Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

DISEÑO DE LA ESTRUCTURA VIRTUAL DEL CURSO DE ELECTRÓNICA 5, APLICADO A CONFIGURACIÓN DE INTERRUPCIONES Y PUERTOS EN LENGUAJE ENSAMBLADOR UTILIZANDO EL CONTROLADOR TM4C123GH6PM, BAJO EL MODELO CONSTRUCTIVISTA DE EDUCACIÓN, EN LA ESCUELA DE INGENIERIA MECÁNICA ELÉCTRICA, FACULTAD DE INGENIERÍA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

Hilda Fabiola España Girón
Asesorado por la Inga. Ingrid Rodríguez

Guatemala, octubre de 2020

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO DE LA ESTRUCTURA VIRTUAL DEL CURSO DE ELECTRÓNICA 5, APLICADO A CONFIGURACIÓN DE INTERRUPCIONES Y PUERTOS EN LENGUAJE ENSAMBLADOR UTILIZANDO EL CONTROLADOR TM4C123GH6PM, BAJO EL MODELO CONSTRUCTIVISTA DE EDUCACIÓN, EN LA ESCUELA DE INGENIERIA MECÁNICA ELÉCTRICA, FACULTAD DE INGENIERÍA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

HILDA FABIOLA ESPAÑA GIRÓN
ASESORADO POR LA INGA. INGRID RODRÍGUEZ

AL CONFERÍRSELE EL TÍTULO DE

INGENIERA EN ELECTRÓNICA

GUATEMALA, OCTUBRE DE 2020

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martinez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Christian Moisés de la Cruz Leal
VOCAL V	Br. Kevin Vladimir Armando Cruz Lorente
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Pedro Antonio Aguilar Polanco
EXAMINADOR	Ing. Byron Odilio Arrivillaga Méndez
EXAMINADOR	Ing. Julio Rolando Barrios Archila
EXAMINADOR	Ing. Carlos Eduardo Guzmán Salazar
SECRETARIA	Inga. Lesbia Magalí Herrera López

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO DE LA ESTRUCTURA VIRTUAL DEL CURSO DE ELECTRÓNICA 5, APLICADO A CONFIGURACIÓN DE INTERRUPCIONES Y PUERTOS EN LENGUAJE ENSAMBLADOR UTILIZANDO EL CONTROLADOR TM4C123GH6PM, BAJO EL MODELO CONSTRUCTIVISTA DE EDUCACIÓN, EN LA ESCUELA DE INGENIERIA MECÁNICA ELÉCTRICA, FACULTAD DE INGENIERÍA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica con fecha 05 de junio de 2019.

Hilda Fabiola España Girón

Guatemala 6 de abril de 2020

Ingeniero
Julio César Solares Peñate
Coordinador del Área de Electrónica
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

Apreciable Ingeniero Solares,

Me permito dar aprobación al trabajo de graduación titulado ¨**Diseño de la estructura virtual del curso de Electrónica 5, aplicado a configuración de interrupciones y puertos en lenguaje ensamblador utilizando el controlador TM4C123GH6PM, bajo el modelo constructivista de educación, en la Escuela de Ingeniería Mecánica Eléctrica, Facultad de Ingeniería, Universidad de San Carlos de Guatemala**¨, de la señorita **Hilda Fabiola España Girón**, por considerar que cumple con los requisitos establecidos.

Por tanto, el autor de este trabajo de graduación y, yo, como su asesora, nos hacemos responsables por el contenido y conclusiones del mismo.

Sin otro particular, me es grato saludarle.

Atentamente,

Inga. Ingrid Rodríguez de Loukota
Colegiada 5,356
Asesora

Ingrid Rodríguez de Loukota
Ingeniera en Electrónica
colegiado 5356

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Guatemala, 12 de mayo de 2020

Señor Director
Armando Alonso Rivera Carrillo
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC

Estimado Señor Director:

Por este medio me permito dar aprobación al Trabajo de Graduación titulado **DISEÑO DE LA ESTRUCTURA VIRTUAL DEL CURSO DE ELECTRÓNICA 5, APLICADO A CONFIGURACION DE INTERRUPCIONES Y PUERTOS EN LENGUAJE ENSAMBLADOR UTILIZANDO EL CONTROLADOR TM4C123GH6PM, BAJO EL MODELO CONSTRUCTIVISTA DE EDUCACIÓN, EN LA ESCUELA DE INGENIERIA MECANICA ELECTRICA, FACULTAD DE INGENIERIA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA,** desarrollado por el estudiante **Hilda Fabiola España Girón**, ya que considero que cumple con los requisitos establecidos.

Sin otro particular, aprovecho la oportunidad para saludarlo.

Atentamente,

ID Y ENSEÑAD A TODOS

Ing. Julio César Solares Peñate
Coordinador de Electrónica

REF. EIME 242.2020.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de Área , al trabajo de Graduación de la estudiante Hilda Fabiola España Girón titulado: **DISEÑO DE LA ESTRUCTURA VIRTUAL DEL CURSO DE ELECTRÓNICA 5, APLICADO A CONFIGURACION DE INTERRUPCIONES Y PUERTOS EN LENGUAJE ENSAMBLADOR UTILIZANDO EL CONTROLADOR TM4C123GH6PM, BAJO EL MODELO CONSTRUCTIVISTA DE EDUCACIÓN, EN LA ESCUELA DE INGENIERIA MECANICA ELECTRICA, FACULTAD DE INGENIERIA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA,** procede a la autorización del mismo.

Ing. Armando Alonso Rivera Carrillo

Guatemala, 26 de julio de 2020.

Decanato
Facultad de Ingeniería
24189101- 24189102

DTG. 314.2020.

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Eléctrica, al Trabajo de Graduación titulado: **DISEÑO DE LA ESTRUCTURA VIRTUAL DEL CURSO DE ELECTRÓNICA 5, APLICADO A CONFIGURACIÓN DE INTERRUPCIONES Y PUERTOS EN LENGUAJE ENSAMBLADOR UTILIZANDO EL CONTROLADOR TM4C123GH6PM, BAJO EL MODELO CONSTRUCTIVISTA DE EDUCACIÓN, EN LA ESCUELA DE INGENIERIA MECÁNICA ELÉCTRICA, FACULTAD DE INGENIERÍA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA,** presentado por el estudiante universitario: **Hilda Fabiola España Girón,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

Inga. Anabela Cordova Estrada
Decana

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANA
FACULTAD DE INGENIERÍA

Guatemala, octubre de 2020

AACE/asga

ACTO QUE DEDICO A:

Dios	Por ser esa parte de mí que aun intangible fue gran parte de este logro.
Mis padres	Edwin España y Evelyn Girón, por siempre apoyarme, regañarme y estar allí cuando los necesitaba. Son los mejores papás que pude tener.
Mi hermano	Rodrigo España, por ser el mejor hermano y amigo, por su apoyo en las buenas y en las malas.
Mami	Hilda Gómez por ser ese angelito que siempre cuido de mí en la tierra, y que me cuida desde el cielo.
Mis abuelos	David Girón, Vitalina Rodas, Ruth Blanco, Guillermo España (q. e. p. d.) y nuevamente Hilda Gómez (q. e. p. d.), por todo su cariño.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por ser la institución que me permitió desarrollarme como profesional y haberme abierto las puertas tanto en aspecto estudiantil como laboral.
Facultad de Ingeniería	Por ayudarme a creer en mí misma y crecer como persona.
A mis tíos, primos y familiares cercanos	Por su apoyo durante no solo carrera sino mi vida en general. Por mencionar algunos, Chevito, Aldo, Tío Noldo, Mamá Carmen, Tío Jorge.
Mis amigos de la Facultad	Especialmente a Manuel Fernández, Miguel Tavico, José Monroy, Byron Paiz, por los repasos antes del examen, el bullying mutuo, los consejos y las comidas compartidas.
Bicicuates	Por hacerme notar lo fuerte que puedo ser si me lo propongo, por todas las amistades que surgieron.
Ingenieros de la Escuela de Ingeniería	Por su apoyo, dedicación y todo lo que me han enseñado. Todos y cada uno han sido parte

importante y fundamental del proceso que tuve en la facultad.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
∞	Infinito
µ	Micro
*****	Multiplicación
E+n	Notación exponencial en base 10 y potencia n positiva.
E-n	Notación exponencial en base 10 y potencia n negativa.
2_	Número binario
0x	Número hexadecimal
π	Número PI
+	Suma
∑	Sumatoria
#	Numeral o Número en programación

GLOSARIO

AND	Operación lógica que se comporta como una multiplicación normal, donde únicamente si ambas entradas son 1, la salida será 1. Cuando una o más sea 0, la salida será 0.
Aritmética	Parte de la matemática que estudia los números y las operaciones que se hacen con ellos.
BIT	*Binary digit* o dígito binario, es la unidad más pequeña de información y permite representar dos valores, 0 y 1.
BUS	En informática, bus es el conjunto de conexiones físicas que comparten los componentes de un microcontrolador.
CI	Circuito integrado, circuito electrónico cuyos componentes, como transmisores y resistencias, están dispuestos en una lámina de material semiconductor.
Código de máquina	Conjunto de instrucciones entendibles directamente por el ordenador, puesto que se componen de unos y ceros.

Display	Dispositivo de un aparato electrónico o pantalla donde se muestra visualmente cierta información.
Global	Una variable global es aquella que puede utilizarse desde cualquier función, generalmente declaradas al inicio del programa.
GPIO	*General Purpose Input Output*, es el sistema de entradas y salidas con las que se comunica el microcontrolador con circuitos externos.
Hipertextual	Son textos con múltiples autores, lo cual permite que cada uno aporte información al texto.
IEEE	Es una organización mundial de ingenieros eléctricos y electrónicos encargada de creación de normas o estándares relacionados a dichas ingenierías.
Interfaz	Dispositivo capaz de transformar las señales generadas por un aparato en señales comprensibles por otro.
Multitasking	Traducido normalmente como multitarea, es la capacidad de una persona o tecnología de realizar múltiples tareas o instrucciones.
Lenguaje de alto nivel	Lenguaje de programación que se caracteriza por ser de sencilla comprensión para el ser humano.

Lenguaje de programación	Es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo por máquinas como las computadoras.
OPCODE	Parte de los lenguajes de máquina que designa la operación que se debe realizar. Abreviatura de *operation code*.
OR	Realiza una suma lógica que da por resultado un 1 si una de las entradas es positiva y un 0 si todas las entradas son negativas.
Oscilador	Circuito electrónico que produce una señal oscilante y periódica; puede ser senoidal o cuadrada.
PIC	Es un circuito integrado programable que posee puertos que le permiten comunicarse con el exterior.
Polling	Es un sondeo constante, generalmente de periféricos externos, sin el uso de interrupciones.
Pseudoinstrucción	Conjunto de símbolos que tienen la apariencia de una instrucción. Es utilizada para dar órdenes a los programas de traducción.

Resistencia pull-up	La acción de pull-up en electrónica se asigna a la acción de elevar una tensión de entrada o salida que tiene un circuito lógico mientras este está en reposo.
Sintaxis	Conjunto de reglas que deben seguirse al escribir el código fuente de los programas.
Switch	También conocido como interruptor, es un dispositivo que permite desviar o interrumpir el curso de una corriente eléctrica.
USB	Es un bus de comunicaciones que sigue un estándar que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre computadoras, periféricos y dispositivos electrónicos.

RESUMEN

El trabajo de tesis de Diseño de la estructura virtual del curso de Electrónica 5, aplicado a configuración de interrupciones y puertos en lenguaje ensamblador utilizando el controlador TM4C123GH6PM, bajo el modelo constructivista de educación, en la Escuela de Ingeniería Mecánica Eléctrica, Facultad de Ingeniería, Universidad de San Carlos de Guatemala, pretende dar una guía introductoria básica a la programación utilizada en el laboratorio del curso Electrónica 5.

El primer capítulo será un esbozo de lo que se pretende hacer en el desarrollo de esta tesis, además de las metodologías que pueden y deben aplicarse en la enseñanza no solo en el laboratorio de electrónica, sino en cualquier otra área.

Los demás capítulos serán una recopilación de conceptos necesarios para la formación profesional de los estudiantes de Ingeniería Electrónica y para cualquier persona que desee instruirse en el mundo de la programación orientada a hardware, desde uno de los primeros lenguajes desarrollados.

Además, se grabará una serie de videos que servirán como material de apoyo al momento de realizar cualquier proyecto que el estudiante quiera.

OBJETIVOS

General

Desarrollar, mediante el método constructivista, los temas de programación en lenguaje ensamblador para el microcontrolador TM4C123GH6PM para el laboratorio de Electrónica 5.

Específicos

1. Realizar videos conceptuales que ayuden con el curso de Electrónica 5.

2. Introducir a los estudiantes a la programación en ensamblador.

3. Enseñar sobre la habilitación de puertos del microcontrolador TM4c123GH6PM.

INTRODUCCIÓN

Existen diversos métodos de enseñanza, los cuales están dirigidos a diferentes tipos de persona. Mientras que para algunos les basta una breve explicación sobre un tema, otros, en cambio, necesitarán practicar constantemente para comprenderlo. Se puede discernir entonces que no es suficiente, en algunos casos, la explicación recibida por parte del educador en clase; y que es necesario contar con un recurso extra para el adecuado aprendizaje de un curso.

Considerando lo anterior, con la utilización de las TICS —que son herramientas para el autoaprendizaje que ayudan a reforzar los conocimientos previamente adquiridos— se podrá acceder desde cualquier medio electrónico en el momento que un estudiante lo necesite.

1. EL CONSTRUCTIVISMO

Para comprender el constructivismo se debe tener claro que el conocimiento debe estar fundamentado; esto quiere decir que no solo se debe creer ciegamente en un concepto, sino que además debe estar fundamentado.

Según F.I. Classtchich "*El proceso del conocimiento humano se compone de dos momentos: El empírico o sensorial y el racional de la comprensión. Toda la base de uno como la del otro es la práctica social de la humanidad*". Esto significa que un individuo no solo aprenderá de forma práctica sino, además, deberá perfeccionar el conocimiento obtenido leyendo o asistiendo a cátedra. A esto se le conoce como construcción del conocimiento.

1.1. Características

- La principal característica es la construcción del conocimiento, el cual crea una base firme en la educación del alumno respecto al tema estudiado.

- El entorno de aprendizaje constructivista debe ser secuencial, para ir aumentando el conocimiento paulatinamente.

- El constructivismo apoya el refuerzo social, lo cual permite forjar mejor el conocimiento.

- El estudiante debe querer aprender y no debe ser solamente por aprobar una materia.

- El maestro es un facilitador del aprendizaje, pero el alumno debe de construir su propio conocimiento.

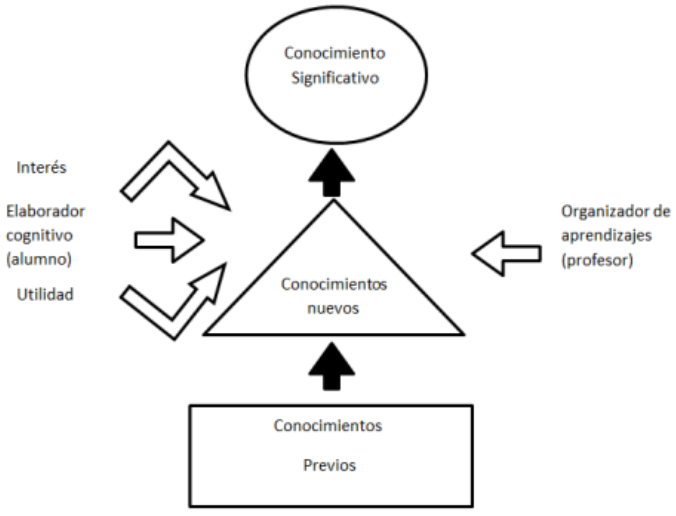
1.2. Condiciones

Existen diferentes condiciones para que el aprendizaje pueda ser exitoso, algunas se presentan a continuación.

1.2.1. Aprendizaje significativo

El profesor debe ser un medio y un participante dentro del aula; no debe ser considerado un déspota que siempre estará en lo correcto. Además, los estudiantes deben tener la motivación propia de aprender. Está comprobado que los alumnos aprenden con mayor facilidad aquello que les interesa.

Figura 1. **Aprendizaje significativo**



Fuente: CALERO PÉREZ, Mavilo. *Constructivismo pedagógico. Teorías y aplicaciones*. p. 124.

1.2.2. Aprendizaje por descubrimiento

Este tipo de aprendizaje lleva al estudiante a descubrir por sí mismo los conceptos y habilidades necesarias, desarrollando así su razonamiento inductivo. El profesor no debe estar ausente en este tipo de aprendizaje, si bien no explica cómo resolver un problema, debe guiar al alumno a descubrir por sí mismo la solución a un problema planteado.

1.2.3. Aprendizaje centrado en la persona

Este tipo de aprendizaje no está centrado en el catedrático, sino en el estudiante, quien debe ser capaz de concentrarse en su propio aprendizaje y crear las condiciones adecuadas para que el proceso sea más sencillo. El objetivo principal de este aprendizaje es que los alumnos tomen la iniciativa de aprender, pero también de colaborar con el aprendizaje de sus semejantes.

1.2.4. Aprendizaje receptivo

El aprendizaje receptivo es el más común, tanto en la educación primaria como en la educación superior. El estudiante obtiene los conocimientos de forma pasiva; esto significa que el catedrático proporciona la información, ya sea por medio del material impreso o audiovisual, y el estudiante la capta y puede aplicar.

1.2.5. Aprendizaje cooperativo

El progreso que se obtiene mediante la interrelación de un alumno con el resto del grupo es esencial para aprender con mayor facilidad. Existen diferentes estudios en los cuales se ha comprobado que la interacción con otros

individuos ayuda a comprender mejores temas que van desde ortografía hasta matemáticas.

1.2.6. Inteligencias múltiples

En 1983 se planteó la teoría de las inteligencias múltiples por el psicólogo Howard Gardner. Esta teoría propone que para cada problema que se debe resolver, existe un tipo diferente de inteligencia capaz de hacerlo. Según Gardner existen 8 tipos de inteligencias:

- Musical: es la capacidad de expresar sentimientos o ideas a través de la música. Las personas con esta inteligencia pueden componer piezas musicales, leer partituras, cantar o tocar un instrumento con mayor facilidad.

- Cinestésica: es la habilidad de controlar el cuerpo de forma segmentada o en su totalidad para poder realizar actividades. Deportistas, bailarines u obreros son representantes de esta habilidad.

- Interpersonal: es la habilidad de “ponerse en los zapatos de los demás”, la capacidad de empatizar con los demás. Son personas capaces de trabajar con grupos numerosos.

- Verbal: es la capacidad de comunicarse con los demás, no necesariamente de forma oral, sino también por medio de gestos. Es poder comprender y transmitir de forma adecuada, sin que esto represente un reto.

- Lógico-matemática: a lo largo de los años, esta habilidad ha sido considerada la más importante dentro del aula. Si una persona demostraba ser buena en matemática, era considerada inteligente; si, por el contrario, resultaba ser una persona hábil en deportes, pero mala en matemáticas, este era considerado “tonto”. En la actualidad aún se considera una de las inteligencias más importantes, mas no la única.

- Naturalista: las personas con este tipo de inteligencia disfrutan estar en contacto con la naturaleza, les interesan las plantas, los animales, el clima y la protección del medio ambiente.

- Intrapersonal: la inteligencia intrapersonal es la capacidad de un ser humano de conocerse a sí mismo, pueden controlar sus emociones y entender el porqué de una reacción.

- Espacial: es la habilidad de abstracción, la cual permite tener más de un punto de vista plano; es poder formar imágenes mentales en dos o tres dimensiones.

1.3. Tipos de saberes

Existen tres tipos de saberes los cuales se diferencian entre sí por los medios de aprendizaje.

1.3.1. Saber conceptual

También conocido como el *saber,* es la teoría o concepto, no necesariamente literal, de un tema. El alumno es capaz de relacionar los nuevos conceptos, con conceptos previamente adquiridos.

1.3.2. Saber procedimental

El saber procedimental o *saber hacer,* hace referencia a la metodología utilizada para realizar una acción o trabajo. Este se desarrolla por medio de la práctica. Se orienta a la realización de procesos exitosos que generen conocimiento.

1.3.3. Saber actitudinal

Hace referencia al *saber ser*, está basado en los valores y actitudes que una persona debe poseer. Estos pueden ser desarrollados por imitación y prácticas relacionadas al lugar donde se desenvuelve el individuo.

1.4. TICs

En la actualidad las TICs han tenido un auge significativo gracias a las nuevas modalidades de enseñanza.

1.4.1. ¿Qué es una TIC?

Las tecnologías de la información y la comunicación (TIC) son algunos de los recursos actuales para obtener y transmitir información. Estas son muy utilizadas por colegios y centros de enseñanza superior para mejorar la educación.

1.4.2. Herramientas del aprendizaje

El internet tiene un gran impacto en la educación, no solo a nivel primario y secundario, sino especialmente a nivel superior. Un estudiante de nivel

primario o secundario nunca ha asistido a una biblioteca en búsqueda de información, puesto que todo puede ser encontrado en internet. Para ello puede utilizar diferentes recursos.

1.4.2.1. Redes sociales

Las redes sociales son una herramienta del siglo XXI. Se ha caracterizado por promover la interconexión entre usuarios, permitiéndoles compartir fotos, videos, chistes o mensajes de forma pública o privada. El docente puede utilizar estos sitios de internet como un recurso didáctico, puesto que los alumnos ya participan activamente en estos.

Dentro de las redes sociales más comunes, se puede mencionar Facebook y Twitter, las cuales permiten crear una cuenta personal, pero también páginas y/o grupos dedicados a un tema en específico.

1.4.2.2. Wikis

Una wiki sirve como una fuente, no siempre fidedigna, de información que ha sido editada o creada por un usuario externo. Provee la facilidad de trabajar con contenido hipertextual, para que cada estudiante pueda realizar sus aportes propios, y puede ser ampliado o corregido por otros estudiantes.

1.4.2.3. Blogs

Es un sitio web que permite al autor publicar distintos artículos de forma sencilla. Existen páginas dedicadas a la publicación de blogs gratuitos, los cuales pueden ser personalizados y orientados a cualquier tema. En este tipo de web, el autor del blog publica gradualmente. La entrada más reciente es la

primera en aparecer en la web. Los estudiantes podrán hacer comentarios o preguntas en cada entrada.

1.4.2.4. YouTube

YouTube es un sitio web 2.0, en el cual cualquier persona tiene la posibilidad de compartir contenido como videos, películas o clips caseros, los cuales pueden ser comentados y calificados desde cualquier parte del mundo. Este sitio controla todo el contenido que es subido a él, por lo cual videos que incumplan los derechos de autor serán removidos de la plataforma.

1.4.3. Características

Algunas de las características aplicables a las TIC son:

1.4.3.1. Inmaterialidad

La inmaterialidad es una característica importante dentro de las TIC, puesto que no tiene que estar ligada a un medio físico. Mediante otra característica, la digitalización, permite acceder desde cualquier medio digital.

1.4.3.2. Interactividad

Permite el intercambio de información entre el usuario y una plataforma, se ajusta a las necesidades y tiempo del alumno, puesto que le permite avanzar a un paso en el que pueda adaptarse, a cualquier horario y momento del día.

1.4.3.3. Interconexión

Una parte importante de las TIC es su capacidad de unir dos tecnologías distintas. Un ejemplo es YouTube, puesto que es una forma de comunicación digital por medio de videos, que utiliza internet para transmitir y un dispositivo digital para que pueda ser reproducido.

1.4.3.4. Instantaneidad

En el siglo pasado, solamente los privilegiados tenían la posibilidad de tener un celular; las comunicaciones eran bastante lentas. Una noticia trágica podía tardar hasta un mes en ser comunicada debido al tiempo en el que se movilizaba un mensajero. Por el contrario, en la actualidad se tiene la facilidad de que la comunicación es instantánea; una llamada o un mensaje de texto llega de forma instantánea al receptor.

1.4.3.5. Digitalización

Es la capacidad de que la información sea universal. Puede estar observando la misma diapositiva en lugares que se encuentran a kilómetros de distancia, gracias a las interconexiones de red que enlazan a todo el mundo mediante la codificación de la información.

1.4.3.6. Innovación

Los seres humanos siempre están evolucionando, por ende, creando nuevos aparatos tecnológicos para hacer más sencillo el día a día. El acceso a la información es uno de los saltos más grandes que tuvo el siglo XXI. Antes se debía ir a bibliotecas o hemerotecas para encontrar información. En la

actualidad es suficiente conectarse a internet para encontrar toda la información que se necesite.

1.4.3.7. Penetración a todos los sectores

Una de las mayores dificultades en la creación de recursos virtuales es que no todas las personas tienen la posibilidad de acceder a un computador o celular inteligente con conexión a internet. Por ende, no se puede migrar a una enseñanza virtual completa y siguen siendo necesarias las clases presenciales.

1.4.3.8. Diversidad

Se refiere al gran campo que poseen las TIC, desde proveer información, servir como un medio de comunicación o una forma de distracción, puesto que no se encuentra únicamente relacionada con la educación.

1.4.4. Tipos de comunicación de las TIC

Existen dos tipos de comunicaciones los cuales dependen de las condiciones del emisor y receptor al momento de impartir la clase.

1.4.4.1. Asíncrona

La comunicación asíncrona sucede cuando el emisor y el receptor no mantienen una comunicación en tiempo real sino en lapsos indeterminados. Un ejemplo de esta comunicación son las cartas e incluso las redes sociales.

1.4.4.2. Síncrona

Este tipo de comunicación no necesariamente presencial requiere que el catedrático y el alumno se encuentren en el mismo momento mientras se realiza el intercambio de conocimiento. Es una forma de crear un vínculo que, si bien no es físico, sí genera cierta socialización entre emisor y receptor, permite el intercambio de ideas, consultas y comentarios.

2. MICROCONTROLADOR TM4C123GH6PM

2.1. Sistemas embebidos

Los sistemas embebidos son parte fundamental de muchos de los sistemas electrónicos utilizados en la vida común.

2.1.1. ¿Qué son los sistemas embebidos?

La definición de sistema es un conjunto de reglas que rigen el correcto funcionamiento de un proceso, un grupo o un trabajo. Un sistema embebido es aquel que, dentro de una misma tarjeta, en este caso un microcontrolador, puede cumplir con un objetivo específico, a diferencia de una computadora que puede realizar muchas más funciones.

2.1.2. Características

Dentro de las características principales de un sistema embebido están:

- El tamaño: su tamaño reducido es un factor importante, puesto que entre más grande sea, el costo de producción aumentará. Además, la capacidad de instalarse en un lugar con espacio reducido permite un mayor número de aplicaciones.

- El costo: debe ser reducido, puesto que sus características y tamaño serán específicos para la función que realizará.

- La funcionalidad: existen diferentes tipos de sistemas embebidos, los cuales contarán con diferentes tipos de periféricos, tiempos de respuesta y espacios en memoria. Esto deberá ser considerado al momento de elegir o diseñar un sistema embebido.

2.1.3. Microcontrolador

Se dará una breve introducción sobre el funcionamiento de un microcontrolador, parte esencial de las prácticas a realizar.

2.1.4. ¿Qué es un microcontrolador?

A diferencia de un CPU, el cual está dedicado al computador personal o de industria, un microcontrolador puede definirse como una microcomputadora integrada todo en un CI. Están diseñados para cumplir una tarea en específico, puesto que una vez haya sido programado, efectuará únicamente lo indicado. Esto no implica que no pueda ser programado nuevamente, pero para esto se deberá volver a conectar a un computador.

2.1.5. Partes de un microcontrolador

Estará compuesto por una unidad central de procesamiento, memoria, periféricos de entrada y salida.

2.1.5.1. Microprocesador

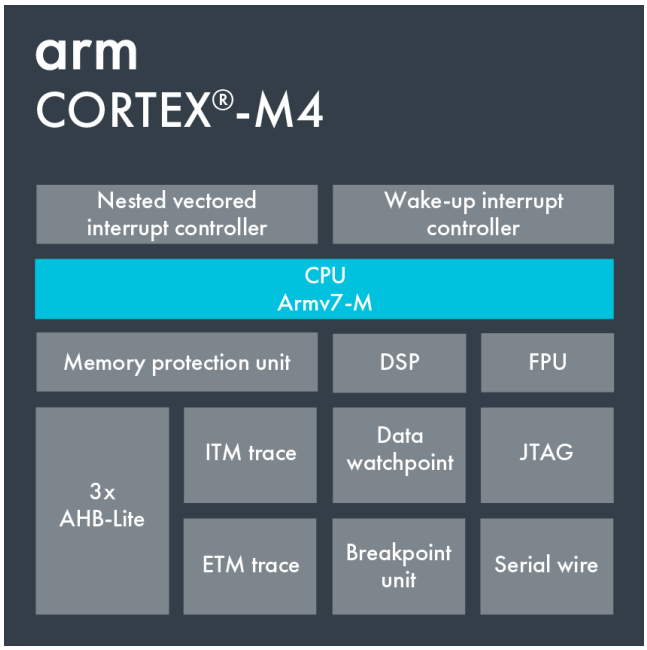
Es un chip con integración a gran escala o VLSI, lo cual implica que estará integrado por miles de transistores que permiten realizar funciones aritméticas, lógicas y de comunicación. Usualmente es confundido con un microcontrolador,

puesto que un microprocesador en una misma placa con interfaces y memorias se convierte en un microcontrolador.

La unidad de procesamiento central o CPU es donde se ejecutarán las operaciones de control del microprocesador. Existen dos tipos de set de instrucciones para un procesador: arquitectura RISC o Reduce Instruction Set Computing, el cual está basado en microcontroladores fáciles de construir y rápidos en el procesamiento; y arquitectura CISC o Complex Instruction Set Computing, el cual es capaz de ejecutar instrucciones complejas, pero consumiendo mayor energía.

Además, cuenta con la unidad aritmética y lógica o ALU, donde se ejecutan las funciones matemáticas en base binaria. La comunicación de un microprocesador con el exterior se realiza a través del bus, el cual es un conjunto de líneas que funcionan como un sistema de comunicación entre el microprocesador y los componentes del microcontrolador. Para cada componente existe un controlador diferente que permite el envío y recepción de datos. La memoria caché es la parte del CPU en la cual se almacenan instrucciones que comúnmente son necesarias para realizar una función; generalmente es de tamaño pequeño pero capaz de realizar tareas a una alta velocidad. Los registros son un espacio de memoria reducido en la cual se almacenan los datos obtenidos al terminar un proceso. Con estos se determina la cantidad de bits que puede procesar.

Figura 2. **Partes de un microprocesador Cortex-M4**



Fuente: arm Developer. *Cortex M-4*. https://developer.arm.com/ip-products/processors/cortex-m/cortex-m4. Consulta: 9 de diciembre de 2019.

2.1.5.2. Memorias

Una de las características principales de un microcontrolador es el tamaño reducido, por lo cual las memorias suelen estar dentro del mismo integrado. No se encontrarán de un gran tamaño debido al espacio reducido.

2.1.5.2.1. RAM

Memoria de acceso aleatorio (RAM) es el lugar donde se almacena la información volátil, puesto que al apagar o desconectar el procesador esta se perderá. Existen dos tipos de RAM: estática o SRAM, la cual mantendrá la información hasta el momento en el que se le retire la potencia, y la dinámica o

DRAM, la cual se sobrescribe constantemente para que no se pierda la información contenida en ella.

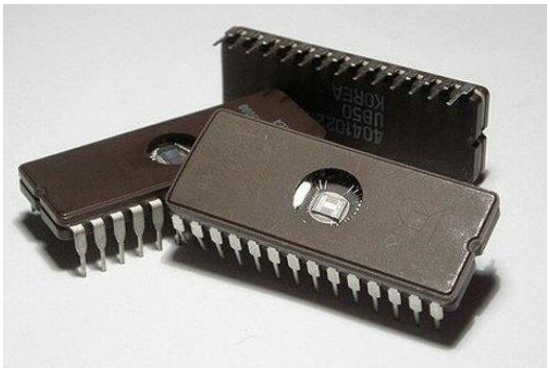
2.1.5.2.2. ROM

La memoria de solo lectura o ROM es utilizada para almacenar datos de forma permanente; el procesador puede leer información mas no escribir información en ella. Al no necesitar una fuente de poder, esta memoria es conocida como memoria no volátil.

2.1.5.2.3. EPROM

ROM borrable-programable o EPROM son memorias que pueden ser reprogramadas si previamente los datos almacenados son borrados con luz ultravioleta, la cual es aplicada por medio de una ventana de cuarzo que se encuentra ubicada sobre el microcontrolador.

Figura 3. **Memoria EPROM**



Fuente: Indiamart. *EPROM Memory Integrated Circuit*. https://www.indiamart.com/proddetail/eprom-memory-integrated-circuit-17156156712.html. Consulta: 6 de marzo de 2020.

2.1.5.3. Puertos

Los puertos I/O digitales tienen la función de enviar y recibir información desde y hacia el microcontrolador. Estos proporcionan interfaces a dispositivos como actuadores y sensores. Cada puerto puede tener más de una función y son programables según los requerimientos del programador.

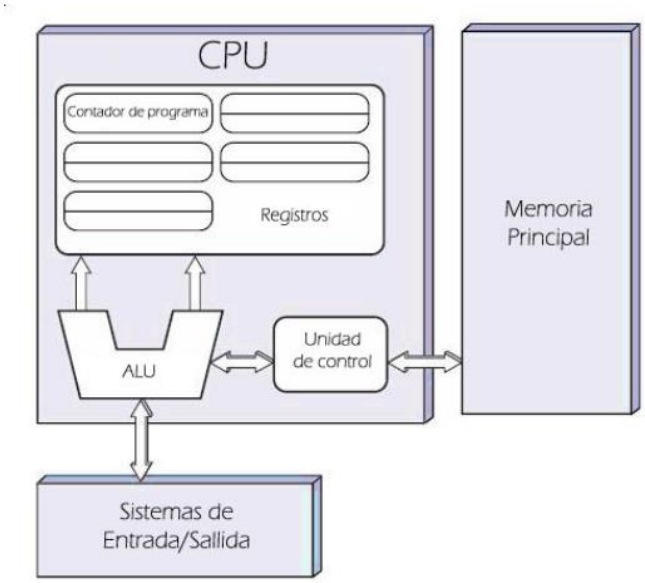
2.1.6. Arquitectura de un microcontrolador

Existen dos arquitecturas que se diferencian entre sí, dependiendo de la forma en que el microcontrolador acata o ejecuta instrucciones.

2.1.6.1. Von Neumann

La característica principal de este tipo de arquitectura es que no cuenta con espacio separado para las instrucciones y los datos, por lo cual utiliza el mismo bus de datos para comunicarse con el CPU. A pesar de que podría tener un procesamiento menor al de Harvard, es el más utilizado debido a la implementación más sencilla.

Figura 4. **Arquitectura Von Neumann**

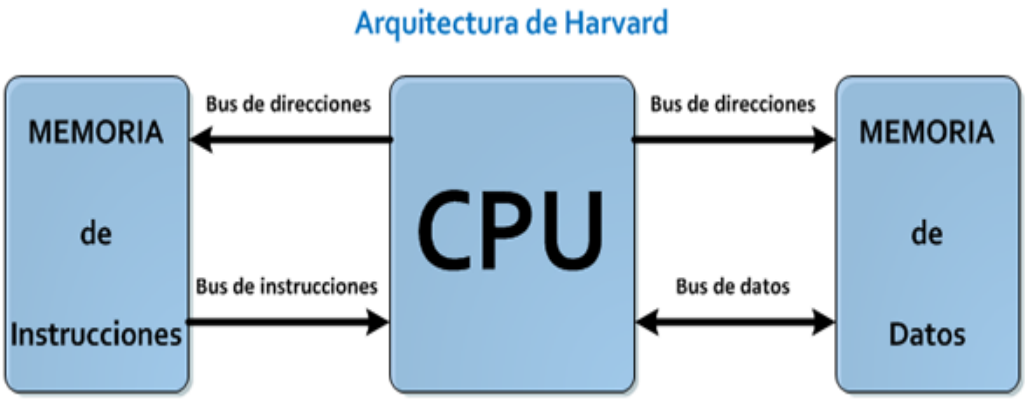


Fuente: Tools. *Diferencias entre los modelos de Von Neumann y Harvard*. https://www.electrontools.com/Home/WP/2018/04/15/diferencias-entre-los-modelos-de-von-neumann-y-harvard/. Consulta: 6 de marzo de 2020.

2.1.6.2. Harvard

Se diferencia principalmente de la arquitectura Von Neumann debido a la separación de instrucciones y datos en dos memorias separadas, por lo cual se utilizan dos buses diferentes de información. No es tan eficiente al utilizar las memorias, puesto que los relojes deben estar sincronizados. Generalmente es utilizado en el procesamiento de audio y video.

Figura 5. **Arquitectura Harvard**



Fuente: Tools. *Diferencias entre los modelos de Von Neumann y Harvard*. https://www.electrontools.com/Home/WP/2018/04/15/diferencias-entre-los-modelos-de-von-neumann-y-harvard/. Consulta: 6 de marzo de 2020.

2.2. ARM

Todas las computadoras, teléfonos, tabletas utilizan procesadores para funcionar, en su gran mayoría procesadores ARM.

