National School of Applied Sciences of Tetouan

Report: Logic based digital queuing system circuit

Computer Architecture & Assembly Language Unit

Supervised by :
Mrs Seltani Rhizlane

Created by :
Akdim Karim
Allali Meryem
Bouayad Saad
Boubrahmi Houda

ACKNOWLEDGEMENT

We want to take this opportunity to thank all who in one way or another contributed in the completion of this project. First, we give thanks to God for protection and ability to do work.

We are so grateful to National School of Applied Sciences of Tetouan for making it possible for us to study there.

We would like to express our special gratitude and thanks to our Professor Madam Rhizlane who encouraged, supported and directed us, for providing necessary information regarding the project. Her challenges brought this work towards a completion. It is with her supervision that this work came into existence.

We are also thankful to my fellow students whose challenges and productive critics, have provided new ideas to the work.

We would like to express thanks to all readers for giving such time to read this report.

Our thanks and appreciations also go to ours friends in developing the project and people who have willingly helped me out with their abilities

May the Almighty God richly bless all of you,

Contents

Abstract

The basis of this project was to create an optimized digital circuit designed to manage waiting lines.

The product was formed to address the prevalent issues of long waiting lines that could be one of the serious sources of daily stress - for example, long queues in banks, DMVs, tax and billing services, and much more.

One of its main objectives, is to delve into our skills in embedded systems and computer architecture to perceive an operational circuit which can be embedded in public structures/offices to avoid deadlocks and long queues.

This report is divided in two parts:

Theorical division which contains the problem's inspection, the different proposed components that could be involved in making the queue system management circuit, with their truth tables and the time diagrams, which would be useful to develop the product's subfunctions.

Practical division, is in fact, a real image of our theorical work. This part shows best what we have previously developed.
Proteus 8 Professional is the software we have chosen to embed our circuit.
The choice was not by accident, but we found out that it contains everything we need to make this digital concept shines.

After using the Proteus platform to draw the circuit, we provided a demonstration to explain its different functions in all cases (when the lines are stuck, when a window is out of function).

Résumé

Le but de ce projet est de concevoir un circuit numérique optimisé conçu pour gérer des files d'attente dans un établissement ou structure.

Ce produit est réalisé pour minimiser le temps d'attente qui pourraient être une des sources sérieuses de stress quotidien chez les clients - par exemple, de longues files d'attente dans des banques, des services des mines, les services d'impôt et facturation, etc.

L'un des objectifs de ce projet, est d'exploiter nos compétences acquises dans le module « Architecture des ordinateurs » pour percevoir un circuit opérationnel qui peut être incorporé dans des bureaux publics pour éviter les interblocages (deadlock) et les longues files d'attente.

Ce rapport est divisé en deux parties principales:

La section théorique qui définit le problème de l'étude du circuit de gestion des files d'attente, expose les différents composants mis en œuvre pour la réalisation du circuit, ainsi que les tableaux de vérité et les chronogrammes nécessaires pour concevoir un circuit bien homogène et cohérent.

La section pratique, est en fait, non seulement une concrétisation à ce qui est construit auparavant, mais aussi une mise en valeur d'un travail d'équipe bien solide.

Proteus 8 Professional est le logiciel que nous avons choisi pour construire notre circuit. C'est un logiciel qui permet de réaliser des simulations des circuits logiques.

La réalisation de ce projet nous a épanoui sur plusieurs perspectives et améliorations que nous pourrons, par la suite, ajouter à notre système afin de l'optimiser ou de développer d'autres fonctionnalités opérationnelles.

Introduction

In the past decades, Technology has known an epic development which has brought many changes in human life. It's meant for bringing innovation and reduces the effort to accomplish a task and has to be used to make a way that smoothens the human's life.

Waiting in line is part of our daily life. Waiting lines are a common situation, they may, for example, take the form of cars waiting in a highway or while going to work. Whether at a grocery shop or waiting in line at the bank, at an amusement park or at Maroc Telecom Office to pay phone bills, we spend a lot of time waiting. Wait-time depends on the number of people waiting before you, the number of servers serving line, and the amount of service time for each individual customer.

