News from the hobbyists

Prof. Dr. Bernd Ulmann

02-NOV-2012

Hochschule fuer Oekonomie und Management, Frankfurt

Introduction

What will be covered by this talk?

- First of all a big thank you to the community.
- News for APL enthusiasts – Lang5 has evolved into `Array::APX`.
- A multiprocessor `simh` developed by Sergey Oboguev.

Thank you

Thank you

I would like to say "thank you" to the OpenVMS community! Not just because I always feel at home when I am with you, but also because of this event:

- On 04-JUL-2012 we had a power outage in my village that lasted for more than 2 hours!
- This, of course, affected FAFNER (`fafner.dyndns.org`) my beloved VAX-7000/820 which was humming along in the basement for the last 12 years.
- Unfortunately the system was heavily affected by this power outage: Two 36 GB SCSI disks had died (fortunately I had a quite recent backup although it could have been more recent :-)) and, worst of all, the HSJ50 controller had died!

Thank you

- In the very same night, while I was trying to devise a work around my phone started ringing and mails began to pour in. Several people from Germany, the netherlands and Great Britain wondered what had happened to FAFNER and offered help.
- One day later I got some inside information of how to bring the HSJ50 back to life. :-)
- A couple of days later a care packet containing half a dozen spare disks arrived.
- Still a couple of days later I got a DLT-drive for future backups.

So thanks to you, the truly wonderful OpenVMS community, FAFNER is back online again and has now plenty of spare disk drives. :-)

Array Languages

Most of you know that I love array languages. I was already fascinated by the ideas of APL when I read an APL book in the late 1980s. When I got access to VAX APL on a VAX 8550 at my university I was hooked.

Then, somehow, the last VAX APL system was shutdown at the university and I tried various other APL systems but none of these provided such a cosy environment as VAX APL and none was running on OpenVMS.

As a result I started developing a new array language called `Lang5` that you may know from last year's talk at the German TUD. Since then a lot has happened...

Do you remember times where you could type something like this to get a list of prime numbers between 2 and 100?

```
ULMANN:FAFNER$ apl
terminal..tty
VAX APL V4.0-894
TNA114: SATURDAY  3-NOV-2012 19:25:29.41 ULMANN [ULMANN]
CLEAR WS
      (~A .EP A .SO . # A) / A _ 1 .DA .IO 100
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
      )OFF
TNA114:  SATURDAY  3-NOV-2012 19:25:44.81
CONNECTED 00:00:15.40  CPU TIME 00:00:00.17
9 STATEMENTS  51 OPERATIONS
615 PAGE FAULTS  37 BUFFERED IO  7 DIRECT IO

ULMANN:FAFNER$
```

If you are not into software archeology you have to consider VAX APL as being dead, so what now?

A couple of years ago development on Lang5 started – an array programming language that runs out of the box on any system with a Perl interpreter, especially OpenVMS systems.

Using Lang5 the example program from above looks like this:

```
ULMANN:FAFNER$ lang5
loading mathlib.5: Const..Basics..Set..Stat..Cplx..P..LA..Graph..NT..
loading stdlib.5:  Const..Misc..Stk..Struct..
lang5> 99 iota 2 + dup dup dup '* outer swap in not select .
[   2    3    5    7   11   13   17   19   23   29   31   37
41   43   47   53   59   61   67   71   73   79   83   89   97  ]
lang5> exit

ULMANN:FAFNER$
```

Although Lang5 turned out to be fun to use it is hard for most users to adapt to a stack based array programming language paradigm. So something new was developed (which, of course, runs on OpenVMS, too :-)):

Using the basic functionality that was developed for the Lang5 interpreter a Perl module named Array::APX has been developed that now offers most of the functions necessary for an array programming language for the Perl environment:

```
ULMANN:FAFNER$ type prime.pl
use strict;
use warnings;
use Array::APX qw(:all);

my $f = sub { $_[0] * $_[1] }; # We need an outer product
my $x;

print $x->select(!($x = iota(199) + 2)->in($x |$f| $x));

ULMANN:FAFNER$ perl prime.pl
[    2    3    5    7   11   13   17   19   23   29   31   37   41   43
47   53   59   61   67   71   73   79   83   89   97  101  103  107  109
113  127  131  137  139  149  151  157  163  167  173  179  181  191  193
197  199 ]
ULMANN:FAFNER$
```

Installation

- Make sure you have a working Perl interpreter on your system.
- Get `array.zip` from
  `http://fafner.dyndns.org/~ulmann/array.zip`
- Perform the following steps (adapt them to your local Perl installation!):

```
SYSTEM:FAFNER$ cd perl_root:[lib.site_perl]

SYSTEM:FAFNER$ unzip disk$user_0:[ulmann.bastel]array.zip
Archive:  DISK$USER_0:[ULMANN.BASTEL]ARRAY.ZIP;1
   creating: [.Array]
  inflating: [.Array]APX.pm
  inflating: [.Array]DeepUtils.pm

SYSTEM:FAFNER$ set prot=w:re array.dir

SYSTEM:FAFNER$ set prot=w:re [.array...]*.*
```

- You are now ready to use `Array::APX` on your OpenVMS system. Have fun with array programming again.

