UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Desarrollo del Software de Control para un Sistema de Adquisición de Imágenes Digitales de Microscopía de Fluorescencia

TESIS

Que para obtener el título de

Ingeniero Eléctrico Electrónico

P R E S E N T A

Brian Jiménez Moedano

DIRECTOR DE TESIS

Dra. Citlali Trueta Segovia

Ciudad Universitaria, Cd. Mx., 2016

"Dios…
concédeme serenidad para aceptar las cosas que no puedo cambiar,
valor para cambiar las que sí puedo,
y sabiduría para distinguir la diferencia."

Agradecimientos:

De primera instancia agradezco enormemente al personal del Instituto Nacional de Psiquiatría "Ramón de la Fuente Muñiz", que hizo posible la prestación de las instalaciones y equipo necesario para que este proyecto de tesis se postulara y desarrollara. En especial y con mucho cariño a la Dra. Citlali Trueta, quien me brindó su total confianza y apoyo desde el primer día que llegué; así como a mis compañeros y compañeras del laboratorio de Neurofisiología, llevándome un cálido recuerdo de todos y cada uno de ellos.

También agradezco al personal docente, administrativo, técnico y a todos aquellos que de alguna u otra manera contribuyen a que la Universidad Nacional Autónoma de México mantenga sus puertas abiertas a las personas que como yo, deseamos un mejor futuro para nosotros y los nuestros. En especial al M.I. Juan Manuel Gómez quien me brindó desinteresadamente su apoyo y orientación académica cuando la necesite, y a quien considero un académico ejemplar. Y con cierta melancolía en el corazón, a todas las personas que se convirtieron en más que compañeros de escuela; a mis amigos y amigas, que convivieron conmigo en el salón de clases, en las jardineras del edificio principal de Ingeniería, en las islas o en cualquier rincón de ciudad universitaria, con cierto temor a dejar fuera a alguno de ellos, solamente diré: ustedes saben quiénes son.

Hago una mención dentro de este espacio a mi amigo Constantino Hinojosa quien me ofreció su amistad sincera, y junto a todos mis demás camaradas de la hacienda me enseñaron una manera muy distinta de ver la vida, una que jamás hubiera podido encontrar en la universidad. Gracias desde el fondo de mi corazón.

A mi padre Luis Jiménez quien a base de numerosos y amorosos sacrificios, me brindó lo necesario para convertirme en un gran hombre, y a quien espero honrar y enorgullecer durante toda mi vida. A mi madre Martha Moedano quien me dio la vida y me educó con el amor de su corazón, su amor de madre jamás me ha abandonado ni un solo día de mi vida. A mis hermanos Luigi y Lorenie a quienes admiro y estoy orgulloso de sus logros, cada uno y a su manera, enseñándome y motivándome a crecer en la vida. A mis abuelos Jorge y Loreto quienes me abrieron las puertas de su hogar y me cuidaron durante mis años de estudiante, no pude haberlo hecho sin ustedes.

Al amor de mi vida Denise, quien después de conocer mis virtudes y defectos, decidió amarme y aceptarme, una estudiante ejemplar, un novia amorosa, y una mujer maravillosa. A la Señora Rosa y a su familia quien me brindó su apoyo y su confianza durante mis largos años de estudiante. A mi amigo Ulises quien me enseño el significado de la palabra mejores amigos. A todos y cada una de las personas que por falta de buena memoria y espacio no tuvieron una mención personal, pero que forman parte de mi vida y que comparten este gran logro conmigo.

… Pero sobre todo a Dios, quien me creó y amó, desde el comienzo de los tiempos. A Dios, le debo todo, a Quien pedí grandeza y me concedió debilidad, para concebir humildad en mi corazón.

ÍNDICE DE CONTENIDO

Capítulo 1

Introducción

El sistema nervioso utiliza sustancias bioquímicas llamadas neurotransmisores para comunicar información entre las células que lo componen, y así actuar en respuesta a los distintos estímulos físicos o fisiológicos tanto internos como externos del cuerpo. La serotonina es un neurotransmisor fundamental para la regulación de la conducta y las emociones humanas, la cual es liberada en regiones sinápticas y extrasinápticas de neuronas específicas. El estudio de los mecanismos que regulan la liberación de serotonina requiere realizar algunos experimentos en los que se utilizan colorantes fluorescentes que permiten estudiar fenómenos biológicos como la secreción dinámica o la concentración de calcio dentro de las neuronas.

