.UBAfiuba
FACULTAD DE INGENIERÍA

Desarrollo de un sistema embebido para la gestión y monitoreo de cultivos verticales

Ing. José Luis Krüger

Carrera de Especialización en Sistemas Embebidos

Director: Esp. Ing. Mariano Campos (FIUBA)

Jurados:

Jurado 1 (pertenencia)
Jurado 2 (pertenencia)
Jurado 3 (pertenencia)

Ciudad de [lugar], [mes] de [año]

Resumen

Este trabajo presenta el desarrollo, construcción, instalación y prueba de un dispositivo semiautomático de monitoreo y gestión para un cultivo hidropónico vertical. Su implementación requirió la aplicación de conocimientos en electrónica, programación, sistemas operativos de tiempo real, testing de software, diseño y ensamblaje de PCB, integración de sensores y protocolos de comunicación.

Los resultados muestran una mejora en la eficiencia del cultivo, reduciendo desperdicios y aumentando la productividad en comparación con los métodos tradicionales. Este avance representa un paso clave hacia la sostenibilidad y la optimización en la producción de alimentos del futuro.

Agradecimientos

Esta sección es para agradecimientos personales y es totalmente **OPCIONAL**.

Índice general

Índice de figuras

Índice de tablas

Dedicado a... [OPCIONAL]

Capítulo 1

Introducción general

En este capítulo se hace una breve introducción a la necesidad que condujo al desarrollo del trabajo. Se presenta el concepto de cultivo vertical y el estado del arte de su integración con la tecnología. Asimismo, se explica el objetivo y los alcances del trabajo.

1.1. Introducción a los cultivos verticales

Con una población mundial que supera los 8000 millones de personas y continúa en crecimiento, la agricultura enfrenta el desafío de volverse más eficiente y sostenible. En este contexto, los cultivos verticales emergen como una solución innovadora que optimiza el uso del espacio y los recursos, especialmente en entornos urbanos, donde el suelo es limitado y costoso.

Este tipo de agricultura utiliza técnicas como la hidroponía, que permite un uso preciso del agua, y la aeroponía, que maximiza el oxígeno disponible para las raíces. Además, las granjas verticales aprovechan áreas infrautilizadas, como edificios abandonados o naves industriales, y permiten cultivar alimentos en zonas donde la agricultura tradicional resulta dificil de desarrollar. Así, se logra una mayor densidad de cultivo y se contribuye a la sostenibilidad al reducir el uso de pesticidas y fertilizantes.

Aunque los cultivos verticales requieren un alto nivel de tecnología y tienen costes energéticos asociados, su capacidad para ahorrar recursos, disminuir la huella de carbono y fomentar la producción.

Este método, también conocido como agricultura urbana, permite el crecimiento de plantas en interiores sin necesidad de luz solar directa, gracias a la aplicación de tecnologías agrícolas avanzadas. Estas optimizan las condiciones de cultivo, facilitando la propagación de plantas jóvenes y la producción de alimentos más saludables sin el uso de pesticidas.

Además, maximizan la producción al emplear sistemas de iluminación especializados de nueva generación, diseñados para estructuras multicapa. Gracias a esta tecnología, es posible obtener mayores rendimientos en un espacio reducido, consolidando a la agricultura vertical como una solución sostenible y eficiente para la producción de alimentos en entornos urbanos y con recursos limitados.

1.1.1. Tipos de cultivos verticales

Existen distintos tipos de agricultura vertical, que optimizan el crecimiento de las plantas sin grandes extensiones de suelo y emplean estructuras especializadas para maximizar la producción.

Hidroponía

La hidroponía es un método de cultivo sin suelo que suministra nutrientes a las plantas a través de una solución acuosa, con raíces en un medio inerte como lana de roca o perlita. Es un sistema eficiente en el uso de agua, permite un control preciso de nutrientes y favorece una mayor densidad de cultivo y un crecimiento más rápido que los métodos tradicionales.

Aeroponía

La aeroponía es un método de cultivo en el que las raíces de las plantas quedan suspendidas en el aire y se rocían con una solución nutritiva. Maximiza la oxigenación, acelera el crecimiento y optimiza el uso de nutrientes, aunque requiere un control preciso para evitar la desecación de las raíces.

Acuaponía

La acuaponía combina la cría de peces con la hidroponía, utilizando los desechos de los peces como fertilizante para las plantas, mientras estas purifican el agua. Es un sistema sostenible y eficiente que minimiza el desperdicio, promueve la biodiversidad y es viable en entornos urbanos y rurales.

1.1.2. Sistemas de agricultura vertical más conocidos

Los sistemas de agricultura vertical varían según el tipo de cultivo y la técnica más adecuada para cada necesidad.

Sistemas de torre

Las torres verticales permiten el cultivo de plantas en estructuras cilíndricas apiladas, maximizando el uso del espacio y facilitando tanto la recolección como el mantenimiento. Este sistema es ideal para cultivos de hoja verde y hierbas, y se integra fácilmente en espacios urbanos como balcones y terrazas.

