**Desarrollo**

En nuestro proyecto, contamos con un archivo index.html, este archivo es el primero que se ejecuta para poder iniciar nuestro proyecto.

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>Electronica</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

  <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500&display=swap" rel="stylesheet">

  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">

  <script src="https://kit.fontawesome.com/2682b753b2.js" crossorigin="anonymous"></script>

</head>

<body class="mat-typography">

  <app-root></app-root>

</body>

</html>

**Fig. 1.** Código del index.html que inicia todo el programa.

Como página de inicio, contamos con unas cartas elaboradas con estilos Bootstrap, en los cuales cada uno de ellos cuenta con un tipo de circuito diferente, como polarización fija, polarización de emisor, divisor de voltaje, realimentación del colector y polarización de base común.

<div class="card border-info mb-3" style="max-width: 18rem;">

    <div class="card-body">

      <h3 class="card-header" style="color: #561d25;">Polarizacion Fija</h3>

      <div class="card-body text-info">

        <h5 class="card-title" style="color: #94a353;"> <font size="+1.5">Valores necesitados</font></h5>

        <div class="card-text" style="color: #94a353;">

          <table>

            <td>

                Vcc Rb Rc Rc y Beta

            </td>

            <td>

              <img class="card-img" src="../assets/final\_fija.jpg" alt="Card image cap">

            </td>

          </table>

        </div>

      </div>

    </div>

    <button (click)="fija()" mat-button style="margin-top: 50px;">Comenzar</button>

  </div>

**Fig. 2.** Ejemplo de una de las cartas completas para cada una de las opciones del menú principal.

Si la opción elegida es de la polarización por divisor, emisor, fija o de realimentación aparecerá un pequeño menú conformado por dos botones con estilos CSS y Bootstrap para poder elegir la configuración de dicho circuito.

<link href="https://fonts.googleapis.com/css2?family=Cookie&display=swap" rel="stylesheet">

<br><br>

<div class="m-5"><h3>Divisor de voltaje</h3></div>

<div (click)="exacto()" class="m-5" id="btn"><span class="noselect">Exacto</span><div id="circle"></div></div>

<div (click)="aprox()" class="m-5" id="btn"><span class="noselect">Aproximado</span><div id="circle"></div></div>

**Fig. 3.** Ejemplo de un submenú al elegir un tipo de circuito de transistor.

Para poder solicitar los datos necesarios en el sitio web, fue necesario implementar una interfaz amigable hacia los usuarios, haciendo esto, ingresamos una imagen del circuito a calcular del lado derecho de la pantalla y del lado izquierdo se muestran los campos a ingresar para poder calcular los demás datos.

<h1 class="mt-5">Polarización por división de voltaje Aproximado</h1>

  <h2>Transistores BJT</h2>

<table class="mb-5" style="margin: auto;">

  <thead>

    <tr>

      <th><h2 class="datosIngresa">Ingresa los siguentes datos</h2></th>

    </tr>

  </thead>

  <tr>

    <td> <span  matBadgeOverlap="false">Vcc= </span></td>

      <td>

        <p>

          <mat-form-field appearance="legacy">

            <mat-label for="VCC">Voltaje de entrada</mat-label>

            <input [(ngModel)]="VCC" type="number" id="VCC" matInput placeholder="valor sin unidad">

          </mat-form-field>

        </p>

      </td>

      <td rowspan="4">

        <img src="../../assets/final\_div\_voltaje.jpg" alt="">

      </td>

    </tr>

    <tr>

      <td> <span  matBadgeOverlap="false">RC=</span></td>

      <td>

        <p>

          <mat-form-field appearance="legacy">

            <mat-label for="RC">Res. de Colector</mat-label>

            <input [(ngModel)]="RC" type="number" id="RC" matInput placeholder="valor sin unidad">

          </mat-form-field>

        </p>

      </td>

    </tr>

    <tr>

      <td> <span  matBadgeOverlap="false">RE=</span></td>

      <td>

        <p>

          <mat-form-field appearance="legacy">

            <mat-label for="RE">Res.emisor</mat-label>

            <input [(ngModel)]="RE" type="number" id="RE" matInput placeholder="valor sin unidad">

          </mat-form-field>

        </p>

      </td>

    </tr>

    <tr>

      <td> <span  matBadgeOverlap="false">Beta=</span></td>

      <td>

        <p>

          <mat-form-field appearance="legacy">

            <mat-label for="Beta">Ganancia</mat-label>

            <input [(ngModel)]="Beta" type="number" id="Beta" matInput placeholder="valor sin unidad">

          </mat-form-field>

        </p>

      </td>

    </tr>

    <tr>

      <td> <span  matBadgeOverlap="false">R1=</span></td>

      <td>

        <p>

          <mat-form-field appearance="legacy">

            <mat-label for="R1">Res 1</mat-label>

            <input [(ngModel)]="R1" type="number" id="R1" matInput placeholder="valor sin unidad">

