Shiraz University

Assembly

Project Documentation

Part-2

Dr. Esfandiari

TA: Ali Maher

Deadline: *will be announced later*

"Part 2: Memory Visualization"

Introduction:

In the last part you have implemented an assembler. And Now in this part you have to display the memory. You need to use the machine codes you have generated in the part 1.

What do we have?

I have invented new microcontroller called ***X1313***. This microcontroller has a 256 Bytes memory. Also, keep in mind that the width of our memory is 8 bits.

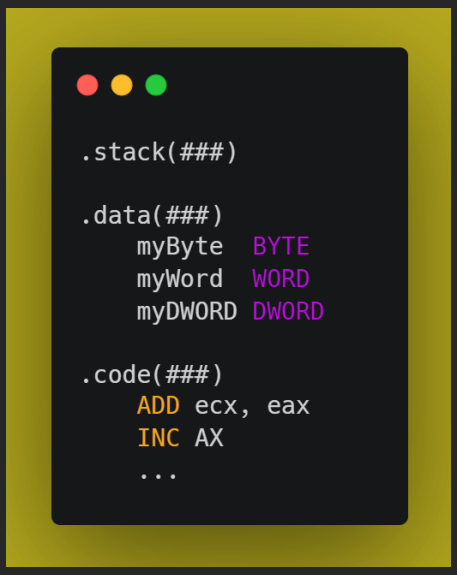
We have three segments: Code Segment, Data Segment and Stack Segment. (*We guarantee that there is no overlapping between these segments*)

Each segment has 32 Byte length.

Inputs:

The input is a txt file like the picture below. It contains from three parts: .stack - .data and .code.

In the parentheses will be some decimal numbers which point to the beginning of that segment. (*we guarantee that the entered number is even*)



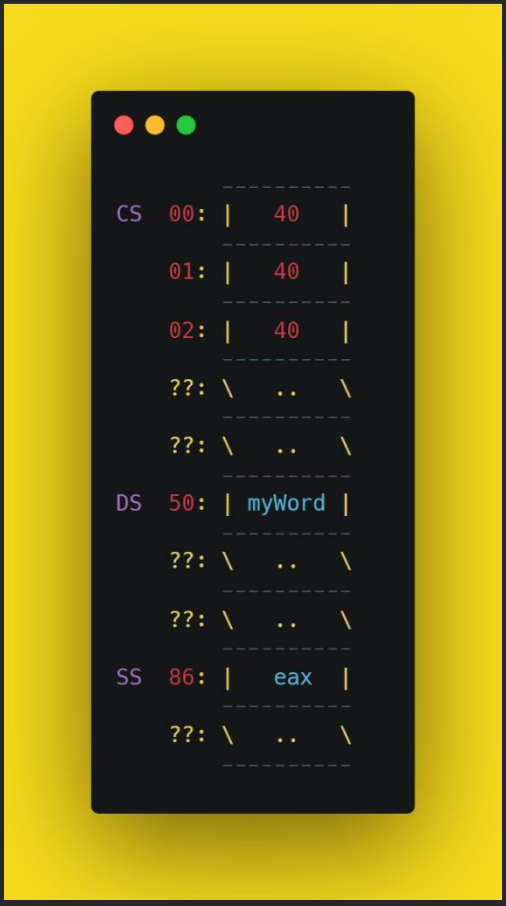
```
.stack(###)

.data(###)
    myByte  BYTE
    myWord  WORD
    myDWORD DWORD

.code(###)
    ADD ecx, eax
    INC AX
    ...
```

Outputs:

The output is the memory visualization corresponding to the input. Like below:



```
          ----------
CS  00: |   40   |
          ----------
    01: |   40   |
          ----------
    02: |   40   |
          ----------
    ??: \   ..   \
          ----------
    ??: \   ..   \
          ----------
DS  50: | myWord |
          ----------
    ??: \   ..   \
          ----------
    ??: \   ..   \
          ----------
SS  86: |   eax  |
          ----------
    ??: \   ..   \
          ----------
```

Memory Filling:

Consider following tips in visualizing the memory:

- **How to fill the stack segment?** consider all the information stored in the stack to be 32-bit. (Consider it Word Align)
  - If a register pushed, based on the register size fill the memory with the name of the register.
  - If a number pushed base on the number size fill the memory with the number.
  - Otherwise, fill them with "MM".

…	…	…	…	…
eax	ax	num8	num16	num32
eax	ax	MM	num16	num32
eax	MM	MM	MM	num32
eax	MM	MM	MM	num32
…	…	…	…	…

- **How to fill the data segment?** This data which is given in the *.data* part of the input, based on their size occupy enough bytes of the memory.
  Examples:

  .data(100)

  myByte ***BYTE***

  myWord ***WORD***

  myDword ***DWORD***

99	…
100	myByte
101	myWord
102	myWord
103	myDword
104	myDword
105	myDword
106	myDword
107	…

- **How to fill the code segment?** You have to fill them with their machine code that you calculated in the part-1 of this project. And note that we have even align here, which means starting of each instruction regardless of the last instruction length must be at a even byte.
  Example:
  .code(0)

  INC ax ➔ *0x40*

0	40
1	MM
2	…

- Fill the bytes that are not used at all and are not in any segment with "XX".
- Fill the bytes that are not used but they are part of a segment with Mouse, for consistency consider it "MM".
- As shown in the sample output make sure to determine the beginning of each segment.
- Each byte has to be addressed. (like the sample output)
- Make sure you follow the right endianness.
- Consider address of the topmost part of the memory zero and address of the lowermost part of the memory 255.

Other Tips:

In the memory of this microprocessor the logical address is exactly the physical address.

Hint: for better handling you can get use of some pointers to determine the offset of each start of each segment to the place they want to right next time. Like SP, DP, IP. (Stack pointer, Data pointer and Instruction pointer)

What to do with labels and jump instructions? Implementation of this part will be considered as bonus. If you see a label you have to add the label in the data segment with its jumping offset.

Scoring criteria: (120 Pts)

- Correctness (100 Pts)
- Report and Documentation (10 Pts)
- Comment (First comment, In code comment) (5 Pts)
- Clean Code (5 Pts)
- Any other creativity will be concerned as bonus.
- The segments starter address controlling is bonus. (SS: 0, DS: 100, CS: 200)

Note: These criteria may change.

You do not score any if you don't attend for presentation.

• Upload the expected files in a zipped file to the page of the course on Quera.

• Do NOT copy from your classmates' answers :)

• Feel free to ask any questions.

Good Luck!

Ali Maher