

Circuito votador, de turno y comparador (junio de 2020)

Carrillo Bryan, Guzmán Bryan, Hernández Bryan

Resumen – En este artículo se presente el diseño de 3 circuitos que tratan de solucionar un problema como el fallo de cualquier sistema de seguridad de un avión, con un circuito votador el cual nos permita escoger la salida de 3 circuitos dado que uno haya fallado, también se implementará un circuito en el cual pueda ingresar la hora en código binario y que en la salida me de el turno en el que el trabajador se encuentra en el momento, y por último se diseñará un circuito el cual compara dos números binarios de 3 bits y nos dice cual es mayor o cual es menor

I. INTRODUCCIÓN

Este documento se realiza con la intención de aplicar el conocimiento adquirido en la materia de circuitos digitales como también consultar nuevos conocimientos, los cuales aplicaremos en el diseño de 3 circuitos. Se empezará el diseño realizando una tabla de verdad para los problemas a resolver, una vez echo eso se procederá a sacar sus funciones y las implementaremos en un simulador con el que comprobaremos su funcionamiento, respaldado por una nueva simulación en un laboratorio virtual, además que se escogerá el primer problema en el que se implementará una aplicación en app inventor.

II. OBJETIVOS

Objetivo general

- Diseñar un circuito votador, un circuito que en su salida nos indique el número de turno ingresando la hora en código binario y un circuito digital comparador de 2 números de 2 bits.

Objetivos específicos

- Implementar los 3 circuitos en un simulador y un laboratorio virtual.

- Diseñar una aplicación en app Inventor del primer problema que tenga las mismas características.
- Implementar a su salida un display de 7 segmentos a los dos últimos problemas planteados

III. ESTADO DE ARTE


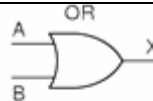

Según (Santos, 2018) en su publicación “Una interpretación lógico-matemática para una dualidad kaingang” estudio la dualidad de compuertas donde nos indica mediante el álgebra de Boole puede llegar hacer operaciones para llegar a una diferente función pero que cumple una misma tabla de verdad donde aplica en un paso la ley de Morgan que nos indica $\overline{\overline{a}} = a$ dándonos la posibilidad de llevar a otras expresiones que necesitemos en este presente documento se usara estas leyes y toda el álgebra de Boole.

IV. MARCO TEÓRICO

A. Compuertas digitales

Un dígito binario se denomina un *bit*. La información está representada en las computadoras digitales en grupos de bits. Utilizando diversas técnicas de codificación los grupos de bits pueden hacerse que representen no solamente números binarios sino también otros símbolos discretos cualesquiera, tales como dígitos decimales o letras de alfabeto. Utilizando arreglos binarios y diversas técnicas de codificación, los dígitos binarios o grupos de bits pueden utilizarse para desarrollar conjuntos completos de instrucciones para realizar diversos tipos de cálculos.

A continuación, se detallarán las compuertas lógicas usadas para la resolución de este ejercicio.

<p>Compuerta AND: (74LS21)</p> <p>Cada compuerta tiene dos variables de entrada designadas por A y B y una salida binaria designada por x.</p> <p>La compuerta AND produce la multiplicación lógica AND: esto es: la salida es 1 si la entrada A y la entrada B están ambas en el binario 1: de otra manera, la salida es 0.</p> <p>Las compuertas AND pueden tener más de dos entradas y por definición, la salida es 1 si todas las entradas son 1.</p>	<p style="text-align: center;">AND</p>  <table border="1" data-bbox="628 321 747 472"><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1
A	B	X														
0	0	0														
0	1	0														
1	0	0														
1	1	1														
<p>Compuerta OR: (74LS32)</p> <p>La compuerta OR produce la función sumadora, esto es, la salida es 1 si la entrada A o la entrada B o ambas entradas son 1; de otra manera, la salida es 0.</p> <p>El símbolo algebraico de la función OR (+), es igual a la operación de aritmética de suma.</p> <p>Las compuertas OR pueden tener más de dos entradas y por definición la salida es 1 si cualquier entrada es 1.</p>	<p style="text-align: center;">OR</p>  <table border="1" data-bbox="638 718 756 875"><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1
A	B	X														
0	0	0														
0	1	1														
1	0	1														
1	1	1														
<p>INVERSOR: (7404)</p> <p>El circuito NOT es un inversor que invierte el nivel lógico de una señal binaria. Produce el NOT, o función complementaria. El símbolo algebraico utilizado para el complemento es una barra sobre el símbolo de la variable binaria.</p> <p>Si la variable binaria posee un valor 0, la compuerta NOT cambia su estado al valor 1 y viceversa.</p> <p>El círculo pequeño en la salida de un símbolo gráfico de un inversor designa un inversor lógico. Es decir, cambia los valores binarios 1 a 0 y viceversa.</p>	<p style="text-align: center;">NOT</p>  <table border="1" data-bbox="643 1119 734 1224"><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0									
A	X															
0	1															
1	0															