El estudio en tiempo real de estos procesos requiere de la adquisición de secuencias de imágenes de microscopía de fluorescencia con una alta resolución temporal en células vivas. La ingeniería eléctrica y electrónica desempeña un papel muy importante en la solución de estas necesidades, ya que integrando el uso de hardware especializado con la potente capacidad de procesamiento de las computadoras modernas de fácil acceso, ha desarrollado una metodología de instrumentación virtual capaz de implementar soluciones asequibles con un alto grado de especialización a través de modernas técnicas de programación, automatización y control de dispositivos mecánicos, eléctricos y electrónicos. Esto elimina gran parte de las limitaciones que los centros de investigación sin acceso a grandes fuentes de financiamiento tienen para adquirir instrumentos comerciales de medición; brindándoles la oportunidad de desarrollar su propio ensamble, programación e implementación.

En este proyecto de tesis se desarrolló un software de control a través de una metodología de instrumentación virtual en LabVIEW (el cual se llamó **FIMAQ,** por sus siglas en inglés **Fluorescence Image Acquisition**); para integrarlo a un equipo de adquisición de imágenes digitales de microscopía de fluorescencia proporcionado por el Instituto Nacional de Psiquiatría “Ramón de la Fuente Muñiz”.

1.1 Descripción del Contexto

El proyecto de tesis aquí expuesto, se desarrolló dentro de las instalaciones del laboratorio de neurofisiología, en el área de investigaciones en neurociencias, del Instituto Nacional de Psiquiatría "Ramón de la Fuente Muñiz" ubicado en la zona de hospitales de Tlalpan, en la Ciudad de México.

Dentro del programa de investigación científica que se desarrolla en el laboratorio de neurofisiología, se encuentra el uso de técnicas de microscopía de fluorescencia para apoyar estudios de neurobiología relacionados con la secreción sináptica y extrasináptica de neurotransmisores y la concentración de calcio intracelular en neuronas identificadas del sistema nervioso de sanguijuelas (*Hirudo medicinalis*) empleadas como modelos biológicos.

Aunque se contaba con el equipo necesario para montar un sistema de adquisición de imágenes digitales de microscopía de fluorescencia (véase **Figura 1.1**), éste había estado inhabilitado debido a la falta de un software o controlador encargado de sincronizar la operación el equipo para realizar cuatro funciones básicas dentro de un experimento típico de microscopía de fluorescencia:

- Control automático de obturadores electrónicos,
- Adquisición de imágenes digitales,
- Gestión de datos y archivos informáticos, y
- Despliegue de resultados.

Figura 1.1 Vista general, del sistema de adquisición de imágenes digitales de microscopía de fluorescencia.

Fue bajo estas circunstancias y necesidades particulares del laboratorio de neurofisiología, que se planteó el proyecto para desarrollar un software de control que permita al usuario realizar estas cuatro funciones en el sistema de adquisición de imágenes digitales de microscopía de fluorescencia con el que se cuenta.

Previo a desarrollar el software de control para el sistema de adquisición de imágenes digitales de microscopía de fluorescencia, fue necesario revisar algunas cuestiones básicas sobre la investigación que se desarrolla en el laboratorio de neurofisiología, así como el funcionamiento y características generales del equipo de microscopía de fluorescencia, para así desarrollar un software de control coherente que cumpla satisfactoriamente con las funciones y necesidades básicas tanto de instrumentación como de investigación. En el siguiente capítulo se presentan estos antecedentes.

Capítulo 2

Marco Teórico

En este capítulo se describen tres conceptos básicos y fundamentales relacionados con la experimentación y la metodología del sistema de adquisición de imágenes digitales de microscopía de fluorescencia, siendo éstas las bases teóricas que sustentan el desarrollo del proyecto de tesis. Resulta de suma importancia abordarlos ya que estos aportan una mejor comprensión y son el punto de referencia clave junto con el hardware descrito en el **Capítulo 5**, al momento de establecer las características generales y específicas del software de control y su desarrollo.

