1. **Introduction:**

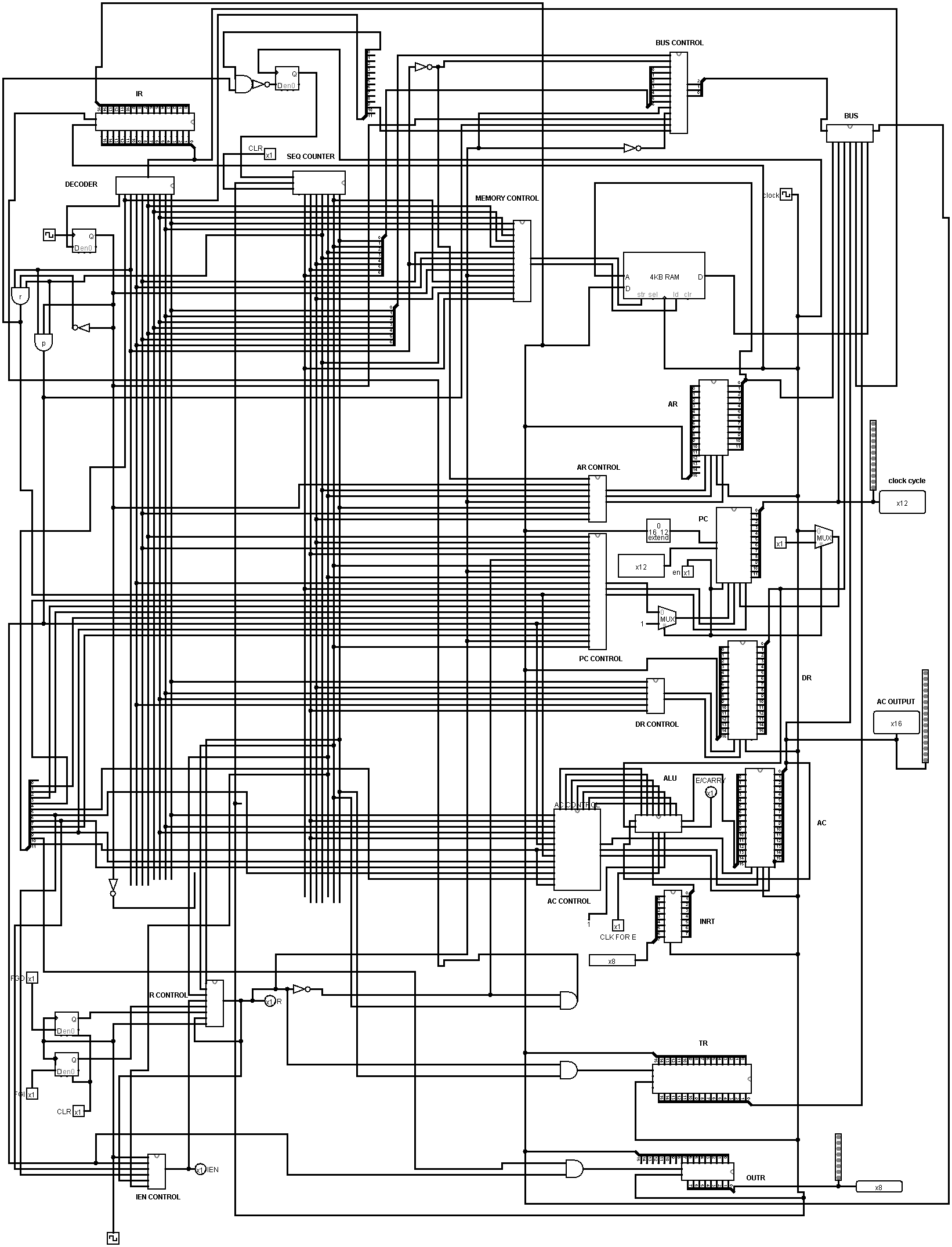
The purpose of this project is to make a completely working basic computer using Logisim open source software. The implemented is of a 16-bit basic computer with mainly eight different registers. Below, a demonstration of how the controls where derived will be shown along with how it all connects together to form one complete basic computer. The basis on which these controls and gates where connected comes from multiple figures found in the Morris Mano 3rd edition text book. Figure 5-3 gave us an idea on the size of each of the main registers while figure 5-4 was used to tell how it all connected together and whether each register has specific inputs such as increment or clear. Table 5-2 was used at the end of the project design, when the circuit was complete, to test different direct and indirect functions such as clear accumulator, circulate left of right. Tables 5-3, 5-4 and 5-5 where used to formulate logic equations, this was simply done by choosing a register, for instance AC and seeing what was the equivalent to it in input combinations. One example of AC’s formula is rB8 and rB9. After these steps where repeated for all registers, drawing on Logisim started. The Logisim circuit relies on bit by bit design to make it easier to understand. The problem was that it was more work but guaranteed. To speed up the drawing processes, things such as 1-bit ALU was made the duplicated to make a 2-bit ALU. The 2-bit ALU was used to make an 8-bit that finally led to making the 16-bit ALU used in our main circuit. The register chips where placed in the main after completion. Each register chip had it own number of input. The input of each register came from their respective control chip. The memory was loaded with instructions and values and an output is displayed at the LEDs.

1. **Registers:**

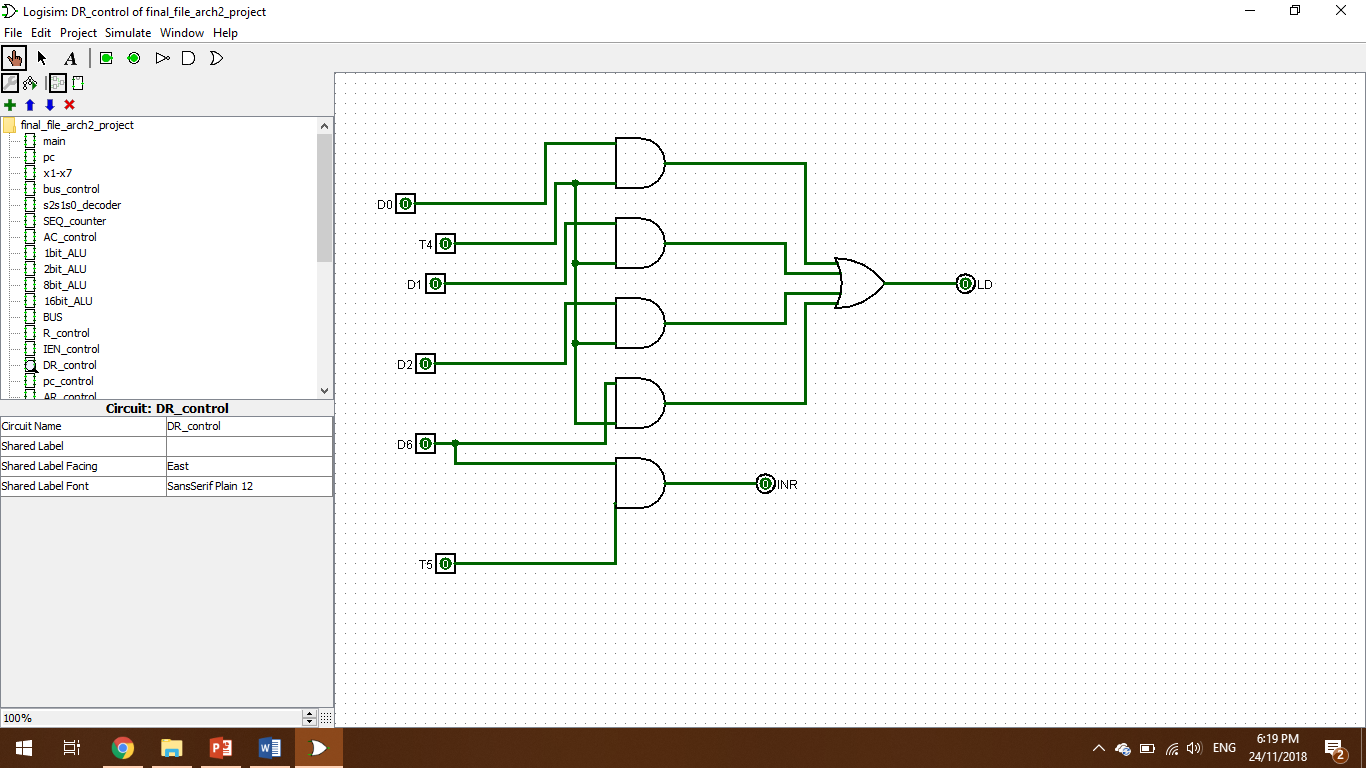
The main registers used where:

1. DR: 16-bits
2. AR: 12-bits
3. AC: 16-bits
4. IR: 16-bits
5. PC: 12-bits
6. TR: 16-bits
7. INPR: 8-bits
8. OUTR: 8-bits

The main challenge in this project was to get everything compatible with each other. This problem initially was caused by the difference in number of bits between registers.

