芝浦工業大学

MINIX Disassembler

Advanced OS and Virtualization

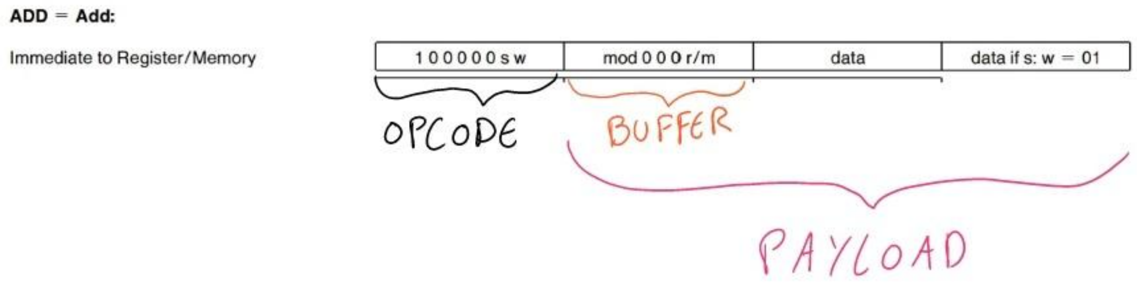
Rodrigo Ferraz Souza
z123086

Implementation Strategy

The disassembler was implemented in C++ using a class-based approach, with each possible command of the Intel 8086 represented as a separate class.

Program Execution

Upon program execution, the user provides the absolute or relative path of the program they wish to analyze as a command-line parameter. This path is then passed as a parameter to the fileReader() function, which reads the program byte by byte. Each byte is compared to the opcode table using a hash table to determine if it corresponds to a valid command. If a matching command is found, an instance of the appropriate command class is created, with the opcode and the following byte passed as parameters to the class constructor. This is necessary as some commands require the first two bytes for proper identification.



ADD = Add:

Immediate to Register/Memory	1 0 0 0 0 0 s w	mod 0 0 0 r/m	data	data if s: w = 01

Parsing Instructions

Once a command is correctly identified, the disassembler determines the size of the command's payload, reads it, and adds it to the instantiated object. The object is then added to a vector<Instruction*>. This vector, containing all the instructions with their respective payloads, is returned to the main() function upon completion of the fileReader().

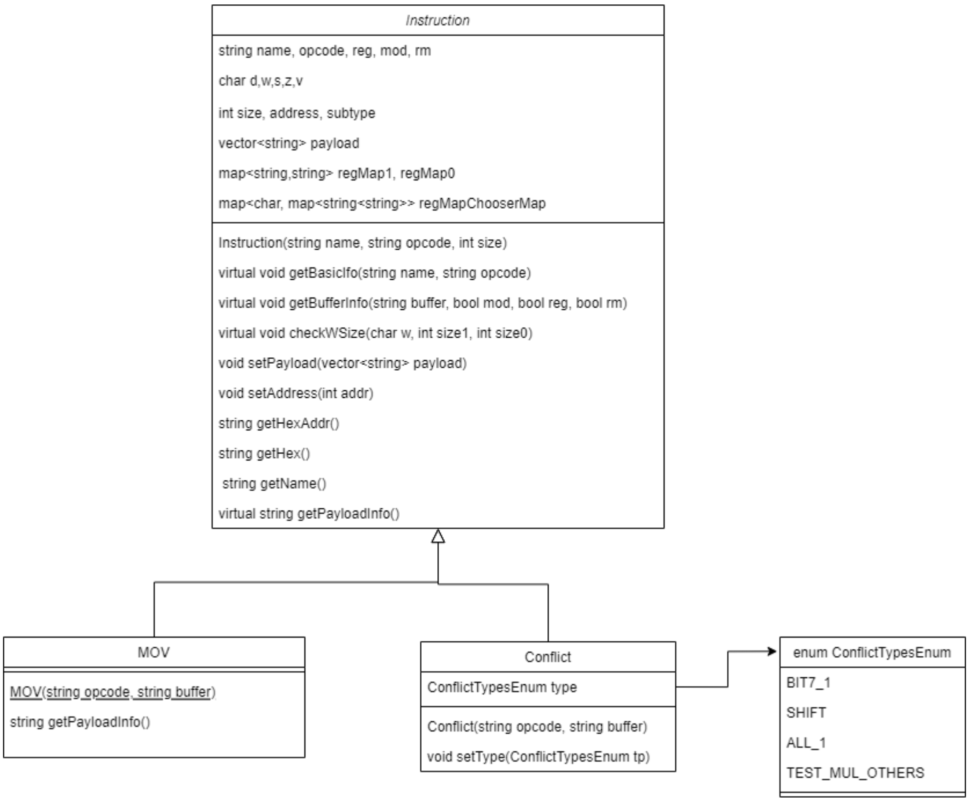
Printing Instructions

In the main() function, each item in the instruction vector is iterated, and the printInst() function is called, passing the instruction pointer as a parameter.

Within the printInst() function, the output is generated by invoking methods from the parent Instruction class to retrieve the instruction's address, hexadecimal code, command name, and payload information, which is converted into assembly code.

Class Structure

The implementation of the disassembler utilizes a class structure, with each command represented by a separate class. The diagram below provides a simplified representation of the class structure for better comprehension.


Instruction
string name, opcode, reg, mod, rm
char d,w,s,z,v
int size, address, subtype
vector<string> payload
map<string,string> regMap1, regMap0
map<char, map<string<string>> regMapChooserMap
Instruction(string name, string opcode, int size)
virtual void getBasicIfo(string name, string opcode)
virtual void getBufferInfo(string buffer, bool mod, bool reg, bool rm)
virtual void checkWSize(char w, int size1, int size0)
void setPayload(vector<string> payload)
void setAddress(int addr)
string getHexAddr()
string getHex()
string getName()
virtual string getPayloadInfo()
MOV
MOV(string opcode, string buffer)
string getPayloadInfo()
Conflict
ConflictTypesEnum type
Conflict(string opcode, string buffer)
void setType(ConflictTypesEnum tp)
enum ConflictTypesEnum
BIT7_1
SHIFT
ALL_1
TEST_MUL_OTHERS

Handling Opcode Conflicts

Several commands in the Intel 8086 architecture have conflicting opcodes, which can only be resolved by considering the second byte. In such cases, the hash map that returns instruction constructors will return a constructor from a special class called Conflict. This class has four different types, specified by an enum.

If a conflict is identified in the fileReader() function, the checkConflict() function is called, passing the buffer containing the second byte. The function uses the object's enum to identify the conflict type and then queries the appropriate hash map using the identifier in the buffer to retrieve the correct constructor.

Disassembler Development and Execution Issue

Despite substantial efforts, the development of the disassembler could not be fully completed within the given time frame. However, it is important to note that the architectural framework in place is capable of supporting the project's eventual realization. The implementation of payload analysis for all commands was partially achieved, with only a subset of commands receiving full analysis due to time constraints. However, further improvements are required to cover the remaining commands and ensure a comprehensive disassembly process.

