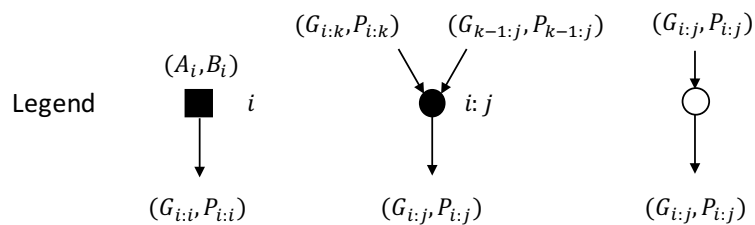
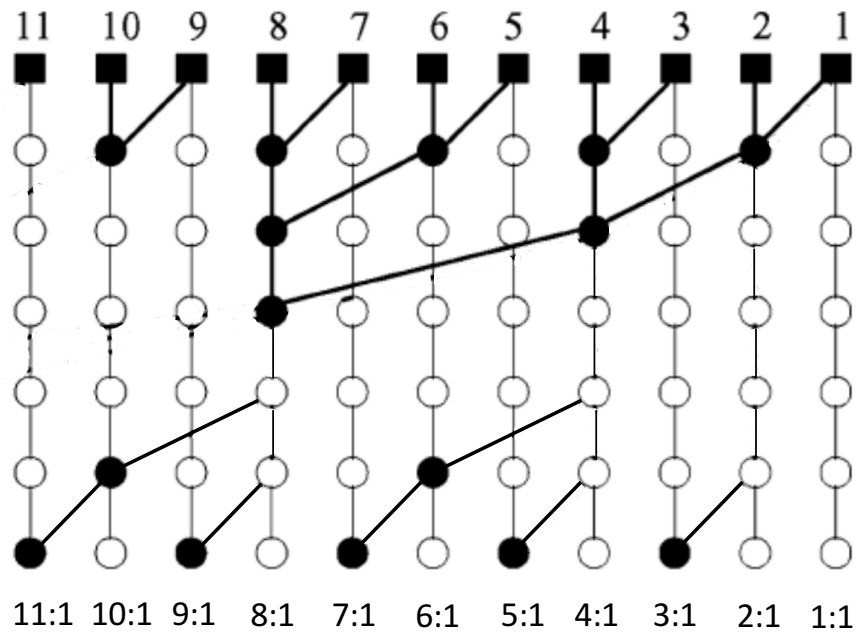


1. (Prefix Adder) Manually simulate the operation of the prefix adder as below. Simulate the addition of 15 and 24. Write the value of $(P_{i,j}, G_{i,j})$ output of each black box/circle. Show the sum $S_i = A_i \oplus B_i \oplus G_{i-1,1}$. Write the value directly on the schematic diagram. Assume that there is no carry in c_{-1} and the lowest bit on right use index 1. (10 points)



- -2: missing/incorrect column generate and propagate.
- -6: missing/incorrect block generate and propagate.
- -2: missing/incorrect sums.
- -1: each wrong pair of propagate and generate or sum.

column i	11	10	9	8	7	6	5	4	3	2	1
15	0	0	0	0	0	0	0	1	1	1	1
24	0	0	0	0	0	0	1	1	0	0	0
$i : j$	11 : 11	10:10	9 : 9	8 : 8	7 : 7	6 : 6	5 : 5	4 : 4	3 : 3	2 : 2	1 : 1
$(G_{i,j}, P_{i,j})$	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,1)	(1,1)	(0,1)	(0,1)	(0,1)
$i : j$		10:9		8 : 7		6 : 5		4 : 3		2 : 1	
$(G_{i,j}, P_{i,j})$		(0,0)		(0,0)		(0,0)		(1,1)		(0,1)	
$i : j$				8 : 5				4 : 1			
$(G_{i,j}, P_{i,j})$				(0,0)				(1,1)			
$i : j$				8 : 1							
$(G_{i,j}, P_{i,j})$				(0,0)							
$i : j$		10:1				6:1					
$(G_{i,j}, P_{i,j})$		(0,0)				(0,0)					
$i : j$	11:1		9:1		7:1		5:1		3:1		
$(G_{i,j}, P_{i,j})$	(0,0)		(0,0)		(0,0)		(1,1)		(0,1)		
S_i	0	0	0	0	0	1	0	0	1	1	1

II. (Carry Save Adder) The carry save adder in lecture 14 adds two numbers for each addition operation. Design a carry save adder that adds **three** numbers for each addition operation. (15 points)

1. Number system and algorithm (7 points)

- -3: wrong/missing number system.
- -4: missing algorithm.
- -2: incorrect number system with justifiable statement.
- -2: incorrect algorithm with justifiable statement.
- -1: correct algorithm based on incorrect number system.

