SISTEMAS EMBEBIDOS

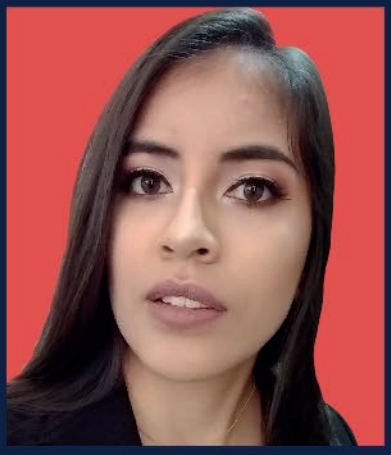
Guía metodológica para su desarrollo

JORGE LUIS ALVA ALARCÓN
NATALI FIORELLA ALCORTA SANTISTEBAN

FONDO EDITORIAL DE LA UNIVERSIDAD PRIVADA ANTENOR ORREGO

JORGE LUIS ALVA ALARCÓN

Maestro en Tecnología de la Información en Fabricación, por la Universidad Politécnica de Madrid, con especialidad en Robótica. Maestro en Ingeniería Eléctrica, Electrónica y Automática, por la Universidad Carlos III de Madrid. Ingeniero Electrónico, por la Universidad Privada Antenor Orrego. Experiencia como docente en la Universidad Carlos III de Madrid y en empresas españolas. Docente universitario y responsable del proceso de I+D+i en el Programa de Ingeniería Electrónica de la Universidad Privada Antenor. Desarrolla proyectos de investigación en las áreas de IoT, robótica e Inteligencia Artificial.



NATALI FIORELLA ALCORTA SANTISTEBAN

Investigadora en el LABINM de Robótica de la Universidad Privada Antenor Orrego. Bachiller de Ingeniería Electrónica, por la Universidad Privada Antenor Orrego. Experiencia laboral en las empresas ECOSAC (Piura) y SiderPerú (Chimbote). Desarrolla proyectos de investigación en las áreas de bio-ingeniería, sistemas embebidos, innovación tecnológica, inteligencia artificial y Big Data.

Jorge Luis Alva Alarcón
Natali Fiorella Alcorta Santisteban

SISTEMAS EMBEBIDOS

Guía metodológica para su desarrollo

FONDO EDITORIAL DE LA UNIVERSIDAD PRIVADA ANTENOR ORREGO

SISTEMAS EMBEBIDOS
Guía metodológica para su desarrollo

Editado por:
© UNIVERSIDAD PRIVADA ANTENOR ORREGO
Av. América Sur N° 3145,
Urb Monserrate, Trujillo, Perú
Teléfono (51) 44 604444, anexo 2087
wwww.upao.edu.pe

Primera edición, versión digital, octubre 2020.
ISBN N° 978-612-4479-13-7

DEDICATORIAS

Con cariño y gratitud a mis padres Ruperto Alva y Pochita Alarcón, a mis hermanos Ana y Pepe, a mis hermanos políticos Moisés y Gina y a mi gran sobrina Ana Sofía.

También va dedicado a mis amigos, colegas docentes, estudiantes y a todos con quienes comparto la pasión por la investigación.

Jorge Luis Alva Alarcón

Dedico este libro a mis papás, a mis hermanas y especialmente a mi abuelita JeshuLinda, por enseñarme que la vida depende de la actitud de cada uno y que una sonrisa puede embellecer el día de cualquier persona.

Asimismo, se lo dedico a mi enamorado, por motivarme a llevar a cabo este libro.

Natali Fiorella Alcorta Santisteban

CONTENIDO

AGRADECIMIENTOS

A Dios, por bendecirnos con cada una de las oportunidades que nos da para superarnos y ser mejores en todos los ámbitos de nuestras vidas.

Nuestro agradecimiento a la Mg. Abg. Ana María Alva Alarcón, quién nos regaló parte de su tiempo contribuyendo con valiosos aportes en la redacción de este libro.

Gracias al Departamento de Investigación de la Universidad Privada Antenor Orrego, especialmente al Dr. Fredy Pérez, por promover e incentivar a los estudiantes a que realicen proyectos de investigación, brindar cursos y ponencias dirigidas a las personas interesadas en los diversos campos de la investigación.

Gracias a nuestro maestro Mg. Ing. Filiberto Azabache por la confianza puesta en nosotros y su guía durante nuestros estudios de Ingeniería Electrónica.

Gracias a la Línea de Automatización y Robótica - LABINM, en particular al Dr. Sixto Ricardo Prado, por confiar en nosotros, permitirnos formar parte de su equipo de investigación y desenvolvernos en el área que más nos apasiona, la investigación. Por sus enseñanzas y críticas constructivas, las cuales nos motivan a seguir con nuestros proyectos. Gracias también a nuestros colegas de laboratorio, puesto que sin ellos los proyectos no se podrían completar satisfactoriamente.

Gracias a la Editorial de la Universidad Privada Antenor Orrego, por permitirnos publicar nuestro primer libro. Por el tiempo dedicado para su revisión y edición.

PRÓLOGO

Este libro va dirigido a todas personas que tomamos la decisión de estudiar una carrera entregada a la tecnología. Asumimos el riesgo de dedicarnos de lleno al estudio constante puesto que, una carrera tecnológica avanza a pasos agigantados, más rápido que un abrir y cerrar de ojos, más rápido que el aleteo de un colibrí, más rápido de lo que uno se pueda imaginar. Y tenemos que ir a la par, a su velocidad, leer y practicar muchísimo, dedicarle tiempo, por no decir una vida entera.

Una vez que se empiezan a materializar las ideas que tenemos en mente, ya no hay marcha atrás. Cuando sucumbimos a la tentación de seguir construyendo e inventando, cuando vemos que nuestros proyectos cobran vida ante nuestros ojos y empiezan a moverse, cuando se hace realidad lo que se había planeado... ya no hay marcha atrás. La emoción de poder conseguir que un proyecto se logre es única, luego de haber dedicado noches y días completos con la adrenalina a flor de piel debido a que, uno puede realizar varias pruebas, incontables pruebas, hasta quemar el último LED, integrado o el mismísimo µC. Y es cuando se puede afirmar que somos unos apasionados por la tecnología.

Poder trabajar en un laboratorio de investigación me ha ampliado el campo de visión con respecto a todas las puertas que se abren cuando uno se atreve a investigar, en cualquier rama de la tecnología no importa cuál, al fin y al cabo todas desembocan en lo mismo, *crear nueva tecnología*. Asimismo, el entorno de trabajo en el cual se vive ayuda a reafirmar aquella decisión tomada al inicio de todo.

Sin embargo, este sentimiento de logro no es exclusivamente de los investigadores, sino que se empieza a vivir desde el primer ciclo de la carrera gracias a los docentes que nos motivan a seguir como es el caso del Ing. Jorge Alva, con quien es un honor compartir la autoría de este libro, que nos enseña con dedicación, que nos transmite con generosidad sus conocimientos fruto de su amplia trayectoria laboral y formación profesional en universidades peruanas y extranjeras, que guía nuestro paso universitario, pero sobre todo, que nos motiva a ser investigadores ya sea desde las aulas y nos alienta a no desfallecer en el intento; ahí es cuando uno se puede dar cuenta que gracias a la carrera tecnológica elegida

y a nuestros esmerados profesores, se pueden llegar conseguir cosas increíbles.

Por otro lado, del Dr. Prado mi mentor, aprendí que la investigación no es para cualquiera, puesto que es una carrera de resistencia y no de velocidad.

La investigación exige que uno se comprometa con la ciencia y la tecnología (este par siempre va entrelazado), pero sobre todo ser perseverante. Habrá incontables veces que caeremos, que no logremos los objetivos planteados pero esto no será impedimento de seguir con nuestros sueños.

Fiorella Alcorta

INTRODUCCIÓN

El presente libro está enfocado principalmente a guiar al estudiante y/o docente interesado en el desarrollo de proyectos electrónicos dentro del ámbito universitario, centrado en el "Internet de las Cosas" o más comúnmente conocido como IoT; incorpora dos metodologías, de las cuales, una está basada en proyectos y la otra en *Kanban*.

Las fases que comprende la evolución de un proyecto, abarcan la idea de proyecto, trabajo grupal, diagrama de bloques, organización de las tareas, presentación en un cronograma, obtención de datos, guardado y análisis de dichos datos, brindando ejemplos en diversas etapas y haciendo comparaciones tecnológicas para que el lector pueda escoger la que más le convenga.

La dilatada experiencia adquirida durante años de trabajo en proyectos internacionales, en el dictado de clases universitarias y la misma experiencia como estudiante universitario, ha permitido plasmar en este libro las ideas y métodos que guiarán al lector paso a paso, dando ejemplos que ayudarán a moldear correctamente su proyecto y ajustarlo a los tiempos que se requieren en un entorno universitario desde el principio del ciclo universitario.

En este libro se pone énfasis en el uso de microcontroladores y microprocesadores y en su interconexión tanto entre sensores y controladores, dando una visión sobre el crecimiento de la tecnología a nivel mundial en el ámbito de los circuitos electrónicos aplicados.

En el capítulo 1 se plantean las consideraciones que se deben observar al tener una idea de proyecto, al diagramar dicha idea, al escoger al equipo de trabajo, al escoger los tiempos para las tareas, así como, se proponen herramientas que se pueden utilizar. También, se refuerzan los métodos que se utilizan al ejecutar un proyecto y, por último, se presentan ejemplos de los temas tratados en este capítulo.

En el capítulo 2 se muestran los microcontroladores, sus usos, modelos de las placas, lenguajes de programación que se pueden utilizar, entornos de desarrollo y herramientas factibles de emplear para su configuración y programación.

En el capítulo 3 se muestran las computadoras de placa única, los sistemas operativos que soportan y las formas de conectar dichas computadoras con los diferentes modelos de microcontroladores.

En el capítulo 4 se guía al lector para conocer la importancia y el uso a nivel mundial de los sistemas electrónicos dentro del ecosistema del Internet de las cosas.

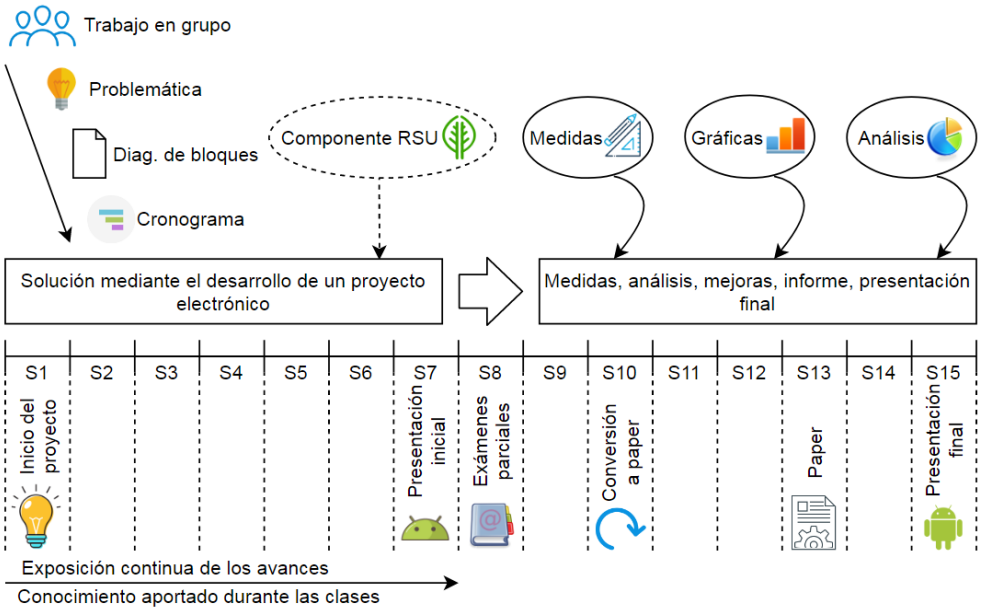
En el capítulo 5 se muestra de forma global el proceso de obtención de los datos, su procesamiento y su análisis mediante los sistemas electrónicos embebidos, y finaliza con ejemplos de proyectos que se desarrollaron por estudiantes universitarios.

CAPÍTULO 1.
CONCEPCIÓN DEL PROYECTO ELECTRÓNICO

En este capítulo se describe una de las formas con las que se puede empezar con el desarrollo de algún proyecto electrónico dirigido a IoT, con el objetivo que la transición entre "idea inicial" y "prototipo funcional" sea sencilla y entretenida. Para esto, a lo largo del libro, se intercala teoría con ejercicios prácticos a seguir.

Durante las clases universitarias, muchas veces se tiene la "obligación" de desarrollar un proyecto electrónico, siendo dicha "obligación" producto de las exigencias del docente o de la materia que se está cursando; así, en este capítulo vamos a guiar tanto al estudiante como al docente durante el transcurso de todo el proceso que comprende la evolución del proyecto.

Figura 1. Diagrama del desarrollo de un proyecto electrónico dentro del ambiente universitario



La figura 1 servirá de guía durante el desarrollo del proyecto, puesto que permite conocer todo el proceso de evolución, para posteriormente ayudar a saber en qué fase del progreso del proyecto se encuentra. Además se muestran los pasos a seguir, desde la generación de la idea del proyecto hasta la culminación del mismo en donde se realizan análisis de las gráficas obtenidas y se presenta un informe con formato de artículo científico. Asimismo, la gráfica mostrada con anterioridad les guiará por el siguiente camino:

1. Reunión del equipo de trabajo.
2. Planteamiento de la problemática a solucionar.
3. Presentación un diagrama de bloques de la solución propuesta.
4. Estimación de los tiempos en los que se llevará a cabo cada una de las fases del proyecto.
5. Toma de mediciones.
6. Graficar los datos obtenidos a partir de las mediciones.
7. Análisis de los datos recolectados.
8. Considerar que el proyecto tenga un componente de RSU (Responsabilidad Social Universitaria).
9. Informe del proyecto, conversión a artículo científico y presentación final del informe en formato de artículo científico (también conocido como *paper*).

Para este tipo de proyectos, se recomienda, que para desarrollarlo, de las catorce semanas que dura en promedio el dictado de clases de un ciclo universitario, en las primeras siete semanas se plantee implementar el sistema principal o *core* del proyecto, posteriormente, durante las siete semanas restantes, automatizar la toma de medidas, realizar el análisis de los datos y, por último, completar el informe final.

1.1. PROBLEMÁTICA Y CONDICIONES

Antes de iniciar cualquier proyecto, ya sea un proyecto electrónico o de otro tipo, primero se deben formular algunas preguntas, tales como las siguientes:

1. ¿Aún no tengo en mente lo que desarrollaré?, ¿sostengo la idea de lo que haré?, o ¿sé lo que quiero hacer?
2. ¿Qué problemática ansío resolver?
3. ¿Qué medidas o gráficas realizaré para demostrar que he resuelto dicha problemática?
4. ¿Qué insumos, herramientas e instrumentos voy a requerir para ejecutar el proyecto?
5. ¿Cuánto tiempo (semanas o meses) dispongo para concretar el proyecto?

Es muy importante conocer todas las condiciones posibles que rodean al proyecto, y si se puede escoger alguna, entonces se hará tratando de limitarla correctamente para no caer en extremos como podría ser un proyecto muy básico únicamente para encender un LED o un proyecto fuera del alcance de este libro como sería desarrollar un cohete a Marte.

1.1.1. La idea

Existen varios libros que tratan este tema de manera más detallada, pero se presenta este subtema para dar a conocer algunas ideas de cómo afrontar las interrogantes que trae consigo la creación del proyecto.

Teniendo en cuenta que muchos estudiantes universitarios sufren con la pregunta de ¿qué proyecto voy a desarrollar? Y, por simple que parezca esta interrogante, es fundamental aclararla cuidadosamente, debido a que de la respuesta dependerá todo el planteamiento y ejecución del proyecto.

En primer lugar, tratar de recordar por qué está estudiando una carrera universitaria, o por qué quiere hacer un proyecto. Muchas personas tienen objetivos ocultos que quizás han olvidado, quizás tienen un familiar a quien se quiere ayudar a partir de lo que se va a desarrollar en base a lo estudiado durante el periodo universitario; o se tiene una vocación por la ciencia y se desea saber el porqué de los fenómenos naturales. Dese un tiempo y procure recordar qué lo impulsó a usted.

En segundo lugar, se aconseja que el estudiante y/o investigador realice una lista con todos los proyectos electrónicos que haya materializado hasta el momento, no importa si el proyecto fue a su parecer muy simple. Se deberá agregar un breve resumen del mismo, de esta manera uno será consiente de la capacidad y del progreso que se ha tenido.

En tercer lugar, averiguar qué es lo que le gusta más, por ejemplo, ¿disfruta de trabajar con ondas electromagnéticas o con microcontroladores, programar, diseñar y/o implementar circuitos electrónicos? En base a su respuesta irá averiguando sus propios gustos. En el caso que le impongan un tipo específico de proyecto, lo mejor es buscar qué parte o área le agrada más de dicho proyecto y aprovecharlo como fuerza motivadora, puesto que es más fácil hacer horas extras, leer, investigar o trasnocharse haciendo algo que le gusta y le haga feliz.

En cuarto lugar, los proyectos que realice deben estar enfocados en solucionar un problema y al final del desarrollo del proyecto debe demostrar que ha podido solucionar dicha problemática.

Finalmente, si la idea es muy ambiciosa, involucrando mucho tiempo o recursos, entonces trate de disgregarla y quedarse con el trozo del proyecto que sí pueda realizar en un periodo de tiempo limitado.

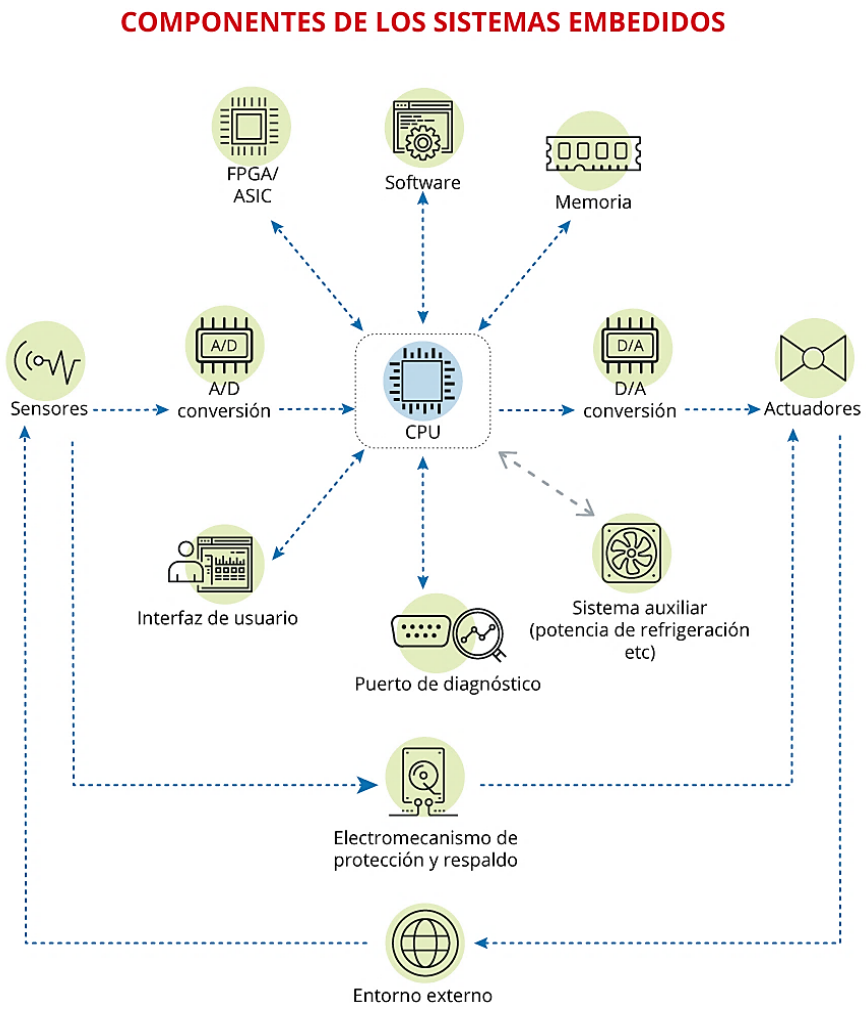
1.2. TEORÍA SOBRE SISTEMAS EMBEBIDOS

Los sistemas embebidos se definen como un conjunto de componentes electrónicos asociados para realizar funciones dedicadas. En pocas palabras, un sistema embebido se diferencia de otro tipo de sistemas principalmente por poseer un procesador central, como se aprecia en la figura 2, el cual es el encargado de recibir, analizar y procesar los datos admitidos por los sensores para, posteriormente, enviar una señal a los actuadores y que éstos realicen la función o funciones específicas requeridas.

Para comprender mejor qué son los sistemas embebidos, a continuación se listan sus características.

- Tienen un procesador central. Los módulos centrales de procesamiento pueden estar formados por microcontroladores, FPGAS, microprocesadores o una mezcla de los anteriores, principalmente.
- Son sistemas confiables.
- Requieren poco o nulo mantenimiento. En su gran mayoría no exigen mantenimiento dedicado.
- Son sistemas seguros. Tienen protección del software incrustado, utilizan protocolos encriptados, se diseñan para ser dispositivos *Anti – Hacking*. Aunque el gran auge de estos sistemas y su uso masivo y muchas veces descuidado está abriendo una brecha en su seguridad.
- Son eficientes en cuanto al consumo de energía.
- Son de propósito específico.
- Tienen interfaz de usuario simple o carecen de ella. Algunos de ellos cuentan con interfaz gráfica de usuario lo cual facilita su programación, configuración y/o control. Cabe resaltar que la interfaz de usuario no es de uso general (como lo pueden ser las interfaces de usuario de una computadora personal), sino, son de uso específico (como el teclado de un cajero automático, por ejemplo).

Figura 2.Componentes de los sistemas embebidos



Fuente (Incibe-Cert, 2018)

1.3. DIAGRAMA DE BLOQUES

Un diagrama de bloques es la representación gráfica de la relación y/o interconexión entre las variables de un sistema o proceso. Se emplean en la primera etapa del diseño y/o concepción del proyecto que se desea llevar a cabo.

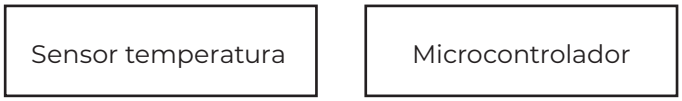
No existe un estándar internacional para realizar un diagrama de bloques, ya que representa una idea que se quiere transmitir. Se sabrá si el diagrama es correcto, cuando al mirarlo se entienda rápidamente el funcionamiento del proyecto con el suficiente detalle pero sin llegar a confundir. Una forma de verificar su comprensión es compartiendo el diagrama con otra persona y pedirle que explique lo que entiende, si coincide con la idea fundamental quiere decir que está bien, caso contrario, habrá que revisarlo para ver en qué parte se ha fallado.

1.3.1. ¿Cómo hacer diagramas de bloques?