VAX MP

A multiprocessor VAX simulator

Sergey Oboguev has developed a simulator called VAX MP that provides virtual multiprocessor capability and allows to run OpenVMS VAX in an SMP environment. Some basic facts:

- VAX MP is based on simh and will become part of the simh code base soon (https://github.com/simh/simh. Please check the simh mailing list at http://mailman.trailing-edge.com/mailman/listinfo/simh for announcements.
- VAX MP simulates a multiprocessor MicroVAX 3900 – something that never existed!
- Preliminary documentation may be found here: https://s3.amazonaws.com/export2/VMS_UG.pdf
- VAX MP is not intended for production systems!
- The multiprocessor support requires a paravirtualization layer for the OpenVMS VAX system:

- VAX MP does not implement any historically existing VAX SMP hardware / buses.
- Thus OpenVMS does not recognize the simulated multiprocessor hardware it is presented with – initially it just sees a MicroVAX 3900.
- Thus VAX MP provides another piece of software beyond simh – a paravirtualization module called VSMP.EXE which must be loaded during startup (or whenever suitable) to expose the VAX MP multiprocessing capabilities to OpenVMS.
- Using the SYSGEN parameter MULTIPROCESSING OpenVMS is forced to load the necessary modules for multiprocessing during boot although the VAX MP at this moment looks like a single CPU MicroVAX 3900.
- Then VSMP.EXE is executed which reconfigures the system by creating additional CPU descriptors and exposing the MP capability of VAX MP to the OpenVMS system.

- VSMP.EXE dynamically modifies the memory-resident copy of the OpenVMS VAX kernel and some of its drivers.
- All in all ca. 30 patches are applied – most of these are necessary due to the weaker memory-consistency model of modern hardware, compared to a real VAX (out-of-order instruction issuing, speculative execution, write/read collapsing, asynchronous multibank caches and the like).
- This is necessary since there are some pieces of code in OpenVMS VAX that do not issue proper memory barriers by executing interlocked instructions (the PU and XQ drivers are examples).
- The number of virtual CPUs provided by VAX MP is limited by the number of CPUs on the host system (including hyperthreaded/SMT units).
- It is not possible to simulate more virtual CPUs than there are logical CPUs in the host system.

- OpenVMS VAX supports a maximum of 32 CPUs.
- Due to `VSMP.EXE` only OpenVMS VAX 7.3 is supported by `VAX MP` although earlier versions of OpenVMS might run as well.
- `VAX MP` currently runs under the following host operating systems:
  - Windows (32 and 64 bit executable)
  - LINUX (32 and 64 bit executable)
  - Max OS X (only 64 bit executable)
- Example session of `VAX MP`:

```
kundry$ vax_mp vax4-demo.ini
VAX MP simulator V3.8-3 RC2
...
KA655X-B V5.3, VMB 2.7
Performing normal system tests.
40..39..38..37..36..35..34..33..32..31..30..29..28..27..26..25..
24..23..22..21..20..19..18..17..16..15..14..13..12..11..10..09..
08..07..06..05..04..03..
Tests completed.
>>>b
(BOOT/R5:80 DUA0
  2..
-DUA0
  1..0..
%SYSBOOT-I-SYSBOOT Mapping the SYSDUMP.DMP on the System Disk
%SYSBOOT-I-SYSBOOT SYSDUMP.DMP on System Disk successfully mapped
%SYSBOOT-I-SYSBOOT Mapping PAGEFILE.SYS on the System Disk
%SYSBOOT-I-SYSBOOT SAVEDUMP parameter not set to protect the PAGEFILE.SYS
   OpenVMS (TM) VAX Version V7.3     Major version id = 1 Minor version id = 0
%WBM-I-WBMINFO Write Bitmap has successfully completed initialization.
$!  Copyright 2001 Compaq Computer Corporation.

%STDRV-I-STARTUP, OpenVMS startup begun at  3-NOV-2012 22:21:26.12
%RUN-S-PROC_ID, identification of created process is 00000206
%DCL-S-SPAWNED, process SYSTEM_1 spawned
...
%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
  SYSTEM       job terminated at  3-NOV-2012 22:26:06.39

  Accounting information:
  Buffered I/O count:            2533         Peak working set size:    1626
  Direct I/O count:               878         Peak page file size:      8424
  Page faults:                  14809         Mounted volumes:             0
  Charged CPU time:          0 00:00:16.80    Elapsed time:     0 00:00:35.35
```

```
 Welcome to OpenVMS (TM) VAX Operating System, Version V7.3

Username: system
Password:
 Welcome to OpenVMS (TM) VAX Operating System, Version V7.3
    Last interactive login on Thursday, 27-SEP-2012 10:57
    Last non-interactive login on Wednesday, 29-AUG-2012 08:55
$ sho cpu

MYVAX, a VAXserver 3900 Series
Multiprocessing is ENABLED. Full checking synchronization image loaded.

PRIMARY CPU = 00
Active CPUs:      00 01 02 03
Configured CPUs:  00 01 02 03
$
```

Thank you for your interest

The author can be reached at

ulmann@vaxman.de