B. Lógica positiva y negativa

• Lógica Positiva

Lógica positiva en esta nomenclatura al 1 le corresponde el nivel más alto de tensión (positivo) y el 0 lógico es el nivel más bajo (negativo), en donde ocurre cuando la señal no está bien definida será de conocer los límites para cada tipo de señal en donde es la tensión.

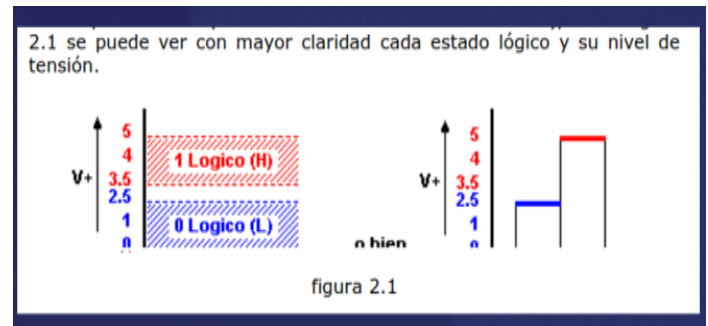


Imagen1: Lógica positiva

• Lógica Negativa

Es donde ocurre lo contrario en donde se presenta el estado 1 con los niveles más bajos de tensión y al 0 con los niveles más altos. En donde suele trabajar con lógica positiva en donde el modulo es la forma más sencilla de representar estos.

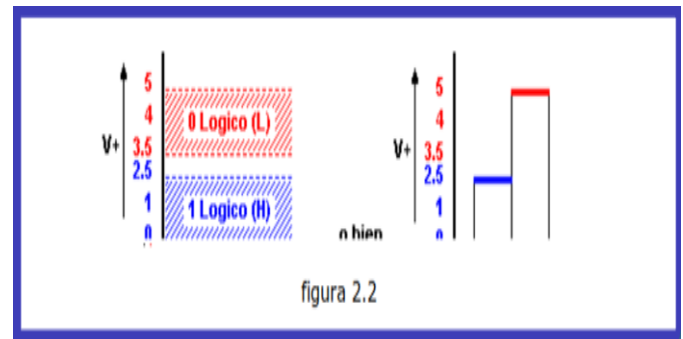


Imagen2: Lógica negativa

C. Funciones lógicas

Función lógica

Dadas n variables lógicas (X_1, X_2, \dots, X_n) cuyos valores pueden tomar 0 o 1, es posible definir una función lógica $f(X_1, X_2, \dots, X_n)$ que tomará un valor 0 o 1 según los valores que tomen cada una de las variables y las operaciones que se realicen.

Símbolos

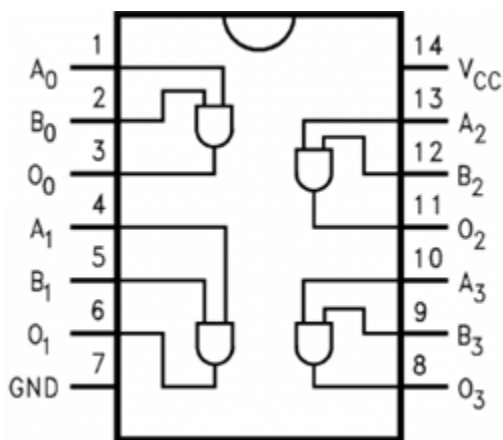
simples se pueden encontrar en encapsulados DIP, SOIC y SSOP.