2.1 Conceptos Básicos de Neurobiología

Ya sea que se hable de las emociones, la conducta, el cerebro o los procesos mentales o biológicos que ocurren en él; el vasto y complejo sistema nervioso, que engloba todos estos aspectos integrales del ser humano y otros animales, resulta una fascinante materia de estudio para científicos y académicos de diversas áreas de las llamadas neurociencias.

La Neurobiología es una disciplina dentro de las neurociencias, y una rama mayor de la Biología; especializada en estudiar las células que componen al sistema nervioso, así como la estructura de dichas células en circuitos funcionales encargados de procesar información y controlar el funcionamiento del cuerpo; sentando las bases biológicas de la conducta, las emociones y demás procesos físicos y mentales relacionados con el sistema nervioso. [2.1]

2.1.1 El Sistema Nervioso del Humano y Otros Animales

El sistema nervioso está compuesto por una vasta red de células, fundamentalmente neuronas y glía (las cuales se describen en la siguiente **Sección 2.1.2**), conectadas entre sí y con todo el cuerpo del organismo por numerosas fibras nerviosas; formando circuitos neuronales encargados de procesar información específica sobre las condiciones y el funcionamiento del organismo así como el de su entorno. En la **Figura 2.1** se puede apreciar la distribución general del sistema nervioso humano a lo largo del cuerpo. [2.1]

El sistema nervioso es estudiado en términos de su distribución en el cuerpo como:

- Sistema Nervioso Central
- Sistema Nervioso Periférico.

