5th International Digital Curation Conference

December 2009

Data Management of nanometre-Scale CMOS Device Simulations

Clive Davenhall,

National e-Science Centre, University of Edinburgh

Bruno Harbulot, Michael Jones

Research Computing Services, University of Manchester

Gordon Stewart, Richard Sinnott,

National e-Science Centre, University of Glasgow

Asen Asenov, Campbell Millar, Gareth Roy, David Reid,

Department of Electronics and Electrical Engineering, University of Glasgow

August 2009

Abstract

In this paper we discuss the problems arising in managing and curating the data generated by simulations of nanometre-scale CMOS (Complementary Metal–Oxide Semiconductor) transistors, circuits and systems and describe the software and operational techniques we have adopted to address them. Such simulations pose a number of challenges including, *inter alia*, multi-TByte data volumes, complex datasets with complex inter-relations between datasets, multi-institutional collaborations including multiple specialisms and a mixture of academic and industrial partners, and demanding security requirements driven by commercial imperatives. This work was undertaken as part of the NanoCMOS project. However, the problems, solutions and experience seem likely to be of wider relevance, both within the CMOS design community and more generally in other disciplines.

The 5th International Digital Curation Conference taking place in London, England over 2-4 December 2009 will address the theme *Moving to Multi-Scale Science: Managing Complexity and Diversity.* Please ensure you use guidance in this template to produce your paper. Please submit your paper in one of the following formats: Microsoft Word (.doc), Open Office (.odt), or Rich Text (.rtf).

Introduction

We discuss the problems arising in managing and curating the data generated by simulations of nanometre-scale CMOS (Complementary Metal–Oxide Semiconductor) transistors, circuits and systems and describe the software and operational techniques we have adopted to address them. It seems likely that both the problems and our emerging solutions will have wider relevance, both within the CMOS design community and more generally in other disciplines. The work presented here was undertaken as part of the EPSRC pilot project *Meeting the design challenges of nano-CMOS Electronics* (NanoCMOS[1]; see Sinnott *et al*., 2006; Sinnott *et al*., 2007; Reid *et al*., 2009). NanoCMOS is a four year project which runs from 2006 until September 2010. It is a collaboration involving 5 different specialist electronic engineering groups from various disciplines, three eScience groups and several industrial partners.

The following section provides the necessary background by first adumbrating the scientific problems that NanoCMOS is addressing and then outlining the computational challenges which this problem entails. Subsequent sections describe the e-Science infrastructure that we have developed to address the problem, with particular emphasis on the data management system. Finally we outline our progress and plans, discuss the lessons learned and consider their wider applicability.

Simulations of nanometre-Scale CMOS Devices

The scientific problem

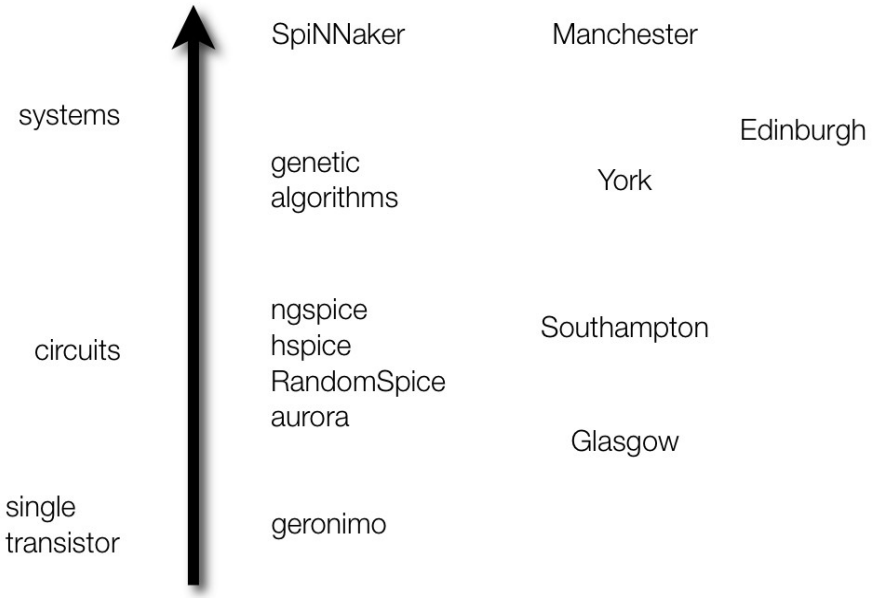
It is a commonplace that smaller and more capable CMOS devices become available as time progresses (and consequently computers become more powerful). This circumstance is captured in 'Moore's Law' which has applied loosely since the integrated circuit was invented in 1958. The 'happy scaling' encapsulated in Moore's Law has led to the enormous success of the computer industry, but it may be coming to an end. CMOS devices consisting of individual transistors 40 nm across are already in mass-production. Transistors smaller than 10 nm are anticipated by 2018 (see, for example, the International Technology Roadmap for Semiconductors, ITRS[2]). A single silicon atom is approximately 0.1 nm in diameter and thus 10 nm is equivalent to a line of about a hundred silicon atoms. For comparison the wavelength of visible light is about 500 nm. Unsurprisingly, there are major challenges in further reducing the transistor size.

