

Assignment 3

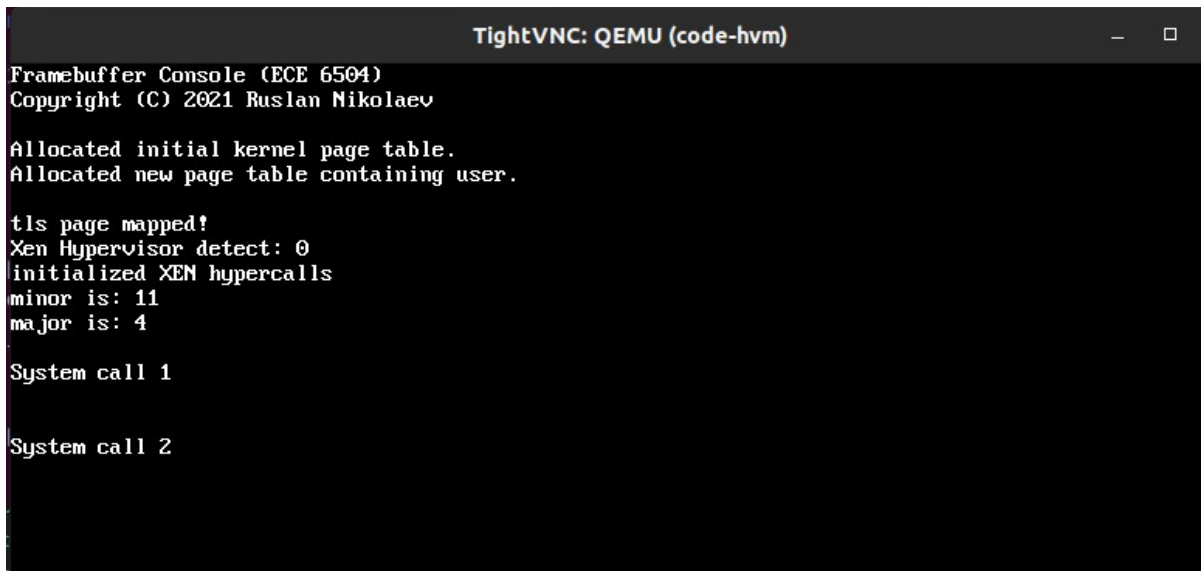
Part-1

2.1 Xen Initialization

Integrated the code to detect xen hypervisor and mapped the shared_info struct in kernel.c.

Also initialized hypercalls in the function initialize_hypercalls().

The xen version major and minor numbers are being printed in the print_xen_version() function in kernel.c.



```
TightVNC: QEMU (code-hvm)
Framebuffer Console (ECE 6504)
Copyright (C) 2021 Ruslan Nikolaev

Allocated initial kernel page table.
Allocated new page table containing user.

tls page mapped?
Xen Hypervisor detect: 0
initialized XEN hypercalls
minor is: 11
major is: 4

System call 1

System call 2
```

2.2 PV Clock

Implemented PV clock in the kernel inside the function pvclock_init().

It sets up the monotonic clock and the wall clock functionalities by calling the required functions and initializing the required data structures.

First, the initial wall clock and monotonic clock values are printed. The actual wall clock value is the monotonic clock value plus the rtc_epochoffset which is the base system value.

Then we implement a busy-wait loop for 1 second using the monotonic time function.

After that we print the new values of the wall clock and the monotonic clock.

```
TightVNC: QEMU (code-hvm)
Framebuffer Console (ECE 6504)
Copyright (C) 2021 Ruslan Nikolaev

Allocated initial kernel page table.
Allocated new page table containing user.

tls page mapped?
Xen Hypervisor detect: 0
initialized XEN hypercalls
Xen version major is: 4
Xen version minor is: 11

initial monotonic clock:13836832254497
initial wall clock:1616881201722953329

clock values after 1 second:
monotonic clock:13837832254528
wall clock:1616881202722953360

System call 1

System call 2
```

Part-2

3.1 Shared Memory

I've created a separate guest domain and a separate configuration file for it called code-other.cfg which refers to the new guest domain.

The main domain initializes the gnttab_table variable by pointing to a buffer and then calls ginit_gnttab().

We need to pass two additional pages from boot.c for this task. One for the gnttab_table variable and another to act as the shared page which we initialize with a null terminated message.

The domain IDs are hard coded based and we need to specify the domain ID of the other side while calling the grant_access function. We pass the frame of the page and not the page address.

The guest domain similarly maps the domain of the other side and the host address and passes it to the HYPERVISOR_grant_table_op function. We also need to pass a grant reference, which is the same one

