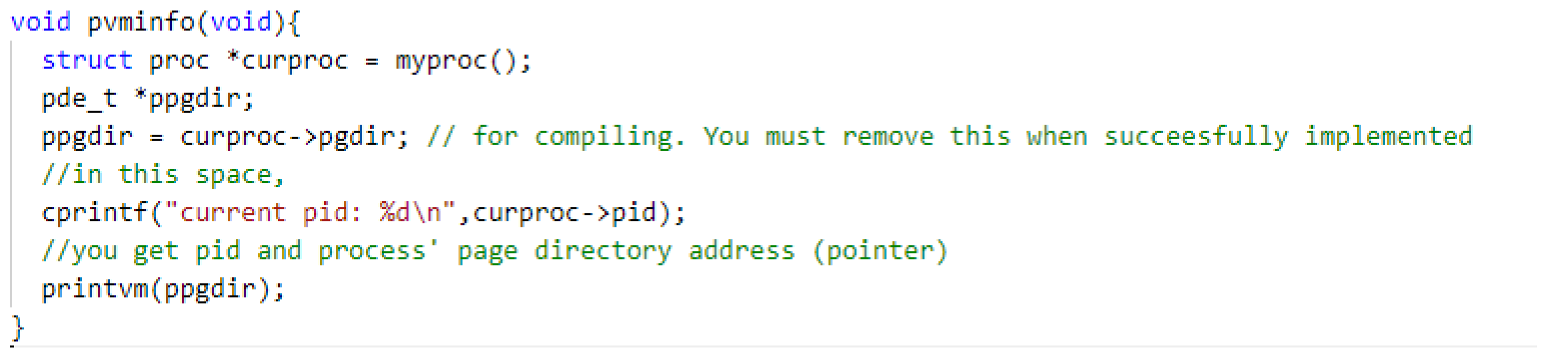
pvminfo

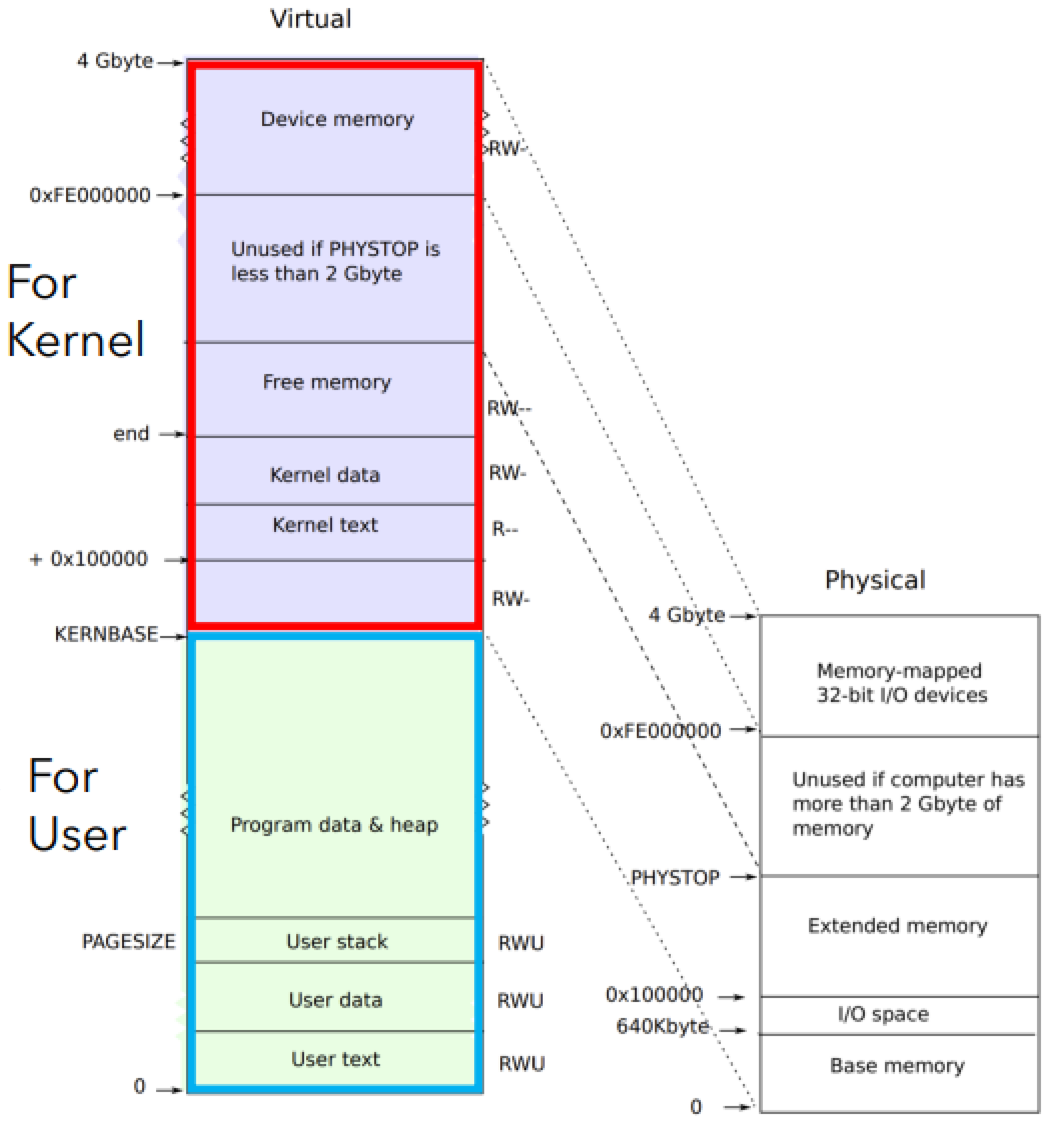


```
void pvminfo(void){
  struct proc *curproc = myproc();
  pde_t *ppgdir;
  ppgdir = curproc->pgdir; // for compiling. You must remove this when succeesfully implemented
  //in this space,
  cprintf("current pid: %d\n",curproc->pid);
  //you get pid and process' page directory address (pointer)
  printvm(ppgdir);
}
```

현재 프로세스를 myproc(); 함수를 curproc에 불러와준 후
curproc()의 page directory 주소를가지고 온 후 printvm에 전달하는 함수이다