Understanding waiting lines or queues and learning how to manage them is one of the most important areas in operations management, and queuing management.

This project has been designed to achieve a logic based circuit of a waiting line. The main aim of this project is to reduce the efforts, waiting time, and to facilitate the service and decrease the amount of time of each individual. This project would be accomplished using basic logic components and devices, like J-K Flip Flop or D latch to conceive the counters and to build registers, BCD 7-seg display, etc. Also Proteus 8 Professional to draw the different circuits.

Our project is going to manage the queuing lines of three windows which receive a number of clients that goes to a thousand of people a day, in an agency. The aim is to avoid deadlocks and long-waiting lines in order to serve more people in less time.

Theorical treatment of the circuit

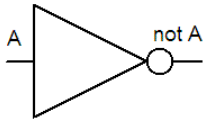
The following analysis will expose the multiple definitions of the devices we used to build this circuit. This will provide truth tables, to explain the way it functions, and behaves with the whole.
Chronograms are used also to describe how an asynchronous counter secretes its values and how it progresses over time.

Basic logic gates

NOT :

NOT gate is a logic gate which implements logical negation, it forwards the negation of its input.

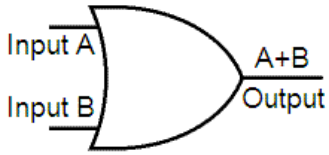
Input	Output
A	NOT A
0	1
1	0



OR :

A logic OR gate or inclusive-OR gate is a type of digital logic gate that has an output which is normally at logic level "0" and only goes "HIGH" to a logic level "1" when one or more of its inputs are at logic level "1". The output, Q of a "logic OR gate" only returns "LOW" again when all of its inputs are at a logic level "0". In other words, for a logic OR gate, any "HIGH" input will give a "HIGH", logic level "1" output.

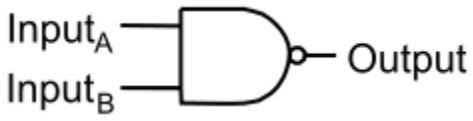
Input 1	Input 2	Output
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	0



NAND :

The logic NAND gate is a combination of the digital logic AND gate with that of an inverter or NOT gate connected together in series. The NAND (Not – AND) gate has an output that is normally at logic level "1" and only goes "LOW" to logic level "0" when ALL of its inputs are at logic level "1". The logic NAND gate is the reverse or "*Complementary*" form of the AND gate we have seen previously.

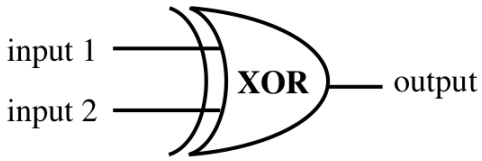
Input 1	Input 2	Output
A	B	A NAND B
0	0	1
0	1	0
1	0	0
1	1	0



XOR :

An XOR gate (sometimes referred to by its extended name, Exclusive OR gate) is a digital logic gate with two or more inputs and one output that performs *exclusive disjunction*. The output of an XOR gate is true only when exactly one of its inputs is true. If both of an XOR gate's inputs are false, or if both of its inputs are true, then the output of the XOR gate is false.

Input 1	Input 2	Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0



Flip flops

Flip flops are actually an application of logic gates. With the help of Boolean logic, you can create memory with them. Flip flops can also be considered as the most basic idea of a Random Access Memory (RAM). When a certain input value is given to them, they will be remembered and executed, if the logic gates are designed correctly. A higher application of flip flops is helpful in designing better electronic circuits.

J-K Flip flop:

The R-S flip flop is the simplest flip flop of all and easiest to understand. It is basically a device which has two outputs, one output being the inverse or complement of the other, and two inputs. A pulse on one of the two inputs to take on a particular logic state.
The outputs will remain in this state until a similar pulse is applied to the other input.
The two inputs are called SET and RESET.

"J-K flip flop is a modified version of R-S flip flop with no invalid or illegal output state."