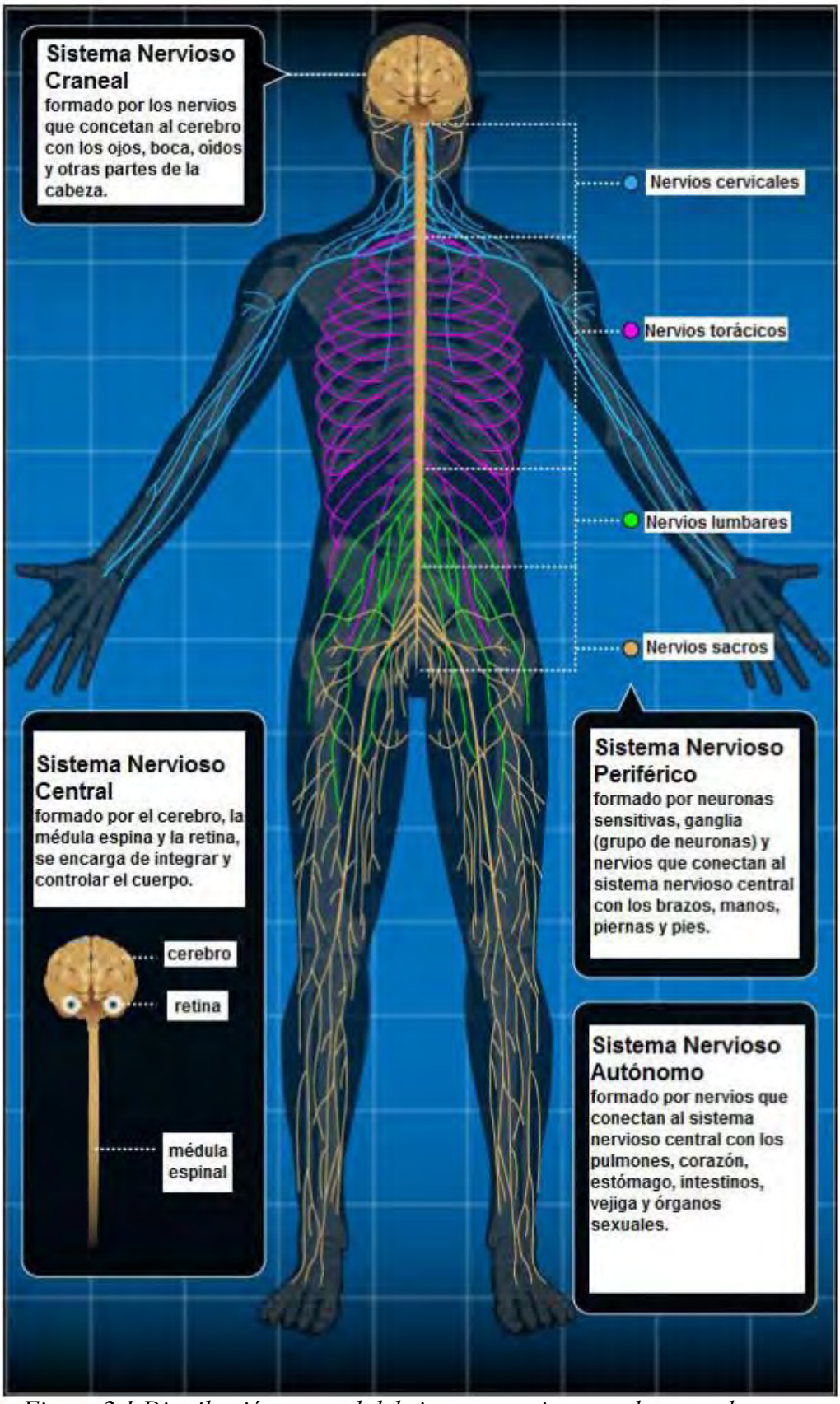


Figura 2.1 Distribución general del sistema nervioso en el cuerpo humano.

Además de esta segmentación, el sistema nervioso también es estudiado en términos de las funciones que desempeña en el organismo como:

- Sistema Sensorial
- Sistema Motor
- Sistema Asociativo

En el ser humano, el sistema nervioso central está compuesto por: cerebro, la médula espinal y diversos nervios craneales. A su vez el cerebro está compuesto por células nerviosas llamadas neuronas, las cuales se apoyan en otras células llamadas gliales para recibir, mandar, almacenar e interpretar información proveniente de todo el cuerpo en forma de impulsos eléctricos. La médula espinal es un elemento de vital importancia para el sistema nervioso central, ya que éste lo une y comunica con el sistema nervioso periférico a través de miles de nervios y numerosas neuronas sensitivas y motoras distribuidas a lo largo de las diversas extremidades, músculos y órganos del cuerpo humano; permitiendo al sistema nervioso central recibir e interpretar información proveniente tanto del cuerpo como de su entorno, actuando o dando una respuesta como consecuencia de estos estímulos. El sistema nervioso central, al asociar al sistema sensorial con el motor, constituye el sistema asociativo, responsable de ejecutar funciones de alto orden como la percepción, atención, aprendizaje, emociones, el pensamiento racional y demás procesos mentales complejos. [2.1]

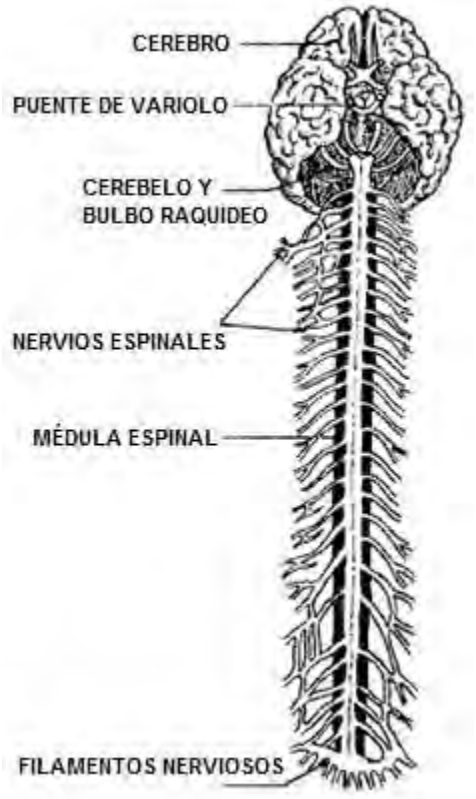


Figura 2.2 Sistema nervioso central humano.

El sistema nervioso periférico, está compuesto por un sistema nervioso somático y un sistema nervioso autónomo; ambos formados por una extensa red de nervios, neuronas sensitivas y motoras, y ganglios que lo conectan con el sistema nervioso central a través de la medula espinal. El sistema nervioso autónomo se encarga de regular procesos inconscientes como la presión sanguínea, secreción hormonal y la respiración, mientras que el sistema nervioso somático está presente en los músculos y en las células receptoras sensitivas en la piel, responsables del movimiento y la mayoría de los procesos conscientes. Los sistemas motores y sensitivos del ser humano, residen en su gran mayoría, en el sistema nervioso periférico. [2.1]