1. **Registers and Derived Controls:**
2. **Main:**
3. **DR control:**

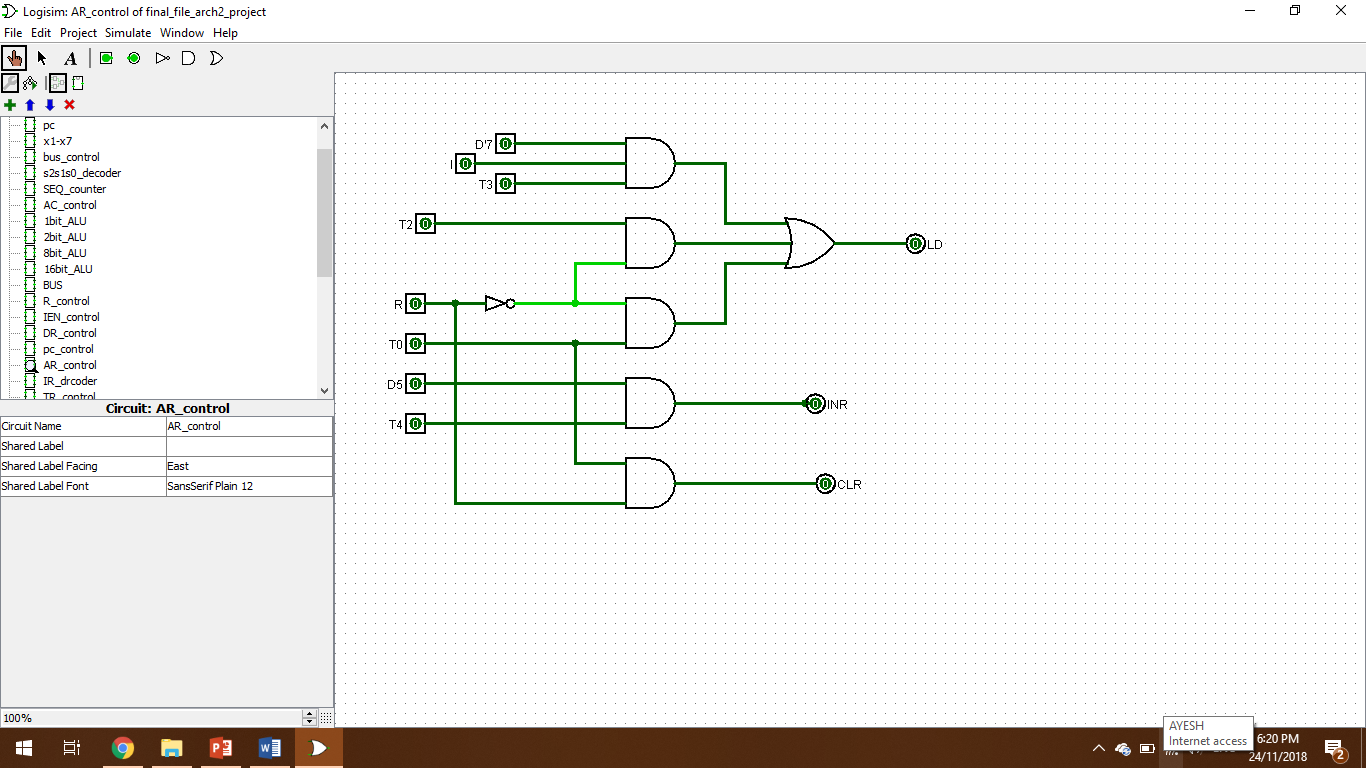
Load=D0.T4+D2.T4+D6.T4

Increment=D6.T5

1. **AR control:**

Load=R’.T0+R’.T2+D7’.I.T3

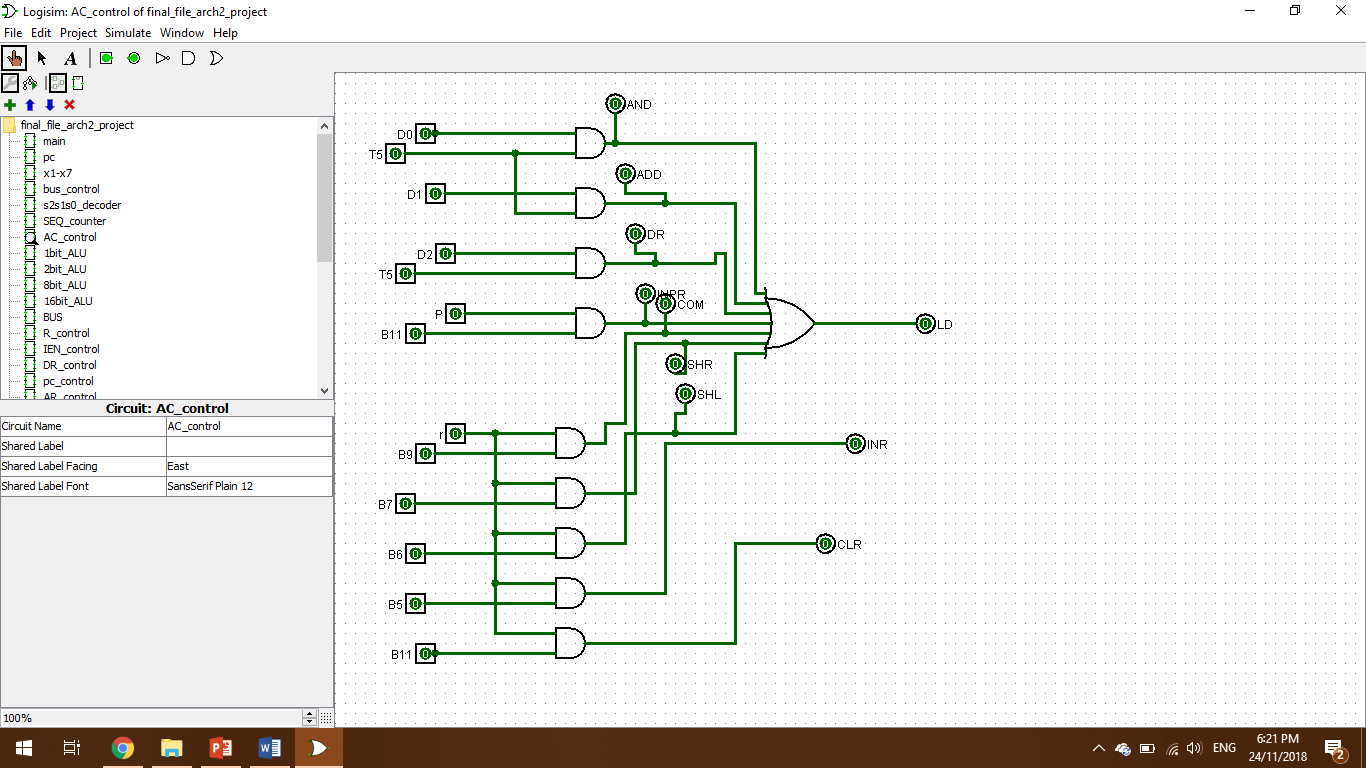
