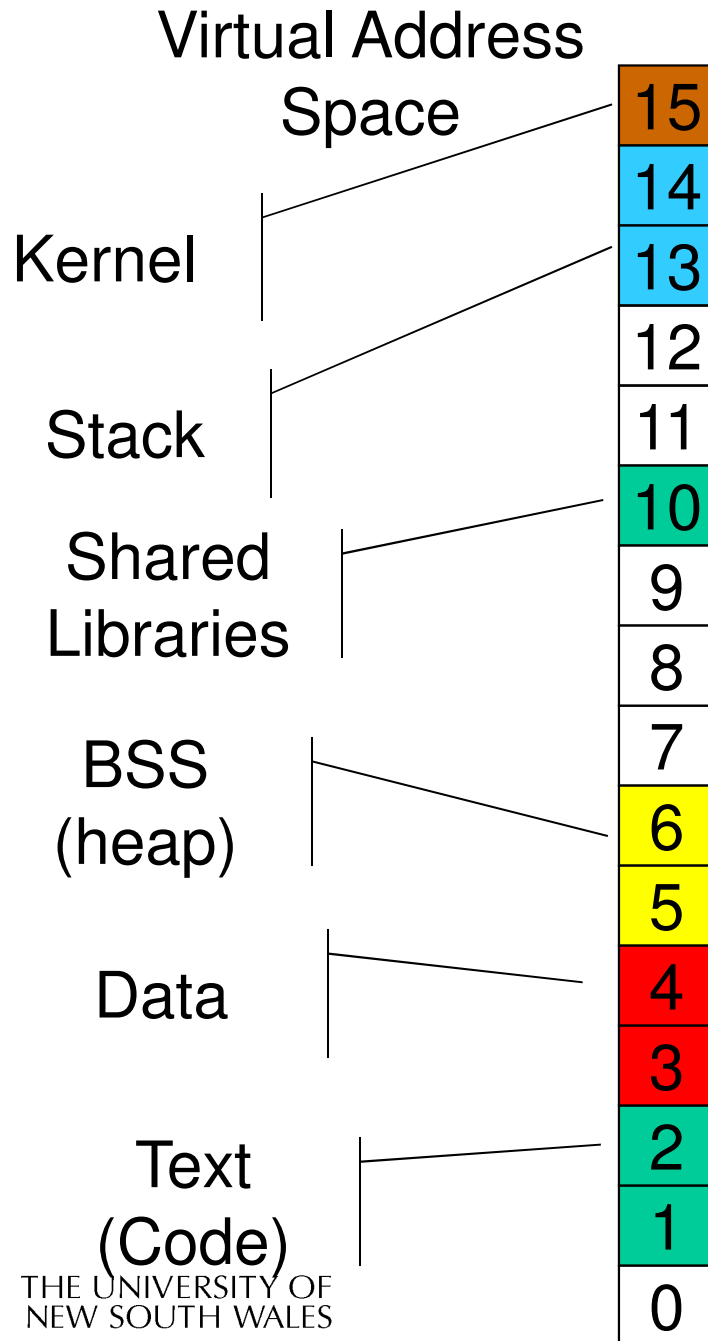


# Assignment 3 tips



# Typical Address Space Layout

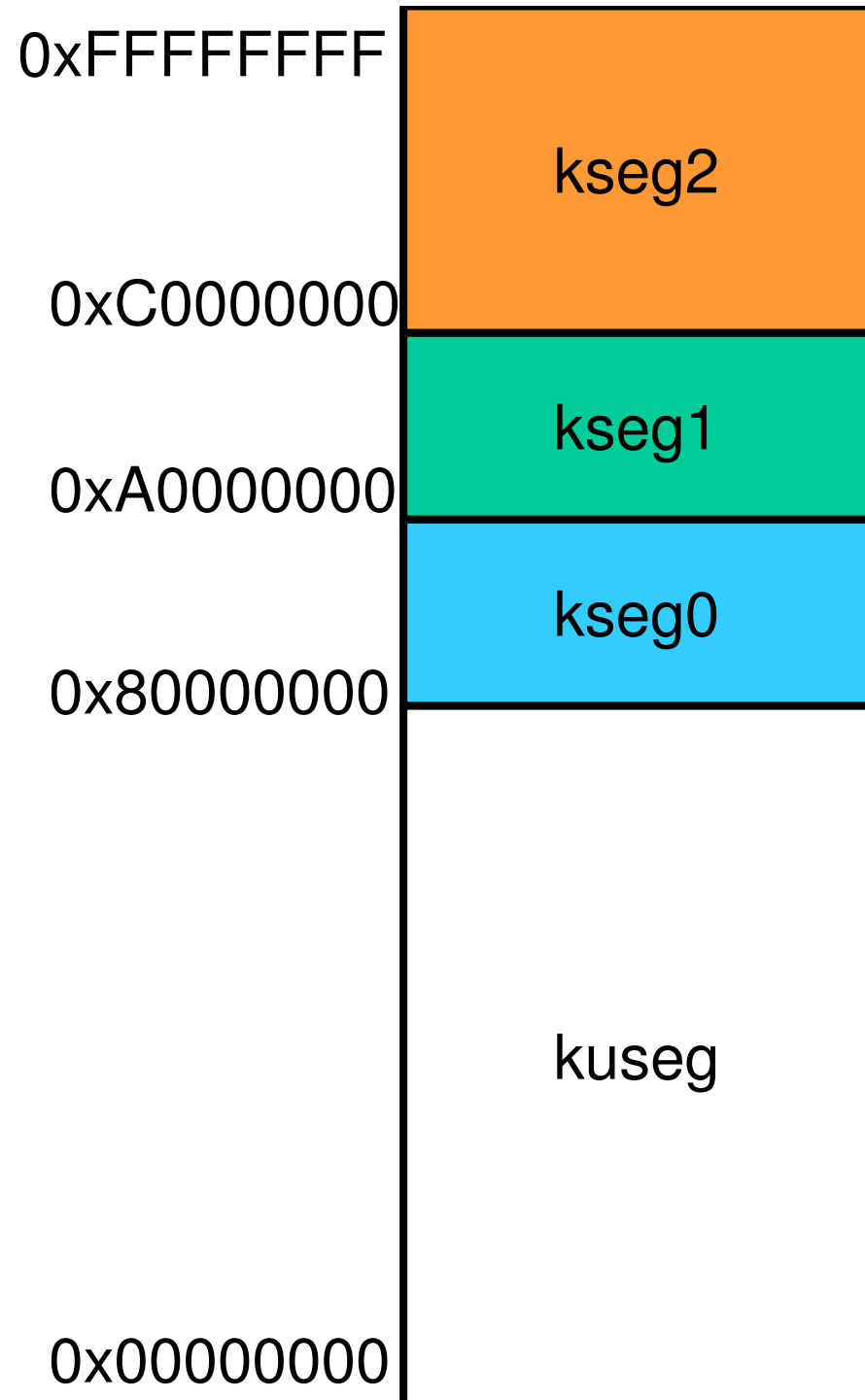


- Stack region is at top, and can grow down
- Heap has free space to grow up
- Text is typically read-only
- Kernel is in a reserved, protected, shared region
