LAB 6: RLE ENCODER (DATA COMPRESSION)

Logic of my code, I have created a signal prev_char which stores the character in the previous clock cycle i.e. the character just before the current input, also I have created a count variable of type integer which keeps the count of the current character, that is the value of the count is handled as:

If prev_char is equal to the input (current character) then I incremented the count by 1, and after incrementing I check whether the prev_char is equal to ESC and count 6 then the output corresponding to this is ESC 6 ESC which is store in buffer which I have created as a variable as 1536 downto 0 (how output is given is explained later) and the value of count is set to 0.

Now I check for If the prev_char is not equal to ESC i.e. according to problem statement it can either be the small alphabet or the space character. So after checking whether the prev_char not equal to ESC and count = 5 then output corresponding to this should be ESC 5 prev_char which is stored in buffer at correct position. This were the case with prev_char = input.

Now lets see the other case that is with prev_char not equal to input. Here we know that the value of count will be from [0,5] the count value 0 should be discarded. And other should be printed with correct format.

Here I have checked if prev_char is ESC then for count>0 put ESC count ESC in the buffer.

Now the cases other than ESC, if count = 1 then put prev_char in buffer for count = 2 put prev_char prev_char in buffer. And for count between (2,6) put ESC count prev_char in buffer.

And then set the value of count = 1 and prev_char <= input.

IMPLEMENTATION OF BUFFER

I have created a variable named buffer_v of size 1536 downto 0. The value 1536 is taken by 64x3x8, this variable will be used as buffer which will store the output that should be given inorder.

I have created 2 variables l, r which are the pointers to the positions in the buffer. The l value points the position whose value should be output and the value of r points to the position where the new output should be buffered that is stored.

Now how am I giving the output?

When l+5 less than r then there is a byte in buffer which is to be given as output in current clock cycle. And also the value of valid given as output should be 1 and increment l by 8.

Else the nothing should be given as output that is the output given should be "UUUUUUUU". And that of valid should be 0.

IMPLEMENTATION OF TESTBENCH

In the testbench first the value of in std_logic for rst = 1 is given as:

rst = 1   clk = 0   input = “00000000”;

rst = 1   clk = 1   input = “00000000”;

rst = 0   clk = 0   input = “00000000”;

then opened the input file i.e input.txt and I have implemented a :

while loop for line_count<129;

if line_count = 0 open the output.txt in write_mode.

Now I increment line count by one.