Simbología de Funciones Lógicas	Americana (MIL/ANSI)	Británica (BS3939)	Alemana	Comisión Internacional de Electrotecnia (IEC)
Buffer No Inversor				
Buffer Inversor (Puerta NOT)				
Puerta AND 2 entradas				
Puerta NAND 2 entradas				
Puerta OR 2 entradas				
Puerta NOR 2 entradas				
Puerta XOR 2 entradas				
Puerta XNOR 2 entradas				

Imagen5: Símbolos

Numeración de circuitos integrados

Las compuertas lógicas, los componentes básicos de muchos otros circuitos integrados, pueden ser encapsuladas en su propio circuito integrado. Algunos de los circuitos integrados de compuertas lógicas pueden contener varias compuertas en un solo encapsulado, como esta compuerta AND de cuatro entradas.



Las compuertas lógicas pueden estar conectadas dentro de un circuito integrado para crear temporizadores, contadores, latches, registro de desplazamiento, y otros circuitos lógicos básicos. La mayoría de estos circuitos

VI. MAPA DE VARIABLES		
Circuito Votador	Circuito de Turno	Comparador
A: Primer votador dice que sí.	A b c d	El número A:
B: Segundo votador dice que sí.	Turno 1 (de 0 al 2 en decimal)	El número B:
C: Tercer votador dice que sí.	Turno 2 (de 3 al 4 en decimal)	M = 1 si A > B
F ₁ : Los votadores resuelven que si se escoge ese circuito.	Turno 3 (de 8 al 10 en decimal)	1 = 1 si A = B
F ₂ : Los votantes resuelven que no se escoge ese circuito.		m = 1 si A < B

Tabla 1 Variables de entrada y salida

VII. EXPLICACIÓN DEL CÓDIGO FUENTE

1. Las normas de seguridad de los modernos aviones exigen que, para señales de vital importancia para la seguridad del aparato, los circuitos deben estar triplicados para que el fallo de uno de ellos no produzca una

catástrofe. En caso de que los tres circuitos no produzcan la misma salida, ésta se escogerá mediante votación. Diseñe el circuito "votador" que ha de utilizarse para obtener como resultado el valor mayoritario de las tres entradas.

Circuito Votador

Diseño

Como tenemos 3 entradas para los 3 votos que necesitamos. Entonces serán 2^3 combinaciones para nuestra tabla de verdad, donde para nuestro caso $n=3$.

$$2^n = 2^3 = 8$$

Declaración de variables

A: Primer votador dice que sí.

B: Segundo votador dice que sí.

C: Tercer votador dice que sí.

F_1 : Los votadores resuelven que si se escoge ese circuito.

F_2 : Los votantes resuelven que no se escoge ese circuito.

A	B	C	F_1	F_2
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

Tabla 2 Tabla de verdad circuito de votación

Ahora sacamos las funciones de salida utilizando los min términos

Para F_1

$$F_1 = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

$F_1 = BC(\bar{A} + A) + A(\bar{B}C + B\bar{C})$ (Agrupamos los términos, y aplicamos $A + \bar{A} = 1$)

$F_1 = BC + A(\bar{B}C + B\bar{C})$ (Aplicamos una regla de boole

$$A + \bar{A}B = A + B)$$

$$F_1 = A(B + C) + BC$$

Para F_2

$F_2 = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C}$ (Ahora reducimos la función)

$F_2 = \bar{A}\bar{B}(\bar{C} + C) + \bar{C}(\bar{A}B + A\bar{B})$ (Agrupamos los términos, y aplicamos $A + \bar{A} = 1$)

$F_2 = \bar{A}\bar{B} + \bar{C}(\bar{A}B + A\bar{B})$ (Aplicamos una regla de boole $A + \bar{A}B = A + B$)

$$F_2 = \bar{A}\bar{B} + \bar{C}(\bar{A} + \bar{B})$$

Una vez encontrado las funciones reducidas, las implementamos en el simulador multisim.

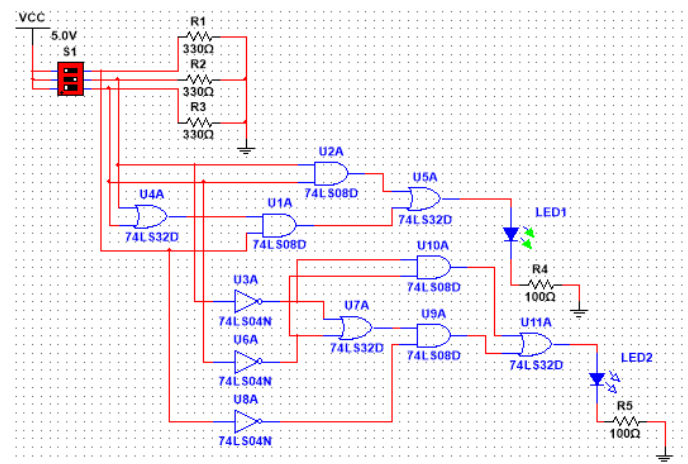


Figura 1. Circuito votador simulado en multisim

Ahora vamos a implementarlo en un laboratorio virtual en línea conocido como Tinkercad.