- 0-th page typically not used, why?



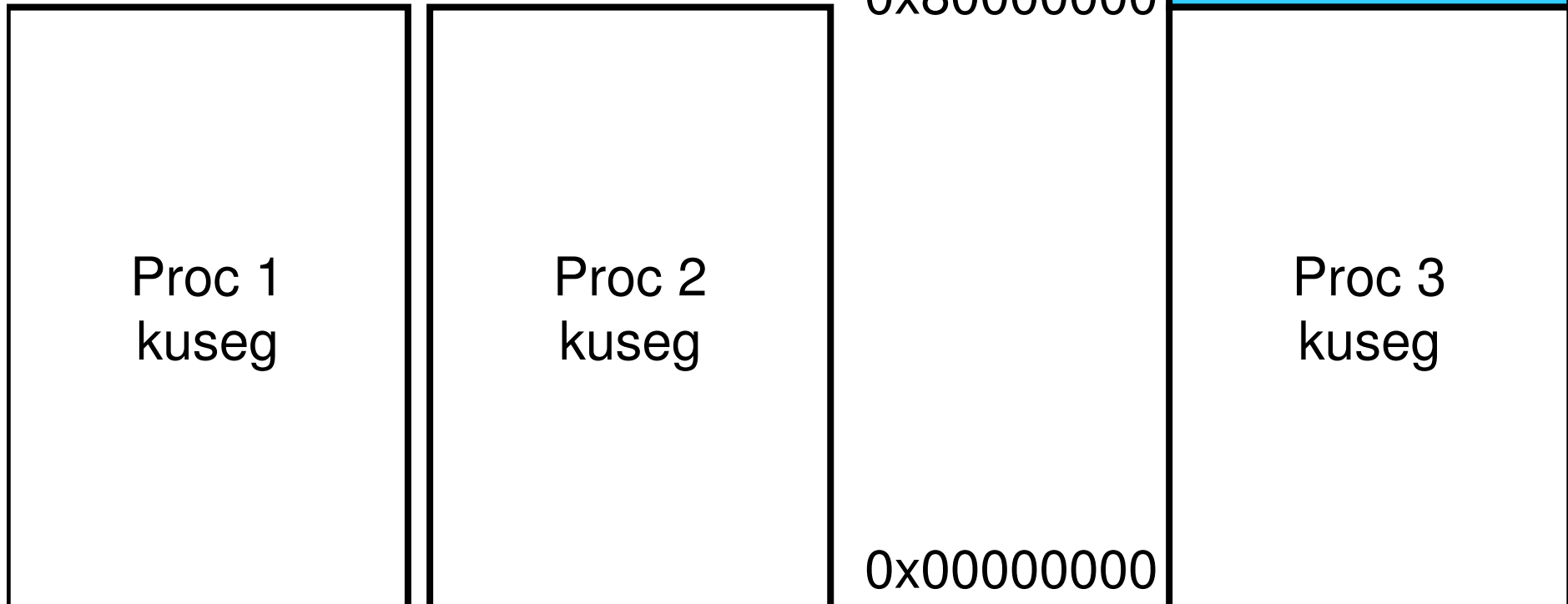
# R3000 Address Space Layout

- kuseg:
  - 2 gigabytes
  - TLB translated (mapped)
  - Cacheable (depending on 'N' bit)
  - user-mode and kernel mode accessible
  - Page size is 4K



