Many-core and Reconfigurable Supercomputing Conference (MRSC) 2008
The Queen's University of Belfast, Northern Ireland

Cellular Automata on Nallatech H101-PCIXM: first results

Anton Shterenlikht
Mechanical Engineering Department, University of Bristol
University Walk, Bristol BS8 1TR, UK
Email mexas@bristol.ac.uk, Tel 0117 928 8233

March 31, 2008

1 Application

The Cellular Automata (CA) – Finite Element (CAFE) model for transitional ductile to brittle fracture in steels [1] relies on CA arrays to represent microstructure.

CA is a finite time machine with discrete cell states, which depend on states of some neighbouring cells via very simple transfer rules.

CA are `integer` or `logical(boolean)` arrays.

Fracture propagation with CA is a change of state from 'alive' to 'dead'. Computationally CA are much cheaper than FE.

1.1 Microstructure generation with CA

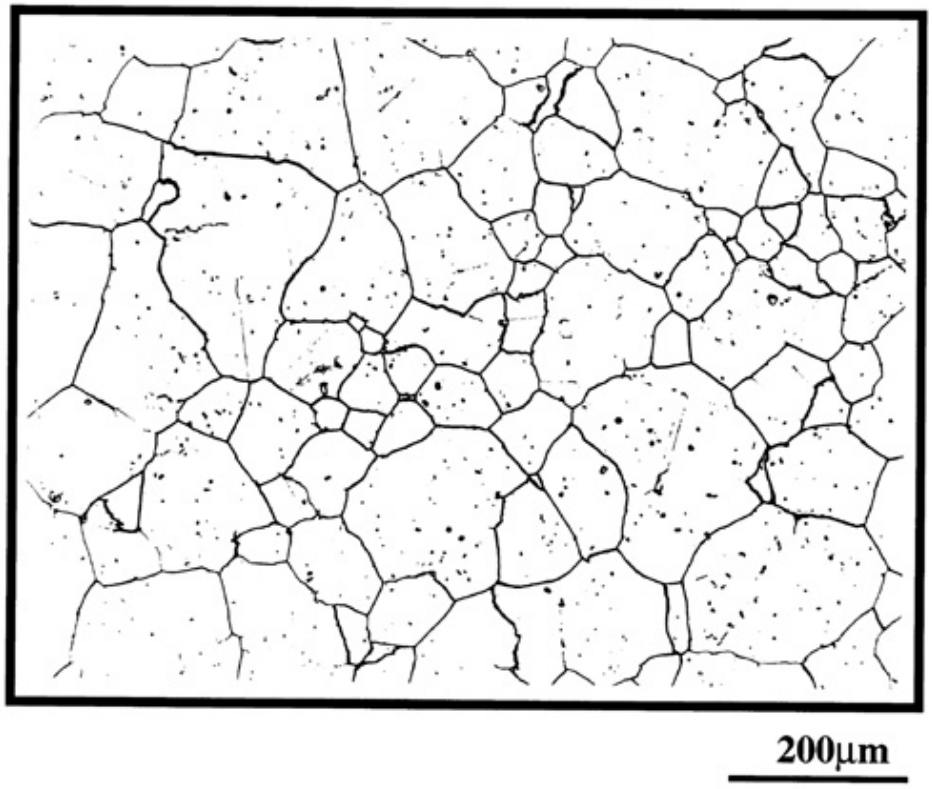


Figure 1: Grain micrograph

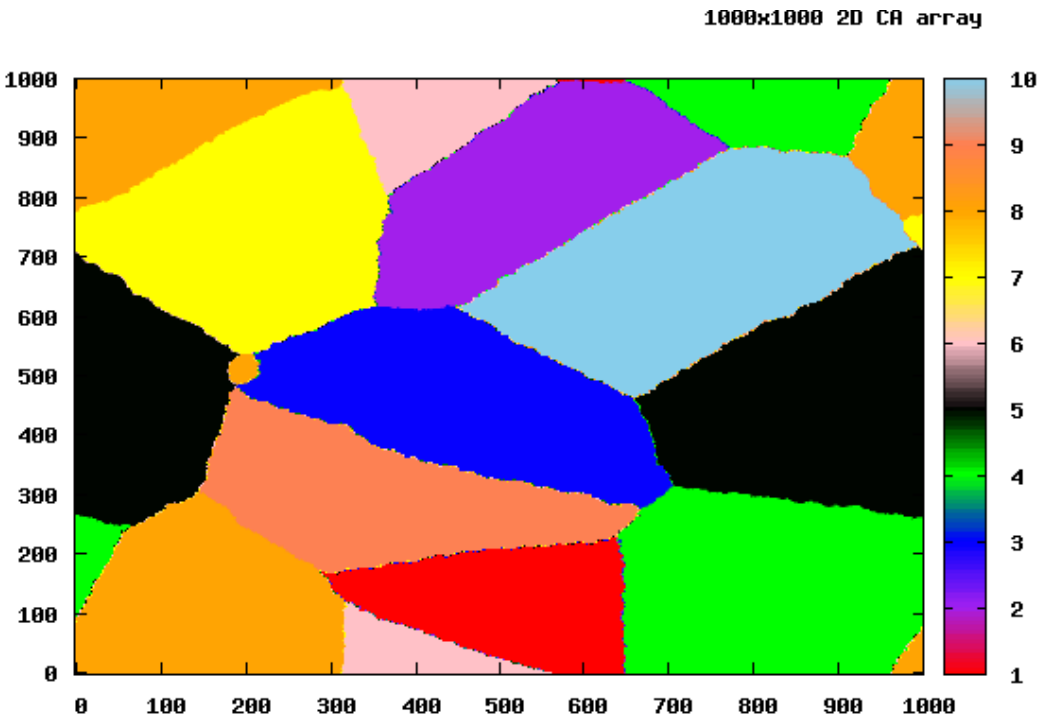


Figure 2: CA simulated 2D grain microstructure, 1M cells, 10 grains (colour denotes grain orientation). CA cells are approx. 2 μm.

