**SPEEDOMETER**

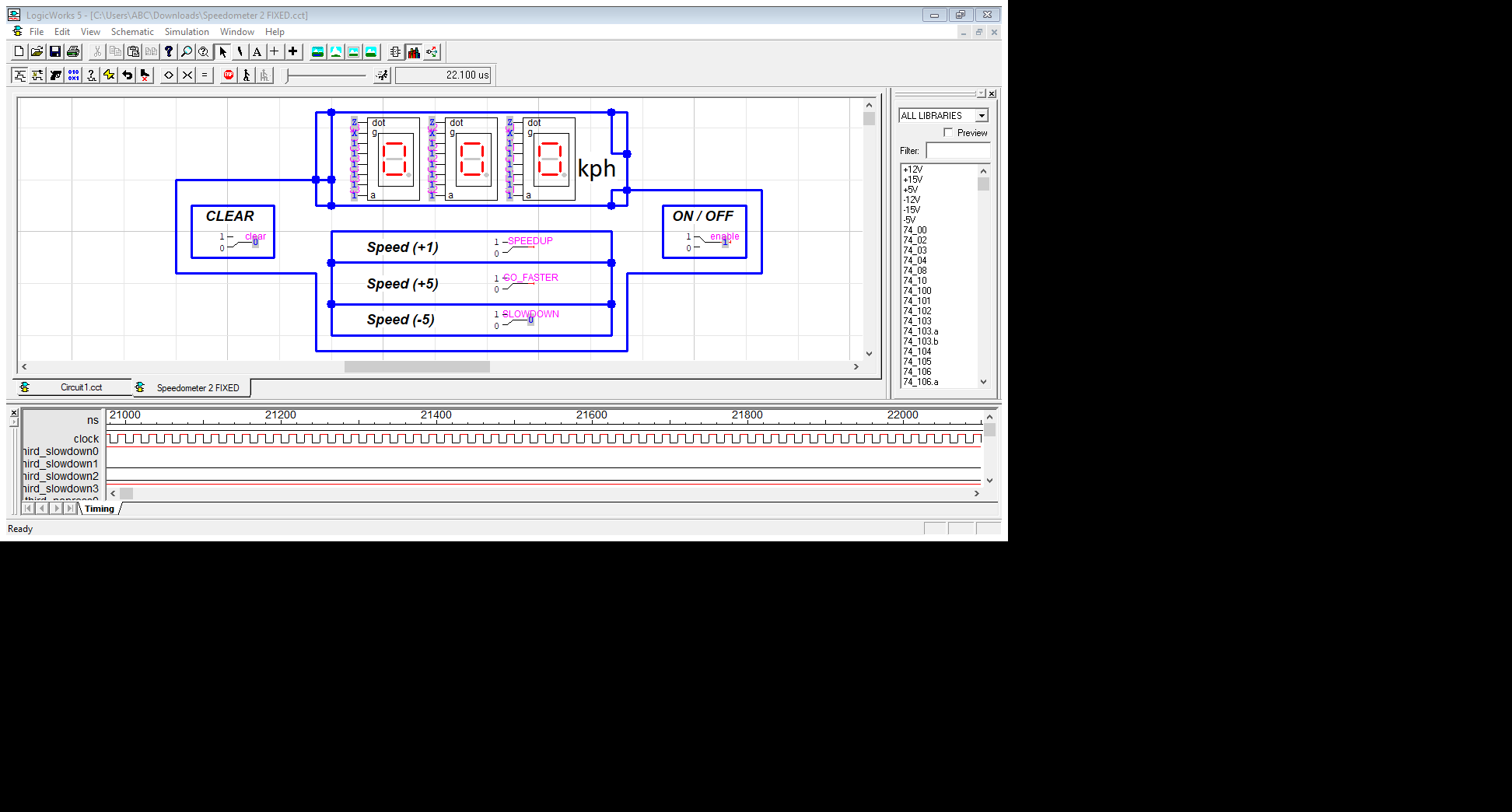
**Submitted By:**

* Roohan ali ( 21L-5629 )
* Mujtaba (21L-5613 )
* Eyad ( 21L-7695 )

**CONNECTIVITY OF WIRES/CONNECTION BETWEEN THE CIRCUITS**

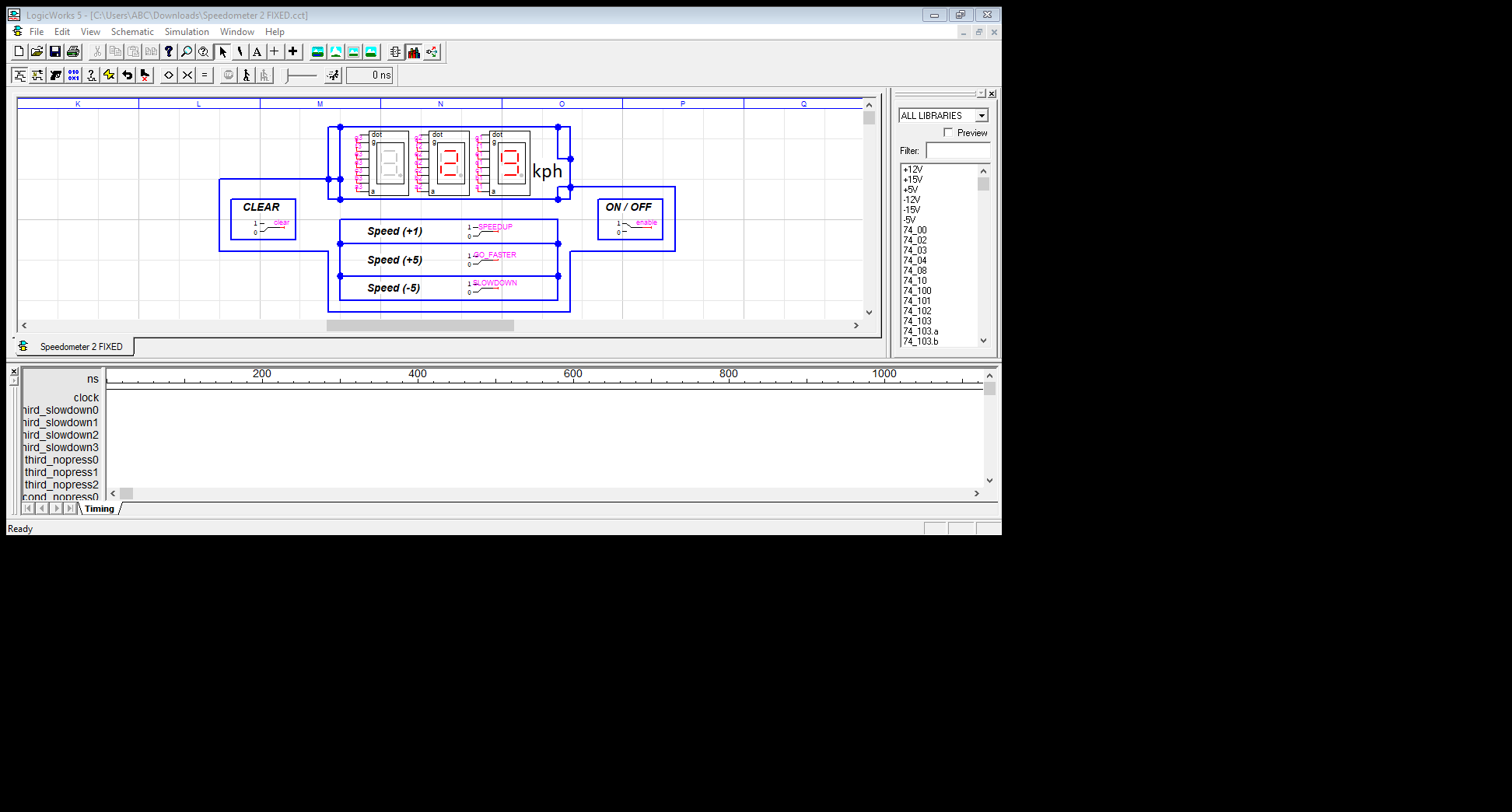
We started this project by joining the wires through the inputs and outputs of the IC’s but as we advanced we faced many difficulties as the circuit looked bulky and so much congested that it was a serious problem for us to explain the circuitry. So after the some research we came to know that by having the same names of the wires the connections can be made easily this reduced the wires traffic in the circuit and made it look simple.

**WORKING OF SPEEDOMETER DISPLAY**

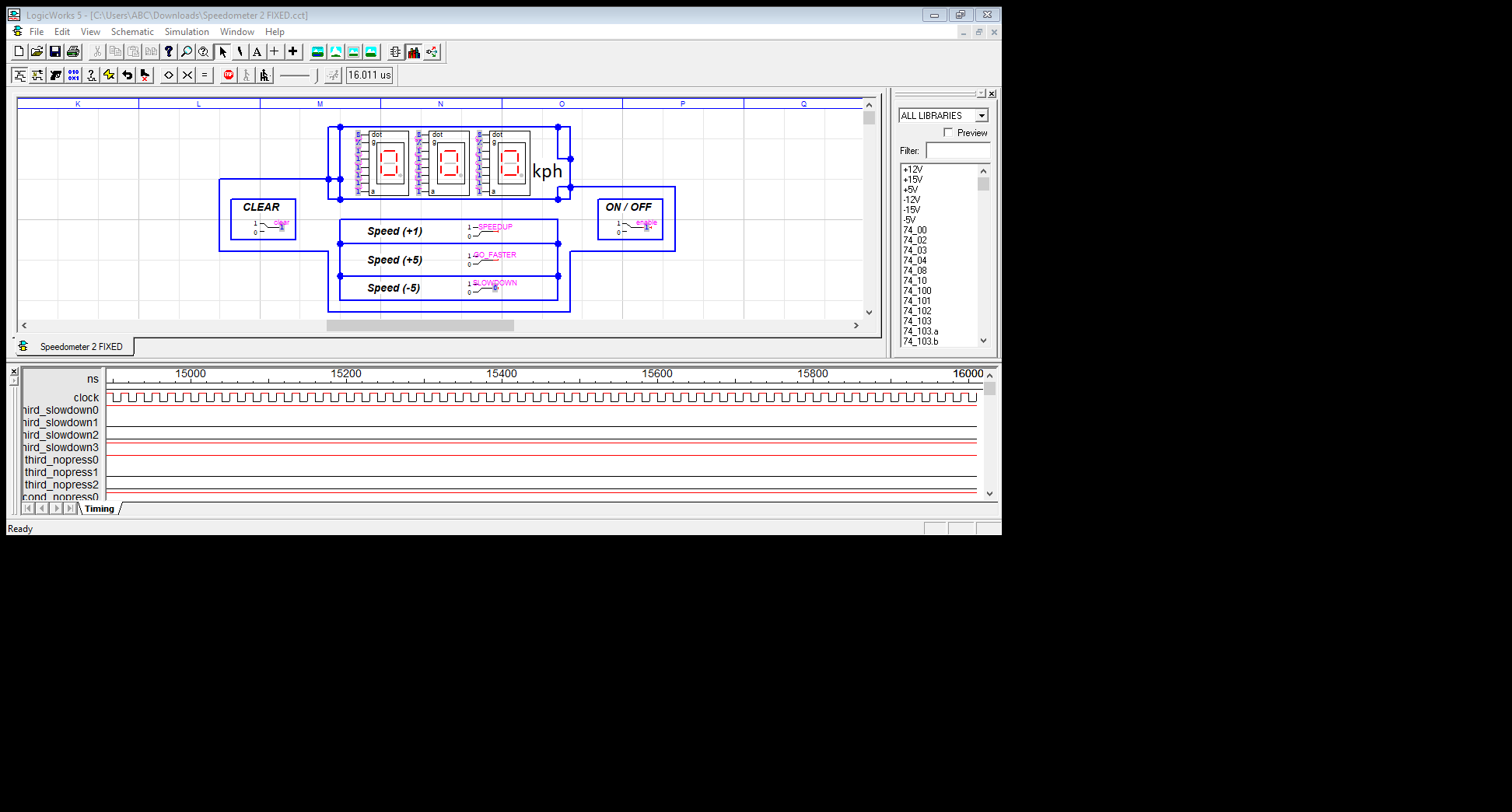


This is the interface of the **speedometer** where the speed will be displayed. The **ON/OFF** switch works as an enable when the switch is given 1 input the **7-segment display** is activated and show output 0 in each 7-segment display. The **Clear switch** is used to reset the output entries its initial value 0, such as:

**BEFORE CLEAR IS PRESSED:**

****

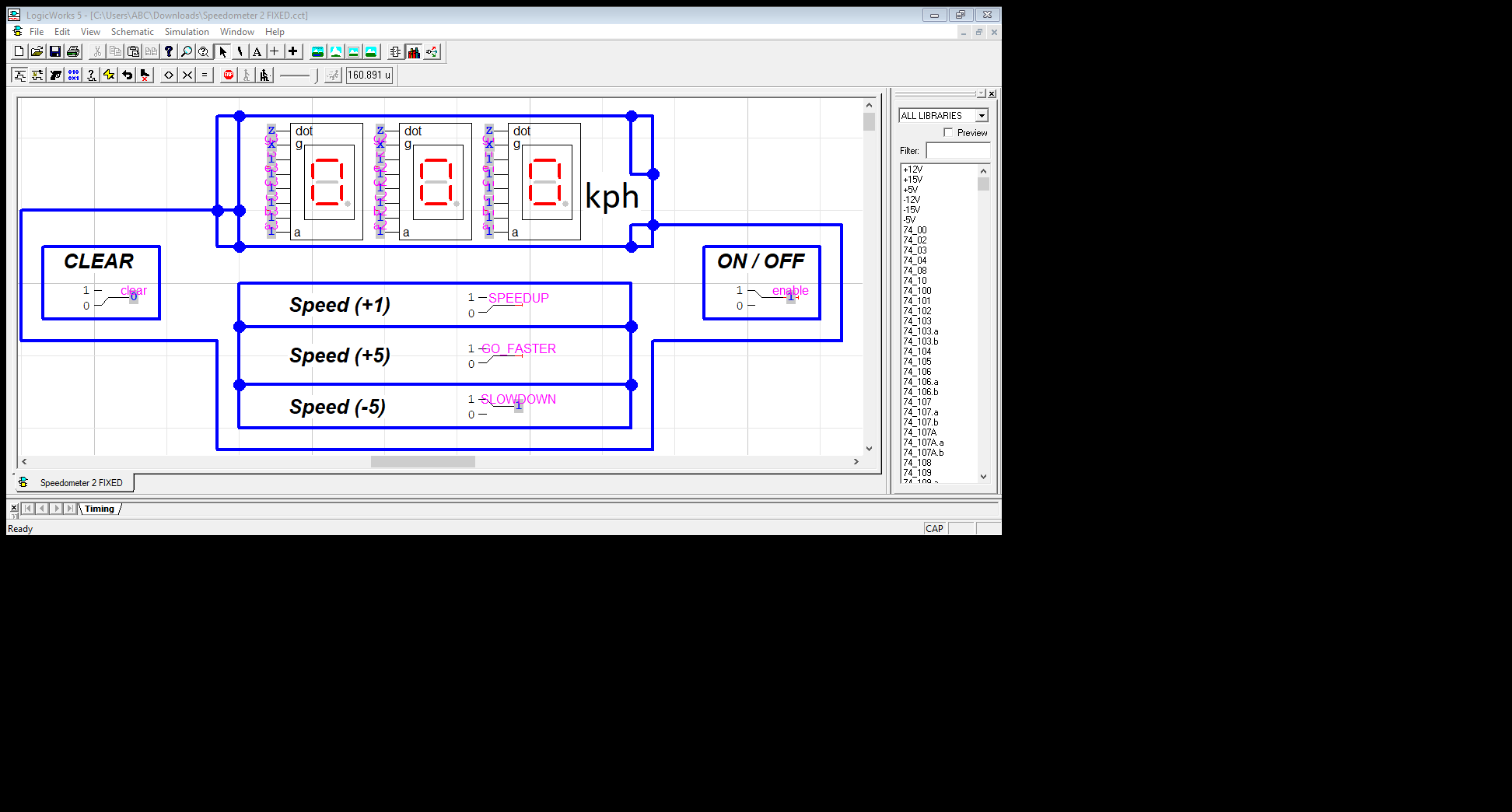
**AFTER CLEAR IS PRESSED:**

****

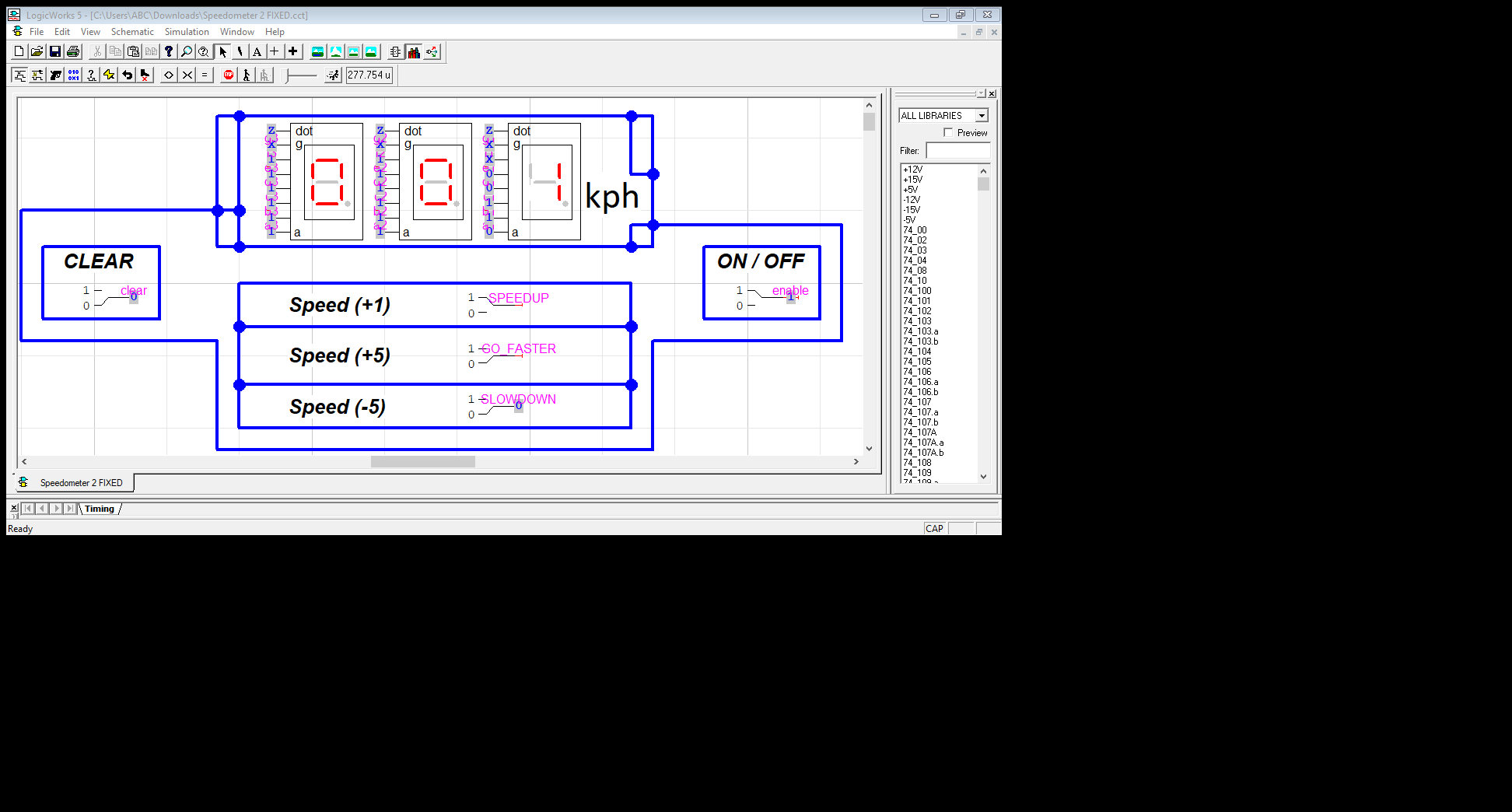
Now explaining the 3 switches mentioned as **Speed (+1), Speed** **(+5) and Speed (-5):-**

* **Speed (+1):** This binary switch is used to increment the speed with 1.

