中国科学技术大学计算机学院
《数字电路实验报告》

实验题目：综合实验

学生姓名：张展翔

学生学号：PB20111669

完成时间：2021.12.26

实验题目

综合实验

实验目的

熟练掌握前面实验中的所有知识点

熟悉几种常用通信接口的工作原理

及使用独立完成具有一定规模的功能电路设计

实验环境

Logisim

gcc and python

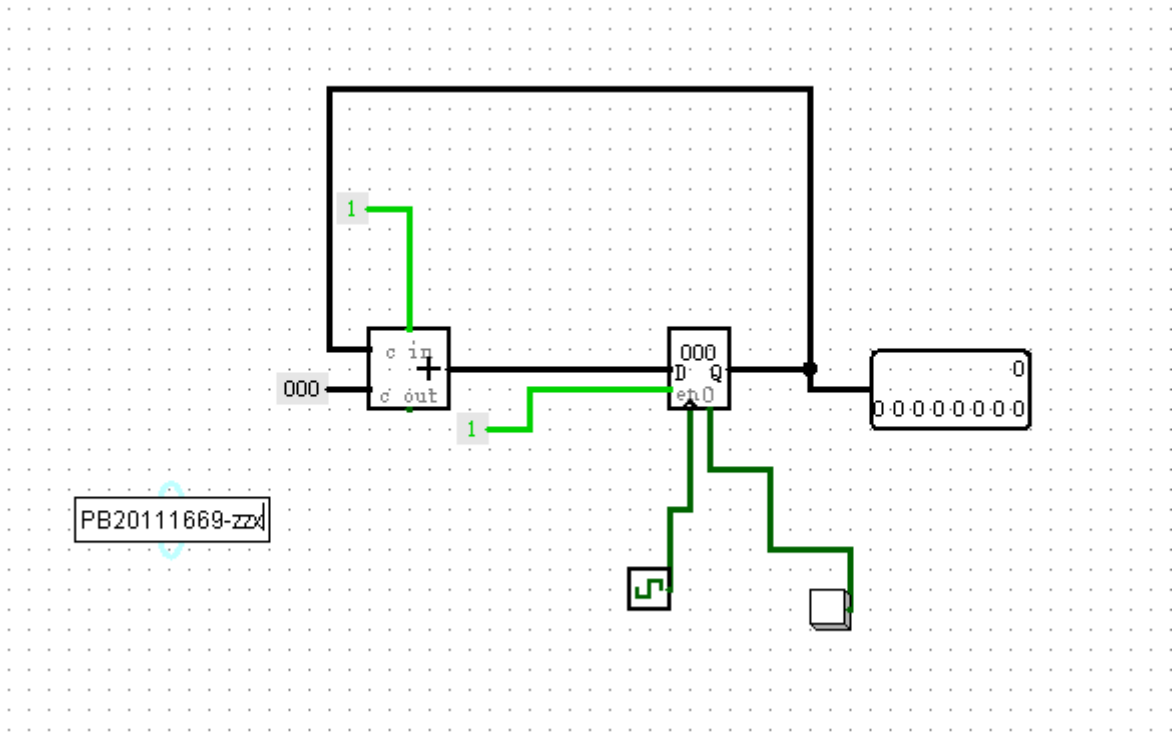
实验效果

本实验实现了由Logisim执行的汉字的循环滚动展示功能，并利用了Python的selenium库来自动获取汉字的点阵坐标，再利用所写程序计算每一个滚动状态的偏移量，从而实现自定义任意字段在Logisim的循环滚动功能。

