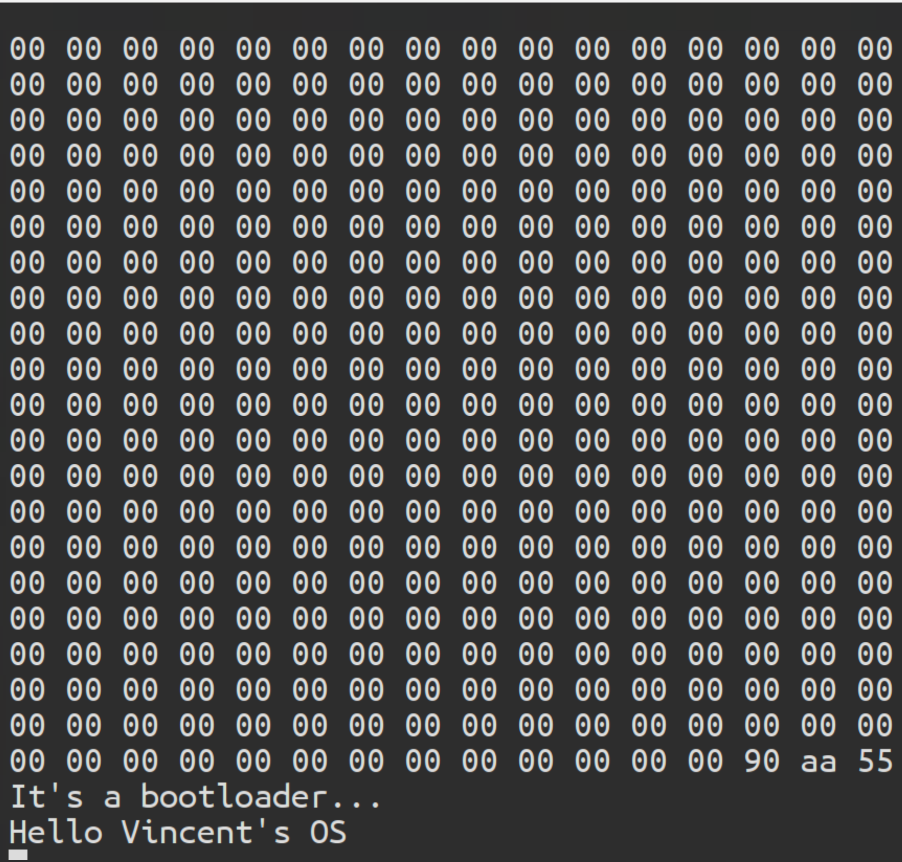
OS EXP
Design Review
#01



```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 90 aa 55
It's a bootloader...
Hello Vincent's OS
```

蔡润泽&张博源

01

Overview on Codes

Overview on Codes
bootblock.s

```
main:
    # 1) task1 call BIOS print string "It's bootblock!"
    la $a0,msg
    lw $t2, printstr
    jal $t2

    # 2) task2 call BIOS read kernel in SD card and jump to kernel start
    lw $a0, kernel
    li $a1, 0x200
    li $a2, 0x200
    lw $t2,read_sd_card
    jal $t2
    lw $t2,kernel_main
    jal $t2
```

Overview on Codes
kernek.c

```c
void __attribute__((section(".entry_function"))) _start(void)
{

    // Call PMON BIOS printstr to print message "Hello OS!"
    void (*p)(char *string);
    p=0x80011100;

    (*p)("Hello Vincent's OS\n");

    return;
}
```

Overview on Codes
createimage.c

```c
Elf32_Phdr *read_exec_file(FILE *opfile)
{
    Elf32_Phdr *phdr_ptr;
    Elf32_Ehdr head;
    phdr_ptr=(Elf32_Phdr*)malloc(sizeof(Elf32_Phdr));

    fread(&head,sizeof(Elf32_Ehdr),1,opfile);
    fseek(opfile,head.e_phoff,SEEK_SET);
    fread(phdr_ptr,sizeof(Elf32_Phdr)*head.e_phnum,1,opfile);
    //printf("e_phnum:%d\n",head.e_phnum);
    // printf("addr:%x\n",phdr_ptr);
    return phdr_ptr;
}
```

Overview on Codes
kernek.c

```c
uint8_t count_kernel_sectors(Elf32_Phdr *Phdr)
{
    uint8_t num;
    num=(Phdr->p_memsz-1)/512+1;            //结果上取整
    // printf("p_memsz:%d\n",Phdr->p_memsz);
    // printf("num:%d\n",num);
    return num;

}
```