Additionally, an issue arose during execution regarding the identification of the last command within the analyzed program. While all other commands were accurately identified, finding a consistent pattern for the last command proved challenging. Attempts were made to determine a pattern based on the binary data size after the last payload, but this functionality was not yet implemented. Addressing this execution issue would contribute to the overall completeness and effectiveness of the disassembler.

Program Compilation, Execution, and Folder Organization

Program Compilation:

The program was provided in a compressed folder in .rar format. Within this folder, at its root, there are two .cpp files and two .hpp files, along with a folder named test. The binaries to be analyzed are located inside the test folder. To compile the program, navigate to the project's root folder where the .cpp files are located and execute the following command:

```
g++ *.cpp -o out
```

Program Execution:

To execute the program, use the following command:

```
./out <PATH TO THE BINARY>
```

For example: ./out test/1c.out

Folder Organization:

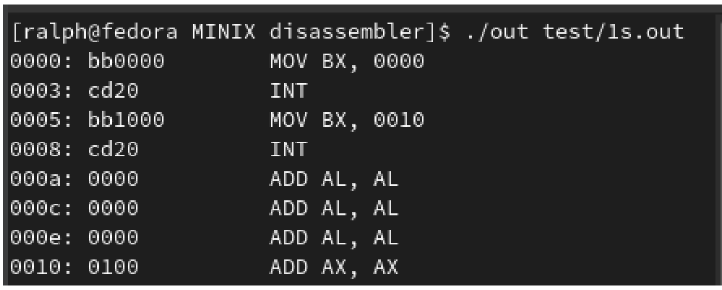
The project folder structure is organized as follows:

Root Folder: Contains the program source files (.cpp and .hpp) and the test folder.

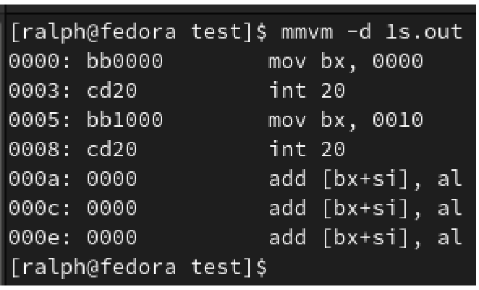
test Folder: Contains the binaries that should be analyzed using the program. It is recommended to place the binary file(s) within this folder and provide the appropriate path when executing the program, as demonstrated in the example above.

Examples of execution

1.s

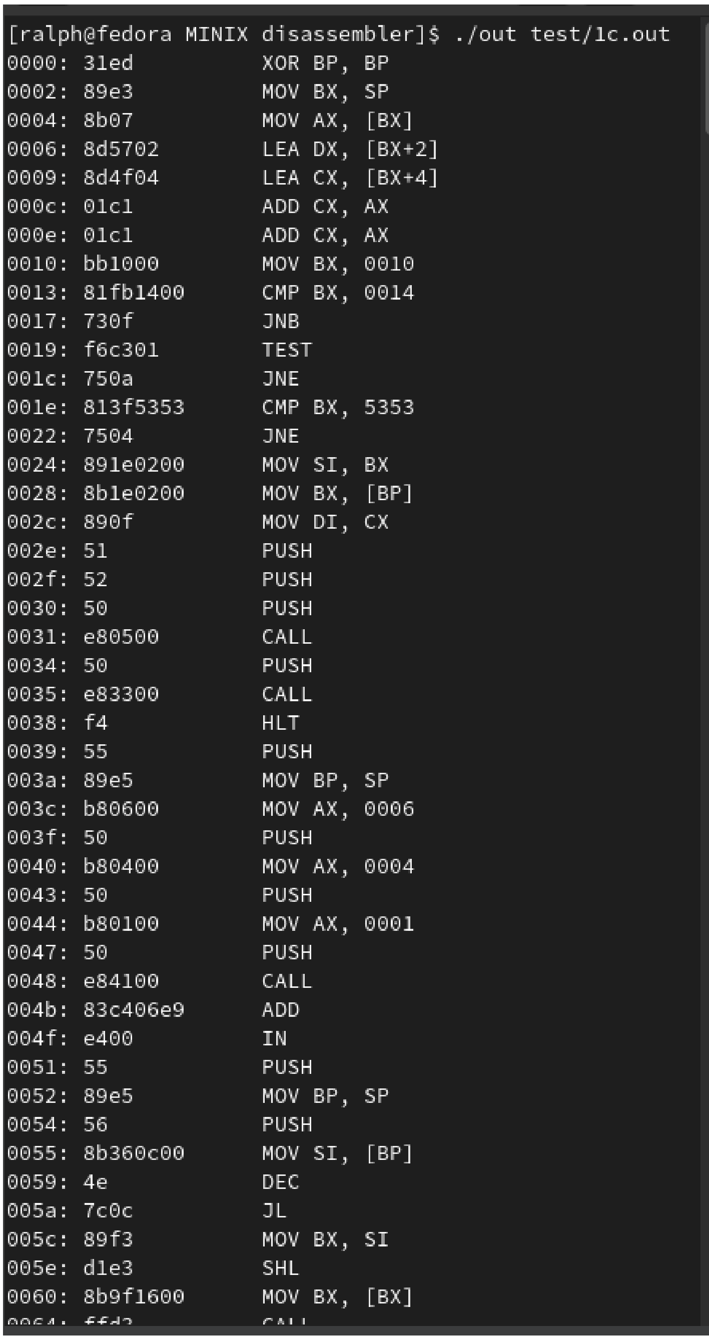


```
[ralph@fedora MINIX disassembler]$ ./out test/1s.out
0000: bb0000        MOV BX, 0000
0003: cd20          INT
0005: bb1000        MOV BX, 0010
0008: cd20          INT
000a: 0000          ADD AL, AL
000c: 0000          ADD AL, AL
000e: 0000          ADD AL, AL
0010: 0100          ADD AX, AX
```

