- 期中考參考解答，有問題的同學，請來找助教討論 ：
  1、2、3 → 謝文雯 R227　4、5、6 → 楊侑儒 R228　7、8、9 → 李岳叡 R227

1. (10%)

(a) (3%) Yield

→ Percentage of good dies from the total number of dies on the wafer.

(b) (3%) SPEC2000

→ A standard set of benchmarks.

(c) (4%) Amdahl's Law :

→ A rule to find out the maximum expected improvement to an overall system when only a part of the system is improved. Amdahl's law can be written in terms of overall speedup as a function

$Speedup = \dfrac{1}{\frac{f}{s} + (1 - f)}$, where f stands for the fraction improved and s stands for the amount of improvement.

2. (15%)

(a) (7%)

$$\frac{Performance_{I1}}{Performance_{I2}} = \frac{Execution\,time_{I2}}{Execution\,time_{I1}} \propto \frac{CPI_{I2}}{CPI_{I1}} \times \frac{Clock\ rate_{I1}}{Clock\ rate_{I2}}$$

$$\frac{1 \times 40\% + 2 \times 40\% + 2 \times 20\%}{2 \times 40\% + 3 \times 40\% + 5 \times 20\%} \times \frac{6GHz}{3GHz} = \frac{1.6}{3.0} \times \frac{2}{1} = \frac{16}{15} \approx 1.067$$

∴ with C1, I1 is about 1.067 times as fast.

(b) (8%)

∵ all other criteria are identical

∴ only take the execution time into consideration

Form (a)

I1 + C1 = $\dfrac{3.0}{6GHz}$　　I2 + C1 = $\dfrac{1.0}{3GHz}$

I1 + C2 = $\dfrac{2 \times 40\% + 3 \times 20\% + 5 \times 40\%}{6GHz} = \dfrac{3.4}{6GHz}$

I2 + C2 = $\dfrac{1 \times 40\% + 2 \times 20\% + 2 \times 40\%}{3GHz} = \dfrac{1.6}{3GHz}$

I1 + C3 = $\dfrac{2 \times 50\% + 3 \times 25\% + 5 \times 25\%}{6GHz} = \dfrac{3.0}{6GHz}$

$$\text{I2 + C3} = \frac{1\times 50\% + 2\times 25\% + 2\times 25\%}{3GHz} = \frac{1.5}{3GHz}$$

∴ purchasing machine I1 using compiler C1, machine I1 using compiler C3 or machine I2 using compiler C3.

3. (15%)

(a) (5%)

Memory-Memory

```
add   Addrb   Addra   Addrc
```

Load-Store

```
lw    $r0, Addra
lw    $r1, Addrc
add   $r0, $r0, $r1
sw    $r0, Addrb
```

(b) (5%)

Code size

Memory-Memory

```
add   Addrb   Addra   Addrc
 1  +   2   +   2   +   2   = 7 (bytes)
```

Load-Store

```
lw    $r0, Addra
1 + .0.5 + 2 = 3.5 ➔ 4 (bytes)
lw    $r1, Addrc
1 + .0.5 + 2 = 3.5 ➔ 4 (bytes)
add   $r0, $r0, $r1
1 + .0.5 + 0.5 + 0.5 = 2.5 ➔ 3 (bytes)
sw    $r0, Addrb
1 + .0.5 + 2 = 3.5 ➔ 4 (bytes)
```

4 + 4 + 3 + 4 = 15 (bytes)

∴ Memory-Memory is more efficient as measured by code size.

(c) (5%)

Data size

Memory-Memory

data from memory to the processor : 8 (bytes)
data from the processor to memory : 4 (bytes)

Load-Store
data from memory to the processor : 8 (bytes)
data from the processor to memory : 4 (bytes)

∵ Memory bandwidth required = code size + data size
Memory-Memory : 7 + 12 = 19 (bytes)
Load-Store : 15 + 12 = 27 (bytes)
∴ Memory-Memory is more efficient as measured by total memory bandwidth required.

4.
基本上這是送分題，只要寫的可以符合 principle 我就會給分。
相信全對的同學絕對不會過來找我討論:P，所以如果這題你覺得被扣的莫名奇妙，請過來 228 找我討論。
以下列出大概的 key word（因爲解答很多種），只要有提到我大概就會給分，頂多扣 1 分。

簡單明瞭有助於一致性
指令簡單所以很容易看懂，或者寫到只有三種 type。
小就是快
register 個數、memory access 比較慢。只有寫 register 大小只有 32bits 的全錯（根本沒關係啊）
使常出現的部份加快
Addi 指令、$zero。
好的設計需要好的折衷方式
指令格式、I-type、J-type 指令。

5.
有人沒有寫你要做誰大於誰，直接寫出程式碼，這樣還要我猜你是在寫($S1>$S2)還是($S2>$S1)嗎？ 希望以後作答的時候有一個觀念：寫清楚答案，讓閱卷者知道你在寫什麼。還有就是要看清楚題目，題目要求做出”大於”以及”小於”，不是要你兩個寫在一起判斷大於的時候做什麼、小於的時候做什麼，原則上確定有完全正確我應該都有給部分分數，頂多扣一兩分。再來就是，題目要求的是滿足”大於”就做（branch），不是小於等於的情況跳走
Great than ($S1>$S2 時跳)

```
slt $S3, $S2, $S1;
```

```
        bne $S3, $zero, L;
```