          </mat-form-field>

        </p>

      </td>

    </tr>

    <tr>

      <td> <span  matBadgeOverlap="false">R2=</span></td>

      <td>

        <p>

          <mat-form-field appearance="legacy">

            <mat-label for="R2">Res 2</mat-label>

            <input [(ngModel)]="R2" type="number" id="R2" matInput placeholder="valor sin unidad">

          </mat-form-field>

        </p>

    </td>

  </tr>

  <tr>

    <td colspan="3">

      <div \*ngIf="!band">

        <div class="alert alert-danger m-auto" style="text-align: center;">

          <strong>¡Oh no!</strong> Los datos ingresados no cumplen con la condicion: <br> <b>β\*Re >= 10\*R2</b>

        </div>

      </div>

    </td>

  </tr>

  <button style="margin-left:auto;margin-right:auto; position:absolute;" (click)="limpiar()" mat-raised-button color="red">limpiar</button>

</table>

<button style="margin-left:auto;margin-right:auto; position:absolute;" (click)="regresar()" mat-raised-button color="primary">Regresar</button>

**Fig. 4.** Ejemplo de cómo se encuentran implementados la página donde se ingresan los datos del circuito.

Para la polarización por división de voltaje aproximado, se tiene que cumplir la condición donde:

Para poder cumplir con la validación, realizamos in \*ngIf ya que esto implementa un “if” en donde veamos en donde si no se cumple la condición (con la variable “band”) entra a la alerta en donde le indica que no se cumple con dicha condición y es hora de modificar algunos datos.

<div \*ngIf="!band">

        <div class="alert alert-danger m-auto" style="text-align: center;">

          <strong>¡Oh no!</strong> Los datos ingresados no cumplen con la condicion: <br> <b>β\*Re >= 10\*R2</b>

        </div>

      </div>

**Fig. 5.** Etiqueta <div> en donde realiza la comprobación de la condición de polarización por división de voltaje aproximado.

Para poder calcular los valores necesarios de cada circuito, se implementa una serie de líneas de código en su “component.ts” que permite realizar código JavaScript. En este archivo se declaran todas las variables necesarias de tipo “number” (número) para poder calcular los valores de cada variable necesaria del circuito.

import { Component, OnInit } from '@angular/core';

import {ActivatedRoute} from '@angular/router';

import {Router} from '@angular/router';

@Component({

  selector: 'app-divisor-aprox',

  templateUrl: './divisor-aprox.component.html',

  styleUrls: ['./divisor-aprox.component.css']

})

export *class* DivisorAproxComponent implements OnInit {

  R1:*number*;

  R2:*number*;

  ETH:*number*;

  RTH:*number*;

  RB:*number*;

  RE:*number*;

  RC:*number*;

  IC:*number*;

  IB:*number*;

  IE:*number*;

  VCC:*number*;

  VCE:*number*;

  VE:*number*;

  ICsat:*number*;

  Beta:*number*;

  VRC:*number*;

  VRB:*number*;

  VB:*number*;

  VC:*number*;

  VBE=0.7;

  VRE:*number*;

  band=true;

*constructor*(private *router*:Router) { }

    limpiar(){

      this.VCC=null;

      this.R1=null;

      this.RC=null;

      this.RE=null;

      this.R2=null;

      this.Beta=null;

    }

    regresar(){

      this.router.navigate([ '/Home']);

     }

    ngOnInit(): *void* {

      setInterval(()*=>*{

        if((this.Beta!=null)&&(this.RE!=null)&&(this.R2!=null)){

          if((this.Beta\*this.RE)>=(10\*this.R2)){

            this.band=true;

            this.VB=(this.R2\*this.VCC)/(this.R1+this.R2);

            this.VE=this.VB-this.VBE;

            this.IE=this.VE/this.RE;

            this.IB=this.IE/(this.Beta+1);

            this.IC=this.IE;//son muy cernaos asi que esto funciona

            this.VCE=this.VCC-this.IC\*(this.RC+this.RE);

            this.RB=this.VB/this.IB;

            this.VRB=this.IB\*this.RB;

            this.VRC=this.IC\*this.RC;

            this.VRE=this.IE\*this.RE;

            this.VC=this.VCC-this.VRC;

            this.ICsat=this.VCC/(this.RC+this.RE);

          }

          else{

            this.band=false;

          }

        }

      },200);

    }

}

**Fig. 6.** Ejemplo de cómo se realizan los cálculos para el circuito polarización por división de voltaje aproximado.

Gracias a nuestro [(ngModel)] podemos modificar los valores en cualquier momento con una actualización inmediata de los valores que éste afecte. Los valores obtenidos se irán registrando en varias tablas según su categoría (voltaje, corriente, etc.) con la sintaxis {{ NOMBRE DE LA VARIABLE }}.

