UCLV
Universidad Central
"Marta Abreu" de Las Villas

FIE
Facultad de
Ingeniería Eléctrica

Departamento de Electrónica y Telecomunicaciones

"Rediseño del software "Amplifiers" para el diseño de amplificadores de pequeña señal con BJT y FET"

Autor: Rolando González Cruz

Tutor: Dr.C. Carlos Roche Beltrán

Santa Clara, junio de 2018

Hago constar que el presente trabajo de diploma fue realizado en la Universidad Central "Marta Abreu" de Las Villas como parte de la culminación de estudios de la especialidad de Ingeniería en Telecomunicaciones y Electrónica, autorizando a que el mismo sea utilizado por la Institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

_____

Firma del Autor

Los abajo firmantes certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del Tutor

Firma del Jefe de Departamento
se defiende el trabajo

_____

Firma del responsable de
Formación Científico-Técnica

PENSAMIENTO

“Lo importante no es el éxito, sino el camino.”

Andrés Iniesta

DEDICATORIA

A mi mamá, por darme la fuerza y ayudarme incondicionalmente a lo largo de este camino.

A mi novia, por ser mi quía y ayuda en estos cinco años.

AGRADECIMIENTOS

A mi mamá, por su dedicación, esfuerzo e impaciencia que me ayudaron a sobrepasar todos los obstáculos.

A mi novia Anabel, por regalarme todo su tiempo y su ayuda sin esperar nada a cambio.

A mi tutor Carlos Roche, por apoyarme en el desarrollo de este proyecto y por brindarme su tiempo para la realización del mismo.

A mi familia, especialmente a mi tío y mis primos, por brindarme su ayuda en estos cinco años.

A mis abuelos Felicia y Soto, por regalarme su amor y ayuda.

A mis amigos Juank, el Poli, Davisito, Alejandro, Sandy, Lizandra y Javier que batallaron conmigo y compartieron los mejores momentos de esta etapa.

A todos los profesores de la Facultad, por inculcarme sus conocimientos.

A mi familia extendida, apoyarnos a Anabel y a mí en todo lo que necesitáramos.

TAREA TÉCNICA

Para dar cumplimiento a los objetivos trazados en la presente tesis se tuvieron en cuenta las siguientes tareas:

1. Búsqueda de información sobre la aplicación de las TIC en la enseñanza y sobre el desarrollo de software educativos, para conformar el marco teórico conceptual de la investigación.
2. Descripción de las herramientas de simulación utilizadas en la electrónica, la aplicación de la simulación para la misma, y las herramientas de programación más utilizadas haciendo énfasis en Python.
3. Rediseño del software Amplifiers con las mejoras requeridas.
4. Simulación y montaje real de un problema que brinde resultados concretos de la investigación y facilite la comparación de resultados.

_____ _____

Firma del autor Firma del Tutor

RESUMEN

El presente trabajo de diploma se enmarca en la elaboración y desarrollo de una nueva versión del software Amplifiers, ajustándose a valores estándares de resistores y logrando mostrar la gráfica del punto de operación de los transistores. Se define la importancia, actualidad, y necesidad del empleo de las Tecnologías de la Información y la Comunicación a la hora de alcanzar los retos planteados por los diferentes sistemas educacionales. Se exponen las principales características y generalidades del diseño y rediseño de software con énfasis en los software educativos. Se describen las características y herramientas más importantes que de los lenguajes de programación y de los simuladores electrónicos, así como, su aplicación en el ámbito universitario. Finalmente se detalla el rediseño de Amplifiers para lograr los objetivos propuestos. Además, se comprueban los resultados obtenidos por la nueva versión de la aplicación, de forma simulada y de forma real y se describe su forma de utilización tanto en el ámbito educacional como en el técnico.

TABLA DE CONTENIDOS

PENSAMIENTO i

DEDICATORIA ii

AGRADECIMIENTOS iii

TAREA TÉCNICA iv

RESUMEN v

TABLA DE CONTENIDOS vi

INTRODUCCIÓN 1

CAPÍTULO 1: Las TIC y los software educativos en la Electrónica. 6

1.1. Las TIC en el proceso de enseñanza-aprendizaje. 6

1.1.1 Ventajas de las TIC. 7

1.1.2 El empleo de las TIC en la Universidad. 10

1.2. Desarrollo de Software educativos. 12

1.2.1 Clasificación del software. 13

1.2.2 Ingeniería del software. 14

1.2.3 Proceso de creación del software. 15

1.2.4 Software educativo. 16

1.3. Diseño de software. 17

1.3.1 Principios para el diseño de software y consideraciones de diseño. 18

1.3.2 Rediseño de software. 20

1.3.3 Software de apoyo al aprendizaje de la Electrónica. 21

1.4. Conclusiones del Capítulo. 24

CAPÍTULO 2: Programación y simulación en la carrera Ingeniería en Telecomunicaciones y Electrónica. 25

2.1 Aplicación de la Programación en la Carrera Ingeniería en Telecomunicaciones y Electrónica. 25

2.1.1 Programación en las disciplinas de la carrera. 27

2.2 La Simulación de circuitos en la Electrónica. 28

2.2.1 Herramientas de simulación. 29

2.2.2 Herramientas de simulación más utilizadas en la Facultad de Ingeniería Eléctrica 31

2.2.3 Aplicación de la simulación. 35

2.2.4 El caso de Amplifiers. 38

2.3 Herramientas de programación. 41

2.3.1 Tipos de lenguajes de programación. 41

2.3.2 Lenguajes de programación más utilizados. 43

2.3.3. Lenguaje de programación “Python”. 46

2.4 Conclusiones del Capítulo. 48

CAPÍTULO 3. Versión mejorada del software Amplifiers. 49

3.1 Rediseño del software Amplifiers. 49

3.2 Como utilizar el software Amplifiers. 51

3.3 Problema resuelto utilizando Amplifiers. 54

3.4 Conclusiones del Capítulo. 61

CONCLUSIONES Y RECOMENDACIONES 62

Conclusiones. 62

Recomendaciones. 63

REFERENCIAS BIBLIOGRÁFICAS 64

ANEXOS 67

Anexo I: Encuesta. 67

Anexo II: Líneas de código para estandarizar los valores de resistencias. 68

Anexo III: Código utilizado para el diseño del esquema del Punto Q.....69

INTRODUCCIÓN

La Electrónica Analógica I, se circunscribe dentro de la Electrónica Aplicada, materia que a su vez pertenece al campo de las tecnologías complejas. Por ello, su aprendizaje es un proceso que requiere del esfuerzo conjunto tanto de docentes, como de estudiantes.

Tradicionalmente los docentes de esta asignatura se han caracterizado por seleccionar y/o desarrollar recursos de aprendizaje, que apliquen adecuadamente las TIC. En esta dirección se ha trabajado en la elaboración de diferentes objetos de aprendizaje (mapas conceptuales, páginas Web, manuales, programas entrenadores, etc.) que unido a la creación de un curso de la asignatura en la plataforma Moodle y el uso de las herramientas de simulación contribuye a mejorar el aprendizaje de los alumnos.

Por otra parte, se ha insistido en la necesidad de que el alumno asuma un rol más activo en el proceso de enseñanza-aprendizaje, que sea más creativo, capaz de producir y no quedarse estancado en un proceso de aprendizaje reproductivo. En este sentido el desarrollo de la habilidad de diseño es fundamental, por lo que se insiste en su formación desde las clases planificadas (laboratorios, seminarios) hasta la realización del examen final escrito.

Mediante la observación de las actividades de aprendizaje y a través del desarrollo de investigaciones se ha podido constatar que uno de los mayores inconvenientes que han presentado los alumnos es en cuanto al desarrollo de la habilidad para afrontar el diseño de amplificadores básicos utilizando transistores bipolares (BJT) y transistores de efecto de campo (FET). Si bien el proceso de análisis de amplificadores de pequeña señal demanda del esfuerzo por parte de los estudiantes, al tener que integrar varios temas de la asignatura, la máxima necesidad recae en el proceso de diseño.

Desde el punto de vista computacional, para comprobar el proceso de diseño, tradicionalmente se ha contado con los software de simulación electrónica (Orcad, Multisim, Proteus, etc.), facilitando establecer comparaciones entre los resultados teóricos y simulados; y realizar las correcciones correspondientes hasta obtener los resultados deseados. Este proceso demanda de disponer del tiempo y las habilidades necesarias y según lo observado se dificulta su cumplimiento.

En el año 2016 un grupo de estudiantes de la carrera Telecomunicaciones y Electrónica desarrollaron un software para el apoyo del diseño de amplificadores de pequeña señal con BJT y FET.

El software llamado Amplifiers tiene implementado un algoritmo para el cálculo de las diferentes etapas amplificadoras que dan solución a un problema que requiere del diseño de un amplificador con una ganancia de voltaje (Av) específica. Al mismo tiempo cuenta con las facilidades de indicar resistencia de entrada (Ri) y salida (Ro) como parte de la solución del problema.

Amplifiers brinda una interface amigable al usuario, cuenta con una barra de herramientas que incluye cinco menús: Archivo, Modo, Base de Datos, Vista y Ayuda. En la ventana principal se puede seleccionar un transistor de la base de datos con la que cuenta la aplicación, así como los parámetros correspondientes a la corriente de colector (Ic), el voltaje de la fuente de colector (Vcc), la Av, Ri y Ro.

El resultado se indica en una nueva ventana donde se pueden observar de forma gráfica las etapas amplificadoras que dan solución al problema y los valores de las resistencias que se deben utilizar [1].

Estos valores de resistores no se rigen por ningún estándar, son el resultado del cálculo real de ecuaciones que presenta el algoritmo.

Debido a lo antes expuesto es necesaria una actualización de Amplifiers que permita la utilización de resistores estándares para la solución de los problemas siguiendo un enfoque práctico y que pueda brindar otra utilidad, como puede ser la gráfica del punto de operación de los transistores.

Enfocar el aprendizaje de la Electrónica Analógica desde una perspectiva de diseño, es un proceso extremadamente complejo que requiere de un esfuerzo conjunto tanto de profesores, como de estudiantes; sin perder la finalidad en sí, es decir el desarrollo de habilidades prácticas que cada alumno debe alcanzar mediante la experimentación real. Otras experiencias muy a tono con las últimas se enmarcan en el cambio de los roles y los escenarios de formación, este es caso del Flippin Classroom (aula invertida). Dichas experiencias aplicadas en grupos de clase pequeños, pueden ser generalizadas a otros

contextos. Para el caso de la Electrónica, alcanzan su máxima expresión cuando los alumnos, ya sea de manera individual o por equipos de trabajo realizan una tarea de diseño y lo comprueban en la práctica, disponiendo de los recursos tecnológicos necesarios.

En la gestión del Plan de Estudios E, próximo a implementarse en la carrera Ingeniería en Telecomunicaciones y Electrónica, el presupuesto de horas para cada disciplina disminuye por lo que es necesario apoyarse en medios que faciliten el aprendizaje de la Electrónica y a su vez promuevan en el alumno el desarrollo de la creatividad y la formación de habilidades prácticas como es el caso de Amplifiers. Por lo antes expuesto se plantea el siguiente problema de investigación: ¿Cómo contribuir al mejoramiento del software Amplifiers de manera que facilite el diseño de amplificadores de pequeña señal para una visión práctica?

Atendiendo a esta interrogante se plantean los siguientes objetivos.

Objetivo general:

Realizar una versión mejorada del software Amplifiers que se acerque a una visión práctica utilizando resistencias de valores estándares.

Objetivos específicos:

1. Determinar tendencias asociadas al desarrollo de software para la enseñanza de la ingeniería.
2. Identificar las demandas de la programación en la carrera Ingeniería en Telecomunicaciones y Electrónica.
3. Presentar herramientas y software que faciliten el diseño de recursos para la Electrónica.
4. Confeccionar una versión mejorada del software Amplifiers.

Interrogantes Científicas:

1. ¿Cuáles son las tendencias actuales en el desarrollo de software que faciliten el proceso de enseñanza-aprendizaje?
2. ¿Qué aplicación tiene la programación en la carrera Ingeniería en Telecomunicaciones y Electrónica?

3. ¿Cuáles son los software y simuladores más utilizados en la enseñanza de la electrónica?
4. ¿Cómo actualizar el software Amplifiers para obtener una versión mejorada?

Con la realización del proyecto se busca llegar a una versión acabada del software Amplifiers que facilite a los estudiantes de la carrera Ingeniería en Telecomunicaciones y Electrónica el diseño de amplificadores de pequeña señal, posibilitando el uso de resistores estándares para su montaje real.

La investigación es indiscutiblemente viable, puesto que la aplicación creada está diseñada a través de un software libre que permite la utilización de una Licencia General Pública. Su desarrollo se basa en materiales y recursos que se encuentran a disposición de todos. No se necesitan recursos financieros que impidan la realización del proyecto.

Para lograr el cumplimiento de los objetivos será necesaria la utilización de métodos del nivel teórico y del nivel empírico.

Dentro de los métodos de nivel teórico se utilizaron: el analítico-sintético para la construcción del marco teórico de la investigación, en la determinación de los recursos educativos que serían manejados en la investigación; el histórico-lógico que permitió el análisis del uso de las TIC en el proceso de enseñanza-aprendizaje de la asignatura Electrónica Analógica I; y el modelado para la concepción y elaboración de los diferentes recursos didácticos digitales.

Del nivel empírico se utilizarán técnicas de búsqueda y selección de diferentes recursos existentes en Internet.

El informe de la investigación se estructurará en introducción, resumen, capitulario, conclusiones, referencias bibliográficas, bibliografía y anexos.

En el capítulo 1 se hará énfasis en las principales características del desarrollo de software educativos y la ingeniería del software. También se comentará de la importancia de las TIC en el proceso enseñanza-aprendizaje y de las ventajas de su uso, así como mencionar ejemplos de software en esta área.

En el Capítulo 2 se hace referencia a las demandas de la programación en las disciplinas principales de la carrera, haciendo énfasis en Electrónica, se realizará un diagnóstico de las

herramientas de simulación y se hará una revisión bibliográfica justificando la selección del lenguaje Python para la confección del software.

El Capítulo 3 se dedicará a la descripción de los resultados, el rediseño del software, definir cómo se va a utilizar el software y estructurar un problema para su resolución mediante la utilización de Amplifiers, para su posterior simulación y montaje real.

Las conclusiones muestran un resumen de los resultados obtenidos y están en función de los objetivos trazados. Las Recomendaciones estarán en función de trabajos futuros. La bibliografía es actualizada y con el rigor requerido para este tipo de trabajo. Al final aparecen los anexos.

CAPÍTULO 1: Las TIC y los software educativos en la Electrónica

En este capítulo se hace referencia a las ventajas y al uso de las TIC en el proceso de enseñanza-aprendizaje en la educación superior y su empleo como apoyo en las carreras técnicas. Se realiza un estudio sobre el desarrollo, creación y diseño de software, haciendo énfasis en los software educativos.

1.1. Las TIC en el proceso de enseñanza-aprendizaje

La sociedad actual ha establecido unas normas donde el potencial humano determina los cambios en todo su sistema de relaciones. Este potencial humano debe verse estimulado por unos sistemas de formación al servicio de lo que las personas necesitan para desarrollarse personalmente y para dar respuesta a los requerimientos que la propia sociedad demanda.

La Universidad es un elemento clave dentro del proceso formativo de las personas. Por tanto, su trabajo debe abarcar desde la identificación de los requerimientos formativos de la sociedad y de las personas hasta propiciar los mecanismos más adecuados para llevar a cabo su acción educativa.

La búsqueda de procesos educativos en términos de eficiencia y calidad debe estar atenta a la oportunidad que proporciona el desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC). Estas tecnologías pueden ser observadas desde su versión más simple hasta la más compleja, sin embargo, su complejidad no determina su valor educativo sino el impacto que alcance esta en la persona que aprende [2].

Se denominan TIC al conjunto de tecnologías que permiten la adquisición, producción, almacenamiento, tratamiento, comunicación, registro y presentación de informaciones, en forma de voz, imágenes y datos contenidos en señales de naturaleza acústica, óptica o electromagnética. Las TIC incluyen la electrónica como tecnología base que soporta el desarrollo de las telecomunicaciones, la informática y el audiovisual.

Las TIC cuentan con un grupo de funciones entre las cuales se encuentran [3]:

Medio de expresión y creación multimedia: para escribir, dibujar, realizar presentaciones multimedia, elaborar páginas web. Se utiliza procesadores de textos, editores de imagen y vídeo, editores de sonido, programas de presentaciones, editores de páginas web.

Canal de comunicación: facilita la comunicación interpersonal, el intercambio de ideas y materiales y el trabajo colaborativo. Se utiliza correo electrónico, chat, videoconferencias, listas de discusión y fórums.

Instrumento de productividad para el proceso de la información: crear bases de datos, preparar informes, realizar cálculos. Se tiene como instrumento hojas de cálculo, gestores de bases de datos, lenguajes de programación, programas para el tratamiento digital de la imagen y el sonido.

Fuente abierta de información y de recursos: en el caso de Internet hay "buscadores" especializados para ayudar a localizar información que se busca. Se utilizan, páginas web de interés educativo en Internet, prensa, radio, televisión.

Instrumento para la gestión administrativa y tutorial: Programas específicos para la gestión de centros y seguimiento de tutorías. Web del centro con formularios para facilitar la realización de trámites on-line

Instrumento para la evaluación: Proporciona corrección rápida y feedback inmediato, reducción de tiempos y costes, posibilidad de seguir el "rastro" del alumno, uso en cualquier ordenador (si es on-line). Programas y páginas web interactivas para evaluar conocimientos y habilidades.

Soporte de nuevos escenarios formativos: Entornos virtuales de enseñanza.

1.1.1 Ventajas de las TIC

Las TIC incluyen una serie de ventajas, entre las cuales se encuentran [3]:

Desde la perspectiva del aprendizaje.