2. Addition table (8 points)

- -1: each mistake in the table.
- -1: use more than 3 bits in the redundant system.
- -4: simulates addition with 2 bits redundant system instead of 3.
- -6: simulates addition with no redundant bit.

II.1 Write your number system and algorithm. (7 points)

Consider the addition of three numbers for each addition, we will have at most one bit carry out of the current digit. However, when the carry out is propagated to the higher digit as one addend in each column would have four one bit numbers. Therefore, we need two bits of carry out and a three bits redundant system. The redundant number system has

- three bits for each digit
 - (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)
- weight 0: (0,0,0)
- weight 1: (0,0,1), (0,1,0), (1,0,0)
- weight 2: (0,1,1), (1,0,1), (1,1,0)
- weight 3: (1,1,1)

In each addition operation, two bits need to be propagated to the redundant system of higher digits. The algorithm of operating this carry save addition system follows the steps

1. Convert the two addend in bits to the redundant system
2. Perform carry save addition for intermediate solution
3. Split the redundant number into three numbers
4. Perform the addition of the three numbers using the carry save adder with two bits redundant system or other kinds of CPAs

II.2 Manually simulate the addition of $n_1 = 7$, $n_2 = 5$, $n_3 = 6$, $n_4 = 11$, $n_5 = 4$. (8 points) Write the results in the following table.

id	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
n_1	000	000	000	000	000	100	100	100
n_2	0	0	0	0	0	1	0	1
n_3	0	0	0	0	0	1	1	0
$A = n_1 + n_2 + n_3$	000	000	000	000	010	110	010	000
n_4	0	0	0	0	1	0	1	1
n_5	0	0	0	0	0	1	0	0
$B = A + n_4 + n_5$	000	000	000	010	010	110	000	100
$Total$	0	0	1	0	0	0	0	1

III. A sequential adder inputs (a_{2i}, b_{2i}) and (a_{2i+1}, b_{2i+1}) , the $2i$ 'th and $(2i + 1)$ 'th bits of two binary numbers in each clock cycle for $i = 0$ to $\frac{n}{2} - 1$ and outputs the sum s_{2i}, s_{2i+1} , where n is the number of bits and is assumed to be even. Implement the adder with a D flip-flop, two **Full Adders** and a minimal network of AND, OR, NOT gates. (15 points)

1. State table (8 points)

- -1: two-bit serial adders without carry out $2i+2$.
- -2: incomplete two-bit serial adders with carries
- -4: two-bit serial adders without carry in or write carry in as output.
- -6: only state table for one-bit serial adder.

2. Diagram (7 points)

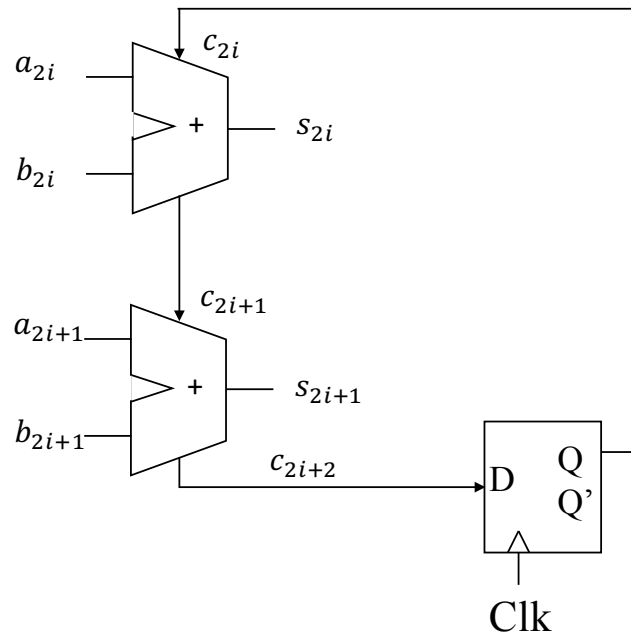
- -2: each mistake.
- -4: wrong diagram with one D F-F and two full adders.

