Current status of 64-bit IRAF

Chisato Yamauchi

Institute of Space and Astronautical Science, Japan Aerospace Exploration Agency
(ISAS/JAXA)
Overview

- Importance of 64-bit IRAF
- Difficulty of porting IRAF to 64-bit
- SPP data models
- Our project - The IRAF64 Project
- Our results
- Screenshots
- Our Plan
Importance of 64-bit IRAF

- IRAF is the de facto standard software for analysis of optical/infrared data.
- Modern large telescopes and satellites still utilize the IRAF.
- IRAF will keep being used in the future.
- Current 32-bit IRAF has the 4GB barrier, which should be removed to handle incredibly high-resolved/wide-field data in the future.
  (e.g. SUBARU Hyper Suprime-Cam will output a 2.8GB of FITS)

next: Difficulty of porting IRAF to 64-bit
Difficulty of porting IRAF to 64-bit

- IRAF requires a small assembler code for each architecture.

```asm
zsvjmp_:
    # %rsi ... &status  %rdi ... &jumpbuf
    movq  %rsi, (%rdi)  # store &status in jmpbuf[0]
    movl  $0, (%rsi)   # zero the value of status
    addq  $8, %rdi     # change point to &jmpbuf[1]
    movl  $0, %esi     # change arg2 to zero
    jmp   _sigsetjmp   # let sigsetjmp do the rest
```

This is zsvjump.s for x86_64 OS. The arguments of a function are received by registers (not by stack).
Difficulty of porting IRAF to 64-bit

- Intrinsic problem is the tacit assumption of size of integer and pointer types in IRAF code.
  i.e., \( \text{sizeof(int)} = \text{sizeof(long)} = \text{sizeof(pointer)} = \text{sizeof(real)}. \)

Examples of actual problems:
- Incompatible pointer args of functions.
- There is ‘Memi’ (for memory access for integer type).
  However, ‘Memp’ (that for pointer) does not exist.
- \text{P2R() and P2I()} macros do not exist.
- There are tricks around ‘struct’ constructions in some codes.
Which data model is suitable for 64-bit SPP?

<table>
<thead>
<tr>
<th></th>
<th>short</th>
<th>int</th>
<th>long</th>
<th>pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILP32</td>
<td>16</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>LP64</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>ILP64</td>
<td>16</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
</tbody>
</table>

- If we select ILP64, a number of tools will work without code modification. However, IRAF contains many codes dependent on 32-bit integer.
- Genuine approach is to renounce the tacit assumption of integer/pointer sizes.

We select LP64 model and revise SPP specifications, and remove the root problem.

next: Our project
Our project - The IRAF64 Project

- Porting the IRAF software to the x86_64 Linux OS with NATIVE x86_64 (Opteron,EM64T) executable.
- This IRAF is developed as a candidate of IRAF Revision 3.0.
- Our project uses the SourceForge.Net (https://sourceforge.net/).
- The source code is maintained using the Subversion, and the repository is public now.
- We review and update all C/SPP codes.
Project members

- Project Admins/Developer:
  Chisato Yamauchi (Japan Aerospace Exploration Agency)

- Developer/Tester:
  Keith Rosema (University of Washington)
  Sergio Pascual (Complutense University of Madrid)
  Yasushi Nakajima (National Astronomical Observatory of Japan)

next: Our results
Our results: general improvements

- Improvements of C code to meet the present standard.
  - Enabled gcc’s ‘-Wall’ option and minimized gcc’s warnings.
  - Replaced ancient style (K&R -> ANSI).
  - Appended complete prototype declarations.
  - Improved a lot of security issues.
  - gcc-4.3 is also OK!

- Rearranged IRAF directory structure.

- Redefined ‘MACH’ environment variable.
  e.g., MACH=x86_64-linux-redhat, MACH=i386-linux-generic, etc.
Our results: general improvements

- Appended an mechanism of auto-detection of OS and architecture when building IRAF.
- Applied ‘GNU make’ to build IRAF easily.
  
  ```
  $ make boot
  $ make iraf
  $ make reboot
  $ make tables
  $ make noao
  $ make install (DESTDIR=...)
  ```
  
  OS and architecture are automatically detected on ‘make boot’ stage.

- Replaced csh-scripts with sh-scripts.
Our results: updates for x86_64

- Wrote an assembler code for Linux x86_64 architecture.
- Prepared the environment for ‘LP64-SPP’ development.
  - Some modifications for SPP preprocessor/convertor and f77 configurations.
  - xc, rpp, xpp and mkpkg are OK!
- Reviewed and updated C and SPP codes for libsys.a, libvops.a, libcur.a, libstg.a, libds.a, libex.a and libc.a.
  - Supported 64-bit memory allocation and file I/O.
  - Supported large data (> 4GB) in the majority of functions.
Our results: updates for x86_64

- ‘cl’ is running on CentOS 4.x/Fedora 8,9.
  - A few tasks of core system are OK.
  - However, the majority of tasks cause PANIC error.
A_A IRAF64 pre-alpha based on NOAO/IRAFNET PC-IRAF Revision 2,14
('v') This is NOT the released version of IRAF64.