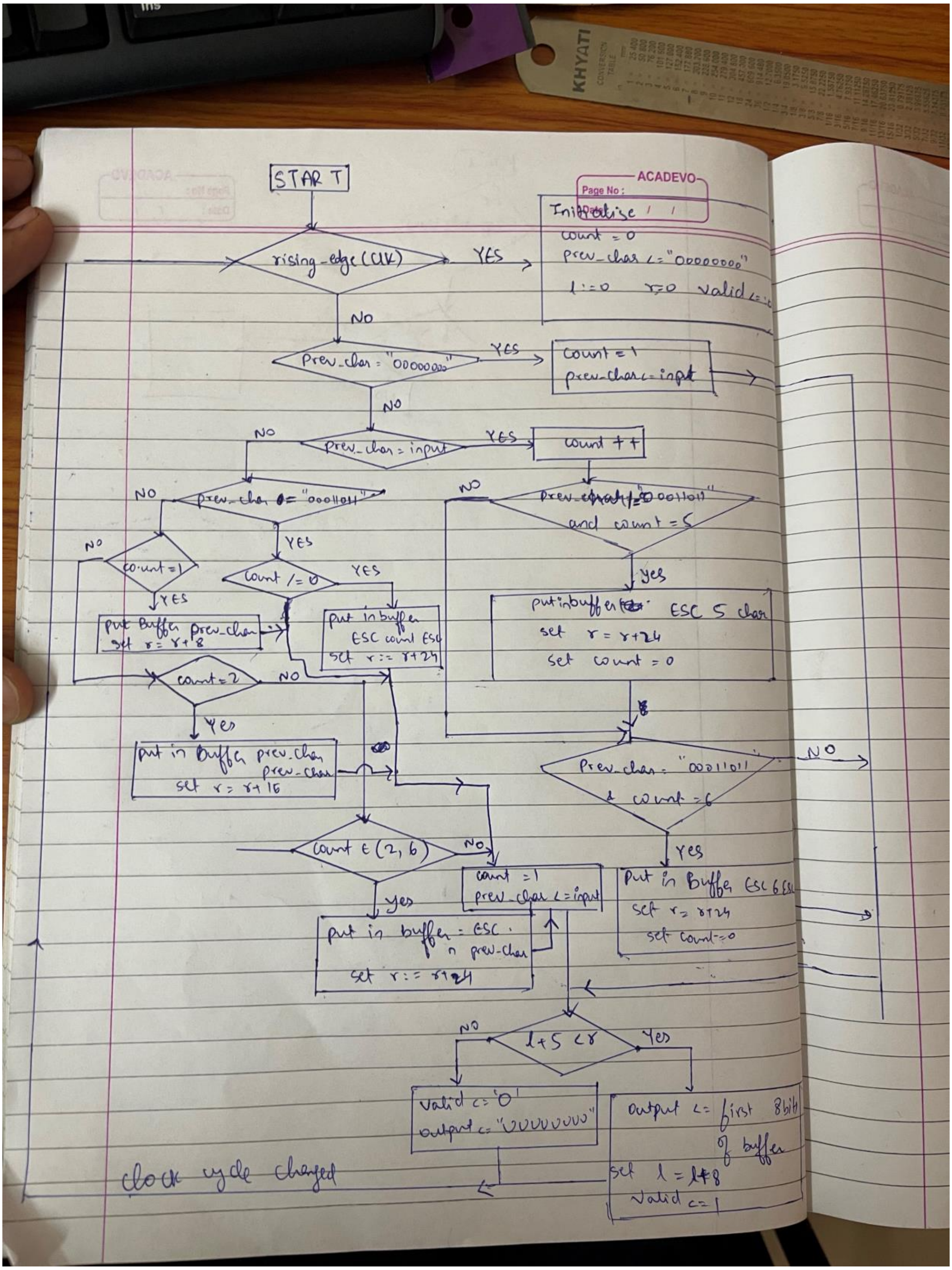
if line_count<65 read from the input file and give the line as input to the code.

If line_count=65 then close input file and I have given input as “00000000” which will not be counted in my code (the way I framed the code).

If line_count>65 just give the input as “00000000” just give input as “00000000”

Now for valid = “1” write the output in the output.txt file. else do nothing.

At end close the output.txt file.


START
rising_edge(CLK)
YES
Initialise
count = 0
prev_char <= "00000000"
l := 0 r = 0 valid <= '0
NO
Prev_char = "00000000"
YES
count = 1
prev_char <= inpt
NO
prev_char = input
YES
count ++
NO
prev_char /= "00011011"
NO
NO
count = 1
YES
put Buffer prev_char
set r = r + 8
YES
count /= 0
YES
put in buffer
ESC count ESC
set r := r + 24
NO
prev_char = "00011011"
and count = 5
yes
put in buffer ESC 5 char
set r = r + 24
set count = 0
count = 2
NO
Yes
put in Buffer prev_char
prev_char
set r = r + 16
Prev_char = "00011011"
& count = 6
NO
Yes
Put in Buffer ESC 6 ESC
set r = r + 24
set count = 0
count ∈ (2, 6)
NO
count = 1
prev_char <= input
yes
put in buffer = ESC .
n prev_char
set r := r + 24
NO
l + 5 < r
Yes
valid <= '0'
output <= "UUUUUUUU"
output <= first 8 bits
of buffer
set l = l + 8
valid <= 1
clock cycle changed

THE END