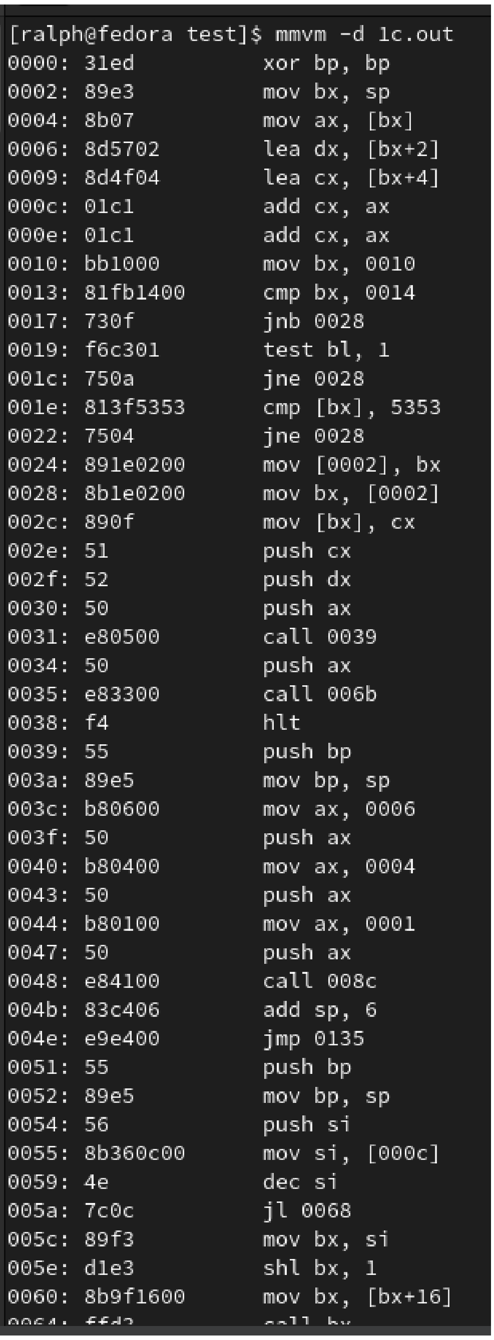


```
[ralph@fedora test]$ mmvm -d 1s.out
0000: bb0000        mov bx, 0000
0003: cd20          int 20
0005: bb1000        mov bx, 0010
0008: cd20          int 20
000a: 0000          add [bx+si], al
000c: 0000          add [bx+si], al
000e: 0000          add [bx+si], al
[ralph@fedora test]$
```

1.c

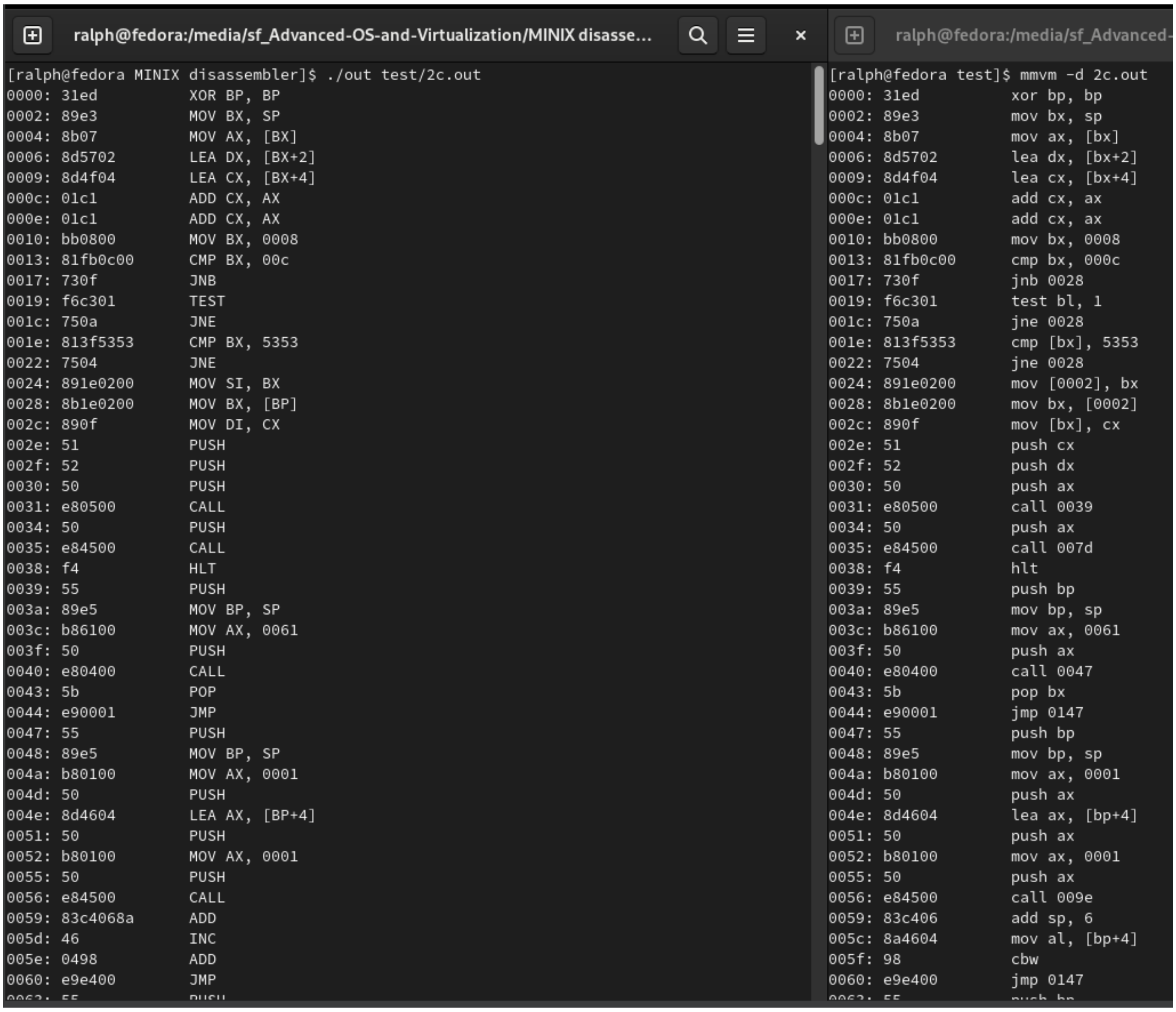


```
[ralph@fedora MINIX disassembler]$ ./out test/1c.out
0000: 31ed          XOR BP, BP
0002: 89e3          MOV BX, SP
0004: 8b07          MOV AX, [BX]
0006: 8d5702        LEA DX, [BX+2]
0009: 8d4f04        LEA CX, [BX+4]
000c: 01c1          ADD CX, AX
000e: 01c1          ADD CX, AX
0010: bb1000        MOV BX, 0010
0013: 81fb1400      CMP BX, 0014
0017: 730f          JNB
0019: f6c301        TEST
001c: 750a          JNE
001e: 813f5353      CMP BX, 5353
0022: 7504          JNE
0024: 891e0200      MOV SI, BX
0028: 8b1e0200      MOV BX, [BP]
002c: 890f          MOV DI, CX
002e: 51            PUSH
002f: 52            PUSH
0030: 50            PUSH
0031: e80500        CALL
0034: 50            PUSH
0035: e83300        CALL
0038: f4            HLT
0039: 55            PUSH
003a: 89e5          MOV BP, SP
003c: b80600        MOV AX, 0006
003f: 50            PUSH
0040: b80400        MOV AX, 0004
0043: 50            PUSH
0044: b80100        MOV AX, 0001
0047: 50            PUSH
0048: e84100        CALL
004b: 83c406e9      ADD
004f: e400          IN
0051: 55            PUSH
0052: 89e5          MOV BP, SP
0054: 56            PUSH
0055: 8b360c00      MOV SI, [BP]
0059: 4e            DEC
005a: 7c0c          JL
005c: 89f3          MOV BX, SI
005e: d1e3          SHL
0060: 8b9f1600      MOV BX, [BX]
0064: ffd3          CALL
```



```
[ralph@fedora test]$ mmvm -d 1c.out
0000: 31ed          xor bp, bp
0002: 89e3          mov bx, sp
0004: 8b07          mov ax, [bx]
0006: 8d5702        lea dx, [bx+2]
0009: 8d4f04        lea cx, [bx+4]
000c: 01c1          add cx, ax
000e: 01c1          add cx, ax
0010: bb1000        mov bx, 0010
0013: 81fb1400      cmp bx, 0014
0017: 730f          jnb 0028
0019: f6c301        test bl, 1
001c: 750a          jne 0028
001e: 813f5353      cmp [bx], 5353
0022: 7504          jne 0028
0024: 891e0200      mov [0002], bx
0028: 8b1e0200      mov bx, [0002]
002c: 890f          mov [bx], cx
002e: 51            push cx
002f: 52            push dx
0030: 50            push ax
0031: e80500        call 0039
0034: 50            push ax
0035: e83300        call 006b
0038: f4            hlt
0039: 55            push bp
003a: 89e5          mov bp, sp
003c: b80600        mov ax, 0006
003f: 50            push ax
0040: b80400        mov ax, 0004
0043: 50            push ax
0044: b80100        mov ax, 0001
0047: 50            push ax
0048: e84100        call 008c
004b: 83c406        add sp, 6
004e: e9e400        jmp 0135
0051: 55            push bp
0052: 89e5          mov bp, sp
0054: 56            push si
0055: 8b360c00      mov si, [000c]
0059: 4e            dec si
005a: 7c0c          jl 0068
005c: 89f3          mov bx, si
005e: d1e3          shl bx, 1
0060: 8b9f1600      mov bx, [bx+16]
0064: ffd3          call bx
```