Sistemas en rack o estanterías

Estos sistemas emplean estanterías apiladas donde las plantas crecen en bandejas o contenedores, son ideales para invernaderos y ambientes controlados. Facilitan el riego, la recolección y el monitoreo, al mismo tiempo que se optimiza el uso del espacio. Los sistemas en rack o estanterías son especialmente útiles para cultivos que requieren diferentes niveles de luz y humedad.

Sistemas montados en pared

En estos sistemas, las plantas crecen en módulos montados en paredes. Su integración en la arquitectura no solo optimiza el espacio, sino que también aporta

beneficios estéticos y funcionales. Los jardines verticales pueden mejorar la calidad del aire y contribuir al aislamiento térmico de los edificios.

Sistemas de bastidor en A

El bastidor en forma de .A"proporciona una estructura estable y accesible para el cultivo vertical, lo que a su vez facilita el acceso a la luz y el riego. Este diseño es particularmente útil para cultivos que necesitan un soporte adicional, como tomates y pepinos, y puede ser utilizado tanto en interiores como en exteriores.

Sistemas de contenedores

Estos sistemas utilizan contenedores apilables que pueden moverse y configurarse según las necesidades, adaptándose a diversos entornos y tipos de cultivo. Los contenedores son ideales para cultivos modulares y pueden ser personalizados para optimizar el uso del espacio y los recursos.

1.2. Motivación

Este trabajo surge como un desarrollo de interés personal, impulsado por el deseo de contribuir positivamente al desarrollo sustentable y la optimización de los recursos naturales en la producción de alimentos. En la actualidad, el crecimiento de la población mundial y el aumento en la demanda de productos agrícolas presentan grandes desafíos ambientales y económicos.

La producción tradicional de alimentos enfrenta dificultades debido a la explotación intensiva de suelos, el uso excesivo de fertilizantes y pesticidas y el desperdicio de recursos esenciales como el agua. Con el fin de maximizar sus ganancias, muchas empresas recurren a métodos poco sostenibles que generan un impacto ambiental negativo, y afectan la biodiversidad, la calidad del suelo y el bienestar de las comunidades locales que dependen de estas fuentes de producción.

Uno de los problemas más críticos es el desperdicio de agua en la agricultura tradicional. Se estima que casi el 90 % del agua utilizada en el riego se pierde por evaporación, drenaje o absorción ineficiente. En un contexto donde la crisis hídrica es una amenaza global, desarrollar sistemas más sostenibles se vuelve una necesidad urgente.

Por otro lado, existe una creciente preocupación entre los consumidores sobre la calidad y procedencia de los alimentos. Cada vez más personas optan por un estilo de vida saludable, adoptando hábitos de consumo responsables, como el vegetarianismo, el veganismo y el autocultivo. Sin embargo, muchos de estos consumidores no cuentan con espacios adecuados ni conocimientos técnicos para producir sus propios alimentos, lo que genera una gran oportunidad para soluciones tecnológicas que faciliten la agricultura doméstica.

A partir de estas problemáticas, nació la motivación para desarrollar un sistema hidropónico inteligente, que no solo optimice el uso del agua, sino que también permita una gestión eficiente del cultivo con mínima intervención del usuario. Este trabajo ofrece una alternativa viable y escalable para la producción de alimentos frescos, reduce la dependencia de sistemas agrícolas tradicionales y promueve un modelo más sustentable para el futuro.

1.3. Estado del arte

Durante la etapa de investigación, se llevó a cabo una búsqueda de productos comerciales tanto en el mercado local como en el internacional. Se identificaron diversas soluciones con características similares al sistema desarrollado.

A continuación, se describen los productos identificados.

NIDO PRO

El NIDO PRO, figura 1.1, es un sistema hidropónico inteligente para cultivos verticales que permite controlar la solución nutritiva a través de una aplicación móvil. Facilita la gestión del pH y la conductividad eléctrica (CE), con algoritmos que realizan comprobaciones diarias para garantizar la estabilidad de la solución nutritiva. Además, admite hasta cuatro ranuras para fertilizantes líquidos y ajustes de pH, lo que optimiza el proceso de nutrición de las plantas.

FIGURA 1.1. Ilustración del módulo NIDO PRO conectado a un sistema hidropónico horizontal.

Xiaomi Mi Flower Care Plant Sensor

El Xiaomi Mi Flower Care Plant Sensor, figura 1.2, monitorea variables ambientales como luz, humedad, nutrientes y temperatura del sustrato. Los sensores se conectan a aplicaciones móviles, lo que permite realizar un seguimiento detallado y controlar las condiciones del cultivo de manera precisa.

Kit de Riego Hydro de Konyks

El sistema de riego inteligente Konyks, figura 1.3, permite controlar la llave de paso de forma remota mediante una aplicación móvil y asistentes de voz como Alexa, Google Home o Siri. Incorpora un medidor de caudal para monitorear el consumo de agua y una toma con relé RF que amplía la conectividad hasta 60 m

FIGURA 1.2. Ilustración del dispositivo Xiaomi Mi Flower Care Plant Sensor y sus prestaciones.

y permite gestionar hasta cuatro grifos. Además, ofrece automatización programable según el clima y los horarios, lo que optimiza el uso del agua y facilita la gestión del riego desde cualquier lugar.

