MODEL INSTITUTE OF ENGINEERING AND TECHNOLOGY JAMMU

A REPORT ON CONVERTING A CODE FROM VIRTUAL ADDRESS TO PHYSICAL ADDRESS

BACHELOR OF ENGINEERING (Computer Science and Engineering)

SUBMITTED BY:

NAMES :- AMJID KHAN, ARJUMAND AYUB, ASHU PAL, JATIN KOUL

ROLL NUMBERS :- 2021A1R106, 2021A1R111, 2021A1R116, 2021A1R120

BRANCH :- CSE

SEMESTER :- 3RD

GROUP :- 7

ACKNOWLEDGEMENT

Through this section of our report, we want to present our gratitude towards our institute MIET. From here, we are able to work on projects that require you to analyse and solve a problem that exists in the real world. It is an experience that one can achieve rarely and we are happy and thankful to get that chance here.

We would also like to thank our mentors Asst. professor Pragti Jamwal and Dr. Swati Goel for being a guiding force for being a guiding force.

TABLE OF CONTENTS

INTRODUCTION

Memory is one of the most important host resources. For workloads to access global system memory, we need to make sure virtual memory addresses are mapped to the physical addresses. There are several components working together to perform these translations as efficient as possible. In this mini-project we develop a code to convert a virtual memory address to physical address.

PHYSICAL ADDRESS

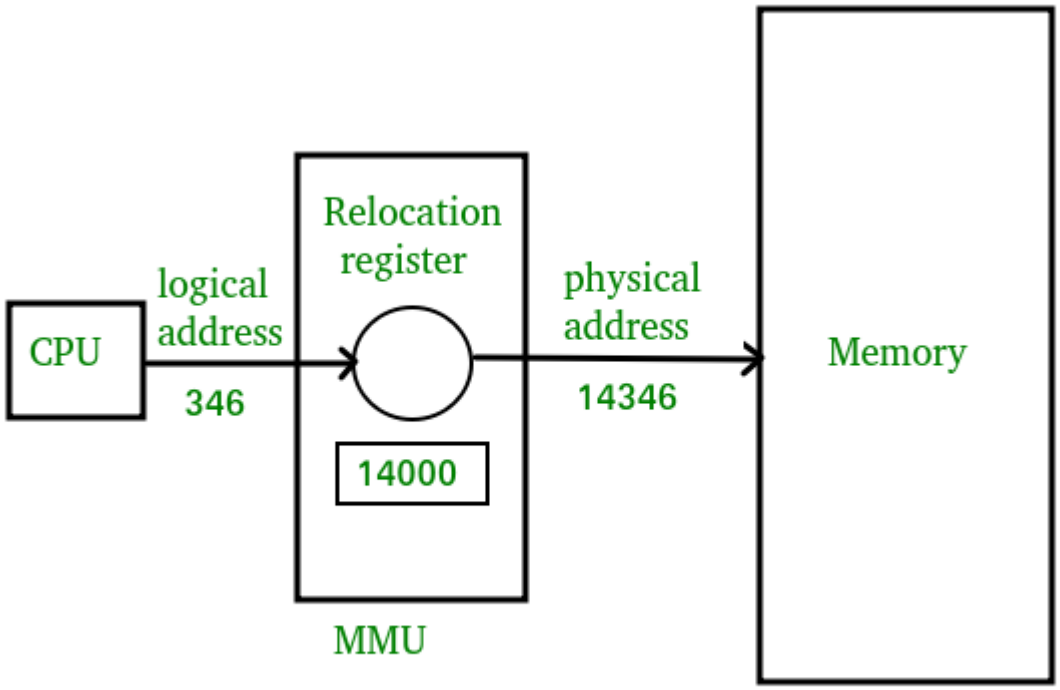
Physical Address identifies a physical location of required data in a memory. The user never directly deals with the physical address but can access by its corresponding logical address. The user program generates the logical address and thinks that the program is running in this logical address but the program needs physical memory for its execution, therefore, the logical address must be mapped to the physical address by MMU before they are used. The term Physical Address Space is used for all physical addresses corresponding to the logical addresses in a Logical address space.

