i. 1

ARCHIVE OF DOCUMENTS AND REFERENCE MATERIALS REGARDING THE IBM ACS-1 MACHINE

Lynn Conway*
February 16, 1999

This volume contains documents and reference materials that I have compiled regarding the IBM Advanced Computing Systems ACS-1 supercomputer. These are copies of original documents dating back to the ACS project itself. Taken together, they may be sufficient to disclose many of the system architectural innovations of the ACS architecture team.

The front-matter for the archive contains a brief, but important overview, of each document, including some details regarding the document's context within the ACS project. Also included is my initial letter to Dr. Mark Smotherman of Clemson University regarding the possibilities of reconstruction of many details of the ACS-1 machine.

CONTENTS: **PAGE:**

* My name was legally changed to Lynn Conway on January 30, 1969. Since I am widely known under my new name, we've chosen to use it on my earlier papers in this archive.

1. "Dynamic Instruction Scheduling", February 23, 1966:
L. Conway, B. Randell, D. Rozenberg, D. Senzig

The background on this paper is as follows. Sometime in late '65, I suddenly visualized a solution to the general multi-issuance and conflict-resolution problem. I quickly compiled block diagrams and notes to capture the ideas, and during the next few days I presented these ideas in staff meetings in the architecture group. There was a rapid, very positive reaction. I was tasked to document the ideas in more detail, to incorporate one of the branching schemes then under study, and to turn the scheme into an architectural "proposal".

Since I was quite junior and had little experience with coordinating and writing ACS proposals, I worked with a number of ACS staff members, including Don Rozenberg, Brian Randell, Don Senzig and others to produce the resulting paper. There was a sense that these weren't just ordinary ideas, and we worked hard to frame the concepts in a tutorial form, so that they would be clear to team members. Brian Randell in particular came up with some wonderful articulations about the DIS schemes, in his inimitable British manner. We hoped to be able to publish the ideas openly later on.

But things then moved fast, and within a year the ideas in the paper had became the basis for, and were implemented within, a fully revised ACS-MPM architecture.

Although the original dynamic instruction scheduling ideas were mine alone, the paper was a team effort. As inventor, I was the lead author, and was followed by Brian Randell, Don Rozenberg and Don Senzig. I think Ed Sussenguth and Herb Schorr gave useful feedback too; had the paper gone on to publication they might have been included as co-authors.

The dynamic instruction scheduling paper is labeled "[DRAFT]". I believe that by late February '66, we saw this paper as a work in progress towards formal publication. The ideas were already, in parallel, being evaluated for use in the actual machine. Thus in this draft I think we stepped back from revealing thinking on exactly how the ideas might be applied in the machine, as, for example, by using dual instruction windows.

But by then we also needed a tutorial on the ideas for those outside the architecture group, such as the logic designers, to use as a reference. Thus this "draft" version of 2-23-67 was released within ACS. After that date, no further work was done on the paper. It was completely overtaken by the escalating events surrounding adoption of this scheme for use in the ACS machine. Thus the invention itself then became quite "secret".

Interestingly, the name "dynamic instruction scheduling" never really entered into the team's "lingo". Instead, the relevant structures were usually just called "instruction queues", or "instruction buffers", or "contender stacks" for short, as is seen in all the later documents. It's possible that many ACS vets won't recall the specific title of the paper. Could that perhaps explain why no one from the team has ever come forward and mentioned this work?

On the other hand, it is very likely that copies of this paper surreptitiously passed into circulation outside IBM during the late 60's and early 70's, providing a path for transfer of this knowledge, and its name, into computer architecture circles outside of IBM.

2. "ACS Simulation Technique", Mar. 15, 1966: D. Rozenberg, L. Conway, R. Riekert

This paper documents the methods used to build the ACS MPM register-transfer level simulator. This paper may prove valuable by helping later analysts better understand and interpret the source code and the output results of the "MPM Timing Simulator".

The simulator was built in FORTRAN IV. Thus it is relatively easy to "read the code" that defines the workings of each module and functional unit. The simulation methods were also aimed at being fast enough to support long runs involving many, many variations of the machine architectural parameters.

The simulator was initially used to take quick looks at architectural variants, watch code passing through them, and figure out why things got blocked or didn't work as expected. Later it was used to gather data on the performance of many serious MPM variants running lots of real code, and then to "balance and tune" the emerging ACS-1 machine.