2 Nallatech H101-PCIXM FPGA

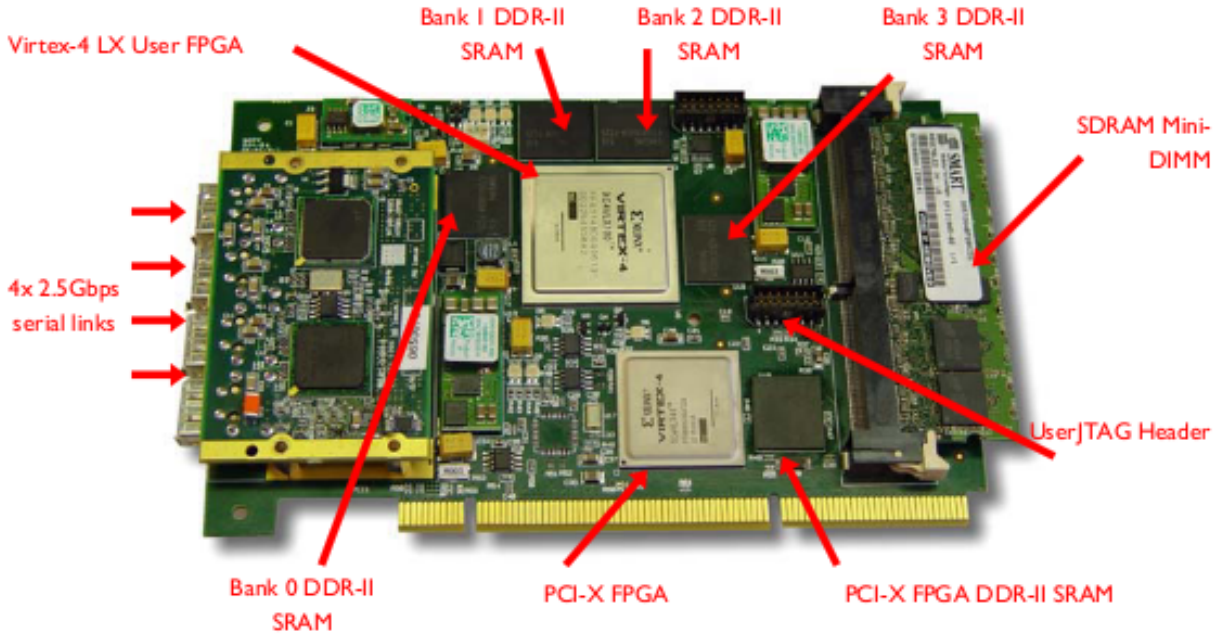


Figure 3: H101-PCIXM top view [2].

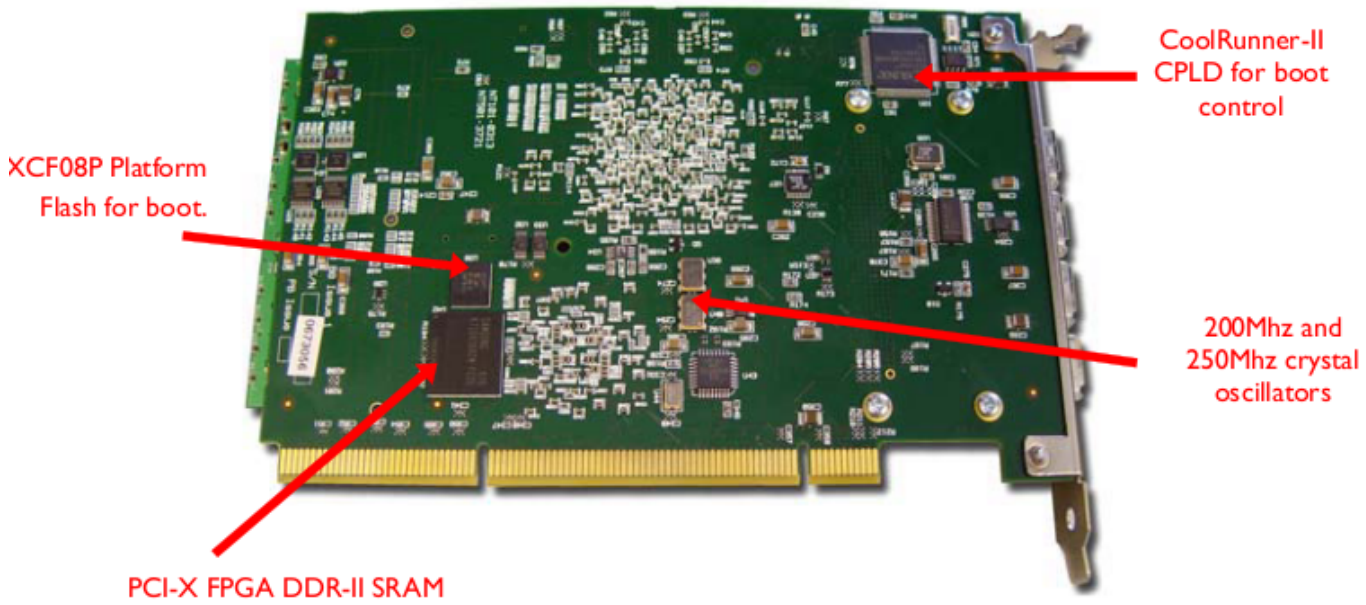


Figure 4: H101-PCIXM bottom view [2].

2.1 Nallatech H101-PCIXM tech data

- Xilinx Virtex-4 LX100 (XC4VLX100-10FF1148C) - FPGA. Maximum FPGA clock frequency - 200 MHz.
- 0.4MB internal block RAM per FPGA.
  0.5 TB/sec bandwidth and can be pipelined.
- 4 banks (4×4=16MB) of DDR-II SRAM.
  6.4 GB/sec bandwidth and can be pipelined.
- 1 bank (512MB) DDR-II SDRAM Mini-DIMM.
  3.2 GB/sec bandwidth and CANNOT be pipelined.
- 133 MHz capable PCI-X interface.
- 4×2.5Gbps external serial link channels (Infiniband).

