Project: Red or Green

Aim:

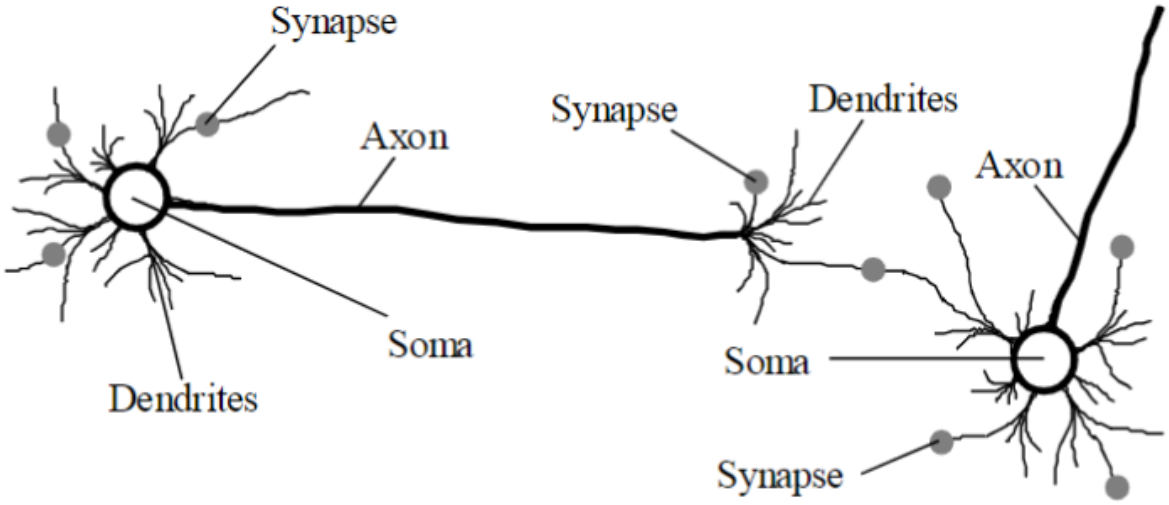
To predict the next color will be click by the user

Background:

To make machine predict like any intelligent human being that machine need to be train or to have knowledge that able to think like human being. As The goal of **artificial intelligence (AI)** as a science is to make machines do things that would require is to make machines do things that would require intelligence if done by humans.

There are numerous of method/ways to make machine think like human. The most popular way is **Neural network**. Neural network is based on the system of human biological neural network (human brain).

The brain consists of a densely interconnected set of nerve cells, or basic information-processing units, called **neurons** with the connections, **synapses** between them. A neuron consists of a cell body, **soma**, a number of fibers called



dendrites, and a single long fiber called **axon**.

Figure 1: Biological neural network

For neural network in machine, they have the same structure except instead of using biologically view, we use blocks, lines, numbers, mathematically view on those units.

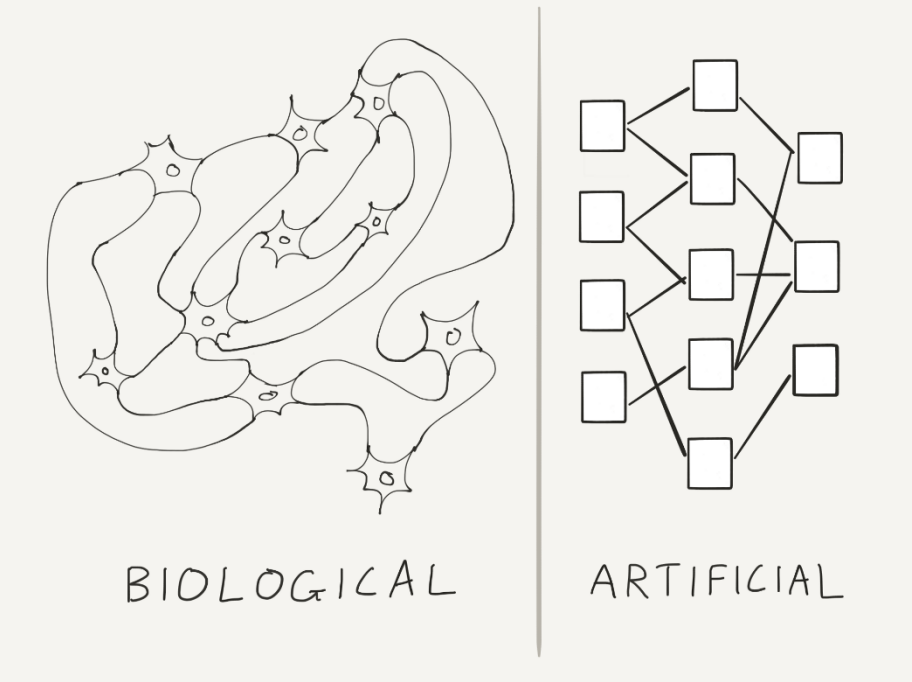


Figure 2: Neural network representation between biological and artificial

Design:

For our case, our neural network looks like this: (To know how this diagram designed, read appendix)

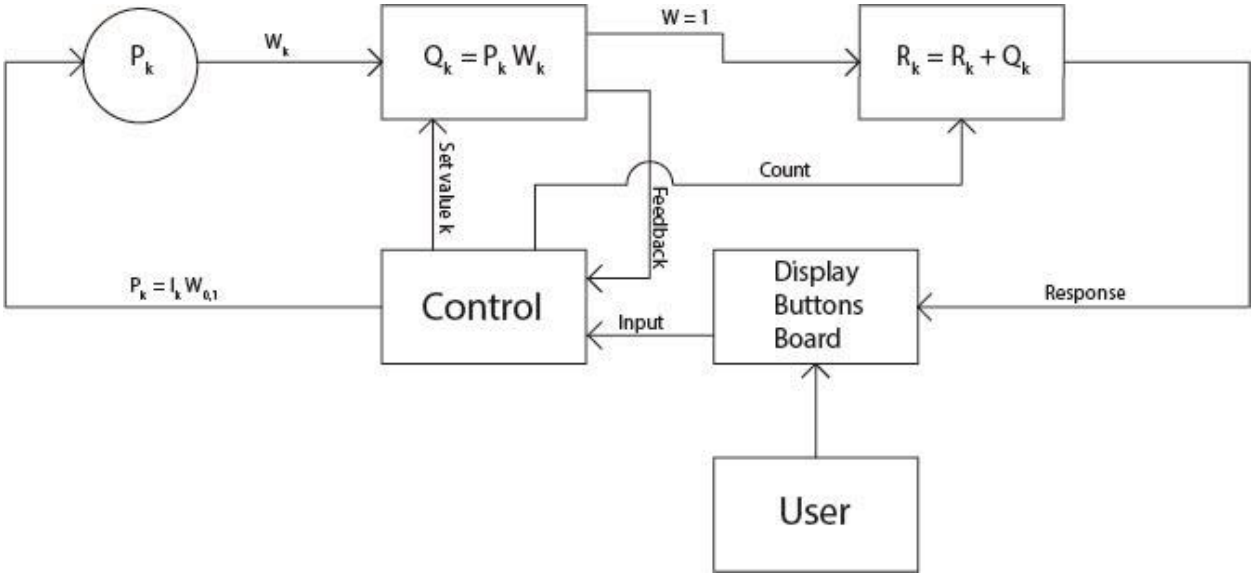
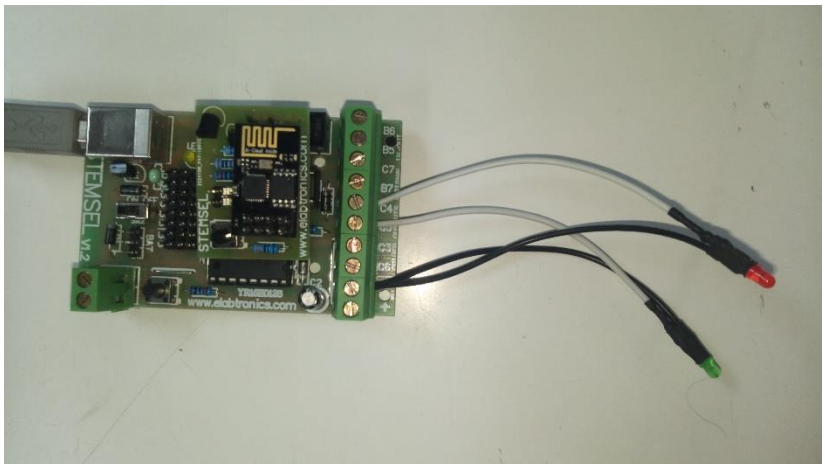


Figure 3: Block diagram of Red-or-Green neural network

Setup:



On STEMSEL board, put red LED in C4, and green LED in C5.