2.2.1. Historia

En 1983, la empresa Acorn Computers Ltd. inició el proyecto ARM. Dos años después, en 1985, el primer prototipo fue desarrollado y tiene el nombre de ARM1. No fue sino hasta el año siguiente en que ARM2 fue lanzado como el primer procesador comercial. Este posee un bus de datos de 32 bits, el cual contenía 30,000 transistores y no contaba con memoria caché, lo cual lo volvió de bajo recursos y de muy rápido procesamiento.

En 1990 se crea la compañía Advanced RISC Machines Ltd. compuesta por tres compañías, Apple Computers, Acorn Computers y VLSI Technologies. Al año siguiente se presentó el procesador ARM6, utilizado especialmente para productos Apple.

Uno de los mayores éxitos de ARM fue la posibilidad de que, tras adquirir una licencia, el usuario podía adecuar el microprocesador a los requerimientos de cada uno. Esto permitió que compañías como IBM, Nintendo, Texas Instruments y Samsung, entre otros, utilizaran hasta la actualidad estos microprocesadores.

2.2.2. Familias de procesadores

Los procesadores ARM se subdividieron en tres diferentes familias, las cuales se clasifican por su funcionalidad, costo y consumo.

2.2.2.1. Córtex-A

Esta familia de procesadores está basada en la aplicación y utilización para terceros. Los fabricantes de diferentes productos compran una licencia de esta familia para adecuarlos al usuario final. Generalmente son utilizados en smartphones, videojuegos y tabletas.

2.2.2.2. Córtex-R

La característica principal de esta familia es la velocidad de procesamiento, puesto que están destinados a aplicaciones en tiempo real, donde se necesita una rápida respuesta. Son utilizados en el campo de equipo médico, aviación y robótica.

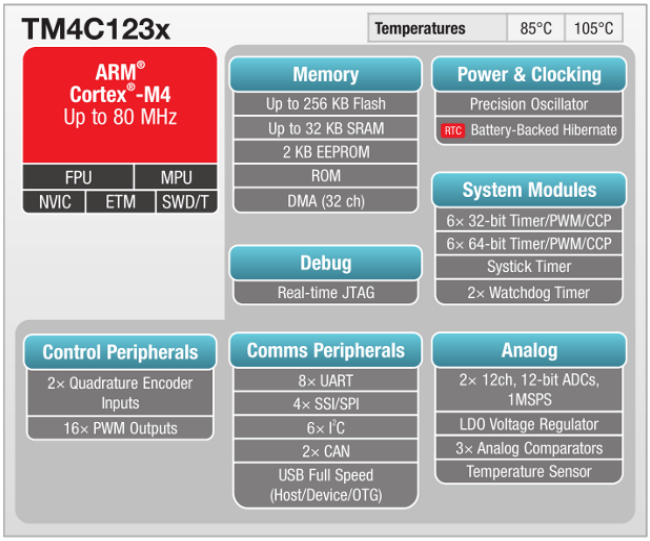
2.2.2.3. Córtex-M

Orientada a sistemas embebidos como los microprocesadores, puesto que tienen un bajo consumo de potencia y bajo costo. Son utilizados por Texas Instrument y Arduino.

2.3. Microcontrolador TM4C123GH6PM

Es un microcontrolador fabricado por Texas Instruments generalmente utilizado en el área educativa. Pertenece a la familia Córtex-M, específicamente Córtex-M4F, el cual posee una arquitectura RISC de 32 y 64 bits.

Figura 6. **Características del microcontrolador TM4C123GH6PM**



Fuente: NBC. *Conociendo el Microcontrolador TM4C123GH6PM, un ARM Orientado para Automatización, de Texas Instruments*. https://www.incb.com.mx/index.php/articulos/78-microcontroladores-y-dsps/1812-conociendo-el-microcontrolador-tm4c123gh6pm-un-arm-orientado-para-automatizacion-de-texas-instruments-mic013s. Consulta: 2 de diciembre de 2019.

2.3.1. Temporizador

El temporizador del sistema o SysTick es parte de todos los procesadores de la familia Cortex-M. Consiste en un contador en decremento de 24 bits que produce una interrupción cuando el contador interno llega a cero. Puede configurarse el valor de recarga, mas no escalas ni parámetros, por lo que su utilización es bastante básica.

2.3.2. MPU

MPU o *Memory Protection Unit* es un sistema de protección especial para ARM7. Protege regiones de memorias, traslapes entre ellas y permiso de accesos.

2.3.3. FPU

Por las iniciales de *Float Point Unit*, es conocido comúnmente por ser un coprocesador matemático encargado de las operaciones con punto flotante, lo cual permite realizar operaciones sin pérdida de decimales. Utiliza el estándar IEEE-754 para mostrar sus valores.

2.3.4. NVIC

NVIC o *Nested Vector Interrupt Controller* provee una mayor facilidad en el control de interrupciones. Puede manejar 240 interrupciones con 256 niveles de prioridad distintos, que pueden ser modificables de forma dinámica.

2.3.5. WatchDog

Parte del sistema de un microcontrolador. Es el encargado, si hubiera una falla, de reaccionar en forma de temporizador, el cual al llegar al valor de cero reiniciará el sistema. Existen dos *watchdog timers*, el primero, WatchDog Timer0 utiliza el reloj del sistema y el segundo, WatchDog Timer1 utiliza el oscilador principal.

2.3.6. JTAG

JTAG (*Joint Test Action Group*) previamente conocido como norma IEEE 1149.1, es utilizado en los puertos de comunicación que permite el acceso de pruebas por medio de una interfaz, para tener un mejor control del código y conocer la funcionalidad exacta de este.

3. ENSAMBLADOR

El lenguaje ensamblador es un lenguaje de programación de bajo nivel con el cual se puede programar computadores, microcontroladores y pics, entre otros. Es la representación más directa al código de máquina, está basado en mnemónicos que simbolizan las instrucciones, los registros del procesador y posiciones en memoria.

Este lenguaje ayudó a los programadores a simplificar su trabajo, puesto que era más natural para el hombre su comprensión en vez de códigos binarios que debía memorizar.

El código en ensamblador es denominado comúnmente código fuente o asm; a partir de esto genera el código de máquina o código objeto. Es un archivo con extensión hex.

3.1. Historia

En los años 50 se comenzaron a desarrollar los primeros lenguajes ensambladores debido a la dificultad por parte de los programadores de recordar instrucciones binarias para cada una de las funciones, por lo que se relacionan los OPCODE con palabras clave mucho más sencillas. De esto se derivan los mnemónicos.

UNIVAC (Universal Automatic Computer) fue el primer computador en utilizar el lenguaje ensamblador.

Para los años 80 el uso del lenguaje ensamblador había sido reducido por la llegada de lenguajes de alto nivel, además de la creación de procesadores de 32 y 64 bits, los cuales ya no estaban regidos por el tamaño del código.

3.2. Instrucciones de CPU

Las instrucciones son generalmente las mismas en los diferentes CPU que existen, pero estas pueden variar dependiendo del CPU. De acuerdo con su función, pueden dividirse de la siguiente forma:

3.2.1. Operaciones con números enteros

Las operaciones con números enteros son realizadas por la ALU. dentro de ellas se encuentran operaciones aritméticas, booleanas y comparaciones.

Las operaciones aritméticas básicas son suma, resta, multiplicación o división. Las operaciones booleanas o lógicas serían operaciones bit a bit, mientras las comparaciones son operaciones donde se comparan dos enteros o registros.

3.2.2. Operaciones con números reales

Si el microprocesador cuenta con el coprocesador para punto flotante, es capaz de realizar operaciones aritméticas como suma, resta, división, multiplicación; operaciones trigonométricas como seno, coseno; operaciones logarítmicas, potencias y raíces.

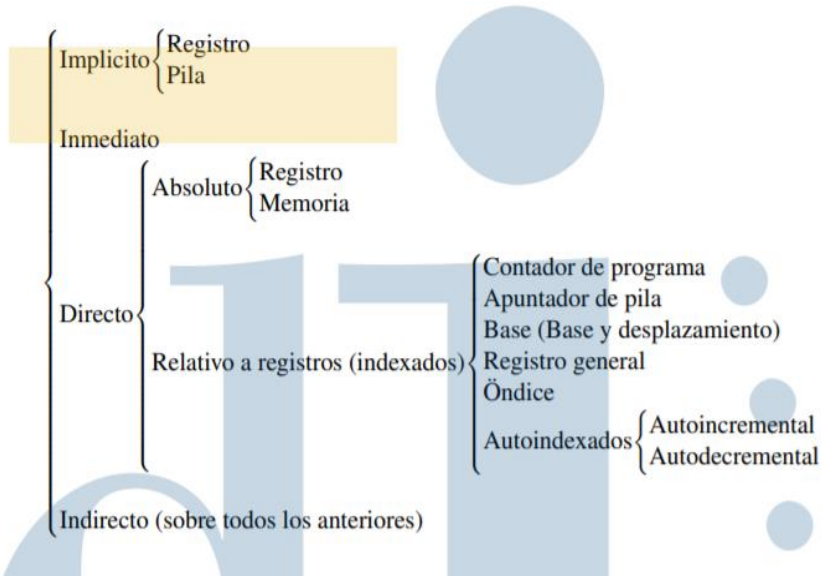
3.2.3. Operaciones de movimiento de datos

Existen diferentes formas de movimiento de datos en ensamblador. Dentro de estas formas es entre registros y memoria, operaciones de pila, para extraer y almacenar datos,

3.3. Modos de direccionamiento

Los modos de direccionamiento indican la forma en que la información se interpretará para especificar un operando en una instrucción. Existe una gran cantidad de modos de direccionamiento que pueden ser complicados si no se comprende cómo operarlos, pero puede simplificar la programación si se sabe cómo utilizarlos.

Figura 7. **Modos de direccionamiento**



Fuente: Departamento de informática, Universidad de Valladolid. *Modos de Direccionamiento.* https://www.infor.uva.es/~bastida/OC/modos.pdf. Consulta 12 de diciembre de 2019.

Existen cuatro modos de direccionamiento de los que se desglosan los demás.

3.3.1. Directo

Dentro de este se encuentra el modo directo absoluto, que puede dividirse en directo por registro —el cual especifica la dirección de un registro del procesador—, directo por memoria —que en la misma instrucción indica la dirección en memoria donde se encuentra el operando—, y el directo indexado o relativo a registros: para este tipo existe el registro índice, el cual es sumado al campo del operando.

3.3.2. Indexado

Dentro del modo indexado se puede encontrar diferentes tipos de direccionamiento, entre ellos el contador de programa, el cual indica la diferencia entre la dirección del dato y la dirección siguiente a la instrucción que se realiza en ese preciso momento. Si el offset es positivo, el operando estará en la dirección siguiente a la dirección; si es negativo, será una anterior a la instrucción. El direccionamiento por base y desplazamiento comúnmente es utilizado en ordenadores que puedan mantener en memoria varios programas. El registro base puede ser modificado para localizar en un nuevo punto del programa sin modificar ningún operando; los direccionamientos autoincremental y autodecremental sirven para aumentar o disminuir la dirección del operando una vez este haya sido operado. Comúnmente es utilizado en el manejo de vectores.

3.3.3. Indirecto

Está basado en cada uno de los modos de direccionamiento e indica la localización de la dirección.

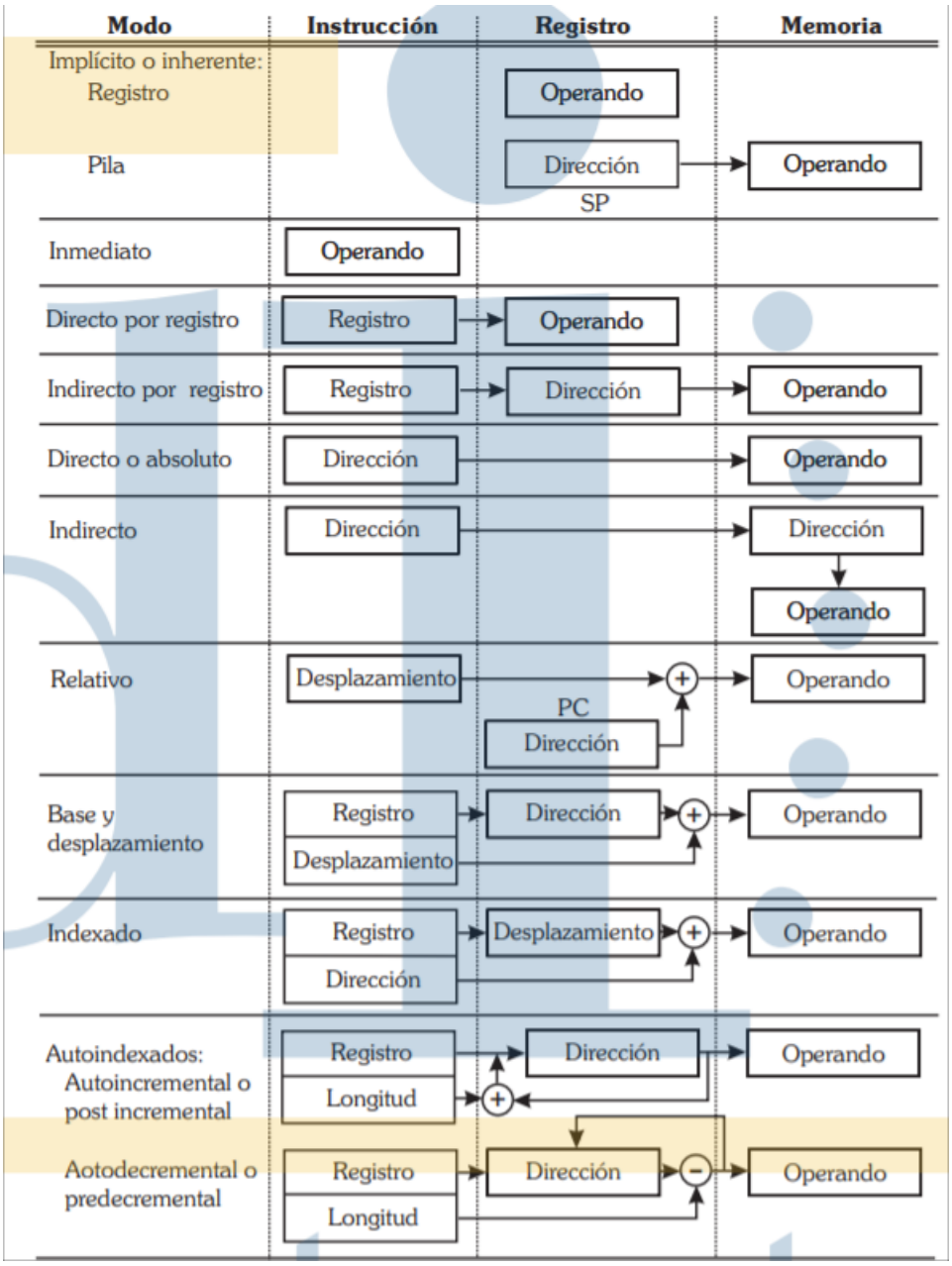
3.3.4. Inmediato

Es el que se asigna en la misma instrucción en vez de tener alguna dirección. Asigna el operando y es utilizado para asignar valores de constantes o al reiniciar una variable.

3.3.5. Implícito

También llamado inherente, el operando se especifica en la misma instrucción, por lo cual no necesita parámetros. Existen dos tipos de direccionamiento implícito: por registros y por operandos de pila. El primero especifica un registro en la misma instrucción, mientras el segundo utiliza el inicio de la pila de forma implícita.

Figura 8. **Funcionamiento de los modos de direccionamiento**



Fuente: Departamento de informática, Universidad de Valladolid. *Modos de Direccionamiento.* https://www.infor.uva.es/~bastida/OC/modos.pdf. Consulta 12 de diciembre de 2019.

3.4. Registros

Los registros son pequeñas memorias a las cuales se puede acceder de forma sumamente veloz. Pueden ser de 4 a 64 bits. Existen los siguientes registros:

3.4.1. Registros de propósito general

También conocido como registros de datos, son utilizados para guardar números enteros. Anteriormente únicamente se utilizaba el registro acumulador.

Dentro de la arquitectura de ARM cuenta con 31 registros, cada uno de 32 bits. De estos, únicamente 16 son visibles para el usuario, y 13 son utilizables, puesto que los otros 3 son registros dedicados. La nomenclatura de los registros será de la forma Rn, siendo n el número de registro específico, R0-R15.

Los tres registros dedicados son:

- R13, Puntero de pila, Stack Pointer o SP, su función es mantener la dirección de la palabra que es procesada en la pila.
- R14, registro de enlace, Link Register o LR, guarda la dirección de la siguiente instrucción a un salto realizado.
- R15, contador de programa, Program Counter o PC, es un registro del procesador que especifica la ubicación de la instrucción ejecutada en dicho momento.

3.4.2. Registros de memorias

Es un pequeño espacio de memoria sumamente veloz, en el cual se almacena información que se utiliza frecuentemente, lo que agiliza la ejecución de instrucciones.

3.4.3. Registros de punto flotante

La FPU no utiliza los registros de propósito general, sino que sus propios registros, llamados pila de registros. Su nomenclatura es Sn, siendo n el número de registro de punto flotante yendo de S0 a S31.

3.4.4. Registros constantes

Los registros constantes son valores de solo lectura definidos por el hardware.

3.4.5. Registros de propósito específico

Guardan información del estado de ejecución del programa.

3.5. Elementos básicos

El lenguaje ensamblador consta de algunos elementos básicos como:

3.5.1. Símbolo

Dentro de los símbolos existen deferentes campos. Uno de ellos es la etiqueta, el cual es el valor que tome la instrucción; el código de operación es la

instrucción en sí, operando los cuales serán los valores a modificar dependiendo del código de operación.

3.5.1.1. Comentario

No es parte del código sino una explicación sencilla de lo que se realiza en una línea en específico o en todo el código, para ayudar a comprender el código.

3.5.2. Mnemónicos

Son un conjunto de símbolos que representarán instrucciones en el código. Fueron ideados para facilitar la programación y evitar recurrir al código de máquina.

3.5.3. Secciones de datos

También conocido como pseudo-instrucción, no se traducirán a código de máquina, pero ayudan en el proceso de ensamblado; indican el tipo, tamaño y alineación de los datos.

3.5.4. Directivas de ensamblador

Una directiva no es parte del código plano sino una orden directa del compilador, el cual le indica al procesador una secuencia de pasos que se debe seguir previo a la ejecución del código. Las directivas no se traducen al código de máquina, solo funcionan durante el ensamblaje.

3.6. ¿Qué es un IDE?

IDE o Entorno de Desarrollo Integrado es un editor de código que cuenta con compilador, depurador y constructor para una interfaz gráfica, que facilitando el desarrollo de un programa. Es común que un mismo IDE funcione con varios lenguajes de programación y no específicamente con uno. Algunos de los IDE más comunes son Eclipse, NetBeans, Oracle, Visual Studio, entre otros.

3.6.1. Características

Aun teniendo distintas funciones, un IDE debe de cumplir con las siguientes características:

- Multiplataforma
- Soporte para diversos lenguajes
- Extensiones y componentes para el IDE
- Depurador
- Múltiples idiomas
- Manual de usuario y ayuda

3.6.2. Componentes

Un IDE debe cumplir con los siguientes componentes

3.6.2.1. Editor de texto

También llamado editor de código fuente, es la parte del editor donde se escribe el código del programa. Algunas de sus funciones son generar el texto

plano que posteriormente se compilará. En él se puede editar el texto, buscar y reemplazar código; además, dependiendo del lenguaje utilizado, resaltará comandos o signos de puntuación. Dentro de los editores más comunes están Notepad++, Vi, Unix entre otros.

3.6.2.2. Compilador

Un compilador es el que después de que un programador realiza el texto plano, traduce o compila a lenguaje de máquina para que pueda ser entendido y procesado por un equipo. Se compone de dos partes: el FrontEnd, que interactúa con el usuario, y el BackEnd, encargado de generar el código del trabajo realizado en el FronEnd.

3.6.2.3. Intérprete

Comúnmente confundido con el compilador, es mucho más rápido pues no traduce el código sino lo ejecuta directamente. Dentro de un IDE es el que permite la opción de *debugging*.

3.6.2.4. Depurador

Depurador o *debugger* es donde se encuentra con mayor facilidad los errores para corregirlos. Este ejecuta el programa principal para examinar paso a paso la secuencia del programa y encontrar el fallo.

3.7. IDE Keil uVision

A continuación, se dará una breve introducción del programa a utilizar.

3.7.1. ¿Qué es?

En un mismo ambiente, el entorno de desarrollo de Keil incluye el editor de texto, compilador, ensamblador y depurador aplicado a varios microcontroladores de ARM.

4. INTRODUCCIÓN A LA PROGRAMACIÓN EN ENSAMBLADOR PARA EL MICROCONTROLADOR TM4C123GH6PM

Teniendo un mayor conocimiento sobre el lenguaje y el microcontrolador que se utilizará, se dará una breve introducción a los elementos que se utilizarán.

4.1. Directivas

Algunas de las directivas más utilizadas son:

4.1.1. ALIGN

La directiva ALIGN permite alinear en una dirección específica el contador de posición. Generalmente se coloca al inicio del segmento de programación. La alineación se realiza insertando ceros. El número que se debe colocar es cualquiera que resulte de elevar un 2 a cualquier número de 0 a 31 más el valor de offset que puede ser cualquier número.

Se escribe de la siguiente manera:

ALIGN {expresión {offset}}

Además de alinear al inicio del código, debe definirse donde terminará. Por lo que se escribirá de la siguiente forma:

ALIGN = 2

….

Código

…

ALIGN

4.1.2. AREA

La directiva área indica que se debe ensamblar un código nuevo. El nombre del área puede ser cualquier nombre elegido por el usuario. Por convención se utiliza |.text| puesto que permite al programador llamar el código desde módulos escritos en C.

La directiva AREA tiene distintos atributos, entre ellos están:

- READONLY: indica que no se debe escribir en la sección. Área de código.
- READWRITE: puede ser leída y escrita, usada para áreas de datos.
- DATA: el área será de datos.
- ALIGN: alinea la sección.

La forma de escribir esta directiva es:

AREA nombre, atributo, atributo

Por ejemplo:

- AREA |.text|, CODE, READONLY, ALIGN=2

- AREA HEAP, NOINIT, READWRITE, ALIGN=3
- AREA |2datos|, CODE, READONLY, ALIGN=4

4.1.3. DCB

La directiva DCB asigna un byte (8 bits) o más de memoria. La sintaxis es:

Nombre DCB Expresión, Expresión

Puede utilizarse para guardar cadena de caracteres, enteros entre -128 y 255, binarios, ASCII, Octales, entre otros, por ejemplo:

- Enteros DCB 3,-100, 50, 10
- Bin DCB 2_01011111

4.1.4. DCD

La directiva DCD define una constante de tipo “Word” o palabra para 32 bits de datos. Puede ser una expresión numérica o una expresión de PC.

La sintaxis de esta directiva es la siguiente:

Etiqueta DCD Expresión, Expresión2… ExpresiónN

Por ejemplo:

- Vector1 DCD 100,43,1,5
- PC DCD mem03 +8

4.1.5. DCFD

La directiva DCFD asigna valores de memoria a puntos flotantes, los que deben estar alineados para que puedan ser utilizados en operaciones aritméticas.

Los rangos para los valores que se pueden asignar son:

- Máximo 1.79769313486231571e+308
- Mínimo 2.22507385850720138e-308.

La sintaxis es:

Etiqueta DCFD ValorPF, ValorPF2,… ValorPFn

Por ejemplo:

- Pi DCFD ,.1416
- PF1 DCFD 3,725e15, 124e13

4.1.6. END

Indica el final del archivo. Todos los archivos deben finalizar con esta directiva aun que se tengan varios archivos de código.

4.1.7. EQU

Esta directiva sirve para asignar valores a constantes, ya que no podrá ser modificado en ninguna parte del programa una vez haya sido ejecutado. La

declaración de constantes puede ser en cualquier parte del programa. Por conveniencia se escribirá al inicio de este.

Comúnmente se escribe de la siguiente forma:

Constante EQU Valor {tipo}

Por ejemplo:

- Contador EQU 20000
- PB1 EQU 0x40005004

El tipo es opcional y puede ser ARM, THUMB, CODE 32, CODE16 y DATA.

4.1.8. EXPORT

La directiva EXPORT indica un símbolo o subrutina que puede ser utilizado desde otro archivo distinto. El sinónimo para EXPORT seria global.

La sintaxis es:

EXPORT Subrutina

Por ejemplo:

- EXPORT Main

4.1.9. IMPORT

Esta directiva permite llamar una subrutina desde otro archivo que previamente haya sido exportada. Su sinónimo es EXTERN.

Su sintaxis es:

IMPORT Subrutina

Por ejemplo:

- IMPORT Start

4.1.10. LTORG

Indica al ensamblador que una todas las expresiones en una piscina de expresiones o *Literal Pool*. Programas largos necesitarán más de una piscina de expresiones. Esta debe colocarse debajo de saltos sin condición.

Su sintaxis es únicamente LTORG.

4.2. Thumb Instruction Set

Es un conjunto de instrucciones de ARM de 32 bits más utilizadas. Las instrucciones Thumb tienen 16 bits de longitud y una instrucción ARM de 32 bits correspondiente que realizan la misma función en el procesador. Gracias a esto, ARM y Thumb pueden trabajar juntos, puesto que las instrucciones Thumb se convierten a instrucciones ARM en tiempo real.

Thumb posee todas las características de un procesador de 32 bits. Entre ellas están:

- Direcciones de 32 bits
- Registros de 32 bits
- Unidad lógica aritmética

4.2.1. BRANCH

También llamados ramificaciones, permiten que el programador pueda modificar el curso de un programa cambiando el contador del programa. Existen diferentes tipos de saltos, entre ellos están los directos, indirectos y relativos.

4.2.1.1. B

La instrucción branch por defecto realiza un salto a la etiqueta adjunta. Su sintaxis es de la siguiente forma:

B {Condición} Etiqueta

Donde “{Condición}” es un sufijo que ayuda en comparaciones de registros, enteros o puntos flotantes, mientras “Etiqueta” es el nombre de la subrutina a donde se redirigirá el contador del programa. De los sufijos se hablará en la sección 4.2.2.

Por ejemplo:

- B Salto1
- BEQ Paso5

- BNE Vector1

4.2.1.2. BL

Mediante la ayuda del registro R14 o LR se podrá comprender con mayor facilidad el uso de este salto. Este registro se mantendrá en su estado por defecto, el cual es 0xFFFFFFFF hasta el momento que se ejecute la instrucción BL; entonces cambiará por la dirección de la siguiente línea en el código y hará el salto a la etiqueta que se haya colocado.

Su sintaxis es:

```
BL  Etiqueta
```

Se debe evitar utilizar la instrucción BL en cada salto pues se perderán las direcciones anteriores.

4.2.1.3. BX

Enlace e intercambio. Este comando realiza el salto hacia la dirección contenida en R14 o LR. Se debe escribir de la siguiente forma:

```
BX LR
```

4.2.2. Sufijos de condición

Existen sufijos que pueden ser utilizados por las instrucciones de condición. Estas no son únicamente para registros, también pueden ser utilizados para comparaciones de punto flotante. Estos se muestran en la tabla I.

Tabla I. **Sufijos de condición**

Sufijo	Significado
EQ	Igual
NE	No Igual
CS	Mayor o igual.
HS	Mayor o igual
CC	Menor o igual
LO	Menor o Igual
MI	Negativo
PL	Positivo
VS	Desbordamiento
VC	Sin desbordamiento
HI	Mayor que
LS	Menor o igual
GE	Mayor o igual
LT	Menor que
GT	Mayor que
LE	Menor o igual
AL	Siempre

Fuente: elaboración propia.

4.2.3. CMP

Para la utilización de esta instrucción, previamente se debe haber realizado una comparación, ya sea entre registros, un registro y un número, un registro y una constante o entre dos puntos flotantes. Este comando realiza una resta entre los dos operandos; no es afectado ninguno de ellos, sino únicamente la bandera de estado.

Su sintaxis es:

CMP Operando1, Operando2

Se usa junto con las instrucciones de salto y los sufijos de condición, puesto que se compara con esta instrucción. Dependiendo del resultado se realizará o no el saldo condicionado.

Esta instrucción también puede ser utilizada para punto flotante, únicamente se agrega VCMP.F32, pero no levantará ninguna bandera. Para lograrlo se deberá colocar la siguiente línea de código:

VMRS APSR_nzcv, FPSCR

4.2.4. Operaciones

Mediante la ALU, el microcontrolador puede realizar las siguientes operaciones:

Tabla II. **Operaciones mediante la ALU**

Instrucción	Explicación	Sintaxis	Ejemplo

ADD	Realiza la operación suma entre dos operandos.	ADD Rn, Rn+1, Rn+2
		
		ADD Rn, Rn+1
			ADD R1, R2, R3 ;
			R1= R2+R3
			
			ADD R1, R2;
			R1=R1+R2

SUB	Realiza la operación resta entre dos operandos.
	(Este comando no está disponible para Keil en ARM)
		SUB Rn, Rn+1, Rn+2
		
		SUB Rn, Rn+1
			SUB R1, R2, R3 ;
			R1= R2-R3
			
			SUB R1, R2;
			R1=R1-R2

MUL	Multiplicación entre dos operandos con 32bits de resultado.	MUL Rn, Rn+1, Rn+2
		
		MUL Rn, Rn+1
			MUL R1, R2, R3 ;
			R1= R2*R3
			
			MUL R1, R2;
			R1=R1*R2

BIC	Realiza una limpieza de bits.	BIC Rn, Rn+1, Operando.	BIC R1, R2, R3 El resultado estará en R1.
ORR	Realiza una función OR lógica.	ORR Rn, Rn+1, Operando	ORR R1, R2, R3 El resultado estará en R1.
AND	Realiza la operación lógica AND	AND Rn, Rn+1, Operando	AND R1, R2, R3
NOP	No realiza nada durante un ciclo de reloj.	NOP	NOP Equivalente a escribir: MOV R0, R0

Fuente: elaboración propia.

4.2.5. Instrucciones de carga y almacenaje

Estas instrucciones tendrán gran importancia al momento de la programación puesto que ellas son las encargadas de mover, almacenar y cargar información desde o hacia memoria, de forma directa o desde otra localidad.

4.2.5.1. LDR

LDR es el equivalente a *Load* o carga. Esta instrucción es utilizada generalmente para cargar un valor en memoria hacia un registro.

La sintaxis generalmente es:

LDR R1, [R2]

Esto quiere decir que el valor que se encuentre dentro de la parte de la memoria que ocupa R2 será asignado en R1. Básicamente es R1 = [R2].

4.2.5.2. STR

STR equivalente a *Store* o almacenar. Es utilizada para guardar un dato que se encuentre en un registro en la memoria.

La sintaxis de esta instrucción es de la siguiente forma:

STR R1, [R2]

Esto indica que el valor de R1 será almacenado en un espacio de memoria de R2. Esto sería [R1] = R2.

4.2.5.3. MOV

Equivalente a *Move* o mover. Generalmente sirve para asignarle un valor a un registro. Este valor no puede ser mayor a 8 bits o 255 en decimales.

Su sintaxis es de la siguiente manera:

```
MOV R1, #2
MOV R1, R2
```

En la primera línea al registro de propósito general R1 se le está asignando el valor de 2 decimal, mientras que en la segunda línea al registro R1 se le asigna el valor que tenga el registro R2.

4.3. Puertos

El microcontrolador TM4C123GH6PM cuenta con puertos que van del A al F, generalmente contarán con un máximo de 8 pines disponibles. La mayoría de los pines pueden ser utilizados de forma digital tanto de lectura como escritura.

4.3.1. Puertos disponibles

No todos los puertos son utilizables, en la siguiente tabla se detallarán los que pueden ser programados por el usuario.

Tabla III. **Tabla con todos los puertos disponibles**

Puerto	Pines disponibles	
Puerto A	PA2, PA3, PA4, PA5, PA6, PA7	Todos los pines pueden ser utilizados como pines digitales.
Puerto B	PB0, PB1, PB2, PB3, PB4, PB5, PB6, PB7	
Puerto C	PC4, PC5, PC6, PC7	
Puerto D	PD0, PD1, PD2, PD3, PD6, PD7	
Puerto E	PE0, PE1, PE2, PE3, PE4, PE5	
Puerto F	PF0, PF1, PF2, PF3, PF4	

Fuente: elaboración propia.

Todos estos puertos pueden ser utilizados como entradas y salidas digitales. Se deberá tomar en cuenta que el puerto F está conectado a los botones de la Tiva (PF0 y PF4) y a los leds internos (PF1-PF3). Estos pueden ser utilizados, pero deberán ser desbloqueados.

Tabla IV. **Direcciones base para cada puerto**

Puerto	Constante
PA	**0x40004000**
PB	**0x40005000**
PC	**0x40006000**
PD	**0x40007000**
PE	**0x40024000**
PF	**0x40025000**

Fuente: elaboración propia.

4.3.2. Puertos dedicados

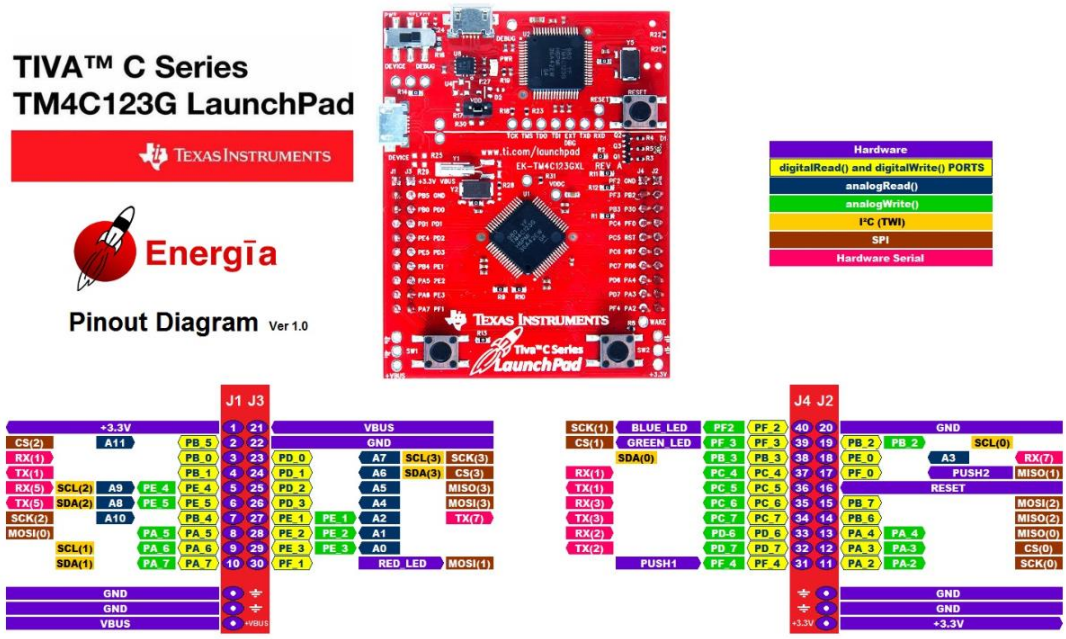
No todos los pines pueden ser utilizados como puertos I/O pero todos pueden ser configurados como tal, por lo que se deberá tener mucho cuidado al momento de habilitar los puertos para no dañar el microcontrolador. Los puertos que no deben ser configurados son:

Tabla V. **Puertos no programables**

Puerto	Pin dedicado	Función
PA	PA0 y PA1	Estos están conectados el puerto serial. U0Rx y U0Tx
PC	PC0-PC3	Estos pines sirven para el funcionamiento de depurador, conectados al JTAG.
PD	PD4 y PD5	Están conectados al USB

Fuente: elaboración propia

Figura 9. **Microcontrolador pinout**



Fuente: IoT. *Getting Started with Tiva C Series MCU*. https://vksgaikwad3.wordpress.com/2016/04/25/getting-started-with-tiva-c-series-mcu/. Consulta: 20 de enero de 2020

4.3.3. Configuración de los puertos

Para poder configurar los puertos como entrada o salida se deberá seguir una serie de pasos. Estos deberán de repetirse para cada puerto que se vaya a utilizar, no para cada pin, a menos que se requiera que algunos pines sean de entrada y otros de salida dentro del mismo puerto.

4.3.3.1. Clock

Como primer punto se deberá de configurar el reloj del microcontrolador. Se creará una constante con el nombre de SYSCTL_RCGCGPIO_R con la dirección de 0x400FE608. Posterior a eso se deberá configurar una máscara en la cual los valores iguales a 1, son los puertos que se utilizarán.

Tabla VI. **Constante para el reloj de cada puerto**

Puerto	BIT	Constante
F	5	X20
E	4	x10
D	3	x08
C	2	x04
B	1	x02
A	0	x01

Fuente: elaboración propia.

Si se quisiera activar el reloj para cualquier puerto se deberá configurar la constante SYSCTL_RCGCGPIO_R EQU 0x400FE608 y posteriormente realizar una operación OR entre los valores de SYSCTL_RCGCGPIO_R y el valor de la constante de reloj para cada puerto.

4.3.3.2. LOCK

LOCK o desbloquear únicamente será necesario si se utilizarán los pines PF0, PF1 y PD7. Se creará una constante con el nombre GPIO_PORTX_LOCK_R EQU 0x400ZZ520 donde X es el puerto, en este caso F, y ZZ será la constante según el puerto de la tabla III.

