! CPSC 355 ASSIGNMENT 2B: Multiplication Using Add and Shift Operations

! =====================================================================

! Completed on 10-11-2013 by HUNG, Michael (UCID: 10099049)

!

!

! This program implements the following algorithm:

!

! negative = multiplier >= 0 ? 0 : 1;

! product = 0;

! for (i = 0; i < 32; i++) {

! if (multiplier & 1)

! product += multiplicand;

! (product and multiplier registers combined as a unit) >> 1;

! }

! if (negative)

! product -= multiplicand;

!

! There will be three uses of the algorithm for these multipliers and multiplicands:

! 1.

! multiplier = 120490892

! multiplicand = 82732983

!

! 2.

! multiplier = -120490892

! multiplicand = 82732983

!

! 3.

! multiplier = -120490892

! multiplicand = -82732983

output: .asciz "Product: %x %x Multiplier: %x Multiplicand: %x\n" ! Output string

.align 4

.global main

main: save %sp, -96, %sp

mov 0, %l3 ! flag for determining where to move

! (negpos or negneg)

mov 0, %l4 ! flag for determining where to move

! (negpos or negneg)

pospos: set output, %o0

clr %l0 ! Product Register

clr %l1 ! Multiplier Register

clr %l2 ! Multiplicand Register

clr %i0 ! Initialize Counter

clr %i1 ! 0 constant

clr %i2 ! Negative flag

set 120490892, %l1 ! multiplier

set 120490892, %o3 ! multiplier for output string

set 82732983, %l2 ! multiplicand

set 82732983, %o4 ! multiplicand for output string

ba ntest

cmp %i1, %l1 ! test for if multiplier is negative;

! filled delay slot

negpos: set output, %o0

clr %l0 ! Product Register

clr %l1 ! Multiplier Register

clr %l2 ! Multiplicand Register

clr %i0 ! Initialize Counter

clr %i1 ! 0 constant

clr %i2 ! negative flag

set -120490892, %l1 ! multiplier

set -120490892, %o3 ! multiplier for output string

set 82732983, %l2 ! multiplicand

set 82732983, %o4 ! multiplicand for output string

ba ntest

cmp %i1, %l1 ! test for if multiplier is negative;

! filled delay slot

negneg: set output, %o0

clr %l0 ! Product Register

clr %l1 ! Multiplier Register

clr %l2 ! Multiplicand Register

clr %i0 ! Initialize Counter

clr %i1 ! 0 constant

clr %i2 ! negative flag

set -120490892, %l1 ! multiplier

set -120490892, %o3 ! multiplier for output string

set -82732983, %l2 ! multiplicand

set -82732983, %o4 ! multiplicand for output string

ba ntest

cmp %i1, %l1 ! test for if multiplier is negative;

! filled delay slot

ntest: bg,a negative

mov 1, %i2 ! Multiplier is negative. Set flag

! to 1; filled delay slot

ba continue ! Not negative

inc %i0 ! Increment counter; filled delay slot

negative: ba continue ! Flag set. Continue

inc %i0 ! Increment counter; filled delay slot

continue: andcc %l1, 0x1, %g0 ! Are the b0 both 1? If so, add the

! multiplier to the product in

! multiply branch

bg,a multiply

add %l0, %l2, %l0 ! b0 are both 1. Add multiplicand to

! product; filled delay slot

ba shift ! The bits are not both 1. Shift

! product and multiplier to right

and %l0, 0x1, %i3 ! grab b0 of product; filled delay slot

multiply: ba shift ! Go to shift branch

and %l0, 0x1, %i3 ! grab b0 of product; filled delay slot

shift: sll %i3, 31, %i3 ! make b0 b31

sra %l0, 1, %l0 ! Shift product to right by one

srl %l1, 1, %l1 ! Shift multiplier to right by one

add %i3, %l1, %l1 ! put b0 of product as b31 of multiplier

cmp %i0, 32 ! Has counter reached 32? If so, move on

bge,a ifneg

andcc %i2, 0x1, %g0 ! Was the multiplier negative? If it

! was, go to submult branch; filled

! delay slot

ba continue ! Loop is not finished

inc %i0 ! Increment counter; filled delay slot

ifneg: bg,a submult

sub %l0, %l2, %l0 ! subtract multiplicand from product;

! filled delay slot

ba outreg ! Multiplier was positive. Go to

! print output

and %l0, %l0, %o1 ! place product into output register;

! filled delay slot

submult: ba outreg ! go to print output

and %l0, %l0, %o1 ! place product into output register;

! filled delay slot

outreg: and %l1, %l1, %o2 ! place updated multiplier into

! output register

cmp %l3, 0 ! pospos is done. Print the product.

be,a printpp

nop

cmp %l4, 0 ! negpos is done. Print the product.

be,a printnp

nop

ba printnn ! negneg is done. Print the product.

nop

printpp: call printf ! Printing pospos product.

nop

ba negpos ! Move to next multiplication, negpos.

mov 1, %l3 ! Change flag to ensure printing of

! negpos product after multiplication

! is done; filled delay slot

printnp: call printf ! Printing negpos product.

nop

ba negneg ! Move to final multiplication, negneg.

mov 1, %l4 ! Change flag to ensure printing of

! negneg product after multiplication

! is done; filled delay slot

printnn: call printf ! Printing negneg product.

! Terminate program.

nop

ba done

mov 1, %g1 ! Finished; filled delay slot

done: ta 0