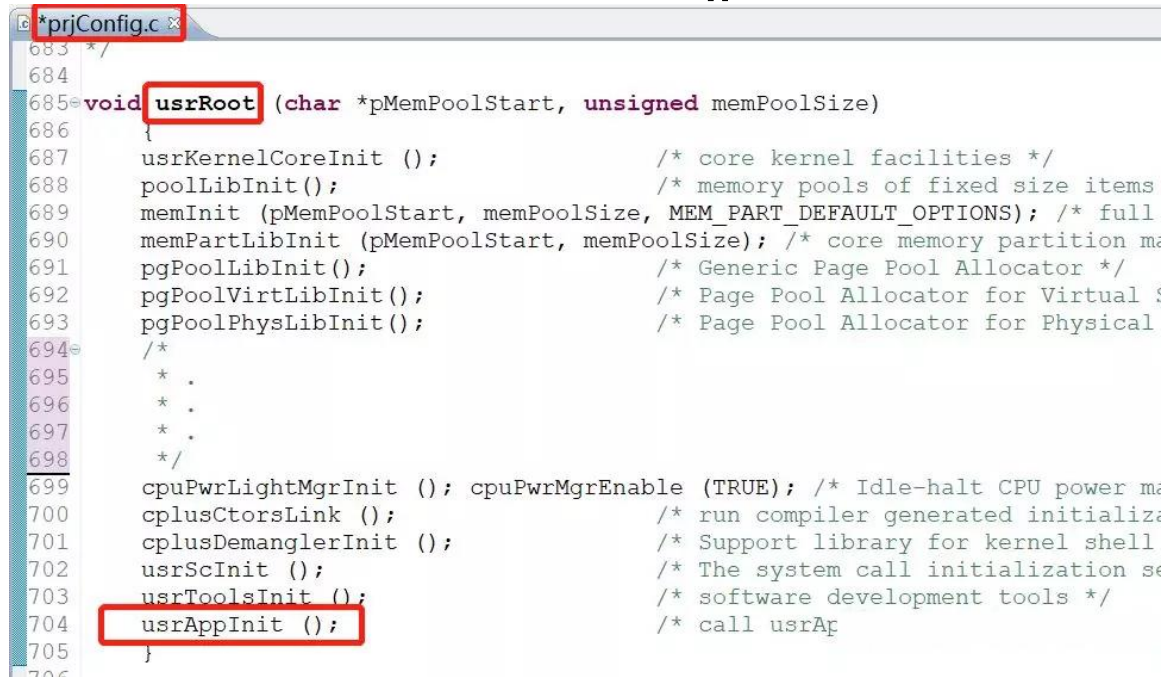


## VxWorks 的应用程序自启动的方法

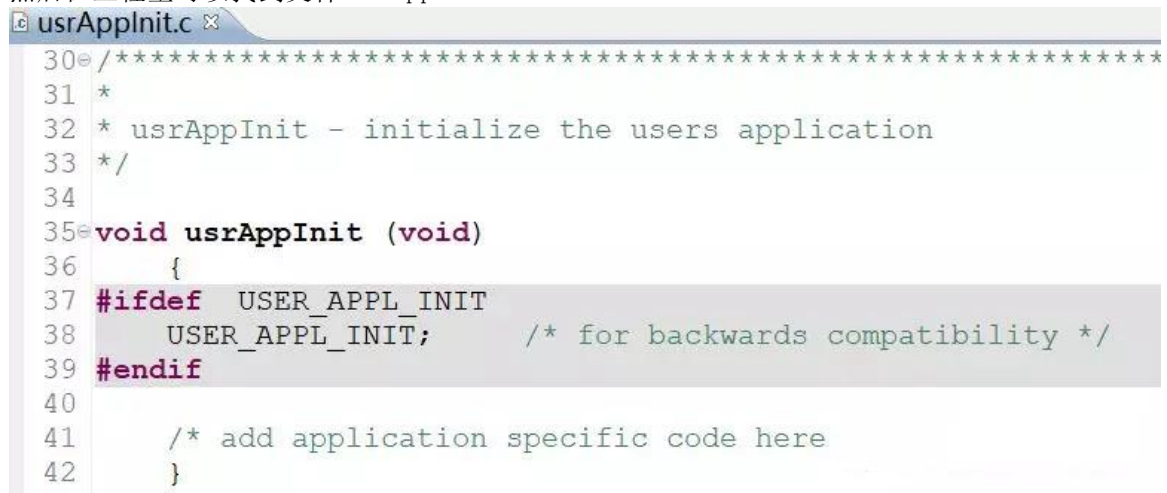
我们知道，要想在 VxWorks 里启动应用程序 (APP/Application/Task+RTP)，常见的方法是使用 `taskSpawn()` / `rtpSpawn()` 来创建并激活相应的 Task/RTP。那我们的第一条 `taskSpawn()` / `rtpSpawn()` 应该放在哪呢？也就是如何让 VxWorks 系统自动启动 APP 呢？只要找到 VxWorks 系统启动时执行的最后一个函数，让这个函数来调用我们自己的 APP 就可以了。