4.3.3.3. AMSEL

AMSEL permite habilitar un pin en modo analógico, pero al no ser esto lo que se necesita, se deberá asignar el valor de 0. Se debe crear una variable con el nombre de GPIO_PORTX_AMSEL_R EQU 0x400ZZ528 donde X es el puerto asignado y ZZ las constantes para dicho puerto. Para esta constante se utiliza la tabla III.

4.3.3.4. PCTL

PCTL permite desactivar funciones alternativas configurándolo con función digital únicamente. Se creará la constante GPIO_PORTX_PCTL_R EQU 0x400ZZ52C. Este será el único paso, donde no se utilizará la dirección del puerto base, sino que teniendo este número en binario 2_11111111, cada 1 representa un pin. El primer 1 de derecha a izquierda es el valor correspondiente al pin 0; el siguiente al pin 1 y en secuencia hasta el pin 7.

4.3.3.5. DIR

Este especifica si será un puerto de entrada o salida. Si en la dirección del pin tiene un 1, este será salida; por el contrario, si tiene un 0, será entrada. Su constante es GPIO_PORTX_DIR_R EQU 0x400ZZ420.

4.3.3.6. AFSEL

Deshabilita las funciones alternativas, deja únicamente el puerto como un puerto digital. De igual forma se debe configurar una constante en este caso será GPIO_PORTX_AFSEL_R EQU 0x400ZZ420, donde X es el puerto a utilizar y ZZ son las constantes de dicho puerto.

4.3.3.7. DEN

DEN habilita la función digital en los puertos, ya que es habilitada este debe de ser 1. Como se vio en la sección 4.2.4, se deberá realizar una operación OR entre los elementos de la constante GPIO_PORTX_DEN_R EQU 0x400ZZ51C y los valores de cada pin.

4.3.3.8. PUR

PUR permite la desactivación de las resistencias internas con las que cuenta el puerto F internamente. Para esto se declarará el valor de la constante GPIO_PORTF_PUR_R EQU 0x400XX510 para poderlos deshabilitar.

4.3.3.9. CR

Este registro previene el desbloquear los pines dedicadas al JTAG del microcontrolador. El valor de esta constante será GPIO_PORTF_CR_R EQU 0x400XX524.

4.3.3.10. Pines

Cada puerto consta de un máximo de 8 pines. Cada pin tiene un valor diferente para ser tanto habilitado como encendido.

Tabla VII. **Direcciones base para el pin**

Bit	Constante
7	0x0200
6	0x0100
5	0x0080
4	0x0040
3	0x0020
2	0x0010
1	0x0008
0	0x0004

Fuente: elaboración propia.

Tabla VIII. **Valor para encender cada pin**

“Encender” un bit	Constante
7	x80
6	x40
5	X20
4	x10
3	x08
2	x04
1	x02
0	x01

Fuente: elaboración propia.

4.4. Arreglos

Un arreglo es un conjunto de datos que pertenecen a un mismo tipo. Consta de posiciones de memoria consecutivas, las cuales pueden ser accedidas de forma ascendente, descendente y en desorden.

Para esto se utilizarán las directivas mostradas en la sección 4.1.3 y 4.1.4. con las cuales se podrá declarar arreglos de distintos tipos de datos. Además, se utilizará una etiqueta especial para cada arreglo. Se declaran de la siguiente forma:

Etiqueta DCD Valores en binario, valores en binario, valores en binario.

Por ejemplo

Arreglo1

DCD 2_00011111,2_10011100,2_00000001

El tamaño del arreglo dependerá del programador y la cantidad de valores que se utilizarán.

4.5. Interrupciones

Una interrupción tiene la función de dar atención a algún circuito interno o externo únicamente cuando es llamada. Esta evita el constante *polling.* El mismo puerto debe ser capaz de indicar cuándo ha sido activada.

4.5.1. Beneficios

Existen muchos beneficios de la utilización de interrupciones:

- Eficiencia en el tiempo de procesamiento del microcontrolador
- Mejora el proceso de multitasking
- Es más veloz al ejecutar instrucciones
- Proporciona niveles de prioridad entre instrucciones

4.5.2. Desventajas

- Puede necesitar de circuitos externos al microcontrolador
- Difícil de programar

Dentro de los tipos de interrupciones se pueden encontrar:

4.5.3. Enmascarables

Una interrupción enmascarable será aquella que puede ser ignorada por software; dicho de otra forma, que el programador puede indicarle al microcontrolador que no atienda la interrupción y que siga con lo que está realizando en ese momento.

4.5.4. No enmascarables

Esta interrupción hace que el microcontrolador deje de hacer lo que está haciendo y realice un salto a una instrucción en particular. Estas no pueden evitarse, puesto que son interrupciones internas del microcontrolador.

El microcontrolador tiene 4 fuentes de interrupciones no enmascarables:

- *Non-maskarable interrupt* (NMI) Pin: para los pines PD7 y PF0, por esto deben desbloquearse al momento de su utilización como puerto digital.

- Falla de verificación del oscilador principal: si el oscilador está funcionando demasiado rápido o demasiado lento, se activa la interrupción.

- INTCTRL o Interrupt Control and State: genera la excepción de mayor prioridad a parte de resert.

- Watchdog Control: este puede ser configurado para generar una señal de reset, únicamente puede ser desactivado con reset de hardware o software.

4.5.5. Timer

Es un módulo interno de los microcontroladores cuya función principal es contar automáticamente a una velocidad determinada, evitando el uso de contadores y del procesador. Esto permite el cumplimiento de ciertas funciones mientras el microcontrolador realiza otras.

Figura 10. **Timer con configuración preestablecida para 16-bit**

Prescale (8-bit value)	# of Timer Clocks (Tc)[a]	Max Time	Units
00000000	1	0.8192	ms
00000001	2	1.6384	ms
00000010	3	2.4576	ms
----------	--	--	--
11111101	254	208.0768	ms
11111110	255	208.896	ms
11111111	256	209.7152	ms

a. Tc is the clock period.

Fuente: Department of Computer Science and Engineering IIT Bombay. *Tiva™ TM4C123GH6PM Microcontroller*. https://www.cse.iitb.ac.in/~erts/html_pages/Resources/Tiva/tm4c123gh6pm-Datasheet.pdf. Consulta: 1 de marzo de 2020

4.5.6. Configuración de interrupciones por timer

Para la configuración de interrupciones se tendrán que seguir los siguientes pasos:

4.5.6.1. RCGCTIMER

Este permite la habilitación y deshabilitación del módulo del timer. Al tenerlo deshabilitado permite ahorrar energía y su acceso a los módulos genera un error; al estar habilitado permite el uso del reloj del timer. Su variable principal será GPTM_RCGCTIMER_R con el valor de 0x400FE604.

4.5.6.2. CFG

Este registro permite configurar de forma global el módulo de los *timers* de propósito general y determina si estará funcionando en 32 o 64 bits, los

cuales son los timer concatenados, o en 16 o 32 bits, que son los modos individuales.

Tabla IX. **CFG según su timer**

TIMER	Valor
TIMER0	0x40030000
TIMER1	0x40031000
TIMER2	0x40032000
TIMER3	0x40033000
TIMER4	0x40034000
TIMER5	0x40035000

Fuente: elaboración propia.

4.5.6.3. CTL

Utilizado junto con CFG para afinar la configuración del timer y habilitar la detención del temporizador y el disparador de salida, este último generalmente es utilizado para iniciar la transferencia en el módulo de ADC. Con los valores de la tabla VIII, únicamente se debe cambiar el último 0 por una C.

Para el Timer0 el valor de la constante sería:

```
GPTM_TIMER0_CTL_R          EQU 0x4003000C
```

4.5.6.4. TIMAMODE

Este registro configura los timers de propósito general. En función de la configuración de CFG, controla los modos del temporizador A cuando se utiliza

solamente este; si se utiliza los timers A y B, este controlará ambos. También conocido como TAMR.

```
GPTM_TIMER0_TIMAMODE_R          EQU 0x40030004
```

Tabla X. **Valores posibles al inicializar el modo del timer A**

Timer A Mode	Valor
Reservado	0x00
Temporizador de disparo	0x01
Periódico	0x02
Modo de captura	0x03

Fuente: BAI, Ying. *Practical Microcontroller Engeniering with ARM® Technology*. p. 707.

4.5.6.5. INTMASK

También llamado IMR o Interrupt Mask Register, este permite al programa la habilitación o deshabilitación del controlador del nivel de interrupción. A este se le debe colocar el offset 0x018, para el timer 0 el valor sería:

```
GPTM_TIMER0_INTMASK_R          EQU 0x40030018;
```

4.5.6.6. MICLR

Conocido también como ICR o Interrupt Clear, este permite limpiar el estado de los registros RIS y MIS. Su offset será 0x024, en el caso del timer 0 este será:

```
GPTM_TIMER0_MICLR_R                EQU   0x40030024;
```

4.5.6.7. RIS

Por su nombre en inglés Raw Interrupt Status, este registro muestra el estado interno de la señal de interrupción. El offset es 0x01C, para el Timer 0 será igual a:

```
GPTM_TIMER0_RIS_R                EQU 0x4003001C;
```

4.5.6.8. MIS

Masked Interrupt Status por sus siglas en inglés, es un registro que muestra el estado del controlador de nivel de interrupción. Si una interrupción es no enmascarable y es activada, el bit es fijado en este registro. Su offset es 0x020, para el timer 0 es:

```
GPTM_TIMER0_MIS_R                EQU 0x40030020;
```

4.5.6.9. TIMAIL

Timer A Interval Load con offset 0x028: este registro carga el valor inicial de la cuenta al timer si está decreciendo; si está aumentando, indica el límite mayor para detener el timer. Para el timer 0 el valor es:

```
GPTM_TIMER0_TIMAILR_R            EQU 0x40030028;
```

4.6. Startup.s

Este archivo es sumamente importante para la ejecución de cualquier programa, pues prepara el microcontrolador para que sepa en qué momento

inicia el programa, cómo se utilizarán las interrupciones, la ubicación de la pila. Este únicamente se utiliza una vez en cada reinicio o nuevo programa.

En él se encuentran las interrupciones enmascarables y las interrupciones por fallo en el código, se crean loops para evitar fallos mayores. Además, en este se habilita los registros FPU.

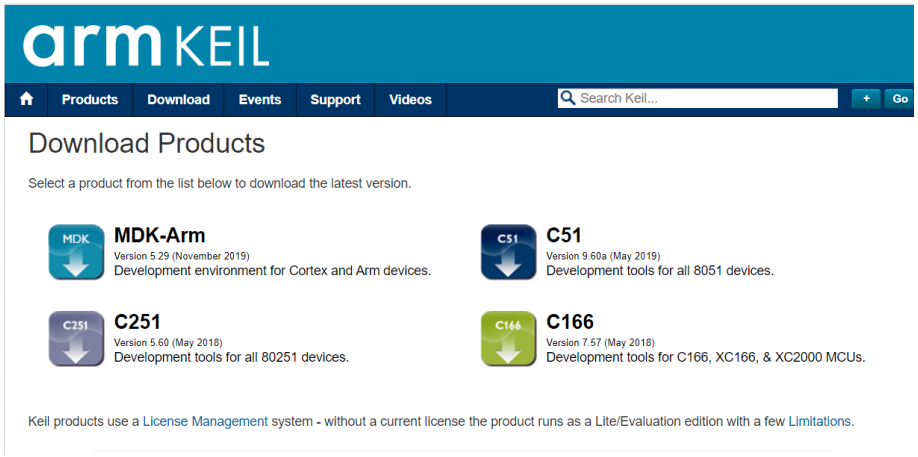
5. INTRODUCCIÓN A LA PROGRAMACIÓN DEL MICROCONTROLADOR TM4C123GH6PM EN LENGUAJE ENSAMBLADOR

5.1. Instalación del programa y creación de un nuevo proyecto

Previo al inicio de las prácticas, el estudiante deberá instalar la o las herramientas necesarias para la programación del microcontrolador.

Se deberá descargar la última versión de MDK-Arm, Development environment for Cortex and Arm devices (https://www.keil.com/download/product/).

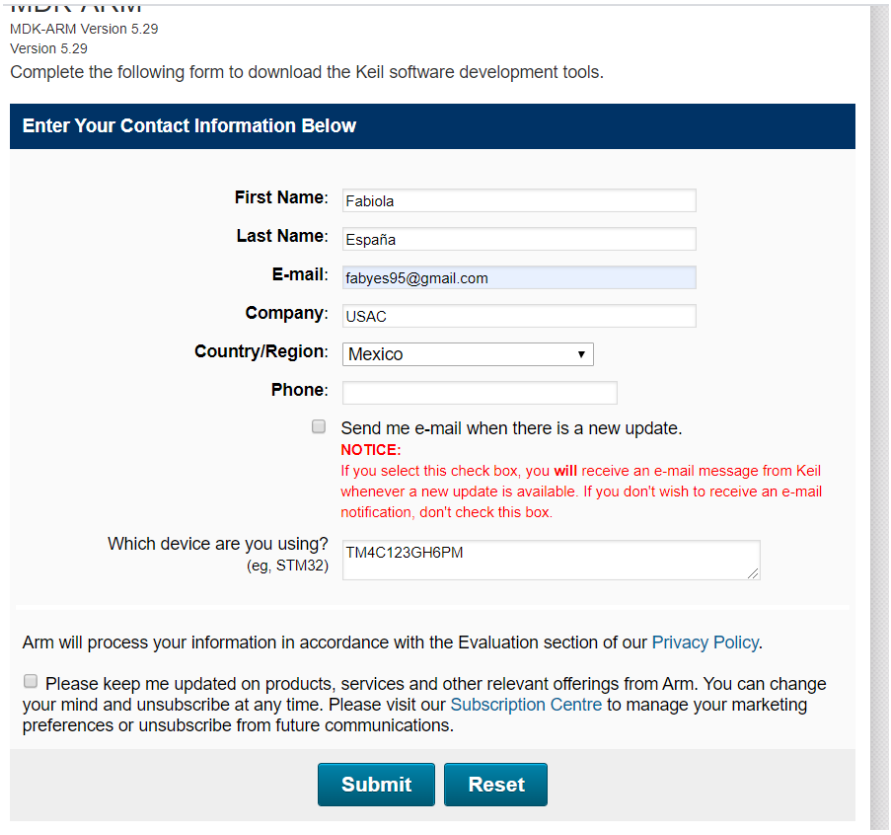
Figura 11. **Página de descarga del programa**



Fuente: elaboración propia.

Esto redirigirá a un link donde se deberán llenar los datos correspondientes para cada estudiante. En país no aparece Guatemala, por lo que puede escoger el que se desee.

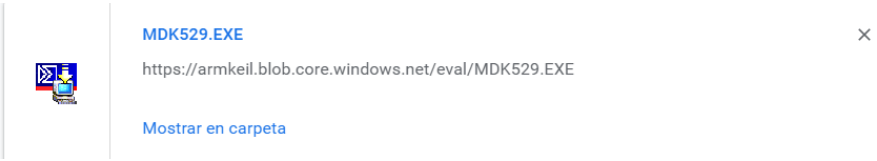
Figura 12. **Datos para descarga del programa**



Fuente: elaboración propia.

Se deberá descargar el ejecutable con el nombre MDK529.EXE de 855,164 K hasta el 10 de febrero de 2020.

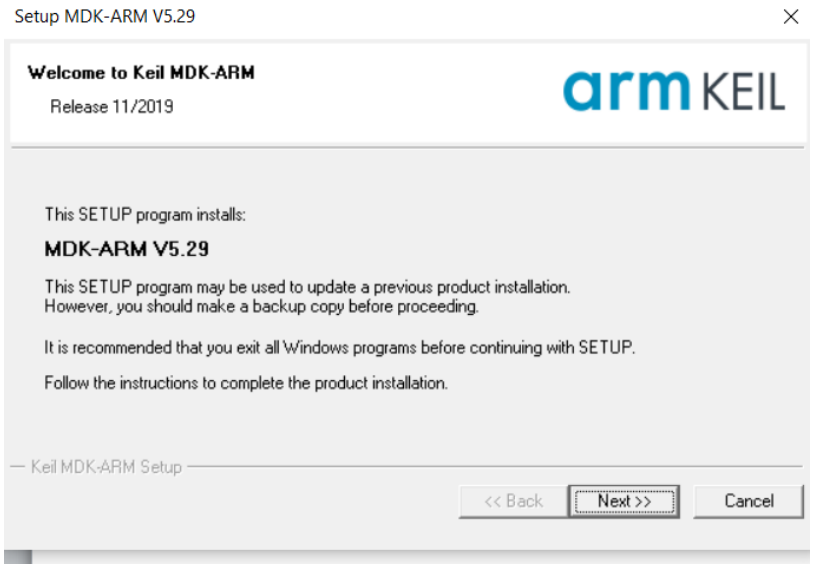
Figura 13. **Archivo descargado**



Fuente: elaboración propia.

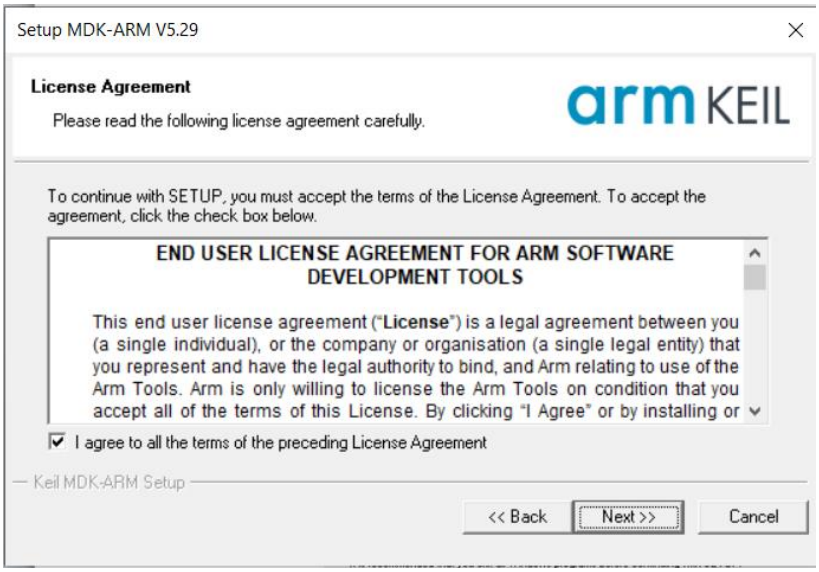
Una vez el archivo haya sido descargado, deberá ejecutarse siguiendo las sencillas instrucciones.

Figura 14. **Inicio de instalación**



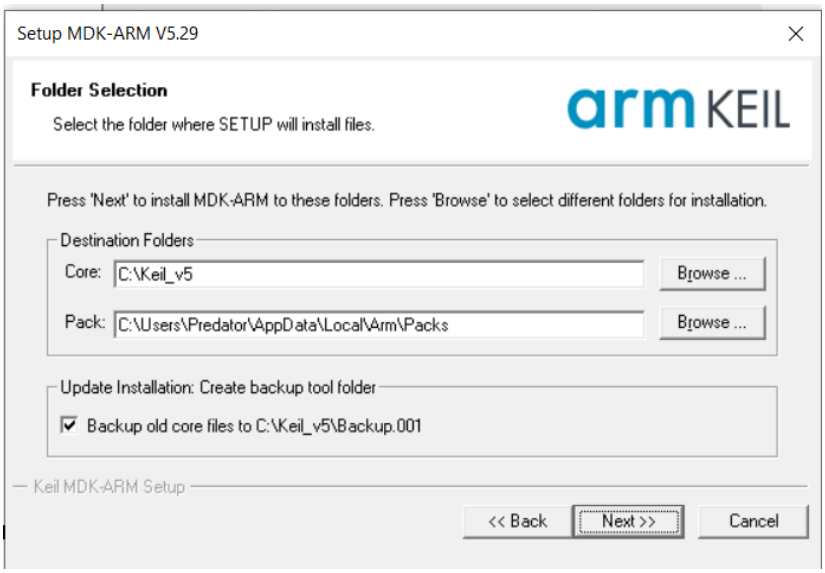
Fuente: elaboración propia.

Figura 15. **Aceptar los términos de la licencia**



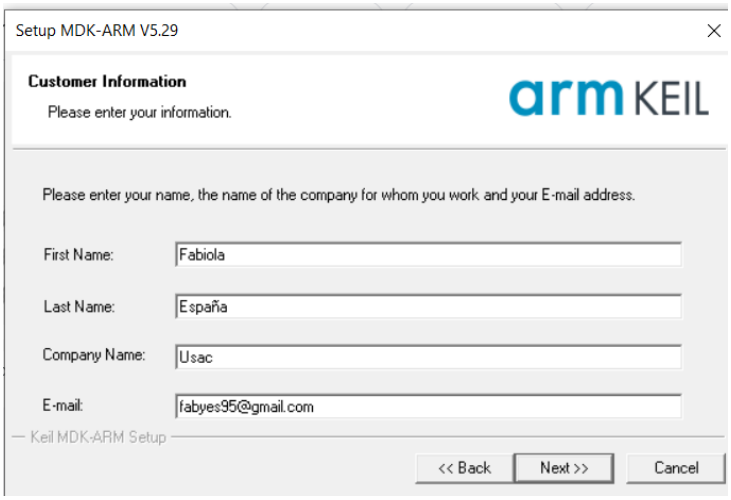
Fuente: elaboración propia.

Figura 16. **Elegir ubicación del programa**



Fuente: elaboración propia.

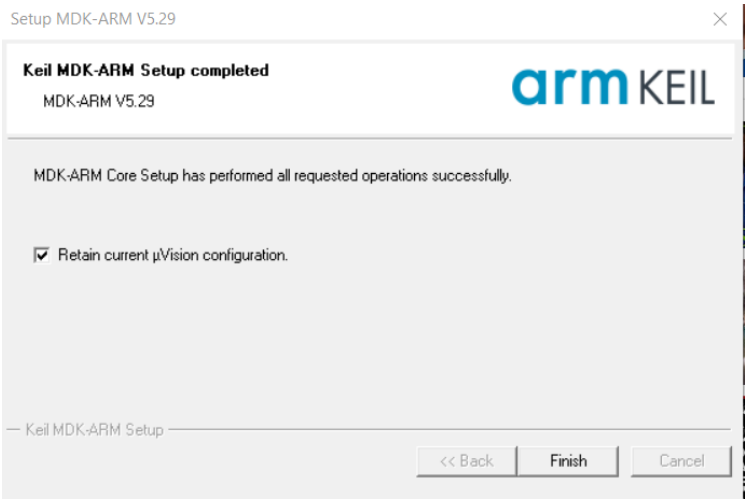
Figura 17. **Información del cliente**



Fuente: elaboración propia.

Se debe completar esta sección con la misma información de la figura 13.

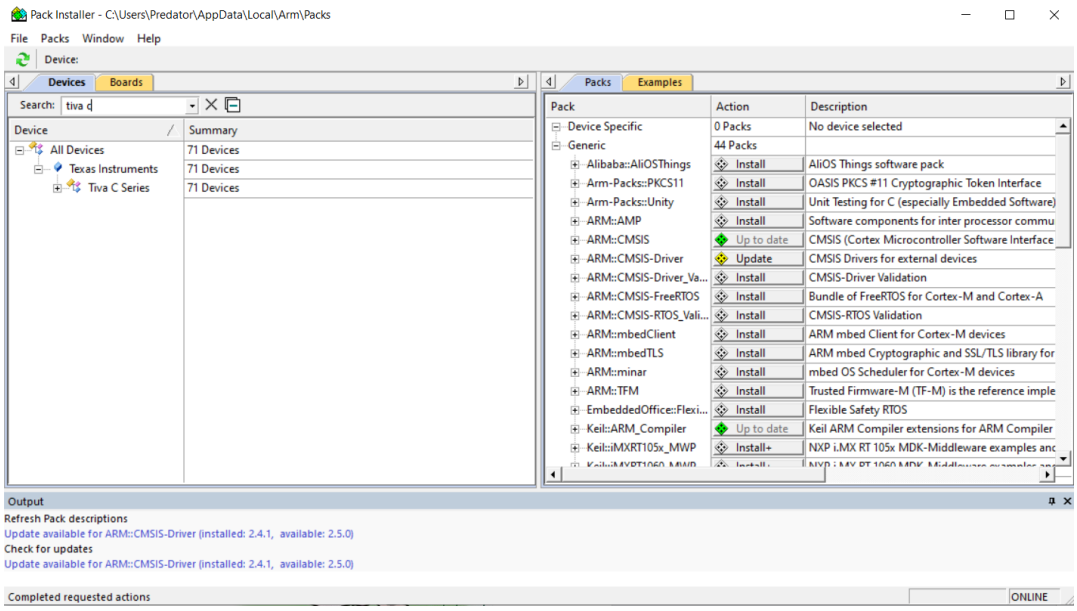
Figura 18. **Termina la instalación**



Fuente: elaboración propia.

Cuando ya esté instalado el programa, se deberá descargar los controladores del dispositivo que se utilizarán.

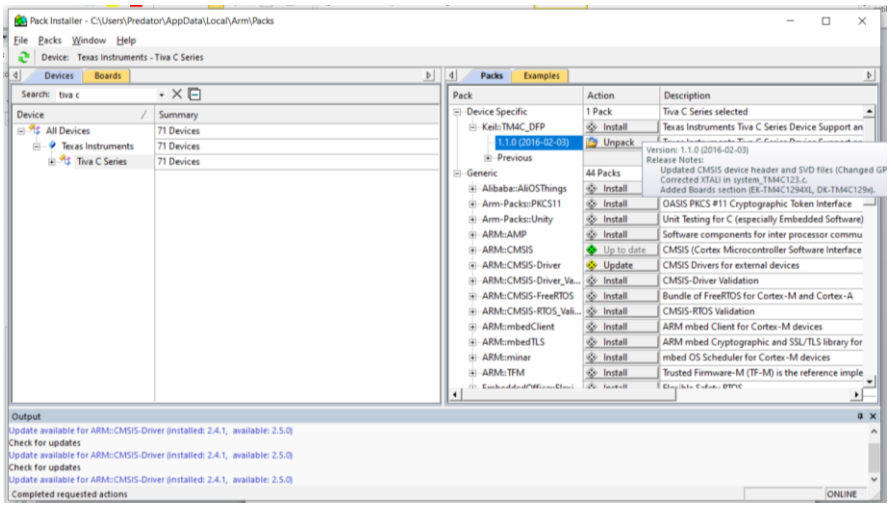
Figura 19. **Instalador de paquetes**



Fuente: elaboración propia.

En el buscador deberá buscarse Tiva C Series, presionar dos veces el clic izquierdo y aparecerá del lado derecho de la pantalla la opción de descargar los paquetes para todos los modelos de Tiva.

Figura 20. **Instalar paquetes**



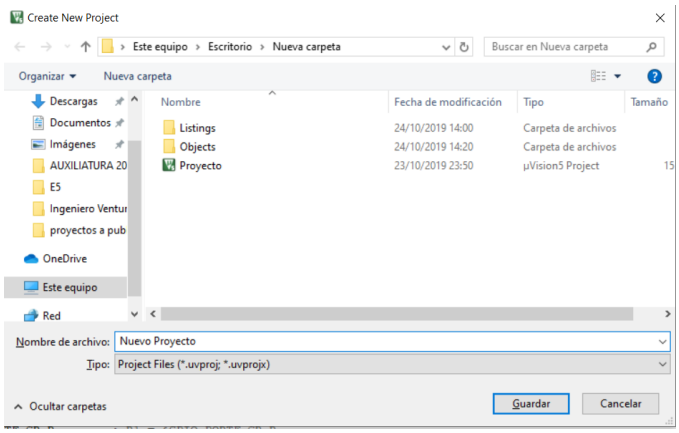
Fuente: elaboración propia.

Ya con los paquetes instalados, se debe crear un nuevo archivo donde se empezará a programar.

Los pasos para crear el archivo son:

- Clic izquierdo a la pestaña de Project
- Clic izquierdo a New uVision Project
- Se escribe el nombre deseado del proyecto

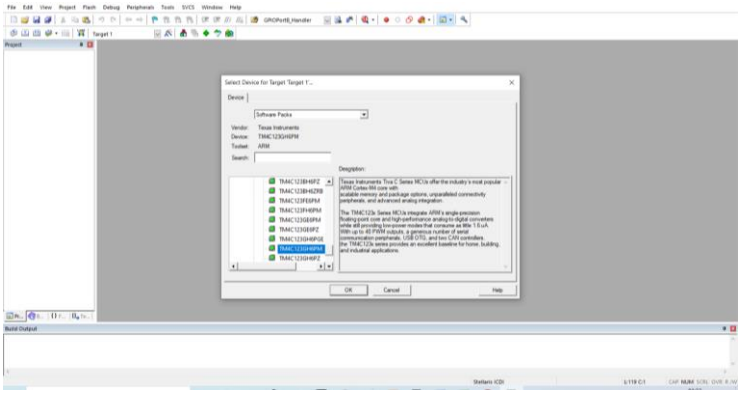
Figura 21. **Nombre del proyecto**



Fuente: elaboración propia.

- Se debe seleccionar el microcontrolador con el que se estará trabajando. En este caso se deberá seleccionar el microcontrolador TM4C123GH6PM

Figura 22. **Seleccionar TM4C123GH6PM**



Fuente: elaboración propia.

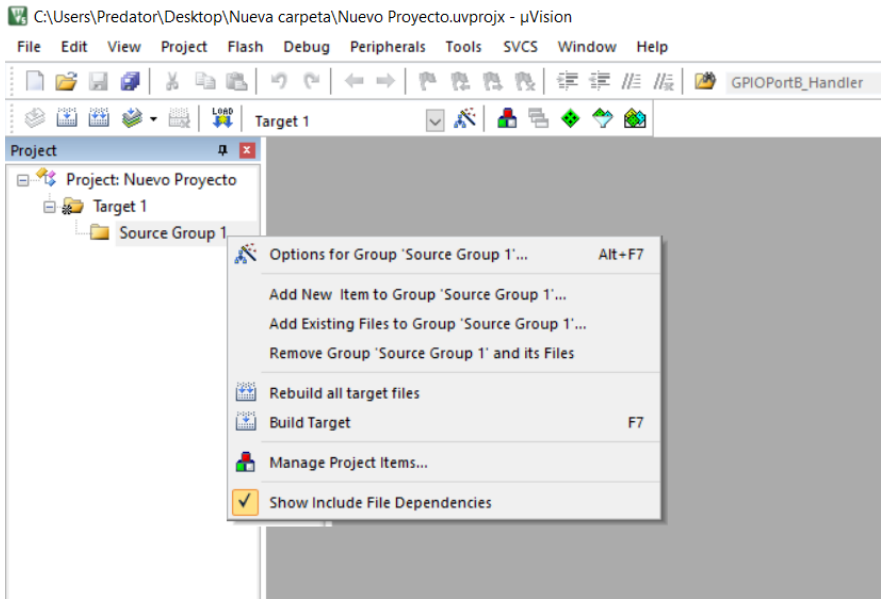
Con esto el usuario estará listo para empezar a programar.

5.2. Agregar archivos existentes al proyecto, Startup.s, directivas o encabezado

Con el archivo nuevo creado, se debe agregar también el archivo Startup.s. Para esto, se realizarán los siguientes pasos:

- Clic derecho en la carpeta de Source Group 1

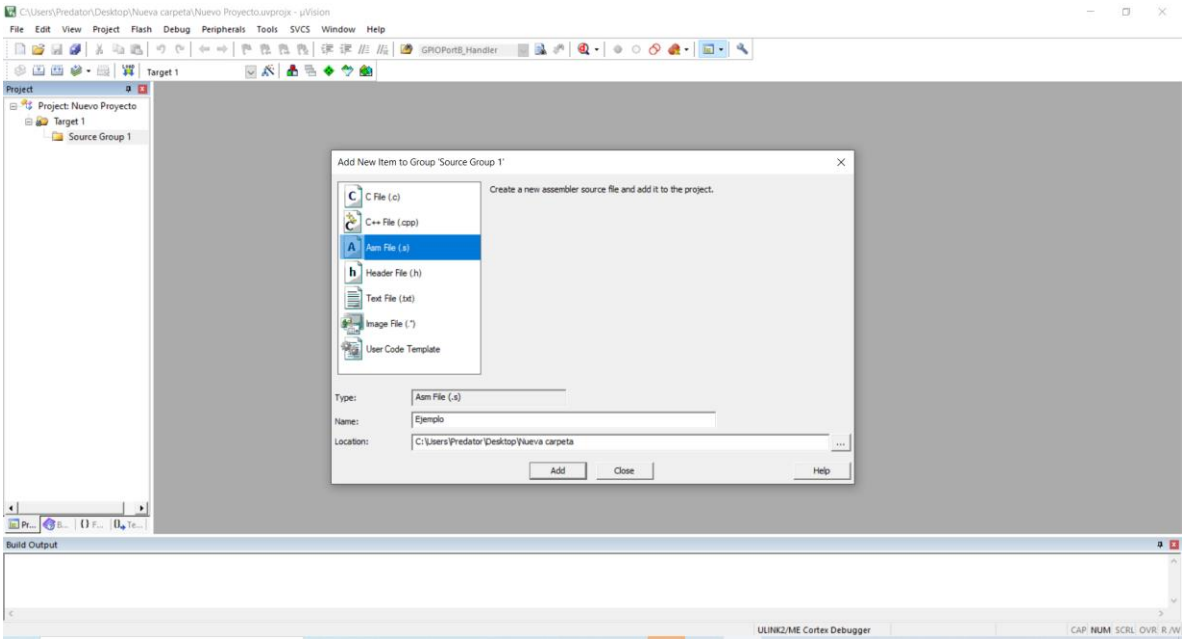
Figura 23. **Recursos del proyecto**



Fuente: elaboración propia.

- Clic derecho en Add New Item To Group ‘Source Group1’

Figura 24. **Nuevo archivo**



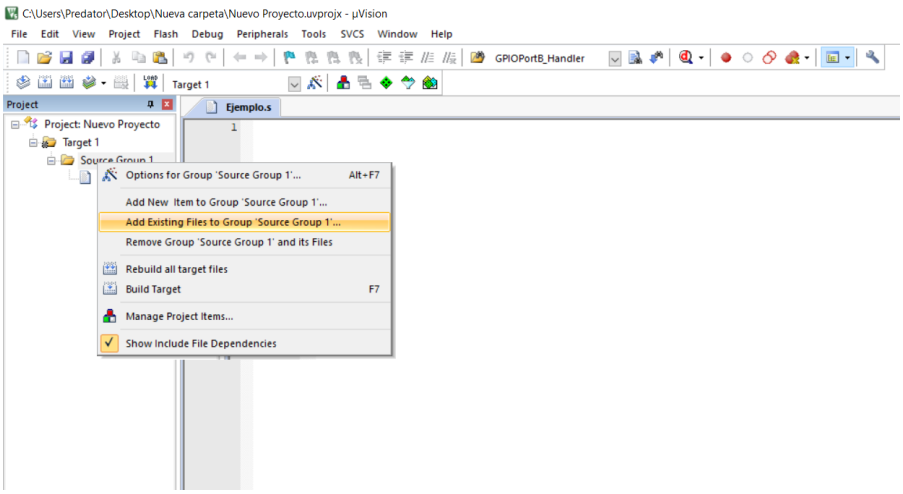
Fuente: elaboración propia.

Se debe especificar el tipo de archivo; en este caso será tipo Asm File. Se escribe el nombre del archivo y su ubicación.

Se debe agregar el archivo Startup.s para indicarle al microcontrolador cuál será su funcionamiento.

- Seleccionar Add Existing File to Group ‘Source Group 1’

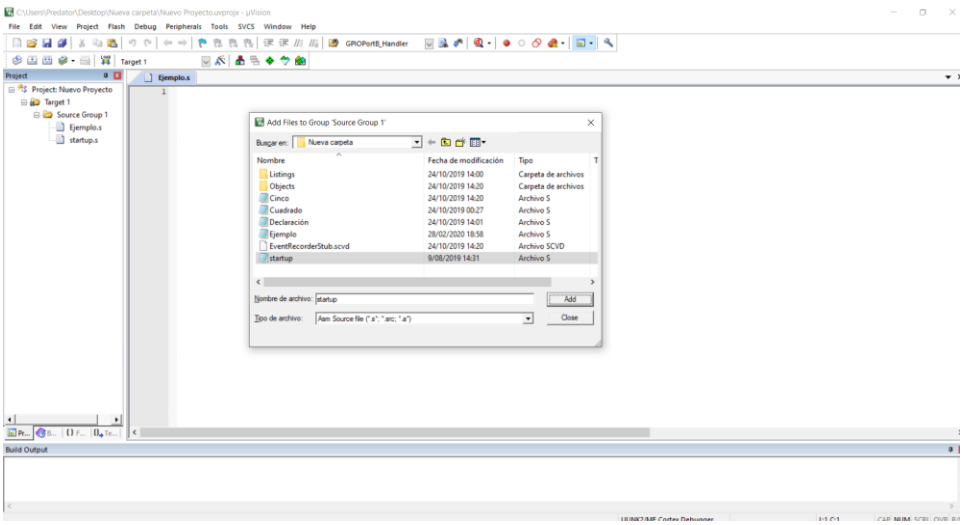
Figura 25. **Recuerdo de grupo 1**



Fuente: elaboración propia.