3 Host computer

Any i386 or x86-64 host can be used. In this work HP Proliant GL360g3 server with dual core Intel® Xeon™ 3.06GHz CPU and 1MB RAM was used.

4 Code execution on a host with FPGA

code.c listing:

[...]

```
/* Write the toggled value}\\
   of Control to kick start} \\
   the DIMEc process} \\
*/
DIMETalk\_Write(hTalk,&Config,     <- FPGA API call
   ONEWORD,address0,
   memory_map_0,timeout)

getchar();                          <- ANSI C

/* Keep reading whilst busy
*/
while ( (Config&0x4) != 0 ) {       <- ANSI C
   DIMETalk_Read(hTalk,           <- FPGA API call
   &Config,ONEWORD,ddress0,
   memory_map_0,timeout);
```

[...]

So it is very easy to delegate tasks to H101 from ANSI C using FPGA API calls, which Nallatech call FUSE API calls [3, 4].

The most important FUSE API calls for end user are

- `DIMETalk_Write` - write data to H101 nodes
- `DIMETalk_Read` - read data from H101 nodes

5 Code design for a host with H101

1. Decide what parts of the original code are best suited for FPGA.
2. Translate those parts to C functions.
3. Optimise C for FPGA.
4. Translate optimised C to VHDL (hardware description language). This is done with Nallatech DIME-C [5] compiler.
5. Program H101, i.e. build a network to be implemented on the Xilinx chip. This is done with Nallatech DIMETalk [6] application builder tool.
6. Add DIMETalk API calls, which will talk to H101, to the source code.
7. Compile the complete code, with e.g. gcc, using Nallatech libraries.
8. Load the network to H101.
9. Run the code on host.

6 Fortran → C → DIME-C

The original CA code was in F90. No automatic Fortran to C conversion tool available for F90 and above, `f2c` works only for F77.

First, all F90 code was translated to C manually !

Second, DIME-C does not support following ANSI C structures, relevant to the CA code. The full list is in [5].

1. pointers
2. multi-dimensional arrays
3. `Switch` (Fortran `CASE`) statements
4. `do-while` loops
5. labeled and `goto` statements
6. `const` type qualifier

7 DIME-C compilation

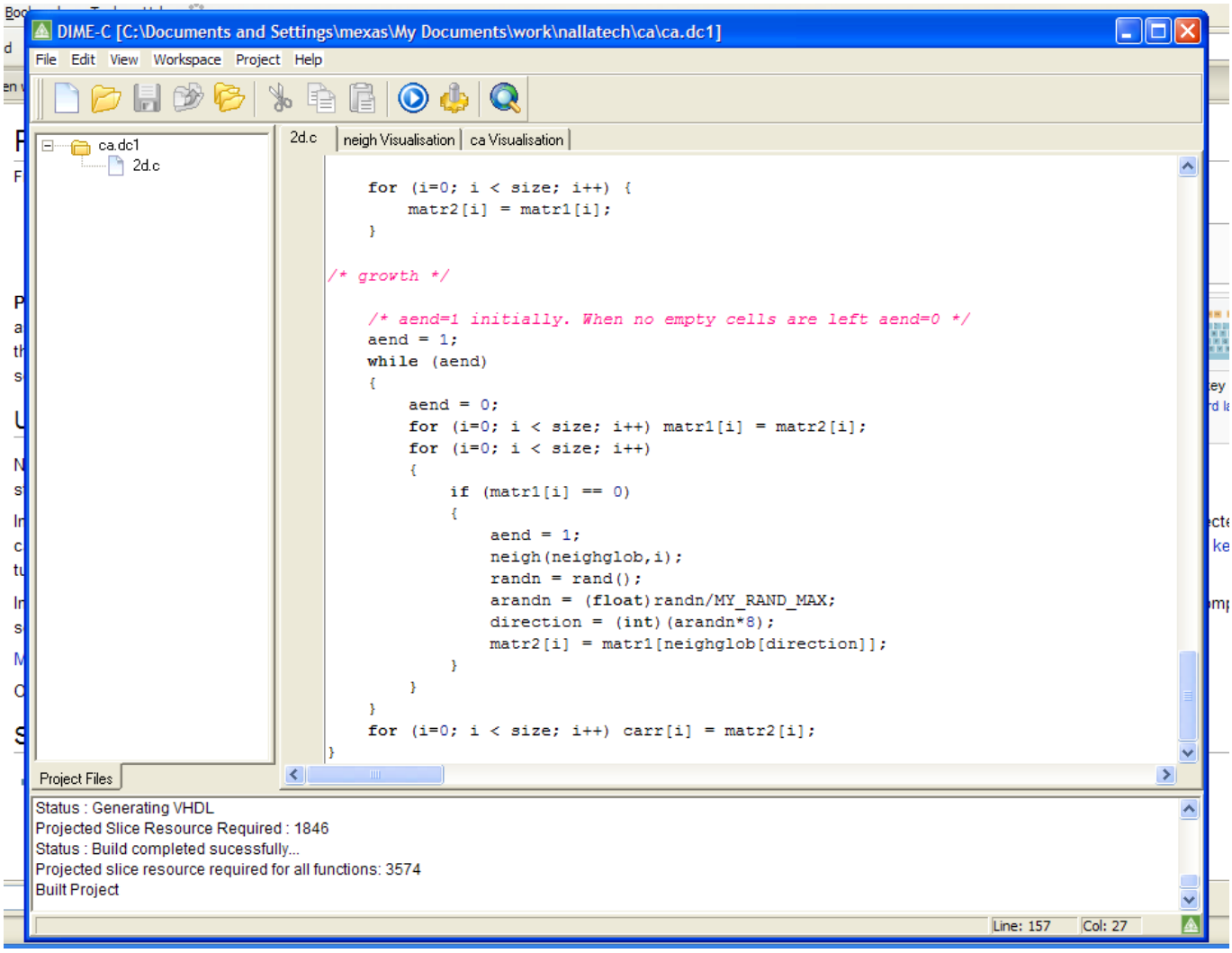


Figure 5: DIME-C screenshot with CA code.