打开工程目录里的 `prjConfig.c` 文件，可以看到 VxWorks 的第一个 `TaskRootTask` 的入口函数 `usrRoot()`。而 `usrRoot()` 调用的最后一个函数是 `usrAppInit()`



```
683 */
684
685 void usrRoot (char *pMemPoolStart, unsigned memPoolSize)
686 {
687     usrKernelCoreInit ();          /* core kernel facilities */
688     poolLibInit();                /* memory pools of fixed size items
689     memInit (pMemPoolStart, memPoolSize, MEM_PART_DEFAULT_OPTIONS); /* full
690     memPartLibInit (pMemPoolStart, memPoolSize); /* core memory partition m
691     pgPoolLibInit();              /* Generic Page Pool Allocator */
692     pgPoolVirtLibInit();          /* Page Pool Allocator for Virtual
693     pgPoolPhysLibInit();         /* Page Pool Allocator for Physical
694     /*
695     *
696     *
697     *
698     */
699     cpuPwrLightMgrInit (); cpuPwrMgrEnable (TRUE); /* Idle-halt CPU power m
700     cplusCtorsLink ();           /* run compiler generated initializ
701     cplusDemanglerInit ();       /* Support library for kernel shell
702     usrScInit ();               /* The system call initialization se
703     usrToolsInit ();            /* software development tools */
704     usrAppInit ();              /* call usrAp
705 }
```

然后在工程里可以找到文件 `usrAppInit.c`



```
30 /******
31 *
32 * usrAppInit - initialize the users application
33 */
34
35 void usrAppInit (void)
36 {
37 #ifdef USER_APPL_INIT
38     USER_APPL_INIT; /* for backwards compatibility */
39 #endif
40
41     /* add application specific code here
42 }
```

从注释里可以看出来了，我们的 APP 可以从这里开始。  
写个小程序试试，例如 `printf()`

```

#include <stdio.h>
#include <taskLib.h>
/*****
 *
 * usrAppInit - initialize the users application
 */

void usrAppInit (void)
{
#ifdef USER_APPL_INIT
    USER_APPL_INIT;    /* for backwards compatibility */
#endif

    /* add application specific code here */
    int pri;
    TASK_DESC desc;
    taskPriorityGet(0, &pri);
    taskInfoGet(0, &desc);
    printf ("\n I'm task %s, priority is %d, stack size is 0x%x\n\n",
            taskName(0), pri, desc.td_stackSize);
}

```

启动 VxWorks，看看效果

```

UUUUUUUU    UUUUUUUU    tttt    iiii    11111111
U:::U:::U    U:::U:::U    ttt::t    i:::i    1:::1
U:::U:::U    U:::U:::U    t:::t    iiii    1:::1
UU:::U:::U    U:::UU    t:::t    l:::1
U:::U:::U    U:::U    ttttttt:::tttttt    iiiiii    1:::1
U:::U:::D    D:::U    t:::t:::t:::t:::t    i:::i    1:::1
U:::U:::D    D:::U    t:::t:::t:::t:::t    i:::i    1:::1
U:::U:::D    D:::U    tttttt:::ttttt    i:::i    1:::1
U:::U:::D    D:::U    t:::t    i:::i    1:::1
U:::U:::D    D:::U    t:::t    i:::i    1:::1
U:::U:::D    D:::U    t:::t    i:::i    1:::1
U:::U:::U    U:::U    t:::t    tttttt    i:::i    1:::1
U:::U:::UUU:::U    t:::t:::t:::t:::ti:::i:::1
UU:::U:::UU    tt:::t:::t:::ti:::i:::1
UU:::U:::UU    tt:::t:::t:::ti:::i:::1
UUUUUUUU    tttttttttt    iiiiii11111111
OS: VxWorks 6.9. BSP: VMWare - Yang.
CPU: VMWARE. Memory Size: 0xfefa000 (~254Mb).
Created: Mar 5 2019 11:51:15
ED&R Policy Mode: Deployed
WDB_COMM_END Ready.

I'm task tRootTask, priority is 0, stack size_is 0x2400
-> _