FIGURA 1.3. Sistema de riego conectado Konyks.

Smart 9 Pro de Click & Grow

El Smart Garden 9 PRO, figura 1.4, es un sistema de cultivo doméstico automatizado con control a través de una aplicación. Ofrece riego automático, iluminación ajustable tanto por control táctil como desde la app, y un suministro equilibrado de nutrientes y oxígeno en las raíces. Permite cultivar hierbas, frutas, ensaladas y flores durante todo el año, con la opción de utilizar cápsulas pre-sembradas

(más de 50 variedades disponibles) o semillas propias. Incluye un set inicial con cápsulas de tomate, albahaca y lechuga.

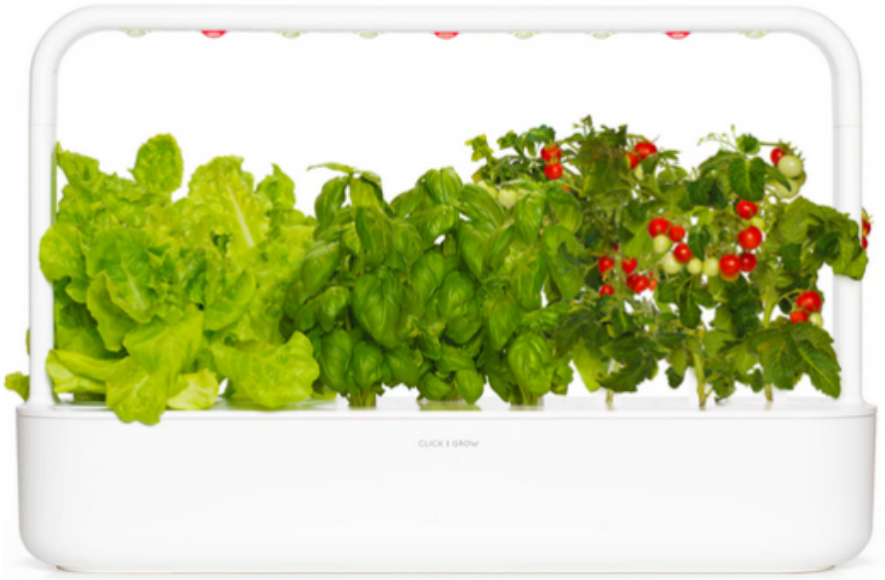


FIGURA 1.4. Módulo Smart Garden 9 PRO.

1.4. Objetivos y alcances

1.4.1. Objetivos

- Desarrollar un prototipo de sistema hidropónico automatizado para monitoreo y gestión eficiente del cultivo.
- Implementar sensores para medir temperatura, humedad, pH, conductividad eléctrica, nivel de agua e intensidad lumínica.
- Optimizar el uso de recursos, reduciendo el consumo de agua y nutrientes mediante control automatizado.
- Diseñar una interfaz web y móvil que permita la supervisión y configuración remota del sistema.
- Evaluar el rendimiento del sistema en comparación con métodos tradicionales, midiendo eficiencia y productividad.
- Poner en práctica los conocimientos adquiridos a lo largo de la carrera de especialización en sistemas embebidos.

1.4.2. Alcances

- El prototipo permitirá el monitoreo en tiempo real y la gestión automatizada del riego, iluminación y ventilación.
- Se desarrollará un módulo de comunicación IoT, asegurando conectividad con dispositivos móviles y servidores locales.
- La solución se enfocará en cultivos hidropónicos verticales de pequeña y mediana escala.

- Se realizarán pruebas de funcionamiento y validación en un entorno controlado, pero no se contempla una implementación comercial en esta etapa.

1.5. Qué incluye esta plantilla

1.5.1. Carpetas

Esta plantilla se distribuye como una único archivo .zip que se puede descomprimir en varios archivos y carpetas. Asimismo, se puede consultar el repositorio git para obtener la última versión de los archivos, https://github.com/patriciobos/Plantilla-CESE.git. Los nombres de las carpetas son, o pretender ser, auto-explicativos.

Appendices – Esta es la carpeta donde se deben poner los apéndices. Cada apéndice debe ir en su propio archivo **`.tex`**. Se incluye un ejemplo y una plantilla en la carpeta.

Chapters – Esta es la carpeta donde se deben poner los capítulos de la memoria. Cada capítulo debe ir un su propio archivo **`.tex`** por separado. Se ofrece por defecto, la siguiente estructura de capítulos y se recomienda su utilización dentro de lo posible:

- Capítulo 1: Introducción general
- Capítulo 2: Introducción específica
- Capítulo 3: Diseño e implementación
- Capítulo 4: Ensayos y resultados
- Capítulo 5: Conclusiones

Esta estructura de capítulos es la que se recomienda para las memorias de la especialización.

