Porting iocCore onto μ-ITRON real-time kernel

EPICS collaboration meeting 2004
12/10/2004

G.Jiang, A.Akiyama, J.Odagiri, N.Yamamoto
What is μ-ITRON?

- An API-specification of real-time kernel
  - Developed by Prof. Ken Sakamura of Tokyo Univ.
  - Many different implementations of products are available on the market
  - Widely used in consumer electronics fields in Japan
    - Mobile phones
    - TV-games
    - Automobile Navigation System
- Details: See
  - http://tron.um.u-tokyo.ac.jp/TRON/ITRON/home-e.html
Why $\mu$-ITRON?

Running iocCore on intelligent device controllers

- Many intelligent device controllers come with an Ethernet interface
- Many of those have an SHx CPU and enough ROM/RAM capacity to run iocCore
- Many of those run $\mu$-ITRON
Intelligent device controllers running μ-ITRON

Commercial Devices

- PLC
  - Yokogawa FA-M3, used for ion-source, vacuum etc.
  - WE7000 measurement station
    - Digital oscilloscope (100kS/s, 1MS/s, 100MS/s and 1GS/s)

Custom Device Controllers

- EMB-LAN100
  - Initially designed for the Power supplies of DTL Q-magnet
- NDIM (Network Device Interface Module)
  - Developed by RIKEN for various control/monitoring
Commercial devices

**FA-M3 PLC**
(Made by Yokogawa)

**WE7000**
(Made by Yokogawa)
Custom device controllers

**EMB-LAN100**  
*(Developed by KEK)*

**N-DIM**  
*(Developed by RIKEN)*
What are the benefits?

- More distributed
  - More load distribution
  - More robust

- Flatter hierarchy
  - Save IOCs being used as “protocol transformer”
  - No need for asynchronous driver
What’s available on the market

- The kernel, NORTi4, from MISPO
- Development environment
  - SHC (Super Hitachi Compiler) with a standard C/C++ library
  - Cygwin on Windows
- Kasago TCP/IP + BSD socket library, from Elmic Systems, Inc.
- We assume BSPs are available from the HW-manufactures
How hard is it?

- We have implemented OSD libraries (not yet tested)
- We have asked Elmic Systems, Inc. to port the TCP/IP + BSD socket onto our target board
- There are some missing functions in STC library needed by iocCore
- Recent versions of EPICS base use lots of C++ codes
  - We have gotten linkage errors with some of those
μ-ITRON has abundant functions for synchronization and communication

It’s relatively easy

For example:

**itrton case:**
```c
const T_CMTX cmtx = {TA_TINHERIT,0,""};
struct epicsMutexOSD *
epicsMutexOsdCreate(void)
{
    ID id;
    ER ercd;
    ercd = acre_sem(&cmtx);
    ...
}
```

**Vxworks case:**
```c
struct epicsMutexOSD *
epicsMutexOsCreate( void )
{
    return((struct epicsMutexOSD *)
    semMCreate(SEM_DELETE_SAFE|SEM_INVERSION_SAFE|SEM_Q_PRIORITY));
}
```
Elmic Systems, Inc. is going to develop libraries which interface their TCP/IP protocol stack with μ-ITRON/LAN controller driver.
Missing functions in STC library

```c
extern FILE *fdopen(int handle, const char * tmp) {...}
extern char *getenv (char * tmp) {...}
extern int putenv (char * tmp) {...}
extern char *tmpnam (char * tmp) {...}
extern FILE *tmpfile (void) {...}
extern int atexit (void (*function)(void)) {...}
extern void exit(int status) {...}
void abort(void) {...}
```
Building problems with C++ code

- iocCore uses some methods not in SHC’s STD C++ library
  - logic_error::logic_error();

- Some of external references can’t be resolved
  - Related with class
  - Related with template
  - Related with inline function
Status and future plans

- We are trying to solve the linkage problems with C++ code.
- If not, we are going to try another compiler, exeGCC, based on gcc.
  - We lose support from the companies on BSP&BSD socket library.
- We hope we can succeed in running iocCore on a target board before Mar. 2005.
MCU made by Nichizou

MCU running μ-ITRON

SH4 (SH7751R-200 MHz)

FLASH ROM 16MB
SDRAM 64MB