

OS 第三次作業

一、分工

鄧智宇：撰寫報告、相關資料查找

黃信維：撰寫程式、相關資料查找

二、作業步驟以及遇到問題

1. 註解.config 的內容
2. 依據題目要求，完成 check_preempt_curr_rt(), pick_next_task_rt() 和 get_rr_interval_rt()三個檔案

check_preempt_curr_rt()

```
if (p->prio < rq->curr->prio) {
    resched_curr(rq);
    return;
}
```

pick_next_task_rt()

```
if(rq->rt_rt_queued <= 0) return NULL;

struct sched_rt_entity *rt_se;
struct rt_rq *rt_rq = &rq->rt;
struct rt_prio_array *array = &rt_rq->active;
struct list_head *queue_head;

// find the offset of first nonzero priority array
int prio_offset;
prio_offset = sched_find_first_bit(array->bitmap);
// shift to this priority
queue_head = array->queue + prio_offset;

// Target: get next scheduable entity address
// Hint: use linux list library and related data structure
// Pseudo code:
// rt_se = get_member_address_of_run_list_from_queue_head_pointer()

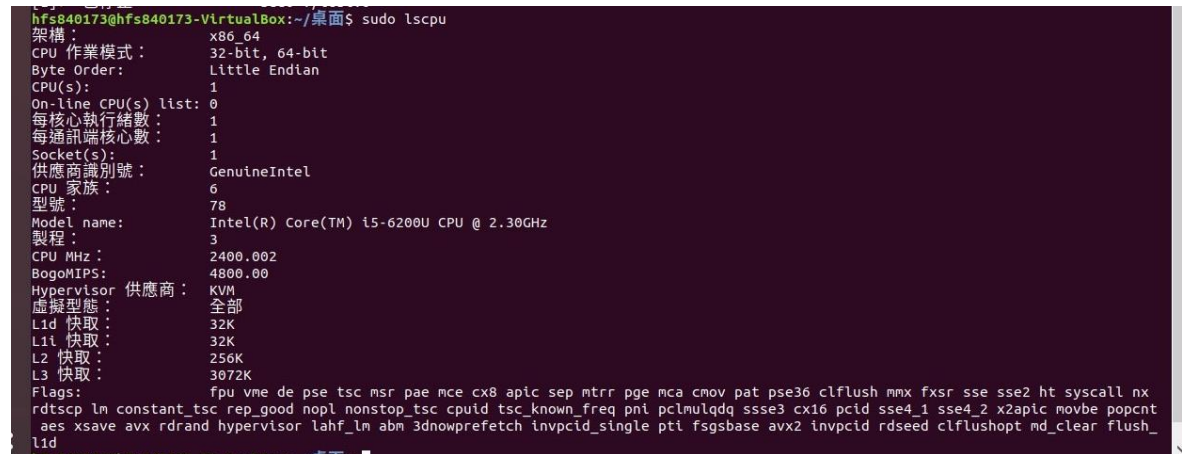
rt_se->run_list.next = queue_head->next;
// p is next runnable task related to scheduable entity
struct task_struct *p;
p = container_of(rt_se, struct task_struct, rt);

set_next_task_rt(rq, p, true);
return p;
```

get_rr_interval_rt()

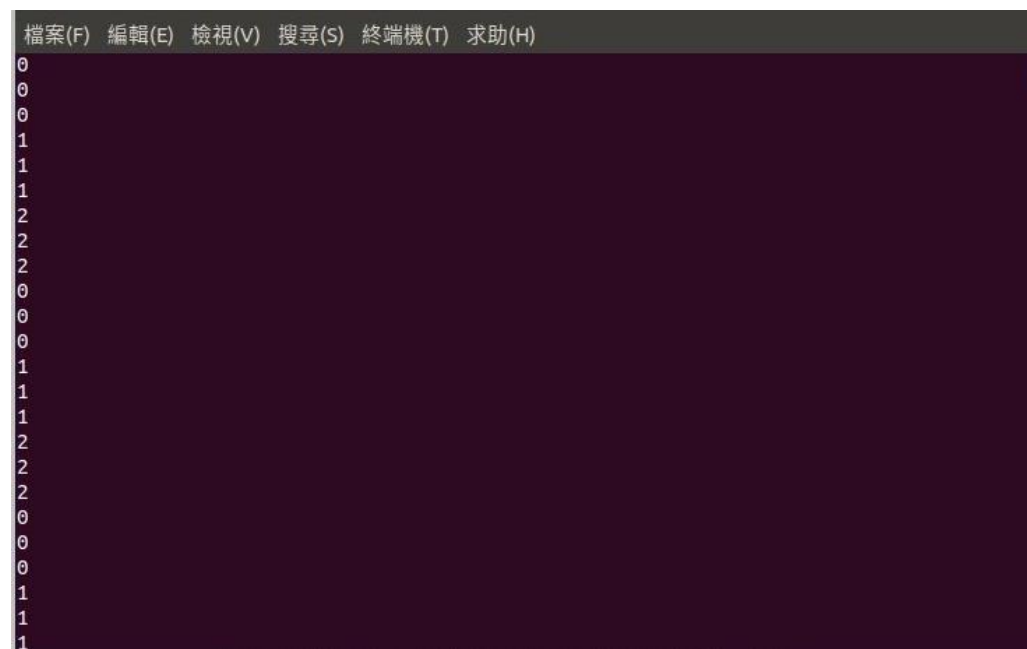
```
if (task->policy == SCHED_RR)
    return sched_rr_timeslice;
else
    return 0;
```

sudo lscpu



```
hfs840173@hfs840173-VirtualBox:~/桌面$ sudo lscpu
架構: x86_64
CPU 作業模式: 32-bit, 64-bit
Byte Order: Little Endian
CPU(s): 1
On-line CPU(s) list: 0
每核心執行緒數: 1
每通訊端核心數: 1
Socket(s): 1
供應商識別號: GenuineIntel
CPU 家族: 6
型號: 78
Model name: Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz
製程: 3
CPU MHz: 2400.002
BogoMIPS: 4800.00
Hypervisor 供應商: KVM
虛擬型態: 全部
L1d 快取: 32K
L1i 快取: 32K
L2 快取: 256K
L3 快取: 3072K
Flags: fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx
rdtscp lm constant_tsc rep_good nopl nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt
aes xsave avx rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_single pti fsgsbase avx2 invpcid rdseed clflushopt md_clear flush_
l1d
```

測試程式 test.c 可成功執行如下。



```
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)
0
0
0
1
1
1
2
2
2
0
0
0
1
1
1
2
2
2
0
0
0
1
1
1
```

3. 遇到的第一個問題，我們資料型態不斷有誤，使得程式無法編譯。後來我們查找了 Bootlin 以及 rt.c 內部的定義，了解其資料型態，以及 priority_queue 的內容，我們將原先使用的「->」改用「.»來代替，就可以成功編譯。

4. 在使用 test.c 測試的時候，我們遇到了下面的狀況：

201201201201201201201201201201

數字照順序固定印出，但一次只印出一個。後來發現，是因為我們並沒有將 CPU 數目設為 1，才有這樣的情形發生。修正後就可印出正確輸出。

三、引用資料

<https://elixir.bootlin.com/linux/v5.7.9/source/include/linux/uaccess.h#L149>