**BEFORE ENABLING Speed (+1):**

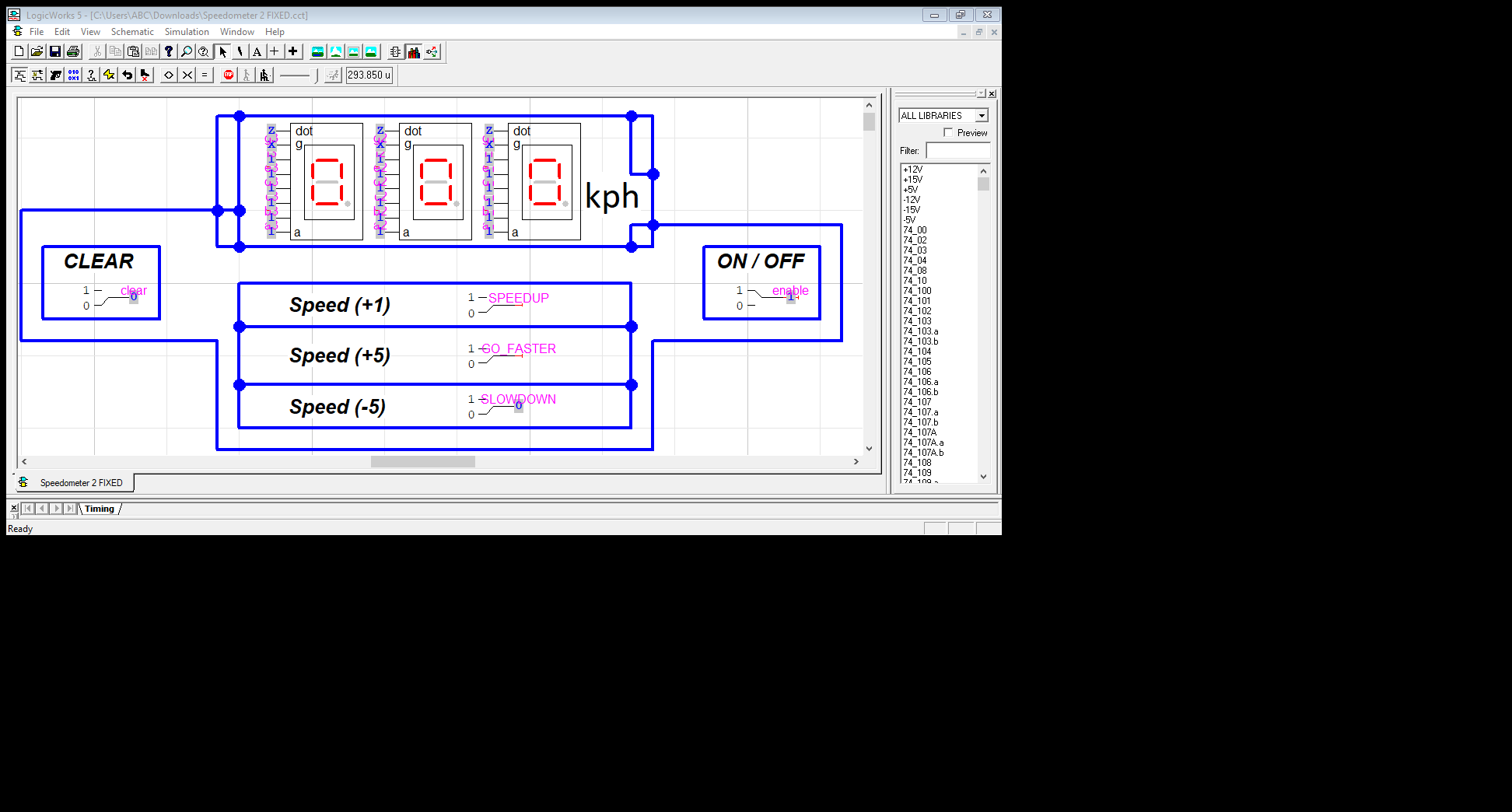


**AFTER ENABLING Speed (+1):**

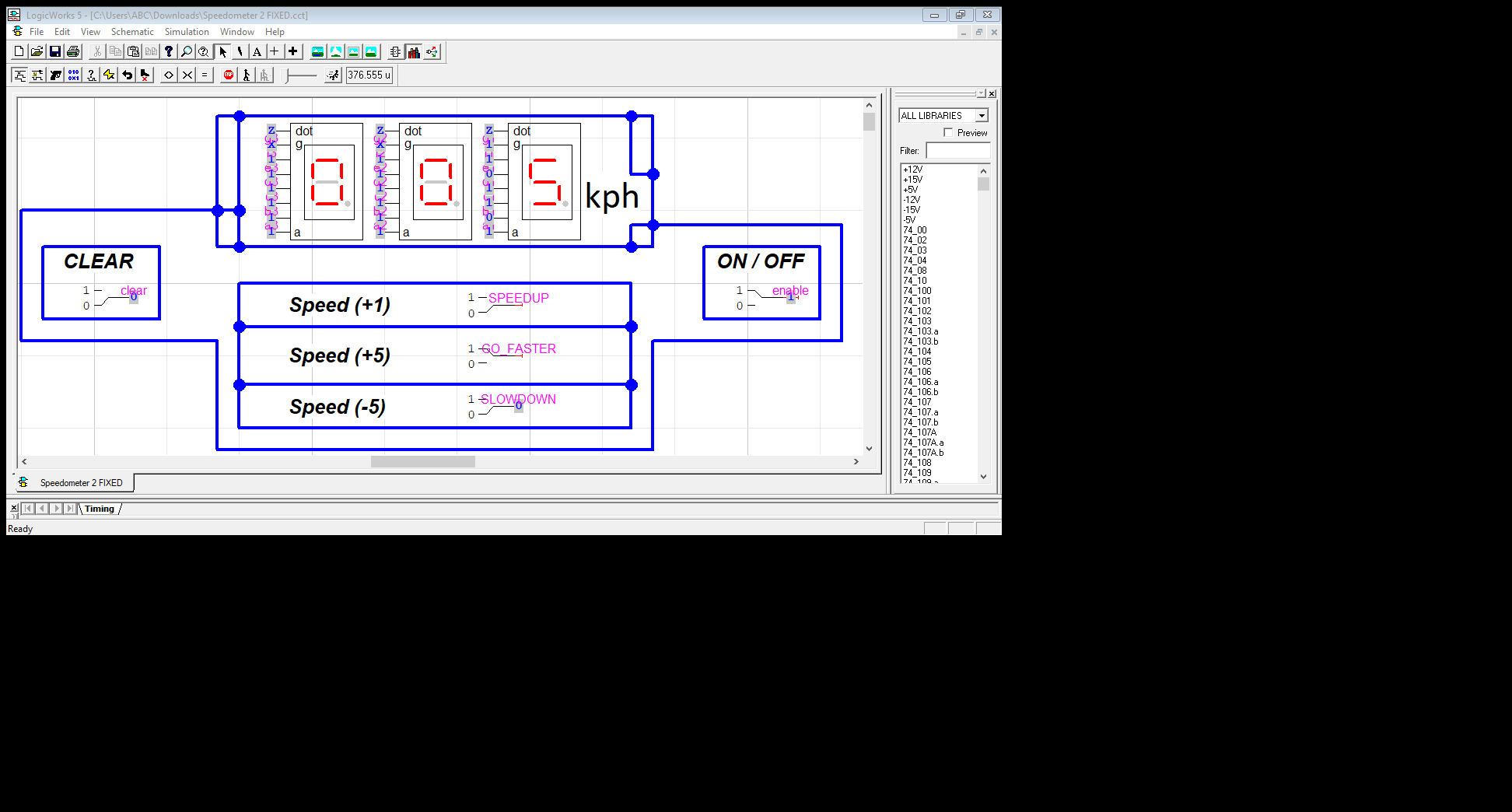
****

* **Speed (+5):** This is used to increment the speed with 5.

**BEFORE ENABLING Speed (+5):**

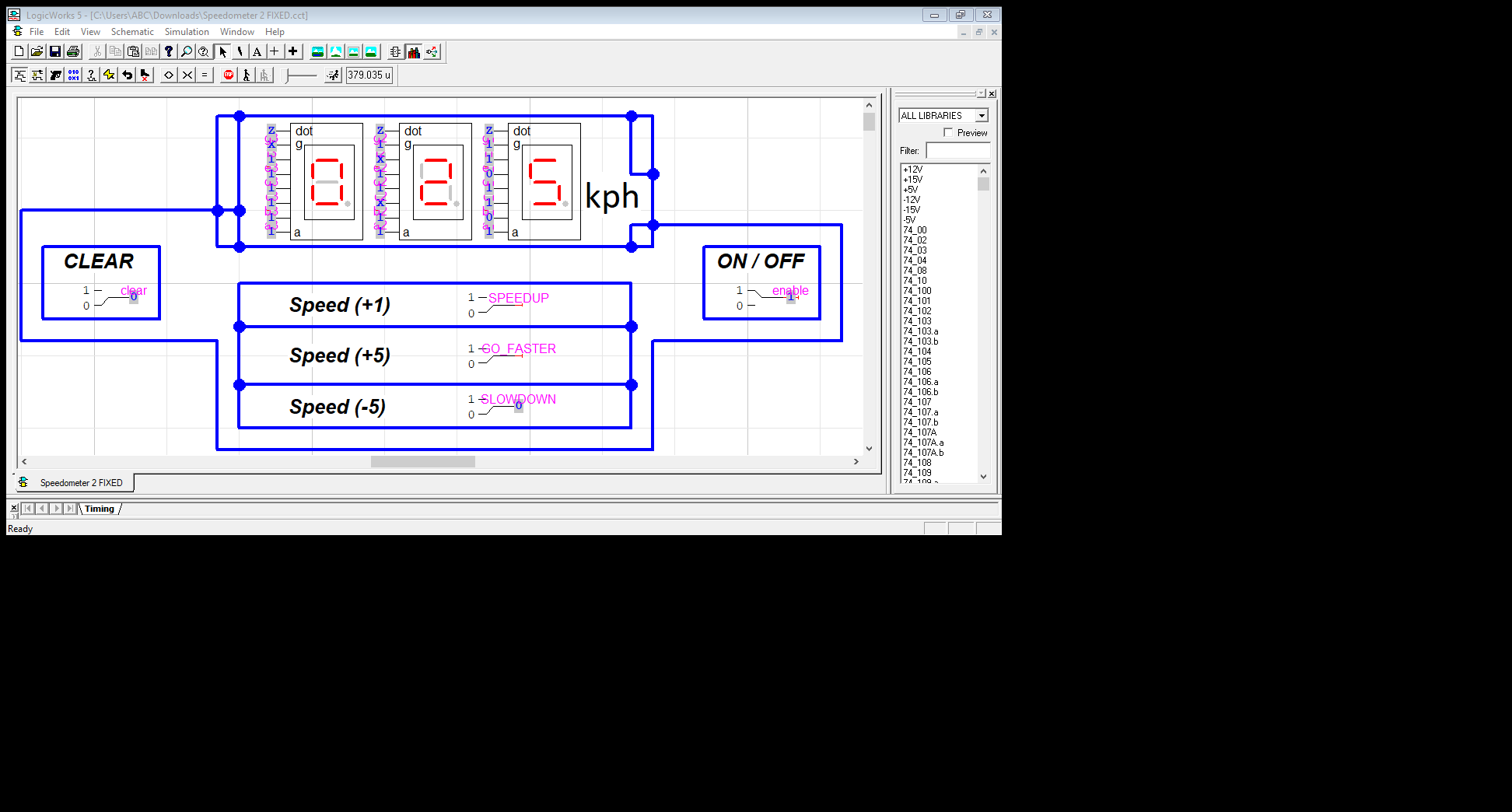
****

**AFTER ENABLING Speed (+5):**

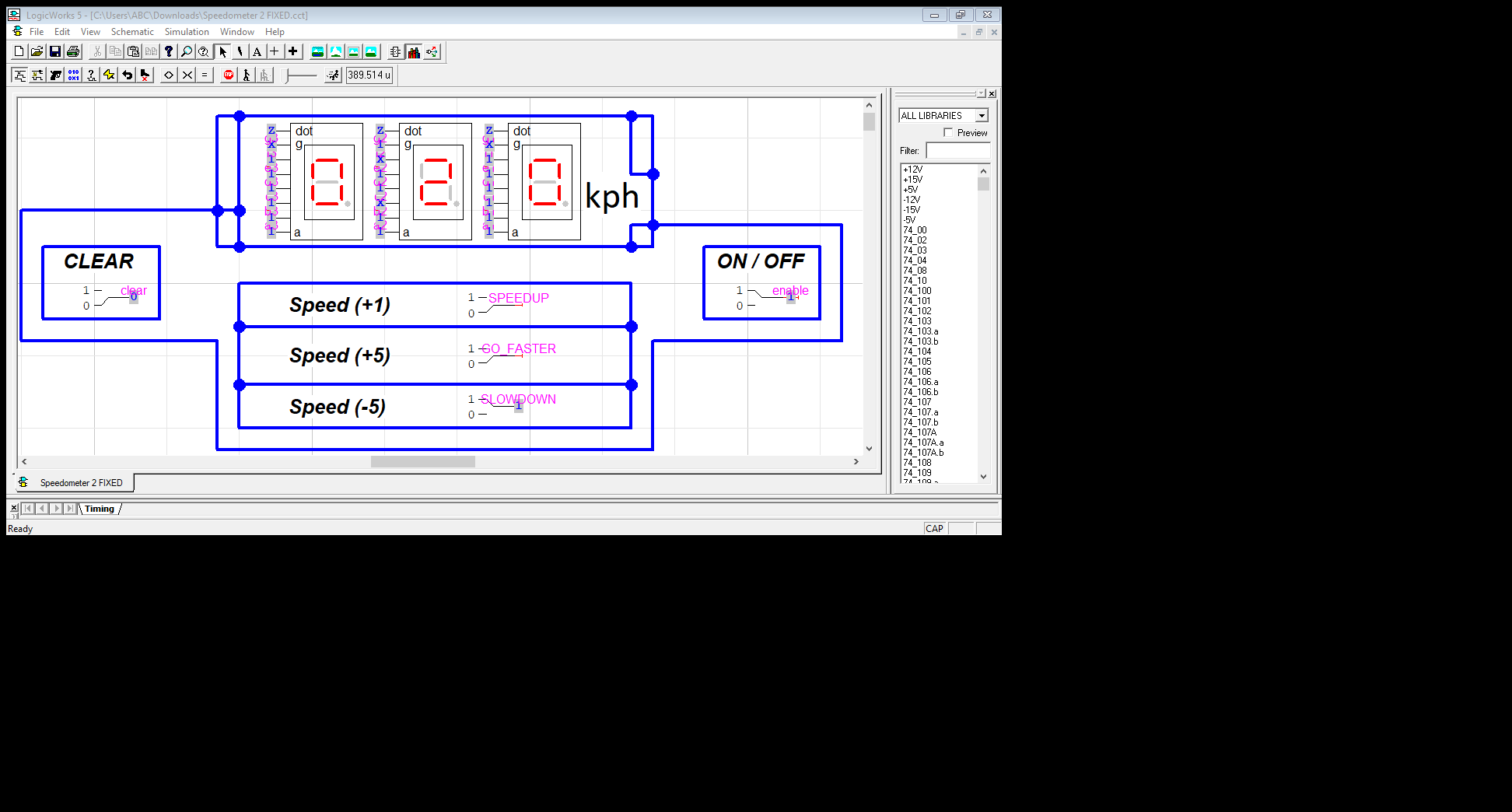
****

* **Speed (-5):** This binary switch works as a brake, which slows the speed with the decrement of 5.

