Instituto Tecnológico y de Estudios

Superiores de Occidente – ITESO



Subject: Computer Architecture

Professor: Aurelio Villalpando

Practice 1: Hanoi Towers

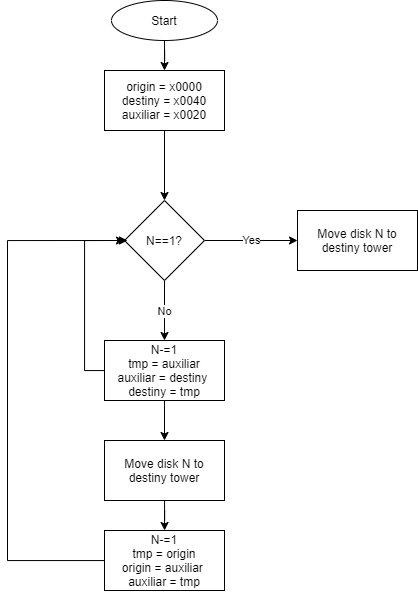
Session: 10

Date: February 22nd 2020

Authors: Godínez Rodríguez Alejandro 714181

Salas Venegas Ricardo 716159

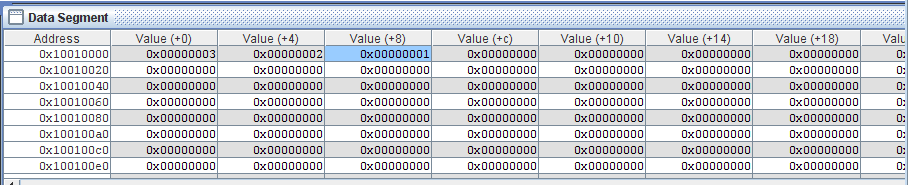
Flowchart

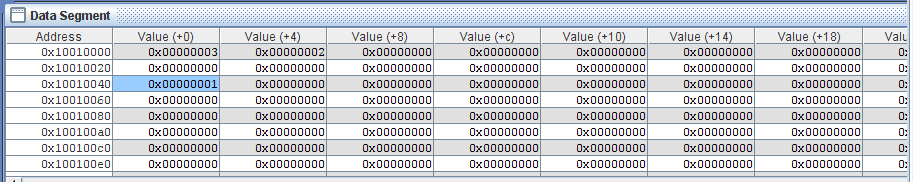


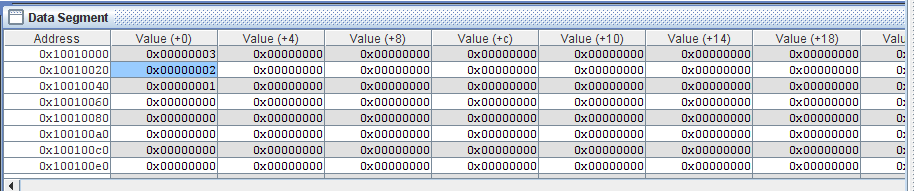
Important decisions for the elaboration of the Program

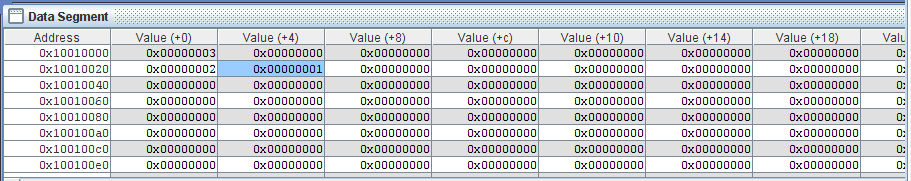
For the elaboration of the algorithm for the Hanoi Towers, we had to take several important decisions in order to make this algorithm recursive. The first one was to know which were the destiny tower, the origin tower, and the auxiliary tower. Therefore, we will be able to make decisions based on that information so that we can develop the base case. Taken that into consideration, addressing the problem became easier once we were able to know which tower was the one that we needed to send the disk.