- Se debe indicar que el archivo es de tipo Asm Source File y se agregará.

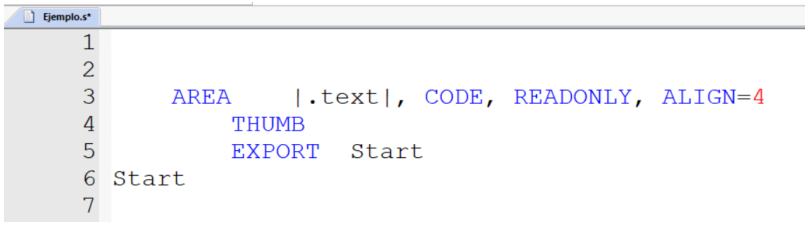
Figura 26. **Añadir Startup.s**



Fuente: elaboración propia.

- Se deberán escribir las directivas necesarias para iniciar un archivo nuevo, en este caso se escribirá de encabezado lo siguiente:

Figura 27. **Encabezado**

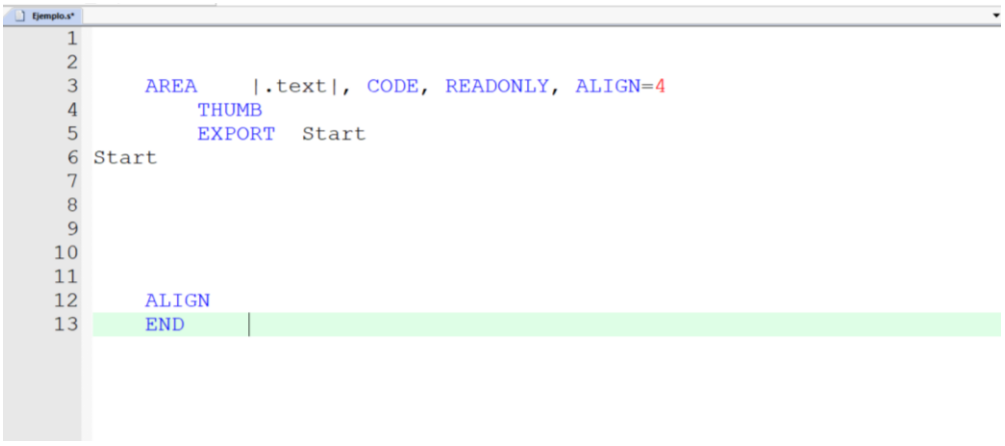


Fuente: elaboración propia.

Esto indicará que a partir de allí se deberá ensamblar un código nuevo.

- Además, se debe señalar el final del documento, donde se indica que termina la alineación y se deja de ensamblar.

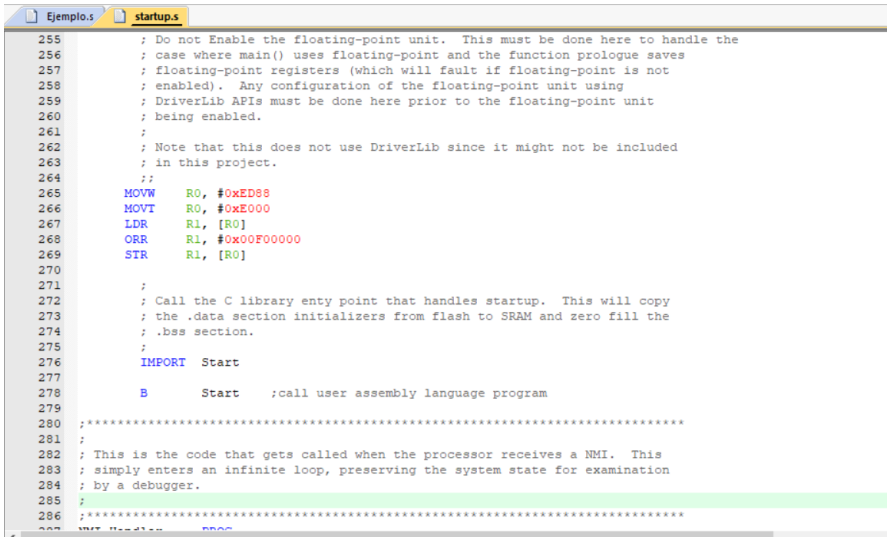
Figura 28. **Final del documento**



Fuente: elaboración propia.

Se puede observar que, además, se está exportando el nombre de la subrutina Start; esto permite su utilización desde otro archivo del mismo proyecto. En este caso, se inicializará desde el archivo Startup.s, de allí la importancia del archivo.

Figura 29. **IMPORT Start**



Fuente: elaboración propia.

Se puede observar en la línea 276 del archivo Startup.s que se importa la subrutina para que esta se pueda llamar con el salto B en la línea 278, la cual redireccionará instantáneamente a la línea 7 del archivo Ejemplo.s en la imagen 28.

5.3. Manejo de saltos e instrucciones de condición, subrutinas, simulador

Como parte de la programación se debe conocer los saltos y sufijos de condición. Para esto se realizará un breve ejemplo sin utilizar el microcontrolador, únicamente el simulador. Se hará la aproximación de Leibniz para el número π.

Figura 30. **Aproximación al número π**

$$\pi = 4\left(\sum_{k=1}^{\infty} \frac{(-1)^{(k+1)}}{(2k-1))}\right)$$
$$= 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots\right)$$

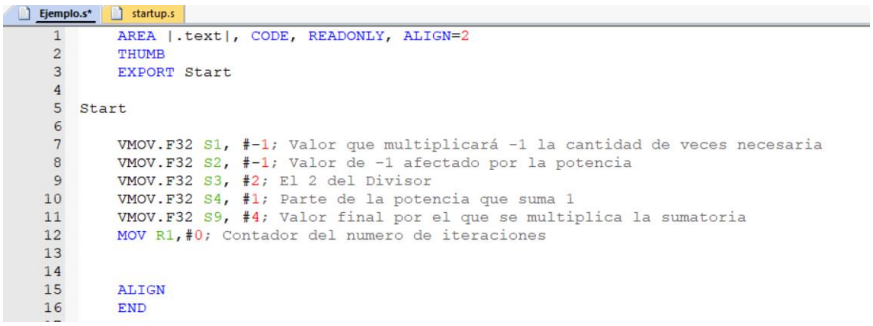
Fuente: elaboración propia.

Con base en la fórmula de la sumatoria de la imagen anterior se realizará la aproximación con un número de 100 iteraciones.

Con el archivo Ejemplo.s, como se puede observar en la imagen 28, se comenzará la programación de la serie. Por sencillez se trabajará con valores de punto flotante.

- Primero se deberá asignar todos los valores que se utilizarán. Para esto se empleará la instrucción VMOV.F32 para punto flotante y MOV para enteros.

Figura 31. **Valores iniciales**

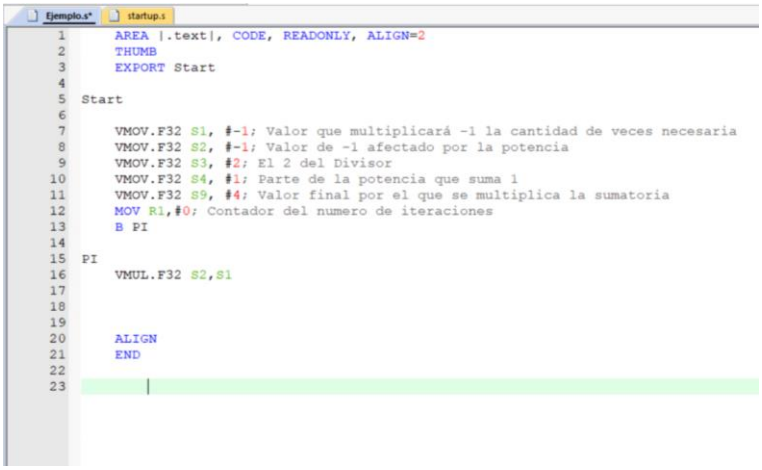


Fuente: elaboración propia.

En este lenguaje no es necesario escribir “;” al finalizar una línea de código, sino únicamente para realizar comentarios.

- Posteriormente se creará la segunda subrutina, la cual indicará el numerador de la operación.

Figura 32. **Numerador**

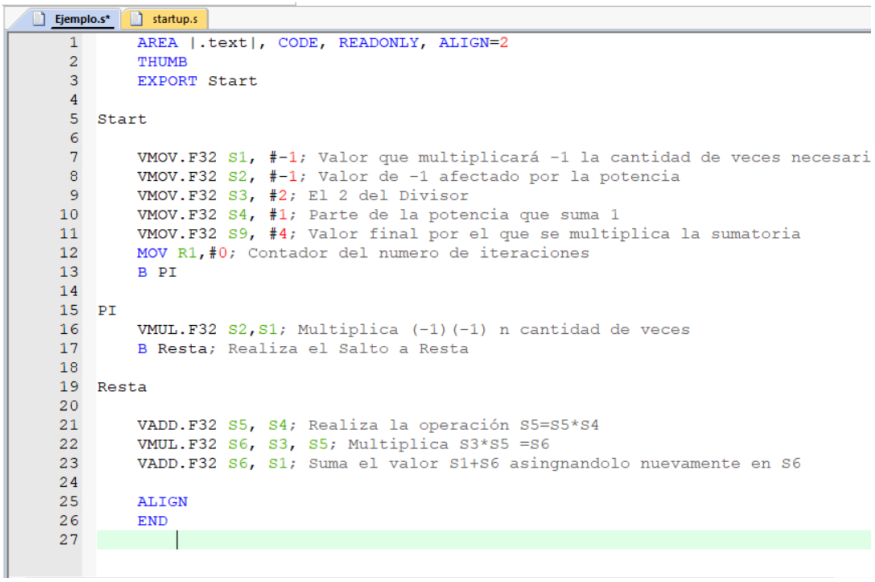


Fuente: elaboración propia.

En este paso se realiza el primer salto del programa, con el comando B y la etiqueta de la subrutina.

- Posteriormente se realiza la operación del denominador

Figura 33. **Denominador**



Fuente: elaboración propia.

- Luego de la realización de las operaciones por separado, se proseguirá a realizar la división.

Figura 34. **División**

```
Ejemplo.s*  startup.s
 5  Start
 6
 7      VMOV.F32 S1, #-1; Valor que multiplicará -1 la cantidad de veces necesaria
 8      VMOV.F32 S2, #-1; Valor de -1 afectado por la potencia
 9      VMOV.F32 S3, #2; El 2 del Divisor
10      VMOV.F32 S4, #1; Parte de la potencia que suma 1
11      VMOV.F32 S9, #4; Valor final por el que se multiplica la sumatoria
12      MOV R1,#0; Contador del numero de iteraciones
13      B PI
14
15  PI
16      VMUL.F32 S2,S1; Multiplica (-1)(-1) n cantidad de veces
17      B Resta; Realiza el Salto a Resta
18
19  Resta
20
21      VADD.F32 S5, S4; Realiza la operación S5=S5*S4
22      VMUL.F32 S6, S3, S5; Multiplica S3*S5 =S6
23      VADD.F32 S6, S1; Suma el valor S1+S6 asignandolo nuevamente en S6
24      B Division
25
26  Division
27      VDIV.F32 S7, S2, S6
28      VADD.F32 S8, S7
29
30
31      ALIGN
32      END
```

Fuente: elaboración propia.

- Se necesitará además un contador. Puesto que no se tendrá la opción de While y For, se realizará con una comparación del Registro R1 con el valor de 100.

Figura 35. **Contador y BEQ**

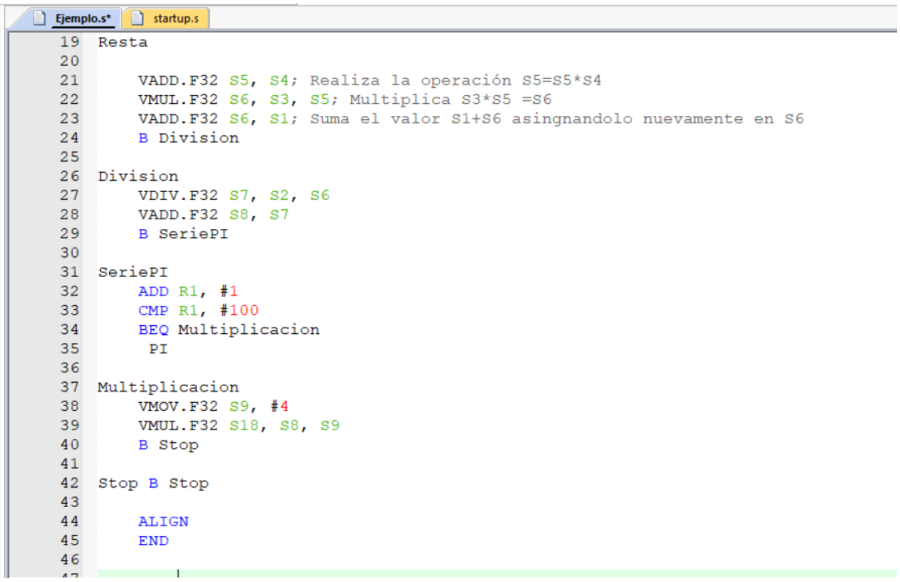
```
Ejemplo.s*  startup.s
17      B Resta; Realiza el Salto a Resta
18
19  Resta
20
21      VADD.F32 S5, S4; Realiza la operación S5=S5*S4
22      VMUL.F32 S6, S3, S5; Multiplica S3*S5 =S6
23      VADD.F32 S6, S1; Suma el valor S1+S6 asignandolo nuevamente en S6
24      B Division
25
26  Division
27      VDIV.F32 S7, S2, S6
28      VADD.F32 S8, S7
29      B SeriePI
30
31  SeriePI
32      ADD R1, #1
33      CMP R1, #100
34      BEQ Multiplicacion
35       PI
36
37  Multiplicacion
38      VMOV.F32 S9, #4
39      VMUL.F32 S10, S8, S9
40
41      ALIGN
42      END
43
44
```

Fuente: elaboración propia.

Cuando ambos valores sean iguales, realizará el salto a la etiqueta Multiplicación; mientras no sean iguales regresará a la subrutina PI.

- Por último, se debe escribir una etiqueta extra como un ciclo *loop* para que el programa realice el último salto sin ningún problema.

Figura 36. **Ciclo infinito**



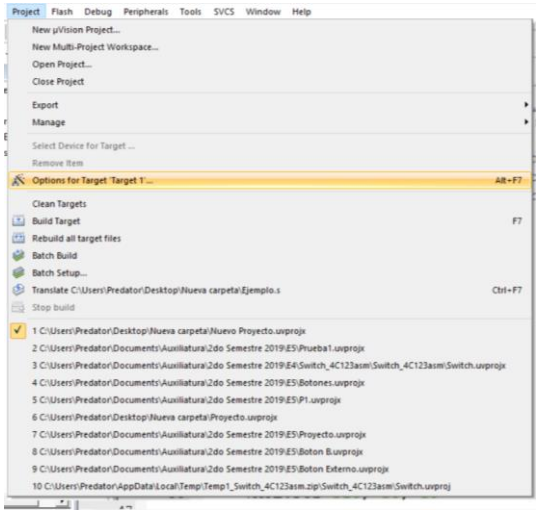
Fuente: elaboración propia.

- Por último, deberá comprobarse que el programa no tenga ningún error de sintaxis. Para esto se deberá construir, presionando F7 o el ícono de Build.

Para comprobar que el programa realice lo requerido se deberá realizar la siguiente configuración:

- En la viñeta Project, se debe acceder a la opción Option for Target

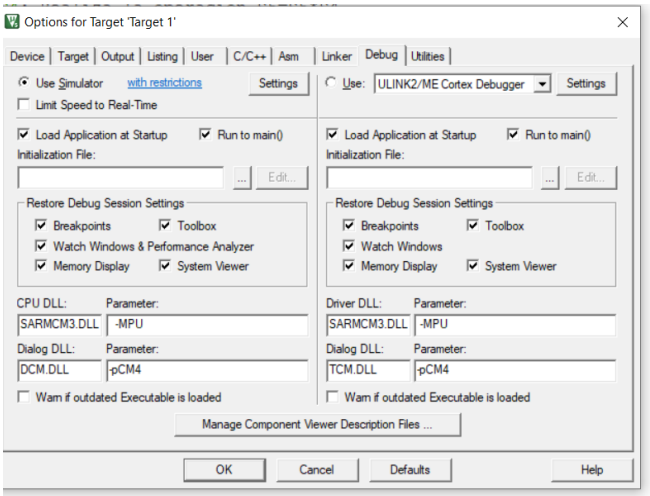
Figura 37. **Simulador**



Fuente: elaboración propia.

- En la viñeta Debug se deberá seleccionar la opción Use Simulator

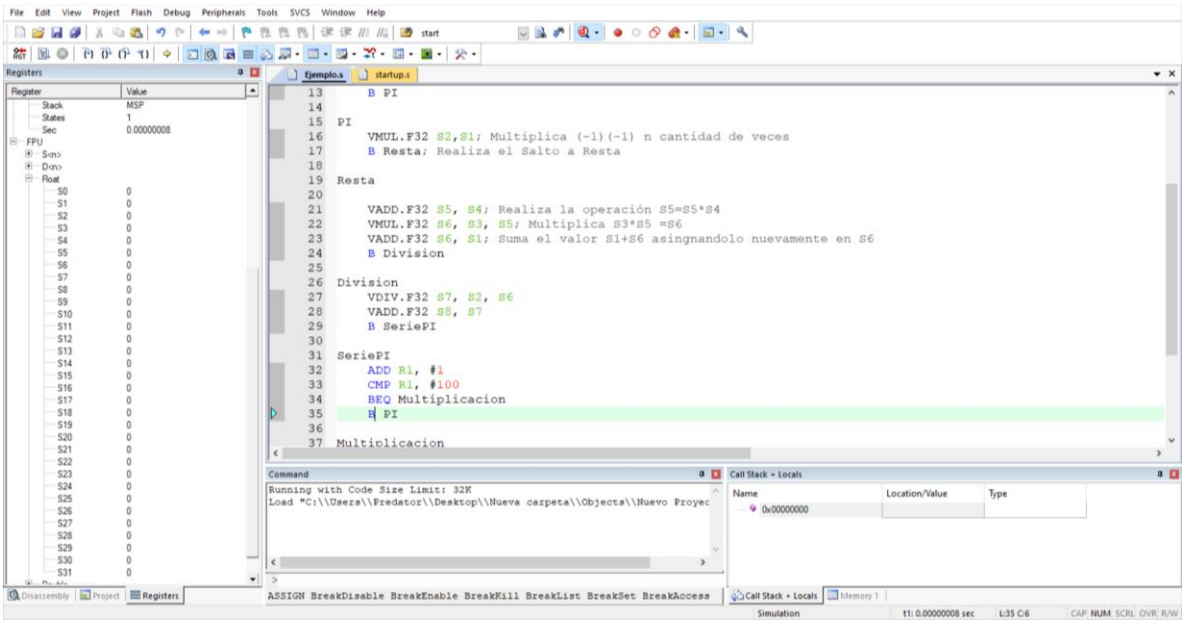
Figura 38. **Option for Target**



Fuente: elaboración propia.

- Se deberá usar el conjunto de teclas Ctrl+ F5 o seleccionar la opción de Start/Stop Debug Sesion.

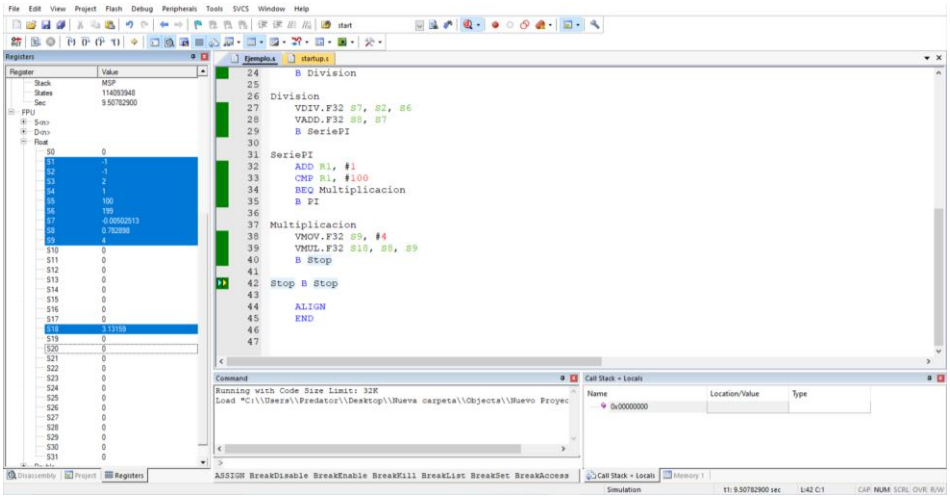
Figura 39. **Debug**



Fuente: elaboración propia.

La pantalla cambiará de vista y se podrán observar, del lado izquierdo de la pantalla, los registros de números enteros y puntos flotantes. Para realizar paso por paso la operación se deberá presionar F11 y se podrá observar el paso a paso de cada línea de código, o bien presionar F5 para correr todo el programa completo hasta la línea de Stop.

Figura 40. **Programa terminado**



Fuente: elaboración propia.

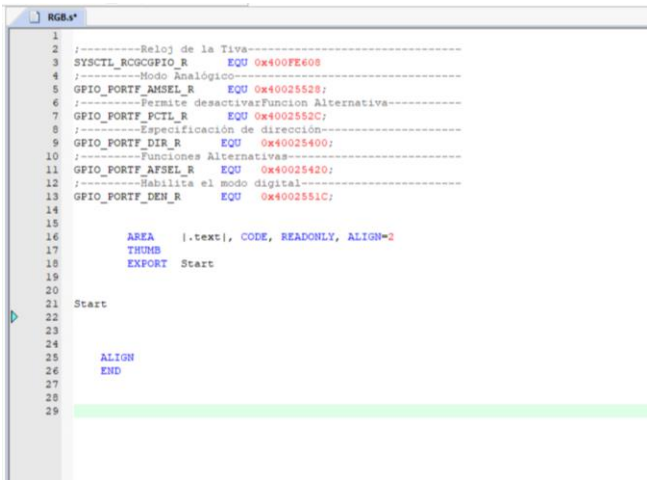
Aparecerá en color azul resaltado los registros que se utilizaron y en verde las líneas de código que ya hubiesen sido recorridas. Se podrá observar el valor final de la aproximación en el registro de punto flotante S18, el cual será de 3.13159 para 100 iteraciones.

5.4. Habilitación de leds RGB dedicados, puertos PF1, PF2 y PF3

Con la idea clara del funcionamiento del IDE, se realizará la primera práctica orientada al microcontrolador. Se siguen los pasos de la creación de un nuevo proyecto, un nuevo archivo y las directivas necesarias para cualquier proyecto. Se deberá recurrir a la sección 4.3 para habilitar el puerto F y sus pines necesarios.

- Como primer paso se debe asignar valores a la constante del reloj y las configuraciones mencionadas en la sección 4.3.3.

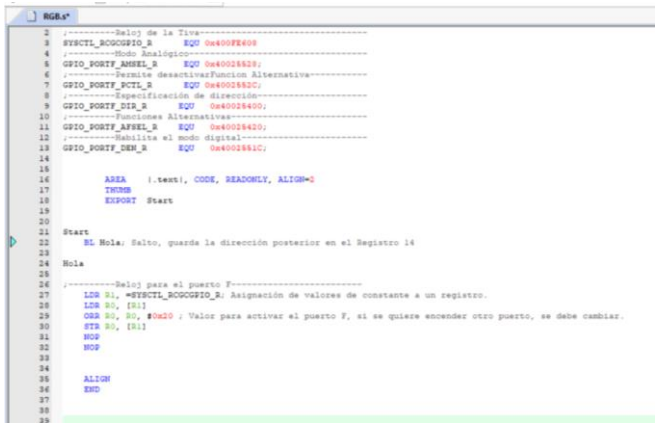
Figura 41. **Asignación de constantes para el puerto F**



Fuente: elaboración propia.

- Luego de inicializar los valores constantes, se debe habilitar el reloj para el o los puertos utilizados. En este caso únicamente se iniciará el reloj del puerto F.

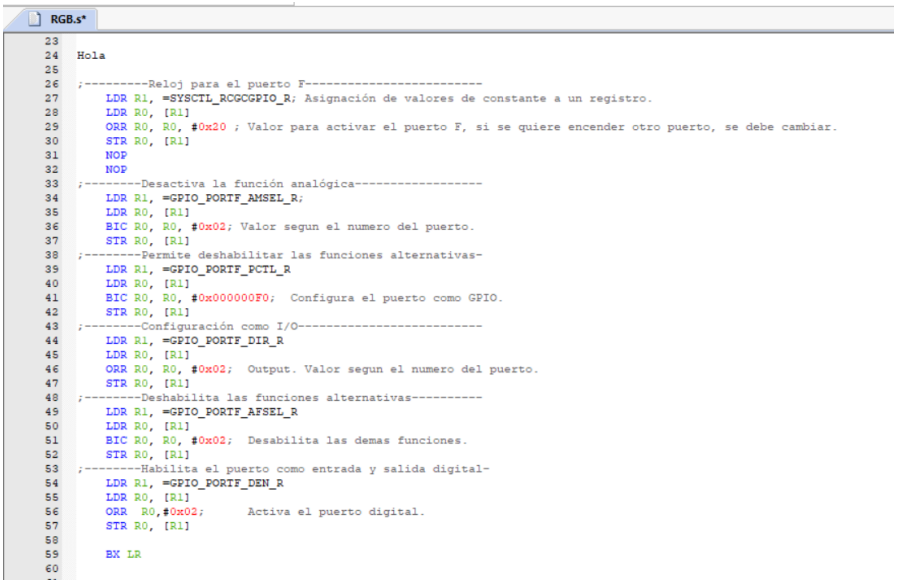
Figura 42. **Habilitar el reloj**



Fuente: elaboración propia.

- Se deberá habilitar y deshabilitar respectivamente cada una de las configuraciones de la sección 4.3.3.

Figura 43. **Configuraciones para el puerto F**

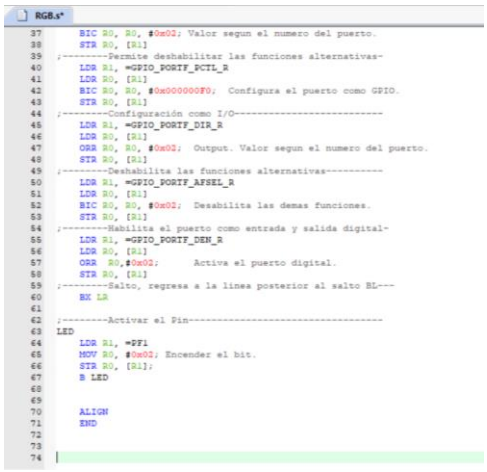


Fuente: elaboración propia.

Se puede observar la utilización del comando de salto BL. Este realizará el salto a la etiqueta Hola, y luego con el comando BX LR regresará una línea después de haber realizado el salto.

- Por último, se activará el pin PF1 con los valores de la tabla VIII.

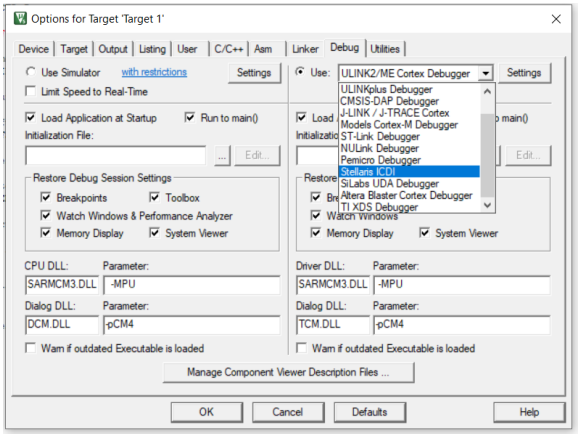
Figura 44. **Activar el pin**



Fuente: elaboración propia.

Al no ser una simulación sino la configuración del microcontrolador se deberá seleccionar en Options for Target en la opción de Debug, Use Stellaris ICDI.

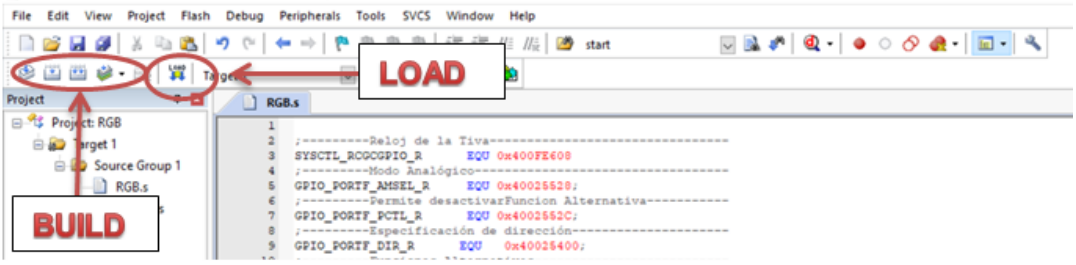
Figura 45. **Stellaris ICDI**



Fuente: elaboración propia.

- Con esto se podrá construir el programa con el botón Build. Una vez no haya errores se podrá cargar el programa al microcontrolador con Load.

Figura 46. **BUILD y LOAD**



Fuente: elaboración propia.

Figura 47. **Código completo pin F1**

```
RGB.s
;---------Reloj de la Tiva----------
SYSCTL_RCGCGPIO_R       EQU 0x400FE608
;---------Modo Analógico----------
GPIO_PORTF_AMSEL_R      EQU 0x40025528;
;---------Permite desactivarFuncion Alternativa----------
GPIO_PORTF_PCTL_R       EQU 0x4002552C;
;---------Especificación de dirección----------
GPIO_PORTF_DIR_R        EQU   0x40025400;
;---------Funciones Alternativas----------
GPIO_PORTF_AFSEL_R      EQU   0x40025420;
;---------Habilita el modo digital----------
GPIO_PORTF_DEN_R        EQU   0x4002551C;
;---------PIN PF1----------
PF1                     EQU 0x40025008

        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Start


Start
    BL Hola; Salto, guarda la dirección posterior en el Registro 14
    B  LED ; Salto

Hola

;---------Reloj para el puerto F----------
    LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
    LDR R0, [R1]
    ORR R0, R0, #0x20 ; Valor para activar el puerto F, si se quiere encender otro puerto, se debe cambiar.
    STR R0, [R1]
    NOP
```

Continuación de la figura 47.

```
34      NOP
35  ;--------Desactiva la función analógica------------------
36      LDR R1, =GPIO_PORTF_AMSEL_R;
37      LDR R0, [R1]
38      BIC R0, R0, #0x02; Valor segun el numero del puerto.
39      STR R0, [R1]
40  ;--------Permite deshabilitar las funciones alternativas-
41      LDR R1, =GPIO_PORTF_PCTL_R
42      LDR R0, [R1]
43      BIC R0, R0, #0x000000F0;  Configura el puerto como GPIO.
44      STR R0, [R1]
45  ;--------Configuración como I/O--------------------------
46      LDR R1, =GPIO_PORTF_DIR_R
47      LDR R0, [R1]
48      ORR R0, R0, #0x02;  Output. Valor segun el numero del puerto.
49      STR R0, [R1]
50  ;--------Deshabilita las funciones alternativas----------
51      LDR R1, =GPIO_PORTF_AFSEL_R
52      LDR R0, [R1]
53      BIC R0, R0, #0x02;  Desabilita las demas funciones.
54      STR R0, [R1]
55  ;--------Habilita el puerto como entrada y salida digital-
56      LDR R1, =GPIO_PORTF_DEN_R
57      LDR R0, [R1]
58      ORR  R0,#0x02;      Activa el puerto digital.
59      STR R0, [R1]
60  ;--------Salto, regresa a la linea posterior al salto BL---
61      BX LR
62
63  ;--------Activar el Pin----------------------------------
64  LED
65      LDR R1, =PF1
66      MOV R0, #0x02; Encender el bit.
67      STR R0, [R1];
68      B LED
69
70
71      ALIGN
72      END
73
74
75
```

Fuente: elaboración propia.

Figura 48. **Pin F1 activo**

Fuente: elaboración propia.

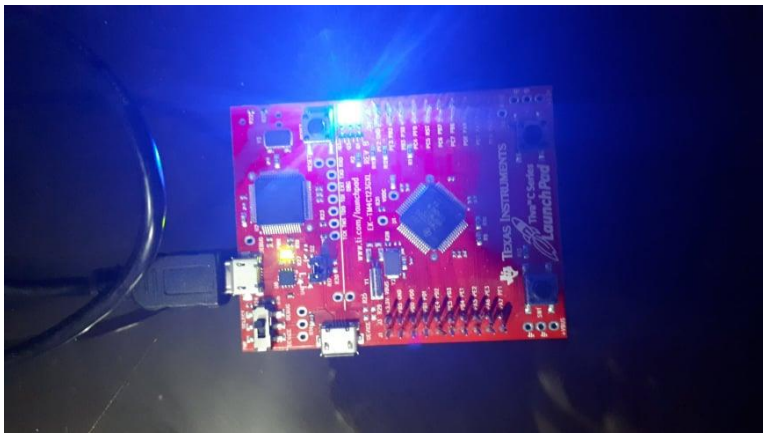
Figura 49. **Código completo pin F2**

RGB.s

```
1
2   ;---------Reloj de la Tiva--------------------------------
3   SYSCTL_RCGCGPIO_R      EQU 0x400FE608
4   ;---------Modo Analógico--------------------------------
5   GPIO_PORTF_AMSEL_R     EQU 0x40025528;
6   ;---------Permite desactivarFuncion Alternativa----------
7   GPIO_PORTF_PCTL_R      EQU 0x4002552C;
8   ;---------Especificación de dirección--------------------
9   GPIO_PORTF_DIR_R       EQU   0x40025400;
10  ;---------Funciones Alternativas-------------------------
11  GPIO_PORTF_AFSEL_R     EQU   0x40025420;
12  ;---------Habilita el modo digital-----------------------
13  GPIO_PORTF_DEN_R       EQU   0x4002551C;
14  ;---------PIN PF1----------------------------------------
15  PF2                    EQU 0x40025010
16
17          AREA    |.text|, CODE, READONLY, ALIGN=2
18          THUMB
19          EXPORT  Start
20
21
22  Start
23      BL Hola; Salto, guarda la dirección posterior en el Registro 14
24      B  LED ; Salto
25
26  Hola
27
28  ;---------Reloj para el puerto F-------------------------
29      LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
30      LDR R0, [R1]
31      ORR R0, R0, #0x20 ; Valor para activar el puerto F, si se quiere encender otro puerto, se debe cambiar.
32      STR R0, [R1]
33      NOP
34      NOP
35  ;--------Desactiva la función analógica-----------------
36      LDR R1, =GPIO_PORTF_AMSEL_R;
37      LDR R0, [R1]
38      BIC R0, R0, #0x04; Valor segun el numero del puerto.
39      STR R0, [R1]
40  ;--------Permite deshabilitar las funciones alternativas-
41      LDR R1, =GPIO_PORTF_PCTL_R
42      LDR R0, [R1]
43      BIC R0, R0, #0x00000F00;  Configura el puerto como GPIO.
44      STR R0, [R1]
45  ;--------Configuración como I/O-------------------------
46      LDR R1, =GPIO_PORTF_DIR_R
47      LDR R0, [R1]
48      ORR R0, R0, #0x04;  Output. Valor segun el numero del puerto.
49      STR R0, [R1]
50  ;--------Deshabilita las funciones alternativas----------
51      LDR R1, =GPIO_PORTF_AFSEL_R
52      LDR R0, [R1]
53      BIC R0, R0, #0x04;  Desabilita las demas funciones.
54      STR R0, [R1]
55  ;--------Habilita el puerto como entrada y salida digital-
56      LDR R1, =GPIO_PORTF_DEN_R
57      LDR R0, [R1]
58      ORR  R0,#0x04;      Activa el puerto digital.
59      STR R0, [R1]
60  ;--------Salto, regresa a la linea posterior al salto BL---
61      BX LR
62
63  ;--------Activar el Pin---------------------------------
64  LED
65      LDR R1, =PF2
66      MOV R0, #0x04; Encender el bit.
67      STR R0, [R1];
68      B LED
69
70
71      ALIGN
72      END
```

Fuente: elaboración propia.

Figura 50. **Pin F2 activo**



Fuente: elaboración propia.