Figures – Esta carpeta contiene todas las figuras de la memoria. Estas son las versiones finales de las imágenes que van a ser incluidas en la memoria. Pueden ser imágenes en formato *raster*[1] como **`.png`**, **`.jpg`** o en formato vectoriales[2] como **`.pdf`**, **`.ps`**. Se debe notar que utilizar imágenes vectoriales disminuye notablemente el peso del documento final y acelera el tiempo de compilación por lo que es recomendable su utilización siempre que sea posible.

1.5.2. Archivos

También están incluidos varios archivos, la mayoría de ellos son de texto plano y se puede ver su contenido en un editor de texto. Después de la compilación inicial, se verá que más archivos auxiliares son creados por LaTeX o BibTeX, pero son de uso interno y no es necesario hacer nada en particular con ellos. Toda la información necesaria para compilar el documento se encuentra en los archivos **`.tex`**, **`.bib`**, **`.cls`** y en las imágenes de la carpeta Figures.

[1] https://en.wikipedia.org/wiki/Raster_graphics

[2] https://en.wikipedia.org/wiki/Vector_graphics

referencias.bib - este es un archivo importante que contiene toda la información de referencias bibliográficas que se utilizarán para las citas en la memoria en conjunto con BibTeX. Usted puede escribir las entradas bibliográficas en forma manual, aunque existen también programas de gestión de referencias que facilitan la creación y gestión de las referencias y permiten exportarlas en formato BibTeX. También hay disponibles sitios web como books.google.com que permiten obtener toda la información necesaria para una cita en formato BibTeX. Ver sección 1.9

MastersDoctoralThesis.cls – este es un archivo importante. Es el archivos con la clase que le informa a LaTeX cómo debe dar formato a la memoria. El usuario de la plantilla no debería necesitar modificar nada de este archivo.

memoria.pdf – esta es su memoria con una tipografía bellamente compuesta (en formato de archivo PDF) creado por LaTeX. Se distribuye con la plantilla y después de compilar por primera vez sin hacer ningún cambio se debería obtener una versión idéntica a este documento.

memoria.tex – este es un archivo importante. Este es el archivo que tiene que compilar LaTeX para producir la memoria como un archivo PDF. Contiene un marco de trabajo y estructuras que le indican a LaTeX cómo diagramar la memoria. Está altamente comentado para que se pueda entender qué es lo que realiza cada línea de código y por qué está incluida en ese lugar. En este archivo se debe completar la información personalizada de las primeras sección según se indica en la sección 1.7.

Archivos que *no* forman parte de la distribución de la plantilla pero que son generados por LaTeX como archivos auxiliares necesarios para la producción de la memoria.pdf son:

memoria.aux – este es un archivo auxiliar generado por LaTeX, si se borra LaTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**.

memoria.bbl – este es un archivo auxiliar generado por BibTeX, si se borra BibTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**. Mientras que el archivo **`.bib`** contiene todas las referencias que hay, este archivo **`.bbl`** contine sólo las referencias que han sido citadas y se utiliza para la construcción de la bibiografía.

memoria.blg – este es un archivo auxiliar generado por BibTeX, si se borra BibTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**.

memoria.lof – este es un archivo auxiliar generado por LaTeX, si se borra LaTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**. Le indica a LaTeX cómo construir la sección *Lista de Figuras*.

memoria.log – este es un archivo auxiliar generado por LaTeX, si se borra LaTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**. Contiene mensajes de LaTeX. Si se reciben errores o advertencias durante la compilación, se guardan en este archivo **`.log`**.

memoria.lot – este es un archivo auxiliar generado por LaTeX, si se borra LaTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**. Le indica a LaTeX cómo construir la sección *Lista de Tablas*.

memoria.out – este es un archivo auxiliar generado por LaTeX, si se borra LaTeX simplemente lo regenera cuando se compila el archivo principal **`memoria.tex`**.

De esta larga lista de archivos, sólo aquellos con la extensión **`.bib`**, **`.cls`** y **`.tex`** son importantes. Los otros archivos auxiliares pueden ser ignorados o borrados ya que LaTeX y BibTeX los regenerarán durante la compilación.

1.6. Entorno de trabajo

Ante de comenzar a editar la plantilla debemos tener un editor LaTeX instalado en nuestra computadora. En forma análoga a lo que sucede en lenguaje C, que se puede crear y editar código con casi cualquier editor, existen ciertos entornos de trabajo que nos pueden simplificar mucho la tarea. En este sentido, se recomienda, sobre todo para los principiantes en LaTeX la utilización de TexMaker, un programa gratuito y multi-plantaforma que está disponible tanto para windows como para sistemas GNU/linux.

La versión más reciente de TexMaker es la 4.5 y se puede descargar del siguiente link: http://www.xm1math.net/texmaker/download.html. Se puede consultar el manual de usuario en el siguiente link: http://www.xm1math.net/texmaker/doc.html.

1.6.1. Paquetes adicionales

Si bien durante el proceso de instalación de TexMaker, o cualquier otro editor que se haya elegido, se instalarán en el sistema los paquetes básicos necesarios para trabajar con LaTeX, la plantilla de los trabajos de Especialización y Maestría requieren de paquete adicionales.