Notice the use of a "memory queue" function as the tutorial example in this paper. I believe that by this time in '66, we were already doing basic simulator implementations and evaluations of various "instruction queuing" structures and controls, as part of our explorations of dynamic instruction scheduling methods. I think we may have just simplified and then "reused" some of that code to create the example in this paper.

Don Rozenberg was lead author, I was second and Bob Riekert was third. Bob had done important work on the simulation methods at Yorktown, but didn't go west with ACS.

3. "Dual Arithmetic on ACS-1", May 1, 1967: T. C. Chen

This paper is an internal proposal from Tien Chi (T. C.) Chen to Jack Bertram regarding methods for implementing dual floating point arithmetic in ACS-1. It contains interesting references to dual arithmetic on the ILLIAC IV machine.

I include this paper as a good example of an ACS "proposal", though I do not recall right now the details of how this particular one turned out.

Note that the data-path register-transfer-level details of the arithmetic-functional units were an independent architectural dimension of the project that had to meet logic design/machine-cycle constraints on the one hand, and bussing/pipelining/issuance-control/architectural constraints on the other.

Thus only the timings of the ACS-1's arithmetic units, and not those units' internal functional details, were modeled in the timing simulator. (An "unroller" processed assembly code input instructions to produce the input instruction stream to the timing simulator). This was in contrast to the OP fetch, Bussing, OP interlocking and issuance, SKIP, Branch and Exit functioning, etc., which were fully modeled in the timing simulator.

4. "Architecturally Critical Paths in the MPM", May 12, 1967: E. Sussenguth

This is an important internal memo from Ed Sussenguth to Herb Schorr that summarizes the results of detailed MPM architectural design studies during the spring of 1967. It pins down the final list of critical paths that must be insured against any performance slippage in any later design iterations.

In each particular case, the critical path functions are identified as needing to be completed within a certain number of machine cycles. Then, for each of these functions, there would have been related critical logic design exposures, wherein specific logic functions had to be completable within a machine cycle .

This memo was the result of an intense period of simulation and tradeoff studies to tune and balance the MPM mechanisms for OP fetching, Bussing, OP interlocking and issuance, SKIP, Branch and EXIT mechanisms, functional unit timings, etc.

Together with the other documents, this paper shows that the near-final form of ACS-1 machine architecture was completed and was being fine-tuned during the spring of '67; thus it supports the inference that generalized dynamic instruction scheduling must have been incorporated into the revised ACS machine architecture sometime in the latter part of '66.

The details in this memo about MPM critical paths should really help during efforts at interpreting other ACS documents, and reconstructing the MPM's architecture.

5. "MPM Timing Simulation", August 25, 1967 (ACS AP #67-115) : L. Conway

This paper is a gold mine of detail on the system architecture of the ACS-1 MPM. It was originally intended as a users' manual that others could reference, in order to submit simulator input and interpret simulator output. I was sole author of this paper.

The simulator was written in FORTRAN IV (H), and ran on an IBM S/360 Mod 75 under OS/360. It operated at a rate of approximately 10 simulated instructions per second; typical programs thus ran at a rate of about 20 instructions per second.

By this date, the simulator was the de facto formal description of the structure and functions of the timing and controls of the ACS-1 MPM. All architecture team members coordinated their work with the making of modifications to the evolving versions of this simulator. Detailed functional modifications were seen to work or not, by whether they functioned as expected during simulation runs.

By the time this document was written, a lot of experience had been gained in the effects on machine performance of variations in machine parameters. In particular, it was clear by then that the 3 out of 8 issuance scheme for A-Ops was near optimal in terms of mean OPs/cycle while meeting the logic-level and machine cycle-time constraints. This paper uses that 3 out of 8 scheme in a very detailed example, including detailed timing diagrams and the corresponding simulator input and output listings.

Therefore, this paper provides a peek inside an ACS-1 MPM actually running code, enabling the reader to see how the OP fetching, Bussing, instruction scheduling, Branch and Exit functions, functional unit timings, etc., all worked together.

The paper defines and elaborates on the mnemonics of all those machine facilities, enabling readers to make detailed interpretations of timing diagrams and simulator output listings. Those mnemonics were used widely within ACS by this date, so these definitions will be helpful in interpreting other ACS documents. This paper includes a list of all instruction mnemonics, but, unfortunately, no detailed descriptions of the instructions themselves.

