Source File: `lab34.asm`
Input: Standard Input
Output: Standard Output
Value: 2

Write a procedure that will multiply two unsigned 32-bit integers using only shifting and addition. The product should be returned in the `EAX` register. The product of two 32-bit integers could be as large as 64 bits. For this assignment, we will assume that the product is never larger than 32 bits. A description of the function as well as client code for testing your implementation is shown in Figure 1, and a sample execution sequence is shown in Figure 2. To use the `Makefile` as distributed in class, add a target of `lab34` to `targets2AsmFiles`.

```
1  [list -]
2  %INCLUDE "Along32.inc"
3  %INCLUDE "Macros_Along.inc"
4  [list +]
5
6  ;-----------------------------------------------------------------------
7  extern  BitwiseMultiply
8  ; HLL prototype:
9  ;   uint BitwiseMultiply(uint *multiplicand, uint *multiplier);
10 ; Receives pointers to two unsigned 32-bit integers.
11 ; Determines the product of the two integers using the bitwise
12 ; multiplication method. Returns the product in the EAX register.
13 ; Receives: EAX = address of multiplicand
14 ;           EBX = address of multiplier
15 ; Returns:  EAX = product of multiplicand and multiplier
16 ;-----------------------------------------------------------------------
17
18 ;-----------------------------------------------------------------------
19 global  PrintInt:function
20 ; HLL prototype:
21 ;   void PrintInt(uint number);
22 ; Prints number in a right-justified field of width 10
23 ; Receives: EAX = number
24 ; Returns:  Nothing
25 ;-----------------------------------------------------------------------
26
27 SECTION .data
28 hrule   times   44 db ('-')
29         db      10,0
30 spacer1 times   2 db ' '
31         db      0
32 spacer2 times   4 db ' '
33         db      0
34 title   times   11 db ' '
35         db      'BITWISE MULTIPLICATION',10,0
36 header  times   2 db ' '
37         db      'Multiplicand'
```

Figure 1. `/usr/local/3304/src/lab34main.asm` (Part 1 of 4)

```
38          times   4 db ' '
39          db      'Multiplier'
40          times   4 db ' '
41          db      '  Product',10,0
42
43  SECTION .bss
44  h       resd    1
45  multiplicand    resd    1
46  multiplier      resd    1
47
48  SECTION .text
49          global  _start
50  _start:
51          finit                           ; initialize floating-point unit
52
53          call    ReadDec                 ; read an unsigned integer
54          mov     [h],eax                 ; move the integer to h
55
56          mov     edx,title               ; write title
57          call    WriteString
58          mov     edx,hrule               ; write hrule
59          call    WriteString
60          mov     edx,header              ; write headings
61          call    WriteString
62          mov     edx,hrule               ; write hrule
63          call    WriteString
64
65  .L0:
66          cmp     dword [h],0             ; while h > 0 do
67          jle     .L1
68
69          mov     edx,spacer1
70          call    WriteString
71          call    ReadDec                 ;   read an unsigned int
72          mov     [multiplicand],eax      ;   store in multiplicand
73          mov     al,' '
74          call    WriteChar
75          mov     eax,[multiplicand]
76          call    PrintInt                ;   write multiplicand to stdout
77          mov     al,' '
78          call    WriteChar
79
80          mov     edx,spacer2
81          call    WriteString
82          call    ReadDec                 ;   read an unsigned int
83          mov     [multiplier],eax        ;   store in multiplier
84          call    PrintInt                ;   write multiplier to stdout
85          mov     edx,spacer2
86          call    WriteString
```