```

可以看到 APP 的 printf() 执行了，不过它是运行在 tRootTask 的上下文里的。这样不好，一来它的优先级太高；二来出了问题后，影响太大。所以我们的 APP 应该由 usrAppInit() 中的 taskSpawn() 来启动，例如

```

#include <stdio.h>
#include <taskLib.h>

void test()
{
    int pri;
    TASK_DESC desc;
    taskPriorityGet(0,&pri);
    taskInfoGet(0,&desc);
    printf("\n I'm task %s, priority is %d, stack size is 0x%x\n\n",
        (int)taskName(0),pri,desc.td_stackSize,0,0,0,0,0,0);
}

/*****
 *
 * usrAppInit - initialize the users application
 */

void usrAppInit (void)
{
#ifdef USER_APPL_INIT
    USER_APPL_INIT;    /* for backwards compatibility */
#endif

    /* add application specific code here */
    taskSpawn("tTest",100,0,0x1000,(FUNCPTR)test,0,
}

```

这样就可以看了

```

UUUUUUUU  UUUUUUUU  tttt      iiii 11111111
U:::U:::U  U:::U:::U  ttt::t    i:::i 1:::1:::1
U:::U:::U  U:::U:::U  t:::t::t   iiii 1:::1:::1
UU:::U:::U  U:::UU    t:::t::t   iiii 1:::1:::1
U:::U:::U  U:::U:::U tttttttt:::ttttttt  iiii 1:::1:::1
U:::U:::D  D:::U:::U t:::t:::t:::t:::t  i:::i 1:::1:::1
U:::U:::D  D:::U:::U t:::t:::t:::t:::t  i:::i 1:::1:::1
U:::U:::D  D:::U:::U tttttt:::ttttttt  i:::i 1:::1:::1
U:::U:::D  D:::U:::U  t:::t::t        i:::i 1:::1:::1
U:::U:::D  D:::U:::U  t:::t::t        i:::i 1:::1:::1
U:::U:::D  D:::U:::U  t:::t::t        i:::i 1:::1:::1
U:::U:::U  U:::U:::U  t:::t::t  tttttt i:::i 1:::1:::1
U:::U:::UUU:::U:::U  t:::t::ttt:::ti:::i 1:::1:::1
UU:::U:::U:::UU      tt:::t:::t:::ti:::i 1:::1:::1
UU:::U:::U:::UU      tt:::t:::t:::ti:::i 1:::1:::1
UUUUUUUUUU          tttttttttt  iiii 11111111111111
OS: VxWorks 6.9.  BSP: UMWare - Yang.
CPU: UMWARE.  Memory Size: 0xfefa000 (~254Mb).
Created: Mar  5 2019 12:04:19
ED&R Policy Mode: Deployed
WDB_COMM_END Ready.
->
I'm task tTest, priority is 100, stack size

```

usrRoot()之所以会调用 usrAppInit(), 是因为包含了组件 INCLUDE\_USER\_APPL

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li> <ul style="list-style-type: none"> <li>RTP Startup Facility: Bootline-encoded RTP list</li> <li>RTP Startup Facility: Command shell startup script</li> <li>RTP Startup Facility: String-encoded RTP list</li> <li>RTP Startup Facility: User-defined code</li> <li><b>application initialization (default)</b></li> <li>post-kernel application initialization</li> <li>pre-kernel application initialization</li> <li>pre-network application initialization</li> </ul> </li> </ul> | <b>FOLDER_APPLICATION</b><br>INCLUDE RTP_APPL_INIT_E<br>INCLUDE RTP_APPL_INIT_C<br>INCLUDE RTP_APPL_INIT_S<br>INCLUDE RTP_APPL_USER<br><b>INCLUDE_USER_APPL</b><br>INCLUDE_USER_POST_KERN<br>INCLUDE_USER_PRE_KERN<br>INCLUDE_USER_PRE_NETWORK |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

