# Assignment 2 Part 3

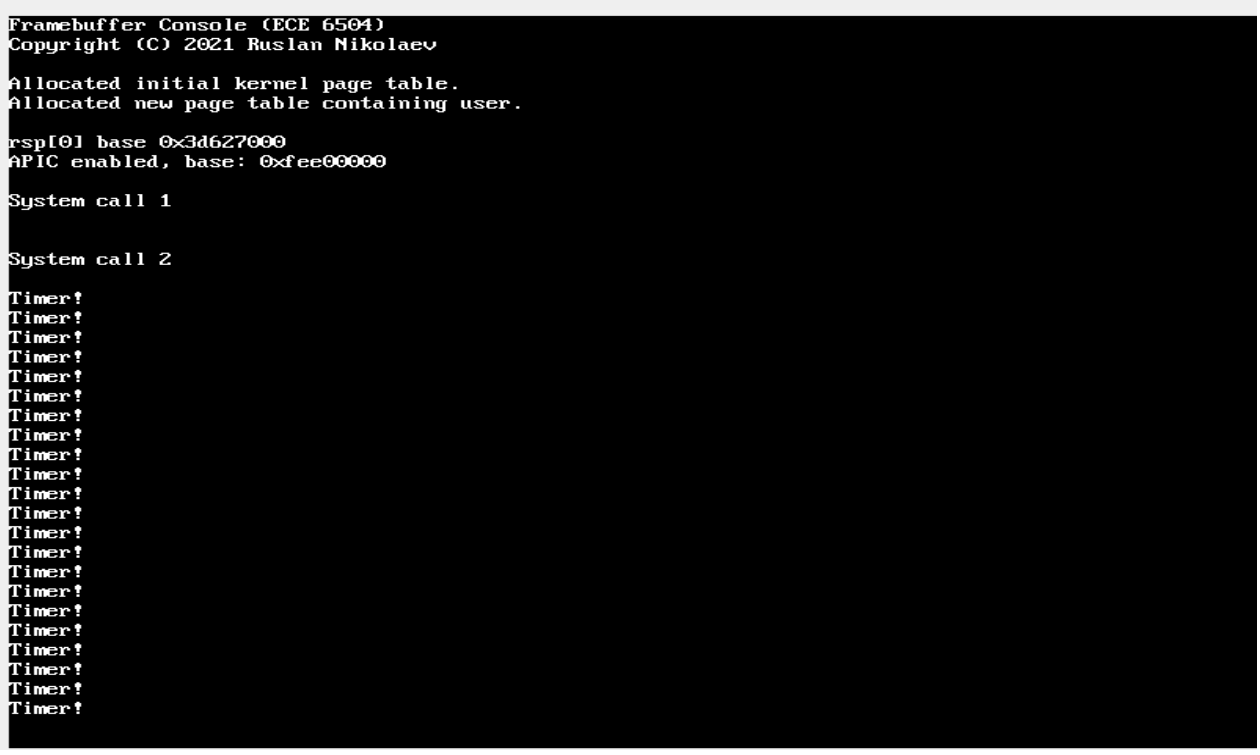
## 4.1 APIC Timer

In this task we need to implement an APIC based periodic timer.
First we need to call x86_lapic_enable() from apic.c which will automatically detect whether the system supports x2APIC or xAPIC. In my case, it detected xAPIC in the virtual box.

Then we need to initialize the timer such that it'll work in periodic mode.
To do that I created a function apic_init() in apic.h and implemented it in apic.c. That function sets the divide config register to divide by 128, or 0xA. It also sets the X86_Lapic_Timer_Init register to set initial count of 0x400000 and configures the X86_Lapic_Timer register to set the interrupt vector as 40 and set the mode as periodic.

I configured vector 40 in my interrupt descriptor table to contain the APIC timer handler function pointer. I also configured this pointer in my kernel_entry.S to point to the APIC handler function which is declared in kernel_asm.S. This handler calls a C function which prints the message "Timer!" on the screen and also acknowledges the interrupt by sending "0" to the EOI register.

The output can be seen in the below screenshot.



## 4.2 Thread Local Storage (TLS)

In this task we need to extend our user.c and kernel.c such that they support TLS.
To set up TLS, I first passed an additional page from boot.c in the user_addr buffer. I mapped this page in the lower 1GB user virtual address space.
TLS uses the FS register to run in Linux so I need to set up the FS register to point to the base TLS virtual address.

I used the “wrmsr” command to write 0xc0000100 to the rcx register (which means the FS register), the lower 32 bits of the TLS virtual address to the rax register and the upper 32 bits to the rdx register.

I also created a tls struct which contained a field *myself which points to itself. So, the first entry in the TLS page will point to itself and the remaining fields in the page will contain the variables (in our case a[i]).

Now, in user.c I include the provided lines of code and run the program without any errors.