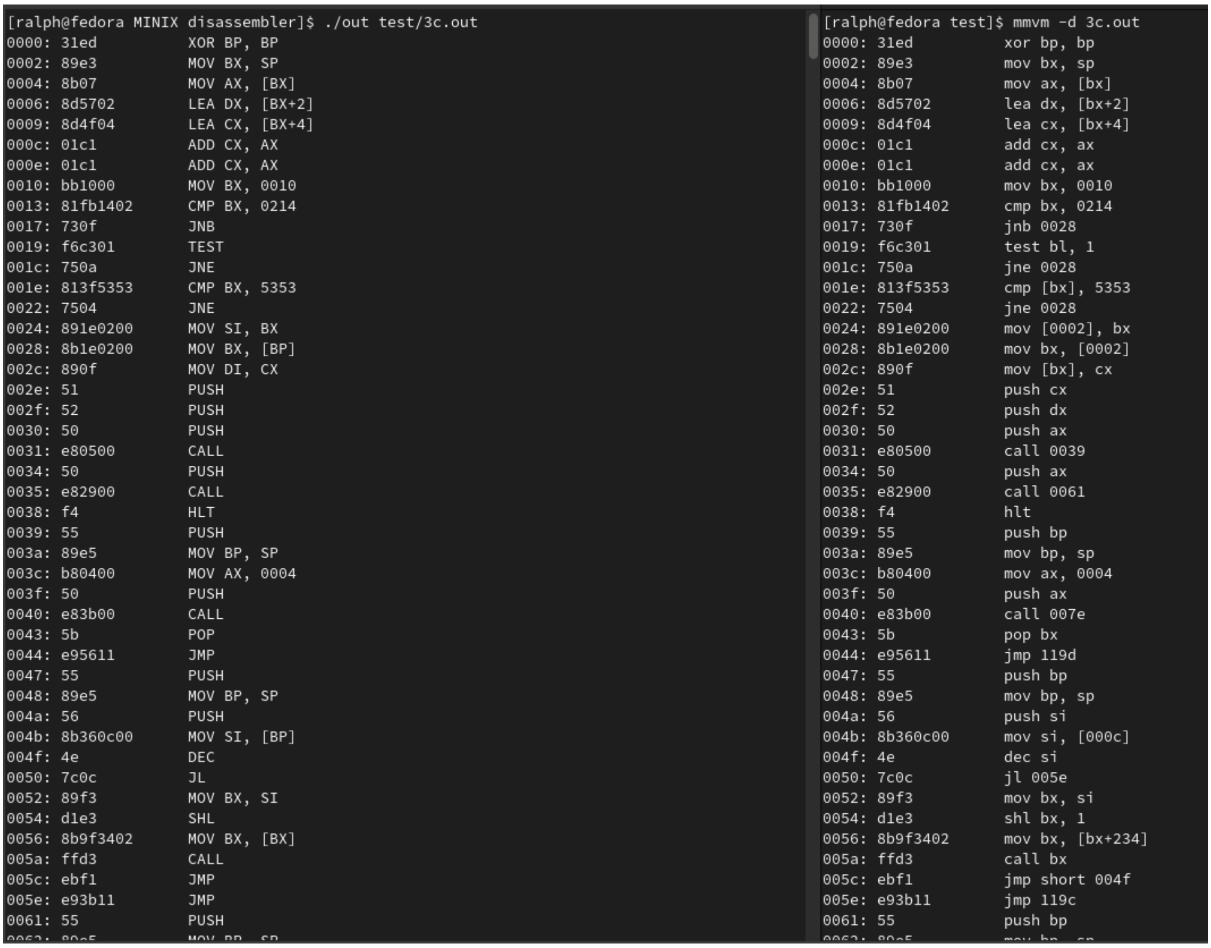
2.c


ralph@fedora:/media/sf_Advanced-OS-and-Virtualization/MINIX disasse…
ralph@fedora:/media/sf_Advanced-

```
[ralph@fedora MINIX disassembler]$ ./out test/2c.out
0000: 31ed          XOR BP, BP
0002: 89e3          MOV BX, SP
0004: 8b07          MOV AX, [BX]
0006: 8d5702        LEA DX, [BX+2]
0009: 8d4f04        LEA CX, [BX+4]
000c: 01c1          ADD CX, AX
000e: 01c1          ADD CX, AX
0010: bb0800        MOV BX, 0008
0013: 81fb0c00      CMP BX, 00c
0017: 730f          JNB
0019: f6c301        TEST
001c: 750a          JNE
001e: 813f5353      CMP BX, 5353
0022: 7504          JNE
0024: 891e0200      MOV SI, BX
0028: 8b1e0200      MOV BX, [BP]
002c: 890f          MOV DI, CX
002e: 51            PUSH
002f: 52            PUSH
0030: 50            PUSH
0031: e80500        CALL
0034: 50            PUSH
0035: e84500        CALL
0038: f4            HLT
0039: 55            PUSH
003a: 89e5          MOV BP, SP
003c: b86100        MOV AX, 0061
003f: 50            PUSH
0040: e80400        CALL
0043: 5b            POP
0044: e90001        JMP
0047: 55            PUSH
0048: 89e5          MOV BP, SP
004a: b80100        MOV AX, 0001
004d: 50            PUSH
004e: 8d4604        LEA AX, [BP+4]
0051: 50            PUSH
0052: b80100        MOV AX, 0001
0055: 50            PUSH
0056: e84500        CALL
0059: 83c4068a      ADD
005d: 46            INC
005e: 0498          ADD
0060: e9e400        JMP
0063: 55            PUSH
```

```
[ralph@fedora test]$ mmvm -d 2c.out
0000: 31ed          xor bp, bp
0002: 89e3          mov bx, sp
0004: 8b07          mov ax, [bx]
0006: 8d5702        lea dx, [bx+2]
0009: 8d4f04        lea cx, [bx+4]
000c: 01c1          add cx, ax
000e: 01c1          add cx, ax
0010: bb0800        mov bx, 0008
0013: 81fb0c00      cmp bx, 000c
0017: 730f          jnb 0028
0019: f6c301        test bl, 1
001c: 750a          jne 0028
001e: 813f5353      cmp [bx], 5353
0022: 7504          jne 0028
0024: 891e0200      mov [0002], bx
0028: 8b1e0200      mov bx, [0002]
002c: 890f          mov [bx], cx
002e: 51            push cx
002f: 52            push dx
0030: 50            push ax
0031: e80500        call 0039
0034: 50            push ax
0035: e84500        call 007d
0038: f4            hlt
0039: 55            push bp
003a: 89e5          mov bp, sp
003c: b86100        mov ax, 0061
003f: 50            push ax
0040: e80400        call 0047
0043: 5b            pop bx
0044: e90001        jmp 0147
0047: 55            push bp
0048: 89e5          mov bp, sp
004a: b80100        mov ax, 0001
004d: 50            push ax
004e: 8d4604        lea ax, [bp+4]
0051: 50            push ax
0052: b80100        mov ax, 0001
0055: 50            push ax
0056: e84500        call 009e
0059: 83c406        add sp, 6
005c: 8a4604        mov al, [bp+4]
005f: 98            cbw