Inputs			Outputs		Comment
Clock	J	K	Qn	NOT Qn	-
↑	0	0	Qn-1	NOT Qn-1	Latch
↑	0	1	0	1	Reset
↑	1	0	1	0	Set
↑	1	1	NOT Qn-1	Qn-1	Toggle
*	X	X	Qn-1	NOT Qn-1	Memory

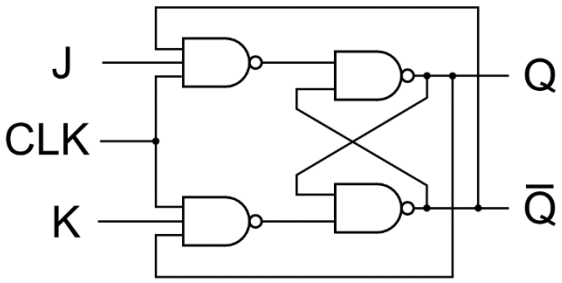


Figure 1: J-K based on logic gates

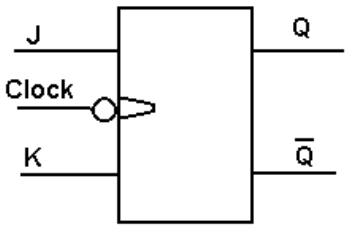


Figure 2: J-K flip flop

D Flip flop:

The D flip-flop tracks the input, making transitions with match those of the input D. The D stands for "data", this flip-flop stores the value that is on the data line. It can be thought of as a basic memory cell.

As long as the clock input is low, changes at the D input make no difference to the outputs. The truth table shows this as a 'don't care' state (X). The basic D Type flip-flop is called a level triggered D Type flip-flop because whether the D input is active or not depends on the logic level of the clock input. Provided that the CK input is high (at logic 1), then whichever logic state is at D will appear at output Q. If D = 1, then S must be 1 and R must be 0, therefore Q is SET to 1. Alternatively, If D = 0 then R must be 1 and S must be 0, causing Q to be reset to 0.

Clock	D	Qn
0	X	Qn-1
↑	0	0
↑	1	1

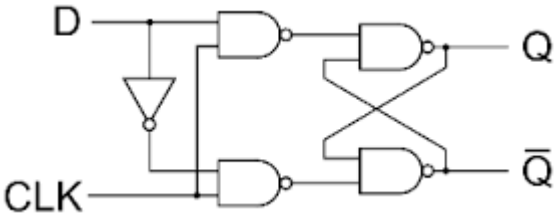


Figure 3: D flip flop based on logic gates

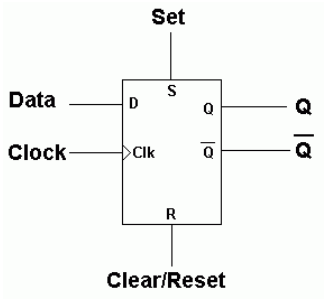


Figure 4: D flip flop

Asynchronous Counter

An asynchronous counter derives its name from the fact that the output of the counter is not directly dependent on the applied clock.

Another way to explain this would be to say that the clocking signal is provided only to one of the flip flops and the remaining (n-1) flip flop's clock pins are driven by the output of the previous stage. So the output of the (n-1) flip flops changes asynchronously to the clocking signal of the system.

For the conception of the counters we need to embed in our circuit, 4 bits are required to count from 0 to 9 (knowing that a 4 bits' counter can count from 0 until 15). In other words, it's a modulo-10 asynchronous counter.

We make the choice of a J-K flip flop instead of a D latch because the toggle state can be activated effortlessly at a forced high states inputs. But you still can do the same with a D latch or a T flip flop (Trigger).

In order to know the number of flip flops needed to make your customized asynchronous counter which counts from 0 to T, the rule below must be respected :

$\mathbf{2^{N-1} - 1 \leq T \leq 2^{N} - 1}$ (N : number of flip flops needed)

Modulo-10 J-K based asynchronous counter:

This counter has the same composition of the 4-bits asynchronous counter, with a reset entry of an AND combination (or NAND if the reset entry is reversed) of the 2nd Q output and the 4th Q output of each flip flop which compound our counter. Note that, the flip flops are mounted in a waterfall structure.