that we receive in the original domain (511 – as can be seen printed in the first screenshot below). If it is successful, it returns a message and we then print the null-terminated shared message on the screen.



```
TIGHTVNC: QEMU (code-hvm)
Framebuffer Console (ECE 6504)
Copyright (C) 2021 Ruslan Nikolaev

Allocated initial kernel page table.
Allocated new page table containing user.

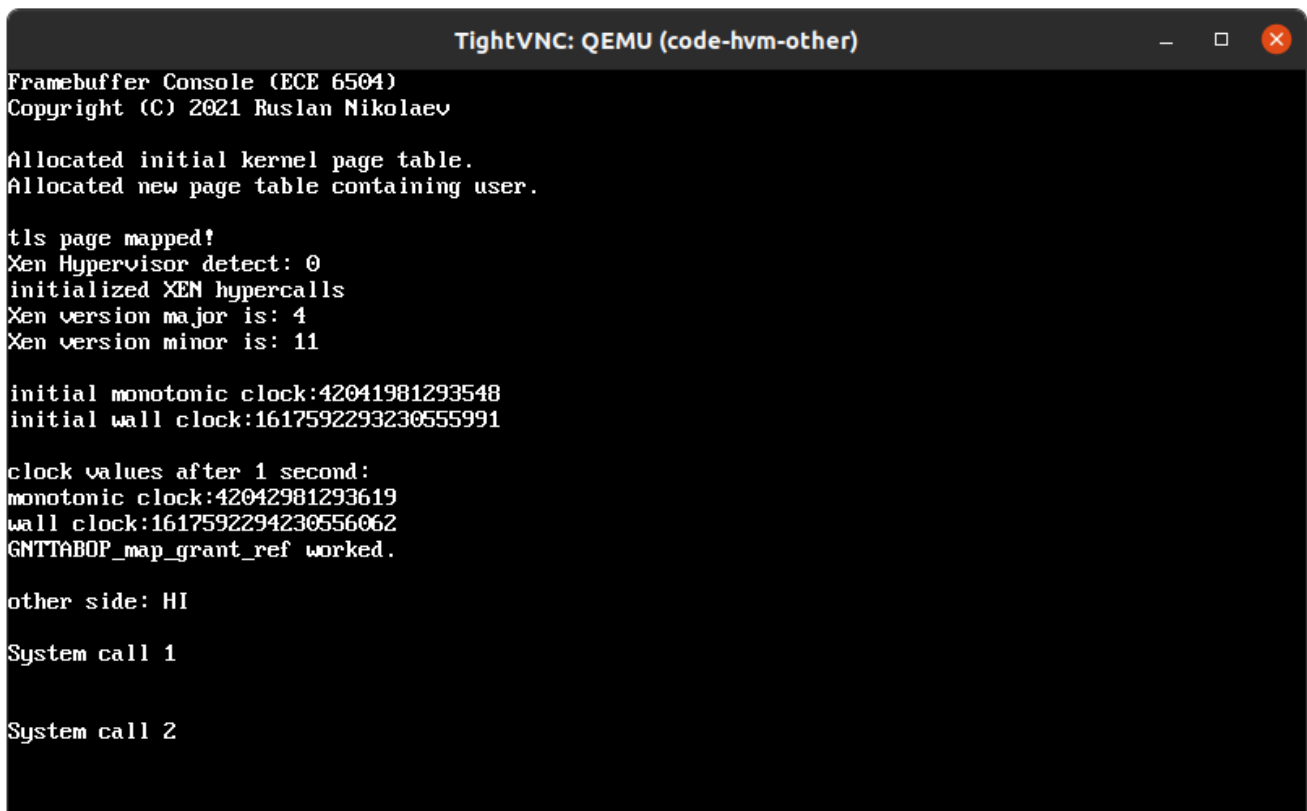
tls page mapped!
Xen Hypervisor detect: 0
initialized XEN hypercalls
Xen version major is: 4
Xen version minor is: 11
gnttab_table mapped at 0x3d18d000.

Writing a message: HI
Grant ref id: 511

System call 1

System call 2
```

The original domain with the statement – “Writing a message: HI”



```
TIGHTVNC: QEMU (code-hvm-other)
Framebuffer Console (ECE 6504)
Copyright (C) 2021 Ruslan Nikolaev

Allocated initial kernel page table.
Allocated new page table containing user.

tls page mapped!
Xen Hypervisor detect: 0
initialized XEN hypercalls
Xen version major is: 4
Xen version minor is: 11

initial monotonic clock:42041981293548
initial wall clock:1617592293230555991

clock values after 1 second:
monotonic clock:42042981293619
wall clock:1617592294230556062
GNTTABOP_map_grant_ref worked.

other side: HI

System call 1

System call 2
```

The guest domain with the statement: “other side: HI”

3.2 Adding a Hypercall

Here we need to implement a new hypercall which prints a message “Hello world!”.

First I added a new C file inside xen/common folder called “print_hello.c” which implements a function do_some_hypercall(int op) which takes an argument, prints a message using printk and returns 0.

```
//print_hello.c
#include <xen/lib.h>
int do_some_hypercall(int op)
{
    printk("Hello world!");
    return 0;
}
```

Some other files that were modified:

xen/arch/x86/hypercall.c –	ARGS(some_hypercall, 1)
xen/arch/x86/guest/xen/hypercall_page.S –	DECLARE_HYPERCALL(some_hypercall)
xen/include/xen/hypercall.h –	Added extern int do_some_hypercall(int op);
xen/include/public/xen.h –	#define __HYPERVISOR_some_hypercall 47
xen/arch/x86/hvm/hypercall.c –	HYPERCALL(some_hypercall)
xen/arch/x86/pc/hypercall.c –	HYPERCALL(some_hypercall)
xen/common/Makefile –	obj-y += print_hello.o

It compiled successfully and the patch has been submitted.