to (e.g.) OSCI

NEEDS GUIDANCE:
Subscribers and GreenSocs Engineering

Central User Resource



Open Source SystemC Infrastructure

The GreenSocs™ website provides collaboratively developed, peer reviewed, Open Source SystemC infrastructure.

The GreenSocs™ mission is to encourage SystemC to be used by designers. It will do this by engineering SystemC infrastructure to compliment the capabilities developed by OSCI and commercial EDA providers, promoting and utilising academic research. The aim is to extend the use and applicability of SystemC, by co-ordinating and promoting collaboratively developed models, methods and utilities, prioritised by designers needs.

GreenSocs releases SystemC 2.1 binaries Including AMD 64 bit (Update, now including MAC binaries)

Monday, 18 September

The principal benefits are:

- To lower infrastructure development, training and adoption costs.
- To enable higher degrees of IP reuse and interchange.
- To enable EDA companies to build tools in support of the standards.

GreenSocs runs a SourceForge project called the GreenSocs project. It's aims are to develop SystemC infrastructure, basic IP, patches and add on library code for eventual standardisation. The GreenSocs project is made up of a number of

- PUBLIC Read/Write "information" Wiki
- GreenSocs projects pages (GreenBus)
- Other project pages
- OPEN - to encourage community…

All Aboard

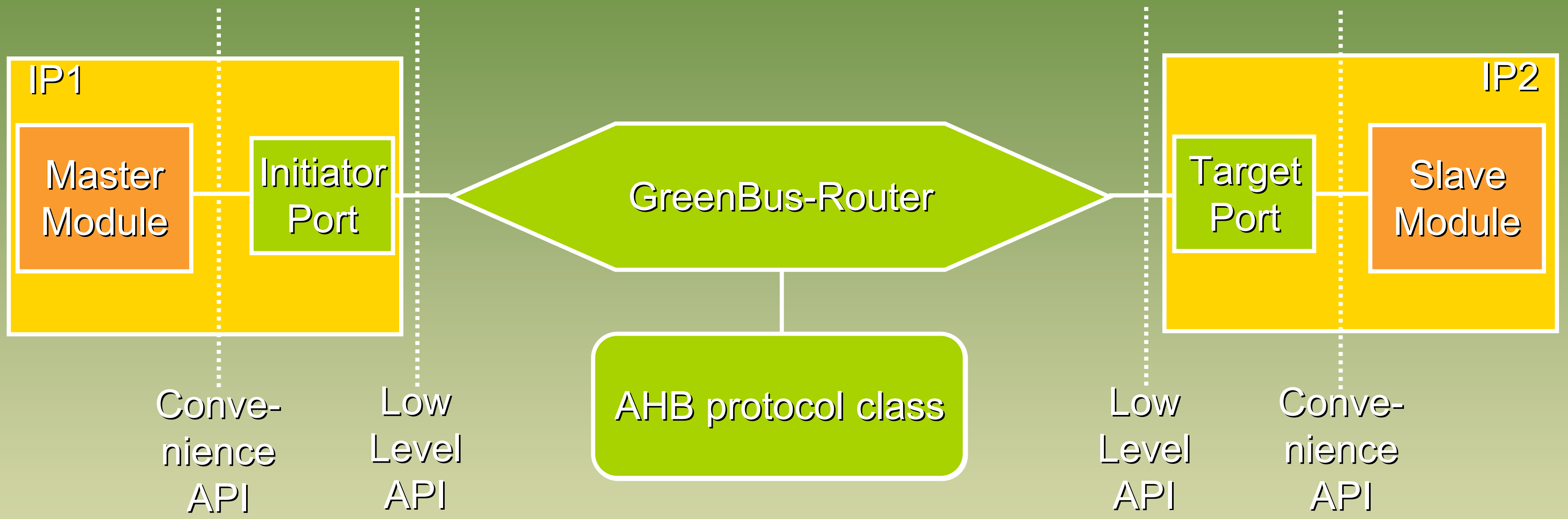
NOKIA Connecting People

simplex sigillum veri VaST™

Texas Instruments

ST PHILIPS

GreenBus (cont’d)



