Operating Systems

Introduction to Lab 4

Kernel Thread Management

Department of Computer Science & Technology
Tsinghua University
IIIS

Outline

- Work Flow & Key Data Structure
- Create & Execute Kernel Threads
- Schedule & Execute Kernel Threads

Work Flow & Key Data Structure

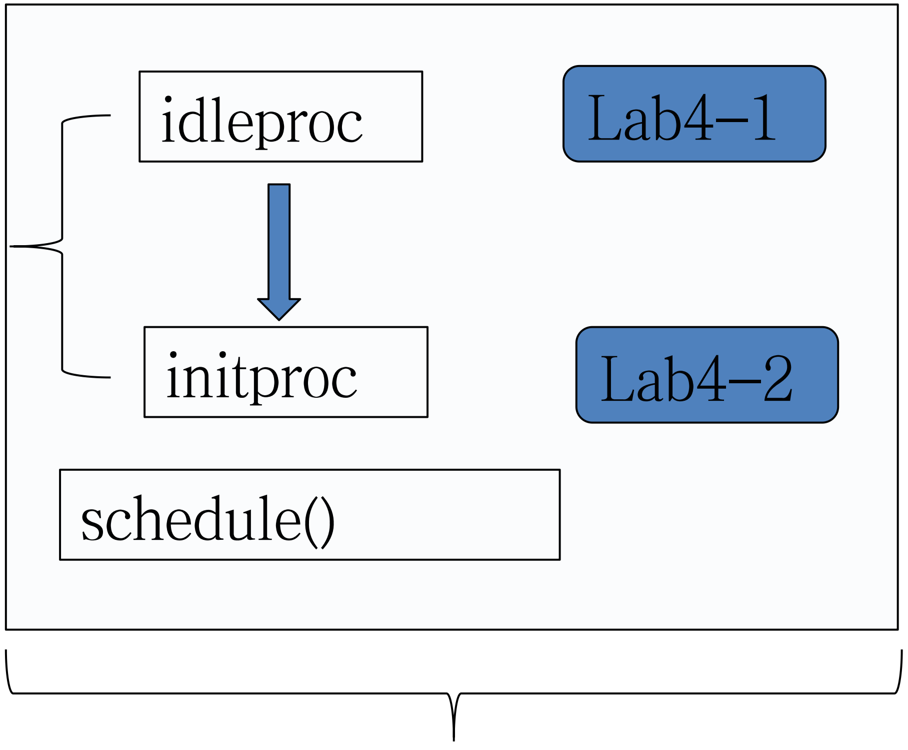
- Work Flow (\kern\init\init.c kern_init())
  - pmm_init()
  - pic_init()
  - idt_init()
  - vmm_init()

  - proc_init()   // init process table

  - ide_init()
  - swap_init()
  - …
  - cpu_idle()   // run idle process

Work Flow & Key Data Structure

- Work Flow
  - …
  - proc_init()
  - …
  - cpu_idle()



\kern\process\proc.[ch]

Work Flow & Key Data Structure

- Key Data Structure

uint32_t flags

char name[PROC_NAME_LEN + 1]