- Interés y motivación.
- Desarrollo de la iniciativa y aprendizaje cooperativo.
- Mayor comunicación entre profesores y alumnos.
- Alfabetización digital y audiovisual.
- Desarrollo de habilidades de búsqueda y selección de información.
- Mejora de las competencias de expresión y creatividad.

Para los estudiantes.

- Acceso a múltiples recursos educativos y entornos de aprendizaje.
- Personalización de los procesos de enseñanza y aprendizaje.
- Autoevaluación y aprendizaje en menor tiempo.
- Mayor proximidad del profesor y flexibilidad en los estudios.
- Instrumentos para el proceso de la información.

Para los profesores.

- Fuente de recursos educativos para la docencia, la orientación y la rehabilitación.
- Por la variedad y amplitud de información permite al profesorado realizar agrupamientos de los estudiantes para utilizar este material.
- Mayor contacto con los estudiantes, profesores y otros centros.
- Facilitan la evaluación, control y actualización profesional.

Desde la perspectiva de los centros.

- Mejora de la administración y gestión de los centros.
- Mejora de la eficacia educativa.
- Nuevos canales de comunicación con las familias y con la comunidad local.
- Recursos compartidos.

La integración de las TIC en las instituciones de educación se debe visualizar como un proceso a largo plazo que requiere un programa sistemático a nivel macro de programas de formación a profesores. Esto, debido principalmente a que los profesores están instruidos en diferentes áreas del conocimiento y demandan capacitación en herramientas TIC para integrarlas a los planes de estudio [4].

La Educación actual necesita ser más personalizada y centrarse en el aprendizaje de los estudiantes y las TIC son una excelente herramienta para lograr esto.

Las universidades crean nuevas modalidades de estudio y establecen formas didácticas innovadoras para hacer más comprensibles los conocimientos, tomando en cuenta las diversidades de la población. Por otro lado, los docentes han tenido que especializarse en los

diferentes medios que han surgido y consideran a la alfabetización digital como prioritaria para estar a la altura de los cambios e innovaciones y como principal apoyo para conocer, dominar e integrar las herramientas tecnológicas y los nuevos elementos culturales en la práctica docente [5].

El empleo de las tecnologías como factor de innovación educativa engloba aspectos tales como: académico, técnico, administrativo, económico, cultural y político. La penetración digital es tanto técnica como cultural y se da en dos formas: formatos múltiples para la creación de software y almacenamiento de la información en forma de voz, vídeo, datos y redes integradas e interactivas para su envío.

Se plantea, que aprovechando las ventajas que aporta la utilización óptima de las TIC, se puedan crear escenarios educativos flexibles y adaptados a las necesidades de formación profesional y académica propias de este milenio. Dichos escenarios, enmarcados en un plan de formación, concientización y difusión, para delimitar en forma clara las responsabilidades de acción con conocimiento de las relaciones entre los actores clave del entorno educativo, los cuales para esta propuesta son: aprendiz, docente, medio.

Hay que destacar que el uso de las TIC no es frecuente en la totalidad de los docentes, no obstante, al estudiante actual, le gusta trabajar en el computador, explorar en Internet y chatear, entre otras cosas. Además, las posibilidades de utilizar las TIC para “hacer deberes académicos” cada vez están al alcance de más alumnos [6].

El ambiente de trabajo de las TIC es atractivo, diversifica la presentación de la información al emplear diferentes canales sensoriales. Dicho fenómeno contribuye a que la información se conserve por mayor cantidad de tiempo además de obtener una representación más acabada y por ende con nexos lógicos más consistentes que distinguen y particularizan dicha información. Además, se conjugan elementos reales y sus representaciones. Hay programas informáticos que permiten analizar con un mayor nivel de profundidad las características de los objetos, descomponerlos, reanalizarlos sin tener necesariamente que acceder al objeto real, sino a una simulación del mismo. Esto exime al individuo del compromiso de sus acciones para con el objeto, ofreciendo determinado grado de seguridad al trabajar. El

carácter interactivo ofrece independencia al estudiante, la estructuración de los programas brinda una lógica de organización que se puede extrapolar a la vida diaria que favorece el desarrollo de las operaciones lógicas del pensamiento y el aprendizaje significativo.

Los programas informáticos además registran y visualizan el progreso de los usuarios contribuyendo a la autorregulación y el replanteo de estrategias de solución por parte del docente y de los propios estudiantes. La no linealidad en la organización del contenido facilita que cada educando despliegue y utilice su estrategia de aprendizaje [7].

1.1.2 El empleo de las TIC en la Universidad

La introducción y el desarrollo de las TIC constituye un reto importante para la Universidad. El valor estratégico que la revolución tecnológica concede a la educación en general y a la Universidad en particular y a la aparición de nuevos trabajos a los que esta debe hacer frente son elementos que amplifican la importancia de la integración de estas tecnologías en la misma.

En reconocidas universidades del mundo se utilizan las TIC como nuevas vías para desencadenar procesos de aprendizaje con la finalidad de orientar al estudiante hacia la creación de su propio conocimiento a partir del conjunto de recursos de información disponibles.

Como ejemplo de esto el Estado español presenta a la Universidad como uno de los ámbitos educativos en los que se han dado los primeros pasos para la integración de las TIC y describe como más representativos los siguientes casos [8]:

- Universidad presencial con servicios complementarios virtuales: las TIC facilitan el acceso a servicios que ofrece la Universidad presencial como puede ser acceso a documentación, comunicación a través de la red con profesores y alumnos, etc. Es el caso de la Universidad Politécnica de Cataluña.
- Universidad virtual, ya sea autónoma respecto a la presencial de la que surge, en este caso se crea una universidad independiente que ofrece formación únicamente a través de la red haciendo uso de las TIC, o adscrita a la universidad presencial. Algunos ejemplos serían el de la Escuela Virtual de Negocios Deusto-Les Heures o el Centro de Estudios de Postgrado de la Universidad Politécnica de Madrid.

Entre otros ejemplos se pueden mencionar los de la Universidad de Palermo, la Universidad Autónoma de México o la famosa Universidad de Stanford. Todas estas instituciones de educación han visto el potencial de este nuevo método de aprendizaje y lo han venido desarrollando de diferentes modos. La Universidad de Palermo ha desarrollado la enseñanza de carreras de pre-grado de modo online; la Universidad de México por su parte ha creado un instituto independiente pero asociado a la universidad que además de impartir clases de pre-grado de manera online, también realiza talleres y cursos de postgrado; por su parte la Universidad de Stanford ha decidido desarrollar una plataforma online donde los usuarios de la red pueden realizar cursos y talleres de forma online completamente gratis [9].

En Cuba, desde la década de los noventa, se comenzó a implementar el Programa Nacional de Informatización de la Sociedad Cubana, definido por el Ministerio de las Comunicaciones y el Ministerio de Ciencia Tecnología y Medio Ambiente (s/a) como "(…) un proceso mediante el cual se aplican las Nuevas Tecnologías de la Información y las Comunicaciones a las diferentes esferas y sectores de la sociedad para lograr, como resultado, una mayor eficacia y eficiencia con la optimización de recursos y el logro de mayor productividad en dichas esferas". En la educación superior el trabajo con las TIC constituye una estrategia curricular que debe abordarse y concretarse desde las asignaturas y las disciplinas hasta los procesos sustantivos que tienen lugar en el centro universitario [7].

En las Universidades cubanas se adoptaron como alternativa las TIC para el apoyo en el proceso de aprendizaje, tanto en carreras Humanísticas como de Ciencias Técnicas, ejemplo de implementaron aulas virtuales que permiten a los estudiantes el acceso a las asignaturas, laboratorios virtuales, conferencias online, foros, exámenes, entre otras facilidades que brinda este entorno virtual. Entre las más destacadas se pueden mencionar el Instituto Superior José Antonio Echeverría (IPSJAE), la Universidad de Oriente (UO) y la Universidad Central "Marta Abreu" de Las Villas (UCLV) [10], [11], [12].

En el aprendizaje de las Tecnologías Complejas, surge un conjunto de problemas para los cuales las TIC pueden ser una vía de solución. En el siguiente esquema se muestran algunas deficiencias y posibles métodos para su erradicación [13].

Figura 1.1. Esquema de apoyo al aprendizaje de las Tecnologías Complejas.

1.2. Desarrollo de Software educativos

Se conoce como software al equipo lógico o soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos llamados hardware. Los componentes lógicos incluyen las aplicaciones informáticas y el llamado software de sistema [14].

El software es un elemento clave en la evolución de sistemas y productos basados en computadoras, y una de las tecnologías más importantes en todo el mundo. En los últimos años, el software ha pasado de ser la solución de un problema especializado y herramienta de análisis de la información a una industria en sí misma.

En la actualidad, el software tiene un papel dual. Es un producto y al mismo tiempo es el vehículo para entregar un producto, es un transformador de información: produce, administra, adquiere, modifica, despliega o transmite información [15].

1.2.1 Clasificación del software

Entre las clasificaciones de software se pueden mencionar:

- Software de sistemas: conjunto de programas escritos para dar servicio a otros programas. Determinado software de sistemas (por ejemplo, compiladores, editores y herramientas para administrar archivos) procesa estructuras de información complejas pero deterministas. El área de software de sistemas se caracteriza por: gran interacción con el hardware de la computadora, uso intensivo por parte de usuarios múltiples, operación concurrente que requiere la secuenciación, recursos compartidos y administración de un proceso sofisticado, estructuras complejas de datos e interfaces externas múltiples [15].
- Software de programación: conjunto de herramientas que permiten al programador desarrollar programas informáticos, usando diferentes alternativas y lenguajes de programación.
- Software de aplicación: permite a los usuarios llevar a cabo una o varias tareas específicas, en cualquier campo de actividad susceptible de ser automatizado o asistido [16].
- Software de ingeniería y ciencias: se ha caracterizado por algoritmos “devoradores de números”.
- Software incrustado: reside dentro de un producto o sistema y se usa para implementar y controlar características y funciones para el usuario final y para el sistema en sí. Ejecuta funciones limitadas y particulares o provee una capacidad significativa de funcionamiento y control.
- Software de línea de productos: es diseñado para proporcionar una capacidad específica para uso de muchos consumidores diferentes. El software de línea de productos se centra en algún mercado limitado y particular o se dirige a mercados masivos de consumidores.
- Aplicaciones web: llamadas “webapps”, esta categoría de software centrado en redes agrupa una amplia gama de aplicaciones. En su forma más sencilla, las webapps son poco más que un conjunto de archivos de hipertexto vinculados que presentan información con uso de texto y gráficas limitadas.

- Software de inteligencia artificial: hace uso de algoritmos no numéricos para resolver problemas complejos que no son fáciles de tratar computacionalmente o con el análisis directo [15].

1.2.2 Ingeniería del software

La ingeniería de software es una tecnología con varias capas. Como se aprecia en la figura 1.2 [15].

Figura 1.2. Capas de la ingeniería de software.

El fundamento en el que se apoya la ingeniería de software es el compromiso con la calidad que está estrechamente vinculado con la capa proceso. El proceso de ingeniería de software es el aglutinante que une las capas de la tecnología y permite el desarrollo racional y oportuno del software de cómputo. El proceso define una estructura que debe establecerse para la obtención eficaz de tecnología de ingeniería de software. El proceso de software forma la base para el control de la administración de proyectos de software, y establece el contexto en el que se aplican métodos técnicos, se generan productos del trabajo, se establecen puntos de referencia, se asegura la calidad y se administra el cambio de manera apropiada.

Los métodos de la ingeniería de software proporcionan la experiencia técnica para elaborar software. Incluyen un conjunto amplio de tareas, como comunicación, análisis de los requerimientos, modelación del diseño, construcción del programa, pruebas y apoyo. Los métodos de la ingeniería de software se basan en un conjunto de principios fundamentales que gobiernan cada área de la tecnología e incluyen actividades de modelación y otras

técnicas descriptivas. Las herramientas de la ingeniería de software proporcionan un apoyo automatizado o semiautomatizado para el proceso y los métodos [15].

1.2.3 Proceso de creación del software

Se define como proceso al conjunto ordenado de pasos a seguir para llegar a la solución de un problema u obtención de un producto.

El proceso de creación de software puede llegar a ser muy complejo, dependiendo de su porte, características y criticidad del mismo. Se divide en tres categorías según su tamaño (líneas de código) o costo: de pequeño, mediano y gran porte.

En el contexto de la ingeniería de software, un proceso no es una prescripción rígida de cómo elaborar software de cómputo. Por el contrario, es un enfoque adaptable que permite que las personas que hacen el trabajo (el equipo de software) busquen y elijan el conjunto apropiado de acciones y tareas para el trabajo [15].

Una estructura de proceso general para la ingeniería de software consta de cinco actividades [15]:

- Comunicación: busca entender los objetivos de los participantes respecto del proyecto, y reunir los requerimientos que ayuden a definir las características y funciones del software.
- Planeación: cualquier viaje complicado se simplifica si existe un mapa. Un proyecto de software es un viaje difícil, y la actividad de planeación crea un “mapa” que guía al equipo mientras viaja. El mapa, llamado plan del proyecto de software, define el trabajo de ingeniería de software al describir las tareas técnicas por realizar, los riesgos probables, los recursos que se requieren, los productos del trabajo que se obtendrán y una programación de las actividades.
- Modelado: Un programador hace un bosquejo al crear modelos a fin de entender mejor los requerimientos del software y el diseño que los satisface.
- Construcción: esta actividad combina la generación de código y las pruebas que se requieren para descubrir errores en este.

- Despliegue: el software (como entidad completa o como un incremento parcialmente terminado) se entrega al consumidor que lo evalúa y que le da retroalimentación, misma que se basa en dicha evaluación.

1.2.4 Software educativo

El software educativo se encuentra dentro de la clasificación de software de aplicación.

Los software educativos (SE), se definen de forma genérica como aplicaciones o programas computacionales que faciliten el proceso de enseñanza aprendizaje. Algunos autores lo conceptualizan como cualquier programa computacional cuyas características estructurales y funcionales sirvan de apoyo al proceso de enseñar, aprender y administrar, o el que está destinado a la enseñanza y el autoaprendizaje y además permite el desarrollo de ciertas habilidades cognitivas; términos que seguramente se replantearán en la medida que se introduzcan nuevos desarrollos tecnológicos para el trabajo en red en Internet [17].

El SE es un programa informático cuyo principal objetivo es la enseñanza o el autoaprendizaje. El uso de software y hardware en educación y capacitación se remonta a principios de los años 40, cuando los investigadores estadounidenses pudieron desarrollar simuladores de vuelo que usaban computadoras analógicas para generar datos de instrumentos simulados a bordo [18].

Las características más generalizadas en los SE son [17]:

- Finalidad: orientados a la enseñanza-aprendizaje en todas sus formas.
- Utilización del computador: el medio utilizado como soporte es el computador.
- Facilidad de uso: son intuitivos y aplica reglas generales de uso y de fácil comprensión para su navegabilidad o desplazamiento y recursividad o posibilidad de regreso a temáticas de interés desde cualquier punto en el ambiente virtual.
- Interactividad: permite un intercambio efectivo de información con el estudiante.

El software educativo integra contenido multimedia y proporciona a los usuarios un alto nivel de interactividad. Las dos características las distinguen de las prácticas de enseñanza tradicionales. El contenido multimedia, como gráficos, imágenes y sonido, ayuda a los alumnos a participar en sus clases. Además, un software de educación en línea beneficia a

los docentes, lo que les permite conectarse mejor con los estudiantes y ayudarlos a mantener interesados en una lección y también promueve un ambiente de aprendizaje productivo [18].

El software educativo, constituye pues una herramienta imprescindible para el desarrollo de los procesos de enseñanza aprendizaje en el presente siglo, en el que se vive ya en una sociedad de la información y el conocimiento y las tecnologías invaden todas las esferas de la vida humana.

1.3. Diseño de software

El proceso de diseño comprende un conjunto de principios, conceptos y prácticas que permiten a un ingeniero de software modelar el sistema o producto que se construirá. Este modelo, conocido como modelo de diseño, se evalúa para determinar su calidad y se revisa antes de que se genere un código y se realicen pruebas. El modelo de diseño proporciona detalles sobre las estructuras de datos implementar el sistema. Los principios de diseño establecen una filosofía general que guía el trabajo de diseño que debe ejecutarse. Deben entenderse los conceptos de diseño antes de aplicar la mecánica de este, y la práctica del diseño en sí lleva a la creación de distintas representaciones del software que sirve como guía para la actividad de construcción que siga [15], [19].

El diseño de software es una fase de la ingeniería de software, en la que se desarrolla un plan para servir como base para construir el sistema de software. IEEE define el diseño de software como “un proceso de definición de la arquitectura, componentes, interfaces y otras características de un sistema o componente y el resultado de ese proceso” [19].

El objetivo del diseño es producir un modelo o representación que tenga resistencia, funcionalidad y belleza. Para lograrlo, debe practicarse la diversificación y luego la convergencia.

La diversificación y la convergencia combinan la intuición y el criterio con base en la experiencia en la construcción de entidades similares, un conjunto de principios heurísticos que guían la forma en la que evoluciona el modelo, un conjunto de criterios que permiten evaluar la calidad y un proceso iterativo que finalmente conduce a una representación del diseño definitivo.

El diseño de software se ubica en el área técnica de la ingeniería de software y se aplica sin importar el modelo del proceso que se utilice. El diseño del software comienza una vez que se han analizado y modelado los requerimientos, es la última acción de la ingeniería de software dentro de la actividad de modelado y prepara la etapa de construcción (generación y prueba de código).

