Machine Learning Intelligent Chip Design
Midterm Examination
Spring 2021

Your Name ____________________
Student ID Number ____________________

Instructions

Exams: Consultation during the exam is not permitted. This is a closed book exam. The dictionary is allowed to look for vocabulary. Don't use scratch paper; show all of your work on these pages.

Regarding Policy: Exams will be accepted for regarding up to two weeks after you get the graded exam. No regrades after two weeks, when submitted, the ENTIRE exam or problem set is up for regarding (and down grading!)

******************** Score Board (to be filled by graders) ****************

	total points	your points
Problem 1	20	
Problem 2	40	
Problem 3	10	
Problem 4	20	
Problem 5	10	
Problem 6	10	
Total	110	

Problem 1: All the following questions are check-all-that-apply questions. You may select at least one option from the alternatives. (20 points)

1. (5%) Memory unit is used to store a large amount of binary information. Please select the incorrect description(s) and provide your reasons.
   (a) Once the data is stored in ROM, it can no longer be modified or deleted.
   (b) ROM have input port to write data.
   (c) RAM allows read and write data at the same time.
   (d) RAM cannot store data when power off.
2. (5%) SC_METHOD, SC_THREAD, and SC_CTHREAD are the popular ways to describe the process in a SystemC module. Please select the improper way(s) to describe the dynamic sensitivity function according to different processes and provide your reason.
   (a) The process by using SC_METHOD can use next_trigger().
   (b) The process by using SC_THREAD can use wait_until().
   (c) The process by using SC_CTHREAD can use wait().
   (d) The process by using SC_CTHREAD can use watching().
3. (5%) If a channel is designed to transfer a data with a certain condition, please select the improper interface description(s) and provide your reason.
   (a) sc_signal_in_if (b) sc_fifo_out_if (c) sc_mutex_if (d) sc_signal_if
4. (5%) Fig. 1 shows that the two modules are connected with a channel. Please determine the wrong declaration(s) and provide your reason.
   (a) sc_out < T > pA (b) sc_port < T > pB
   (c) sc_fifo < T > C (d) sc_port < sc_fifo_out<T> > pA

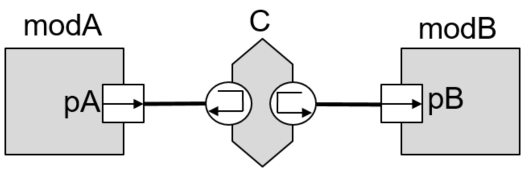


Fig. 1

Answer:

1.
   (b) ROM have no input port to write data.
   (c) RAM cannot read and write data at the same time.
2.
   (b) The process by using SC_THREAD can use wait().
   (c) The process by using SC_CTHREAD can use wait_until().
3.
   (d) sc_signal_if doesn't exist.
4. (b) there is no interface in sc_port declaration.

Problem 2: Calculation system (40 points)

Fig. 2 shows a calculation system, which is composed of (a) one LFSR-based random pattern generator (10%), (b) a channel block (10%), (c) a full adder (10%), and (d) a output buffer (10%). We assume that the initial value of the (D0, D1, D2, D3) and (D4, D5, D6, D7) in the two LFSRs are both (0, 1, 0, 0). Please design the corresponding SC_MODULE of each block in this calculation system with SystemC language.