Here is a representation of a modulo10 J-K based asynchronous counter with a reversed entry.

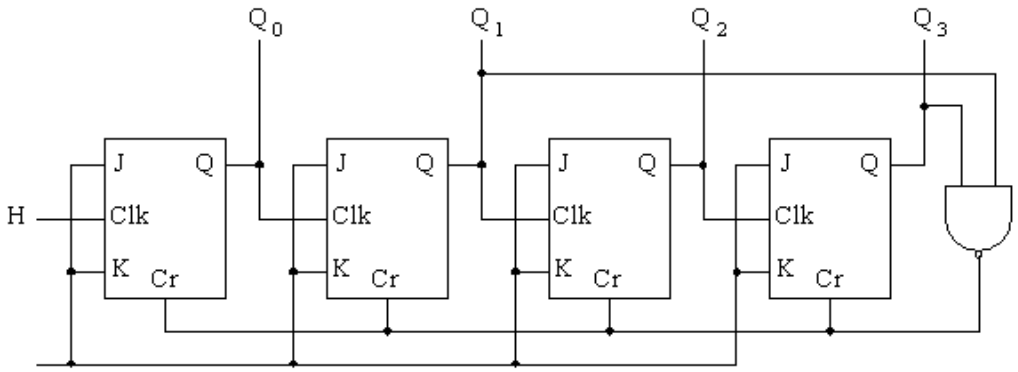


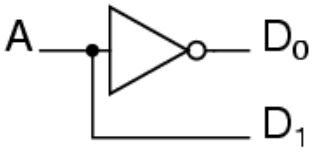
Figure 5: Mod-10 Asynchronous Counter

Number of pulses	QD	QC	QB	QA
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10 (Reset)	0	0	0	0

Decoder

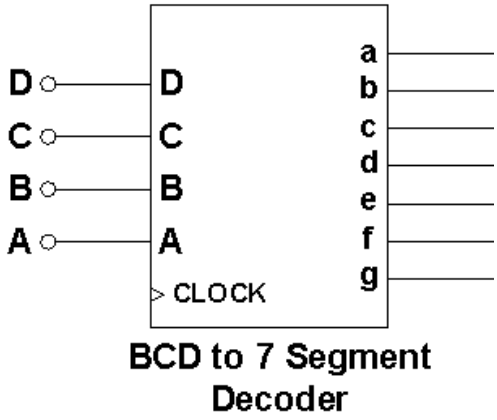
A decoder is a circuit that changes a code into a set of signals. It is called a decoder because it does the reverse of encoding, but we will begin our study of encoders and decoders with decoders because they are simpler to design.
A common type of decoder is the line decoder which takes an n-digit binary number and decodes it into 2^n lines. The simplest is the 1-to-2-line decoder.

A	D1	D0
0	0	1
1	1	0



7 Segment Decoder

It is the one we are going to use in our circuit. This decoder allows, thanks to its outputs values, and with coordination with the BCD display (which is going to be evoked later), to lit the 7 LEDs of the BCD display when one or more outputs are active.

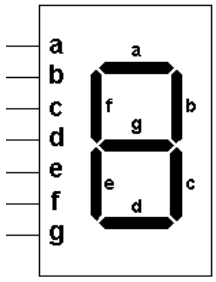


BCD to 7 Segment Decoder

	BCD Inputs				Segments Outputs						
Decimal	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1

BCD Display

Typically 7-segment displays consist of seven individual colored LED's (called the segments), within one single display package. In order to produce the required numbers or HEX characters from0 to 9 and A to F respectively, on the display the correct combination of LED segments need to be illuminated. A standard 7-segment LED display generally has 8 input connections, one for each LED segment and one that acts as a common terminal or connection for all the internal display segments. Some single displays have also an additional input pin to display a decimal point in their lower right or left hand corner.