El diseño de software es un proceso iterativo por medio del cual se traducen los requerimientos en un "plano" para construir el software. El diseño se representa en un nivel alto de abstracción, en el que se rastrea directamente el objetivo específico del sistema y los requerimientos más detallados de datos, funcionamiento y comportamiento. A medida que tienen lugar las iteraciones del diseño, las mejoras posteriores conducen a niveles menores de abstracción.

Sin importar el método de diseño que se utilice, debe aplicarse un conjunto de conceptos básicos al diseño en el nivel de datos, arquitectura, interfaz y componente [15].

1.3.1 Principios para el diseño de software y consideraciones de diseño

Los principios básicos de diseño permiten al ingeniero de software navegar por el proceso de diseño. A continuación, se mencionan algunos de ellos:

- El diseño debe "minimizar la distancia intelectual" entre el software y el problema tal como existe en el mundo real. Es decir, la estructura del diseño del software debe, siempre que sea posible, imitar la estructura del dominio del problema [20].
- El diseño del software debe ser uniforme e integrado. El diseño del software se considera uniforme e integrado, si las interfaces están definidas correctamente entre los componentes de diseño. Para esto, las reglas, el formato y los estilos se establecen antes de que el equipo de diseño comience a diseñar el software.
- El diseño del software debe estructurarse para degradarse suavemente. El software debe diseñarse para manejar cambios y circunstancias inusuales, y si surge la necesidad de terminación, debe hacerlo de manera adecuada para que la funcionalidad del software no se vea afectada [19].
- El diseño no codifica, la codificación no es diseño. Incluso cuando se crean diseños de procedimientos detallados para los componentes del programa, el nivel de

abstracción del modelo de diseño es más alto que el código fuente. Las únicas decisiones de diseño tomadas a nivel de codificación deben abordar los pequeños detalles de implementación que permiten codificar el diseño del procedimiento.

- El diseño debe evaluarse para la calidad tal como está siendo creada, no después del hecho. Una variedad de conceptos de diseño y medidas de diseño están disponibles para ayudar al diseñador a evaluar la calidad en todo el proceso de desarrollo.
- El diseño debe revisarse para minimizar los errores conceptuales (semánticos). A veces hay una tendencia a centrarse en las minucias cuando se revisa el diseño, faltando el bosque para los árboles. Un equipo de diseño debe asegurarse de que se hayan abordado los principales elementos conceptuales del diseño (omisiones, ambigüedad, inconsistencia) antes de preocuparse por la sintaxis del modelo de diseño [20].
- El diseño del software debe representar la correspondencia entre el software y el problema del mundo real. El diseño del software debe estructurarse de manera que siempre se relacione con el problema del mundo real.
- Reutilización del software. Los componentes de software deben diseñarse de tal manera que puedan reutilizarse de manera efectiva para aumentar la productividad [19].

Hay muchos aspectos a considerar en el diseño de una pieza de software. La importancia de cada consideración debe reflejar los objetivos y expectativas que el software se está creando para cumplir. Algunos de estos aspectos son [21]:

Compatibilidad: el software puede funcionar con otros productos diseñados para la interoperabilidad con otro producto. Por ejemplo, un software puede ser compatible con una versión anterior de sí mismo.

Extensibilidad: se pueden agregar nuevas capacidades al software sin grandes cambios en la arquitectura subyacente.

Modularidad: el software resultante comprende componentes independientes bien definidos que permiten un mejor mantenimiento. Los componentes podrían luego implementarse y

probarse de forma aislada antes de integrarse para formar un sistema de software deseado. Esto permite la división del trabajo en un proyecto de desarrollo de software.

Tolerancia a fallas: el software es resistente y capaz de recuperarse de fallas en los componentes.

Capacidad de mantenimiento: una medida de la facilidad con la que se pueden corregir errores o realizar modificaciones funcionales. La alta capacidad de mantenimiento puede ser el producto de la modularidad y la extensibilidad.

Confiabilidad (durabilidad del software): el software puede realizar una función requerida bajo las condiciones establecidas por un período de tiempo específico.

Reutilización: la capacidad de utilizar algunos o todos los aspectos del software preexistente en otros proyectos con poca o ninguna modificación.

Robustez: el software puede funcionar bajo tensión o tolerar entradas impredecibles o no válidas. Por ejemplo, puede diseñarse con resistencia a condiciones de poca memoria.

Seguridad: el software es capaz de resistir y resistir actos e influencias hostiles.

Usabilidad: la interfaz de usuario del software debe ser utilizable para su usuario / audiencia objetivo. Los valores predeterminados para los parámetros se deben elegir para que sean una buena opción para la mayoría de los usuarios.

Rendimiento: el software realiza sus tareas dentro de un marco de tiempo que es aceptable para el usuario y no requiere demasiada memoria.

Portabilidad: el software debe poder utilizarse en diferentes condiciones y entornos.

Escalabilidad: el software se adapta bien al aumento de datos o al número de usuarios.

1.3.2 Rediseño de software

Una actividad de diseño importante que se sugiere para muchos métodos ágiles es el rediseño, técnica de reorganización que simplifica el diseño (o código) de un componente sin cambiar su función o comportamiento [15].

El proceso de mejora de software (SPI) se ha centrado tradicionalmente en abordar cómo mejorar las capacidades de una organización de desarrollo de software mediante la maduración y la comparación de los procesos de software.

El proceso de rediseño de software (SPR) se preocupa por la identificación, aplicación y refinamiento de nuevas formas de SPI. Los procesos de software de interés incluyen no solo los asociados con el desarrollo de software, sino también los de adquisición de sistemas de software, uso y evolución. La heurística de rediseño sirve como la principal fuente de conocimiento sobre cómo mejorar drásticamente el ciclo de tiempo, la prevención de defectos y la rentabilidad de varios tipos de software [22].

Cuando se rediseña el software, se examina el diseño existente en busca de redundancias, elementos de diseño no utilizados, algoritmos ineficientes o innecesarios, estructuras de datos mal construidas o inapropiadas y cualquier otra falla del diseño que pueda corregirse para obtener un diseño mejor [15].

Los modelos de procesos de software se pueden analizar de varias maneras. Estos análisis generalmente están dirigidos a mejorar la calidad del modelo de proceso, así como también a detectar o prevenir los errores y omisiones comunes que aparecen en los modelos grandes.

En primer lugar, es necesario analizar la consistencia, la integridad, la trazabilidad y la corrección de múltiples modelos de procesos interrelacionados. Esto es algo análogo a lo que ocurre en un proyecto de desarrollo de software cuando se usan notaciones múltiples, por lo tanto, se requiere un análisis a través de las notaciones del software.

En segundo lugar, es necesario tener en cuenta los recursos del proceso de software durante todo el esfuerzo de rediseño. SPR puede cambiar el flujo de recursos a través de un proceso y se observan estos cambios en el rendimiento del proceso [22].

1.3.3 Software de apoyo al aprendizaje de la Electrónica

En 2009 un grupo de profesionales desarrollaron un software para la electrónica que se implementa como laboratorio virtual. El mismo describe un sistema de autoevaluación, llamado SIPAE (Sistema Integrado para aprendizaje y Evaluación) que combina un conjunto de cuestionarios con experimentos virtuales interactivos. El laboratorio virtual, que forma el conjunto de experimentos, muestra al alumno el circuito práctico bajo análisis e incluso lo

genera de acuerdo con su respuesta, y le da la oportunidad de experimentar con él como si se encontrase en el laboratorio real.

La herramienta de autoevaluación se plantea desde una perspectiva pedagógica de evaluación por competencias y está constituida por un conjunto de cuestionarios, cada uno de los cuales está asociado a un experimento del laboratorio virtual. Proporciona al alumno diversas competencias generales y específicas situadas entre la clase magistral y el laboratorio.

Para formular los cuestionarios de autoevaluación sobre los dispositivos electrónicos se utiliza la tabla 1.1.

Tabla 1.1. Metodología para la elaboración de cuestionarios.

TAXONOMÍA DE BLOOM	MODELO IDEAL	APROXIMA-CIÓN	MODELO REAL	COMPETENCIAS
CONOCIMIENTO	Principios de funcionamiento	Estructura del dispositivo con nociones del componente real	Estructura del dispositivo real	Describir estructura y funcionamiento
CONOCIMIENTO Y COMPRENSIÓN	Funcionamiento grafico	Funcionamiento grafico	Funcionamiento gráfico	Comprender el funcionamiento
CONOCIMIENTO Y COMPRENSIÓN	Leyes de funcionamiento Ecuaciones	Leyes de funcionamiento Ecuaciones	Leyes de funcionamiento Ecuaciones	Comprender el funcionamiento
ANÁLISIS APLICACIÓN	Estudio de circuitos básicos de aplicación	Estudio de circuitos básicos de aplicación	Estudio de circuitos básicos de aplicación	Analizar y utilizar en circuitos básicos
APLICACIÓN SINTESIS	Resolución de problemas	Resolución de problemas	Resolución de problemas	Construir, analizar y reparar circuitos

SINTESIS Y EVALUACIÓN	Diseño de circuitos	Diseño de circuitos	Diseño de circuitos	Diseñar circuitos básicos electrónicos

Cada celda interior está formada por un cuestionario de cinco preguntas (reactivos). Cada pregunta consta de un enunciado, una figura o esquema electrónico, un enlace con el experimento virtual interactivo y un conjunto de respuestas (opciones) de las que una es correcta y las demás son incorrectas (elementos de distracción). En total y en las celdas centrales de la tabla se elaboran 18 cuestionarios, cada uno de los cuales tiene cinco preguntas que a su vez constan de cinco elementos de distracción y una respuesta correcta. En total, en la tabla, hay 90 preguntas.

El alumno, debe resolver los cuestionarios avanzando por las celdas interiores de la tabla, moviéndose de izquierda a derecha y de arriba abajo, para ser consciente en todo momento del nivel aprendizaje que alcanza. Dicho nivel lo conoce cualitativamente por la posición en la tabla, porque en cada instante sabe cuáles son las competencias que está desarrollando y cuales le faltan, y cuantitativamente porque el sistema le da una valoración numérica [13].

En el organigrama de la figura 1.3 se muestra, de forma resumida, la metodología desarrollada para el proceso de autoevaluación y aprendizaje.

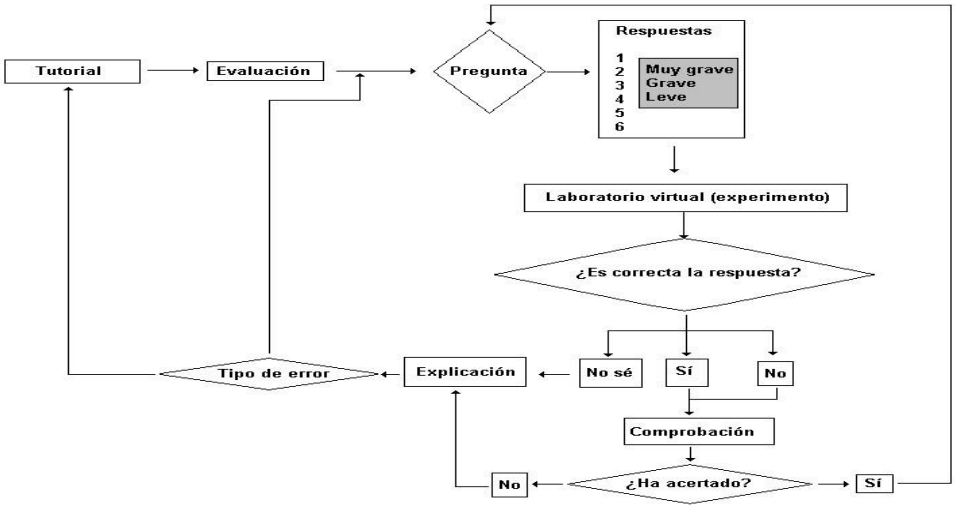


Figura 1.3. Organigrama del sistema SIPAE.

En la figura 1.4 se muestra un ejemplo sencillo del sistema SIPAE en el tema de diodos.

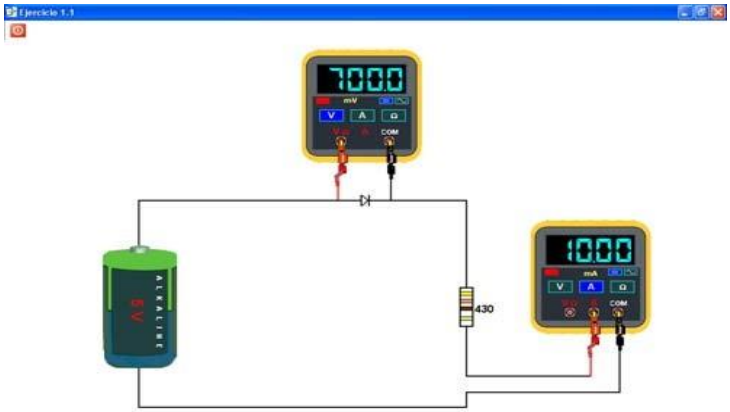


Figura 1.4. Tensión en bornes del diodo y corriente que circula por el circuito.

1.4. Conclusiones del Capítulo

En el capítulo quedó reflejada la influencia e impacto de las TIC en la universidad actual y se comprobó la importancia de los software educativos como una vía para el apoyo al proceso de enseñanza-aprendizaje, con énfasis en las carreras técnicas.

CAPÍTULO 2: Programación y simulación en la carrera Ingeniería en Telecomunicaciones y Electrónica

El presente capítulo se centra en el estudio de la importancia y el uso de la programación en la carrera Ingeniería en Telecomunicaciones y Electrónica, la necesidad de la utilización de herramientas de simulación para el aprendizaje de la asignatura de Electrónica. Además, se da a conocer la aplicación Amplifiers como una variante para el apoyo al aprendizaje de la asignatura y se exponen las razones por la cual se elige Python como lenguaje de programación.

2.1 Aplicación de la Programación en la Carrera Ingeniería en Telecomunicaciones y Electrónica

En la carrera Ingeniería en Telecomunicaciones y Electrónica la programación es uno de los pilares fundamentales para el desempeño de muchas asignaturas, ya sea mediante la utilización de programas y software como en la propia programación de aplicaciones que contribuyen al desarrollo de las mismas.

La disciplina de Computación para el plan vigente en la carrera cuenta con una serie de asignaturas dedicadas específicamente a la programación, esta disciplina tiene como objetivos generales instructivos:

- Familiarizarse con los conceptos fundamentales de la algorítmica y las herramientas básicas de las computadoras personales:
    - Dominar los elementos fundamentales de la técnica de computación, sus partes integrantes y las relaciones entre ellas.
    - Sentar las bases para utilizar bibliotecas de programas orientados a un mejor empleo de los recursos de la PC.
- Elaborar programas de computación en lenguajes de alto nivel, de utilización general para aplicaciones específicas de la profesión:
    - Explotar un sistema específico con utilitarios, bibliotecas, facilidades de edición y puesta a punto, etc.

- Utilizar las computadoras personales como herramientas de diferentes asignaturas.
    - Desarrollar algoritmos eficientes de computación.
    - Aplicar eficientemente las microcomputadoras personales y sus herramientas básicas de programación.
    - Elaborar programas que utilicen las facilidades de los sistemas operativos.
    - Aplicar las herramientas que brindan ambientes de programación. Para la simulación y caracterización de sistemas, así como para el comercio electrónico.

A continuación, se expondrán las asignaturas pertenecientes a la disciplina de computación y los objetivos de las mismas.

Programación I:

- Utilizar eficientemente un lenguaje de alto nivel de propósito general en la programación de algoritmos relacionados con el cálculo científico y aplicaciones a su perfil de trabajo.
- Explotar un sistema específico de programación con sus utilitarios, bibliotecas, facilidades de edición y puesta a punto, etc.
- Elaborar programas que permitan fácilmente la interpretación por el hombre y que en su ejecución brinden facilidades de interacción con el sistema.
- Describir estructuras de datos básicos (arreglos y estructuras) y los algoritmos generales para su manipulación.
- Utilización de algoritmos relacionados con las aplicaciones afines a la carrera.
- Analizar y sintetizar algoritmos de mediana complejidad.

Programación II:

- Conocer de forma avanzada el lenguaje C#
- Saber los aspectos avanzados de la programación orientada a objeto
- Saber los aspectos avanzados de la programación con ventanas o interfaz gráfica de usuarios.

- Elaborar programas de computación en lenguaje de alto nivel para aplicaciones específicas de la profesión [23].

Además, se incluyen las asignaturas Introducción a la Computación e Instrumentación Virtual como optativas de primero y tercer año respectivamente.

2.1.1 Programación en las disciplinas de la carrera.

En el mundo actual ya todo es programado desde el diseño de un circuito hasta la interfaz gráfica, de ahí destacar la importancia de la programación en sentido general.

La carrera Ingeniería en Telecomunicaciones y Electrónica está conformada por 5 disciplinas catalogadas como principales, las cuales agrupan la mayor cantidad de asignaturas vinculadas a las esferas de actuación del ingeniero:

1. Electrónica
2. Teoría de las Comunicaciones
3. Sistemas de Telecomunicaciones
4. Sistemas de Radiocomunicaciones

Con el objetivo de indagar sobre las posibilidades de aplicación de la programación en asignaturas de la carrera, se desarrolló una entrevista a los jefes de disciplina de la misma, donde se les preguntó la importancia que le concedían a la programación para la formación del profesional en las asignaturas de cada disciplina y los software más utilizados en estas para apoyar el proceso de enseñanza aprendizaje.

En la disciplina de Sistemas de Radiocomunicaciones desde el punto de vista docente no es bien aprovechada, por la falta de proyectos que faciliten la realización de herramientas para su apoyo. Se pudiera utilizar con una mayor frecuencia en asignaturas como: Radiopropagación, Sistemas de Radiocomunicaciones, Comunicaciones móviles, Televisión Digital, entre otras. En la asignatura que más se emplea es Antena, para el cálculo y diseño de las mismas, que en muchos casos se realizan iteraciones.