Inicialmente se debe poner en rectángulos una palabra o palabras que describan a groso modo cada uno de los módulos del proyecto.

a. Suponiendo que se pretende medir la temperatura de un sistema, entonces se necesita un sensor de temperatura. En este punto no es prioritario preocuparse de cuál será dicho sensor (ni su rango, resolución o precio), lo que se hará es ponerlo tal cual en una caja de la siguiente manera:

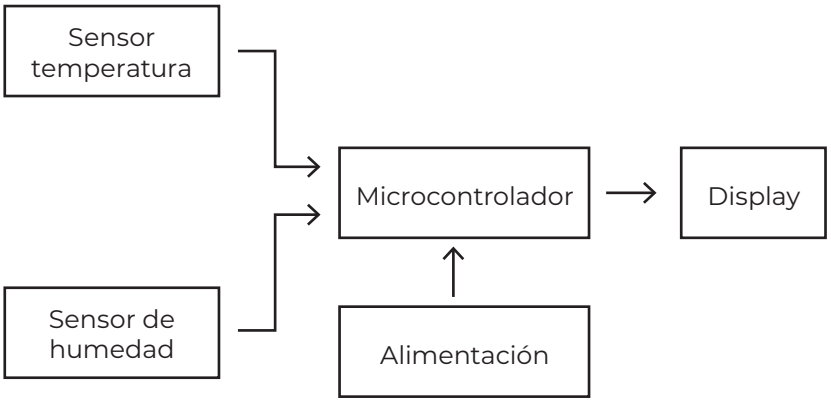
b. En este punto se necesita una unidad central que procese dicho valor, por lo que se pensará en utilizar un microcontrolador. Tampoco es necesario saber cuál microcontrolador se utilizará (ni su velocidad, pines de entrada/salida o su precio), lo que se hará es también ponerlo en una caja de la siguiente manera:



c. Teniendo el sensor y el microcontrolador, se hará la siguiente pregunta: ¿de dónde es emitida la señal y hacia dónde viaja? Entonces se responderá con: “la señal emitida por el sensor viaja hacia el microcontrolador” y con esta respuesta se procederá a dibujar una flecha que salga del sensor y que acabe en el casillero del microcontrolador.

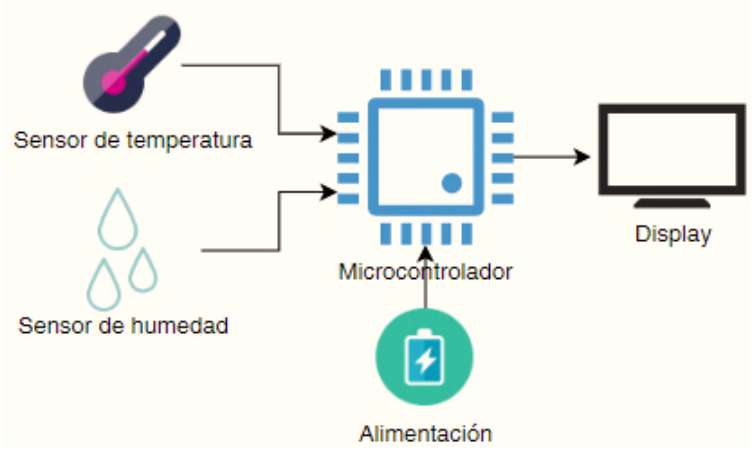
d. También añadimos que el valor de la temperatura medida sea visualizada en un display (o pantalla), e igualmente se dará cuenta que la información viaja desde el microcontrolador hacia el display; de esta forma, se tendrá la figura a continuación:

e. También se podrá resaltar la importancia de una fuente de alimentación para el microcontrolador, y así como se mide la temperatura también se desea medir la humedad, por lo que el diagrama quedaría así:



f. Dependiendo también de los gustos, como se observa en la figura 3 es procedente incluir íconos que representen lo que se quiere mostrar, pero es importante agregar el texto de lo que representa o significa cada uno de los bloques.

Figura 3. Diagrama de bloques de un sistema que mide temperatura y humedad



Como se va mostrando, el proceso para crear el diagrama de bloques de un sistema es una tarea que avanza paso a paso ordenadamente y que concluye en un diagrama que representa de una forma clara y concisa qué es lo que se desea elaborar.

Estos diagramas son utilizables al principio de los informes de los proyectos ya que permiten introducir al lector la idea que se desea transmitir. La forma, tamaño, colores y detalles de cada diagrama de bloques queda limitado solamente por la imaginación de los integrantes del equipo.

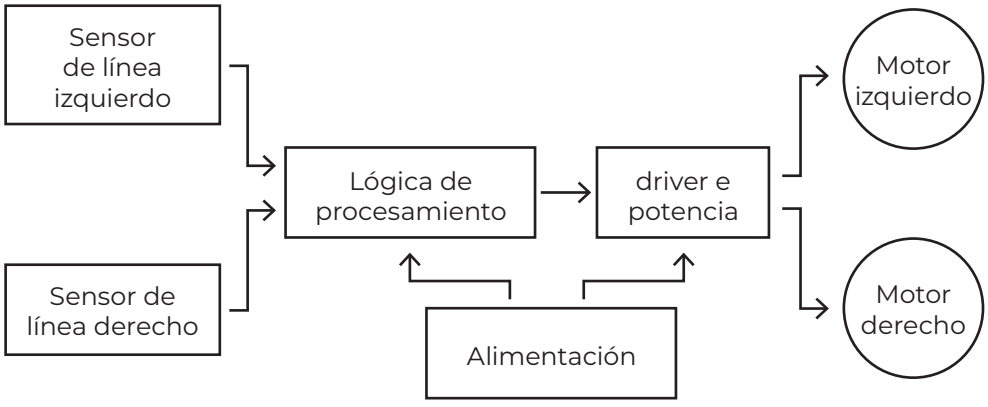
1.3.2. Ejemplos de diagramas de bloques

En este apartado, se mostrarán ejemplos de diagramas de bloques y se comentarán, así se tendrá una mejor idea de cómo hacer uno propio.

Diagrama de bloques de un seguidor de línea

Un seguidor de línea es el *"hola mundo"* de la robótica. Este es un robot muy sencillo y cuya única función es seguir una línea marcada en el suelo, esta línea es generalmente una línea negra de 2 cm de grosor y está pintada sobre un fondo blanco. Existen varias estrategias para construir un seguidor de línea, pero en esencia el robot debe saber si se encuentra a la derecha o a la izquierda de la línea para girar, y si es que se requiere, en sentido contrario. Así, como mínimo se necesitan dos sensores para detectar si se ubica al lado derecho o al lado izquierdo de la línea. En la figura 4 se puede observar el diagrama de bloques de un seguidor de línea.

Figura 4.Diagrama de bloques de un carrito seguidor de línea



El diagrama de bloques del seguidor de línea indica que tiene solamente dos sensores (de acuerdo a este diseño, no es obligatorio usar únicamente dos), un sensor izquierdo y uno derecho, ambos sensores envían su información hacia una lógica de procesamiento y esta lógica de procesamiento envía su resultado hacia un driver de potencia que controla el giro de los dos

motores: el motor de la izquierda y el motor de la derecha. En el diagrama también se puntualiza qué tanto la lógica de procesamiento como el driver de potencia, serán alimentados directamente; muchas veces no se especifica la alimentación, pero este criterio depende del autor.

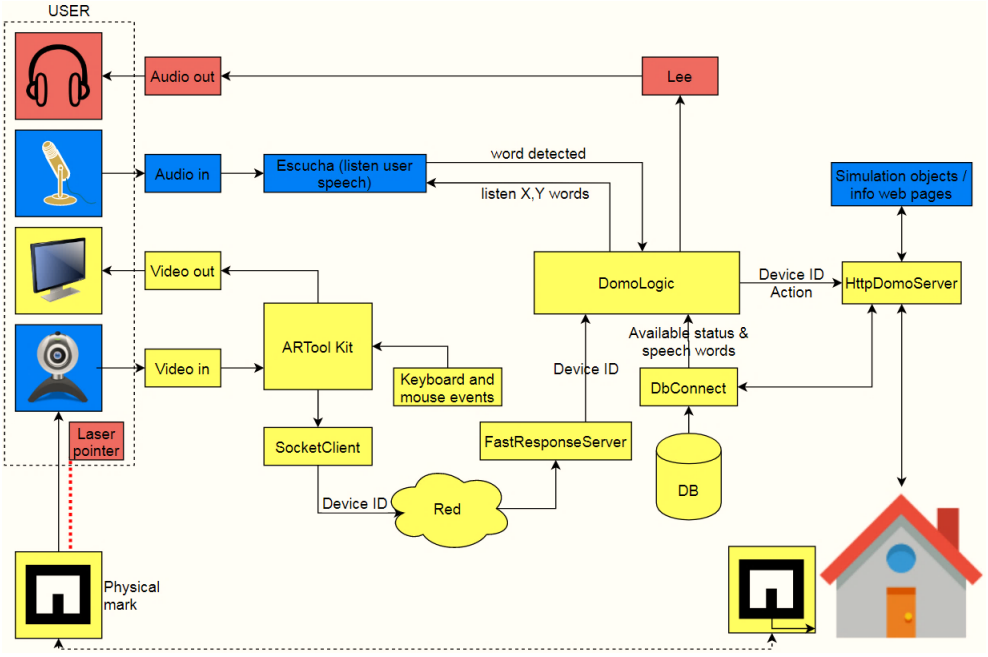
En este diagrama no se detalla cómo funciona internamente la lógica de procesamiento, pero se puede intuir que los sensores de luz detectan si el carrito se encuentra o no dentro de un camino y así accionar el motor derecho o izquierdo para corregir la ruta.

Diagrama de bloques de un sistema domótico para el control de una vivienda

La domótica comprende aquellos sistemas encargados de automatizar las tareas de un hogar o de un edificio. Aportan servicios de seguridad, gestión energética, bienestar y comunicación, integrando redes de comunicación interiores y de exteriores. Proviene de dos palabras *domus*, que significa 'casa', y a*utónomo,* que significa 'así mismo' (Wikipedia: Domótica, 2020).

A medida que se van viendo los diagramas de bloques de diferentes proyectos, se irá dando cuenta que existen infinidad de formas y arreglos que se pueden realizar con el fin de expresar una idea. Por ejemplo, en la figura 5 se tiene el sistema de control de entorno de una casa domótica.

Figura 5. Diagrama de bloques de un sistema de control de entorno domótico

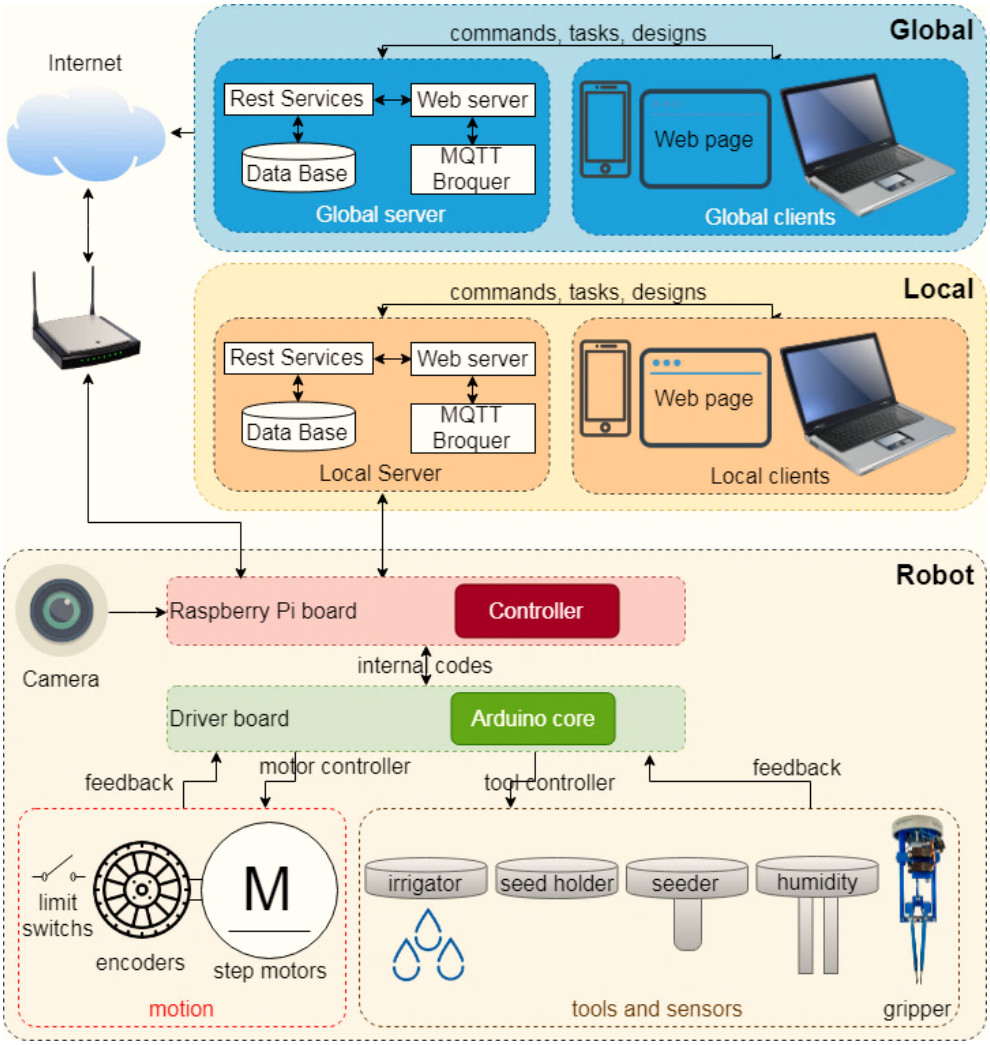


Fuente (Alva-Alarcón & Jardón-Huete, 2018)

Diagrama de bloques de un huerto automatizado mediante IoT

Los diagramas de bloques también pueden mostrar relaciones con sistemas robóticos e interconexión entre sus componentes como el que se puede ver en la figura 6.

Figura 6. Diagrama de bloques de un huerto automatizado usando IoT en un entorno doméstico



Fuente (Choque M., Linares O., Alcorta S., Alva A., & Prado G., 2019)

Como se puede observar, se utilizan bloques rectangulares para expresar una idea o incluir una gama de figuras y colores que ayuden a mejorar la compresión del proyecto.

1.3.3. Herramientas online / offline para crear diagramas de bloques

Existen diversas herramientas para producir diagramas de bloques, algunas necesitan licencia, otras, son de uso libre y funcionan bajo diferentes plataformas, como Windows en sus diferentes versiones o Linux con sus múltiples ramas.

• Draw.io

Es un aplicativo diseñado para construir diagramas. Posee una aplicación gratuita para ser utilizada por Web. Se puede utilizar en cualquiera de los navegadores existentes como: Firefox, Chrome, etc. entrando a la página web https://www.draw.io/. Este aplicativo es uno de los más utilizados en el mundo y es capaz de integrarse con muchos sitios web como por ejemplo con Google Drive, OneDrive (Seibert Media Corp., 2020).

• Power Point

Es un programa de uso común, principalmente para realizar presentaciones, pero también se puede aprovechar para efectuar diagramas de bloques ya que su funcionamiento lo hace fácil de utilizar para este propósito. Es un software de pago y está desarrollado para funcionar sobre plataformas Windows, aunque tiene la versión online Microsoft 360 que permite trabajar las diapositivas utilizando un navegador web (Microsoft Corporation, 2020).

• Paint

Aunque es un programa básico para dibujar, es uno de los que más se emplea puesto que viene incorporado en el sistema operativo Windows. Como recomendación se puede utilizar para pequeños diagramas de bloques, pero no para diagramas más complejos debido a que es algo limitado.

• Dia

Es un aplicativo de uso gratuito bajo licencia GPLv2 (General Public License), desarrollada originalmente como parte del proyecto GNOME. Tiene versiones para Windows, Linux y MacOS. Este software posibilita la creación de diagramas de entidad-relación, diagramas UML, diagramas de redes, diagramas de flujo, diagramas de circuitos eléctricos, entre otros (The Dia Developers, 2014).

1.4. DIAGRAMA DE GANTT (CRONOGRAMA)

Un diagrama de Gantt es una herramienta gráfica que permite visualizar el tiempo que se dedicará a diversas tareas o actividades a lo largo de un tiempo total determinado. Permite planificar, programar, realizar el seguimiento y el control del progreso de cada una de las etapas de un proyecto; asimismo, reproduce gráficamente las tareas, su duración y secuencia, adicionalmente al calendario general del proyecto.

El gráfico del diagrama de Gantt es, en realidad, un sistema de coordenadas con dos ejes esenciales: en el eje vertical se ubican las tareas a realizar desde el inicio hasta el fin del proyecto, mientras en el horizontal se ponen los tiempos. En función del tipo de actividades que conformen el proyecto, los valores ubicados en el eje horizontal deben definirse en días, semanas, meses, semestres o, incluso, años (OBS Business School, 2018).

1.4.1. Estimaciones relativas de tiempo

Antes de dar los pasos para realizar un diagrama de Gantt, es necesario saber cómo elaborar una estimación lo más aproximada del tiempo que demorará hacer una tarea. Por ejemplo, ¿cuánto tiempo podría tardar caminar 10 metros?, algunas personas podrán opinar que entre unos 8 a 12 segundos; es fácil dar una estimación para una distancia corta. No obstante, si se vuelve a realizar la misma pregunta pero con una distancia mucho más larga (1 Km), la estimación sería más difícil de efectuar. Si se piensa cuánto tiempo podría demorar en caminar 100 km no se conseguiría elaborar con una regla de tres simple, debido a que se tendría que tener en cuenta nuevos factores, como por ejemplo: si el recorrido es de subida o bajada, a qué altura del mar está el camino, si es peligroso o no, entre otros.

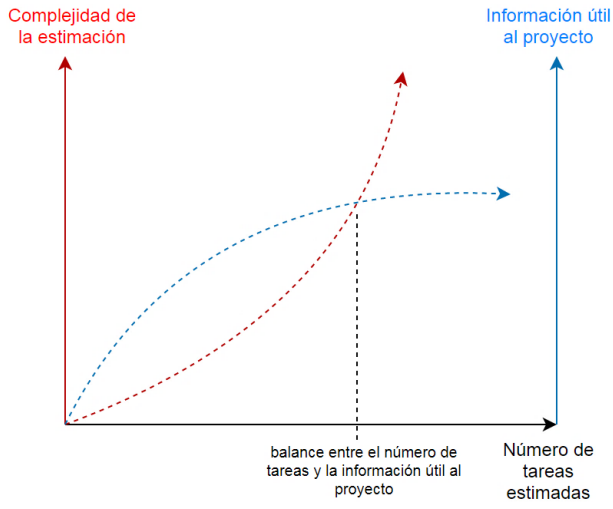
1.4.2. Desglosar las tareas

Cuando se plantea un proyecto se procurará dividirlo en las tareas que se puedan estimar. Si se ve que una tarea es muy genérica y con un tiempo difícil de estimar entonces se desglosará dicha tarea en otras más pequeñas.

Se procurará no desmenuzar tanto el proyecto haciendo una lista muy larga de tareas, ya que se podría utilizar más tiempo en su planificación y estimación que el tiempo que se necesitaría para su ejecución.

En la figura 7 se representan dos curvas que muestran la relación entre la complejidad de estimación de las tareas y la información útil que se puede dar al proyecto. Es importante encontrar el número adecuado de tareas que no sea tan complejo de estimar, pero que den información importante de qué se debe hacer.

Figura 7. Balance entre el número de tareas estimadas y su información útil.



Para entender el efecto producto del análisis, se verá un proyecto como ejemplo. Para esto se planifican las tareas necesarias para realizar un carrito seguidor de línea. En la tabla 1 se propone inicialmente una sola tarea.

Tabla 1. Lista de tareas resumida

Tarea	Duración
Implementar un carrito seguidor de línea	2 semanas

En la tabla anterior la tarea propuesta coincide con lo solicitado, pero la planificación de la tarea no está aportando información al trabajo. Se puede mejorar la lista de actividades desglosando la tarea en subtareas como se muestra en la tabla 2.

Tabla 2.Lista de tarea desglosada

Tarea	Duración
Informe inicial	2 días
Implementación del prototipo	10 días

En la tabla anterior se ha desglosado la tarea principal; sin embargo, aún no es suficiente información de la cual guiarse. Por lo que, se debe tomar la molestia de dividir aún más la lista de tareas.

Tabla 3. Lista de tareas más desglosada

Tarea	Duración
Desarrollo y presentación del plan del proyecto	2 días
Selección y compra de los componentes	3 días
Construcción del chasis del carrito	3 días
Incorporación de los motores y la rueda loca al chasis	1 día
Incorporación de los sensores de luz en el chasis	3 días
Desarrollo de la lógica de procesamiento	6 días
Incorporación de la batería	1 día
Pruebas de funcionamiento	4 días
Presentación final e informe	2 días

En la tabla 3 se muestra de mejor manera la lista de tareas que se deben realizar y permite saber más claramente qué es lo que se debe desarrollar y en cuánto tiempo se realizará. Se puede seguir desglosando las tareas hasta hacerlas muy finas, pero se debe encontrar el balance correcto entre desglosar las tareas y la información que ellas aporten.

1.4.3. Influencia en las estimaciones de tiempo dentro de un equipo de trabajo

Si ya se ha tenido la oportunidad de estar en un equipo de trabajo, se habrá notado que cuando surge la pregunta "¿cuánto demorará hacer esta tarea T?", si el primero que responde es el "líder" del equipo y dice X días, entonces el resto del equipo asiente con la cabeza o dice valores muy similares. Esto ocurre debido la presión del equipo, la cual empuja a dar una opinión muy similar a la que tiene el equipo, lo que impide manifestar la verdadera estimación.

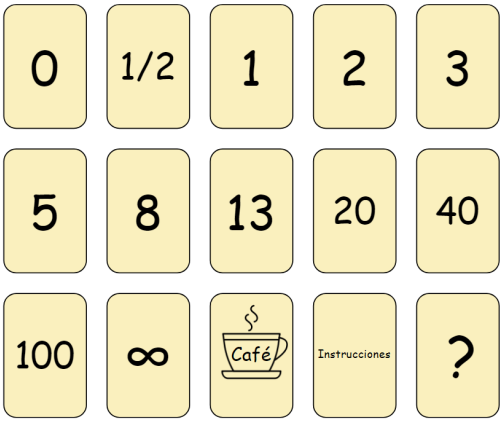
No escuchar a todos los miembros del equipo es un error común y para solucionar el error de no considerar a todos los miembros del equipo existen varias técnicas, una de ellas es el *Planning Poker*.

Implementando dicha técnica, se genera una dinámica ágil en la que se reúne el equipo de trabajo portando cada uno de ellos una baraja de póker modificada y se hacen rondas de estimación con ayuda de estas cartas (Casanova, 2016).

En la figura 8 se pueden ver los números que conforman la baraja del *Plannin Poker*; los números en cada carta representan la cantidad de días u horas que puede conllevar realizar una tarea.

Como curiosidad muchas de estas barajas incluyen una carta de "café" que indica que ya es tiempo de hacer una pausa en las tareas y tomarse un buen café, sin azúcar, que es más saludable.

Figura 8. Planning Poker



El método del *Planning Poker* permite quitarse el miedo de opinar y beneficiarse de la *"sabiduría de la multitud"* o también llamada *"inteligencia colectiva"*.

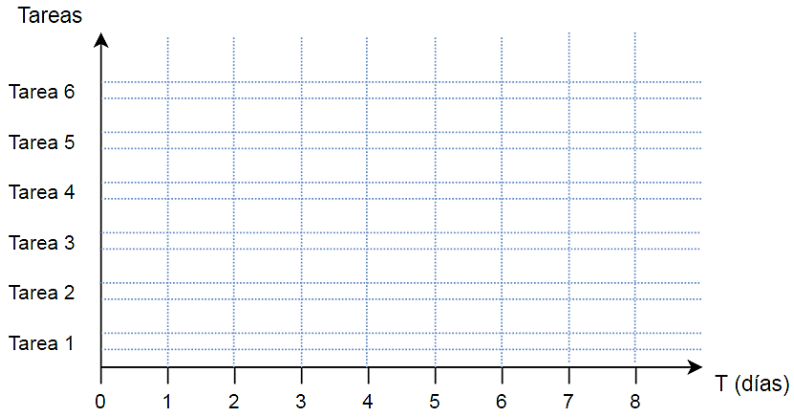
1.4.4. ¿Cómo hacer un diagrama de Gantt?

