CMPE 230 HW 1 REPORT

Students: Olcayto Türker - 2016400180, Sertay Akpınar – 2016400075

In this project, we implemented an A86 assembly language program that will input postfix expression involving hexadecimal quantities and evaluate it.

First of all, we implemented a read label to get the input command. Then we created some conditions that check the decimal of the input characters and jump to the related label according to their decimals. (Such as con2: checks the "+" sign then jump to the jmpaddition label.) We also created the intermediate labels to jump to the essential labels.

We implemented the operations in division, multiplication and addition labels. One register is taking the first operand, other register is taking the second. After the operation is done, we push the result to the stack. Then come back to "read1" to get the next input. In the bitwise labels we did nearly the same thing in the operation labels. "ax" pops the last value and the "bx" pops the second value, did the operation, push the result to the stack then come back to the "read1".

We implemented four stack labels. We created "_stack" and "stack3" label for the inputs more than one digit. In "_stack" we subtracted 30h(0's hexadecimal) from the given input to get the right value and we multiplied the top of the stack by 16 to get the right value(for the numbers more than 9). In the end of these labels, we decrement "bp" to control the number of digits in the input and come back to the read label again. In "stack3" we nearly did the same thing we do in "_stack". Stack3 is implemented for the "A, B, C, D, E, F" inputs. And "cx" multiply the top of the stack by 16.

“stack1” and “stack2” are for making the stack operations if the given input is a number or a letter. If it is number, we subtract 30h in stack1. “_a, _b, _c, _d, _e” labels subtract 55d from the input to get right value and jump to stack2. “space” label means there is nothing to read, so set the bp value to zero and keep on reading. “_enter” label means the input is ended so seperate the digits of the number and jump to the print label. We divide the result by 16 to get the every digits of the number one by one. We implemented a basic loop here like while the quotient is not zero jump to the _enter label. If the quotient is zero that means our code is ready for the print command.

In “print” label we compared the character’s value by A’ s hexadecimal. If it’s above or equal to 0Ah it is a letter so jump to “ifletter” label if not, it is a number so jump to “ifnumber” label. We jump to “print1” from the ifnumber and ifletter labels. In this label, we do the print command. We implemented a basic loop here like while the si’s value is not 0(that means the output is not over), jump to the “print” label. In the end, we give some inputs to try our code and we get the correct results.