- **Interoperability Layer: Generic Protocol**
- **User Layer: IP-specific convenience APIs**

GreenBus

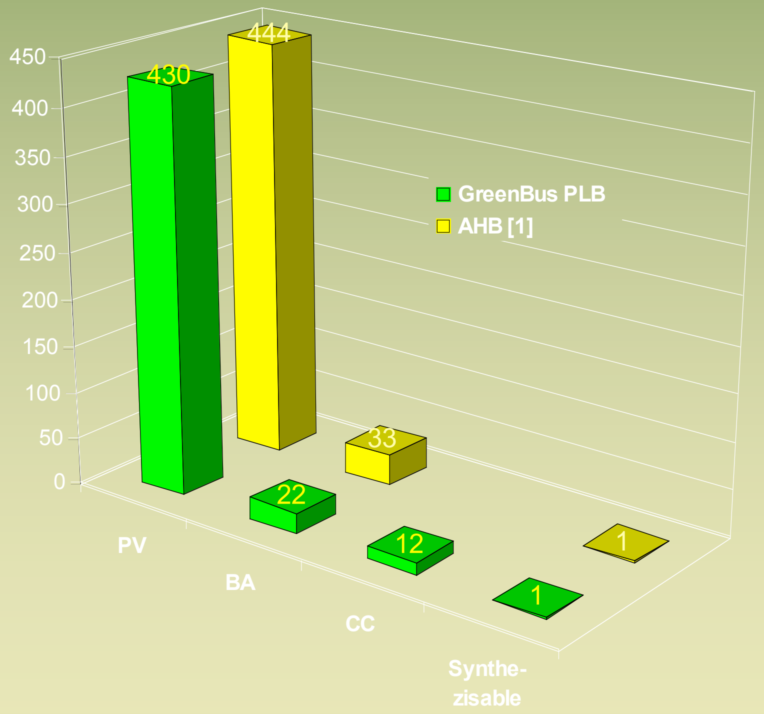
- GreenBus is not a bus!

- Inter-operability
  - Simple API between models
  - Defined data structures
- Safety
  - Automatic memory management for data structures
  - Event semantics between models
- Speed
  - Data passed by shared pointers (payload-events)
  - Only required events used
  - Model free to use methods (rather than wait() calls)

AVAILABLE

GreenBus (cont'd)

- **Other features of GreenBus:**
  - **Open Source**
  - **Built for/on proposed TLM 2.0 standard**



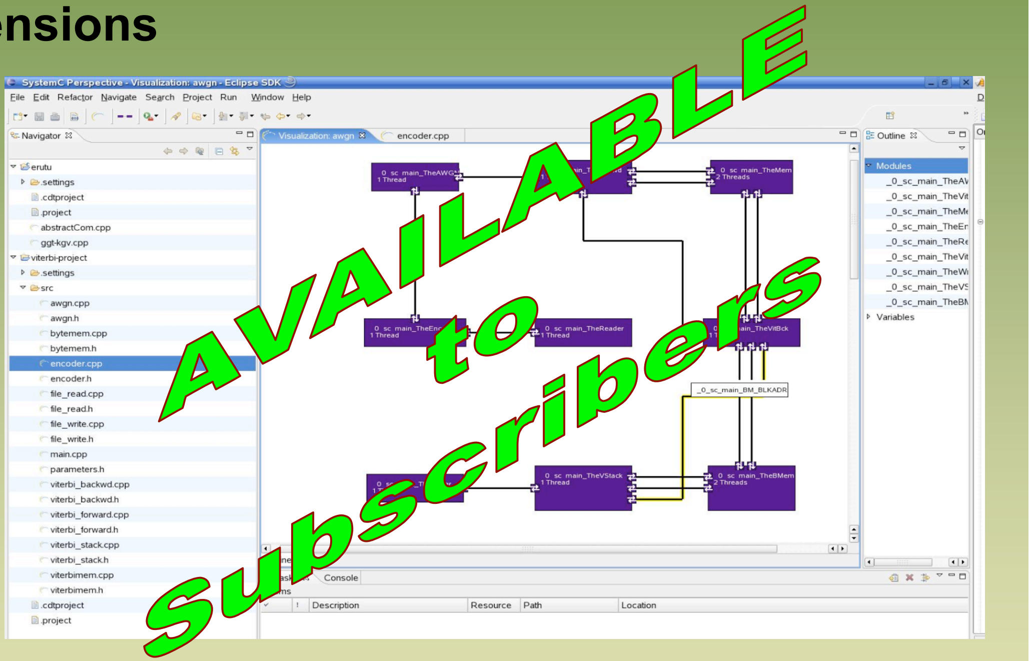
- **Efficient (0.5 million atoms per second)**
- **Multiple abstraction levels**
- **Native support for other user API's (OCP-IP etc)**