**BEFORE ENABLING Speed (-5):**

****

**AFTER ENABLING Speed (-5):**

****

**WORKING OF INCREMENT AND DECREMENT:**

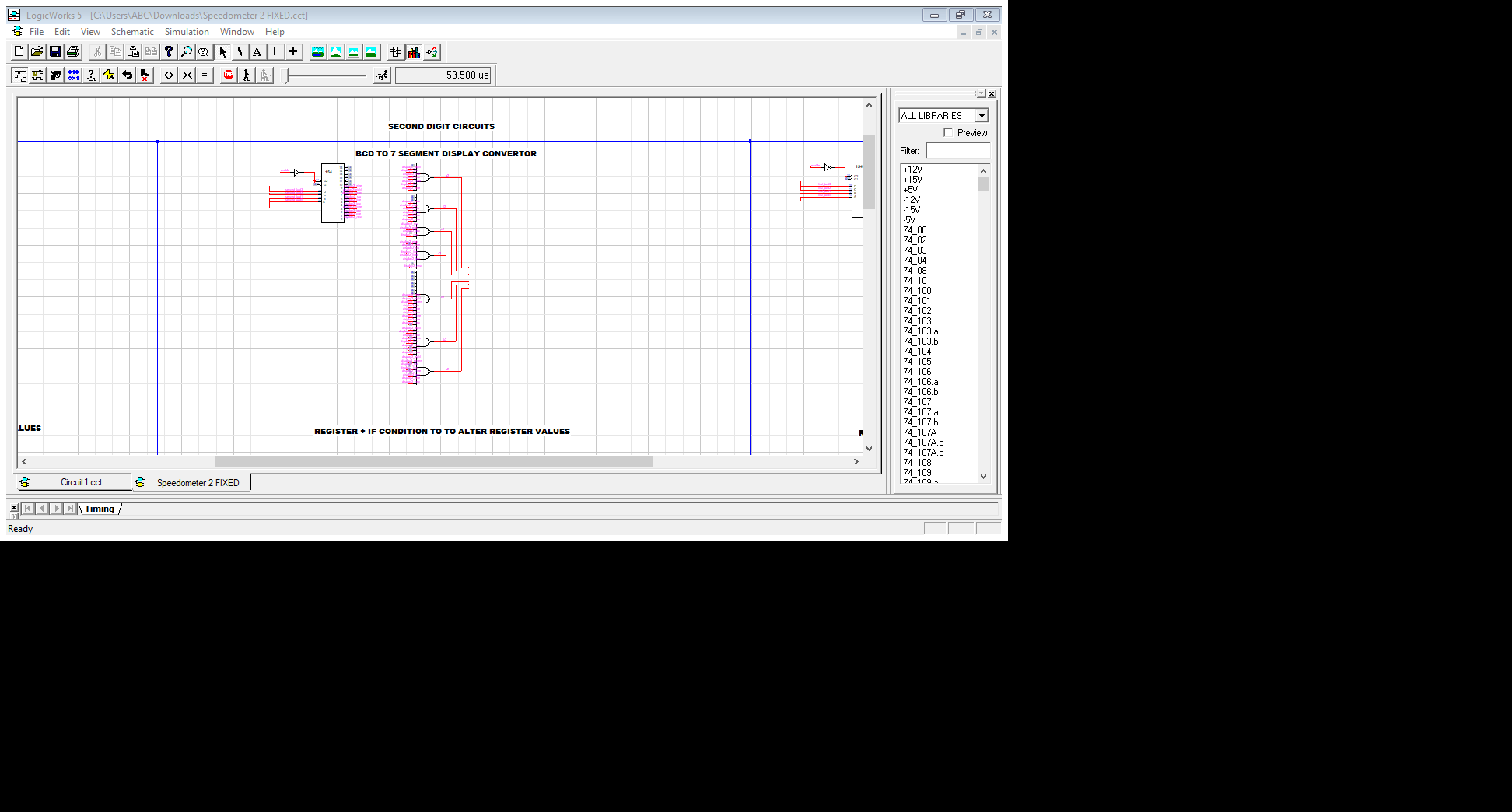
The increment and decrement of the Speedometer works only if the enable receives input 1. If the Speed (+1) receives input as 1 the speed of speedometer will start increasing as 0, 1, 2, 3 and so on. If Speed (+5) switch receives input as 1 then the speed of speedometer will start increasing as 0 , 5 , 10 , 15 and so on(these two acts as an accelerator). But when you need to return to the initial value 0 or to slow down the speed there are only 2 options:

1. Either give both of the Speed (+) switches with binary input of 0.
2. Give Speed (-5) switch a binary input 1.

When both of the increment switches are closed or given input 0. Then the speed displayed on **speedometer (7-segment display)** will start decreasing with 1 like 20, 19, 18, and so on.

But when you want to decrease the speed much faster than **Speed (-5)** is given an input value 1. The speed will start decreasing with 5 like 20, 15, 10, 5, 0.This acts as a brake.

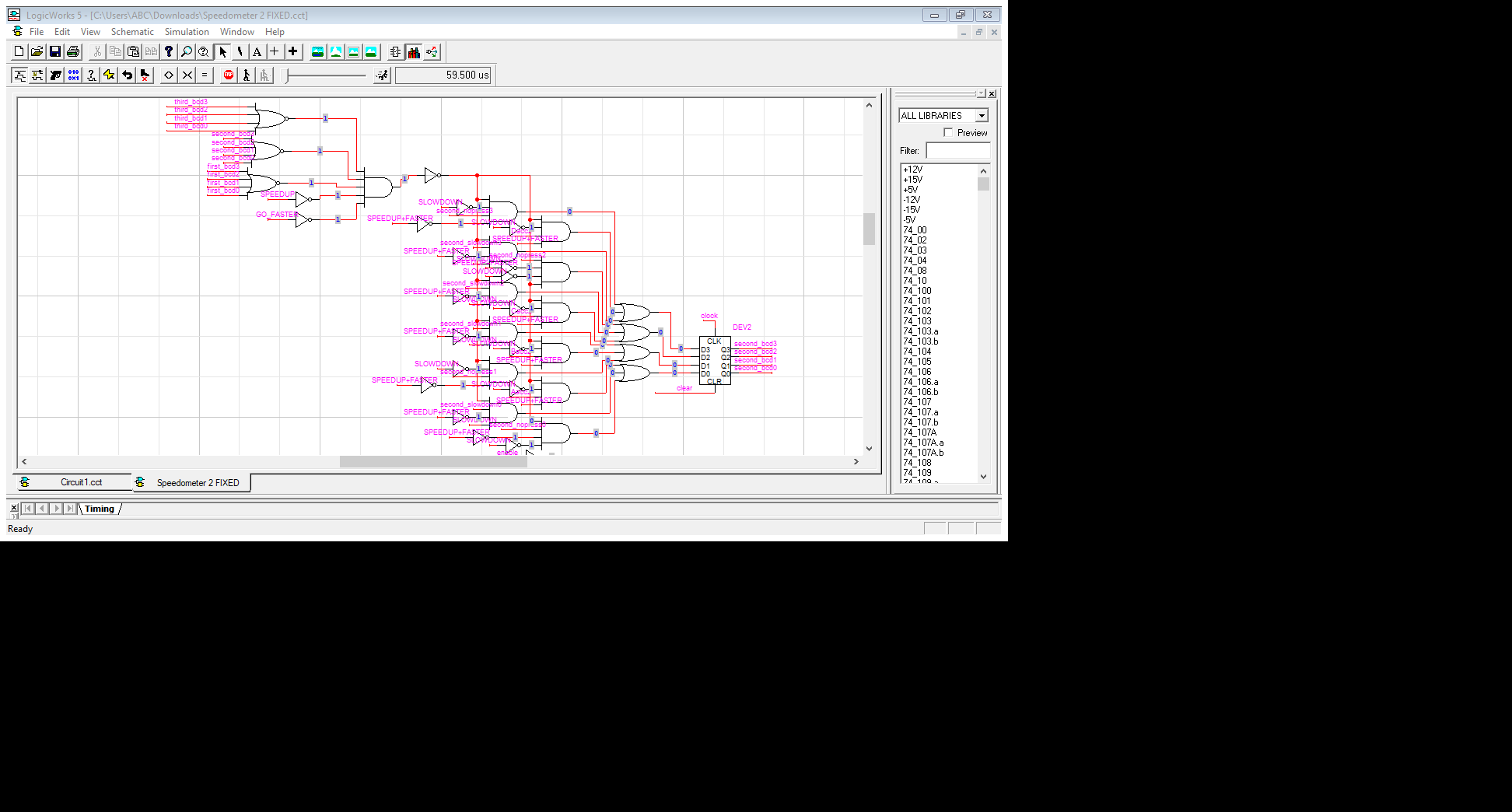
**WORKING OF BCD TO 7-SEGMENT DISPLAY CONVERTOR**



As there are three 7-segment displays so each 7 segment display has its own circuitry which helps it to display a value.

