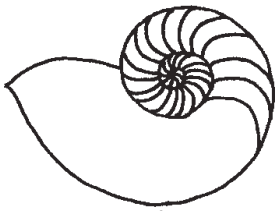
Volume 4, Number 4 February 1992 $2.50



TUUG Lines

Technical UNIX® User Group

Newsletter of the Technical UNIX® User Group

A Group, by Any Other Name, ...

One of the common misconceptions about our group is that we focus on very technical issues, rather than aiming for more novice users. Our membership is a very mixed bag, indeed, ranging from people who've worked with UNIX for over a decade to those who've only begun to struggle with it. Therefore, we try, in the newsletter and at the meetings, to balance things out, and offer something for everyone.

One of the reasons for the misconception, it turns out, is simply our name. The reason for the word "Technical" in the name is historical – it was to distinguish ourselves from another local UNIX group which dealt more with UNIX industry and market related issues than the hands-on, technical issues on which we focussed back then. Since then, our group has become the only active UNIX group in Manitoba (to our knowledge, at least), has widened its focus, and has increased dramatically both in size and in variety of members.

It is now perhaps appropriate for the group to consider changing its name. Some of the suggestions that have been made to date are "Winnipeg UNIX User Group" (for the obvious reason that we are Winnipeg based), and "Manitoba UNIX User Group" (since we do have members outside of the city, and we are not likely to be stepping on anyone's toes by claiming Manitoba as our area of coverage). Also, as we are considering affiliation with UniForum Canada, a name that indicates the region of Canada we cover would be desirable.

We are open to suggestions from the entire membership for a new name. Please contact an executive member if you have a name to suggest. We hope to be able to pick the best one (or a few of the best) and have a vote on the name change at a future meeting.

By any other name, the group will continue with its goal of providing a forum for all UNIX users, novice, technical, and management alike, to exchange ideas and information about all aspects of UNIX and Open Systems in general.

THIS MONTH'S MEETING

Meeting Location:
This month, the meeting is to be held at the Senate Chambers, room 245 in the new Engineering Building of the University of Manitoba, Ft. Garry Campus. (See the enclosed map of the campus.) The meeting is set to start at the usual time – 7:30 PM, on February 11, 1992. The Senate Chambers are on the main floor, near the entrance that faces University Centre.

Meeting Agenda:
See last page for details.

INSIDE THIS ISSUE

Newsletter Editor's Ramblings
President's Corner
The Fortune File
Feedback: Ask Monsieur Ex; Local UNIX Courses
Hands-on: RPC Programming; An "lpquit" Script
Technology: Motorola's MC88110 RISC Processor
January 14th Meeting Minutes
February 11th Meeting Agenda

UNIX is a registered trademark of UNIX System Laboratories, Inc.

RAMBLINGS

The "Squeaky Wheel"

By Gilbert Detillieux

There's an old saying that goes "the squeaky wheel gets the oil." In last month's newsletter, I must have squeaked quite loudly, when I begged for more material to be submitted, judging by the amount of stuff that has since poured in.

I would like to thank everyone who submitted something, particularly Scott Balneaves, for his article on RPC programming, and the promise that there will be more in the series, and Allan Moulding, for his article on the MC88110 processor.

I've also received various small bits of information, and several questions about UNIX. So this month, there's a new page with the heading "Feedback." I hope to continue this page in the future, and use it for a variety of submitted information (a "letters" column), and a Q&A column (provided the questions continue to come in, and provided Monsieur Ex, a mysterious Frenchman who claims to be an old editor and an expert in UNIX, is willing to give the answers).

I hope the flow will continue, as there are many more newsletters to come this year. I'll need more questions, fortunes, letters, and articles for next month. I would particularly like to see something under the "Industry" heading for next month.

Note to UniForum members: we can include articles from UniForum publications, as we've done in the past, but we have to type them in. If you read an interesting article, and are a reasonable typist, please type it in and submit it to me. That will help me considerably, as I'm a slow typist. Also, my throat is sore from squeaking.