Figure 4: STEMSEL board with red and green LED

On runlinc control page, assign “Red” in C4, and “Green” in C5.

Codes:

HTML:

```
<div>
   <h1>Red or Green</h1>
   <button style="background-color: rgb(255, 103, 103)" onclick="play('red')">Red</button>
   <button style="background-color: rgb(118, 255, 118)"
onclick="play('green')">Green</button>
   <button onclick="play('default')">Reset</button>
</div>
<br>
<laber>Sequence: </laber><output id="seq">-----</output><br>
<laber>Predict next color will be </laber><output id="predict">-----</output><br><br><br>
```

Javascript:

```
var n = 8;
   var sequence = [];
   var count = 0;

   function play(colour) {
      if (sequence.length > 7) {
         sequence.shift();
      }
      switch (colour) {
         case 'red':
            sequence.push("1");
            calculation();
            break;

         case 'green':
            sequence.push("0");
            calculation();
            break;

         default:
            weight = 0;
            sequence = [];
            document.getElementById("seq").innerHTML = "-----";
            document.getElementById("predict").innerHTML = "-----";
            turnOff(Red);
            turnOff(Green);
```

```
            break;
        }
    }

    //Basic machine learning
    function calculation() {
        var input = [];
        var weightEach = [];
        var weight = 0;
        var SUM = 0;
        var DEC = 0;
        for (var i = 0; i < sequence.length; i++) {
            if (sequence[i] == 0) {
                input[i] = 0.1;
            }
            if (sequence[i] == 1) {
                input[i] = 1;
                count++;
            }
            SUM = SUM + sequence[i] * Math.pow(10, 7 - i);
            DEC = SUM.toString(10);

            weightEach[i] = (input[i] / ((1 / DEC) + n));

            weight += weightEach[i];
        }
        document.getElementById("seq").innerHTML = sequence.join(" ");
        prediction(weight);
    }

    //AI facts method
    function prediction(pred) {
        if (sequence.length < 7) {
            document.getElementById("predict").innerHTML = "-----";
            turnOff(Red);
            turnOff(Green);
        } else {
            if (pred >= 0.5499999931763907) {
                document.getElementById("predict").innerHTML = "red";
                turnOn(Red);
                turnOff(Green);
            }
            if (pred <= 0.5374999333107895) {
                document.getElementById("predict").innerHTML = "green";
```

```
            turnOff(Red);
            turnOn(Green);
        }
        if (pred == 0.6624999923295869) {
            document.getElementById("predict").innerHTML = "green";
            turnOff(Red);
            turnOn(Green);
        }
        if (pred == 0.41249951870961277) {
            document.getElementById("predict").innerHTML = "red";
            turnOn(Red);
            turnOff(Green);
        }
    }
}
```

Appendix:

The general architecture of an artificial neural network:

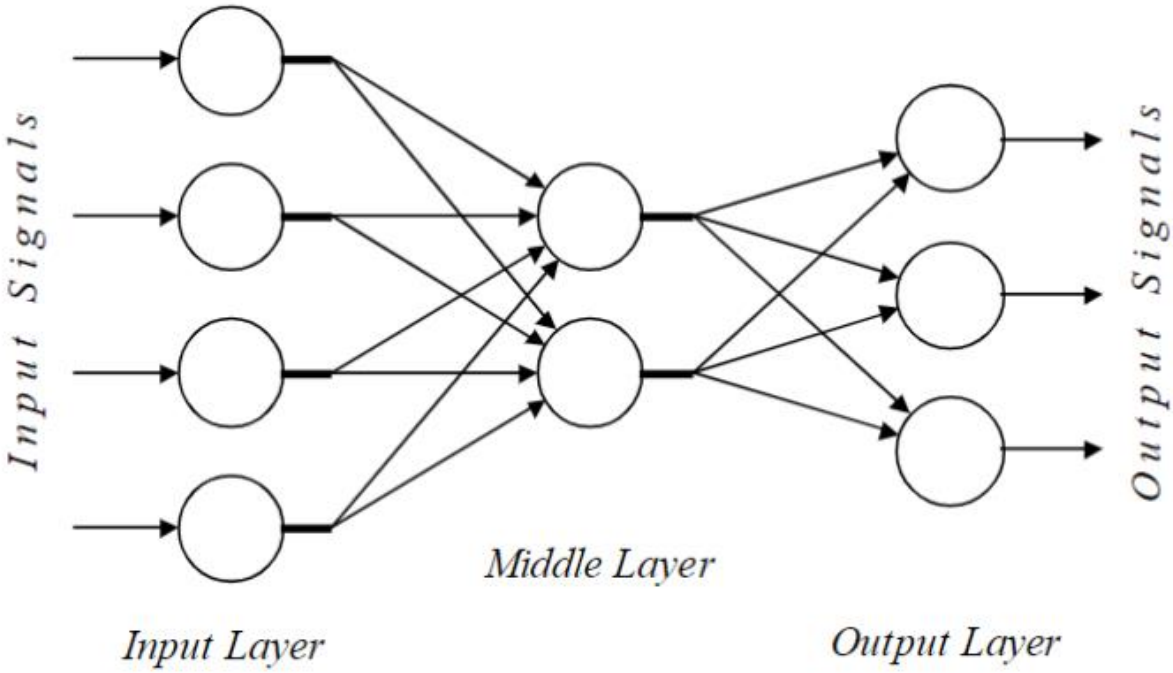


Figure A1: Architecture of an artificial neural network

To put it into analogy between biological and artificial neural networks, we have:

- Some as Neuron
- Dendrite as Input
- Axon as Output
- Synapse as Weight

And to simplify artificial neural network as a computing element, we have

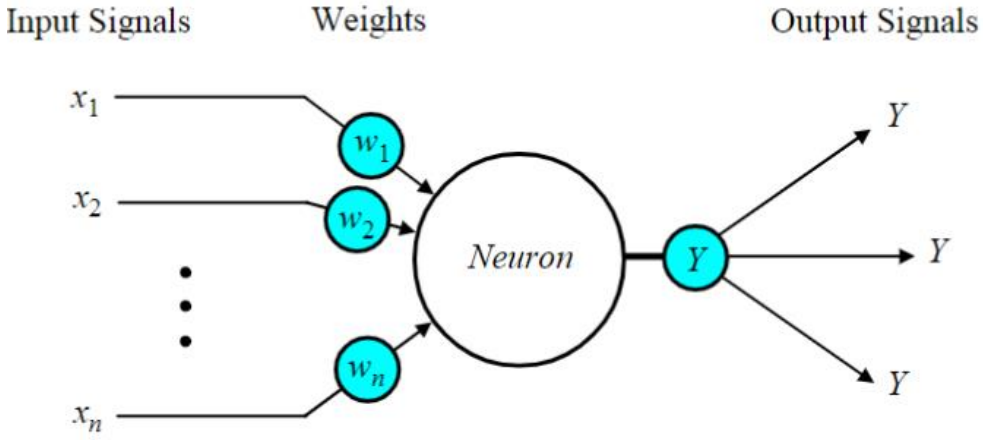


Figure A2: Diagram of a neuron

Before designing our block diagram, figure 3, we need to know our situations:

- 2 inputs with individual weight
- Total of 256 combination in our sequence, from 00000000 to 11111111 (8 bits)
- Each of the combination need to have difference weights
- Only record 8 bits

Let's start with 2 inputs with 2 bits. The possible combination of 2 bits, 0 and 1 are:

Decimal	A	B
0	0	0
1	0	1
2	1	0
3	1	1

Table 1: Possible combination of 2 bits

Therefore, in neural network:

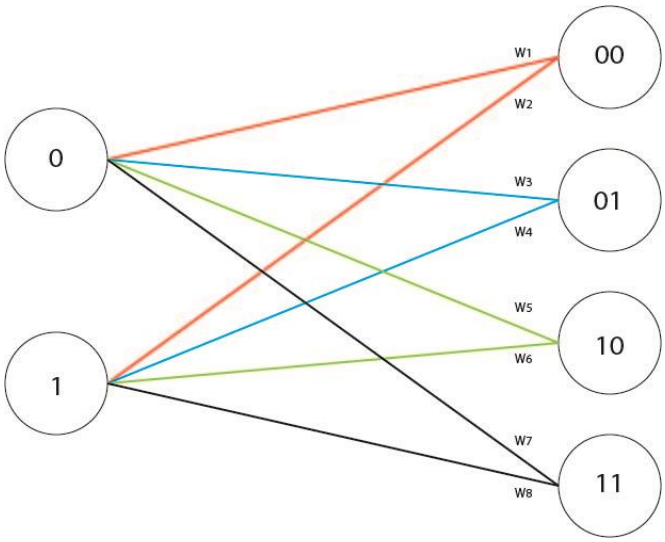


Figure A3: Neural network for 2 bits

Same method with 4 bits and 8 bits:

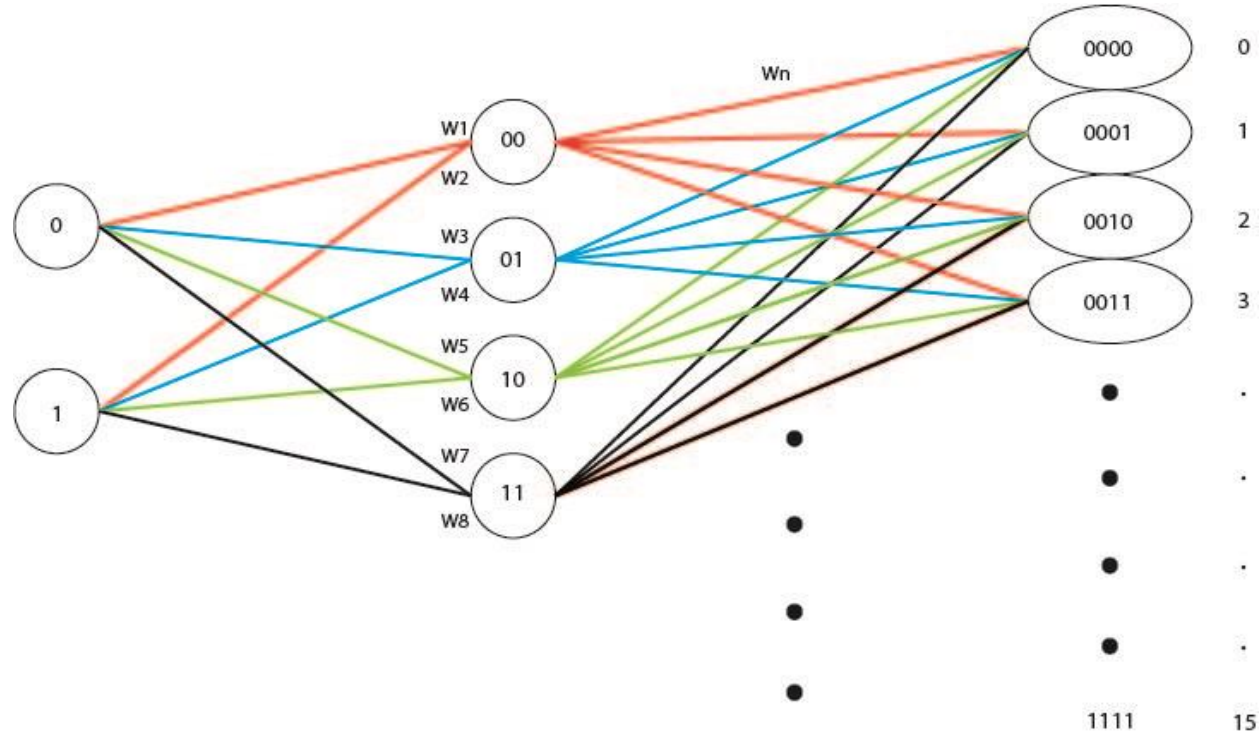


Figure A4: Neural network for 4 bits

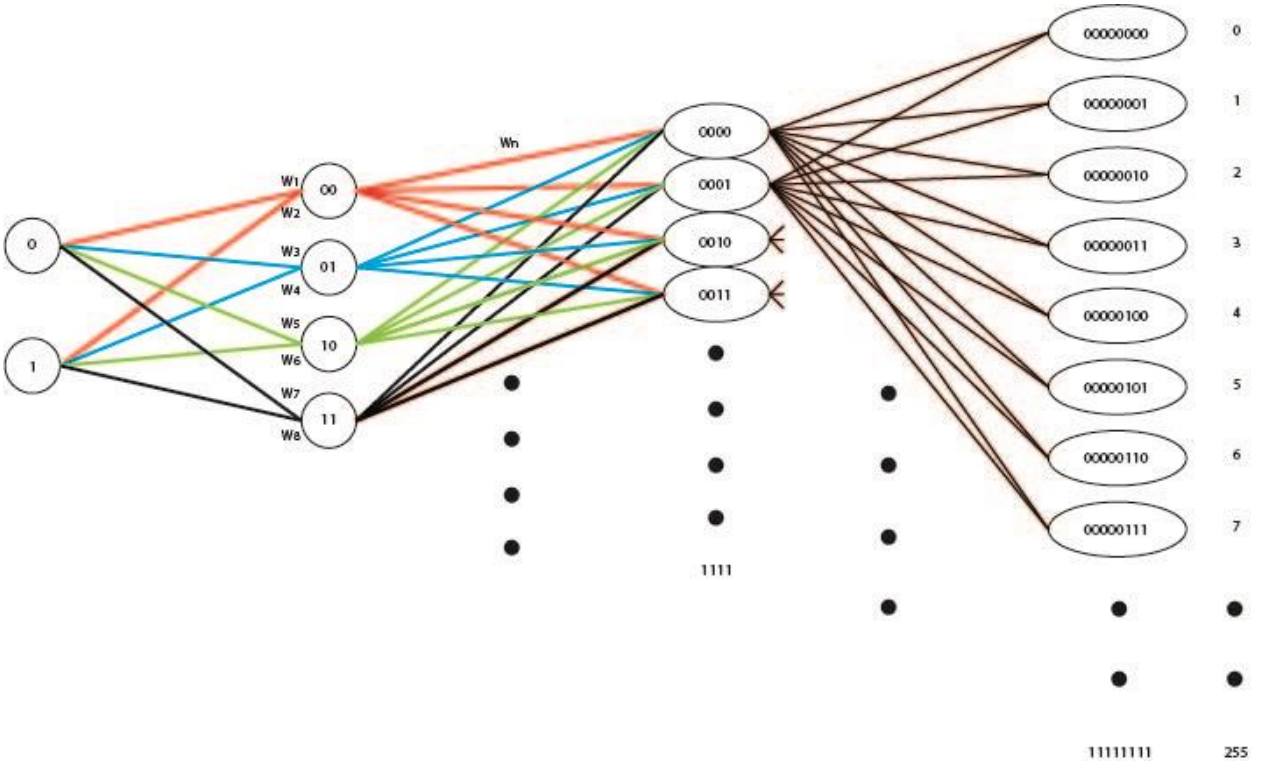


Figure A5: Neural network for 8 bits

Here comes the trick, we could split 8 bits into 4 parts, and it will make things easier. Example:

	AB (first part)	CD (second part)	EF (third part)	FG (forth part)
00000000	00	00	00	00
00000001				01
00000010				10
00000011				11
00000100			01	00
00000101				01
00000110				10
		.		
		.		
		.		
11111111	11	11	11	11

With this split, we could see each part are the combination of 2 bits and we can group them, and assign number, k for the order of part to let machine know which part go first towards last part. Therefore:

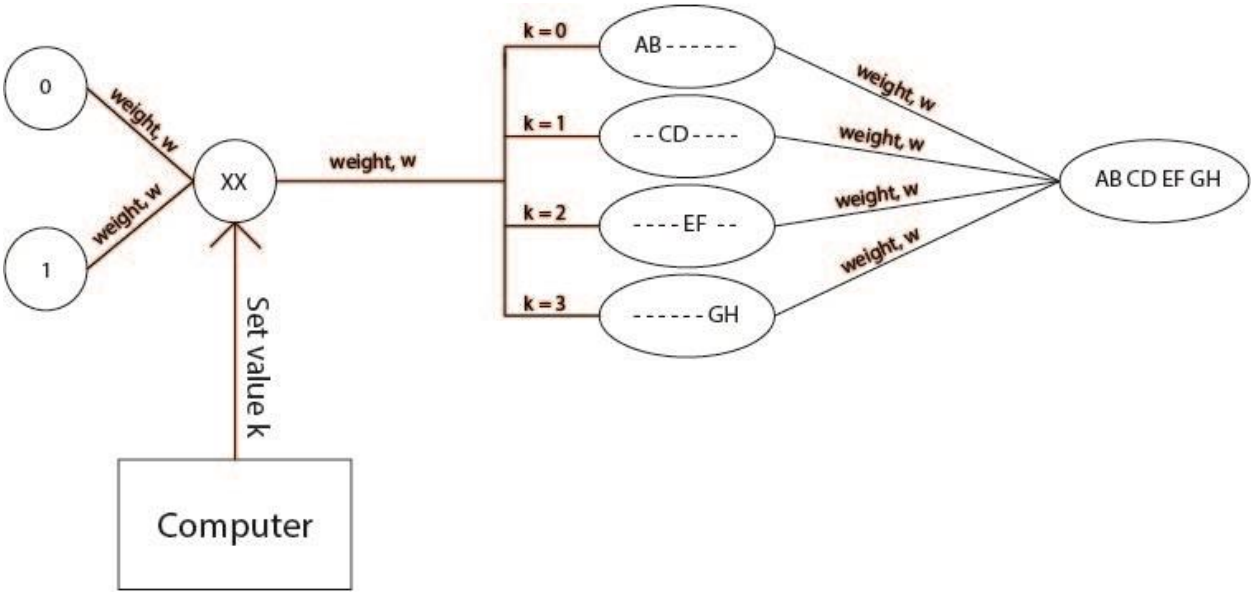


Figure A6: 8 bits into 4 parts block diagram

Furthermore, we can split them from 4 to 2 parts, that mean k count from 0 to 7.

	A	B	C	D	E	F	G	H
00000000	0	0	0	0	0	0	0	0
00000001	0	0	0	0	0	0	0	1
00000010	0	0	0	0	0	0	1	0
00000011	0	0	0	0	0	0	1	1
00000100	0	0	0	0	0	1	0	0
00000101	0	0	0	0	0	1	0	1
00000110	0	0	0	0	0	1	1	0
				.				
11111111	1	1	1	1	1	1	1	1

Finally, group them into a block, change computer to control, add user:

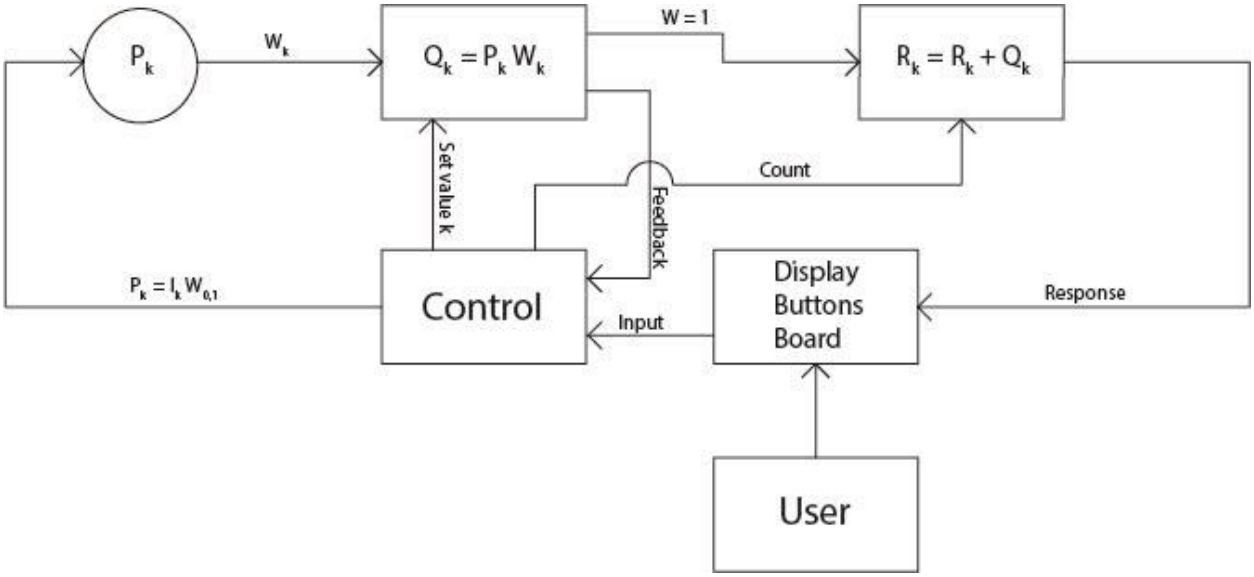


Figure 3: Block diagram of Red-or-Green neural network