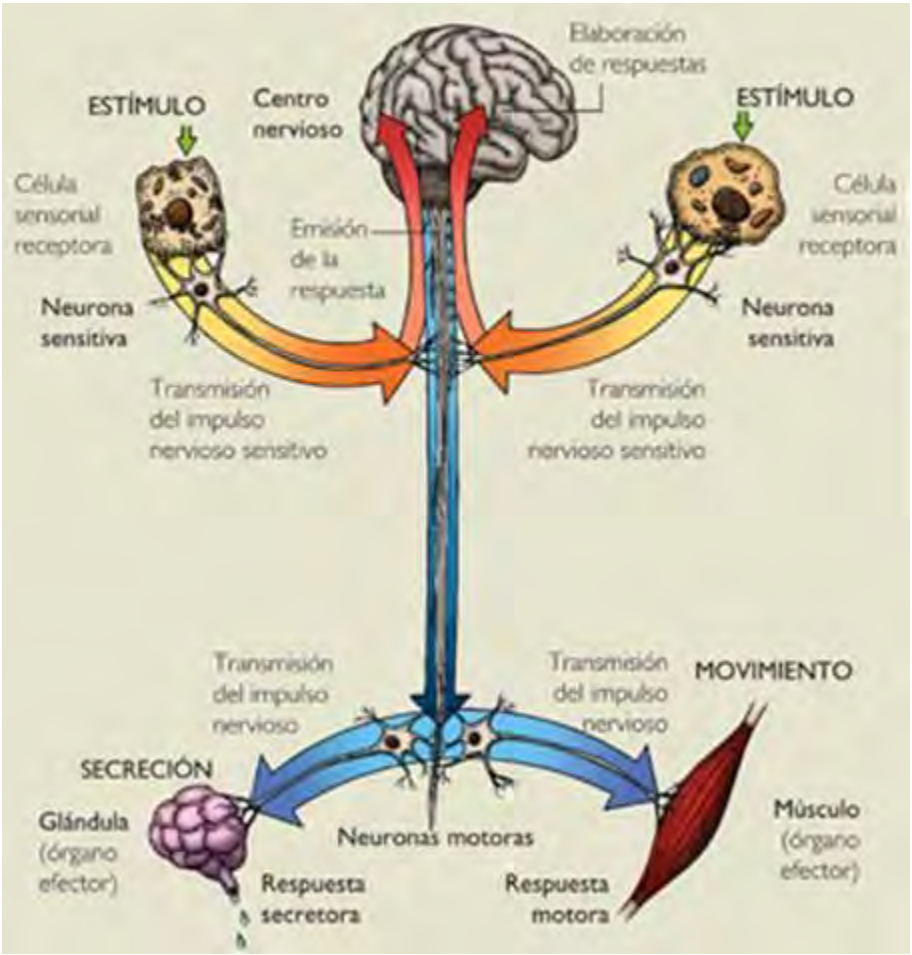


Figura 2.3 Sistema nervioso periférico humano.

En la **Figura 2.2** y **2.3** se pueden observar de manera general, la distribución y funcionamiento del sistema nervioso central y periférico humano respectivamente.

Debido a la compleja estructura y funcionalidad del sistema nervioso humano, algunos científicos del área han dedicado esfuerzos en hallar un modelo de estudio con un sistema nervioso mucho más sencillo en cuanto a estructura, y similar en cuanto a funcionalidad al humano. La sanguijuela es un animal cuyo sistema nervioso se tiene bien identificado en anatomía y fisiología, y su estructura y distribución resulta más sencilla de estudiar: posee muy pocas neuronas, las cuales son mucho más grandes que las de los mamíferos, y a

diferencia de estos, las de la sanguijuela se pueden aislar y mantener en cultivos; están localizadas en sitios estereotipados por lo que se pueden identificar fácilmente; y también poseen un repertorio de comportamientos muy limitado. Es por esto que se ha preferido como modelo de estudio en el laboratorio de neurofisiología en donde se desarrolló este proyecto de tesis. La mayor parte de las células que se desean observar y estudiar con el sistema de adquisición de imágenes digitales de microscopía de fluorescencia, provienen de los ganglios identificados del sistema nervioso de la especie conocida como *Hirudo medicinalis*.