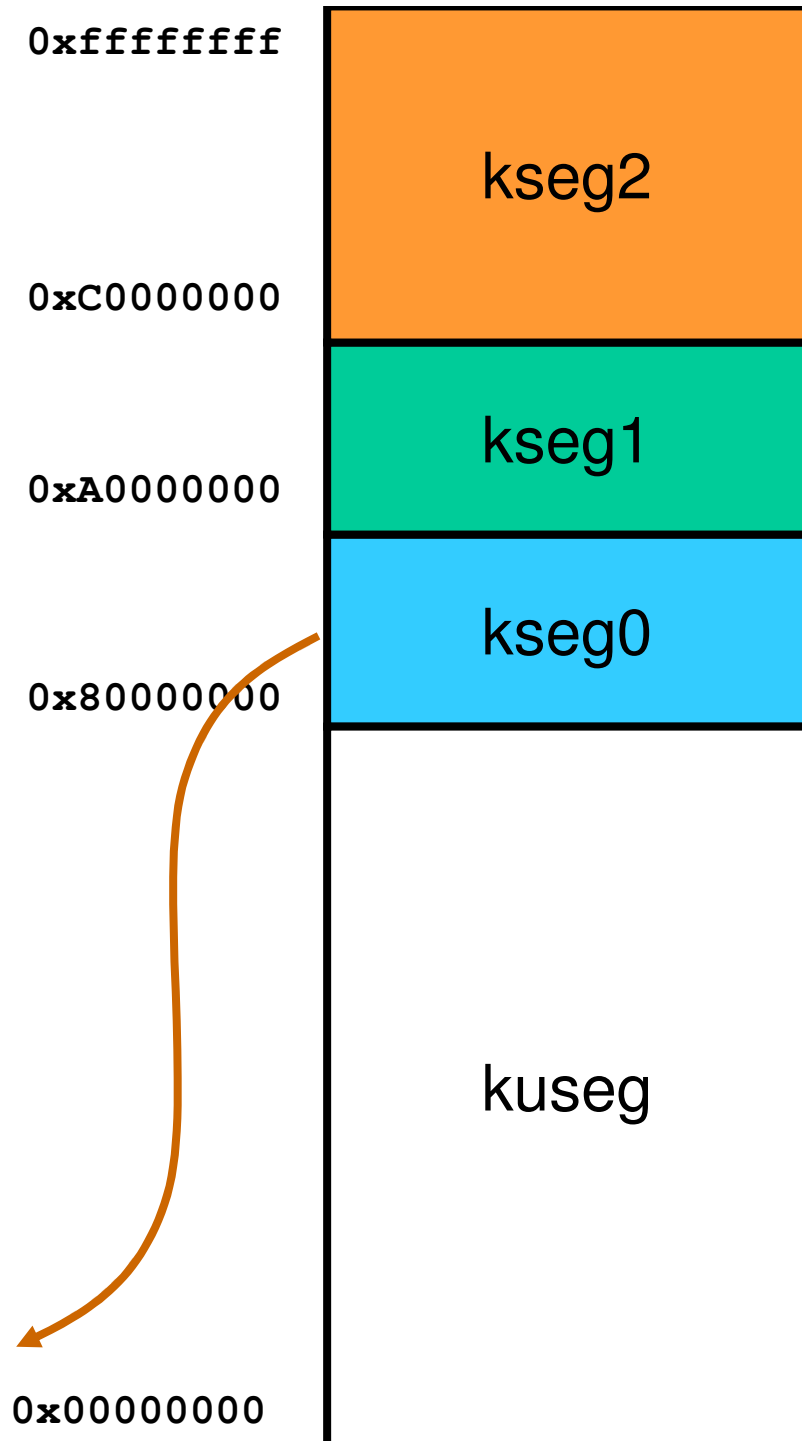
# R3000 Address Space Layout

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



# R3000 Address Space Layout

- kseg0:
  - 512 megabytes
  - Fixed translation window to physical memory
    - $0x80000000 - 0x9fffffff$  virtual =  $0x00000000 - 0x1fffffff$  physical
    - TLB not used
  - Cacheable
  - Only kernel-mode accessible
  - Usually where the kernel code is placed

