Raj kumar Nagarajan
rknagara@cs.ncsu.edu

Report: Jan 08 – Jan 15' 2003

Tasks Underway:

Understanding basic block profiling (Edge profiling) implementation

Read the paper
Thomas Ball and James R. Larus, "Optimally Profiling and Tracing Programs" ACM Transactions on Programming Languages and Systems (TOPLAS), vol. 16, no. 4, July 1994, pp. 1319-1360.

The paper describes the approach to perform program profiling and instruction tracing using edge profiling. The algorithm inserts monitoring code along the edges of the CFG to reduce the overhead of profiling and tracing. Advantage over vertex-profiling:

- simple and efficient
- more information than vertex profiling
- lower overhead, instruction and execution time.

Issues and To-Do:
The o/p from program profiling and tracing is to be formatted as gmon.out format, which can be read by gprof tool.
Would it be possible to write it as gmon.out format?

Study the feasibility and implementation of the approach on Microblaze

Currently the EDK provides call-graph and histogram profiling information. The o/p is in gmon.out format, which can be read by mb-gprof (gprof for microblaze). The following steps are followed:

1. Build the profile library, option to EDK.
2. Compile the program with –pg option and link it with the profile library.
3. Run the program using XMD. The profile information is collected by XMD and it generated the gmon.out file.
4. mb-gprof can be used to read the gmon.out file.

gmon.out can also be generated by collecting information from running the program in modelsim.

Issues for implementing the new method:

1. Code size – What would be the increase in code size of the application after compilation (modified application code).

   Memory sizes in different chips and reference boards:
   BRAMs – Block RAM on FPGA

Spartan II	*4k*
Spartan IIE	*8k*

Virtex/VirtexE 16k
Virtex II/Virtex II Pro 64k
Memec Microblaze Boards
SDRAM 4M – 32M
P160 – Flash 8M
P160 – SRAM 1M

** P160 module is an attachment to most of the Memec reference boards, which apart from the Memory also has Ethernet Phy., UART, RS-232, etc.*
Microblaze Demo Board (The board with me)
*5 * 1M – ZBT RAM*

Memory access cycles for Memory.
LMB BRAM 1-2 cycles
OPB BRAM 3-4 cycles
*External Memory 10 + cycles. ****

2. Execution time – What would be the increase in the execution time, because of profiling and tracing activity. And also would it (intrusive method) affect the semantics of the program being executed. E.g. Would it cause the program to miss its deadline, etc..
   What example programs do we plan to use for testing ?
3. How is the trace information going to be represented (memory limitations and also timing constraints) and how would it be sent to XMD or other tool to generate the gmon.out format ? Only way of communicating with host machine is through UART or JTAG, which can be very slow if large amount of data needs to be sent across.

Alternatives:
Generate the profiling and tracing information from the list of executed PCs. The PCs can be got using 2 methods:

1. Running modelsim and get the program trace. This information is already available and used to generate the gmon.out file. But the execution time is very slow due to simulation. Interrupts and other real-time programs may not be work in this method !! (don't know)
2. Have a hardware tracing peripheral. This peripheral can list the executed PCs of the program. Non-intrusive method of collecting information, but information maybe humongous.

Plan for next week:

1. Think, Read and Discuss on the following topic.
2. Long term goal of my work, write an abstract of the work that needs to be done.
3. Identify what this profiled information can be used for ?
4. More related work in this area.