Historically circuit and system design has treated the individual transistors comprising a device as being uniform and similar. As the size of the transistors shrinks this circumstance no longer holds; the variability between transistors increases with decreasing size. This effect has several causes, some fundamental to the quantisation

[1]See URL: http://www.nanocmos.ac.uk/
[2]See URL: http://www.itrs.net

of charge and matter, though the details are not germane here. These effects are stochastic and cannot be mediated by better manufacturing process control.

Though this variability between transistors is inherent it can be characterised. Circuits and systems can then be designed in a way that accommodates it.To perform this characterisation the NanoCMOS engineers use a layered suite of application programs (Figure 1). The arrow in the diagram represents increasing device complexity. At the bottom are applications to model individual transistors and characterise their range of properties. Applications in the higher layers are concerned with designing individual circuits and then agglomerations of circuits into systems. Typically each application will use data from lower applications. Figure 1 encompasses a considerable range of complexity; large, modern VLSI (Very Large Scale Integration) systems may contain several thousand million transistors.



[Caption] Figure 1. The increase in complexity in CMOS devices from single transistors to circuits and then systems. The middle column lists relevant simulation programmes and systems. The NanoCMOS group working in each area is shown on the right.

The computational problem

Addressing the above scientific problem involves the following computational challenges.

Computational cost and complexity. Simulating an individual transistor once is costly and requires considerable processing time, of the order of 100 CPU hours. In the case of nanometre-scale transistors each one must be simulated many times to characterise the range of properties empirically. For a small ensemble of 200 devices the total time is approximately in the range 20,000 CPU hours. This ensemble is

required because the resulting distribution of threshold voltage, for example, is not Gaussian (briefly, it is similar to a Gaussian but with the addition of skew). Similarly circuits must be simulated many times to characterise their range of properties, selecting random values from the permitted distribution of individual transistor values. Consequently a great deal of processing time is required. However, there is often a coarse-grained parallelism inherent in the problem, which is well-matched to super-computing clusters comprising many independent nodes.

Large volumes of data. The simulations often generate large volumes of data. In some cases the entire dataset does not need to be kept permanently, but rather a pertinent sub-set can be extracted and retained. For example in the above set of 200 devices one of the stored parameters, the mobility, leads to 168 Gbyte of generated data after compression. This quantity can be multiplied many times for the various physical properties under investigation. To fully characterise all of the parameters of an individual transistor will take ensembles of up to 100,000 devices. For use in circuit design this process must be performed twice (for both n- and p- channel devices). In total the simulations generated during the NanoCMOS project will occupy multiple TBytes of disk space.

Distributed access. Groups of users in different institutions distributed around the country will typically need access to the data.

Absence of standards for data and metadata. Unlike some other branches of science and engineering, semiconductor electronics has no standards for data and metadata to facilitate communicating information between applications. Such standardisation as exists is *de facto* in the form of the format of the input files required by many of the applications.

Tracking the work done. Within NanoCMOS a large number of simulations will be performed. In many cases each simulation will involve many invocations of an executable program on a node of a super-computer. Each such invocation will generate its own, potentially large, output files. However, often the simulation will ultimately be characterised by a small amount of critically relevant data which will be used by one or more higher-level simulations. It is impossible to track all this information by hand. Rather, what is required is a metadata handling system that records the details of individual simulations: their input parameters, the files generated, and the relations between them: which group of simulations they are part of, which lower-level simulations they use *etc*.

Security. Most of the groups work with industrial partners on designs that are commercially sensitive. For these firms the designs are intellectual property that is the core of their business. The firms usually collaborate with single groups, or individuals in groups, not the entire multi-organisation academic consortium. Thus, access to defined sets of results must be restricted to groups of individuals within the consortium, and these groups can be cross-institutional. Restricted access is required not just to the bulk data but also to the metadata.