int pid

uintptr_t kstack

int runs

struct context context

struct proc_struct

uintptr_t cr3

struct mm_struct *mm

enum proc_state state

struct trapframe *tf

volatile bool need_resched

list_entry_t hash_link

list_entry_t list_link

struct proc_struct *parent

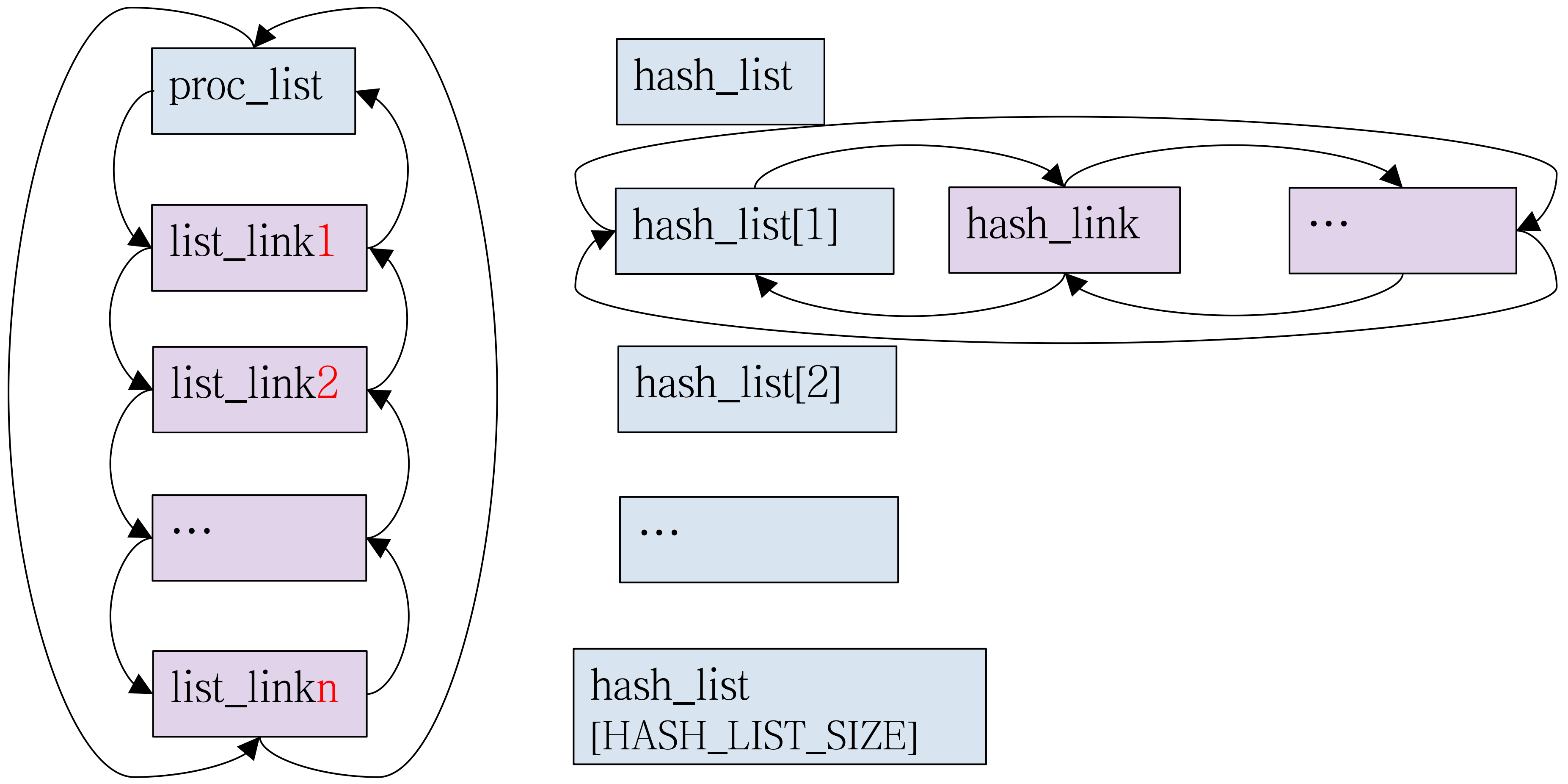
Work Flow & Key Data Structure

- Key structures (\kern\mm\vmm.h)

```
struct mm_struct {
        // linear list link which sorted by start addr of vma
        list_entry_t mmap_list;
        // current accessed vma, used for speed purpose
        struct vma_struct *mmap_cache;
        pde_t *pgdir;  // the PDT of these vma  =cr3=boot_cr3
        int map_count; // the count of these vma
        void *sm_priv;          // the private data for swap manager
};
```