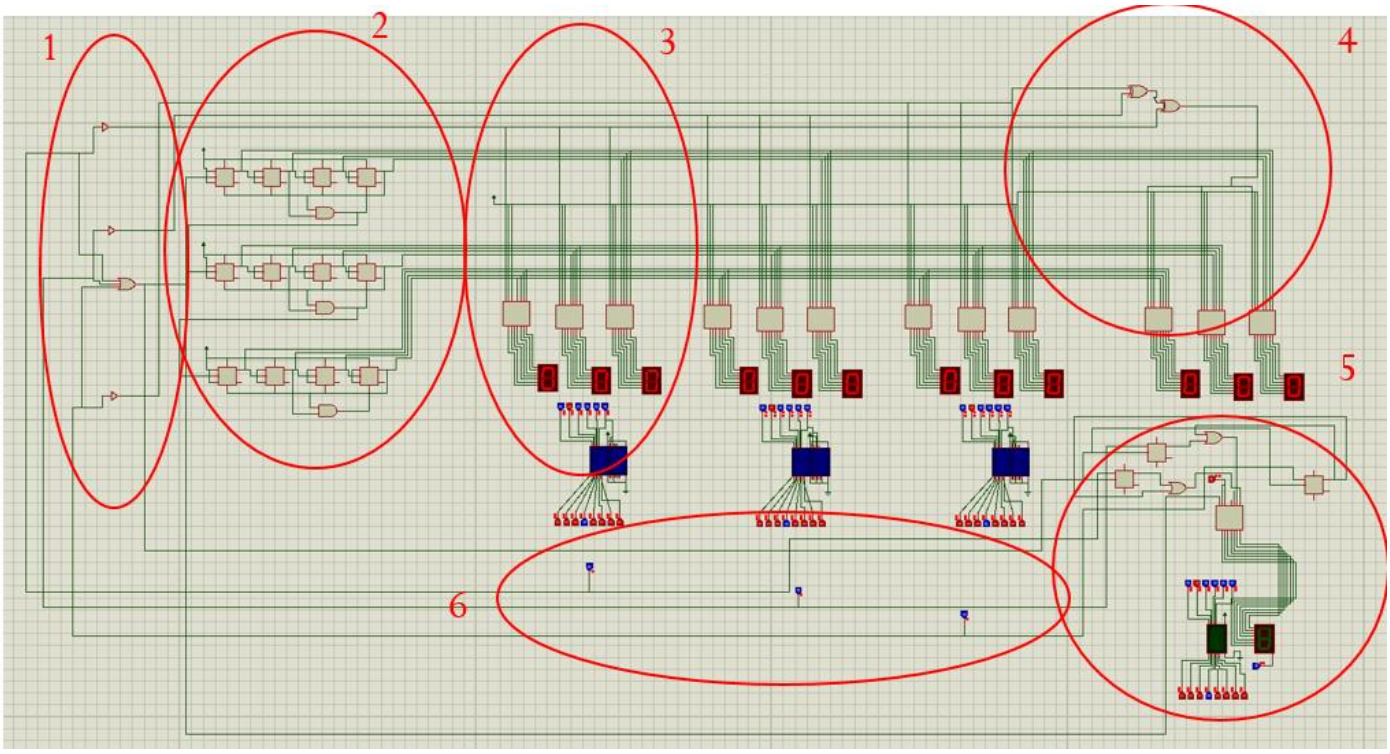


7-Segment
LED Display

Practical treatment of the circuit

Now comes the pragmatic step of the project, which consists to merge the previous syntheses in the perfect combination to answer all the necessary queries.
Although, this is not the only composition of the elements, multiple schemes could be reliable, it all depends on the followed thinking algorithm, since we are not handcuffed by any complexity criteria or contract specifications.