可以看到还有几个组件与 startup 相关，有兴趣的话，我们试试其它组件

INCLUDE RTP\_APPL\_USER

## Components

| Component Configuration                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                                                                                                                                                                                                                                   |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | Name                                                                                                                                                                                                                                              |
| <ul style="list-style-type: none"> <li> <ul style="list-style-type: none"> <li>RTP Startup Facility: Bootline-encoded RTP list</li> <li>RTP Startup Facility: Command shell startup script</li> <li>RTP Startup Facility: String-encoded RTP list</li> <li><b>RTP Startup Facility: User-defined code</b></li> <li><b>application initialization (default)</b></li> <li>post-kernel application initialization</li> <li>pre-kernel application initialization</li> <li>pre-network application initialization</li> </ul> </li> </ul> | <b>FOLDER_APPLICATION</b><br>INCLUDE RTP_APPL_INIT_BOOT<br>INCLUDE RTP_APPL_INIT_CMD_<br>INCLUDE RTP_APPL_INIT_STRIN<br><b>INCLUDE RTP_APPL_USER</b><br><b>INCLUDE_USER_APPL</b><br>INCLUDE_USER_POST_KERNEL_<br>INCL<br>INCLUDE_USER_PRE_NETWORK |

```

720  cpuPwrLightMgrInit (); cpuPwrMgrEnable (TRUE); /* Idle-halt CPU power management */
721  cplusCtorsLink (); /* run compiler generated initialization function */
722  cplusDemanglerInit (); /* Support library for kernel shell and loader: */
723  usrScInit (); /* The system call initialization sequence */
724  usrToolsInit (); /* software development tools */
725  usrAppInit (); /* call usrAppInit() in your usrAppInit.c project */
726  usrRtpAppInit (); /* Launch RTP from */
727  }

```

```

usrRtpAppInit.c
20 /*****
21  *
22  * usrRtpAppInit - initialize the users RTP applications.
23  */
24
25 void usrRtpAppInit (void)
26 {
27     /* TODO - add your own application lau
28     }

```

可以看到，包含了这个组件后，工程里多出来一个与 `usrAppInit()` 类似的 `usrRtpAppInit()`。从注释来看，它是用来启动 RTP 的。不过它俩默认都是空函数，没什么本质区别的。也就是说，在 `usrRtpAppInit()` 里来启动 Kernel Task 也是没问题的。

#### INCLUDE\_STARTUP\_SCRIPT

|                                          |                         |
|------------------------------------------|-------------------------|
| kernel shell components                  | FOLDER_SHELL            |
| > shell commands                         | FOLDER_SHELL_CMD        |
| > shell editing mode (default)           | SELECT_SHELL_EDIT_MODE  |
| > shell interpreter selection (default)  | SELECT_SHELL_INTERP     |
| > debugging facilities (default)         | INCLUDE_DEBUG           |
| < kernel shell startup script            | INCLUDE_STARTUP_SCRIPT  |
| maximum retry limit for opening script   | OPEN_SCRIPT_MAX_RETRY_C |
| < shell banner (default)                 | INCLUDE_SHELL_BANNER    |
| < shell history saving/loading mechanism | INCLU                   |
| < shell hooks                            | INCLUDE_SHELL_HOOKS     |

这个组件在另一个目录里，包含之后，`usrRoot()` 会在调用 `usrShell()` 之前解析并执行 bootrom 的 startup script 参数

```

prjConfig.c
408
409 void usrShellInit (void)
410 {
411     shellLibInit (); /* Handles the shell core files */
412     { extern size_t dbgLibTrcBufSz; dbgLibTrcBufSz = DEBUG_STACK_TRACE_BUF_SIZE;
413     vxdbgRtpLibInit (); /* Initialize process debugging library */
414     usrBanner (); /* Display the WRS banner on startup */
415     ledModeRegister (viLedLibInit); /* Editing mode similar to the Vi editor */
416     shellInterpRegister (shellInterpCInit); /* The C interpreter for the kernel */
417     shellInterpRegister (shellInterpCmdInit); /* The command interpreter for the kernel */
418     usrShellCmdInit (); /* The kernel shell commands initialization */
419     usrStartupScript (startupScriptFieldSplit (sysBootParams.startupScript));
420     usrShell (); /* Interpreter
421 }

```

```
[VxWorks Boot]: c

'.' = clear field;  '-' = go to previous field;

boot device          : lnPci0
processor number     : 0
host name            : host
file name            : vxworks
inet on ethernet (e) : 192.168.11.111:ffffff00
inet on backplane (b):
host inet (h)        : 192.168.11.1
gateway inet (g)     :
user (u)             : vm
ftp password (pw) (blank = use rsh): vm
flags (f)            : 0x8
target name (tn)     : vm
startup script (s)   :
```

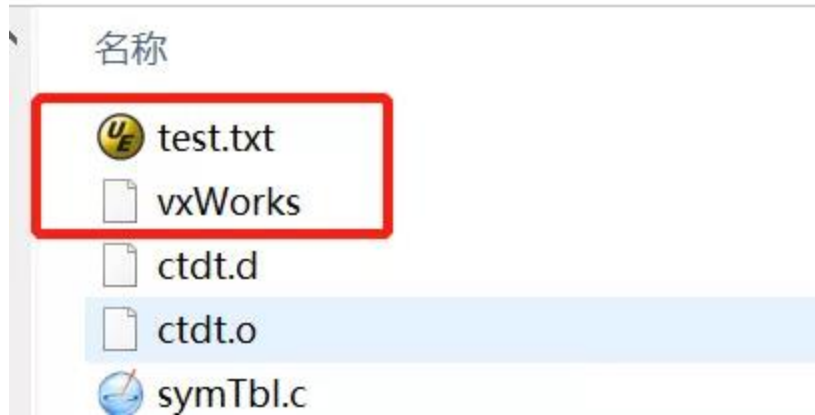
我们写一个 script 文件来试试这个功能：

新建一个文件，为了方便编辑，可以把文件后缀写作 txt，在文件中添加几行 **可以在 VxWorks 的 Kernel Shell 里执行的语句**。我们之前定义了一个函数 test()，这次也调用它