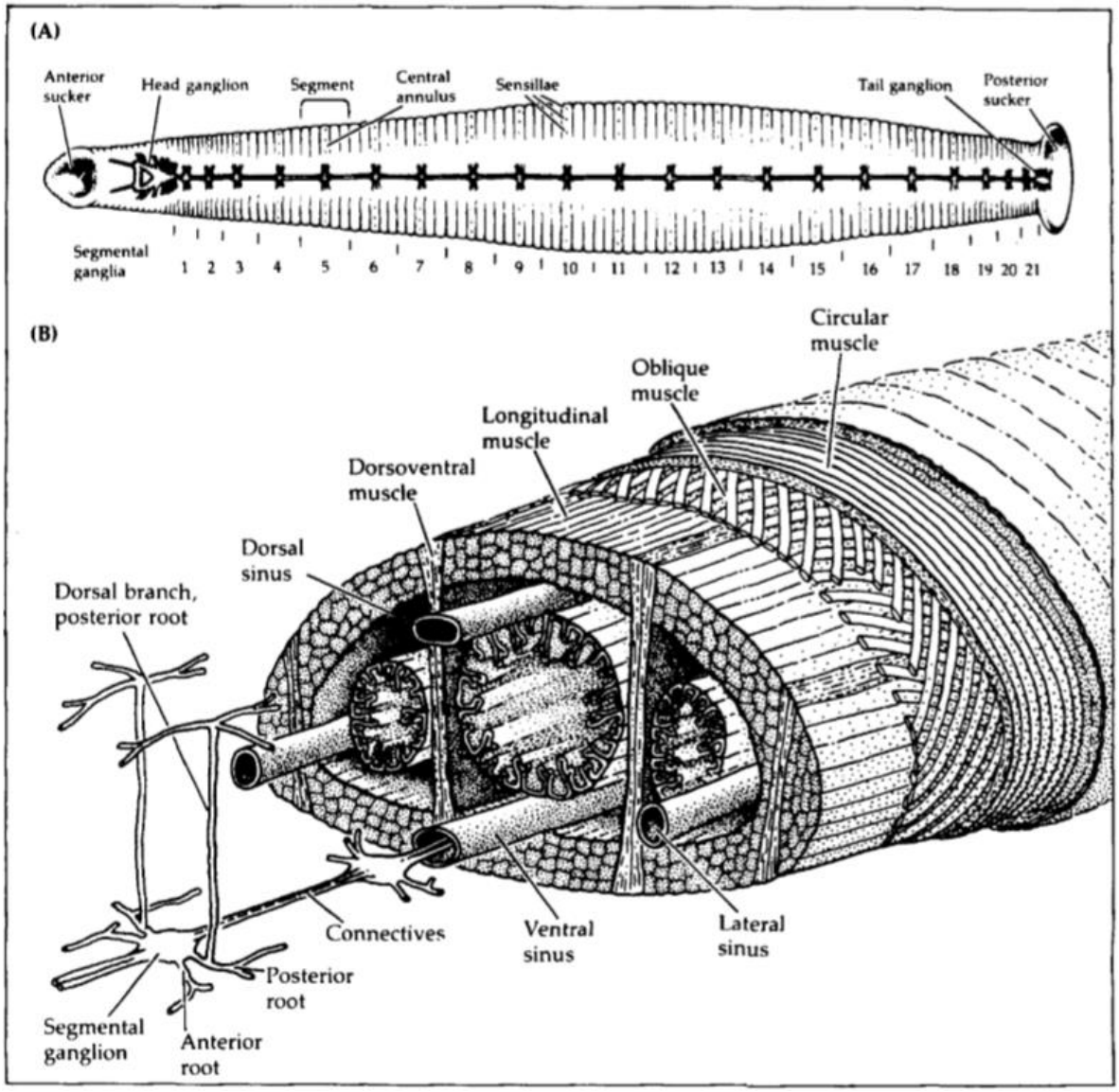


Figura 2.4 Sistema nervioso central de la sanguijuela [2.3].