Se indican a continuación los comandos que se deben introducir en la consola de Ubuntu (ctrl + alt + t) para instalarlos:

```
$ sudo apt install texlive-lang-spanish texlive-science
$ sudo apt install texlive-bibtex-extra biber
$ sudo apt install texlive texlive-fonts-recommended
$ sudo apt install texlive-latex-extra
```

1.6.2. Configurando TexMaker

Una vez instalado el programa y los paquetes adicionales se debe abrir el archivo memoria.tex con el editor para ver una pantalla similar a la que se puede apreciar en la figura 1.5. Una vez instalado el programa y los paquetes adicionales se debe abrir el archivo memoria.tex con el editor para ver una pantalla similar a la que se puede apreciar en la figura 1.5. Una vez instalado el programa y los paquetes adicionales se debe abrir el archivo memoria.tex con el editor para ver una pantalla similar a la que se puede apreciar en la figura 1.5. Una vez instalado el programa y los paquetes adicionales se debe abrir el archivo memoria.tex con el editor para ver una pantalla similar a la que se puede apreciar en la figura 1.5.

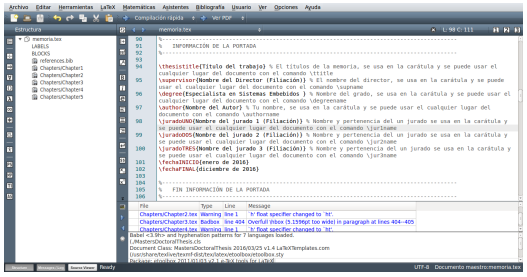


FIGURA 1.5. Entorno de trabajo de texMaker.

Notar que existe una vista llamada Estructura a la izquierda de la interfaz que nos permite abrir desde dentro del programa los archivos individuales de los capítulos. A la derecha se encuentra una vista con el archivo propiamente dicho para su edición. Hacia la parte inferior se encuentra una vista del log con información de los resultados de la compilación. En esta última vista pueden aparecen advertencias o *warning*, que normalmente pueden ser ignorados, y los errores que se indican en color rojo y deben resolverse para que se genere el PDF de salida.

Recordar que el archivo que se debe compilar con PDFLaTeX es **`memoria.tex`**, si se tratara de compilar alguno de los capítulos saldría un error. Para salvar la molestia de tener que cambiar de archivo para compilar cada vez que se realice una modificación en un capítulo, se puede definir el archivo **`memoria.tex`** como "documento maestro" yendo al menú opciones ->"definir documento actual como documento maestro", lo que permite compilar con PDFLaTeX memoria.tex directamente desde cualquier archivo que se esté modificando . Se muestra esta opción en la figura 1.6.

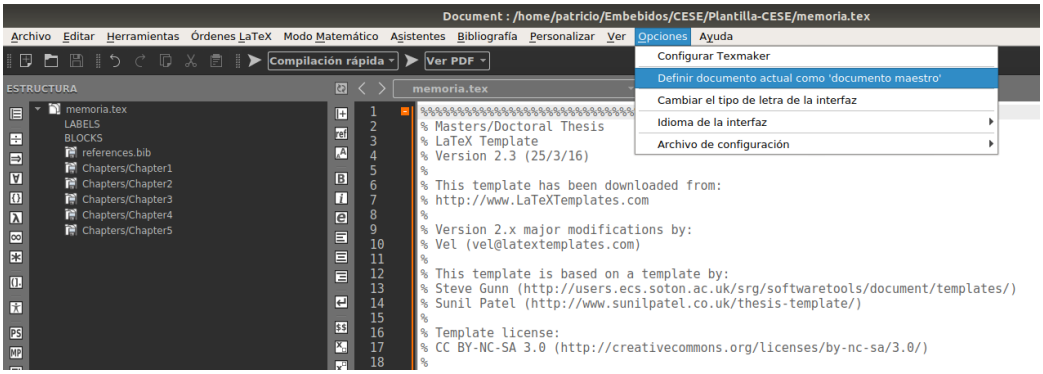


FIGURA 1.6. Definir memoria.tex como documento maestro.

En el menú herramientas se encuentran las opciones de compilación. Para producir un archivo PDF a partir de un archivo .tex se debe ejecutar PDFLaTeX (el shortcut es F6). Para incorporar nueva bibliografía se debe utilizar la opción BibTeX del mismo menú herramientas (el shortcut es F11).

Notar que para actualizar las tablas de contenidos se debe ejecutar PDFLaTeX dos veces. Esto se debe a que es necesario actualizar algunos archivos auxiliares antes de obtener el resultado final. En forma similar, para actualizar las referencias bibliográficas se debe ejecutar primero PDFLaTeX, después BibTeX y finalmente PDFLaTeX dos veces por idénticos motivos.

1.7. Personalizando la plantilla, el archivo `memoria.tex`

Para personalizar la plantilla se debe incorporar la información propia en los distintos archivos **.tex**.

