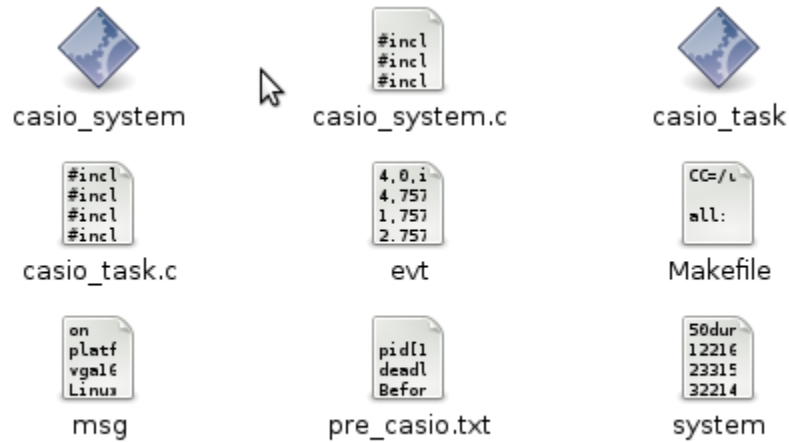
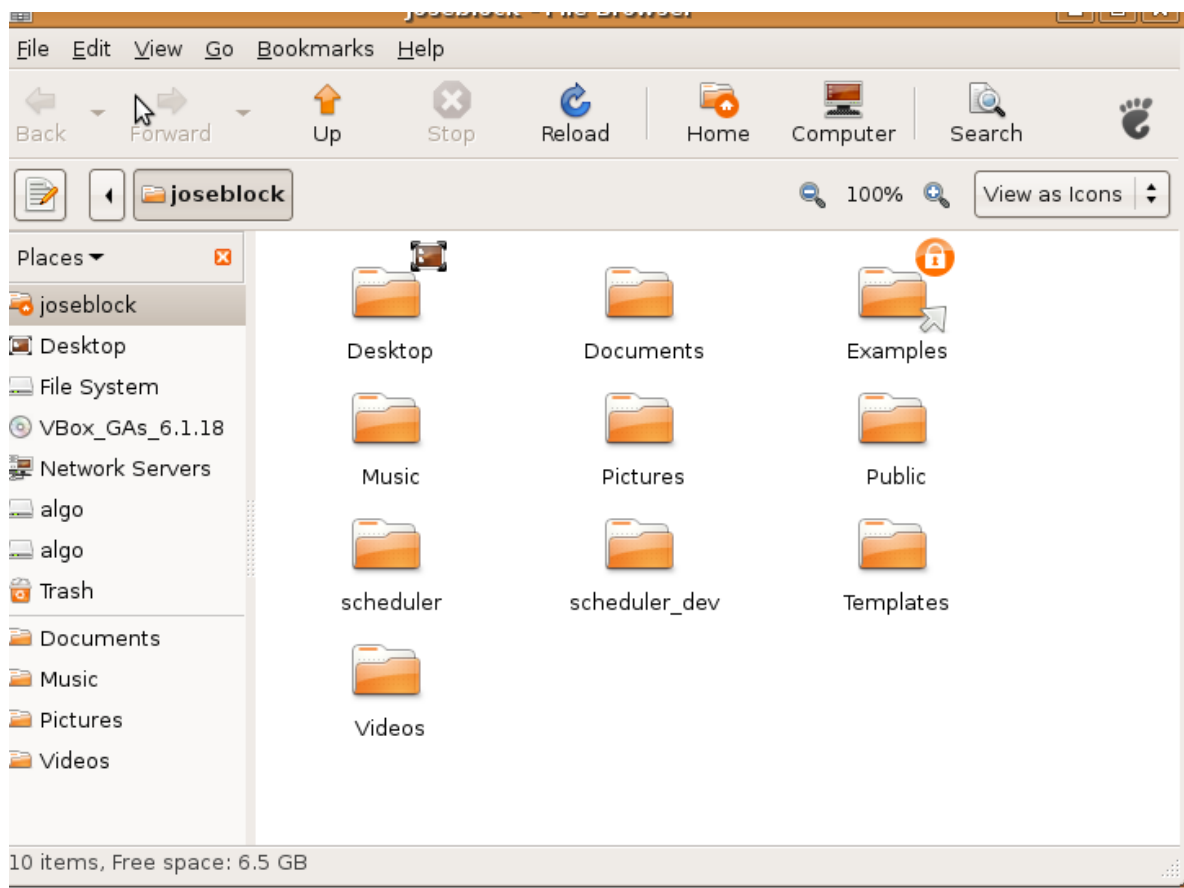


### Calendarización de CPU



```
joseblock@joseblock-laptop:/etc/apt$ sudo nano sources.list
[sudo] password for joseblock:
joseblock@joseblock-laptop:/etc/apt$
```



```
joseblock@joseblock-laptop:~/Desktop/linux-2.6.24-casio/arch/x86$ sudo nano Kconfig
joseblock@joseblock-laptop:~/Desktop/linux-2.6.24-casio/arch/x86$
```

Investigue y resuma:

Funcionamiento y sintaxis de uso de structs.

Hace referencia a un bloque contiguo de memoria física, generalmente delimitado (dimensionado) por límites de longitud de palabra. Corresponde a la función de nombre similar disponible en algunos ensambladores para procesadores Intel. Al ser un bloque de memoria contigua, cada campo dentro de una estructura se ubica en un cierto desplazamiento fijo desde el principio.

[https://en.wikipedia.org/wiki/Struct\\_\(C\\_programming\\_language\)](https://en.wikipedia.org/wiki/Struct_(C_programming_language))

```
struct structureName
{
    dataType member1;
    dataType member2;
    ...
};
```

Here is an example:

```
struct Person
{
    char name[50];
    int citNo;
    float salary;
};
```

<https://www.programiz.com/c-programming/c-structures>

Propósito y directivas del preprocesador.

Es el primer programa invocado por el compilador y procesa directivas como #include, #define e #if. Estas directivas no son específicas de C. En realidad pueden ser usadas con cualquier tipo de archivo. El preprocesador utiliza 4 etapas denominadas Fases de traducción. Aunque alguna implementación puede elegir hacer alguna o todas las fases simultáneamente, debe comportarse como si fuesen ejecutadas paso a paso.

[https://es.wikipedia.org/wiki/Preprocesador\\_de\\_C#Precedencia](https://es.wikipedia.org/wiki/Preprocesador_de_C#Precedencia)

Diferencia entre \*y &en el manejo de referencias a memoria (punteros).

El símbolo \* es un puntero y & es una referencia. Un puntero puede ser inicializado con cualquier valor en cualquier momento, mientras que una referencia debe ser inicializada en el momento que se declara.

Solo el puntero puede tener valor nulo. Solo el puntero puede ser cambiado de direccionamiento, con la condición de que la variable a la que apunta sea del mismo tipo que la original.