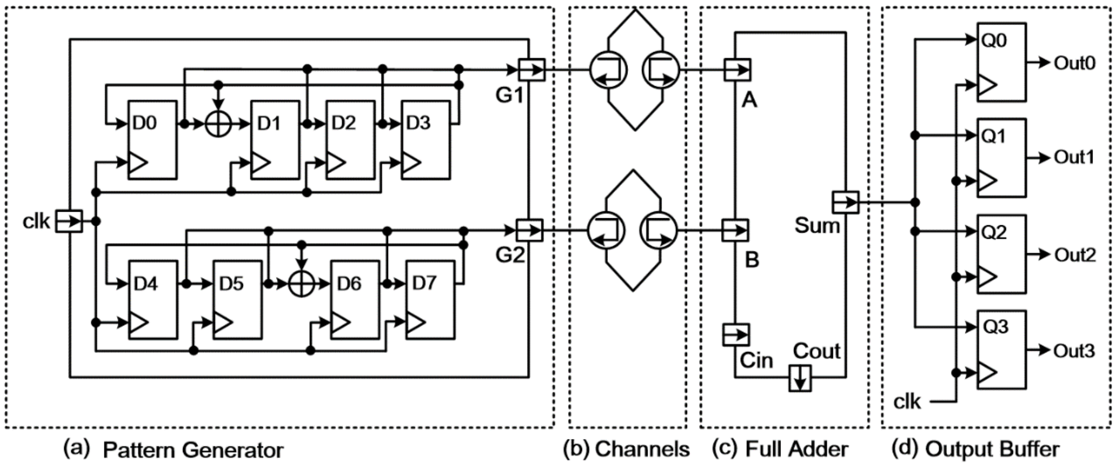
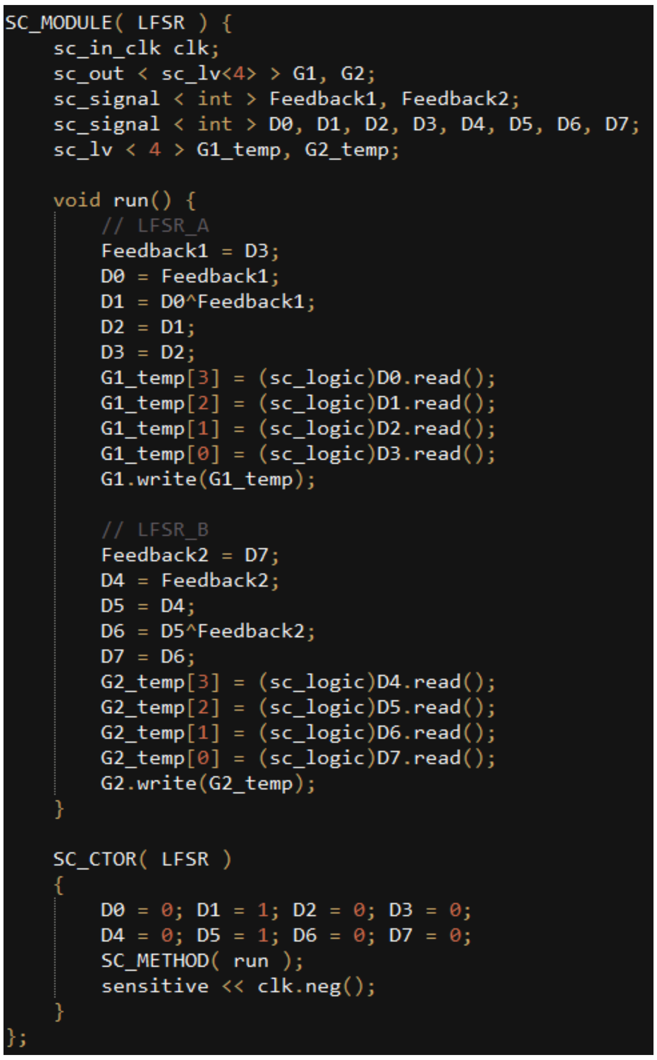


Fig. 2

Answer:

(a)



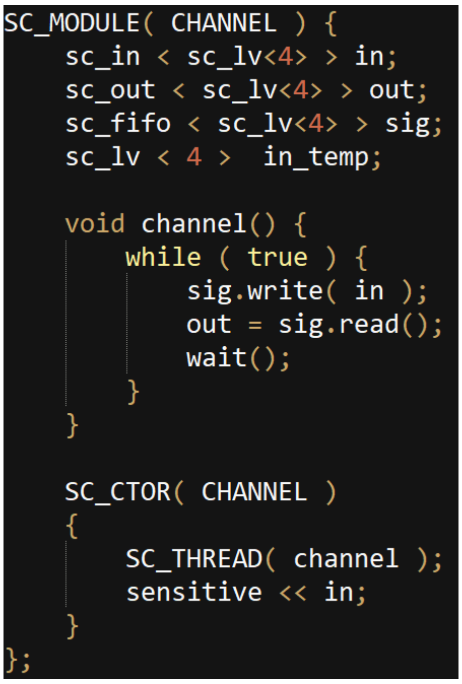
```
SC_MODULE( LFSR ) {
    sc_in_clk clk;
    sc_out < sc_lv<4> > G1, G2;
    sc_signal < int > Feedback1, Feedback2;
    sc_signal < int > D0, D1, D2, D3, D4, D5, D6, D7;
    sc_lv < 4 > G1_temp, G2_temp;

    void run() {
        // LFSR_A
        Feedback1 = D3;
        D0 = Feedback1;
        D1 = D0^Feedback1;
        D2 = D1;
        D3 = D2;
        G1_temp[3] = (sc_logic)D0.read();
        G1_temp[2] = (sc_logic)D1.read();
        G1_temp[1] = (sc_logic)D2.read();
        G1_temp[0] = (sc_logic)D3.read();
        G1.write(G1_temp);

        // LFSR_B
        Feedback2 = D7;
        D4 = Feedback2;
        D5 = D4;
        D6 = D5^Feedback2;
        D7 = D6;
        G2_temp[3] = (sc_logic)D4.read();
        G2_temp[2] = (sc_logic)D5.read();
        G2_temp[1] = (sc_logic)D6.read();
        G2_temp[0] = (sc_logic)D7.read();
        G2.write(G2_temp);
    }

    SC_CTOR( LFSR )
    {
        D0 = 0; D1 = 1; D2 = 0; D3 = 0;
        D4 = 0; D5 = 1; D6 = 0; D7 = 0;
        SC_METHOD( run );
        sensitive << clk.neg();
    }
};
```

