Assignment 5 report

1.Please realize merging free blocks in default_free_pages()

code:

```
154   //----------------------合并空闲块--------------------
155    list_entry_t* frontier = list_prev(&(base->page_link));//the node before base->page_link in free_link
156    list_entry_t* next = list_next(&(base->page_link));//the node after base->page_link in free_link
157    if(next != NULL || next != &free_list){
158     struct Page* page = le2page(next,page_link);
159     if(base + base->property == page){// base can merge with the page after it
160       base->property += page->property;// update the page->property
161       ClearPageProperty(page);
162       list_del(next);// remove the page after base from free_list as it will be integrated into base
163     }
164    }
165    if(frontier != &free_list){
166     struct Page* page = le2page(frontier,page_link);
167     if(page + page->property == base){// base can merge with the page before it
168       page->property += base->property;// update the page->property
169       ClearPageProperty(base);
170       list_del(&(base->page_link));// remove base from free_list as it will be integrated into the page bef
171     }
172    }
173   //------------------------------------------------
```

result:

```
OpenSBI v0.6
   ____                    ______ ____ ______
  / __ \                  / ____|  _ \_   _|
 | |  | |_ __   ___ _ __ | (___ | |_) || |
 | |  | | '_ \ / _ \ '_ \ \___ \|  _ < | |
 | |__| | |_) |  __/ | | |____) | |_) || |_
  \____/| .__/ \___|_| |_|_____/|____/_____|
        | |
        |_|

Platform Name          : QEMU Virt Machine
Platform HART Features : RV64ACDFIMSU
Platform Max HARTs     : 8
Current Hart           : 0
Firmware Base          : 0x80000000
Firmware Size          : 120 KB
Runtime SBI Version    : 0.2

MIDELEG : 0x0000000000000222
MEDELEG : 0x000000000000b109
PMP0    : 0x0000000080000000-0x000000008001ffff (A)
PMP1    : 0x0000000000000000-0xffffffffffffffff (A,R,W,X)
os is loading ...
memory management: default_pmm_manager
physcial memory map:
  memory: 0x0000000007e00000, [0x0000000080200000, 0x0000000087ffffff].
check_alloc_page() succeeded!
QEMU: Terminated
xin12110714@xin12110714-virtual-machine:~/Desktop/lab9$
```

2.Realize bestfit in best_fit_pmm.c

code:

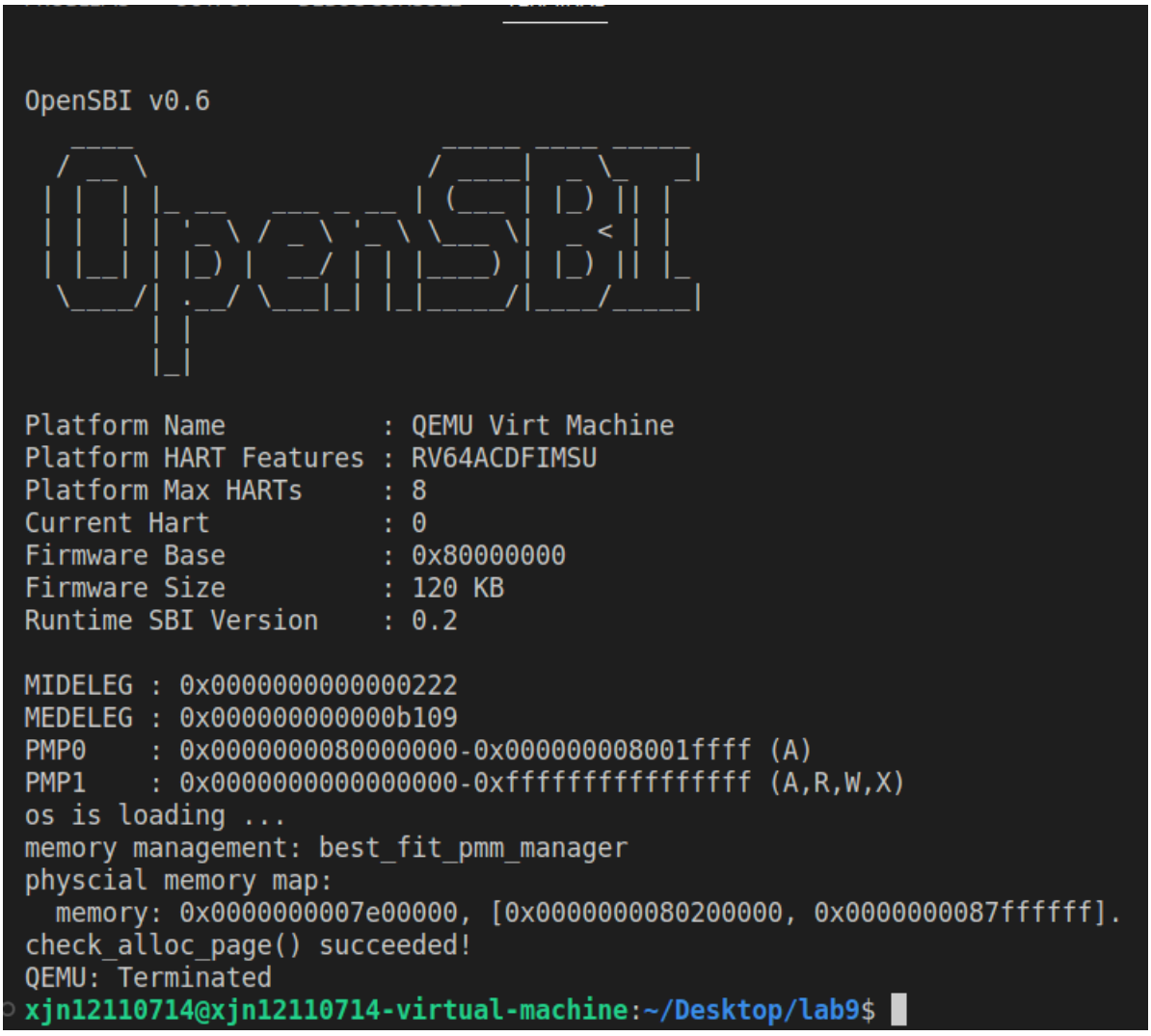
```
12  static void
13  best_fit_init(void)
14  {
15      list_init(&free_list);// initialize the free_list
16      nr_free = 0;// set # of free blocks as zero
17  }
18
19  static void
20  best_fit_init_memmap(struct Page *base, size_t n)
21  {
22      //set base as a page with property of and insert it into free_list
23      assert(n > 0);
24      struct Page *p = base;
25      for (; p != base + n; p ++) {
26          assert(PageReserved(p));
27          p->flags = p->property = 0;
28          set_page_ref(p, 0);
29      }
30      base->property = n;
31      SetPageProperty(base);
32      nr_free += n;
33      if (list_empty(&free_list)) {
34          list_add(&free_list, &(base->page_link));
35      } else {
36          list_entry_t* le = &free_list;
37          while ((le = list_next(le)) != &free_list) {
38              struct Page* page = le2page(le, page_link);
39              if (base < page) {
40                  list_add_before(le, &(base->page_link));
41                  break;
```

```
42              } else if (list_next(le) == &free_list) {
43                  list_add(le, &(base->page_link));
44              }
45          }
46      }
47  }
48
49  static struct Page *
50  best_fit_alloc_pages(size_t n)
51  {
52      struct Page *page = NULL;
53      assert(n > 0);
54      if (n > nr_free) { // if n > nr_free, none of free blocks satisfied
55          return NULL;
56      }
57      list_entry_t *le = &free_list;
58      size_t min = __SIZE_MAX__;
59      while ((le = list_next(le)) != &free_list) { // traversal to find the page whose property is larger t
60          struct Page *p = le2page(le, page_link);
61          if (p->property >= n) {
62              if(p->property < min){
63                  min = p->property;
64                  page = p;
65              }
66          }
67      }
68      if (page != NULL) {// if we find such a page then allocate the page we need
69          list_entry_t* prev = list_prev(&(page->page_link));
70          list_del(&(page->page_link));
```

```
        if (page->property > n) {
            struct Page *p = page + n;
            p->property = page->property - n;
            SetPageProperty(p);
            list_add(prev, &(p->page_link));
        }
        nr_free -= n;
        ClearPageProperty(page);
    }
    return page;

}

static void
best_fit_free_pages(struct Page *base, size_t n)
{
    // free a series of blocks started from base with property n
    assert(n > 0);
    struct Page *p = base;
    for (; p != base + n; p ++) {
        assert(!PageReserved(p) && !PageProperty(p));
        p->flags = 0;
        set_page_ref(p, 0);
    }
    base->property = n;
    SetPageProperty(base);
    nr_free += n;

    if (list_empty(&free_list)) {
```

```
    } else {
        list_entry_t* le = &free_list;
        while ((le = list_next(le)) != &free_list) {
            struct Page* page = le2page(le, page_link);
            if (base < page) {
                list_add_before(le, &(base->page_link));
                break;
            } else if (list_next(le) == &free_list) {
                list_add(le, &(base->page_link));
            }
        }
    }

    //----------------------合并空闲块-------------------
    list_entry_t* frontier = list_prev(&(base->page_link));
    list_entry_t* next = list_next(&(base->page_link));
    if(next != NULL || next != &free_list){
     struct Page* page = le2page(next,page_link);
     if(base + base->property == page){
       base->property += page->property;
       ClearPageProperty(page);
       list_del(next);
     }
    }
    if(frontier != &free_list){
     struct Page* page = le2page(frontier,page_link);
     if(page + page->property == base){
       page->property += base->property;
       ClearPageProperty(base);
       list_del(&(base->page_link));
```

result:


OpenSBI v0.6
Platform Name : QEMU Virt Machine
Platform HART Features : RV64ACDFIMSU
Platform Max HARTs : 8
Current Hart : 0
Firmware Base : 0x80000000
Firmware Size : 120 KB
Runtime SBI Version : 0.2
MIDELEG : 0x0000000000000222
MEDELEG : 0x000000000000b109
PMP0 : 0x0000000080000000-0x000000008001ffff (A)
PMP1 : 0x0000000000000000-0xffffffffffffffff (A,R,W,X)
os is loading ...
memory management: best_fit_pmm_manager
physcial memory map:
memory: 0x0000000007e00000, [0x0000000080200000, 0x0000000087ffffff].
check_alloc_page() succeeded!
QEMU: Terminated
xjn12110714@xjn12110714-virtual-machine:~/Desktop/lab9$