En la **Figura 2.4** (A) se puede observar el sistema nervioso central de la sanguijuela, el cual consiste en una cadena de 21 ganglios segmentales, un ganglio cefálico, y un ganglio caudal. El cuerpo de la sanguijuela, está dividido en segmentos. [2.3]

(B) El cordón nervioso se halla en la parte ventral del cuerpo dentro de un seno venoso. Los ganglios se encuentran unidos entre ellos por enlaces de axones (los nervios conectivos), y además inervan la pared del cuerpo por las raíces laterales. Los músculos se encuentran acomodados en tres principales capas: circular, oblicua y longitudinal. Adicionalmente se tienen músculos dorsoventrales que achatan al animal y fibras inmediatamente debajo de la piel que lo elevan en crestas. [2.3]

2.1.2 Neuronas y Neurotransmisores

La neurona es la célula y elemento fundamental del sistema nervioso. Su composición interna es similar a cualquier otra célula conocida, contiene: un núcleo, mitocondrias, aparato de Golgi, retículo endoplásmico y demás organelos contenidos en un cuerpo celular llamado soma. De este cuerpo celular o soma, sobresalen ramificaciones llamadas dendritas y una prolongación especializada llamada axón, recubierta por una funda o capa de Mielina que funge como aislante eléctrico. Este axón termina en un botón sináptico que comunica a la primera neurona (llamada presináptica), con otra neurona adyacente (llamada postsináptica), al establecer una comunicación con las dendritas de ésta; formando así extensos circuitos neuronales en el cerebro tan complejos o sencillos como el número de dendritas y conexiones sinápticas permita la particular morfología y estructura de las neuronas implicadas. Existen neuronas con ramificaciones de dendritas tan extensas como las de un árbol de edad avanzada, y las hay con muy pocas o nulas ramificaciones, dependiendo de la región y función particular de la neurona o célula nerviosa. [2.9]

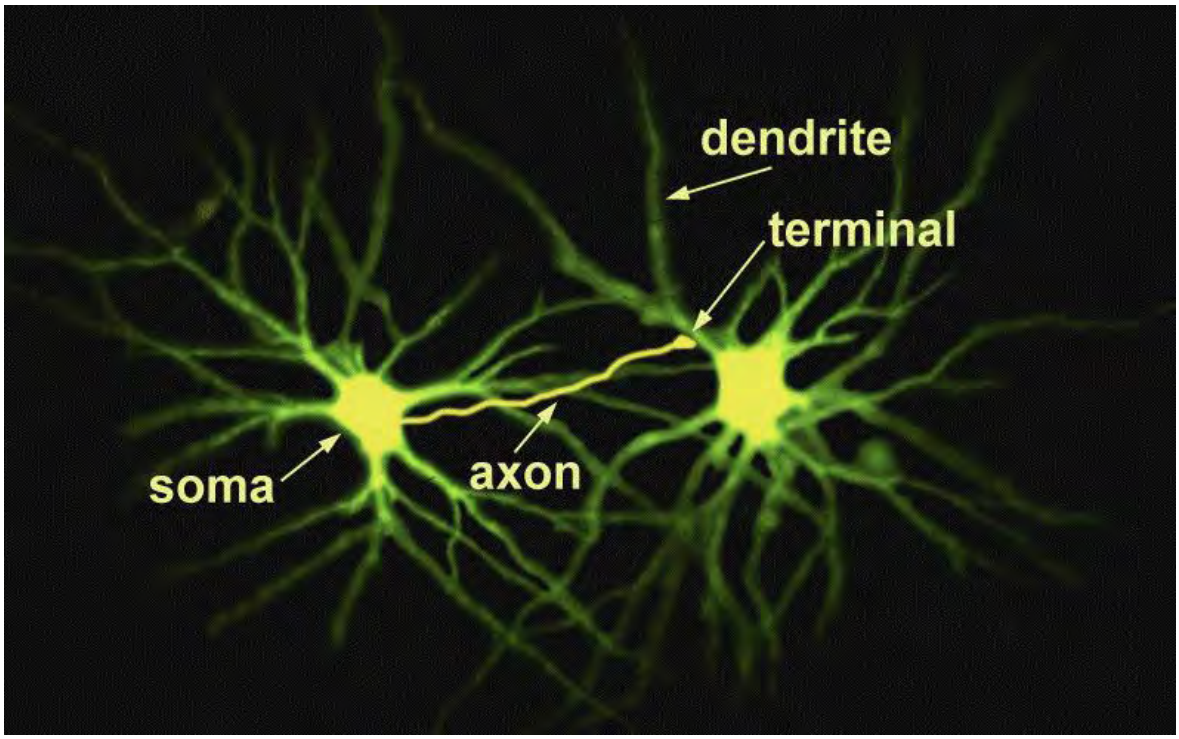


Figura 2.5 Par de neuronas inyectadas con colorantes fluorescentes. Se puede observar su anatomía básica y estructura en circuitos neuronales.