The 1991-1992 Executive

President:	Susan Zuk	(W) 788-7312
Past President:	Eric Carsted	1-883-2570
Vice-President:	Richard Kwiatkowski	589-4857
Treasurer:	Rick Horocholyn	(W) 474-4533
Secretary:	Roland Schneider	1-482-5173
Membership Sec.:	Allan Moulding	269-8054
Mailing List:	Gilles Detillieux	489-7016
Meeting Coordinator:	Eric Carsted	1-883-2570
Newsletter editor:	Gilbert Detillieux	489-7016
Information:	Susan Zuk	(W) 788-7312
	(or)Gilbert Detillieux	489-7016

Copyright Policy and Disclaimer

This newsletter is ©opyrighted by the Technical UNIX User Group. Articles may be reprinted without permission, for non-profit use, as long as the article is reprinted in its entirety and both the original author and the Technical UNIX User Group are given credit.

The Technical UNIX User Group, the editor, and contributors of this newsletter do not assume any liability for any damages that may occur as a result of information published in this newsletter.

Our Address

Technical UNIX User Group
P.O. Box 130
Saint-Boniface, Manitoba
R2H 3B4

Internet E-mail:
tuug@cs.umanitoba.ca

Group Information

The Technical UNIX User Group meets at 7:30 PM the second Tuesday of every month, except July and August. The newsletter is mailed to all paid up members one week prior to the meeting. Membership dues are $20 annually and are due at the October meeting. Membership dues are accepted by mail and dues for new members will be pro-rated accordingly.

PRESIDENT'S CORNER

More Thoughts About Open Systems

By Susan Zuk, President

It seems that UNIX and Open Systems are front and centre in terms of questions and promotions in the world of computers. Computer departments are trying to tie together their various heterogeneous computer systems and are wondering if they should be looking at Open Systems or to a specific operating system like UNIX. A better understanding of these questions needs to be obtained. Many people are discussing what Open Systems are, and how to ensure that computer departments have what it takes to become open. This is one of the reasons why we offered the UNIX Symposium and is probably one of the reasons why another symposium is being held in Toronto.

A UNIX and Open Systems Executive Symposium is to be held in Toronto on March 4th and 5th. This symposium, sponsored by Datapro Canada, McGraw Hill, Unican Marketing Services and UniForum Canada, is aimed at helping executives who are concerned about preparing their organisations for the 90's and considering the use of Open Systems in that process. Anyone who would like more information about this event can give me a call.

Continuing on with Open Systems, there was our previous meeting which had Rocky Nystrom discuss Migrating to Open Systems. In his presentation, Rocky stated that the means of getting to Open Systems is as much of a requirement as just deciding on taking that route. He provided the listener with a process on developing Open Systems environments as well as some examples which his firm has handled. This was a well attended meeting and showed how hardy we Winnipegers are, as it was bitterly cold that night. A special thank you to Paul Hope for allowing us to use the St-Boniface Research Centre theatre.

Also at our last meeting, we had the pleasure of being visited by Robert Li of the Computer Post. He came not only to see what our group was doing but also to provide us with an invitation. The Computer Post is attempting to provide a channel of communication for the Winnipeg computing community. Robert would like to dedicate a section of the Computer Post to Winnipeg's major user groups. The idea is to allow all of us to share in the pool of valuable information that transpires at each user group. He would like us to write an article each month that catches some of this information. If you would like to be the contributor, or would like to provide us with ideas on what type of information would be most beneficial to contribute to the Computer Post, please give me a call at 788-7312.

We have included a survey with your newsletter. Please help us to provide you with the type of UNIX group you would like to see by completing the survey and either mailing or FAXing it back to us. A FAX and a mailing address are located at the top of the survey. Any additional comments would be greatly appreciated.

At the last meeting, a question was discussed about a communications line between members. If you are interested in helping to set up a communications gateway and also helping to provide a seminar to show members how to use this system, call me and I will put you in touch with the appropriate people.

A new column will be provided for your interest. In this column there will be the questions and answers from the previous meeting's round table. If you have any question and would also like to share the information with other group members, FAX your questions to Gilbert Detillieux at 269-9178.

That's all for this month; and I thought I had nothing to write about. Take care and please send in those surveys!

THE FORTUNE FILE

Misc.gems

From Kirk Marat (taken from "fortunes"):

A UNIX saleslady, Lenore,
Enjoys work, but she likes the beach more.
She found a good way
To combine work and play:
She sells C shells by the seashore.