<https://stackoverflow.com/questions/45610026/what-is-the-difference-between-a-a-and-a>

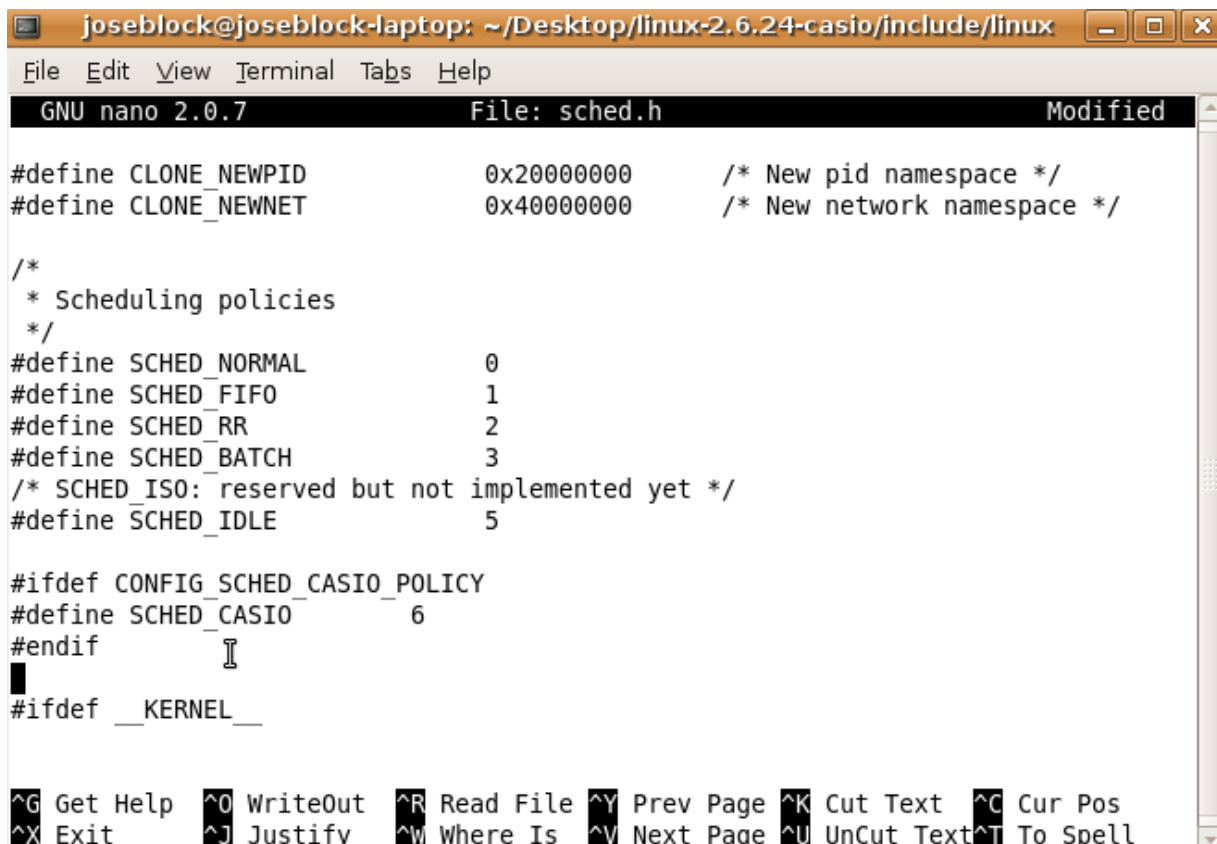
Propósito y modo de uso de APT y dpkg.

APT simplifica en gran medida la instalación y eliminación de programas en los sistemas GNU/Linux. No existe un programa apt en sí mismo, sino que APT es una biblioteca de funciones C++ que se emplea por varios programas de línea de comandos para distribuir paquetes. En especial, apt-get y apt-cache.

[https://es.wikipedia.org/wiki/Advanced\\_Packaging\\_Tool](https://es.wikipedia.org/wiki/Advanced_Packaging_Tool)

El programa dpkg es la base del sistema de gestión de paquetes de Debian GNU/Linux. Es una herramienta de bajo nivel; se necesita un frontal de alto nivel para traer los paquetes desde lugares remotos o resolver conflictos complejos en las dependencias de paquetes. Debian cuenta con apt para esta tarea. Debian posee una serie de herramientas que es necesario llamar para construir un paquete:

- dpkg-source
- dpkg-gencontrol
- dpkg-shlibdeps
- dpkg-genchanges
- dpkg-buildpackage
- dpkg-distaddfile
- dpkg-parsechangelog



```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/include/linux
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.h Modified

#define CLONE_NEWPID          0x20000000    /* New pid namespace */
#define CLONE_NEWNET          0x40000000    /* New network namespace */

/*
 * Scheduling policies
 */
#define SCHED_NORMAL          0
#define SCHED_FIFO            1
#define SCHED_RR              2
#define SCHED_BATCH           3
/* SCHED_ISO: reserved but not implemented yet */
#define SCHED_IDLE            5

#ifdef CONFIG_SCHED_CASIO_POLICY
#define SCHED_CASIO           6
#endif

#ifdef __KERNEL__

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text ^T To Spell
```

```

define SCHED_BATCH    3
endif

define SCHED_CASIO 6

ifndef __USE_MISC
* Cloning flags. */
define CSIGNAL    0x00000000
#define CLONE_VM    0x00000001

```

¿Cuál es el propósito de los archivos sched.h modificados?

Contiene la referencia de las struct, const y otras variables que colaboran con la creación de las políticas de calendarización.

¿Cuál es el propósito de la definición incluida y las definiciones existentes en el archivo?