MMU(Memory Management Unit)

The run time mapping between Virtual address and Physical Address is done by a hardware device known as MMU.

In memory management, the Operating System will handle the processes and move the processes between disk and memory for execution . It keeps track of available and used memory.

Dynamic relocation using a relocation register



1. CPU will generate logical address for eg: 346
2. MMU will generate a relocation register (base register) for eg: 14000
3. In memory, the physical address is located eg: (346+14000= 14346)

WHY WE NEED TO CONVERT VIRTUAL ADDRESS TO PHYSICAL ADDRESS?

To store the data and to manage the processes, we need a large-sized memory and, at the same time, we need to access the data as fast as possible. But if we increase the size of memory, the access time will also increase and, as we know, the CPU always generates addresses for secondary memory, i.e. logical addresses. But we want to access the main memory, so we need Address translation of logical address into physical address.

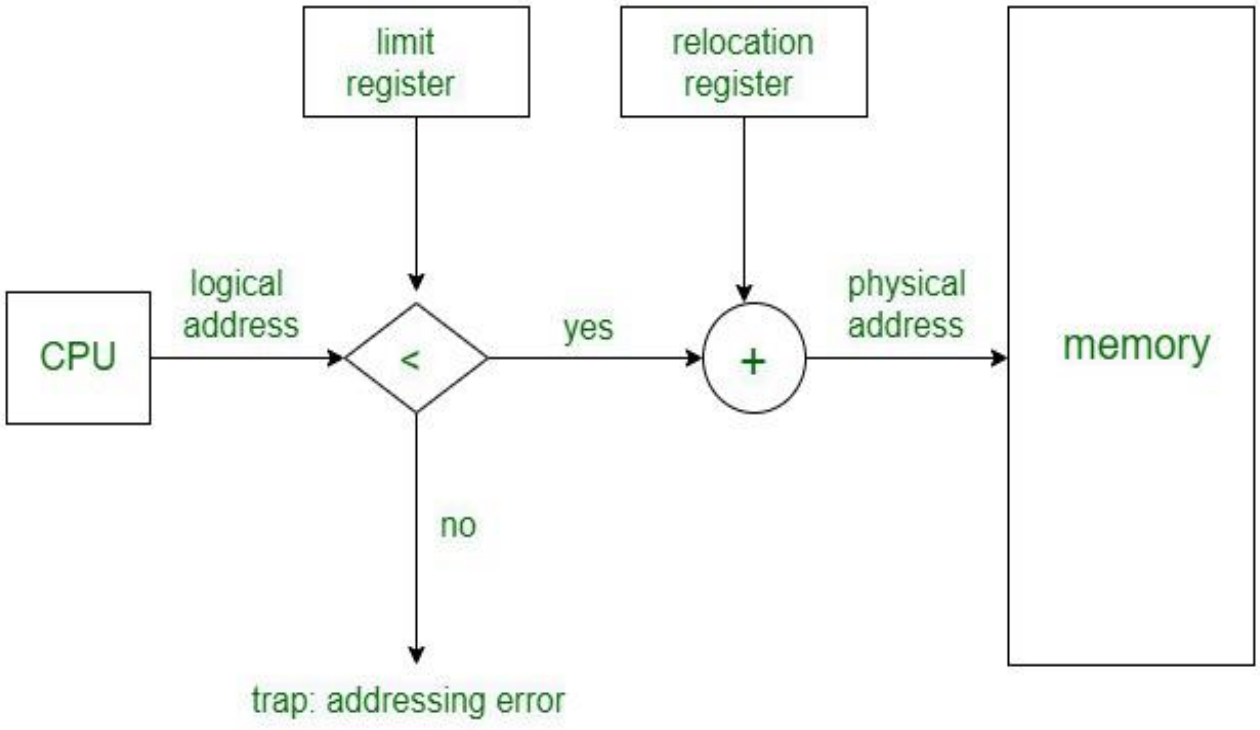
Mapping Virtual Addresses to Physical Addresses

In Contiguous memory allocation mapping from virtual addresses to physical addresses is not a difficult task, because if we take a process from secondary memory and copy it to the main memory, the addresses will be stored in a contiguous manner, so if we know the base address of the process, we can find out the next addresses.