From a signature in a news posting:

People never lie so much as after a hunt, during a war or before an election. — Otto von Bismarck

And another from a signature in a news posting:

SVR4: The first system so open that everyone dumps their garbage there.

From Kevin McGregor:

Just remember, games don't kill productivity, people do!

From a network service sign-on screen:

You couldn't even prove the White House staff sane beyond a reasonable doubt. — Ed Meese, on the Hinckley verdict

From Renate Scheidler:

Here's a good quote from the UNIX System V.2 Administrator's Guide (which I saw on 'fortune'):

> Making files is easy on the UNIX operating system. Therefore, users tend to create numerous files using large amounts of file space. It has been said that the only standard thing about all UNIX systems is the message-of-the-day telling users to clean up their files.

Seems to me all you have to do is replace the word 'file' by 'GIF file' and it'll be perfectly tailored to us here.

Ask Monsieur Ex

A column in which our resident Unix expert answers questions submitted by members, or discussed at round table sessions.

By Gilbert Detillieux

Q. In a Bourne shell script of mine, the new value being assigned to a variable in a "`if`" statement wasn't being propagated outside the "`if`." Is this because the conditionally executed statements are run in a subshell?

A. *Mais non!* The commands within Bourne shell control structures (if, while, case, etc.) aren't executed in a subshell unless you do that explicitly, e.g. with a parenthesized command list, or a pipeline, etc. Without seeing the script, I can't tell for sure. Is it possible that the body of the "if" is not even being executed? (You can use "echo" to add trace prints, or run the script with "sh -x" to trace the execution.)

Q. Is there an equivalent to `/etc/profile` for the C-shell? That is a file such as `.login` that's executed for every user?

A. *Je regrette*, but the answer is no. I wish such a thing existed, so that system administrators could make changes to the login procedure for all users without the problems associated with changing individual users' `.login` files. What can be done (and the sooner, the better), is set up each user's `.login` file to check for, and source, a system-wide file that could then be maintained by sys–admins. This is how our `.login` files begin:

```
if ( -r /usr/local/script/.login.global ) then
      source /usr/local/script/.login.global
endif
```

Q. When I use `rsh` to run a command, what is done with the standard input, standard output, and standard error output of the remote command?

A. The standard input to `rsh` will be read by `rsh` and passed along to the remote command (unless the `-n` option is given). The standard output and error output of the remote command are both passed along to the respective outputs of `rsh`. *Les gurus* will note that this requires two socket connections between `rsh` and its remote command; one is used for stdin/stdout, the other for stderr and for passing signals on to the remote command.

Q. How do you view a file in hex on UNIX systems?

A. The `od` command, which normally produces octal dumps (from *le bon vieux temps* of the PDP-11, which used octal), can produce hex dumps as well, by specifying the `-h` option. It prints out the values as 16 bit integers, i.e. 4 hex digits per value. If your machine is little-endian, the byte ordering will be reversed, so you may want to see if your `od` command lets you specify a word size with the `-w` option if you want individual bytes in the right order; if you can't use `-w`, you can use `-b` to get individual bytes in octal (sigh).

Q. I create a shell script and make it executable using `chmod`. When I'm logged in as `root`, I can't run it from my current directory, unless I give its full path name. As any other user, it works OK. What is happening?

A. *Ah, mon ami*, you're experiencing a feature, not a bug. All UNIX shells search for commands (when no explicit path name is given) only in the directories specified in your PATH variable. This gives you the option of searching the current directory at whatever point you like, or not at all. On most systems, the PATH used by `root` only has a small list of trusted directories, and excludes the current directory, for security reasons – it would be easy for `root` to inadvertently run a *Trojan horse* program otherwise.

Job Wanted

Competent, hardworking, and responsible individual seeking Unix related employment (permanent/casual) in the areas of:

- Unix system administration.
- Unix related hardware and software marketing/customer support.
- Unix networking support.

Experience:

- 4 years of Unix experience in large installation of Sun workstations and other Unix systems.
- Skilled in diagnosing and solving system problems.
- Hands-on experience with system admin., networking, X, C, TCP/IP, E-mail, etc.
- Very sensitive to user and customer needs.
- Up to date with current Unix hardware and software trends.

Contact Andrew Chan at 275-2456 or e-mail <andchan@ccu.umanitoba.ca>.

(Full resume and references available.)