[DAC'06]

Enabling the Open SystemC User Community

Eclipse Environment

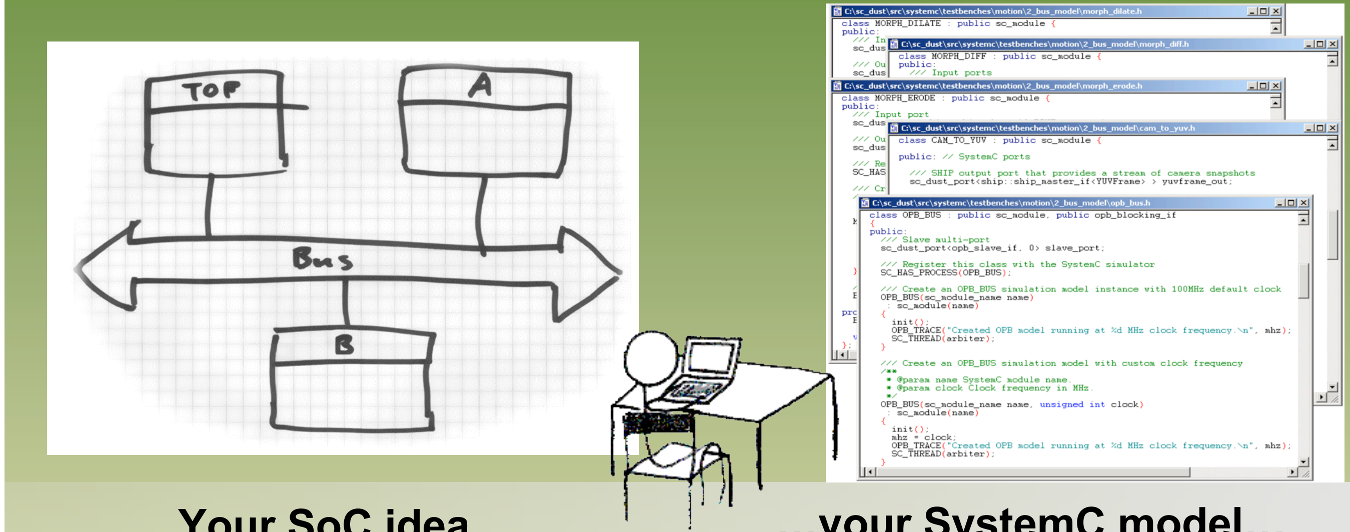
- Modular and extensible IDE for SystemC designs
- SystemC-specific features and extensions
  - Project management
  - Configuration
  - Visualization
  - Model execution
  - Module tree viewer
  - Structural representation



- Availability
  - Available for GreenSocs subscribers
  - Further customer-specific licenses and extensions are possible
  - GreenSocs funding will ensure public availability

DUST

Structure and Communication Analysis



C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\morph_dilate.h

```
class MORPH_DILATE : public sc_module {
public:
  /// In
  sc_dus

  /// Ou
  sc_dus
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\morph_diff.h

```
class MORPH_DIFF : public sc_module {
public:
  /// Input ports
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\morph_erode.h

```
class MORPH_ERODE : public sc_module {
public:
  /// Input port
  sc_dus

  /// Ou
  sc_dus

  /// Re
  SC_HAS

  /// Cr
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\cam_to_yuv.h

```
class CAM_TO_YUV : public sc_module {

public: // SystemC ports

  /// SHIP output port that provides a stream of camera snapshots
  sc_dust_port<ship::ship_master_if<YUVFrame> > yuvframe_out;
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\opb_bus.h