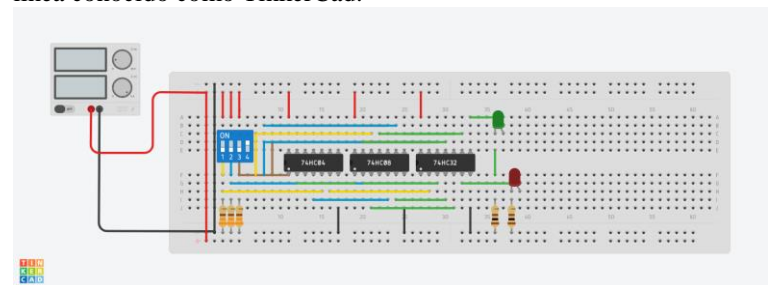


Figura 2. Circuito votador simulado en lab virtual tinkercad

2. El horario laboral de una factoría es de 8 horas diarias, divididas en tres turnos: de 8 a 11 (primer turno), de 11 a 13 (segundo turno), de 13 a 16 (descanso) y de 16 a 19 (tercer turno). Se pretende diseñar un circuito que tenga como entradas las representaciones binarias de la hora actual menos

ocho y que proporcione a la salida el número de turno que está trabajando si procede (si procede) o “0” si es hora de descanso

Variables de entrada:

A B C D

Variables de salida:

Turno 1 (de 0 al 2 en decimal)

Turno 2 (de 3 al 4 en decimal)

Turno 3 (de 8 al 10 en decimal)

Tabla de verdad

TURNOS	HORA	HORA MENOS	8	A	B	C	D	F
turno 1	8	0		0	0	0	0	1
turno 1	9	1		0	0	0	1	1
turno 1	10	2		0	0	1	0	1
turno 2	11	3		0	0	1	1	1
turno 2	12	4		0	1	0	0	1
descanso	13	5		0	1	0	1	0
descanso	14	6		0	1	1	0	0
descanso	15	7		0	1	1	1	0
turno 3	16	8		1	0	0	0	1
turno 3	17	9		1	0	0	1	1
turno 3	18	10		1	0	1	0	1

Tabla 2 Tabla de verdad circuito de turno

Función simplificada por turnos

TURNOS		
TURNOS 1		
$A'B'C'D'$	$A'B'C'$	
$+$		
$A'B'C'D$	$A'B'C'$	
$+$		
$A'B'CD'$	$A'B'CD'$	$A'B'(C'+CD')$ = $A'B'(C'+D')$
TURNOS 2		
$A'B'CD$	$A'B'CD+A'BC'D'$	$A'(B'CD'+BC'D')$
$+$		
$A'BC'D'$		
TURNOS 3		
$AB'C'D'$	$AB'C'$	
$+$		
$AB'C'D$	$AB'C'$	
$+$		
$AB'CD'$	$AB'CD'$	$AB'(C'+CD')$ = $AB'(C'+D')$