El objetivo del diagrama de Gantt para un proyecto es el de tener una forma clara y rápida de organizar el tiempo y saber si se va avanzando o no correctamente durante el desarrollo del proyecto.

Se comienza dibujando un plano, cuyo eje "X" será el tiempo, medido en semanas o días (dependiendo de la duración del proyecto) y en el eje "Y" se pondrán las tareas que se piensa desarrollar, preferentemente ordenadas de forma sucesiva.

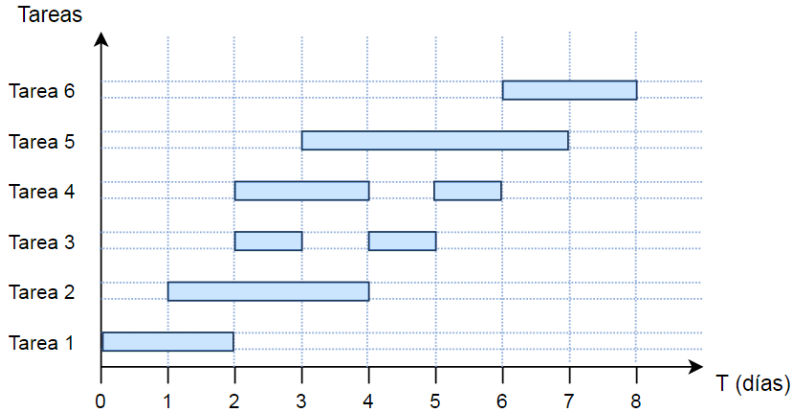
Como se ha visto en el apartado anterior se deben desglosar las tareas del proyecto de modo que no se tengan muy pocas, ya que no permitirían estimar el desarrollo correctamente, o que se tengan demasiadas tareas porque podrían confundir la organización del avance. Para construir el diagrama de Gantt se puede empezar con un plano como el que se observa en la figura 9.

Figura 9. Base gráfica para empezar a desarrollar nuestro diagrama de Gantt



El siguiente paso consiste en ir ubicando el tiempo en el que empieza y termina cada una de las tareas que se han distribuido en el eje "Y". Por ejemplo, se pueden ubicar rectángulos que muestren el inicio y fin de cada tarea como se aprecia en la figura 10.

Figura 10. Diagrama de Gantt de nuestro proyecto



Recuerde que hay muchos estilos para realizar diagramas de Gantt y que las tareas y la organización de las mismas dependen de cada desarrollador.

1.4.5. Ejemplos de diagramas de Gantt para proyectos de Electrónica

Ejemplo 1: Diagrama de Gantt para un carrito seguidor de línea

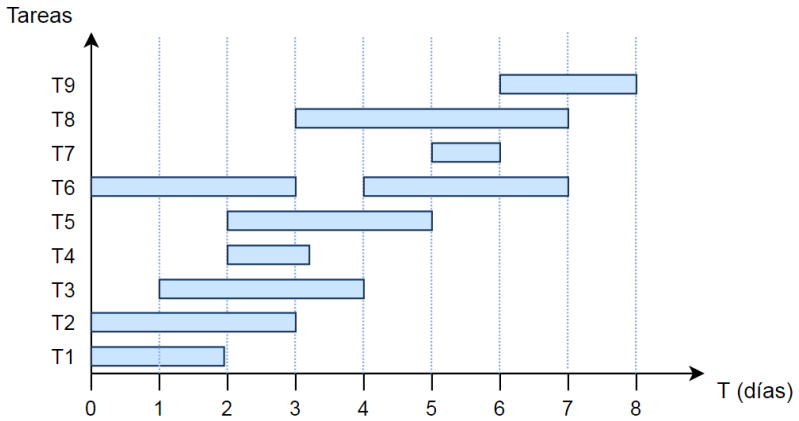
Se pide desarrollar un carrito seguidor de línea, para lo cual se tiene que incluir en el informe un diagrama de Gantt del proyecto. Por lo que se empezará listando las tareas que se deben realizar, como se observa en tabla 4.

Tabla 4.Tabla con las tareas del proyecto a desarrollar

Código	Descripción
T1	Desarrollo y presentación del plan del proyecto
T2	Selección y compra de los componentes
T3	Construcción del chasis del carrito
T4	Incorporación de los motores y la rueda loca al chasis
T5	Incorporación de los sensores de luz en el chasis
T6	Desarrollo de la lógica de procesamiento
T7	Incorporación de la batería
T8	Pruebas de funcionamiento
T9	Presentación final e informe

Una vez que se tenga la lista de las tareas que se deben efectuar se procede a ubicarlas en el diagrama junto con una estimación del inicio y fin de la tarea. La figura 11 muestra el diagrama de Gantt para el desarrollo de un carrito seguidor de línea.

Figura 11. Diagrama de Gantt para el desarrollo de un carrito seguidor de línea



Los tiempos propuestos en la figura 11 dependerán mucho de la experiencia del equipo de trabajo, de cuántos integrantes son, del tipo de lógica que se quiere implementar, del número de sensores de luz se planean utilizar, asimismo si el carrito será un prototipo básico o será un modelo de competencia, etc.

Ejemplo 2: Diagrama de Gantt para el diseño de una estación meteorológica

Se pide desarrollar un sistema que exhiba la temperatura y la humedad en un display, de forma que se obtenga una pequeña estación meteorológica. Por lo tanto, se empezará listando las tareas que se tienen que cumplir, conforme se muestran en la tabla 5.

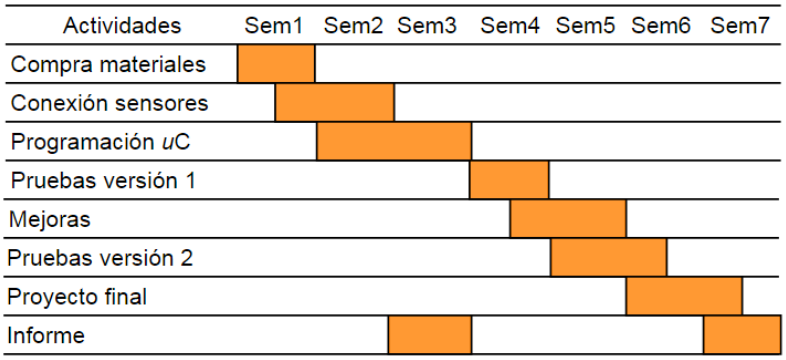
Tabla 5.Tabla con las tareas de nuestra estación meteorológica

Código	Descripción
T1	Compra de materiales
T2	Conexión de los sensores
T3	Programación del microcontrolador
T4	Pruebas de la versión 1
T5	Mejoras
T6	Pruebas de la versión 2
T7	Proyecto final
T8	Informe

Similar al proyecto anterior, una vez se tenga la lista de las tareas a desarrollar, se procederá a ubicarlas en el cronograma. A diferencia del ejemplo anterior, este cronograma empieza de arriba hacia abajo, en vez de abajo hacia arriba; además se planeó para ser acabado en la semana 7.

En la figura 12 se presenta el cronograma para el desarrollo de un proyecto de estación meteorológica.

Figura 12. Diagrama de Gantt para el desarrollo de una estación meteorológica



Conforme se observa, los tiempos y las tareas se escogerán por los miembros del equipo de acuerdo al proyecto que se desarrolla y a las exigencias del mismo, tales como: el número de semanas para la presentación, número de personas del equipo, si los materiales se pueden comprar en la propia ciudad o hay que pedir el envío de estos, si se requiere hacer un informe y un artículo final, etc.

1.5. METODOLOGÍA BASADA EN PROYECTOS

El aprendizaje basado en proyectos es una metodología que permite a los estudiantes adquirir los conocimientos y competencias clave en el siglo XXI, mediante la elaboración de proyectos que dan respuesta a problemas de la vida real. Los estudiantes se convierten en protagonistas de su propio aprendizaje y desarrollan su autonomía y responsabilidad, puesto que son ellos los encargados de planificar, estructurar el trabajo y elaborar el producto para resolver la cuestión planteada. Por otro lado, la labor del docente es guiarlos y apoyarlos a lo largo del proceso, adquiriendo un rol menos activo.

Mediante esta metodología los estudiantes no solo memorizan o recogen información, sino que aprenden haciendo (AulaPlaneta, 2015) (EDUforics, 2017).

En la figura 13 sobre el ciclo del aprendizaje basado en proyectos, se observa cada uno de los pasos que guiarán a los estudiantes en su motivación por aprender gracias al intercambio de ideas, la creatividad y la colaboración.

A lo largo de este libro, se dan pautas que guían tanto a los estudiantes como a los docentes, con el fin que la materialización de los proyectos (principalmente proyectos electrónicos) se realice de manera progresiva, considerando las semanas de aprendizaje de un semestre académico.

Figura 13.Metodología basada en proyectos

Fuente (AulaPlaneta, 2015)

1.5.1. Seguimiento continuo (semana a semana)

En muchos casos, lamentablemente, los estudiantes se concentran en hacer el proyecto del curso en uno o dos días antes de la fecha de entrega, dejando todo el trabajo, que se debería haber realizado durante todo el semestre, para el final. Esta deficiencia se puede corregir haciendo un seguimiento continuo de los proyectos, pidiéndoles a los estudiantes que semana a semana expongan el avance que han tenido, de esta manera se conseguirá que:

- Los estudiantes se preocupen por avanzar semana a semana.
- Haya una realimentación hacia el resto de los equipos que también desarrollan proyectos y puedan ver cómo van avanzando sus compañeros.

Durante esta etapa, el docente debe realizar una ficha para hacer el seguimiento de cada proyecto y evitar que los estudiantes expongan contenido repetido. También es fundamental que el docente ejerza un papel de asesor, interviniendo en el trabajo, guiándolos durante cada una de las semanas (sin excepción), entendiendo los problemas que sufre cada equipo y ayudándoles a resolverlos.

Es pertinente comentar que dentro de nuestra experiencia docente hemos presenciado diversos casos de estudiantes que al momento de tratar de explicar el funcionamiento de su proyecto no sabían comunicar la idea central del mismo. Esto se debe a que muchas veces no sabemos expresarnos cuando queremos transmitir un mensaje; y hay que tener en cuenta que es muy importante saber hacerlo, utilizando un lenguaje sencillo, pero a la vez claro y apropiado, más aún, tratándose de proyectos de investigación donde el investigador debe saber explicar en qué consiste el mismo.

En las siguientes figuras se comparte una ficha para la revisión semanal de los proyectos. Esta ficha puede ser utilizada tanto por docentes como por estudiantes.

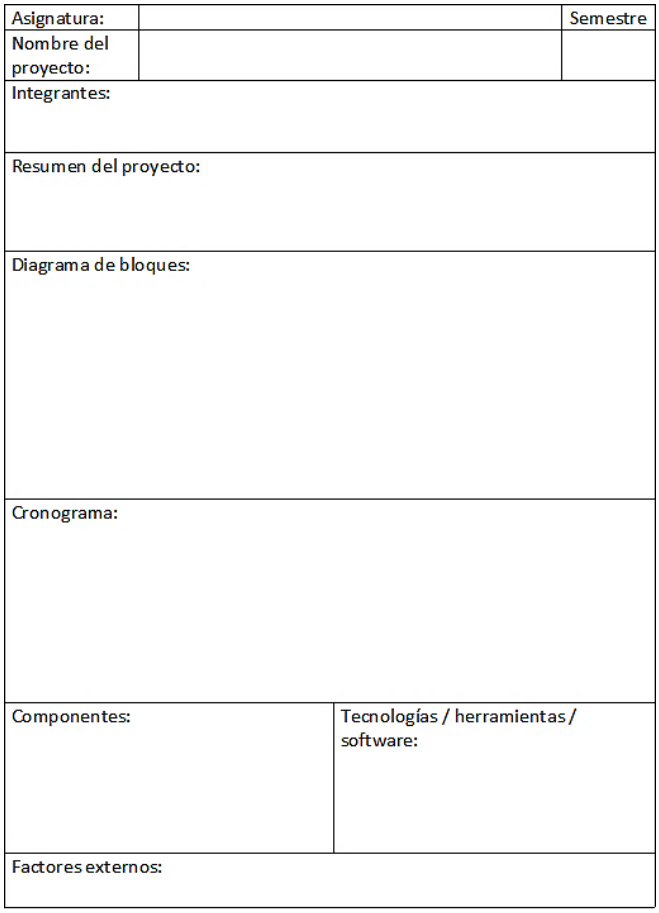
La primera página de la ficha, como se presenta en la figura 14, será rellenada por los miembros del equipo en donde deberán resumir las características más importantes de su proyecto como son:

- Título.
- Resumen del proyecto, mediante el cual expresarán de manera concisa cuál será el resultado final que desean obtener.
- Diagrama de bloques del proyecto, de forma que a través de este se pueda entender rápidamente cuál es el funcionamiento del mismo.
- Cronograma de las acciones que los integrantes planean realizar durante el intervalo de tiempo del curso. Este cronograma es un diagrama de tiempo tentativo y que puede modificarse a lo largo de las semanas, pero es importante puesto que guía el desarrollo del proyecto. Es frecuente que este cronograma cambie en el transcurso del desarrollo debido a

que muchos de los estudiantes no tendrán experiencia planificando proyectos.

- Componentes electrónicos necesarios, como pueden ser: microcontroladores, sensores, baterías, motores, drivers, pulsadores, displays, etc.
- Tecnologías, herramientas, software que planean utilizar, como pueden ser: comunicación inalámbrica Wifi/Bluetooth/lora, comunicación I2C/RS232/otras, diferentes softwares de programación de microcontroladores entre otros.
- Factores externos que los miembros consideren que pueda afectar al proyecto, como podrían ser: líquidos en entornos húmedos, ruido eléctrico, cableados no blindados, etc.

Figura 14. Fichas de revisión de proyectos: página 1



Asignatura:		Semestre
Nombre del proyecto:		
Integrantes:		
Resumen del proyecto:		
Diagrama de bloques:		
Cronograma:		
Componentes:	Tecnologías / herramientas / software:	
Factores externos:		

Nota. Para imprimir ver Anexo 2. Ficha de seguimiento semanal de proyectos

Una vez que se tiene la información de la página 1, es posible recordar rápidamente cuál es el proyecto que se está desarrollando y llevar a cabo la evaluación correspondiente.

La segunda página, como se puede ver en la figura 15, posibilitará escribir un resumen de la presentación que se ejecuta semanalmente. También permitirá escribir qué es lo que se espera que el proyecto consiga para la siguiente sesión. Finalmente se alcanzará colocar una nota referencial por cada semana.

Figura 15. Ficha de revisión de proyectos semanal: página 2

S	Apuntes	Revisión	Nota
2			
3			
4			
5			
6			
7			
9			
10			
11			
12			
13			
14			

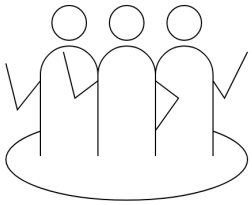
La página 2 de la ficha de revisión de proyectos tiene 12 casillas (de la semana 2 a la 14, excluyendo la semana 1 y 8), el número de semanas se explica con mayor detalle en el apartado 1.6.2.

1.6. TRABAJO DE EQUIPO

Varios proyectos en los que se participará se tendrán que desarrollar junto con otras personas, investigadores o estudiantes. Muchas veces en dichos equipos existe un jefe de equipo o un coordinador que dirá qué hacer a cada integrante, pero hay tendencias de tener equipos auto-organizados o equipos de trabajo que conversan horizontalmente, combinando su conocimiento con el objetivo de que el proyecto asignado salga adelante y, a su vez, que los miembros menos expertos del equipo crezcan tanto como

individuos como integrantes del equipo. La figura 16 representa la nueva filosofía de equipos de trabajo horizontal y auto-organizado.

Figura 16. Equipos de trabajo auto-organizados



Nota: Actualmente las empresas buscan tener equipos horizontales, sin posiciones jerárquicas, distribuyendo mejor la responsabilidad.

Existen técnicas de desarrollo ágiles las cuales tienen en mente aprovechar el trabajo de equipos de desarrollo horizontales como son *Scrum*. En este libro no se describirá *Scrum*, sino que se incitará a usar otra técnica que es mucho más sencilla conocida como *Kanban* que será descrita en el apartado 1.7.

1.6.1. Tamaño de los equipos

Cuando se trabaja con proyectos grandes es importante que los estudiantes interactúen en equipo, puesto que:

- Permite que desarrollen su capacidad colaborativa.
- Reduce el coste total del proyecto.
- La envergadura del proyecto puede ser mayor, permitiendo conseguir incluso artículos científicos de los mismos.
- La cantidad de proyectos que el docente debe calificar disminuye, haciendo que dedique más tiempo y mayor atención a cada uno de ellos.

En contrapartida también se tienen *peros* acerca del trabajo en equipo:

- Algunos miembros del equipo no participan en el proyecto.
- En equipos no homogéneos (algunos estudiantes se esfuerzan más por lo que tienden a tener mayor discernimiento de los temas) el conocimiento es monopolizado por pocos miembros.
- Discrepancias o rencillas entre sus miembros, llegando a casos en los que un integrante prefiere no trabajar y desaprobar para que el resto de sus compañeros también desaprueben como castigo por no trabajar.

Y debido a dichos “peros”, el docente debe ver formas de cómo calificar a todos los miembros incluyendo estas consideraciones.

Teniendo las consideraciones mencionadas anteriormente y otras más, dependiendo de cada realidad social, el tamaño de los equipos queda a criterio del docente.

1.6.2. Duración en semanas de las clases universitarias en Perú

En el Perú, la Ley Universitaria 30220, del 9 julio del 2014 establece que, para los estudios universitarios de pregrado presenciales, se debe tener no menos de 35 créditos de estudios generales y no menos de 165 créditos en estudios de especialidad. Todos estos créditos repartidos durante 10 semestres, lo que da una media de 20 créditos semestrales. Además, cada crédito corresponde a 16 horas de teoría o 32 horas de prácticas. Para conseguir esto, las universidades tienen semestres de 16 a 18 semanas. Antes de proponer la cantidad de semanas en las que se puede realizar un proyecto durante un semestre, se analizará la tabla 6 con cantidad de semanas que tienen algunas de las principales universidades de Perú.

Tabla 6. Tabla de número de semanas entre inicio de clases y principio de exámenes finales para el semestre académico 2019-20 en algunas universidades del Perú

Universidad	Inicio clases semestre 2019-20	Último día antes de exámenes finales	N° de semanas entre inicio de clases y principio de exámenes finales
Pontificia Universidad Católica del Perú	19/08/2019	30/11/2019	15
Universidad Nacional Mayor de San Marcos	19/08/2019	06/12/2019	16
Universidad Peruana de Ciencias Aplicadas	12/08/2019	23/11/2019	15
Universidad Nacional Agraria La Molina	19/08/2019	07/12/2019	16
Universidad Nacional de Ingeniería	19/08/2019	29/11/2019	15
Universidad del Pacífico	12/08/2019	23/11/2019	15
Universidad Privada Antenor Orrego	19/08/2019	30/11/2019	15
Universidad Privada del Norte	26/08/2019	07/12/2019	15
Universidad Cesar Vallejo	02/09/2019	07/12/2019	14

Nota: Datos recabados por los cronogramas publicados por las universidades para el ciclo académico 2019-20.

La tabla 6 alcanza el número de semanas entre el inicio de clases y el último día anterior a los exámenes finales para el semestre académico 2019-20, teniendo en cuenta que muchas veces se tiene una semana para exámenes

parciales y otra para exámenes finales, sin contar los feriados. Por lo tanto, se considera que para realizar un proyecto se tienen de manera efectiva catorce semanas, de las cuales siete son antes de los exámenes parciales y las siguientes siete son después de dichas pruebas.

1.6.3. Distribución por semanas de un proyecto

Como se estudió en el apartado anterior, en la realidad universitaria actual, se pueden desarrollar proyectos durante las semanas de clases que se pueden resumir en tabla 7.

Tabla 7. Distribucion semanal del proyecto

Semana	Descripción
1	Inicio de clases. Selección inicial del proyecto. Exposición sobre su selección.
2	Selección final del proyecto. Presentación de avances.
3	Desarrollo del proyecto y presentación de avances.
4	Desarrollo del proyecto y presentación de avances.
5	Desarrollo del proyecto y presentación de avances.
6	Desarrollo del proyecto y presentación de avances.
7	Presentación parcial del proyecto. Con demostración del prototipo.
8	Examen parcial. No se revisa el proyecto
9	Se retoma el desarrollo del proyecto. Presentación de avances.
10	Desarrollo del proyecto y presentación de avances.
11	Desarrollo del proyecto y presentación de avances.
12	Desarrollo del proyecto y presentación de avances.
13	Desarrollo del proyecto y presentación de avances.
14	Desarrollo del proyecto y presentación de avances.
15	Presentación final del proyecto. Incluye informe en formato de artículo científico.

En resumidas cuentas, se dispone de catorce semanas para tener la idea, materializarla, desarrollarla, consolidarla en un prototipo y presentar su versión final así como los informes respectivos.

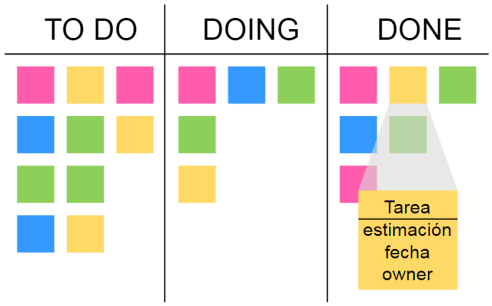
1.7. *KANBAN*

Kanban es un sistema visual para gestionar el trabajo a medida que se avanza dentro de un proceso. Este sistema permite visualiza tanto el proceso (el flujo de trabajo) como el trabajo real que pasa por dicho proceso. El objetivo de *Kanban* es identificar posibles cuellos de botella en el proceso y solucionarlos para que el trabajo pueda fluir de manera rentable a una velocidad o rendimiento óptimos (Digité, 2016).

Así, esta herramienta permite tener en un esquema las tareas que se deben realizar, y además es un cuadro que debe estar ubicado a la vista de todos los integrantes del equipo, ya que así podrán saber el estado del proyecto y si alguien tiene una falla pueda ser vista por otro miembro del equipo y apoyar en su solución.

La figura 17 muestra un tablero *Kanban* con las columnas más comúnmente utilizadas.

Figura 17. Tablero *Kanban*



Principalmente un esquema *Kanban* tiene los siguientes estados:

- ToDo: lista de tareas que se deben ejecutar “to do”.
- Doing: lista de tareas que se encuentran en proceso de producción.
- Done: lista de tareas que se han finalizado.

Esta lista de estados puede crecer tanto como se imagine, así también se pueden tener varios sub-estados. Por el contrario, cuando se está empezando a utilizar esta herramienta es recomendable no tener tantos estados.

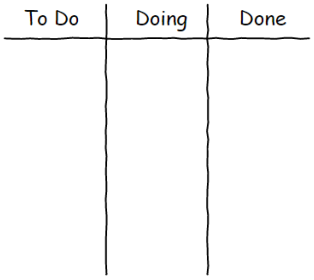
El cuadro de *Kanban* tiene muchas variaciones y reglas, cada equipo de trabajo agrega sus propias estrategias, pero la esencia debe seguir siendo la misma.