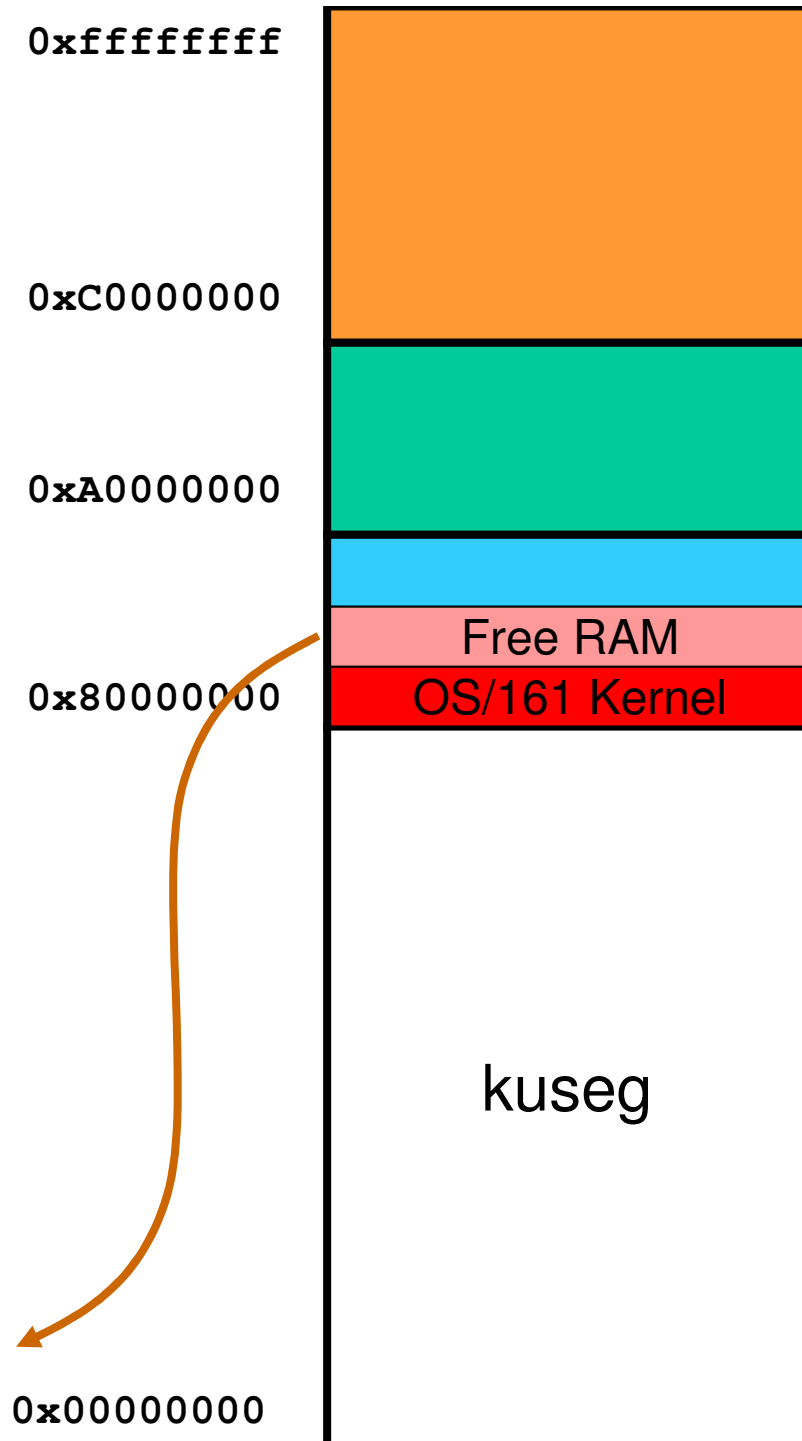
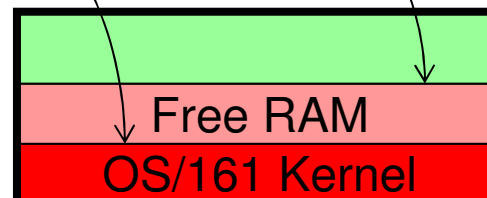


# OS/161 Kernel

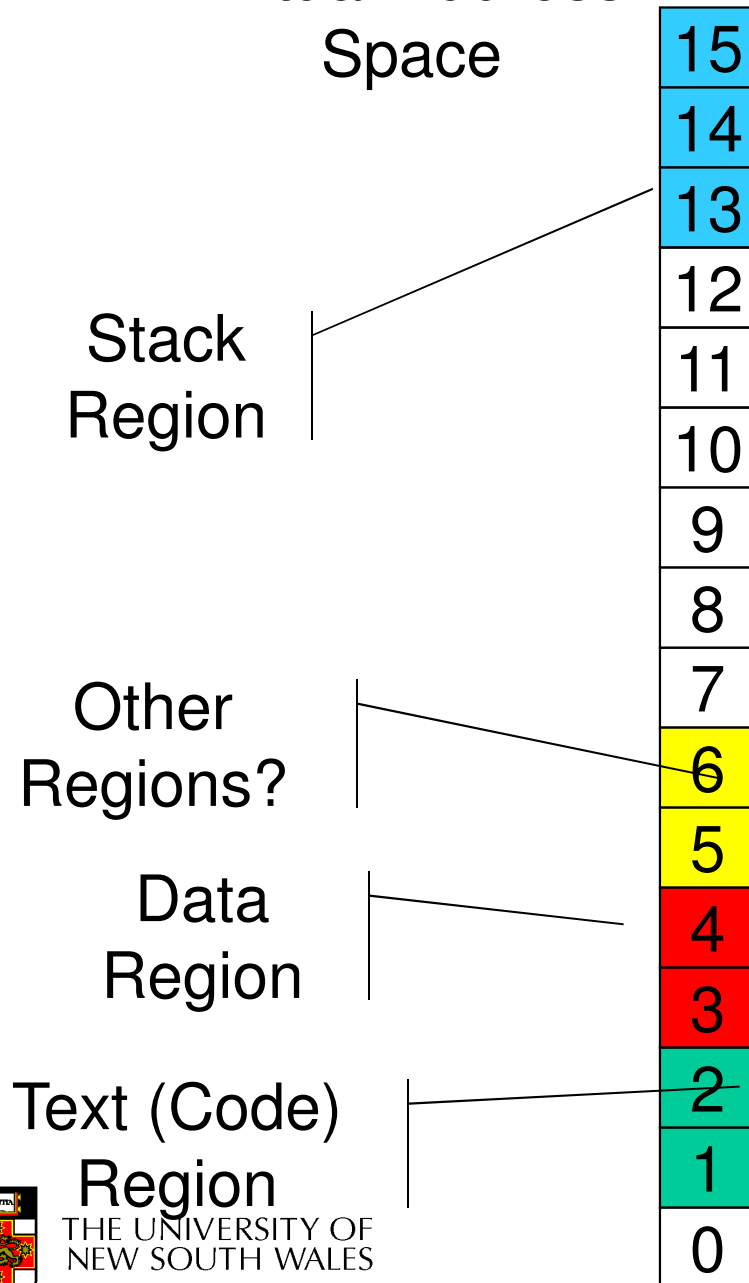
- Placed in Kseg0
  - lower part of physical memory
  - 16 meg of physical RAM
    - 31 busctl ramsize=16777216, in sys161.conf
  - **#define**  
**PADDR\_TO\_KVADDR(paddr)**  
**((paddr)+MIPS\_KSEG0)**
  - **void ram\_getsize**  
**(paddr\_t \*lo, paddr\_t \*hi);**



