Operating Systems

Lab. Class

Week 6

Project Plan

- 6 projects
  0. Install xv6
  1. System call
  2. Scheduling
  3. Virtual memory 1
  4. Virtual memory 2
  5. Concurrency
  6. File system

- Individual projects

Address Translation in Intel x86

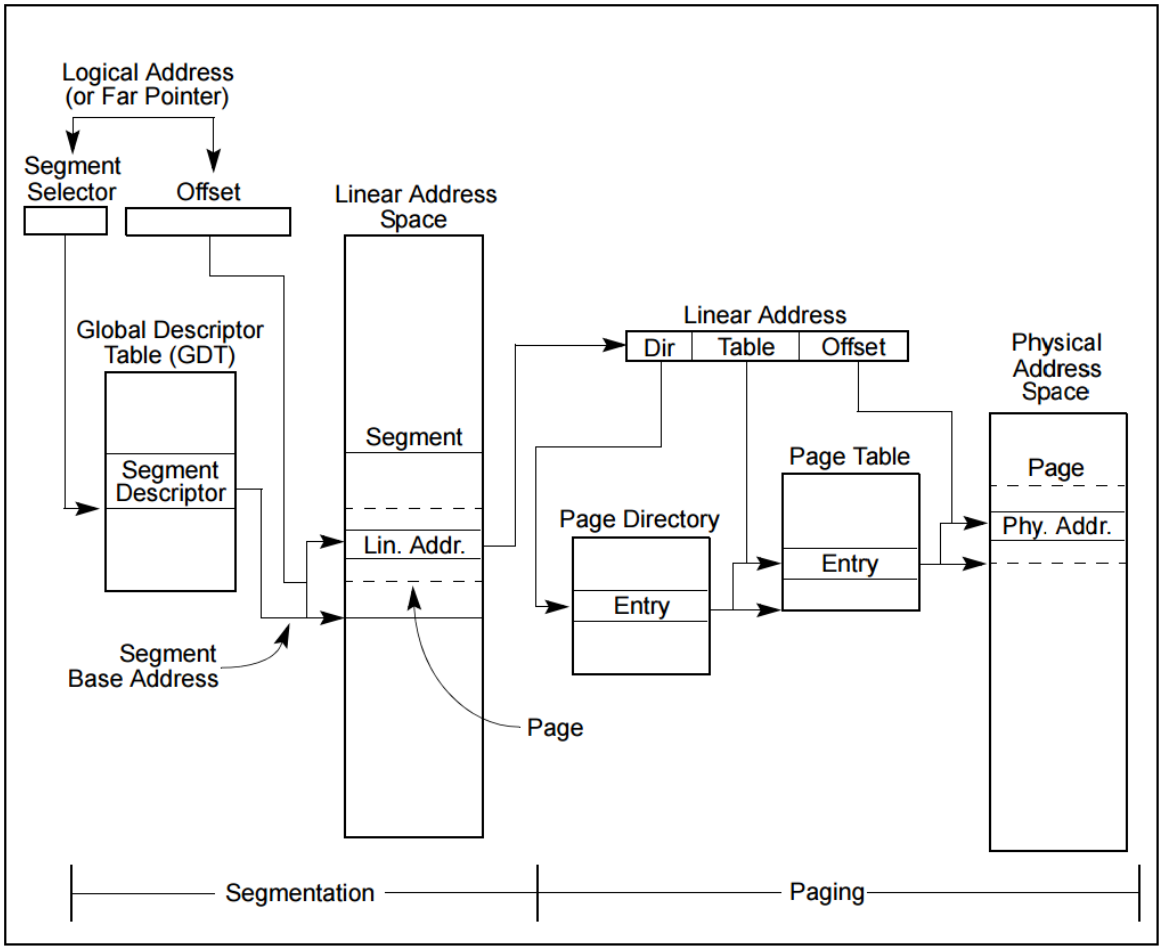
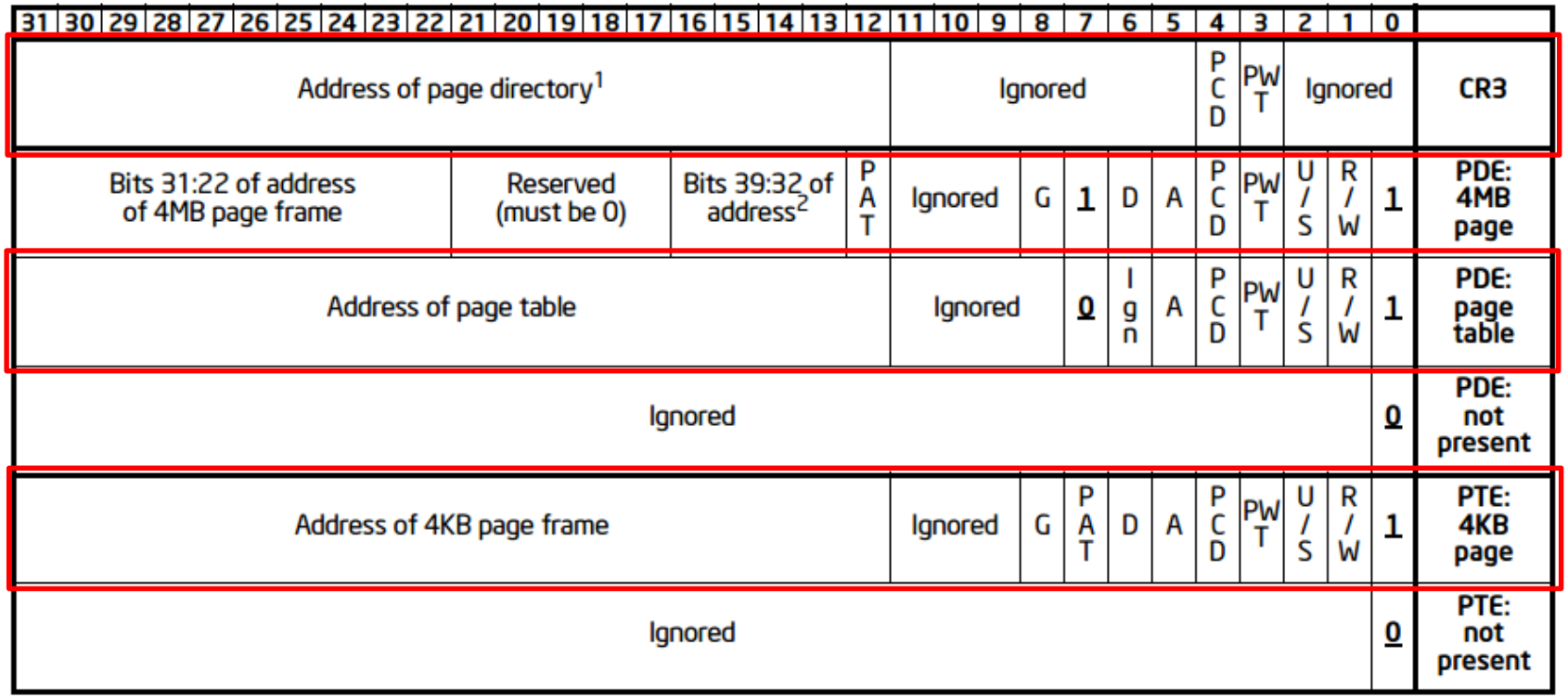