The Memory Management Unit is a combination of 2 registers –

1. Base Register (Relocation Register)
2. Limit Register.

Mapping Virtual Addresses to Physical Addresses



Code to convert virtual address to physical address

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
int main(int argc, char *argv[]) {
// Get the virtual address from the user
```

```
printf("Enter a virtual address: ");
char input[128];
fgets(input, 128, stdin);

// Parse the virtual address from the input string
    void *virtual_address = (void *)strtoul(input, NULL, 16);

    // Open the /proc/self/pagemap file
    int pagemap_fd = open("/proc/self/pagemap", O_RDONLY);
    if (pagemap_fd < 0) {
    perror("Error opening /proc/self/pagemap");
    return 1;
    }
// Seek to the correct entry in the pagemap file
    off_t pagemap_offset = (unsigned long)virtual_address /
    getpagesize() * 8;
    if (lseek(pagemap_fd, pagemap_offset, SEEK_SET) !=
    pagemap_offset)
perror("Error seeking in /proc/self/pagemap");
    return 1;
    }
    // Read the entry from the pagemap file
    unsigned long pagemap_entry;
    if (read(pagemap_fd, &pagemap_entry, 8) != 8) {
```

```
        perror("Error reading from /proc/self/pagemap");
        return 1;
        }
        // Extract the physical page number from the pagemap entry
        unsigned long physical_page_number = pagemap_entry &
        ((1ull << 55) - 1);
// Compute the physical address by adding the offset within the
page
        void *physical_address = (void *)(physical_page_number *
        getpagesize() +
        (unsigned long)virtual_address % getpagesize());
        printf("Virtual address: %p\n", virtual_address);
        printf("Physical address: %p\n", physical_address);
        return 0;
        }
/*This program prompts the user to enter a virtual address and
then uses the /proc/self/pagemap file to look up the
corresponding physical address. The /proc/self/pagemap file is a
special file that is provided by the Linux kernel and contains the
entries in the current process's page table.

Keep in mind that this is just one way to convert a virtual address
to a physical address, and the exact method for doing so will
depend on the specific operating system and hardware being used
*/
```

OUTPUTS

```
arjumand@arjumand-virtualbox:~$ gedit code.c
arjumand@arjumand-virtualbox:~$ chmod u+x code.c
arjumand@arjumand-virtualbox:~$ ./code
Enter a virtual address: 215371
Virtual address: 0x215371
Physical address: 0x371
arjumand@arjumand-virtualbox:~$ ./code
Enter a virtual address: 123456
Virtual address: 0x123456
Physical address: 0x456
arjumand@arjumand-virtualbox:~$ ./code
Enter a virtual address: a1b5cd4f
Virtual address: 0xa1b5cd4f
Physical address: 0xd4f
arjumand@arjumand-virtualbox:~$ ./code
Enter a virtual address: 483
Virtual address: 0x483
Physical address: 0x483
arjumand@arjumand-virtualbox:~$ ./code
Enter a virtual address: 25
Virtual address: 0x25
Physical address: 0x25
arjumand@arjumand-virtualbox:~$ ./code
Enter a virtual address: 1
Virtual address: 0x1
Physical address: 0x1
arjumand@arjumand-virtualbox:~$
```

LOGICAL ADDRESS

- **Logical Address** is generated by CPU while a program is running. The logical address is virtual address as it does not exist physically, therefore, it is also known as Virtual Address. This address is used as a reference to access the physical memory location by CPU. The term Logical Address Space is used for the set of all logical addresses generated by a program's perspective.
- The hardware device called Memory-management Unit is used for mapping logical address to its corresponding physical address.