En las asignaturas de Microprocesadores es un elemento fundamental, que contribuye a obtener una lógica de programación sin importar el lenguaje con que se esté trabajando. Se

utiliza lenguaje de programación de bajo nivel (Ensamblador) que se asemeja al lenguaje C utilizado en otras asignaturas de la carrera.

En las Telecomunicaciones actuales cada vez se utilizan más software que requieren conocimientos básicos de programación esto se evidencia en la disciplina de Teoría de las Comunicaciones donde en todas sus asignaturas se emplea el asistente matemático Matlab de gran aplicación práctica. Un ejemplo de esto lo constituye las redes de telefonía móvil de quinta generación donde toda la configuración se realiza por software.

En la disciplina de Sistemas de Telecomunicaciones la programación constituye un pilar fundamental, las nuevas tecnologías como las Redes Definidas por Software hacen de la red un ambiente totalmente programable, donde Python y Java son los principales lenguajes. Cada administrador puede correr su propio script de configuración de red, además de crear aplicaciones propias.

En la disciplina de Electrónica la programación complementa la teoría con la práctica mediante la simulación. Se crean entornos virtuales que facilitan el apoyo al proceso de enseñanza-aprendizaje, debido a limitaciones para ejercer laboratorios prácticos modernos. La programación permite realizar esquemas tanto con resistores, condensadores, bobinas como con elementos más avanzados. Además, se ha incursionado en el desarrollo de programas entrenadores

2.2 La Simulación de circuitos en la Electrónica

Un simulador de circuitos electrónicos es una herramienta de software utilizada por profesionales en el campo de la electrónica y los estudiantes de las carreras de tecnologías de información. Ayuda a crear algún circuito que se desee ensamblar, ayudando a entender mejor el mecanismo, y ubicar las fallas dentro del mismo de manera sencilla y eficiente.

Los simuladores de circuitos cuentan con múltiples herramientas que te permiten realizar casi cualquier circuito, se pueden colocar circuitería básica como resistores, condensadores, fuentes de voltaje o LED; también se pueden usar semiconductores como compuertas AND,

OR, XOR, XAND y circuitería más compleja como un temporizador, biestables (flip-flop), Buffer y Unidades aritmétputasica y lógica [24].

2.2.1 Herramientas de simulación

Para la simulación de circuitos electrónicos se utilizan un gran número de herramientas de simulación, algunas más utilizadas que otras. Entre las más comunes se encuentran las ejemplificadas a continuación:

TinyCad

Es un software de dibujo esquemático para Windows de la empresa Sourceforge. Compatible con el dibujo esquemático, desarrollo de diseño y simulación de circuitos. Este programa ayuda al dibujo de diagramas de circuitos e incluye bibliotecas de símbolos para comenzar de forma inmediata. Además de ser capaz de imprimir sus diseños, puede utilizar TinyCAD para publicar dibujos, copiar y pegar en un documento Word o guardar como un mapa de bits PNG para web.

Dia

Es un software de dibujo básico adecuado para la elaboración de diagramas de bloques y proporciona acceso a algunos componentes importantes de la electrónica. Este software es recomendado sólo para un principiante o alguien nuevo en el área de dibujo de circuitos electrónicos. El software tiene una licencia GPL y está hecho para Mac y Linux.

Orcad

Es un software altamente popular de la empresa Cadence para el dibujo de circuitos, el desarrollo de diseño y simulación.

Tina

Es una solución asequible para las pequeñas industrias y autónomos. Facilita el dibujo de circuitos, desarrollos de diseño, simulación y otras características como permitir la prueba en tiempo real de los circuitos [25].

SPICE

Es un acrónimo inglés de *Simulation Program with Integrated Circuits Emphasis* (Programa de simulación con énfasis en circuitos integrados). Es un estándar internacional cuyo objetivo es simular circuitos electrónicos analógicos compuestos por resistencias, condensadores, diodos, transistores, etc. Para ello hay que describir los componentes, el circuito y luego elegir el tipo de simulación (temporal, en frecuencia, en continua, paramétrico, Montecarlo, etc.) [26].

EveryCircuit

Es una herramienta para el diseño de circuitos que a través de su simulador interactivo deja ver qué es lo que hace cada uno de los componentes que se agregan y cómo se comportan cuando están conectados entre sí. Entre los componentes que se pueden agregar se encuentran resistencias, condensadores, transistores, transformadores, interruptores, lámparas, entre otros. Se puede probar EveryCircuit en Windows, Linux, Mac, también está disponible para dispositivos iOS y Android. La herramienta requiere tener nociones básicas sobre circuitos, al menos para saber el nombre o la función de cada uno de los componentes que se pueden agregar [27].

ElectroDroid

Es considerada como la navaja suiza de la electricidad y electrónica. Contiene una gran colección de herramientas y tablas de referencias imprescindibles para quién se dedique a este sector. Además de múltiples recursos no catalogados en estas clasificaciones. Es sin duda una de las herramientas imprescindibles a la hora de llevarla con nosotros.

Dentro de la aplicación, ElectroDroid se encuentra dividido en tres categorías diferentes: Calculadoras, para realizar distintas operaciones numéricas, Pin-out, tablas y diagramas de conexiones y la categoría de Recursos, diferentes recursos de la electrónica y la electricidad.

En el apartado de las calculadoras, se encuentran una lista de 23, para realizar los cálculos pertinentes de una forma sencilla. Se basa en una serie de valores editables donde se introducen los valores deseables y se obtienen las cifras buscadas de una forma sencilla.

Las calculadoras disponibles son las siguientes: Código de colores de resistencias, código de resistencias SMD, código de colores de inductores, calculador de reactancia, filtros,

EveryCircuit, Circuit Simulator, divisor resistivo, ratio de resistencias, resistencias valores serie/paralelo, carga del condensador, amplificadores operacionales, resistencias para LED, calculadora LM317, calculadora NE555, disipación de energía, cálculo de vida de baterías, herramienta de diseño de inductores, calculadora de caída de voltaje, calculadora de ancho de pista para PCB, conversor de decibelios, conversor de frecuencias y conversor analógico-digital [28].

2.2.2 Herramientas de simulación más utilizadas en la Facultad de Ingeniería Eléctrica

En la Facultad de Ingeniería Eléctrica de la Universidad “Marta Abreu” de Las Villas se utilizan también simuladores de circuitos electrónicos y en este caso los más utilizados son:

MultiSim (Electronics Workbench)

Multisim es el mejor entorno de simulación SPICE estándar de la industria. Es la piedra angular de la solución de enseñanza de circuitos de National Instrument (NI) para construir experiencia a través de la aplicación práctica en el diseño, creación de prototipos y prueba de circuitos eléctricos. El enfoque de diseño Multisim ayuda a guardar prototipos de iteraciones y optimizar diseños de placas de circuitos impresos (PCB) al principio del proceso [29].

Multisim es una poderosa herramienta para el diseño electrónico. Fue creado acorde a las necesidades de educadores y estudiantes, además de cumplir ampliamente con los requerimientos de los ingenieros y diseñadores a nivel profesional. Cuenta con nuevas características técnicas como puntas de prueba industriales, intercambio de datos con instrumentos virtuales y "reales", corrector de errores y sugerencias de cambios sobre el circuito, simulación integrada con microcontroladores [30].

Los estudiantes pueden usar 20 análisis diferentes en Multisim para ayudarlos a comprender completamente el comportamiento del circuito en las clases de potencia analógica, digital y de potencia en toda la universidad. Multisim incluye análisis desde transitorios básicos y simulación AC hasta barridos avanzados de parámetros y simulación de ruido.

Multisim está equipado con más de 36,000 componentes validados por los principales fabricantes de semiconductores. La completa biblioteca Multisim de amplificadores, diodos,

transistores y fuentes de alimentación conmutadas actualizadas junto con simulación avanzada hace posible cubrir una amplia variedad de temas más rápidamente.

Se integra perfectamente con myDAQ y el NI Instrumental Educativo Virtual Instrumentation Suite (NI ELVIS). Con estas plataformas de educación y laboratorio, los estudiantes pueden comparar los resultados simulados de la tarea con los resultados de laboratorio adquiridos en un solo entorno.

Aplicaciones de enseñanza

Los estudiantes pueden usar Multisim para optimizar el rendimiento de diseño de su circuito y guardar prototipos de iteraciones en diferentes áreas de aplicación, como diseño analógico, electrónica de potencia, energía renovable y diseños completos de nivel de sistema analógico / digital.

Aplicaciones de diseño de circuito

Los ingenieros pueden usar Multisim para optimizar el rendimiento de diseño de su placa de circuito impreso (PCB) y guardar prototipos de iteraciones en diferentes áreas de aplicación como diseño de circuitos analógicos, electrónica de potencia, energía renovable y simulación completa de sistema analógico / digital con fácil integración de hardware [29].

Simulink

Simulink es un entorno de programación visual, que funciona sobre el entorno de programación Matlab.

Es un entorno de programación de más alto nivel de abstracción que el lenguaje interpretado Matlab (archivos con extensión .m). Simulink genera archivos con extensión. mdl (de "model").

En las imágenes, se puede apreciar el diagrama en bloques de un Radar, en el cuál se muestra que uno de sus bloques de procesamiento de señal, es un filtro Kalman realizado en un script de Matlab.

Luego, se puede apreciar un sistema de control automático, junto a su modelización y finalmente un sistema de un automóvil, vinculando la simulación a un entorno de realidad virtual.

Simulink viene a ser una herramienta de simulación de modelos o sistemas, con cierto grado de abstracción de los fenómenos físicos involucrados en los mismos. Se hace hincapié en el análisis de sucesos, a través de la concepción de sistemas (cajas negras que realizan alguna operación).

Es ampliamente usado en Ingeniería Electrónica en temas relacionados con el procesamiento digital de señales (DSP), involucrando temas específicos de ingeniería biomédica, telecomunicaciones, entre otros. También es muy utilizado en Ingeniería de Control y Robótica [31], [32].

Proteus

Proteus VSM es un completo entorno de diseño, que permite realizar todas las tareas de diseño de circuitos electrónicos, tales como: dibujo de esquemas de circuitos, simulación interactiva de circuitos analógicos, digitales, y con microcontroladores, con animación en tiempo real, además del diseño de circuitos impresos. Cuenta con una extensa librería de componentes genéricos y específicos [30].

Proteus es una aplicación para la ejecución de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño del esquema electrónico, programación del software, construcción de la placa de circuito impreso, simulación de todo el conjunto, depuración de errores, documentación y construcción.

Sin la utilización de la suite Proteus, el proceso para construir un equipo electrónico basado en un microprocesador se compone de cinco etapas. Sólo al final del proceso se de detectan los errores y cualquier problema exige volver a ejecutar el ciclo completo.

El depurado de errores puede convertirse en una labor ardua en tiempo y recursos, lo que conlleva un alto coste económico. Sin embargo, con la herramienta Proteus el proceso queda definido de la siguiente manera:

Fases del diseño utilizando Proteus

Con Proteus las fases de prueba no suponen la necesidad de volver a construir nuevos prototipos, con el ahorro de costos y tiempo que ello supone.

Los diferentes módulos que componen Proteus se pueden adquirir de forma independiente añadiendo nuevas funcionalidades a medida que aumentan las necesidades de desarrollo y producción. Además, la capacidad de simular cada una de las familias de microprocesadores también es objeto de adquisición por separado. De esta manera podemos empezar adquiriendo unas funcionalidades básicas e ir adquiriendo progresivamente nuevas características aprovechando al máximo nuestras inversiones en la herramienta y asegurar al máximo los costes de inversión en el software.

En el mundo de la formación, Proteus se muestra como una herramienta que permite al alumno realizar modificaciones tanto en el circuito como en el programa, experimentando y comprobando de forma inmediata los resultados y permitiéndole de esta forma aprender de forma práctica y sin riesgos de estropear materiales de elevado coste.

Si se desea simular el funcionamiento electrónico del circuito, el funcionamiento lógico del programa cargado en el microprocesador, construir la placa de circuito impreso, documentar todo el proceso y obtener vistas en tres dimensiones, Proteus le ofrece una herramienta completa a un precio competitivo [33].

Por otra parte, este software está conformado por dos dispositivos principales: ISIS y ARES. El primero de ellos será el encargado de la generación de circuitos reales. Una ventaja que te ofrecerá esta herramienta es la de evaluar el funcionamiento de tu trabajo en un PCB. De ese modo prevendrás errores y corregirás fallas.

En cuanto a ARES, su función consiste en enrutar, situar y editar los diferentes componentes que hacen a la fabricación de placas PCB. En este sentido, convierte tu diseño de modo que pueda ser aplicado en cada pieza de un circuito impreso. Además, se dispone de un calificado espacio de simulación generado con la mejor tecnología. En concreto, se cuenta con los beneficios y potencial de VSM, a cargo de la modelación de sistemas virtuales [34].

2.2.3 Aplicación de la simulación

Para comprobar la importancia que tiene la utilización de software en las carreras técnicas, específicamente en la carrera Ingeniería en Telecomunicaciones y Electrónica se realizó una encuesta (ver anexo 1), a los estudiantes de segundo, tercer y quinto año que cursan esta carrera en la Universidad Central "Marta Abreu" de Las Villas.

La encuesta realizada tiene como objetivo el de identificar las necesidades de la utilización de software en la carrera Ingeniería en Telecomunicaciones y Electrónica.

Para la misma se tuvieron en cuenta los siguientes indicadores:

- Repercusión de la programación y la utilización de software en la carrera.
- Posibilidades de aplicación de la programación y el trabajo con software educativos en las diferentes asignaturas y trabajos científicos.
- Utilización de software con la disciplina de Electrónica.

Se tomó una muestra de aproximadamente el 33% de los estudiantes de cada año y se obtuvieron los siguientes resultados:

El 100% de los estudiantes consideran importante la programación en la carrera y asignaturas en particular argumentando que la misma desarrolla el pensamiento lógico, contribuye a ampliar el perfil del Ingeniero en Telecomunicaciones y Electrónica siendo aplicable en la vida profesional, está acorde a las exigencias del mundo actual donde casi todo es programable, es la base para la creación de software y aplicaciones y constituye una necesidad del país.

En cuanto al trabajo con software en el transcurso de la carrera, los estudiantes reflejaron que la frecuencia con la cual lo utilizan como apoyo en las diferentes asignaturas es el mostrado en la tabla 2.1:

Tabla 2.1. Frecuencia de utilización de software en la carrera

	Frecuencia de utilización de software		
Año	Siempre	Casi Siempre	Algunas Veces

2do	20%	45%	35%
3ro	35%	55%	10%
5to	35%	60%	5%

La mayoría de los estudiantes utilizan con gran frecuencia software de apoyo en asignaturas de la carrera, notándose un incremento de su empleo a medida que aumentan los años y el rigor en los estudios de las disciplinas fundamentales.

En el caso específico de la Electrónica los alumnos dieron su criterio sobre la suficiencia o no de los software que contribuyen al apoyo de la misma, obteniéndose como resultado que la mayoría de los software utilizados son de propietarios y que debido a la tendencia actual del país de utilizar cada vez más software libres, se debe de trabajar más en la facilitación y creación de estos últimos.

Tabla 2.2. Resultados acerca de la suficiencia de los software en las asignaturas de Electrónica.

	Suficiencia de los software	
Año	Suficiente	Insuficiente
2do	70%	30%
3ro	50%	50%
5to	85%	15%

Se realizó una entrevista a los jefes de disciplinas y a algunos profesores reconocidos por su experiencia en la Facultad de Ingeniería Eléctrica, con el objetivo de confirmar la necesidad de la utilización de software en las asignaturas de la carrera Ingeniería en Telecomunicaciones y Electrónica.

Los software dan la medida práctica de lo que se ve teóricamente, elemento imprescindible en las asignaturas de la disciplina Sistemas de Radiocomunicaciones, donde se dificulta la demostración real de los contenidos generalmente por la falta de equipamiento técnico. Entre

los principales software que se utilizan están: Radio Mobile para la simulación de enlaces de radiocomunicaciones, Mmana y CCT para el diseño de antenas y Matlab en el desarrollo de la Carta de Smith.

En las asignaturas de Microprocesadores los software son la principal fuente de aprendizaje, porque son la única forma para comprobar el correcto funcionamiento de un programa para un microprocesador cuando no se cuenta con el componente real. La mayoría de los software que se emplean en esta rama son de propietarios, provocando dificultades para su empleo en las universidades cubanas. Específicamente es utilizado el software libre 8051 que es un demo y presenta limitaciones, además el 8086 que no es libre.

En la disciplina de Teoría de las Comunicaciones los software son utilizados en experimentos y simulaciones para demostrar el correcto funcionamiento de los principales objetivos de las asignaturas como: el análisis de espectros de señales, las formas de codificación y decodificación y las modulaciones. Los software más utilizados son Matlab, Opnet y NC3, este último es el único libre.

Los software son utilizados como herramientas didácticas y sustitutos de equipos reales para su simulación en la disciplina de Sistemas de Telecomunicaciones. Para poder realizar un diseño real de una red lo más viable es simularla primero dando un criterio de sus funcionalidades, de manera que permita la rectificación de errores y minimizar el costo de esta. De esta forma los estudiantes tienen acceso al funcionamiento de los principales equipos y pueden crear propuestas de redes sin afectar una red real. Los software utilizados en esta disciplina son: Opnet, Packet Tracer, GNS3 y el software libre MiniNet.

En el campo de la electrónica el uso de simuladores se hace más imprescindible, la variedad de componentes que se puede añadir al circuito y la complejidad de este, obliga a hacer simulaciones y diseñar el circuito desde el propio ordenador para ajustar los requerimientos de los parámetros antes de programar el circuito en un chip programable o montarlo en una plaza de conexiones. En las asignaturas de la disciplina de Electrónica los software son un pilar fundamental para que los estudiantes asimilen con mayor facilidad los contenidos, tengan una mejor visión del funcionamiento de componentes electrónicos y pueden apreciar

los diferentes gráficos que contengan circuitos. Los principales software utilizados son los simuladores como: Proteus, Orcad y Multisim.