把这个文件存放到 VxWorks 文件的目录里

> default



修改 bootrom 的 startup script 参数

```
[UxWorks Boot]: c
'. ' = clear field; '-' = go to previous field; ^D = quit

boot device      : lnPci0
processor number  : 0
host name        : host
file name        : vxworks
inet on ethernet (e) : 192.168.11.111:ffffff00
inet on backplane (b):
host inet (h)     : 192.168.11.1
gateway inet (g)  :
user (u)          : vm
ftp password (pw) (blank = use rsh): vm
flags (f)         : 0x8
target name (tn)  : vm
startup script (s) : test.txt
```

启动 VxWorks

```

UU::::::::::::UU          tt::::::::::::ti::::il::
UU::::::::::::UU          tt::::::::::::tti::::il::
UUUUUUUUUU          tttttttttttt  iiiiiiiiiiii
OS: VxWorks 6.9.  BSP: VMWare - Yang.
CPU: VMWARE.  Memory Size: 0xfefa000 (~254Mb).
Created: Mar  5 2019 14:43:40
ED&R Policy Mode: Deployed
WDB_COMM_END Ready.
Executing startup script 'test.txt'...

printf "\nhello world!"

hello world!value = 13 = 0xd
1+2
value = 3 = 0x3
test

I'm task tShell0, priority is 1, stack size is 0x10000

value = 58 = 0x3a = ':'

Done executing startup script 'test.txt'.

->

```

可以看到，在 Kernel Shell 的提示符“->”出现之前，VxWorks 解析并执行了 test.txt 中的语句。而且这些语句是在 Kernel Shell 的上下文里执行的。因此，最好也使用 taskSpawn() 来创建自己的 Task 环境。

#### INCLUDE\_RTP\_APPL\_INIT\_BOOTLINE

| application components                                                                                                                                                                                                                                                                                                                                                                          | FOLDER_APPLICATION                                                                                                                                |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>RTP Startup Facility: Bootline-encoded RTP list</b>                                                                                                                                                                                                                                                                                                                                          | <b>INCLUDE_RTP_APPL_INIT_BOOTLINE</b>                                                                                                             |
| <ul style="list-style-type: none"> <li> RTP Startup Facility: Command shell startup script           <ul style="list-style-type: none"> <li> Command shell script file.</li> </ul> </li> <li> RTP Startup Facility: String-encoded RTP list           <ul style="list-style-type: none"> <li> String-encoded RTP list.</li> </ul> </li> <li> RTP Startup Facility: User-defined code</li> </ul> | INCLUDE_RTP_APPL_INIT_CMD_SHELL_S...<br>RTP_APPL_CMD_SCRIPT_FILE<br>INCLUDE_RTP_APPL_INIT_STRING<br>RTP_APPL_INIT_STRING<br>INCLUDE_RTP_APPL_USER |
| <b>application initialization (default)</b>                                                                                                                                                                                                                                                                                                                                                     | <b>INCLUDE_USER----</b>                                                                                                                           |
| post-kernel application initialization                                                                                                                                                                                                                                                                                                                                                          | INCLUDE_USER_POST_KERNEL_APPL_INIT<br>INCLUDE_USER_POST_KERNEL_APPL_INIT                                                                          |