Local UNIX Courses

The University of Manitoba Downtown is offer the following Unix/Xenix courses:

Introduction to SCO Xenix System V — provides the technical overview of the SCO System V hardware and operating system enviroment as well as hands-on experience using commands and files. March 9, 10, & 11.

SCO System V Administration for End Users — provides non-technical users the information to preform routine system maintenance tasks, and hands-on experience in SCO system adminstration. March 12 & 13.

These courses are SCO authorized and can be used for credits towards the SCO ACE program. For more information, and for costs, call Greg Anderson at 474-8028.

Unisys Canada is offering "C Language Programming," a 5 day course, starting March 16, in Winnipeg. It's aimed at programmers experienced in other high level languages.

This is the first of a series of generic UNIX courses to be offered. Others will be scheduled based on the training needs in Manitoba. For more information, call Susan Zuk or Debbie Harapiak at 788-7400.

Sun RPC Programming

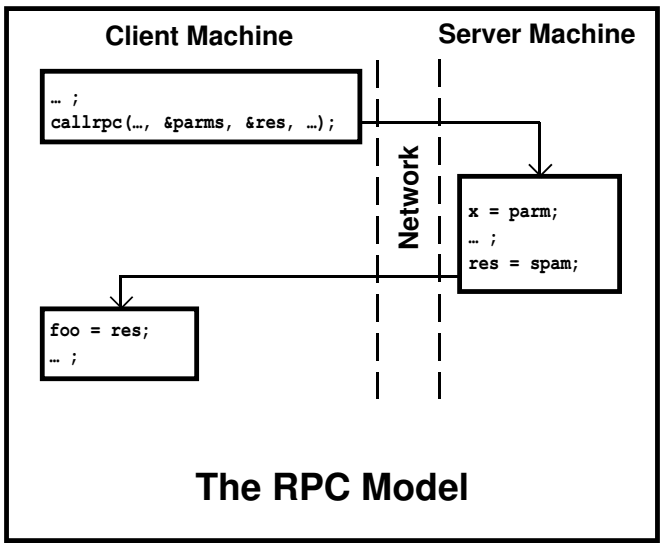
Network programming for client/server applications can be a tricky business. In a series of articles, the author hopes to shed some light on the subject.

By Scott Balneaves

There's a lot of applications being written nowadays using the "client-server" architecture that Sun Microsystems helped to pioneer. I had occasion to use RPC functions a few months ago, and I've found that there are a myriad of uses for them. I hope I can interest a few other people out there through this series of articles.

What I'm going to assume

I'm going to make this series of articles relatively technical, with an emphasis on actually doing some programming. I'm going to assume that you are familiar with Unix networking concepts, and that you have programmed in 'C' before. The examples I'll use will be tested on a Sun network, but will work on SCO, AIX, or HP Unix if you use the proper load libraries. I'll give you an overview, and code to try, but I won't discuss anything in great detail. This is just to get you started. You'll have to read the manuals for all the gory details (hey, I had to, so why shouldn't you?).



The RPC Model

It looks simple pictorially, and it's simple conceptually as well. All RPC (Remote Procedure Call) does is the same thing as a regular function call, only the function that you're calling isn't part of the same program. In fact, it doesn't even need to be on the same machine. It's a form of inter-process communication that can be carried out over the network. You simply send the RPC server your parameters, and it executes the function, and returns the results to you. It's all transparent to you how the data gets there and back. The server's job is to simply sit and listen for requests to come in, and send answers back.

The Nitty Gritty

The two simplest functions you will need to start off with are registerrpc() and callrpc(). You use them as follows:

registerrpc(PROGNUM, VERSNUM, PROCNUM, functionname, xdr_parmtype, xdr_restype);

This registers your function <functionname> as an RPC server. PROGNUM is the RPC's program number, in the range of 0x20000000 - 0x3fffffff (check your manuals for details). VERSNUM is the version number of your routine. You usually start this at 1. You can have multiple versions of your routines running. This is so if you make an improvement to your RPC program, your clients can access the new version or the old for compatibility. PROCNUM is the procedure number, usually starting at 1. The parameter and return types are specified afterwards. Again, check the manual for the boring details.

callrpc(server, PROGNUM, VERSNUM, PROCNUM, xdrparmtype, &parm, xdr_restpe, &result);

This is the RPC call function. Notice that the parameters must all be passed by address. The <server> is the character string name of the machine on which the server program is running.