2.2.4 El caso de Amplifiers

Amplifiers es un programa que constituye un recurso que, siendo utilizado adecuadamente por los estudiantes, ayuda en el proceso de diseño de amplificadores, facilitando la comprensión y aplicación de criterios de diseño y la obtención de resultados óptimos con los que puedan trabajar y familiarizarse.

Para la realización del software “Amplifier”se utilizó el lenguaje de programación Python.

Para lograr que la aplicación ofrezca un circuito amplificador en una o varias etapas, dependiendo de la ganancia de voltaje deseada por el usuario, se utilizaron como bloques funcionales etapas en las configuraciones emisor común y colector común. El algoritmo implementado se muestra en la Figura 2.1, donde se pueden realizar varias iteraciones hasta definir la cantidad de etapas para facilitar la obtención de la ganancia especificada.

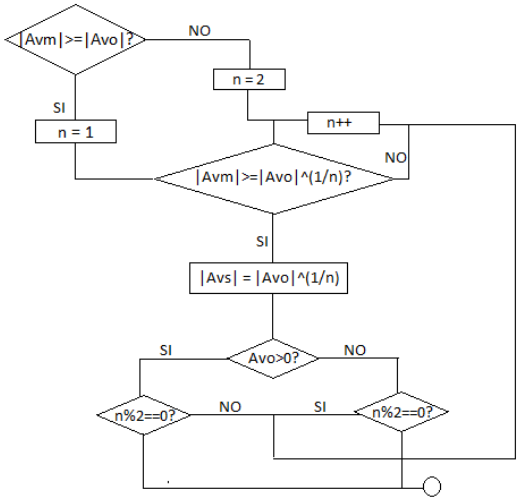


Figura 2.1. Algoritmo general de la aplicación.

La Figura 2.2 muestra la interfaz gráfica inicial de la aplicación, con la cual el usuario puede interactuar de forma rápida, especificando el dispositivo de amplificación, fuente de polarización y parámetros que se desean obtener o se especifican como requisitos para el

diseño. Para ello primeramente debe realizarse un análisis del problema que se está abordando.

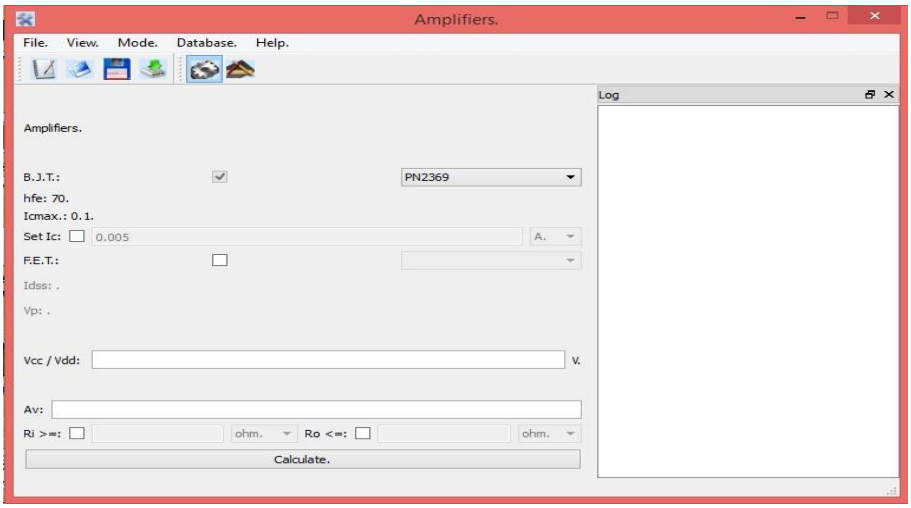


Figura 2.2. Ventana principal de la aplicación.

En la Figura 2.3 se observa la barra de herramientas de la aplicación que cuenta con cinco menús.

En el menú Archivo se encuentran las opciones de crear un nuevo archivo, guardar el archivo, abrir un archivo previamente guardado y exportar, este último se puede hacer en modo texto, para facilitar la obtención de los datos.

En el menú de vista se puede elegir la opción de mostrar u ocultar el log que tiene la aplicación.

En el menú Modo se puede elegir la opción profesional o la opción de aprendizaje. La primera muestra un resultado inmediato, mientras que la segunda ofrece resultados transitando por cada uno de los pasos de diseño que utiliza el programa para alcanzar los requisitos deseados. Precisamente, esta segunda opción, puede ser utilizada para facilitar el entrenamiento de los estudiantes en la apropiación del método de diseño y el aprendizaje de los principales criterios a tomar en cuenta.

En el menú Base de Datos se pueden encontrar la base de datos de los transistores que utiliza la aplicación. Además, se ofrece la posibilidad al usuario de introducir nuevos transistores de ser necesario.

En el menú de Ayuda se brinda información acerca de cómo utilizar la aplicación y su funcionamiento. Además, aparece la identificación completa de la aplicación y la información de sus creadores.

Debajo de este menú también se pueden encontrar diferentes accesos rápidos que pueden ser desplazados de lugar para comodidad del usuario.

Figura 2.3. Menú de herramientas de la aplicación.

La aplicación cuenta con un log el cual lleva un registro de los cambios y modificaciones realizados en la aplicación, de manera que, si el usuario desea volver a comprobar los cambios realizados desde el inicio de sesión del programa, pueda saber con exactitud lo que ha hecho.

La aplicación creada está diseñada a través de un software libre que permite la utilización de una Licencia General Pública, por sus siglas en inglés G.P.L., su idioma base es el inglés, para hacer más factible la compatibilidad a la hora de emigrar a otros idiomas.

Su algoritmo de programación está basado en los criterios de diseño, los esquemas de amplificadores y las ecuaciones de resolución descritas en la bibliografía básica de la asignatura Electrónica Analógica I [1].

La aplicación fue diseñada de manera tal que los valores de resistores obtenidos en el resultado son completamente teóricos, no se utiliza ninguna aproximación que muestre valores estándares de las mismas, lo que dificulta su montaje real, además no cuenta con una forma de visualizar el correcto funcionamiento de los transistores teniendo en cuenta la ubicación de su punto de operación y la región en la cual el transistor se encuentra trabajando.

Ninguna de las aplicaciones mencionadas en los subepígrafes 2.2.1 y 2.2.2 permite diseñar a partir de valores teóricos establecidos circuitos amplificadores con la utilización de transistores de efecto de campo y transistores bipolares, sino que se parte de un diseño ya establecido por el usuario para su posterior simulación. En muchos casos se necesita poseer altos conocimientos en herramientas de simulación y de electrónica, además de un limitando enfoque hacia el proceso de enseñanza-aprendizaje de muchos de estos simuladores.

La aplicación que más se acerca a esta función es ElectroDroid, ya que dentro de su apartado de calculadoras tiene la de amplificadores operacionales (AO). Esta calculadora permite obtener los valores de los resistores de los AO en sus diferentes configuraciones, pero no se centra en las diferentes etapas presentes en amplificadores de pequeña señal.

2.3 Herramientas de programación

Un lenguaje de programación es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo por máquinas como las computadoras.

Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana [35].

2.3.1 Tipos de lenguajes de programación

Básicamente hay tres tipos de lenguajes de programación de computadora.

- Lenguajes de programación de bajo nivel.
- Lenguajes de programación de alto nivel.
- Lenguajes de programación de nivel medio.

Lenguajes de programación de bajo nivel.

Estos son lenguajes de programación dependientes de la máquina como Binary (código de máquina) y lenguaje Assembly.

Como la computadora solo entiende el lenguaje binario, que son instrucciones en forma de 0 y 1 (señales, que pueden ser altas o bajas), estos lenguajes de programación son la mejor manera de dar señales (instrucciones binarias) directamente a la computadora.

Machine Code (Binary Language) no necesita ningún intérprete o compilador para convertir el lenguaje en cualquier forma, porque la computadora entiende estas señales directamente.

Sin embargo, el lenguaje de ensamblaje debe convertirse en un código binario equivalente, para que la computadora pueda entender las instrucciones escritas en ensamblado. El ensamblador se usa para convertir un código ensamblador a su código binario equivalente.

Los códigos escritos en ese tipo de idiomas son difíciles de escribir, leer, editar y comprender; los programas no son portátiles para ningún otro sistema informático.

Los programas de lenguaje de programación de bajo nivel son más rápidos que los programas de lenguaje de programación de alto nivel, ya que tienen menos palabras clave, símbolos y ninguna necesidad (menos necesidad) de convertir en código de máquina.

Lenguajes de programación de alto nivel.

Estos son los lenguajes de programación independientes de la máquina, que son fáciles de escribir, leer, editar y comprender.

Los lenguajes como Java, .Net, Pascal, COBOL, C ++, C, C # y otros (que se utilizan para desarrollar aplicaciones de usuario final), se incluyen en la categoría de lenguaje de programación de alto nivel. Estos tienen palabras clave, funciones y bibliotecas de clase especiales al usarlas.

La computadora no entiende el programa escrito en dichos idiomas directamente, ya como se menciona anteriormente la computadora solo entiende código binario. Por lo tanto, aquí los traductores de programación deben convertir un programa de alto nivel a su código máquina equivalente.

Traductores de programación como compiladores e intérpretes son los software del sistema que convierten un programa escrito en lenguajes de programación específicos en su código máquina equivalente.

Algunas de las características de los lenguajes de programación de alto nivel son:

Los programas están escritos en lenguajes de programación de alto nivel y son independientes, lo que significa que un programa escrito en un sistema puede ejecutarse en otro sistema.

Fácil de entender: dado que estos lenguajes de programación tienen palabras clave, funciones y bibliotecas de clase (que son similares a las palabras en inglés), podemos entender fácilmente el significado de un término particular relacionado con ese lenguaje de programación.

Fácil de codificar, leer y editar: los programas escritos en lenguajes de programación de alto nivel son fáciles de codificar, leer y editar. Incluso podemos editar programas escritos por otros programadores fácilmente teniendo poco conocimiento de ese lenguaje de programación.

A pesar de que los programas de idiomas de alto nivel son más lentos que los programas de idiomas de bajo nivel; aun así, son lenguajes de programación populares para desarrollar aplicaciones finales de usuario.

Lenguaje de programación de nivel medio.

Dado que no existe tal categoría de lenguajes de programación de computadoras, los lenguajes de programación que tienen características de bajo nivel y nivel pertenecen a esta categoría [36].

2.3.2 Lenguajes de programación más utilizados

Los 5 lenguajes de programación más populares de la actualidad son: Java, C, C++, Python y C#. Según el Índice TIOBE en su actualización de junio 2017.

El Índice TIOBE se elabora a partir de diversas variables, como, por ejemplo, número de ingenieros cualificados en determinado lenguaje, las búsquedas que hacen los usuarios a través de los buscadores solicitando información de los distintos lenguajes de programación, la demanda de cursos o los lenguajes que están siendo más utilizados. El índice no dice cuál es el mejor, o en qué lenguaje de programación se escribió la mayor cantidad de líneas de

código. Más bien sirve para que un programador pueda determinar si sus conocimientos en un determinado lenguaje han quedado obsoletos, o si por el contrario sus conocimientos están vigentes. También puede ser de utilidad si tienes que tomar una decisión estratégica respecto a qué lenguaje debes utilizar para la construcción de determinado software.

1. Java

Reconocido por su legibilidad y simplicidad, Java es uno de los lenguajes de programación más adoptados: más 9 millones de desarrolladores lo usan y está presente en 7 mil millones de dispositivos en todo el mundo. Desde 2001 se mantiene en las primeras posiciones, llegando al número puesto número 2 como la más baja de todas en marzo de 2015. Su enorme popularidad se debe a su poder de permanencia, cuestión que asegura el funcionamiento a largo plazo de las aplicaciones que lo utilizan.

Asimismo, cabe destacar que el manejo de Java está incluido dentro de las 20 habilidades más valoradas por los empleadores en 2016, según un estudio elaborado por LinkedIn.

2. C

Creado entre 1969 y 1972 en los Laboratorios Bell, es uno de los más utilizados en el mundo. Si bien es ejecutado en la mayoría de los sistemas operativos, es de propósito general, con lo cual es muy flexible. Es muy popular para el desarrollo de aplicaciones de escritorio, como el conocido editor gráfico GIMP.

3. C++

Conocido por el nombre “C Plus Plus”, este lenguaje de programación orientado a objetos surge como una continuación y ampliación del C. Hay una gran cantidad de programas escritos en C++, como por ejemplo los paquetes de Adobe.

4. Python

Un lenguaje de programación multiplataforma y multiparadigma, que también es de propósito general. Esto significa que soporta la orientación a objetos, la programación imperativa y funcional. Su simpleza, legibilidad y similitud con el idioma inglés lo convierten en un gran lenguaje ideal para principiantes.

5. C#

También llamado “C Sharp”, este lenguaje de programación orientado a objetos fue desarrollado en el año 2000 por Microsoft para ser empleado en una amplia gama de aplicaciones empresariales ejecutadas en el framework .NET. C Sharp es una evolución del C y C++ que se destaca por su sencillez y modernidad.

6. Visual Basic. NET

Pasó del número 9 en junio de 2016 al sexto lugar en 2017. Es utilizado por una gran cantidad de personas a lo largo del mundo que no cuentan con conocimientos profundos como desarrolladores, quienes encuentran en visual basic, además de una sintaxis sencilla, la posibilidad de automatizar sus propios procesos y crear sus propias aplicaciones web. Es visto como uno de los lenguajes más amigables para los que recién comienzan, sobre todo a comparación de C#.

7. JavaScript

JavaScript es un lenguaje de programación que puede ser utilizado para crear programas que luego son acoplados a una página web o dentro de programas más grandes. Sirve para crear efectos y realizar acciones interactivas. Se puede ver el funcionando este lenguaje en servicios como el chat, calculadoras, buscadores de información y un sin fin de utilidades más.

8. PHP

Creado en 1994 por el programador canadiense Rasmus Lerdorf, nunca pretendió ser un lenguaje de programación, sino que fue creado con la intención de contar con un conjunto de herramientas para el mantenimiento de las páginas web. Es de fácil acceso para nuevos programadores y a su vez ofrece a los más experimentados muchas posibilidades.

9. Perl

Creado Larry Wall, Pearl es una sigla que significa Practical Extraction and Report Language. Es un lenguaje de propósito general que sirve prácticamente para todo, como

puede ser la generación y tratamiento de ficheros, para analizar datos y textos, y muchas otras cosas más. Si bien hay proyectos complejos y completos escritos en Pearl, son los menos.

10. Assembly language (ASL)

Assembly language (lenguaje ensamblador) saltó de la posición número 12 en 2016 al número 10 en 2017. Se trata de un lenguaje de programación de bajo nivel utilizado para interactuar con hardware informático. Utiliza comandos estructurados, en sustitución de los números, permitiendo a las personas tener una mayor legibilidad de los códigos. Si bien es más sencillo de leer que el código binario se trata de un lenguaje difícil que muchas veces es sustituido por uno superior, como por ejemplo C [37].

2.3.3. Lenguaje de programación "Python"

El lenguaje Python surgió a principios de los 90 e inicialmente fue desarrollado por Guido Van Rossum, un ingeniero holandés que trabajaba en ese momento en el CWI de Ámsterdam, el Centro de Investigación de Ciencias de la Computación holandés.

Python surgió como un hobby para Guido y su nombre, Python, fue tomado del grupo cómico británico Monty Python, del que Guido era un gran fan. Desde sus comienzos, Python nació como un proyecto de software libre y posiblemente deba parte de su éxito a la decisión de hacerlo código abierto.

Actualmente, la evolución del lenguaje Python es gestionada por la Python Software Foundation, una sociedad sin ánimo de lucro dedicada a dar difusión al lenguaje y apoyar su evolución. Guido sigue totalmente involucrado en el desarrollo y en la toma de decisiones de diseño.

Python está licenciado bajo licencia PSFL, derivada de BSD y compatible con GPL. Muchas empresas y organizaciones, como Google, Microsoft o Red Hat, hacen un gran uso de Python y tienen influencia en su evolución, pero ninguna ejerce un control sobre el mismo. Esto diferencia a Python de otros lenguajes.

Python es open source, cualquiera puede contribuir a su desarrollo y divulgación. Además, no es necesario pagar ninguna licencia para distribuir software desarrollado con este lenguaje. Hasta su intérprete se distribuye de forma gratuita para diferentes plataformas.

Python es un lenguaje de programación de alto nivel, interpretado y multipropósito. En los últimos años su utilización ha ido constantemente creciendo y en la actualidad es uno de los lenguajes de programación más empleados para el desarrollo de software.

Python puede ser utilizado en diversas plataformas y sistemas operativos, entre los que podemos destacar los más populares, cómo Windows, Mac OS X y Linux. Pero, además, Python también puede funcionar en smartphones, Nokia desarrolló un intérprete de este lenguaje para su sistema operativo Symbian [38].

Características de Python

Python tiene una serie de características que lo hacen muy particular y que le aportan muchas ventajas y están en la raíz de su uso tan extendido.

Python es un lenguaje multiparadigma, esto significa que combina propiedades de diferentes paradigmas de programación. Principalmente es un lenguaje orientado a objetos, todo en Python es un objeto, pero también incorpora aspectos de la programación imperativa, funcional, procedural y reflexiva.

Una de las características más reseñables de Python es que es un lenguaje interpretado, esto significa que no se compila a diferencia de otros lenguajes como Java o C/C++, sino que es interpretado en tiempo de ejecución [38]. Python tiene muchas de las características de los lenguajes compilados, por lo que se podría decir que es semi interpretado. En Python, como en Java y muchos otros lenguajes, el código fuente se traduce a un seudo código máquina intermedio llamado bytecode la primera vez que se ejecuta, generando archivos .pyc o .pyo, que son los que se ejecutarán en sucesivas ocasiones [1].