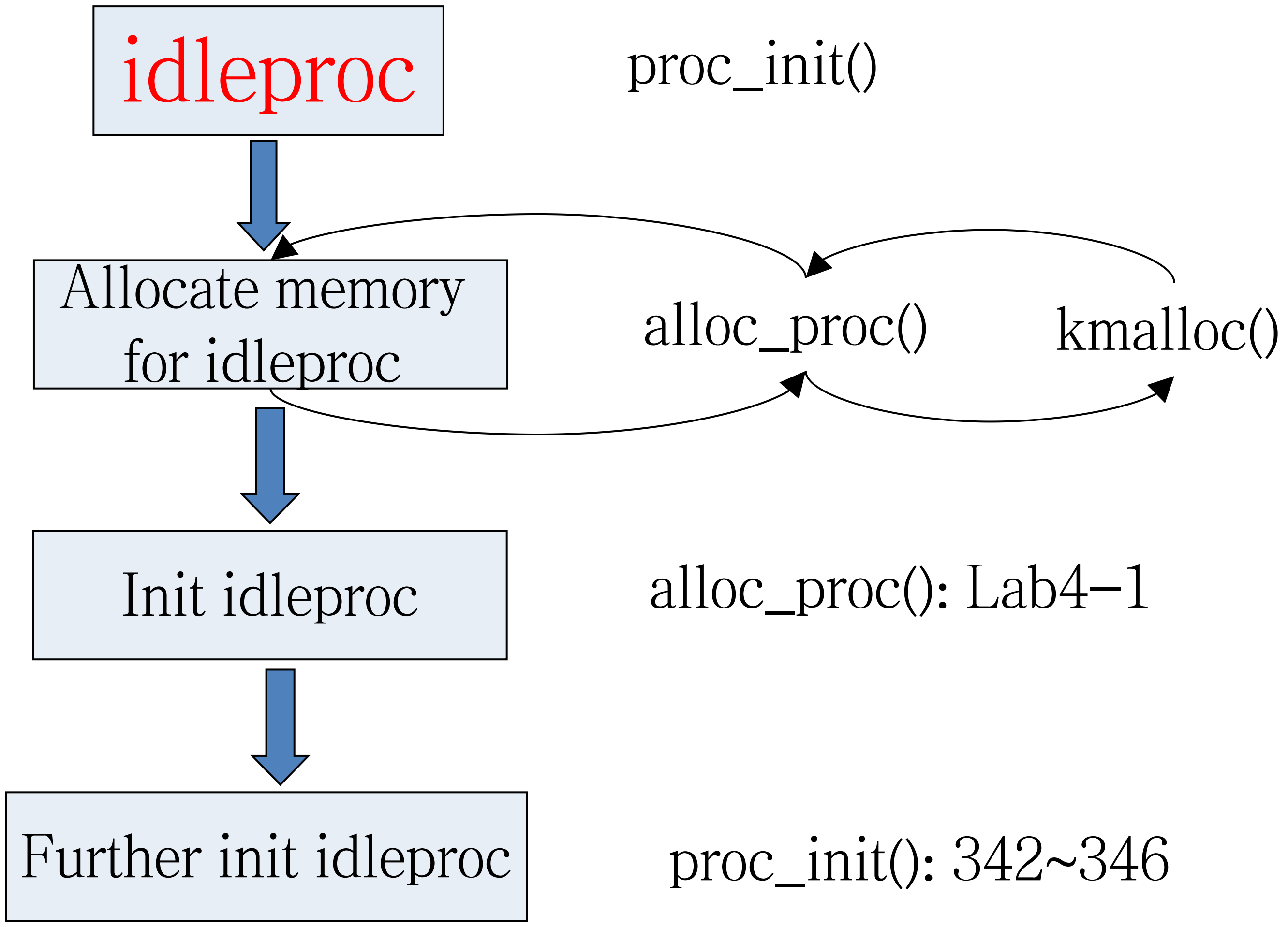
Work Flow & Key Data Structure

- Key Data Structure (\kern\process\proc.c)



