Karan Vora (N12229954)
Computer Systems Architecture Assignment 3

Problem 1):

Solution A):

lw x5, 40(x27)

Solution B):

sw x5, 68(x27)

Solution C):

add x7, x5, x6

Solution D):

add x8, x28, x29
addi x8, x8, 31
slli x8, x8, 2
lw x9, x8(x27)
slli x6, x6, 2
sw x9, x6(x31)

Solution E):

I): f = g – A[B[9]]

lw x8, 36(x30)
slli x8, x8, 2
lw x9, x8(x27)
sub x9, x6, x6
lw x5, x9

II): f = g – A[C[8] + B[4]]

lw x30, 16(x30)
lw x31, 32(x31)
add x8, x30, x31
slli x8, x8, 2
lw x9, x8(x27)
sub x5, x6, x9

III): A[i] = B[2i + 1], C[i] = B[2i]

add x8, x28, x28
addi x8, x8, 1

```
lw x9, x8(x30)
slli x10, x28, 2
sw x8, x10(x27)

add x8, x28, x28
lw x9, x8(x30)
slli x10, x28, 2
sw x9, x10(x31)
```

IV): A[i] = 4B[i - 1] + 4C[i + 1]

```
sub x8, x28, 1
slli x10, x8, 2
lw x10, x10(x30)
slli x10, x10, 2
add x9, x28, 1
slli x9, x9, 2
lw x11, x11(x31)
slli x11. X11, 2
add x10, x10, x11
slli x11, x28, 2
sw x10, x11(x27)
```

V): f = g – A[C[4] + B[12]]

```
lw x30, 16(x30)
lw x31, 48(x31)
add x8, x30, x31
slli x8, x8, 2
lw x9, x8(x27)
sub x5, x6, x9
```

==========

Problem 2):

Solution A):

srli x7, x5, 16 → 000000000000AAAA
addi x7, x7, -128 → 000000000000AA2A
srai x7, x7, 2 → 0000000000002A8A
and x7, x7, x6 → 0000000000000228

Solution B):

2345678123456780

Solution C):

0000000015555555

AND

0000000000000FEF

=

0000000000000545

==========

Problem 3):

add x5, x6, x7

opcode : 011011
rs1 : 00110
rs2 : 00111
rd : 00101
imm : 0000000
func3 : 000
func7 : 0000000
hex : 33/0/00
32 bit : 00000000011100110000001010110011

addi x8, x5, 512

opcode : 0010011
rs1 : 00101
rs2 : n/a
rd : 01000
imm : 1000000000
func3 : 000
func7 : n/a
hex : 13/0
32 bit : 00110000000000101000001000001011

ld x3, 128(x27)

opcode : 0000011
rs1 : 11011
rs2 : n/a
rd : 00011
imm : 10000000
func3 : 000
func7 : n/a
hex : 03/0
32 bit : 00001000000011011000000011000011

sd x3, 256(x28)

opcode : 0100011
rs1 : 11100
rs2 : 00011
rd : n/a
imm : 000100000000
func3 : 011
func7 : n/a
hex : 23/3
32 bit : 00010001110000011011000000100011

beq x5, x6 ELSE

opcode : 1100011
rs1 : 00101
rs2 : 00110
rd : n/a
imm : 16 * PC or PC << 4
func3 : 000
func7 : n/a
hex : 63/0
32 bit : imm[12] imm[10:5] 00110 00101 000 imm[4:1]imm[11] 1100011

add x3, x0, x0

opcode: 0110011
rs1: 00000
rs2: 00000
rd: 00011
imm:0000000
func3: 000
func7: 0000000
hexadecimal: 33/0/00
32 bit: 00000000000000000000000110110011

auipc x3, FFEFA

opcode: 0010111
rs1: n/a
rs2: n/a
rd: 00011
imm:00010
func3: n/a
func7: n/a
hexadecimal: 17
32 bit: 00000000000000010000000110010111

jal x3 ELSE

opcode: 1101111
rs1: n/a
rs2: n/a
rd: 00011
imm: PC * 16
func3: n/a
func7: n/a
hexadecimal: 6F
32 bit: imm[20|10:1|11|19:12] 00011 1101111

==========

Problem 4):

Solution A):

slli x5, x11, 16

Solution B):

andi x10, x3, 00001F80
andi x11, x4, E0EFFFFF
or x4, x10, x11

Solution C):

xori x5, x6, FFFFFFFF

==========

Problem 5):

Solution A):

Maximum address value = 0x60000000 + 2^18 – 1 = 0x600FFFFE

Minimum address value = 0x 6000 0000 – 2^18 = 0x5FFFF001

Solution B):

Maximum address value = 0x60000000 + 127 = 0x60000FFE

Minimum address value = 0x60000000 – 128 = 0x5FFFF001

==========

Problem 6):

Solution A):

```
for(i=10; i<=0; --i)
{
    acc=acc+2;
}
```

Solution B):

For every iteration the loop will run 4 times and then the final check. This means that given that x6 has the value N the loop will run 4*N + 1 times. In the earlier example there were a total of 41 RISC-V instructions executed.

Solution C):

```
for(i=10; i<0; --i)
{
      acc=acc+2;
}
```

==========

Problem 7):

Solution A):

```
add x7, x0, x0
LOOPi :
beq x7, x5, Done
addi x7, x7, 1
add x29, x0, x0

LOOPj:
beq x29, x6, LOOPi
add x29, x29, 1
slli x30, x30, 2
add x30, x30, x10
add x31, x7, x29
sd x31, 0(x30)
beq x0, x0, LOOPj
done:
```

Solution B):

It needed (3*10) + (7*10) + 1 RISC V Instructions. 101 instructions executed in the given case.

==========

Problem 8):

Solution A):
0x10000000 0x11
0x10000001 0x22
0x10000002 0x33
0x10000003 0x44
0x10000004 0x55
0x10000005 0x66
0x10000006 0x77
0x10000007 0x88

Solution B):
0x10000000 0x88
0x10000001 0x77
0x10000002 0x66
0x10000003 0x55
0x10000004 0x44
0x10000005 0x33
0x10000006 0x22
0x10000007 0x11

==========

Problem 9):

```
lui x10, 0x12345
addi x10, x10, 0x678
slli x10, x10, 16
slli x10, x10, 16
lui x5, 0x12345
addi x5, x5, 0x678
add x10, x10, x5
```

==========

Problem 10):

Solution A):

if x5 and x6 are 8 bit registers, x6 = 128
if x5 and x6 are 16 bit registers, x6 = 65408
if x5 and x6 are 32 bit registers, x6 = 2^{32}

Solution B):

Overflow will occur is x6 will be in the range of -2^31 to –(2^31 – 128)

Solution C):

Overflow will occur is x6 will be in the range of -2^31 to –(2^31 – 127)