printvm

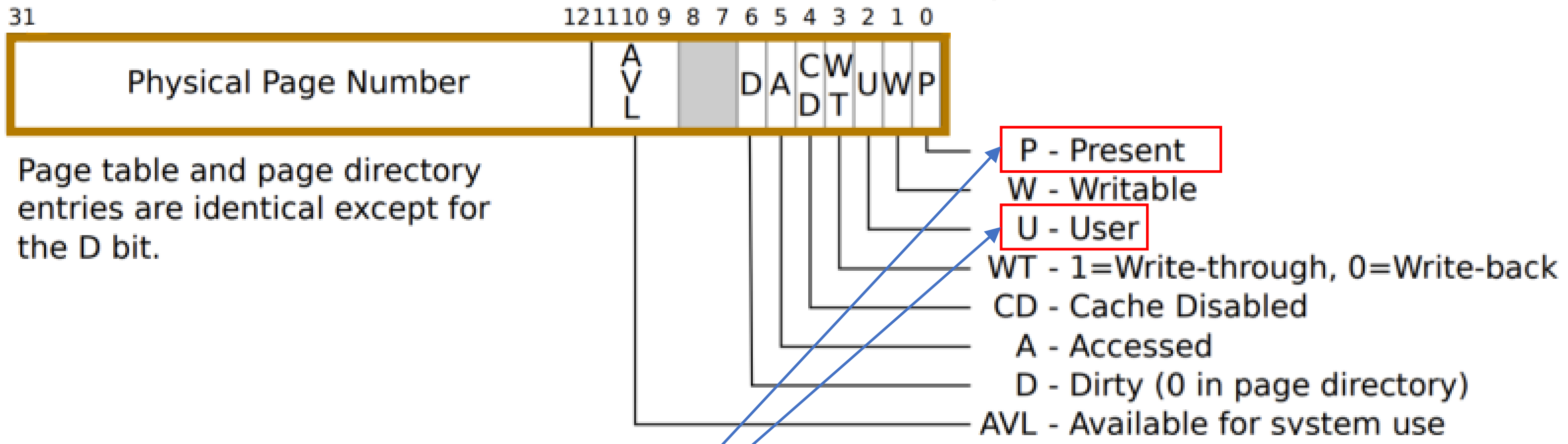
```c
void
printvm(pde_t *pgdir)
{
  pde_t *pde;
  pde_t *pgtab;
  pde_t *pte;
  cprintf("Page directory VA: 0x%x \n",pgdir);
  int i;
  int j;

  for(i = 0; i < 512; i++){
    pde = &pgdir[i];
    if ((*pde & 0x005) ==  0x005 ){
      cprintf("---%d: PDE: 0x%x PA: 0x%x \n",i,*pde,PTE_ADDR(pgdir));
      pgtab = (pte_t*)P2V(PTE_ADDR(*pde));
      for(j = 0; j < NPDENTRIES; j++){
        pte = &pgtab[j];
        if ((*pte & 0x005) == 0x005 ){
          cprintf("------%d: PTE: 0x%x PA: 0x%x \n",j,*pte,PTE_ADDR(*pde));
        }
      }
    }
  }
}
```



우리가 인쇄해야 하는 것은 VM for User 이므로 NPDENTRIES의 절반인 0~511까지만 pde를 for loop 문으로 탐색해야 한다.

printvm

```
void
printvm(pde_t *pgdir)
{
  pde_t *pde;
  pde_t *pgtab;
  pde_t *pte;
  cprintf("Page directory VA: 0x%x \n",pgdir);
  int i;
  int j;

  for(i = 0; i < 512; i++){
    pde = &pgdir[i];
    if ((*pde & 0x005) ==  0x005 ){
      cprintf("---%d: PDE: 0x%x PA: 0x%x \n",i,*pde,PTE_ADDR(pgdir));
      pgtab = (pte_t*)P2V(PTE_ADDR(*pde));
      for(j = 0; j < NPDENTRIES; j++){
        pte = &pgtab[j];
        if ((*pte & 0x005) == 0x005 ){
          cprintf("------%d: PTE: 0x%x PA: 0x%x \n",j,*pte,PTE_ADDR(*pde));
        }
      }
    }
  }
}
```



Page table and page directory entries are identical except for the D bit.

00000000101

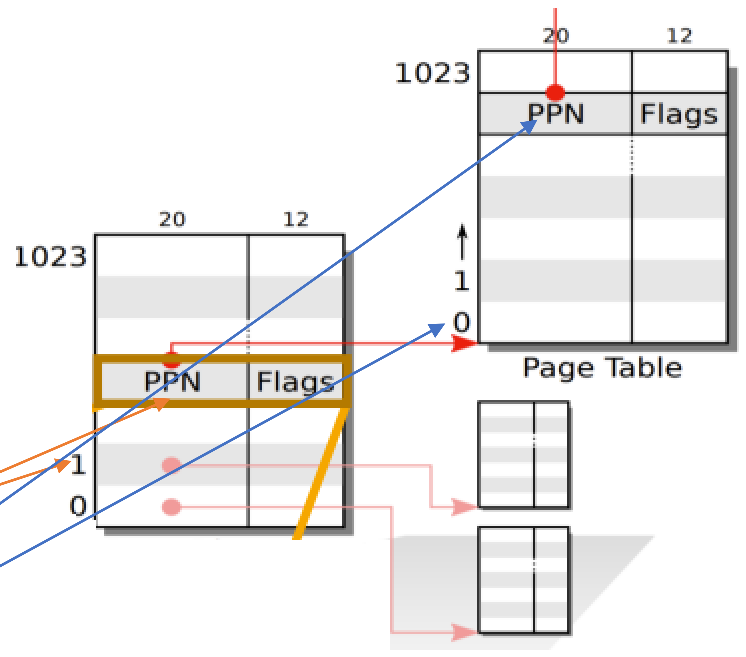
우리가 인쇄해야 하는 것은 VM for User 이면서 활성화 되어 있는 즉 0번째와 2번째 비트가 1인 것이므로
0x005 = 00000000101을 곱하여 0x005 가 나왔는지 확인한다.