Previous practice

Prior to NanoCMOS individual groups worked predominantly on separate areas (corresponding to layers in Figure 1). There was no automatic transfer of data from one application to another. Rather, it was done 'by hand', for example: by editing files into new formats, via the published literature or using information from manufacturers' data sheets *etc*. A user would run jobs on his own desktop machine or on departmental clusters, which tend to be small.

The bottom layer, simulating individual transistors, is only done by the Glasgow device modelling group who are recognised international experts in this area. They use a suite of extremely complex programs, written in-house in Fortran 77/90 and developed over many years. In a sense this code encapsulates the consensus understanding within the group of the underlying physics of transistors. Higher up the stack 'off-the-shelf' public-domain or commercial applications are usually used. Sometimes there are several implementations of the same program; the circuit simulation program SPICE[3] is a good example, with several commercial and open source versions available. As mentioned, inter-operation has been predominantly through *ad hoc* file formats and, for example, there are subtle but important differences in file format between the various versions of SPICE.

The NanoCMOS eScience Infrastructure

Following the discussion in the preceding section, previous practice was not suitable for simulating devices and circuits containing nanometre-scale transistors with a range of properties, because an ensemble of simulations must be run to establish the range of variability relevant to large-scale VLSI circuit design. In order to address this computational problem the simulations need to be run on large super-computer clusters remote from the users, such as the NGS[4] (National Grid Service) and ScotGrid[5] systems. Further, a data management system is required to keep track of the simulations performed and the data generated.

NanoCMOS has developed an e-Science infrastructure to support this new methodology. The data management system is the focus of this paper and is described in the next section. The job submission and data storage systems are discussed rather more briefly immediately below. The security infrastructure is described by Sinnott *et al*. (2008).

Job Submission System

In NanoCMOS simulations are typically generated by jobs running on shared, public super-computing clusters. The details of submitting and tracking jobs, and

[3]See, for example, URL: http://bwrc.eecs.berkeley.edu/Classes/icbook/SPICE/
[4]See URL: http://www.grid-support.ac.uk/
[5]See URL: http://www.scotgrid.ac.uk/

subsequently retrieving the results, all vary between systems. Users, conversely, may wish to run simulations on a variety of resources, making use of a variety of systems and exploiting the currently least-busy.

Consequently we have developed a standard job submission 'harness' which allows a user to submit a simulation to a system of his choice. The harness wraps the underlying application performing the simulation and makes the necessary submission while hiding the details of the target resource. It is relatively straightforward to extend the harness to wrap an additional application or to submit to a new resource.

The technicalities of the submission harness are beyond the scope of this paper, but briefly, it is implemented as a series of Web services hosted on a central server. The user communicates with these services by running a client on his local machine. This client is all he must install locally, thus avoiding having to install the often difficult and arcane job submission software for the various resources.

The data file which defines the simulation to be performed and the resource where it is to run is formatted as an XML file. Internally the submission harness uses standards where possible. For example jobs are described using JSDL (Job Submission Description Language; Anjomshoaa *et al.*, 2005; Humphrey *et al.*, 2007; Savva 2008).

Finally, to relieve the user from having to retrieve the simulations generated from the remote cluster they are automatically deposited in a secure virtual file system, as described below. Where practical the cluster is configured to write the results directly into this file system to avoid unnecessary transfers.

Data Storage System

The storage requirements for NanoCMOS are secure, distributed access to a large volume of data, with fine-grained control over who is allowed access to which dataset.

The project adopted the Andrew File System (AFS; Howard *et al,* 1988) as a good match to these requirements. AFS is a distributed file system originally developed in the 1980s. Several implementations of AFS have been produced. We chose OpenAFS[6], a mature public-domain version which is the only one now in widespread use.

Files in an AFS system appear to the user and behave as though they were local files. For example, in a Unix system AFS files appear in the same directory hierarchy as other files, but descending from an AFS root directory (usually /afs). Files can be copied between the AFS and local directories just like copying between two local directories. For a well-configured system writing files to a local AFS directory is not significantly less efficient than writing files to a conventional local file system. Access control and file synchronisation are handled transparently.

AFS has a fine-grained file access permission system based on access control lists of individuals and groups which allows access to individual directories to be controlled. This system provides precisely the fine-grained access required for NanoCMOS. Users are identified or authenticated to AFS using the well-known

[6]See URL: http://www.openafs.org/

Kerberos[7] protocol. We use version V of the protocol.