This 64-bit version of IRAF is ported by the IRAF64 project hosted by
Chisato Yamauchi (ISAS/JAXA).

Welcome to IRAF. To list the available commands, type ? or ??.
To get detailed information about a command, type `help <command>'.
To run a command or load a package, type its name. Type `bye' to exit a
package, or `logout' to get out of the CL. Type `news' to find out
what is new in the version of the system you are using.

The following commands or packages are currently defined:

dataio, language, obsolete, softools, utilities.
dbms, lists, plot, system.
images, noao, proto, tables.

cl> !ldd /opt/local/iraf/iraf/bin/cl.e
libreadline.so.4 => /usr/lib64/libreadline.so.4 (0x0000000000000000)
libncurses.so.5 => /usr/lib64/libncurses.so.5 (0x0000000000000000)
libm.so.6 => /lib64/tls/libm.so.6 (0x0000000000000000)
libc.so.6 => /lib64/tls/libc.so.6 (0x0000000000000000)
/lib64/ld-linux-x86-64.so.2 (0x0000000000000000)
Screenshots (showcap)

```
cl> showcap

set device
  "\"" (to dump full graphcap entry
  cc [arg1 [arg2 [arg3]]]

cc:
  a two character capcode (e.g., 'cm')
  an encoder program (non alpha first char)

* set vt640
  RC=(1$2)^X\E[24;65H\E[7mLIGHT PEN READY\E[0m($$)^\E"(1$2)5($D)4($$)g;\n  WC="]\t\e/f:0w=150^"; CW="\E[24;OH\E["K;GE=150^"; GD="\E[24;OH\E["K;\n  lt#5;nc#2;se:CL=50^";E^L;x=640;yr#480;ar#,.57;xs#,23;ys#,13;ar#,70;\n  ch#,0294;co#80;cw#,0125;in;k1#1;k2#127;kf=cl;li#35;lt#5;nc#1;\n  nk#127;pl:pm;th#4:t1#1:t2#2:t3#3:t4#4:tx:wc:xr#1024;yr#780;xs#,20;\n  ys#,14;CD="M;CN#6;LT=\";E/(1$0)1d\E`($-5)0d\E(1_+.D)0d\E(`$;\n  MS=\034;FL;RC=\E^Z;SC=(,13, & *, &+1, & *, &+12;TH=\E(1#47+;\n  TS=\"];Vs=\";DE=\";X1#0;X2#1023;XY%=Y1#0;Y2#779;OW=\";]\_\;\n  CW=(#682!2#011)^]\t_;GE=\";CL=1000(#32!9)\E^L;LR=(#32!9;GD=(#9!-99$031!9$d\n  #22*!2#011)^]\t_;\n
* TH 2

program: ^[(1#47+.
encode: ^[1
      status = 0
```
Current status of 64-bit IRAF – p.16/19
Welcome to IRAF. To list the available commands, type "?" or "??". To get detailed information about a command, type help "<command>". To run a command or load a package, type its name. Type "bye" to exit a package, or "logout" to get out of the IL. Type "news" to find out what is new in the version of the system you are using.

The following commands or packages are currently defined:

database, language, database, softools, utilities, done, list, plot, system, images, redo, proto, tables, cli, interactive
cli> display dos:oil 1
cli> llist /usr/local/iraf/iraf/bin/cli
libenvironment.so.4 => /usr/lib64/libenvironment.so.4 (0x0000000000000000)
libcurses.so.5 => /usr/lib64/libcurses.so.5 (0x0000000000000000)
libеньк.so.6 => /usr/lib64/libcctor.so.6 (0x0000000000000000)
libc.so.6 => /usr/lib64/libcctor.so.6 (0x0000000000000000)
libmalloc.so.2 => /usr/lib64/libmalloc.so.2 (0x0000000000000000)
cli> primary spe3d.MODEL
iraf
This 64-bit version of IRAF is ported by the IRAF64 project hosted by Chisato Yamazaki (iras64@IRAF).

Welcome to IRAF. To list the available commands, type ? or ?? To get detailed information about a command, type help command. To run a command or load a package, type its name. Type 'exit' to quit a package, or 'logout' to get out of the CL. Type 'new' to find out what is new in the version of the system you are using.

The following commands or packages are currently defined:

dataio, language, obsolete, softools, utilities,
dmx, lists, plot, system,
images, nom, proto, tables.

% disp1_1comp.fits
frame to be written into (1/10) (1):
% z=78.9584 y2=97.688
% is
1_1comp.fits cihistory.txt 020080412_05_iras,Fits login,cl upmem kg st
% luname-9
Linux bourla,plain,iras,jpa,ja 2,8,9-67,0,15,Elmer #1 SSH Thu May 8 10:50:20 EDT 2008 x86.64 x86_64 x86_64 GNU/Linux
Our plan

- First, we intend to provide 64-bit support of core system. (i.e., libraries and basic tasks in sys/ pkg/ and math/ directories).
- Tables library and NOAO standard utilities will be updated after support of core system.
- Collaboration with STScI/NOAO may be wanted for updates of Tables library and NOAO standard utilities.