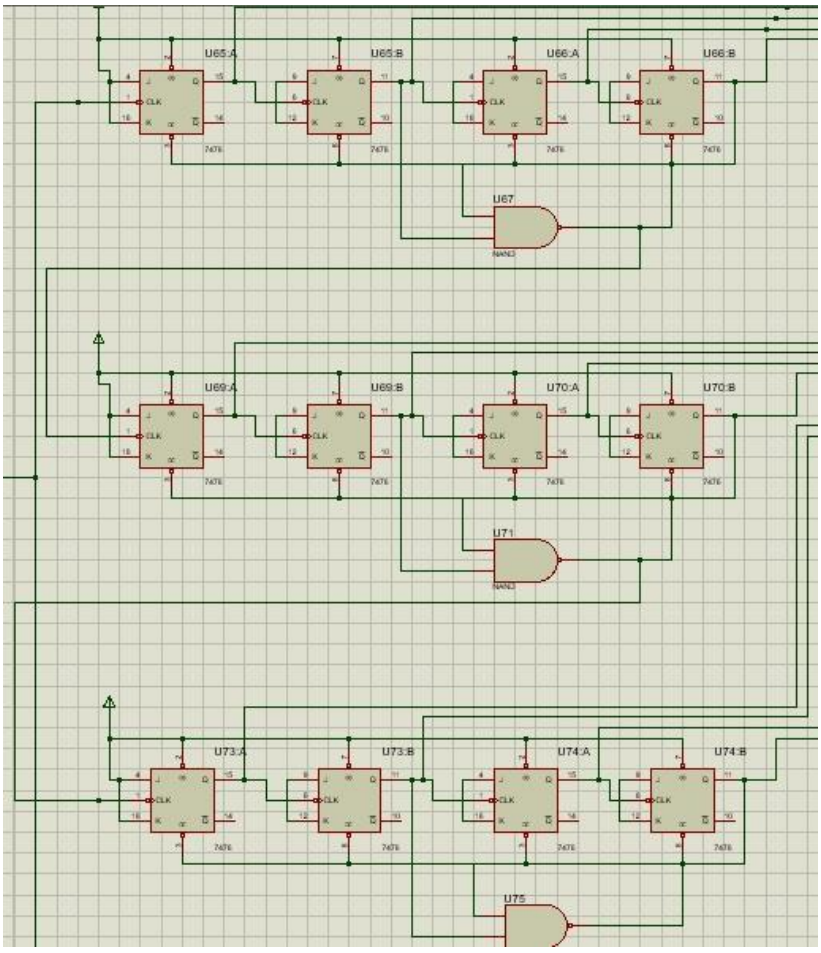
Adopting an inductive structure, we are going to show the whole circuit, and then we are going to subdivide it into blocks with a same function.



Counter block (2)

The three counters are made to increase the output number with every toggle by 1. Going from the top to the bottom, the first counter determines the number of units of the client. The second counter has to determine the number of decades, and the third determines the number of hundreds.

Every MSB (most significant bit) output of each counter in connected to every window display. We will need an extra combination of logic gates to hand turns from window to another without displaying the same number twice in more than one window.

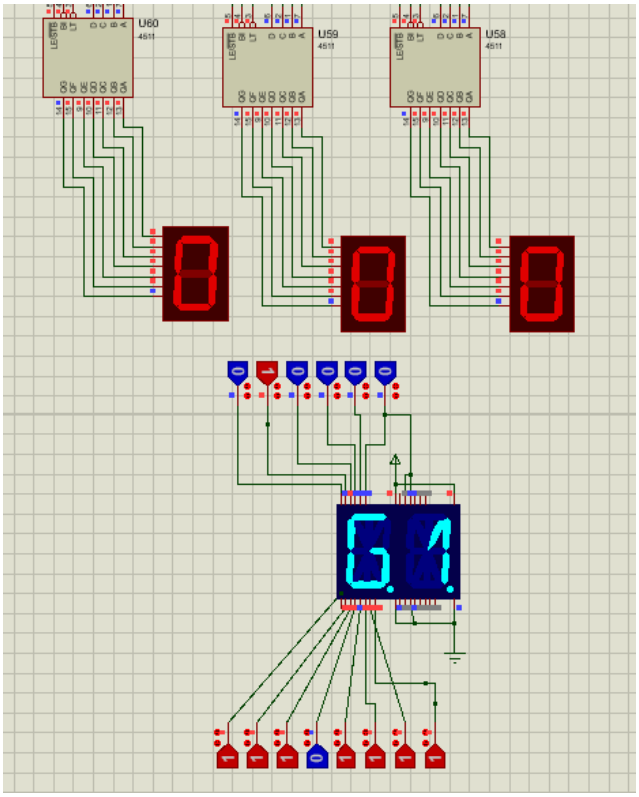


Units output

Decades output

Hundreds output

Window block (3)

