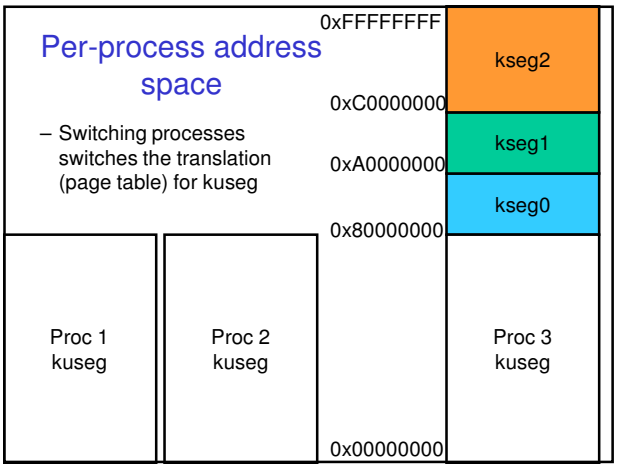
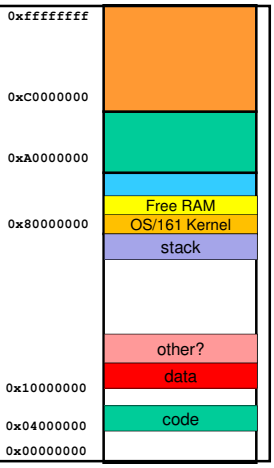
Assignment 2

Per-process address space

– Switching processes switches the translation (page table) for kuseg



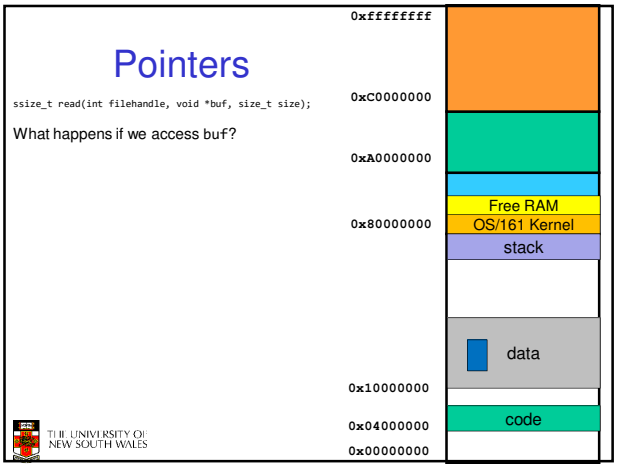
Process Layout



Pointers

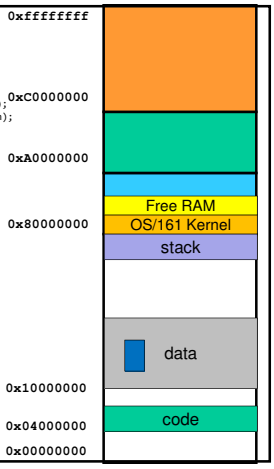
```
ssize_t read(int filehandle, void *buf, size_t size);
```

What happens if we access buf?



Copyin/out/str

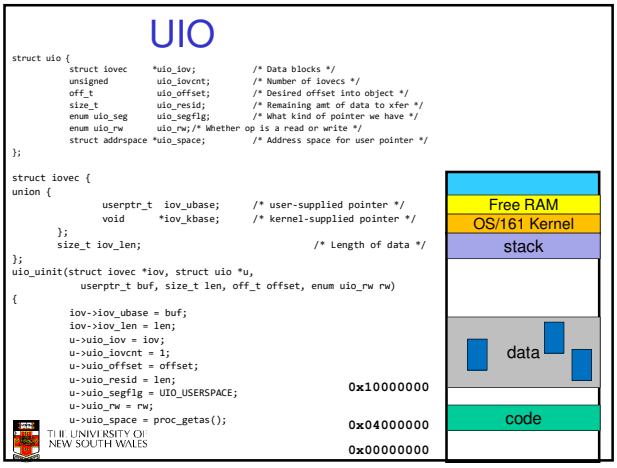
```
int copyin(const_userptr_t usersrc, void *dest, size_t len);
int copyout(const void *src, userptr_t userdest, size_t len);
int copyinstr(const_userptr_t usersrc, char *dest,
              size_t len, size_t *got);
int copyoutstr(const char *src, userptr_t userdest,
               size_t len, size_t *got);
```



UIO

```
struct uio {
          struct iovec     *uio_iov;          /* Data blocks */
          unsigned          uio_iovcnt;       /* Number of iovecs */
          off_t             uio_offset;       /* Desired offset into object */
          size_t            uio_resid;        /* Remaining amt of data to xfer */
          enum uio_seg      uio_segflg;       /* What kind of pointer we have */
          enum uio_rw       uio_rw;/* Whether op is a read or write */
          struct addrspace *uio_space;        /* Address space for user pointer */
};

struct iovec {
union {
                userptr_t  iov_ubase;     /* user-supplied pointer */
                void      *iov_kbase;     /* kernel-supplied pointer */
        };
        size_t iov_len;                         /* Length of data */
};
uio_uinit(struct iovec *iov, struct uio *u,
          userptr_t buf, size_t len, off_t offset, enum uio_rw rw)
{
          iov->iov_ubase = buf;
          iov->iov_len = len;
          u->uio_iov = iov;
          u->uio_iovcnt = 1;
          u->uio_offset = offset;
          u->uio_resid = len;
          u->uio_segflg = UIO_USERSPACE;
          u->uio_rw = rw;
          u->uio_space = proc_getas();
```



System Call Implementation

open()	vfs_ope()
close()	vfs_close()
read()	VOP_READ()
write()	VOP_WRITE()
lseek()	VOP_ISSEEKABLE()
	VOP_STAT()
dup2()	

Lseek Offset

```
off_t lseek(int filehandle, off_t pos, int whence);
```

```
uint64_t offset;
int whence;
off_t retval64;

join32to64(tf->tf_a2, tf->tf_a3, &offset);

copyin((userptr_t)tf->tf_sp + 16, &whence, sizeof(int));

split64to32(retval64, &tf->tf_v0, &tf->tf_v1);
```