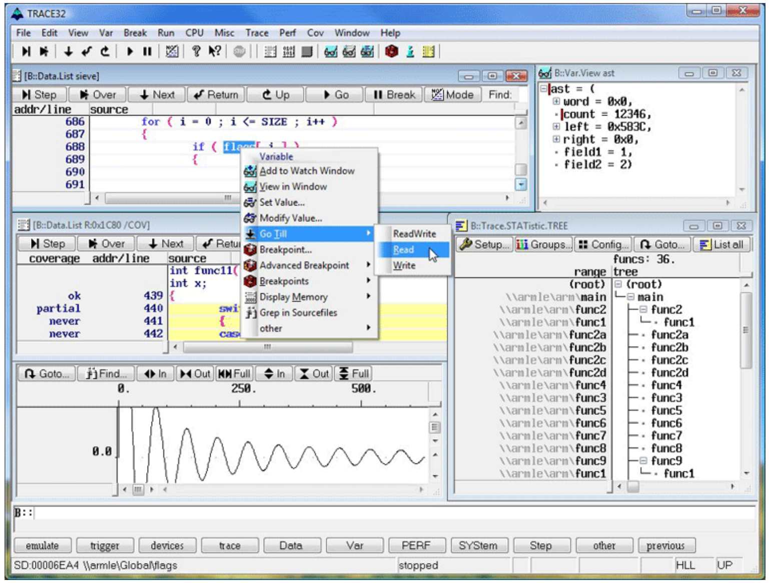
# Uniform Look-And-Feel for all Processors


TRACE32
File Edit View Var Break Run CPU Misc Trace Perf Cov Window Help
[B::Data.List sieve]
Step Over Next Return Up Go Break Mode Find:
B::Var.View ast
Variable
Add to Watch Window
View in Window
Set Value...
Modify Value...
Go Till
Breakpoint...
Advanced Breakpoint
Breakpoints
Display Memory
Grep in Sourcefiles
other
ReadWrite
Read
Write
[B::Data.List R:0x1C80 /COV]
Step Over Next Retur
B::Trace.STATistic.TREE
Setup... Groups... Config... Goto... List all
Goto... Find... In Out Full In Out Full
0. 250. 500.
0.0
emulate trigger devices trace Data Var PERF SYStem Step other previous

```
addr/line  source
      686        for ( i = 0 ; i <= SIZE ; i++ )
      687        {
      688                if ( flags[ i ] )
      689                {
      690
      691
```

```
ast = (
  word = 0x0,
  count = 12346,
  left = 0x583C,
  right = 0x0,
  field1 = 1,
  field2 = 2)
```

```
coverage  addr/line  source
                     int func11(
                     int x;
      ok        439  {
 partial        440          swi
   never        441          {
   never        442          cas
```

```
                  funcs: 36.
                range tree
               (root) (root)
    \\armle\arm\main  main
   \\armle\arm\func2    func2
   \\armle\arm\func1      func1
  \\armle\arm\func2a    func2a
  \\armle\arm\func2b    func2b
  \\armle\arm\func2c    func2c
  \\armle\arm\func2d    func2d
   \\armle\arm\func4    func4
   \\armle\arm\func3    func3
   \\armle\arm\func5    func5
   \\armle\arm\func6    func6
   \\armle\arm\func7    func7
   \\armle\arm\func8    func8
   \\armle\arm\func9    func9
   \\armle\arm\func1      func1
```

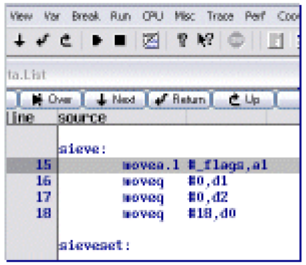
```
B::
```

```
SD:00006EA4 \\armle\Global\flags    stopped    HLL    UP
```

## Highlights

- ASM Debugger/HLL Debugger
- FLASH Flash Programmer
- CTS Context Tracking System
- STATISTIC and PERFORMANCE Analyzer
- COVERAGE Analyzer
- RTOS Debugger
- INT Tools Integration
- Front-End to TRACE32 Instruction Set Simulator
- Front-End to Third-Party Core Simulators/Virtual Prototypes
- Front-End to Third-Party Target Servers