Además, es de tipado dinámico, aunque opcionalmente desde la versión 3.5 se puede hacer uso de tipado estático [38]. El tipado dinámico se refiere a que no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en

tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.

No se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario convertir de forma explícita dicha variable al nuevo tipo previamente [1].

Python es cross plataforma, es decir, se puede ejecutar en diferentes sistemas operativos como Windows o Linux simplemente usando el intérprete correspondiente. Además Python ofrece dialectos como el Jython, que se utiliza para escribir en Java [39], [38].

Entre las principales razones para elegir Python, se argumenta que sus principales características lo convierten en un lenguaje muy productivo. Se trata de un lenguaje potente, flexible y con una sintaxis clara y concisa. Además, no requiere dedicar tiempo a su compilación debido a que es interpretado [38].

Empresas de alto prestigio utilizan Python para programar todo tipo de aplicaciones y servicios.

Python se encuentra en multitud de aplicaciones y servicios que se usa habitualmente. Ostenta una gran lista de usuarios de gran calibre como Google, YouTube o Facebook, los cuales utilizan este lenguaje de programación. Poco a poco Python ha ganado territorio y, entre los entendidos, se ha convertido en uno de los lenguajes más solicitados y, sobretodo, más esenciales del momento. Esto indica que programar en Python es una opción muy viable y efectiva que hay ahora mismo en el mercado [39].

Se puede decir que Python es un lenguaje maduro, con una gran base de desarrolladores, documentación y proyectos en producción [38].

2.4 Conclusiones del Capítulo

En este capítulo se expusieron las ventajas del lenguaje de programación Python. Se realizó un estudio de los principales simuladores utilizados y su importancia en la carrera Ingeniería en Telecomunicaciones y Electrónica. Se mostró la aplicación Amplifiers como un software de apoyo en el estudio de la Electrónica. Al inicio del capítulo se realizó un estudio en el cual se dio a conocer la necesidad de la programación en las principales disciplinas de la carrera.

CAPÍTULO 3. Versión mejorada del software Amplifiers

En el presente capítulo se describirá el rediseño del software Amplifiers y su forma de empleo dentro de la asignatura de Electrónica Analógica I. Se comprobarán los resultados que brinda la aplicación mediante problemas propuestos simulados en el Multisim y su montaje real.

3.1 Rediseño del software Amplifiers

Como se observó en capítulos anteriores el software Amplifiers contaba con varias deficiencias, por lo que se decidió realizar una nueva restructuración del mismo. Para esto se diseñó un nuevo algoritmo que permite obtener valores comerciales de resistores a partir de una base de datos incorporada a la aplicación donde se encuentran las tablas de tolerancia de algunos estándares reconocidos [40].

Esta base de datos cuenta con los valores reales de los estándares E12, E24, E48 y E96 como se muestra en la figura 3.1.

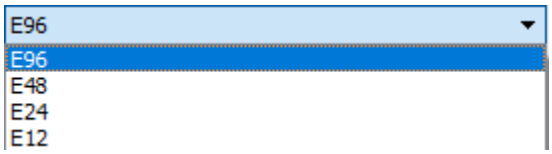


Figura 3.1. Ventana desplegable para la selección de los estándares de resistores.

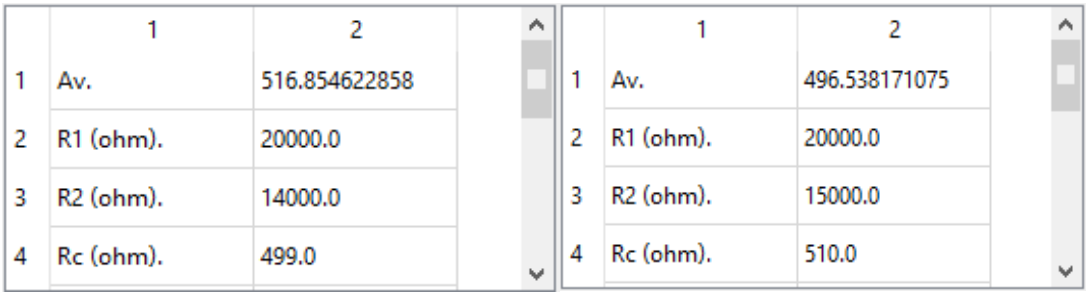
Al seleccionar uno de estos, el software ejecuta la línea de código que se muestra en el anexo 2, reevaluando los valores de las resistencias teniendo en cuenta el que más se aproxime al teórico y a su vez se calcula el nuevo valor de ganancia de voltaje con los parámetros reales.

En un ejemplo donde se busca un diseño con una ganancia de voltaje igual a 500 los valores teóricos de resistencias obtenidos fueron los mostrados en la figura 3.2.

	1	2
1	R1 (ohm).	19959.6778655
2	R2 (ohm).	14062.5003143
3	Rc (ohm).	500.000011176
4	Re (ohm).	496.987962916

Figura 3.2. Valores teóricos de resistencias.

Para el mismo diseño en la figura 3.3 se observa la sección de la interfaz del software dedicada a esta nueva ganancia y valores de resistencias para el estándar E96 a la izquierda y E24 a la derecha.



	1	2
1	Av.	516.854622858
2	R1 (ohm).	20000.0
3	R2 (ohm).	14000.0
4	Rc (ohm).	499.0

	1	2
1	Av.	496.538171075
2	R1 (ohm).	20000.0
3	R2 (ohm).	15000.0
4	Rc (ohm).	510.0

Figura 3.3. Valores estándares de resistencias y nueva ganancia.

Como se observa en la figura 3.3 mientras mayor es la tolerancia del estándar más se alejan los valores teóricos de los comerciales.

De las imágenes anteriores se puede expresar que la aproximación realizada por Amplifiers es muy cercana entre los valores teóricos y los comerciales de resistencias, al igual que en el caso de las ganancias de voltaje, siempre dependiendo del estándar escogido.

Además del reajuste de estos parámetros se incluyó una gráfica que muestra las curvas de salida de los transistores BJT y su punto de operación (Q), en la que se puede constatar que los valores obtenidos cumplen con los requerimientos de diseño y que los transistores trabajan en la región activa, como se muestra en la figura 3.4.

En el anexo 3 se muestra el código utilizado para la confección de esta parte del rediseño de la aplicación.

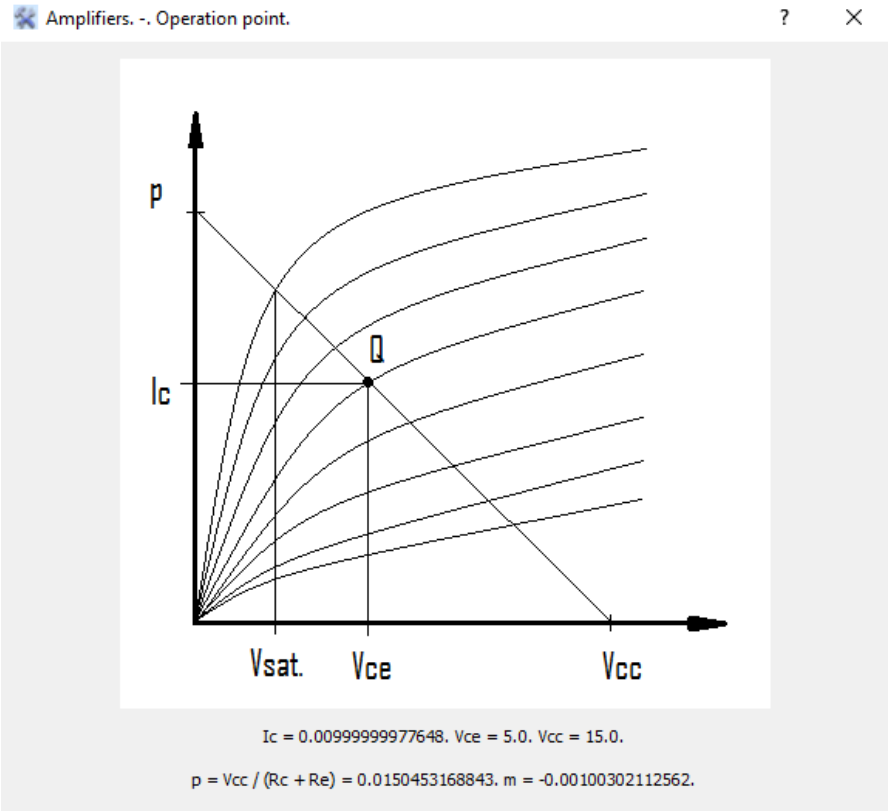


Figura 3.4. Característica transferencial de salida de un BJT.

3.2 Como utilizar el software Amplifiers

En la figura 3.5 se muestra una parte de la interfaz de la aplicación donde se puede seleccionar un transistor de efecto de campo o un transistor bipolar en dependencia con el cual el usuario desea trabajar. La aplicación cuenta con una base de datos que brinda varios modelos específicos de transistores con sus características generales, como son hfe y la Ic máxima para el caso de BJT y la Idss y Vp para el caso de FET.

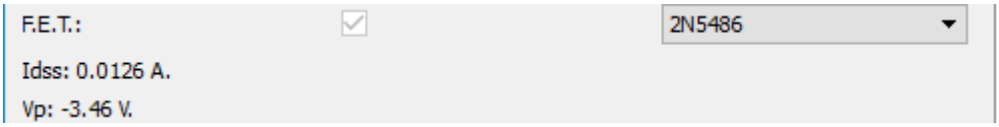


Figura 3.5. Selección de los parámetros de los transistores.

Aparece una Ic por defecto que puede ser cambiada por la deseada, siempre y cuando no supere el valor de corriente máximo definido en el modelo del transistor como se observa en la figura 3.6. En el caso de la Idss solo varía cuando se cambia el transistor debido a que es un cálculo que realiza la propia aplicación, por lo tanto, el usuario no lo puede variar.

Figura 3.6. Selección de Ic por el usuario.

En la Figura 3.7 se muestra otra parte de la interfaz de la aplicación donde el usuario debe introducir el voltaje de polarización (VCC o VDD según el caso), los valores de voltajes establecidos en la aplicación son los utilizados con generalidad en la asignatura de Electrónica Analógica I. La ganancia de voltaje se puede introducir con el valor deseado que se está trabajando. Además, se tiene la opción de obtener un diseño acotado a una resistencia de salida y una resistencia de entrada con valores que pueden ser especificados. Aparece también el botón calcular el cual se utiliza para la obtención de los resultados.

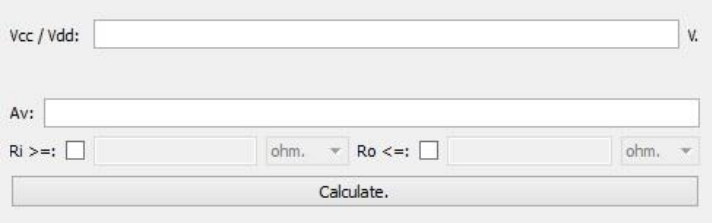


Figura 3.7. Parte inferior de la interfaz de la aplicación.

Es importante mencionar que si los parámetros introducidos por el usuario no se encuentran dentro de los rangos de diseño de este tipo de amplificadores la aplicación mostrará una advertencia que informa sobre el error que se ha cometido [1].

La figura 3.8 muestra el resultado que la aplicación brinda, en esta se muestra de forma gráfica las diferentes etapas calculadas, así como las configuraciones amplificadoras necesarias para lograr los resultados esperados. Haciendo clic en el botón siguientes, el

usuario podrá ver cada una de las etapas las cuales deben ser colocadas en cascadas para obtener el diseño completo. A la derecha se encuentras los valores de los componentes utilizados, tanto los valores de resistencias teóricos como los reales, además de que se recalcula el valor de la ganancia de voltaje con los parámetros estándares.

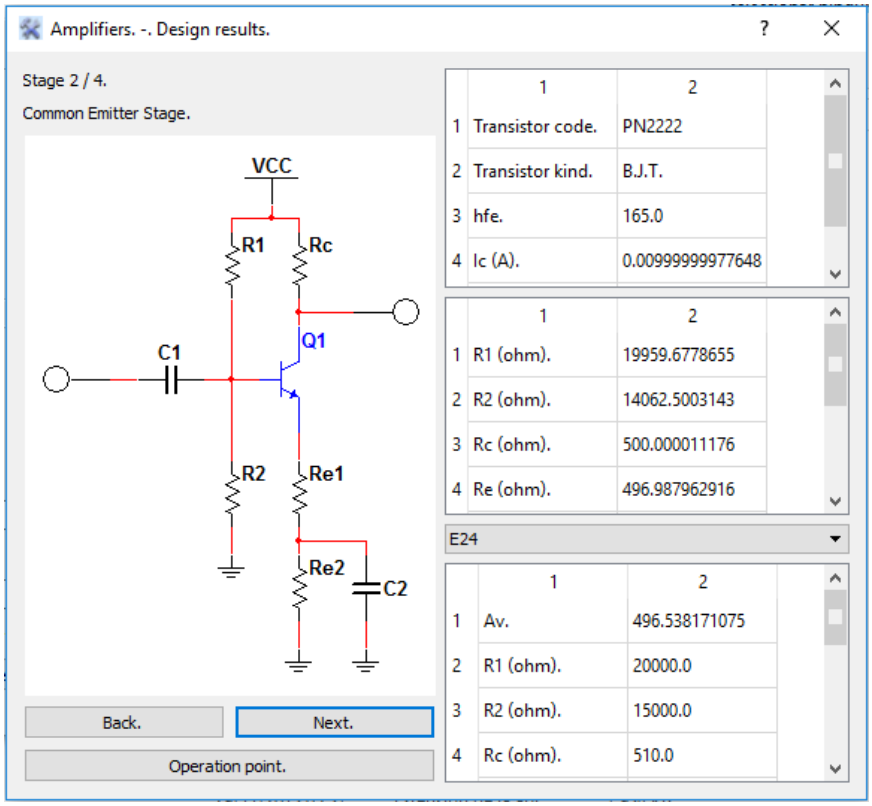


Figura 3.8. Resultado que brinda la aplicación.

En la ventana que muestra los resultados aparecen tres botones: Back, Next y Operation Point. Los dos primeros se utilizan para observar las etapas que presenta el diseño del amplificador y el último muestra la característica transferencial de salida del transistor con el Punto Q.

La enseñanza de la Electrónica se desarrolla desde tres ángulos diferentes: Analítico, Simulado y Experimental. De lo anterior se concluye el importante papel que juega la utilización de la computadora en la enseñanza de esta disciplina, tanto para la simulación como en la instrumentación virtual.

En el nuevo plan de estudios E se disminuye el número de horas clases dedicadas a la disciplina con respecto al plan anterior, de manera que el estudiante debe jugar un papel determinante mediante su trabajo individual para apropiarse de los conocimientos.

Entre los objetivos instructivos de la Electrónica se encuentra el de diseñar los esquemas lógicos y eléctricos de circuitos y sistemas electrónicos básicos, analógicos y digitales y el de explotar programas para el estudio del comportamiento y diseño de circuitos electrónicos [41].

Amplifiers puede ser utilizado como un medio de apoyo al estudio de amplificadores de pequeña señal, complementando los laboratorios reales propuestos y el seminario dos en la asignatura Electrónica Analógica I. Los estudiantes pueden comprobar los resultados obtenidos en clases y compararlos con los ofrecidos por la aplicación.

Esta aplicación no es un sustituto de lo que el alumno debe realizar con los conocimientos adquiridos en el estudio de la asignatura, sino otra fuente de apoyo para la consolidación de los mismos.

3.3 Problema resuelto utilizando Amplifiers

Para comprobar el correcto funcionamiento de Amplifiers utilizando los nuevos valores resistivos obtenidos se decidió realizar un problema sencillo con características similares a los propuestos en la Electrónica Analógica I.

Los parámetros escogidos fueron: ganancia de voltaje mayor o igual a 100, resistencia de entrada mayor o igual a 20kΩ, Vcc de 15V, una fuente de entrada de 1mVpk a 1KHz y se seleccionó un transistor NPN de la serie BC548.

Este transistor no estaba dentro de la base de datos de la aplicación, por lo que se incluyó manualmente, constatando que cualquier usuario pude incluir sus propios modelos de transistor según las exigencias del ejercicio que se proponga realizar.

Después de pasarle los parámetros a la aplicación se obtuvieron los resultados que se muestran en las figuras 3.9, 3.10 y 3.11.

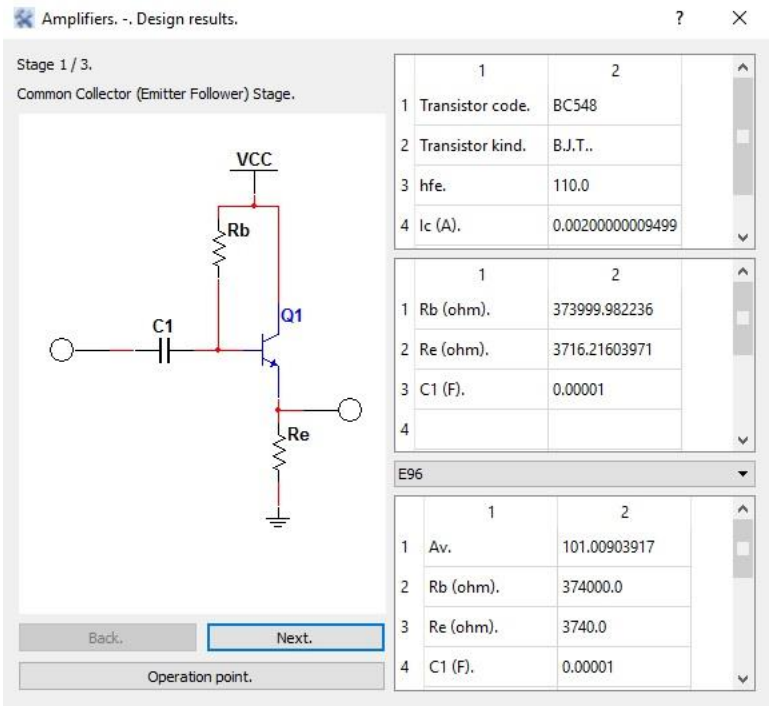


Figura 3.9. Primera etapa del amplificador.