Son las encargadas de definir las variables const que sirven para relacionar las políticas de calendarización del sistema.

```

joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/include/linux
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.h Mod

#endif
#ifdef CONFIG_FAULT_INJECTION
    int make_it_fail;
#endif
    struct prop_local_single dirties;

#ifdef CONFIG_SCHED_CASIO_POLICY
    unsigned int casio_id;
    unsigned long long deadline;
#endif

};

/*
 * Priority of a process goes from 0..MAX_RT_PRIO-1, valid RT
 * priority is 0..MAX_RT_PRIO-1, and SCHED_NORMAL/SCHED_BATCH
 * tasks are in the range MAX_RT_PRIO..MAX_RT_PRIO-1. Priority
 * values are inverted: lower p->prio value means higher priority.
 *
File Name to Write: sched.h
^G Get Help      ^T To Files      M-M Mac Format   M-P Prepend
^C Cancel        M-D DOS Format   M-A Append       M-B Backup Fi

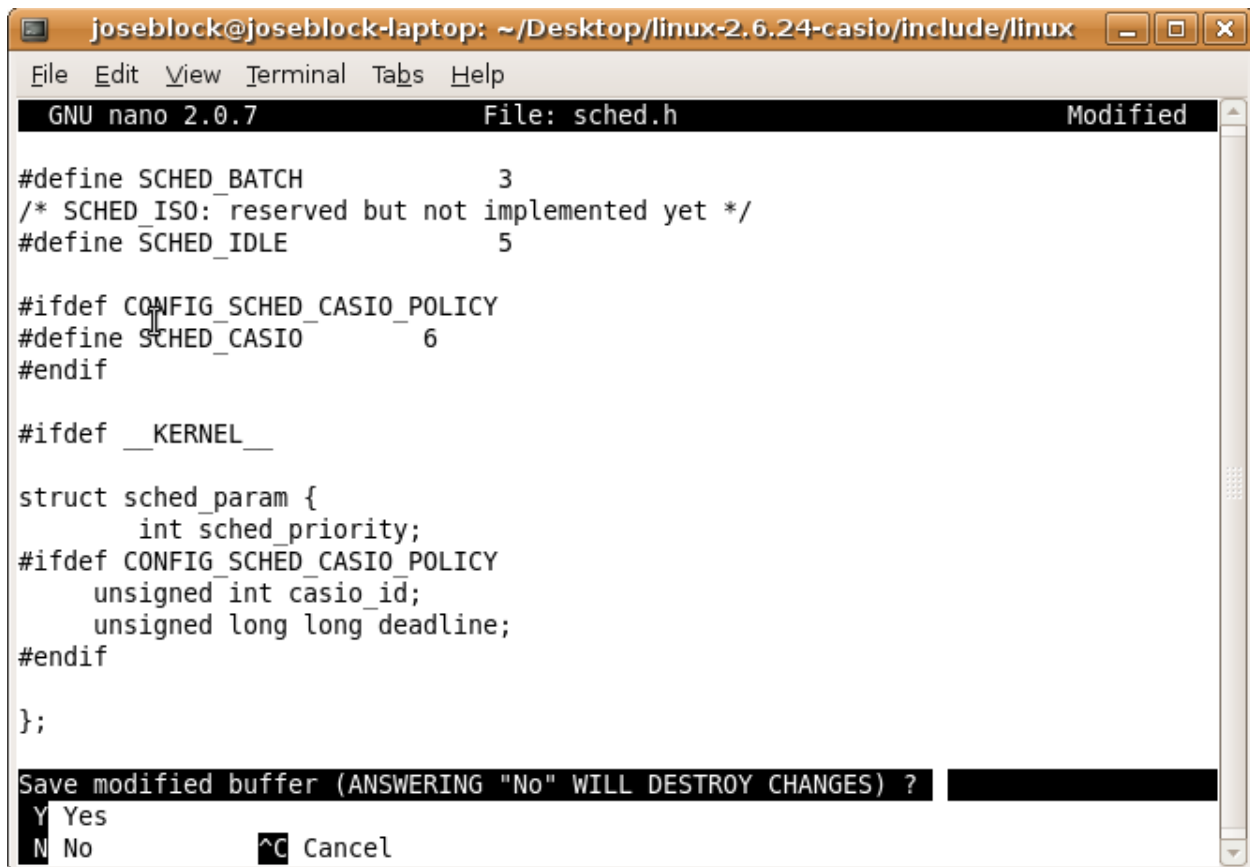
```

¿Qué es una task en Linux?

Es equivalente al famoso; proceso.

¿Cuál es el propósito de `task_struct` cuál es su análogo en Windows?

Es un PCB en Linux, que por ejemplo en Windows es un KPROCESS con el EPROCESS y el PEB.



```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/include/linux
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.h Modified

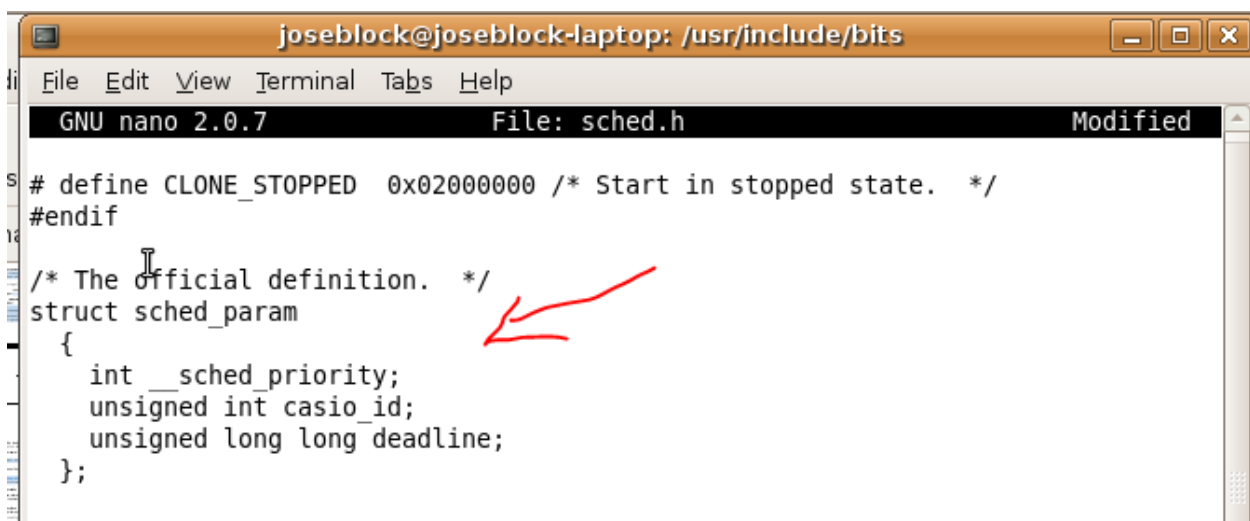
#define SCHED_BATCH          3
/* SCHED_ISO: reserved but not implemented yet */
#define SCHED_IDLE           5

#ifdef CONFIG_SCHED_CASIO_POLICY
#define SCHED_CASIO          6
#endif

#ifdef __KERNEL__

struct sched_param {
    int sched_priority;
#ifdef CONFIG_SCHED_CASIO_POLICY
    unsigned int casio_id;
    unsigned long long deadline;
#endif
};

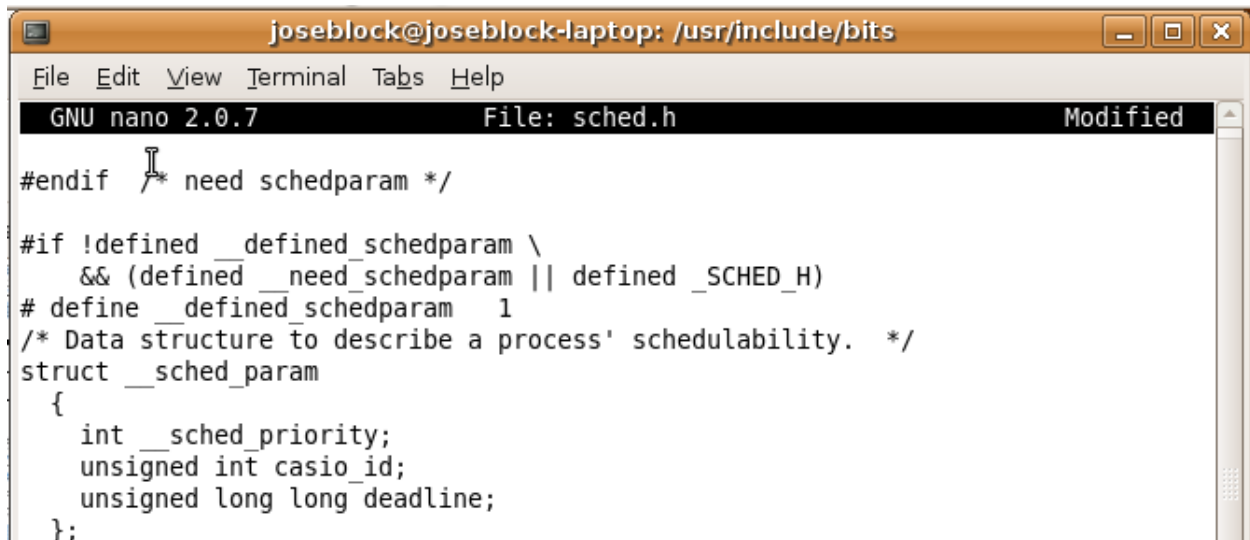
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
Y Yes
N No ^C Cancel
```



```
joseblock@joseblock-laptop: /usr/include/bits
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.h Modified

# define CLONE_STOPPED 0x02000000 /* Start in stopped state. */
#endif

/* The official definition. */
struct sched_param
{
    int __sched_priority;
    unsigned int casio_id;
    unsigned long long deadline;
};
```