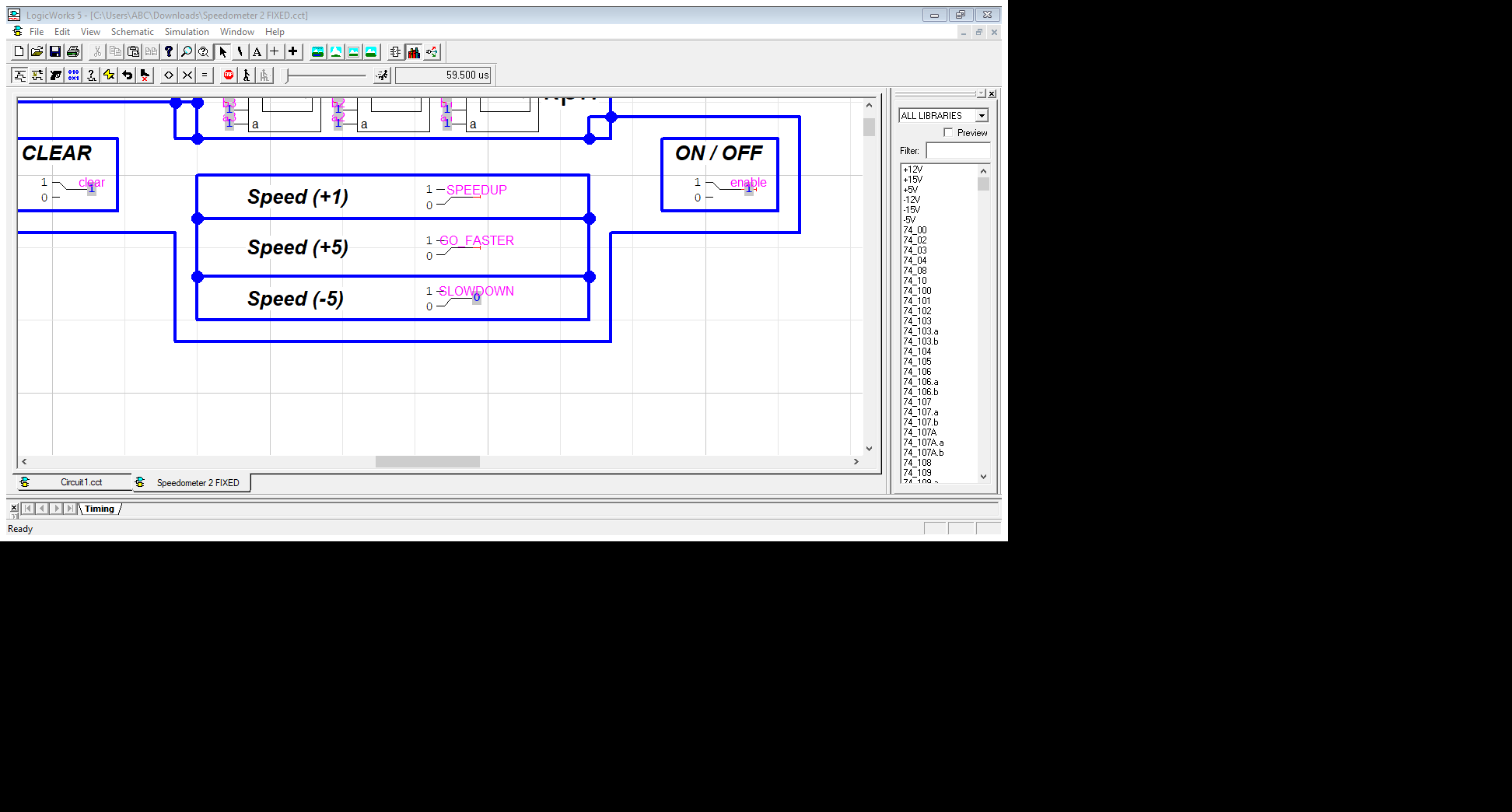
This converter is used so it can convert the BCD value that is stored in register so that it can be displayed on 7 segment display.

**REGISTER + IF CONDITION TO ALTER REGISTER VALUES:**

****

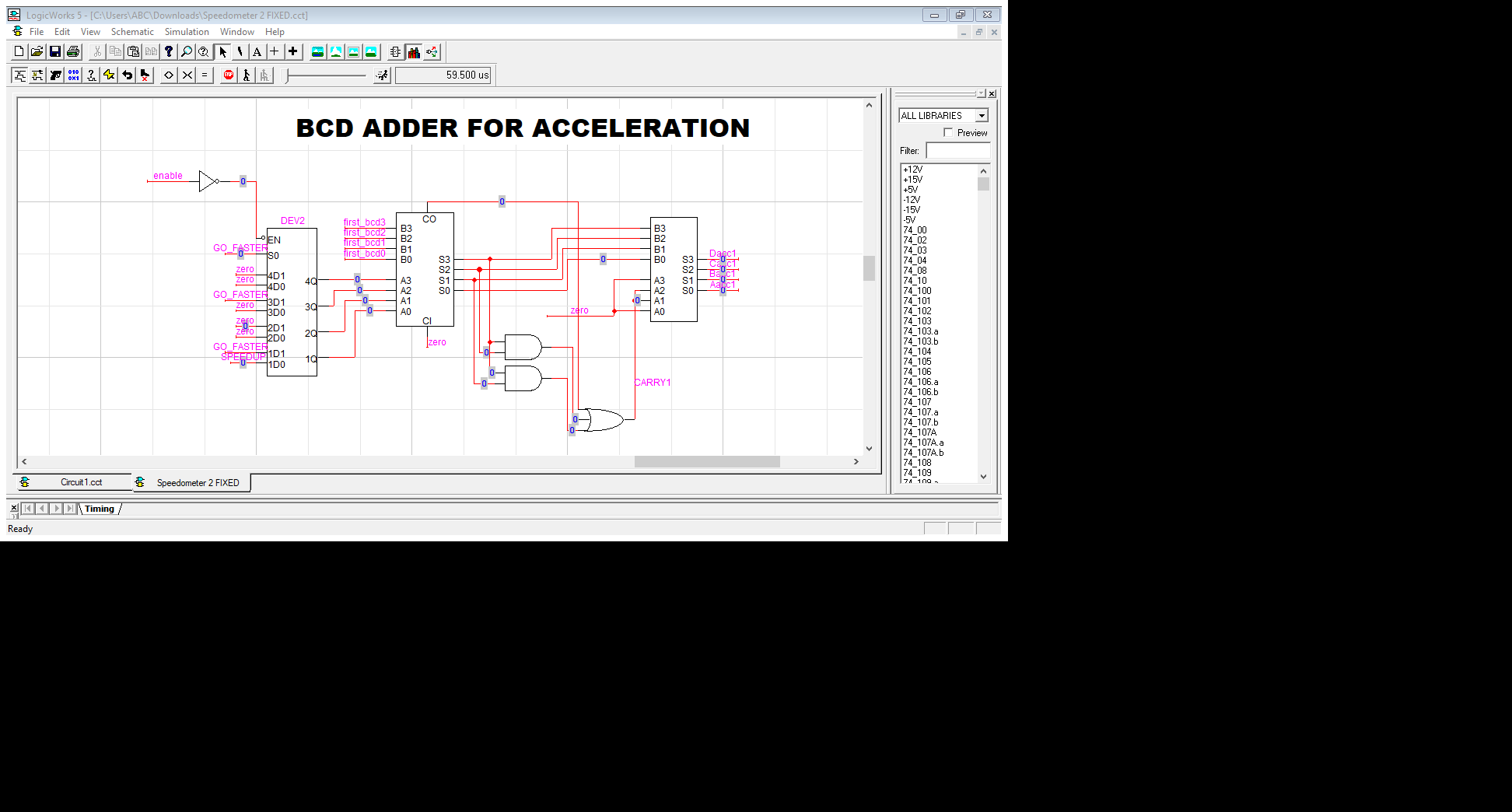
Every 7 segment display has its own register like shown in the above picture.

It consists of some if conditions, it depends on the value that is passed from above interface of speedometer.



Like if Speed up is given input 1 than it will pass the values of speed up adder (will be discussed below) to the register and if slowdown is given input 1 than it will pass the slowdown values. It depends upon the switch that is on or given input 1 it will just pass the values according to it.

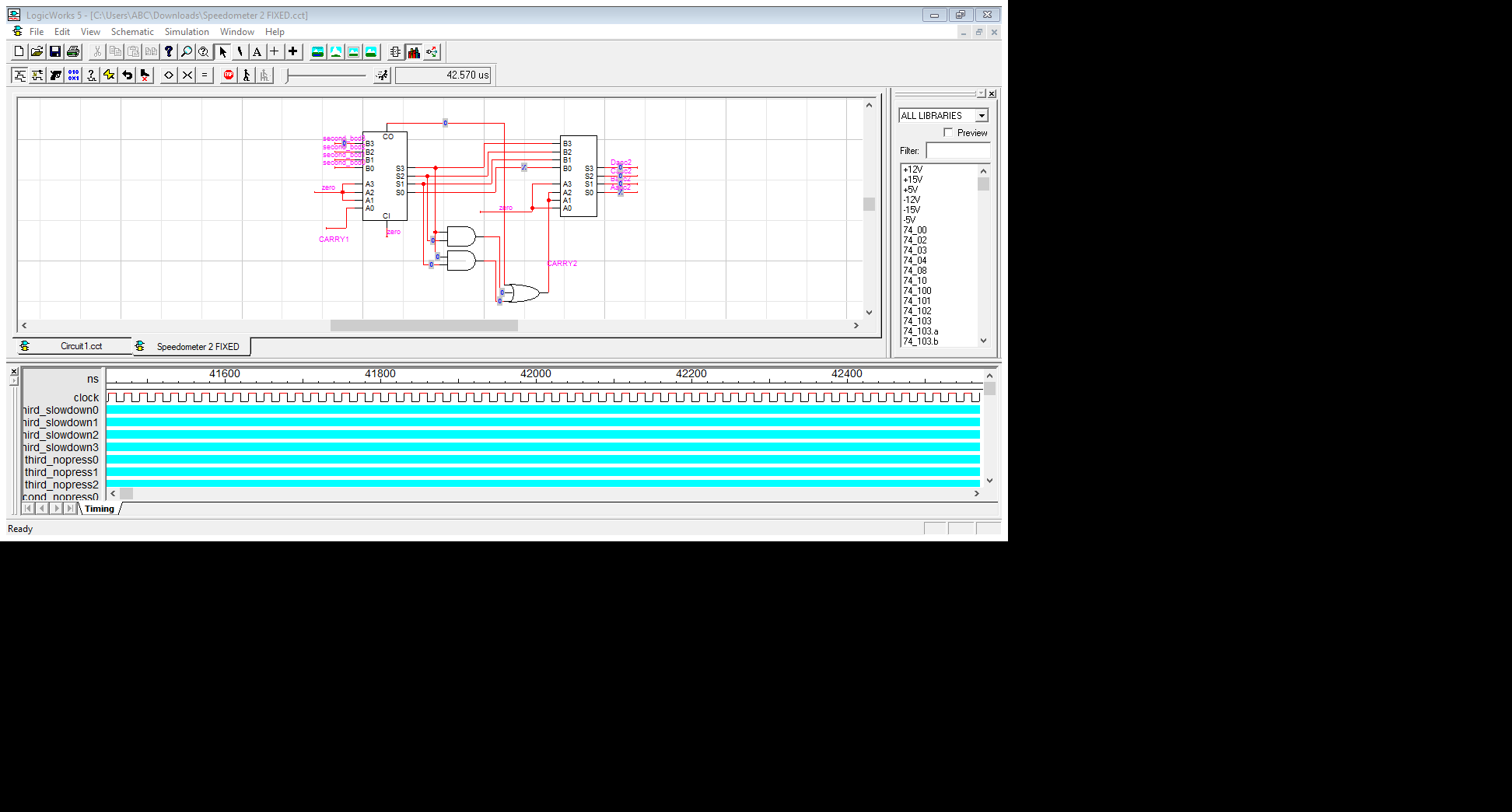
**BCD ADDER FOR ACCELERATION:**



The **MUX** in the start generates output according to the 2 options of fast acceleration (+5) and slow acceleration (+1). If the user selects fast acceleration (+5) than the binary of 5

(0101) will be passed to the adder and if user selects slow acceleration (+1) than the binary value of 1 (0001) will be passed on to the binary adder.