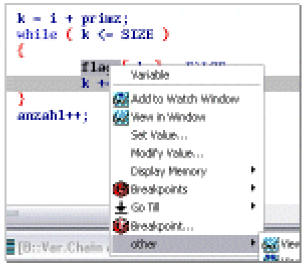
# Features



```
sieve:
        movea.l #_flags,a1
        moveq   #0,d1
        moveq   #0,d2
        moveq   #18,d0

sieveset:
```

## ASM Debugger

- Supports almost all file formats
- Assembler source-level debugging
- Advanced memory display
- Inline assembler
- Memory tests
- Customizable windows
- Peripheral windows
- Terminal window
- Semi-hosting
- Flash programming
- Full support for peripherals



```
k = i + primz;
while ( k <= SIZE )
{
        flag
        k +
}
anzahl++;
```

## High-Level-Language Debugging

- Supports multiple languages
- Full support for C++
- Integrated into TRACE32 environment
- Supports most compilers and hosts
- Same user interface on different hosts
- High speed download
- Debugs optimized code
- Display of function nesting
- Display of linked lists
- Powerful expression evaluation

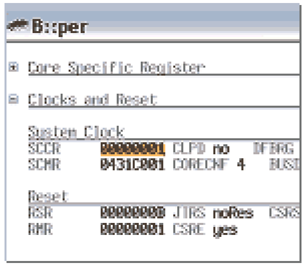
```
k = i + primz;
while ( k <= SIZE )
{
        flag
        k +
}
anzahl++;
```

## SIM Instruction Set Simulators

- Easy high-level and assembler debugging
- Interface to all compilers
- Trace Buffer
- Powerful script language
- Software compatible to all TRACE32 tools
- Hardware simulation

## ROM Monitor

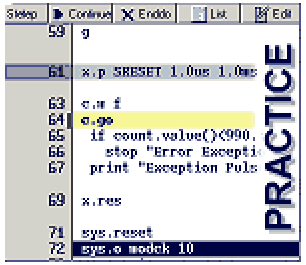
- Compatible with Emulator
- Support for C,C++ and ASM
- Communication via Eprom Simulator
- Communication via RS232 or customized .DLL link
- Windows9x, WindowsNT and Unix
- Monitor Code with Source
- Monitor Code Royalty Free



```
B::per
⊞ Core Specific Register
⊟ Clocks and Reset
  System Clock
  SCCR      00000001 CLPD no      DFBRG
  SCMR      0431C001 CORECNF 4    BUSD
  Reset
  RSR       0000000D JTRS noRes   CSRS
  RMR       00000001 CSRE yes
```

## Logical Display of Peripherals

- Display of onchip peripherals
- User definable windows
- Interactive window definition with softkey support
- Pulldown menus for selection of choices
- Additional description for each field



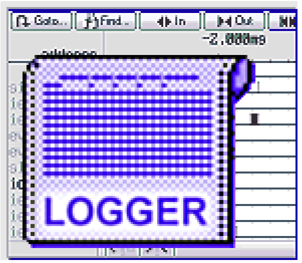
```
g

x.p SRESET 1.0us 1.0ms

c.m f
c.go
 if count.value()<990.
   stop "Error Excepti
 print "Exception Puls

x.res

sys.reset
sys.o modck 10
```

## Script Language PRACTICE

- Structured Language
- Menu Support
- Command Logs
- Custom Menus
- Custom Toolbars and Buttons
- Custom Dialog Windows
- 64-Bit Arithmetic
- Numeric, Logical and String Operators
- Direct Access to System States

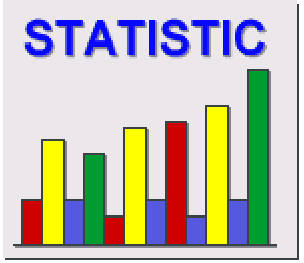
## CTS Trace-based Debugging

- Allows re-debugging of a traced program section
- Provides forward and backward debugging capabilities
- High-level language trace display including all local variables
- Timing and function nesting display
- Has the ability to fill most trace gaps caused by the limited bandwidth of trace port