Clear=R.T0

INR=D5.T4

1. **AC control:**

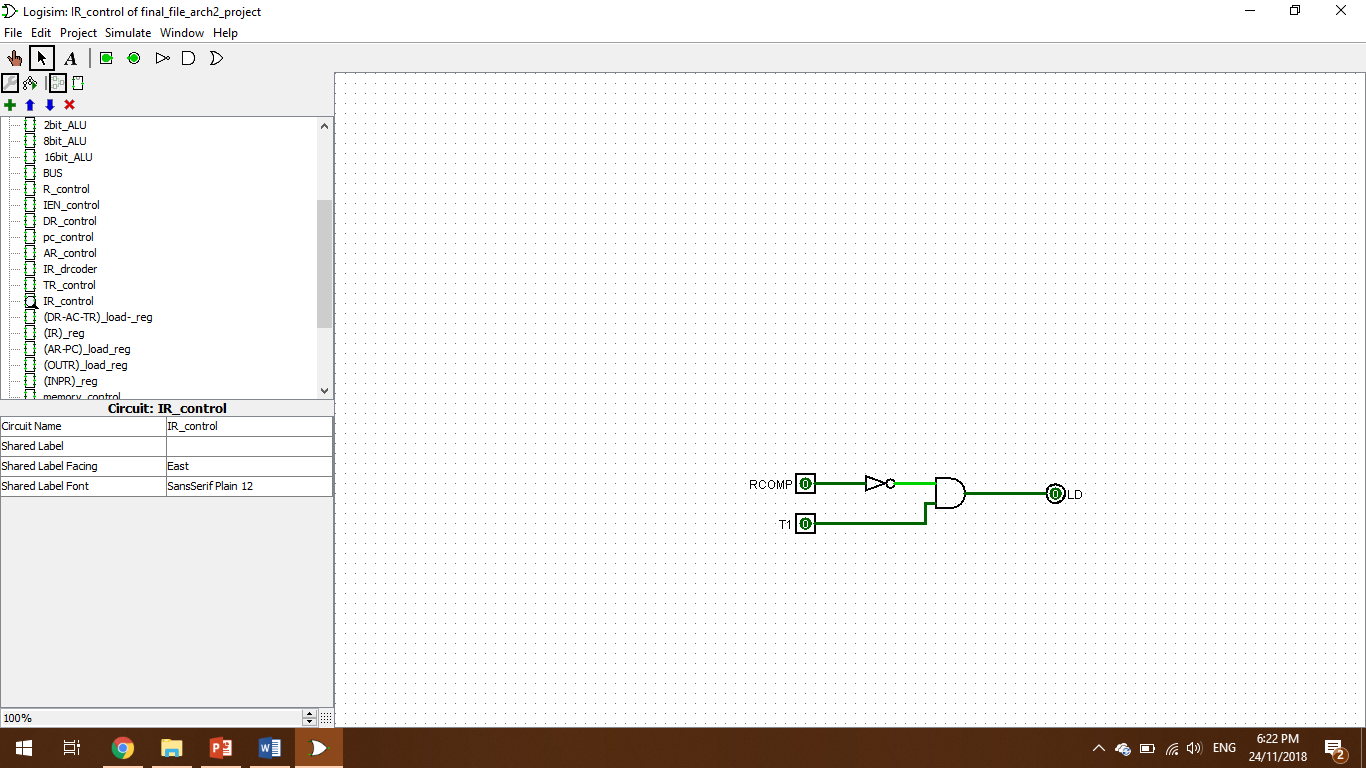
Load=D0T5+D1T5+D2T5+pB11+rB9+rB7+rB6

INR=rB5

Clear=rB11

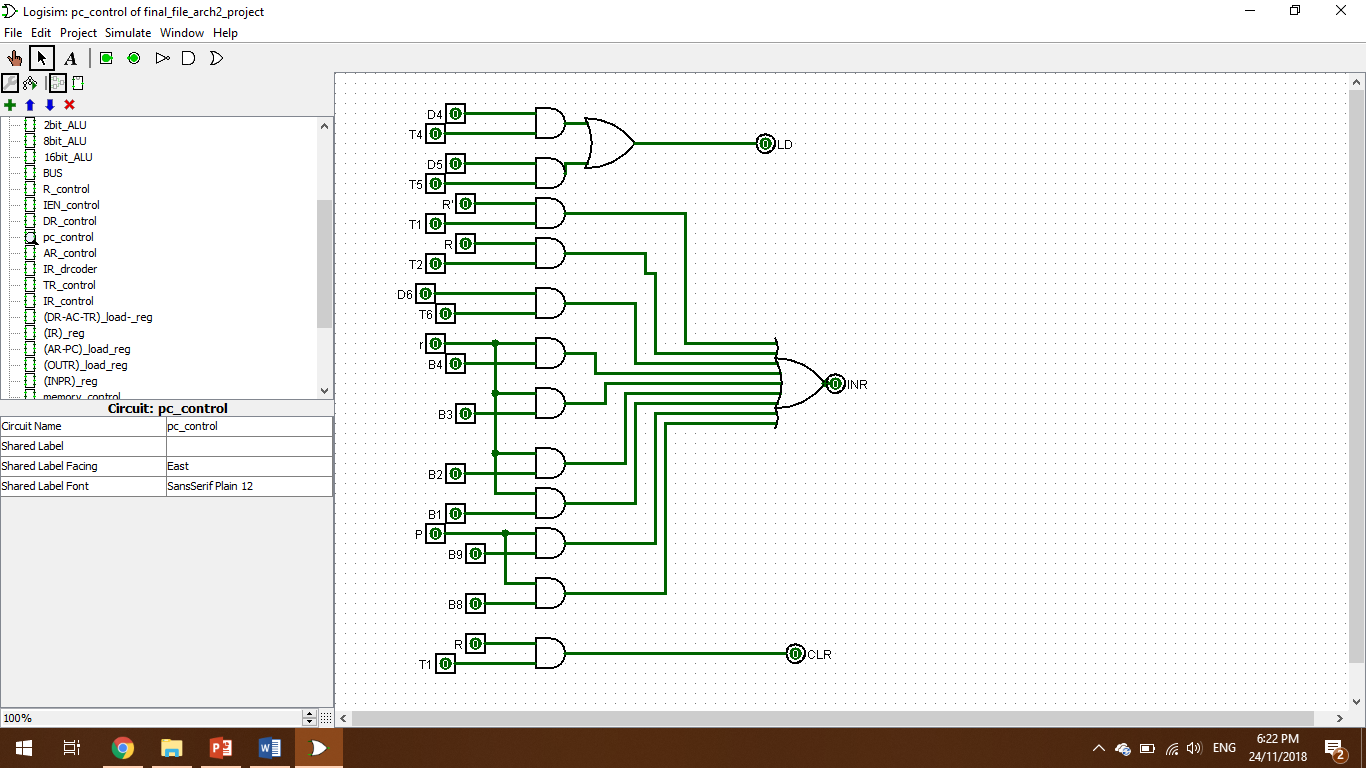
1. **IR control:**

Load=R’.T1