printvm

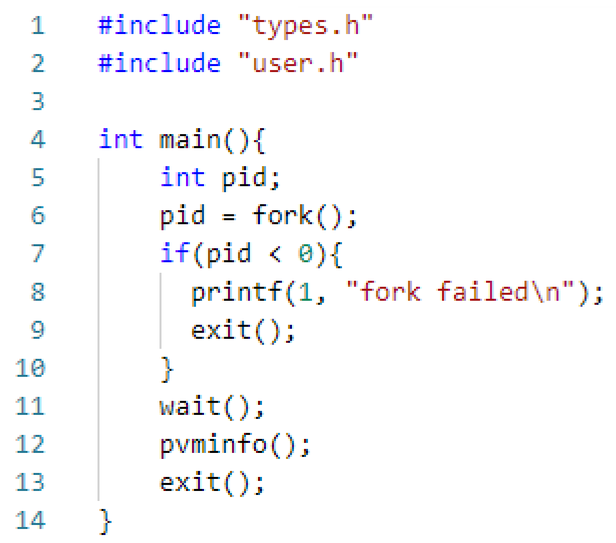
```c
void
printvm(pde_t *pgdir)
{
  pde_t *pde;
  pde_t *pgtab;
  pde_t *pte;
  cprintf("Page directory VA: 0x%x \n",pgdir);
  int i;
  int j;

  for(i = 0; i < 512; i++){
    pde = &pgdir[i];
    if ((*pde & 0x005) ==  0x005 ){
      cprintf("---%d: PDE: 0x%x PA: 0x%x \n",i,*pde,PTE_ADDR(pgdir));
      pgtab = (pte_t*)P2V(PTE_ADDR(*pde));
      for(j = 0; j < NPDENTRIES; j++){
        pte = &pgtab[j];
        if ((*pte & 0x005) == 0x005 ){
          cprintf("------%d: PTE: 0x%x PA: 0x%x \n",j,*pte,PTE_ADDR(*pde));
        }
      }
    }
  }
}
```