Primero abrir **memoria.tex** con TexMaker (o el editor de su preferencia). Se debe ubicar dentro del archivo el bloque de código titulado *INFORMACIÓN DE LA PORTADA* donde se deben incorporar los primeros datos personales con los que se construirá automáticamente la portada.

1.8. El código del archivo `memoria.tex` explicado

El archivo **memoria.tex** contiene la estructura del documento y es el archivo de mayor jerarquía de la memoria. Podría ser equiparable a la función *main()* de un programa en C, o mejor dicho al archivo fuente .c donde se encuentra definida la función main().

La estructura básica de cualquier documento de LaTeX comienza con la definición de clase del documento, es seguida por un preámbulo donde se pueden agregar funcionalidades con el uso de `paquetes` (equiparables a bibliotecas de C), y finalmente, termina con el cuerpo del documento, donde irá el contenido de la memoria.

```
\documentclass{article}  <- Definicion de clase
\usepackage{listings}    <- Preambulo

\begin{document}         <- Comienzo del contenido propio
        Hello world!
\end{document}
```

El archivo **memoria.tex** se encuentra densamente comentado para explicar qué páginas, secciones y elementos de formato está creando el código LaTeX en cada línea. El código está dividido en bloques con nombres en mayúsculas para que resulte evidente qué es lo que hace esa porción de código en particular. Inicialmente puede parecer que hay mucho código LaTeX, pero es principalmente código para dar formato a la memoria por lo que no requiere intervención del usuario de la plantilla. Sí se deben personalizar con su información los bloques indicados como:

- Informacion de la memoria
- Resumen
- Agradecimientos
- Dedicatoria

El índice de contenidos, las listas de figura de tablas se generan en forma automática y no requieren intervención ni edición manual por parte del usuario de la plantilla.

En la parte final del documento se encuentran los capítulos y los apéndices. Por defecto se incluyen los 5 capítulos propuestos que se encuentran en la carpeta /Chapters. Cada capítulo se debe escribir en un archivo .tex separado y se debe poner en la carpeta *Chapters* con el nombre **Chapter1**, **Chapter2**, etc... El código para incluir capítulos desde archivos externos se muestra a continuación.

```
\include{Chapters/Chapter1}
\include{Chapters/Chapter2}
\include{Chapters/Chapter3}
\include{Chapters/Chapter4}
\include{Chapters/Chapter5}
```

Los apéndices también deben escribirse en archivos .tex separados, que se deben ubicar dentro de la carpeta *Appendices*. Los apéndices vienen comentados por defecto con el caracter `%` y para incluirlos simplemente se debe eliminar dicho caracter.

Finalmente, se encuentra el código para incluir la bibliografía en el documento final. Este código tampoco debe modificarse. La metodología para trabajar las referencias bibliográficas se desarrolla en la sección 1.9.

1.9. Bibliografía

Las opciones de formato de la bibliografía se controlan a través del paquete de latex *biblatex* que se incluye en la memoria en el archivo memoria.tex. Estas opciones determinan cómo se generan las citas bibliográficas en el cuerpo del documento y cómo se genera la bibliografía al final de la memoria.

En el preámbulo se puede encontrar el código que incluye el paquete biblatex, que no requiere ninguna modificación del usuario de la plantilla, y que contiene las siguientes opciones:

```
\usepackage[backend=bibtex,
        natbib=true,
        style=numeric,
        sorting=none]
{biblatex}
```

En el archivo **reference.bib** se encuentran las referencias bibliográficas que se pueden citar en el documento. Para incorporar una nueva cita al documento lo primero es agregarla en este archivo con todos los campos necesario. Todas las entradas bibliográficas comienzan con @ y una palabra que define el formato de la entrada. Para cada formato existen campos obligatorios que deben completarse. No importa el orden en que las entradas estén definidas en el archivo .bib. Tampoco es importante el orden en que estén definidos los campos de una entrada bibliográfica. A continuación se muestran algunos ejemplos:

```
@ARTICLE{ARTICLE:1,
    AUTHOR="John Doe",
    TITLE="Title",
    JOURNAL="Journal",
    YEAR="2017",
}

@BOOK{BOOK:1,
    AUTHOR="John Doe",
    TITLE="The Book without Title",
    PUBLISHER="Dummy Publisher",
    YEAR="2100",
}
```

```
@INBOOK{BOOK:2,
    AUTHOR="John Doe",
    TITLE="The Book without Title",
    PUBLISHER="Dummy Publisher",
    YEAR="2100",
    PAGES="100-200",
}

@MISC{WEBSITE:1,
    HOWPUBLISHED = "\url{http://example.com}",
    AUTHOR = "Intel",
    TITLE = "Example Website",
    MONTH = "12",
    YEAR = "1988",
    URLDATE = {2012-11-26}
}
```

Se debe notar que los nombres *ARTICLE:1*, *BOOK:1*, *BOOK:2* y *WEBSITE:1* son nombres de fantasía que le sirve al autor del documento para identificar la entrada. En este sentido, se podrían reemplazar por cualquier otro nombre. Tampoco es necesario poner : seguido de un número, en los ejemplos sólo se incluye como un posible estilo para identificar las entradas.