(b)

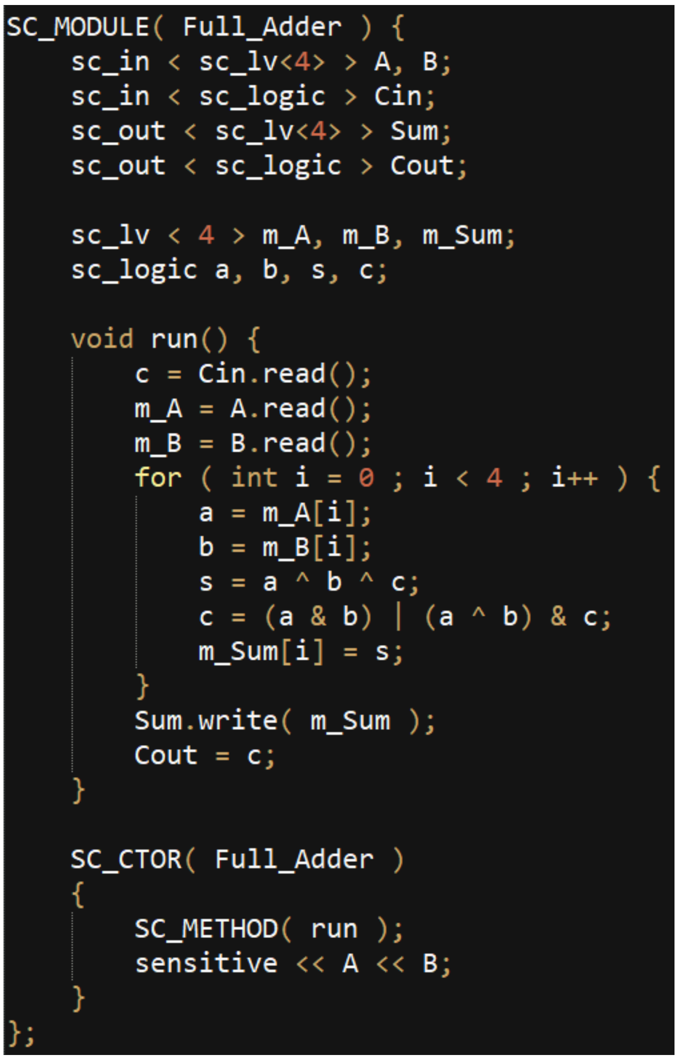


```
SC_MODULE( CHANNEL ) {
    sc_in < sc_lv<4> > in;
    sc_out < sc_lv<4> > out;
    sc_fifo < sc_lv<4> > sig;
    sc_lv < 4 >  in_temp;

    void channel() {
        while ( true ) {
            sig.write( in );
            out = sig.read();
            wait();
        }
    }

    SC_CTOR( CHANNEL )
    {
        SC_THREAD( channel );
        sensitive << in;
    }
};
```

(c)



