

# [PATCH 3/3] syslet demos – collecting timer expirations and child status

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-05/msg13851.html>

---

- *From:* Jeff Dike <[jdike@xxxxxxxxxxx](mailto:jdike@xxxxxxxxxxx)>
  - *Date:* Thu, 31 May 2007 14:04:31 -0400
- 

wait-sleep uses the `async_exec` mechanism to collect timer expirations and child process wait status. It randomly fires off a number of sleeps and forks, waiting for them to finish, and firing off something new to replace anything that finishes.

Signed-off-by: Jeff Dike <[jdike@xxxxxxxxxxxxxxxxxxx](mailto:jdike@xxxxxxxxxxxxxxxxxxx)>

---  
Makefile | 4 +  
wait-sleep.c | 123 +++  
2 files changed, 127 insertions(+)

Index: `async-test-v5/Makefile`

=====  
--- `async-test-v5.orig/Makefile` 2007-05-31 13:30:17.000000000 -0400  
+++ `async-test-v5/Makefile` 2007-05-31 13:45:35.000000000 -0400  
@@ -17,3 +17,7 @@ `evserver_epoll_threadlet`: `evserver_epoll`

`aio-read`: `aio-read.c` `syslet.h` `sys.h` `Makefile`  
`gcc -O2 -g -Wall -o aio-read aio-read.c -fomit-frame-pointer`  
+  
+`wait-sleep`: `wait-sleep.c` `syslet.h` `sys.h` `Makefile`  
+ `gcc -O2 -g -Wall -o wait-sleep wait-sleep.c -fomit-frame-pointer`  
+

Index: `async-test-v5/wait-sleep.c`

=====  
--- `/dev/null` 1970-01-01 00:00:00.000000000 +0000  
+++ `async-test-v5/wait-sleep.c` 2007-05-31 13:32:39.000000000 -0400  
@@ -0,0 +1,123 @@  
+#include <stdio.h>  
+#include <stdlib.h>  
+#include <errno.h>  
+#include <sys/time.h>  
+#include <sys/wait.h>  
+#include "sys.h"  
+  
+/\*  
+ \* Set up a syslet atom:  
+ \*/

[PATCH 3/3] syslet demos – collecting timer expirations and child status

```
+static void
+init_atom(struct syslet_uatom *atom, int nr,
+ void *arg_ptr0, void *arg_ptr1, void *arg_ptr2,
+ void *arg_ptr3, void *arg_ptr4, void *arg_ptr5,
+ void *ret_ptr, unsigned long flags, struct syslet_uatom *next,
+ void *private)
+{
+ atom->nr = nr;
+ atom->arg_ptr[0] = (u64)(unsigned long)arg_ptr0;
+ atom->arg_ptr[1] = (u64)(unsigned long)arg_ptr1;
+ atom->arg_ptr[2] = (u64)(unsigned long)arg_ptr2;
+ atom->arg_ptr[3] = (u64)(unsigned long)arg_ptr3;
+ atom->arg_ptr[4] = (u64)(unsigned long)arg_ptr4;
+ atom->arg_ptr[5] = (u64)(unsigned long)arg_ptr5;
+ atom->ret_ptr = (u64)(unsigned long)ret_ptr;
+ atom->flags = flags;
+ atom->next = (u64)(unsigned long)next;
+ atom->private = (u64) (u32) private;
+}
+
+struct something {
+ enum { SLEEPER, RUNNER } type;
+ int time;
+ int status;
+};
+
+void start_something(struct syslet_uatom *atom, struct something *s)
+{
+ struct syslet_uatom *done;
+
+ s->time = rand() % 10 + 1;
+ if((rand() % 2) == 0){
+ struct timespec ts = { s->time, 0 }, *timeptr = &ts;
+ struct timespec *rem = NULL;
+
+ s->type = SLEEPER;
+ printf("0x%p : Sleeping for %d seconds\n", atom, s->time);
+ init_atom(atom, __NR_sys_nanosleep, &timeptr, &rem,
+ NULL, NULL, NULL, NULL, NULL, 0, NULL, s);
+ }
+ else {
+ int pid = fork();
+ if(pid < 0){
+ perror("fork");
+ exit(1);
+ }
+ else if(pid > 0){
+ int *stat = &s->status, flags = 0;
+ s->type = RUNNER;
+ init_atom(atom, __NR_sys_waitpid, &pid, &stat,
+ &flags, NULL, NULL, NULL, NULL, 0, NULL, s);
```

[PATCH 3/3] syslet demos – collecting timer expirations and child status

```
+ }
+ else {
+ int status = rand() % 128;
+ sleep(s->time);
+ printf("0x%p : child %d exiting with status %d\n", atom,
+ getpid(), status);
+ exit(status);
+ }
+ }
+ if(async_head.new_thread_stack == 0)
+ async_head.new_thread_stack = thread_stack_alloc();
+ done = sys_async_exec(atom, &async_head);
+ if(done == atom){
+ printf("something finished too fast\n");
+ exit(1);
+ }
+ }
+
+#define ARRAY_SIZE(a) (sizeof(a) / sizeof(a[0]))
+
+int main(int argc, char *argv[])
+{
+ struct something something[ARRAY_SIZE(completion_ring)], *s;
+ struct syslet_uatom atoms[ARRAY_SIZE(completion_ring)], *done;
+ int err, i;
+
+ async_head_init();
+ for(i = 0; i < ARRAY_SIZE(atoms); i++){
+ start_something(&atoms[i], &something[i]);
+ }
+
+ while(1){
+ err = sys_async_wait(1, async_head.user_ring_idx, &async_head);
+ if(err){
+ perror("sys_async_wait");
+ exit(1);
+ }
+ while(completion_ring[async_head.user_ring_idx] != 0){
+ done = (struct syslet_uatom *) (u32)
+ completion_ring[async_head.user_ring_idx];
+ completion_ring[async_head.user_ring_idx++] = 0;
+ async_head.user_ring_idx %= ARRAY_SIZE(completion_ring);
+ s = (struct something *) (u32) done->private;
+ switch(s->type){
+ case SLEEPER:
+ printf("0x%p : Slept for %d seconds\n", done,
+ s->time);
+ break;
+ case RUNNER:
+ printf("0x%p : Slept for %d seconds, then "
+ "exited with status %d\n", done, s->time,
```

[PATCH 3/3] syslet demos – collecting timer expirations and child status

```
+ WEXITSTATUS(s->status));  
+ break;  
+ default:  
+ fprintf(stderr, "Bad something type – %d\n",  
+ s->type);  
+ exit(1);  
+ }  
+ start_something(done, &something[done – atoms]);  
+ }  
+ }  
+}  
–
```

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in  
the body of a message to majordomo@xxxxxxxxxxxxxxxxxxx  
More majordomo info at <http://vger.kernel.org/majordomo-info.html>  
Please read the FAQ at <http://www.tux.org/lkml/>