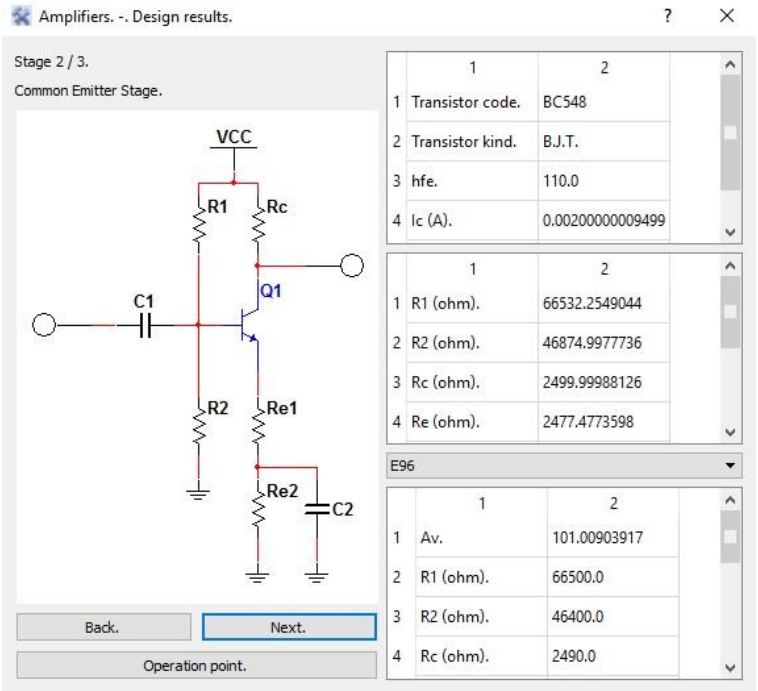


Figura 3.10. Estructura de la segunda y tercera etapa del amplificador.

	1	2
6	Re1 (ohm).	187.0
7	Re2 (ohm).	2260.0
8	C1 (F).	0.00001
9	C2 (F).	0.00001

	1	2
6	Re1 (ohm).	150.0
7	Re2 (ohm).	2320.0
8	C1 (F).	0.00001
9	C2 (F).	0.00001

Figura 3.11. Distribución de Re en las etapas 2 (a la izquierda) y 3 (a la derecha).

Para la comprobación de los resultados se realizó la simulación de la cascada amplificadora en el Multisim. En la figura 3.12 se observa el circuito montado.

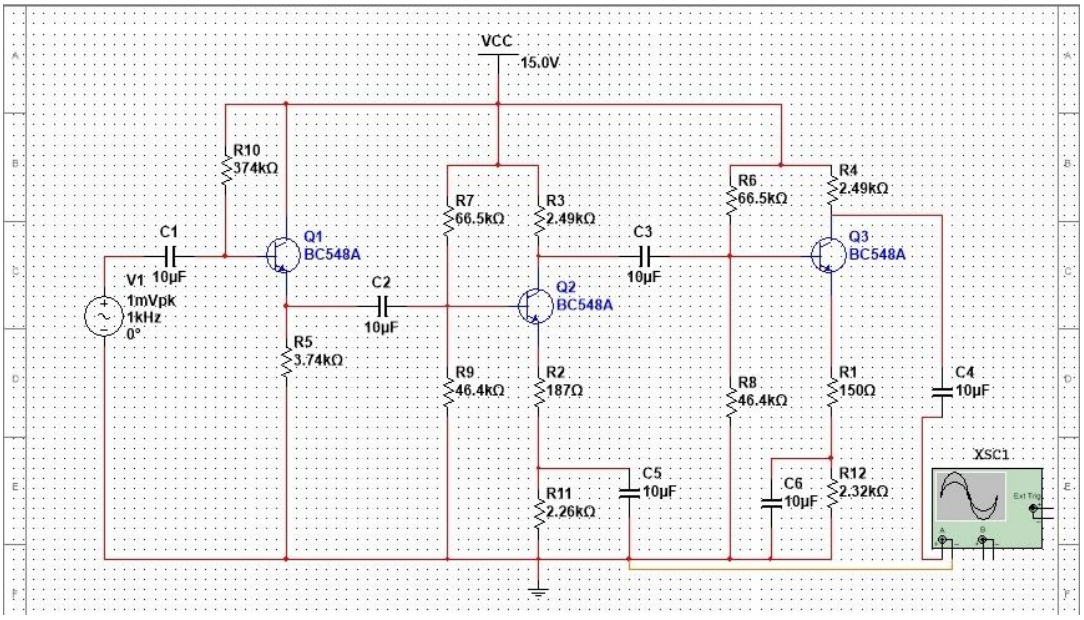


Figura 3.12. Circuito montado en el Multisim.

Para una entrada de 1mVpk y una ganancia de voltaje igual a 100, se espera a la salida aproximadamente 100mVpk, por lo que se colocó un osciloscopio en el cual se pude ver los valores de salida. En la figura 3.13 se observa la señal de salida.

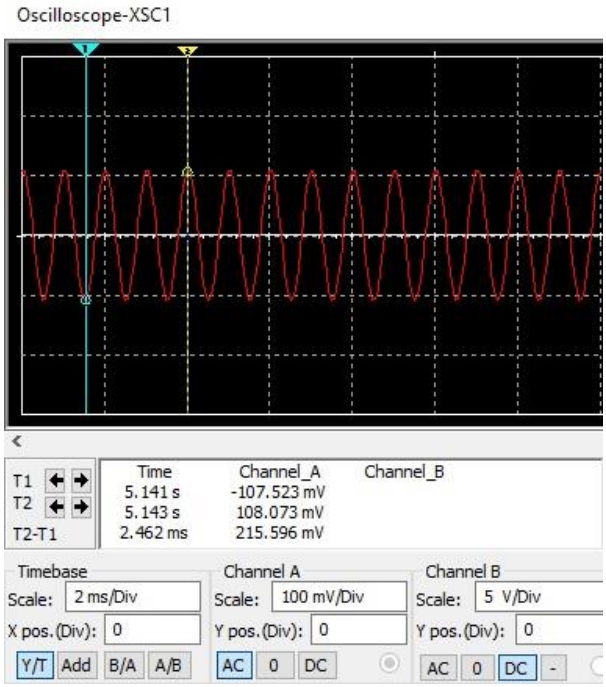
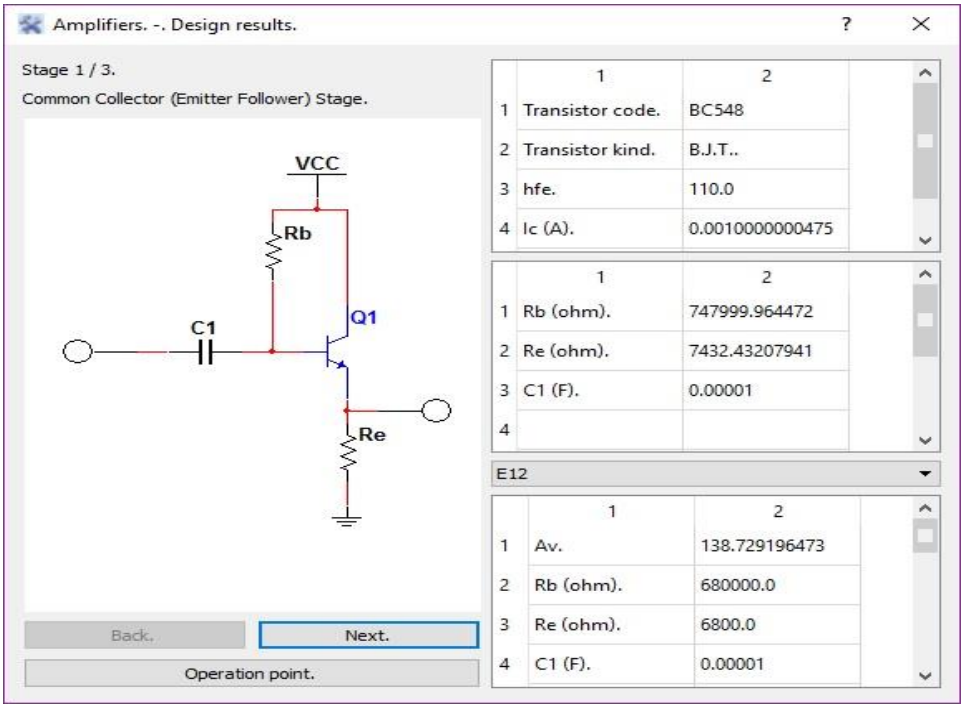


Figura 3.13. Señal de salida del amplificador.

Como se puede ver en la figura anterior la salida oscila entre los 108mVpk y los 107mVpk, lo que demuestra que el diseño realizado por la aplicación es correcto y que los valores reales de resistencias solo varían ligeramente el voltaje final.

Para comprobar resultados con otro estándar se realizó otro problema utilizando el E12, los parámetros escogidos fueron: Vcc =15V, Av =150, Ri mayor o igual que 20kΩ y el transistor BC548. Los resultados obtenidos por la aplicación se muestran en las figuras 3.14, 3.15 y 3.16.

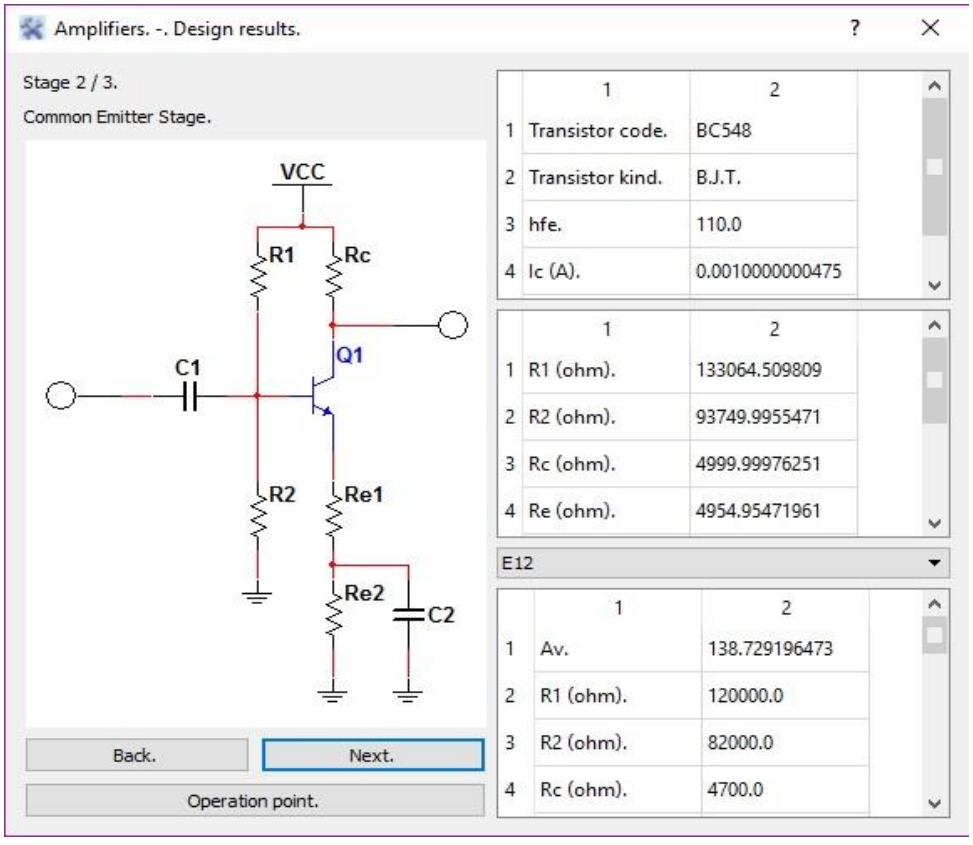


	1	2
1	Transistor code.	BC548
2	Transistor kind.	B.J.T..
3	hfe.	110.0
4	Ic (A).	0.0010000000475

	1	2
1	Rb (ohm).	747999.964472
2	Re (ohm).	7432.43207941
3	C1 (F).	0.00001
4		

	1	2
1	Av.	138.729196473
2	Rb (ohm).	680000.0
3	Re (ohm).	6800.0
4	C1 (F).	0.00001

Figura 3.14. Configuración de la primera etapa amplificadora.



	1	2
1	Transistor code.	BC548
2	Transistor kind.	B.J.T.
3	hfe.	110.0
4	Ic (A).	0.0010000000475

	1	2
1	R1 (ohm).	133064.509809
2	R2 (ohm).	93749.9955471
3	Rc (ohm).	4999.99976251
4	Re (ohm).	4954.95471961

	1	2
1	Av.	138.729196473
2	R1 (ohm).	120000.0
3	R2 (ohm).	82000.0
4	Rc (ohm).	4700.0

Figura 3.15. Configuración de las etapas 2 y 3 amplificadoras.

E12

	1	2
5	Re (ohm).	4700.0
6	Re1 (ohm).	270.0
7	Re2 (ohm).	4700.0
8	C1 (F).	0.00001

E12

	1	2
5	Re (ohm).	4700.0
6	Re1 (ohm).	180.0
7	Re2 (ohm).	4700.0
8	C1 (F).	0.00001

Figura 3.16. Distribución de Re de la etapa 2 (izquierda) y etapa 3 (derecha).

Como se puede ver los valores reales de los resistores están más alejados de los obtenidos por la aplicación debido a la tolerancia del estándar escogido.

El montaje del circuito en el Multisim se puede observar en la figura 3.17.

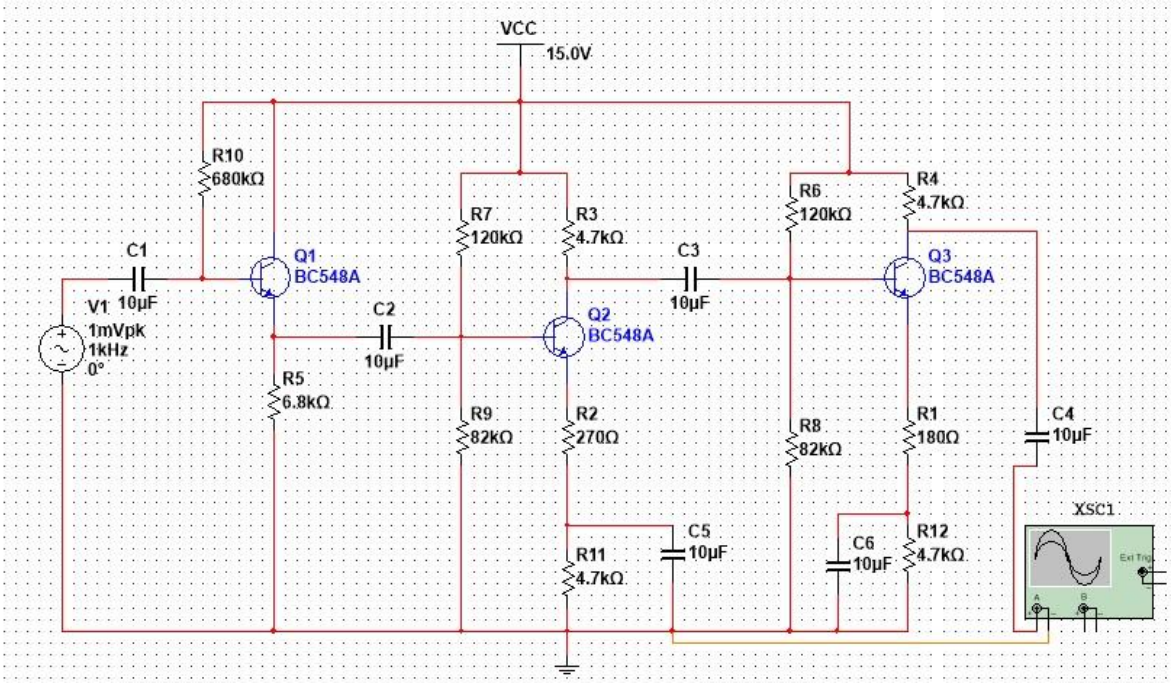


Figura 3.17. Circuito montado en el Multisim.

La forma de onda obtenida por el osciloscopio para este ejercicio se muestra en la figura 3.18.

