CENG 3420
Computer Organization & Design

HW1 Review

LIU Hongduo
CSE Department, CUHK
hdliu21@cse.cuhk.edu.hk

Spring 2022

Overview

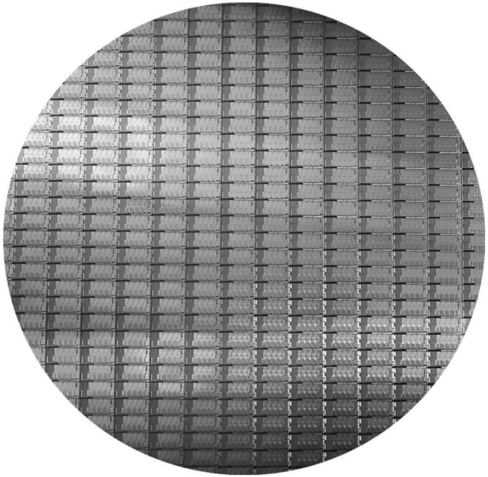
Overview

Q1 Description

Assume a 20 cm diameter wafer has a cost of 15, contains 100 dies, and has 0.030 defects/cm^2.

1. Find the yield of this wafer.
2. Find the cost per die for this wafer.

Related Course Contents


A 12-inch (300 mm) wafer of Intel Core i7 (Courtesy Intel).

Related Course Contents

wafer	A slice composed of silicon, used to create chips.
die	The individual rectangular sections that are cut from a wafer, more informally known as chips.
defect	A microscopic flaw in a wafer that can result in the failure of the die containing that defect.
yield	The percentage of good dies from the total number of dies on the wafer.

Solution 1.1

$$\text{Die area} \approx \frac{\text{Wafer area}}{\text{Dies per wafer}} \tag{1}$$

This equation is an approximation, since the area near the border of the wafer **cannot** accommodate the rectangular dies. According to equation Equation (1), Die area $\approx \frac{\pi \times (\frac{20}{2})^2}{100} \approx 3.14\text{cm}^2$

$$\text{Yield} = \frac{1}{(1 + \frac{\text{Defects per area} \times \text{Die area}}{2})^2} \tag{2}$$

This equation is based on empirical observations of yields at the integrated circuit factories. According to equation Equation (2), Yield $= \frac{1}{(1+\frac{0.030\times 3.14}{2})^2} = 0.9121$.

Solution 1.2

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{yield}} \tag{3}$$

According to Equation (3), Cost per die $= \frac{15}{100 \times 0.9121} = 0.1645$.

Overview

Q2 Description

Suppose we developed a new processor that has 75% of the capacitive load of the older processor. Also it can reduce voltage 20% compared to previous generation. What is the impact on dynamic power if the frequency keeps unchanged? Give the ratio of $\frac{\text{Power}_{\text{new process}}}{\text{Power}_{\text{old processor}}}$.

Solution of Q2

$$\text{Power} = \frac{1}{2} \times \alpha \times \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency switched} \tag{4}$$

According to Equation (4), $\frac{\text{Power}_{\text{new}}}{\text{Power}_{\text{old}}} = 0.48$.

Overview

Q3 Description

For the following C statement, write the corresponding RISC-V assembly code. Assume that the C variables f, g, and h, have already been placed in registers `x5`, `x6`, and `x7` respectively. Use a minimal number of RISC-V assembly instructions.

```
f = g + (h-8)
```

Solution of Q3

As shown below,

1. addi x5, x7, -8
2. add x5, x5, x6

Overview

Q4 Description

Assume the following register contents:
`x5 = 0xAAAAAAAA, x6 = 0x12345678`
1. For the register values shown above, what is the value of `x7` for the following sequence of instructions?

```
slli x7, x5, 4
or x7, x7, x6
```

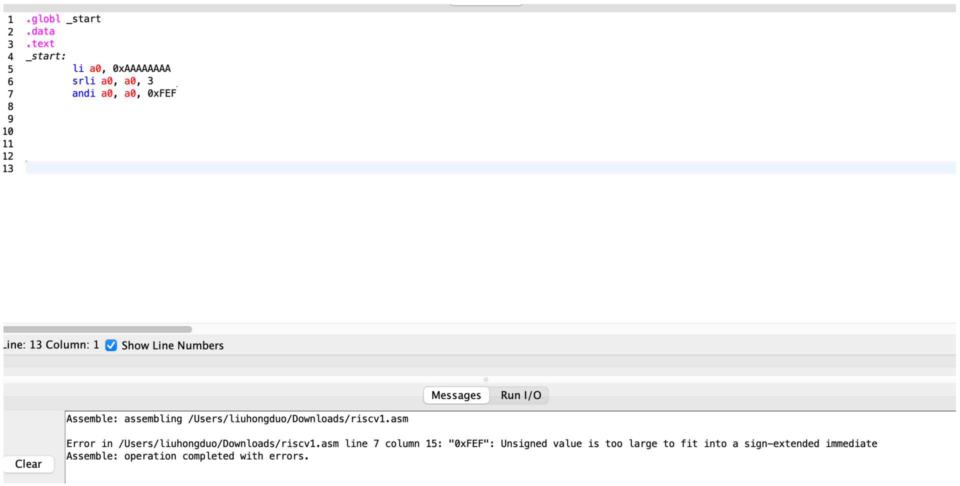
2. For the register values shown above, what is the value of `x7` for the following sequence of instructions?

```
srli x7, x5, 3
andi x7, x7, 0xFEF
```