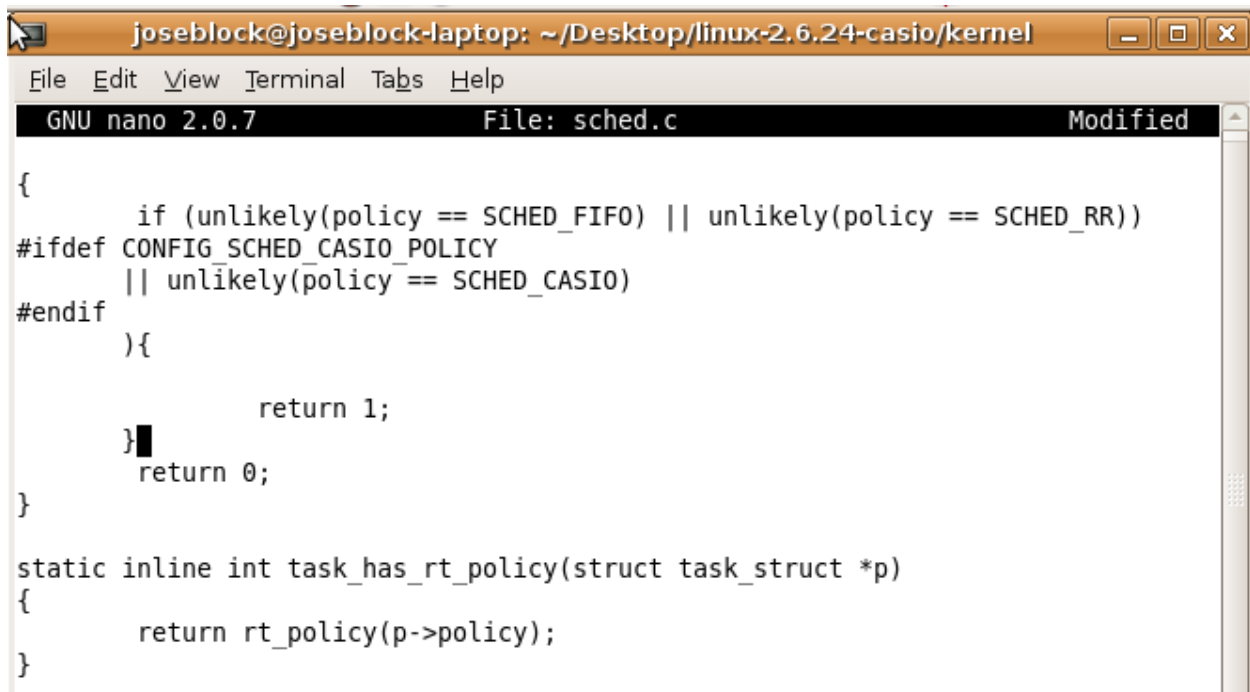
A screenshot of a nano editor window. The title bar shows the user 'joseblock' on a 'joseblock-laptop' at the path '/usr/include/bits'. The menu bar includes 'File', 'Edit', 'View', 'Terminal', 'Tabs', and 'Help'. The status bar at the bottom indicates 'GNU nano 2.0.7', 'File: sched.h', and 'Modified'. The code content shows the end of a conditional compilation block for 'need schedparam' and the start of a block for 'defined \_\_defined\_schedparam'. It defines a struct 'struct \_\_sched\_param' with fields: 'int \_\_sched\_priority;', 'unsigned int casio\_id;', and 'unsigned long long deadline;'.

```
#endif /* need schedparam */

#if !defined __defined_schedparam \
    && (defined __need_schedparam || defined _SCHED_H)
# define __defined_schedparam 1
/* Data structure to describe a process' schedulability. */
struct __sched_param
{
    int __sched_priority;
    unsigned int casio_id;
    unsigned long long deadline;
};
```

¿Qué información contiene sched\_param?

Almacena los params(parametros) que cumplen con las políticas de calendarización. Solo se enfoca en la calendarización, y luego de cada acción que se haga hay un medidor de tiempo en nanosegundos que define el tiempo de finalizar un proceso(task).

A screenshot of a nano editor window. The title bar shows the user 'joseblock' on a 'joseblock-laptop' at the path '~/Desktop/linux-2.6.24-casio/kernel'. The menu bar includes 'File', 'Edit', 'View', 'Terminal', 'Tabs', and 'Help'. The status bar at the bottom indicates 'GNU nano 2.0.7', 'File: sched.c', and 'Modified'. The code content shows a conditional compilation block for 'CONFIG\_SCHED\_CASIO\_POLICY' and a function 'task\_has\_rt\_policy' that returns the result of 'rt\_policy(p->policy)'.

```
{
    if (unlikely(policy == SCHED_FIFO) || unlikely(policy == SCHED_RR))
#ifdef CONFIG_SCHED_CASIO_POLICY
    || unlikely(policy == SCHED_CASIO)
#endif
){
    return 1;
}
return 0;
}

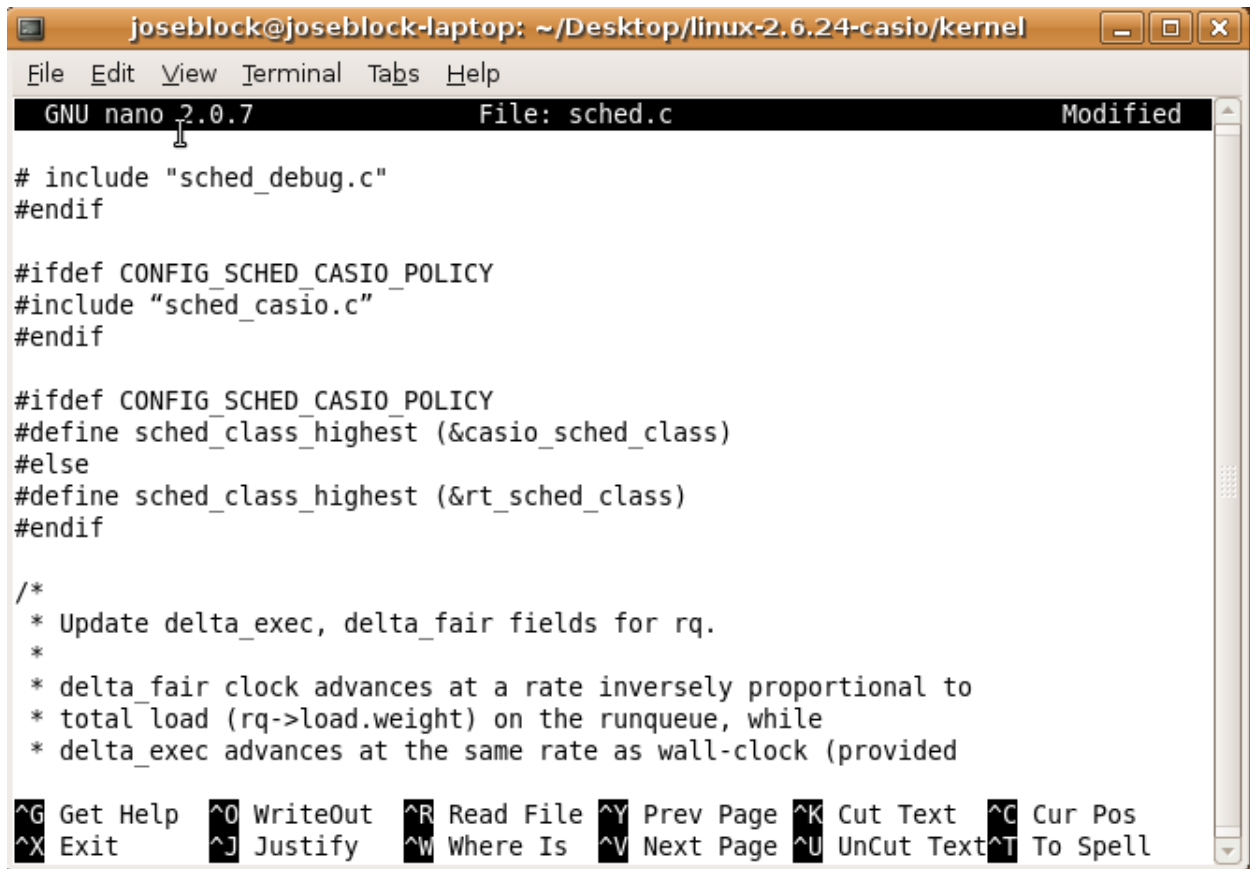
static inline int task_has_rt_policy(struct task_struct *p)
{
    return rt_policy(p->policy);
}
```