The top 4 input lines of binary adder is the speed that is to be incremented with value of 5 or 1. There are two carry inputs in the other two BCS ADDERS like:

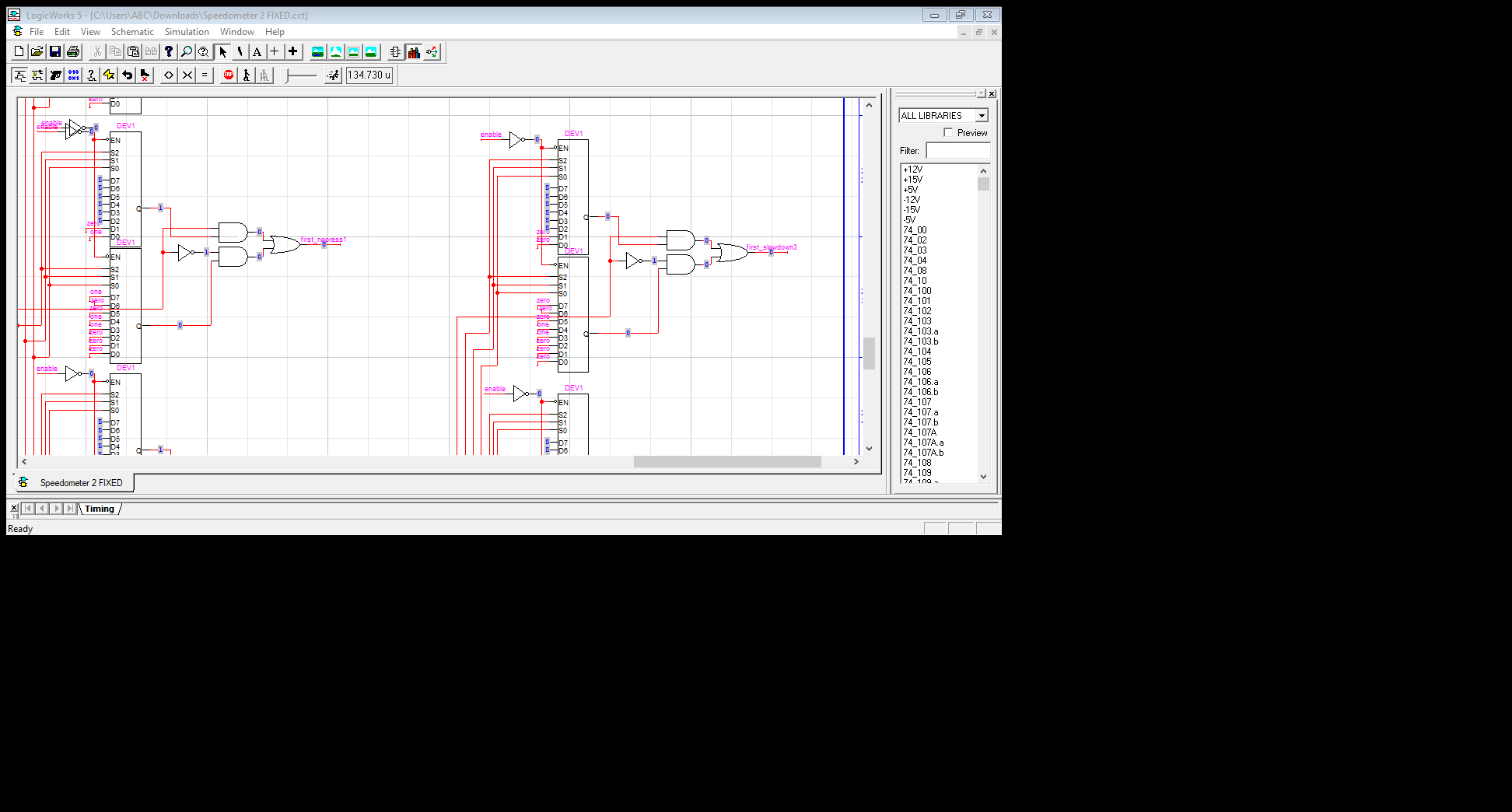


This carry input works only if the output is greater than 9. Just take an example if the binary adder of first 7 segment display is 8 and user selects fast acceleration (+5) which is used to increment 5 in speed which means

**8 + 5 = 13**

This cannot be displayed on 7 segment display as it can display from 0 – 9. So on the first 7 segment display 3 will be displayed and 1 will be taken as a carry in the next BCD adder. The carry will be added in the input of **MUX** of the middle 7 segment display. And the above mentioned working will be same in this adder and the next adder respectively.

**BCD SUBSTRACTOR FOR DECELERATION:**

****

This type of IC’s are been used for each of the 7-segment display.This is the subtractor that was not part of our syllabus but we were told that we had to research about some IC’s and its working on our own. So first of all

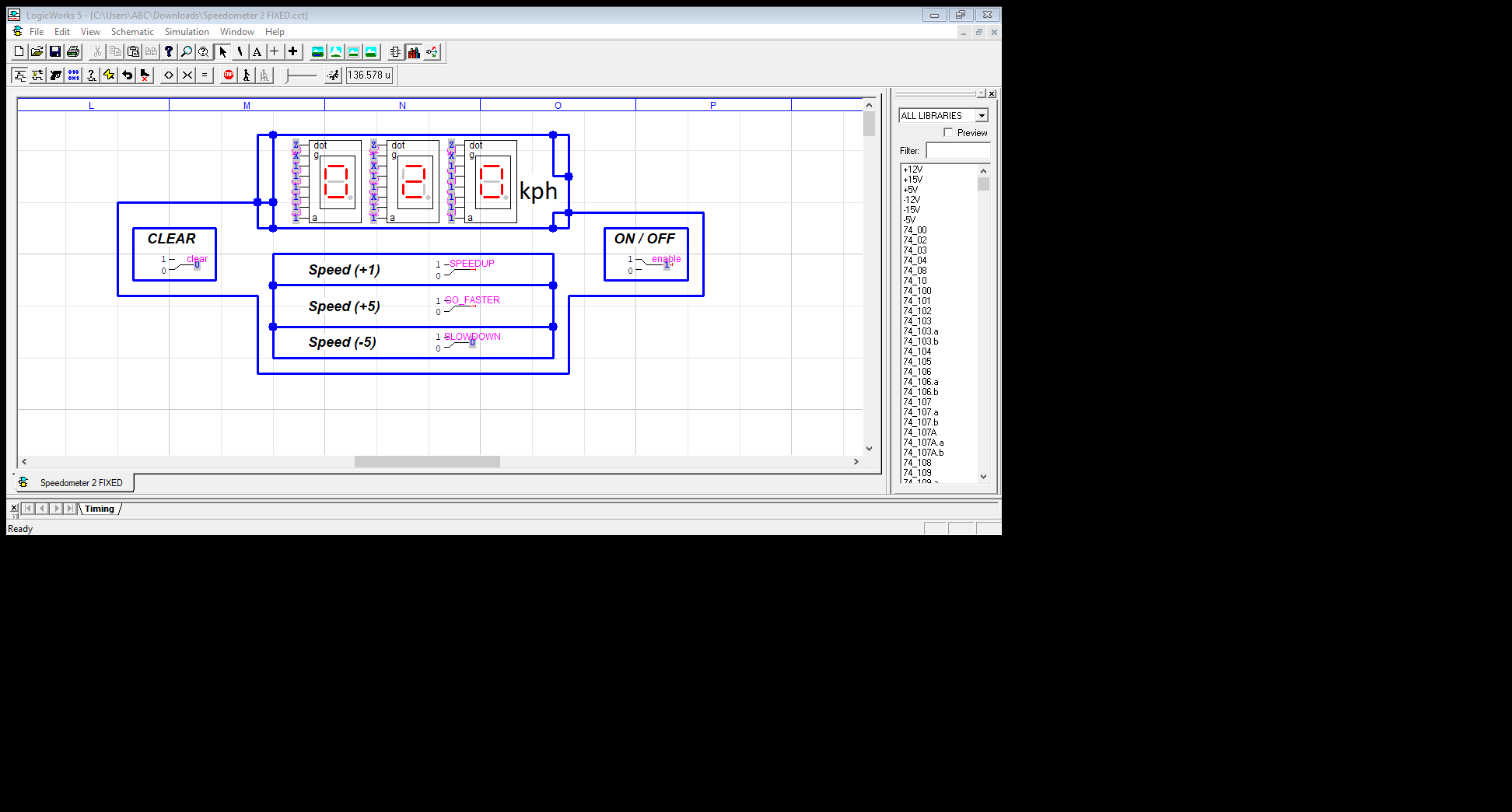
**What is BCD substractor?**

**A substractor circuit is required to perform a subtraction operation on two decimal numbers**. BCD subtraction is slightly different from BCD addition. Performing subtraction operation by taking the 9's or 10's complement of the subtrahend and adding it to the minuend is economical.

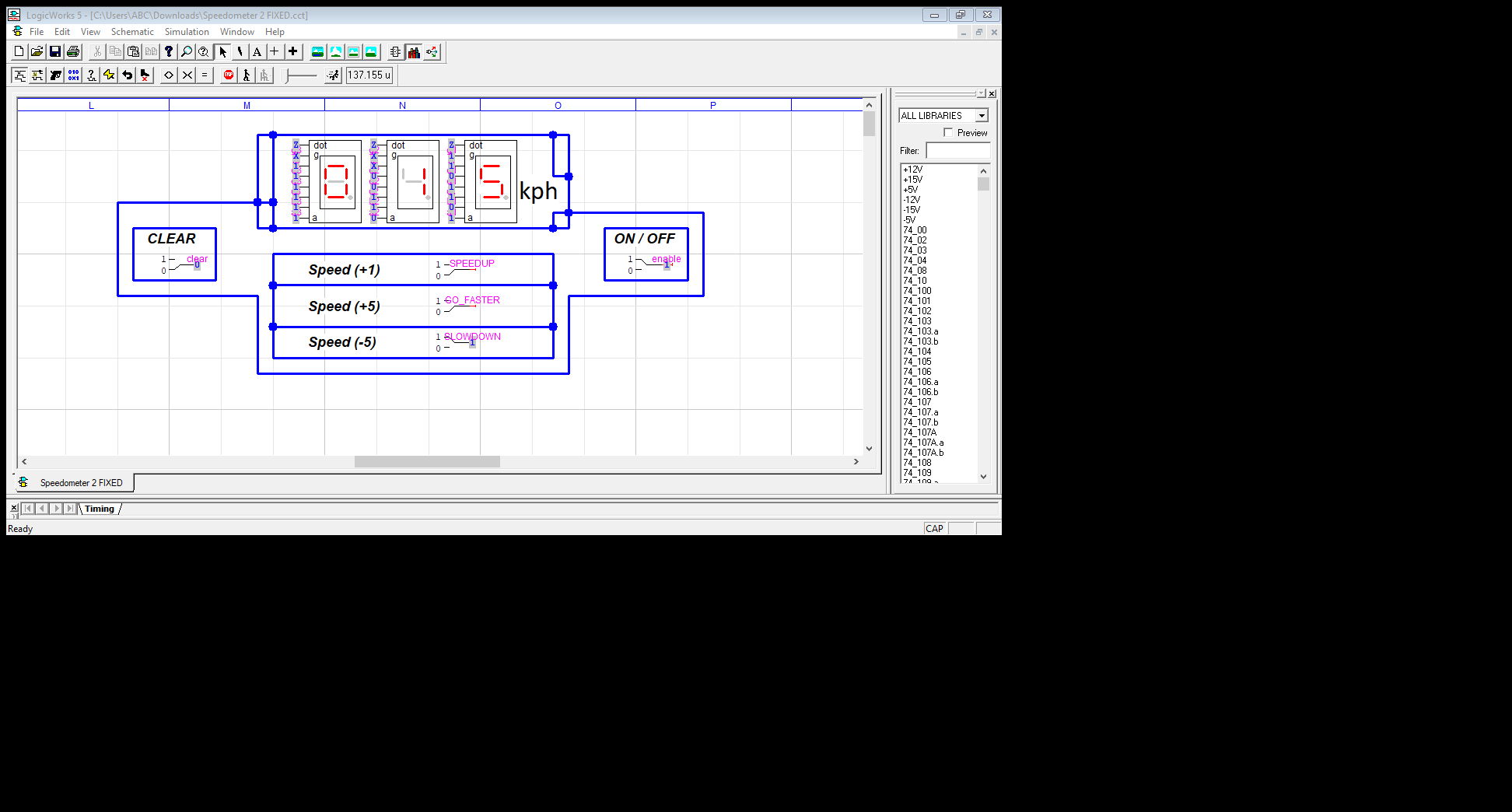
**WORKING:**

* Take 10′s or 9’s complement for B.
* Add it to A using BCD addition.
* If addition is invalid BCD then add 6.
* If carry then add it to the next bits.
* In final result, if carry is occurred then it is ignored.