Figura 51. **Configuración pines F1, F2 y F3 activos**

RGB.s

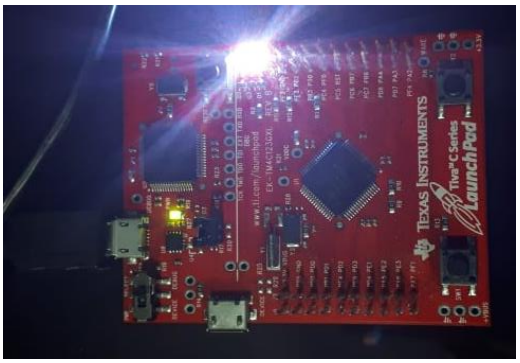
```
1
2  ;---------Reloj de la Tiva--------------------------------
3  SYSCTL_RCGCGPIO_R      EQU 0x400FE608
4  ;---------Modo Analógico----------------------------------
5  GPIO_PORTF_AMSEL_R     EQU 0x40025528;
6  ;---------Permite desactivarFuncion Alternativa-----------
7  GPIO_PORTF_PCTL_R      EQU 0x4002552C;
8  ;---------Especificación de dirección---------------------
9  GPIO_PORTF_DIR_R       EQU   0x40025400;
10 ;---------Funciones Alternativas--------------------------
11 GPIO_PORTF_AFSEL_R     EQU   0x40025420;
12 ;---------Habilita el modo digital------------------------
13 GPIO_PORTF_DEN_R       EQU   0x4002551C;
14 ;---------PIN PF1-----------------------------------------
15 PF123                  EQU 0x40025038; Esta suma se realiza en hexadecimal
16 PCTL                   EQU 0x00000FFF; Valor agregado, mayores a 255.
17
18         AREA    |.text|, CODE, READONLY, ALIGN=2
19         THUMB
20         EXPORT  Start
21
22
23 Start
24     BL Hola; Salto, guarda la dirección posterior en el Registro 14
25     B  LED ; Salto
26
27 Hola
28
29 ;---------Reloj para el puerto F-------------------------
30     LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
31     LDR R0, [R1]
32     ORR R0, R0, #0x20 ; Valor para activar el puerto F, si se quiere encender otro puerto, se debe cambiar.
33     STR R0, [R1]
```

Continuación de la figura 50.

```
34      NOP
35      NOP
36  ;--------Desactiva la función analógica------------------
37      LDR R1, =GPIO_PORTF_AMSEL_R;
38      LDR R0, [R1]
39      BIC R0, R0, #0x0E; Valor segun el numero del puerto. Se realiza la suma de los valores en hexadecimal.
40      STR R0, [R1]
41  ;--------Permite deshabilitar las funciones alternativas-
42      LDR R1, =GPIO_PORTF_PCTL_R
43      LDR R0, [R1]
44      LDR R2, =PCTL
45      LDR R3,[R2]
46      BIC R0, R0, R3;  Configura el puerto como GPIO.
47      STR R0, [R1]
48  ;--------Configuración como I/O--------------------------
49      LDR R1, =GPIO_PORTF_DIR_R
50      LDR R0, [R1]
51      ORR R0, R0, #0x0E;  Output. Valor segun el numero del puerto.
52      STR R0, [R1]
53  ;--------Deshabilita las funciones alternativas----------
54      LDR R1, =GPIO_PORTF_AFSEL_R
55      LDR R0, [R1]
56      BIC R0, R0, #0x0E;  Desabilita las demas funciones.
57      STR R0, [R1]
58  ;--------Habilita el puerto como entrada y salida digital-
59      LDR R1, =GPIO_PORTF_DEN_R
60      LDR R0, [R1]
61      ORR  R0,#0x0E;      Activa el puerto digital.
62      STR R0, [R1]
63  ;--------Salto, regresa a la linea posterior al salto BL---
64      BX LR
65
66  ;--------Activar el Pin---------------------------------
67  LED
68      LDR R1, =PF123
69      MOV R0, #0x0E; Encender el bit.
70      STR R0, [R1];
71      B LED
72
73
74      ALIGN
75      END
```

Fuente: elaboración propia.

Figura 52. **Pines F1, F2 y F3 activos simultáneamente**



Fuente: elaboración propia.

5.5. Secuencia de leds con contador

Una vez esté configurada la práctica anterior para los pines F1, F2 y F3, se puede realizar con mayor facilidad una secuencia de leds. El orden será el siguiente:

Rojo, azul, verde, morado, turquesa, amarillo, blanco, repitiéndose constantemente. No se necesita configurar una constante por color, únicamente se debe saber el valor de activación de cada pin.

Tabla XI. **Tabla de colores**

Color	Pines involucrados	Valor de activación de los pines
Rojo	PF1	0x02
Azul	PF2	0x04
Verde	PF3	0x08
Morado	PF1 y PF2	0x06
Turquesa	PF2 y PF3	0x0C
Amarillo	PF1 y PF3	0x0A
Blanco	PF1, PF2, PF3	0x0E

Fuente: elaboración propia

Partiendo de la figura anterior, se crearán 7 subrutinas, una por cada color. Además, se creará un contador, el cual disminuirá hasta llegar a 0, lo cual significará un cambio de color.

Figura 53. **Programación completa**

RGB.s*

```
;---------Reloj de la Tiva--------------------------------
SYSCTL_RCGCGPIO_R      EQU 0x400FE608
;---------Modo Analógico----------------------------------
GPIO_PORTF_AMSEL_R     EQU 0x40025528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTF_PCTL_R      EQU 0x4002552C;
;---------Especificación de dirección---------------------
GPIO_PORTF_DIR_R       EQU   0x40025400;
;---------Funciones Alternativas--------------------------
GPIO_PORTF_AFSEL_R     EQU   0x40025420;
;---------Habilita el modo digital------------------------
GPIO_PORTF_DEN_R       EQU   0x4002551C;
;---------PIN PF1-----------------------------------------
PF123                  EQU 0x40025038; Esta suma se realiza en hexadecimal
PCTL                   EQU 0x00000FFF; Valor agregado, mayores a 255.
Cont                   EQU 1000000

        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Start


Start
    BL Hola; Salto, guarda la dirección posterior en el Registro 14
    B  Rojo ; Salto

Hola

;---------Reloj para el puerto F-------------------------
    LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
    LDR R0, [R1]
    ORR R0, R0, #0x20 ; Valor para activar el puerto F, si se quiere encender otro puerto, se debe cambiar.
    STR R0, [R1]
    NOP
    NOP
;--------Desactiva la función analógica------------------
    LDR R1, =GPIO_PORTF_AMSEL_R;
    LDR R0, [R1]
    BIC R0, R0, #0x0E; Valor segun el numero del puerto. Se realiza la suma de los valores en hexadecimal.
    STR R0, [R1]
;--------Permite deshabilitar las funciones alternativas-
    LDR R1, =GPIO_PORTF_PCTL_R
    LDR R0, [R1]
    LDR R2, =PCTL
    LDR R3,[R2]
    BIC R0, R0, R3;  Configura el puerto como GPIO.
    STR R0, [R1]
;--------Configuración como I/O--------------------------
    LDR R1, =GPIO_PORTF_DIR_R
    LDR R0, [R1]
    ORR R0, R0, #0x0E;  Output. Valor segun el numero del puerto.
    STR R0, [R1]
;--------Deshabilita las funciones alternativas----------
    LDR R1, =GPIO_PORTF_AFSEL_R
    LDR R0, [R1]
    BIC R0, R0, #0x0E;  Desabilita las demas funciones.
    STR R0, [R1]
;--------Habilita el puerto como entrada y salida digital-
    LDR R1, =GPIO_PORTF_DEN_R
    LDR R0, [R1]
    ORR  R0,#0x0E;      Activa el puerto digital.
    STR R0, [R1]
;--------Salto, regresa a la linea posterior al salto BL---
    BX LR
```

Fuente: elaboración propia.

Figura 54. **Continuación programación completa**

```
66
67  ;--------Contador de iteraciones--------------------------
68
69  Contador
70      SUB R10, #1 ; Resta
71      CMP R10, #0 ; Comparación
72      BNE Contador ; Condición No igual
73      BX LR
74
75
76  ;--------Activar PF1 ---------------------------------
77  Rojo
78      LDR R1, =PF123
79      MOV R0, #0x02; Encender el bit.
80      STR R0, [R1];
81      LDR R10, =Cont
82      BL Contador
83  ;--------Activar PF2 ---------------------------------
84  Azul
85      LDR R1, =PF123
86      MOV R0, #0x04; Encender el bit.
87      STR R0, [R1];
88      LDR R10, =Cont
89      BL Contador
90  ;--------Activar PF3 ---------------------------------
91  Verde
92      LDR R1, =PF123
93      MOV R0, #0x08; Encender el bit.
94      STR R0, [R1];
95      LDR R10, =Cont
96      BL Contador
97  ;--------Activar PF1 y PF2 ---------------------------------
98  Morado
99      LDR R1, =PF123
100     MOV R0, #0x06; Encender el bit.
101     STR R0, [R1];
102     LDR R10, =Cont
103     BL Contador
104 ;--------Activar PF2 y PF3 ---------------------------------
105 Turquesa
106     LDR R1, =PF123
107     MOV R0, #0x0C; Encender el bit.
108     STR R0, [R1];
109     LDR R10, =Cont
110     BL Contador
111 ;--------Activar PF1 y PF3 ---------------------------------
112 Amarillo
113     LDR R1, =PF123
114     MOV R0, #0x0A; Encender el bit.
115     STR R0, [R1];
116     LDR R10, =Cont
117     BL Contador
118 ;--------Activar PF1, PF2 y PF3---------------------------------
119 Blanco
120     LDR R1, =PF123
121     MOV R0, #0x0E; Encender el bit.
122     STR R0, [R1];
123     LDR R10, =Cont
124     BL Contador
125     B Rojo
126
127     ALIGN
128     END
129
```

Fuente: elaboración propia.

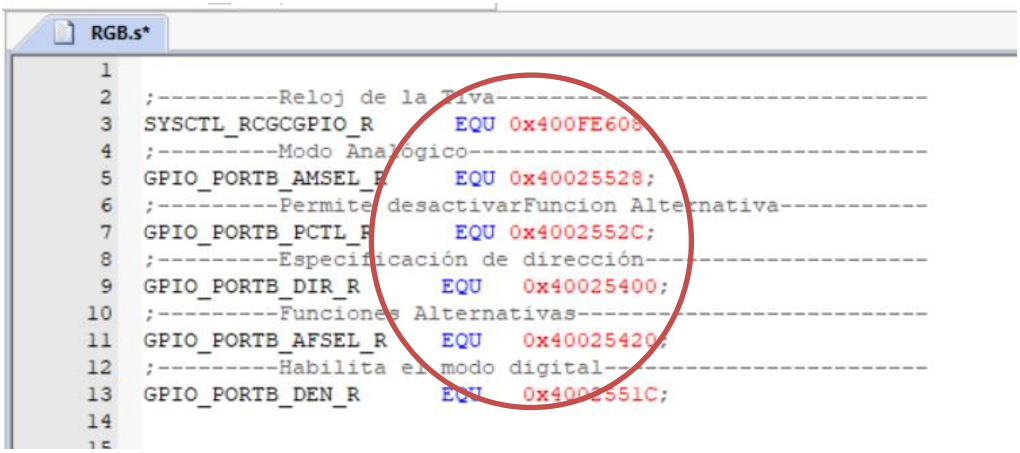
Se puede observar las nuevas etiquetas creadas que representan cada color. Además, la constante creada con el nombre Cont que se inicializa al finalizar cada etiqueta para volver a iniciar de un millón.

5.6. Habilitar como salida el puerto B completo

Una vez habilitado el puerto F, la habilitación del puerto B será mucho más sencilla. Esta se realizará con el fin de aclarar dudas del estudiante al momento de habilitar un puerto que no sea el F. Para que pueda observar las modificaciones correspondientes, se trabajará sobre la práctica 5.

- Como primer punto se modificarán los nombres de las constantes, únicamente para que su comprensión sea más sencilla. Notar que la constante del reloj se mantendrá.

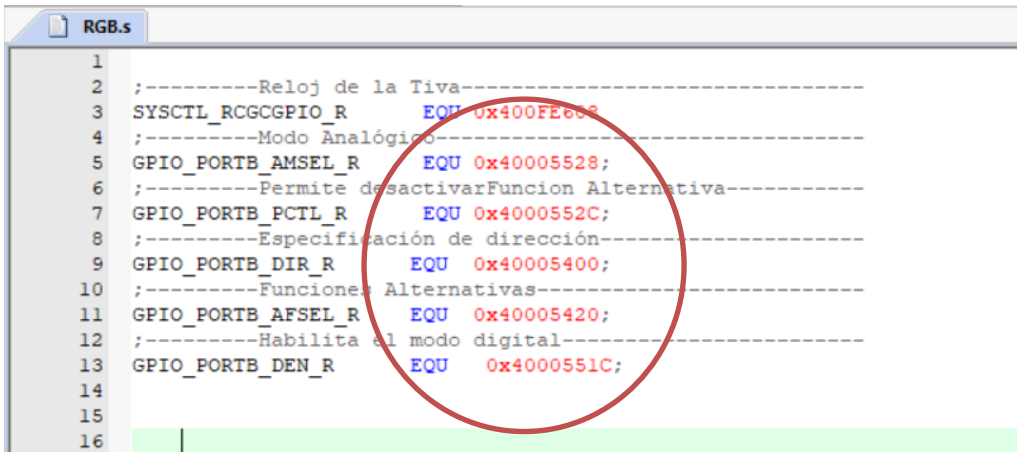
Figura 55. **Cambio de nombre de las constantes**



Fuente: elaboración propia.

- Luego se deberá cambiar los valores de las constantes. Tomando como referencia la tabla III se deberá modificar cada valor para el puerto B.

Figura 56. **Cambio del valor de las constantes**



Fuente: elaboración propia.

- Se debe modificar la constante para el puerto B. Como se utilizará todo el puerto se nombrará PB y el valor debe ser modificado con los valores de la tabla VI. Esta suma se realiza en hexadecimal.

Figura 57. **Constante puerto B**

```
;---------Puerto B----------
PB              EQU 0x400053FC; Esta suma se realiza en hexadecimal.
```

Fuente: elaboración propia.

- Como se utilizarán todos los pines del puerto, a la constante PCTL de la práctica anterior se le asignará el valor de 2_11111111.

Figura 58. **Constante PCTL**

```
22  ;---------Constante PCTL----------------------------------
23  PCTL                EQU 2_11111111; Como se utilizarán todos los pines.
24
25
```

Fuente: elaboración propia.

- Se debe configurar el reloj para el puerto B. Este será de 0x02 según la tabla V.

Figura 59. **Reloj para el puerto B**

```
RGB.s*
27  Cont                EQU 1000000; Valor del contrador
28
29          AREA    |.text|, CODE, READONLY, ALIGN=2
30          THUMB
31          EXPORT  Start
32
33
34  Start
35      BL Conf; Salto, guarda la dirección posterior en el Registro 14
36      B  PuertoB ; Salto
37
38
39  Conf
40
41  ;---------Reloj para el puerto B-------------------------
42      LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
43      LDR R0, [R1]
44      ORR R0, R0, #0x02 ; Valor para activar el reloj del puerto B.
45      STR R0, [R1]
46      NOP
47      NOP
48
49
```

Fuente: elaboración propia.

- Luego de esto se debe realizar la configuración del puerto. Como se utilizarán todos los puertos se suman los valores de la tabla VII, el cual será de 0xFF.

Figura 60. **Configuración puerto B**

RGB.s*

```
48
49
50
51  ;--------Desactiva la función analógica------------------
52      LDR R1, =GPIO_PORTB_AMSEL_R;
53      LDR R0, [R1]
54      BIC R0, R0, #0xFF; Valor segun el numero del puerto. Se realiza la suma de los valores en hexadecimal.
55      STR R0, [R1]
56  ;--------Permite deshabilitar las funciones alternativas-
57      LDR R1, =GPIO_PORTB_PCTL_R
58      LDR R0, [R1]
59      LDR R2, =PCTL
60      LDR R3,[R2]
61      BIC R0, R0, R3;  Configura el puerto como GPIO.
62      STR R0, [R1]
63  ;--------Configuración como I/O---------------------------
64      LDR R1, =GPIO_PORTB_DIR_R
65      LDR R0, [R1]
66      ORR R0, R0, #0xFF;  Output. Valor segun el numero del puerto.
67      STR R0, [R1]
68  ;--------Deshabilita las funciones alternativas----------
69      LDR R1, =GPIO_PORTB_AFSEL_R
70      LDR R0, [R1]
71      BIC R0, R0, #0xFF;  Desabilita las demas funciones.
72      STR R0, [R1]
73  ;--------Habilita el puerto como entrada y salida digital-
74      LDR R1, =GPIO_PORTB_DEN_R
75      LDR R0, [R1]
76      ORR  R0,#0xFF;      Activa el puerto digital.
77      STR R0, [R1]
78  ;--------Salto, regresa a la linea posterior al salto BL---
79      BX LR
80
```

Fuente: elaboración propia

- Posteriormente ya se podrán activar todos los pines del puerto B. Para esto, con el mismo contador utilizado en la práctica anterior se realizará un bucle que hará intermitentes todos los pines del puerto B.

Figura 61. **Bucle intermitente**

RGB.s

```
79      BX LR
80
81
82
83  ;--------Contador de iteraciones---------------------------
84
85  Contador
86      SUB R10, #1 ; Resta
87      CMP R10, #0 ; Comparación
88      BNE Contador ; Condición No igual
89      BX LR
90
91
92  ;--------Activar PB1 ---------------------------------
93  PuertoBActivo
94      LDR R1, =PB
95      MOV R0, #0xFF; Encender el bit.
96      STR R0, [R1];
97      LDR R10, =Cont
98      BL Contador
99
100 PuertoBInactivo
101     LDR R1, =PB
102     MOV R0, #0xFF; Encender el bit.
103     STR R0, [R1];
104     LDR R10, =Cont
105     BL Contador
106
107     ALIGN
108     END
109
110
111
```

Fuente: elaboración propia.

Figura 62. **Código práctica 6 parte 1**

RGB.s

```
;---------Reloj de la Tiva--------------------------------
SYSCTL_RCGCGPIO_R      EQU 0x400FE608
;---------Modo Analógico----------------------------------
GPIO_PORTB_AMSEL_R     EQU 0x40005528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTB_PCTL_R      EQU 0x4000552C;
;---------Especificación de dirección---------------------
GPIO_PORTB_DIR_R       EQU  0x40005400;
;---------Funciones Alternativas--------------------------
GPIO_PORTB_AFSEL_R     EQU  0x40005420;
;---------Habilita el modo digital------------------------
GPIO_PORTB_DEN_R       EQU   0x4000551C;



;---------Puerto B----------------------------------------
PB                EQU 0x400053FC; Esta suma se realiza en hexadecimal.



;---------Constante PCTL----------------------------------
PCTL                   EQU 2_11111111; Como se utilizarán todos los pines.



Cont                   EQU 1000000; Valor del contrador

        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Start


Start
    BL Conf; Salto, guarda la dirección posterior en el Registro 14
    B  PuertoBActivo ; Salto


Conf

;---------Reloj para el puerto B------------------------
    LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
    LDR R0, [R1]
    ORR R0, R0, #0x02 ; Valor para activar el reloj del puerto B.
    STR R0, [R1]
    NOP
    NOP



;--------Desactiva la función analógica-----------------
    LDR R1, =GPIO_PORTB_AMSEL_R;
    LDR R0, [R1]
    BIC R0, R0, #0xFF; Valor segun el numero del puerto.
    STR R0, [R1]
;--------Permite deshabilitar las funciones alternativas-
    LDR R1, =GPIO_PORTB_PCTL_R
    LDR R0, [R1]
    LDR R2, =PCTL
    LDR R3,[R2]
    BIC R0, R0, R3;  Configura el puerto como GPIO.
    STR R0, [R1]
;--------Configuración como I/O-------------------------
    LDR R1, =GPIO_PORTB_DIR_R
    LDR R0, [R1]
```

Fuente: elaboración propia.

Figura 63. **Código práctica 6 parte2**

```
66      ORR R0, R0, #0xFF;  Output. Valor segun el numero del puerto.
67      STR R0, [R1]
68  ;--------Deshabilita las funciones alternativas----------
69      LDR R1, =GPIO_PORTB_AFSEL_R
70      LDR R0, [R1]
71      BIC R0, R0, #0xFF;  Desabilita las demas funciones.
72      STR R0, [R1]
73  ;--------Habilita el puerto como entrada y salida digital-
74      LDR R1, =GPIO_PORTB_DEN_R
75      LDR R0, [R1]
76      ORR  R0,#0xFF;      Activa el puerto digital.
77      STR R0, [R1]
78  ;--------Salto, regresa a la linea posterior al salto BL---
79      BX LR
80
81
82
83  ;--------Contador de iteraciones---------------------------
84
85  Contador
86      SUB R10, #1 ; Resta
87      CMP R10, #0 ; Comparación
88      BNE Contador ; Condición No igual
89      BX LR
90
91
92  ;--------Activar PB1 ---------------------------------
93  PuertoBActivo
94      LDR R1, =PB
95      MOV R0, #0xFF; Encender el bit.
96      STR R0, [R1];
97      LDR R10, =Cont
98      BL Contador
99
100 PuertoBInactivo
101     LDR R1, =PB
102     MOV R0, #0x00; Encender el bit.
103     STR R0, [R1];
104     LDR R10, =Cont
105     BL Contador
106     B PuertoBActivo
107
108     ALIGN
109     END
110
111
```

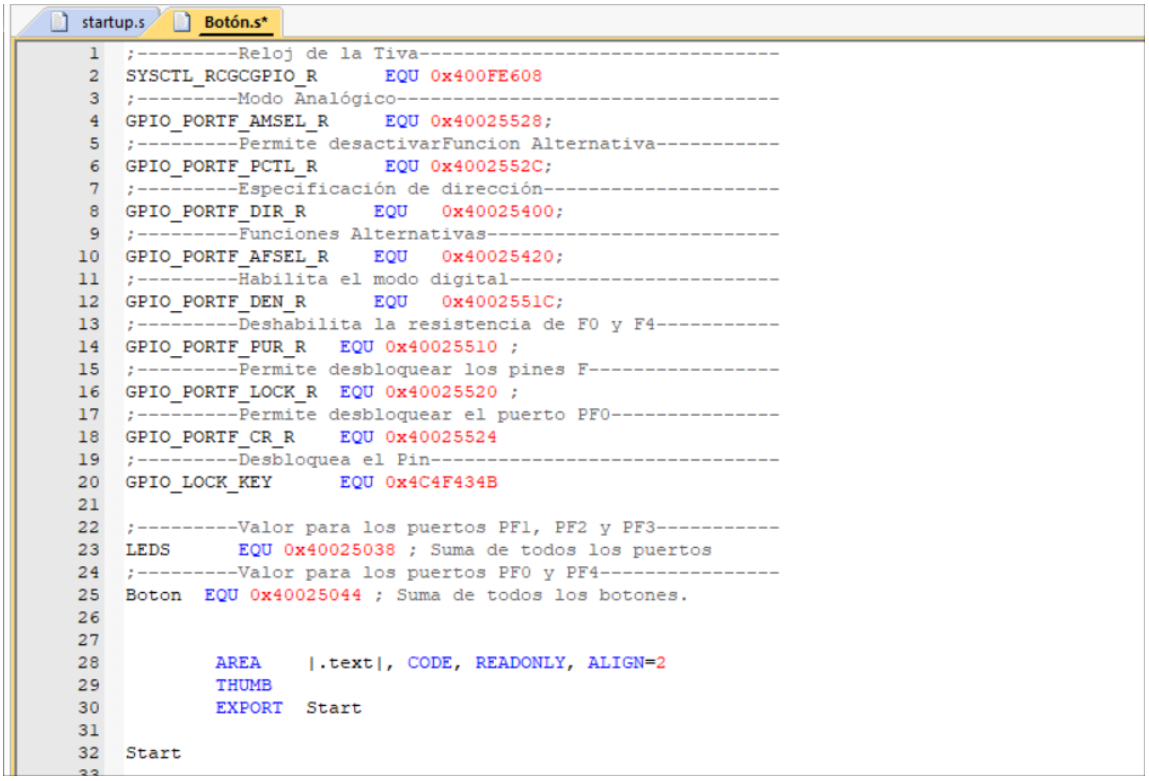
Fuente: elaboración propia.

5.7. Habilitar los puertos SW1- PF4 y SW2-PF0 como entradas

Una vez se puede modificar los puertos y pines como salida, se debe considerar que también es necesario utilizarlos como entrada. En la presente práctica se enseñará cómo habilitar los puertos PF0 y PF4, botones dedicados de la Tiva.

- Primeramente, se utilizarán todas las constantes de la sección 4.3.3, puesto que LOCK, PUR y CR únicamente se utilizarán para desbloquear los puertos PF0 y PF4. Además de las constantes para habilitar los pines, estarán dos constantes más, leds, el cual tendrá los valores de los puertos PF1-PF3 y botón que tendrá los valores de PF0 y PF4.

Figura 64. **Asignación de constantes para puertos de entrada y salida**



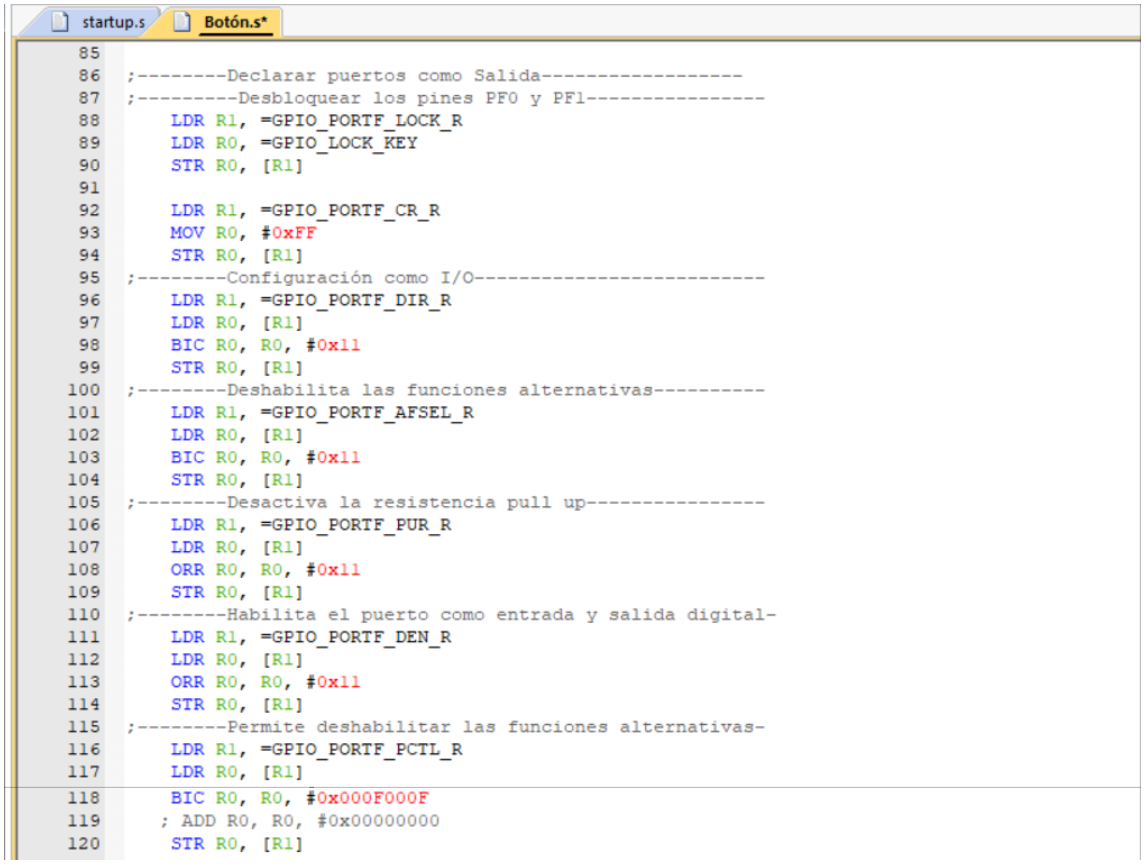
Fuente: elaboración propia.

- Se debe configurar el reloj para el puerto F. No es necesario repetir este paso al declararlo entrada y salida.

- Posteriormente se configurarán los puertos como entrada, como se puede observar en la figura 34, práctica 5.4.

- Ya habilitados los valores para PF1-PF3, se debe habilitar los valores para los switch integrados del microcontrolador. La configuración es similar a cuando son valores de entrada y salida, únicamente será diferente la dirección o DIR y que además se deben agregar los comando LOCK, PUR y CR.

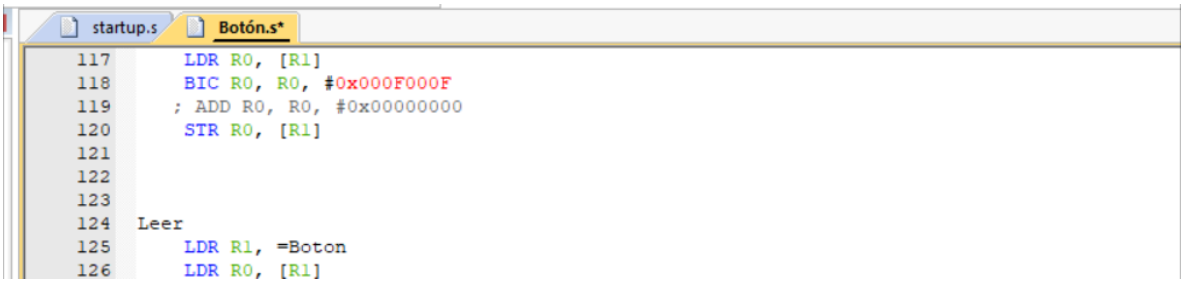
Figura 65. **Puertos PF0 y PF4 como salida**



Fuente: elaboración propia.

- Una vez habilitados los puertos, se deberá leer el estado de los switch constantemente. Con la constante Boton se le asignará el valor a un registr;, en este caso, el registro R1 tendrá la dirección de los pines y R0 tendrá el valor de los pines.

Figura 66. **Leer el estado de los botones PF0 y PF4**



Fuente: Elaboración propia

- Con el valor guardado en el registro R0 se podrá comparar con los valores de un pin activo; en este caso, siguiendo la tabla VII se puede observar que para el puerto PF0 será un valor de 0x01 y para el puerto PF4 será 0x10.

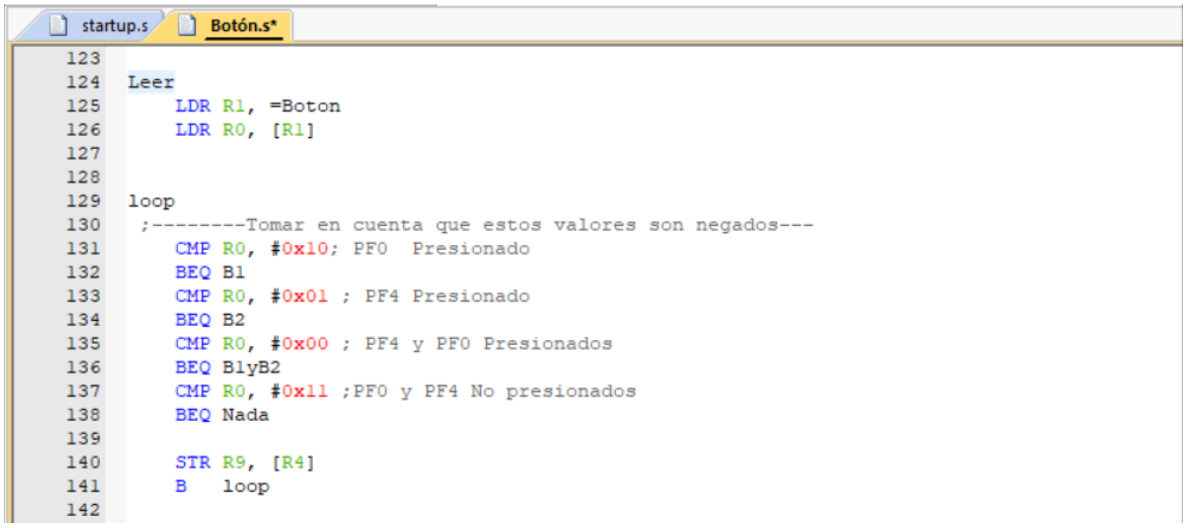
Tabla XII. **Combinación de botones**

Puerto de entrada	Valor	Color	Subrutina
PF0	0x10	Rojo	B1
PF1	0x01	Azul	B2
PF0 y PF1	0x00	Morado	B1yB2
Ninguno	0x11	Apagado	Nada

Fuente: elaboración propia.

En la tabla XI se podrá observar el funcionamiento del programa. Se deberá considerar que, por la resistencia activa, estos valores estarán negados; además, se debe tomar en cuenta que el puerto PF0 será el switch 2 del microcontrolador y el PF4 el switch 1.

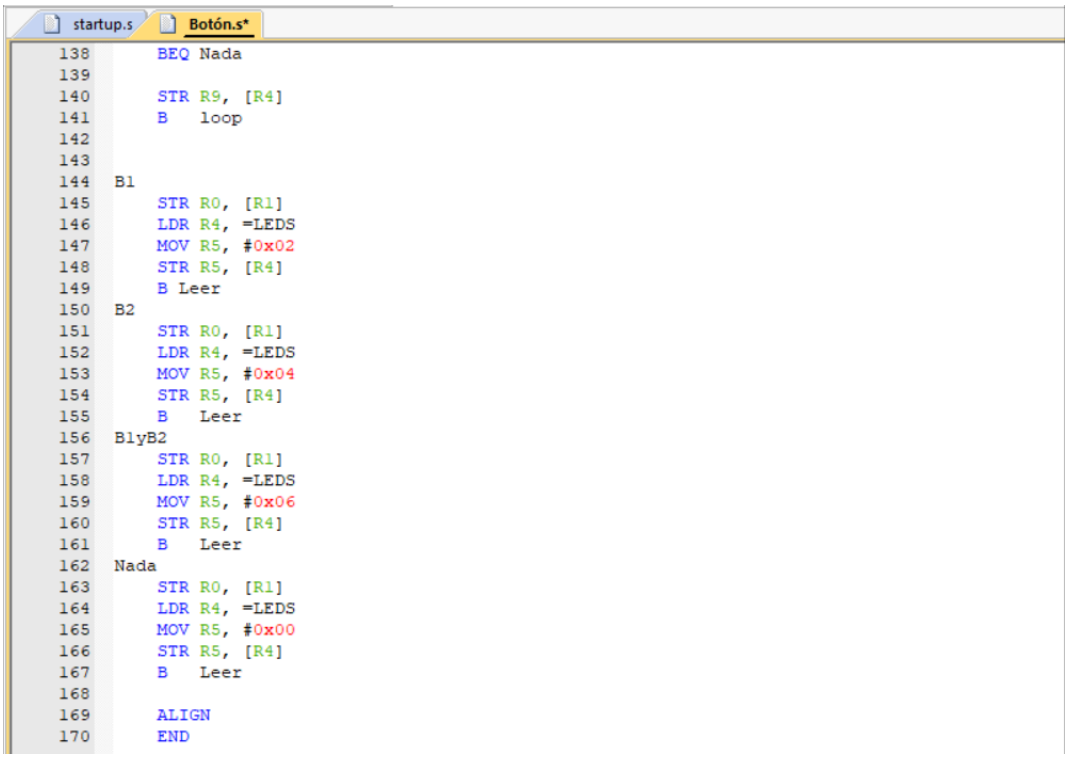
Figura 67. **Comparar el estado de los puertos PF0 y PF4**



Fuente: elaboración propia.

- Cada una de estas comparaciones redirigirá a una subrutina, la cual será diferente según el botón presionado, como se puede observar en la tabla IX.

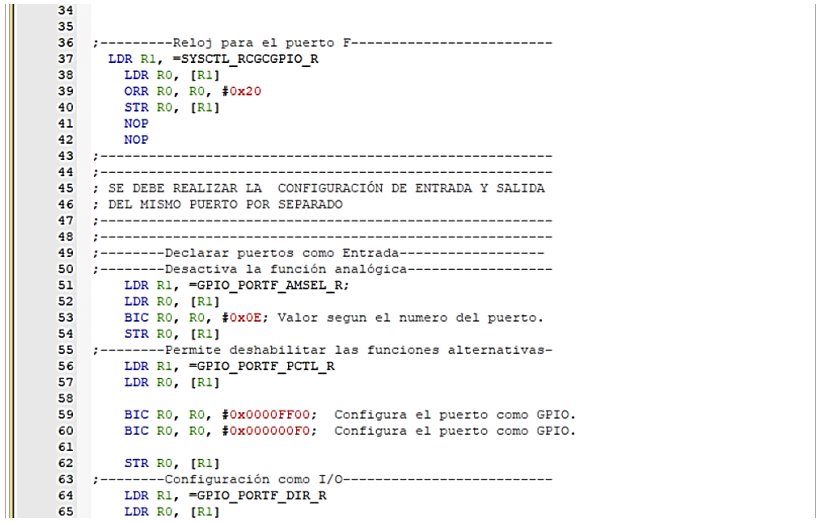
Figura 68. **Subrutinas que cumplen cada condición**



Fuente: elaboración propia.

Se debe observar que cada rutina termina con el salto a leer para que siempre esté leyendo el estado de los pines PF0 y PF4, por si llegara a cambiar el estado de estos.