0060: e9e400        jmp 0147
0063: 55            push bp
```

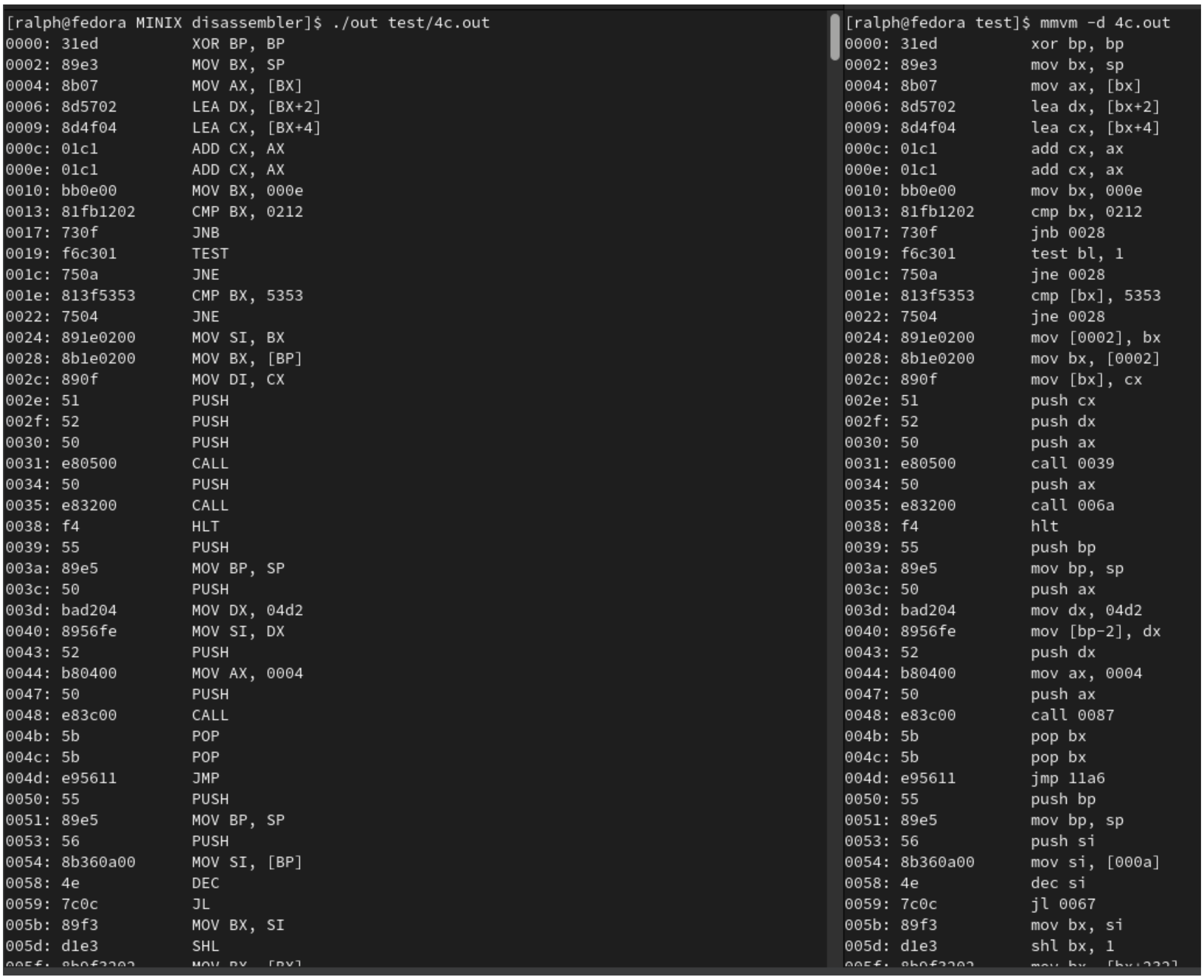
3.c



```
[ralph@fedora MINIX disassembler]$ ./out test/3c.out
0000: 31ed          XOR BP, BP
0002: 89e3          MOV BX, SP
0004: 8b07          MOV AX, [BX]
0006: 8d5702        LEA DX, [BX+2]
0009: 8d4f04        LEA CX, [BX+4]
000c: 01c1          ADD CX, AX
000e: 01c1          ADD CX, AX
0010: bb1000        MOV BX, 0010
0013: 81fb1402      CMP BX, 0214
0017: 730f          JNB
0019: f6c301        TEST
001c: 750a          JNE
001e: 813f5353      CMP BX, 5353
0022: 7504          JNE
0024: 891e0200      MOV SI, BX
0028: 8b1e0200      MOV BX, [BP]
002c: 890f          MOV DI, CX
002e: 51            PUSH
002f: 52            PUSH
0030: 50            PUSH
0031: e80500        CALL
0034: 50            PUSH
0035: e82900        CALL
0038: f4            HLT
0039: 55            PUSH
003a: 89e5          MOV BP, SP
003c: b80400        MOV AX, 0004
003f: 50            PUSH
0040: e83b00        CALL
0043: 5b            POP
0044: e95611        JMP
0047: 55            PUSH
0048: 89e5          MOV BP, SP
004a: 56            PUSH
004b: 8b360c00      MOV SI, [BP]
004f: 4e            DEC
0050: 7c0c          JL
0052: 89f3          MOV BX, SI
0054: d1e3          SHL
0056: 8b9f3402      MOV BX, [BX]
005a: ffd3          CALL
005c: ebf1          JMP
005e: e93b11        JMP
0061: 55            PUSH
0062: 89e5          MOV BP, SP
```

```
[ralph@fedora test]$ mmvm -d 3c.out
0000: 31ed          xor bp, bp
0002: 89e3          mov bx, sp
0004: 8b07          mov ax, [bx]
0006: 8d5702        lea dx, [bx+2]
0009: 8d4f04        lea cx, [bx+4]
000c: 01c1          add cx, ax
000e: 01c1          add cx, ax
0010: bb1000        mov bx, 0010
0013: 81fb1402      cmp bx, 0214
0017: 730f          jnb 0028
0019: f6c301        test bl, 1
001c: 750a          jne 0028
001e: 813f5353      cmp [bx], 5353
0022: 7504          jne 0028
0024: 891e0200      mov [0002], bx
0028: 8b1e0200      mov bx, [0002]
002c: 890f          mov [bx], cx
002e: 51            push cx
002f: 52            push dx
0030: 50            push ax
0031: e80500        call 0039
0034: 50            push ax
0035: e82900        call 0061
0038: f4            hlt
0039: 55            push bp
003a: 89e5          mov bp, sp
003c: b80400        mov ax, 0004
003f: 50            push ax
0040: e83b00        call 007e
0043: 5b            pop bx
0044: e95611        jmp 119d
0047: 55            push bp
0048: 89e5          mov bp, sp
004a: 56            push si
004b: 8b360c00      mov si, [000c]
004f: 4e            dec si
0050: 7c0c          jl 005e
0052: 89f3          mov bx, si
0054: d1e3          shl bx, 1
0056: 8b9f3402      mov bx, [bx+234]
005a: ffd3          call bx
005c: ebf1          jmp short 004f
005e: e93b11        jmp 119c
0061: 55            push bp
0062: 89e5          mov bp, sp
```