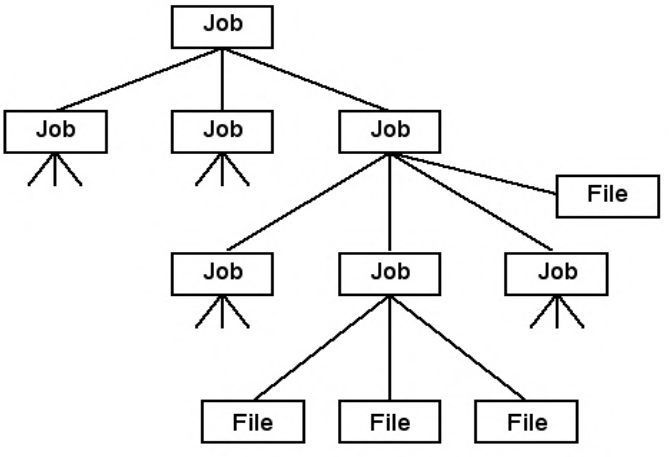
Authentication with Kerberos usually involves the user entering a password. This approach is satisfactory for interactive use, but unsuitable for remote or autonomous operation, when no user is present. For this latter case we have adopted the gssklog utility which uses GSS (Generic Security Services) to perform Kerberos authentication from a proxy X.509 certificate[8], thus avoiding the need to enter a password.

It is clearly important that data stored in the AFS partition are regularly backed up as a hedge against misadventure. The current arrangements are as follows. The AFS partition is located on two Sun Fire X4500 servers. These systems have 48 disk drives configured in a RAID 5+0 arrangement. This system provides redundancy against drive failure. In addition snapshot copies of the partition are taken automatically every night. These arrangements are adequate, but are currently being improved by adding support for secure off-site copies.

The Data Management System

The purpose of the NanoCMOS data management system (DMS) is to store and manage the metadata which describes the execution of simulations and their output. It allows details of the simulations performed, and crucially their inter-relations and inter-dependencies, to be captured and preserved. Subsequently the results and details of simulations can be queried, retrieved and re-used, for example in publications or subsequent higher-level simulations.

The basic entity of the DMS is a unit of identifiable computational work: a 'job'. A DMS job is a flexible and abstract concept: a job can contain an arbitrary list of sub-jobs which are themselves jobs. Jobs can also have files, either input or output, associated with them. This hierarchical relationship is illustrated schematically in Figure 2. In the DMS jobs are represented by job records and files by file records. Two examples might illustrate the structure.



[Caption] Figure 2. The hierarchy of files and jobs in the DMS.

[7]See URL: http://web.mit.edu/Kerberos/

[8]See, for example, URL: http://www.itu.int/rec/T-REC-X.509/en

Firstly, at the bottom of the device complexity hierarchy, a job record may correspond to a set of simulations of individual transistors, for example based on a particular semiconductor technology or a particular manfacture's design of devices. This job record might have as its sub-job records simulations for individual transistors in the set. The DMS job record for each transistor might correspond to an array job submitted to a super-computer cluster, with each member of the array job causing the execution of a program on a node of the cluster. Each such execution would then become a DMS sub-job record of the DMS job record corresponding to the array job.

Secondly, a higher-level simulation, perhaps for simulating an individual circuit, would include in its metadata links to the files and transistor simulations from which it was obtaining its input data. Thus, it is possible to track the chain of assumptions and input values used to generate the higher level simulation. This facility is not exactly tracking provenance, but it is closely akin to it.

All jobs and files have the same structure. They consist of a single core-data item and an optional list of annotations. The core-data is mandatory and contains a fixed set of items (slightly different for jobs and files; see Table 1) which give basic information about the item (when it was created *etc*). For a file record this information includes the AFS file name of the corresponding file. This name is unique within the AFS installation and so can function as a URI. The annotations are optional and an arbitrary number may be present. Each annotation can contain an arbitrary list of metadata items (though in practice the list is likely to be similar for a given type of job or file).

Job Record		**File Record**	
Item	Occurs	Item	Occurs
owner	1	owner	1
jobCreationTime	1	location	1
jobStartTime	1	type	1
jobEndTime	1	jobRecord	1
parentJob	0-1		
childJob	0-n		
fileRecord	0-n		

[Caption] Table 1. Core-data items for job and file records. The names of the items are self explanatory.