Tener un cuadro *Kanban* guarda una ventaja añadida. En efecto, un sentimiento de alegría se ve reflejado al momento de mover una tarjeta del estado “doing” al “done” y es importante que el propio desarrollador mueva su tarjeta personalmente.

1.7.1. ¿Cómo hacer un tablero Kanban propio?

Para crear tablero *Kanban* propio es procedente empezar con una hoja A4, la que será dividida en 3 partes como se puede apreciar en la figura 18.

Figura 18. Tablero Kanban simple



En la columna de “To-do” se ponen las tareas que se han desglosado (ver ejemplo para trocear tareas en el apartado 1.4.2). Para colocar las tareas se pueden utilizar pequeños retazos de papel y pegarlos en dicha columna (ejemplo post-it).

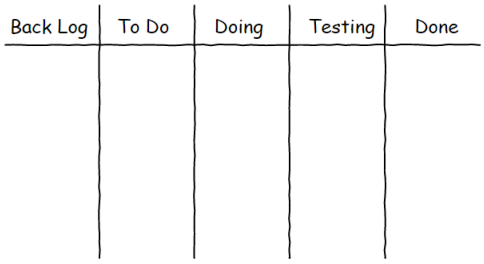
Como no se trabajará en todas las tareas simultáneamente, lo que se hará es que la tarea que se vaya a desarrollar será la que se moverá de la columna “To-do” a la columna “Doing”. Si se pertenece a un equipo de trabajo entonces también se pondrán los nombres de los integrantes en las tarea que se están desarrollando, así el resto del equipo de trabajo sabrá qué es lo que está realizando cada miembro del equipo.

Finalmente, cuando se haya concluido la tarea, ésta será movida de la columna “Doing” a la columna “Done”.

Existen muchas más reglas cuando se trabaja con el esquema *Kanban*, reglas que se pueden encontrar en libros especializados o reglas propias de un equipo de trabajo en particular, pero la esencia de este tablero es tener de forma gráfica, ordenada y pública (dentro del equipo de trabajo) las tareas que se están llevando a cabo.

Una de las mejoras comunes de este tablero es agregarle una columna de “Backlog” al inicio y otra columna llamada “Testing” antes de la columna “Done” como se puede ver en la figura 19.

Figura 19. Tablero Kanban con “Backlog” y “Testing”



La columna de “Backlog” es la utilizada para poner toda la lista de tareas e ideas que se tengan en mente hacer en un futuro cercano o lejano y que no son muy urgentes. Por ejemplo, la columna “To-do” solo tendrá las tareas que se pretenden hacer a lo largo de la semana.

La columna “Testing” será utilizada cuando algún integrante se encuentre en la fase de verificación y pruebas de la tarea en desarrollo.

1.7.2. Ejemplos

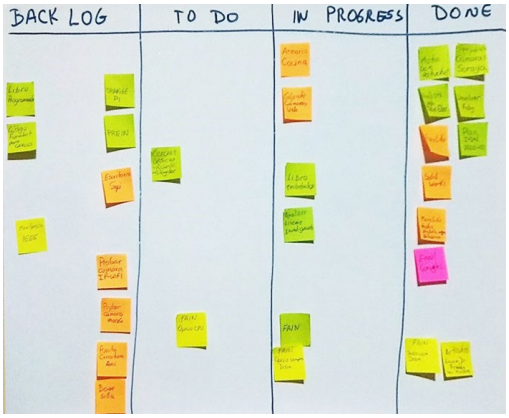
En esta sección se exponen ejemplos de tableros *Kanban* para el desarrollo de diversos proyectos de ingeniería.

En la figura 20 se muestra un ejemplo de dos tableros *Kanban* creados por dos equipos de estudiantes universitarios que competían en desarrollar un software matemático. Este tablero les permitió dividirse las tareas y observar su avance y el de sus compañeros.

Figura 20. Ejemplo de dos tableros Kanban simples

Y la figura 21 representa un tablero *Kanban* que también considera un Backlog para agrupar ideas o tareas que no están programadas inicialmente. Este Backlog es un espacio que permite ubicar todas las ideas que se tienen; sólo las tareas que consideran oportunas pasan a la columna de To-do, luego de un análisis detallado.

Figura 21. Ejemplo de tablero *Kanban* sencillo



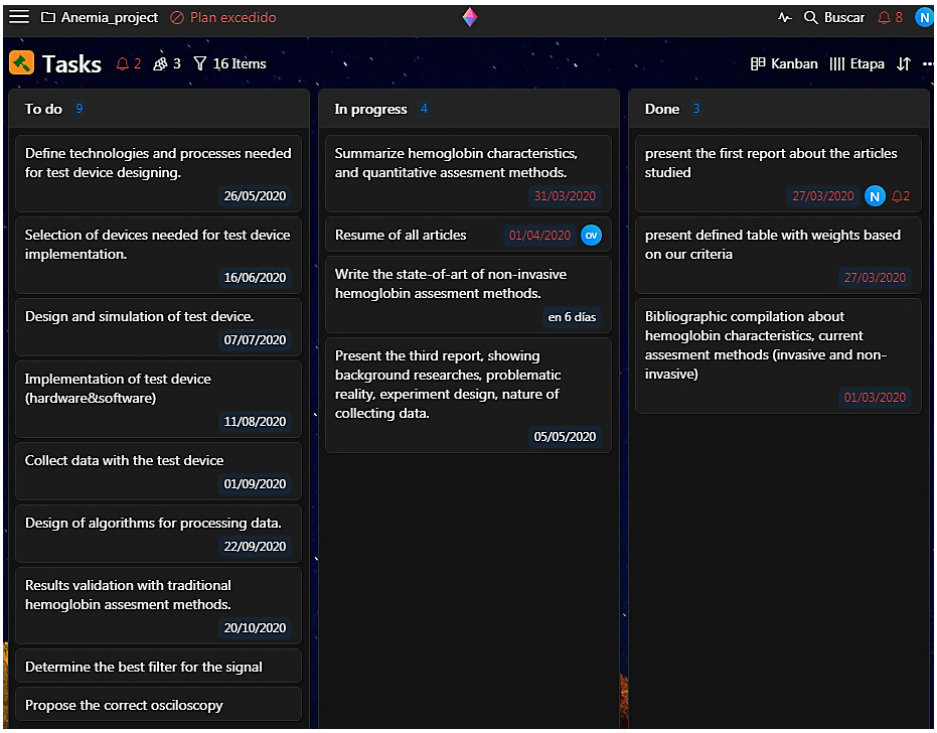
Existen varios softwares y aplicaciones en línea que permiten trabajar con el tablero *Kanban* de manera automática ya sea en el Smartphone o en la laptop y compartirlo con todo el equipo. Asimismo, algunos poseen las facultades de mover las tareas de un estado a otro de manera virtual, asignarlas, crear alarmas, hacer seguimiento entre otras funcionalidades. Como son algunos ejemplos de software: Trello, Monday, Kanbanize, MeisterTask, Clarizen, Zenkit, entre otros.

La figura 22 expone un tablero *Kanban* virtual que presenta las tareas que se van desarrollando para la implementación de un carrito seguidor de líneas.

Figura 22. Ejemplo de tablero *Kanban* virtual para implementar un carrito seguidor de línea

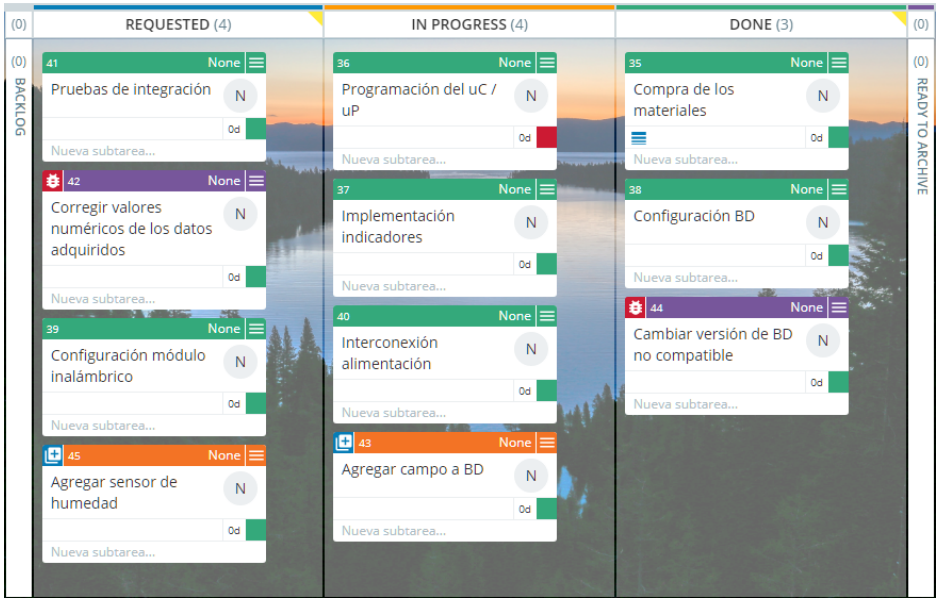
Los aplicativos que permiten compartir las tareas utilizando el Internet, hacen posible que varios investigadores y desarrolladores trabajen remotamente, cada uno tomando la tarea que le corresponde. En la figura 23 se muestra el tablero *Kanban* de un proyecto enfocado a desarrollar un dispositivo bio-electrónico para medir anemia.

Figura 23. Ejemplo de tablero *Kanban* virtual para el desarrollo de un proyecto bio-electrónico



En la figura 24 se puede ver otro proyecto desarrollado utilizando las aplicaciones web, además, se puede observar la presencia de tarjetas con diferentes colores y etiquetas, que representan distintas acciones, como pueden ser la corrección de un problema o el surgimiento de un nuevo requisito.

Figura 24. Ejemplo de tablero *Kanban* virtual para el desarrollo de un dispositivo IoT



CAPÍTULO 2.

MICROCONTROLADORES VS. PLACAS DE DESARROLLO BASADAS EN MICROCONTROLADORES

En el mercado existen muchos microcontroladores así como también muchas placas de desarrollo basadas en estos microcontroladores. En este capítulo se verán cuáles son las diferencias entre un microcontrolador propiamente dicho y una placa de desarrollo basada en microcontrolador.

2.1 ¿QUÉ ES UN MICROCONTROLADOR?

Un microcontrolador es un computador completo de limitadas prestaciones, el cual está contenido en un microchip de un único circuito integrado y su función es la de realizar una sola tarea. Posee una memoria de programa que es en donde se graban las instrucciones que el microcontrolador efectuará y será exactamente lo que se indique en esas líneas de código. Es necesario conocer exactamente cuál es la tarea que deberá ejecutar y todas sus especificaciones. También se debe considerar que realicen diferentes programas que den el mismo resultado, pero se deberá optar siempre por el que no realice operaciones innecesarias o utilicen mucha memoria (Apaza-Condori, 2010).

2.2 DIFERENCIAS ENTRE MICROCONTROLADORES Y PLACAS DE DESARROLLO

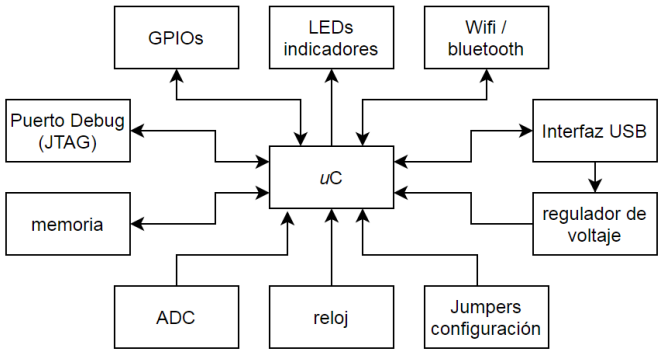
Internamente los microcontroladores tienen una arquitectura que les permite ser una pequeña computadora, tienen una unidad lógica aritmética, registros, puertos de entrada y salida y más; todo esto contenido en un único chip.

Generalmente, cuando se trabaja con un proyecto que necesita un microcontrolador se estará en la obligación de usar circuitería complementaria: una fuente reguladora de voltaje, un programador, cristales de oscilación, módulos de wifi o bluetooth, entre otros; y todos estos elementos se utilizarán en un protoboard o en una placa de baquelita que uno mismo tendrá que elaborar.

Por el contrario, se podría decir que las placas de desarrollo basadas en microcontroladores ya incorporan gran parte de estos módulos adicionales, de forma que cuando se necesite realizar un prototipado rápido solo se tendrá que "conectar y programar".

En la figura 25 se observa cómo es por lo general una placa de desarrollo basada en microcontrolador.

Figura 25. Diagrama de bloques de una placa de desarrollo basada en uC



Una placa de desarrollo basada en un microcontrolador puede tener los siguientes módulos:

- uC: microcontrolador que es el corazón de la placa.
- Interfaz USB: conexión que permite la comunicación por USB con otros dispositivos. Principalmente es utilizado para programar la placa empleando la computadora personal.
- Regulador de voltaje: circuito que utiliza la energía del puerto USB para regularlo y alimentar a la placa.
- Jumpers de configuración: son jumpers que permiten configurar el comportamiento de la placa.
- Reloj: circuito oscilador encargado de proporcionar la señal de reloj al uC. Muchas placas incluyen un circuito de RTC (Real Time Clock).
- ADC: son módulos conversores de señales analógicas a valores digitales. La resolución de estos módulos varía dependiendo de la placa.
- Memoria: es la memoria de la placa, por lo general es memoria Flash.
- Puerto debug: es un puerto que permite la conexión directamente al módulo de verificación del microcontrolador. Es una "puerta trasera" a la placa de desarrollo.
- GPIOs: son puertos de entrada o salida generales (General Purpose Input / Output). Son entradas o salidas digitales. Es importante tener en cuenta esto, ya que varias veces se confunden con pines de entrada/salida analógicos.

- Leds indicadores: son Leds incorporados en la placa que indican cuál es su comportamiento. Por ejemplo se tienen: led de encendido, led de transmisión, led de recepción, entre otros.
- Wifi / bluetooth: son módulos de comunicación inalámbrica que se encuentran en muchas placas, estos módulos han tenido un auge con el crecimiento de Internet de las Cosas (IoT), ya que permiten que la placa de desarrollo se comunique con la nube.

2.3 LISTA DE MICROCONTROLADORES Y PLACAS DE DESARROLLO

La tabla 8 ofrece una lista con los microcontroladores y las placas de desarrollo más utilizadas en la ciudad de Trujillo - Perú. Esta lista está obtenida en base a la experiencia como docente (del Ing. Alva) y como investigadores en el área de la Electrónica Digital.

Tabla 8. Lista de microcontroladores y placas basadas en uC

Nombre	*u*C	Placa	Descripción
PIC	✓		Familia de microcontroladores fabricada por Microchip Technology Inc. Existen muchas gamas de PIC de diferentes tamaños y potencia.
AVR (Atmega)	✓		Familia de microcontrolador fabricado por Atmel Corporation (Fue adquirida por Microchip Technology Inc. en el 2016). Uno de los microcontroladores más populares son los ATmega.
Arduino		✓	Placa de desarrollo basadas en los microcontroladores AVR. Son fabricadas por la compañía Arduino. Existen muchas placas clónicas.
Teensy		✓	Placa de desarrollo basadas en diversos microcontroladores fabricadas por la empresa PJRC. COM LLC. Muchas de sus placas usan arquitectura ARM.
STM32	✓		Familia de microcontroladores de 32 bits fabricada por la empresa STMicroelectronics.
STM32xxxxDISCOVERY		✓	Placas de desarrollo que utilizan como base los microcontroladores STM32.

Blue Pill		✓	Placa de desarrollo basada en el *u*C STM
ESP8266	✓		Más que un *u*C es un SoC (System on Chip). Es fabricado por la empresa Espressif Systems. Se caracteriza por incluir un módulo de comunicaciones Wifi.
Wemos		✓	Placa de desarrollo basada en el chip ESP8266
NodeMCU		✓	Placa de desarrollo basada en el chip ESP8266. Similar a la placa Wemos, pero ligeramente con mayor que la placa Wemos y con más pines.
ESP32	✓		Es el sucesor del ESP8266, también fabricado por la empresa Espressif Systems. Tiene 2 núcleos de 32bits y su módulo de comunicaciones inalámbrica permite comunicarse por Wifi y Bluetooth.
NodeMCU-32		✓	Placa de desarrollo basado el microcontrolador ESP32.

2.4 IDE DE PROGRAMACIÓN DE MICROCONTROLADORES

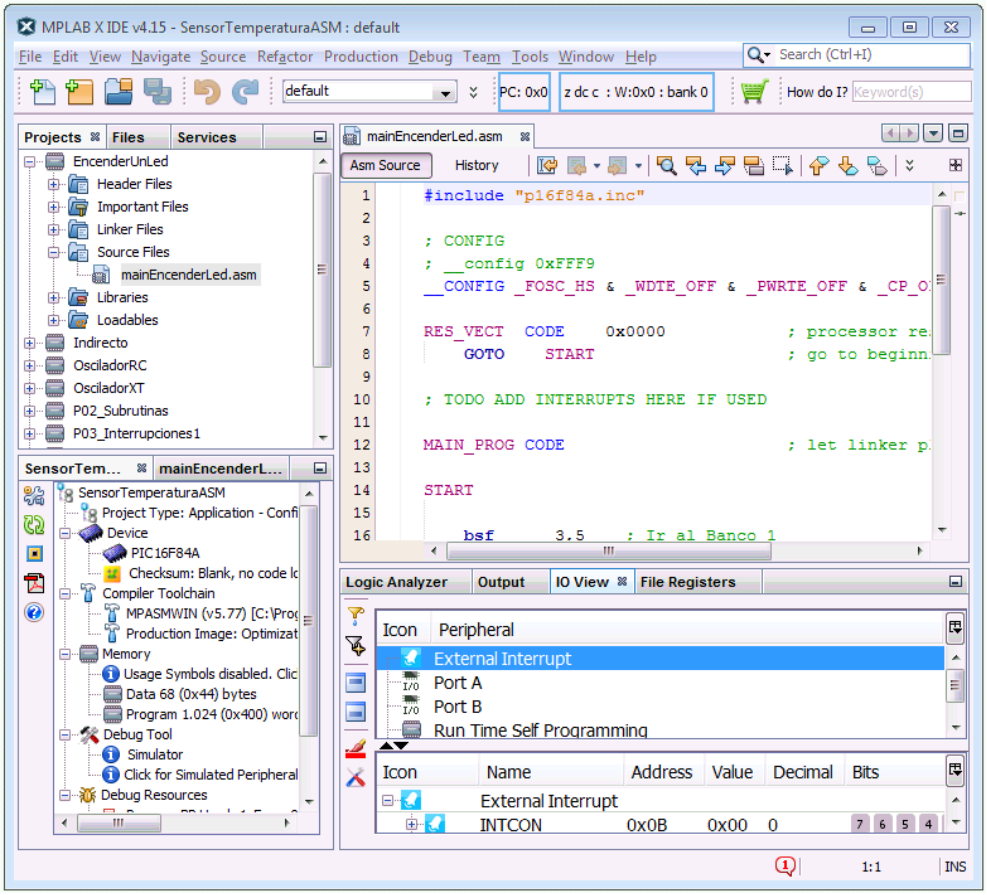
Un IDE (Integrated Development Environment) o entorno de desarrollo integrado, es una aplicación que se utiliza en una computadora para un fin en específico. En este caso, un IDE para programar *u*C permitirá tener los recursos necesarios para programar microcontroladores o placas de desarrollo basada en *u*C. A continuación, se describirán los más conocidos:

2.4.1. MPLab X IDE

Es un IDE gratuito para programar microcontroladores de la marca Microchip. Por lo general, es utilizado para programar microcontroladores PIC empleando lenguaje ensamblador. Entre sus características figuran que cuenta con visualizadores de datos, visualizadores de los pines I/O, simuladores paso a paso, coloreo de palabras clave, entre otros. Se pueden descargar en la página web: https://www.microchip.com/mplab/mplab-x-ide (Microchip Technology Inc., 2020).

La figura 26 es una evidencia del entorno de desarrollo MPLab X IDE con diferentes ventanas de trabajo, como son la ventana de lista de archivos, codificación, entradas y salidas y configuración del proyecto.

Figura 26. Entorno de desarrollo MPLab X IDE

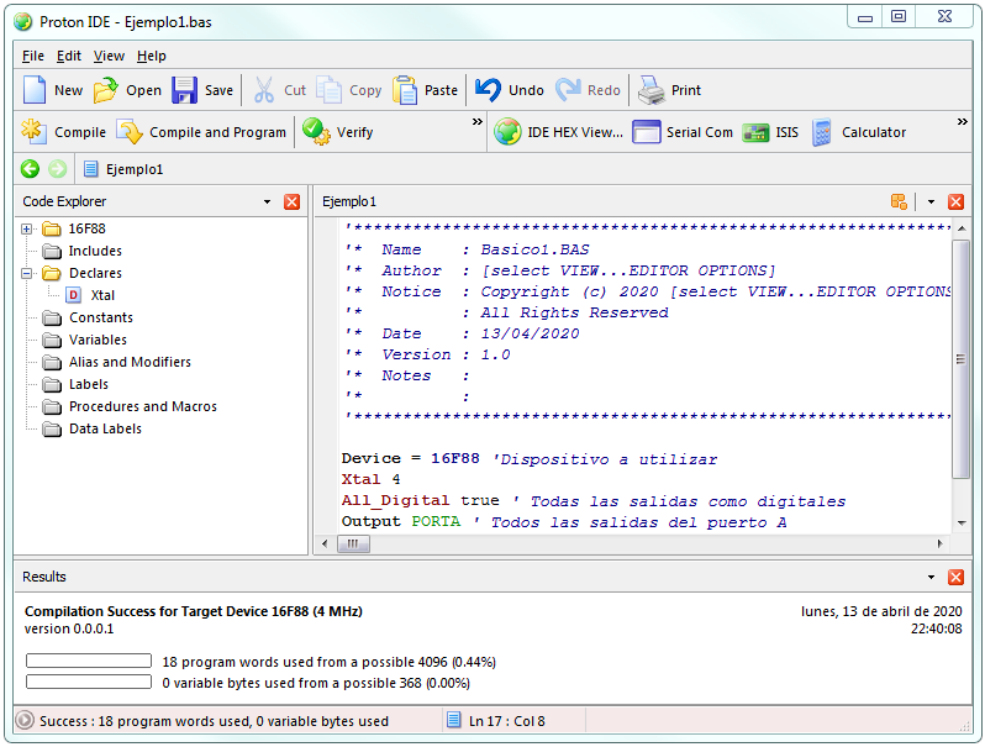


2.4.2. Proton IDE

Es un software para programar microcontroladores PIC desarrollado por la empresa Crownhill Associates. Ese aplicativo está enfocado en programar los microcontroladores utilizando el lenguaje Basic. Existe una versión gratuita en la que se puede programar y compilar los microcontroladores más comunes. Al comprar la licencia se activan algunos cientos más. Se puede descargar de http://www.protonbasic.co.uk/content.php/1450-Proton-Compiler-Updates (Crownhill Associates, 2020).

La figura 27 ofrece el entorno de programación Proton IDE, donde principalmente se observan 3 vistas, que son: el explorador de código, la ventana de codificación y el cuadro de resultados.