THE UNIVERSITY OF  
NEW SOUTH WALES



Virtual Address  
Space



# KUseg layout

- Stack region is at top, and can grow down
- Other regions determined by ELF file
  - see `load_elf()`
  - number can vary
  - permissions specified also
  - `cs161-objdump -p testbin/huge`

```
thresher% cs161-objdump -h ../bin/true
```

```
../bin/true: file format elf32-tradbigmips
```

```
Sections:
```

Idx	Name	Size	VMA	LMA	File off	Align	
0	.reginfo	00000018	00400094	00400094	00000094	2**2	CONTENTS, ALLOC, LOAD, READONLY, DATA, LINK_ONCE_SAME_SIZE
1	.text	000001d0	004000b0	004000b0	000000b0	2**4	CONTENTS, ALLOC, LOAD, READONLY, CODE
2	.data	00000000	10000000	10000000	00001000	2**4	CONTENTS, ALLOC, LOAD, DATA
3	.sbss	00000008	10000000	10000000	00001000	2**2	ALLOC
4	.bss	00000000	10000010	10000010	00001008	2**4	ALLOC
5	.comment	00000036	00000000	00000000	00001008	2**0	CONTENTS, READONLY
6	.pdr	000004a0	00000000	00000000	00001040	2**2	CONTENTS, READONLY
7	.mdebug.abi32	00000000	00000000	00000000	000014e0	2**0	CONTENTS, READONLY

```
thresher%
```





```
thresher% cs161-objdump -p ../bin/true
```

```
../bin/true:  file format elf32-tradbigmips
```

```
Program Header:
```

```
0x70000000 off  0x00000094 vaddr 0x00400094 paddr 0x00400094 align 2**2
```

```
    filesz 0x00000018 memsz 0x00000018 flags r--
```

```
LOAD off  0x00000000 vaddr 0x00400000 paddr 0x00400000 align 2**12
```

```
    filesz 0x00000280 memsz 0x00000280 flags r-x
```

```
LOAD off  0x00001000 vaddr 0x10000000 paddr 0x10000000 align 2**12
```

```
    filesz 0x00000000 memsz 0x00000010 flags rw-
```

```
private flags = 1001: [abi=O32] [mips1] [not 32bitmode]
```

```
thresher%
```

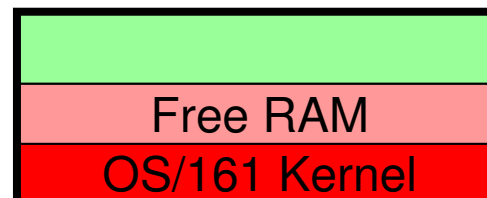


# Process Layout

- Process layout in KUseg
  - regions specified by calls to
    - `as_define_stack()`
    - `as_define_region()`
      - usually implemented as a linked list of region specifications
    - `as_prepare_load()`
      - make READONLY regions READWRITE for loading purposes
    - `as_complete_load()`
      - enforce READONLY again