包含这个组件后，usrRoot() 会在 usrAppInit() 之后再调用一个函数 usrRtpAppInitBootline()。

```

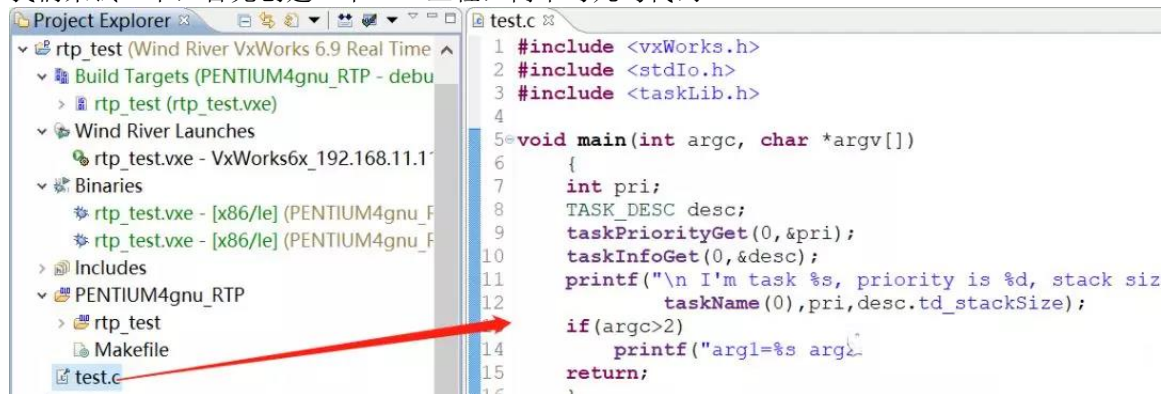
prjConfig.c
723     usrScInit ();
724     usrToolsInit ();
725     usrAppInit ();
726     usrRtpAppInitBootline ();
727 }

```

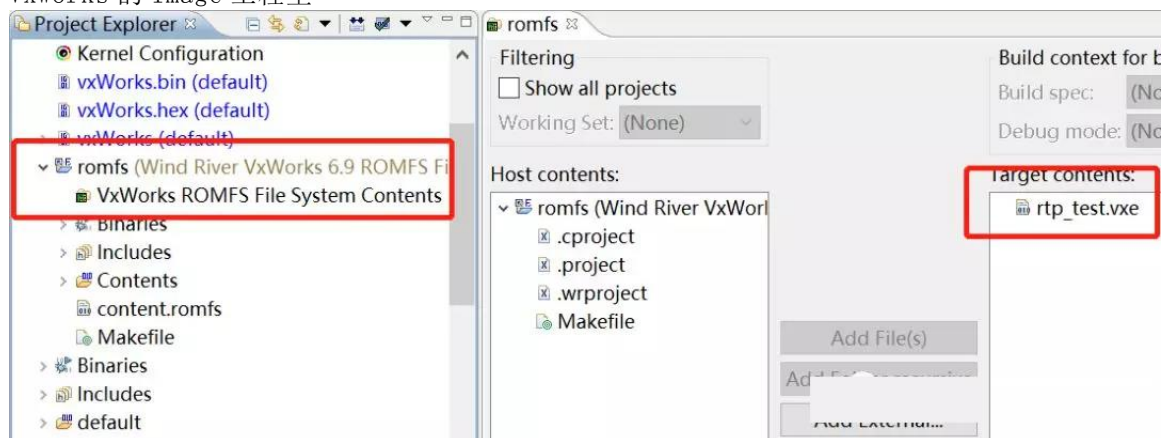
/\* The system call initialization  
 /\* software development tools \*/  
 /\* call usrAppInit() (in your user  
 /\* Launch R...

这个函数的作用也是解析 startup script，不过它解析的不是 script 里的文件，而是直接使用 startup script 指定的 RTP 文件。格式是：#RTP 文件 1^参数 1^参数 2...#RTP 文件 2... 即符号#之后是 RTP 的文件名，符号^之后是 RTP 的入参。有几个 RTP 就用几个#，有几个参数就用几个^

我们来试一下，首先创建一个 RTP 工程，简单写几句代码



新建一个 ROMFS 工程，用于存放 RTP 工程编译后的 rtp\_test.vxe，然后把 ROMFS 工程放到 VxWorks 的 Image 工程里



修改 bootrom 的 startup script 参数，可以保留之前的 script 文件，不过要放到第一个#之前。并给 RTP 传递两个参数：x 和 y

```
boot device      : lnPci
unit number      : 0
processor number  : 0
host name        : host
file name        : vxworks
inet on ethernet (e) : 192.168.11.111:ffffff00
host inet (h)     : 192.168.11.1
user (u)         : vm
ftp password (pw) : vm
flags (f)        : 0x8
target name (tn)  : vm
startup script (s) : test.txt#/romfs/rtp_test.vxe x y
```

这次 script 文件 test.txt 的内容如下



启动 VxWorks，看看得到了什么



VxWorks 在打印 Kernel Shell 提示符 “->” 之前执行了 test.txt 里的语句；在打印 “->” 之后，又启动了 startup script 里指定的 RTP。  
这就是 INCLUDE\_RTP\_APPL\_INIT\_BOOTLINE 的作用：执行 bootrom 参数 startup script 中符号#之后的 RTP