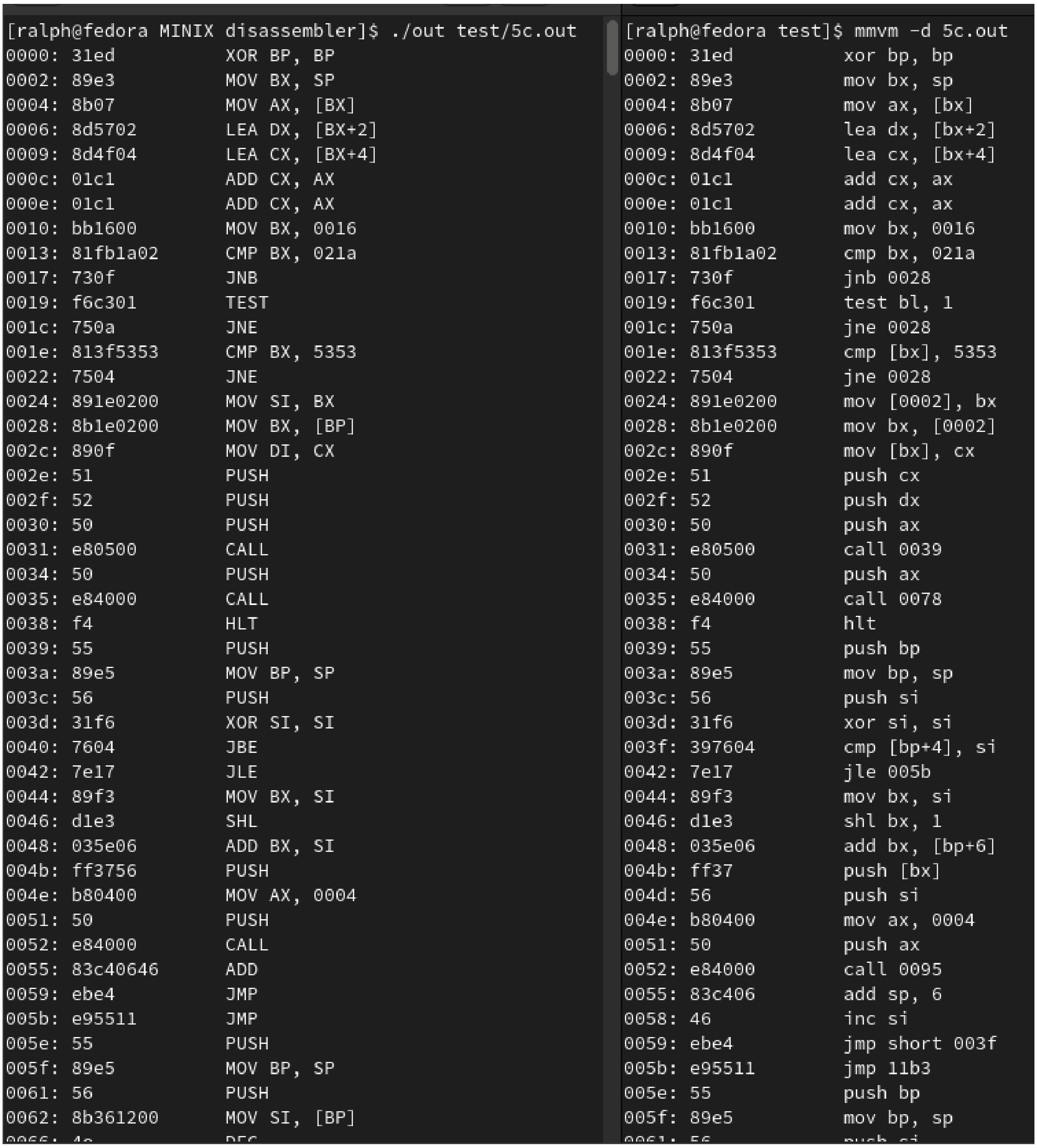
4.c



```
[ralph@fedora MINIX disassembler]$ ./out test/4c.out
0000: 31ed          XOR BP, BP
0002: 89e3          MOV BX, SP
0004: 8b07          MOV AX, [BX]
0006: 8d5702        LEA DX, [BX+2]
0009: 8d4f04        LEA CX, [BX+4]
000c: 01c1          ADD CX, AX
000e: 01c1          ADD CX, AX
0010: bb0e00        MOV BX, 000e
0013: 81fb1202      CMP BX, 0212
0017: 730f          JNB
0019: f6c301        TEST
001c: 750a          JNE
001e: 813f5353      CMP BX, 5353
0022: 7504          JNE
0024: 891e0200      MOV SI, BX
0028: 8b1e0200      MOV BX, [BP]
002c: 890f          MOV DI, CX
002e: 51            PUSH
002f: 52            PUSH
0030: 50            PUSH
0031: e80500        CALL
0034: 50            PUSH
0035: e83200        CALL
0038: f4            HLT
0039: 55            PUSH
003a: 89e5          MOV BP, SP
003c: 50            PUSH
003d: bad204        MOV DX, 04d2
0040: 8956fe        MOV SI, DX
0043: 52            PUSH
0044: b80400        MOV AX, 0004
0047: 50            PUSH
0048: e83c00        CALL
004b: 5b            POP
004c: 5b            POP
004d: e95611        JMP
0050: 55            PUSH
0051: 89e5          MOV BP, SP
0053: 56            PUSH
0054: 8b360a00      MOV SI, [BP]
0058: 4e            DEC
0059: 7c0c          JL
005b: 89f3          MOV BX, SI
005d: d1e3          SHL
005f: 8b9f3202      MOV BX, [BX]
```

```
[ralph@fedora test]$ mmvm -d 4c.out
0000: 31ed          xor bp, bp
0002: 89e3          mov bx, sp
0004: 8b07          mov ax, [bx]
0006: 8d5702        lea dx, [bx+2]
0009: 8d4f04        lea cx, [bx+4]
000c: 01c1          add cx, ax
000e: 01c1          add cx, ax
0010: bb0e00        mov bx, 000e
0013: 81fb1202      cmp bx, 0212
0017: 730f          jnb 0028
0019: f6c301        test bl, 1
001c: 750a          jne 0028
001e: 813f5353      cmp [bx], 5353
0022: 7504          jne 0028
0024: 891e0200      mov [0002], bx
0028: 8b1e0200      mov bx, [0002]
002c: 890f          mov [bx], cx
002e: 51            push cx
002f: 52            push dx
0030: 50            push ax
0031: e80500        call 0039
0034: 50            push ax
0035: e83200        call 006a
0038: f4            hlt
0039: 55            push bp
003a: 89e5          mov bp, sp
003c: 50            push ax
003d: bad204        mov dx, 04d2
0040: 8956fe        mov [bp-2], dx
0043: 52            push dx
0044: b80400        mov ax, 0004
0047: 50            push ax
0048: e83c00        call 0087
004b: 5b            pop bx
004c: 5b            pop bx
004d: e95611        jmp 11a6
0050: 55            push bp
0051: 89e5          mov bp, sp
0053: 56            push si
0054: 8b360a00      mov si, [000a]
0058: 4e            dec si
0059: 7c0c          jl 0067
005b: 89f3          mov bx, si
005d: d1e3          shl bx, 1
005f: 8b9f3202      mov bx, [bx+232]
```

