CPU宏与CPU核心集表示

set是一个指向十六进制位的整数的指针，整数的从低位到高位的每个bit代表从编号从0开始的CPU，1表示指定CPU在集合中，0表示不在集合中。

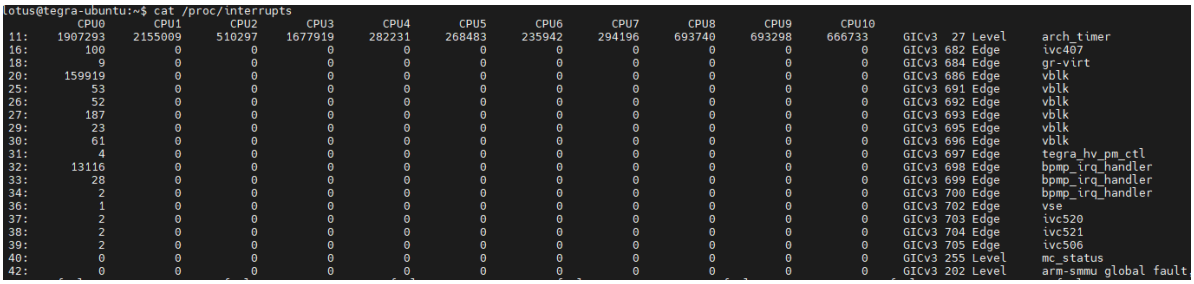
```
void CPU_ZERO(cpu_set_t *set)   //初始化CPU集
void CPU_SET(int cpu,cpu_set_t *set)  //向CPU集中添加某个CPU
void CPU_CLR(int cpu,cpu_set_t *set)  //从CPU集中移除某个CPU
int CPU_ISSET(int cou,cpu_set_t *set)  //检查CPU集中是否存在某个CPU
int CPU_COUNT(cpu_set_t *set)          //返回CPU集中CPU的个数
```

中断绑定CPU

中断请求（IRQ）有一个亲和度属性smp_affinity，smp_affinity决定允许哪些CPU核心处理IRQ。

某一特定IRQ的亲和度值储存在/proc/irq/IRP_NUMBER/smp_affinity文件中，储存的值是一个十六进制位掩码(hexadecimal bit-mask)，代表着系统的所有CPU核心。从低位到高位的每个bit代表从编号从0开始的CPU能否接受此IRQ，1表示接受，0表示拒绝。

cat /proc/interrupts查看所有设备的interrupts信息，第一列为IRP_NUMBER。(截图为部分中断信息)



```
lotus@tegra-ubuntu:~$ cat /proc/interrupts
           CPU0       CPU1       CPU2       CPU3       CPU4       CPU5       CPU6       CPU7       CPU8       CPU9       CPU10
11:    1907293    2155009     510297    1677919     282231     268483     235942     294196     693740     693298     666733     GICv3  27 Level     arch_timer
16:        100          0          0          0          0          0          0          0          0          0          0     GICv3 682 Edge      ivc407
18:          9          0          0          0          0          0          0          0          0          0          0     GICv3 684 Edge      gr-virt
20:     159919          0          0          0          0          0          0          0          0          0          0     GICv3 686 Edge      vblk
25:         53          0          0          0          0          0          0          0          0          0          0     GICv3 691 Edge      vblk
26:         52          0          0          0          0          0          0          0          0          0          0     GICv3 692 Edge      vblk
27:        187          0          0          0          0          0          0          0          0          0          0     GICv3 693 Edge      vblk
29:         23          0          0          0          0          0          0          0          0          0          0     GICv3 695 Edge      vblk
30:         61          0          0          0          0          0          0          0          0          0          0     GICv3 696 Edge      vblk
31:          4          0          0          0          0          0          0          0          0          0          0     GICv3 697 Edge      tegra_hv_pm_ctl
32:      13116          0          0          0          0          0          0          0          0          0          0     GICv3 698 Edge      bpmp_irq_handler
33:         28          0          0          0          0          0          0          0          0          0          0     GICv3 699 Edge      bpmp_irq_handler
34:          2          0          0          0          0          0          0          0          0          0          0     GICv3 700 Edge      bpmp_irq_handler
36:          1          0          0          0          0          0          0          0          0          0          0     GICv3 702 Edge      vse
37:          2          0          0          0          0          0          0          0          0          0          0     GICv3 703 Edge      ivc520
38:          2          0          0          0          0          0          0          0          0          0          0     GICv3 704 Edge      ivc521
39:          2          0          0          0          0          0          0          0          0          0          0     GICv3 705 Edge      ivc506
40:          0          0          0          0          0          0          0          0          0          0          0     GICv3 255 Level     mc_status
42:          0          0          0          0          0          0          0          0          0          0          0     GICv3 202 Level     arm-smmu global fault
```