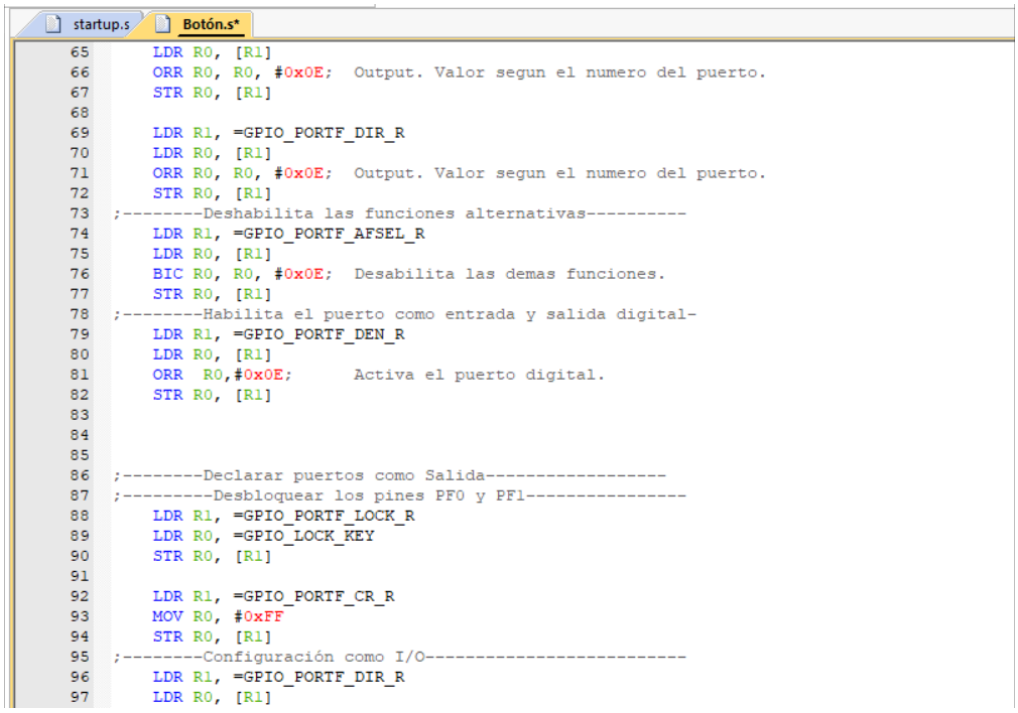
Figura 69. **Código completo práctica 5.7 parte 1**



```
34
35
36  ;---------Reloj para el puerto F-------------------------
37    LDR R1, =SYSCTL_RCGCGPIO_R
38      LDR R0, [R1]
39      ORR R0, R0, #0x20
40      STR R0, [R1]
41      NOP
42      NOP
43  ;-------------------------------------------------------
44  ;-------------------------------------------------------
45  ; SE DEBE REALIZAR LA  CONFIGURACIÓN DE ENTRADA Y SALIDA
46  ; DEL MISMO PUERTO POR SEPARADO
47  ;-------------------------------------------------------
48  ;-------------------------------------------------------
49  ;--------Declarar puertos como Entrada------------------
50  ;--------Desactiva la función analógica-----------------
51      LDR R1, =GPIO_PORTF_AMSEL_R;
52      LDR R0, [R1]
53      BIC R0, R0, #0x0E; Valor segun el numero del puerto.
54      STR R0, [R1]
55  ;--------Permite deshabilitar las funciones alternativas-
56      LDR R1, =GPIO_PORTF_PCTL_R
57      LDR R0, [R1]
58
59      BIC R0, R0, #0x0000FF00;  Configura el puerto como GPIO.
60      BIC R0, R0, #0x000000F0;  Configura el puerto como GPIO.
61
62      STR R0, [R1]
63  ;--------Configuración como I/O--------------------------
64      LDR R1, =GPIO_PORTF_DIR_R
65      LDR R0, [R1]
```

Fuente: elaboración propia.

Figura 70. **Código completo práctica 5.7 parte 2**



startup.s | Botón.s*

```
65      LDR R0, [R1]
66      ORR R0, R0, #0x0E;  Output. Valor segun el numero del puerto.
67      STR R0, [R1]
68
69      LDR R1, =GPIO_PORTF_DIR_R
70      LDR R0, [R1]
71      ORR R0, R0, #0x0E;  Output. Valor segun el numero del puerto.
72      STR R0, [R1]
73  ;--------Deshabilita las funciones alternativas----------
74      LDR R1, =GPIO_PORTF_AFSEL_R
75      LDR R0, [R1]
76      BIC R0, R0, #0x0E;  Desabilita las demas funciones.
77      STR R0, [R1]
78  ;--------Habilita el puerto como entrada y salida digital-
79      LDR R1, =GPIO_PORTF_DEN_R
80      LDR R0, [R1]
81      ORR  R0,#0x0E;      Activa el puerto digital.
82      STR R0, [R1]
83
84
85
86  ;--------Declarar puertos como Salida------------------
87  ;---------Desbloquear los pines PF0 y PF1----------------
88      LDR R1, =GPIO_PORTF_LOCK_R
89      LDR R0, =GPIO_LOCK_KEY
90      STR R0, [R1]
91
92      LDR R1, =GPIO_PORTF_CR_R
93      MOV R0, #0xFF
94      STR R0, [R1]
95  ;--------Configuración como I/O--------------------------
96      LDR R1, =GPIO_PORTF_DIR_R
97      LDR R0, [R1]
```

Continuación de la figura 69.

```
    BIC R0, R0, #0x11
    STR R0, [R1]
;--------Deshabilita las funciones alternativas----------
    LDR R1, =GPIO_PORTF_AFSEL_R
    LDR R0, [R1]
    BIC R0, R0, #0x11
    STR R0, [R1]
;--------Desactiva la resistencia pull up----------------
    LDR R1, =GPIO_PORTF_PUR_R
    LDR R0, [R1]
    ORR R0, R0, #0x11
    STR R0, [R1]
;--------Habilita el puerto como entrada y salida digital-
    LDR R1, =GPIO_PORTF_DEN_R
    LDR R0, [R1]
    ORR R0, R0, #0x11
    STR R0, [R1]
;--------Permite deshabilitar las funciones alternativas-
    LDR R1, =GPIO_PORTF_PCTL_R
    LDR R0, [R1]
    BIC R0, R0, #0x000F000F
   ; ADD R0, R0, #0x00000000
    STR R0, [R1]


Leer
    LDR R1, =Boton
    LDR R0, [R1]
```

Fuente: elaboración propia.

Figura 71. **Código completo práctica 5.7 parte 3**

```
loop
 ;--------Tomar en cuenta que estos valores son negados---
    CMP R0, #0x10; PF0  Presionado
    BEQ B1
    CMP R0, #0x01 ; PF4 Presionado
    BEQ B2
    CMP R0, #0x00 ; PF4 y PF0 Presionados
    BEQ B1yB2
    CMP R0, #0x11 ;PF0 y PF4 No presionados
    BEQ Nada

    STR R9, [R4]
    B   loop


B1
    STR R0, [R1]
    LDR R4, =LEDS
    MOV R5, #0x02
    STR R5, [R4]
    B Leer
B2
    STR R0, [R1]
    LDR R4, =LEDS
    MOV R5, #0x04
    STR R5, [R4]
    B   Leer
B1yB2
    STR R0, [R1]
    LDR R4, =LEDS
    MOV R5, #0x06
    STR R5, [R4]
    B   Leer
```

Continuación de la figura 70.

```
162  Nada
163      STR R0, [R1]
164      LDR R4, =LEDS
165      MOV R5, #0x00
166      STR R5, [R4]
167      B   Leer
168
169      ALIGN
170      END
171
```

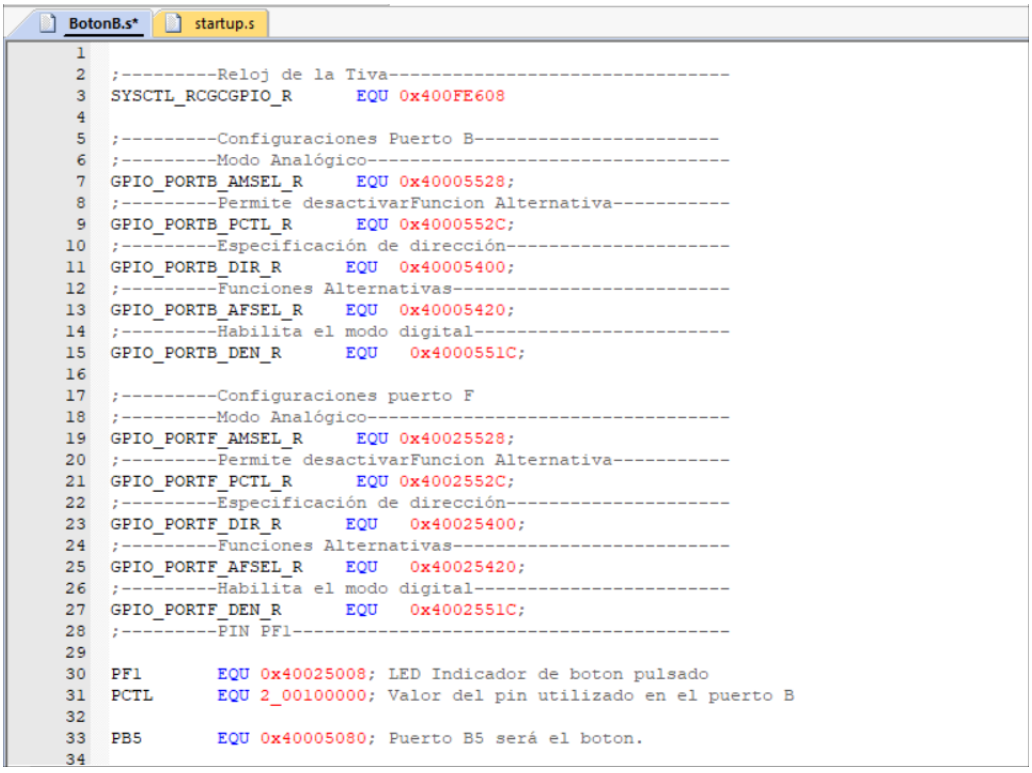
Fuente: elaboración propia.

5.8. Habilitar un puerto no dedicado como entrada

La habilitación de los puertos dedicados del microcontrolador, como se pudo observar en la práctica 5.7, no es tan complicada, aunque existen pequeñas diferencias entre la habilitación de un puerto común y un puerto dedicado. En la siguiente práctica se realizará la habilitación del pin PB0 como puerto de entrada.

- Lo primero al habilitar cualquier puerto, como se ha visto en las prácticas anteriores, es declarar las constantes. En este caso se utilizará el puerto PB5 como demostración de un puerto no dedicado como entrada; además, se declarará el puerto PF1 para tener una demostración visual al activar el pin.

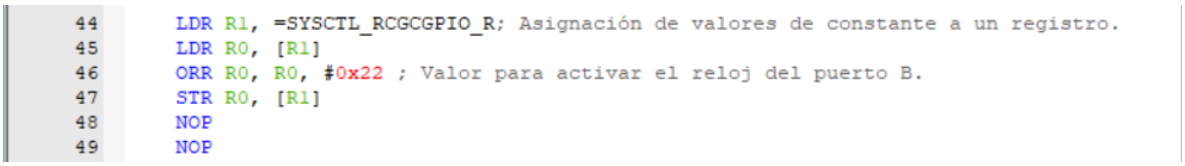
Figura 72. **Constantes puertos B y F**



Fuente: elaboración propia.

- Como se utilizarán 2 puertos distintos, se deben habilitar ambos relojes. En este caso se realizará la suma de ambos valores.

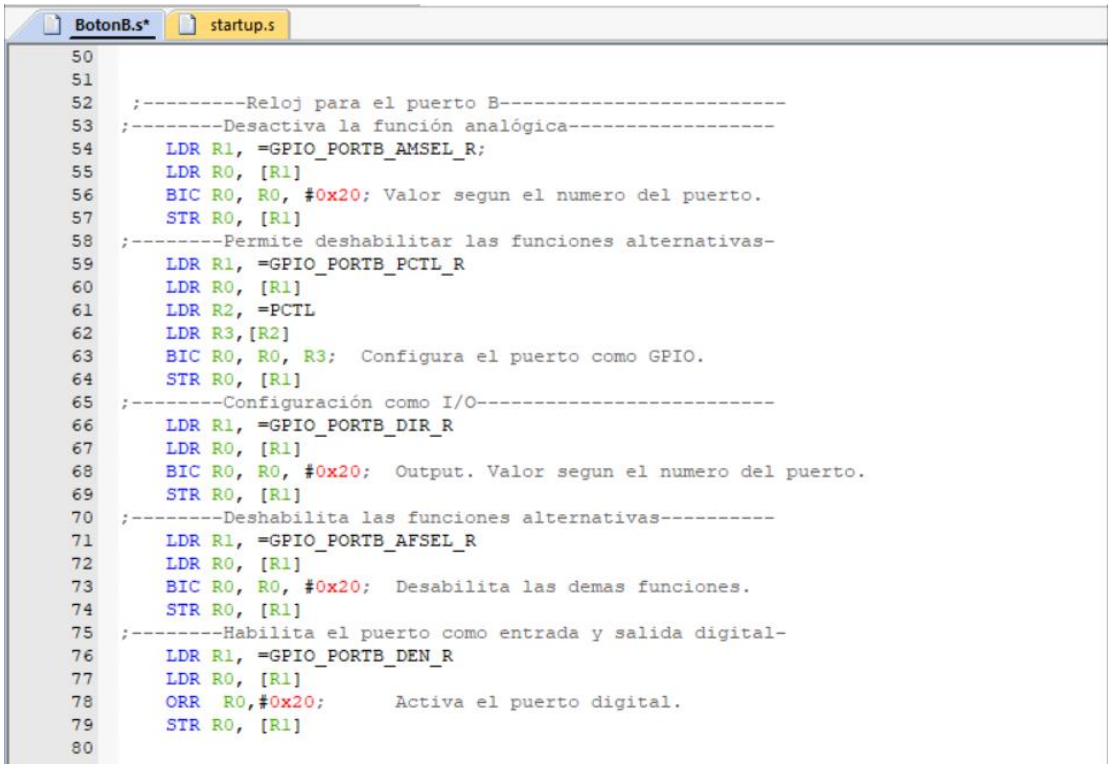
Figura 73. **Reloj para los puertos B y F**



Fuente: elaboración propia.

- Posteriormente se realiza la configuración para activar el puerto PB5 como entrada. A diferencia del puerto F, no necesita habilitar las resistencias pull-up, ni desbloquear los puertos.

Figura 74. **Configuración del puerto B como entrada**



Fuente: elaboración propia.

Se debe notar la diferencia entre entrada y salida que únicamente será en DIR. Para entrada se realiza la operación ORR y para salidas, la operación BIC.

- Luego se activará el puerto PF1 como salida, con las configuraciones previamente utilizadas.

- Luego de esto se realizará la lectura del puerto PB5. A diferencia de los puertos PF0 y PF4, los puertos no dedicados estarán activos cuando el valor sea igual de la tabla VII.

Figura 75. **Lectura del puerto PB5**

```
110
111 ;--------Lee el estado del puerto PB5----------------------
112 Lectura
113     LDR R1, =PB5
114     LDR R0, [R1]
115
116
117 ;--------Ciclo de comparación--------------------------------
118 Ciclo
119     CMP R0, #0x20; Boton pulsado
120     BEQ Activo
121     CMP R0, #0x00; Boton sin presionar
122     BEQ Apagado
123     B   Lectura
124 ;--------Subrutina si el boton está pulsado-----------------
125 Activo
126     LDR R1, =PF1
127     MOV R0, #0x02; Encender el bit.
128     STR R0, [R1];
129     B   Lectura
130 ;--------Subrutina si el boton no está pulsado-------------------
131 Apagado
132     LDR R1, =PF1
133     MOV R0, #0x00; Apagar el bit.
134     STR R0, [R1];
135     B   Lectura
136
137
138     ALIGN
139     END
```

Fuente: elaboración propia.

El código completo será el siguiente

Figura 76. **Código práctica 5.8 parte 1**

BotonB.s* startup.s

```
;---------Reloj de la Tiva---------------------------------
SYSCTL_RCGCGPIO_R      EQU 0x400FE608

;---------Configuraciones Puerto B-----------------------
;---------Modo Analógico-----------------------------------
GPIO_PORTB_AMSEL_R     EQU 0x40005528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTB_PCTL_R      EQU 0x4000552C;
;---------Especificación de dirección---------------------
GPIO_PORTB_DIR_R       EQU  0x40005400;
;---------Funciones Alternativas--------------------------
GPIO_PORTB_AFSEL_R     EQU  0x40005420;
;---------Habilita el modo digital------------------------
GPIO_PORTB_DEN_R       EQU   0x4000551C;

;---------Configuraciones puerto F
;---------Modo Analógico-----------------------------------
GPIO_PORTF_AMSEL_R     EQU 0x40025528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTF_PCTL_R      EQU 0x4002552C;
;---------Especificación de dirección---------------------
GPIO_PORTF_DIR_R       EQU   0x40025400;
;---------Funciones Alternativas--------------------------
GPIO_PORTF_AFSEL_R     EQU   0x40025420;
;---------Habilita el modo digital------------------------
GPIO_PORTF_DEN_R       EQU   0x4002551C;
;---------PIN PF1-----------------------------------------

PF1       EQU 0x40025008; LED Indicador de boton pulsado
PCTL      EQU 2_00100000; Valor del pin utilizado en el puerto B

PB5       EQU 0x40005080; Puerto B5 será el boton.

    AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Start
Start
    BL Configuracion
    B Lectura

Configuracion

    LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
    LDR R0, [R1]
    ORR R0, R0, #0x22 ; Valor para activar el reloj del puerto B.
    STR R0, [R1]
    NOP
    NOP


 ;---------Reloj para el puerto B------------------------
;--------Desactiva la función analógica-----------------
    LDR R1, =GPIO_PORTB_AMSEL_R;
    LDR R0, [R1]
    BIC R0, R0, #0x20; Valor segun el numero del puerto.
    STR R0, [R1]
;--------Permite deshabilitar las funciones alternativas-
    LDR R1, =GPIO_PORTB_PCTL_R
    LDR R0, [R1]
    LDR R2, =PCTL
    LDR R3,[R2]
    BIC R0, R0, R3;  Configura el puerto como GPIO.
    STR R0, [R1]
```

Fuente: elaboración propia.

Figura 77. **Código práctica 5.8 parte 1**

BotonB.s* startup.s

```
65  ;--------Configuración como I/O--------------------------
66      LDR R1, =GPIO_PORTB_DIR_R
67      LDR R0, [R1]
68      BIC R0, R0, #0x20;  Output. Valor segun el numero del puerto.
69      STR R0, [R1]
70  ;--------Deshabilita las funciones alternativas----------
71      LDR R1, =GPIO_PORTB_AFSEL_R
72      LDR R0, [R1]
73      BIC R0, R0, #0x20;  Desabilita las demas funciones.
74      STR R0, [R1]
75  ;--------Habilita el puerto como entrada y salida digital-
76      LDR R1, =GPIO_PORTB_DEN_R
77      LDR R0, [R1]
78      ORR  R0,#0x20;      Activa el puerto digital.
79      STR R0, [R1]
80
81
82      ;--------Desactiva la función analógica------------------
83      LDR R1, =GPIO_PORTF_AMSEL_R;
84      LDR R0, [R1]
85      BIC R0, R0, #0x02; Valor segun el numero del puerto.
86      STR R0, [R1]
87  ;--------Permite deshabilitar las funciones alternativas-
88      LDR R1, =GPIO_PORTF_PCTL_R
89      LDR R0, [R1]
90      BIC R0, R0, #2_00000100;  Configura el puerto como GPIO.
91      STR R0, [R1]
92  ;--------Configuración como I/O--------------------------
93      LDR R1, =GPIO_PORTF_DIR_R
94      LDR R0, [R1]
95      ORR R0, R0, #0x02;  Output. Valor segun el numero del puerto.
96      STR R0, [R1]
97  ;--------Deshabilita las funciones alternativas----------
98      LDR R1, =GPIO_PORTF_AFSEL_R
99      LDR R0, [R1]
100     BIC R0, R0, #0x02;  Desabilita las demas funciones.
101     STR R0, [R1]
102 ;--------Habilita el puerto como entrada y salida digital-
103     LDR R1, =GPIO_PORTF_DEN_R
104     LDR R0, [R1]
105     ORR  R0,#0x02;      Activa el puerto digital.
106     STR R0, [R1]
107 ;--------Salto, regresa a la linea posterior al salto BL---
108     BX LR
109
110
111 ;--------Lee el estado del puerto PB5---------------------
112 Lectura
113     LDR R1, =PB5
114     LDR R0, [R1]
115
116
117 ;--------Ciclo de comparación-----------------------------
118 Ciclo
119     CMP R0, #0x20; Boton pulsado
120     BEQ Activo
```

Fuente: elaboración propia.

Figura 78. **Código práctica 5.8 parte 3**

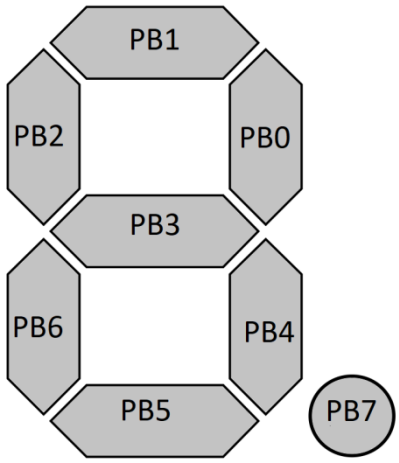
```
124  ;--------Subrutina si el boton está pulsado-----------------
125  Activo
126      LDR R1, =PF1
127      MOV R0, #0x02; Encender el bit.
128      STR R0, [R1];
129      B   Lectura
130  ;--------Subrutina si el boton no está pulsado-----------------
131  Apagado
132      LDR R1, =PF1
133      MOV R0, #0x00; Apagar el bit.
134      STR R0, [R1];
135      B   Lectura
136
137
138      ALIGN
139      END
```

Fuente: elaboración propia.

5.9. Manejo de arreglos para la programación de un 7-segmentos

Una vez se puede activar y desactivar puertos como salida o entrada, la siguiente práctica es el manejo de estos puertos con la ayuda de vectores, para simplificar la utilización de todos los pines de un puerto. Para esto se utilizará el puerto B, puesto que es el único que tiene 8 bits completos que pueden ser programados.

Figura 79. **7-Segmentos PIN por segmento**



Fuente: elaboración propia.

Tabla XIII. **Valores correspondientes a cada número**

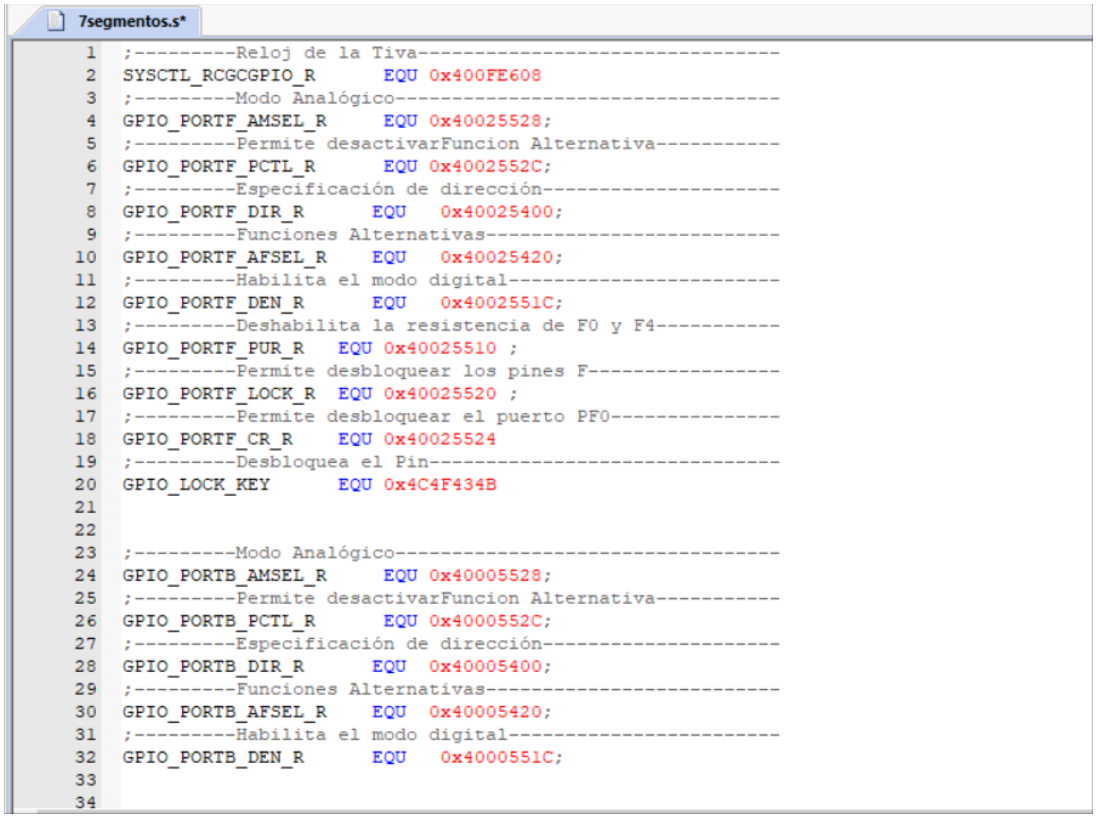
Número	Pines a utilizar	Valor correspondiente al arreglo
0	PB0, PB1, PB2, PB4, PB5, PB6	2_01110111
1	PB0, PB4	2_00010001
2	PB0, PB1,PB3, PB5, PB6	2_01101011
3	PB0, PB1, PB3, PB4, PB5	2_00111011
4	PB0, PB2, PB3, PB4	2_00011101
5	PB1, PB2, PB3, PB4, PB5	2_00111110
6	PB2, PB3, PB4, PB5, PB6	2_01111110
7	PB0, PB1, PB4	2_00010011
8	PB0, PB1, PB2, PB3, PB4, PB5, PB6	2_01111111
9	PB0, PB1, PB2, PB3, PB4.	2_00011111
Punto	PB7	2_10000000

Fuente: elaboración propia.

En la tabla XII se observa que cada pin corresponde a un valor en el vector; el número PB0 es el de la derecha y el PB7, el de la izquierda.

- Como en las prácticas anteriores, se debe inicializar las constantes de los puertos que se utilizarán. En este caso se utilizarán los mismos puertos de la práctica anterior, los cuales son B y F.

Figura 80. **Constantes para la práctica 5.9**



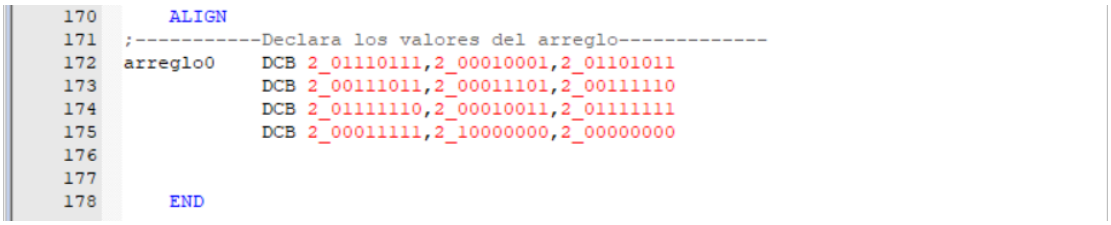
7segmentos.s*

```
;---------Reloj de la Tiva---------------------------------
SYSCTL_RCGCGPIO_R      EQU 0x400FE608
;---------Modo Analógico-----------------------------------
GPIO_PORTF_AMSEL_R     EQU 0x40025528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTF_PCTL_R      EQU 0x4002552C;
;---------Especificación de dirección---------------------
GPIO_PORTF_DIR_R       EQU   0x40025400;
;---------Funciones Alternativas--------------------------
GPIO_PORTF_AFSEL_R     EQU   0x40025420;
;---------Habilita el modo digital------------------------
GPIO_PORTF_DEN_R       EQU   0x4002551C;
;---------Deshabilita la resistencia de F0 y F4-----------
GPIO_PORTF_PUR_R   EQU 0x40025510 ;
;---------Permite desbloquear los pines F-----------------
GPIO_PORTF_LOCK_R  EQU 0x40025520 ;
;---------Permite desbloquear el puerto PF0---------------
GPIO_PORTF_CR_R    EQU 0x40025524
;---------Desbloquea el Pin-------------------------------
GPIO_LOCK_KEY      EQU 0x4C4F434B


;---------Modo Analógico-----------------------------------
GPIO_PORTB_AMSEL_R     EQU 0x40005528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTB_PCTL_R      EQU 0x4000552C;
;---------Especificación de dirección---------------------
GPIO_PORTB_DIR_R       EQU  0x40005400;
;---------Funciones Alternativas--------------------------
GPIO_PORTB_AFSEL_R     EQU  0x40005420;
;---------Habilita el modo digital------------------------
GPIO_PORTB_DEN_R       EQU   0x4000551C;


```

Fuente: elaboración propia.

- Posteriormente se declarará el arreglo o arreglos que se utilizarán. Estos irán ubicados en medio de ALIGN y END justo al final del documento. Declarado con la etiqueta arreglo.

Figura 81. **Arreglo para el 7-segmentos**

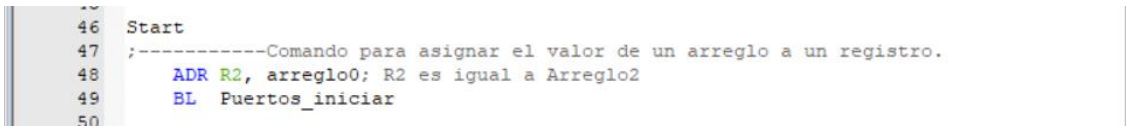


```
170     ALIGN
171 ;-----------Declara los valores del arreglo-------------
172 arreglo0    DCB 2_01110111,2_00010001,2_01101011
173             DCB 2_00111011,2_00011101,2_00111110
174             DCB 2_01111110,2_00010011,2_01111111
175             DCB 2_00011111,2_10000000,2_00000000
176
177
178     END
```

Fuente: elaboración propia.

- Teniendo el arreglo configurado, estos datos deberán ser asignados a un registro. Con el comando ADR se asigna la dirección donde inicia el vector en el registro R2.

Figura 82. **Asignación de dirección del vector**



```
46  Start
47  ;-----------Comando para asignar el valor de un arreglo a un registro.
48      ADR R2, arreglo0; R2 es igual a Arreglo2
49      BL  Puertos_iniciar
50
```

Fuente: elaboración propia.

- Se indicará la secuencia que tomará el programa, para que siempre se encuentre leyendo el estado del botón y además se mantenga encendido el display.

Figura 83. **Secuencia de ejecución**

```
50
51  ;--------------Orden en el que se ejecutarán los comandos------
52  Ciclo
53      LDR R7, =ONESEC
54      BL  retardo
55      BL  Display
56
57      BL  Boton
58      BL  Reset
59      B   Ciclo
```

Fuente: elaboración propia

- Se debe configurar el reloj para ambos puertos.

Figura 84. **Habilitación del reloj práctica 5.9**

```
61  ;------------Habilitar Reloj-----------------------------
62  Puertos_iniciar
63      LDR R1, =SYSCTL_RCGCGPIO_R      ; activar reloj
64      LDR R0, [R1]
65      ORR R0, R0, #0x22
66      STR R0, [R1]
67      NOP
68      NOP
69      NOP
```

Fuente: elaboración propia

- Se deberá configurar los puertos como entradas y salidas dependiendo del puerto. En este caso la entrada es el pin PF4 y la salida el puerto PB.

Figura 85. **Configuración de los puertos**

7segmentos.s*

```
    ;PORT F
;--------Declarar puertos como Entrada-----------------
;---------Desbloquear los pines PF0 y PF1----------------
    LDR R1, =GPIO_PORTF_LOCK_R
    LDR R0, =GPIO_LOCK_KEY
    STR R0, [R1]

    LDR R1, =GPIO_PORTF_CR_R
    MOV R0, #0xFF
    STR R0, [R1]
;--------Configuración como I/O--------------------------
    LDR R1, =GPIO_PORTF_DIR_R
    LDR R0, [R1]
    BIC R0, R0, #0x10
    STR R0, [R1]
;--------Deshabilita las funciones alternativas----------
    LDR R1, =GPIO_PORTF_AFSEL_R
    LDR R0, [R1]
    BIC R0, R0, #0x10
    STR R0, [R1]
;--------Desactiva la resistencia pull up----------------
    LDR R1, =GPIO_PORTF_PUR_R
    LDR R0, [R1]
    ORR R0, R0, #0x10
    STR R0, [R1]
;--------Habilita el puerto como entrada y salida digital-
    LDR R1, =GPIO_PORTF_DEN_R
    LDR R0, [R1]
    ORR R0, R0, #0x10
    STR R0, [R1]
;--------Permite deshabilitar las funciones alternativas-
    LDR R1, =GPIO_PORTF_PCTL_R
    LDR R0, [R1]
    BIC R0, R0, #0x000F0000
    STR R0, [R1]

    ;PORT B
;--------Configuración como I/O--------------------------
    LDR R1, =GPIO_PORTB_DIR_R
    LDR R0, [R1]
    ORR R0, R0, #0xFF;  Output. Valor segun el numero del puerto.
    STR R0, [R1]                ;
;--------Deshabilita las funciones alternativas----------
    LDR R1, =GPIO_PORTB_AFSEL_R
    LDR R0, [R1]
    BIC R0, R0, #0xFF;  Desabilita las demas funciones.
    STR R0, [R1]
;--------Habilita el puerto como entrada y salida digital-
    LDR R1, =GPIO_PORTB_DEN_R
    LDR R0, [R1]
    ORR  R0,#0xFF;      Activa el puerto digital.
    STR R0, [R1]
;--------Permite deshabilitar las funciones alternativas-
    LDR R1, =GPIO_PORTB_PCTL_R
    LDR R0, [R1]
    BIC R0, R0, #2_11111111;  Configura el puerto como GPIO.
    STR R0, [R1]

;--------Desactiva la función analógica-----------------
    LDR R1, =GPIO_PORTB_AMSEL_R;
    LDR R0, [R1]
    BIC R0, R0, #0xFF; Valor segun el numero del puerto.
    STR R0, [R1]
```

Fuente: elaboración propia.

- En esta práctica se ha agregado un contador, el cual permite que los números puedan visualizarse mientras está presionado el botón PF4 durante 1 segundo antes de volver a leer el estado del botón y realizar el cambio.

Figura 86. **Retardo y botón**

```
138 ;---------Retardo para que no ocurra rebote-----------------
139 retardo
140     SUBS R7, R7, #1
141     BNE retardo
142     BX  LR
143 ;----------Leerá el estado del pin PF4----------------------
144 Boton
145     LDR R4, =PF4
146     LDR R0, [R4]
147     CMP R0, #0x10
148     BEQ Boton
149     BX LR
```

Fuente: elaboración propia.

- Posteriormente se realizará la asignación del vector en la posición 1 al vector del puerto B. Para esto se utiliza el comando LDRB, el cual permite asignar los valores como LDR que se conoce y además mover a la posición siguiente el registro R2.

Figura 87. **Display asignación**

```
151 ;------------Encender Display---------------------------------
152 ;LDRB: A R3 se le asigna los valores que se encuentran de la dirección
153 ;guardada en R2 y a R2 se le indica que se moverá una posicion
154 ;dentro del vector
155 ;-------------------------------------------------------------
156 Display
157     LDRB R3,[R2],#1;
158     LDR R1, =PB; A R1 se le asigna el vector de PB
159     STR R3, [R1]; En esta linea se le asignan los valores de R3 a R1,
160                 ;En otras palabras los valores del vector a PB
161     BX  LR
```

Fuente: elaboración propia.

- Por último, si el vector no se vuelve a inicializar comenzará a generar números random que se encuentran ubicados en la memoria del microcontrolador, por lo que deberá hacerse una comparación de donde se desea que termine el vector e inicializarlo de nuevo.

Figura 88. **Reinicio de la posición del vector**

```
162
163 ;------------Reset------------
164 Reset
165     CMP R3, #2_10000000; Compara si ya tiene el valor deseado
166                        ;Aun que hayan mas valores no continuará.
167     BEQ Start
168     BX  LR
169
```

Fuente: elaboración propia.

Con esto ya se tendrá un contador de 0 a 9, el cual contará mientras el botón PF4 esté presionado. Al soltarse este guardará el último valor que mostró, hasta que se vuelva a presionar. A continuación, el código completo.