¿Para qué sirve la función rt\_policy y para qué sirve la llamada unlikely en ella?

Para saber si una modificación en la calendarización se llevará a cabo a una clase de tiempo real. Por otro lado, unlikely aumenta la rapidez al ejecutar instrucciones que incumplen una condición en sí. Por las mismas circunstancias, se ve aumentada la velocidad de respuesta, al no involucrar políticas SCHEDRR, SCHED\_FIFO y SCHED\_CASIO\_POLICY en la modificación de calendarización por lo que se debería de hacer modificaciones en clases de prioridad normal.

¿Qué tipo de tareas calendariza la política EDF, en vista del método modificado?

Según las modificaciones realizadas y el nombre de las pólizas, estas tasks tienen un tiempo límite (deadline) que se da en tasks(procesos) que se llevan a cabo en tiempo de operación del mismo.



```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
GNU nano 2.0.7 File: sched.c Modified

# include "sched_debug.c"
#endif

#ifdef CONFIG_SCHED_CASIO_POLICY
#include "sched_casio.c"
#endif

#ifdef CONFIG_SCHED_CASIO_POLICY
#define sched_class_highest (&casio_sched_class)
#else
#define sched_class_highest (&rt_sched_class)
#endif

/*
 * Update delta_exec, delta_fair fields for rq.
 *
 * delta_fair clock advances at a rate inversely proportional to
 * total load (rq->load.weight) on the runqueue, while
 * delta_exec advances at the same rate as wall-clock (provided
```

Describe la precedencia de prioridades para las políticas EDF, RT y CFS, de acuerdo con los cambios realizados hasta ahora.

Se ordenan por prioridad hacia abajo

1. EDF
2. RT
3. CFS

```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.c Modified

        p->sched_class = &rt_sched_class;
        break;
    }

    p->rt_priority = prio;
    p->normal_prio = normal_prio(p);
    /* we are holding p->pi_lock already */
    p->prio = rt_mutex_getprio(p);
    set_load_weight(p);
#ifdef CONFIG_SCHED_CASIO_POLICY
    case SCHED_CASIO:
        p->sched_class = &casio_sched_class;
        break;
#endif
}

        policy := SCHED_NORMAL or policy
        policy != SCHED_IDLE
/*)*/
#ifdef CONFIG_SCHED_CASIO_POLICY
    && policy != SCHED_CASIO
#endif
    )

        return -EINVAL;
/*
 * Valid priorities for SCHED_FIFO and SCHED_RR
 * 1 - MAX_USER_RT_PRIO - 1 - valid priority for SCHED_FIFO
 */
        return -EPERM;
    }

#ifdef CONFIG_SCHED_CASIO_POLICY
    if (policy == SCHED_CASIO){
        p->deadline = param->deadline;
        p->casio_id = param->casio_id;
    }
#endif
```



```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.c Modified

    struct rt_prio_array active;
    int rt_load_balance_idx;
    struct list_head *rt_load_balance_head, *rt_load_balance_curr;
};
#ifdef CONFIG_SCHED_CASIO_POLICY
    struct casio_task{
        struct rb_node casio_rb_node;
        unsigned long long absolute_deadline;
        struct list_head casio_list_node;
        struct task_struct* task;
    };
    struct casio_rq{
        struct rb_root casio_rb_root;
        struct list_head casio_list_head;
        atomic_t nr_running;
    };
#endif
/*
```

Explique el contenido de la estructura casio\_task.

Es un struct que define la calendarización de las tasks por SCHED\_CASIO\_POLICY. Es referenciada con un nodo en un "red-black tree" para calendarizar el momento en el que se ejecuta casio\_task. Se va haciendo referencia a través de un nodo en una lista que registra las casio\_task en el sistema. Por último, hace referencia a una task que hace referencia a la determinación del tiempo límite(deadline) que define el tiempo límite que debe tomar la task para ser terminada.

```
    struct casio_rq rq;
#ifdef CONFIG_SCHED_CASIO_POLICY
    struct casio_rq casio_rq;
#endif
```

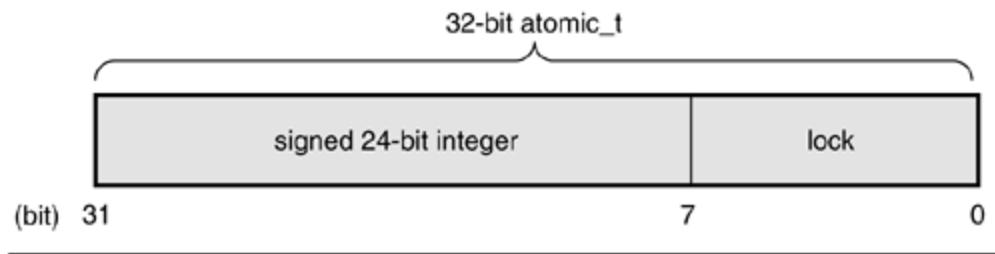
Explique el propósito y contenido de la estructura casio\_rq.

Se encarga de mantener a la cola de casio\_task por procesador preparadas para ejecutar cada proceso.

¿Qué es y para qué sirve el tipo atomic\_t? Describa brevemente los conceptos de operaciones RMW (read-modify-write) y mapeo de dispositivos en memoria (MMIO).