Tabla 3 Simplificación de funciones

Función implementada en proteos

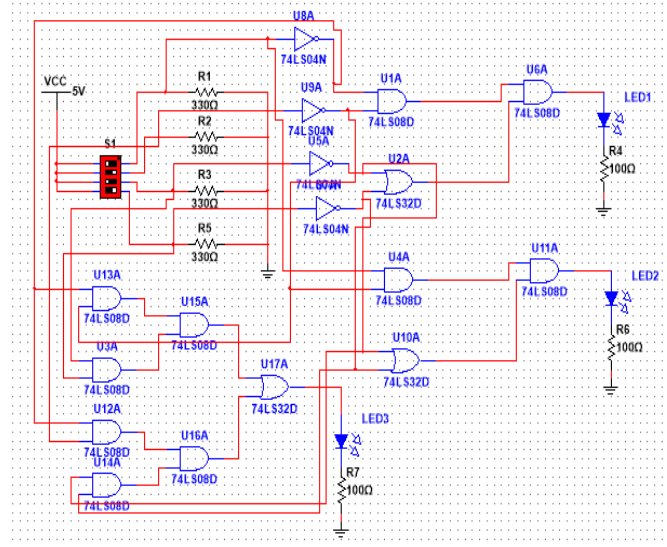


Figura 3. Circuito de turno simulado en multisim

3. Se pretende diseñar un circuito comparador de 2 números de 2 bits, $A=(a_1, a_0)$ y $B=(b_1, b_0)$. Dicho circuito deberá tener tres salidas M, I, m, de tal forma que:

$M = 1$ si $A > B$

$I = 1$ si $A = B$

$m = 1$ si $A < B$

Diséñese exclusivamente con puertas NOR.

Variables de entrada:

El número A:

$a_1 a_0$

El número B:

$b_1 b_0$

Variables de salida:

$M = 1$ si $A > B$

$I = 1$ si $A = B$

$m = 1$ si $A < B$

a_1	a_0	b_1	b_0	M	l	m
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

Simplificación mediante mapas K

A > B

$a_1 a_0$ $b_1 b_0$	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	0

Función simplificada

$$A > B = (a_1 + a_0)(a_1 + \bar{b}_0)(a_1 + \bar{b}_1)(a_0 + \bar{b}_1)(\bar{b}_1 + \bar{b}_0)$$

Función para implementar solo con NOR

$$A > B = \overline{\overline{(a_1 + a_0)(a_1 + \bar{b}_0)(a_1 + \bar{b}_1)(a_0 + \bar{b}_1)(\bar{b}_1 + \bar{b}_0)}} = \overline{(a_1 + a_0) + (a_1 + \bar{b}_0) + (a_1 + \bar{b}_1) + (a_0 + \bar{b}_1) + (\bar{b}_1 + \bar{b}_0)}$$

A = B

$a_1 a_0$ $b_1 b_0$	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

Función simplificada

$$A = B = (a_0 + \bar{b}_0)(a_1 + \bar{b}_1)(\bar{a}_0 + b_0)(\bar{a}_1 + b_1)$$

Función para implementar solo con NOR

$$A = B = \overline{\overline{(a_0 + \bar{b}_0)(a_1 + \bar{b}_1)(\bar{a}_0 + b_0)(\bar{a}_1 + b_1)}} = \overline{(a_0 + \bar{b}_0) + (a_1 + \bar{b}_1) + (\bar{a}_0 + b_0) + (\bar{a}_1 + b_1)}$$

A < B

$a_1 a_0$ $b_1 b_0$	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

Función simplificada

$$A < B = (b_1 + b_0)(\bar{a}_0 + b_1)(\bar{a}_1 + b_1)(\bar{a}_1 + b_0)(\bar{a}_1 + \bar{a}_0)$$

Función para implementar solo con NOR

$$A < B = \overline{\overline{(b_1 + b_0)(\bar{a}_0 + b_1)(\bar{a}_1 + b_1)(\bar{a}_1 + b_0)(\bar{a}_1 + \bar{a}_0)}} = \overline{(b_1 + b_0) + (\bar{a}_0 + b_1) + (\bar{a}_1 + b_1) + (\bar{a}_1 + b_0) + (\bar{a}_1 + \bar{a}_0)}$$

Para el tercer enunciado diseñe un módulo adicional que permita visualizar en un display de 7 segmentos conectado a cada salida el número 0 si han ocurrido uno de los 3 casos.

M	l	m	O
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

$a_1 a_0$ $b_1 b_0$	00	01	11	10
0	0	1	0	1
1	1	0	0	0

Función simplificada

$$O = (M + L + m)(\bar{M} + \bar{m})(\bar{M} + \bar{L})(\bar{L} + \bar{m})$$

Función para implementar solo con NOR

$$A < B = \overline{\overline{(M + L + m)(\bar{M} + \bar{m})(\bar{M} + \bar{L})(\bar{L} + \bar{m})}} = \overline{(M + L + m) + (\bar{M} + \bar{m}) + (\bar{M} + \bar{L}) + (\bar{L} + \bar{m})}$$