cat /proc/irq/32/smp_affinity可以看到IRQ号为32的亲和度。orin的默认值为7ff，代表这个IRP能被所有CPU接受处理。

```
lotus@tegra-ubuntu:~$ cat /proc/irq/32/smp_affinity
7ff
```

echo 1 >/proc/irq/32/smp_affinity把IRQ号为32的亲和度值设为1,代表这个IRP仅能被CPU0接受处理。（注：需要在root下执行）

```
root@tegra-ubuntu:~# echo 1 > /proc/irq/32/smp_affinity
root@tegra-ubuntu:~# cat /proc/irq/32/smp_affinity
001
root@tegra-ubuntu:~#
```

代码级别的亲和度设定:

linux提供系统调用irq_set_affinity来设置IRQ的CPU亲和性。

```
int irq_set_affinity(unsigned int irq, const cpu_set_t *mask);
/* 第一个参数irq表示中断编号，第二个指针参数mask通常指向一个32位整数，整数的二进制下的每个位表示一个CPU，1表示irq使用该CPU，0相反。*/
/* *mask 通过CPU宏设置 */
```

在默认情况下，有一个irqbalance进程在对IRQ进行负载均衡，在设置IRQ亲和性之前需要先暂停负载均衡进程，以保证手动绑定的更改不会被覆盖。

照此，我们可以据要求绑定指定IRQ到CPU。

进程/线程绑定CPU

设定某个进程的CPU亲和度

头文件包含：#include <sched.h>

```
int sched_setaffinity(pid_t pid, size_t cpusetsize, const cpu_set_t *mask);
/* 设定进程号为pid的进程运行在mask所设定的CPU上，第二个数cpusetsize是mask所指定的数的长度，通常为sizeof(cpu_set_t)。如果pid的值为0，则表示的是当前进程。*/
/* *mask 通过CPU宏设置 */
```

设定某个线程的CPU亲和度

头文件包含：#include <pthread.h>

```
int pthread_setaffinity_np(pthread_t thread, size_t cpusetsize, const cpu_set_t *cpuset);
/* 设定线程号为thread的线程运行在mask所设定的CPU上，第二个数cpusetsize是mask所指定的数的长度，通常为sizeof(cpu_set_t)。如果thread的值为0，则表示的是当前线程。*/
/* *mask 通过CPU宏设置 */
```

进程独占CPU

1.CPU隔离

如果想让特定进程或线程独占某一或某些CPU，我们需要避免其它进程运行在该CPU上。因此,我们所做的第一步是CPU隔离。

CPU隔离的方法:

- 修改Linux内核的启动参数isolcpus。 isolcpus将从线程调度器中移除选定的CPU,这些被移除的CPU称为"isolated" CPU. 若想要在被隔离的CPU上run进程,必须调用CPU亲和度相关的syscalls。

  具体的修改方法是在/boot/grub/grub.conf的kernel列最末尾加上isolcpus=x,y,... (代表将CPUx CPUy隔离)

- 另一方法利用了CPU亲和性的继承性，即子进程会继承父进程的CPU亲和性。由于所有进程都是init的子进程，我们可以设置init的CPU亲和性，这样一来，所有的进程都具有了与init相同的CPU亲和性。然后我们可以更改我们需要的进程的CPU亲和性来达到独占。

