Simulador de S.O.

Práctica: Asignacion Continua

Asignación Continua

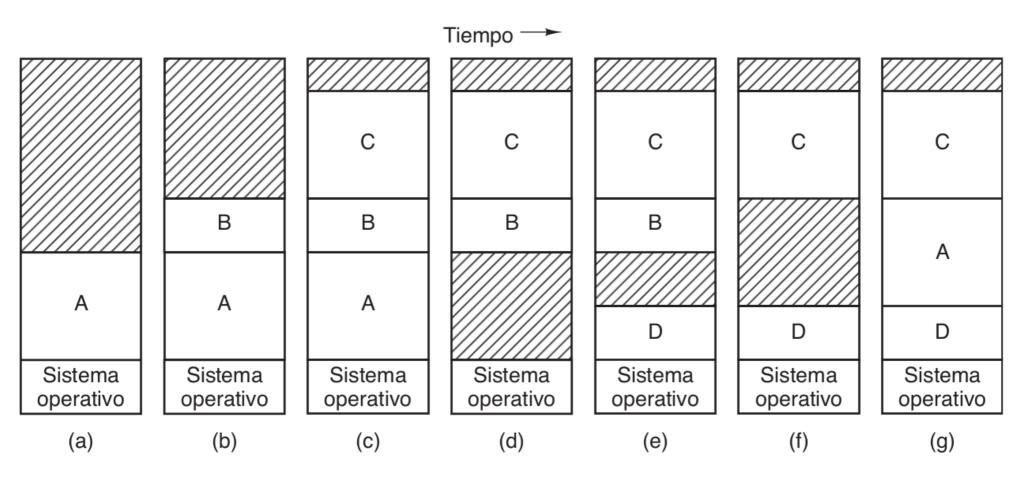
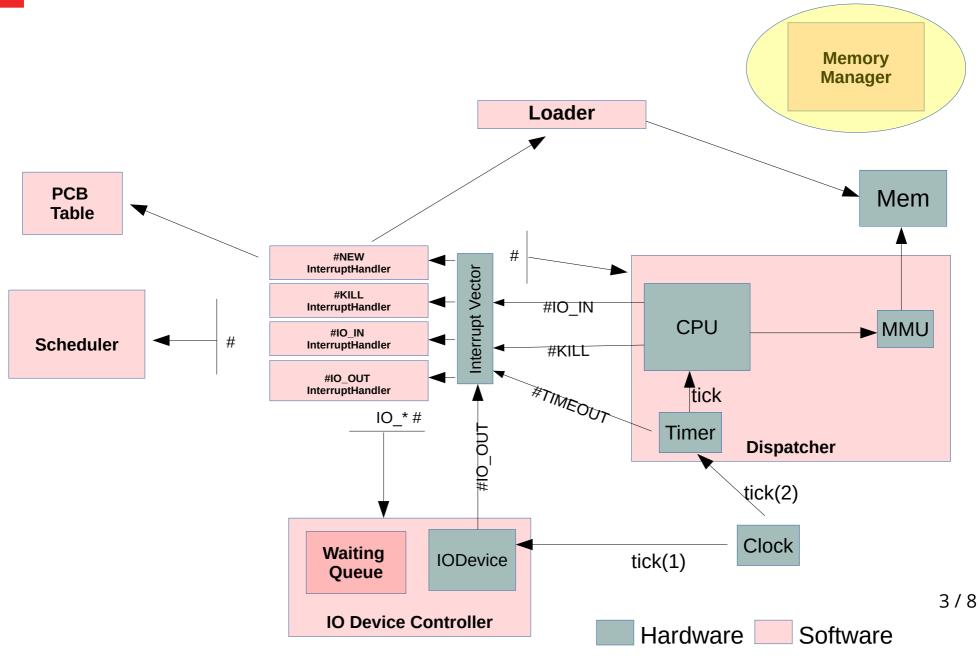
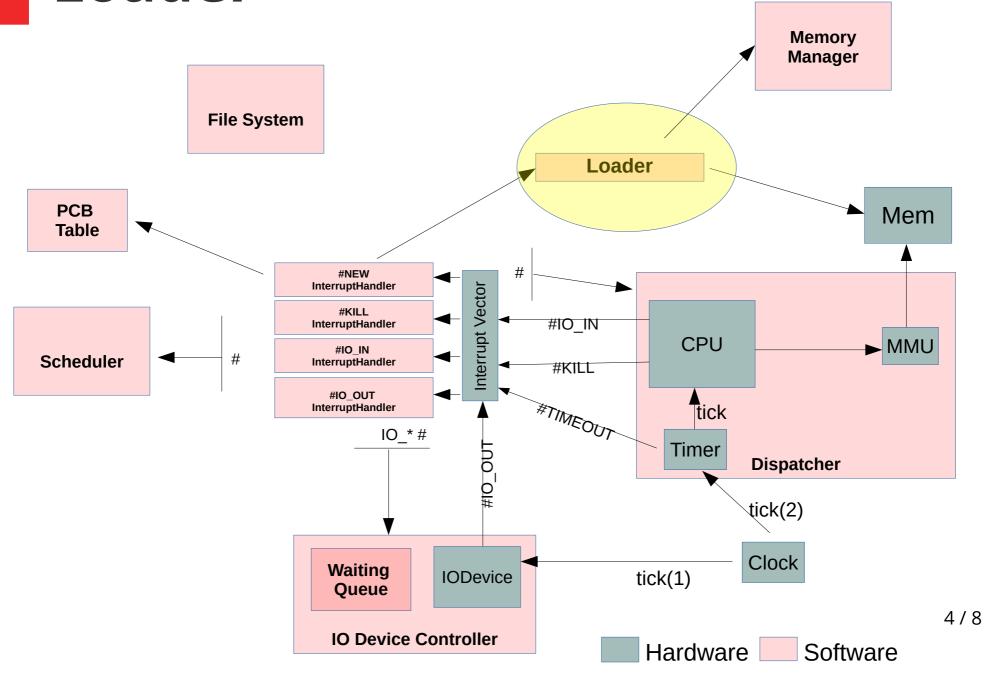