Simulación en multisim

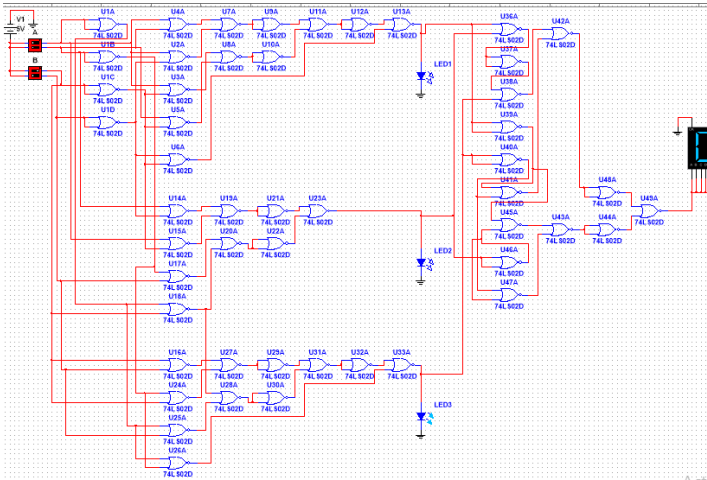


Figura 4. Circuito comparador simulado en multisim

Simulación en laboratorio virtual

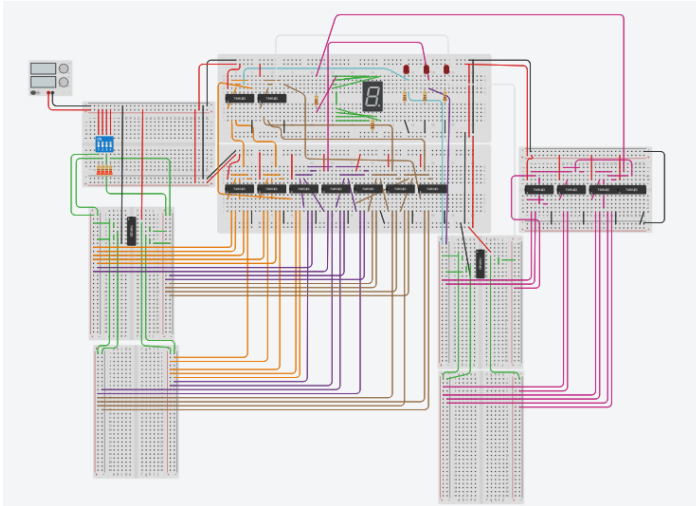


Figura 5. Circuito comparador simulado en laboratorio virtual

Para el diseño de la app:

Vista Diseñador Screen1



En este bloque de código armamos un splash screen, aplicando el sensor de reloj.

inicializar global cont como tomar el valor inicial

cuando Voto_N°1 - Clic
ejecutar abrir otra pantalla Nombre de la pantalla "Voto_1"

cuando Resultado - Clic
ejecutar si tomar global cont <= 1
entonces poner Result - Texto como "La mayoría de votos nos dan un resultado de "No"
sino poner Result - Texto como "La mayoría de votos nos dan un resultado de "Si"

cuando Borrar - Clic
ejecutar poner Result - Texto como "0"
poner global cont a 0

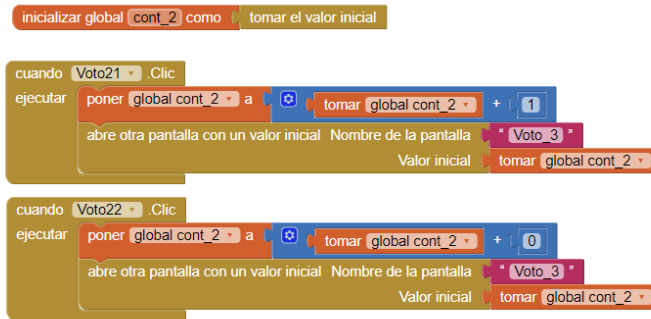
Ahora en este bloque se asignamos una función a cada botón de nuestro screen principal, cuando presionamos el botón comenzar nos lanza a una nueva screen en donde comenzará el primer voto, para el botón resultado verificamos el valor si es menor que 1 para contabilizar los votos y dar una sentencia de si la mayoría votó que no, caso contrario quiere decir que la mayoría de los votos fueron si, y visualizamos en un label.