So this IC’s works under the rule of 9’s complement, it takes the reading from the **switch of Speed (-5)** like take an example if the speedometer is showing the reading of **20** and if the user gives 1 as the input of **Speed (-5)** so the speed will start decrementing with 5, it will take exactly 4 second to reach from 20 to 0 (initial value) depending upon the speed of the clock provided. Like:



**After switch is pressed:**

****

The speed will continue to decrement until it reached 0.

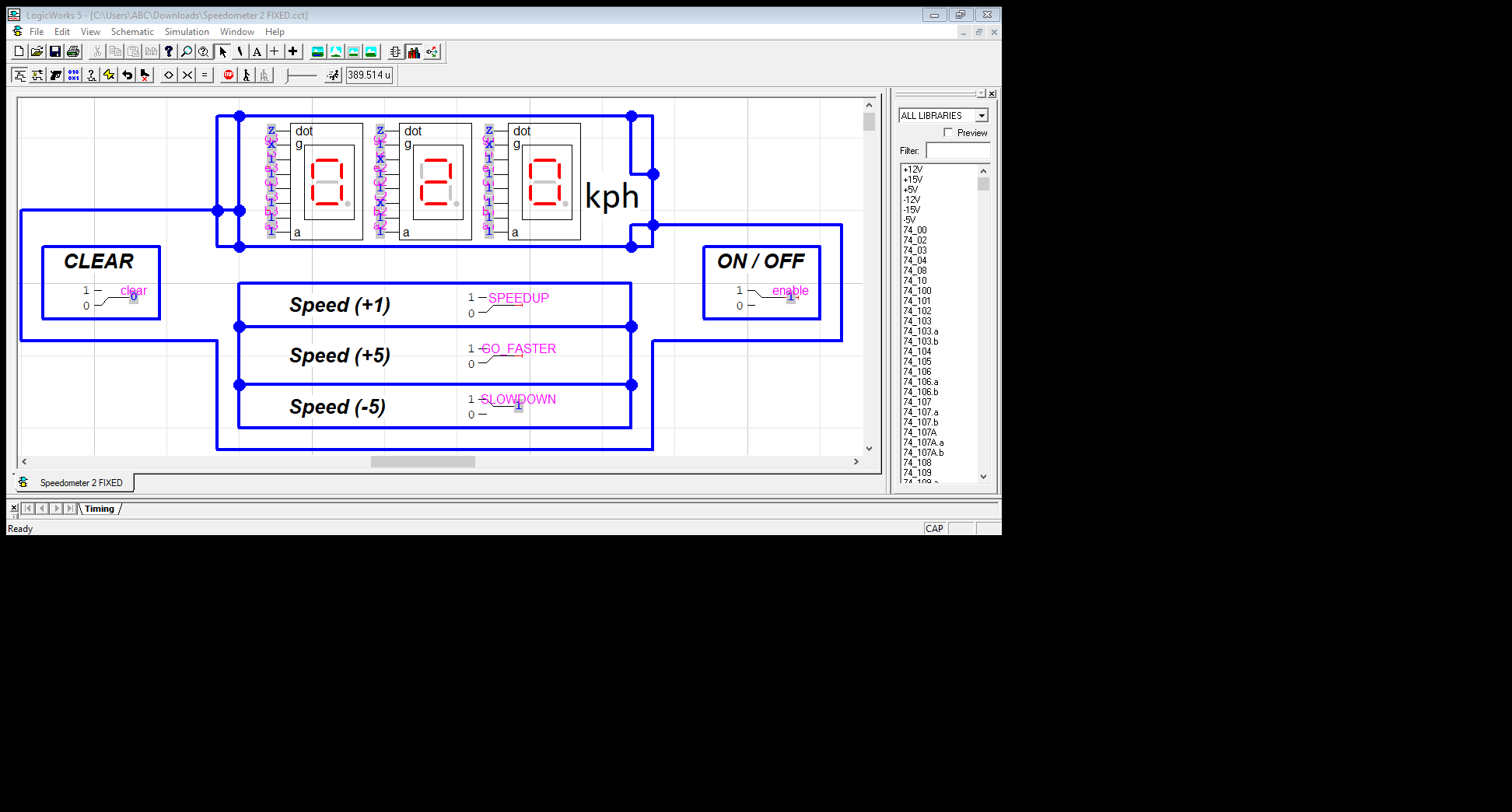
The **Speed(-5) switch** acts as an **enable** for the substractor when it is pressed the circuitry of deceleration or brakes starts working.

**EXAMPLE FROM REAL LIFE:**

Just take an example from your daily life you are driving a car and you want to slowdown your car or stop it either you will stop pressing the accelerator ( the working of binary adder ) that speeds up your car which will cause the car to slowdown but it will be very much time taking process but if you press brakes (Speed(-5)) your car will stop in matter of some seconds.

So to stop the car substractor has its own working either it will stop the car with decrement of 1 or it will stop faster with the decrement of 5.

**7-SEGMENT DISPLAY WORKING**

****

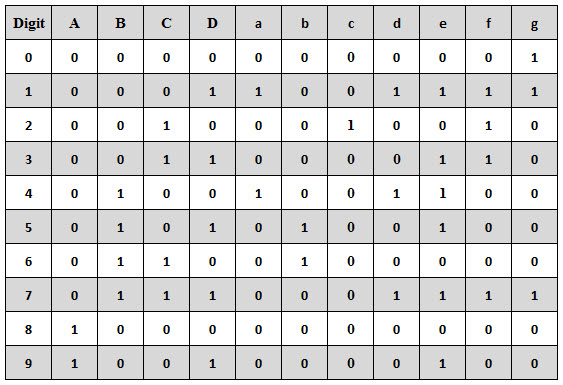
**Step 1:**The first step of the design involves analysis of the common cathode 7-segment display.  A 7-segment display consists of an arrangement of LEDs in an ‘H’ form.  A truth table is constructed with the combination of inputs for each decimal number. For example, decimal number 1 would command a combination of b and c.

**Step 2:**The second step involves constructing the truth table listing the 7 display input signals, decimal number and corresponding 4 digit binary numbers.

The truth table for the decoder design depends on the type of 7-segment display. As we mentioned above that for a common cathode seven-segment display, the output of decoder or segment driver must be active high in order to glow the segment.

The figure below shows the truth table of a BCD to seven-segment decoder with common cathode display. In the truth table, there are 7 different output columns corresponding to each of the 7 segments.

Suppose the column for segment a shows the different combinations for which it is to be illuminated. So ‘a’ is active for the digits 0, 2, 3, 5, 6, 7, 8 and 9.

[](https://www.electronicshub.org/wp-content/uploads/2014/03/BCD-to-common-anode-7-segment-truth-table.jpg)

From the above truth table, the Boolean expressions of each output functions can be written as

**a = F1 (A, B, C, D) = ∑m (0, 2, 3, 5, 7, 8, 9)**

**b = F2 (A, B, C, D) = ∑m (0, 1, 2, 3, 4, 7, 8, 9)**

**c = F3 (A, B, C, D) = ∑m (0, 1, 3, 4, 5, 6, 7, 8, 9)**

**d = F4 (A, B, C, D) = ∑m (0, 2, 3, 5, 6, 8)**

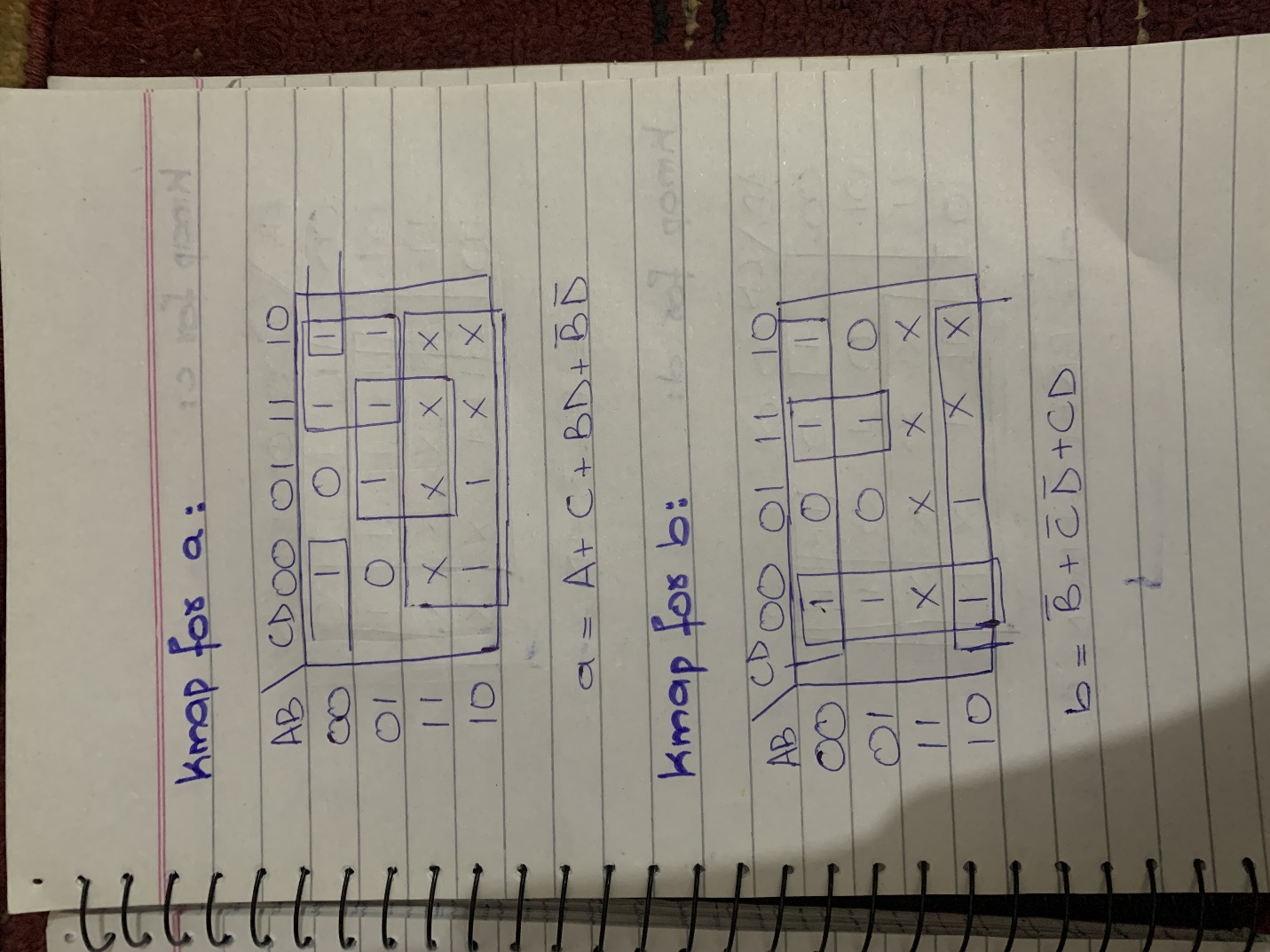
**e = F5 (A, B, C, D) = ∑m (0, 2, 6, 8)**