## Logger

- Software trace of any size stored in an array structure on the target
- General trace format provided by TRACE32-PowerView
- Configuration and display commands provided by TRACE32-PowerView
- Works as trace with address and data information
- Works as a program flow trace (SH4, PowerPC)
- Time stamp possible
- Predefined algorithms to fill the trace provided by Lauterbach
- User defined algorithms to fill the trace also possible

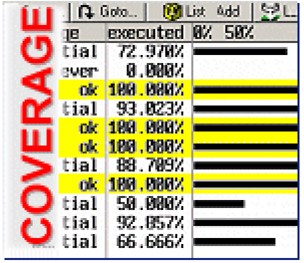


## Trace-based Profiling

- Detailed analysis of function run-times
- Detailed analysis of task run-times and state
- Graphical analysis of variable values over the time
- Analysis of the time interval of a single event (e.g. Interrupt)
- Analysis of the time interval between 2 defined events

## Sample-based Profiling

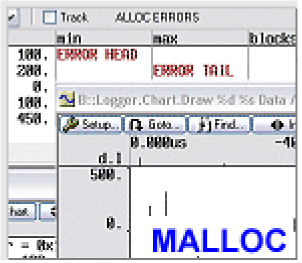
- Long-time performance analysis for functions
- Long-time performance analysis for tasks
- Long-time analysis of the contents of a variable or memory location and more



| ge | executed | 0% 50% |
|---|---|---|
| tial | 72.970% | |
| ever | 0.000% | |
| ok | 100.000% | |
| tial | 93.023% | |
| ok | 100.000% | |
| ok | 100.000% | |
| tial | 88.709% | |
| ok | 100.000% | |
| tial | 50.000% | |
| tial | 92.857% | |
| tial | 66.666% | |

## Trace-based Code Coverage

- Real-time code coverage without instrumentation
- Suitable for long-term testing
- Analysis for both assembly and source code level
- Off-line review capabilities
- Full support of multicore chips



## Memory Analysis

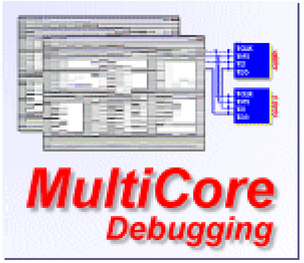
- Display of allocated memory blocks
- Memory allocation Statistics
- Check for out-of-bounds writes
- Trace of allocation calls
- Graphical displays of memory usage

## NOR FLASH Programming

- Internal and/or external NOR FLASH memories
- All common NOR FLASH types
- Programming of multiple NOR FLASH devices
- Provided by debuggers and in-circuit emulators

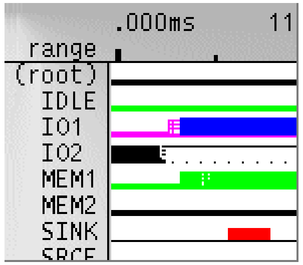
## NAND FLASH Programming

- Generic and CPU-specific NAND FLASH controllers
- Support all common NAND FLASH devices
- Bad block treatment (skipped, reserved block area)
- ECC generation



## Multicore Debugging

- Debugging support for homogeneous and heterogeneous multiprocessor and multicore systems
- High quality standard debuggers can be combined for multiprocessor and multicore systems
- All TRACE32-ICD debuggers are designed to work together in a multiprocessor/multicore debugging environment
- Fast integration of third party debuggers
- Several processors in a single piece of silicon can share the same debug port
- Start and stop synchronisation



## OS-aware Debugging

- Statistic evaluation and graphic display of task run times
- Task related evaluation of function run times
- Statistic evaluation and graphic display of task states
- Manual executed system calls
- Task stack coverage
- PRACTICE functions for OS data
- RTOS related pull-down menu
- Task selective debugging

## Help System

- Acrobat Based Documentation
- Fast Text Search
- Device Specific Filtering
- Basic and Advanced Help
- Training Manuals Included
- WWW Update

**DISTRIBUTOR**
**FLASH TECHNOLOGY PTE LTD**
**Website : www.flashtech.com.sg**
**Email : sales_sg@flashtech.com.sg**
**Tel : +65 6749 6168**