Figure 3-1. Segmentation and Paging

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

Formats of Paging Entries in Intel x86



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Address of page directory[1]																				Ignored							PCD	PWT	Ignored			CR3
Bits 31:22 of address of 4MB page frame										Reserved (must be 0)						Bits 39:32 of address[2]			PAT	Ignored			G	1	D	A	PCD	PWT	U/S	R/W	1	PDE: 4MB page
Address of page table																				Ignored				0	Ign	A	PCD	PWT	U/S	R/W	1	PDE: page table
Ignored																															0	PDE: not present
Address of 4KB page frame																				Ignored			G	PAT	D	A	PCD	PWT	U/S	R/W	1	PTE: 4KB page
Ignored																															0	PTE: not present

Figure 4-4. Formats of CR3 and Paging-Structure Entries with 32-Bit Paging

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

Page Fault Exception in Intel x86

- Conditions
  1. There is no translation for the linear address
  2. There is a translation for the linear address, but its access rights do not permit the access

- CR2 stores the linear address that caused a page fault

- Processor triggers interrupt 14 (page fault)

Control Registers in Intel x86

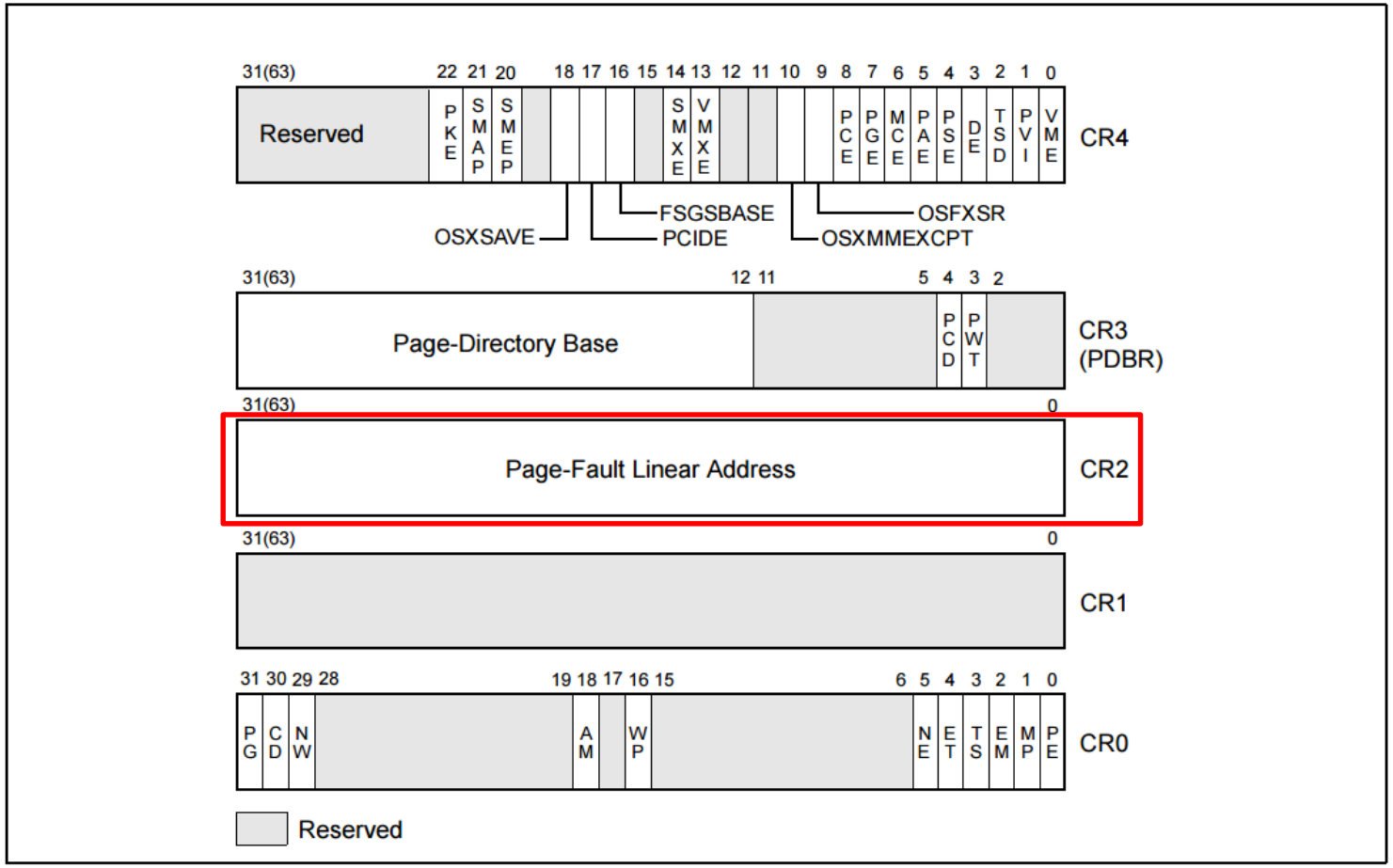


Figure 2-7. Control Registers

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

Page Fault Error Code in Intel x86

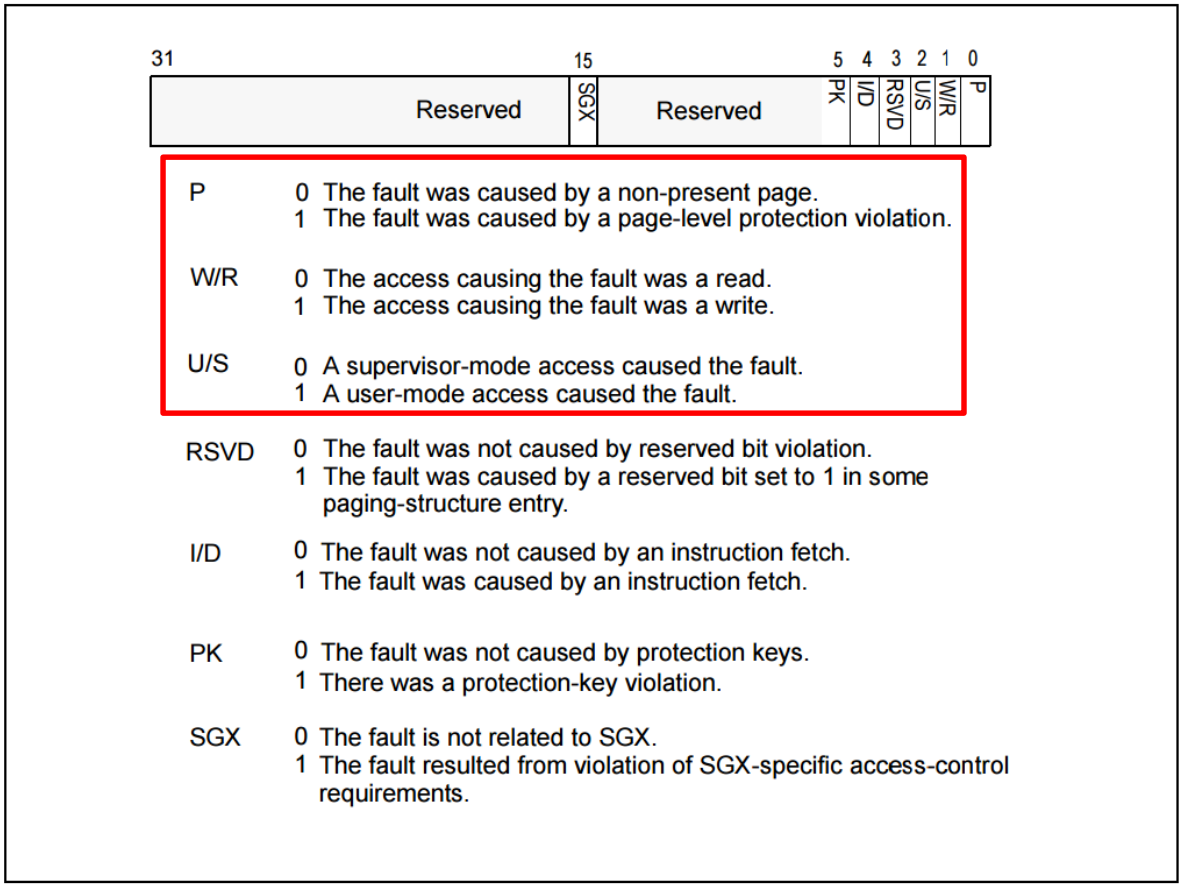


Figure 4-12. Page-Fault Error Code

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

Exception & Interrupt Handling in xv6

- Follows Intel x86 architecture

- Procedures
  1. Assign certain interrupt to interrupt descriptor table (IDT)
  2. All interrupts jump to alltraps() and build trap frame
  3. Handle each interrupt depending on its trap number