INCLUDE\_RTP\_APPL\_INIT\_CMD\_SHELL\_SCRIPT

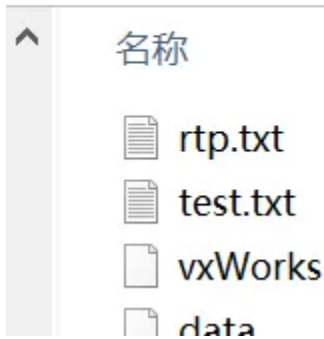
### Components

| Component Configuration                            |                  |        |           |
|----------------------------------------------------|------------------|--------|-----------|
| Description                                        | Name             | Type   | Value     |
| application components                             | FOLDER_APP...    |        |           |
| RTP Startup Facility: Bootline-encoded RTP list    | INCLUDE_RTP...   |        |           |
| RTP Startup Facility: Command shell startup script | INCLUDE_RTP...   |        |           |
| Command shell script file.                         | RTP_APPL_C...    | string | "rtp.txt" |
| RTP Startup Facility: String-encoded RTP list      | INCLUDE_RTP_...  |        |           |
| String-encoded RTP list.                           | RTP_APPL_INIT... | string | ""        |
| RTP Startup Facility: User-defined code            | INCLUDE_R        |        |           |
| application initialization (default)               | INCLUDE_USE...   |        |           |

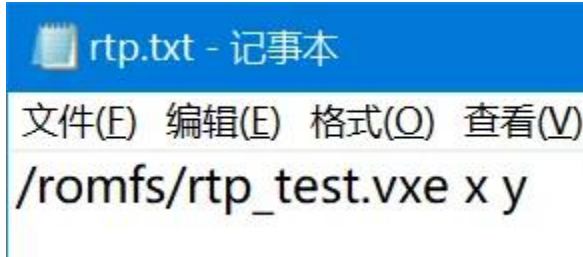
这个组件也是在 VxWorks 启动 Kernel Shell 之后来启动 RTP 的。不过它的输入是由参数 RTP\_APPL\_CMD\_SCRIPT\_FILE 来指定的文件，而且它解析文件时，用的是 VxWorks6 特有的 Command Interpreter。

例如我们在 VxWorks 镜像文件的目录里新建一个 rtp.txt 文件，并把这个文件名赋值给参数 RTP\_APPL\_CMD\_SCRIPT\_FILE

> default



文件里放一条 Command Interpreter 能识别的语句



为了看的更清晰，我们先把 bootrom 的参数 startup script 清空

```
[VxWorks Boot]: c
'. ' = clear field;  '-' = go to previous field;  ^D = quit

boot device           : lnPci0
processor number      : 0
host name             : host
file name             : vxworks
inet on ethernet (e)  : 192.168.11.111:ffffff00
inet on backplane (b):
host inet (h)         : 192.168.11.1
gateway inet (g)      :
user (u)              : vm
ftp password (pw) (blank = use rsh): vm
flags (f)             : 0x8
target name (tn)      : vm
startup script (s)    :
```

然后启动 VxWorks

```

U:::U D:::U t:::t i:::i l:::l
U:::U D:::U t:::t i:::i l:::l
U:::U U:::U t:::t tttttt i:::i l:::l
U:::U U:::U t:::t tttttt i:::i l:::l
UU:::UUU:::U t:::t tttttt i:::i l:::l
UU:::UUU:::U tt:::tt i:::i l:::l
UU:::UUU:::U tt:::tt i:::i l:::l
UUUUUUUUU tttttttttt iiii1111111111
OS: VxWorks 6.9. BSP: UMWare - Yang.
CPU: UMWARE. Memory Size: 0xfefa000 (~254Mb).
Created: Mar 5 2019 16:34:25
ED&R Policy Mode: Deployed
WDB_COMM_END Ready.

-> Executing startup script 'rtp.txt'...

/romfs/rtp_test.vxe x y
Launching process '/romfs/rtp_test.vxe' ...
Process '/romfs/rtp_test.vxe' (process Id = 0x1643010) launched.

I'm task iRtp_test, priority is 220, stack size is 12445312
arg1=x arg2=y

```

可以看到，组件 INCLUDE\_RTP\_APPL\_INIT\_CMD\_SHELL\_SCRIPT 的作用就是使用 Command Interpreter 来解析 RTP\_APPL\_CMD\_SCRIPT\_FILE 指定的文件