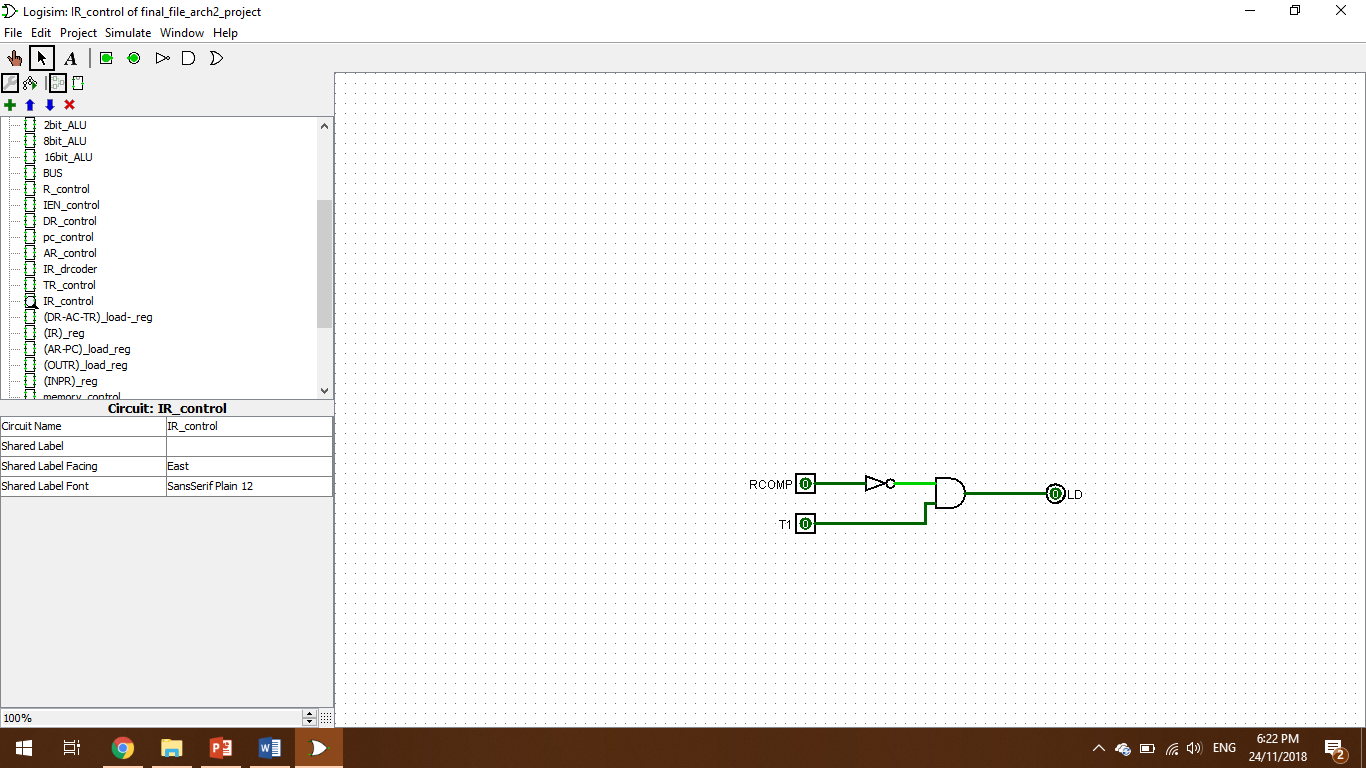
1. **PC control:**

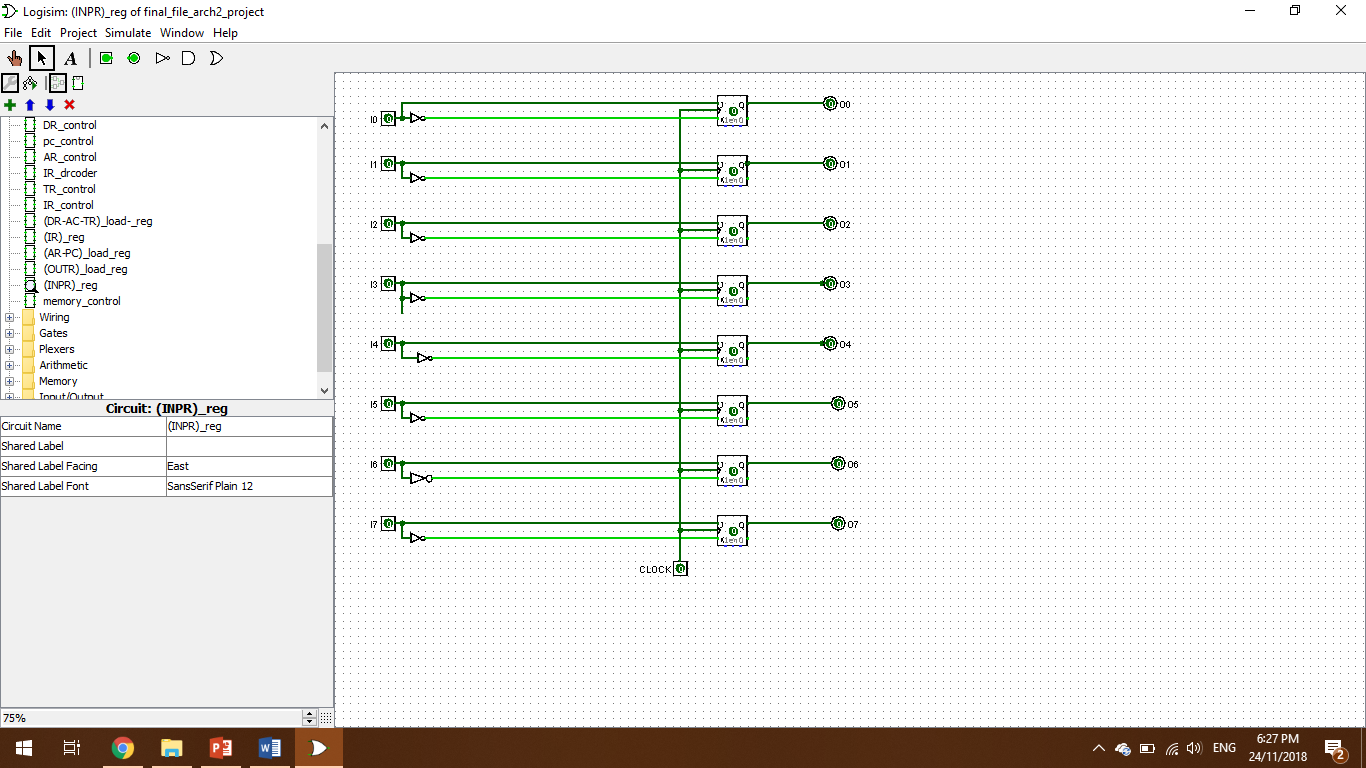
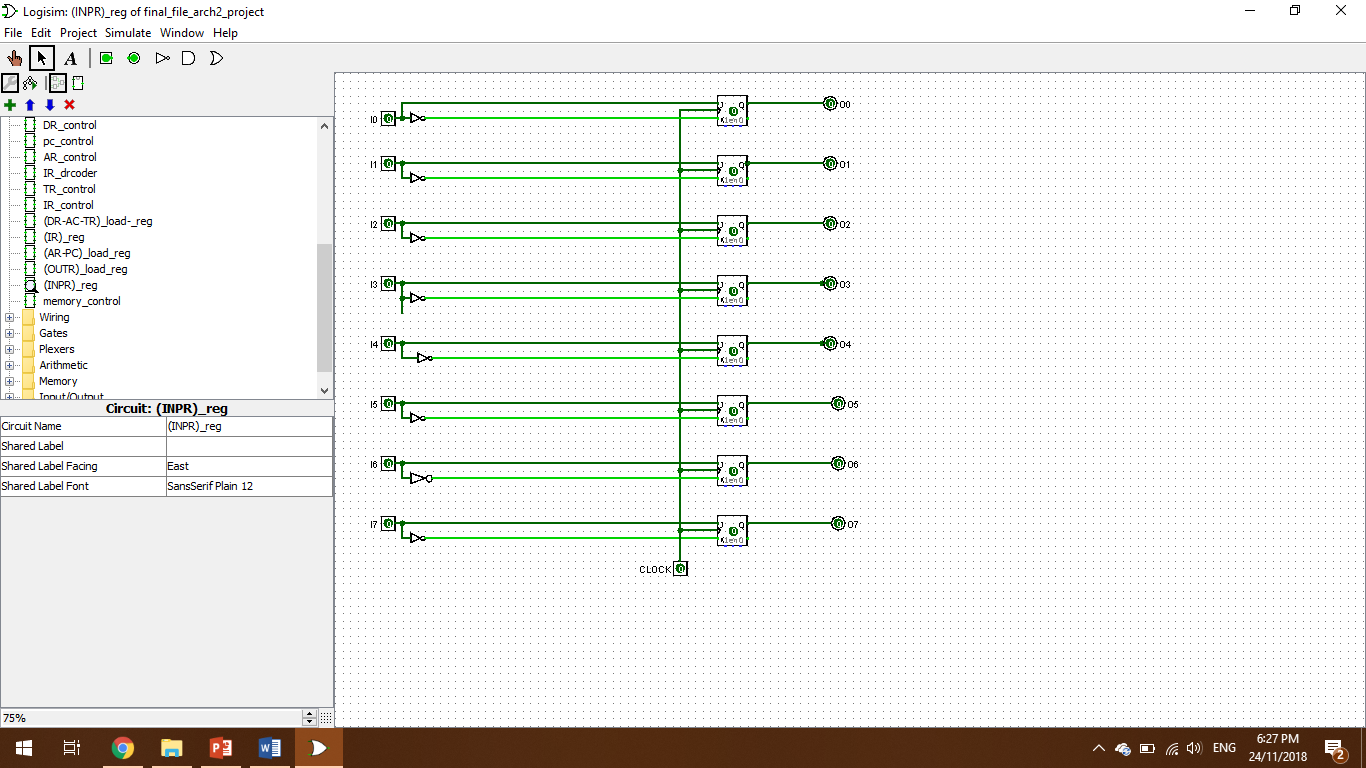
Load=D4.T4+D5.T5

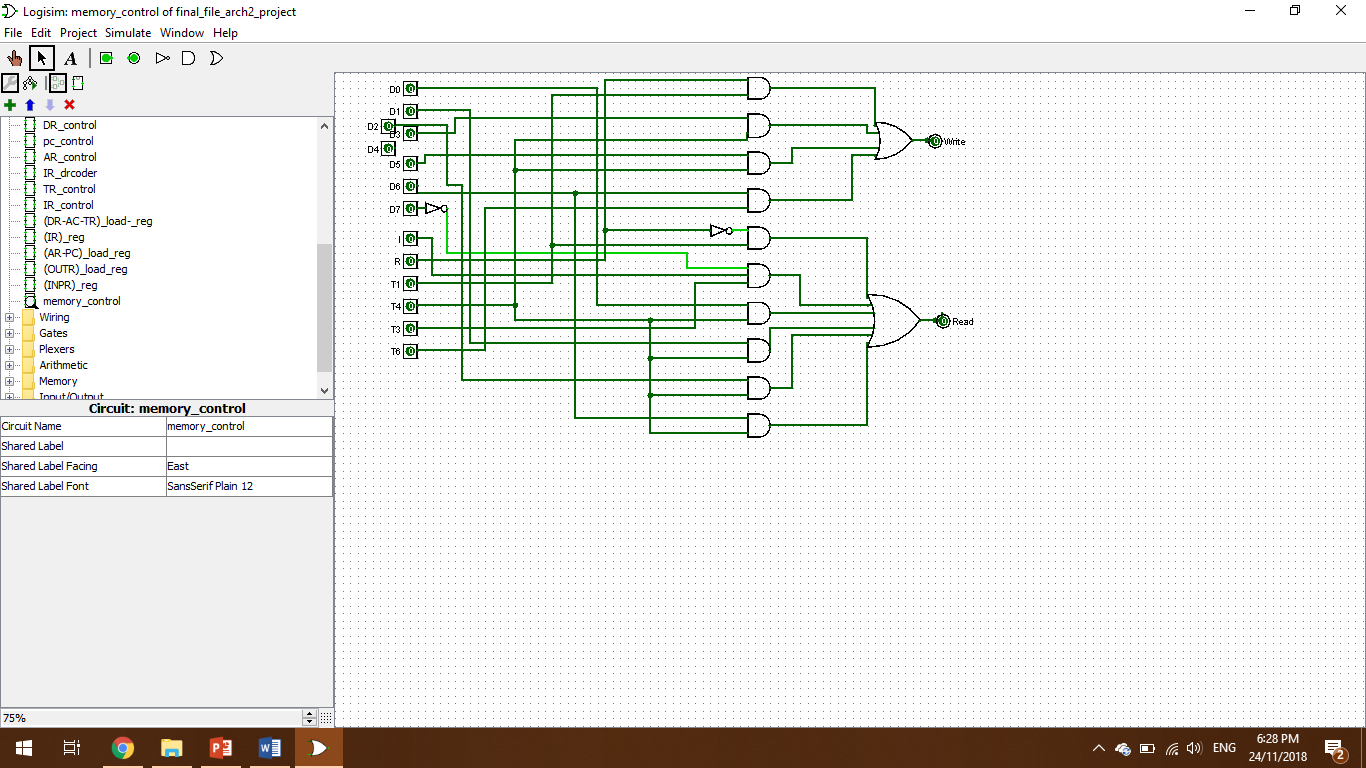
Increment=R’.T1+R.T2+D6.T6+r.B4+r.B3+r.B2+r.B1+p.B9+p.B8