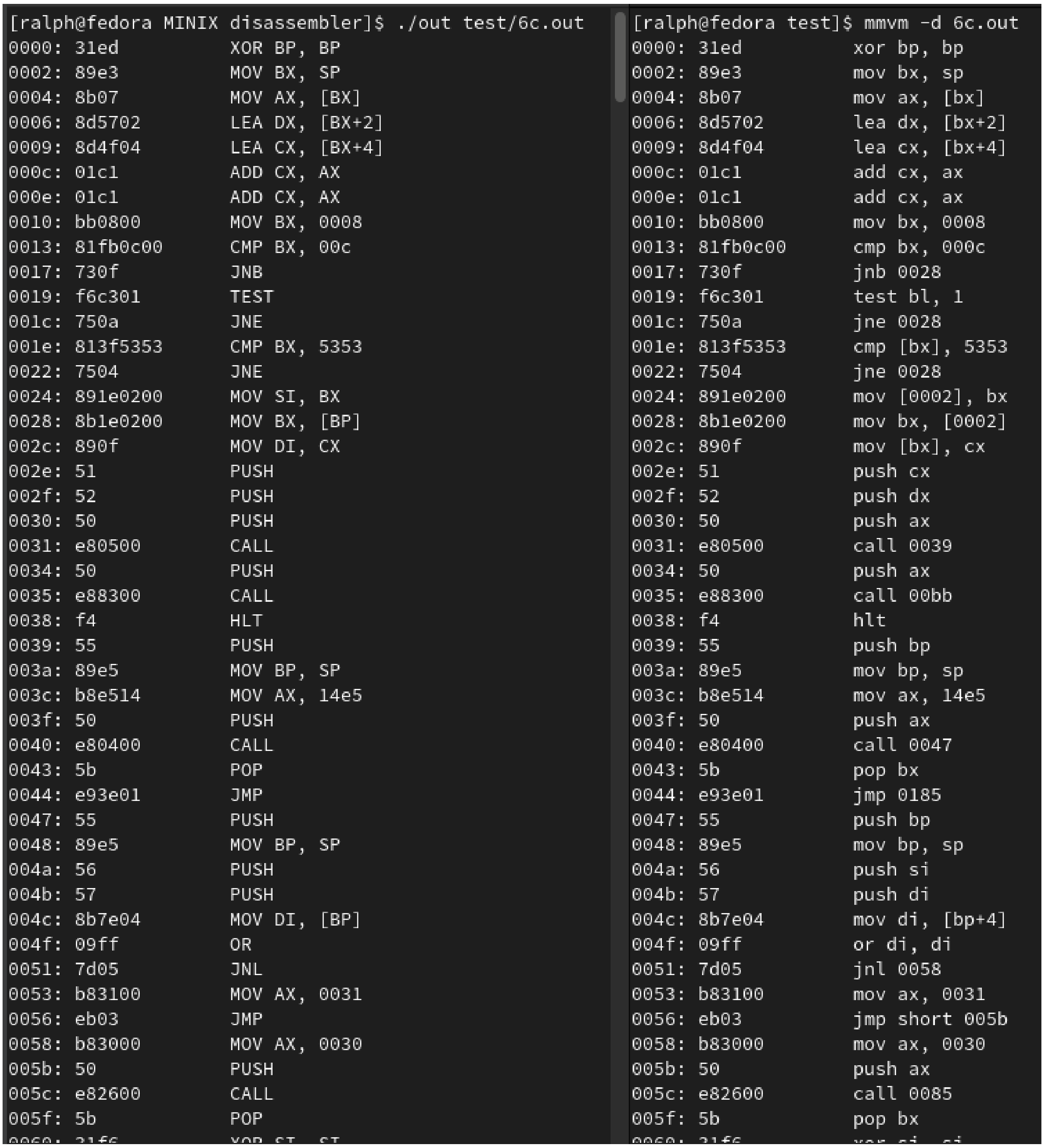
5.c



```
[ralph@fedora MINIX disassembler]$ ./out test/5c.out
0000: 31ed          XOR BP, BP
0002: 89e3          MOV BX, SP
0004: 8b07          MOV AX, [BX]
0006: 8d5702        LEA DX, [BX+2]
0009: 8d4f04        LEA CX, [BX+4]
000c: 01c1          ADD CX, AX
000e: 01c1          ADD CX, AX
0010: bb1600        MOV BX, 0016
0013: 81fb1a02      CMP BX, 021a
0017: 730f          JNB
0019: f6c301        TEST
001c: 750a          JNE
001e: 813f5353      CMP BX, 5353
0022: 7504          JNE
0024: 891e0200      MOV SI, BX
0028: 8b1e0200      MOV BX, [BP]
002c: 890f          MOV DI, CX
002e: 51            PUSH
002f: 52            PUSH
0030: 50            PUSH
0031: e80500        CALL
0034: 50            PUSH
0035: e84000        CALL
0038: f4            HLT
0039: 55            PUSH
003a: 89e5          MOV BP, SP
003c: 56            PUSH
003d: 31f6          XOR SI, SI
0040: 7604          JBE
0042: 7e17          JLE
0044: 89f3          MOV BX, SI
0046: d1e3          SHL
0048: 035e06        ADD BX, SI
004b: ff3756        PUSH
004e: b80400        MOV AX, 0004
0051: 50            PUSH
0052: e84000        CALL
0055: 83c40646      ADD
0059: ebe4          JMP
005b: e95511        JMP
005e: 55            PUSH
005f: 89e5          MOV BP, SP
0061: 56            PUSH
0062: 8b361200      MOV SI, [BP]
0066: 4e            DEC
```

```
[ralph@fedora test]$ mmvm -d 5c.out
0000: 31ed          xor bp, bp
0002: 89e3          mov bx, sp
0004: 8b07          mov ax, [bx]
0006: 8d5702        lea dx, [bx+2]
0009: 8d4f04        lea cx, [bx+4]
000c: 01c1          add cx, ax
000e: 01c1          add cx, ax
0010: bb1600        mov bx, 0016
0013: 81fb1a02      cmp bx, 021a
0017: 730f          jnb 0028
0019: f6c301        test bl, 1
001c: 750a          jne 0028
001e: 813f5353      cmp [bx], 5353
0022: 7504          jne 0028
0024: 891e0200      mov [0002], bx
0028: 8b1e0200      mov bx, [0002]
002c: 890f          mov [bx], cx
002e: 51            push cx
002f: 52            push dx
0030: 50            push ax
0031: e80500        call 0039
0034: 50            push ax
0035: e84000        call 0078
0038: f4            hlt
0039: 55            push bp
003a: 89e5          mov bp, sp
003c: 56            push si
003d: 31f6          xor si, si
003f: 397604        cmp [bp+4], si
0042: 7e17          jle 005b
0044: 89f3          mov bx, si
0046: d1e3          shl bx, 1
0048: 035e06        add bx, [bp+6]
004b: ff37          push [bx]
004d: 56            push si
004e: b80400        mov ax, 0004
0051: 50            push ax
0052: e84000        call 0095
0055: 83c406        add sp, 6
0058: 46            inc si
0059: ebe4          jmp short 003f
005b: e95511        jmp 11b3
005e: 55            push bp
005f: 89e5          mov bp, sp
0061: 56            push si
```

