Ví dụ viết system call SC_Create

Hai thủ tục copy vùng nhớ từ User Space vào System Space và ngược lại

```
/*
Input: - User space address (int)
       - Limit of buffer (int)
Output:- Buffer (char*)
Purpose: Copy buffer from User memory space to System memory space
*/

char* User2System(int virtAddr,int limit)
{
 int i;// index
 int oneChar;
 char* kernelBuf = NULL;

 kernelBuf = new char[limit +1];//need for terminal string
 if (kernelBuf == NULL)
  return kernelBuf;

 memset(kernelBuf,0,limit+1);

 //printf("\n Filename u2s:");
 for (i = 0 ; i < limit ;i++)
  {
   machine->ReadMem(virtAddr+i,1,&oneChar);
   kernelBuf[i] = (char)oneChar;
   //printf("%c",kernelBuf[i]);
   if (oneChar == 0)
        break;
  }

  return kernelBuf;
}


/*
Input: - User space address (int)
       - Limit of buffer (int)
       - Buffer (char[])
Output:- Number of bytes copied (int)
Purpose: Copy buffer from System memory space to User  memory space
*/
int   System2User(int virtAddr,int len,char* buffer)
{
```

```
  if (len < 0) return -1;
  if (len == 0)return len;
  int i = 0;
  int oneChar = 0 ;
  do{
    oneChar= (int) buffer[i];
    machine->WriteMem(virtAddr+i,1,oneChar);
    i ++;
  }while(i < len && oneChar != 0);

  return i;
}
```

Lập trình cho system call **void Create (char* name);** nghĩa là chúng ta viết xử ly sự kiện **which = SyscallException** và **type=SC_Create** trong file userprog/exception.cc,
Sau đây là một ví dụ mã nguồn cho system call này, chưa phải là thật hoàn chỉnh, các bạn có thể phải viết để kiểm tra thêm các ngoại lệ

```
/*
Input: reg4 -filename (string)
Output: reg2    -1: error and 0: success
Purpose: process the event SC_Create of System call
*/

case SyscallException:
...
case SC_Create:
{
  int virtAddr;
  char* filename;

  DEBUG(dbgFile,"\n SC_Create call ...");
  DEBUG(dbgFile,"\n Reading virtual address of filename");

  // check for exception
  virtAddr = machine->ReadRegister(4);
  DEBUG (dbgFile,"\n Reading filename.");
  filename = User2System(virtAddr,MaxFileLength+1); // MaxFileLength là = 32
  if (filename == NULL)
    {
      printf("\n Not enough memory in system");
      DEBUG(dbgFile,"\n Not enough memory in system");
      machine->WriteRegister(2,-1); // trả về lỗi cho chương trình người dùng
      delete filename;
      return;
```

```
    }



  DEBUG(dbgFile,"\n Finish reading filename.");
  //DEBUG(dbgFile,"\n File name : '"<<filename<<"'");

// Create file with size = 0
// Dùng đối tượng fileSystem của lớp OpenFile để tạo file, việc tạo file này là sử dụng các
// thủ tục tạo file của hệ điều hành Linux, chúng ta không quản ly trực tiếp các block trên
// đĩa cứng cấp phát cho file, việc quản ly các block của file trên ổ đĩa là một đồ án khác

 if (!fileSystem->Create(filename,0))
    {
      printf("\n Error create file '%s'",filename);
      machine->WriteRegister(2,-1);
      delete filename;
      return;
    }

  machine->WriteRegister(2,0); // trả về cho chương trình người dùng thành công

  delete filename;
  return;
}
```

Sau đó các bạn viết chương trình để thử nghiệm system call này. Ví dụ viết chương trình createfile.c chương trình này lưu trong thư mục Test/createfile.c. Nhớ là phải thêm phần biên dich chương trình createfile này trong test/Makefile và biên dịch lại trước khi chạy thử chương trình createfile của mình bằng HĐH Nachos.