Important:

- Only innermost loops are pipelined, so nested loops are bad.
- If a variable is assessed and assigned in a scope, the scope cannot be pipelined.
- `math.h` and generic `rand()` are provided.

8 DIME-C pipelining (parallelisation)

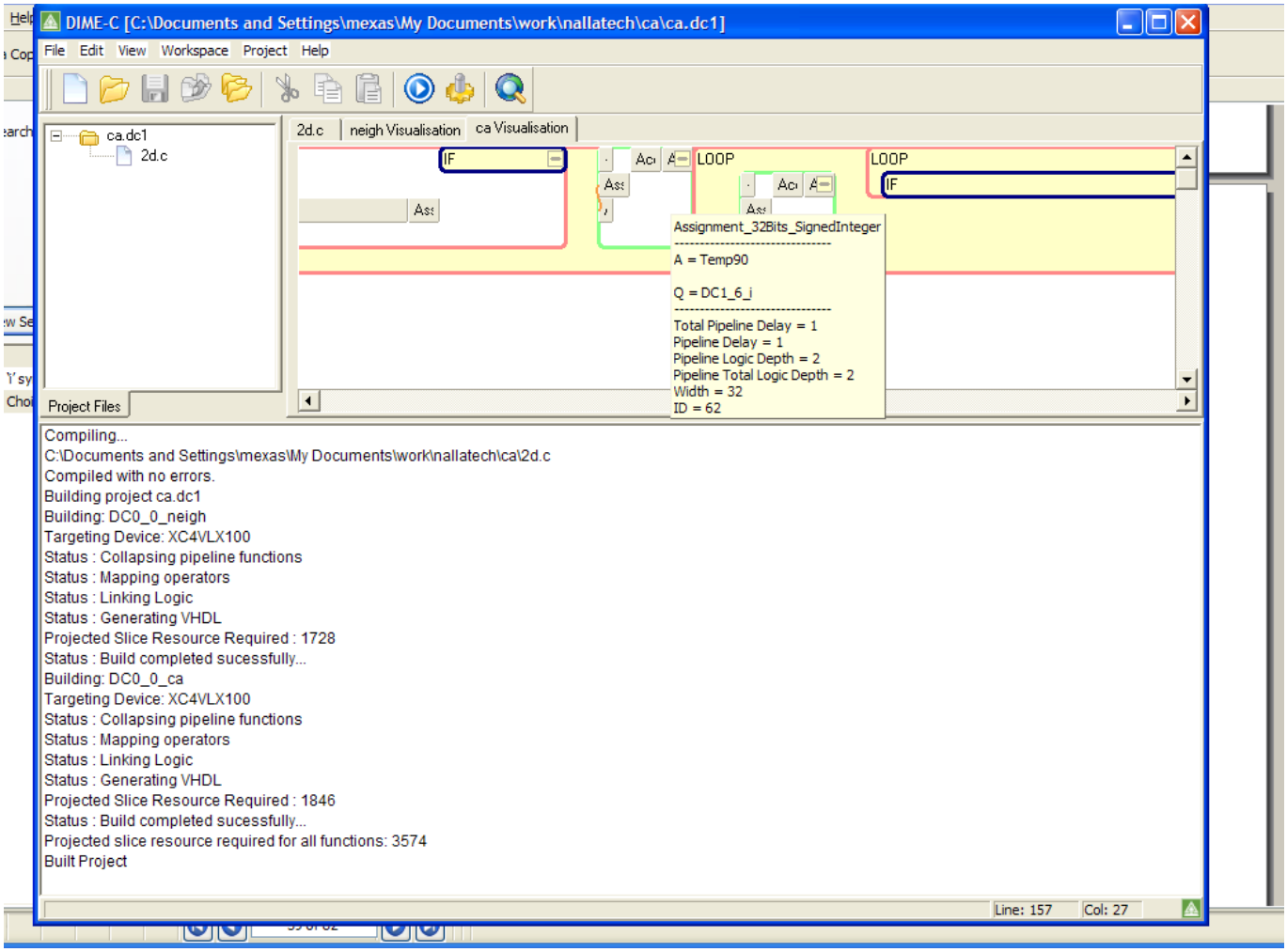


Figure 6: DIME-C pipelining analysis for CA code.

The secret of FPGA success is parallelisation or pipelining. DIME-C provides the pipelining report. The colourscheme is:

Green	pipelined loop - fast
Pink	not pipelined - delays
Blue	choice - not pipelined -delays

Also the size of resulting VHDL is reported in slices. H101 Virtex chip has 49k slices. This code needs 3.5k.

9 DIMETalk – building FPGA network

DIMETalk is a graphical tool for interconnecting pre-existing VHDL primitives, plus the user-defined VHDL, and creating a complete FPGA network.