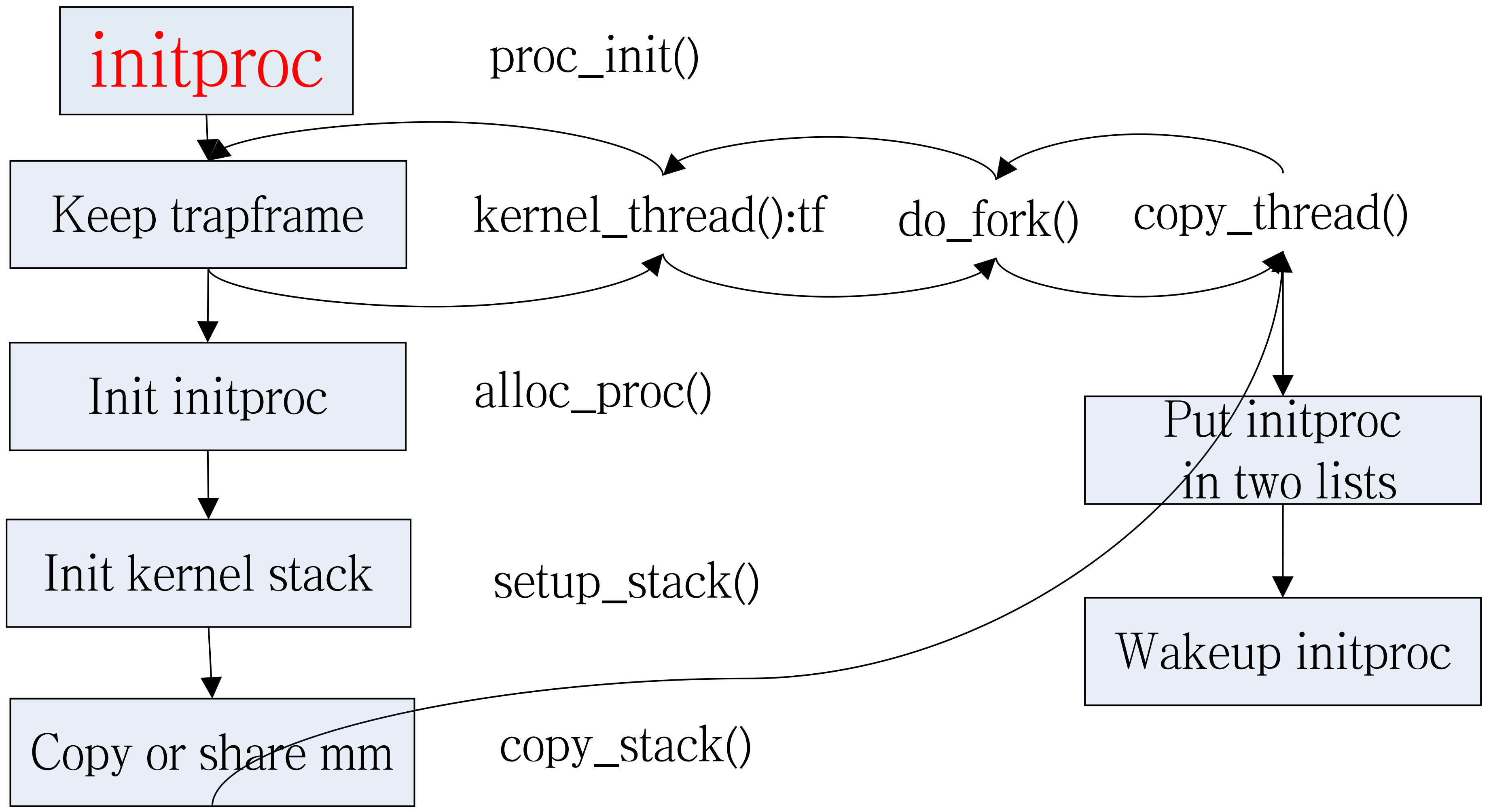
Create & Execute Kernel Thread

- Create the Zero Kernel Thread (\kern\process\proc.c)



Create & Execute Kernel Thread

- Create the 1st Kernel Thread (\kern\process\proc.c)