Las operaciones atómicas proporcionan instrucciones que se ejecutan de forma atómica sin interrupción. Así como originalmente se pensó que el átomo era una partícula indivisible, los operadores atómicos son instrucciones indivisibles. Por ejemplo, como se discutió en el capítulo anterior, un incremento atómico puede leer e incrementar una variable en uno en un solo paso indivisible e ininterrumpido.

**Figure 9.1. Old layout of the 32-bit `atomic_t` on SPARC.**



<http://books.gigatux.nl/mirror/kerneldevelopment/0672327201/ch09lev1sec1.html>

RMW es una clase de operaciones atómicas (como test-and-set, fetch-and-add, y compare-and-swap) que leen una ubicación de memoria y escriben un nuevo valor en ella simultáneamente, ya sea con un valor completamente nuevo o alguna función del valor anterior. Estas operaciones evitan condiciones de carrera en aplicaciones de subprocesos múltiples. Normalmente se utilizan para implementar mutex o semáforos. Estas operaciones atómicas también se utilizan mucho en la sincronización sin bloqueo.

<https://en.wikipedia.org/wiki/Read%E2%80%93modify%E2%80%93write>

MMIO es un método complementario de relocalización de input/ output (I/O) entre la unidad central de procesamiento (CPU) y los dispositivos periféricos en una computadora. Un enfoque alternativo es usar procesadores de I/O dedicados, comúnmente conocidos como canales en computadoras centrales, que ejecutan sus propias instrucciones.

Las I/O asignadas en memoria utilizan el mismo espacio de direcciones para direccionar tanto la memoria como los dispositivos de I/O. La memoria y los registros de los dispositivos de I/O están mapeados (asociados con) valores de dirección. Por lo tanto, cuando la CPU accede a una dirección, puede referirse a una parte de la RAM física o, en su lugar, puede referirse a la memoria del dispositivo de I/O.

[https://en.wikipedia.org/wiki/Memory-mapped\\_I/O](https://en.wikipedia.org/wiki/Memory-mapped_I/O)

```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched.c Modified

        policy = oldpolicy = -1;
        __task_rq_unlock(rq);
        spin_unlock_irqrestore(&p->pi_lock, flags);
        goto recheck;
    }
#ifdef CONFIG_SCHED_CASIO_POLICY
    if (policy == SCHED_CASIO){
        add_casio_task_2_list(&rq->casio_rq, p);
    }
#endif
    update_rq_clock(rq);
```

```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched_casio.c Modified

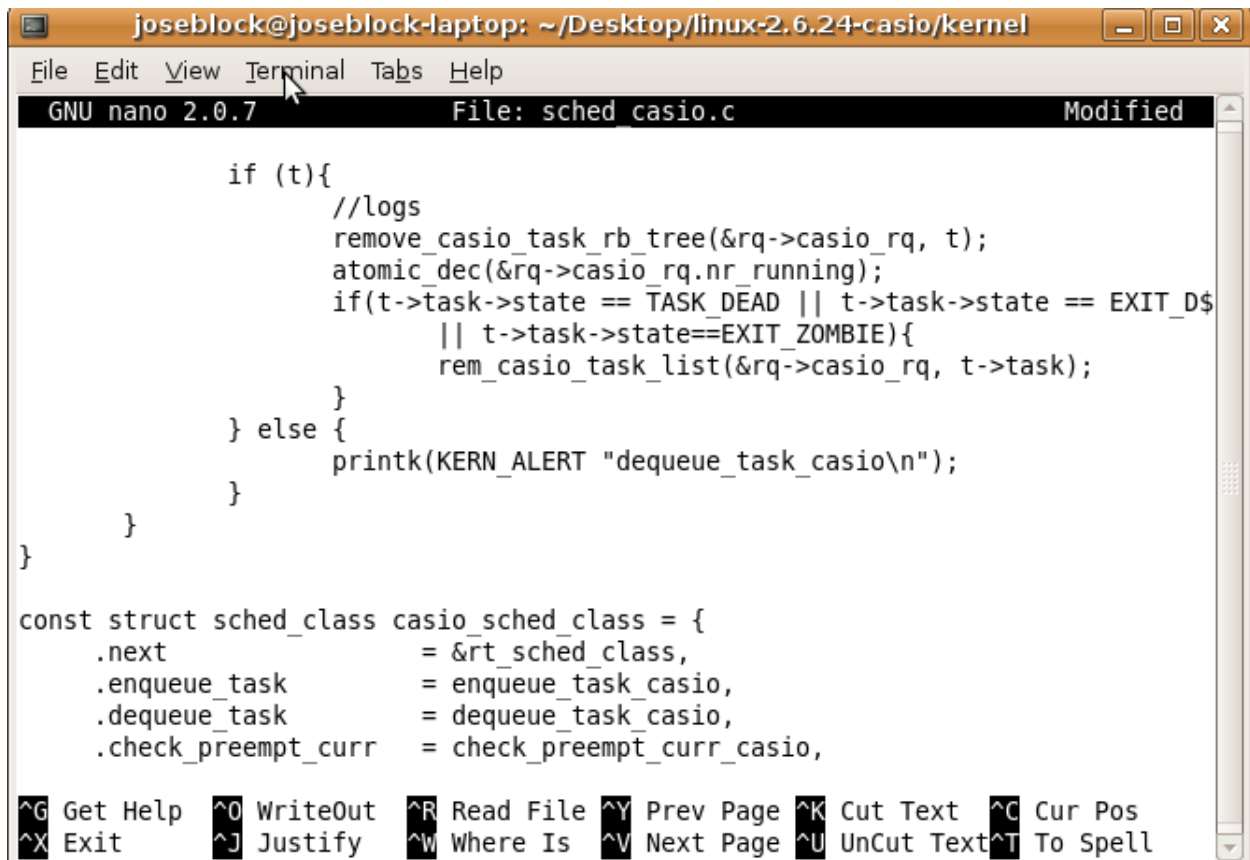
        node = node->rb_left;
    }
    p = rb_entry(node, struct casio_task, casio_rb_node);
    return p;
}

const struct sched_class casio_sched_class = {
    .next                = &rt_sched_class,
    .enqueue_task        = enqueue_task_casio,
    .dequeue_task        = dequeue_task_casio,
    .check_preempt_curr  = check_preempt_curr_casio,
    .pick_next_task      = pick_next_task_casio,
    .put_prev_task       = put_prev_task_casio,
#ifdef CONFIG_SMP
    .load_balance        = load_balance_casio,
    .move_one_task       = move_one_task_casio,
#endif
    .set_curr_task       = set_curr_task_casio,
    .task_tick           = task_tick_casio,
};

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text ^T To Spell
```

¿Qué indica el campo .next de esta estructura?