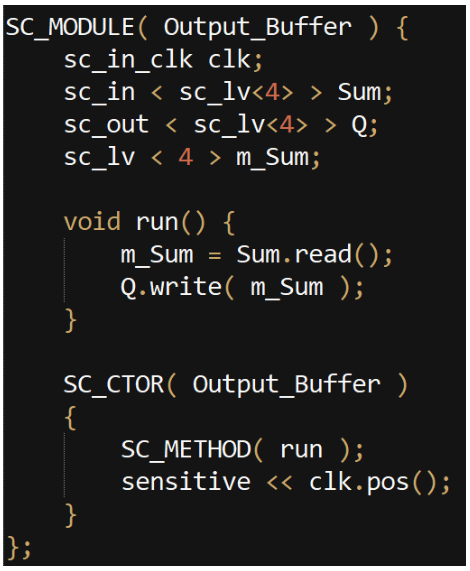
```
SC_MODULE( Full_Adder ) {
    sc_in < sc_lv<4> > A, B;
    sc_in < sc_logic > Cin;
    sc_out < sc_lv<4> > Sum;
    sc_out < sc_logic > Cout;

    sc_lv < 4 > m_A, m_B, m_Sum;
    sc_logic a, b, s, c;

    void run() {
        c = Cin.read();
        m_A = A.read();
        m_B = B.read();
        for ( int i = 0 ; i < 4 ; i++ ) {
            a = m_A[i];
            b = m_B[i];
            s = a ^ b ^ c;
            c = (a & b) | (a ^ b) & c;
            m_Sum[i] = s;
        }
        Sum.write( m_Sum );
        Cout = c;
    }

    SC_CTOR( Full_Adder )
    {
        SC_METHOD( run );
        sensitive << A << B;
    }
};
```

(d)



```
SC_MODULE( Output_Buffer ) {
    sc_in_clk clk;
    sc_in < sc_lv<4> > Sum;
    sc_out < sc_lv<4> > Q;
    sc_lv < 4 > m_Sum;

    void run() {
        m_Sum = Sum.read();
        Q.write( m_Sum );
    }

    SC_CTOR( Output_Buffer )
    {
        SC_METHOD( run );
        sensitive << clk.pos();
    }
};
```

(A) (5%)

```
SC_MODULE( Hello_SystemC_A ) {
        sc_in < sc_logic > x1, x2;
        sc_out < sc_bit > x_out;
        void run() {
                while ( true ) {
                        if ( x1 == 1 )
                                x_out = x2;
                        else
                                x_out = ‘1’;
                }
        }
        SC_CTOR( Hello_SystemC_A )
        {
                SC_METHOD( run, x1, x2 );
        }
};
```

Answer:

```
SC_MODULE( Hello_SystemC_A ) {
        sc_in < sc_logic > x1, x2;
        sc_out < sc_bit > x_out;
        void run() {
                if ( x1 == 1 )
                        x_out = (sc_bit)x2;
                else
                        x_out = (sc_bit)’1’;
        }
        SC_CTOR( Hello_SystemC_A )
        {
                SC_METHOD( run );
                Sensitive << x1 << x2;
        }
};
```

(B) (5%)

```
SC_MODULE( Hello_SystemC_B ) {
        sc_in < sc_bit > x1;
        sc_out < int > x_out;
        void run() {
                if ( x1 == ‘z’ )
                        x_out = 5;
                else
                        x_out = 10;
        }
        SC_CTOR( Hello_SystemC_B )
        {
                SC_THREAD( run );
        }
};
```

Answer:

```
SC_MODULE( Hello_SystemC_B ) {
        sc_in < sc_logic > x1;
        sc_out < int > x_out;
        void run() {
                while ( true ) {
                        if ( x1 == ‘z’ )
                                x_out = 5;
                        else
                                x_out = 10;
                        wait();
                }
        }
        SC_CTOR( Hello_SystemC_B )
        {
                SC_THREAD( run );
                sensitive << x1;
        }
};
```

Problem 4: Shift register (20 points)

Fig. 3 shows a shift register, triggered by the positive edge of the clock.