PA directory 주소(pgdir)에서 PA 추출

PDE에서 PPN 추출

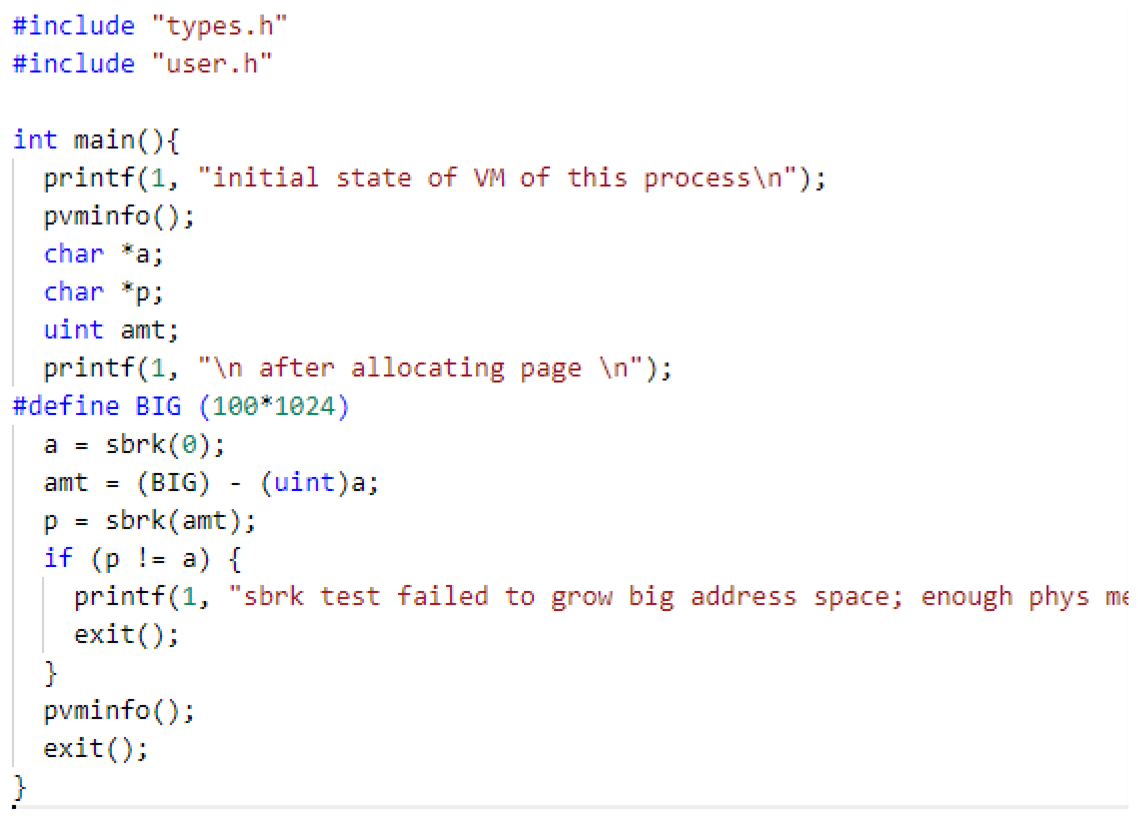


firsttest

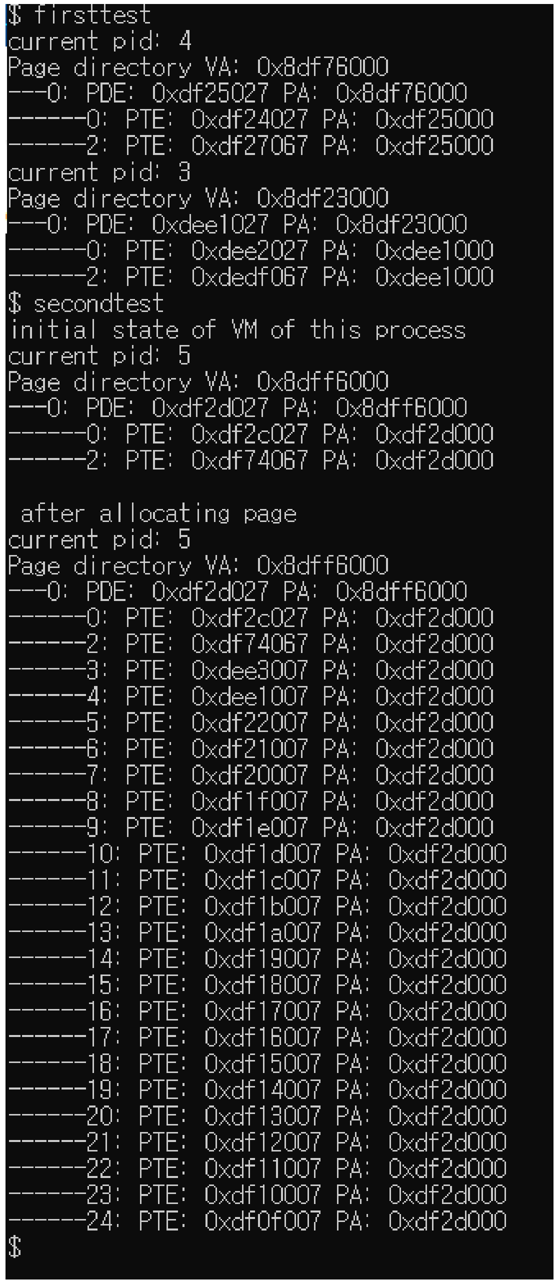


```c
#include "types.h"
#include "user.h"

int main(){
    int pid;
    pid = fork();
    if(pid < 0){
      printf(1, "fork failed\n");
      exit();
    }
    wait();
    pvminfo();
    exit();
}
```