III.1. Write the state table to describe the next states and outputs as functions of present state and inputs. (8 points)

Answer: Input $a_{2i+1:2i}$ and $b_{2i+1:2i}$, output sum $s_{2i+1:2i}$, the carry in c_{2i} to digit $2i$ and carry out c_{2i+2} from digit $(2i + 1)$ are connected by the D F-F,

$$D = c_{2i+2}, \quad Q = c_{2i}.$$

III.2. Draw the schematic diagram. (7 points)



$a_{2i+1:2i}$	$b_{2i+1:2i}$	c_{2i}	c_{2i+1}	c_{2i+2}	$s_{2i+1:2i}$
00	00	0	0	0	00
00	00	1	0	0	01
00	01	0	0	0	01
00	01	1	1	0	10
00	10	0	0	0	10
00	10	1	0	0	11
00	11	0	0	0	11
00	11	1	1	1	00
01	00	0	0	0	01
01	00	1	1	0	10
01	01	0	1	0	10
01	01	1	1	0	11
01	10	0	0	0	11
01	10	1	1	1	00
01	11	0	1	1	00
01	11	1	1	1	01
10	00	0	0	0	10
10	00	1	0	0	11
10	01	0	0	0	11
10	01	1	1	1	00
10	10	0	0	1	00
10	10	1	0	1	01
10	11	0	0	1	01
10	11	1	1	1	10
11	00	0	0	0	11
11	00	1	1	1	00
11	01	0	1	1	00
11	01	1	1	1	01
11	10	0	0	1	01
11	10	1	1	1	10
11	11	0	1	1	10
11	11	1	1	1	11

IV. Counter: Given a modulo 8 counter, construct a counter that counts in a repeated sequence (1,2,5,6,7,0) with minimal networks of AND, OR gates. Assume that NOT gates are free. (15 points)

1. Truth table (8 points)

- -1: for each wrong/missing entry.

2. Diagram (7 points)

- -1: missing labels (CLK, CNT, LD, D and Q).
- -1: not using minimal networks for LD and D.
- -2: wrong LD or D.
- -1: wrong CNT.

IV.1. Write the truth table to describe the load LD , and (D_2, D_1, D_0) as functions of present state (Q_2, Q_1, Q_0) . (8 points)

Id	Q_2	Q_1	Q_0	LD	D_2	D_1	D_0
0	0	0	0	0	-	-	-
1	0	0	1	0	-	-	-
2	0	1	0	1	1	0	1
3	0	1	1	-	-	-	-
4	1	0	0	-	-	-	-
5	1	0	1	0	-	-	-
6	1	1	0	0	-	-	-
7	1	1	1	0	-	-	-

The minimal boolean expressions of LD and (D_2, D_1, D_0) are derived through K-maps.

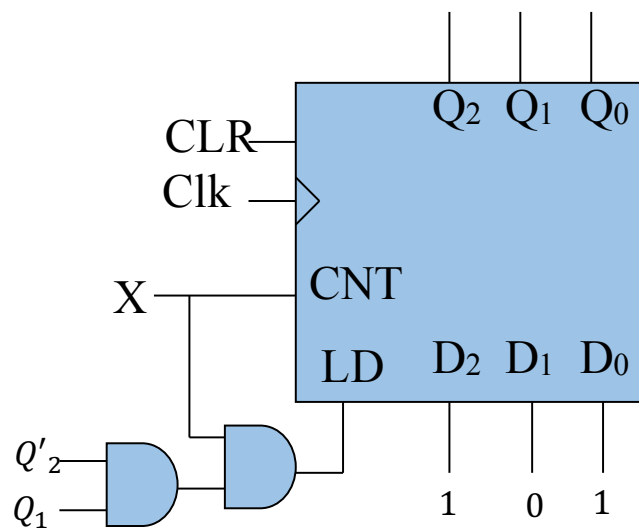
$$LD = Q_2'Q_1$$

$$D_2 = 1$$

$$D_1 = 0$$

$$D_0 = 1$$

IV.2. Draw the schematic diagram. (7 points)



V. (System Designs) Implement the following algorithm: (33 points)

```
Alg(X,Y,Z,W,start,done)
Input X[7:0], Y[7:0], Z[7:0], start;
Output W[7:0], done;
Local-object A[7:0], B[7:0],C[7:0];
S0: if start' goto S0 || done  $\leftarrow$  1;
S1: A  $\leftarrow$  X || B  $\leftarrow$  Y || C  $\leftarrow$  Z || done  $\leftarrow$  0;
S2: if A[7] goto S5 || B  $\leftarrow$  Add(B,C);
S3: A  $\leftarrow$  Add(A,B);
S4: if B[7] goto S2;
S5: if C[7] goto S2 || C  $\leftarrow$  INC(C);
S6: W  $\leftarrow$  A || done  $\leftarrow$  1 || goto S0;
end Alg
```

1. Instructions and components (6 points)

- -3: missing instruction.
- -3: missing component.
- -1: each missing/incorrect item.