<div \*ngIf="band">

  <table class="container mb-5" style="margin-left:auto;margin-right:auto">

    <thead>

      <tr>

        <th><h1>Voltaje</h1></th>

        <th><h1>Volts</h1></th>

      </tr>

    </thead>

    <tbody>

      <tr>

        <td>VCC</td>

        <td>{{VCC}}</td>

      </tr>

      <tr>

        <td>VCE</td>

        <td>{{VCE}}</td>

      </tr>

      <tr>

        <td>VRC</td>

        <td>{{VRC}}</td>

      </tr>

      <tr>

        <td>VRB</td>

        <td>{{VRB}}</td>

      </tr>

      <tr>

        <td>VB</td>

        <td>{{VB}}</td>

      </tr>

      <tr>

        <td>VC</td>

        <td>{{VC}}</td>

      </tr>

      <tr>

        <td>VE</td>

        <td>{{VE}}</td>

      </tr>

    </tbody>

  </table>

  <!--Corriente-->

  <table class="container mb-5" style="margin-left:auto;margin-right:auto">

    <thead>

      <tr>

        <th><h1>Corriente</h1></th>

        <th><h1>A</h1></th>

        <th><h1>mA</h1></th>

        <th><h1>µA</h1></th>

      </tr>

    </thead>

    <tbody>

      <tr>

        <td>IB</td>

        <td>{{IB}}</td>

        <td>{{IB \* 1000}}</td>

        <td>{{IB \* 1000000}}</td>

      </tr>

      <tr>

        <td>IC</td>

        <td>{{IC}}</td>

        <td>{{IC \* 1000}}</td>

        <td>{{IC \* 1000000}}</td>

      </tr>

      <tr>

        <td>ICsat</td>

        <td>{{ICsat}}</td>

        <td>{{ICsat \* 1000}}</td>

        <td>{{ICsat \* 1000000}}</td>

      </tr>

      <tr>

        <td>IE</td>

        <td>{{IE}}</td>

        <td>{{IE \* 1000}}</td>

        <td>{{IE \* 1000000}}</td>

      </tr>

    </tbody>

  </table>

  <!--Resistencias-->

  <table class="container mb-5" style="margin-left:auto;margin-right:auto">

    <thead>

      <tr>

        <th><h1>Resistencia</h1></th>

        <th><h1>Ohms</h1></th>

        <th><h1>Kilo Ohms</h1></th>

      </tr>

    </thead>

    <tbody>

      <tr>

        <td>RC</td>

        <td>{{RC}}</td>

        <td>{{RC / 1000}}</td>

      </tr>

      <tr>

        <td>RE</td>

        <td>{{RE}}</td>

        <td>{{RE / 1000}}</td>

      </tr>

      <tr>

        <td>RB</td>

        <td>{{RB}}</td>

        <td>{{RB / 1000}}</td>

      </tr>

      <tr>

        <td>R1</td>

        <td>{{R1}}</td>

        <td>{{R1 / 1000}}</td>

      </tr>

      <tr>

        <td>R2</td>

        <td>{{R2}}</td>

        <td>{{R2 / 1000}}</td>

      </tr>

    </tbody>

  </table>

  <!--Ganancia-->

  <table class="container mb-5" style="margin-left:auto;margin-right:auto">

    <thead>

      <tr >

        <th><h1>Ganancia</h1></th>

        <th><h1></h1></th>

      </tr>

    </thead>

    <tbody>

      <tr>

        <td>Beta</td>

        <td>{{Beta}}</td>

      </tr>

    </tbody>

  </table>

</div>

**Fig. 7.** Ejemplo de cómo son impresos los valores obtenidos en las tablas que se muestran debajo de los valores ingresados anteriormente.

Como se puede observar, es muy sencillo poder obtener actualizaciones al instante sin tener que recargar la pantalla completa. Esta es una de las razones principales por las que optamos realizar el proyecto con Angular.

Conclusión:

Jessica Guerrero Carrera: Al término del proyecto aprendí como poder implementar las fórmulas que vimos en clases para poder realizar diversos cálculos de los transistores BJT. Durante la realización de la calculadora de circuitos con transistor BJT, pude reforzar mis conocimientos sobre en qué casos se pueden utilizar algunas fórmulas, así como también entender la lógica de cómo funcionan los circuitos con sus diferentes formas. Me pareció bastante interesante el objetivo del proyecto, ya que al principio del semestre creía que se realizaría algún circuito en físico y presentarlo hacía el profesor, sin embarco por la pandemia me pareció muy interesante el implementar la programación web (en el caso de mi equipo) con la electrónica.

A pesar de que vimos la mayor parte del curso por medio de clases de en línea, puedo decir que los temas vistos en clase fueron muy bien explicados, a tal grado de que el profesor siempre buscó la manera en poder brindarnos el mejor entorno de clase posible, ya sea presencial o virtual.