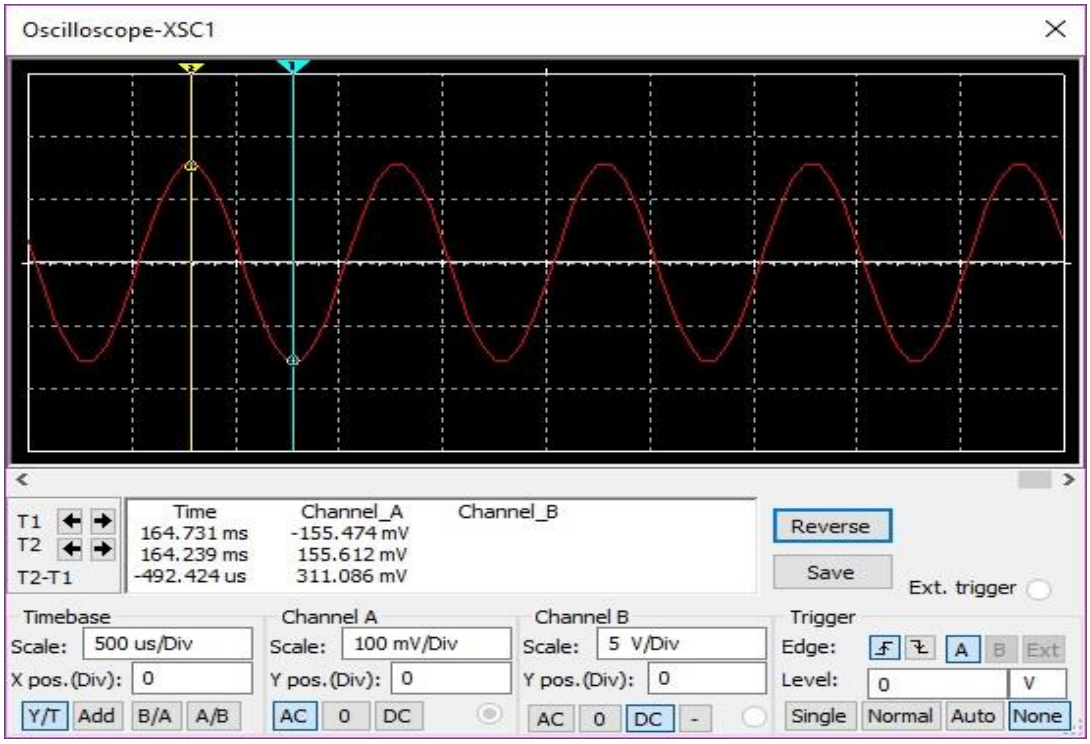


Figura 3.18. Señal de salida obtenida por el osciloscopio.

Como se puede observar en la figura anterior la señal de salida oscila entre los 155mV y -155mV, aumentando el valor esperado debido al estándar escogido que dista ligeramente de los valores ofrecidos por la aplicación.

Después de haber simulado el circuito se pasó a realizar su montaje real como se muestra en la figura 3.19.

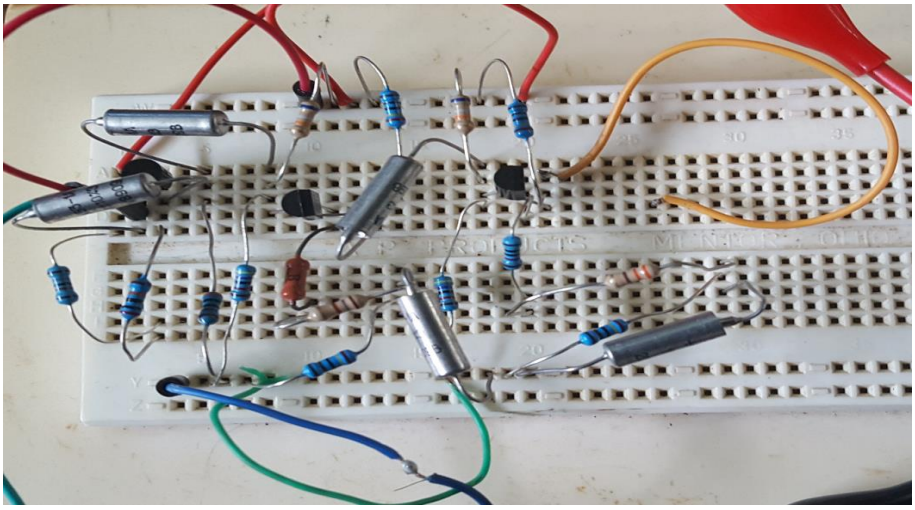


Figura 3.19. Circuito amplificador montado de forma real.

Los resultados obtenidos se analizaron mediante la utilización del osciloscopio disponible en el laboratorio. Para generar la señal de entrada se utilizó una señal de 1mVpp generada con un frecuencímetro y para el Vcc se generaron los 15 V con una fuente de voltaje. En la figura 3.20 se pueden ver los resultados mostrados por el osciloscopio.

Figura 3.20. Resultados mostrados por el osciloscopio en el montaje real.

3.4 Conclusiones del Capítulo

En el capítulo se obtuvo como resultado una versión mejorada del software Amplifiers. Se comprobó el correcto funcionamiento de la nueva versión y se constató que puede ser utilizada como un medio de apoyo para la comprobación de resultados de los problemas propuestos en la asignatura Electrónica Analógica I.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

En el presente trabajo se realizó una versión mejorada de la aplicación Amplifiers, para su utilización como herramienta de apoyo al proceso de enseñanza-aprendizaje de la asignatura de Electrónica Analógica I. Durante su realización se arribó a las siguientes conclusiones:

- Las TIC contribuyen al desarrollo de recursos que favorecen el proceso de enseñanza-aprendizaje de la Electrónica estas permiten diseñar y/o rediseñar software educativos en carreras técnicas que apoyan el aprendizaje de asignaturas de las mismas, fomentando el interés de los estudiantes para su estudio.
- En la carrera Ingeniería en Telecomunicaciones y Electrónica de la Universidad Central "Marta Abreu" de Las Villas, la programación y la utilización de software constituyen un pilar fundamental en la formación del futuro profesional.
- Se identificaron y describieron los problemas fundamentales de la primera versión de la aplicación Amplifiers: la inexistencia de un algoritmo que brindara los valores estándares de las resistencias obtenidas como resultado de la aplicación y la falta de un recurso gráfico para mostrar el punto de operación de los transistores utilizados, de manera que se visualice la ubicación en la región de trabajo.
- Se utilizaron para la versión mejorada de Amplifiers tablas de tolerancias que corresponden a los estándares de resistores E96, E48, E24 y E12.
- La introducción del cálculo la ganancia de voltaje de la cascada con la utilización de valores estándares de resistores demuestra que, cuanto mayor sea la tolerancia del estándar de resistencia, más se alejará el valor de ganancia de voltaje real del teórico requerido.
- Los valores calculados en la realización de la gráfica para el cálculo del punto de operación cumplen con los criterios de diseño para el correcto funcionamiento de un transistor.
- Los resultados obtenidos por la simulación y el montaje real corresponden con los esperados, acercándose en gran medida al requerido por el usuario.

Recomendaciones

Se considera que las siguientes recomendaciones pueden ser de utilidad para enriquecer el estudio realizado y los resultados obtenidos:

- Utilizar la aplicación como un complemento para la asignatura Electrónica Analógica I.
- Incorporar a la aplicación otros contenidos de la asignatura como el cálculo de las componentes de frecuencia.
- Seguir desarrollando software con la utilización de lenguajes de programación libres para el apoyo de asignaturas de carreras técnicas.

REFERENCIAS BIBLIOGRÁFICAS

[1] O. E. Abreu Cruz, A. Estevez Salgado, R. González Cruz, L. S. Linares Pérez, y D. E. Ruiz Guirola, «Aplicación de la programación como herramienta de apoyo al diseño de amplificadores». 2016.

[2] M. Gisbert Cervera, J. M. Cela-Ranilla, y S. Isus Barado, «Las simulaciones en entornos TIC como herramienta para la formación en competencias transversales de los estudiantes universitarios.», *Teoría Educ. Educ. Cult. En Soc. Inf.*, vol. 11, n.º 1, 2010.

[3] R. Mayta Huatuco y W. León Velásquez, «El uso de las TIC en la enseñanza profesional», *Ind. Data*, vol. 12, n.º 2, 2009.

[4] T. Guzmán Flores, M. T. García Ramírez, C. Espuny Vidal, y R. Chaparro Sánchez, «Formación docente para la integración de las TIC en la práctica educativa», *Apertura*, vol. 3, n.º 1, 2011.

[5] K. Rodríguez Salas, «Las TIC como apoyo al proceso de enseñanza-aprendizaje en Bibliotecología, IIBI, UNAM, México», 13-feb-2018. [En línea]. Disponible en: http://iibi.unam.mx/publicaciones/280/tic-educacion-bibliotecologica-las-TICs-Karla-Rodriguez-Salas.html. [Accedido: 13-feb-2018].

[6] M. Delgado, X. Arrieta, y V. Riveros, «Uso de las TIC en educación, una propuesta para su optimización», *Omnia*, vol. 15, n.º 3, 2009.

[7] I. Fernández Silva, «El uso de las tic en la carrera Lic. En educación, especialidad Logopedia: consideraciones para el diseño del proceso de enseñanza-aprendizaje.», *9/7/2015*, vol. IV, n.º 2, p. 43, mar. 2015.

[8] G. Meneses Benítez, «Las TICs en la Universidad», Universidad Rovira I Virgili, España, 2010.

[9] Atlantic International University, «E learning (aprendizaje electrónico), el futuro de la educación», *Vota Tu Profesor | Califica tu profesor y tu Centro de Educación*, 23-oct-2017. [En línea]. Disponible en: https://www.votatuprofesor.com/blog/item/685-elearning-el-futuro-de-la-educacion. [Accedido: 05-mar-2018].

[10] C. B. Busoch Morlán, R. Garrote, C. Regueiro Busoch, y A. Regueiro Gómez, «LA FORMACIÓN INVESTIGATIVA CURRICULAR CON AYUDA DE LA INTEGRACIÓN DE UN SISTEMA DE GESTIÓN DEL APRENDIZAJE (MOODLECEBIO), TIC Y E-LEARNING PARA INGENIERÍAS». 24-nov-2014.

[11] D. C. R. Tamayo Cuenca, Ms. L. Álvarez Hernández, y D. C. L. Bao Pavón, «Estretegia metodológica para el uso de la plataforma Mooodle en la Universidad de Holguín», *Conectando Soc.*, mar. 2016.

[12] Grupo de Aprendizaje Virtual y Educación a Distancia, UCLV, «Moodle UCLV», 05-mar-2018. [En línea]. Disponible en: https://moodle.uclv.edu.cu/. [Accedido: 05-mar-2018].

[13] P. Fernández Sánchez, A. Salaverría Garnacho, J. González Dacosta, y E. Mandado Pérez, «El aprendizaje activo mediante la autoevaluación utilizando un laboratorio virtual», *IEEE-RITA*, vol. 4, n.º 1, feb. 2009.

[14] Diccionario de la Lengua española, «Software», *Diccionario de la Lengua Española*. Espasa-Clape, 2005.

[15] R. S. Pressman y J. M. Troya, «Ingeniería del software», 1988.

[16] IEEE Standards Association, «Software Engineering Standerd: Glosary of Software Engineering Terminology», 2015.
[17] M. Vidal Ledo, F. Gómez Martínez, R. Piedra, y A. M, «Software educativos», *Educ. Médica Super.*, vol. 24, n.º 1, pp. 97-110, mar. 2010.
[18] N. Sharon, «What You Need To Know About Educational Software», *eLearning Industry*, 10-mar-2017. [En línea]. Disponible en: https://elearningindustry.com/need-know-educational-software. [Accedido: 30-abr-2018].
[19] «Principles of Software Design & Concepts in Software Engineering». [En línea]. Disponible en: http://ecomputernotes.com/software-engineering/principles-of-software-design-and-concepts. [Accedido: 30-abr-2018].
[20] G. Booch, *Análisis y Diseño Orientado a Objetos con Aplicaciones*, 3ra ed. EE.UU: Edison Wesley, 2004.
[21] J. Carroll, *Diseño basado en escenarios: visión de trabajo y tecnología en el desarrollo de sistemas*. New York, 1995.
[22] M. Sinking, «Software redesign and modernization of existing software», 11-ago-2014. [En línea]. Disponible en: https://www.edsd.com/about/blog/software-redesign-and-modernization. [Accedido: 30-abr-2018].
[23] Facultad de Ingeniería Eléctrica UCLV, «Programa de la disciplina: Computación». 2007.
[24] C. Roth, «Fundamentos de diseños de circuitos». 5ta edición.
[25] «Software de diseño de circuitos electrónicos - Monografias.com», 23-mar-2018. [En línea]. Disponible en: http://www.monografias.com/trabajos107/software-diseno-circuitos-electronicos/software-diseno-circuitos-electronicos.shtml. [Accedido: 23-mar-2018].
[26] W. J. M, «A program for the Nonlinear DC Analysis of Bipolar Transistor Circuits», *IEEEE J. Solid State Circuits*. pp. 14-19, 1971.
[27] J. Pinto Bruno, «Every Circuit», *Biblioteca Escolar Digital*, 16-feb-2018. [En línea]. Disponible en: http://bibliotecaescolardigital.es/comunidad/BibliotecaEscolarDigital/recurso/every-circuit/18cfbb7c-8986-45bc-aea6-031302176261. [Accedido: 16-feb-2018].
[28] A. Sanchez, «ElectroDroid: En tus manos una aplicación rápida, fácil y sencilla para la Electrónica», *Azul Web*, 31-ago-2014. [En línea]. Disponible en: https://www.azulweb.net/electrodroid-en-tus-manos-una-aplicacion-rapida-facil-y-sencilla-para-la-electronica/. [Accedido: 16-feb-2018].
[29] National Instruments, «NI Multisim Applications». [En línea]. Disponible en: http://www.ni.com/multisim/applications/. [Accedido: 30-abr-2018].
[30] Ilirey SA, «Programas de diseño electrónico y simulación... - YoReparo», 16-feb-2018. [En línea]. Disponible en: https://yoreparo.com/electronica/laboratorios-virtuales/preguntas/274599/programas-de-diseno-electronico-y-simulacion. [Accedido: 16-feb-2018].
[31] Wikipedia, la enciclopedia libre, «Simulink», *es.wikipedia.org*. 13-dic-2017.
[32] «Simulink - Simulación y diseño basado en modelos», 16-feb-2018. [En línea]. Disponible en: https://es.mathworks.com/products/simulink.html. [Accedido: 16-feb-2018].
[33] MathWork, «¿Qué es proteus?», 16-feb-2018. [En línea]. Disponible en: http://www.hubor-proteus.com/proteus-pcb/proteus-pcb/2-proteus.html. [Accedido: 16-feb-2018].

[34] Anonimo, «Proteus Professional v8.5 SP0 + Portable, Suite Profesional de Simulación de Circuitos Electrónicos - IntercambiosVirtuales», 16-feb-2018. .
[35] M. Lutz, *Learning Python*, Cuarta. O´Roelly, 2010.
[36] IncludeHelp, «Computer programming languages and its types». [En línea]. Disponible en: https://www.includehelp.com/basics/computer-programming-languages.aspx. [Accedido: 30-abr-2018].
[37] Universia Argrntina, «Los 10 lenguajes de programación más populares en la actualidad». [En línea]. Disponible en: http://noticias.universia.com.ar/consejos-profesionales/noticia/2016/02/22/1136443/conoce-cuales-lenguajes-programacion-populares.html. [Accedido: 17-may-2018].
[38] Cedejob, «¿Qué es Python? - Aprende a Programar», 16-feb-2018. [En línea]. Disponible en: https://www.codejobs.biz/es/blog/2013/03/02/que-es-python. [Accedido: 16-feb-2018].
[39] Bejob, «7 razones para programar en Python». .
[40] D. Xiloj Curruchiche, «Resistencias y sus valores estándar», 02-dic-2007. [En línea]. Disponible en: https://denjohx.wordpress.com/2007/12/02/resistencias-y-sus-valores-estandar/. [Accedido: 22-jun-2018].
[41] Facultad de Ingeniería Eléctrica UCLV, «Programa de la Disciplina: Electrónica». 2007.

ANEXOS

Anexo I: Encuesta estudiantil para determinar la importancia de la programación y la utilización de software en la carrera Ingeniería en Telecomunicaciones y Electrónica

Carrera: Año:

¿Usted considera que la programación es importante para la carrera Ingeniería en Telecomunicaciones y Electrónica?

___Si ___No

¿por qué?_____

¿En el transcurso de la carrera usted ha utilizado software como apoyo en asignaturas de la misma?

___Siempre ___Casi siempre ___Algunas veces ___Casi nunca ___Nunca

¿Cree usted que la programación tiene aplicación práctica en las asignaturas de la carrera?

___Si ___No

¿por qué?_____

¿Considera usted suficiente el número de software de apoyo en disciplinas como la Electrónica?

___Suficiente __Insuficiente

¿por qué?_____

Anexo II: Líneas de código para estandarizar los valores de resistencias

```
def standardValue(self, standard=1):
    r = self.R
    i = 0
    while r > 10.0:
        r = r / 10.0
        i += 1
    d = 10 ** i
    ki = e96
    if standard == 2:
        ki = e48
    elif standard == 3:
        ki = e24
    elif standard == 9:
        ki = e12
    kn = 10
    di = 10
    for k in ki:
        if abs(k - r) < di:
            di = abs(k - r)
            kn = k
    return d * kn
```

Anexo III: Código utilizado para el diseño del esquema del Punto Q

```
def __init__(self, voltage, Rc, Ic, Re, parent=None):
    super(pointDialog, self).__init__(parent)
    self.setWindowTitle("Amplifiers. -. Operation point.")
    self.setFixedSize(575, 500)
    self.imageLabel = QLabel()
    self.imageLabel.setMinimumSize(276, 376)
    self.imageLabel.setAlignment(Qt.AlignCenter)
    image = QImage(":/op.png")
    self.imageLabel.setPixmap(QPixmap.fromImage(image))
    p = voltage / (Rc + Re)
    m = - 1.0 / (Rc + Re)
    self.label01 = QLabel("Ic = " + str(Ic) + ". Vce = " + str(voltage / 3) + ". Vcc = " +
str(voltage) + ".")
    self.label01.setAlignment(Qt.AlignCenter)
    self.label02 = QLabel("p = Vcc / (Rc + Re) = " + str(p) + ". m = " + str(m) + ".")
    self.label02.setAlignment(Qt.AlignCenter)
    self.layout02 = QVBoxLayout()
    self.layout02.addWidget(self.imageLabel)
    self.layout02.addWidget(self.label01)
    self.layout02.addWidget(self.label02)
    self.setLayout(self.layout02)
```