Solution of Q4

1. 0xbabefef8
2. When we treat OxFEF as 0x00000FEF, the answer is 0x545.
When we treat OxFEF as 0xFFFFFFEF, the answer is 0x15555545.
In general, the immediate of `andi` instruction will be signed-extended to 12 bits. Since here is an ambiguity, we accept both solutions when grading. Please note that when we actually run the code in RARS simulator, the RARS simulator treats the immediate 0xFEF as an unsigned value 4079 and sign-extents it to a 12-bit value will cause an error.

Solution of Q4



```
.globl _start
.data
.text
_start:
        li a0, 0xAAAAAAAA
        srli a0, a0, 3
        andi a0, a0, 0xFEF
```

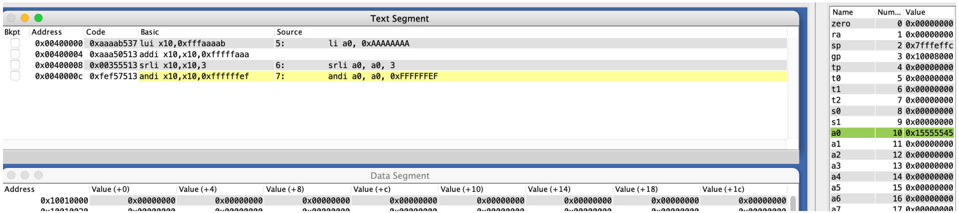
Line: 13 Column: 1 ✓ Show Line Numbers

```
Assemble: assembling /Users/liuhongduo/Downloads/riscv1.asm

Error in /Users/liuhongduo/Downloads/riscv1.asm line 7 column 15: "0xFEF": Unsigned value is too large to fit into a sign-extended immediate
Assemble: operation completed with errors.
```

Run the code directly in RARS simulator

Solution of Q4



Text Segment

Bkpt	Address	Code	Basic	Source
☐	0x00400000	0xaaaab537	lui x10,0xfffaaaab	5: li a0, 0xAAAAAAAA
☐	0x00400004	0xaaa50513	addi x10,x10,0xfffffaaa	
☐	0x00400008	0x00355513	srli x10,x10,3	6: srli a0, a0, 3
☐	0x0040000c	0xfef57513	andi x10,x10,0xffffffef	7: andi a0, a0, 0xFFFFFFEF

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Name	Num...	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7fffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x15555545
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000

Treat OxFEF as OxFFFFFFEF

Overview

Q5 Description

Consider the following RISC-V loop:

```
LOOP: beq x6, x0, DONE
      addi x6, x6, -1
      addi x5, x5, 2
      jal x0, LOOP
DONE:
```

1. Assume that the register `x6` is initialized to the value 10. What is the final value in register `x5` assuming the `x5` is initially zero?
2. For the loop above, write the equivalent C code. Assume that the registers `x5` and `x6` are integers `acc` and `i`, respectively.
3. For the loop written in RISC-V assembly above, assume that the register `x6` is initialized to the value N. How many RISC-V instructions are executed?
4. For the loop written in RISC-V assembly above, replace the instruction "`beq x6, x0, DONE`" with the instruction "`blt x6, x0, DONE`" and write the equivalent C code.

Solution of Q5

1. 20

2.
   ```
   while (i != 0) {
   acc += 2;
   i = i-1; }
   ```
3. 4N+1

4.
   ```
   while (i >= 0) {
   acc += 2;
   i = i-1; }
   ```

Overview

Q6 Description

Some RISC-V assembly instructions are shown below. Assume that the variables f, g are assigned to registers `x5, x6`, respectively. Assume that the base address of the arrays A and B are in registers `x10` and `x11`, respectively.

```
slli x30, x5, 2 // x30 = f*4
add x30, x10, x30 // x30 = &A[f]
slli x31, x6, 2 // x31 = g*4
add x31, x11, x31 // x31 = &B[g]
lw x5, 0(x30) // x5 = A[f]
addi x12, x30, 4
lw x30, 0(x12)
add x30, x30, x5
sw x30, 0(x31)
```

1. What's the meaning of the last four instructions.
2. What is corresponding C statement?

Solution of Q6

1.
   ```
   addi x12, x30, 4 // x12 = &A[f]+4 (i.e.  &A[f+1])
   lw x30, 0(x12) // x30 = A[f+1]
   add x30, x30, x5 // x30 = A[f+1] + A[f]
   sw x30, 0(x31) // B[g] = x30 (i.e.  A[f+1] + A[f])
   ```
2. The corresponding C code is:
   ```
   B[g]= A[f] + A[f+1]
   ```