2. Data Subsystem (8 points)

- -1: each operation that is not implemented.
- -1: each wrong/missing component/connection/label.

3. Control signals (7 points)

- -1: each missing/wrong state/control signal of the table.

4. State diagram (5 points)

- +1: correct states.
- +4: correct transitions.

5. One hot design (7 points)

- +1: represent states correctly as flip flops.
- +3: use OR gates correctly.
- +3: use AND gates correctly.

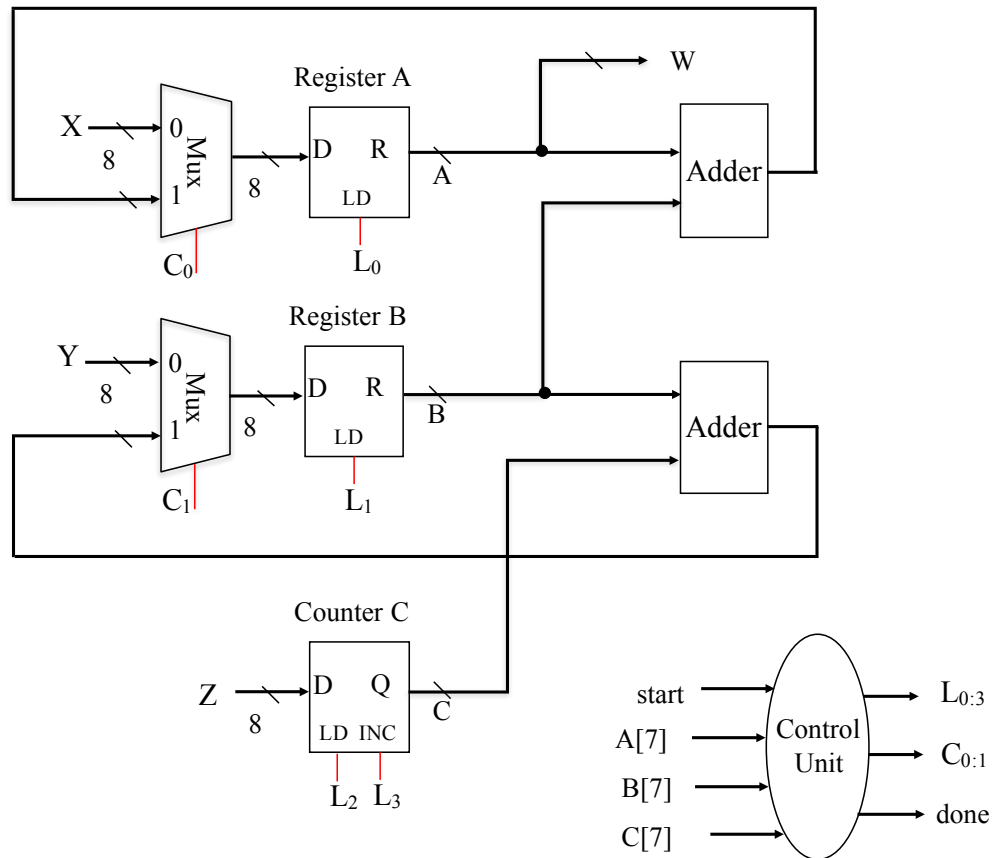
V.1. Design a data subsystem that is adequate to execute the algorithm by answering the following questions.

V.1.1. Use a table to list the instructions and the corresponding components that should be used in the data path subsystem. (6 points)

State	Instruction	Component
S1	$A \leftarrow X$	Register
S1	$B \leftarrow Y$	Register
S1	$C \leftarrow Z$	Register*
S2	$B \leftarrow \text{Add}(B,C)$	Register, Adder, Mux
S3	$A \leftarrow \text{Add}(A,B)$	Register, Adder, Mux
S5	$C \leftarrow \text{Inc}(C)$	counter/register (with Inc module)

*: any other storage component is fine, e.g. counter.