inicializar global cont_1 como 0

cuando Voto_11 - Clic
ejecutar poner global cont_1 a 1
abrir otra pantalla con un valor inicial Nombre de la pantalla "Voto_2"
Valor inicial tomar global cont_1

cuando Voto_12 - Clic
ejecutar si tomar global cont_1 <= 1
entonces abrir otra pantalla Nombre de la pantalla "Voto_2"
sino poner global cont_1 a 0
abrir otra pantalla con un valor inicial Nombre de la pantalla "Voto_2"
Valor inicial tomar global cont_1

En esta parte del código nos encontramos en la screen Voto_1 en donde inicializamos la variable cont_1 en 0 para empezar con los votos y cuando damos clic en sí. Le asignamos un valor de 1 y si es el botón No, le asignamos un valor de cero, en cualquiera de los dos casos nos lanza a la pantalla siguiente en donde mandamos el valor de cont_1.



En esta parte del código nos encontramos en la screen Voto_2 en donde inicializamos la variable cont_2 en con el valor que mandamos desde la screen voto 1, cuando damos clic en sí, le asignamos la suma del valor que tenga nuestra variable cont_2 +1 y si es el botón No le sumamos 0, en cualquiera de los dos casos nos lanza a la pantalla siguiente en donde mandamos el valor de cont_2.



En esta parte del código nos encontramos en la screen Voto_3 en donde inicializamos la variable cont_3 en con el valor que mandamos desde la screen voto 2, cuando damos clic en sí, le asignamos la suma del valor que tenga nuestra variable cont_2 +1 y si es el botón No le sumamos 0, en cualquiera de los dos casos nos lanza a la pantalla principal en donde podemos ya mostrar ya los resultados con el valor que mandamos de cont_3.



VIII. DESCRIPCIÓN DE PRERREQUISITOS Y CONFIGURACIÓN
Para el circuito que en su salida nos muestra el horario de turno de un trabajador, hay que tomar en cuenta que se debe ingresar la hora en código binario restándole 8 horas.

En cuanto a la aplicación desarrollada en app inventor se debe tomar en cuenta que los 3 votantes deben solo realizaran el ejercicio una sola vez y solo podrá escoger la opción de “Sí” y “No”.

IX. APORTACIONES

Uso de binarios de tres bits para la transformación a cualquier base en este caso a octal y decimal como se indica en la siguiente imagen mencionada en Circuitos digitales []

OTROS MÉTODOS DE CONVERSIÓN

- Los sistemas octal y hexadecimal están relacionados con el binario, ya que sus bases son potencias exactas de 2 (la base binaria). Esto permite convertir entre estos sistemas de forma muy sencilla:
 - OCTAL a BINARIO:** Convertir cada dígito en 3 bits.
Ejemplo: $735_8 = 111\ 011\ 101_2$
 - BINARIO a OCTAL:** Agrupar en grupos de 3 bits y convertirlos de forma independiente a octal.
Ejemplo: $1\ 011\ 100\ 011_2 = 1343_8$

Fig. 8. Método de conversión de 3 bits

Uso de laboratorio virtual Tinkercad, este simulador es muy eficiente al momento de armar nuestros circuitos acercándose mucho a lo que es la realidad, además que posee diversos componentes para el diseño de circuitos digitales.

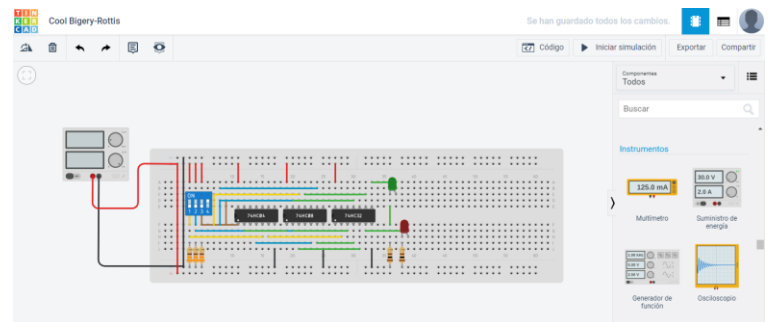


Figura 6. Circuito comparador simulado en laboratorio virtual

X. CONCLUSIONES

Se diseñó el circuito votador el cual nos permite tomar una decisión para escoger una salida tomando en cuenta la mayoría de los votos, dado una falla de un sistema de seguridad de un avión

Se implementó el ejercicio votador en app inventor

El uso de display de 7 segmentos se puede usar con un decodificador.