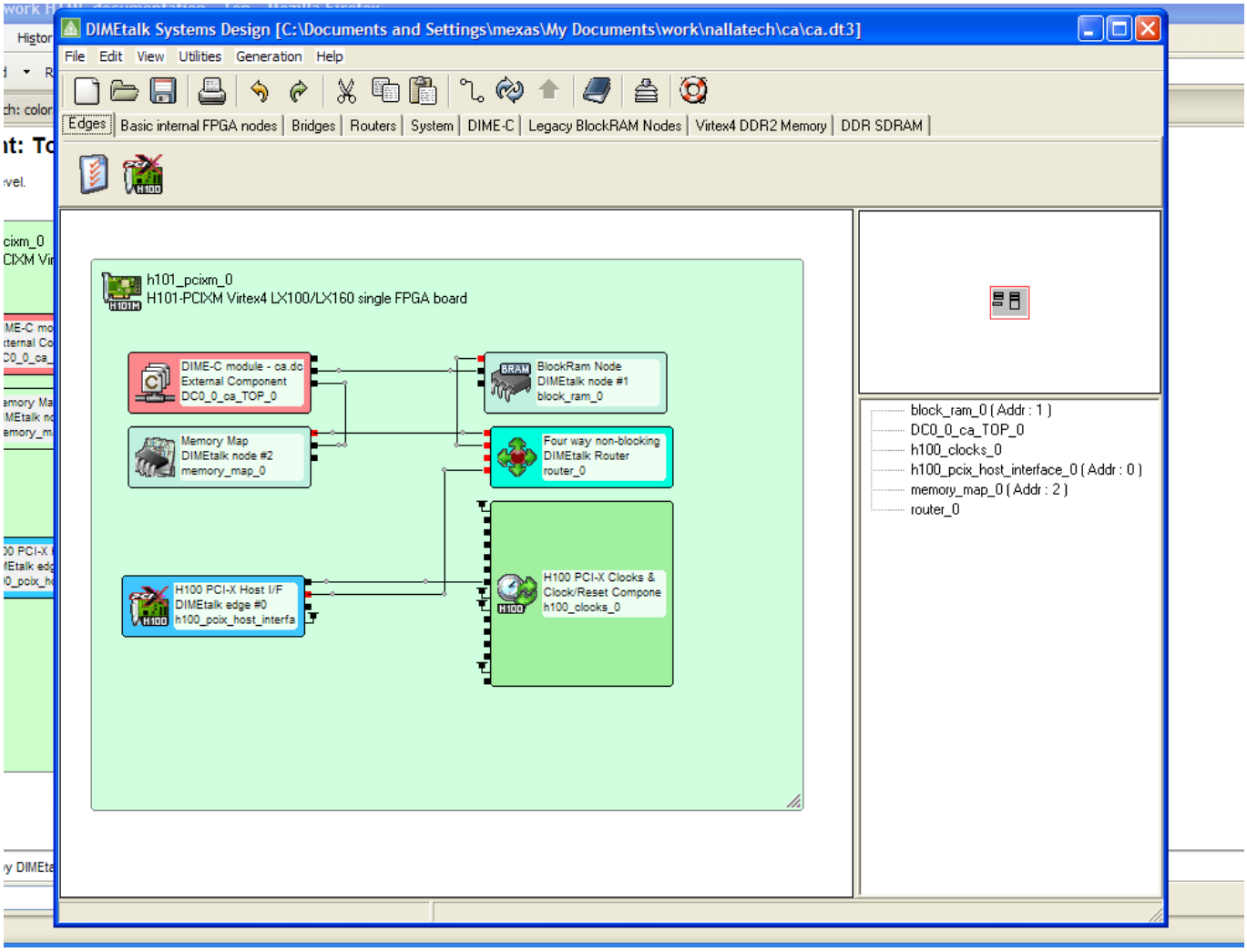
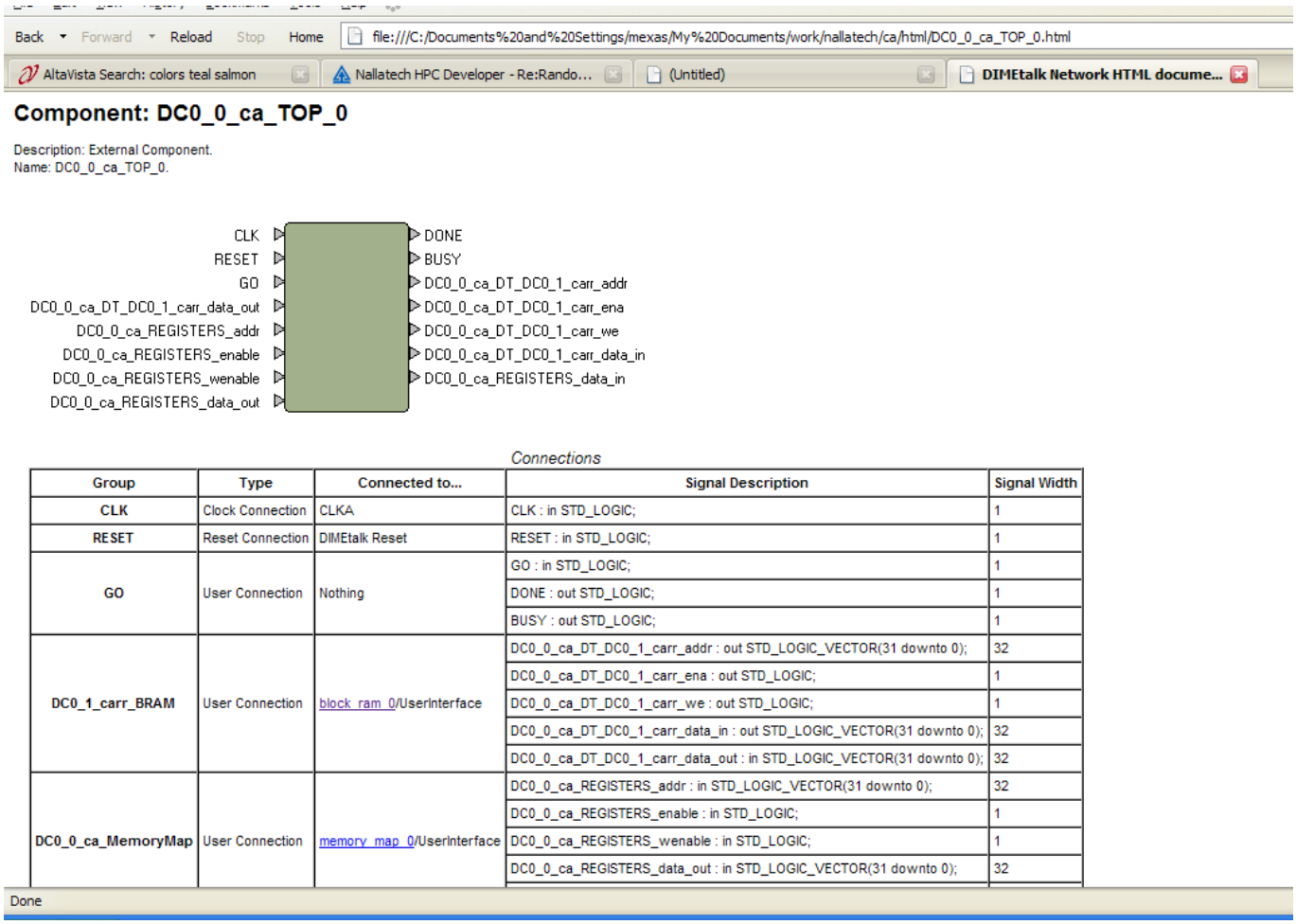