La entradas se citan en el documento con el comando:

```
\citep{nombre_de_la_entrada}
```

Y cuando se usan, se muestran así: [**ARTICLE:1**], [**BOOK:1**], [**BOOK:2**], [**WEBSITE:1**]. Notar cómo se conforma la sección Bibliografía al final del documento.

Finalmente y como se mencionó en la subsección 1.6.2, para actualizar las referencias bibliográficas tanto en la sección bibliografía como las citas en el cuerpo del documento, se deben ejecutar las herramientas de compilación PDFLaTeX, BibTeX, PDFLaTeX, PDFLaTeX, en ese orden. Este procedimiento debería resolver cualquier mensaje Çitation xxxxx on page x undefined".

Capítulo 2

Introducción específica

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

2.1. Estilo y convenciones

2.1.1. Uso de mayúscula inicial para los título de secciones

Si en el texto se hace alusión a diferentes partes del trabajo referirse a ellas como capítulo, sección o subsección según corresponda. Por ejemplo: "En el capítulo 1 se explica tal cosa", o "En la sección 2.1 se presenta lo que sea", o "En la subsección 2.1.2 se discute otra cosa".

Cuando se quiere poner una lista tabulada, se hace así:

- Este es el primer elemento de la lista.
- Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

Si se desea poner una lista numerada el formato es este:

1. Este es el primer elemento de la lista.
2. Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

2.1.2. Este es el título de una subsección

Se recomienda no utilizar **texto en negritas** en ningún párrafo, ni tampoco texto subrayado. En cambio sí se debe utilizar *texto en itálicas* para palabras en un idioma extranjero, al menos la primera vez que aparecen en el texto. En el caso de palabras que estamos inventando se deben utilizar "comillas", así como también para citas textuales. Por ejemplo, un *digital filter* es una especie de "selector" que permite separar ciertos componentes armónicos en particular.

La escritura debe ser impersonal. Por ejemplo, no utilizar "el diseño del firmware lo hice de acuerdo con tal principio", sino "el firmware fue diseñado utilizando tal principio".

El trabajo es algo que al momento de escribir la memoria se supone que ya está concluido, entonces todo lo que se refiera a hacer el trabajo se narra en tiempo pasado, porque es algo que ya ocurrió. Por ejemplo, "se diseñó el firmware empleando la técnica de test driven development".

En cambio, la memoria es algo que está vivo cada vez que el lector la lee. Por eso transcurre siempre en tiempo presente, como por ejemplo:

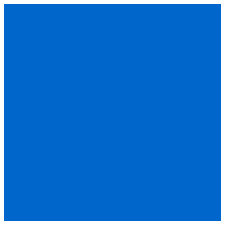
“En el presente capítulo se da una visión global sobre las distintas pruebas realizadas y los resultados obtenidos. Se explica el modo en que fueron llevados a cabo los test unitarios y las pruebas del sistema”.

Se recomienda no utilizar una sección de glosario sino colocar la descripción de las abreviaturas como parte del mismo cuerpo del texto. Por ejemplo, RTOS (*Real Time Operating System*, Sistema Operativo de Tiempo Real) o en caso de considerarlo apropiado mediante notas a pie de página.

Si se desea indicar alguna página web utilizar el siguiente formato de referencias bibliográficas, dónde las referencias se detallan en la sección de bibliografía de la memoria, utilizado el formato establecido por IEEE en [**IEEE:citation**]. Por ejemplo, “el presente trabajo se basa en la plataforma EDU-CIAA-NXP [**CIAA**], la cual...”.

2.1.3. Figuras

Al insertar figuras en la memoria se deben considerar determinadas pautas. Para empezar, usar siempre tipografía claramente legible. Luego, tener claro que **es incorrecto** escribir por ejemplo esto: “El diseño elegido es un cuadrado, como se ve en la siguiente figura:”



La forma correcta de utilizar una figura es con referencias cruzadas, por ejemplo: “Se eligió utilizar un cuadrado azul para el logo, como puede observarse en la figura 2.1”.

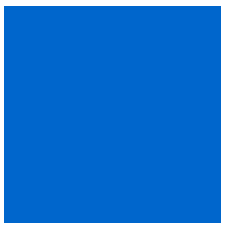


FIGURA 2.1. Ilustración del cuadrado azul que se eligió para el diseño del logo.

El texto de las figuras debe estar siempre en español, excepto que se decida reproducir una figura original tomada de alguna referencia. En ese caso la referencia

FIGURA 2.2. Imagen tomada de la página oficial del procesador[1].

de la cual se tomó la figura debe ser indicada en el epígrafe de la figura e incluida como una nota al pie, como se ilustra en la figura 2.2.

La figura y el epígrafe deben conformar una unidad cuyo significado principal pueda ser comprendido por el lector sin necesidad de leer el cuerpo central de la memoria. Para eso es necesario que el epígrafe sea todo lo detallado que corresponda y si en la figura se utilizan abreviaturas entonces aclarar su significado en el epígrafe o en la misma figura.