V.1.2. Draw the schematic diagram to show the data path subsystem. Label the inputs, outputs, and control signals of all components. (8 points)

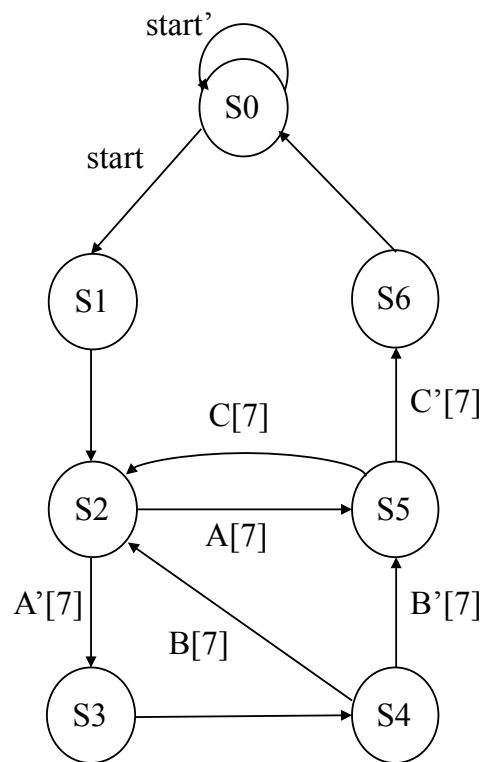


V.2. Design the control subsystem by answering the following questions.

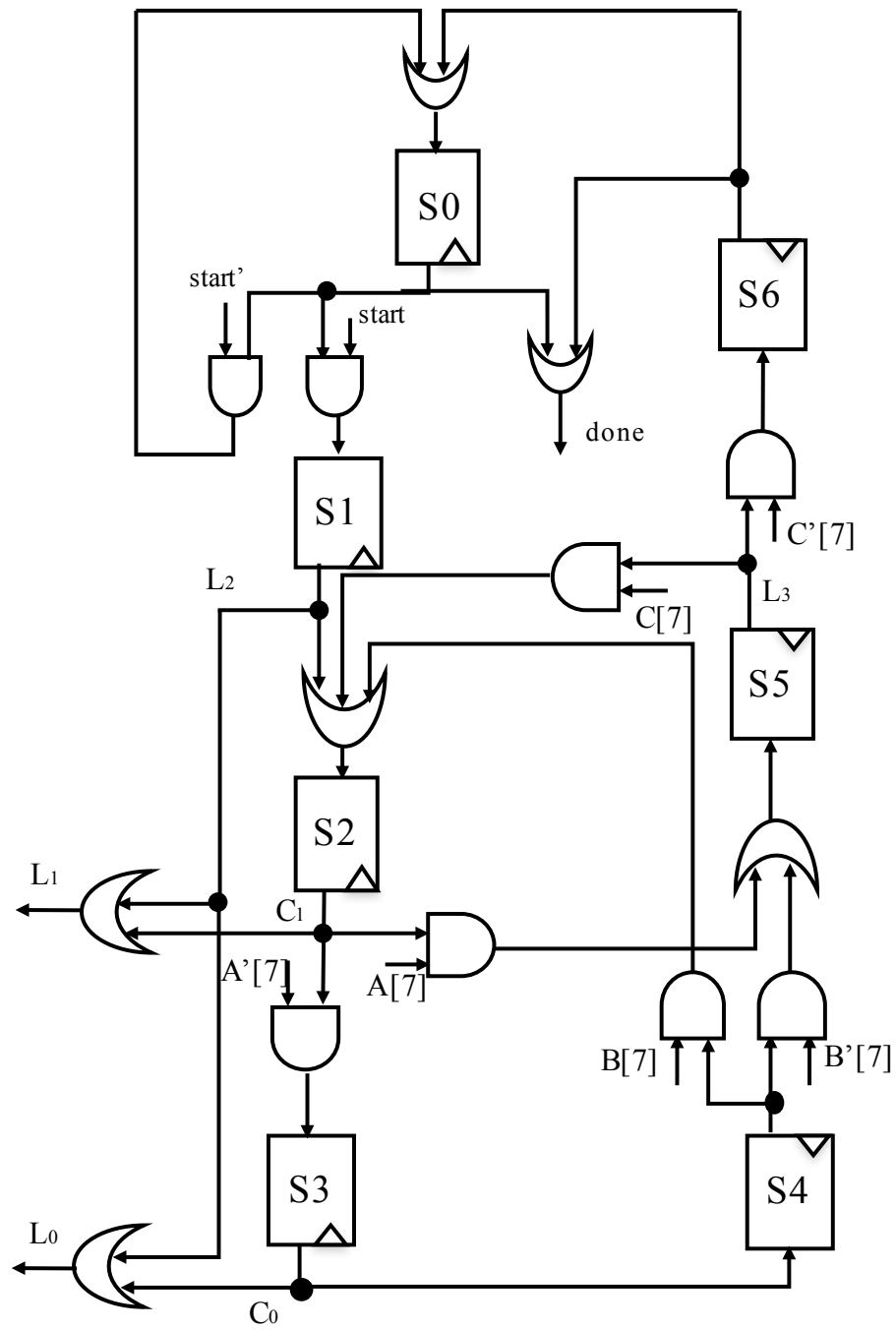
V.2.1. Use a table to list the value of control signals for every state. (7 points)