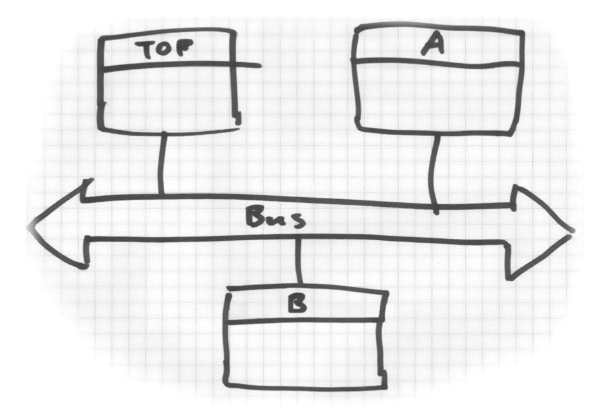
```
class OPB_BUS : public sc_module, public opb_blocking_if
{
public:
  /// Slave multi-port
  sc_dust_port<opb_slave_if, 0> slave_port;

  /// Register this class with the SystemC simulator
  SC_HAS_PROCESS(OPB_BUS);

  /// Create an OPB_BUS simulation model instance with 100MHz default clock
  OPB_BUS(sc_module_name name)
   : sc_module(name)
  {
    init();
    OPB_TRACE("Created OPB model running at %d MHz clock frequency.\n", mhz);
    SC_THREAD(arbiter);
  }

  /// Create an OPB_BUS simulation model with custom clock frequency
  /**
   * @param name SystemC module name.
   * @param clock Clock frequency in MHz.
   */
  OPB_BUS(sc_module_name name, unsigned int clock)
   : sc_module(name)
  {
    init();
    mhz = clock;
    OPB_TRACE("Created OPB model running at %d MHz clock frequency.\n", mhz);
    SC_THREAD(arbiter);
  }
```

Your SoC idea… …your SystemC model…

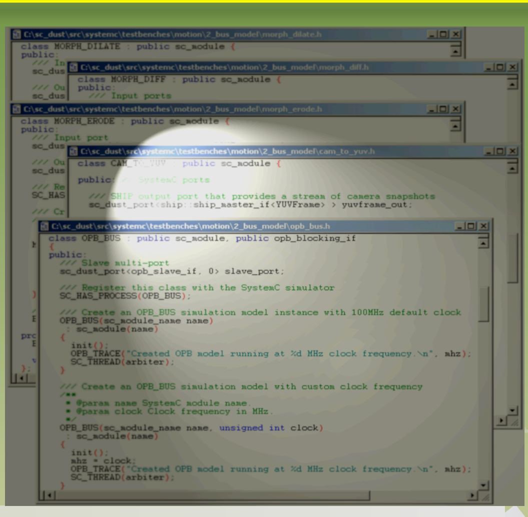
- Live design structure analysis
- Interactive transaction recording
- Component / Bus / MSC / Gantt View
- Works with any LRM-2.1+ kernel
- GreenBus support

DUST

Structure and Communication Analysis



Your SoC idea…



C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\morph_dilate.h

```
class MORPH_DILATE : public sc_module {
public:
  /// In
  sc_dus

  /// Ou
  sc_dus
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\morph_diff.h

```
class MORPH_DIFF : public sc_module {
public:
  /// Input ports
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\morph_erode.h

```
class MORPH_ERODE : public sc_module {
public:
  /// Input port
  sc_dus

  /// Ou
  sc_dus

  /// Re
  SC_HAS

  /// Cr
```