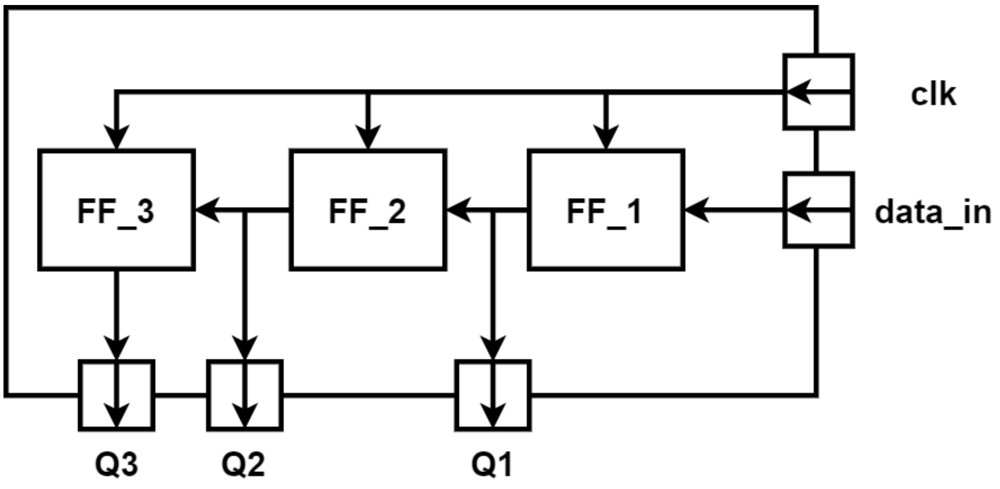
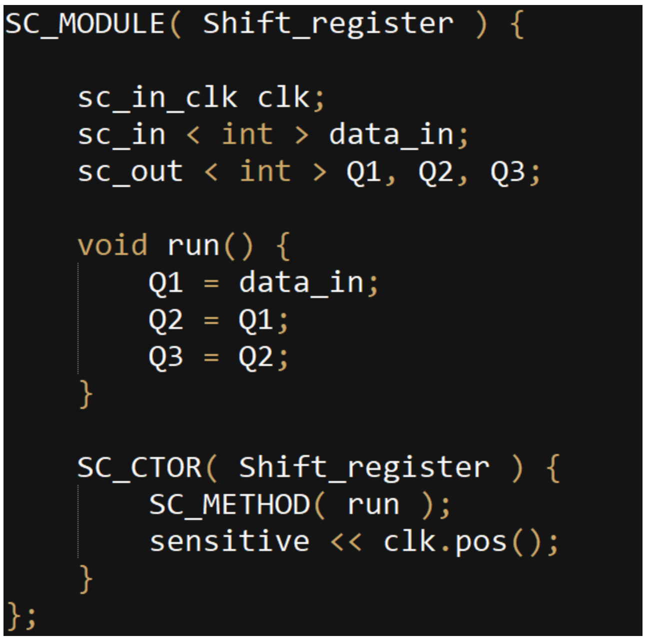


Fig. 3

1. (10%) Please provide a SystemC code to describe the module of shift register by use SC_METHOD.
2. (10%) Please provide a SystemC code to describe the module of shift register by use SC_THREAD.

Answer:

(1)



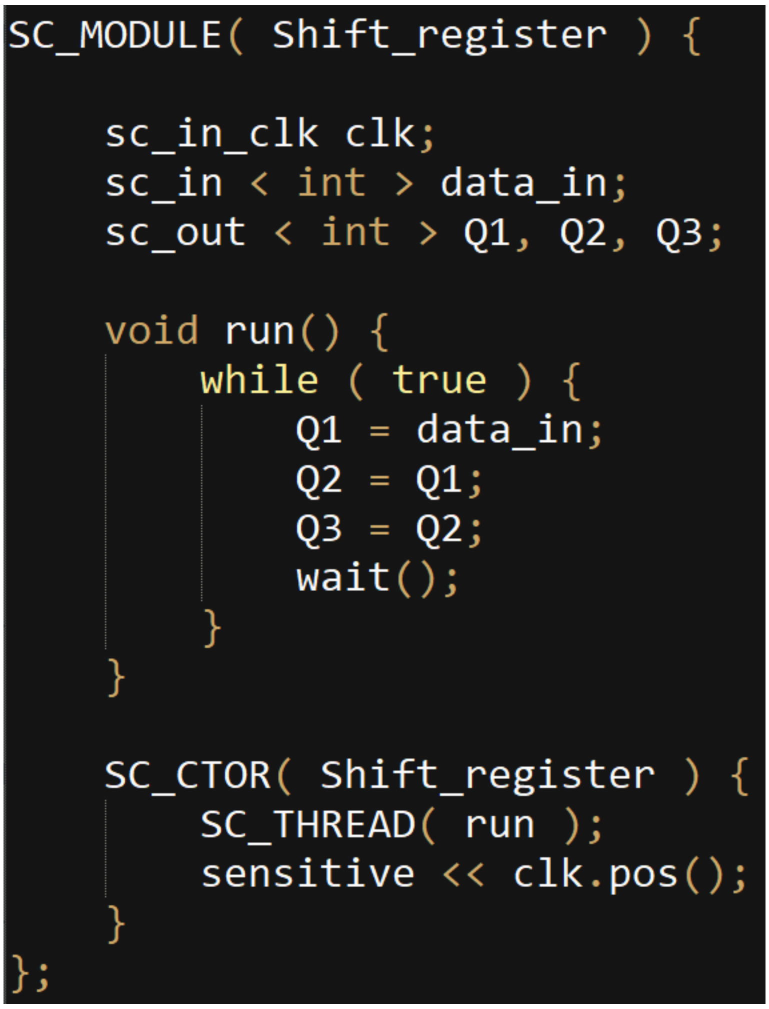
```
SC_MODULE( Shift_register ) {

    sc_in_clk clk;
    sc_in < int > data_in;
    sc_out < int > Q1, Q2, Q3;

    void run() {
        Q1 = data_in;
        Q2 = Q1;
        Q3 = Q2;
    }

    SC_CTOR( Shift_register ) {
        SC_METHOD( run );
        sensitive << clk.pos();
    }
};
```