THE UNIVERSITY OF  
NEW SOUTH WALES



0xffffffff

0xc0000000

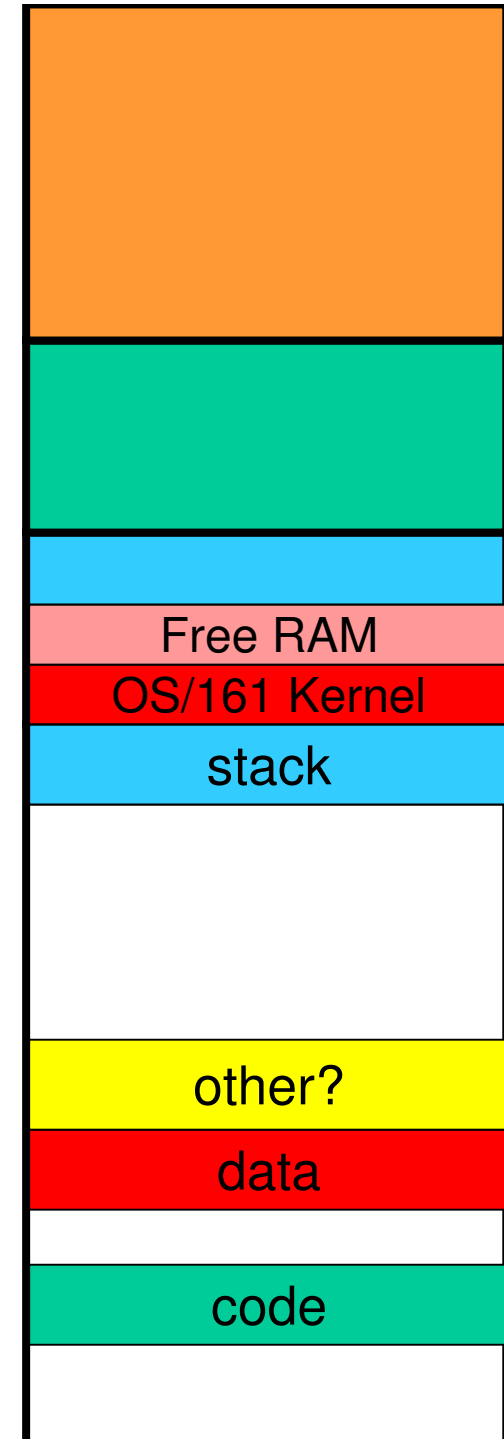
0xa0000000

0x80000000

0x10000000

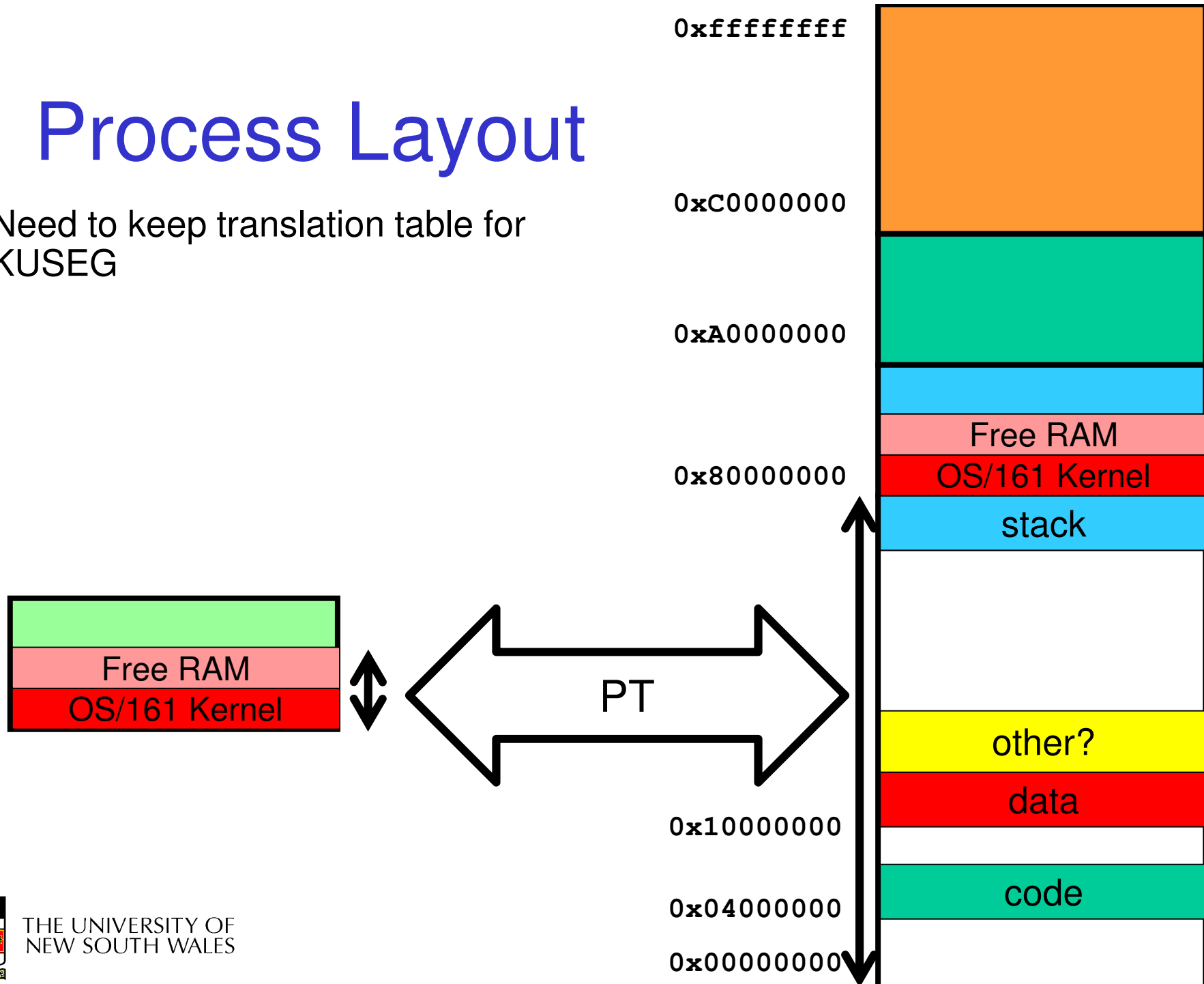
0x04000000

0x00000000



# Process Layout

- Need to keep translation table for KUSEG



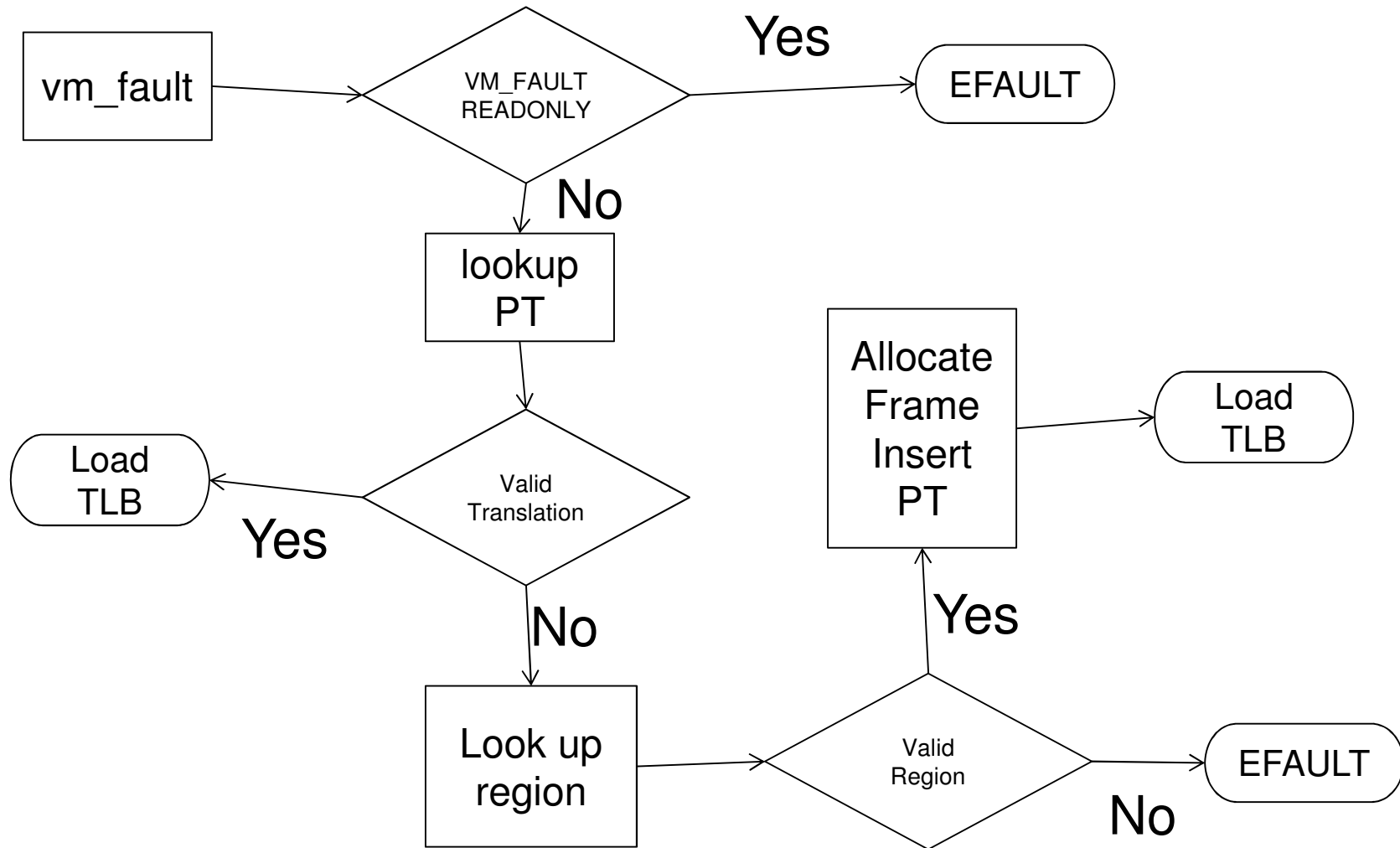
- `as_create()`
  - allocate a data structure used to keep track of an address space
    - i.e. regions
    - PT Level 1
    - `curthread->t_vmspace` used to get access to current address space struct
    - `pid_t` some type defined by you
- `as_destroy()`
  - deallocate book keeping and page tables.
    - deallocating frames used



- `as_copy()`
  - allocates a new (destination) address space
  - adds all the same regions as source
  - roughly, for each mapped page in source
    - allocate a frame in dest
    - copy contents from source frame to dest frame
    - add PT entry for dest
- `as_activate()`
  - flush TLB



# VM Fault Approximate Flow Chart



kprintf()

- Do not use it in `vm_fault()` after the TLB write!!!!