Overview on Codes
kernek.c

```c
void write_bootblock(FILE *image, FILE *file, Elf32_Phdr *phdr)
{
    char *temp;
    temp=(char *)malloc(sizeof(char)*512);
    char zero[512]="";
    uint16_t end=0x55aa;

    fseek(file, phdr->p_offset, SEEK_SET);          //get program from bootblock
    fread(temp,phdr->p_filesz,1,file);

    fwrite(temp,phdr->p_filesz,1,image);            //write bootblock to image

    if(phdr->p_filesz%512){
        fwrite(zero,1,512-2-phdr->p_filesz%512,image);  //add zero
        fwrite(&end,2,1,image);                         //add 0x55aa in the end
    }

    free(temp);
}
```

```c
void write_kernel(FILE *image, FILE *knfile, Elf32_Phdr *Phdr, int kernelsz)
{
    Elf32_Ehdr kn_head;                         //get Ehdr of kernel
    fseek(knfile,0,SEEK_SET);
    fread(&kn_head,sizeof(Elf32_Ehdr),1,knfile);
    fseek(knfile,0,SEEK_SET);

    char *temp;
    char zero[512]="";
    int i=0;

    Elf32_Phdr total_phdr [kn_head.e_phnum];    //get all Phdr(s) of kernel
    fseek(knfile, kn_head.e_phoff, SEEK_SET);
    fread(total_phdr, sizeof(Elf32_Phdr), kn_head.e_phnum, knfile);

    temp = (char*)malloc(sizeof(char) * kernelsz);

    for(i = 0; i < kn_head.e_phnum; i++){       //get program from kernel
        fseek(knfile, total_phdr[i].p_offset, SEEK_SET);
        fread(temp,total_phdr[i].p_filesz, 1, knfile);
    }

    fwrite(temp, kernelsz, 1, image);           //write kernel to image

     if(kernelsz % 512){                        //add zero in the end
         fwrite(zero, 1, 512-kernelsz%512, image);
     }
    free(temp);
}
```

Overview on Codes
kernek.c

```c
void record_kernel_sectors(FILE *image, uint8_t kernelsz)
{
    fseek(image,509,SEEK_SET);
    fwrite(&kernelsz,1,1,image); //kernelsz is kernel's sector_num
}

void extent_opt(Elf32_Phdr *Phdr_bb, Elf32_Phdr *Phdr_k, int kernelsz)
{
    printf("kernelsz : %d\n",kernelsz * 512);
    printf("virtual addredd: 0x%x\n", Phdr_bb->p_vaddr);
    printf("segment size in file: 0x%x\n", Phdr_bb->p_filesz+Phdr_k->p_filesz);
    printf("segment size in memory: 0x%x\n", Phdr_bb->p_memsz+Phdr_k->p_memsz);

}
```

Overview on Codes
kernek.c

```c
void record_kernel_sectors(FILE *image, uint8_t kernelsz)
{
    fseek(image,509,SEEK_SET);
    fwrite(&kernelsz,1,1,image); //kernelsz is kernel's sector_num
}

void extent_opt(Elf32_Phdr *Phdr_bb, Elf32_Phdr *Phdr_k, int kernelsz)
{
    printf("kernelsz : %d\n",kernelsz * 512);
    printf("virtual addredd: 0x%x\n", Phdr_bb->p_vaddr);
    printf("segment size in file: 0x%x\n", Phdr_bb->p_filesz+Phdr_k->p_filesz);
    printf("segment size in memory: 0x%x\n", Phdr_bb->p_memsz+Phdr_k->p_memsz);

}
```