Process User Stack in xv6

- 1 stack page & 1 guard page

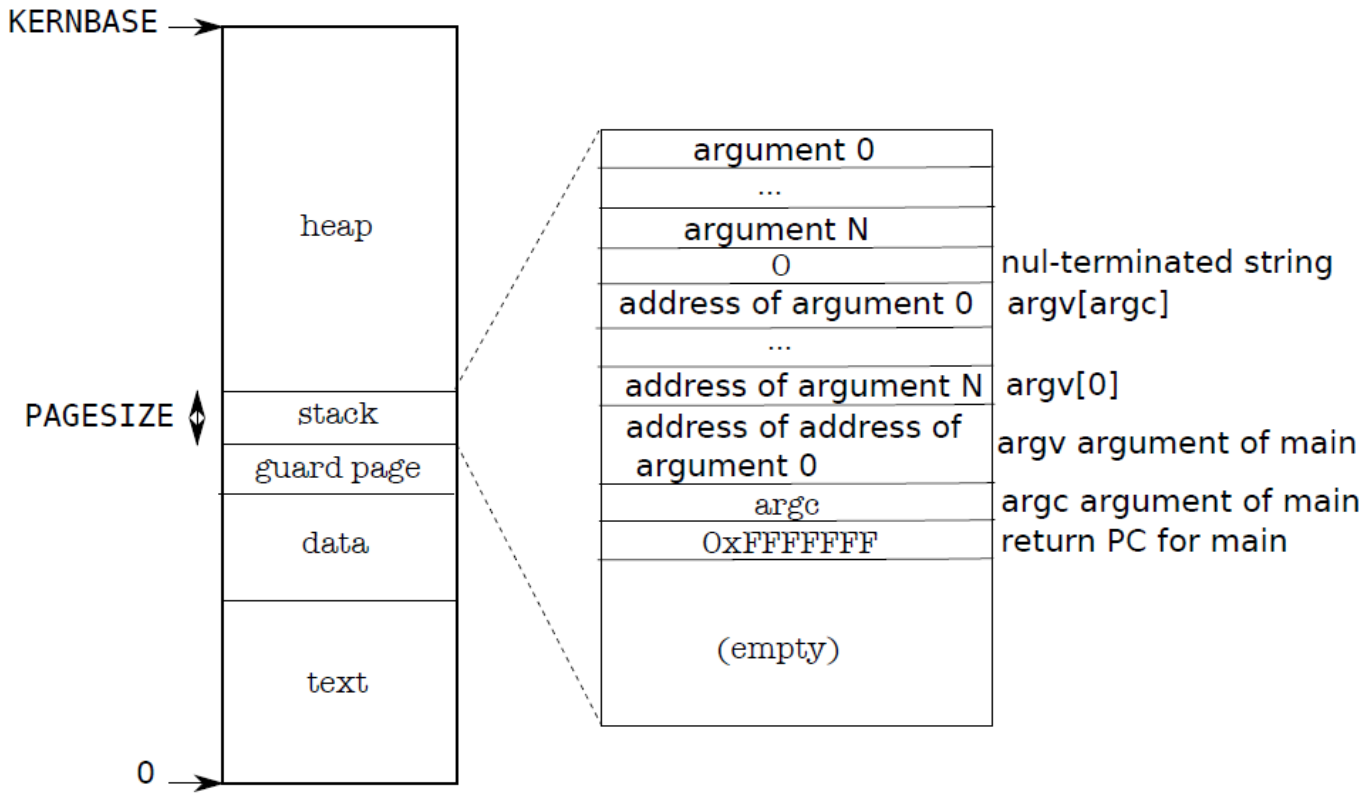
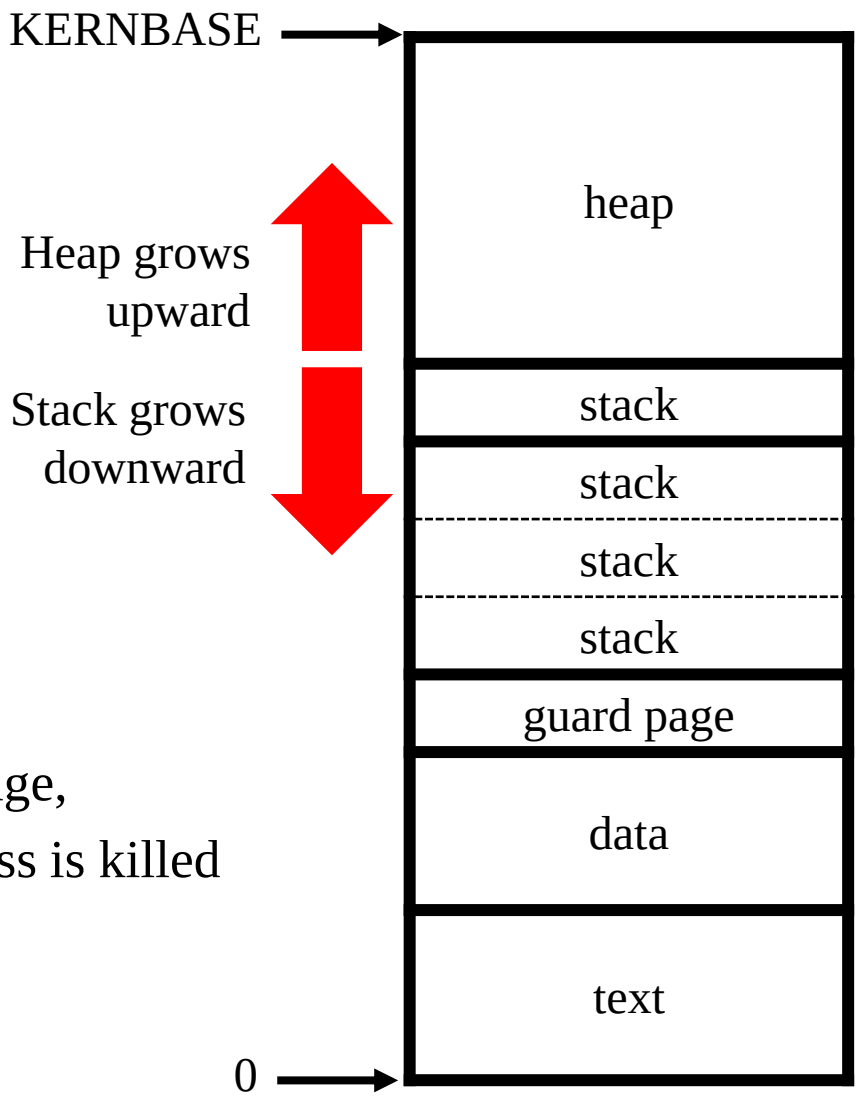


Figure 2-3. Memory layout of a user process with its initial stack.

Stack Growth in xv6

- 4 reserved pages for stack
- Stack grows when current stack is full
- Stack pointer can move up to 32bytes (pushal instruction)
  - Otherwise, invalid access
- When stack pointer reaches guard page, stack overflow occurs and the process is killed



Project Assignment #3 - Stack Growth

- Implement stack growth on xv6

- Submission deadline
  - 2016-04-24 23:59

Project Assignment #3 Template Code

- Download from http://sys.skku.edu
- Modifications
  - Remove debug messages
    - Do not print any messages on screen
  - halt system call
    - Halt xv6 program
  - make tarball
    - Compress your source codes into one .tar.gz file for submission
    - You should enter your ID & project no. on Makefile

Project Submission Procedure

- http://sys.skku.edu
- Register an account
  - You must type your real name & student ID
  - Other parts are free

- Since 2nd submission, -5% penalty of the project score
- Every one day delay, -25% penalty of the project score
  - You can use up to 5 *slip* days

Project Assignment #3 Test Cases

- Test cases will be uploaded to http://csl.skku.edu/SWE3004S16/Projects