Hace referencia a la clase que le sigue en la jerarquía de prioridades.



```
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched_casio.c Modified

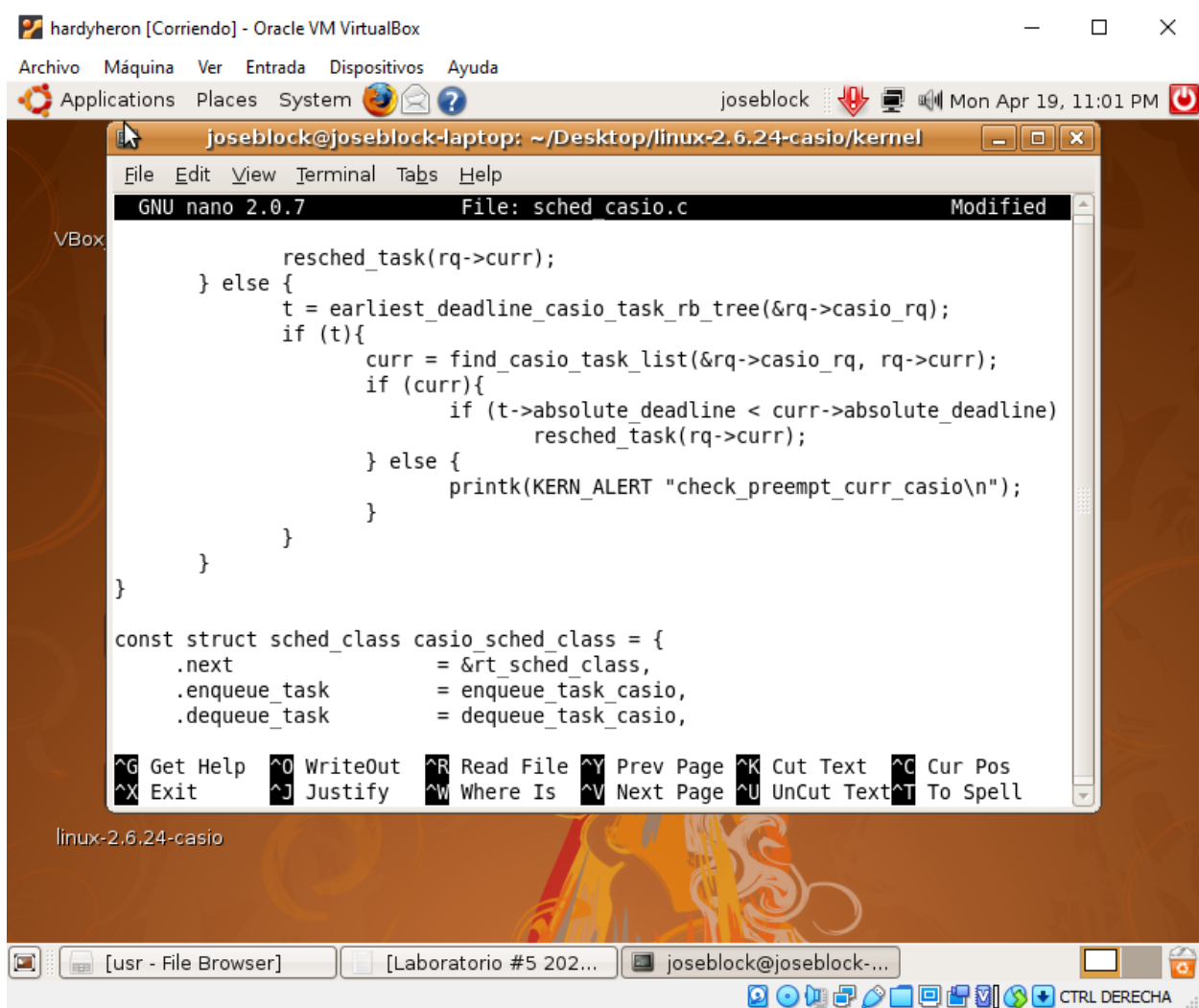
        if (t){
            //logs
            remove_casio_task_rb_tree(&rq->casio_rq, t);
            atomic_dec(&rq->casio_rq.nr_running);
            if(t->task->state == TASK_DEAD || t->task->state == EXIT_D$
                || t->task->state==EXIT_ZOMBIE){
                rem_casio_task_list(&rq->casio_rq, t->task);
            }
        } else {
            printk(KERN_ALERT "dequeue_task_casio\n");
        }
    }
}

const struct sched_class casio_sched_class = {
    .next                = &rt_sched_class,
    .enqueue_task        = enqueue_task_casio,
    .dequeue_task        = dequeue_task_casio,
    .check_preempt_curr  = check_preempt_curr_casio,
};

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text ^T To Spell
```

Tomando en cuenta las funciones para manejo de lista y red-black tree de `casio_tasks`, explique el ciclo de vida de una `casio_task` desde el momento en el que se le asigna esta clase de calendarización mediante `sched_setscheduler`. El objetivo es que indique el orden y los escenarios en los que se ejecutan estas funciones, así como las estructuras de datos por las que pasa. ¿Por qué se guardan las `casio_tasks` en un red-black tree y en una lista encadenada?

Cuando se ejecuta `sched_setscheduler`, se suma el proceso actual a una lista con la acción `add_casio_task_2_list`, que a simple vista se puede notar que se crea una nueva `casio_task`. En el momento en que una `casio_task` está "ready", se agrega a través de `enqueue_task_casio` al "red-black-tree" con su respectivo tiempo límite, por lo tanto hay que encontrar el proceso con `find_casio_task_list` e insertarla en el árbol con `insert_casio_task_rb_tree`. Ahora una `casio_task` deja de estar "ready", por lo que se saca del "red-black-tree" con la acción `dequeue_task_casio`, que activa a `find_casio_task_list` y a `remove_casio_task_rb_tree`.



```
hardyheron [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Applications Places System joseblock Mon Apr 19, 11:01 PM
joseblock@joseblock-laptop: ~/Desktop/linux-2.6.24-casio/kernel
File Edit View Terminal Tabs Help
GNU nano 2.0.7 File: sched_casio.c Modified

    resched_task(rq->curr);
} else {
    t = earliest_deadline_casio_task_rb_tree(&rq->casio_rq);
    if (t){
        curr = find_casio_task_list(&rq->casio_rq, rq->curr);
        if (curr){
            if (t->absolute_deadline < curr->absolute_deadline)
                resched_task(rq->curr);
        } else {
            printk(KERN_ALERT "check_preempt_curr_casio\n");
        }
    }
}

const struct sched_class casio_sched_class = {
    .next          = &rt_sched_class,
    .enqueue_task  = enqueue_task_casio,
    .dequeue_task  = dequeue_task_casio,
    .get_rr_interval = get_rr_interval_casio,
};

linux-2.6.24-casio

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
[usr - File Browser] [Laboratorio #5 202...] joseblock@joseblock-... CTRL DERECHA
```

¿Cuándo preemptea una `casio_task` a la task actualmente en ejecución?