Figura 27. Entorno de desarrollo Proton IDE

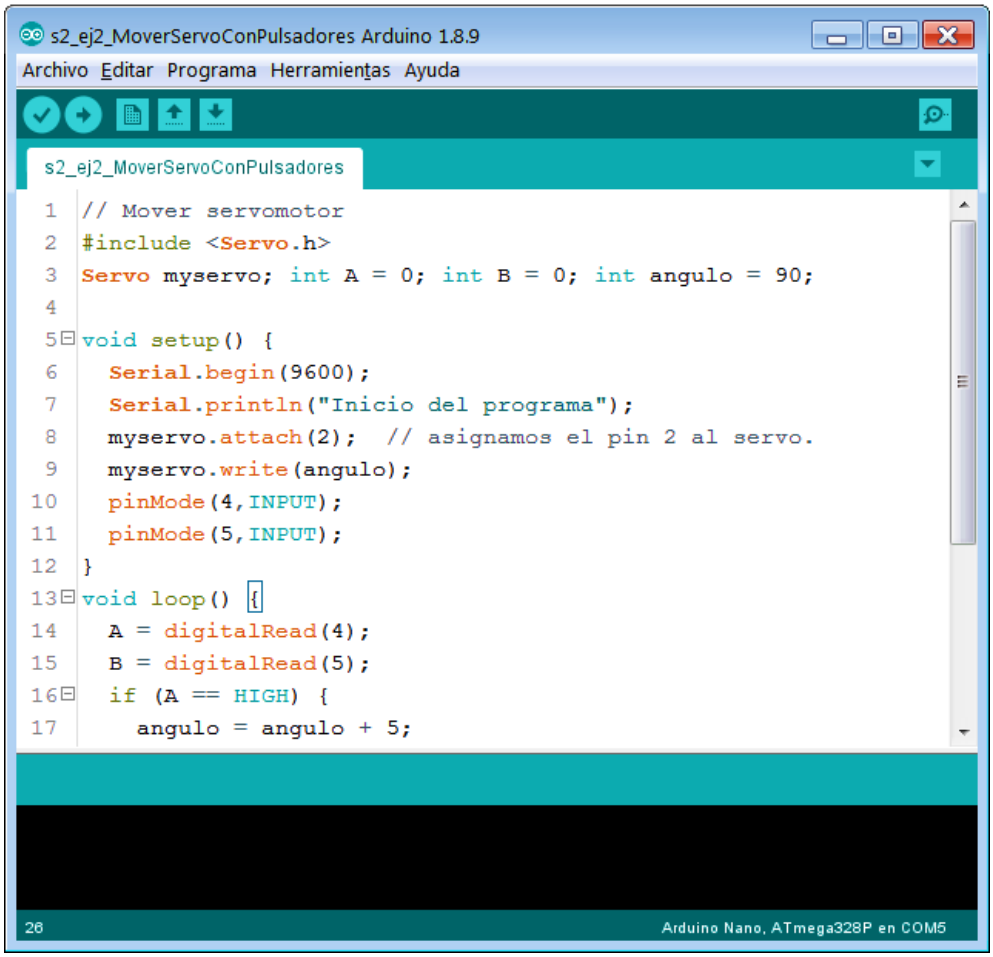


2.4.3. Arduino

Arduino es una compañía italiana dedicada a la elaboración de productos electrónicos que comercializa placas de desarrollo que llevan el nombre Arduino grabado en sus diferentes modelos y cuyo software de programación o IDE también es conocido con dicho nombre. En este apartado se explicará el software Arduino, el cual es una aplicación gratuita para computadora (de los sistemas operativos: Mac, Linux y Windows) que permite programar diversas placas de desarrollo Arduino: Arduino UNO, Arduino MEGA, Arduino DUO, etc. Además, este software tiene plugins (módulos adicionales) que permiten programar otro tipo de placas de desarrollo. Este software se puede descargar de la página web https://www.arduino.cc/en/Main/Software (Arduino, 2020).

La figura 28 muestra el entorno de programación Arduino IDE, dividido principalmente en la ventana de programación y el cuadro de resultados.

Figura 28. Entorno de desarrollo ArduinoIDE

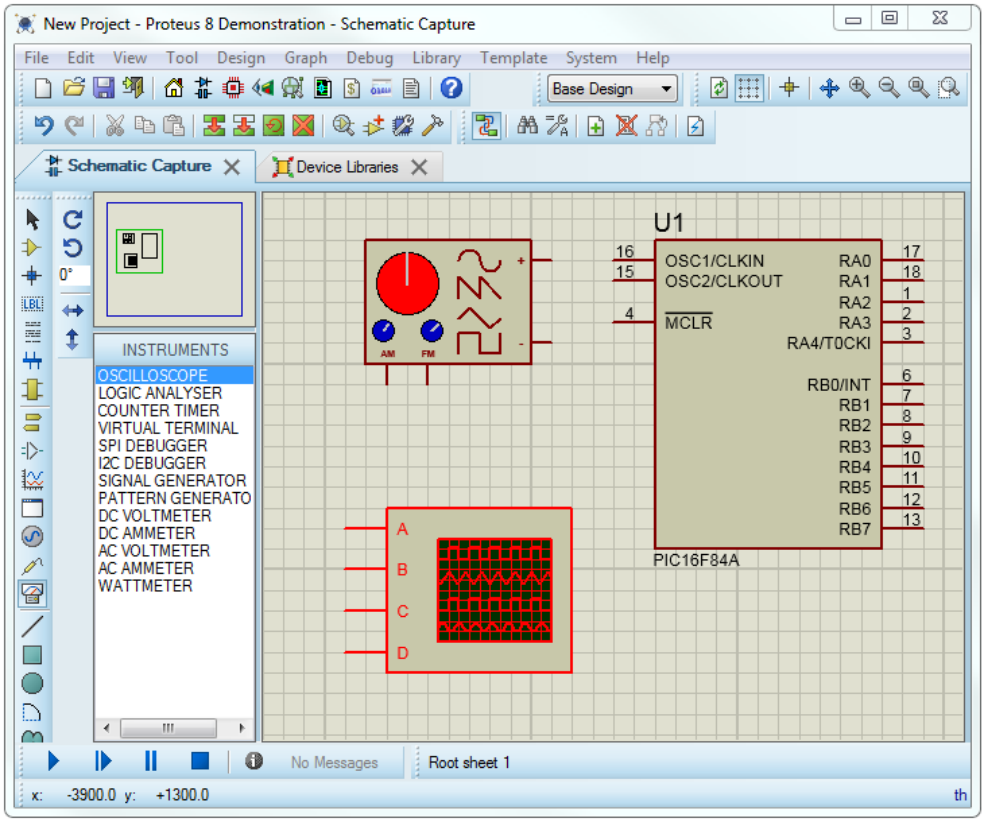


2.4.4. Proteus Design Suite

Proteus Design Suite es un software de diseño avanzado, desarrollado por la empresa Labcenter Electronics Ltd. Este software tiene una versión de pago con todas sus funcionalidades, pero también tiene una versión gratuita con funcionalidades limitadas. Tiene dos módulos principales: ISIS, Intelligent Schematic Input System (Sistema de Enrutado de Esquemas Inteligente) y ARES, Advanced Routing and Editing Software (Software de Edición y Ruteo Avanzado). Este software se puede descargar de la siguiente página web https://www.labcenter.com/ (Labcenter Electronics Ltd., 2020).

En la figura 29 se observa el entorno de desarrollo Proteus, en su versión de demostración.

Figura 29. Entorno de desarrollo Proteus

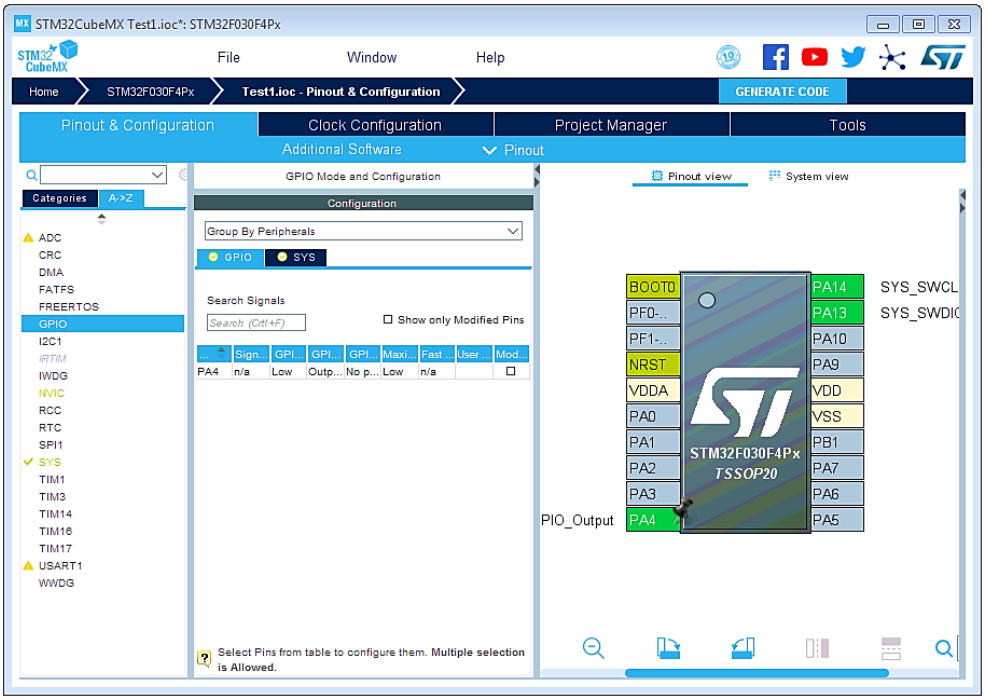


2.4.5. STM32Cube

El software STM32Cube es una familia de herramientas, formada principalmente por las aplicaciones STM32CubeMX, STM32CubeID, STM32CubeMonitoy STM32CubeProgrammer, que permite configurar de manera sencilla los microcontroladores STM. Estas herramientas han sido desarrolladas por la empresa STMicroelectronics. Poseen interfaz gráfica de configuración, interfaz para codificar el código en lenguaje C, interfaz para programar el código en el microcontrolador y utilidades de monitoreo y verificación. Estas aplicaciones pueden ser descargadas de la página https://www.st.com/en/development-tools/stm32-software-development-tools.html (STMicroelectronics, 2020).

La figura 30 muestra el entorno de desarrollo STM32CubeMX, que es uno de los aplicativos necesarios para programar los microcontroladores STM32.

Figura 30. Entorno de desarrollo STM32CubeMX



2.5 LENGUAJES DE PROGRAMACIÓN PARA MICROCONTROLADORES

Para programar los microcontroladores se tienen principalmente los siguientes lenguajes de programación.

2.5.1. Ensamblador

Es un lenguaje de programación de bajo nivel que también es conocido como assembler. Utiliza nemónicos que representan instrucciones básicas para ser ejecutadas en un microprocesador o microcontrolador. Cuando este lenguaje se ejecuta en un microcontrolador, cada instrucción dura un tiempo fijo del ciclo de reloj, así se puede determinar cuánto dura el programa sabiendo cuántas instrucciones se ejecutarán y cuántos ciclos de reloj durará cara instrucción (Wikipedia: Lenguaje ensamblador, 2020).

Figura 31. Ejemplo de programación en ensamblador para un PIC modelo 16F84

```
LIST P=16f84A           ;Indica el tipo de procesador a programar
#include "p16f84a.inc"  ;Incluye en el programa el fichero de definiciones del uC seleccionado
                        ;Ruta C:\PROGRAM FILES (X86)\MICROCHIP\MPLABX\V4.15\MPASMX\

; TODO INSERT CONFIG CODE HERE USING CONFIG BITS GENERATOR
 __CONFIG _FOSC_EXTRC & _WDTE_ON & _PWRTE_OFF & _CP_OFF

RES_VECT  CODE    0x0000            ;processor reset vector
    GOTO    START                   ;go to beginning of program


MAIN_PROG CODE                      ; let linker place main program

START

    BCF STATUS, RP0     ;nos vamos al banco 0
    CLRF PORTB          ;nos aseguramos que el PuertoB está a cero
    BSF STATUS, RP0     ;nos vamos al banco 1
    movlw 0x00          ;ponemos un cero en el registro w
    movwf TRISB         ;movemos el cero del registro w hacia el registro TRISB
                        ;es decir, ponemos el puerto B a Output

    BCF STATUS, RP0 ; nos vamos al banco 0


bucle:
    incf PORTB
    GOTO bucle                      ; loop forever

    END
```

El programa mostrado en la figura 31 se encarga de poner ciertos pines GPIO del microcontrolador como salida para luego incrementar digitalmente el valor de dichos pines, es decir, primero se tendrá el valor b'00000000', luego el valor b'00000001', posteriormente el valor b'00000010' y así sucesivamente.

2.5.2. Proton BASIC

Es un lenguaje de programación básico, principalmente utilizado en el software Proton IDE para programar microcontroladores PIC. El objetivo de este lenguaje es permitir que mediante un lenguaje sencillo y amigable se puedan programar diversas familias de microcontroladores PIC. En la figura 32 se observa un ejemplo de programación con lenguaje Proton BASIC.

Figura 32. Ejemplo de programación en Proton BASIC para el PIC 16F1829

```
' Un voltímetro digital, utilizando el ADC incorporado
Device = 16F1829
Declare Xtal = 4
Declare Hserial_Baud = 9600     ' se stablecer la velocidad en baudios para HRSOut
Declare Adin_Res = 10           ' resultado de 10 bits requerido
Declare Adin_Tad = cFRC         ' RC OSC escogidos para el ADC
Declare ADin_Delay = 50         ' 50us de tiempo de muestra

Dim vADC_Raw As Word   ' definimos variable
Dim voltaje As Float   ' definimos variable
Symbol q = 5.0 / 1024  ' se calcula el valor de cuantificación
ADCON1 = %10000000     ' se establece la entrada analógica en PORTA.0
Do                             ' se crea un bucle
    vADC_Raw = ADIn 0          ' leemos el ADC
    voltaje = vADC_Raw * q     ' se convierte el valor del voltaje
    HRSOutLn Dec2 voltaje,"V" ' se transmite el valor en voltios al terminal serie
    DelayMS 300                ' retraso
Loop                           ' bucle para siempre
```

2.5.3. C/C++

C/C++ es un lenguaje común para programar sistemas embebidos que contiene un conjunto de librerías y comandos de alto nivel que permiten programar el microcontrolador. Muchos microcontroladores aceptan código C/C++, pero uno de los entornos que popularizaron este lenguaje fue le empresa Arduino. En la figura 33 se puede ver un programa en C++ encargado de mover un servomotor.

Figura 33. Ejemplo de un programa escrito en C++ para mover un servomotor

```
1  // Mover servomotor
2  #include <Servo.h>
3  Servo myservo;  //creamos un objeto servo
4
5  void setup() {
6    Serial.begin(9600);
7    Serial.println("Inicio del programa");
8    myservo.attach(2);  // asignamos el pin 2 al servo motor.
9  }
10 void loop() {
11   Serial.println("moviendo a 0 grados");
12   myservo.write(0);  // enviamos el valor deseado al servo.
13   delay(1000);
14   Serial.println("moviendo a 90 grados");
15   myservo.write(90);  // enviamos el valor deseado al servo.
16   delay(1000);
17   Serial.println("moviendo a 180 grados");
18   myservo.write(180);  // enviamos el valor deseado al servo.
19   delay(1000);
20 }
```

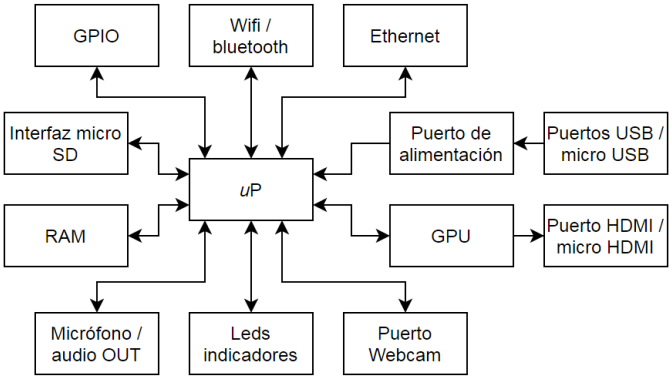
CAPÍTULO 3.
COMPUTADORAS DE PLACA ÚNICA (SBC)

Las computadoras de placa única o SBCs (Single Board Computer) se caracterizan por ser toda una computadora entera contenida dentro de una sola placa de circuito impreso. En muchos casos eso no significa que se conectará el dispositivo y ya podrá funcionar, sino que se necesitarán ciertos elementos adicionales (que serán descritos en el siguiente apartado) así como también la instalación y configuración de su sistema operativo. Para ver en más detalle la instalación de una SBC dirigirse al *Anexo 1: Configuración de una placa Raspberry Pi y ejercicios*.

3.1. PARTES DE COMPUTADORAS DE PLACA ÚNICA

Cuando se habla de una computadora de placa única, o computadora de placa reducida, se refiriere a un dispositivo que en una sola placa impresa tendrá la mayoría de módulos necesarios para arrancar un sistema operativo. Este dispositivo tendrá incluido un regulador de voltaje, memoria integrada, módulo gráfico, módulos de interconexión con el exterior, entre otros, tal como se muestra en la figura 34.

Figura 34. Diagrama de bloques de una SBC



Los módulos que principalmente contiene una computadora de placa única son:

- *u*P. Microprocesador, que es el corazón del sistema. Generalmente utiliza arquitectura ARM.
- GPU. Unidad gráfica de procesos (Graphic Process Unit).
- Puertos de conexión con el exterior. Como son puertos HDMI, USB, de micrófono, audio, Ethernet.
- RAM. Memoria de acceso aleatorio. Este tipo de memoria viene soldada a la placa y no se puede cambiar.
- Slot micro SD. Slot que alberga la memoria micro SD que se utiliza como "disco duro", en donde se podrá colocar la memoria micro SD que por lo general tiene instalado el sistema operativo. Hay casos en que la SBC tiene incorporada una memoria flash desde la cual se podrá arrancar el sistema.
- Wifi / bluetooth. Muchas placas llevan incorporados estos módulos de conexión inalámbrica.
- GPIO. Puertos de entrada y salida principalmente digitales.

3.2. ARQUITECTURA ARM VS ARQUITECTURA X86

Después de un gran auge basado en computadores que utilizaban procesadores con arquitectura x86, comenzaron a aparecer los dispositivos portátiles cimentados en procesadores con una arquitectura ARM (Advanced Risk Machine). La tabla 9 se muestra las ventajas, desventajas y diferencias entre estas arquitecturas.

Tabla 9. Diferencias entre arquitecturas ARM y X86

Aspecto	ARM	X86
Consumo de energía	Relativamente bajo	Alto
Uso	En celulares, tablets, sistemas embebidos, juguetes, discos duros, etc.	En laptops y computadoras de escritorio
Potencia de cómputo	Mediana a alta	Alta a muy alta
Empresas que los fabrican	Qualcomm, Samsung, Media Tek	Intel, AMD
Arquitectura	RISK (Reduced Instruction Set Computers)	CISC (Complex Instruction Set Computers)
Multitarea	No están específicamente diseñados	Diseñados para realizar multitarea
Sistemas operativos	Android, linux embebido, iOS, etc (listados en el apartado 3.6)	Windows, Linux, MacOS

3.3. ELEMENTOS ADICIONALES NECESARIOS

Aunque la SBC esté diseñada para no acoplarle módulos adicionales, muchas de ellas necesitan los siguientes elementos para su correcto funcionamiento.

- Fuente de alimentación. Se debe estimar correctamente el voltaje y amperaje necesario para el funcionamiento de la placa incluido el consumo eléctrico de los periféricos adicionales que se utilicen.
- Memoria micro SD. Memoria en la que por lo general se guarda el sistema operativo de la placa. Ejemplos de los sistemas operativos se pueden ver en el apartado 3.6.
- Adaptador micro SD a SD. Adaptador que será necesario cuando se vaya a escribir el sistema operativo desde la laptop hacia la memoria micro SD.
- Adaptador micro HDMI a HDMI. Este adaptador sólo será necesario cuando se requiera ver el video de la SBC, pero únicamente cuando esta placa solo tenga salida micro HDMI.
- Monitor. Si se desea ver la salida de video, entonces se necesitará un monitor con entrada HDMI.
- Teclado / mouse. Como toda computadora, si queremos interactuar con nuestra SBC, un teclado y un mouse serán necesarios. Hay casos en los que no son necesarios, pero son configuraciones más avanzadas.
- Case. Como todos los circuitos de la SBC se encuentran expuestos a la intemperie, es recomendable utilizar un case o funda como protección.
- Cable de red / Wifi router / switch. La conexión con el mundo exterior o con Internet siempre será indispensable para proyectos que pretendan comunicar a otros sistemas o servicios las medidas tomadas.
- Entrada y salida de audio. Solo serán necesarios si el monitor HDMI que se tenga no posea audio y micrófono incorporados.

3.4. COMPUTADORAS DE PLACA ÚNICA VS COMPUTADORAS DE ESCRITORIO

A continuación, en la tabla 10, se listan las principales diferencias entre las computadoras de placa única y las computadoras tradicionales de escritorio o de sobremesa. Hay casos específicos donde estas diferencias no se dan, pero la idea de este listado es guiar al lector a comprender las principales diferencias entre la generalidad de estos dispositivos.

Tabla 10.Comparación entre las SBCs y las computadoras de sobremesa

Aspecto	Computadoras de placa única	Computadora de sobremesa
Dimensiones	Dimensiones muy pequeñas. Muchas placas son más pequeñas que una tarjeta de crédito.	Dimensiones medianas a grandes
Precio	Precio reducido.	Precio elevado.
Potencia de cómputo	Reducida potencia de cómputo	Elevada potencia de cómputo
Consumo eléctrico	Reducido consumo de electricidad (generalmente menos de 10 vatios)	Elevado consumo de electricidad
RAM	RAM fija. No se puede cambiar	RAM modular. Se puede aumentar o disminuir.
Disco duro	Generalmente viene a ser la tarjeta micro SD	Se puede utilizar discos duros de mucha capacidad.
Tarjeta de video	Mínima y fija	Hay muchas configuraciones y posibilidades. Se puede poner tarjetas de video muy potentes
Tarjetas de expansión	No tiene	Tiene 2 a 4 tarjetas
GPIOs	Tiene un puerto dedicado a este propósito	No tiene. En caso de necesitar se podría utilizar una tarjeta de expansión
Uso	Generalmente enfocada para usos puntuales	Gran cantidad de usos

3.5. LISTA DE COMPUTADORAS DE PLACA ÚNICA

A continuación se describe brevemente los modelos de SBCs más conocidos en el mercado.

- Raspberry Pi. Una de las SBCs más populares y utilizada en el mundo. Fue desarrollada por la *Raspberry Pi Foundation*. Cuenta con diversos modelos y tiene una gran comunidad que da soporte a dudas y al desarrollo de nuevas aplicaciones.

