Brian Perel, Jon Petani

Operating Systems Internals – HW2

Assembly Code, Symbol Table, Machine Code

**Total of 4 programs are included:**

1. Null Program
2. Program #1: Dynamic memory allocation and freeing
3. Program #2: Using stack
4. Program #3: Performing input and output operations

**Null Program:**

**Assembly code with comments:**

*Label:* *Mnemonic:* *Operands:* *Comment:*

Main Function Null function start

Start 0 Function starts at address 0

loop Branch loop Branch to loop

Halt Halt execution

End loop End of program

**Symbol Table:**

|  |  |
| --- | --- |
| **Symbol** | **Address** |
| main | 0 |
| loop | 0 |

**Machine code:**

***Address:* *Content:***

0 60000 // Branch has no operands (0), address of loop. Branch Opcode is 6

1 0 // Address to branch, value 0

2 0 // halt is 0, opcode = 0

-1 0 // pc value is set to 0

**User Program #1:**

**Assembly code with comments:**

*Label:* *Mnemonic:* *Operands:* *Comment:*

Main Function Main function start

Size long 150 Declare variable ‘size’ of type long and set value to 150

Move r2, Size Move size into register 2

System call 4 make system call, operand is 4

(memory allocation function call)

R long 5 Declare variable ‘R’ of type long and set to random value

Move r3, R Move R into register 3

M long 2 Declare variable ‘M’ of type long and set to value 2

Move r4, M Move M into register 4

Counter long 10 Declare variable counter for the loop conditional

Move r5, Counter Move Counter into register 5

Loop Multiply r3, r4 Multiply r3 (R) by r4 (M)

Add r4, 1 Increment r4 (M) by 1

Subtract r5, 1 Decrement r5 (Counter) by 1

Branch on plus Counter, loop Set conditional to increment loop

Halt Stop the program

Move R, r1 move R into register 1

System call 5 make system call, operand is 5

End Main Terminate the program

**Symbol Table:**

|  |  |
| --- | --- |
| **Symbol** | **Address** |
| Main | 0 |
| R | 5 |
| M | 2 |
| Counter | 10 |
| Loop | 0 |

**Machine Code:**

***Address:* *Content:***

3 51260 // Declare variable size

4 150 // Assign value to variable size

5 126000 // Memory allocation system call

6 4 // System call ID for memory allocation

7 51360 // Declare variable R

8 5 // Assign value to variable R

9 51460 // Declare variable M

10 2 // Assign value to variable M

11 51560 // Declare variable counter

12 10 // Assign value to variable counter

13 31314 // Multiply R (register 3) by M (register 4)

14 11434 // Increment M (register 4)

15 21545 // Decrement counter (register 5)

16 81500 // Check if counter is 0, if not go back into loop

17 0 // Halt

18 51361 // Move result of R into register 1

19 12600 // Memory free system call

20 5 // System call ID for memory freeing

-1 0 // End program

**User Program #2:**

**Assembly code with comments:**

*Label:* *Mnemonic:* *Operands:* *Comment:*

Main Function Main function start

M long 11 Declare variable ‘M’ of size 11

Move r1, M Move value of variable M into register 1

V1 long 4321 Declare variable ‘v1’ of size 4321

Move r2, V1 Move value of variable V1 into register 2

V2 long 3142 Declare variable ‘v2’ of size 3142

Move r3, V2 Move value of variable V2 into register 3

V3 long 1111 Declare variable ‘v3’ of size 1111

Move r4, V3 Move value of variable V3 into register 4

V4 long 2233 Declare variable ‘v4’ of size 2233

Move r5, V4 Move value of variable V4 into register 5

loop Push V1 Doing a push operation on V1

Push V2 Doing a push operation on V2

Push V3 Doing a push operation on V3

Push V4 Doing a push operation on V4

Pop V1 Doing a pop operation on V1

Pop V2 Doing a pop operation on V2

Pop V3 Doing a pop operation on V3

Pop V4 Doing a pop operation on V4

Subtract r1, 1 Decrement register 1

BrOnPlus M, loop If M is 0 terminate the loop

Halt Stop the program

End Main Program terminated

**Symbol Table:**

|  |  |
| --- | --- |
| **Symbol** | **Address** |
| Main | 0 |
| Loop | 0 |
| M | 11 |
| V1 | 4321 |
| V2 | 3142 |
| V3 | 1111 |
| V4 | 2233 |