FIGURA 2.3. ¿Por qué de pronto aparece esta figura?

Nunca colocar una figura en el documento antes de hacer la primera referencia a ella, como se ilustra con la figura 2.3, porque sino el lector no comprenderá por qué de pronto aparece la figura en el documento, lo que distraerá su atención.

Otra posibilidad es utilizar el entorno *subfigure* para incluir más de una figura, como se puede ver en la figura 2.4. Notar que se pueden referenciar también las figuras internas individualmente de esta manera: 2.4a, 2.4b y 2.4c.

(A) Un caption. (B) Otro. (C) Y otro más.

FIGURA 2.4. Tres gráficos simples.

El código para generar las imágenes se encuentra disponible para su reutilización en el archivo **`Chapter2.tex`**.

2.1.4. Tablas

Para las tablas utilizar el mismo formato que para las figuras, sólo que el epígrafe se debe colocar arriba de la tabla, como se ilustra en la tabla 2.1. Observar que

[1]Imagen tomada de https://goo.gl/images/i7C70w

sólo algunas filas van con líneas visibles y notar el uso de las negritas para los encabezados. La referencia se logra utilizando el comando `\ref{<label>}` donde label debe estar definida dentro del entorno de la tabla.

```
\begin{table}[h]
\centering
\caption[caption corto]{caption largo más descriptivo}
\begin{tabular}{l c c}
\toprule
\textbf{Especie}     & \textbf{Tamaño} & \textbf{Valor}\\
\midrule
Amphiprion Ocellaris & 10 cm           & \$ 6.000 \\
Hepatus Blue Tang    & 15 cm           & \$ 7.000 \\
Zebrasoma Xanthurus  & 12 cm           & \$ 6.800 \\
\bottomrule
\hline
\end{tabular}
\label{tab:peces}
\end{table}
```

TABLA 2.1. caption largo más descriptivo.

Especie	**Tamaño**	**Valor**
Amphiprion Ocellaris	10 cm	$ 6.000
Hepatus Blue Tang	15 cm	$ 7.000
Zebrasoma Xanthurus	12 cm	$ 6.800

En cada capítulo se debe reiniciar el número de conteo de las figuras y las tablas, por ejemplo, figura 2.1 o tabla 2.1, pero no se debe reiniciar el conteo en cada sección. Por suerte la plantilla se encarga de esto por nosotros.

2.1.5. Ecuaciones

Al insertar ecuaciones en la memoria dentro de un entorno *equation*, éstas se numeran en forma automática y se pueden referir al igual que como se hace con las figuras y tablas, por ejemplo ver la ecuación 2.1.

$$ds^2 = c^2 dt^2 \left(\frac{d\sigma^2}{1 - k\sigma^2} + \sigma^2 \left[d\theta^2 + \sin^2 \theta d\phi^2 \right] \right) \tag{2.1}$$

Es importante tener presente que si bien las ecuaciones pueden ser referidas por su número, también es correcto utilizar los dos puntos, como por ejemplo "la expresión matemática que describe este comportamiento es la siguiente:"

$$\frac{\hbar^2}{2m} \nabla^2 \Psi + V(\mathbf{r}) \Psi = -i\hbar \frac{\partial \Psi}{\partial t} \tag{2.2}$$

Para generar la ecuación 2.1 se utilizó el siguiente código:

```
\begin{equation}
\label{eq:metric}
```

```
ds^2 = c^2 dt^2 \left( \frac{d\sigma^2}{1-k\sigma^2} +
\sigma^2\left[ d\theta^2 +
\sin^2\theta d\phi^2 \right] \right)
\end{equation}
```

Y para la ecuación 2.2:

```
\begin{equation}
\label{eq:schrodinger}
\frac{\hbar^2}{2m}\nabla^2\Psi + V(\mathbf{r})\Psi =
-i\hbar \frac{\partial\Psi}{\partial t}
\end{equation}
```

Capítulo 3

Diseño e implementación

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

3.1. Análisis del software

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epígrafe descriptivo"]
las líneas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
#define MAX_SENSOR_NUMBER 3
#define MAX_ALARM_NUMBER  6
#define MAX_ACTUATOR_NUMBER 6

uint32_t sensorValue[MAX_SENSOR_NUMBER];
FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
state_t alarmState[MAX_ALARM_NUMBER];           //ON or OFF
state_t actuatorState[MAX_ACTUATOR_NUMBER];     //ON or OFF

void vControl() {

  initGlobalVariables();

  period = 500 ms;

  while(1) {

    ticks = xTaskGetTickCount();

    updateSensors();

    updateAlarms();

    controlActuators();

    vTaskDelayUntil(&ticks, period);
  }
```

```
28 }
```

CÓDIGO 3.1. Pseudocódigo del lazo principal de control.

Capítulo 4

Ensayos y resultados

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

4.1. Pruebas funcionales del hardware

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.

Capítulo 5

Conclusiones

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

En esta sección no se deben incluir ni tablas ni gráficos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.