secondtest



```c
#include "types.h"
#include "user.h"

int main(){
  printf(1, "initial state of VM of this process\n");
  pvminfo();
  char *a;
  char *p;
  uint amt;
  printf(1, "\n after allocating page \n");
#define BIG (100*1024)
  a = sbrk(0);
  amt = (BIG) - (uint)a;
  p = sbrk(amt);
  if (p != a) {
    printf(1, "sbrk test failed to grow big address space; enough phys me
    exit();
  }
  pvminfo();
  exit();
}
```



```
$ firsttest
current pid: 4
Page directory VA: 0x8df76000
---0: PDE: 0xdf25027 PA: 0x8df76000
------0: PTE: 0xdf24027 PA: 0xdf25000
------2: PTE: 0xdf27067 PA: 0xdf25000
current pid: 3
Page directory VA: 0x8df23000
---0: PDE: 0xdee1027 PA: 0x8df23000
------0: PTE: 0xdee2027 PA: 0xdee1000
------2: PTE: 0xdedf067 PA: 0xdee1000
$ secondtest
initial state of VM of this process
current pid: 5
Page directory VA: 0x8dff6000
---0: PDE: 0xdf2d027 PA: 0x8dff6000
------0: PTE: 0xdf2c027 PA: 0xdf2d000
------2: PTE: 0xdf74067 PA: 0xdf2d000

 after allocating page
current pid: 5
Page directory VA: 0x8dff6000
---0: PDE: 0xdf2d027 PA: 0x8dff6000
------0: PTE: 0xdf2c027 PA: 0xdf2d000
------2: PTE: 0xdf74067 PA: 0xdf2d000
------3: PTE: 0xdee3007 PA: 0xdf2d000
------4: PTE: 0xdee1007 PA: 0xdf2d000
------5: PTE: 0xdf22007 PA: 0xdf2d000
------6: PTE: 0xdf21007 PA: 0xdf2d000
------7: PTE: 0xdf20007 PA: 0xdf2d000
------8: PTE: 0xdf1f007 PA: 0xdf2d000
------9: PTE: 0xdf1e007 PA: 0xdf2d000
------10: PTE: 0xdf1d007 PA: 0xdf2d000
------11: PTE: 0xdf1c007 PA: 0xdf2d000
------12: PTE: 0xdf1b007 PA: 0xdf2d000
------13: PTE: 0xdf1a007 PA: 0xdf2d000
------14: PTE: 0xdf19007 PA: 0xdf2d000
------15: PTE: 0xdf18007 PA: 0xdf2d000
------16: PTE: 0xdf17007 PA: 0xdf2d000
------17: PTE: 0xdf16007 PA: 0xdf2d000
------18: PTE: 0xdf15007 PA: 0xdf2d000
------19: PTE: 0xdf14007 PA: 0xdf2d000
------20: PTE: 0xdf13007 PA: 0xdf2d000
------21: PTE: 0xdf12007 PA: 0xdf2d000
------22: PTE: 0xdf11007 PA: 0xdf2d000
------23: PTE: 0xdf10007 PA: 0xdf2d000
------24: PTE: 0xdf0f007 PA: 0xdf2d000
$
```