Figura 89. **Código completo práctica 5.9 parte 1**

7segmentos.s*

```
;---------Reloj de la Tiva---------------------------------
SYSCTL_RCGCGPIO_R      EQU 0x400FE608
;---------Modo Analógico-----------------------------------
GPIO_PORTF_AMSEL_R     EQU 0x40025528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTF_PCTL_R      EQU 0x4002552C;
;---------Especificación de dirección---------------------
GPIO_PORTF_DIR_R       EQU   0x40025400;
;---------Funciones Alternativas--------------------------
GPIO_PORTF_AFSEL_R     EQU   0x40025420;
;---------Habilita el modo digital------------------------
GPIO_PORTF_DEN_R       EQU   0x4002551C;
;---------Deshabilita la resistencia de F0 y F4-----------
GPIO_PORTF_PUR_R   EQU 0x40025510 ;
;---------Permite desbloquear los pines F-----------------
GPIO_PORTF_LOCK_R  EQU 0x40025520 ;
;---------Permite desbloquear el puerto PF0---------------
GPIO_PORTF_CR_R    EQU 0x40025524
;---------Desbloquea el Pin-------------------------------
GPIO_LOCK_KEY      EQU 0x4C4F434B


;---------Modo Analógico-----------------------------------
GPIO_PORTB_AMSEL_R     EQU 0x40005528;
;---------Permite desactivarFuncion Alternativa-----------
GPIO_PORTB_PCTL_R      EQU 0x4000552C;
;---------Especificación de dirección---------------------
GPIO_PORTB_DIR_R       EQU   0x40005400;
;---------Funciones Alternativas--------------------------
GPIO_PORTB_AFSEL_R     EQU   0x40005420;
;---------Habilita el modo digital------------------------
GPIO_PORTB_DEN_R       EQU   0x4000551C;


;------------Pines que se utilizarán------------------------
PB             EQU  0x400053FC;Todos los pines del puerto B
PF4            EQU 0x40025040;Puerto F4
 ;-----------Delay de un segundo----------------------------
ONESEC         EQU 5333333  ;[[16*10E6]/3]=5333333.333, numero asociado a 1 segundo

    AREA    |.text|, CODE, READONLY, ALIGN=2
    THUMB
    EXPORT  Start


Start
;-----------Comando para asignar el valor de un arreglo a un registro.
    ADR R2, arreglo0; R2 es igual a Arreglo2
    BL  Puertos_iniciar

;--------------Orden en el que se ejecutarán los comandos------
Ciclo
    LDR R7, =ONESEC
    BL  retardo
    BL  Display

    BL  Boton
    BL  Reset
    B   Ciclo

;------------Habilitar Reloj-------------------------------
Puertos_iniciar
    LDR R1, =SYSCTL_RCGCGPIO_R      ; activar reloj
    LDR R0, [R1]
```

Fuente: elaboración propia.

Figura 90. **Código completo práctica 5.9 parte 2**

```
65      ORR R0, R0, #0x22
66      STR R0, [R1]
67      NOP
68      NOP
69      NOP
70      ;PORT F
71  ;--------Declarar puertos como Entrada-----------------
72  ;---------Desbloquear los pines PF0 y PF1----------------
73      LDR R1, =GPIO_PORTF_LOCK_R
74      LDR R0, =GPIO_LOCK_KEY
75      STR R0, [R1]
76
77      LDR R1, =GPIO_PORTF_CR_R
78      MOV R0, #0xFF
79      STR R0, [R1]
80  ;--------Configuración como I/O---------------------------
81      LDR R1, =GPIO_PORTF_DIR_R
82      LDR R0, [R1]
83      BIC R0, R0, #0x10
84      STR R0, [R1]
85  ;--------Deshabilita las funciones alternativas----------
86      LDR R1, =GPIO_PORTF_AFSEL_R
87      LDR R0, [R1]
88      BIC R0, R0, #0x10
89      STR R0, [R1]
90  ;--------Desactiva la resistencia pull up----------------
91      LDR R1, =GPIO_PORTF_PUR_R
92      LDR R0, [R1]
93      ORR R0, R0, #0x10
94      STR R0, [R1]
95  ;--------Habilita el puerto como entrada y salida digital-
96      LDR R1, =GPIO PORTF DEN R
97      LDR R0, [R1]
98      ORR R0, R0, #0x10
99      STR R0, [R1]
100 ;--------Permite deshabilitar las funciones alternativas-
101     LDR R1, =GPIO_PORTF_PCTL_R
102     LDR R0, [R1]
103     BIC R0, R0, #0x000F0000
104     STR R0, [R1]
105
106     ;PORT B
107 ;--------Configuración como I/O---------------------------
108     LDR R1, =GPIO_PORTB_DIR_R
109     LDR R0, [R1]
110     ORR R0, R0, #0xFF;  Output. Valor segun el numero del puerto.
111     STR R0, [R1]                ;
112 ;--------Deshabilita las funciones alternativas----------
113     LDR R1, =GPIO_PORTB_AFSEL_R
114     LDR R0, [R1]
115     BIC R0, R0, #0xFF;  Desabilita las demas funciones.
116     STR R0, [R1]
117 ;--------Habilita el puerto como entrada y salida digital-
118     LDR R1, =GPIO_PORTB_DEN_R
119     LDR R0, [R1]
120     ORR  R0,#0xFF;      Activa el puerto digital.
121     STR R0, [R1]
122 ;--------Permite deshabilitar las funciones alternativas-
123     LDR R1, =GPIO_PORTB_PCTL_R
124     LDR R0, [R1]
125     BIC R0, R0, #2_11111111;  Configura el puerto como GPIO.
126     STR R0, [R1]
127
128 ;--------Desactiva la función analógica-----------------
```

Fuente: elaboración propia.

Figura 91. **Código completo práctica 5.9 parte 3**

```
129     LDR R1, =GPIO_PORTB_AMSEL_R;
130     LDR R0, [R1]
131     BIC R0, R0, #0xFF; Valor segun el numero del puerto.
132     STR R0, [R1]
133
134     BX  LR
135
136
137
138 ;---------Retardo para que no ocurra rebote-----------------
139 retardo
140     SUBS R7, R7, #1
141     BNE retardo
142     BX  LR
143 ;----------Leerá el estado del pin PF4----------------------
144 Boton
145     LDR R4, =PF4
146     LDR R0, [R4]
147     CMP R0, #0x10
148     BEQ Boton
149     BX LR
150
151 ;------------Encender Display---------------------------------
152 ;LDRB: A R3 se le asigna los valores que se encuentran de la dirección
153 ;guardada en R2 y a R2 se le indica que se moverá una posicion
154 ;dentro del vector
155 ;-------------------------------------------------------------
156 Display
157     LDRB R3,[R2],#1;
158     LDR R1, =PB; A R1 se le asigna el vector de PB
159     STR R3, [R1]; En esta linea se le asignan los valores de R3 a R1,
160                 ;En otras palabras los valores del vector a PB
161     BX  LR
162
163 ;------------Reset------------
164 Reset
165     CMP R3, #2_10000000; Compara si ya tiene el valor deseado
166                        ;Aun que hayan mas valores no continuará.
167     BEQ Start
168     BX  LR
169
170     ALIGN
171 ;-----------Declara los valores del arreglo-------------
172 arreglo0    DCB 2_01110111,2_00010001,2_01101011
173             DCB 2_00111011,2_00011101,2_00111110
174             DCB 2_01111110,2_00010011,2_01111111
175             DCB 2_00011111,2_10000000,2_00000000
176
177
178     END
```

Fuente: elaboración propia

5.10. Interrupción de falla

Existen dos tipos de interrupciones, una de ellas son las interrupciones no enmascarables que básicamente son interrupciones que el microprocesador debe cumplir y el programador no puede evitar. A modo de ejemplo se realizará un error en la práctica 5.4 en la cual se activa el pin PF1.

Figura 92. **Práctica 5.4 con error parte 1**

Interrupción.s | startup.s

```
1
2  ;---------Reloj de la Tiva--------------------------------
3  SYSCTL_RCGCGPIO_R      EQU 0x400FE608
4  ;---------Modo Analógico-----------------------------------
5  GPIO_PORTF_AMSEL_R     EQU 0x40025528;
6  ;---------Permite desactivarFuncion Alternativa-----------
7  GPIO_PORTF_PCTL_R      EQU 0x4002552C;
8  ;---------Especificación de dirección---------------------
9  GPIO_PORTF_DIR_R       EQU   0x40025400;
10 ;---------Funciones Alternativas--------------------------
11 GPIO_PORTF_AFSEL_R     EQU   0x40025420;
12 ;---------Habilita el modo digital------------------------
13 GPIO_PORTF_DEN_R       EQU   0x4002551C;
14 ;---------PIN PF1-----------------------------------------
15 PF1                    EQU 0x40025008
16
17         AREA    |.text|, CODE, READONLY, ALIGN=2
18         THUMB
19         EXPORT  Start
20
21
22 Start
23     B Hola; Salto, guarda la dirección posterior en el Registro 14
24     B  LED ; Salto
25
26 Hola
27
28 ;---------Reloj para el puerto F-------------------------
29     LDR R1, =SYSCTL_RCGCGPIO_R; Asignación de valores de constante a un registro.
30     LDR R0, [R1]
31     ORR R0, R0, #0x20 ; Valor para activar el puerto F
32     STR R0, [R1]
33     NOP
```

Fuente: elaboración propia.

Figura 93. **Práctica 5.4 con error parte 2**

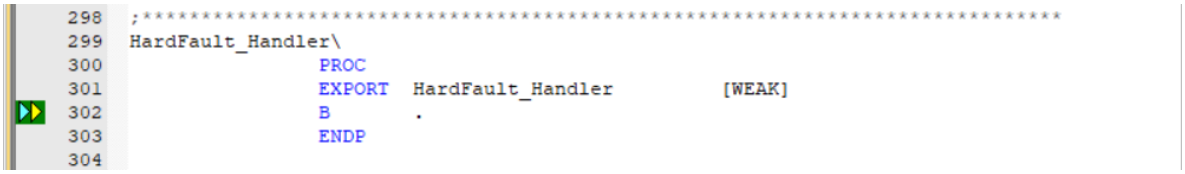
```
33      NOP
34      NOP
35  ;--------Desactiva la función analógica-----------------
36      LDR R1, =GPIO_PORTF_AMSEL_R;
37      LDR R0, [R1]
38      BIC R0, R0, #0x02; Valor segun el numero del puerto.
39      STR R0, [R1]
40  ;--------Permite deshabilitar las funciones alternativas-
41      LDR R1, =GPIO_PORTF_PCTL_R
42      LDR R0, [R1]
43      BIC R0, R0, #0x000000F0;  Configura el puerto como GPIO.
44      STR R0, [R1]
45  ;--------Configuración como I/O--------------------------
46      LDR R1, =GPIO_PORTF_DIR_R
47      LDR R0, [R1]
48      ORR R0, R0, #0x02;  Output. Valor segun el numero del puerto.
49      STR R0, [R1]
50  ;--------Deshabilita las funciones alternativas----------
51      LDR R1, =GPIO_PORTF_AFSEL_R
52      LDR R0, [R1]
53      BIC R0, R0, #0x02;  Desabilita las demas funciones.
54      STR R0, [R1]
55  ;--------Habilita el puerto como entrada y salida digital-
56      LDR R1, =GPIO_PORTF_DEN_R
57      LDR R0, [R1]
58      ORR  R0,#0x02;      Activa el puerto digital.
59      STR R0, [R1]
60  ;--------Salto, regresa a la linea posterior al salto BL---
61      BX LR
62
63  ;--------Activar el Pin----------------------------------
64  LED
65      LDR R1, =PF1
66      MOV R0, #0x02; Encender el bit.
67      STR R0, [R1];
68      B LED
69
70
71      ALIGN
72      END
```

Fuente: elaboración propia.

La única diferencia entre el código correcto y el código error se encuentra en la línea 23, donde no se guarda la dirección siguiente al salto, por lo que el registro R14 se mantendrá por defecto. Al momento de construir el código, este no dará ningún error, puesto que para el compilador es válido este valor. El error se mostrará hasta el momento en que se ejecute la línea de instrucción 61, donde debería realizar el salto. Esto generará una interrupción que el microcontrolador no puede ignorar y dirigirá al error en la línea 299,

HardFault_Handler, el cual mantendrá en un ciclo el código y evitará un daño al microcontrolador.

Figura 94. **Error no enmascarable**



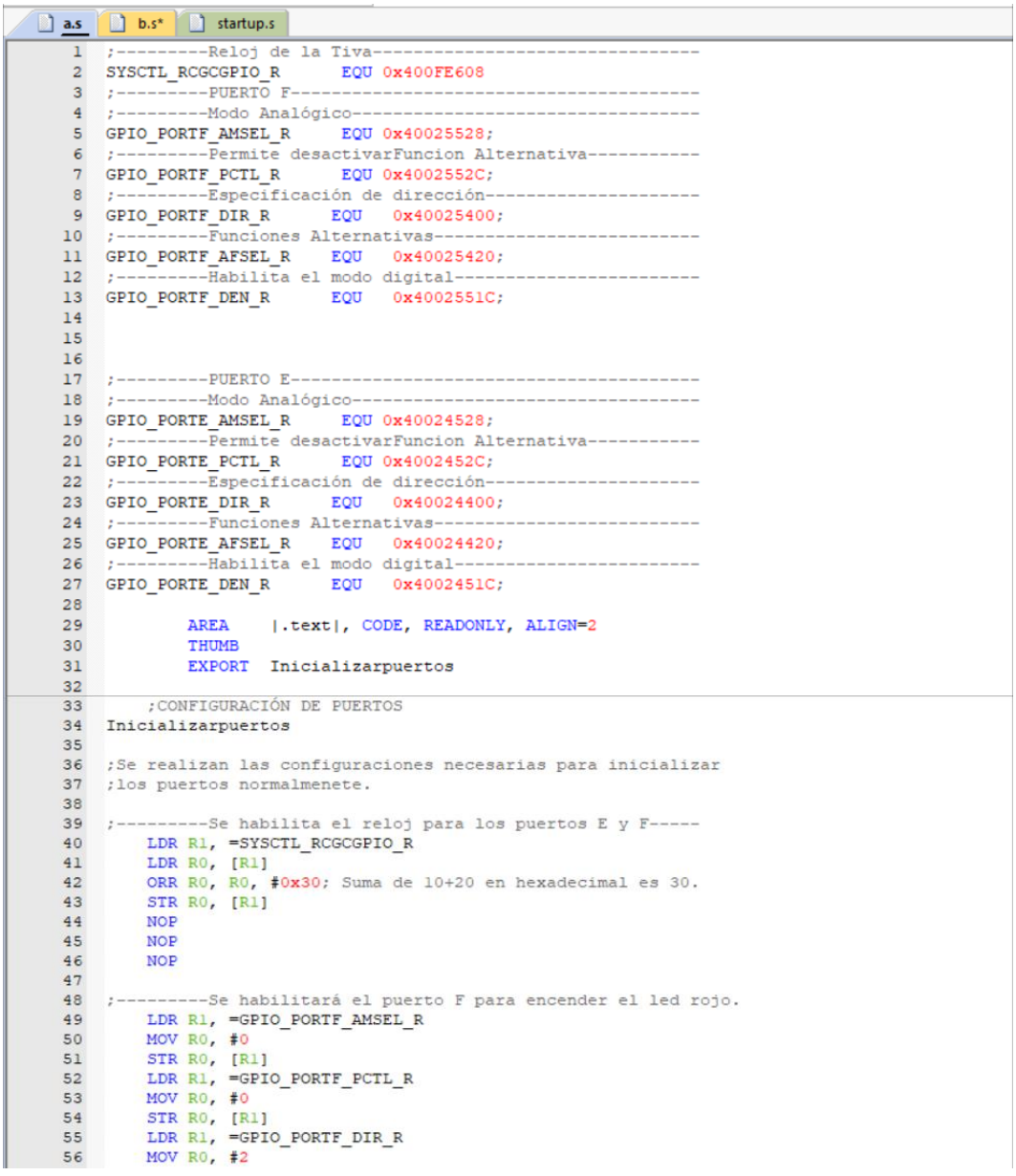
Fuente: elaboración propia.

5.11. Interrupción por temporizador

Se realizará una interrupción por temporizador o timer, en la cual, al evitar el uso de contadores, se encenderán los pines PF1 y PE4. Para esto se utilizarán dos archivos diferentes.

- El primer archivo (a) contendrá los valores para inicializar los puertos PF y PE. Este es mandado a llamar desde el segundo archivo (b).

Figura 95. **Inicialización puerto F y B parte 1**



```
;---------Reloj de la Tiva-------------------------------
SYSCTL_RCGCGPIO_R      EQU 0x400FE608
;---------PUERTO F---------------------------------------
;---------Modo Analógico---------------------------------
GPIO_PORTF_AMSEL_R     EQU 0x40025528;
;---------Permite desactivarFuncion Alternativa----------
GPIO_PORTF_PCTL_R      EQU 0x4002552C;
;---------Especificación de dirección--------------------
GPIO_PORTF_DIR_R       EQU   0x40025400;
;---------Funciones Alternativas-------------------------
GPIO_PORTF_AFSEL_R     EQU   0x40025420;
;---------Habilita el modo digital-----------------------
GPIO_PORTF_DEN_R       EQU   0x4002551C;



;---------PUERTO E---------------------------------------
;---------Modo Analógico---------------------------------
GPIO_PORTE_AMSEL_R     EQU 0x40024528;
;---------Permite desactivarFuncion Alternativa----------
GPIO_PORTE_PCTL_R      EQU 0x4002452C;
;---------Especificación de dirección--------------------
GPIO_PORTE_DIR_R       EQU   0x40024400;
;---------Funciones Alternativas-------------------------
GPIO_PORTE_AFSEL_R     EQU   0x40024420;
;---------Habilita el modo digital-----------------------
GPIO_PORTE_DEN_R       EQU   0x4002451C;

        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Inicializarpuertos

    ;CONFIGURACIÓN DE PUERTOS
Inicializarpuertos

;Se realizan las configuraciones necesarias para inicializar
;los puertos normalmenete.

;---------Se habilita el reloj para los puertos E y F-----
    LDR R1, =SYSCTL_RCGCGPIO_R
    LDR R0, [R1]
    ORR R0, R0, #0x30; Suma de 10+20 en hexadecimal es 30.
    STR R0, [R1]
    NOP
    NOP
    NOP

;---------Se habilitará el puerto F para encender el led rojo.
    LDR R1, =GPIO_PORTF_AMSEL_R
    MOV R0, #0
    STR R0, [R1]
    LDR R1, =GPIO_PORTF_PCTL_R
    MOV R0, #0
    STR R0, [R1]
    LDR R1, =GPIO_PORTF_DIR_R
    MOV R0, #2
```

Fuente: elaboración propia.

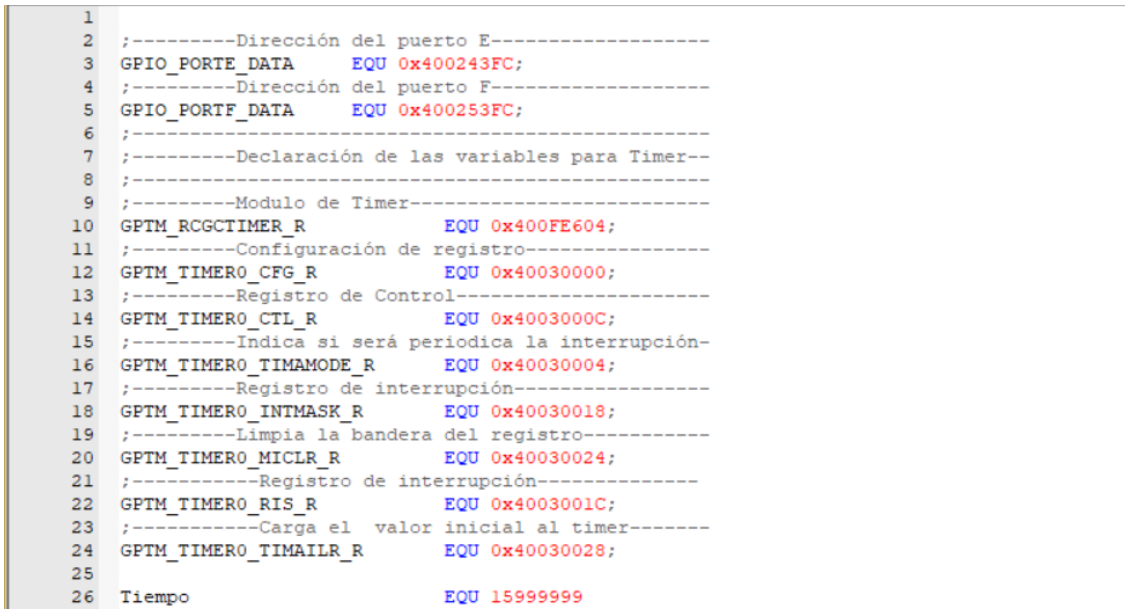
Figura 96. **Inicialización puerto F y B parte 2**

```
56      MOV R0, #2
57      STR R0, [R1]
58      LDR R1, =GPIO_PORTF_AFSEL_R
59      MOV R0, #0
60      STR R0, [R1]
61      LDR R1, =GPIO_PORTF_DEN_R
62      MOV R0, #0xFF
63      STR R0, [R1]
64
65  ;----------El puerto E se estará utilizando  como el detonante
66  ;----------de la interrupción.
67          LDR R1, =GPIO_PORTE_PCTL_R
68          LDR R0, [R1]
69          BIC R0, R0, #0x0
70          STR R0, [R1]
71          LDR R1, =GPIO_PORTE_AMSEL_R
72          MOV R0, #0
73          STR R0, [R1]
74          LDR R1, =GPIO_PORTE_AFSEL_R
75          MOV R0, #0
76          STR R0, [R1]
77          LDR R1, =GPIO_PORTE_DIR_R
78          MOV R0, #0xFF
79          STR R0, [R1]
80          LDR R1, =GPIO_PORTE_DEN_R
81          MOV R0, #0xFF
82          STR R0, [R1]
83
84          BX  LR
85
86              ALIGN
87              END
88
```

Fuente: elaboración propia.

- En el segundo archivo se deben declarar las constantes estudiadas en la sección 4.5.6.

Figura 97. **Constantes para interrupción por temporizador**



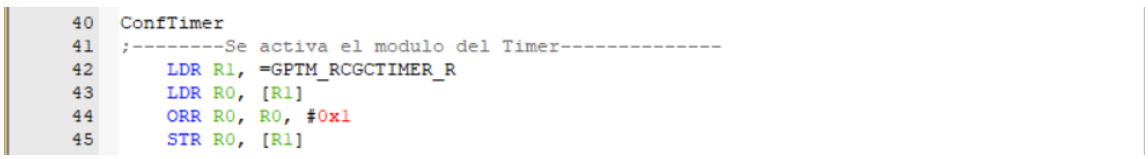
```
1
2  ;---------Dirección del puerto E------------------
3  GPIO_PORTE_DATA     EQU 0x400243FC;
4  ;---------Dirección del puerto F------------------
5  GPIO_PORTF_DATA     EQU 0x400253FC;
6  ;-----------------------------------------------
7  ;---------Declaración de las variables para Timer--
8  ;-----------------------------------------------
9  ;---------Modulo de Timer-------------------------
10 GPTM_RCGCTIMER_R            EQU 0x400FE604;
11 ;---------Configuración de registro---------------
12 GPTM_TIMER0_CFG_R           EQU 0x40030000;
13 ;---------Registro de Control---------------------
14 GPTM_TIMER0_CTL_R           EQU 0x4003000C;
15 ;---------Indica si será periodica la interrupción-
16 GPTM_TIMER0_TIMAMODE_R      EQU 0x40030004;
17 ;---------Registro de interrupción----------------
18 GPTM_TIMER0_INTMASK_R       EQU 0x40030018;
19 ;---------Limpia la bandera del registro----------
20 GPTM_TIMER0_MICLR_R         EQU 0x40030024;
21 ;-----------Registro de interrupción-------------
22 GPTM_TIMER0_RIS_R           EQU 0x4003001C;
23 ;-----------Carga el  valor inicial al timer-------
24 GPTM_TIMER0_TIMAILR_R       EQU 0x40030028;
25
26 Tiempo                      EQU 15999999
```

Fuente: elaboración propia.

El valor de tiempo quedará a discreción del estudiante para indicar el tiempo que necesite.

- Posteriormente se debe configurar el timer, pero primero se debe activar el módulo de este.

Figura 98. **Módulo de timer**

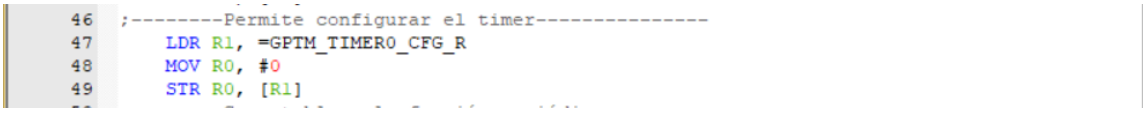


```
40 ConfTimer
41 ;--------Se activa el modulo del Timer-------------
42     LDR R1, =GPTM_RCGCTIMER_R
43     LDR R0, [R1]
44     ORR R0, R0, #0x1
45     STR R0, [R1]
```

Fuente: elaboración propia.

- Se debe permitir en la utilización de interrupciones de timer.

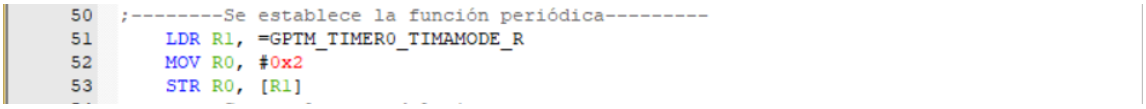
Figura 99. **Permite el manejo de las NMI**



Fuente: elaboración propia.

- Se debe establecer que la función será periódica. El valor 0x2 indica esta opción, como se ve en la tabla IX.

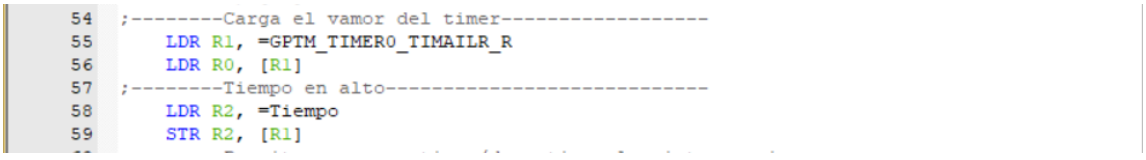
Figura 100. **Función periódica del timer**



Fuente: elaboración propia.

- Se deberá inicializar el valor del timer, para que cuando se active, comience el contador de ciclos a decrecer.

Figura 101. **Valor del timer**



Fuente: elaboración propia.

- Se deberá permitir que se activen y desactiven las interrupciones. Para esto se utiliza el registro INTMASK.

Figura 102. **Máscara de interrupción**

```
60  ;--------Permite que se activen/desactiven las interrupciones.
61      LDR R1, =GPTM_TIMER0_INTMASK_R
62      LDR R0, [R1]
63      ORR R0, R0, #0x1
64      STR R0, [R1]
```

Fuente: elaboración propia.

- Se debe ajustar el temporizador para que comience a contar a partir de 0.

Figura 103. **Registro de control**

```
66      LDR R1, =GPTM_TIMER0_CTL_R
67      LDR R0, [R1]
68      ORR R0,R0, #0x1
69      STR R0, [R1]
```

Fuente: elaboración propia.

- Ya teniendo configurado la interrupción se activará el puerto E4 como salida.

Figura 104. **Activa el puerto E4**

```
70  ;---------Valores del puerto E--------------------
71      LDR R1, =GPIO_PORTE_DATA
72      MOV R0, #0x10
73      STR R0, [R1]
```

Fuente: elaboración propia.

- Se deberá comprobar que la bandera que compara el final del timer ha sido levantada, si es así, realizará un salto a led E4; de lo contrario, esperará hasta que el contador del timer llegue a 0x1

Figura 105. **Ciclo de timer**

```
77  Ciclo
78  ;---------Levanta la bandera del estado en alto---------
79      LDR R1, =GPTM_TIMER0_RIS_R
80      LDR R0, [R1]
81      CMP R0, #0x1
82      BEQ LedE4
83      B Ciclo
```

Fuente: elaboración propia.

- Luego de realizar el salto del estado de la interrupción, saltará a led E4 donde se encenderá el pin PE4, se apagará PF1 y se reiniciará el contador del timer.

Figura 106. **Activa pin PE4 y apaga pin F1**

```
85  LedE4
86  ;--------------Enciende el led E4---------------
87      LDR R1, =GPIO_PORTE_DATA
88      LDR R0, [R1]
89      CMP R0, #0x10
90      BEQ LedF1
91      MOV R0, #0x10
92      STR R0, [R1]
93  ;---------Apaga Led F1-----------------------------
94      LDR R1, =GPIO_PORTF_DATA
95      MOV R0, #0x0
96      STR R0, [R1]
97  ;---------Limpia el contador interno del timer-------
98      LDR R1, =GPTM_TIMER0_MICLR_R
99      LDR R0, [R1]
100     ORR R0, R0, #0x1
101     STR R0, [R1]
102     B   Ciclo
```

Fuente: elaboración propia.

De la misma forma que en el anterior, pero encendiendo el pin PF1 y apagando el pin PE4, reiniciará el conteo y saltará a la función Ciclo.

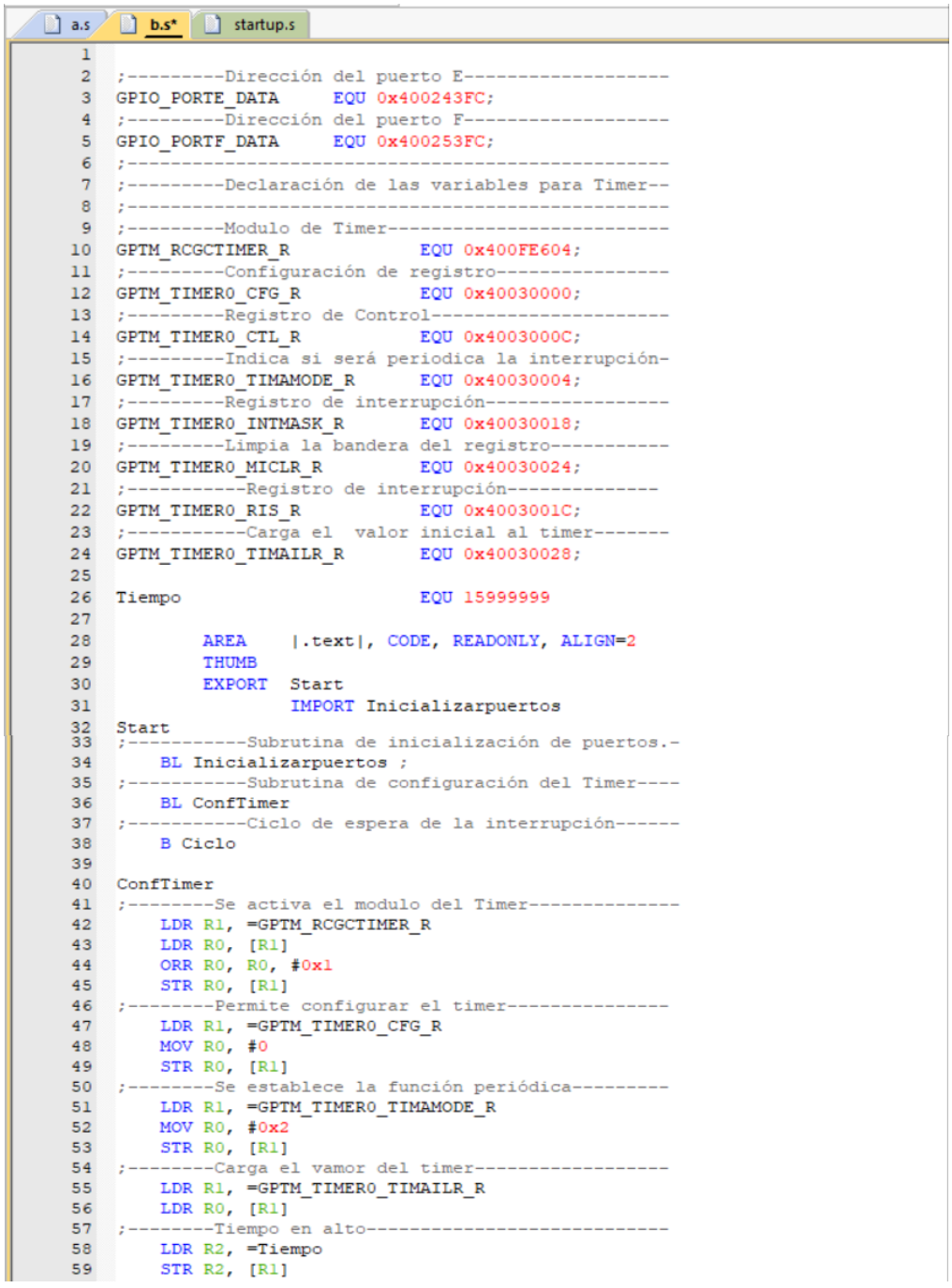
Figura 107. **Activa pin PF1 y apaga pin PE4**

```
104 ;------Se enciende el LED F1 y se apaga el LED E4--
105 LedF1
106 ;---------Apaga Led E4----------------------------
107     LDR R1, =GPIO_PORTE_DATA
108     MOV R0, #0x0
109     STR R0, [R1]
110 ;---------Enciende Led F1-------------------------
111     LDR R1, =GPIO_PORTF_DATA
112     MOV R0, #0x2
113     STR R0, [R1]
114 ;---------Limpia el contador interno del timer-------
115     LDR R1, =GPTM_TIMER0_MICLR_R
116     LDR R0, [R1]
117     ORR R0, R0, #0x01
118     STR R0, [R1]
119     B   Ciclo
120     ALIGN
121     END
122
```

Fuente: elaboración propia.

Con esto se tendrá una interrupción por temporizador. El código completo del archivo b.s será:

Figura 108. **Código completo archivo b.s parte 1**



```
a.s | b.s* | startup.s

;---------Dirección del puerto E------------------
GPIO_PORTE_DATA     EQU 0x400243FC;
;---------Dirección del puerto F------------------
GPIO_PORTF_DATA     EQU 0x400253FC;
;-------------------------------------------------
;---------Declaración de las variables para Timer--
;-------------------------------------------------
;---------Modulo de Timer-------------------------
GPTM_RCGCTIMER_R            EQU 0x400FE604;
;---------Configuración de registro---------------
GPTM_TIMER0_CFG_R           EQU 0x40030000;
;---------Registro de Control---------------------
GPTM_TIMER0_CTL_R           EQU 0x4003000C;
;---------Indica si será periodica la interrupción-
GPTM_TIMER0_TIMAMODE_R      EQU 0x40030004;
;---------Registro de interrupción----------------
GPTM_TIMER0_INTMASK_R       EQU 0x40030018;
;---------Limpia la bandera del registro----------
GPTM_TIMER0_MICLR_R         EQU 0x40030024;
;-----------Registro de interrupción--------------
GPTM_TIMER0_RIS_R           EQU 0x4003001C;
;-----------Carga el  valor inicial al timer-------
GPTM_TIMER0_TIMAILR_R       EQU 0x40030028;

Tiempo                      EQU 15999999

        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Start
                IMPORT Inicializarpuertos
Start
;-----------Subrutina de inicialización de puertos.-
    BL Inicializarpuertos ;
;-----------Subrutina de configuración del Timer----
    BL ConfTimer
;-----------Ciclo de espera de la interrupción------
    B Ciclo

ConfTimer
;--------Se activa el modulo del Timer--------------
    LDR R1, =GPTM_RCGCTIMER_R
    LDR R0, [R1]
    ORR R0, R0, #0x1
    STR R0, [R1]
;--------Permite configurar el timer---------------
    LDR R1, =GPTM_TIMER0_CFG_R
    MOV R0, #0
    STR R0, [R1]
;--------Se establece la función periódica---------
    LDR R1, =GPTM_TIMER0_TIMAMODE_R
    MOV R0, #0x2
    STR R0, [R1]
;--------Carga el vamor del timer------------------
    LDR R1, =GPTM_TIMER0_TIMAILR_R
    LDR R0, [R1]
;--------Tiempo en alto----------------------------
    LDR R2, =Tiempo
    STR R2, [R1]
```

Fuente: elaboración propia.

Figura 109. **Código completo archivo b.s parte 2**

```
60  ;--------Permite que se activen/desactiven las interrupciones.
61      LDR R1, =GPTM_TIMER0_INTMASK_R
62      LDR R0, [R1]
63      ORR R0, R0, #0x1
64      STR R0, [R1]
65  ;--------Registro de control que inicia el conteo-------
66      LDR R1, =GPTM_TIMER0_CTL_R
67      LDR R0, [R1]
68      ORR R0,R0, #0x1
69      STR R0, [R1]
70  ;---------Valores del puerto E---------------------
71      LDR R1, =GPIO_PORTE_DATA
72      MOV R0, #0x10
73      STR R0, [R1]
74
75
76
77  Ciclo
78  ;---------Realiza una interrupcion si se ha realizado otra interrupcion------
79      LDR R1, =GPTM_TIMER0_RIS_R
80      LDR R0, [R1]
81      CMP R0, #0x1
82      BEQ LedE4
83      B Ciclo
84
85  LedE4
86  ;--------------Enciende el led E4---------------
87      LDR R1, =GPIO_PORTE_DATA
88      LDR R0, [R1]
89      CMP R0, #0x10
90      BEQ LedF1
91      MOV R0, #0x10
92      STR R0, [R1]
93  ;---------Apaga Led F1-----------------------------
94      LDR R1, =GPIO_PORTF_DATA
95      MOV R0, #0x0
96      STR R0, [R1]
97  ;---------Limpia el contador interno del timer-------
98      LDR R1, =GPTM_TIMER0_MICLR_R
99      LDR R0, [R1]
100     ORR R0, R0, #0x1
101     STR R0, [R1]
102     B   Ciclo
103
104 ;------Se enciende el LED F1 y se apaga el LED E4--
105 LedF1
106 ;---------Apaga Led E4-----------------------------
107     LDR R1, =GPIO_PORTE_DATA
108     MOV R0, #0x0
109     STR R0, [R1]
110 ;---------Enciende Led F1--------------------------
111     LDR R1, =GPIO_PORTF_DATA
112     MOV R0, #0x2
113     STR R0, [R1]
114 ;---------Limpia el contador interno del timer-------
115     LDR R1, =GPTM_TIMER0_MICLR_R
116     LDR R0, [R1]
117     ORR R0, R0, #0x01
118     STR R0, [R1]
119     B   Ciclo
120     ALIGN
121     END
```

Fuente: elaboración propia.

