Report

Handwritten

2.8

```
A[1] = &A[0]
f = A[1] + &A[0]
```

2.9

```
addi x30, x10, 8
immediate = 8, rs1 = 10, funct3 = 0, rd = 30, opcode = 19

addi x31, x10, 0
immediate = 0, rs1 = 10, funct3 = 0, rd = 31, opcode = 19

sd x31, 0(x30)
immed[11:5] = 0, rs2 = 31, rs1 = 30, funct3 = 3, opcode = 35

ld x30, 0(x30)
immediate = 0, rs1 = 30, funct3 = 3, rd = 30, opcode = 3

add x5, x30, x31
funct7 = 0, rs2 = 31, rs1 = 30, funct3 = 0, rd = 5, opcode =
51
```

2.16

128 registers need 7 bits for addressing, four time of instructions make opcode have another 2 bits.

2.16.1

Original number of bits:

funct7 = 7 , rs2 = 5, rs1 = 5, funct3 = 3, rd = 5, opcode = 7

After expansion:

funct7 = 1, rs2 = 7, rs1 = 7, funct3 = 1, rd = 7, opcode = 9

2.16.2

Original number of bits:

immediate = 12, rs1 = 5, funct3 = 3, rd = 5, opcode = 7

After expansion:

immediate = 6, rs1 = 7, funct3 = 3, rd = 7, opcode = 9

2.16.3

The proposed change could decrease the size of a RISCK-V assembly program by decreasing number of instructions for loading and storing registers, since the number of registers has increased.

The proposed change could increase the size of a RISCK-V assembly program by increasing the number of instructions needed for adding a constant, since the number of bits of immediate field in I-type become fewer.

Report on matrix multiplication:

It takes 16892568 cycles by doing the naïve matrix multiplication written by myself.

```
root@4f0f554cef5a:~/Problems/matrix# make
riscv64-unknown-elf-gcc -O3 -o matrix matrix.c matrix.s
root@4f0f554cef5a:~/Problems/matrix# make test
spike pk ./matrix
bbl loader
Took 16892568 cycles
root@4f0f554cef5a:~/Problems/matrix#
```

It roughly needs 2 *128^3 = 4194304 load and 128^2 = 16384 store.

It roughly needs 128^3 = 2097152 loop controls.

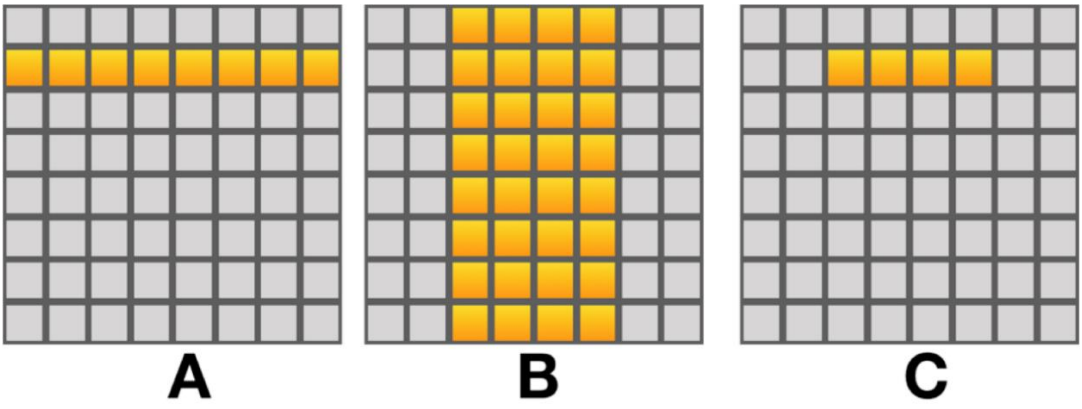
My final result using 1 by 16 blocking takes 8951472 cycles.

```
root@4f0f554cef5a:~/Problems/matrix# make
riscv64-unknown-elf-gcc -O3 -o matrix matrix.c matrix.s
root@4f0f554cef5a:~/Problems/matrix# make test
spike pk ./matrix
bbl loader
Took 8951472 cycles
root@4f0f554cef5a:~/Problems/matrix#
```

It roughly needs 128^3 + (128^3)/16 = 2228224 load and 128^2 = 16384 store.

It roughly needs (128^3)/16 = 131072 loop controls.

Blocking is a way to keep registers being used as much as possible before they are replaced.



The figure is an example of 1 by 4 block. It calculates 4 element of C at the same time.

I use 1 by 16 block in my program. By using blocking, a register loaded with elements in A can be used 16 times before being replaced. Therefore, it can reduce the number of loading elements from A, and reduce the number of cycles.