Figure 7: DIMETalk screenshot showing the network.

- Internal block RAM to store CA arrays.
- All components are placed on the H101 device.
- 2 nodes: block RAM and Memory Map.
- The Memory Map is required to start the code.
- Clock, Edge and Router are 3 other required components.

10 DIMETalk – CA user-defined VHDL signals



Group	Type	Connected to...	Signal Description	Signal Width
CLK	Clock Connection	CLKA	CLK : in STD_LOGIC;	1
RESET	Reset Connection	DIMEtalk Reset	RESET : in STD_LOGIC;	1
GO	User Connection	Nothing	GO : in STD_LOGIC;	1
			DONE : out STD_LOGIC;	1
			BUSY : out STD_LOGIC;	1
DC0_1_carr_BRAM	User Connection	block_ram_0/UserInterface	DC0_0_ca_DT_DC0_1_carr_addr : out STD_LOGIC_VECTOR(31 downto 0);	32
			DC0_0_ca_DT_DC0_1_carr_ena : out STD_LOGIC;	1
			DC0_0_ca_DT_DC0_1_carr_we : out STD_LOGIC;	1
			DC0_0_ca_DT_DC0_1_carr_data_in : out STD_LOGIC_VECTOR(31 downto 0);	32
			DC0_0_ca_DT_DC0_1_carr_data_out : in STD_LOGIC_VECTOR(31 downto 0);	32
DC0_0_ca_MemoryMap	User Connection	memory_map_0/UserInterface	DC0_0_ca_REGISTERS_addr : in STD_LOGIC_VECTOR(31 downto 0);	32
			DC0_0_ca_REGISTERS_enable : in STD_LOGIC;	1
			DC0_0_ca_REGISTERS_wenable : in STD_LOGIC;	1
			DC0_0_ca_REGISTERS_data_out : in STD_LOGIC_VECTOR(31 downto 0);	32

Figure 8: DIMETalk: CA user-defined VHDL signals.

- Each DIME-C array must have its own RAM bank.
- Signal group DC0_1_carr_BRAM is for communication with RAM. 1 is the first data array and, accordingly, the first RAM bank. `carr` is the only data array.
- Signal group DC0_0_ca_MemoryMap is for communication with the Memory Map node.

11 Optional: DIMECheck – FPGA network test

As a part of network build a `dimetest.wish` script is generated. It calls DIMECheck - network interactive diagnostics tool.