**Machine Code:**

**Address: Content:**  
21 51260 // Declare variable ‘M’

22 11 // Assign value to variable ‘M’

23 51360 // Declare variable ‘V1’

24 4321 // Assign value to variable ‘V1’

25 51460 // Declare variable ‘V2’

26 3142 // Assign value to variable `V2`

27 51560 // Declare variable `V3`

28 1111 // Assign value to variable `V3`

29 51660 // Declare variable `V4`

30 2233 // Assign value to variable `V4`

31 106000 // Push variable `V1` onto the stack

32 4321 // Immediate value of `V1`

33 106000 // Push variable `V2` onto the stack

34 3142 // Immediate value of `V2`

35 106000 // Push variable `V3` onto the stack

36 1111 // Immediate value of `V3`

37 106000 // Push variable `V4` onto the stack

38 2233 // Immediate value of `V4`

39 115000 // Pop variable from stack

40 2700 // Variable is stored into memory location

41 115000 // Pop variable from stack

42 2700 // Variable is stored into memory location

43 115000 // Pop variable from stack

44 2700 // Variable is stored into memory location

45 115000 // Pop variable from stack

46 2700 // Variable is stored into memory location

47 21545 // Decrement variable `M`

48 81500 // Check if Counter is 0, if not go back into loop

49 0 // Halt

-1 0 // Get PC

**User Program #3:**

5-letter word: Yacht

**Assembly code with comments:**

*Label:* *Mnemonic:* *Operands:* *Comment:*

Main Function Main function start

Counter long 5 Declare variable Counter of size 5 for 5-letter word

Move r1, Counter Move variable Counter into register 1

L1 long 89 Declare variable L1 (for first letter of word) of size 89

Move r2, L1 Move variable L1 into register 2

L2 long 97 Declare variable L2 (for second letter of word) of size 97

Move r3, L2 Move variable L2 into register 3

L3 long 99 Declare variable L3 (for third letter of word) of size 99

Move r4, L3 Move variable L3 into register 4

L4 long 104 Declare variable L4 (for fourth letter of word) of 104

Move r5, L4 Move variable L4 into register 5

L5 long 116 Declare variable L5 (for fifth letter of word) of 116

Move r6, L5 Move variable L5 into register 6

System call 4 Call memAllocSystemCall()

Loop System call 8 Call io\_getcSystemCall()

Subtract r1, 1 Decrement register 1

Move r1, GPR7 Move register 1 into GPR7

BrOnPlus r1, loop If counter is 0, move onto loop 2

Counter2 long 5 Declare variable counter 2 of size 5

Move r1, Counter2 Move variable counter2 into register 1

Loop2 System call 9 Call io\_putcSystemCall()

Subtract r1, 1 Decrement register 1

Move r5, r0 Put IO content into register 0

BrOnPlus r1, loop2 If counter is 0, move to termination

System call 5 Call memFreeSystemCall()

Halt 0 Halt program

End Main End program

**Symbol Table:**

|  |  |
| --- | --- |
| **Symbol:** | **Address:** |
| Main | 0 |
| Counter | 5 |
| L1 | 89 |
| L2 | 97 |
| L3 | 99 |
| L4 | 104 |
| L5 | 116 |
| Counter2 | 5 |
| Loop | 0 |
| Loop2 | 0 |

**Machine Code:**

**Address: Content:**  
50 51160 // Declare variable Counter

51 5 // Assign value to Counter

52 51260 // Declare variable L1

53 89 // Assign value to L1

54 51360 // Declare variable L2

55 97 // Assign value to L2

56 51460 // Declare variable L3

57 99 // Assign value to L3

58 51560 // Declare variable L4

59 104 // Assign value to variable L4

60 51660 // Declare variable L5

61 116 // Assign value to variable L5

62 12600 // system call ID for memory allocation

63 4 // system call ID for system call memory allocation

64 12600 // system call for io\_getcSystemCall()

65 8 // system call ID for system get

66 21145 // Decrement register 1

67 51127 // Move into register 1

68 81500 // Check if Counter is 0, if not go back into loop

69 12600 // system call ID for io\_putcSystemCall()

70 9 // system call ID for system put

71 21145 // Decrement register 1

72 51127 // Move into register 1

73 81500 // Check if Counter2 is 0, if not go back into loop

74 12600 // system call ID for memFreeSystemCall()

75 5 // system call ID for memory free

76 0 // Halt

-1 31 // PC is 31