1. **TR control:**

Load=R.T0



1. **INPR; 8-bits:**
2. **OUTR; 8-bits:**
3. **Memory control:**

Load=R.T1+D3.T4+D6.T6+D5.T4

1. **X1 till X7:**

X1=D4T4+D5T5

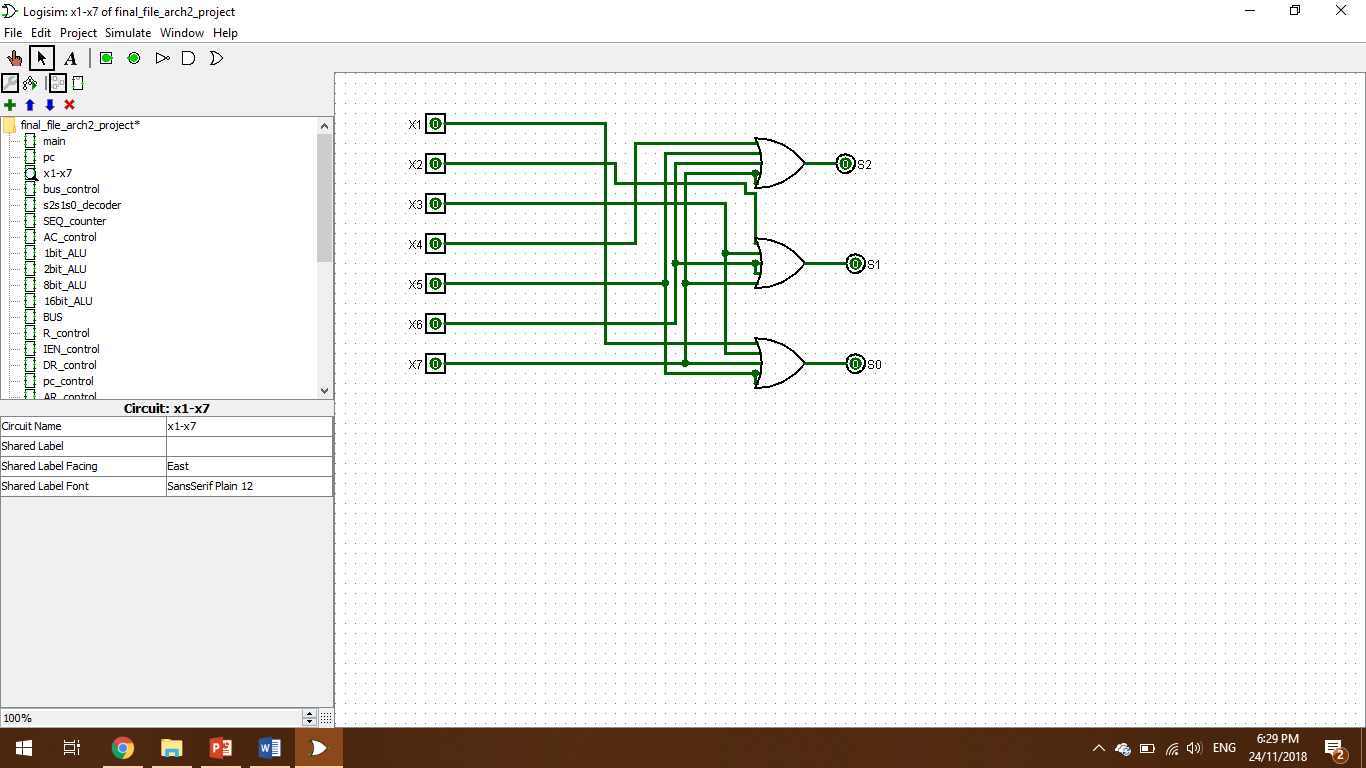
X2=R’.T0+R.T0+D5.T4

X3=D2.T5+D6.T6

X4=D3.T4

X5=R’T2

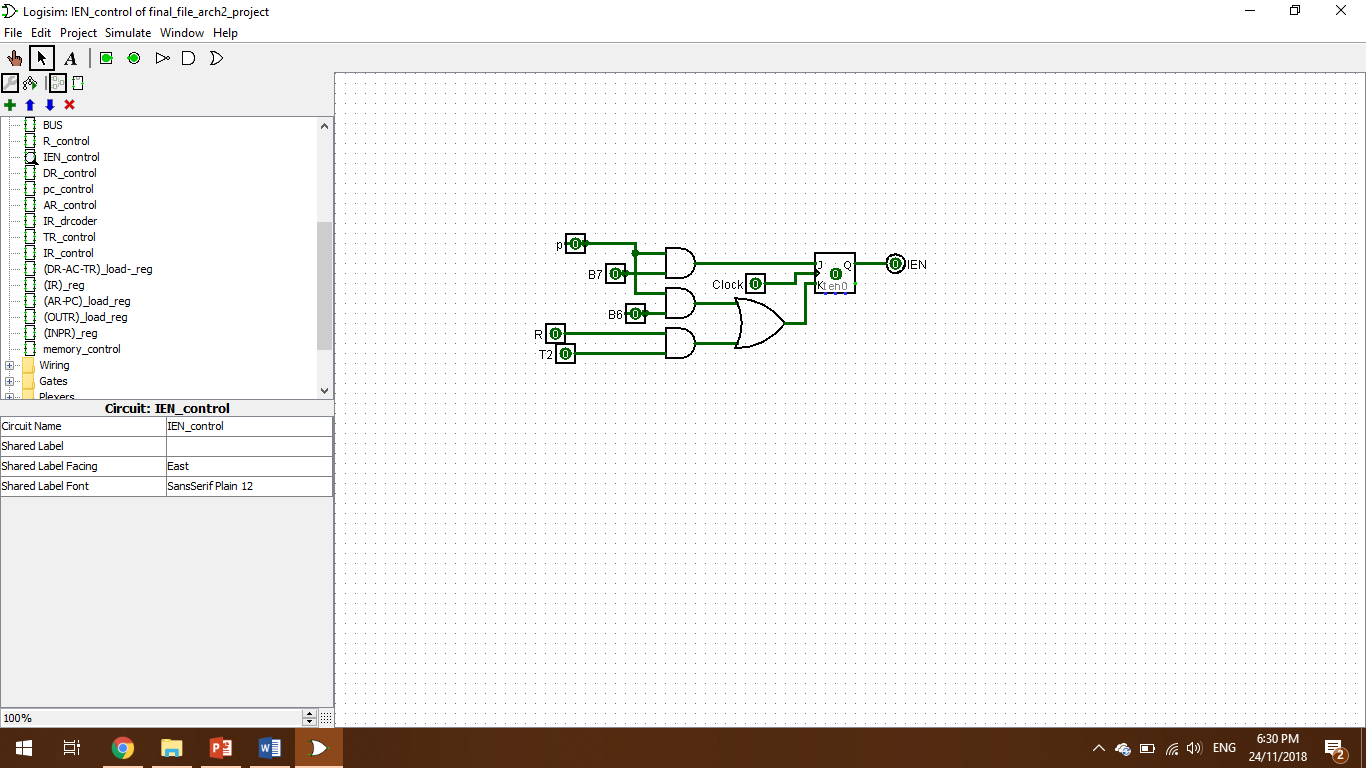
X6=RT1