2.中断解除绑定

被隔离的CPU虽然没有进程run在上面,但是仍会收到interrupt。第二步我们需要将中断绑定到其他CPU上实现中断解除绑定。

3.绑定进程到CPU

在以上两步摒除了一部分外界的干扰后，最后一步绑定进程到隔离的CPU上。

进程调度策略、优先级配置

头文件包含：#include <sched.h>

```
int sched_setscheduler(pid_t pid, int policy, const struct sched_param *param);
/* policy有非实时策略：
SCHED_OTHER:标准循环分时策略；
SCHED_BATCH：用于“批处理”样式的进程执行；
SCHED_IDLE：用于运行优先级较低的后台作业；
实时策略：
SCHED_FIFO：先进先出策略；
SCHED_RR：循环策略。*/
/* 非实时调度策略sched_priority必须指定为0，实时策略sched_priority指定为1~99，且数字越大优先级越高 */
/* fork产生的子进程会继承调度策略和优先级 */
```

```
/* param优先级配置 */
struct sched_param param;
param.sched_priority = NUM;
```

特殊的策略：SCHED_DEADLINE

头文件包含：#include <sched.h> #include <sys/syscall.h>

```
int syscall(SYS_sched_setattr, pid_t pid, struct sched_attr *attr,
                    unsigned int flags);
```

- attr结构体设置

  需要设置的参数有：sched_runtime，sched_deadline，sched_period。sched_runtime需要略大于该线程的平均执行时间，可以设置为WCET；sched_deadline设置为相对时限；sched_period设置为任务周期。

  kernel要求sched_runtime <= sched_deadline <= sched_period，如果sched_period设置为0，它会被自动设置为与sched_deadline相同。并且所有这些值要求大于1024，单位为微秒。

- flag位为保留位，目前必须设置为0

SCHED_DEADLINE策略运行的线程将有最高的优先级（高于前面任意一种策略），当这个线程是就绪状态时，他会抢占任何其他策略的线程。该线程不能调用fork。

ORIN-X CPU调度能力测试及使用建议

以下通过指定线程数，测试用不同的核数的性能对比（注：启用多处理器时，线程被平均的绑定到启用的核上；每个线程被唤醒的间隔为10ms，所有时间单位均为us）

100线程

核数	平均核利用率	平均时延	最大时延
1	1%	5	45
2	0.5%	5	47
3	0.7%	3	28
4	0.5%	3	17
5	0.5%	3	15
6	0.5%	3	23
7	0.5%	3	26
8	0.5%	4	23
9	0.5%	4	23
10	0.5%	4	21

500线程

核数	平均核利用率	平均时延	最大时延
1	4%	10	110
2	4%	3	51
3	2%	3	97
4	2%	4	31
5	1.5%	3	43
6	1.5%	3	42
7	1.5%	3	30
8	1%	3	27
9	1%	3	35
10	1%	3	36

1000线程

核数	平均核利用率	平均时延	最大时延
1	37%	10	202
2	5%~10%	10	59
3	4%	7	77
4	4%	4	101
5	3%	3	61
6	3%	3	50
7	2%	3	43
8	2%	3	49
9	1.5%	3	25
10	1.5%	3	24

2000线程

核数	平均核利用率	平均时延	最大时延
1	35%	10	306
2	35%	20	194
3	25%	10	64
4	18%	8	129
5	10%	7	95
6	20%	6	75
7	5%	5	53
8	10%	5	55
9	5%	4	44
10	4%	4	78

3000线程

核数	平均核利用率	平均时延	最大时延
1	100%	N/A	N/A
2	45%	15	102
3	30%	10	112
4	25%	10	147
5	20%	8	67
6	10%	7	82
7	15%	6	65
8	15%	5	76
9	10%	5	88
10	6%	5	44

结论表格：

线程数	100	500	1000	2000	3000
最佳启用核数	5	7~8	9~10	9	10