Report

(1)

For the following values of A and B, how many clock cycles are needed to execute your first program from Lab 7 on your baseline MIPS processor, before adding optimizations of Lab 9? Assuming that we run the MIPS processor at 20 MHz, how much time (in seconds) would that take? (You can assume that the loading of the numbers requires only one instruction)

Value of A	Value of B	Number of cycles	Time in seconds
0	8		
6	8		
0	90’000		
89’996	90’002		

```
main:
    addi    $s0,    $0,     0           # A = 0
    addi    $s1,    $0,     8           # B = 8
    addi    $t2,    $0,     0           # S = 0
    slt     $t1,    $s0,    $s1         # $t1 = A < B ? 1 : 0
    beq     $t1,    $0,     end         # If A > B, jump to end
    j       loop                        # Jump to loop

loop:
    add     $t2,    $t2,    $s0         # S = S + A
    beq     $s0,    $s1,    end         # If A == B, jump to end
    addi    $s0,    $s0,    1           # A = A + 1
    j       loop                        # jump to loop

end:
    j       end                         # Infinite loop at the end
```

The formula to calculate the number of cycles is given by

$$N(A,B) = \begin{cases} 6 + 4 \cdot (B - A) + 2, & A \leq B \\ 5, & A > B \end{cases}$$

A	B	Number of cycles	Time in seconds
0	8	40	0.000002s
6	8	16	0.0000008s
6	90'000	359'984	0.0179992s
89'996	90'002	32	0.0000016s

(2)

Fill in the new values for the Table in Exercise 1 when using the modified MIPS architecture running the optimized code, as discussed in the manual for Lab 9. (You can assume that the loading of the numbers requires only one instruction)

Value of A	Value of B	Number of cycles	Time in seconds
0	8		
6	8		
0	90’000		
89’996	90’002		

```
.text
main:
        addi    $t0,    $0,     0               # $t0 = A
        addi    $t1,    $0,     8               # $t1 = B
        addi    $t2,    $t0,    -1              # A-1
        multu   $t0,    $t2                             # A (A-1)
        mflo    $t0                                             # mult result in t0
        srl     $t0,    $t0,    1               # divide by two

        addi    $t2,    $t1,    1               # B+1
        multu   $t1,    $t2                             # B (B+1)
        mflo    $t1                                             # mult result in t1
        srl     $t1,    $t1,    1               # divide by two
        sub     $t2,    $t1,    $t0             # end result is the difference

end:
        j               end                                             # loop t2 is
the result
```

The formula to calculate the number of cycles is given by

$$N(A, B) = 11$$

A	B	Number of cycles	Time in seconds
0	8	11	0.00000055s
6	8	11	0.00000055s
6	90'000	11	0.00000055s
89'996	90'002	11	0.00000055s

(3)

Compare the size/device utilization of the two implementations (before and after the modifications in Lab manual 9). What differences do you see? Briefly comment on them.
Hint: Look into the synthesis report.

Without additional instructions implemented:

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	461	0	20800	2.22
LUT as Logic	205	0	20800	0.99
LUT as Memory	256	0	9600	2.67
LUT as Distributed RAM	256	0		
LUT as Shift Register	0	0		
Slice Registers	31	0	41600	0.07
Register as Flip Flop	31	0	41600	0.07
Register as Latch	0	0	41600	0.00
F7 Muxes	0	0	16300	0.00
F8 Muxes	0	0	8150	0.00

with additional instructions implemented:

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	629	0	20800	3.02
LUT as Logic	373	0	20800	1.79
LUT as Memory	256	0	9600	2.67
LUT as Distributed RAM	256	0		
LUT as Shift Register	0	0		
Slice Registers	64	0	41600	0.15
Register as Flip Flop	64	0	41600	0.15
Register as Latch	0	0	41600	0.00
F7 Muxes	0	0	16300	0.00
F8 Muxes	0	0	8150	0.00

As evident, implementing the additional functionality increases the number of used LUTs (logic) by nearly 50 percent and doubles the number of registers. This surprised us a lot!