EPICS: A Retrospective on Porting iocCore to Multiple Operating Systems

M.R. Kraimer, J.B. Anderson, ANL
J.O. Hill, LANL
W.E. Norum, University of Saskatchewan
**iocCore: Important EPICS Component**

- IOC Input/Output Controller originally
  - Meant a VME/ VXI based system
  - Required the vxWorks operating system
- ICALEPCS99 paper presented plan to port to other OSs
  - This talk describes the successful implementation of the plan.
  - The basic plan was followed but many details changed.
  - vxWorks, RTEMS, Solaris, Linux, WIN32 now supported
Major Problems - Solution

- VME/ VXI – Hardware Support Unbundled.
- vxWorks libraries – Define/ Implement OSI interfaces.
- vxWorks dynamic loading – Registry.
- Build Environment – Major changes.
- vxWorks shell – iocsh (ioc shell)
- Interrupt Level support
  - Use a global mutex if OS doesn’t allow interrupt level support.
### Operating System Independent Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Library</th>
<th>Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>epicsRing</td>
<td>rngLib</td>
<td>Generic</td>
</tr>
<tr>
<td>epicsTimer</td>
<td>wdLib,osiTimer</td>
<td>Generic</td>
</tr>
<tr>
<td>epicsAssert</td>
<td>epicsAssert</td>
<td>Default, vxWorks</td>
</tr>
<tr>
<td>epicsEvent</td>
<td>semLib</td>
<td>RTEMS, WIN 32, POSIX, vxWorks</td>
</tr>
<tr>
<td>epicsFindSymbol</td>
<td>symFindByName</td>
<td>Default, vxWorks</td>
</tr>
<tr>
<td>epicsInterrupt</td>
<td>intLib</td>
<td>RTEMS, default, vxWorks</td>
</tr>
<tr>
<td>epicsMutex</td>
<td>semLib</td>
<td>RTEMS, WIN 32, POSIX, vxWorks</td>
</tr>
<tr>
<td>epicsThread</td>
<td>taskLib</td>
<td>RTEMS, WIN 32, POSIX, vxWorks</td>
</tr>
<tr>
<td>epicsTime</td>
<td>tickLib,osiTime</td>
<td>RTEMS, WIN 32, POSIX, vxWorks</td>
</tr>
<tr>
<td>osiPoolStatus</td>
<td>memLib</td>
<td>RTEMS, WIN 32, default, vxWorks</td>
</tr>
<tr>
<td>osiProcess</td>
<td>osiProcess</td>
<td>RTEMS, WIN 32, POSIX, vxWorks</td>
</tr>
<tr>
<td>osiSigPipeIgnore</td>
<td>osiSigPipeIgnore</td>
<td>WIN 32, default, POSIX, vxWorks</td>
</tr>
<tr>
<td>osiSock</td>
<td>osiSock</td>
<td>Linux, RTEMS, WIN, default, solaris, vxWorks</td>
</tr>
</tbody>
</table>
Some Details

- **Registry**
  - vxWorks `symFindByName` - bind to global symbol.
  - iocCore dynamically binds record/device/driver/etc support.
  - While building application:
    - A perl program generates a C function
    - C function is linked with application. During startup the C function is called. It registers the support.

- **Build Environment**
  - Extensive changes, more functionality, easier to use.
  - `iocsh` - simple command interpreter, built in commands.

- **Interrupt Level Support**
  - iocCore has minimal use.
  - vxWorks, RTEMS real support. Default uses global mutex.
Status of Port

- **3.14.0beta1** – First beta release tested on
  - vxWorks 5.4, RTEMS 4.6, solaris 8, Redhat 6.2 7.1, NT,98,2000.

- **Work Remaining**
  - No new functionality will be added to 3.14.
  - With more testing will have 3.14.1.
  - HPUX – Work on HPUX-11 is in progress.

- **Hardware Support**
  - Much of unbundled hardware support tested. Only on vxWorks.
  - Unbundled GPIB support works with 3.14,
    The LANGPIB support works on all platforms
  - Unbundled Sequencer works with 3.14.
  - Some unbundled hardware support is available for RTEMS.
Supporting a New Platform

- Environment must have: multithreading, GNU make, and PERL.
- `<base>/src/libCom/osi/os`
  - default
  - posix
  - `<platform specific>`
- Implementation provided three ways:
  - default - If this version works nothing to do.
  - posix - Requires POSIX real time and POSIX threads. If this works nothing to do.
  - `<platform specific>` - If neither default or posix works.
**Lines of code: src/libCom/osi/os**

<table>
<thead>
<tr>
<th>OS</th>
<th>Lines of Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vxWorks</td>
<td>1283</td>
<td>Does not use posix.</td>
</tr>
<tr>
<td>RTEMS</td>
<td>1488</td>
<td>Does not use posix.</td>
</tr>
<tr>
<td>Solaris</td>
<td>89</td>
<td>Uses posix</td>
</tr>
<tr>
<td>Linux</td>
<td>145</td>
<td>Uses posix</td>
</tr>
<tr>
<td>WIN32</td>
<td>3651</td>
<td>Almost no use of posix or default</td>
</tr>
<tr>
<td>posix</td>
<td>1262</td>
<td></td>
</tr>
<tr>
<td>default</td>
<td>950</td>
<td></td>
</tr>
</tbody>
</table>
Channel Access 3.14 changes

- CA client now thread safe and requires multithreading.
- Use plug compatible interfaces for in-memory services.
- Supports unlimited vector lengths.
- Client specified dispatch priorities.
Compatibility

- Converting R3.13 IOC applications to 3.14.
  - The old 3.13 build rules are still supported.
  - Using old build rules requires few changes. However, all hardware support is unbundled.
  - Ultimately should convert to new rules.
- Old CA client interface still supported
  - Many CA client applications have been built with 3.14.