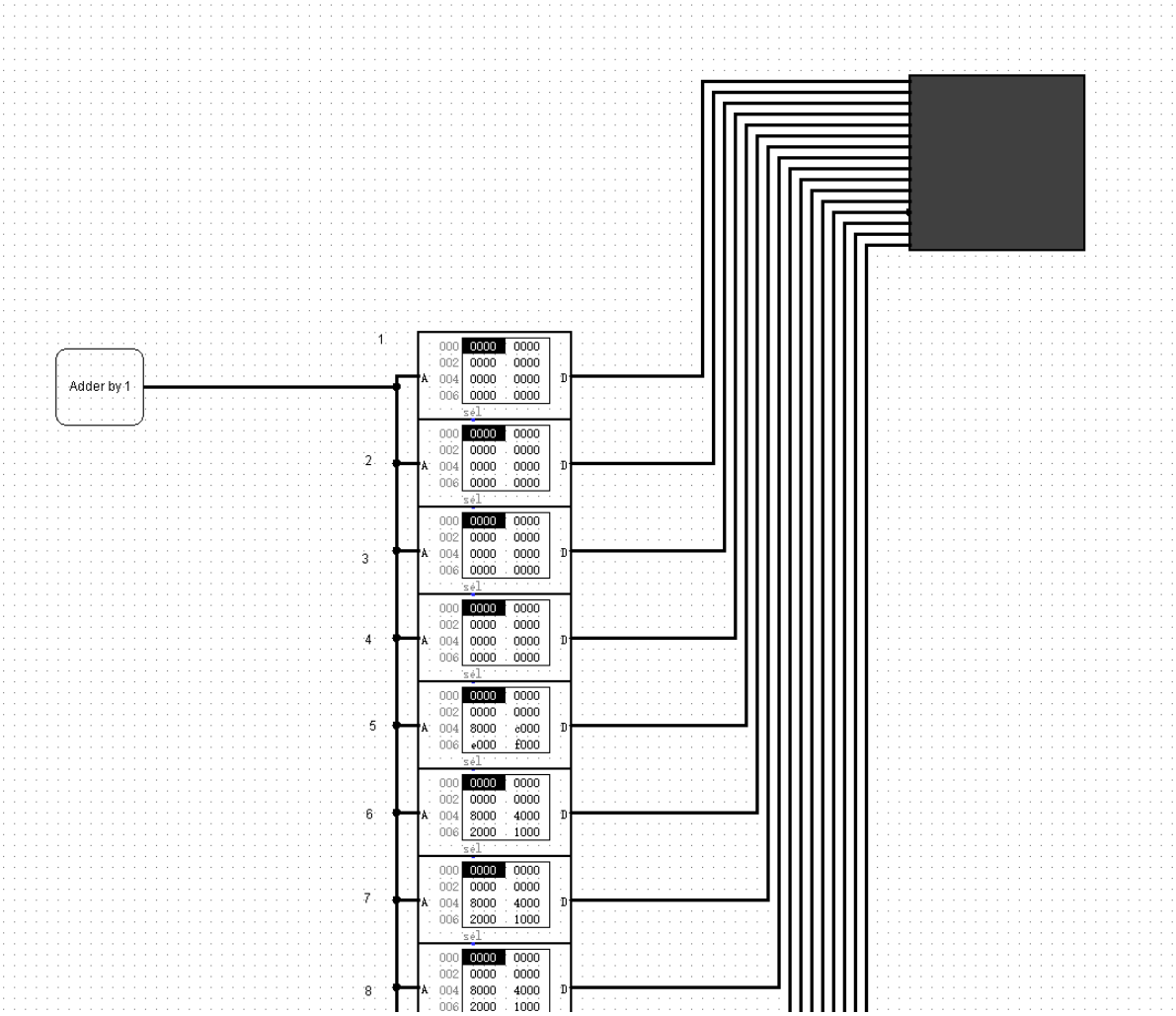
实验过程

一、电路设计

首先，利用Adder和Register原件设计一个9位的自增加法器，设计原理图如下：



然后，利用此加法器外接16个ROM原件来控制这16个ROM原件的状态变化，这16个ROM原件再分别与LED显示屏的16个接口相连来控制LED的亮灭情况。



那么如何实现不同汉字的循环显示呢?