This manual, together with the detailed "Timing Simulator Notebook" and the "Timing Simulator Source Code Listing", provides sufficient information to possibly enable later analysts to reconstruct a running version of the ACS timing simulator.

This document, with all its details of how the ACS-1 processed instructions, may also have passed into circulation outside of IBM, and thus helped to propagate ACS architectural concepts into the computer architecture community.

6. MPM Architecture and Simulator Notebook, August 1967: L. Conway

This notebook contains my working documentation of the ACS-1 machine architecture, and materials regarding translation of that architecture into the MPM Timing Simulator. It contains very detailed information on the ACS-1 as of late August 1967, which was a mature point in the machine's evolution, and the design point for which important benchmarks have been described elsewhere. The notebook consists of about 120 pages of flowcharts, tables and notes, in addition to the ACS AP #67-115 paper.

Unfortunately, these notes do not contain a description of the OP set itself, as it was documented in a separate memo that, I believe, was entitled "ACS-1 MPM Instruction Manual" (we should really try to find a copy of that one, if one still exists). However, many important details regarding the OP set, including the OP Tags, are included in these notes. A listing of the contents of this notebook is included on the following page.

Listing of contents of the Timing Simulator Notebook (draft listing, as of 1-21-99) :

7. Timing Simulator Source Code Listings, August 1967: L. Conway

This notebook contains a set of listings of the source code for the near-final version of the ACS machine's register-transfer level timing simulator. There are about 5000 lines of FORTRAN IV (H) source code in these 100 or so pages of listings. This is probably the version of the code used to generate the examples in the ACS AP #67-115 paper.

By mid-67, the timing simulator was the de facto formal description of the overall team-coordinated details of the evolving ACS-1 architecture. Therefore, these listings, when taken together with the Timing Simulator Manual and the additional diagrams, flowcharts and other details in the Timing Simulator Notebook, provide a very detailed account of the ACS-1 system architecture.

8. "ACS Logic Design Conventions: A Guide for the Novice", Nov. 29, 67: L. Conway

On joining ACS, I found that there was no single convenient source for this information. Some of the information was not documented in any available references. Since most of the logic designers used different notations and conventions, it proved to be a time consuming and confusing process to learn the precise details of this very simple, basic material. Many of the designers related to me that they had had similar initial experiences.

At the time I made some notes for my own personal use, and later formed these notes into this memorandum in the hope that it might prove useful to newcomers to ACS. This memo may prove useful in ACS retrospectives and reconstructions by enabling more precise analysis of original ACS DRKS design records.

9. "A Proposed ACS Logic Simulation System", Oct. 31, 1967: L. Conway

This memo proposes an LSS to provide a means for debugging the logic design of the ACS machine. Included is a means to extract design partitions from DRKS files and run simulations on the partitions based on interface signals extracted from the equivalent partition of the system-level (MPM timing) simulator. Considerable detail in the form of block diagrams, flow-charts and calculations are included to clarify interfaces and interaction in the overall system. One requirement for such a system to work would be formal acceptance of the system-level simulator as the formal description of machine structure and functions, and forcing of logic design partitions to implement the functions of the equivalent system-level partitions. This seemed feasible at the time, since the MPM Timing Simulator had already become the de-facto formal description of the machine. This memo may provide useful insights into various practical aspects of ACS logic design and engineering at the time.

10. "The Computer Design Process: A Proposed Plan for ACS", Aug 6, 68: L. Conway

This memo builds on item 9, and proposes a detailed design for the overall ACS machine design process, including system architecture, logic design and engineering, physical specification and process automation, and maintenance. The thesis is that proper design of the design process is as important as proper design of the machine itself. It exploits the System-level Simulator as the overall machine specification, and discusses the overall integration and protocols for use of that simulator with the LSS, DRKS, Physical Specification and Process Automation tools. It addresses many concerns, such as the fact that design phases do not follow serially but overlap in time, that some partitions may be far along in specification while others may be quite tentative, and that later design phases constantly feedback feasibility or cost issues to earlier (higher-level) phases. This proposal was fairly widely circulated and had gained considerable support just before the project was cancelled. This memo provides useful insights into practical aspects of ACS system architecture, logic design, engineering, physical specification and process automation at the time. [Also, taken together with the other materials, all this work substantially informed my later explorations at Xerox PARC on VLSI design and implementation methodologies].