- Orange Pi. Es una placa de código abierto y está basada en la placa Raspberry Pi. Generalmente son placas de menor precio, pero su comunidad no es tan amplia. Estas placas son fabricadas por la empresa Shenzhen Xunlong Software CO en China.
- Banana Pi. Similares a las placas Orange Pi, son la competencia de las Raspberry Pi. Son placas pequeñas y de bajo coste desarrolladas por la empresa Shenzhen SINOVOIP Co., Ltd.
- Rock Pi. Son placas de desarrollo similares a las anteriormente mencionadas, fabricadas por la empresa Rockchip.
- Nano Pi. Son placas de desarrollo basadas en un procesador ARM, desarrolladas por la empresa FriendlyARM y se caracterizan porque son generalmente más pequeñas que su competencia, de ahí viene su nombre de "nano".
- Tinker Board. Placa de desarrollo fabricado por la empresa Asus, cuyo punto fuerte es el GPU. Está diseñada especialmente para trabajar con juegos, visión artificial, procesamiento de imágenes, reconocimiento de gestos, etc.
- Odroid. Son placas de desarrollo fabricados por la empresa Hardkernel. Están basadas en procesadores ARM y existen muchos modelos.
- VoCore. Son placas de desarrollo del tamaño de una moneda, que incluyen conexión inalámbrica. Han sido pensadas para ser utilizadas dentro del mundo de Internet de las cosas (IoT). La empresa que las fabrica es VoCore Studio.
- Jetson nano. Es una placa de desarrollo fabricada por Nvidia. Integran internamente el procesador Tegra. Son utilizadas para trabajar con visión computacional y deep learning.

3.6. SISTEMAS OPERATIVOS PARA SBCS

Dependiendo del tipo de placa que estemos utilizando y también del objetivo que tengamos, existe una gran cantidad de sistemas operativos utilizables, que van desde centro de recreación hasta servidores de procesos y bases de datos. A continuación se indican y se resumen los más conocidos.

- Linux. Es uno de los sistemas operativos más populares para este tipo de placas. Muchos de los sistemas operativos siguientes tiene como base el core de Linux, pero en este punto se mencionarán a los más representativos que son: Raspbian, Fedora, Arch Linux, Armbian, Noobs, Ubuntu Core.
- OpenWRT. Es un firmware basado en Linux, que se caracteriza por necesitar muy poco espacio para funcionar. Está principalmente desarrollado para trabajar como sistema operativo de los Routers.

- ROS. Aunque su nombre signifique Robotic Operating System (Sistema Operativo Robótico), no es un sistema operativo en sí, sino una serie de paquetes que se pueden instalar sobre el sistema operativo y así poder utilizar todas sus funcionalidades. Una de sus principales características es que se puede instalar en varios dispositivos y comunicar los nodos entre sí de manera relativamente sencilla.
- FreeRTOS. Es un sistema diseñado para ser pequeño, simple, funcionar en sistemas embebidos y principalmente ser un sistema que opere en tiempo real.
- Android. Es el sistema operativo por excelencia que funciona sobre gran parte de los celulares en todo el mundo. Está basado en el kernel de Linux y tiene una amplia gama de aplicaciones.
- Windows 10 IoT Core. Es la versión de Windows desarrollada por Microsoft para dispositivos ARM.
- Gestores multimedia. También se han desarrollado sistemas operativos específicamente enfocados en ser utilizados como centros de ocio. Entre estos sistemas se tienen: OSMC, Kodi, OpenElec, LibreElec.
- Juegos. Las computadoras de placas reducidas también han entrado al mundo de los juegos y entre los principales sistemas operativos que permiten jugar se tienen: RetroPie, RecalboxOS, Batocera Linux, Lakka.

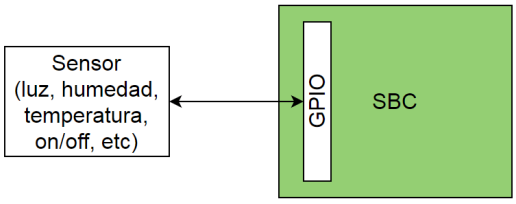
3.7. INTERCONEXIÓN

En este apartado se ayudará al lector a responder la pregunta "¿cómo puedo leer los datos de un sensor?" Para esto se proponen tres formas de realizarlo.

3.7.1. Conexión directa usando los puertos GPIO

Se debe recordar que la placa de desarrollo posee un puerto GPIO, a través del cual se puede conectar con otros dispositivos. Para esto, se considerará que se tiene un sensor digital de luz, humedad, temperatura o, incluso, un sensor ON/OFF. El cableado será como el que se muestra en la figura 35.

Figura 35. Conexión directa usando puertos GPIO



Como se puede ver, la transmisión de datos pasa directamente desde el sensor hacia la SBC y/o viceversa.

En la imagen siguiente se puede ver el código necesario, programado en Python, para encender un LED conectado al pin 9 del puerto GPIO en una placa Raspberry Pi.

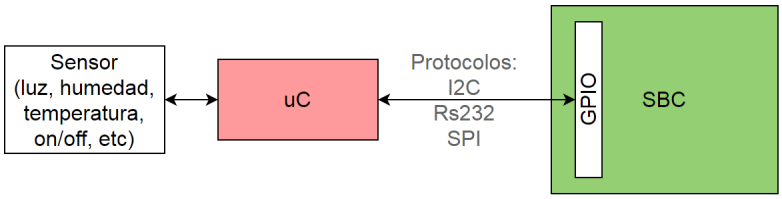
```
1  from gpiozero import LED
2  led = LED(9)
3  led.on( )
4  quit( )
```

Para casos en los que el sensor tenga un protocolo serie, ejemplo el RS232, I2C o SPI, se debe encontrar una librería que le permita comunicarse con este sensor. Para el caso que no se tenga la facilidad de importar una librería, entonces se deberán conocer el protocolo bit a bit y codificarlo.

3.7.2. Conexión indirecta cableada utilizando un microcontrolador

Hay casos en los que se tiene un microcontrolador que será el encargado de leer los datos de los sensores, ya sea porque el sensor que se tiene es analógico y la SBC no cuente con entradas analógicas, o porque un circuito previamente desarrollado y se deba incorporar una placa para obtener y guardar los datos. En este caso, como se puede observar en la figura 36, se necesitará conectar la SBC al microcontrolador. Para ello, es factible hacer uso de los protocolos de comunicación serie, como pueden ser el I2C, RS232 o SPI. Se debe recordar que muchas veces se necesitarán importar las librerías necesarias para poder leer este tipo de flujo de datos.

Figura 36.Conexión indirecta cableada utilizando un microcontrolador

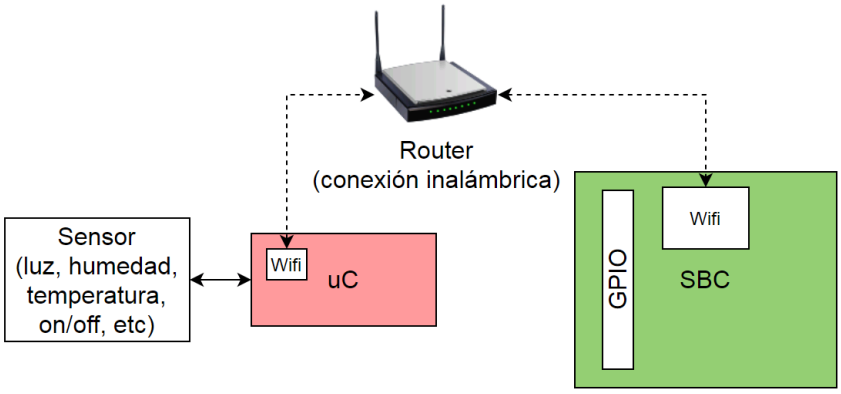


Otra forma de conectarse al microcontrolador es utilizando el mismo puerto USB incorporados en algunas placas de desarrollo. Este puerto USB también se utiliza para enviar y recibir datos.

3.7.3. Conexión indirecta inalámbrica utilizando un microcontrolador

Finalmente, se puede utilizar tecnología inalámbrica para conectar la SBC al microcontrolador, siendo necesario tener un router o que el microcontrolador tenga la capacidad de ser un punto de acceso. La conexión necesaria será conforme la que se aprecia en la figura 37.

Figura 37. Conexión indirecta inalámbrica utilizando un microcontrolador



Con este tipo de conexión también se obtiene información desde otros microcontroladores conectados a la red inalámbrica. Para este tipo de red también será necesario que se tenga conocimientos de cómo trabajar con protocolos de comunicación como HTTP o MQTT.

3.8. IDES COMPLEMENTARIOS

Muchas veces, cuando se trabaja con microcontroladores y microprocesadores, se tendrá acceso a mucha información que será necesaria procesar a través de diferentes medios. Por ejemplo, de forma gráfica, se tendrá que desarrollar aplicaciones que ayuden a visualizar estos datos, modificarlos, procesarlos, etc. Por la potencia requerida, estos IDE se deben instalar en la computadora de trabajo y los complementos que se desarrollen se ejecutarán en el SBC.

A continuación, se listan algunos IDE que permiten desarrollar estos complementos.

- Eclipse. Es un IDE famoso por su entorno de desarrollo integrado para desarrollar programas en Java, pero teniendo además otros IDE enfocados a otros lenguajes, incluidos IDE C / C ++, IDE JavaScript / TypeScript, IDE PHP y más. Este IDE es gratuito y se puede descargar de la página web https://www.eclipse.org/downloads/ (Eclipse Foundation, 2020).
- Pycharm. El un IDE específicamente desarrollado para trabajar con Python. Tiene dos versiones, la de pago y la community. Se puede descargar de la página web https://www.jetbrains.com/es-es/pycharm/ (Jet Brains, 2020).
- NotePad++. Es un editor gratuito, de código fuente abierto y reemplaza el Bloc de Notas. Admite varios idiomas, funciona sobre el entorno Microsoft Windows y su uso se rige por la licencia GPL. Se puede descargar de la página web https://notepad-plus-plus.org/ (Ho, 2020).

- Sublime. Se define como un editor de texto sofisticado. Una de sus ventajas es que incluye muchos atajos y funcionalidades especiales que otros editores de texto no poseen. Se puede descargar desde su página web https://www.sublimetext.com/ (Skinner, 2020).
- Intelij. Es un IDE con el cual se puede desarrollar programas escritos en el lenguaje JAVA. Se puede descargar desde su página web https://www.jetbrains.com/es-es/idea/ (JetBrains, 2020).
- Codeblocks. Es un IDE gratuito escrito en C, C ++ y Fortran creado para codificar y compilar los programas desarrollados. Está diseñado para ser muy extensible y configurable. Esta aplicación se puede descargar de http://www.codeblocks.org/ (The Code::Blocks team, 2020).
- Visual C++. También conocido como Microsoft Visual C++, es un entorno de desarrollo integrado para lenguajes de programación C, C++ y C++/CLI. Funciona sobre la plataforma Microsoft Windows. Existe una versión Express, llamada Microsoft Visual C++ Express Edition, que es gratuita y se puede descargar desde el sitio de https://visualstudio.microsoft.com/es/vs/express/ (Microsoft Corporation, 2020).
- Visual Studio Code. Es un editor de código con licencia Freeware, compatible con varios lenguajes de programación y con una gran cantidad de plugins para adaptarlo a las necesidades de los programadores. Aunque ha sido desarrollado por Microsoft Corporation, se puede utilizar sobre Windows, Linux o MacOs. Se puede descargar de https://code.visualstudio.com/ (Microsoft Corporation, 2020)
- Gambas. Es un lenguaje de programación libre, similar al Basic, que funciona sobre la plataforma Linux. Permite crear botones, formularios, cuadros de texto, controles y enlazarlos con diferentes bases de datos como MySql, PostgreSQL o SQLite. Una de sus principales características es que permite realizar aplicaciones gráficas. Se puede descargar de la url http://gambas.sourceforge.net/en/main.html. (Minisini, 2020)

CAPÍTULO 4.
ECOSISTEMA IOT

Como ya se ha mencionado en los capítulos anteriores, los microcontroladores, microprocesadores y sus placas de desarrollo, permiten tener acceso a una gran cantidad de datos, así como tele-controlar diversos dispositivos. Al ser una tendencia a nivel mundial la conexión de todos los dispositivos entre sí, se dedica este capítulo a entender cómo funciona el ecosistema IoT, representado por la figura 38, pero sin dejar de lado hacia dónde se dirige toda esta tecnología.

Figura 38. Conectemos el mundo

4.1 IOT

El Internet de las cosas (IoT) es un tipo de red que conecta cualquier *cosa* con Internet. Dicha red está basada en protocolos estipulados a través de equipos de detección de información para llevar a cabo el intercambio de información y las comunicaciones, con el fin de lograr reconocimientos inteligentes, posicionamiento, rastreo, monitoreo y administración.

Se puede clasificar el IoT en tres categorías que interactúan a través de internet: (1) gente a gente, (2) gente a máquina/cosas, (3) cosas/máquina a cosas/máquina.

El objetivo del Internet de las cosas es permitir que las *cosas* se conecten en cualquier momento y lugar; asimismo, posibilitar la conexión con diferentes *cosas* y personas que utilicen idealmente la red y/o servicio de Internet.

Sus características fundamentales son: interconectividad, protección de la privacidad, heterogeneidad, gran escala y coherencia semántica entre las *cosas* físicas y sus *cosas* virtuales asociadas (Keyur K & Sunil M, 2016).

4.2 VENTAJAS Y DESVENTAJAS DEL IOT DESDE EL PUNTO DE VISTA DE LA DOMÓTICA

Con el boom del IoT, han surgido muchos productos enfocados a ser utilizados dentro de los hogares (domótica). En este apartado se detallarán sus ventajas y desventajas.

La figura 39 muestra cómo el IoT ha ido reemplazando a la domótica tradicional (que utilizan dispositivos desarrolladas por grandes y conocidas empresas y que principalmente se interconectan por cable) a través del uso cada vez más común de dispositivos que se interconectan a la red wifi del hogar.

Figura 39. Un nuevo concepto de domótica ha llegado

Fuente (Carretero, 2017)

"Estos dispositivos IoT están revolucionando el mercado, por su bajo coste, por su concepto de plug & play, su fácil instalación y porque resuelven soluciones sencillas s*in tener que entrar en el mundo profesional de los sistemas*" (Carretero, 2017).

Se listas a continuación las ventajas y desventajas al utilizar los dispositivos IoT de forma masiva para interconectar los diversos artefactos dentro de los hogares.

Ventajas

- Inversión inicial baja.
- Dispositivos de fácil instalación y configuración (enfocados al cliente final).
- Fácil comunicación (wifi o radio).
- Ideal para pequeñas casas o pisos.
- Productos novedosos e interesantes en cada momento.
- Productos accesibles a nivel mundial.

Desventajas

- Muchos productos creados por Startups (pequeñas empresas emergentes que se dedican al desarrollo tecnológico partiendo de una idea innovadora), si tienen una vida corta, entonces apagan sus servidores.
- Al ser productos de bajo coste, la calidad y la estética dejan qué desear.
- Las actualizaciones pueden hacer que el producto deje de funcionar.
- Carencia de soluciones profesionales, ya que los productos aún son muy genéricos.
- Falta de comunicación entre marcas y/o productos (falta de integración similar a sistemas domóticos con estándares).
- Baja fiabilidad (gran parte por la comunicación inalámbrica).
- Nivel de seguridad bajo.

4.3 TENDENCIAS TECNOLÓGICAS ALREDEDOR DEL IOT

En este apartado se muestra cómo ha ido cambiando la tendencia de las diez principales tecnologías estratégicas a lo largo de estos últimos cinco años. Para ello, se ha recopilado información de la Consultora Gartner Inc., que es una empresa dedicada a la consultoría y a la investigación en TI con sede en Estados Unidos (Gartner Inc., 2020). En los siguientes apartados, se presentan gráficas que muestran diversas tendencias tecnológicas, de las cuales, las más relacionadas con el IoT han sido marcadas con un círculo rojo.

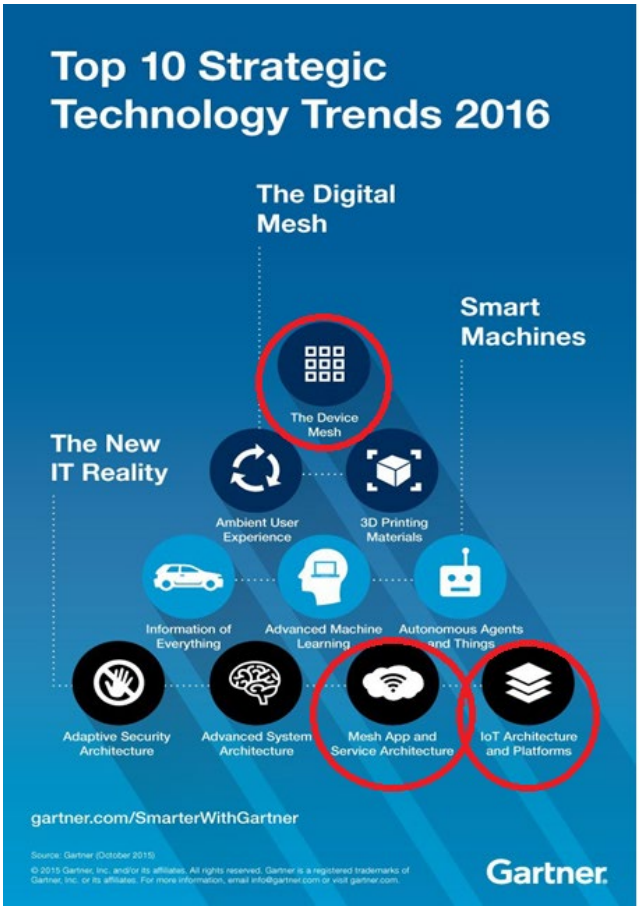
4.3.1. Las 10 principales tendencias tecnológicas estratégicas 2016

La tendencia tecnológica para el 2016, presentada por la consultara Garner, Inc, se puede apreciar en la figura 40 y fue dada por:

- The device mesh
- Ambient user experience
- 3D printing materials

- Information of everything
- Advanced machine learning
- Autonomous agents and things
- Adaptative security architecture
- Advanced system architecture
- Mesh app and service architecture
- IoT architecture and platforms

Figura 40. Top 10 Strategic Technology Trends 2016



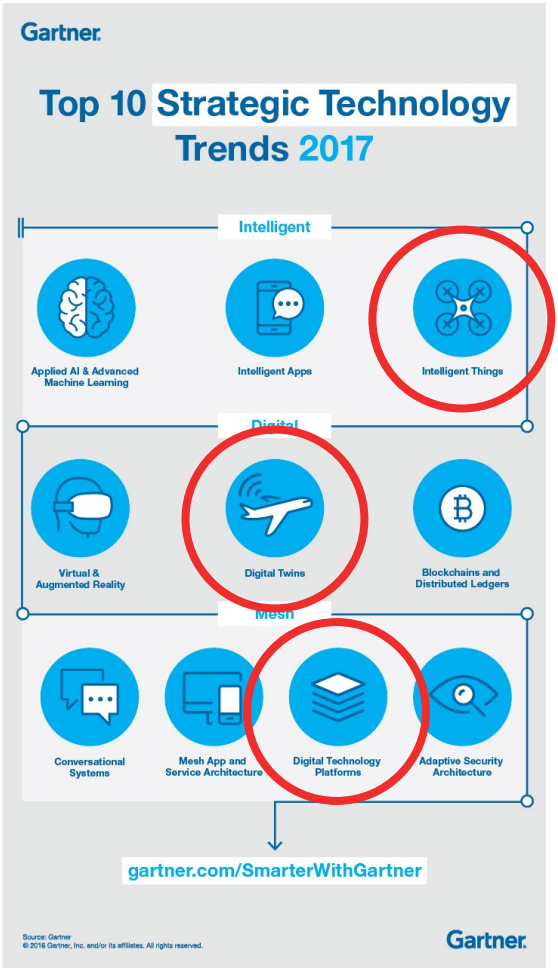
Fuente (Gartner, Inc, 2015)

4.3.2. Las 10 principales tendencias tecnológicas estratégicas 2017

La tendencia tecnológica para el 2017, dada por la consultora Garner, Inc. se puede apreciar en figura 41, y fue dada por:

- Applied AI & advanced machine learning
- Intelligent apps
- Intelligent things
- Virtual & augmented reality
- Digital twins
- Blockchains and distributed ledgers
- Conversational systems
- Mesh app and service architecture
- Digital technology platforms
- Adaptive security architecture

Figura 41. Top 10 Strategic Technology Trends 2017



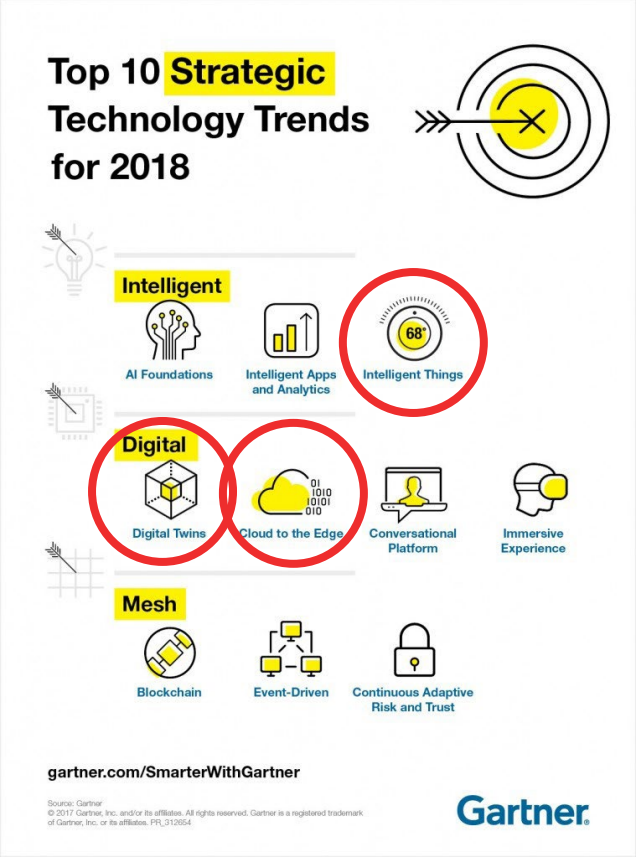
Fuente (Gartner, Inc, 2016)

4.3.3. Las 10 principales tendencias tecnológicas estratégicas 2018

La tendencia tecnológica para el 2018, presentada por la consultara Gartner, Inc. se muestra en la figura 42, y fue dada por:

- AI foundations
- Intelligent apps and analytics
- Intelligent things
- Digital twins
- Cloud to the edge
- Conversational platform
- Inmersive experience
- Blockchain
- Even-driven
- Continous adaptive risk and trust

Figura 42. Top 10 Strategic Technology Trends 2018



Fuente (Gartner, Inc, 2017)

4.3.4. Las 10 principales tendencias tecnológicas estratégicas 2019

La tendencia tecnológica para el 2019, presentada por la consultora Gartner, Inc. se muestra en la figura 43, y fue dada por:

- Autonomous things
- Digital twins
- Blockchain
- Augmented analytics
- ·Empowered edge
- Smart spaces
- AI-Driven develpment
- Inmersive technologies
- Digital ethics and privacy
- Quantum computing

Figura 43. Top 10 Strategic Technology Trends 2019

Fuente (Gartner, Inc, 2018)

4.3.5. Las 10 principales tendencias tecnológicas estratégicas 2020

La tendencia tecnológica para el 2020, presentada por la consultora Gartner, Inc. se muestra en la figura 44, y fue dada por:

- Hyperautomation
- Empowered edge
- Multiexperience
- Distributed cloud
- Democratization
- Autonomous things
- Human augmentation
- Practical blockchain
- Transparency and traceability
- AI security

Figura 44. Top 10 Strategic Technology Trends 2020

Fuente (Gartner, Inc, 2019)

4.4 CUADRO RESUMEN DE LAS TENDENCIAS TECNOLÓGICAS ALREDEDOR DEL IOT

Una vez vistas las tendencias tecnológicas a lo largo de estos últimos cinco años, observamos que invertir tiempo y recursos en aprender a desarrollar tecnologías que permitan controlar, programar, configurar los microcontroladores, microprocesadores y sus tarjetas de desarrollo, se convierte en una decisión acertada.

