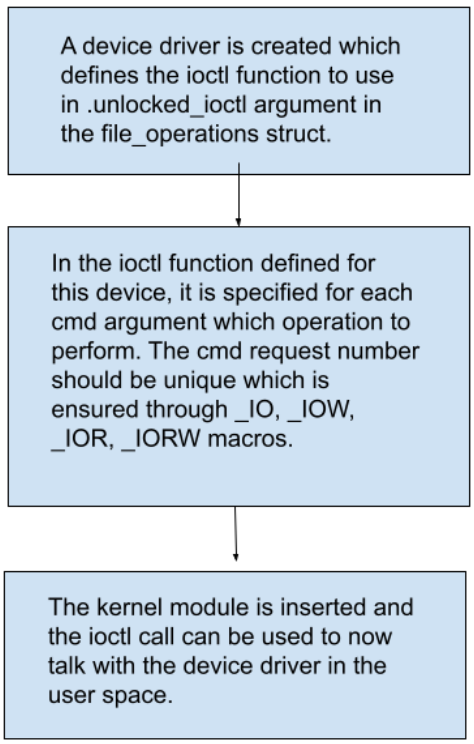
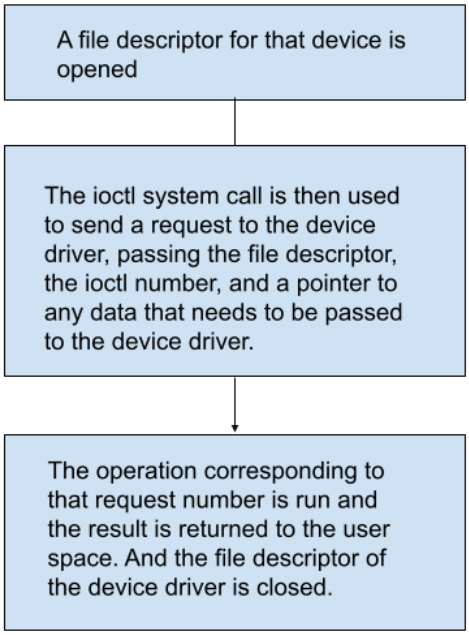
CS 695-A3 Answers - 190050013

1. A. The system call `static long ioctl(struct file *f, unsigned int cmd, unsigned long arg)` call takes three arguments
    a. `struct file* f` - which is a pointer to a file descriptor of an opened driver, the command is run on this driver
    b. `unsigned int cmd` - a command (request number) to run in the driver. The implementation of ioctl has a switch over this cmd and does a different operation on different commands. The request number needs to be unique across the system, which is achieved through the usage of _IO, _IOW, _IOR, and _IORW macros that generate unique ioctl modifiers.
    c. `unsigned long arg` - Any argument that needs to be sent to the ioctl function is packed in a struct, and the address of a pointer to that struct is sent over through this argument. It is an unsigned long that is type-casted to the corresponding struct's pointer and can be used for the get or set operations.

    B. Kernel side



User side



2. A. Some of the KVM API calls used by the hypervisor to set up and create VM:
    a. KVM_CREATE_VM: creates a new KVM virtual machine.
    b. KVM_GET_SUPPORTED_CPUID: retrieves the supported features of the virtual machine.
    c. KVM_VM_SETUP_MEMORY_REGION: sets up a memory region for the virtual machine.
    d. KVM_CREATE_VCPU: creates a virtual CPU for the virtual machine.
    e. KVM_RUN: runs the virtual machine.
    f. KVM_SET_USER_MEMORY_REGION: sets up a user-space memory region for the virtual machine.
    g. KVM_SET_TSS_ADDRESS: sets the address of the task-state segment for the virtual machine.
    h. KVM_SET_SREGS - Sets the system register state for the virtual CPU.
    i. KVM_SET_CR0 - Sets the control register 0 for the virtual CPU.
    j. KVM_SET_CR3 - Sets the control register 3 for the virtual CPU.
    k. KVM_SET_CR4 - Sets the control register 4 for the virtual CPU.

    B.
    a. KVM_CREATE_VM - Creates a KVM virtual machine.
    b. KVM_SET_TSS_ADDRESS: sets the address of the task-state segment for the virtual machine.
    c. KVM_GET_API_VERSION - get the API version of KVM

d. KVM_RUN - Starts the execution of the virtual machine.
e. KVM_GET_SREGS - Retrieves the system register state of the virtual machine.
f. KVM_SET_SREGS - Sets the system register state of the virtual machine.
g. KVM_GET_REGS - Retrieves the register state of the virtual machine.
h. KVM_SET_REGS - Sets the register state of the virtual machine.
i. KVM_SET_USER_MEM_REGION - Sets the memory region information for a virtual machine.
j. KVM_CREATE_VCPU - Creates a virtual CPU for a virtual machine.
k. KVM_GET_VCPU_MMAP_SIZE - get the size of memory that needs to be mmapped to communicate with the virtual CPU.
l.

3. a. The mapping of Guest Virtual Address (GVA) to Host Physical Address (HPA) varies across different modes of operation.
In real mode, GVA is directly mapped to HPA using linear address calculation.
However, in protected mode and paged 32-bit mode, GVA is mapped to HPA using segmentation and paging mechanisms. The segment selector and offset of the GVA are used to access the descriptor table and calculate the linear address, which is then mapped to a physical address using paging.
The address space in paged 32-bit mode is limited to 4 GB, while in protected mode, it is limited to 1 MB.
In long mode, which has a much larger address space, the PML4 table is used in addition to the segmentation and paging mechanisms to map GVA to HPA. The PML4 table maps 48-bit virtual addresses to 52-bit physical addresses (if PAE is set).
b. The size allocated in vm_init is 2MB (0x200000) .
c. *static void setup_long_mode(struct vm *vm, struct kvm_sregs *sregs)* function sets up the page table and control registers values of the vm running in long mode. It sets up a 3 level page table at the bottom of the guest physical address space. Then it sets the control register values in sregs to point to that page table.
sregs were set while setting up the page table in setup_long_mode function which also had set the sregs.cr3 register which points to the pml4 of the vm. Now in run_long_mode function KVM_SET_SREGS api call is used to make KVM aware of the set sregs.

d. In CR4 register only the "Physical Address Extension (PAE)" bit is set. If set, changes page table layout to translate 48-bit virtual addresses into extended 52-bit physical addresses.

4. a. The guest starts its execution at guest virtual address 0 set using regs.rip = 0;in run_long_mode function. This address is configured in setup_64bit_code_segment function and assigned to cs register.
b. In run_vm(struct vm *vm, struct vcpu *vcpu, size_t sz) function vm is run using ioctl(vcpu->fd, KVM_RUN, 0) API call which happens inside a for loop.

5. a. In guest. C prints each character on the port `0xE9` which causes exit to the hypervisor. The serial port is emulated using a memory-mapped I/O mechanism

b. In hypervisor, the value is read using using ((char *) vcpu->kvm_run) + vcpu->kvm_run->io.data_offset. The KVM_EXIT_IO should be the exit reason as it is an IO call.
c. The number 42 is written to 0x400 = 1024 Bytes in guest's virtual memory by guest.c. The value 42 is used for testing purposes, it is written to and read from the emulated serial port to verify the correctness of the program. Any value could have been used in place of 42.