注意到ROM中可以储存许多不同的状态，故可利用每一个状态来表示汉字不同的位置，从而实现汉字的循环滚动功能。

二、循环状态值的计算

注意到ROM有许多不同的状态值，每一个状态都对应着不同的led显示情况，要实现滚动，就要实现每一个状态为上一个状态整体向右平移一位。

经计算与观察可得，某状态所对应的16进制数值除以2即为整体向右平移1格所对应的下一个状态的16进制数值，乘2即为左移一个下一个状态所对应的16进制数值。

下面为我所写出的计算每一个16进制滚动各个左右平移状态数值计算所对应的程序

```
void left(int x)
{
    int t[16] = {0};
    x *= 2;
    for (int i = 0; i < 16; i++)
    {
        t[15 - i] = x;
        x *= 2;
    }
    for (int i = 0; i < 16; i++)
    {
        if (i == 8)
            cout << endl;
        cout << hex << t[i] << " ";
```

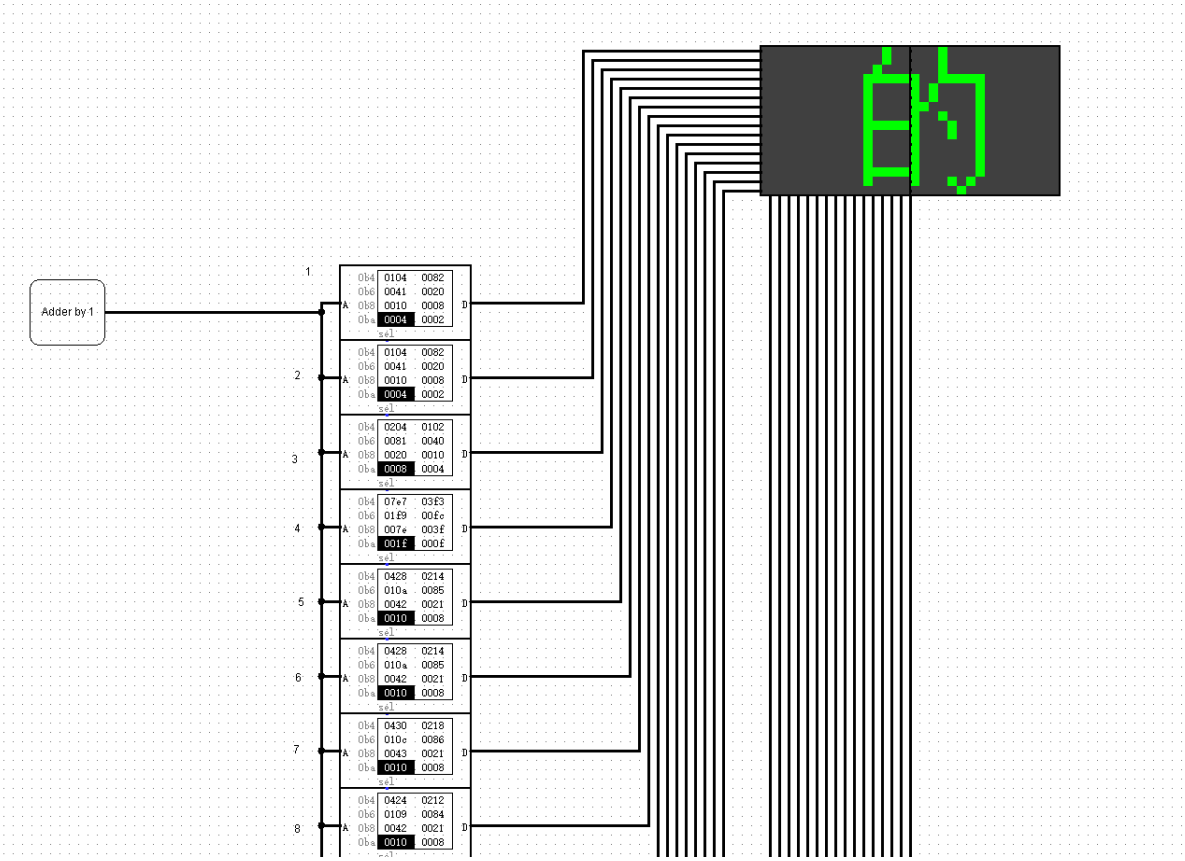
```
    }
    cout << endl;
}

void right(int x)
{
    for (int i = 0; i < 16; i++)
    {
        if (i == 8)
            cout << endl;
        cout /*<<setw(4)<<setfill('0')*/ << hex << x << " ";
        x /= 2;
        //x *= 2;
    }
    cout << endl;
}
```