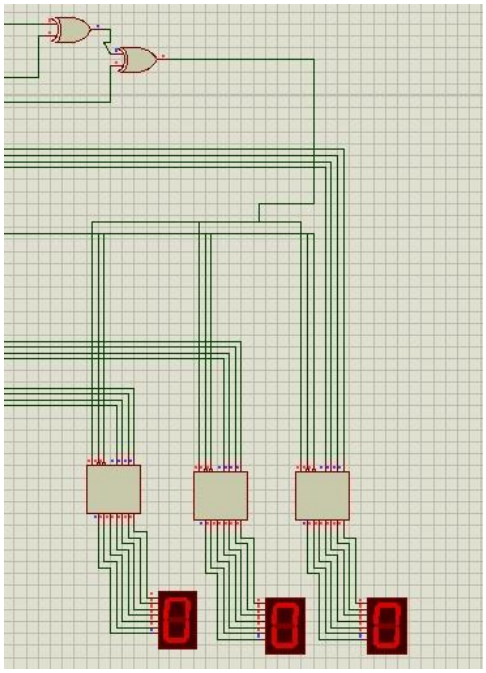


The window block has two elements, the first one is the serving client number which is composed of 3 digits, from the right to the left, the first one is the unit digit the middle one is the decade digit and the one in the left represents the hundreds digit.

The BCD Decoder is used to decode the signal coming from each counter.

In the other hand, we used the fixed input values 14 segment BCD display (the blue display) to exhibit the windows number (G1 means Guichet 1).

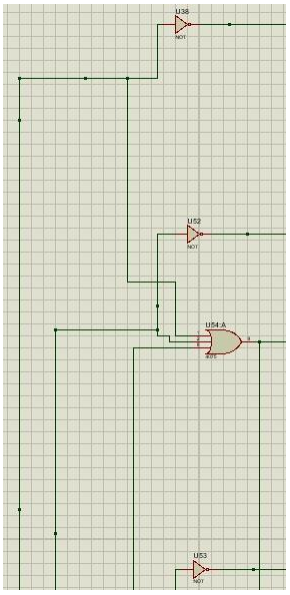
Next client block – waiting room (4)



This block it very important in our circuit, it affects values of the window incremented at the toggle, to be exhibited at the waiting room displays. This way, a new client is notified so he can get to the line of the corresponding window.

The block has 2 XORs, built to conduct only one value while one toggle button is pushed.

Synchronization block (1)

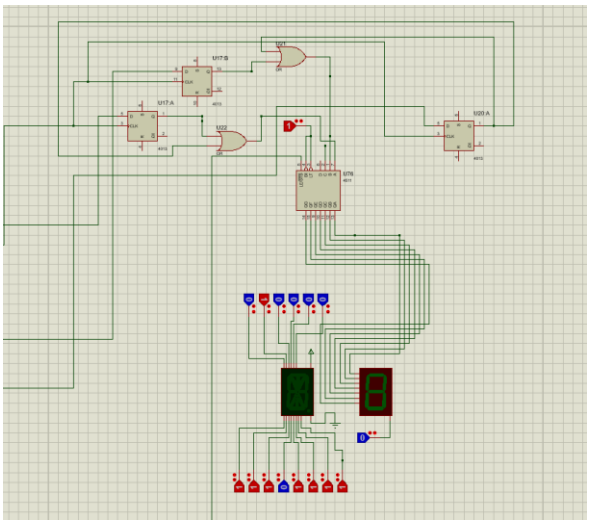


Synchronization circuit is the one who commands the pipe going to the entrance LE of the decoder. So it can increase for the corresponding window, while saving the old values for the other windows. Once the toggle button is pushed, a notification is raised, and only one window increases its value, and goes directly to the one after the last served client's number.

