



IOC Application Development/Debugging

Shanghai EPICS Seminar

Tuesday, 8/29

J.Odagiri



Before Getting Started...

- √ 如果有付公不明白的地方、請別客氣。
- √ This lecture includes...
 - √ What VxWorks is
 - √ How you boot EPICS
- √ We have 90 minuets...
 - √ 15 minuets for general information
 - √ 75 minuets for working on exercise



VxWorks is Not Unix

- v It's true that VxWorks is a Unix like OS, but ...
- v Tasks share a single address space.
 - v There can be only one global variable, or one function, for a specific symbol name in the whole system.
 - v Any tasks can see any global symbols in the system.

VxWorks is Not Unix (continued)

- ✓ **Tasks always run in the Kernel Mode**
 - ✓ Tasks does not use system trap for “system calls”.
 - ✓ Applications can call functions in the kernel by a usual function call.
- ✓ **Tasks are scheduled on their fixed priority.**
 - ✓ VxWorks is a real-time OS, not a TSS system.



Comments on VxWorks

- v **Seamless unification between VxWorks and Applications**
- v **The line between VxWorks and applications is just the matter of “Who wrote the code?”**
- v **When you are developing applications under VxWorks, it’s more like you are developing the kernel itself under Unix.**



Good Points and Downsides

- ✓ **Good Points**

- ✓ High performance
- ✓ Lots of flexibility

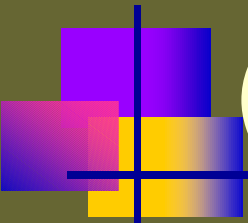
- ✓ **Downsides**

- ✓ No protection, not at all
- ✓ Hard to debug



The First 3 Points to Get

- ✓ **System Symbol Table**
 - ✓ Holds all addresses of the functions and variables which have an external linkage
- ✓ **Linking Loader**
 - ✓ Resolves external references in object modules upon the loading by referring to the System Symbol Table



The First 3 Points to Get (continued)

- ✓ **Target Shell**
 - ✓ Looks up the typed command(function) in the System Symbol Table
 - ✓ If found, Shell calls it inside the shell's own context, without creating a new task.



Creating Tasks

```
taskSpawn(  
    char *name,  
    int  priority,  
    int  options,  
    int  stackSize,  
    FUNCPTR *entryPt,  
    int  arg1, ... ,  
    int  arg10 );
```



Interrupt Service Routines (ISRs)

- v Not scheduler but hardware triggers ISRs.
- v ISRs run in a special context outside of any task's context.
- v ISRs must not invoke functions that might cause the caller to block.



Startup Scripts

- v VxWorks shell allows you to invoke commands with a script file.

an example of startup scripts

```
cd "/users/yangcn/startup/st.cmd"
```

```
pwd
```

```
ld < testModule.o
```

```
moduleShow
```

```
myInitRoutine( "tHello", 50, 0 )
```



Loading iocCore and Others

- v **iocCore**
 - v Channel Access
 - v Database Access . . .
- v **seq**
 - v Run-time sequencer
- v **shanghaiAppLib**
 - v Record Support
 - v Device Support
 - v Driver Support



Loading Run-time database

- v **dbLoadDatabase(“testAppLib.dbd”)**
 - v **Loads database definitions into memory**
 - v **Record-related definitions**
 - v **Device-related definitions**
 - v **Driver-related definitions**
- v **dbLoadRecords(“test.db”)**
 - v **Loads database record instances into memory**



iocInit() – the Entry Point

- ✓ **iocInit() initializes iocCore.**
- ✓ **base/src/db/iocInit.c**
 - ✓ **If you take a look of it, you will get the picture on how the EPICS world comes up.**
- ✓ **When iocInit() returns, iocCore has been up and running.**
- ✓ **Sevral tasks are spawned in the initialization sequence.**



EPICS Tasks

- v Channel Access related tasks
- v Scan tasks
- v General purpose callback tasks
- v Watchdog task
- v Error handling task
- v ...



Working on Lessons

- v In, /export/home/odagiri/shanghaiApp/src you have the files named
Lesson_01 ~ Lesson_12
Lesson_extra_01 ~ Lesson_extra_04.
- v They explains how to work on the exercises.
- v Please follow the instructions.