Tabla 11. Cuadro resumen de las tendencias tecnológicas alrededor del IoT para los últimos 5 años

Año	Tendencias tecnológicas relacionadas a los sistemas embebidos e IoT		
2016	The device mesh	Mesh app and service architecture	IoT architecture and platforms
2017	Intelligent things	Digital twins	Digital technology platforms
2018	Intelligent things	Digital twins	Cloud to the edge
2019	Autonomous things	Digital twins	Empowered edge / Smart spaces
2020	Autonomous things	Hyperautomation	Empowered edge

En la tabla 11 se puede apreciar una tendencia, desde los años 2017 al 2020, hacia las tecnologías como *Intelligent things* (Objetos inteligentes), *Digital twins* (Gemelos digitales) y *Autonomous things* (Dispositivos autónomos), las cuales están muy relacionadas al uso de sistemas embebidos. Por otro lado, se observan tecnologías relacionadas con la comunicación de todos los dispositivos embebidos o IoT hacia la nube, con *Empowered edge,* cuyo fin es mejorar las comunicaciones con *la nube* mediante routers, switches y dispositivos que pre-procesen toda esa gran cantidad de datos provenientes desde todos los dispositivos inteligentes.

CAPÍTULO 5.

VISIÓN GLOBAL DE UN SISTEMA EMBEBIDO: OBTENCIÓN, TRANSMISIÓN, PROCESAMIENTO Y ANÁLISIS DE LOS DATOS

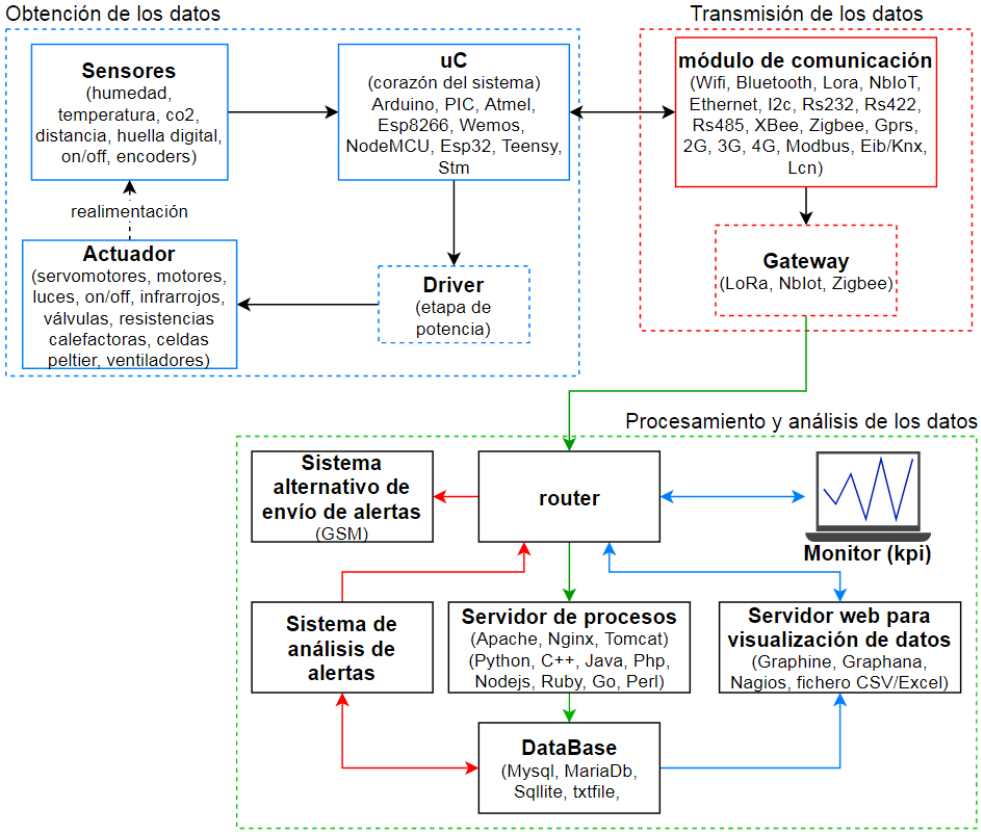
Cuando se desarrolla un proyecto para un sistema, se necesita tener en mente una foto del sistema en general. En este capítulo se muestra un sistema con todas sus partes, haciendo uso de los conocimientos que se han dado en los capítulos anteriores.

5.1 MÓDULOS DEL PROYECTO

En este apartado se verá de manera global cómo está constituido un proyecto, desde la toma de datos con los sensores, hasta el análisis de las alertas y su envío. Es importante saber hasta qué punto se va a realizar el proyecto, para no hacer uno muy pequeño, que sea muy básico o elemental, o uno tan complejo que no pueda ser finalizado en el plazo deseado.

En la siguiente figura se verá cada uno de los módulos que componen un sistema embebido con toma de datos, proceso y análisis de la información, así como también algunos ejemplos de los tipos de circuitos con los que se pueden encontrar mientras se diseña el proyecto.

Figura 45. Sistema embebido – obtención, transmisión, procesamiento y análisis de los datos



La figura anterior muestra la relación entre los distintos módulos del sistema. Se debe recordar que el tamaño y complejidad del sistema dependerá de los desarrolladores. A continuación se describe cada uno de dichos módulos.

5.1.1. Módulo de sensado

Si se desea obtener información del mundo exterior, estos circuitos son esenciales y es mediante los cuales obtendremos la información básica para desarrollar el proyecto. Se debe tener en cuenta las diferentes características de los sensores, como son, rango, resolución, repetitividad, etc., por cuanto si no son los sensores adecuados o si no tenemos cuidado al interpretar los datos podríamos obtener información errónea. Podrán ser sensores de humedad, temperatura, CO2, de distancia, huella digital, de on/off, encoders, etc.

5.1.2.BMódulo de procesamiento central: µC

Este módulo se encargará del procesamiento central de la información tomada a través de los sensores, procesamiento que se realiza usando un microcontrolador o placa adecuada para recibir el dato proveniente del sensor o los sensores (como se vio en el apartado 2.3, en donde se alcanzó un listado de microcontroladores y placas de desarrollo). Luego, se analiza y se procesa la información rescatada por los sensores.

5.1.3.BMódulo de potencia o Driver

Este módulo está conformado por un elemento de potencia o *driver* encargado de suministrar corriente al actuador que se desea controlar. Se obtienen drivers puente H, drivers Pololu, drivers MOSFET, transistores, relés, etc.

5.1.4. Módulo actuador

En este módulo se tendrán servomotores, motores, luces, dispositivos on/off, infrarrojos, válvulas, resistencias calefactoras, celdas peltier, ventiladores, etc.

5.1.5. Módulo de comunicación

En este apartado se cuenta módulos de comunicación para Wifi, Bluetooth, LoRa, NbIoT, Ethernet, I2c, Rs232, Rs422, Rs485, XBee, Zigbee, Gprs, 2G, 3G, 4G, Modbus, Eib / Knx, Lcn, etc.

5.1.6. Gateway

También se podrá contar con gateways para protocolos que aún no son suficientemente conocidos como son LoRa, NbIoT, Zigbee, etc.

5.1.7. Router

En este módulo será necesario utilizar un dispositivo encargado de llevar los datos de la red privada a la red pública y viceversa.

5.1.8. Servidor de procesos

El servidor de procesos podrá ser uno que los mismos desarrolladores programen. o también ser algún framework que ya exista como Apache, Nginx, Tomcat, etc. Y estos servidores de procesos podrán utilizar lenguajes de programación como Python, C/C++, Java, Javascript, Jsp, Php, Nodejs, Ruby, Go, Perl, etc. Algunas personas serán capaces de utilizar una solución más completa como Wamp o Lamp que incluyen bases de datos (Wamp o Lamp no está recomendado para servicios finales de producción).

Actualmente también existen muchos servicios en la nube a los que es factible subscribir, como Google Cloud, Amazon WS, Microsoft Azure, entre otros.

5.1.9. Base de datos

Entre las bases de datos que a utilizar se tiene a MySql, MariaDb, SqlLite, PostgreSQL, Mongo DB, txt file (un fichero de texto simple), csv file (fichero de datos separados por comas), etc.

5.1.10. Servidor web para visualización de datos

Si se desea visualizar los datos almacenados, se utilizarían servidores como Graphite, Graphana, Nagios o incluso hojas de cálculo (utilizando ficheros csv), etc.

5.1.11. Monitor de Kpis

Un Kpi, o indicador clave de rendimiento, es aquella medición que se considerará clave para medir el rendimiento general del sistema.

5.1.12. Sistema de análisis de alertas

Este módulo estará compuesto por un proceso o procesos que analizarán los datos cada equis minutos con el fin de encontrar anomalías, valores máximos, mínimos o valores fuera de rangos de los datos sensados, con el fin de disparar una alerta que deberá ser procesada inmediatamente o lo antes posible por un equipo de trabajo. Estas alertas serán guardadas en una base de datos con el fin de analizar su periodicidad.

5.1.13. Sistema alternativo de envío de alertas

Generalmente, la forma principal de comunicar datos es mediante Internet, pero se debe considerar eventos en el que no se tenga acceso a Internet, entonces se incluirán sistemas como el de envío de mensajes de texto o SMS.

5.2 EJEMPLO DE PROYECTOS DESARROLLADOS

Aquí se mencionan, de manera resumida, algunos de los proyectos que se realizaron en el curso de Sistemas Embebidos del Programa de Estudios de Ingeniería Electrónica de la Universidad Privada Antenor Orrego, los cuales servirán como ejemplos de proyectos que orientan, de forma práctica, el aprendizaje del lector.

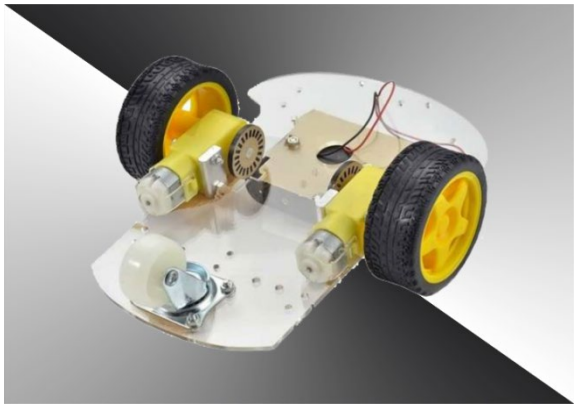
5.2.1 Seguidor de línea negra con compuertas lógicas

Este proyecto describe el procedimiento que se llevó a cabo para el diseño y montaje de un carrito seguidor de línea negra, con todas sus partes. El comportamiento del circuito está constituido en base a compuertas lógicas alimentado por un sistema de voltaje transformado a 5.0 V que

entrega directamente el Driver L298n (circuito electrónico utilizado para dar potencia a motores). Este circuito recibe la información de un módulo de sensores infrarrojos, estos detectan el camino o guía basado en una cinta negra que mide aproximadamente 2 cm de ancho y un fondo blanco; además, el Driver L298n está alimentado con 7.4V (2 amperios) para sus salidas conectadas a los 2 motores reductores DC (corriente directa) con sus respectivas ruedas y una rueda de giro libre para facilitar el movimiento en el camino (Blanco-Lezama, Diestra-Flores, Santillan-Requelme, Moreno-Barrientos, & Alva-Alarcón, 2018).

La figura 46 exhibe el chasis del carrito seguidor de línea que se utilizó.

Figura 46. Chasis del carrito seguidor de línea



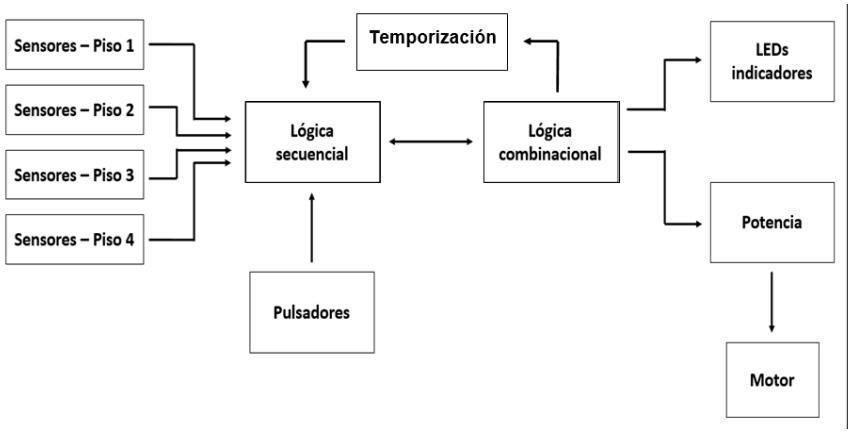
Fuente (Blanco-Lezama, Diestra-Flores, Santillan-Requelme, Moreno-Barrientos, & Alva-Alarcón, 2018)

5.2.2 Ascensor digital a escala de cuatro pisos

Exhibe este trabajo el desarrollo de un prototipo a escala de un ascensor de cuatro pisos, utilizando lógica digital (combinacional y secuencial). Se trabajó con compuertas lógicas, latches, registros y timers. El principio de funcionamiento de este proyecto es el mismo que el de un ascensor real, exceptuando la lógica de prioridades y los sistemas de seguridad. Este ascensor cuenta con almacenamiento de llamadas, puerta automática con temporización, display indicador del piso y los pulsadores tanto del ascensor como los de llamada (Moncada-Calmet, Ventura-Caballero, Rodríguez-Masumura, García-Honores, Campos-Chiang, & Alva-Alarcón, 2018).

En la figura 47 se presenta el diagrama de bloques diseñado para el ascensor digital a escala de cuatro pisos.

Figura 47. Diagrama de bloques del ascensor digital a escala de 4 pisos



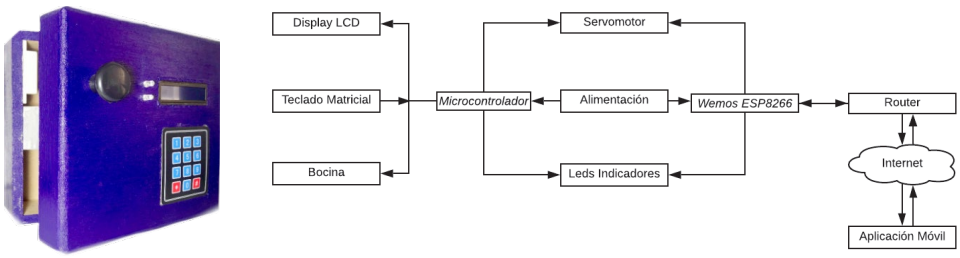
Fuente (Moncada-Calmet, Ventura-Caballero, Rodríguez-Masumura, García-Honores, Campos-Chiang, & Alva-Alarcón, 2018)

5.2.3 Cerradura electrónica controlada por wifi para un prototipo de caja fuerte

Este proyecto presenta el prototipo de un sistema de cerradura electrónica para una caja fuerte que es accionado por un teclado físico y por un aplicativo móvil conectado a la red. Se trabajó con un microcontrolador y un módulo Wemos-ESP8266 (tarjeta electrónica de desarrollo orientada al Internet de las cosas). El principal funcionamiento de este proyecto es el mismo que el de una caja fuerte convencional, con la diferencia que este cuenta con una aplicación en el celular para desbloquearlo de manera remota. Este prototipo cuenta con un teclado matricial para ingresar la contraseña, una pantalla LCD para visualizar los mensajes y un servomotor (motor especial utilizado en un sistema de control) para accionar el seguro de la puerta (Barrionuevo-Infante, De-La-Cruz-Mariños, Pantigoso-Ferreyra, & Alva-Alarcón, 2019).

La figura 48 muestra tanto la vista frontal como el diagrama de bloques de la caja fuerte con control digital y acceso wifi desarrollada.

Figura 48.Vista frontal y diagrama de bloques de la caja fuerte



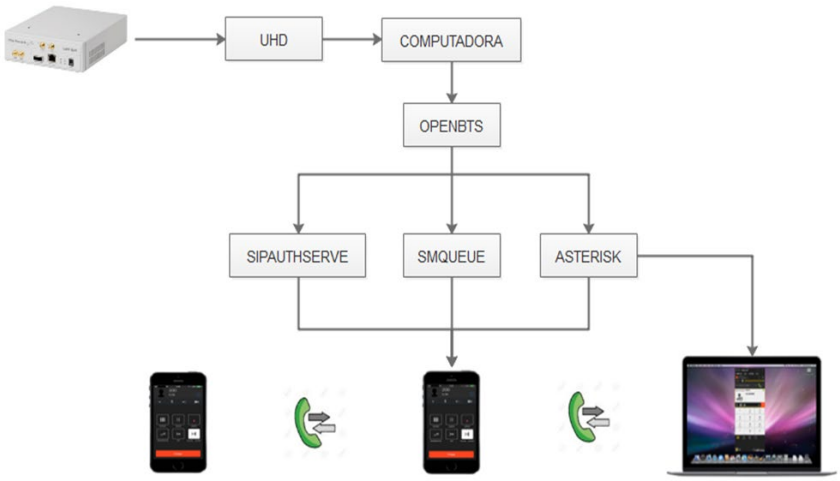
Fuente (Barrionuevo-Infante, De-La-Cruz-Mariños, Pantigoso-Ferreyra, & Alva-Alarcón, 2019)

5.2.4 Implementación de una red celular GSM mediante software OPENBTS

El presente trabajo empleó herramientas de hardware y software de licencia libre para el establecimiento de una estación base celular (BTS) a baja escala. Partiendo de conceptos técnicos que facilitan la instalación del sistema Operativo Ubuntu (dstribución Linux), el software libre OpenBTS y empleando el hardware USRP N200 (Universal Software Radio Peripheral), se desplegó una red análoga al estándar de telefonía móvil (GSM), tal como se establece en Perú. Fueron usados los teléfonos móviles como extensiones SIP (Session Initiation Protocol), desde el sistema Asterisk (software con funcionalidades de gestión telefónica), logrando ejecutar llamadas entre los terminales, mensajes de texto (SMS), llamadas desde un terminal OpenBTS hacia otra terminal móvil, entre otros servicios (Araujo-Garcia, Pomatanta-Rodríguez, Ñanez-Ruiz, & Alva-Alarcón, 2019).

La figura 49 muestra el diagrama de bloques del sistema GSM desarrollado.

Figura 49. Diagrama de bloques del sistema GSM



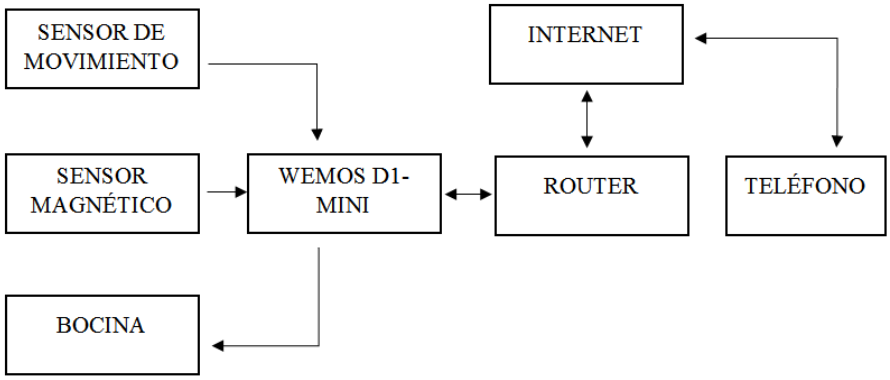
Fuente (Araujo-Garcia, Pomatanta-Rodríguez, Ñanez-Ruiz, & Alva-Alarcón, 2019)

5.2.5 Sistema de alarma doméstica a escala controlado por un aplicativo móvil

El proyecto presenta el diseño e implementación de una alarma doméstica a escala. Para ello se hizo uso de la tarjeta electrónica de desarrollo Wemos D1 mini, el entorno de desarrollo Arduino IDE y la plataforma para aplicaciones IoT, Blynk. El objetivo del funcionamiento es brindar seguridad, detectando movimientos no habituales en el hogar, enviando alertas al teléfono inteligente enlazado y, a su vez, controlando este sistema desde un aplicativo móvil; sin importar la distancia a la que el usuario se encuentre (Padilla-Alayo, Acevedo-Celis, Dionicio-Guzmán, & Alva-Alarcón, 2019).

La figura 50 muestra el diagrama de bloques del sistema de alarma doméstico desarrollado.

Figura 50. Diagrama de bloques del sistema de alarma doméstica



Fuente (Padilla-Alayo, Acevedo-Celis, Dionicio-Guzmán, & Alva-Alarcón, 2019)

BIBLIOGRAFÍA

Alva-Alarcón, J. L., & Jardón-Huete, A. (2018). Sistema amigable de control de entorno basado en realidad aumentada y reconocimiento de voz. Pueblo *Continente, 29(2)*, 355-365.

Apaza-Condori, D. (2010). *Microcontroladores PIC. Fundamentos y aplicaciones. Un enfoque didáctico.* Arequipa: Universida Autónoma San Francisco.

Araujo-Garcia, J. H., Pomatanta-Rodríguez, C. J., Ñanez-Ruiz, F. E., & Alva-Alarcón, J. L. (2019). Implementación de una red celular GSM mediante software OPENBTS. *Pueblo Continente*, 101-108.

Arduino. (2020). *Arduino*. Recuperado el 11 de 03 de 2020, de https://www.arduino.cc/en/Main/Software

AulaPlaneta. (02 de 12 de 2015). *Cómo aplicar el aprendizaje basado en proyectos en diez pasos.* Recuperado el 10 de 01 de 2020, de https://www.aulaplaneta.com/2015/02/04/recursos-tic/como-aplicar-el-aprendizaje-basado-en-proyectos-en-diez-pasos/

Barrionuevo-Infante, C. A., De-La-Cruz-Mariños, G. F., Pantigoso-Ferreyra, P. C., & Alva-Alarcón, J. L. (2019). Cerradura electrónica controlada por WiFi para un prototipo de caja fuerte. *Pueblo Continente, 30*(1), 109-116.

Blanco-Lezama, A., Diestra-Flores, J., Santillan-Requelme, P., Moreno-Barrientos, L., & Alva-Alarcón, J. L. (2018). Seguidor de línea negra con compuertas lógicas. *Pueblo Continente, 29*(2), 385-389.