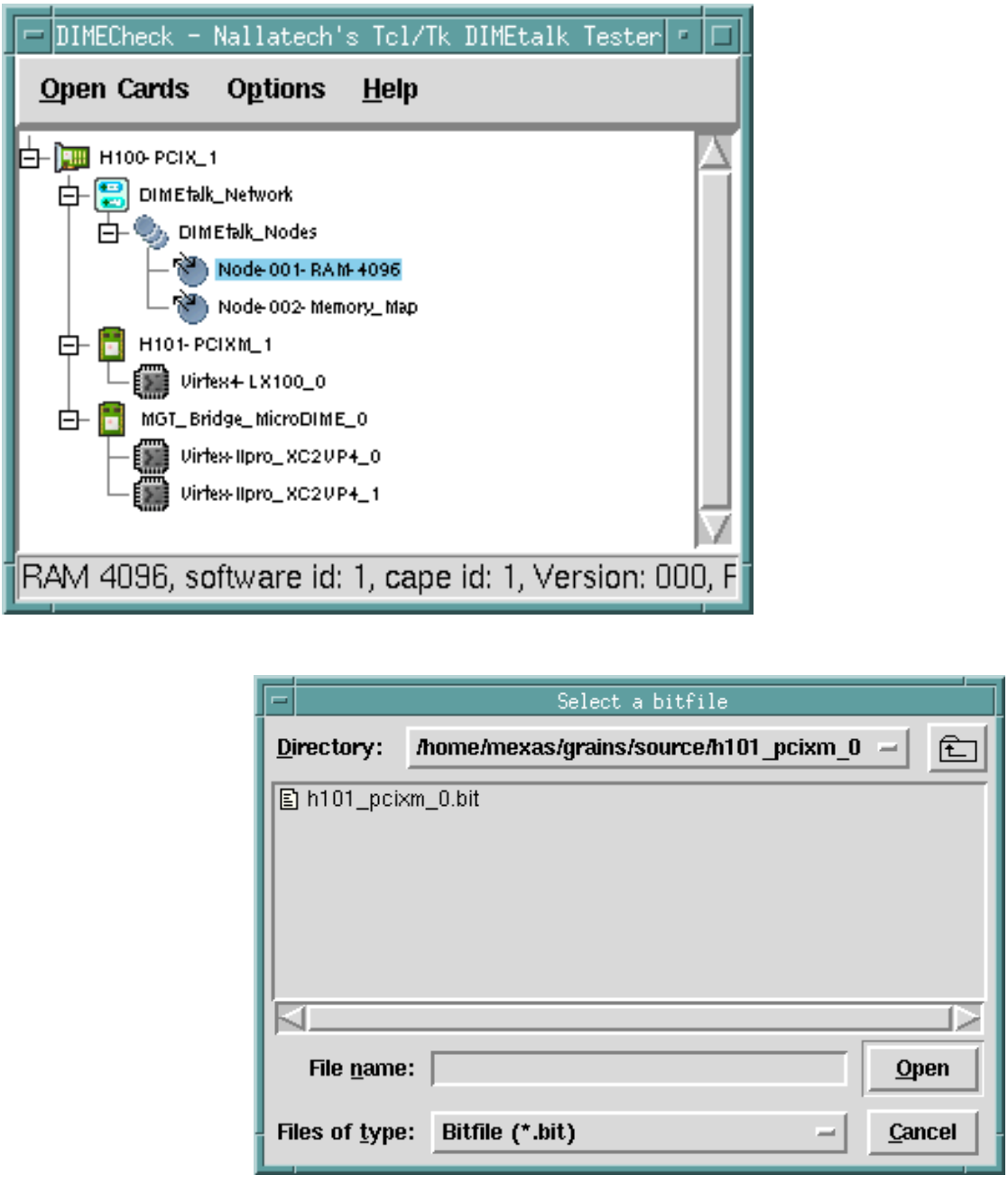


Figure 9: DIMECheck: loading the bitfile onto H101.

- In this network BlockRAM is limited to $2^{12} = 4096$ int words.
- The network can be triggered interactively from DIMECheck.

12 DIMETalk API calls - add to main c code on host

```
259 void cahost(int *A,int *B, DIME_HANDLE hTalk)
260 {
261   DWORD Config;
262   DWORD timeout = 1000;
263   int address0 = 0;
264   int ONEWORD = 1;
265
266   //Write the array to the FPGA memory node
267  DIMETalk_Write(hTalk,(DWORD*)A,(DWORD)sizem,address0,block_ram_0,timeout);
268
269   //Read the value of the 'GO' bit in the memorymap and toggle
270  DIMETalk_Read(hTalk,&Config,ONEWORD,address0,memory_map_0,timeout);
271   if (Config&0x1)
272     Config = 0;
273   else
274     Config = 1;
275
276  //Write the toggled value of Control to kick start the DIMEc process
277  DIMETalk_Write(hTalk,&Config,ONEWORD,address0,memory_map_0,timeout);
278
279         getchar();
280
281   //Keep reading whilst busy
282   while ( (Config&0x4) != 0 ) {
283    DIMETalk_Read(hTalk,&Config,ONEWORD,address0,memory_map_0,timeout);
284   }
285   //Read the results back
286  DIMETalk_Read(hTalk,(DWORD*)B,(DWORD)sizem,address0,block_ram_0,timeout);
287 }
```

13 DIMETalk API calls explained

Line 267: write array to FPGA memory node starting from 0. The memory node is Block RAM (`block_ram_0`).

Lines 270-274: read the zero ('GO') bit of Memory Map address 0 and toggle it. Togging the GO bit starts the user code.

Line 277: write the toggled GO bit to Memory Map - start the code.

Lines 282-283: bit 4 of Memory Map address 0 is 'BUSY' read it until it is zero.

Line 286: when the FPGA code is complete read the array back from the Block RAM.

14 FPGA result 1 – Block RAM smaller than array

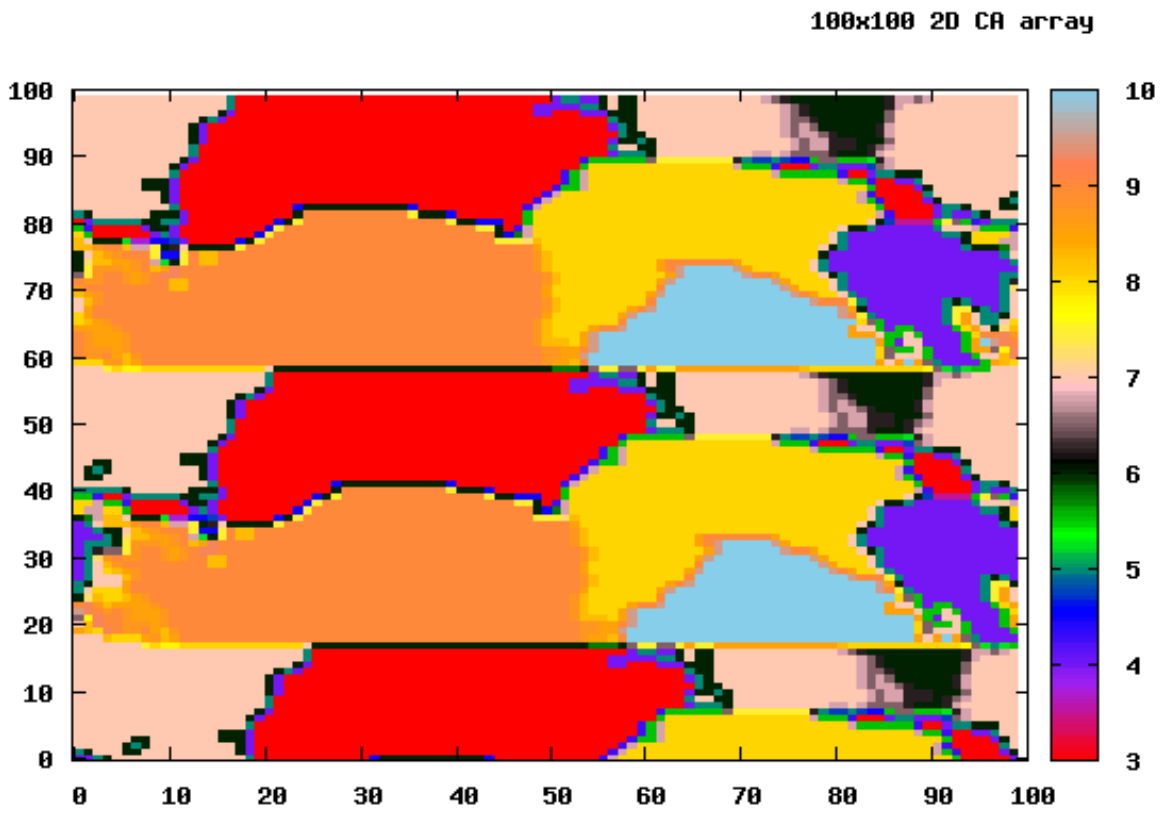


Figure 10: 100x100 cell CA array with 10 grains simulated with H101.