Figure 1. `/usr/local/3304/src/lab34main.asm` (Part 2 of 4)

```
87
88          mov     eax,multiplicand        ;   &multiplicand in eax
89          mov     ebx,multiplier          ;   &multiplier in ebx
90          call    BitwiseMultiply
91          call    PrintInt                ;   write product to stdout
92
93          mov     al,'\n'
94          call    WriteChar
95
96          dec     dword [h]               ;   decrement h
97          jmp     .L0                     ; end while
98  .L1:
99          mov     edx,hrule               ; write hrule
100         call    WriteString
101
102         Exit    {0}
103
104 PrintInt:
105 SECTION .bss
106 .width  resd    1                       ; make width local with .
107 .number resd    1                       ; make number local with .
108
109 SECTION .text
110         pushad                          ; save all 32-bit GP registers
111
112         mov     [PrintInt.number],eax
113         mov     dword [PrintInt.width],0; initialize width at 0
114 ;
115 ; if (number == 0)
116 ;   width = 0
117 ; else
118 ;   width = log_10(number) = log_2(number) / log_2(10)
119 ;
120         cmp     dword[PrintInt.number],0
121         je      .L0
122
123         fld1                            ; st(0) = 1.0 (load constant 1.0)
124         fild    dword [PrintInt.number] ; st(0) = number; st(1) = 1.0
125         fyl2x                           ; computes st(0) = st(1) * log_2(st(0))
126         fldl2t                          ; st(0) = log_2(10.0)
127                                         ; st(1) = log_2(multiplicand)
128         fdiv                            ; st(1) = st(1) / st(0)
129                                         ; pop stack
130         fisttp  dword [PrintInt.width]  ; store truncated integer and pop
131 .L0:
132         inc     dword [PrintInt.width]  ; add one to width
```

Figure 1. `/usr/local/3304/src/lab34main.asm` (Part 3 of 4)

```
133 ;
134 ; insert enough spaces to eventually right justify number in a field of
135 ; width 10
136 ;
137         mov     ecx,10
138         sub     ecx,[PrintInt.width]
139 .L1:
140         cmp     ecx,0
141         jle     .L2
142         mov     al,' '
143         call    WriteChar
144         dec     ecx
145         jmp     .L1
146 .L2:
147         mov     eax,[PrintInt.number]
148         call    WriteDec
149
150         popad                           ; restore all 32-bit GP registers
151         ret
```

Figure 1. `/usr/local/3304/src/lab34main.asm` (Part 4 of 4)

```
newuser@csunix ~/3304/34> cp /usr/local/3304/data/34/* .
newuser@csunix ~/3304/34> cp /usr/local/3304/src/Makefile .
newuser@csunix ~/3304/34> cp /usr/local/3304/src/lab34main.asm .
newuser@csunix ~/3304/34> touch lab34.asm
newuser@csunix ~/3304/34> make lab34
nasm -f elf32  -l lab34main.lst -o lab34main.o lab34main.asm -I/usr/local/3304/include/ -I.
nasm -f elf32  -l lab34.lst -o lab34.o lab34.asm -I/usr/local/3304/include/ -I.
ld -m elf_i386 --dynamic-linker /lib/ld-linux.so.2 -o lab34 lab34main.o lab34.o \
/usr/local/3304/src/Along32.o -lc
newuser@csunix ~/3304/34> ../irvine_test.sh lab34 01.dat
           BITWISE MULTIPLICATION
-----------------------------------------
  Multiplicand    Multiplier      Product
-----------------------------------------
             1     2147483647    2147483647
    2147483647              1    2147483647
            11           3304         36344
          3304             11         36344
             0     2147483647             0
    2147483647              0             0
             1             12            12
           123           1234        151782
         12345         123456    1524064320
       1234567           1234    1523455678
      12345678            123    1518518394
     123456789             12    1481481468
             0              0             0
          3304           3304      10916416
         46340          46340    2147395600
             1              1             1
             4              4            16
            10             10           100
            32             32          1024
           100            100         10000
           317            317        100489
          1000           1000       1000000
          3163           3163      10004569
         10000          10000     100000000
         31623          31623    1000014129
-----------------------------------------
newuser@csunix ~/3304/34> ../irvine_test.sh lab34 01.dat > my.out
newuser@csunix ~/3304/34> diff 01.out my.out
newuser@csunix ~/3304/34>
```

Figure 2. Commands to Assemble, Link, & Run Lab 34