(2)



```
SC_MODULE( Shift_register ) {

    sc_in_clk clk;
    sc_in < int > data_in;
    sc_out < int > Q1, Q2, Q3;

    void run() {
        while ( true ) {
            Q1 = data_in;
            Q2 = Q1;
            Q3 = Q2;
            wait();
        }
    }

    SC_CTOR( Shift_register ) {
        SC_THREAD( run );
        sensitive << clk.pos();
    }
};
```

Problem 5: (10 points)

1. (5%) Please explain the definitions of 1) Deadlock, 2) Livelock, and (3) Starvation problems in NoC systems and describe the situations, caused these problems.
2. (5%) Please explain the definitions of 1) Deterministic routing and 2) Adaptive routing and analyze the tradeoff of the two routing algorithms design.

Answer:

(1) Deadlock:

Deadlock: A packet does not reach its destination, because it is blocked at some intermediate resource.

(2) Livelock:

Livelock: A packet does not reach its destination, because it enters a cyclic path.

(3) Starvation:

Starvation: A packet does not reach its destination, because some resource does not grant access (wile it grants access to other packets).

(4) Deterministic routing:

Deterministic algorithms are simple and inexpensive, but they do utilize path diversity and thus are weak on load balancing.

(5) Adaptive routing:

Adaptive algorithms although in theory superior, are complex and power hungry.

Problem 7: (10 points)

Fig. 4 is a 4-by-4 mesh-based NoC. If the source node is *Src* and the destination node is *Dst*. Besides, we use minimal fully adaptive routing algorithm in this case. How many paths are there from the source node to the destination node?

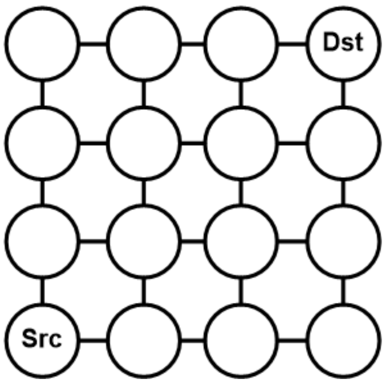
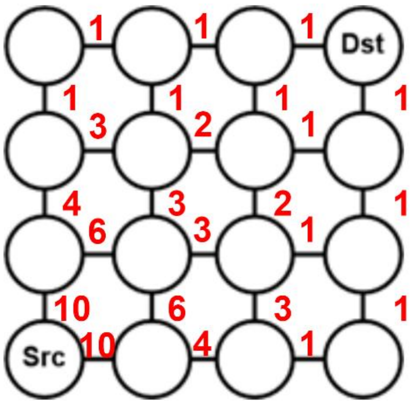


Fig. 4

Answer:

20 paths


Dst
Src
1
1
1
1
1
1
1
3
2
1
4
3
2
1
6
3
1
10
6
3
1
10
4
1