TP3: Multitarea con desalojo

static_assert

La macro definida en jos se define a base de un switch, en el cual se evalúa la condición del assert. Si esta es 0, se continúa, en caso contrario, se genera un error en tiempo de compilación. Esto se debe a que no se definen todos los casos en el switch.

env_return

Una vez terminada la función umain() de un proceso el kernel retoma la ejecución en la función libmain(), donde llamará a exit(). Esta a su vez llama a sys_env_destroy(), dentro de la cual se ejecuta envid2env para convertir el id de proceso al puntero al mismo. Finalmente se llama a env_destroy, que a su vez ejecuta env_free para desalocar la memoria del proceso y destruir el proceso. La diferencia con el anterior TP, es que en este caso, una vez destruido el proceso, se verifica si el proceso es el mismo al que se estaba ejecutando y si es así, se llama a sched_yield() para que el kernel pueda ejecutar algún otro proceso en condiciones de hacerlo (estado: ENV_RUNNABLE).

sys_yield

```
SMP: CPU 0 found 1 CPU(s)
enabled interrupts: 1 2
[00000000] new env 00001000
[00000000] new env 00001001
[00000000] new env 00001002
Hello, I am environment 00001000.
Hello, I am environment 00001001.
Hello, I am environment 00001002.
Back in environment 00001002, iteration 0.
Back in environment 00001002, iteration 1.
Back in environment 00001002, iteration 2.
Back in environment 00001002, iteration 3.
Back in environment 00001002, iteration 4.
All done in environment 00001002.
[00001002] exiting gracefully
[00001002] free env 00001002
```