After you have registered your RPC, you will need to execute "svc_run();"

This routine should never return. It's the dispatcher that sits and listens for requests to come in. The only other point to note is that the return values from your remote server must be declared as static. I'll explain why next time.

Example

This example is somewhat contrived, but it gives you the general idea. The client function passes a userid, and the server will tell you whether that user is currently logged into the machine that the server is running on. Simplistic, yes, but it gives you the idea. The three files are:

`who.h`

`who_svc.c` (*server code*)

`ison.c` (*client code*).

You compile the server with:

```
cc who_svc.c -o who_svc -lrpcsvc
```

and the client with

```
cc ison.c -o ison -lrpcsvc
```

The client syntax is:

```
ison <machine> <user>
```

e.g. `ison silver sbalneav`

Have fun, and see you next month!

```
/*
 * Filename: who.h
 * Author:   Scott Balneaves, Jan 15, 1992
 */

#include <rpc/types.h>

#define WHO_PROG (u_long)0x20001000
#define WHO_VERS (u_long)1
#define WHO_PROC (u_long)1

int *whos_there_1();
```

```
/*
 * Filename: who_svc.c
 * Author:   Scott Balneaves, Jan 15, 1992
 */

#include <stdio.h>
#include <signal.h>
#include <string.h>
#include <rpc/rpc.h>
#include "who.h"

void catch_term();
#define PIPECMD \
   "/usr/bin/who | /usr/bin/awk '{print $1}'"

int main()
{
       int    retval;

       /* Parent exits, child continues */
       if (fork())
              return(0);

       /*
        * catch sigterm so we will exit
        * gracefully, and de-register
        * ourselves from the system.
        */

       signal(SIGTERM, catch_term);

       /*
        * Register our RPC program.
        * Our remote procedure
        * is whos_there_1, with input
        * paramter type of string (char *)
        * and output parameter type of int.
        */

       retval = registerrpc(WHO_PROG,
                     WHO_VERS, WHO_PROC,
                     whos_there_1, xdr_string,
                     xdr_int);

       /*
        * check for success in registering.
        */

       if (retval) {
              fprintf(stderr,
                     "Couldn't register\n");
              exit(1);
              }

       svc_run();
       /* This should never return! */
       fprintf(stderr,
              "svc_run() returned\n");
       exit(1);
}

void
catch_term()
{
       fprintf(stderr, "Caught SIGTERM\n");

       /*
        * Unregister ourselves (to be nice)
        */

       svc_unregister(WHO_PROG, WHO_VERS);
       exit(1);
}

int *
whos_there_1(parm)
       char **parm;
{
       static int    retval;
                     /* NOTE STATIC DECL */
       FILE          *pipe;
       char          buf[BUFSIZ];

       /*
        * This is lazy, but what the hey!
        */

       if ((pipe = popen(PIPECMD, "r"))
                     == (FILE *) NULL) {
              fprintf(stderr,
              "Error: couldn't open pipe\n");
              exit(1);
       }
```

```
        retval = 0;

        /*
         * Find if our user is logged on.
         */

        while (fgets(buf, BUFSIZ, pipe)
                                != NULL)
                if (!strncmp(buf, *parm,
                             strlen(*parm)))
                        retval = 1;

        pclose(pipe);

        return(&retval);  /* return results */
}
```

```
/*
 * Filename: ison.c
 * Author:    Scott Balneaves, Jan 15, 1992
 */

#include <rpc/rpc.h>
#include <sys/types.h>
#include <sys/time.h>
#include <stdio.h>
#include "who.h"


int main(argc, argv)
        int argc;
        char *argv[];
{
        int    result, retcode;

        if (argc != 3) {
                fprintf(stderr,
               "%s: useage: %s <servername> <user>\n",
                                argv[0], argv[0]);
                exit(1);
                }

        /*
         * Make our RPC call to the server.
         */
        retcode = callrpc(argv[1], WHO_PROG,
                        WHO_VERS, WHO_PROC,
                        xdr_string, &argv[2],
                        xdr_int, &result);

        if (retcode) {
                fprintf(stderr,
                        "callrpc() failed\n");
                exit(1);
                }

        if (result)
                printf(
                "user %s is logged onto %s\n",
                        argv[2], argv[1]);
        else
                printf(
            "user %s is NOT logged onto %s\n",
                        argv[2], argv[1]);

        return(0);
}
```

Lpquit – An Interactive Cancel Command for the System V LP Spooler.