CONCLUSIONES

1. Los temas de programación en lenguaje ensamblador se desarrollaron para el microcontrolador TM4C123GH6PM con base en el método constructivista, como apoyo al estudiante que cursa el laboratorio de Electrónica 5.

2. Se realizaron videos conceptuales con base en las once prácticas detalladas, para una comprensión más sencilla de los temas del laboratorio de Electrónica 5.

3. Se introdujo al estudiante a los temas básicos de programación en ensamblador para el microcontrolador TM4C123GH6PM.

4. Se indicó la forma en la que se habilitan los puertos, los registros necesarios y la declaración de estos.

RECOMENDACIONES

1. Utilizar el Startup.s que se encuentra publicado junto con el código de las prácticas, de no ser así, el código no se ejecutará. Además, es importante no cambiar la ubicación del archivo, puesto que el programa dejará de funcionar si no lo encuentra en la ubicación especificada.

2. Existe una amplia cantidad de interrupciones y sus diferentes formas de configuración. En el presente trabajo se presentó a grandes rasgos los dos tipos más extensos y la configuración de uno de ellos. Se exhorta a la investigación y configuración de las demás interrupciones según necesidad de los estudiantes.

3. El IDE utilizado es KeiluVision. De la misma forma podría utilizarse el IDE de Code Composer Studio, el cual permite la configuración en ensamblador para el microcontrolador TM4C123GH6PM, pero deberán realizarse las modificaciones correspondientes.

4. Se deberá tener sumo cuidado en la configuración del registro PCTL, puesto que, si no se hace de forma correcta, puede bloquear el microcontrolador. Si se habilitaran los puertos PC0-PC3 generará un fallo en el microcontrolador y no permitirá cargar nuevos programas. Para esto se debe utilizar el programa LMFLASHPROGRAMMER y, además, el reinicio del computador, puesto que este fallo podría bloquear los puertos.

5. Consultar con el catedrático temas puntuales sobre la programación en ensamblador, ya con una base de conocimiento.

6. Crear una plataforma eficiente donde encontrar enlaces, videos, simulaciones y salas de consulta entre usuarios, que sea de fácil acceso para estudiantes de ingeniería electrónica de la escuela de Ingeniería Mecánica Eléctrica de la Universidad de San Carlos de Guatemala.

BIBLIOGRAFÍA

1. ALEGSA. *Definición de Registro de memoria*. [en línea]. <http://www.alegsa.com.ar/Dic/registro_de_memoria.php>. [Consulta 13 de diciembre de 2019.]

2. ANGULO AGUIRRE, Luis; CHÍRINOS ARMAS, Daniel. *TIC en la educación, informática y herramientas digitales*. Perú: Editorial MACRO. 37 p.

3. Arm Developer. *Cortex-A*. [en línea]. <http://infocenter.arm.com/help/topic/com.arm.doc.set.cortexa/index.html>. [Consulta 9 de diciembre de 2019.]

4. _____. *Cortex-M.* [en línea]. <http://infocenter.arm.com/help/topic/com.arm.doc.set.cortexm/index.html>. [Consulta 9 de diciembre de 2019.]

5. _____. *Cortex-R*. [en línea] <http://infocenter.arm.com/help/topic/com.arm.doc.set.cortexr/index.html>. [Consulta 9 de diciembre de 2019.]

6. _____. *DCD and DCDU.* [en línea]. <http://infocenter.arm.com/help/topic/com.arm.doc.dui0489c/Babbfcga.html>. [Consulta 18 de diciembre de 2019.]

7. arm Keil. *ADD*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289861747.htm>. [Consulta 12 de enero de 2020.]

8. _____. *ALIGN*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290002364.htm>. [Consulta 16 de diciembre de 2019.]

9. _____. *AREA*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290002714.htm>. [Consulta 16 de diciembre de 2019.]

10. _____. *B*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289863797.htm>. [Consulta 9 de enero de 2020.]

11. _____. *BIC*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289864906.htm>. [Consulta 20 de enero de 2020.]

12. _____. *BL.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289865686.htm>. [Consulta 9 de enero de 2020.]

13. _____. *CMP and CMN.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289868786.htm>. [Consulta 12 de enero de 2020.]

14. _____. *Comparison of condition code meanings in integer and floating-point code*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1359731162080.htm>. [Consulta 11 de enero de 2020.]

15. _____. DCB. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290005584.htm>. [Consulta 16 de diciembre de 2019.]

16. _____. *DCFD and DCDFU.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290006584.htm>. [Consulta 18 de diciembre de 2019.]

17. _____. *END*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290008253.htm>. [Consulta 20 de diciembre de 2019.]

18. _____. *EQU.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290008953.htm>. [Consulta 16 de diciembre de 2019.]

19. _____. *EXPORT and GLOBAL.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290016692.htm>. [Consulta 9 de enero de 2020.]

20. _____. *IMPORT and EXTERN.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290016692.htm>. [Consulta 20 de diciembre de 2019.]

21. _____. *LDR.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289874275.htm>. [Consulta 20 de enero de 2020.]

22. _____. *LTORG.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361290018422.htm>. [Consulta 9 de enero de 2020.]

23. _____. *MUL.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289882394.htm>. [Consulta 20 de enero de 2020.]

24. _____. *STR.* [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289907270.htm>. [Consulta 20 de enero de 2020.]

25. _____. *SUB*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_dom1361289908389.htm>. [Consulta 15 de enero de 2020.]

26. _____. *VMRS*. [en línea]. <http://www.keil.com/support/man/docs/armasm/armasm_pge1423836177994.htm>. [Consulta 12 de enero de 2020.]

27. AZERYA. *Memory instructions: load and store*. [en línea]. <https://azeria-labs.com/memory-instructions-load-and-store-part-4/>. [Consulta 20 de enero de 2020.]

28. BAI, Ying. *Practical Microcontroller Engineering with ARM Technology*. Estados Unidos: Editorial John Wiley y Sons, Ltd. 987 p.

29. BARRIENTOS ROJAS, David Josué. *Desarrollo del curso introducción al diseño de sistemas embebidos, utilizando el controlador tm4c123gh6pm como actualización del laboratorio de microcontroladores.* Trabajo de graduación de Ing. Electrónica. Universidad de San Carlos de Guatemala, Facultad de Ingeniería, 2017. 128 p.

30. CABANES, Nacho. *Directivas del procesador*. [en línea]. <http://www.aprendeaprogramar.com/mod/resource/view.php?id=666>. [Consulta 14 de diciembre de 2019.]

31. CALERO PEREZ, Mavilo. *Constructivismo pedagógico. Teorías y aplicaciones básicas.* México: Alfaomega Grupo Editor S.A., 176 p.

32. Caleroadrina93. *Entorno de desarrollo integrado (IDE).* [en línea]. <https://caleroadrian93.wordpress.com/2014/01/24/entorno-de-desarrollo-integrado-ide/>. [Consulta 13 de diciembre de 2019.]

33. CRUZ MEDINA, Marie Chantelle. A*nálisis arquitectónico de perfiles cortex-m y cortex-a en procesadores arm y diseño de guía introductoria en su programación de bajo nivel con lenguaje ensamblador.* Trabajo de graduación de Ing. Electrónica. Universidad de San Carlos de Guatemala, Facultad de Ingeniería, 2019. 454 p.

34. Culturacion. *¿Qué es un compilador?* [en línea]. <http://culturacion.com/que-es-un-compilador/>. [Consulta 14 de diciembre de 2019.]

35. Departamento de Informática, Universidad de Valladolid. *Modos de direccionamiento.* [en línea]. <https://www.infor.uva.es/~bastida/OC/modos.pdf>. [Consulta 11 de diciembre de 2019.]

36. Department of Computer Science and Engineering IIT Bombay. *Tiva™ TM4C123GH6PM Microcontroller*. [en línea]. <https://www.cse.iitb.ac.in/~erts/html_pages/Resources/Tiva/tm4c123gh6pm-Datasheet.pdf>. [Consulta: 1 de marzo de 2020.]

37. Ecured. *ARM.* [En Línea]. <https://www.ecured.cu/ARM>. Consulta 9 de diciembre de 2019.

38. _____. *IDE de Programación*. [en línea]. <https://www.ecured.cu/IDE_de_Programaci%C3%B3n#Caracter.C3.ADsticas>. [Consulta 13 de diciembre de 2019.]

39. EdWiki. *TM4C123 timer programming*. [en línea]. <http://shukra.cedt.iisc.ernet.in/edwiki/EmSys:TM4C123_Timer_Programming>. [Consulta: 9 de marzo de 2020.]

40. HERNÁNDEZ, Sherlin. *Arquitectura de microcontroladores.* [en línea]. <https://sherlin.xbot.es/microcontroladores/introduccion-a-los-microcontroladores/arquitectura-de-microcontroladores>. [Consulta 10 de diciembre de 2019.]

41. Lenguaje ensamblador. *Aprende Lenguaje Ensamblador*. [en línea] <http://lenguaje-ensamblador.blogspot.com/2012/09/historia-del-lenguaje-ensamblador.html>. [Consulta 10 de diciembre de 2019.]

42. Microcontrollers Tips. *What are compilers, translators, interpreters, and assemblers?* [en línea]. <https://www.microcontrollertips.com/compilers-translators-interpreters-assemblers-faq/>. [Consulta 14 de diciembre de 2019.]

43. Punto Flotante, S.A. *Manejo de interrupciones en el timer 0, en lenguaje ensamblador, con el microcontrolador 18F2550*. [en línea]. <http://www.puntoflotante.net/INTERRUPCIONES-18F2550-TIMER-0.htm>. [Consulta 1 de marzo de 2020.]

44. Red Hat. *El concepto de IDE.* [en línea]. <https://www.redhat.com/es/topics/middleware/what-is-ide>. [Consulta 13 de diciembre de 2019.]

45. ROUSE, Margaret. *Program counter.* [en línea]. <https://whatis.techtarget.com/definition/program-counter>. [Consulta 13 de diciembre de 2019.]

46. Sonoma. *Chapter 3 ARM Assembly*. [en línea]. <https://web.sonoma.edu/users/f/farahman/sonoma/courses/es310/310_arm/lectures/Chapter_3_Instructions_ARM.pdf>. [Consulta 16 de diciembre de 2019.]

47. Texas Instrument. *Tiva™ C Series TM4C1294 Connected LaunchPad Evaluation Kit EK-TM4C1294XL User's Guide*. [en línea]. <http://www.ti.com/lit/ug/spmu365c/spmu365c.pdf>. [Consulta 9 de diciembre de 2019.]

48. Universidad Autónoma de Barcelona. *La teoría de las inteligencias múltiples.* [en línea]. <http://bioinformatica.uab.cat/base/documents/genetica_gen/portfolio/La%20teor%C3%ADa%20de%20las%20Inteligencias%20m%C3%BAltiples%202016_5_25P23_3_27.pdf>. [Consulta 24 de septiembre de 2019.]

49. Universidad Autónoma del Estado de Hidalgo. *Expresiones de punto flotante para ensamblador*. [en línea]. <http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro20/311_expresiones_de_punto_flotante_para_ensamblador.html>. [Consulta 13 de diciembre de 2019.]

50. Universidad de Valencia. *Entornos virtuales de formación.* [en línea]. <https://www.uv.es/bellochc/pedagogia/EVA1.wiki?4>. [Consulta 24 de septiembre de 2019.]

51. Universidad del País Vasco. *Directivas de control de segmento.* [en línea] <http://www.sc.ehu.es/sbweb/webcentro/automatica/web_avr/archivos/Ensamblador%20AVRs/directivas/directivas_control_segmentos.htm>. [Consulta 16 de diciembre de 2019.]

52. _____. *Directivas de ensamblador*. [en línea]. <http://www.sc.ehu.es/sbweb/webcentro/automatica/web_avr/archivos/Ensamblador%20AVRs/directivas/directivas_de_ensamblador.htm#Arriba>. [Consulta 14 de diciembre de 2019.]

53. _____. *La clase matriz y la clase vector*. [en línea]. <http://www.sc.ehu.es/sbweb/fisica/cursoJava/numerico/matrices/matriz/matriz.htm>. [Consulta 25 de febrero de 2020.]

54. _____. *Mnemónico.* [en línea]. <http://www.sc.ehu.es/sbweb/webcentro/automatica/WebCQMH1/PAGINA%20PRINCIPAL/PROGRAMACION/LENGUAJES%20DE%20PROGRAMACION/MNEMONICO/mnemonico.htm>. [Consulta 25 de febrero de 2020.]

55. Universidad Marcelino Champagnat. *Contenidos declarativos (factuales, conceptuales), procedimentales y actitudinales*. [en línea]. <http://umch.edu.pe/arch/hnomarino/58_Contenidos%20declarativos%20procedimentales%20y%20actitudinales.pdf>. [Consulta 25 de septiembre de 2019.]

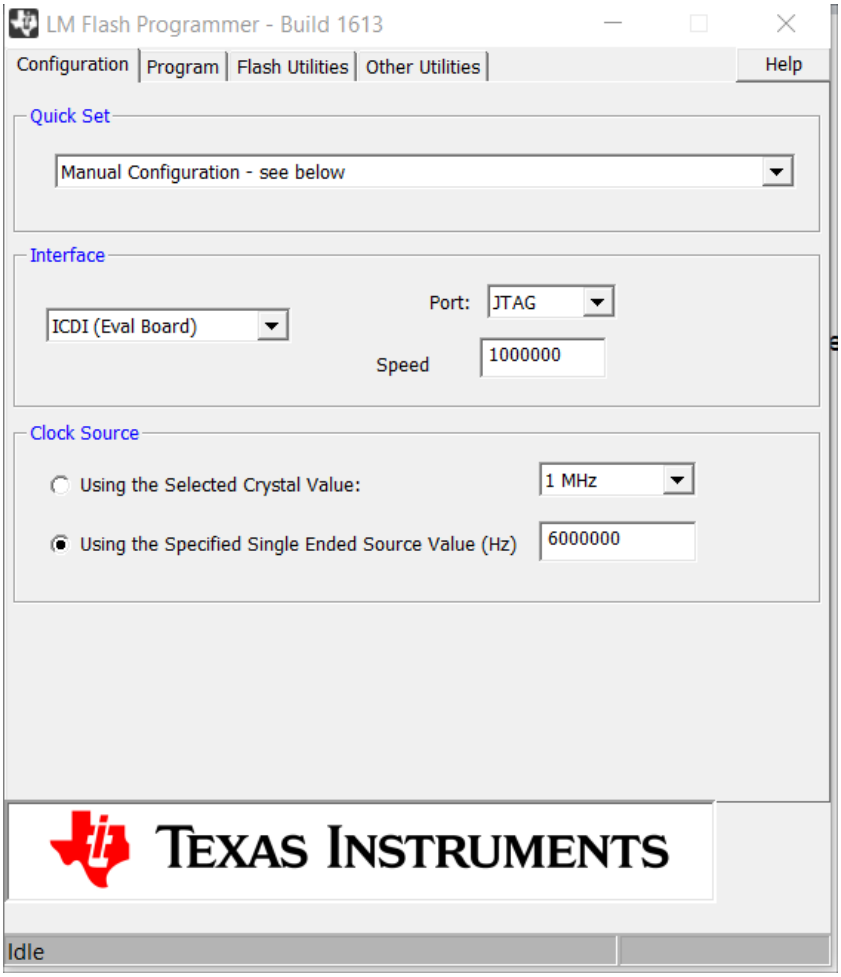
56. Universidad Politécnica de Valencia. *Temporización mediante el temporizador del sistema SysTick en microcontroladores ARM Cortex-M.* [en línea]. <https://riunet.upv.es/bitstream/handle/10251/32259/Art%C3%ADculo%20docente%20temporizaci%C3%B3n%20mediante%20SysTick%20ARM%20Cortex-M.pdf?sequence=3>. [Consulta 2 de diciembre de 2019.]

57. VALVERDE VILLARÁN, Andres. *Sistema de desarrollo PIC18F452.* [en línea]. <http://bibing.us.es/proyectos/abreproy/11301/fichero/Memoria%252FCap%C3%ADtulo+2.pdf>. [Consulta 10 de diciembre de 2019.]

APÉNDICE

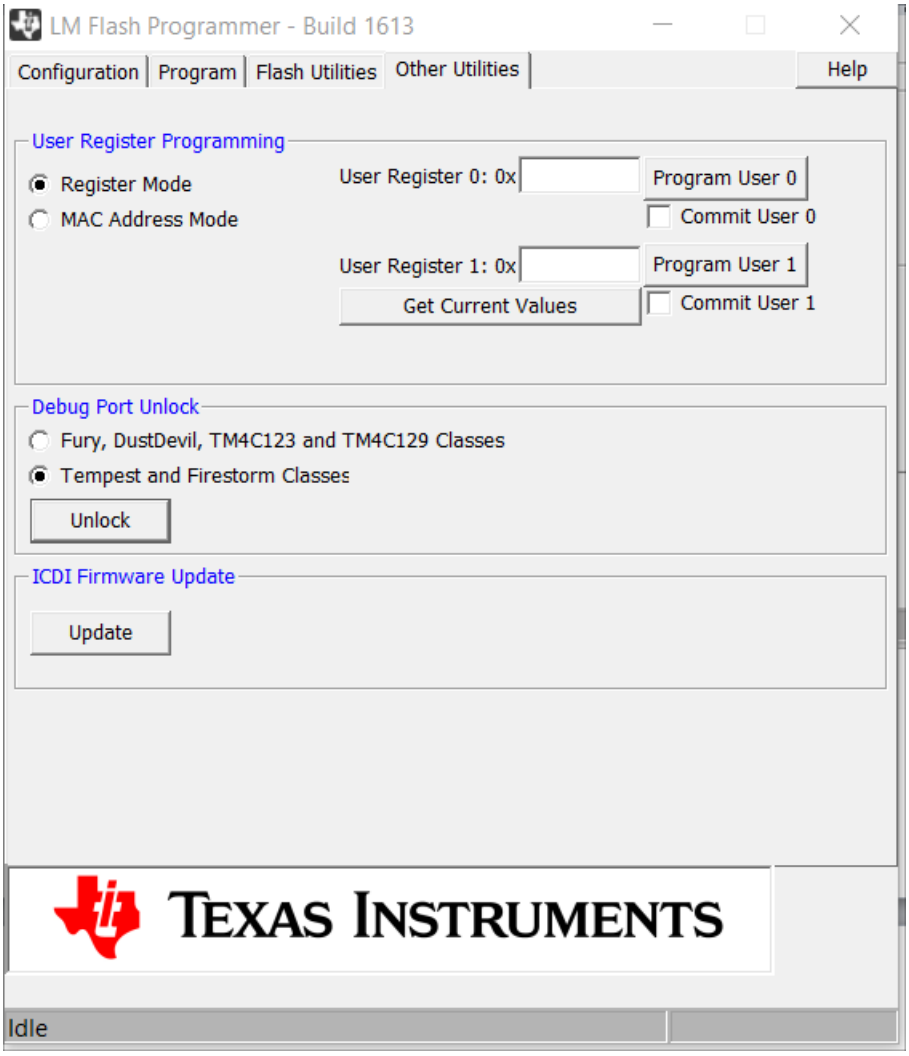
Apéndice 1. **Desbloquear una Tiva**

Si el microcontrolador se llegara a bloquear, se debe utilizar el programa LMFLASHPROGRAMMER (Descarga: http://www.ti.com/tool/LMFLASHPROGRAMMER) para resetear a valores de fábrica el microcontrolador.



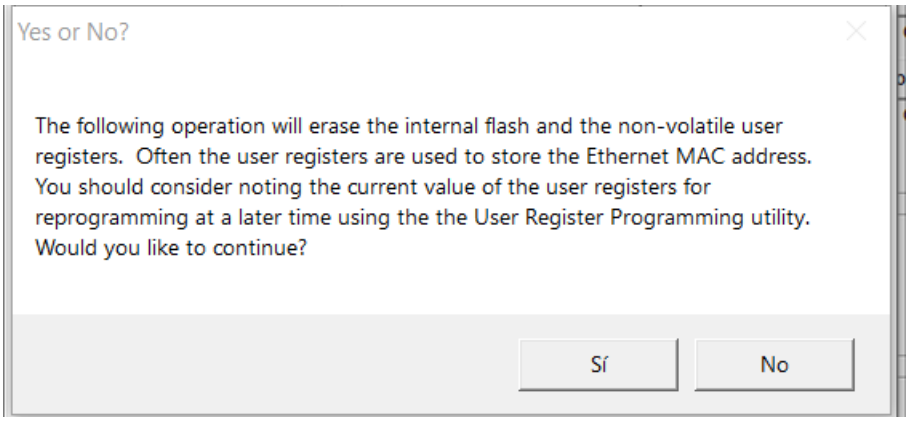
Continuación del apéndice 1.

En la viñeta de Other Utilities se deberá seleccionar en el área de User Register Programming, Register Mode y en el área de Debug Port Unlcok, Tempest and Firestorm Classes.

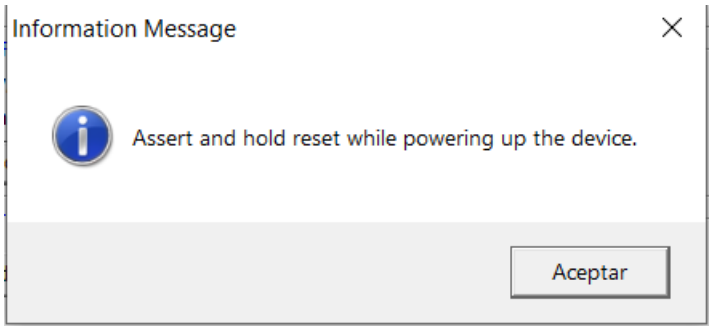


Se deberá presionar el botón Unlock y seguir los pasos que indicará el programa.

Continuación del apéndice 1.

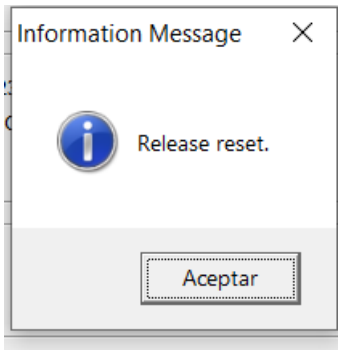


Aparecerá el mensaje de si queremos que la memoria flash y los registros no volaties sean borrados. Se presiona el botón sí.

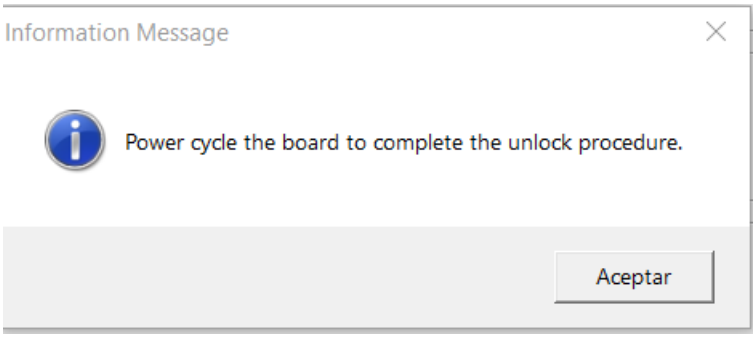


Presionar el botón de reset de la tiva mientras se enciende y se apaga el microcontrolador, habiendo hecho este paso presionar Aceptar.

Continuación del apéndice 1.



Posteriormente indicará que se debe soltar el botón reset, se deberá presionar aceptar habiendo soltado el botón.

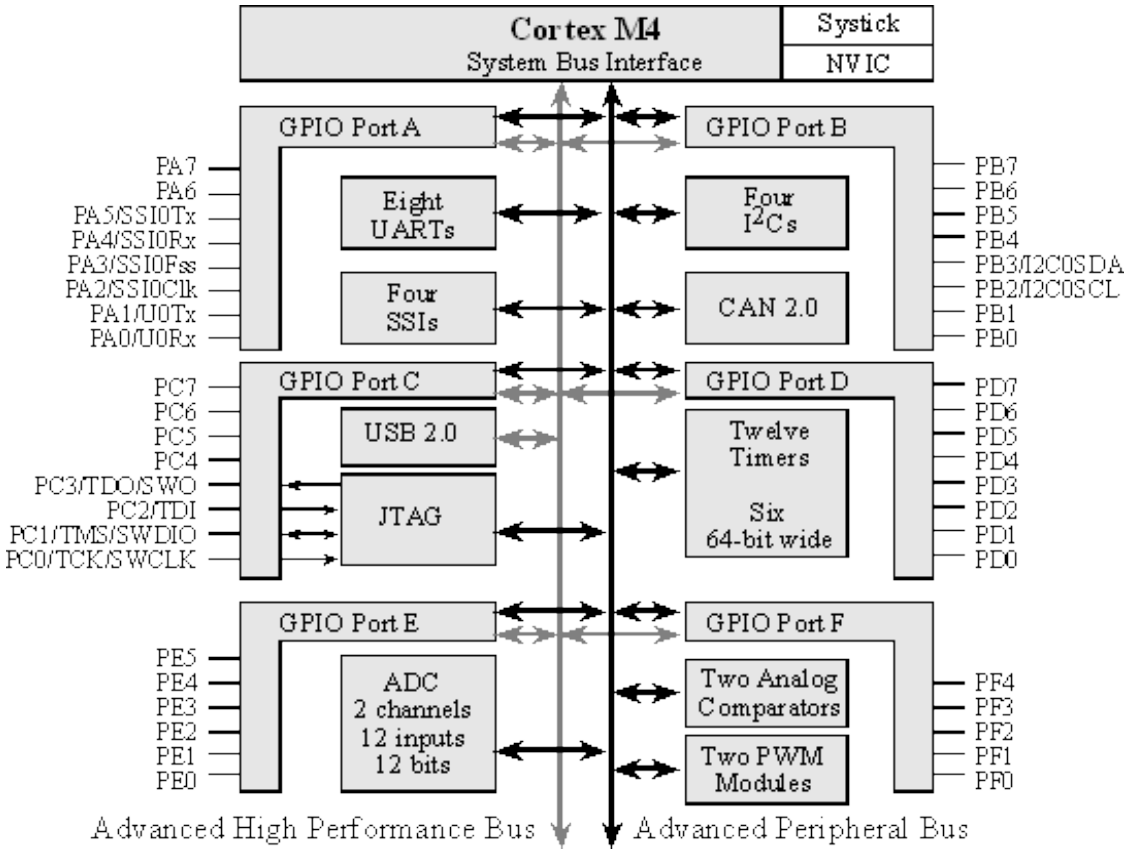


Se deberá apagar y encender el microcontrolador, presionar Aceptar y con lo que se habrá desbloqueado el microcontrolador.

Fuente: elaboración propia.

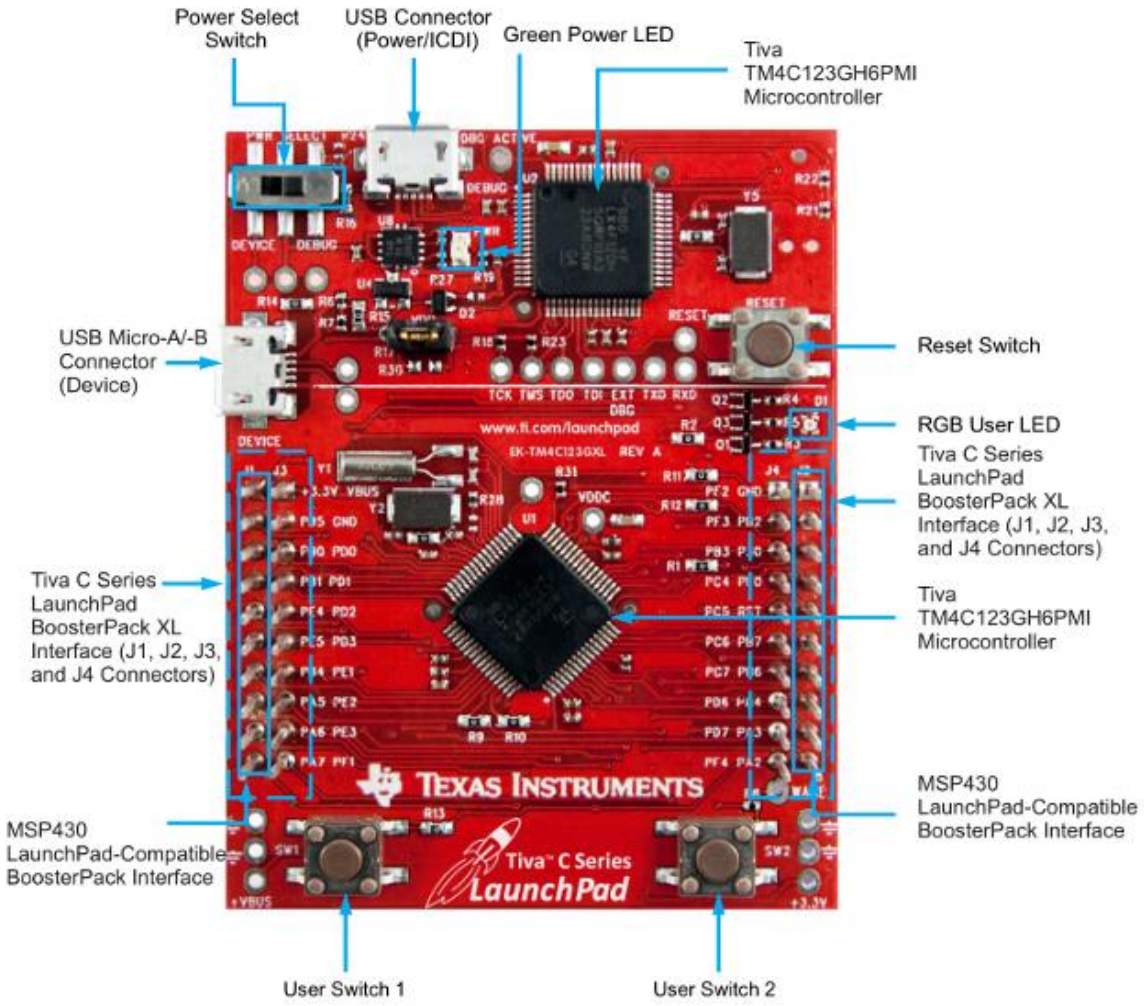
ANEXOS

Anexo 1. **Pines I/O del microcontrolador TM4C123GH6PM**



Fuente: VALVANO y Jonathan. YERRABALLI, Ramesh. *Embedded Systems - Shape The World*. http://users.ece.utexas.edu/~valvano/Volume1/E-Book/Appendix.htm. Consulta: marzo de 2020.

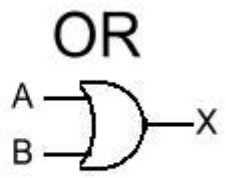
Anexo 2. **Tarjeta de desarrollo**



Fuente: NCB. *Conociendo el Microcontrolador TM4C123GH6PM, un ARM Orientado para Automatización, de Texas Instruments. (MIC013S)*. https://www.incb.com.mx/index.php/articulos/78-microcontroladores-y-dsps/1812-conociendo-el-microcontrolador-tm4c123gh6pm-un-arm-orientado-para-automatizacion-de-texas-instruments-mic013s. Consulta marzo de 2020.

Anexo 3. **Tabla de verdad compuerta OR**

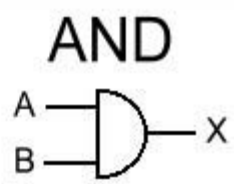
Tabla de verdad OR		
A	**B**	**X**
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**1**



Fuente: Electrónica Digital. *Practica 1. Simulación de tablas de verdad de compuertas lógicas.* https://noloelecdig.wordpress.com/2014/09/09/practica-1-simulacion-de-tablas-de-verdad-de-compuertas-logicas/#more-2. Consultado Marco 2020.

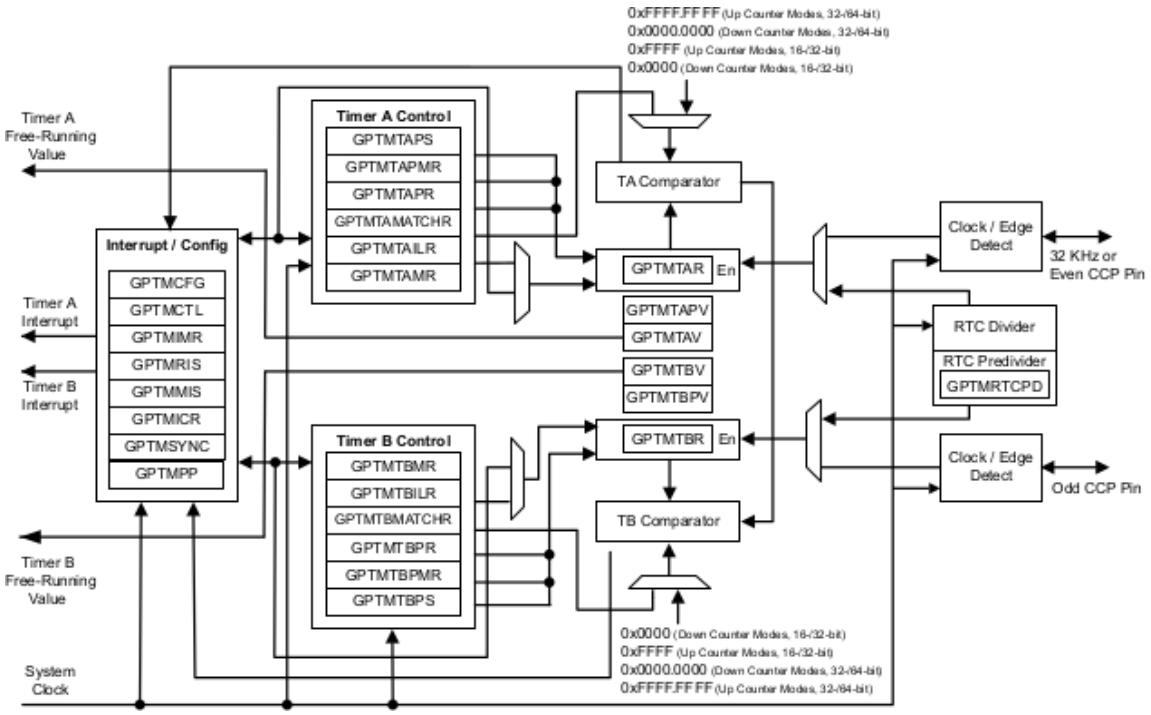
Anexo 4. **Tabla de verdad compuerta AND**

Tabla de verdad AND		
A	**B**	**X**
0	**0**	**0**
0	**1**	**0**
1	**0**	**0**
1	**1**	**1**



Fuente: Electrónica Digital. Practica 1. Simulación de tablas de verdad de compuertas lógicas. https://noloelecdig.wordpress.com/2014/09/09/practica-1-simulacion-de-tablas-de-verdad-de-compuertas-logicas/#more-2. Consultado Marco 2020

Anexo 5. GPTM Module Block Diagram

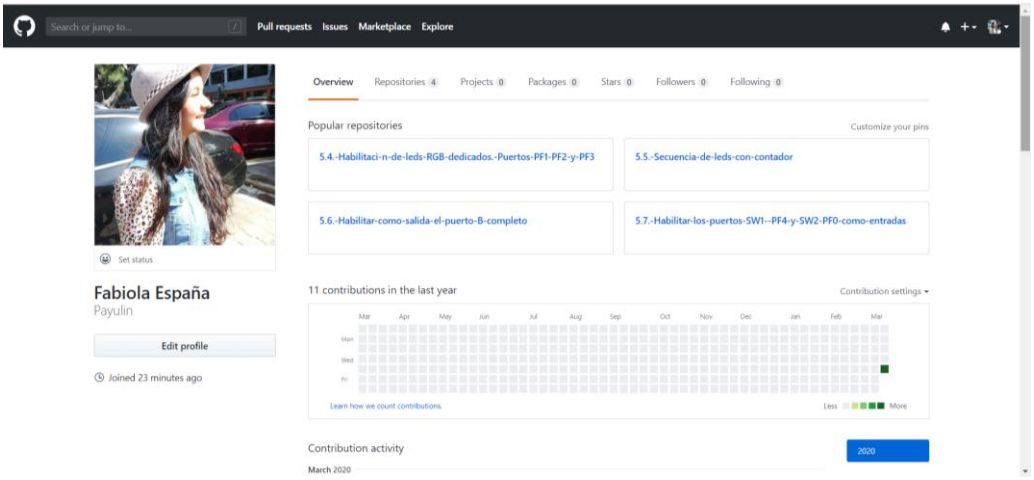


Fuente: EdWiki. *TM4C123 Timer Programming*. http://shukra.cedt.iisc.ernet.in/edwiki/EmSys:TM4C123_Timer_Programming. Consulta: 9 de marzo de 2020.

Anexo 6. **Página del laboratorio de electrónica 5, Ingeniería, USAC**

Fuente: Laboratorio de electrónica*. Electrónica 5.* http://labelectronica.weebly.com/electronica5.html Consulta marzo 2020

Anexo 7. **Repositorio con los códigos de las prácticas**



Fuente: GitHub. *Fabiola España*. https://github.com/Payulin. Consulta marzo 2020.