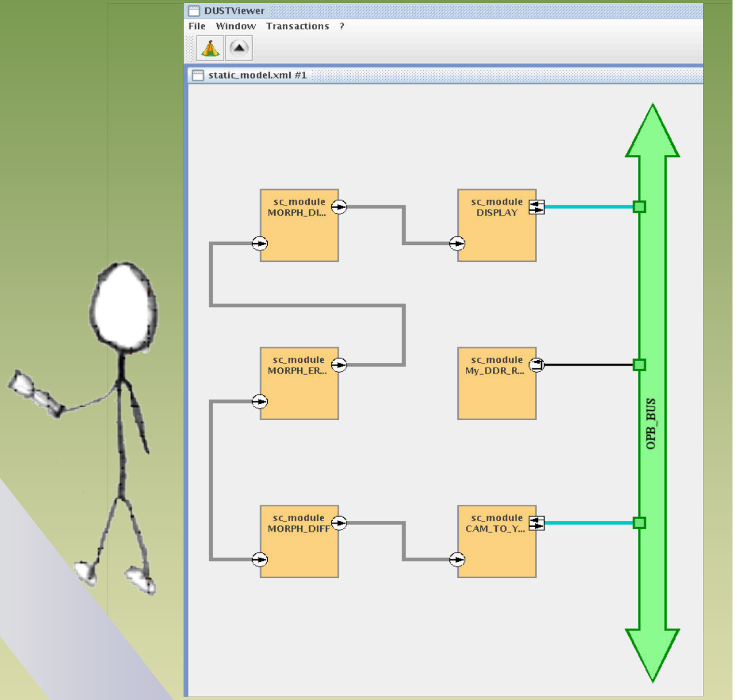
C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\cam_to_yuv.h

```
class CAM_TO_YUV : public sc_module {

public: // SystemC ports

  /// SHIP output port that provides a stream of camera snapshots
  sc_dust_port<ship::ship_master_if<YUVFrame> > yuvframe_out;
```

C:\sc_dust\src\systemc\testbenches\motion\2_bus_model\opb_bus.h

```
class OPB_BUS : public sc_module, public opb_blocking_if
{
public:
  /// Slave multi-port
  sc_dust_port<opb_slave_if, 0> slave_port;

  /// Register this class with the SystemC simulator
  SC_HAS_PROCESS(OPB_BUS);

  /// Create an OPB_BUS simulation model instance with 100MHz default clock
  OPB_BUS(sc_module_name name)
   : sc_module(name)
  {
    init();
    OPB_TRACE("Created OPB model running at %d MHz clock frequency.\n", mhz);
    SC_THREAD(arbiter);
  }

  /// Create an OPB_BUS simulation model with custom clock frequency
  /**
   * @param name SystemC module name.
   * @param clock Clock frequency in MHz.
   */
  OPB_BUS(sc_module_name name, unsigned int clock)
   : sc_module(name)
  {
    init();
    mhz = clock;
    OPB_TRACE("Created OPB model running at %d MHz clock frequency.\n", mhz);
    SC_THREAD(arbiter);
  }
```