In this example Block RAM was 4096 int words, therefore when the data is written back from FPGA, extra array elements are repeated.

Something is also wrong with the random numbers – no grains 1 or 2.

15 FPGA result 2 – array fitting inside Block RAM

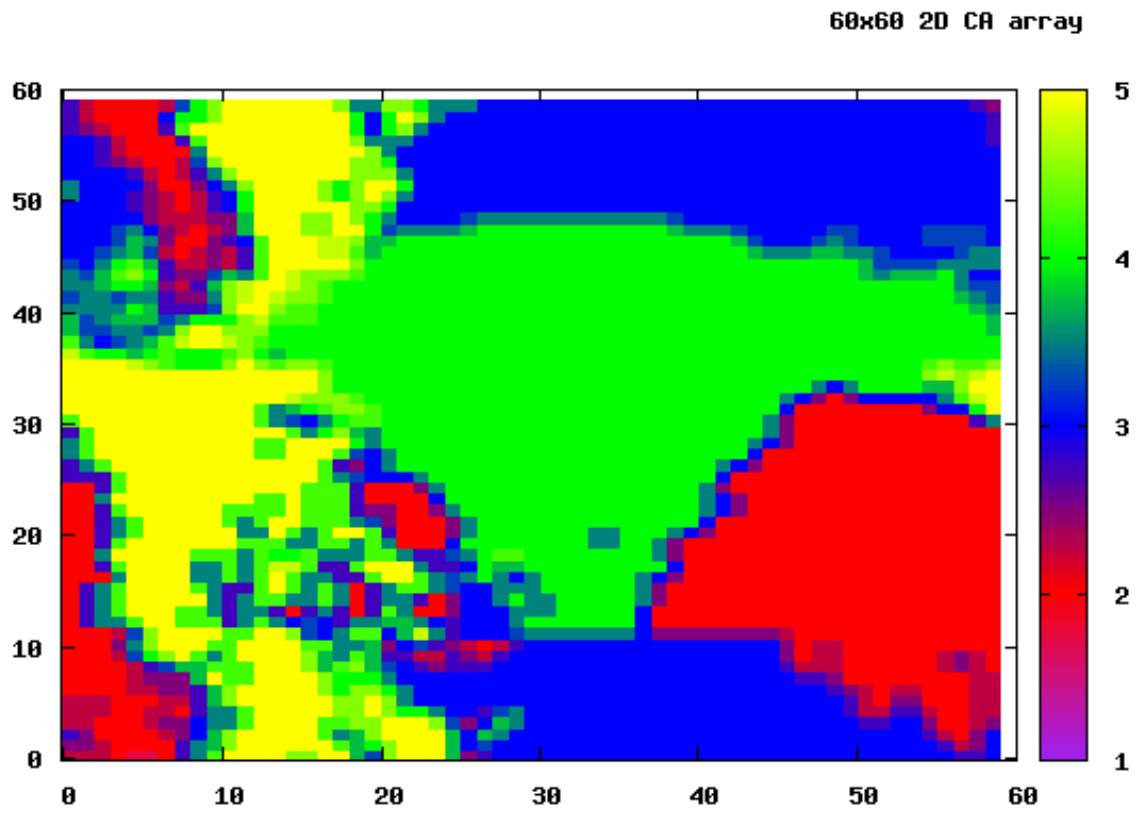


Figure 11: 60x60 cell CA array with 5 grains simulated with H101.

In this example data fits within Block RAM, so no repetition is present.

Still grain 1 is only one cell – bugs.

16 Grumble

- DIME-C and DIMETalk are only available for MS Windows. Even worse, full administrative privileges must be used to run each program!
- DIME-C manual is out of date. Many important features are not documented.
- DIMETalk build of CA takes about 30 min on dual core AMD64 laptop with 3GB RAM. Any change in user code means network rebuild – slow debugging and code development.

17 Future

- Try SRAM and SDRAM for bigger models.
- H101 speed-up measurements
- Floating point code on H101, e.g. numerical solution of a system of PDEs.

18 Conclusions

- Steep learning curve, complex development – bad.
- Fortran codes must be translated to C – lots of manual work – bad.
- Most algorithms will have to be rethought due to the constraints of DIME-C, e.g. no multi-dimensional arrays – bad.
- A completed FPGA network is used simply with FUSE API calls. Can be called from any C code – good.
- `hpc-nallatech.com` user and developer forum – all Nallatech users welcome!

19 Acknowledgements

The author gratefully acknowledges financial support from The Royal Society in form of the Research Grant. He would also like to acknowledge help and advice he received from Nallatech engineers Daniel Denning, Robin Bruce and Rich Deiner.

References

[1] A. Shterenlikht and I. C. Howard. The CAFE model of fracture – application to a TMCR steel. *Fatigue and Fracture of Engineering Materials and Structures*, 29(9-10):770–787, 2006.

[2] Nallatech. *H101-PCIX Reference Guide, Issue 1*, 2007.

[3] Nallatech. *FUSE C/C++ API Developer's Guide, Issue 11*, 2007.

[4] Nallatech. *FUSE System Software User Guide, Issue 8*, 2007.

[5] Nallatech. *DIME-C User Guide, Issue 1*, 2005.

[6] Nallatech. *DIMEtalk 3.1 User Guide, Issue 4.1*, 2007.