

COMP 235 — Assembly Study

You've been given a short C program along with the assembly generated from that file. Analyze the assembly and determine the answers to the following:

1. For *main*

- (a) attribute all lines of assembly to lines of C or to function call prologue and epilogue code done at the assembly level. Label assembly line numbers with C line numbers.
- (b) determine the size of the stack frame in bytes
- (c) determine the location within the frame for each of the 3 arrays. List the locations relative to `%rbp`. For example, `[-10, -5]` would refer to bytes `%rbp-10` through `%rbp-5`.
- (d) draw a diagram showing how each byte of the stack frame is used. If bytes are allocated on the stack but not directly used, label them as unused.
- (e) determine the primary operand specifier for the variables A, B, and C.

2. For *initAll*

- (a) attribute all lines of assembly to lines of C or to function call prologue and epilogue code done at the assembly level. Label assembly line numbers with C line numbers.
- (b) determine how much stack space is used
- (c) determine the primary operand specifier for variables a,b,c,i, and j.

3. For *matmul*

- (a) attribute all lines of assembly to line of C or to function call prologue and epilogue code done at the assembly level. Label assembly line numbers with C line numbers.
- (b) determine how much stack space is used
- (c) determine the primary operand specifier for variables a,b,c,i,j, and k

We'll spend some time in class getting started on this. This exercise is open book, open notes, open professor, open classmates, but not open internet. Do not consult with AI.

matmul.c

```
1  #include <stdio.h>
2
3  void mult(int lhs[][4],int rhs[][4],int prod[][4]){
4      int i,j,k;
5      int sum;
6      for(i=0;i<4;i++){
7          for(j=0;j<4;j++){
8              for(k=0;k<4 ;k++){
9                  prod[i][j] += lhs[i][k]*rhs[k][j];
10             }
11         }
12     }
13 }
14
15 void printMat(int m[][4]){
16     int i,j;
17     for(i=0;i<4;i++){
18         for(j=0;j<4;j++){
19             printf("%3d ",m[i][j]);
20         }
21         printf("\n");
22     }
23 }
24
25 void initAll(int a[][4], int b[][4], int c[][4]){
26     int i,j;
27     for(i = 0; i < 4; i++){
28         for(j=0; j<4; j++ ){
29             c[i][j] = 0;
30             a[i][j] = 4*i + j;
31             b[i][j] = 16-(4*i+j);
32         }
33     }
34 }
35
36 int main(){
37     int A[4][4];
38     int B[4][4];
39     int C[4][4];
40
41     initAll(A,B,C);
42
43     printMat(A);
44     printMat(B);
45     printMat(C);
46
47     mult(A,B,C);
48     printMat(C);
49
50 }
```


matmul.s

```
1      .file   "matmul.c"
2      .text
3      .globl  mult
4      .type   mult, @function
5  mult:
6      endbr64
7      pushq   %rbp
8      movq    %rsp, %rbp
9      movq    %rdi, -24(%rbp)
10     movq    %rsi, -32(%rbp)
11     movq    %rdx, -40(%rbp)
12     movl    $0, -12(%rbp)
13     jmp     .L2
14  .L7:
15     movl    $0, -8(%rbp)
16     jmp     .L3
17  .L6:
18     movl    $0, -4(%rbp)
19     jmp     .L4
20  .L5:
21     movl    -12(%rbp), %eax
22     cltq
23     salq    $4, %rax
24     movq    %rax, %rdx
25     movq    -40(%rbp), %rax
26     addq    %rax, %rdx
27     movl    -8(%rbp), %eax
28     cltq
29     movl    (%rdx,%rax,4), %esi
30     movl    -12(%rbp), %eax
31     cltq
32     salq    $4, %rax
33     movq    %rax, %rdx
34     movq    -24(%rbp), %rax
35     addq    %rax, %rdx
36     movl    -4(%rbp), %eax
37     cltq
38     movl    (%rdx,%rax,4), %edx
39     movl    -4(%rbp), %eax
40     cltq
41     salq    $4, %rax
42     movq    %rax, %rcx
43     movq    -32(%rbp), %rax
44     addq    %rax, %rcx
45     movl    -8(%rbp), %eax
46     cltq
47     movl    (%rcx,%rax,4), %eax
48     movl    %edx, %ecx
49     imull   %eax, %ecx
50     movl    -12(%rbp), %eax
```

```

51     cltq
52     salq    $4, %rax
53     movq    %rax, %rdx
54     movq    -40(%rbp), %rax
55     addq    %rax, %rdx
56     addl    %esi, %ecx
57     movl    -8(%rbp), %eax
58     cltq
59     movl    %ecx, (%rdx,%rax,4)
60     addl    $1, -4(%rbp)
61 .L4:
62     cmpl    $3, -4(%rbp)
63     jle .L5
64     addl    $1, -8(%rbp)
65 .L3:
66     cmpl    $3, -8(%rbp)
67     jle .L6
68     addl    $1, -12(%rbp)
69 .L2:
70     cmpl    $3, -12(%rbp)
71     jle .L7
72     nop
73     nop
74     popq    %rbp
75     ret
76     .size   mult, .-mult
77     .section .rodata
78 .LC0:
79     .string "%3d "
80     .text
81     .globl  printMat
82     .type   printMat, @function
83 printMat:
84     endbr64
85     pushq   %rbp
86     movq    %rsp, %rbp
87     subq    $32, %rsp
88     movq    %rdi, -24(%rbp)
89     movl    $0, -8(%rbp)
90     jmp     .L9
91 .L12:
92     movl    $0, -4(%rbp)
93     jmp     .L10
94 .L11:
95     movl    -8(%rbp), %eax
96     cltq
97     salq    $4, %rax
98     movq    %rax, %rdx
99     movq    -24(%rbp), %rax
100    addq    %rax, %rdx
101    movl    -4(%rbp), %eax
102    cltq