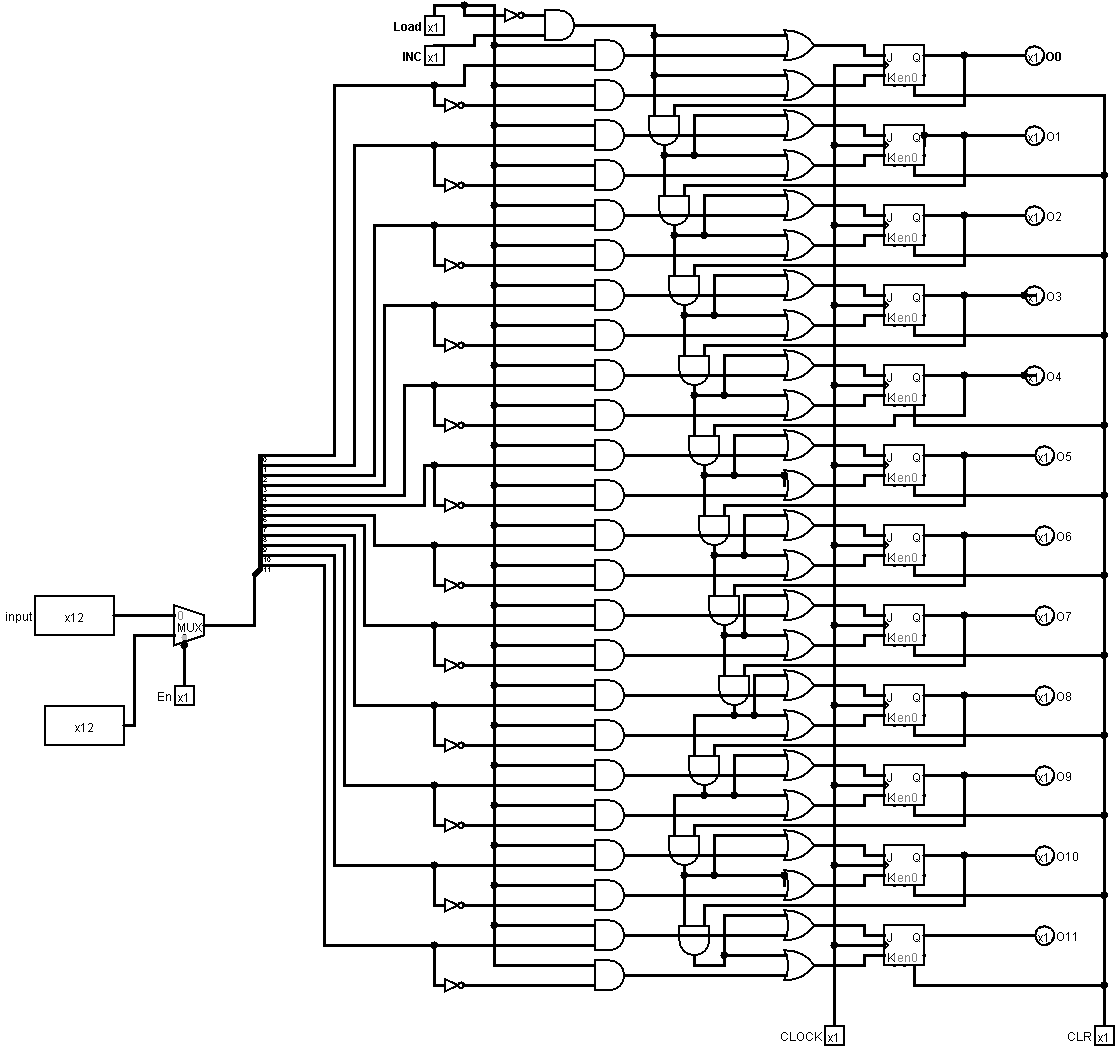
X7=R’.T1+D7’.T3’+(D0+D1+D2+D6).T4

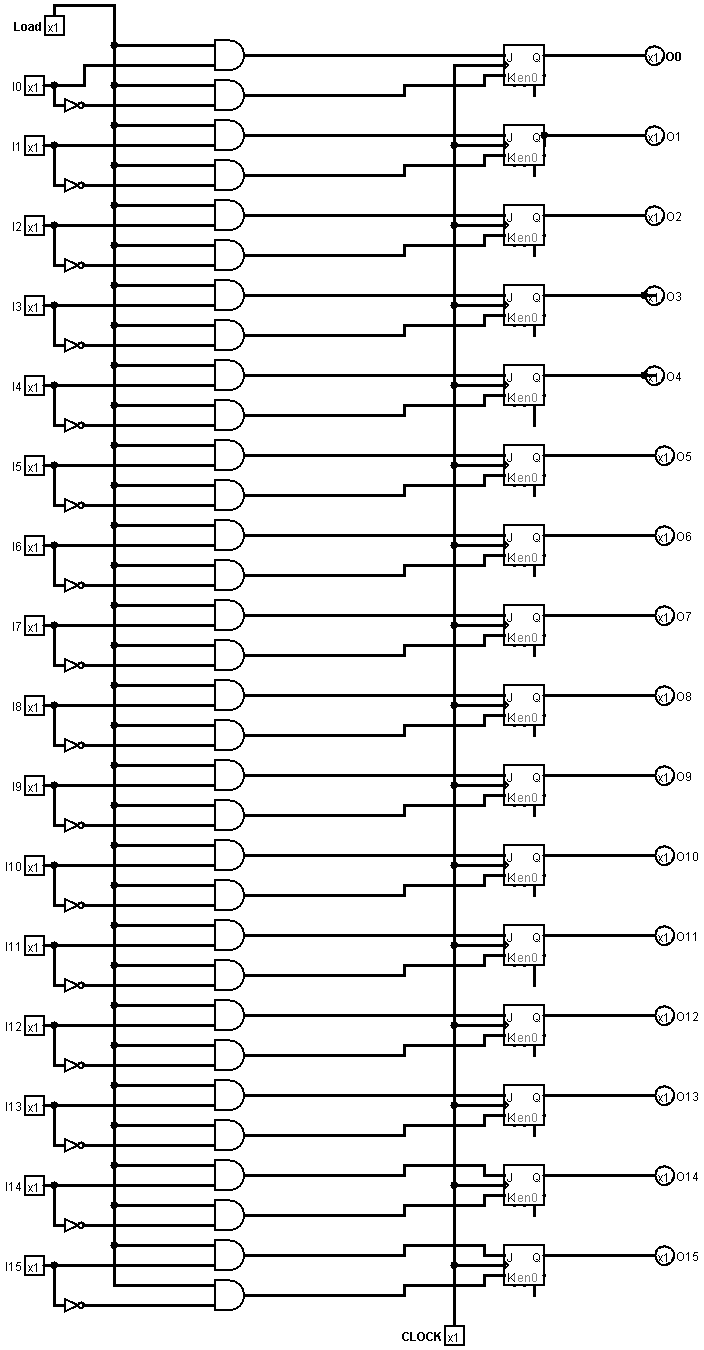
1. **IEN:**

Clear=R.T2+Pb6

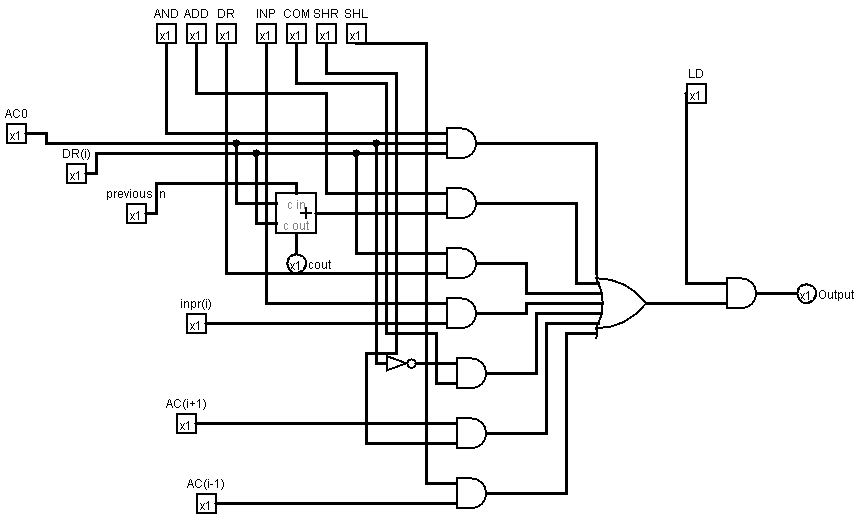
Set1=p.B7

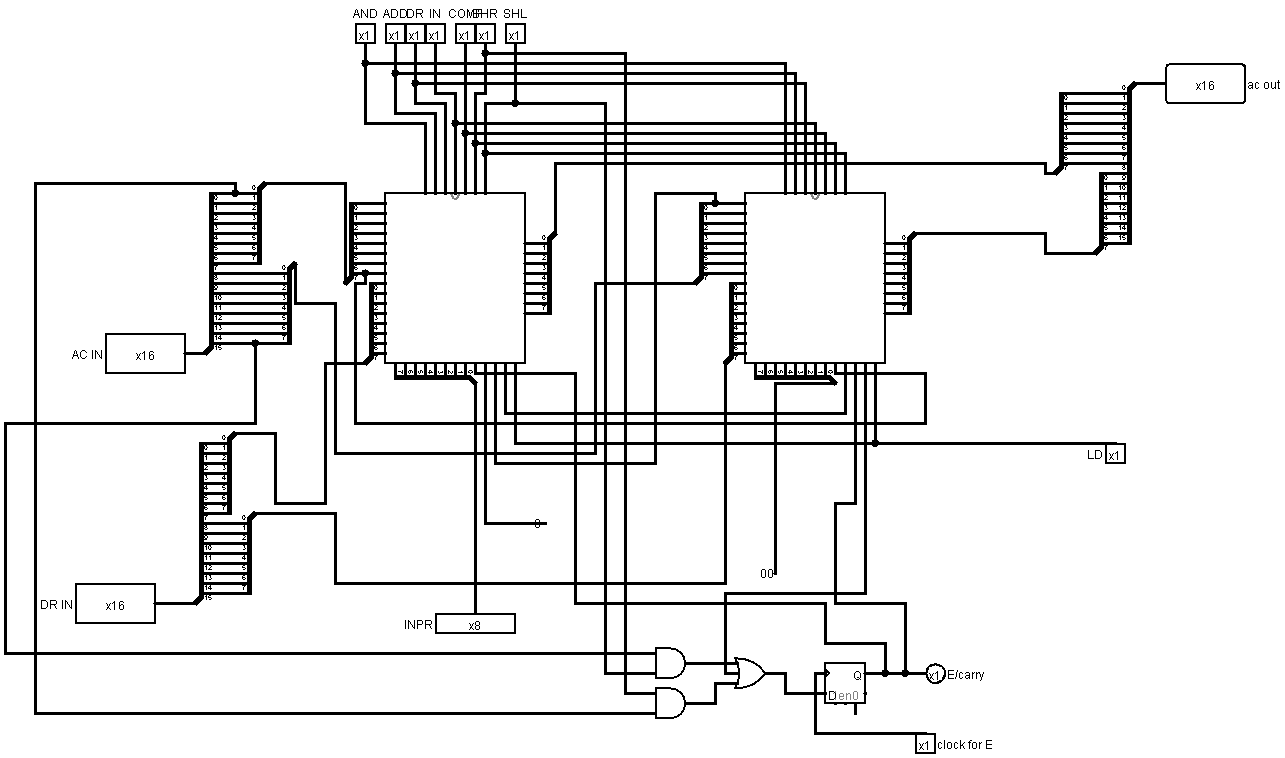


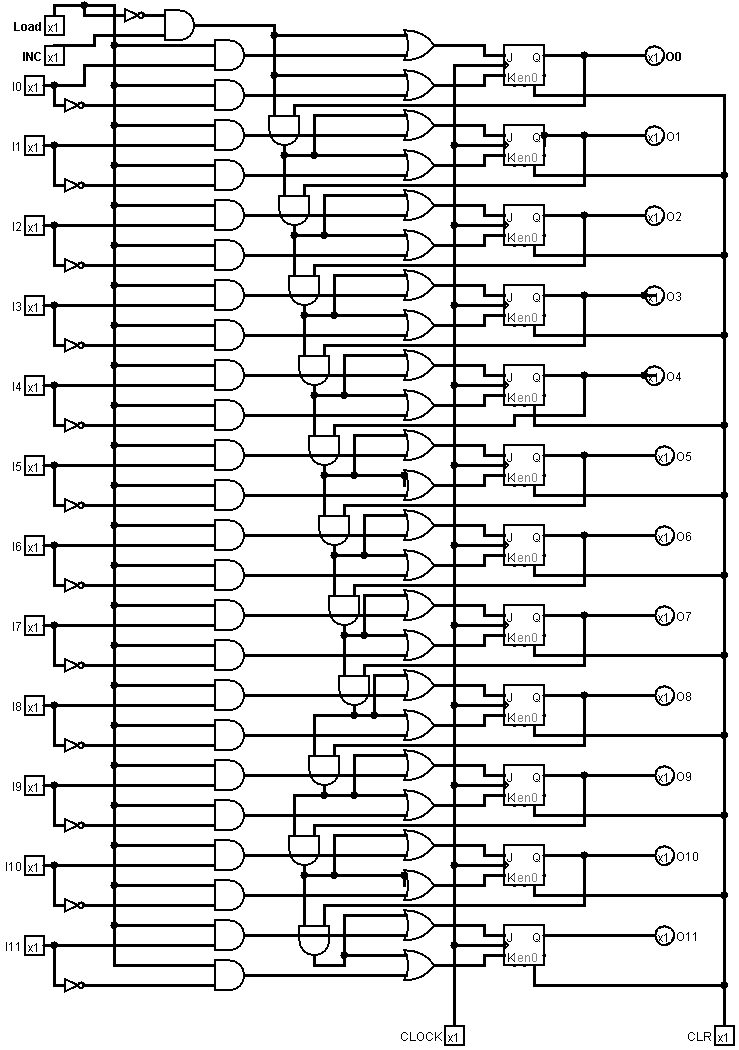
1. **PC; 12-bits:**
2. **IR register; 16-bits:**