…your SystemC model…



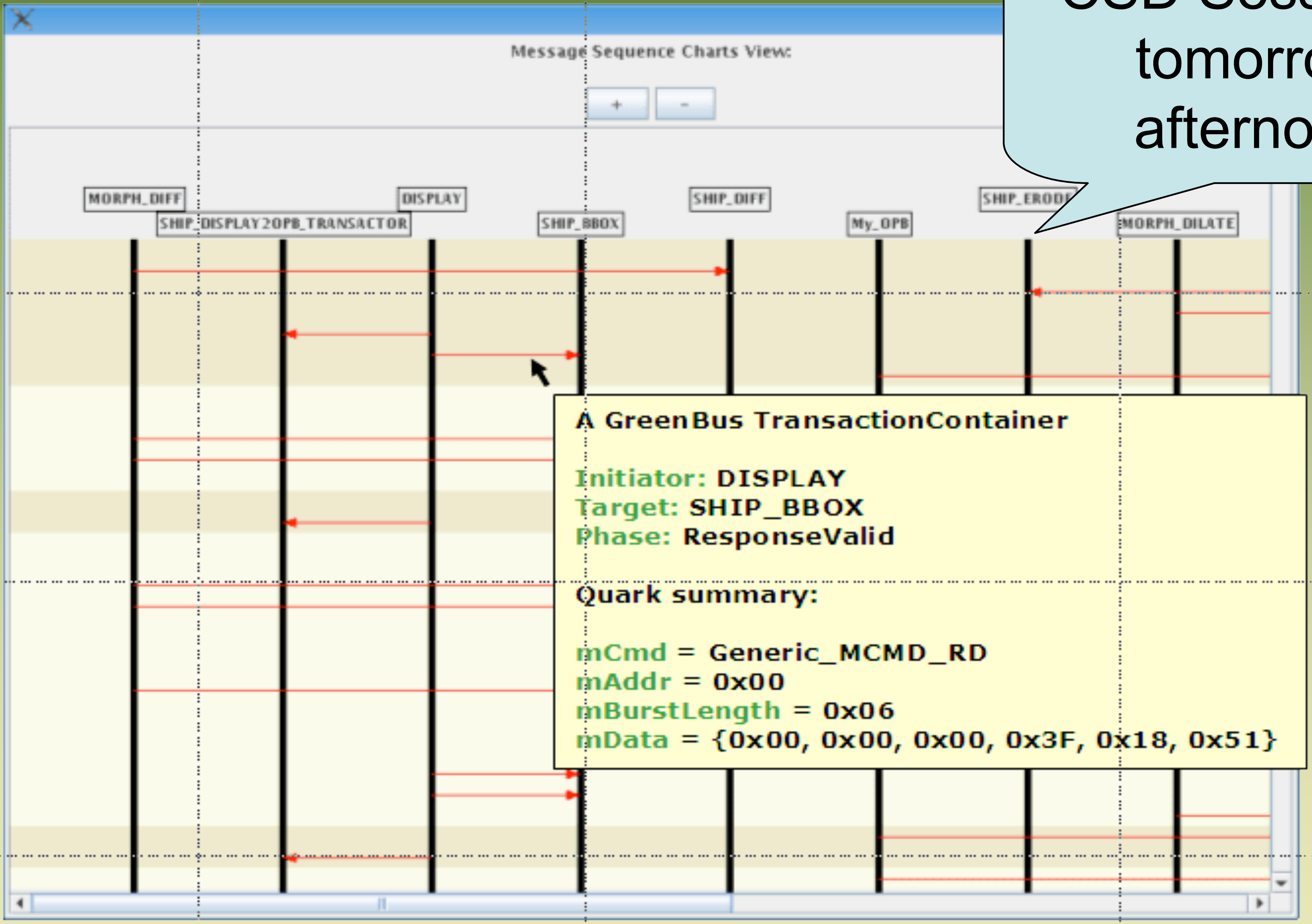
…the visual analysis

- Live design structure analysis
- Interactive transaction recording
- Component / Bus / MSC / Gantt View
- Works with any LRM-2.1+ kernel
- GreenBus support

DUST

Structure and Communication Analysis

CSD Session 3 tomorrow afternoon



A GreenBus TransactionContainer

Initiator: DISPLAY
Target: SHIP_BBOX
Phase: ResponseValid

Quark summary:

```
mCmd = Generic_MCMD_RD
mAddr = 0x00
mBurstLength = 0x06
mData = {0x00, 0x00, 0x00, 0x3F, 0x18, 0x51}
```

Package Management

EASE OF USE

- Provide a common environment for packages
- Make it easy to
  Install / Evaluate / Remove / Upgrade
- Keep track of dependencies, etc….
- Allow multiple installed "worlds"
- All packages can be qualified in a known environment (gcc, systemc, …).
- Based on open source applications
- Available to ALL ESL vendors (can be used with flex etc)

Package Management

EASE OF USE

- Provide a common environment for packages
- Make
  Ins
- Keep
- Allow
- All pa
  enviro
- Based on open source applications
- Available to ALL ESL vendors (can be used with flex etc)

```
> tar -zxf gsroot_2006.06-1_i386.tar.gz
> ./gsroot/bin/gsconfig
> ./gsroot/bin/gsh
$ gs-apt-get update
$ gs-apt-get install gstlm
$ cd examples/exaples_simple
$ gscsim example_simple.cpp
```

Package Management

EASE OF USE

- Provide a common e
- Make it easy to
  Install / Evalu
- Keep track of d
- Allow multiple
- All packages ca
  environment (gcc
- Based on open sou
- Available to ALL E
  flex etc)

- Lots of packages in the queue waiting to be done….

- ST TAC
- CoWare SCML
- Mentor's O/S package
- Etc…

That's it

Thank you!

Wolfgang.Klingauf@greensocs.com

www.greensocs.com