6.c



```
[ralph@fedora MINIX disassembler]$ ./out test/6c.out
0000: 31ed          XOR BP, BP
0002: 89e3          MOV BX, SP
0004: 8b07          MOV AX, [BX]
0006: 8d5702        LEA DX, [BX+2]
0009: 8d4f04        LEA CX, [BX+4]
000c: 01c1          ADD CX, AX
000e: 01c1          ADD CX, AX
0010: bb0800        MOV BX, 0008
0013: 81fb0c00      CMP BX, 00c
0017: 730f          JNB
0019: f6c301        TEST
001c: 750a          JNE
001e: 813f5353      CMP BX, 5353
0022: 7504          JNE
0024: 891e0200      MOV SI, BX
0028: 8b1e0200      MOV BX, [BP]
002c: 890f          MOV DI, CX
002e: 51            PUSH
002f: 52            PUSH
0030: 50            PUSH
0031: e80500        CALL
0034: 50            PUSH
0035: e88300        CALL
0038: f4            HLT
0039: 55            PUSH
003a: 89e5          MOV BP, SP
003c: b8e514        MOV AX, 14e5
003f: 50            PUSH
0040: e80400        CALL
0043: 5b            POP
0044: e93e01        JMP
0047: 55            PUSH
0048: 89e5          MOV BP, SP
004a: 56            PUSH
004b: 57            PUSH
004c: 8b7e04        MOV DI, [BP]
004f: 09ff          OR
0051: 7d05          JNL
0053: b83100        MOV AX, 0031
0056: eb03          JMP
0058: b83000        MOV AX, 0030
005b: 50            PUSH
005c: e82600        CALL
005f: 5b            POP
0060: 31f6          XOR SI, SI
```

```
[ralph@fedora test]$ mmvm -d 6c.out
0000: 31ed          xor bp, bp
0002: 89e3          mov bx, sp
0004: 8b07          mov ax, [bx]
0006: 8d5702        lea dx, [bx+2]
0009: 8d4f04        lea cx, [bx+4]
000c: 01c1          add cx, ax
000e: 01c1          add cx, ax
0010: bb0800        mov bx, 0008
0013: 81fb0c00      cmp bx, 000c
0017: 730f          jnb 0028
0019: f6c301        test bl, 1
001c: 750a          jne 0028
001e: 813f5353      cmp [bx], 5353
0022: 7504          jne 0028
0024: 891e0200      mov [0002], bx
0028: 8b1e0200      mov bx, [0002]
002c: 890f          mov [bx], cx
002e: 51            push cx
002f: 52            push dx
0030: 50            push ax
0031: e80500        call 0039
0034: 50            push ax
0035: e88300        call 00bb
0038: f4            hlt
0039: 55            push bp
003a: 89e5          mov bp, sp
003c: b8e514        mov ax, 14e5
003f: 50            push ax
0040: e80400        call 0047
0043: 5b            pop bx
0044: e93e01        jmp 0185
0047: 55            push bp
0048: 89e5          mov bp, sp
004a: 56            push si
004b: 57            push di
004c: 8b7e04        mov di, [bp+4]
004f: 09ff          or di, di
0051: 7d05          jnl 0058
0053: b83100        mov ax, 0031
0056: eb03          jmp short 005b
0058: b83000        mov ax, 0030
005b: 50            push ax
005c: e82600        call 0085
005f: 5b            pop bx
0060: 31f6          xor si, si
```