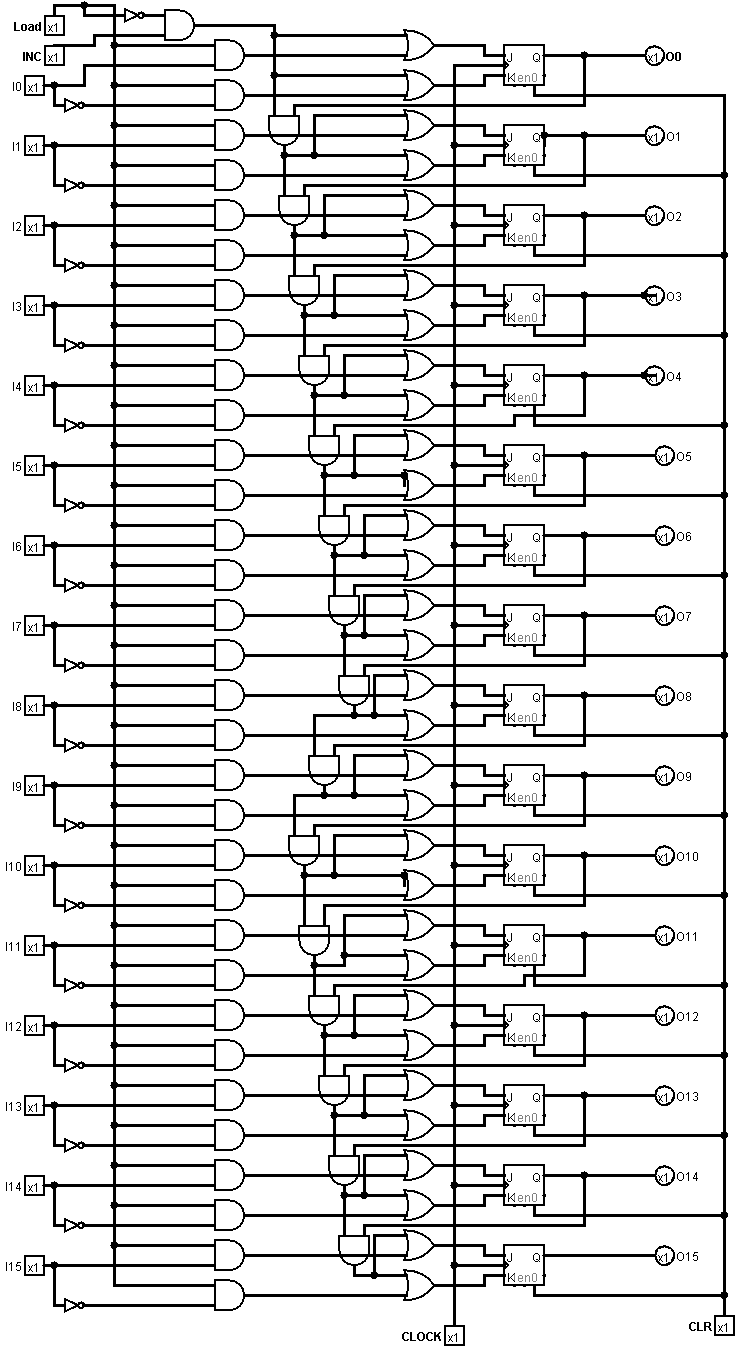


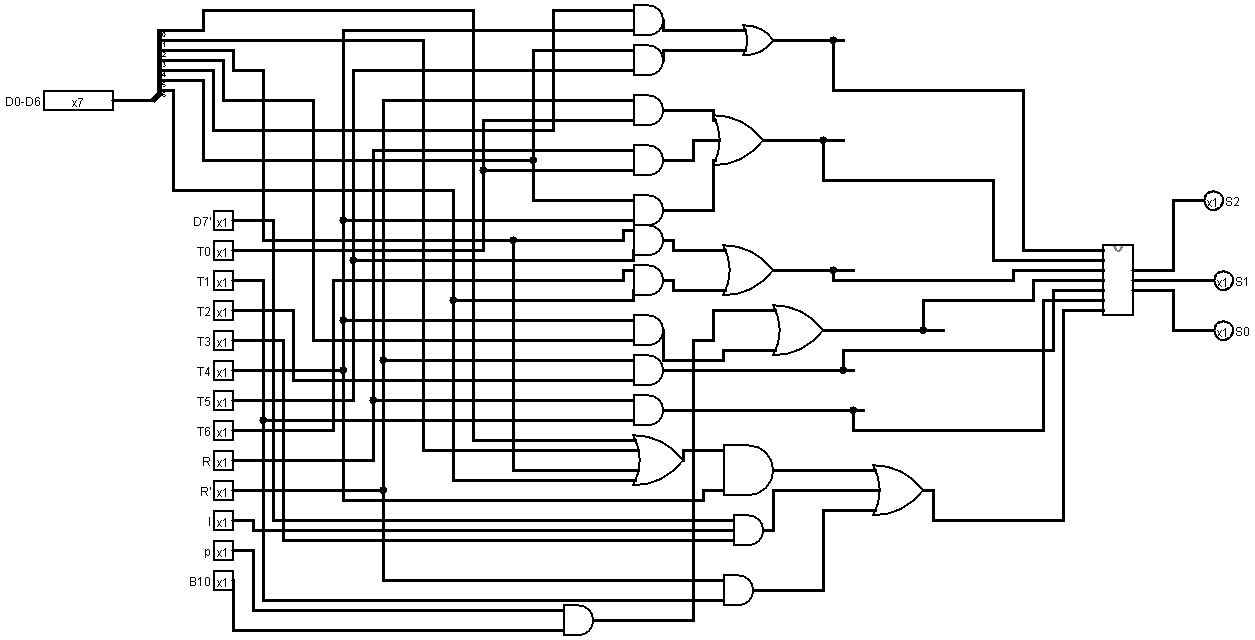
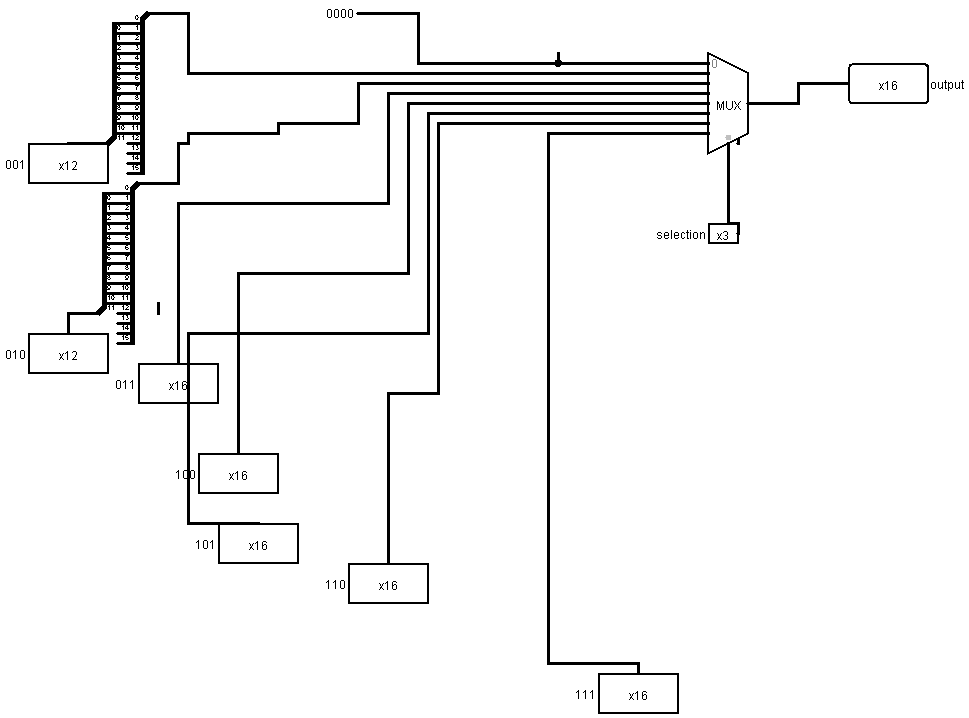
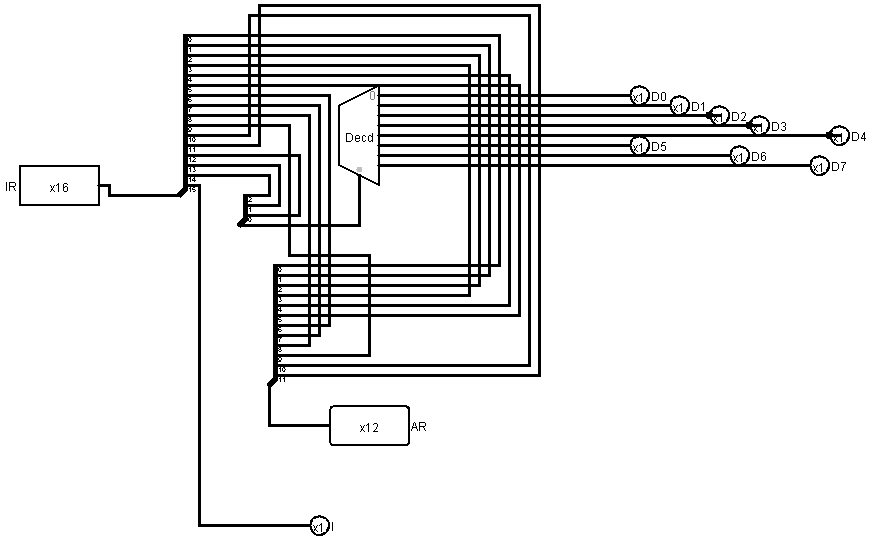
1. **ALU; 1-bit and 16-bits respectively:**

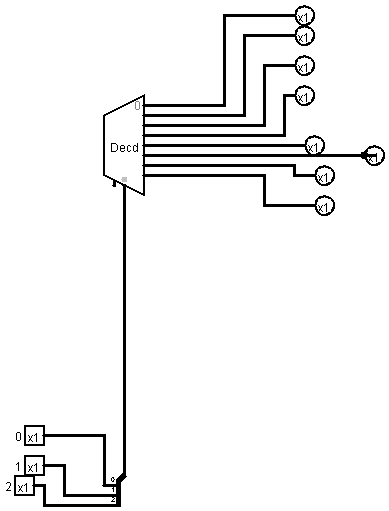
This 16-bit ALU is comprised of an 8-bit ALU which is made from two 4-bit ALUs. Each 4-bits ALU is made from 2-bits and that 2-bit ALU is made from a 1-bits ALU. These all can be found in the Logism file.

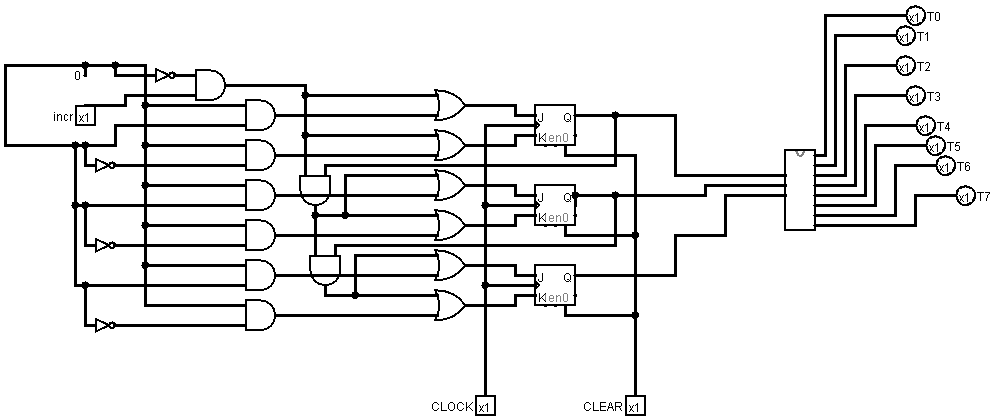


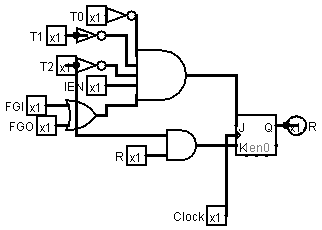
1. **PC and AC load registers:**
2. **DR, AC and TR load registers:**



1. **Bus control:**
2. **Bus:**
3. **D0 to D7:**
4. **S2S1S0:**



1. **Sequence counter:**
2. **R:**



1. **FGI:**

Clear=p.B11

Set1=p.B9

1. **FGO:**

Clear=pB10

Set1=pB8

1. **Multiplication Assembly code:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Address** | **Instruction** | **Machine code 16-bit IR** | **PC** | **AR** | **DR** | **AC** |
|  | INIT | -- | 300 | -- | -- | -- |
| 300 | CLA | 7800 | 301 | 300 |  | 0000 |
| 301 | 0 STA 102 | 3102 | 302 | 102 |  | 0000 |
| 302 | INC | 7020 | 303 | 302 |  | 0001 |
| 303 | INC | 7020 | 304 | 303 |  | 0002 |
| 304 | INC | 7020 | 305 | 304 |  | 0003 |
| 305 | INC | 7020 | 306 | 305 |  | 0004 |
| 306 | 0 STA 100 | 3100 | 307 | 100 |  | 0004 |
| 307 | INC | 7020 | 308 | 307 |  | 0005 |