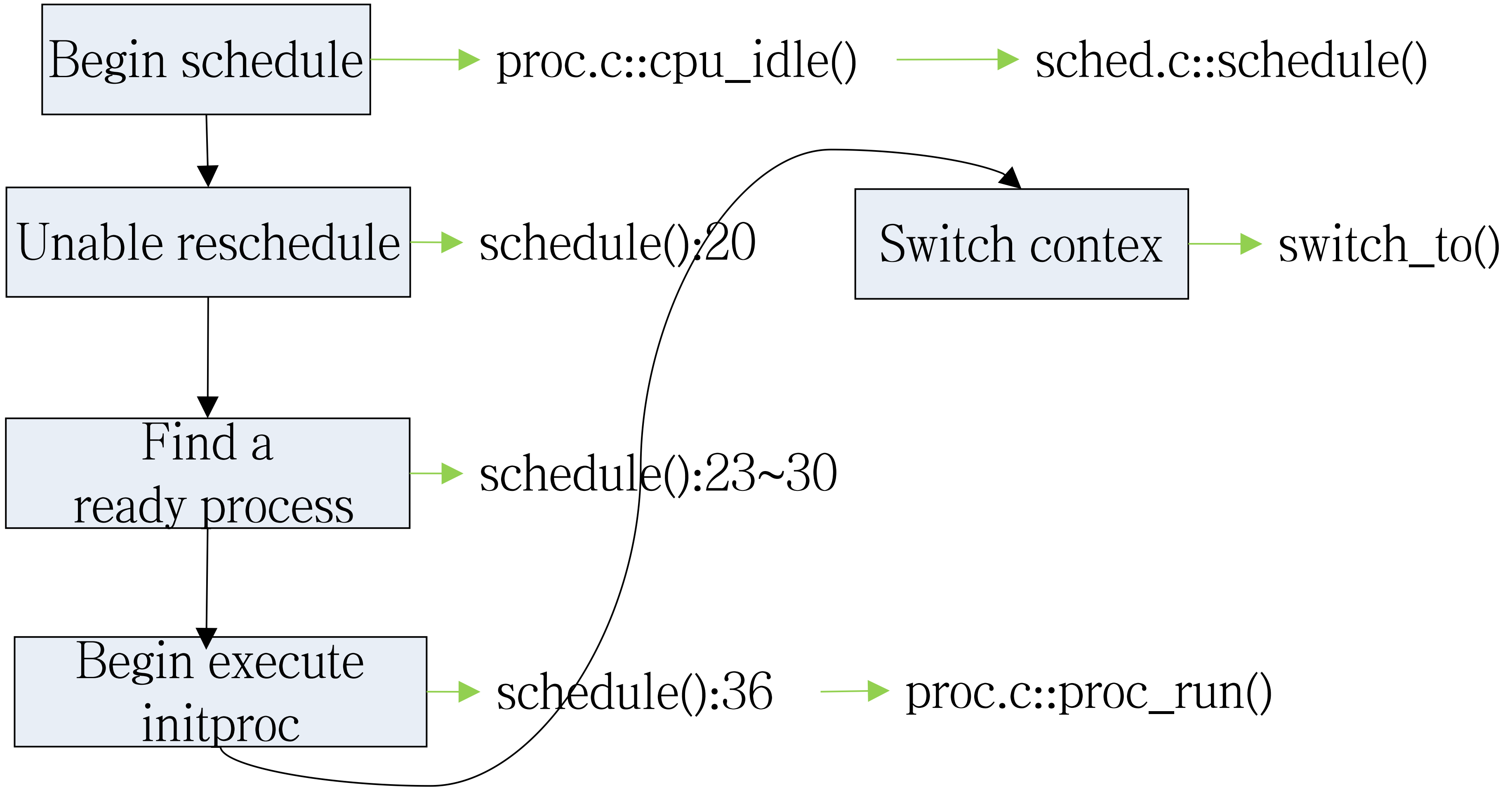
Create & Execute Kernel Thread

- Create the 1st Kernel Thread (\kern\process\proc.c)

```
initproc->tf= (proc->kstack+KSTACKSIZE) – sizeof (struct trapframe);

initproc->tf.tf_cs = KERNEL_CS;
initproc->tf.tf_ds = initproc->tf.tf_es = initproc->tf.tf_ss = KERNEL_DS;

initproc->tf.tf_regs.reg_ebx = (uint32_t)init_main;
initproc->tf.tf_regs.reg_edx = (uint32_t) ADDRESS of "Hello world!!";
initproc->tf.tf_eip = (uint32_t)kernel_thread_entry;

initproc->tf.tf_regs.reg_eax = 0;
initproc->tf.tf_esp = esp;
initproc->tf.tf_eflags |= FL_IF;
```

Schedule & Execute Kernel Thread

- Schedule Kernel Thread (\kern\process\proc.c, kern\schedule\sched.[ch])



That's all. Thanks!