Figura 3-4. La asignación de la memoria cambia a medida que llegan procesos a la memoria y salen de ésta. Las regiones sombreadas son la memoria sin usar.

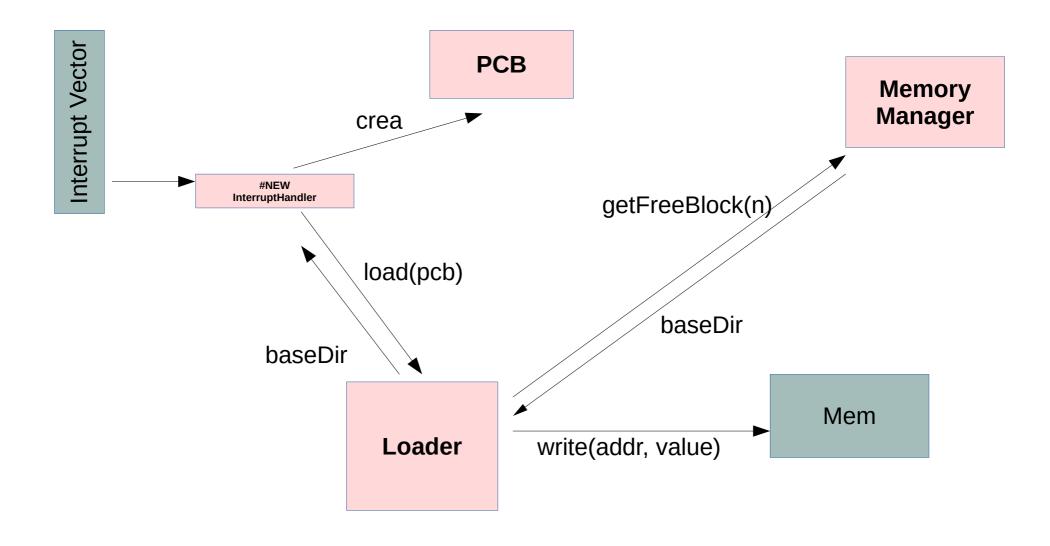
Memory Manager

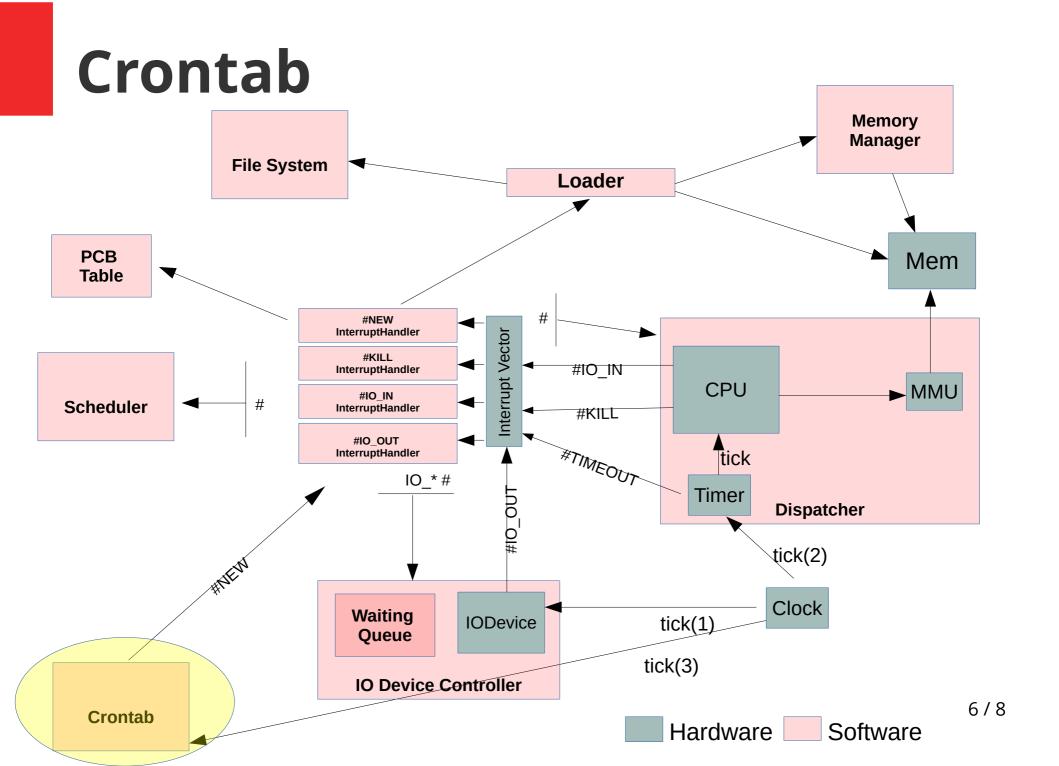


Loader



El #New queda así





Main con Crontab

```
## setup our hardware and set memory size to 2 "cells"
HARDWARE.setup(20)
prg1 = Program("prg1.exe", [ASM.CPU(1)])
prg2 = Program("prg2.exe", [ASM.CPU(3)])
prg3 = Program("prg3.exe", [ASM.CPU(3)])
prg4 = Program("prg4.exe", [ASM.CPU(3)])
prg5 = Program("prg5.exe", [ASM.CPU(5)])
prq6 = Program("prq6.exe", [ASM.CPU(4)])
prq7 = Program("prq7.exe", [ASM.CPU(8)])
## Switch on computer
HARDWARE.switchOn()
# Usar Scheduler Prioridad no expropiativo
# Prioridad mas alta: 1
# Prioridad mas baja: 5
# execute all programs
kernel.run(prg1, 4) ## 4 = prioridad del proceso
kernel.run(prg2, 2) ## 2 = prioridad del proceso
kernel.run(prg3, 1) ## 1 = prioridad del proceso
kernel.run(prg4, 3) ## 3 = prioridad del proceso
kernel.run(prg5, 5) ## 5 = prioridad del proceso
kernel.crontab.add_job(13 ,prg6, 4) # En el tick 13 corre el prg6 con prioridad 4
kernel.crontab.add_job(15 ,prg7, 5) # En el tick 15 corre el prg7 con prioridad 5
```

Final: Asignacion Continua Memory Manager **File System** Loader **PCB** Mem **Table** # #NEW InterruptHandler Interrupt Vector #KILL #IO IN InterruptHandler **CPU** MMU #IO IN Scheduler InterruptHandler #KILL **#IO OUT** #TIMEOUT InterruptHandler tick IO_* # Timer Dispatcher tick(2) Clock Waiting **IODevice** tick(1) Queue tick(3) **IO Device Controller** 8/8 Crontab Hardware Software