Submitted by Kirk Marat

```
#!/bin/sh
#lpquit   9/28/91 R.  Nunlist UC Berkeley.
#  Script to interactively cancel
#  lp requests.
#
Version=092891
echo "$0 $Version "
i=1
if  [ "`lpstat -o`" = "" ]
then   echo "No lp requests pending!"
       sleep 1
       exit
fi
echo "List of Print or Plot Lp Requests:"
lpstat -o
echo ""
LPS="   "

while [ "$LPS" != "" ]
do   LPS=`lpstat -o | tail -$i | sed  1p `
     if [ "$LPS" = "" ]
     then  echo "No lp requests pending!"
           sleep 3
           exit
     fi
     LPID=`echo $LPS |cut -f1 -d" "`
     USER=`echo $LPS |cut -f2 -d" "`
     echo "Cancel $LPID $USER ?" \
           "(y to cancel, q to quit) \c"
     read answ
     if [ "$answ" = "y" ]
     then  cancel $LPID
     fi
     if  [ "$answ" = "q" ]
     then exit
     fi
     i=`expr $i + 1 `
done
```

Motorola's MC88110 RISC Microprocessor Debuts

By Allan Moulding

Late last year, Motorola announced its next generation RISC microprocessor, the MC88110. This chip contains the following 10 units:

- Superscalar Instruction Unit
- 64-bit Graphics, Integer, and 80-bit Floating Point Multiply Execution Unit
- 64-bit Integer and 80-bit Floating Point Divide Execution Unit
- 80-bit Extended Precision Floating Point Add Execution Unit
- Two 64-bit 3-D Graphics Execution Units
- Two 32-bit Integer Arithmetic Logic (ALU) Execution Units
- 32-bit Bit-Field Execution Unit
- Data Unit with Load Buffers and Store Reservation Stations

There is also two eight ported register files:

- Thirty-two 32-bit General Purpose Registers for Operand Storage
- Thirty-two 80-bit Extended Registers for Additional Floating Point Operand Storage

All 88000 family computational instructions are register to register, or register plus intermediate value instructions. This eliminates memory access delays, and modern compilers can move address computations out of critical loops, thus increasing instruction throughput. All instructions are implemented as single word (32-bit) opcodes, greatly simplifying instruction pipelining. There are two levels of privilege: supervisor mode, which is primarily used by operating systems, and user mode, the lower privilege level of the two, which is used by application software. Supervisor mode prevents application software from corrupting critical operating system resources.

An item which aids in the extensibility of the 88000 family is the concept of special function units (SFU's). A SFU is defined as a set of instructions, with a common opcode field. This provides additional functionality to the base architecture. An example is the graphics execution units, which is defined as SFU2. The previous generation chip, the MC88100, didn't have the graphics execution units. Up to seven SFU's are supported by the 88000 family architecture.

The two 32-bit integer ALU's are identical, thus 2 ALU instructions can be issued simultaneously. Since ALU instructions execute in one clock cycle, processing isn't stalled due to ALU unavailability.

The graphics processing unit is targeted for improving the interactive performance, raster conversion, and image processing. Other things like viewpoint transformation, lighting, and display are handled by other execution units. The pixel add, and the pixel pack execution units support the graphics processing unit. Pixels are packed into 64-bit fields and stored in register pairs in the general register file. Graphics instructions process the individual fields in parallel.

The instruction cache and data caches are both 8 KB in size and are two-way set associative. The instruction and data memory management units (MMU's) provide two 4 GB logical address spaces each, one each for supervisor and user mode. The MC88100 used separate cache memory management unit (CMMU) chips, the MC88200 or the more advanced MC88204. Cache coherency is maintained by bus snooping. This is especially important in multiprocessor systems.