| 308 | 0 STA 101 | 3101 | 309 | 101 |  | 0005 |
| 309 | CLA | 7800 | 30A | 309 |  | 0000 |
| 30A | CMA | 7200 | 30B | 30A |  | FFFF |
| 30B | 0 STA 103 | 3103 | 30C | 103 |  | FFFF |
| 30C | 0 LDA 102 | 2102 | 30D | 102 | 0000 | 0000 |
| 30D | 0 ADD 101 | 1101 | 30E | 101 | 0004 | 00004 |
| 30E | 0 STA 102 | 3102 | 30F | 102 | 0004 | 00004 |
| 30F | 0 LDA 100 | 2100 | 310 | 100 | 0005 | 00004 |
| 310 | 0 ADD 103 | 1103 | 311 | 103 | FFFF | 00004 |
| 311 | 0 STA 100 | 2100 | 312 | 100 | FFFF | 00004 |
| 312 | SZA | 7004 | 313 | 30C | FFFF | 00004 |
| 313 | BUN 20C | 430C | 314 |  | FFFF | 00004 |
| 314 | HLT | 7001 |  |  |  | 00004 |

|  |  |
| --- | --- |
| **Machine code 16-bit IR** | |
| -- | -- |
| 7800 | 0111 1000 0000 0000 |
| 3102 | 0011 0001 0000 0010 |
| 7020 | 0111 0000 0010 0000 |
| 7020 | 0111 0000 0010 0000 |
| 7020 | 0111 0000 0010 0000 |
| 7020 | 0111 0000 0010 0000 |
| 3100 | 0011 0001 0000 0000 |
| 7020 | 0111 0000 0010 0000 |
| 3101 | 0011 0001 0000 0001 |
| 7800 | 0111 1000 0000 0000 |
| 7200 | 0111 0010 0000 0000 |
| 3103 | 0011 0001 0000 0011 |
| 2102 | 0011 0001 0000 0010 |
| 1101 | 0001 0001 0000 0001 |
| 3102 | 0011 0001 0000 0010 |
| 2100 | 0010 0001 0000 0000 |
| 1103 | 0001 0001 0000 0011 |
| 2100 | 0010 0001 0000 0000 |
| 7004 | 0111 0000 0000 0100 |
| 430C | 0100 0011 0000 1100 |
| 7001 | 0111 0000 0000 0001 |

1. **Machine language:**