HW3

2.3

```
sub x30, x28, x29  // x30 ← i-j
slli x30, x30, 3   // x30 ← 8*(i-j)
add x30, x30, x10  // x30 ← A+8*(i-j)
ld  x30, 0(x30)    // x30 ← A[i-j]
sd  x30, 64(x11)   // B[8] ← A[i-j]
```

2.4 B[g] = A[f+1] + A[f]

2.7

```
slli x30, x28, 3    // x30 = 8*i
add  x30, x30, x10  // x30 = &A[i]
ld   x30, 0(x30)    // x30 = A[i]
slli x31, x29, 3    // x31 = 8*j
add  x31, x31, x10  // x31 = &A[j]
ld   x31, 0(x31)    // x31 = A[j]
add  x30, x30, x31  // x30 = A[i] + A[j]
sd   x30, 64(x11)   // B[8] = A[i] + A[j]
```

2.8 f = 2 * &A

2.9

	imm12	rs2	rs1	funct3	rd	opcode
	7 bits	5 bits	5 bits	3 bits	5 bits	7 bits
addi x30, x10, 8	0000000	01000	01010	000	11110	0010011
addi x31, x10, 0	0000000	00000	01010	000	11111	0010011
sd x31, 0(x30)	0000000	11111	11110	111	00000	0100011
ld x30, 0(x30)	0000000	00000	11110	011	11110	0000011
add x5, x30, x31	0000000	11111	11110	000	00101	0110011

2.21 $x6 = 0 | 2 = 2 = 0x0000000000000002$

原因：x5的值>0，bge跳转至ELSE，做 x6 = x0 按位或 2

2.22

2.22.1 jal 指令使用offset20，offset的范围为$[-2^{20}, 2^{20}-2] = [0xFFF00000, 0x000FFFFE]$

∴ PC+offset的范围是 $[0x1FF00000, 0x200FFFFE]$，且最低位一定为0

2.22.2 beq指令使用offset12，范围是$[-2^{12}, 2^{12}-2] = [0xFFFFF000, 0x00000FFE]$

∴ PC+offset的范围是 $[0x1FFFF000, 0x20000FFE]$，且最低位一定是0

2.23

2.23.1 UJ-format

2.23.2
```
    bge  x0, x29, Exit
    sub  x29, x29, 1
    jal  x0, loop
Exit
```

2.24

2.24.1 x5 = 20

2.24.2
```
while (i!=0) {
    i--;
    acc+=2;
}
```

2.24.3 执行了(4N+1)条指令

对于 x6 = N, N-1, ..., 1，4条指令都要执行

当x6 = 0时，执行一次beq指令后3跳出循环

2.24.4
```
while (i≥0) {
    i--;
    acc+=2;
}
```

2.29

argument n — x10

result fib(n) — x11

```
Fib: addi sp,sp,-16
     sd   x1,8(sp)    // save return address
     sd   x10,0(sp)   // save argument n
     add  x5,x0,2     // x5 = 2
     bge  x10,x5,L1   // if n≥2, go to L1
     addi sp,sp,16    // recover sp
     jalr x0,0(x1)
L1:  addi x10,x10,-1  // x10 = n-1

     jal  x1,Fib      // x11 = fib(n-1)
     add  x6,x11,x0   // x6 = fib(n-1)
     addi x10,x10,-1  // x10 = n-2
     jal  x1,Fib      // x11 = fib(n-2)
     add  x11,x11,x6  // x11 = fib(n-1)+fib(n-2)
     ld   x10,0(sp)
     ld   x1,8(sp)    // restore argument and return address
     addi sp,sp,16
     jalr x0,0(x1)
```