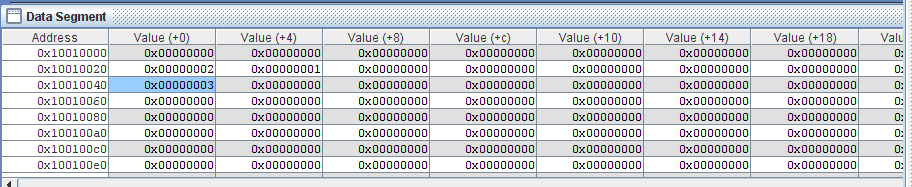
Mars Simulation with three Disks

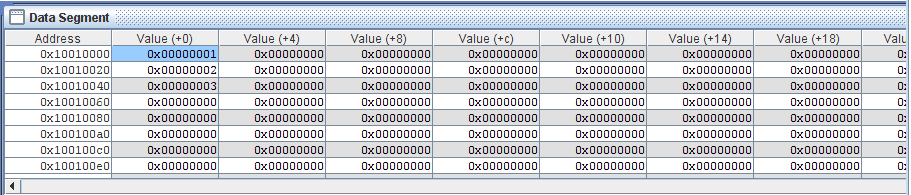


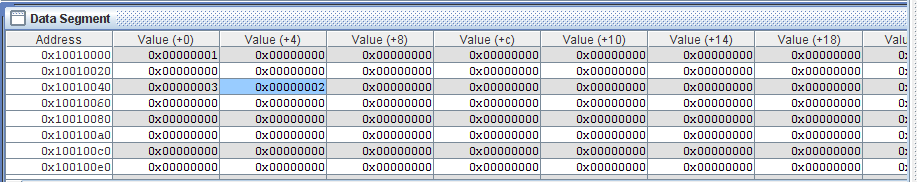


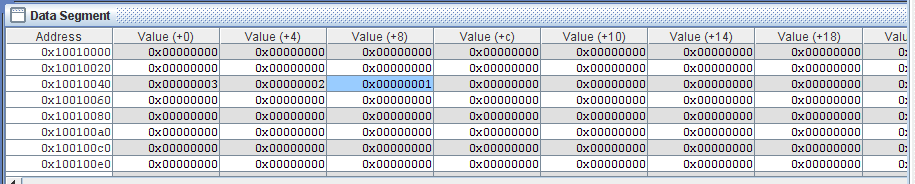








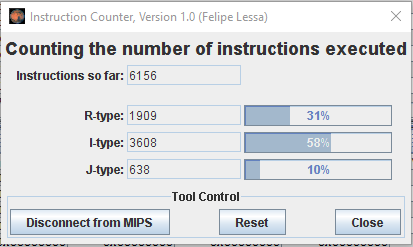


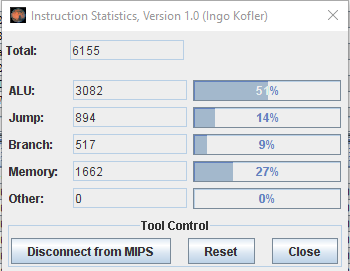


Stack Behavior Analysis

In order for us to be able to do recursive calls on the function MOV\_HANOI we had to use the register $ra for each call on the function. This is the reason we had to use stack. Also, in order for us to be able to get back to the previous disk, we ha to store along the $ra register, the N variable stored in the register $s0. So, in that way on the first memory block of the stack we stored the recursive call of the function ($ra) and on the second memory block we stored the register that stored the disk that had to be moved in that call of the function. So, every time we made a poll on the stack we used the $ra register to backtrack the recursive call of the function MOV\_HANOI, along with the disk register.

IC and Program size in bytes





Conclusions

Alejandro Godinez:

This problem was at first trickier to address, this is because we started solving this problem without a recursive algorithm. This meant a blocking in our progress once we realized that the algorithm needed to be recursive. Nevertheless, once we implemented a recursive algorithm the most difficult part of the program to solve was to program the computer to do the correct switching of the disks from the origin tower to the destiny one. Therefore, we had the problem solved on its majority but this difficulty meant a huge blocking in our progress because the solution did not came as easy.

Ricardo Salas:

Our first big problem was the time. We started making our own algorithm with a single issue, we didn’t needed stack or a recursive way. After we noticed we were obligated to use both, we need to search for lots of information.

After starting, we noticed a second problem. The tower addresses were just going crazy all around each other. We had to make a debugging exercise for hours. Every time we thought we were just around the corner; we noticed a new complication in our way. I really hated this practice.