Carretero, R. (2017). *Introducción al IOT "Un nuevo concepto de domótica ha llegado".* Recuperado el 12 de 03 de 2020, de http://raulcarretero.com/2018/03/18/introduccion-al-iot-un-nuevo-concepto-de-domotica/

Casanova, S. (2016). *Estimación ágil con la técnica Planning Poker.* Recuperado el 17 de 03 de 2020, de https://samuelcasanova.com/2016/01/estimacion-agil-con-la-tecnica-planning-poker/

Choque M., C., Linares O., P., Alcorta S., N., Alva A., J., & Prado G., S. (2019). IoT Automated Orchard in a Domestic Environment. *IEEE Xplore Digital Library.*

Crownhill Associates. (2020). *Proton Basic.* Recuperado el 13 de 04 de 2020, de http://www.protonbasic.co.uk/content.php/118-ProtonBASIC

Digité. (2016). *What is Kanban?* Recuperado el 18 de 03 de 2020, de https://www.digite.com/kanban/what-is-kanban/

Eclipse Foundation. (2020). *Eclipse desktop & web IDEs*. Recuperado el 18 de 03 de 2020, de https://www.eclipse.org/ide/

EDUforics. (30 de 04 de 2017). *Aprendizaje basado en proyectos. Cómo hacer que un proyecto sea auténtico y real.* Recuperado el 10 de 01 de 2020, de http://www.eduforics.com/es/aprendizaje-basado-proyectos/

Gartner Inc. (2020). *Smarter With Gartner.* Recuperado el 12 de 03 de 2020, de https://www.gartner.com/smarterwithgartner

Gartner, Inc. (06 de 10 de 2015). *Top 10 Strategic Technology Trends for 2016: At a Glance.* Obtenido de https://www.gartner.com/en/documents/3143618/top-10-strategic-technology-trends-for-2016-at-a-glance

Gartner, Inc. (18 de 10 de 2016). *Gartner's Top 10 Strategic Technology Trends for 2017.* Obtenido de https://www.gartner.com/smarterwithgartner/gartners-top-10-technology-trends-2017/

Gartner, Inc. (3 de 10 de 2017). *Gartner Top 10 Strategic Technology Trends for 2018.* Obtenido de https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2018

Gartner, Inc. (15 de 10 de 2018). *Gartner Top 10 Strategic Technology Trends for 2019.* Obtenido de https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2019/

Gartner, Inc. (21 de 10 de 2019). *Gartner Top 10 Strategic Technology Trends for 2020.* Obtenido de https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2020/

Ho, D. (2020). *What is Notepad++.* Recuperado el 18 de 03 de 2020, de https://notepad-plus-plus.org/

Incibe-Cert. (08 de 02 de 2018). *Introducción a los sistemas embebidos*. Recuperado el 08 de 01 de 2020, de https://www.incibe-cert.es/blog/introduccion-los-sistemas-embebidos

Jet Brains. (2020). *PyCharm: el IDE de Python para desarrolladores profesionales.* Recuperado el 18 de 03 de 2020, de https://www.jetbrains.com/es-es/pycharm/

JetBrains. (2020). *IntelliJ IDEA: El entorno de desarrollo integrado de Java para desarrolladores.* Recuperado el 18 de 03 de 2020, de https://www.jetbrains.com/es-es/idea/

Keyur K, P., & Sunil M, P. (2016). Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges. *International Journal of Engineering Science and Computing, 6*(5), 6122-6131.

Labcenter Electronics Ltd. (2020). *Proteus Design Suite.* Recuperado el 12 de 03 de 2020, de https://www.labcenter.com/

Microchip Technology Inc. (2020). *MPLAB® X Integrated Development Environment (IDE).* Recuperado el 11 de 03 de 2020, de https://www.microchip.com/mplab/mplab-x-ide

Microsoft Corporation. (2020). *Desarrolle aplicaciones con C y C++*. Recuperado el 18 de 03 de 2020, de https://visualstudio.microsoft.com/es/vs/features/cplusplus/

Microsoft Corporation. (2020). *Microsoft Power Point.* Recuperado el 17 de 03 de 2020, de https://products.office.com/es/powerpoint

Microsoft Corporation. (2020). *Visual Studio Code - Code Editing.* Redefined. Recuperado el 09 de 04 de 2020, de https://code.visualstudio.com/

Minisini, B. (2020). *Gambas Almost Means BASIC!* Recuperado el 09 de 04 de 2020, de http://gambas.sourceforge.net/en/main.html

Moncada-Calmet, L. F., Ventura-Caballero, C. J., Rodríguez-Masumura, S. H., García-Honores, M. G., Campos-Chiang, A. A., & Alva-Alarcón, J. L. (2018). Ascensor digital a escala de 4 pisos. *Pueblo Continente, 29*(2), 379-383.

OBS Business School. (24 de 10 de 2018). *¿Qué es un diagrama de Gantt y para qué sirve?* Recuperado el 10 de 01 de 2020, de https://obsbusiness.school/es/blog-project-management/diagramas-de-gantt/que-es-un-diagrama-de-gantt-y-para-que-sirve

Padilla-Alayo, C. M., Acevedo-Celis, P. Z., Dionicio-Guzmán, D. A., & Alva-Alarcón, J. L. (2019). Sistema de alarma doméstica a escala controlado por un aplicativo móvil. *Pueblo Continente,* 93-99.

Seibert Media Corp. (2020). *Draw.io.* Recuperado el 17 de 03 de 2020, de https://drawio-app.com/

Skinner, J. (2020). *Sublime Text - A sophisticated text editor for code, markup and prose*. Recuperado el 18 de 03 de 2020, de https://www.sublimetext.com/

STMicroelectronics. (2020). *STM32CubeMX - STM32Cube initialization code generator - STMicroelectronics.* Recuperado el 03 de 12 de 2020, de https://www.st.com/en/development-tools/stm32-software-development-tools.html

The Code::Blocks team. (2020). *Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms*. Recuperado el 18 de 03 de 2020, de http://www.codeblocks.org/

The Dia Developers. (2014). Dia Diagram Editor. Recuperado el 17 de 03 de 2020, de http://dia-installer.de/

Wikipedia: *Domótica. (*2020). Domótica. Recuperado el 17 de 03 de 2020, de https://es.wikipedia.org/wiki/Dom%C3%B3tica

Wikipedia: Lenguaje ensamblador. (2020). *Lenguaje ensamblador.* Recuperado el 11 de 03 de 2020, de https://es.wikipedia.org/wiki/Lenguaje_ensamblador

ANEXO

Anexo 1. Configuración de una placa Raspberry Pi y ejercicios

Como la placa Raspberry Pi es de las placas más utilizadas, se describe a continuación un tutorial de cómo configurarla y empezar a utilizarla. Además se proponen 4 ejercicios con esta placa.

Materiales necesarios:

Item	Descripción
Computadora de placa reducida	En este caso se utilizará una Raspberry Pi 3B o Raspberry Pi 3B+, preferentemente con su case.
Memoria micro SD	Para grabar el sistema operativo, esta memoria debe tener una capacidad mayor a 8GB. Se recomienda una memoria clase 10.
Adaptador para memoria micro SD	Este adaptador permitirá grabar el SO utilizando la Laptop. Puede ser un adaptador SD a micro SD, o un adaptador USB a micro SD.
Fuente de alimentación	Debe ser una fuente de 5 voltios con salida micro USB (similar a la usada para cargar los celulares). Se recomienda una fuente de alimentación de más de 2 amperios.
Jumpers, resistencias y leds	Para hacer pruebas del puerto GPIO de la Raspberry.
Laptop o una computadora	En la que se pueda trabajar y grabar el sistema operativo.
Monitor, teclado y mouse	Es opcional. En caso de no tenerlos se trabajará por SSH o VNC (instalación 'headless')
Cable de red	Para conectar inicialmente la Raspberry al router.
Router o switch	Preferentemente que sea administrable y así poder ver las IPs de los clientes conectados.
Conexión a Internet	Con el fin de poder realizar las instalaciones de los paquetes requeridos para los diferentes ejercicios.

Pasos:

1. Copiar todos los ficheros e instaladores necesarios en la Laptop. Estos archivos se listan a continuación:
   a. Imagen de nuestro sistema operativo Raspbian.
   b. Software SD-Card-Formater.
   c. Software 7-zip
   d. Software Rufus
   e. Herramienta Putty
   f. Cliente VNC

2. Copiar la imagen del sistema operativo
    a. Poner la memoria micro SD en su adaptador y conectarla a nuestra PC.
    b. Formatear la SD utilizando el software SD-Card-Formater.
    c. Descomprimir la imagen xxx-raspbian-buster-full.zip (usar preferentemente el programa 7-Zip).
    d. Copiar el fichero imagen (.img) en nuestra SD utilizando la herramienta Rufus.
    e. [Opcional]. En caso de realizar una instalación "headless" (sin monitor ni teclado ni mouse), crear un fichero vacío con el nombre de "ssh" en la memoria micro SD.
    f. Retirar la memoria micro SD y colocarla en la Rasbperry.
3. Realizar el cableado necesario y al último conectar la alimentación.

NOTA. Evitar desenchufar la Raspberry sin antes haberla apagado por comandos, ya que en ciertos casos la memoria SD se está escribiendo y podría dañarse si se retira la alimentación de golpe.

4. Se conecta la alimentación y se debe esperar a que el sistema operativo arranque.

OPCIONAL. Para el caso de una instalación *"headless"* se debe obtener la IP de la Raspberry. Una opción es conectarla por cable Ethernet al router y ver qué IP se le asignó. Cuando es conocida la IP, se procederá a conectarse por *ssh* utilizando la herramienta *"putty"*. Una vez conectados por *ssh* se procederá a habilitar la conexión *vnc* si se desea conectarse gráficamente. Si se quiere, también se podrá realizar una conexión desde el celular si previamente se ha instalado el cliente *vnc*. Para aumentar el tamaño de la ventana VNC utilizar: *vncserver -randr=800x600*.

5. Configuración de la Rasbperry. Para hacer esta configuración se podrá hacer uso de las herramientas gráficas que se encuentran en "Inicio / preferencias", o hacer uso del comando:

    sudo raspi-config

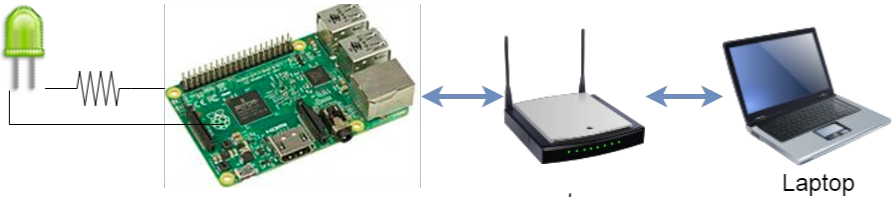
    Así se tendrá acceso a configurar los siguientes puntos importantes:

    a. País
    b. Disposición del teclado
    c. Wifi
    d. Expansión de la memoria SD (Expandir sistema de ficheros)
    e. Se debe actualizar las librerías con:

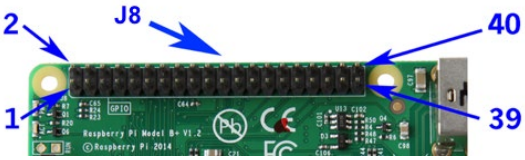
    sudo apt-get update
    sudo apt upgrade

Ejercicios

Ejercicio 1. Encendido de un Led



Distribución de los pines:



Raspberry Pi 3 Model B (J8 Header)

GPIO#	NAME			NAME	GPIO#
	3.3 VDC Power	1	2	5.0 VDC Power	
8	GPIO 8 SDA1 (I2C)	3	4	5.0 VDC Power	
9	GPIO 9 SCL1 (I2C)	5	6	Ground	
7	GPIO 7 GPCLK0	7	8	GPIO 15 TxD (UART)	15
	Ground	9	10	GPIO 16 RxD (UART)	16
0	GPIO 0	11	12	GPIO 1 PCM_CLK/PWM0	1
2	GPIO 2	13	14	Ground	
3	GPIO 3	15	16	GPIO 4	4
	3.3 VDC Power	17	18	GPIO 5	5
12	GPIO 12 MOSI (SPI)	19	20	Ground	
13	GPIO 13 MISO (SPI)	21	22	GPIO 6	6
14	GPIO 14 SCLK (SPI)	23	24	GPIO 10 CE0 (SPI)	10
	Ground	25	26	GPIO 11 CE1 (SPI)	11
30	SDA0 (I2C ID EEPROM)	27	28	SCL0 (I2C ID EEPROM)	31
21	GPIO 21 GPCLK1	29	30	Ground	
22	GPIO 22 GPCLK2	31	32	GPIO 26 PWM0	26
23	GPIO 23 PWM1	33	34	Ground	
24	GPIO 24 PCM_FS/PWM1	35	36	GPIO 27	27
25	GPIO 25	37	38	GPIO 28 PCM_DIN	28
	Ground	39	40	GPIO 29 PCM_DOUT	29

Attention! The GIPO pin numbering used in this diagram is intended for use with WiringPi / Pi4J. This pin numbering is not the raw Broadcom GPIO pin numbers.

http://www.pi4j.com

Pasos:

1. Si se tiene dudas sobre cómo están distribuidos los pines de la Raspberry, se puede ejecutar el comando:

   pinout

2. Se ejecuta el comando de apagado de la Raspberry con el comando:

   sudo shutdown –h now

3. Se debe esperar unos segundos hasta observar que el led de encendido de la Raspberry se encuentra apagado.
4. Se conecta la resistencia y el led al puerto GPIO 9 siguiendo la polaridad correcta.
5. Se debe entrar a una consola del sistema y utilizando Python se deberán ejecutar los siguientes comandos:

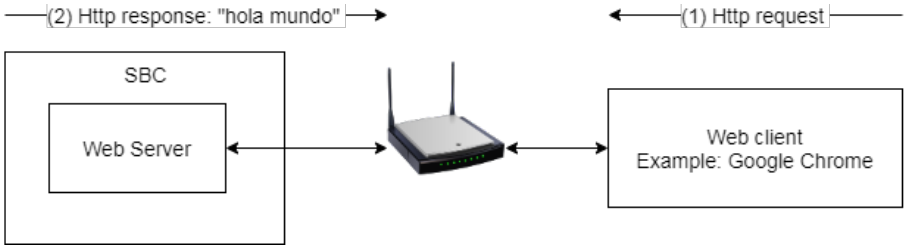
   python
   from gpiozero import LED
   led = LED(9)
   led.on( )
   led.off( )
   quit( )

6. También se puede utilizar comandos propios del shell como:

   echo 9 > /sys/class/gpio/export
   echo out > /sys/class/gpio/gpio9/direction
   echo 1 > /sys/class/gpio/gpio9/value
   echo 0 > /sys/class/gpio/gpio9/value
   echo 9 > /sys/class/gpio/unexport

Extra: en este ejercicio se han utilizado los puertos GPIO para encender y apagar un led, pero los puertos GPIO también pueden utilizarse para conexión PWM, SPI, I2C, Serial.

Ejercicio 2. “Hola mundo” para un servidor web



Resumen:

Este ejercicio muestra cómo se puede conectar la SBC utilizando el protocolo HTTP. Para esto, se deberá programar un pequeño servidor web en Python utilizando el framework ‘Flask’.

Pasos:

1. Se descargará la librería ‘Flask’ para Python con el comando:

   pip install Flask

2. Se creará un fichero llamado ‘servidor.py’ con el siguiente contenido:

```
#!/usr/bin/python
from flask import Flask, request
app = Flask(__name__)
@app.route("/")
def hello():
return "Hola mundo"

if __name__ == "__main__":
app.run(debug=True, host='0.0.0.0', port=5000)
@app.errorhandler(500)
def server_error(e):
return 'An internal error occurred.' + str(e), 500
```

3. Se arrancará el servidor con el comando:

   python servidor.py

4. Por defecto, si no se ha especificado ningún puerto, el servidor arrancará escuchando el puerto 5000.

5. Desde la propia minicomputadora, en un navegador web, para acceder al servidor que se ha creado se copiará la siguiente url:

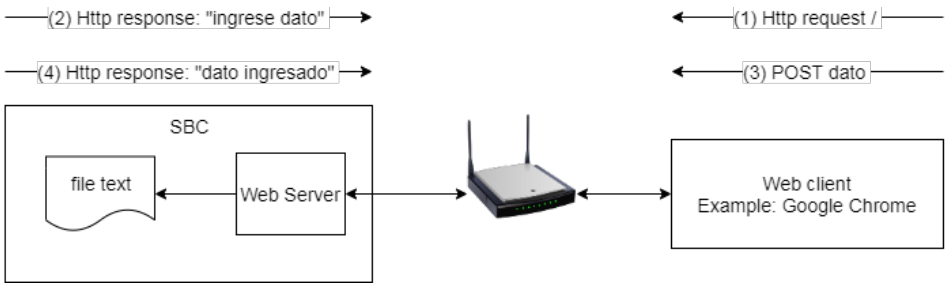
   http://localhost:5000

6. Se debe ver que responde con el texto “hola mundo”.

7. Si se desea hacer peticiones desde otra computadora, entonces se deberá utilizar la url:

   http://<ip_de_la_raspberry>:5000

Ejercicio 3. Grabar datos recibidos por peticiones http



Resumen:

En este ejercicio se desarrolla un servidor web que recibe los datos que vienen de otros dispositivos y los guarda en un fichero con formato *csv*.

Pasos:

1. Con la librería 'flask' previamente descargada, se procederá a crear un fichero de texto con el nombre de 'servidor2.py' y el siguiente contenido:

```
#!/usr/bin/python
from flask import Flask, request
f = open("datos.csv","w+")
app = Flask(__name__)
@app.route("/")
def hello():
   return '''<form action="" method="post">
   Texto: <input type="text" name="dato">
   <input type="submit" value="Enviar">
</form>'''
@app.route('/', methods=['POST'])
def recibeDato():
   dato = request.form["dato"]
   print(dato)
   f.write("dato; %s\n" % dato)
   f.flush()
   return "recibido dato: " + str(dato)
@app.errorhandler(500)
def server_error(e):
   return 'An internal error occurred.' + str(e), 500
if __name__ == "__main__":
   app.run(debug=True, host='0.0.0.0', port=5000)
```

2. Se ejecuta el servidor con el siguiente comando:

   python servidor2.py

3. Se realiza una petición web desde la propia Raspberry con la siguiente url:

   http://localhost:5000

4. Se ingresan los valores y se pulsa 'enviar'.
5. Se abre el fichero *csv* que se ha creado y se observan los valores guardados.

Ejercicio 4. Instalación de una base de datos en la Raspberry, con acceso desde las Laptops

Resumen:

Se instalará un gestor de bases de datos en la Raspberry. Posteriormente se creará una base de datos así como una tabla de datos. Se configurarán los permisos para poder acceder desde otra computadora a la base de datos previamente creada.

Pasos:

1. Instalar el servidor de MariaDB con los siguientes comandos:

   sudo apt-get install mariadb-server-10.0 --fix-missing

   sudo mysql_secure_installation

   pip install mysql-connector-python

2. Se permite el acceso a la base de datos desde otra computadora modificando:

   sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

   cambiar: bind-address = 0.0.0.0

3. Se crea un usuario que tenga permiso de acceso a una base de datos de prueba:

   sudo mysql -u root -p

   *GRANT ALL PRIVILEGES ON *.* TO 'usertest'@'%' IDENTIFIED BY 'passtest' WITH GRANT OPTION; flush privileges;*

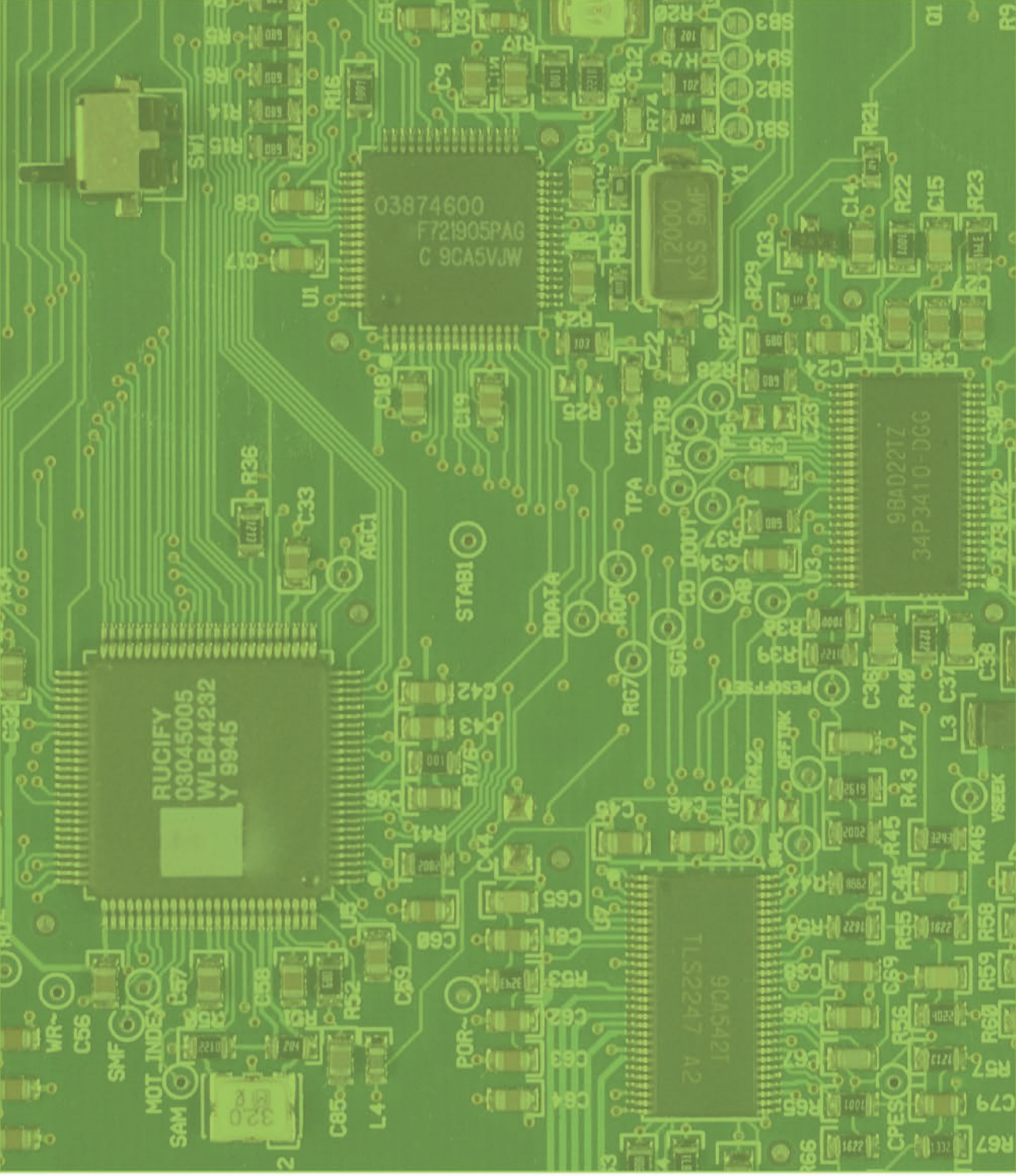
Como punto extra de este ejercicio se puede guardar en base de datos los valores enviados desde otra computadora con el siguiente código:

```
#!/usr/bin/python
from flask import Flask, request
import mysql.connector as mariadb
mariadb_connection = mariadb.connect(user='usertest',
password='passtest', database='test')
app = Flask(__name__)
@app.route("/")
def hello():
   return ''<form action="" method="post">
   Texto: <input type="text" name="dato">
   <input type="submit" value="Submit">
  </form>'''
@app.route('/', methods=['POST'])
ef recibeDato():
dato = request.form["dato"]
print("Dato recibido: " + dato)
cursor = mariadb_connection.cursor()
cursor.execute("INSERT INTO Datos (valor) VALUES (%d)"%
(int(dato)))
mariadb_connection.commit()
mariadb_connection.close()
return "recibido dato: " + str(dato)

@app.errorhandler(500)
def server_error(e):
        return 'An internal error occurred.' + str(e), 500

if __name__ == "__main__":
        app.run(debug=True, host='0.0.0.0', port=5000)
```

Anexo 2. Ficha de seguimiento semanal de proyectos

Asignatura:		Semestre
Nombre del proyecto:		
Integrantes:		
Resumen del proyecto:		
Diagrama de bloques:		
Cronograma:		
Componentes:	Tecnologías / herramientas / software:	
Factores externos:		

S	Apuntes	Revisión	Nota
2			
3			
4			
5			
6			
7			
9			
10			
11			
12			
13			
14			



UPAO | FONDO EDITORIAL