Less than and equal ($S1<=$S2 時跳)

```
        slt $S3, $S2, $S1;
        beq $S3, $zero, L;
```

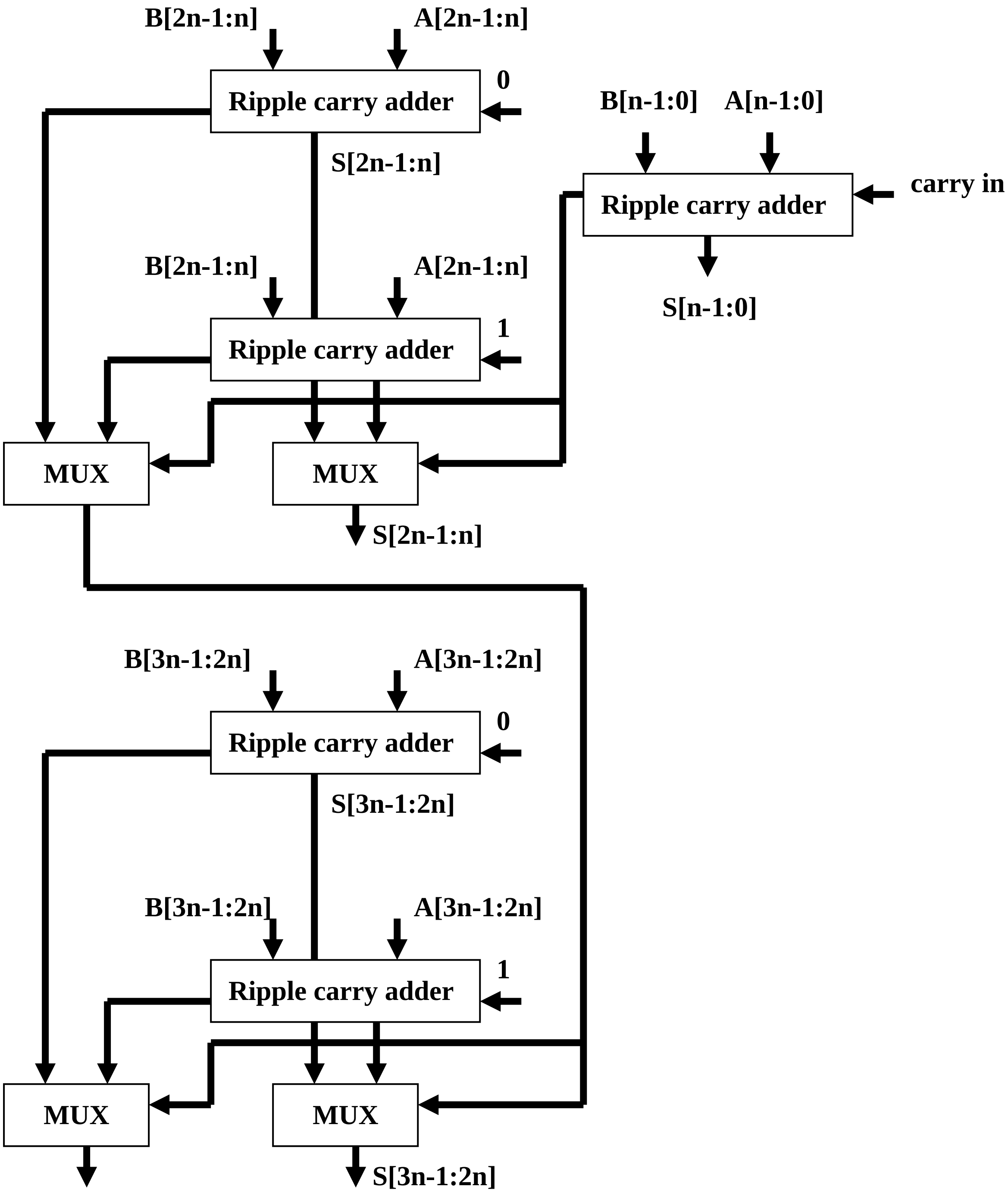
6.

Add	0	0	1	0					
Sub	0	1	1	0	或是	0	1	1	0
Nor	1	1	0	0					
OR	0	0	0	1					

題目要求寫出”bits” ，請不要自作主張寫成 decimal，所以扣了一點點分數。

7.

(a)



(b) 1 * n-bit adder delay + 2 * mux delay

8.

-7 = 1001

-5 = 1011

(a)

```
              1001
              1011
------------------------------
         +)11111001
         +)1111001
         +)00000
         -)11011
------------------------------
           00100011
```

(b)

```
0000   1011   0   → 10     ( sub
0111   1011   0   →         ( shift right
0011   1101   1
------------------------------
0011   1101   1   → 11     ( do nothing
0011   1101   1   →        ( shift right
0001   1110   1
------------------------------
0001   1110   1   → 01     ( add
1010   1110   1   →        ( shift right
1101   0111   0
------------------------------
1101   0111   0   → 01     ( sub
0100   0111   0   →        ( shift right
0010   0011 ( Ans )
```

9.

(a) pattern : 1   00001011   01010….010 (中間….代表連續 0)

S   Exponent   Fraction

→ $(-1)^{S} * (1 + \text{Fraction}) * 2^{(\text{Exponent - Bias})}$

$(-1) * (1 + 1/2^{2} + 1/2^{4} + 1/2^{22}) * 2^{(11 - 127)}$

(b)

1   10000011   10001011101011100001010