XI. RECOMENDACIONES

Se recomienda que para el uso de la App Inventor se debe tener conocimientos previos de lenguaje de programación como Java, C++, PYTHON.

Es importante conocer el álgebra de Boole para la simplificación de funciones

Es necesario investigar mapas K para la mejor simplificación y optimización de tiempo en simplificar funciones

Es preciso planificar un cronograma con diagramas de Grant en las diferentes aplicaciones que existen y para el desarrollo se recomienda el software Project.

XII. CRONOGRAMA

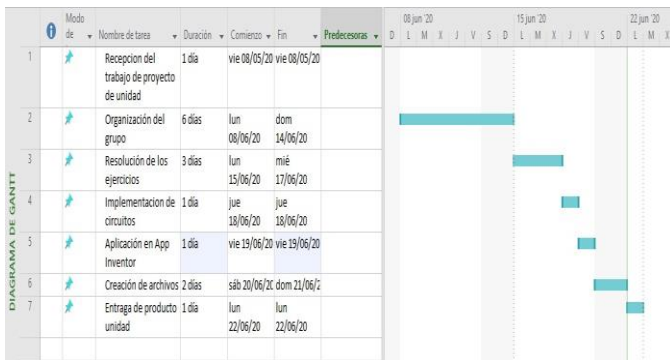


Figura 7. Cronograma de actividades

XIII. ANEXOS



Figura 8. Interfaz de la aplicación

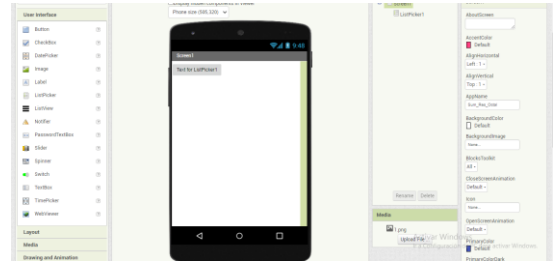


Figura 9. Entorno de trabajo parte grafica



Figura 10. Entorno de trabajo parte de bloques

XIV. MANUAL DE USUARIO

Circuito Votación

La siguiente aplicación es una para contador de votos entre “sí” y “no”.

En la interfaz principal se encontrará con tres botones; uno que es “comenzar” que nos permite comenzar con el proceso de sufragar; el voto será discreto para tres votantes y se debe sufragar en orden y solo una vez por votante entre las opciones de “sí” o “no”.

El botón “Ver resultado” nos permite visualizar quien obtuvo más votos entre las opciones de “sí” o “no”, dándonos como mensaje “La mayoría de votos nos dan un resultado de Si” si es si el ganador y si el vencedor es no “La mayoría de votos nos dan un resultado de No”

Y para borrar los datos almacenados en ver resultados lo hacemos con el botón borrar resultado así nuestra pantalla principal quedara vacía y además de que nos permite volver a realizar el sufragio una vez más realizando el mismo proceso desde el botón comenzar.

XV. HOJA TÉCNICA

Ficha Técnica de Calculadora de Circ Vot**Lenguaje de Programación**

Lenguaje de programación	Programación orientada a objetos y base de datos
Software de edición y programación	App inventor II
Formato de empaquetamiento	.APK
Software	Móvil Android

Tab. 1. Especificaciones

La aplicación móvil de circuito votador, desarrollada elementos de código (lenguaje) programación orientada a objetos.

Utilizando App Inventor II como plataforma de edición y desarrollo, el cual optimiza las funciones de la app con los sistemas operativos Android al utilizar en algunas funciones Java así como POO permite ofrecer algunas funcionalidades en navegadores de cualquier sistema operativo para dispositivos móviles de escritorio a través de un código QR.

REFERENCIAS

- [1] Floyd, T. (2006). *Fundamentos de sistemas digitales*. Madrid: Pearson.
- [2] Ricoy, A. (14 de Junio de 2020). *Appinventor en español*. Obtenido de <https://sites.google.com/site/appinventormegusta/primeros-pasos>
- [3] Santos, J. (2018). Algoritmo de las operaciones aritmeticas aplicadas a los codigos binarios, octal y hexadecimal con sus respectivas conversiones. *Revista Latinoamericana de Etnomatemática*.