The OR 3 input gate behave the same as two OR 2 input gates.

While one button is pushed, the signal is equal to 1 at the output, and the value displayed at the window increases!

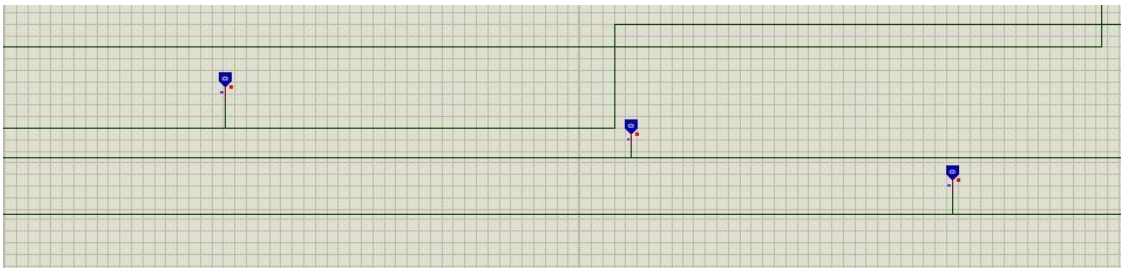
Next available window block – waiting room (5)



Three D latches are implemented to save the number of the next available window and keep it shown on the display. The expulsion of the three latches, can destroy the main functionality of this block, which is to provide the next available window number. The 2 OR used in serial structure, so to alternate between the 3 command buttons.

We are also using the green LCD, to make the difference between the displays used in windows, and the one used in the waiting room.

Command block (6)



The command area is based on 3 toggle buttons, each one commands a window. From left to right, the first one is conducting the window 1 to the left one which is commanding the window 3, the second is in the middle.

Each button is pushed when the resource is available, or when the window is ready to serve a new client. So the signal shift 0 to 1 and 1 to 0 quickly, the high edge is being recognized by our flip flop, and then by our counter, which increments the output value and goes further, in wave, to devastate the whole circuit, and answer to the logical function of the it.

Weaknesses and perspectives

This project was a challenge for us as engineering students to test our skills in this matter, but as expected we faced some difficulties such as:

- We chose the Proteus 8 Professional software as a platform to create our circuit but it was pretty difficult to work with since the circuit was big enough and to handle the assembling between devices.
- We were not allowed to use embedded devices to create our circuit.
- Because we didn't study the LED display of three numbers, we had to replace it with 3 LED displays for each number which made the circuit more complicated and less optimized.

Although this solution is not perfect it can be enhanced with adding few more options like a display table that shows the number of the following customer in the waiting room make him get ready to get to the window, this will contribute to the efficiency of the client's service.

Conclusion

Through our project, we were able to create a queuing management circuit that has been a revolution in managing waiting lines crisis in most of the client's service agencies.

Making this project real has been an opportunity to apply our knowledge and skills in computer architecture and assembly language not only by using different flip flops, logical gates, decoder display and asynchronous counters, but also, by seeking for more information and trying to make this solution as reliable and useful as it could be.

However, this solution cannot be considered as the ultimate one to the queuing management problem since it can be treated in different ways. And as we mention it before, the solution can be developed differently due to the diversity of the contract specifications; whether by adding more options or it in order to minimize the cost of the final product.

Although, this concept could be implemented in any real structures that struggle from the queuing lines problem, and to make this work, it should be commended using task management and other convenient algorithms for this aim.

Bibliography & Webography

Books :

__ " Digital design", 5th Edition by Morris Mano.

__ " Digital Electronics And Logic Design", By A.P.Godse

Websites :

__ www.allaboutcircuits.com

__ www.electrical4u.com/digital-electronics

__ www.physics.leidenuniv.nl/digital-electronics