# trace161 can help with debugging

<http://cgi.cse.unsw.edu.au/~cs3231/06s1/os161/man/sys161/index.html>

- The following additional options control trace161's tracing and are ignored by sys161:
- *-f tracefile*
  - Set the file trace information is logged to. By default, stderr is used. Specifying *-f* sends output to stdout instead of stderr.
- *-t traceflags*
  - Tell System/161 what to trace. The following flags are available:
    - d Trace disk I/O
    - e Trace emufs I/O
    - j Trace jumps and branches
    - k Trace instructions in kernel mode
    - n Trace network I/O
    - t Trace TLB/MMU activity
    - u Trace instructions in user mode
    - x Trace exceptions
- **Caution:** tracing instructions generates huge amounts of output that may overwhelm smaller host systems.





wagner% trace161 -tt kernel

sys161: System/161 release 1.12, compiled Jun 21 2005 10:34:06

sys161: Tracing enabled: tlb

trace: tlb: 81000/000 -> 00000 ----: [0]  
trace: tlb: 81001/000 -> 00000 ----: [1]  
trace: tlb: 81002/000 -> 00000 ----: [2]  
trace: tlb: 81003/000 -> 00000 ----: [3]  
trace: tlb: 81004/000 -> 00000 ----: [4]  
trace: tlb: 81005/000 -> 00000 ----: [5]  
trace: tlb: 81006/000 -> 00000 ----: [6]  
trace: tlb: 81007/000 -> 00000 ----: [7]  
trace: tlb: 81008/000 -> 00000 ----: [8]  
trace: tlb: 81009/000 -> 00000 ----: [9]  
trace: tlb: 8100a/000 -> 00000 ----: [10]  
trace: tlb: 8100b/000 -> 00000 ----: [11]  
trace: tlb: 8100c/000 -> 00000 ----: [12]  
trace: tlb: 8100d/000 -> 00000 ----: [13]  
trace: tlb: 8100e/000 -> 00000 ----: [14]  
trace: tlb: 8100f/000 -> 00000 ----: [15]  
trace: tlb: 81010/000 -> 00000 ----: [16]  
trace: tlb: 81011/000 -> 00000 ----: [17]  
trace: tlb: 81012/000 -> 00000 ----: [18]  
trace: tlb: 81013/000 -> 00000 ----: [19]  
trace: tlb: 81014/000 -> 00000 ----: [20]



```
trace: tlbp:      81015/000 -> 00000 ----: [21]
.....
trace: tlbp:      8103c/000 -> 00000 ----: [60]
trace: tlbp:      8103d/000 -> 00000 ----: [61]
trace: tlbp:      8103e/000 -> 00000 ----: [62]
trace: tlbp:      8103f/000 -> 00000 ----: [63]
trace: tlbp:      81040/000 -> NOT FOUND
trace: tlbwi: [ 0] 81000/000 -> 00000 ---- ==> 81040/000 -> 00000 ----
trace: tlbp:      81041/000 -> NOT FOUND
trace: tlbwi: [ 1] 81001/000 -> 00000 ---- ==> 81041/000 -> 00000 ----
trace: tlbp:      81042/000 -> NOT FOUND
trace: tlbwi: [ 2] 81002/000 -> 00000 ---- ==> 81042/000 -> 00000 ----
trace: tlbp:      81043/000 -> NOT FOUND
trace: tlbwi: [ 3] 81003/000 -> 00000 ---- ==> 81043/000 -> 00000 ----
trace: tlbp:      81044/000 -> NOT FOUND
trace: tlbwi: [ 4] 81004/000 -> 00000 ---- ==> 81044/000 -> 00000 ----
trace: tlbp:      81045/000 -> NOT FOUND
trace: tlbwi: [ 5] 81005/000 -> 00000 ---- ==> 81045/000 -> 00000 ----
trace: tlbp:      81046/000 -> NOT FOUND
trace: tlbwi: [ 6] 81006/000 -> 00000 ---- ==> 81046/000 -> 00000 ----
trace: tlbp:      81047/000 -> NOT FOUND
trace: tlbwi: [ 7] 81007/000 -> 00000 ---- ==> 81047/000 -> 00000 ----
trace: tlbp:      81048/000 -> NOT FOUND
```



trace: tlbp: 81015/000 -> 00000 ----: [21]

.....

trace: tlbp: 8107c/000 -> NOT FOUND

trace: tlbwi: [60] 8103c/000 -> 00000 ---- ==> 8107c/000 -> 00000 ----

trace: tlbp: 8107d/000 -> NOT FOUND

trace: tlbwi: [61] 8103d/000 -> 00000 ---- ==> 8107d/000 -> 00000 ----

trace: tlbp: 8107e/000 -> NOT FOUND

trace: tlbwi: [62] 8103e/000 -> 00000 ---- ==> 8107e/000 -> 00000 ----

trace: tlbp: 8107f/000 -> NOT FOUND

trace: tlbwi: [63] 8103f/000 -> 00000 ---- ==> 8107f/000 -> 00000 ----

OS/161 base system version 1.10

Copyright (c) 2000, 2001, 2002, 2003

President and Fellows of Harvard College. All rights reserved.

Put-your-group-name-here's system version 0 (ASST1 #1)

Cpu is MIPS r2000/r3000

344k physical memory available

Device probe...

lamebus0 (system main bus)

emu0 at lamebus0