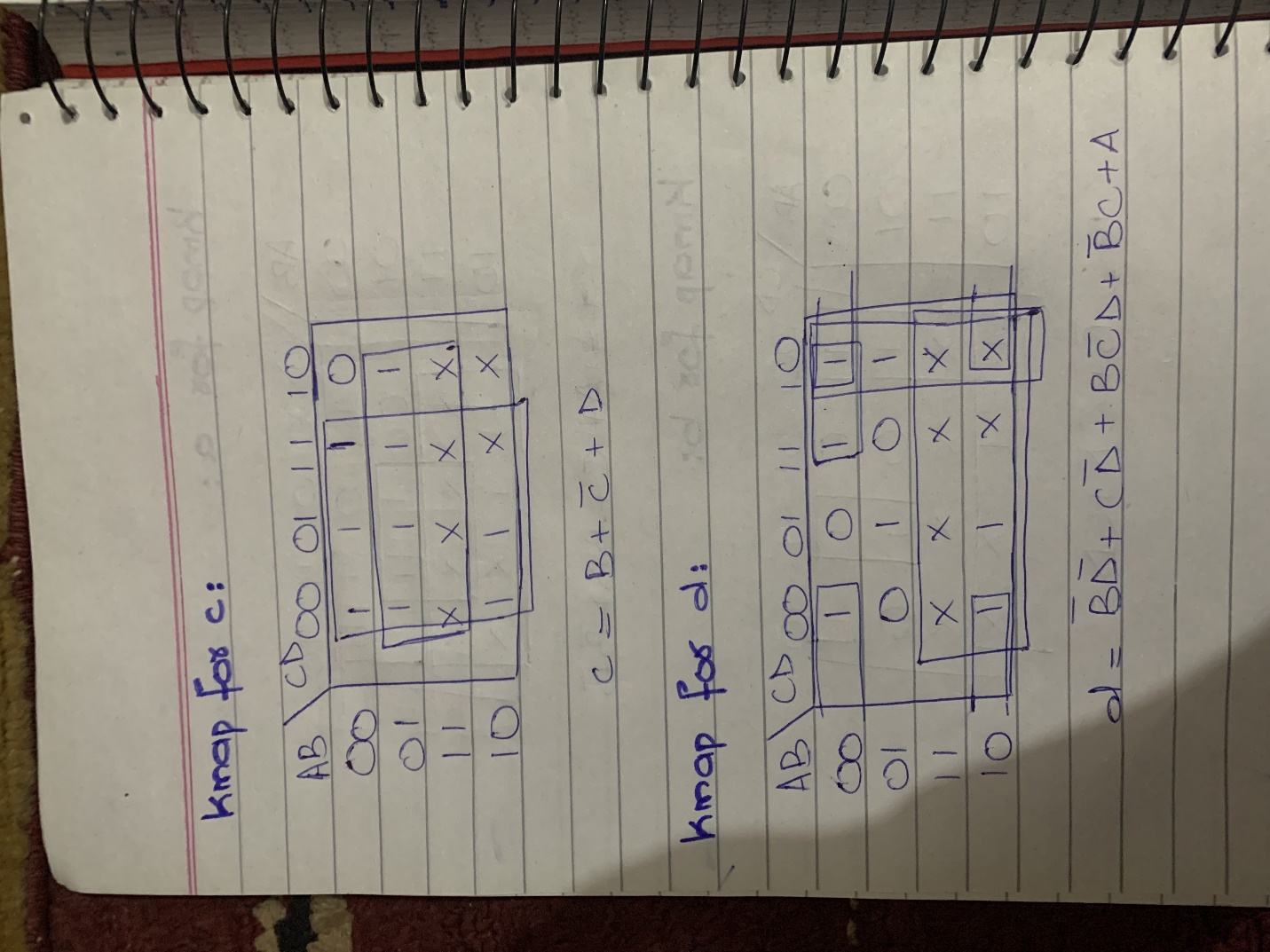
**f = F6 (A, B, C, D) = ∑m (0, 4, 5, 6, 8, 9)**

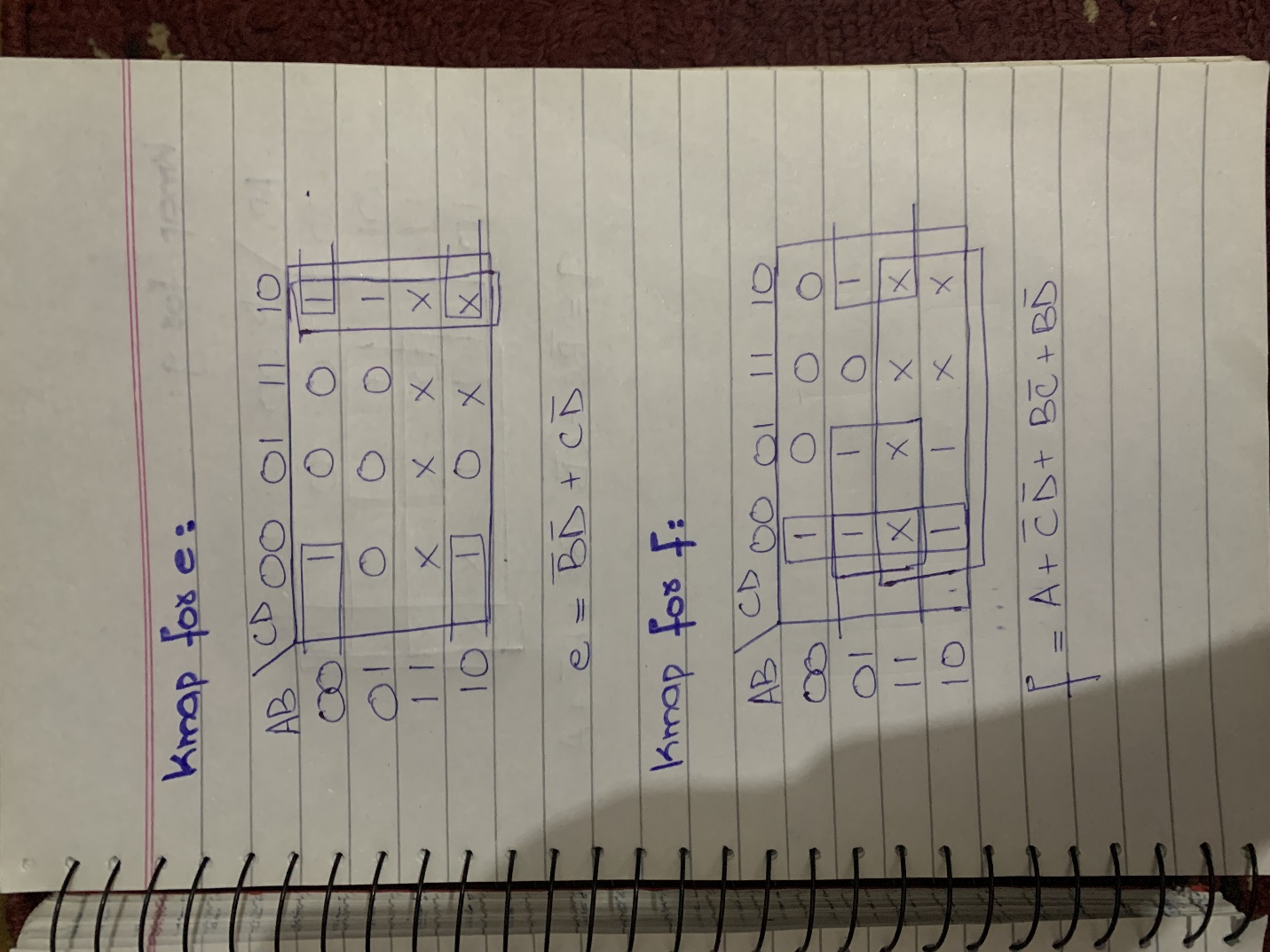
**g = F7 (A, B, C, D) = ∑m (2, 3, 4, 5, 6, 8, 9)**

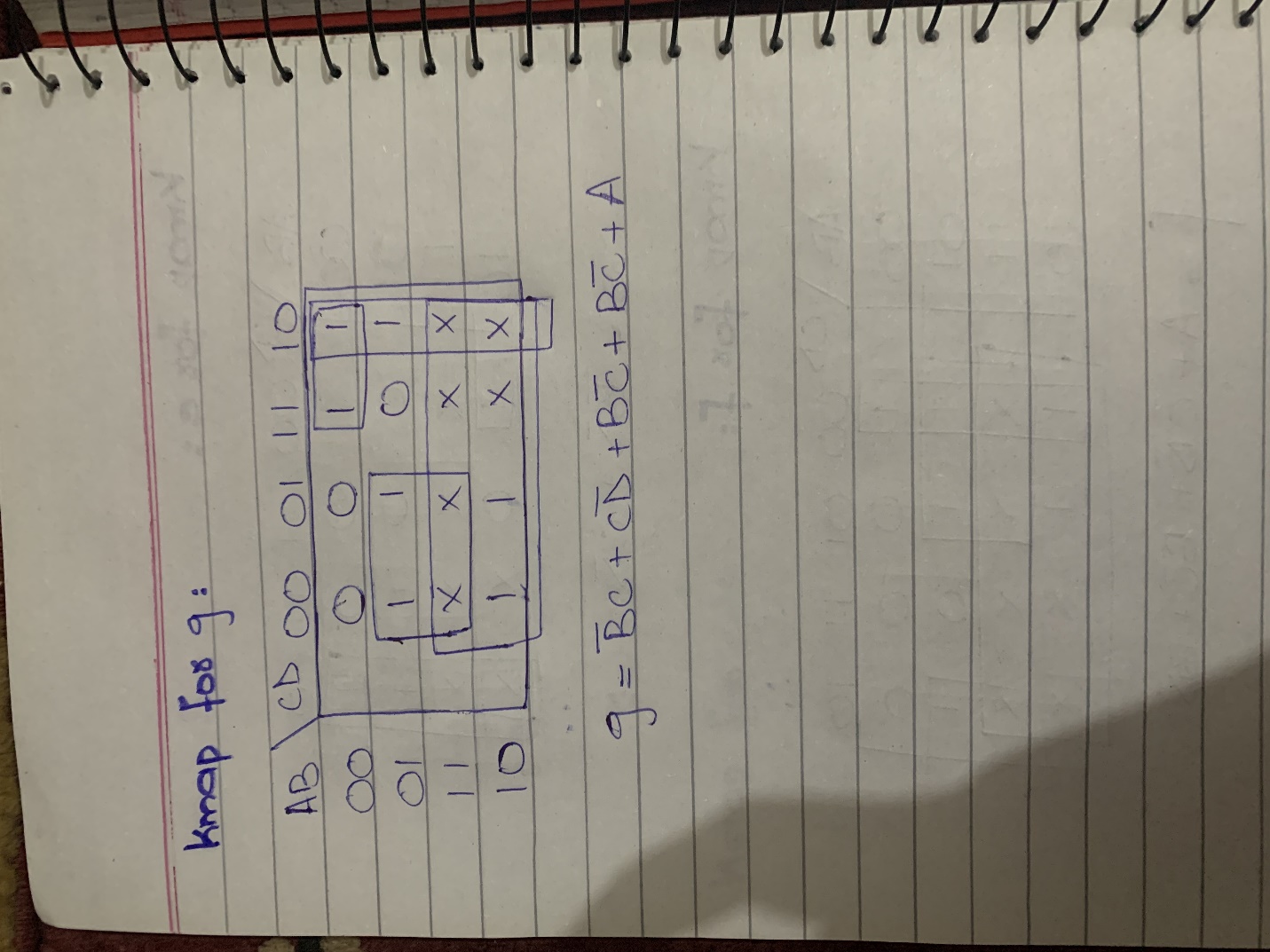
**Step 3:**The third step involves constructing the K-map for each output term and then simplifying them to obtain a logic combination of inputs for each output.

**K-MAP SIMPLIFICATION:**

****





****