State	L ₀	L ₁	L ₂	L ₃	C ₀	C ₁	done
S0	0	0	0	0	-	-	1
S1	1	1	1	0	0	0	0
S2	0	1	0	0	-	1	0
S3	1	0	0	0	1	-	0
S4	0	0	0	0	-	-	0
S5	0	0	0	1	-	-	0
S6	0	0	0	0	-	-	1

V.2.2. Draw the state diagram. (5 points)



V.2.3. Implement the control subsystem using a one hot encoding design. Draw the logic diagram.
(7 points)



VI. (Floating Point Processes) According to IEEE floating number standard 754 for 32 bit width numbers, we have 1 sign bit, 8 bits for exponent and 23 bits for mantissa. Write the solution in the normalized format whenever possible. (12 points)

VI.1. $1.010_2 \times 2^{12} \times 1.100_2 \times 2^{-10}$ (3points)

- -0.5: incorrect / no rounding (to three decimal places).
- -1: no normalization (hidden bit is ok); wrong exponent /base; wrong mantissa.

Answer: $1.111_2 \times 2^2$

$$\begin{array}{r} 1.010 \ 2^{12} \\ \times 1.100 \ 2^{-10} \\ \hline .101 \\ 1.01 \ 2^2 \\ \hline 1.111 \ 2^2 \end{array}$$

VI.2. $1.101_2 \times 2^{120} \times 1.010_2 \times 2^{90}$ (3points)

- -0.5: no mention of overflow with process shown.
- -2: no mention of overflow.

With calculation process,

- -0.5: no normalization.
- -1: wrong exponent /base; wrong mantissa

Answer: Overflow

VI.3. $1.000,000,000,01_2 \times 2^{10} - 1.000,000,000,000,010,10_2 \times 2^{-1}$ (3points) (rounding to the nearest even)

- -1: wrong subtraction result.
- -1: wrong rounding result (based on subtraction result).
- -1: wrong format (normalization, round bits, and other rare ones).

Answer: $1.000,000,000,001_2 \times 2^2$

$$\begin{array}{r} 1.000,000,000,01 \qquad \qquad \qquad 2^{10} \\ -0.000,000,000,010,000,000,000,101,0 \ 2^{10} \\ \hline 0.111,111,111,111,111,111,111,11,011,0 \ 2^{10} \\ \downarrow \\ 1.000,000,000,000,000,000,000,00 \ 2^{10} \end{array}$$

23
G R S
G=0, R=1, S=1
R(G+S) = 1 round up

Figure 1: Rounding process of VI.3.

VI.4. What is the relative error for the representation of π ? Estimate the error in a tight bound with the knowledge of the first three digits 3.14. (3points)

- +3: Correct.
- +2: utilize absolute error bound and real number 3.14.
- +1: utilize absolute error bound or real number 3.14.

Answer: the real number 3.14 is normalized with exponent 2^1 as a floating point number with 32 bit width. The absolute rounding error is bounded by

$$err_{abs} \leq 0.5 \times 2^{-23} \times 2^1.$$

The relative error is bounded by

$$err_{rel} = \frac{err_{abs}}{3.14} \leq \frac{1 \times 2^{-23}}{3.14} \approx 3.80 \times 10^{-8}.$$