## INCLUDE\_RTP\_APPL\_INIT\_STRING Components

| Component Configuration                            |                                   |        |           |
|----------------------------------------------------|-----------------------------------|--------|-----------|
| Description                                        | Name                              | Type   | Value     |
| application components                             | FOLDER APPLICATION                |        |           |
| RTP Startup Facility: Bootline-encoded RTP list    | INCLUDE RTP_APPL_INIT_BOOTLINE    |        |           |
| RTP Startup Facility: Command shell startup script | INCLUDE RTP_APPL_INIT_CMD_SH...   |        |           |
| Command shell script file.                         | RTP_APPL_CMD_SCRIPT_FILE          | string | "rtp.txt" |
| RTP Startup Facility: String-encoded RTP list      | INCLUDE RTP_APPL_INIT_STRING      |        |           |
| String-encoded RTP list.                           | RTP_APPL_INIT_STRING              | string | ""        |
| RTP Startup Facility: User-defined code            | INCLUDE RTP_APPL_USER             |        |           |
| application initialization (default)               | INCLUDE_USER_APPL                 |        |           |
| post-kernel application initialization             | INCLUDE_USER_POST_KERNEL_APPL ... |        |           |

这个组件的作用类似于前文中的 INCLUDE\_RTP\_APPL\_INIT\_BOOTLINE，解析的格式也是一样的。不过它的输入不是 bootrom 的 startup script 参数，而是自己的参数 RTP\_APPL\_INIT\_STRING。

我们修改一下这个参数试试，同样为了看的清晰，我们把上个例子中的 RTP\_APPL\_CMD\_SCRIPT\_FILE 置为空，并把 RTP 的参数改为 a 和 b

| Component Configuration                            |                                 |        |                           |
|----------------------------------------------------|---------------------------------|--------|---------------------------|
| Description                                        | Name                            | Type   | Value                     |
| application components                             | FOLDER APPLICATION              |        |                           |
| RTP Startup Facility: Bootline-encoded RTP list    | INCLUDE RTP_APPL_INIT_BOOTLINE  |        |                           |
| RTP Startup Facility: Command shell startup script | INCLUDE RTP_APPL_INIT_CMD_SH... |        |                           |
| Command shell script file.                         | RTP_APPL_CMD_SCRIPT_FILE        | string | ""                        |
| RTP Startup Facility: String-encoded RTP list      | INCLUDE RTP_APPL_INIT_STRING    |        |                           |
| String-encoded RTP list.                           | RTP_APPL_INIT_STRING            | string | "/romfs/rtp_test.vxe^a^b" |
| RTP Startup Facility: User-defined code            | INCLUDE RTP_APPL_USER           |        |                           |
| application initialization (default)               | INCLUDE_USER_APPL               |        |                           |

启动 VxWorks

```
U:::::U   U:::::U           t:::::t   tttttt i::::i  l::::l
U:::::UUU:::::U           t:::::ttttt:::::ti:::::il:::::l
UU:::::UU           tt:::::ttti:::::il:::::l
  UU:::::UU           tt:::::ttti:::::il:::::l
    UUUUUUUU           tttttttttt  iiiiilllllllll
OS: VxWorks 6.9.  BSP: UMWare - Yang.
CPU: UMWARE.  Memory Size: 0xfefa000 (~254Mb).
Created: Mar  5 2019 16:58:26
ED&R Policy Mode: Deployed
WDB_COMM_END Ready.

Spawning RTP: /romfs/rtp_test.vxe
->
I'm task iRtp_test, priority is 220, stack size is 12445352
arg1=a arg2=b
```

总结一下

|   | Component                              | Application Position              | Interpreter               |
|---|----------------------------------------|-----------------------------------|---------------------------|
| 1 | INCLUDE_USER_APPL                      | usrApplInit.c/usrApplInit()       | C                         |
| 2 | INCLUDE_RTP_APPL_USER                  | usrRtpApplInit.c/usrRtpApplInit() | C                         |
| 3 | INCLUDE_STARTUP_SCRIPT                 | bootrom/startup script            | Kernel Shell              |
| 4 | INCLUDE_RTP_APPL_INIT_BOOTLINE         | bootrom/startup script            | #RTP1^arg1^arg2#RTP2^arg1 |
| 5 | INCLUDE_RTP_APPL_INIT_CMD_SHELL_SCRIPT | RTP_APPL_CMD_SCRIPT_FILE          |                           |
| 6 | INCLUDE_RTP_APPL_INIT_STRING           | RTP_APPL_INIT_STRING              | #RTP1^arg1^arg2#RTP2^arg1 |

1、3 用于启动 Task，2、4-6 用于启动 RTP。  
1、2 在代码中；3、4 在引导参数中；5、6 在组件参数中。  
1、2 使用 C 语法；3 使用 Shell 命令，类似于 C 语法；4、6 使用符号#和^；5 使用专用于 RTP 的 Command 命令。