The DMS is designed for geographically dispersed users who are remote from the computer hosting it. It has a RESTful interface, that is, one which conforms to the REST[9] (Representational State Transfer; see for example Fielding, 2000; Fielding & Taylor 2002) principles. In practice this approach means that every item in the system, that is every job or file, has an identifier; a unique name by which it may be referenced

[9]See, for example, URL: http://www.xfront.com/REST-Web-Services.html

and displayed. Similarly there are temporary identifiers for sets of search results and standard identifiers for lists of all the jobs and files and for the function to specify search criteria. The identifiers used are standard URIs, in order to facilitate access using the standard Web protocols, notably HTTP.

The DMS is written in Java and the RESTful interface is implemented using the RESTLET[10] framework. The core-data and annotations constitute 'resources' within the REST framework and may be returned using a XML, HTML or JSON (JavaScript Object Notation) representation. The job and file details are stored in an SQL relational database which is accessed using the Hibernate[11] object-relational persistence framework. Hiberate supports various back-end databases and we are variously using ORACLE[12] and PostgresSQL[13]. The user interface is implemented using GWT[14] (Google Web Toolkit) and is accessed using a standard Web browser.

The project currently supports two primary DMS installations. One of these is for public NanoCMOS data and is deployed and hosted on the NGS. The other is hosted in Glasgow and is primarily used for more secure datasets. We are currently exploring installing further DMS instances at partner sites. Users select the DMS that they wish to use as part of the process for job creation and submission.

Progress and Plans

The NanoCMOS project is still in progress and development of the e-Science infrastructure continues. The basic system is in place, notably the harness for submitting jobs to super-computing clusters and the DMS.

When developing and wrapping applications in the stack (Figure 1) work necessarily started at the bottom and worked up, since the higher levels require data from the lower ones. To date two applications have been targeted and made available through the infrastructure: Geronimo, at the lower level of the stack, which simulates individual transistors and RandomSpice which makes multiple invocations of the SPICE circuit simulator to determine the range of properties of individual circuits by randomly selecting constituent transistors with a range of properties.

Future work will address applications higher up the stack. However, simulating individual transistors and circuits are autonomous operations that, once their input parameters are defined, can proceed from start to finish without further human intervention. Thus they are amenable to automation and remote submission to super-computer clusters. Conversely, applications higher up the stack tend to be more interactive. Adapting them seems likely to involve significant challenges.

We will continue to enhance the DMS, in particular improving both the features

[10]See URL: http://www.restlet.org/
[11]See URL: https://www.hibernate.org/
[12]See URL: http://www.oracle.com/
[13]See URL: http://www.postgresql.org/
[14]See URL: http://code.google.com/webtoolkit/

for searching the database and the user interface. Also, further work is required on some of the security functions.

Effort to date has necessarily concentrated on developing the basic infrastructure. We also need to ensure that the simulations generated are properly curated. In this context the particulars of the NanoCMOS project are pertinent: though we have a large volume of complex scientific data we also have a small and specialised user community. Further, the simulations are not required indefinitely in their current format (the semiconductor industry evolves quite rapidly). Thus we are concerned with short- and medium-term, rather than long-term, curation. It would be useful to assess our current curation practices and identify any areas for improvement, perhaps using some of the techniques adopted by the DCC[15] (Digital Curation Centre), such as the DAF[16] (Data Audit Framework) and DRAMBORA[17] (Digital Repository Audit Method Based on Risk Assessment).

Discussion

We have developed an infrastructure for managing the data generated by simulations of nanometre-scale CMOS devices. Enhancements to this infrastructure continue. However, the following lessons, some of which seem likely to be of wider applicability, are already apparent.

1. The job submission harness uses less standard software than originally intended. We initially attempted to use such items, but many were either found to be unsuitable or they did not scale adequately when running thousands of computationally intensive jobs.
2. Configuring the OpenAFS installation proved more troublesome than expected, due to a combination of security configuration and networking issues, though it is now working satisfactorily. We chose AFS because it offered significant performance advantages over systems such as the SRB[18] (Storage Resource Broker).
3. The DMS initially had a fixed hierarchy comprising: experiments, tasks, jobs and files. However, this arrangement was found to be insufficiently flexible to accurately represent the work undertaken and was replaced with the flexible scheme described above.
4. The motivation for the REST architecture is that it captures or mimics the way that the Web protocols behave. Thus, applications designed with a RESTful interface present a clean and intuitive Web interface. Our experience confirms this supposition.
5. Some users are, understandably, reluctant to adopt the new way of

[15]See URL: http://www.dcc.ac.uk/
[16]See URL: http://www.data-audit.eu/DAF_Methodology.pdf
[17]See URL: http://www.dcc.ac.uk/FAQs/self-auditing/
[18]See URL: http://www.sdsc.edu/srb/index.php/Main_Page

working. It is worth investing effort to ensure that the new infrastructure is straightforward to use. However, once users have adopted the new system they usually soon appreciate its power and flexibility.