En caso de no ser una `casio_task`.

```
joseblock@joseblock-laptop: ~/scheduler/linux-2.6.24-casio
File Edit View Terminal Tabs Help
Testing module (CRYPTO_TEST) [M/n/?] m
Authenc support (CRYPTO_AUTHENC) [M/n/y/?] m
*
* Hardware crypto devices
*
Hardware crypto devices (CRYPTO_HW) [Y/n/?] y
  Support for VIA PadLock ACE (CRYPTO_DEV_PADLOCK) [Y/n/m/?] y
    PadLock driver for AES algorithm (CRYPTO_DEV_PADLOCK_AES) [M/n/y/?] m
    PadLock driver for SHA1 and SHA256 algorithms (CRYPTO_DEV_PADLOCK_SHA) [M/
n/y/?] m
    Support for the Geode LX AES engine (CRYPTO_DEV_GEODE) [M/n/y/?] m
*
* Library routines
*
CRC-CCITT functions (CRC_CCITT) [M/y/?] m
CRC16 functions (CRC16) [M/y/?] m
CRC ITU-T V.41 functions (CRC_ITU_T) [M/y/?] m
CRC32 functions (CRC32) [Y/?] y
CRC7 functions (CRC7) [M/n/y/?] m
CRC32c (Castagnoli, et al) Cyclic Redundancy-Check (LIBCRC32C) [M/y/?] m
#
# configuration written to .config
#
joseblock@joseblock-laptop:~/scheduler/linux-2.6.24-casio$
```

```
joseblock@joseblock-laptop: ~/scheduler/linux-2.6.24-casio
File Edit View Terminal Tabs Help
-Plinux-image-2.6.24-casio -P/home/joseblock/scheduler/l
linux-2.6.24-casio/debian/linux-image-2.6.24-casio/
create_md5sums_fn () { cd $1 ; find . -type f ! -regex '.*DEBIAN/.*' ! -regex '
./etc/.*' ! -regex '.*lib/modules/[^/]*/modules\..*' -printf '%P\0' | xargs
-r0 md5sum > DEBIAN/md5sums ; if [ -z "DEBIAN/md5sums" ] ; then rm -f "DEBIAN/m
d5sums" ; fi ; } ; create_md5sums_fn /home/joseblock/scheduler/linu
x-2.6.24-casio/debian/linux-image-2.6.24-casio
chmod -R og=rX /home/joseblock/scheduler/linux-2.6.24-casio/debi
an/linux-image-2.6.24-casio
chown -R root:root /home/joseblock/scheduler/linux-2.6.24-casio/debi
an/linux-image-2.6.24-casio
dpkg --build /home/joseblock/scheduler/linux-2.6.24-casio/debi
an/linux-image-2.6.24-casio ..
dpkg-deb: building package `linux-image-2.6.24-casio' in `../linux-image-2.6.24-
casio_2.6.24-casio-10.00.Custom_i386.deb'.
make[1]: Leaving directory `/home/joseblock/scheduler/linux-2.6.24-casio'
===== making target stamp-kernel-image [new prereqs: linux-image-2.6.24-casio l
inux-image-2.6.24-casio]=====
This is kernel package version 11.001.
echo done > stamp-kernel-image
===== making target kernel_image [new prereqs: stamp-configure stamp-build-kern
el stamp-kernel-image]=====
This is kernel package version 11.001.
joseblock@joseblock-laptop:~/scheduler/linux-2.6.24-casio$ sudo dpkg -
```

```
joseblock@joseblock-laptop: ~/scheduler
File Edit View Terminal Tabs Help
Selecting previously deselected package linux-image-2.6.24-casio.
(Reading database ... 98840 files and directories currently installed.)
Unpacking linux-image-2.6.24-casio (from linux-image-2.6.24-casio_2.6.24-casio-1
10.00.Custom_i386.deb) ...
Done.
Setting up linux-image-2.6.24-casio (2.6.24-casio-10.00.Custom) ...
Running depmod.
Finding valid ramdisk creators.
Using mkinitramfs-kpkg to build the ramdisk.
Running postinst hook script update-grub.
Searching for GRUB installation directory ... found: /boot/grub
Searching for default file ... found: /boot/grub/default
Testing for an existing GRUB menu.lst file ... found: /boot/grub/menu.lst
Searching for splash image ... none found, skipping ...
Found kernel: /boot/vmlinuz-2.6.24-casio
Found kernel: /boot/vmlinuz-2.6.24-26-generic
Found kernel: /boot/memtest86+.bin
Replacing config file /var/run/grub/menu.lst with new version
Updating /boot/grub/menu.lst ... done

Examining /etc/kernel/postinst.d.
run-parts: executing /etc/kernel/postinst.d/vboxadd

joseblock@joseblock-laptop:~/scheduler$
```

```
hardyheron [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda

La máquina virtual informa que el SO invitado no soporta integración del ratón en el modo de vídeo actual. Se necesita capturar el

Ubuntu 8.04.4 LTS, kernel 2.6.24-casio
Ubuntu 8.04.4 LTS, kernel 2.6.24-casio (recovery mode)
Ubuntu 8.04.4 LTS, kernel 2.6.24-26-generic
Ubuntu 8.04.4 LTS, kernel 2.6.24-26-generic (recovery mode)
Ubuntu 8.04.4 LTS, memtest86+

Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the
commands before booting, or 'c' for a command-line.
```

Ejecute nuevamente el archivo `casio_systemtal` como se hizo al inicio del laboratorio, pero guardando los resultados en un archivo diferente. Adjunte ambos archivos de resultados de `casio_systema` su entrega, comentando sobre sus diferencias.

- Antes
  - Tiene solo una task en la que se define el thread que se usa, las prioridades con tres Jobs con sus tiempos de corrida predeterminados y muestra cuando se termina una task.
  - Termina el programa
- Después
  - Tiene cuatro tasks en la que se define el thread que se usa, las prioridades con tres Jobs con sus tiempos de corrida predeterminados (diferentes a los de Antes) y muestra cuando se termina una task.
  - Termina el programa

Ubique el archivo de log de eventos registrados por la calendarización implementada. Adjunte este archivo con su entrega.

Agregue comentarios explicativos a los archivos `casio_task.c` y `casio_system.c` que permitan entender el propósito y funcionamiento de este código. Asegúrese de aclarar el uso de instrucciones y estructuras que no conozca (como, por ejemplo, los timers y la estructura `itimerval`). ¿Qué información contiene el archivo `system` que se especifica como argumento en la ejecución de `casio_system`?

En este ejercicio, se lleva a cabo la configuración para las tasks, con una calendarización nueva. Todo esto requiere de saber los parámetros temporales de un tiempo límite de cada task.

Investigue el concepto de aislamiento temporal en relación con procesos. Explique cómo el calendarizador `SCHED_DEADLINE`, introducido en la versión 3.14 del kernel de Linux, añade al algoritmo EDF para lograr aislamiento temporal

El aislamiento temporal es la forma de un sistema para asegurar que las acciones que hace y el tiempo que le toma, estén separados de otros procesos que también se están llevando a cabo. `SCHED_DEADLINE` es un calendarizador de CPU disponible en el kernel de Linux desde la versión 3.14, basado en los algoritmos Earliest Deadline First (EDF) y Constant Bandwidth Server (CBS), que admite reservas de recursos: cada tarea programada bajo tales La política está asociada con un presupuesto  $Q$  (también conocido como tiempo de ejecución) y un período  $P$ , correspondiente a una declaración al kernel de que esa tarea requiere  $Q$  unidades de tiempo cada  $P$  unidades de tiempo, en cualquier procesador. El calendarizador es aislamiento temporal ya que espera a un proceso de manera individual, para evitar pérdidas de información o duplicación de la misma, etc.

[https://www.researchgate.net/publication/47336303\\_Temporal\\_isolation\\_effects\\_in\\_recognition\\_and\\_serial\\_recall](https://www.researchgate.net/publication/47336303_Temporal_isolation_effects_in_recognition_and_serial_recall)