Performance-wise, the MC88110 has a simulated SPECmark rating of 63.7 SPECmarks (using SPEC release 1.2) at 50 MHz, with no secondary cache. The geometric means for integer and floating point are 51.0 and 73.9 SPECmarks respectively. The simulator is called XSim, and is an accurate instruction level model of the MC88110 that presents timing information on a clock-by-clock basis. There are a few things that would lower the SPECmark ratings in actual systems, but Motorola suspects that it would be offset by future compiler improvements. Motorola estimated that there would be a 20-25% improvement in performance by using a reasonably large secondary cache. Here are the individual simulated SPECmark ratings:

Benchmark	Rating
gcc	46.5
expresso	48.1
spice	34.7
dudoc	41.4
nasa7	67.9
xlisp	57.0
eqntot	52.9
matrix300	357.8
fpppp	64.4
tomcatv	72.2

The same configuration had a simulated benchmark of 185,000 Dhrystones using version 2.1 of the Dhrystone benchmark.

*The information in this article, except for the performance data, was taken from Motorola's publication "*Technical Summary MC88110 RISC Microprocessor*" (order code MC88110/D).*

***Allan Moulding** attended the University of Manitoba, graduating with a B.Sc. in Statistics in 1985. For the past 5 years, he's run his own consulting firm, A.B.M. Services, specialising in the CAD area. His areas of interest are graphics and programming languages.*

TUUG Meeting Minutes

Tuesday, January 14, 1992, 7:30 PM
St-Boniface Hospital Research Centre
Theatre, Main Floor, 351 Taché

Chair: Susan Zuk
Attendance: 29

Business Meeting:

a) President's Report

1) Computer Post editor Robert Li would like to encourage TUUG members to complete the survey in the December issue of the Computer Post. He also would like articles for the paper.

2) Uniforum

- Benefits of joining: Uniforum provides funding for projects like the Symposium, etc., and would allow us to find out what other organizations in Canada are doing.
- Winnipeg is one of the few major cities without representation on Uniforum.
- Individual memberships cost $100 per year. For this you get Uniforum monthly, biweekly information mailings, and UNIX product information.
- There is no group membership fee, and not everyone in TUUG has to join Uniforum.
- TUUG only has to sign a charter saying we promote open systems.

3) Name change – The "Technical" part of TUUG is intimidating to many potential members. A proposal for a name change will be included in the February newletter.

b) Membership Secretary's Report

Current membership is 58, up from the mid 20's last year.

c) Newsletter Editor's Report

As always, the newsletter is looking for submissions.

d) Treasurer's Report

The TUUG bank account currently stands at $1590, with another $8000 expected from CIPS as the TUUG share of the Symposium profit. TUUG members are encouraged to make suggestions for the best ways to use this money.

e) Meeting Coordinator's Report

- SCO will make a presentation in April or March
- Breakfast or lunch meeting in May will be an hour to an hour and a half. We will bring in a speaker. There will be a nominal charge for those attending.
- Amdahl may make a presentation in May.
- The February meeting may be a presentation on databases, hosted by the University of Manitoba.
- Other suggestions: Uniplex office automation.

Presented topic:

Videotape of Rocky Nystrom's Symposium talk "Migrating to Open Systems"

Agenda

for
Tuesday, February 11, 1992, 7:30pm
Senate Chambers, 245 Engineering Bldg.
University of Manitoba, Ft. Garry Campus

1. Round Table 7:30

2. Business Meeting 8:00
   a) President's Report
   b) Membership Secretary's Report
   c) Newsletter Editor's Report
   d) Treasurer's Report
   e) Meeting Coordinator's Report
   f) New Business

3. Break 8:20

5. Presented Topic 8:30

Networking and Unix at the U of M

Networking and Unix are essential and evolving components of Computer Services at the University of Manitoba. Bill Reid, manager of Networking, and Kathy Norman, Unix Administrator, will discuss the present and the future in their respective areas. A brief tour of the facilities will follow.

6. Adjourn 9:30

Note: Please try to arrive at the meeting between 7:15 and 7:30 pm. Thank You.

Next Month

Meeting:

Our March meeting is scheduled for Tuesday, March 10, at 7:30 PM. The presented topic is "Writing Software for Portability," by Gilbert Detillieux. The meeting will start off with our usual round table and business meeting. Location TBA.

Newsletter:

We will likely continue with our Q&A column, and RPC Programming by Scott Balneaves, next month. I would like to see something for next month's "Industry" and "Technology" headings, or any other topics that would be of interest to members. Any other aspiring writers out there?