6. Even in the relatively small NanoCMOS community users vary enormously in their level of computing expertise. Also different groups have noticeably different practices. Particularly towards the lower end of the complexity stack there was little demand for a GUI or Web portal for job submission. Users were perfectly happy submitting jobs from their Unix command line, provided that the procedure was straightforward and adequately documented.

7. Finally, users do not know *a priori* what metadata needs to be included in their annotations, nor what queries they will want to make on the metadata. The process is new to them. An iterative approach must be adopted, with the metadata stored and queries supported being adapted to the users' changing requirements as they develop.

The NanoCMOS infrastructure addresses the significant challenge of managing multi-TByte volumes of specialised, complex scientific data which require distributed access for users in multiple institutions in the presence of demanding security requirements. The data are accessed by a stack of specialised applications which simulate CMOS devices on a wide range of scales, from single transistors to systems with many millions of transistors. Finally, the DMS, in particular, contains little that is specific to NanoCMOS and we are actively pursuing applying it to manage other large, complex datasets.

Acknowledgements

The NanoCMOS project is funded by the UK Engineering and Physical Sciences Research Council (EPSRC).

References

[report]Anjomshoaa, A., Brisard, F., Drescher, M., Fellows, D., Ly, A., McGough, S., Pulsipher, D. and Savva, A. (2005, November). *Job Submission Description Language* (JSDL), V1.0. (GFD-R.56). Open Grid Forum (http://www.ogf.org/).

[thesis]Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine.

[journalarticle]Fielding, Roy T.; Taylor, Richard N. (2002). Principled Design of the Modern Web Architecture. *ACM Transactions on Internet Technology (TOIT)*.

New York: Association for Computing Machinery, 2 (2): 115–150

[journalarticle]Howard, J.H., Kazar, M.L., Nichols, S.G., Nichols, D.A., Satyanarayanan, M., Sidebotham, R.N., & West, M.J. (1988, February). Scale and Performance in a Distributed File System. *ACM Transactions on Computer Systems*. New York: Association for Computing Machinery, 6 (1): 51-81.

[report]Humphrey, M., Smith, C., Theimer, M. and Wasson, G. (2007, August), *JSDL HPC Profile Application Extension, V1.0*. (GFD.111). Open Grid Forum.

[report]Savva, A. (2008, August). *JSDL SPMD Application Extension, V1.0*. (GFD. 115). Open Grid Forum.

[proceedings]Sinnott, R.O., Asenov, A., Berry, D., Furber, S., Millar, C., Murray, A., Pickles, S. , Roy, S., Tyrell A. & Zwolinski. M. (2006, September). *Meeting the Design Challenges of nanoCMOS Electronics: An Introduction to an EPSRC Pilot Project*. UK e-Science All Hands Meeting, Nottingham UK.

[proceedings]Sinnott, R.O., Asenov, A., Brown, A., Millar, C., Roy, G., Roy, S. Stewart, G. (2007, September) Grid Infrastructures for the Electronics Domain: Requirements and Early Prototypes from an EPSRC Pilot Project. UK e-Science All Hands Meeting, Nottingham, UK (best paper award).

[proceedings]Sinnott, R.O., Bayliss, C., Davenhall, C., Doherty, T., Harbulot, B.,Jones, M., Martin, D., Millar, C., Roy, G., Roy, S., Stewart, G., Watt,J. & Asenov. A. (2008, December). Integrating Security Solutions to Support nanoCMOS Electronics Research. IEEE International Symposium on Parallel and Distributed Processing Systems with Applications, Sydney, Australia.

[journalarticle]Reid, D., Sinnott, R.O., Millar, C., Roy, G., Roy, S., Stewart, Gordon, Stewart, Graeme & Asenov, A. (2009, July). Enabling Cutting-edge Semiconductor Simulation through Grid Technology. *Journal of the Philosophical Transactions of the Royal Society A*. London, 367:2573-2584.