```

```

103     movl    (%rdx,%rax,4), %eax
104     movl    %eax, %esi
105     leaq    .LC0(%rip), %rax
106     movq    %rax, %rdi
107     movl    $0, %eax
108     call    printf@PLT
109     addl    $1, -4(%rbp)
110 .L10:
111     cmpl    $3, -4(%rbp)
112     jle     .L11
113     movl    $10, %edi
114     call    putchar@PLT
115     addl    $1, -8(%rbp)
116 .L9:
117     cmpl    $3, -8(%rbp)
118     jle     .L12
119     nop
120     nop
121     leave
122     ret
123     .size   printMat, .-printMat
124     .globl  initAll
125     .type   initAll, @function
126 initAll:
127     endbr64
128     pushq   %rbp
129     movq    %rsp, %rbp
130     movq    %rdi, -24(%rbp)
131     movq    %rsi, -32(%rbp)
132     movq    %rdx, -40(%rbp)
133     movl    $0, -8(%rbp)
134     jmp     .L14
135 .L17:
136     movl    $0, -4(%rbp)
137     jmp     .L15
138 .L16:
139     movl    -8(%rbp), %eax
140     cltq
141     salq    $4, %rax
142     movq    %rax, %rdx
143     movq    -40(%rbp), %rax
144     addq    %rax, %rdx
145     movl    -4(%rbp), %eax
146     cltq
147     movl    $0, (%rdx,%rax,4)
148     movl    -8(%rbp), %eax
149     leal    0(,%rax,4), %ecx
150     movl    -8(%rbp), %eax
151     cltq
152     salq    $4, %rax
153     movq    %rax, %rdx
154     movq    -24(%rbp), %rax

```

```

155     addq    %rax, %rdx
156     movl    -4(%rbp), %eax
157     addl    %eax, %ecx
158     movl    -4(%rbp), %eax
159     cltq
160     movl    %ecx, (%rdx,%rax,4)
161     movl    -8(%rbp), %eax
162     leal    0(,%rax,4), %edx
163     movl    -4(%rbp), %eax
164     leal    (%rdx,%rax), %esi
165     movl    -8(%rbp), %eax
166     cltq
167     salq    $4, %rax
168     movq    %rax, %rdx
169     movq    -32(%rbp), %rax
170     addq    %rax, %rdx
171     movl    $16, %eax
172     subl    %esi, %eax
173     movl    %eax, %ecx
174     movl    -4(%rbp), %eax
175     cltq
176     movl    %ecx, (%rdx,%rax,4)
177     addl    $1, -4(%rbp)
178 .L15:
179     cmpl    $3, -4(%rbp)
180     jle .L16
181     addl    $1, -8(%rbp)
182 .L14:
183     cmpl    $3, -8(%rbp)
184     jle .L17
185     nop
186     nop
187     popq    %rbp
188     ret
189     .size   initAll, .-initAll
190     .globl  main
191     .type   main, @function
192 main:
193     endbr64
194     pushq   %rbp
195     movq    %rsp, %rbp
196     subq    $208, %rsp
197     movq    %fs:40, %rax
198     movq    %rax, -8(%rbp)
199     xorl    %eax, %eax
200     leaq    -80(%rbp), %rdx
201     leaq    -144(%rbp), %rcx
202     leaq    -208(%rbp), %rax
203     movq    %rcx, %rsi
204     movq    %rax, %rdi
205     call    initAll
206     leaq    -208(%rbp), %rax

```

```

207     movq    %rax, %rdi
208     call   printMat
209     leaq   -144(%rbp), %rax
210     movq    %rax, %rdi
211     call   printMat
212     leaq   -80(%rbp), %rax
213     movq    %rax, %rdi
214     call   printMat
215     leaq   -80(%rbp), %rdx
216     leaq   -144(%rbp), %rcx
217     leaq   -208(%rbp), %rax
218     movq    %rcx, %rsi
219     movq    %rax, %rdi
220     call   mult
221     leaq   -80(%rbp), %rax
222     movq    %rax, %rdi
223     call   printMat
224     movl    $0, %eax
225     movq    -8(%rbp), %rdx
226     subq    %fs:40, %rdx
227     je     .L20
228     call   __stack_chk_fail@PLT
229 .L20:
230     leave
231     ret
232     .size   main, .-main
233     .ident  "GCC: (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0"
234     .section .note.GNU-stack,"",@progbits
235     .section .note.gnu.property,"a"
236     .align  8
237     .long   1f - 0f
238     .long   4f - 1f
239     .long   5
240 0:
241     .string "GNU"
242 1:
243     .align  8
244     .long   0xc0000002
245     .long   3f - 2f
246 2:
247     .long   0x3
248 3:
249     .align  8
250 4:
251

```