三、多显示屏的扩展

注意到上述电路图只是用了一个LED显示屏，无法满足实际需求，因此我们可以通过增加ROM元件的数目来控制更多显示屏的加入。

要注意不同显示屏直接的衔接。



四、汉字字库点阵值的获取

助教在群内所发字库16进制点阵查询，故我采取爬虫技术来实现了获取每个汉字的16进制点阵

```
import re
from time import sleep
from typing import KeysView
from selenium import webdriver
import webbrowser
```

```
import os


filename="1616.html"



driver=webdriver.Chrome()
str=input("请输入所要显示的字段:")
driver.get('1616.html')
blank=driver.find_element_by_xpath("/html/body/div[1]/input")
blank.click()
blank.clear()
blank.click()
blank.send_keys(str)
sleep(2)
driver.find_element_by_id("btn").click()
hextext=driver.find_element_by_id("wrap")
print(hextext.text)
hex_all=re.findall(r".*16进制:(.*)2进制",hextext.text,re.S)
print(''.join(hex_all))
hex=hex_all[0].splitlines()
print(len(hex))
f=open('hexnum.txt','w')
for i in range(1,17):
    f.write(hex[i])
    f.write('\n')
f.close()
driver.close()
```

五、每个ROM状态值的计算

为了更快的获取每个状态所对应的16进制数值，我写出了如下程序来实现（~~状态数好多~~）

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <sstream>
using namespace std;

int Atoint(string s, int radix)
{
    int ans = 0;
    for (int i = 0; i < s.size(); i++)
    {
        char t = s[i];
        if (t >= '0' && t <= '9')
            ans = ans * radix + t - '0';
        else
            ans = ans * radix + t - 'a' + 10;
    }
    return ans;
}

int num_length(int n)
{
    int sum = 0;
```

```
    while (n)
    {
        sum++;
        n /= 10;
    }
    return sum;
}

void left(int x)
{
    int t[16] = {0};
    x *= 2;
    for (int i = 0; i < 16; i++)
    {
        t[15 - i] = x;
        x *= 2;
    }
    for (int i = 0; i < 16; i++)
    {
        if (i == 8)
            cout << endl;
        cout << hex << t[i] << " ";
    }
    cout << endl;
}

void right(int x)
{
    for (int i = 0; i < 16; i++)
    {
        if (i == 8)
            cout << endl;
        cout /*<<setw(4)<<setfill('0')*/ << hex << x << " ";
        x /= 2;
        //x *= 2;
    }
    cout << endl;
}
int main()
{
    string str[16];
    string line;
    ifstream file;
    file.open("C:/Users/Lenovo/Desktop/hexnum.txt");
    int i = 0;
    while (getline(file, line))
    {
        str[i++] = line;
    }
    int n = str->size() / 4;
    int x[18];
    for (int k = 0; k < n; k++)
    {
        int j = 0;
        string s;
        for (int i = 0; i < str->size(); i++)
        {
            if (i % 4 == 0 && i != 0)
```

```
            {
                x[j++] = Atoint(s, 16);
                s.clear();
            }
            s += str[k][i];
        }
        cout<<"ROM"<<k+1<<":"<<endl;
        for (int i = 0; i < 16; i++)
        {
            left_gun(x[i]);
            right_gun(x[i]);
        }
        cout<<endl;
    }
}
```

然后将上述程序所计算出的一串16进制数值替换掉.circ文件中所对应的数值即可

总结与思考

通过这次综合实验，我更加深入的了解了课堂实验中所学知识，并将所学知识更好的去应用到实验当中去，大大的提高了我对所学知识的综合运用能力，收获很大。