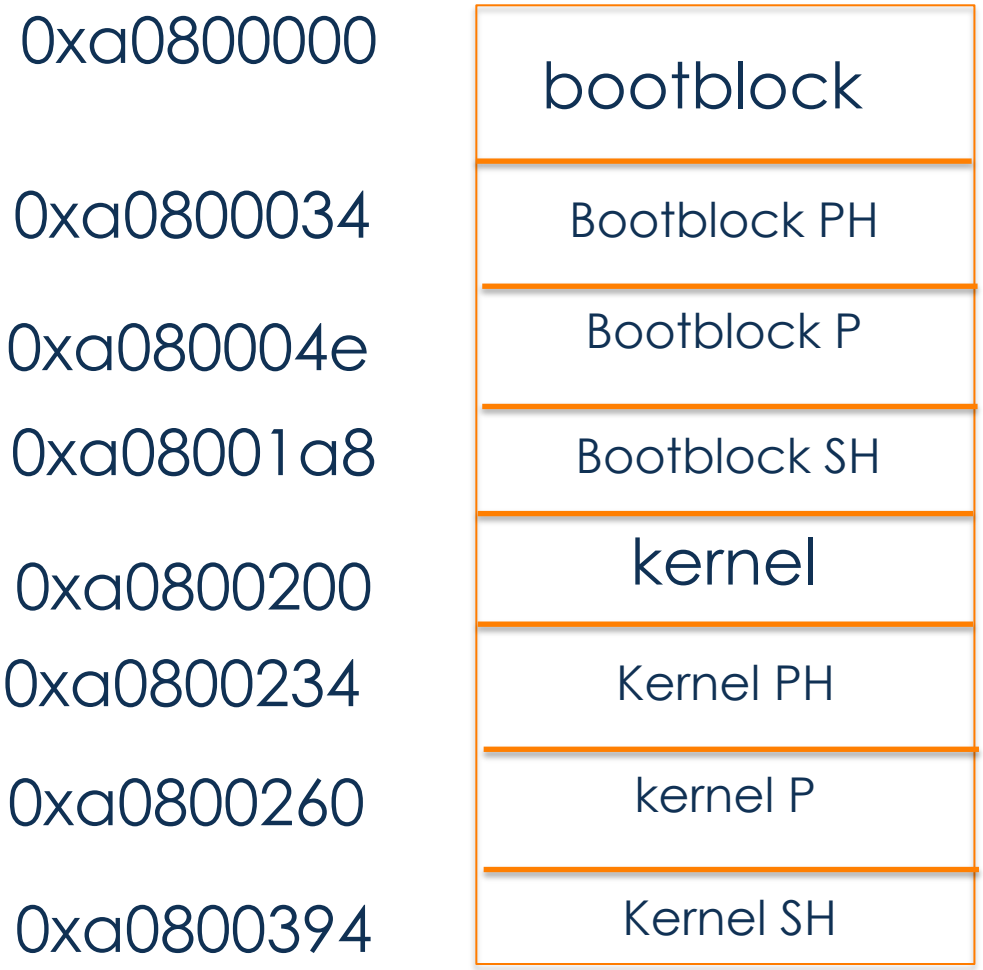
```c
int get_kernelsz( FILE *knfile, Elf32_Phdr *Phdr){
    Elf32_Ehdr kn_head;
    int i=0,total_size=0;

    fseek(knfile,0,SEEK_SET);                       //get Ehdr of kernel
    fread(&kn_head,sizeof(Elf32_Ehdr),1,knfile);
    fseek(knfile,0,SEEK_SET);


    Elf32_Phdr total_phdr [kn_head.e_phnum];        //get all Phdr(s) of kernel
    fseek(knfile, kn_head.e_phoff, SEEK_SET);
    fread(total_phdr, sizeof(Elf32_Phdr), kn_head.e_phnum, knfile);


    for(i = 0; i < kn_head.e_phnum; i++){           //get total size of kernel
        fseek(knfile, total_phdr[i].p_offset, SEEK_SET);
        total_size += total_phdr[i].p_filesz;
    }
    fseek(knfile, 0, SEEK_SET);                     //ptr reset
    return total_size;

}
```

```c
int main()
{
    uint8_t sector_num;int kernelsz;
    FILE* bootblock_file,*kernel_file;
    Elf32_Phdr *bootblock_phdr,*kernel_phdr;

    bootblock_file=fopen("bootblock","rb");
    kernel_file=fopen("kernel","rb");

    bootblock_phdr=read_exec_file(bootblock_file);
    kernel_phdr=read_exec_file(kernel_file);
    sector_num=count_kernel_sectors(kernel_phdr);
    FILE* image_file=fopen("image","wb");

    write_bootblock(image_file,bootblock_file,bootblock_phdr);
    kernelsz=get_kernelsz(kernel_file,kernel_phdr);
    write_kernel(image_file,kernel_file,kernel_phdr,kernelsz);

    record_kernel_sectors(image_file,(uint8_t)kernelsz);
    extent_opt(bootblock_phdr,kernel_phdr,kernelsz);

    fclose(bootblock_file);
    fclose(kernel_file);
    fclose(image_file);
    return 0;
}
```

02

Q&A

Show the in-memory layout of your bootblock and kernel



How do you invoke BIOS function? Show example code to invoke read_sd_card .

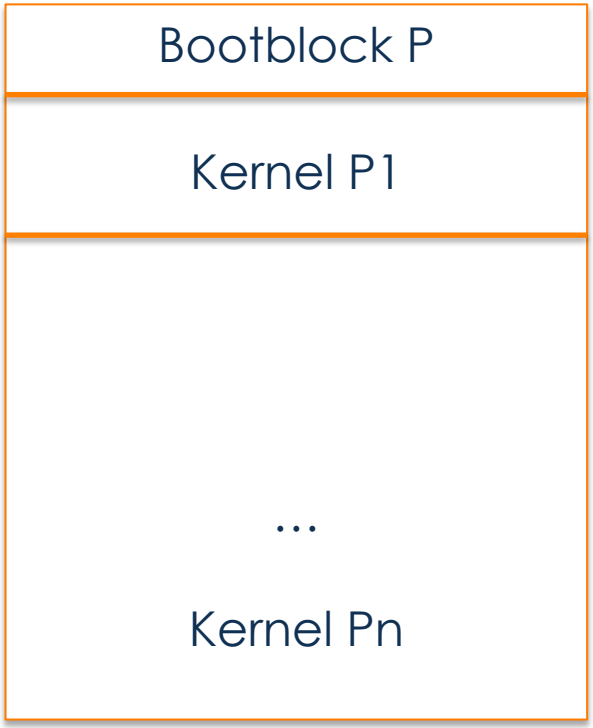
```
lw $a0, kernel
li $a1, 0x200
li $a2, 0x200
lw $t2,read_sd_card
jal $t2
```

Or

```
void (*r)(void *addr,int offset,int size);
r= 0x80011000;

(*r)(addr,offset,size);
```

How do you combine kernel.o and bootloader into an bootable image?

The same as kernel and bootloader.



03

Next Week

Next Week

main() :

argc , argv

Version :

2019 (itoa)

Bouns

Thanks