Dentro de los botones sinápticos se acumulan vesículas que contienen biomoléculas llamadas neurotransmisores, sintetizadas previamente en el soma o directamente en las terminaciones del axón, y en cuyo procesamiento suelen participan las células gliales (las cuales son células de soporte para las neuronas en el procesamiento de la información y la regulación del microambiente celular). Entre los principales neurotransmisores se encuentran: el glutamato, el GABA (ácido gamma-aminobutírico), la acetilcolina, la dopamina, la noradrenalina, la serotonina, entre otros; cada uno cumpliendo funciones específicas en el sistema nervioso que abarcan desde la salivación, reducción de la frecuencia cardiaca, respuesta sexual, micción, inhibición o actividad de estados emocionales asociados con la ira, la depresión, el estrés, el humor, el sueño, procesos mentales complejos como el comportamiento, la cognición, la motivación o el aprendizaje. Estos neurotransmisores promueven o inhiben la generación o propagación de señales eléctricas cuando son liberados por la neurona presináptica hacia la neurona postsináptica, lo que en el macro mundo se traduce en la generación o inhibición de respuestas emocionales, fisiológicas o mentales del individuo ante los estímulos internos o externos de su cuerpo. [2.10]

2.1.3 Liberación Sináptica y Extrasináptica de Neurotransmisores

Según la región anatómica de la neurona presináptica en donde se lleva a cabo, la liberación de neurotransmisores se clasifica de la siguiente manera:

- Liberación sináptica de neurotransmisores, cuando se lleva a cabo en las terminaciones del axón de la neurona presináptica sobre las dendritas de la neurona postsináptica.
- Liberación extrasináptica de neurotransmisores, cuando se lleva a cabo en regiones distintas a las identificadas para la liberación sináptica; como por ejemplo en el soma de la neurona presináptica.

En el caso de la liberación sináptica de neurotransmisores, los botones sinápticos que se encuentran al final del axón no se unen a las dendritas a través de un contacto directo como tal, sino que se encuentran separados por un estrecho espacio o hendidura sináptica, en donde se hallan multitud de proteínas extracelulares que modifican la difusión y propagación de neurotransmisores entre la neurona presináptica y postsináptica. [2.2]

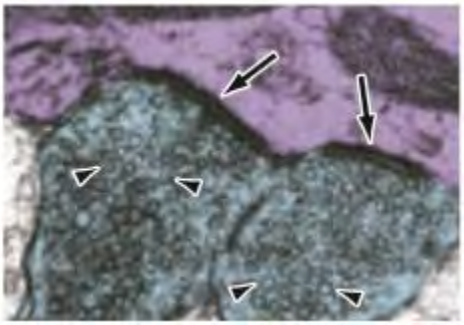


Figura 2.6 Fotografía real de un botón sináptico (azul) cargado con vesículas llenas de neurotransmisores (cabezas de flecha), formando sinapsis (flechas) con una dendrita (púrpura) [2.2].

Las neuronas codifican la información en forma de despolarizaciones breves de su membrana llamadas potenciales de acción, los cuales viajan a través del soma hasta el axón, y finalmente hasta el botón sináptico por medio de un efecto de despolarización en cadena. En el botón sináptico el potencial de acción, en conjunto con otros procesos químicos y biológicos, promueve la movilización de las vesículas a la membrana del botón y éstas se fusionan, liberando los neurotransmisores contenidos en ellas, los cuales difunden por la hendidura sináptica hasta las dendritas de la neurona postsináptica. Estos neurotransmisores promueven o inhiben la apertura de canales iónicos en la neurona postsináptica, causando una despolarización (cuando entran iones positivos acumulados en el exterior de la neurona o salen iones negativos), o una hiperpolarización (cuando entran iones negativos o salen iones positivos); de esta manera el proceso de señalización continúa propagándose a otras neuronas. La neurona postsináptica, es capaz de recibir varios estímulos a la vez; la suma total de los estímulos promotores o inhibidores causados por la liberación de neurotransmisores, darán como resultado que la neurona postsináptica aumente o disminuya su frecuencia de disparo de potenciales de acción. [2.2]

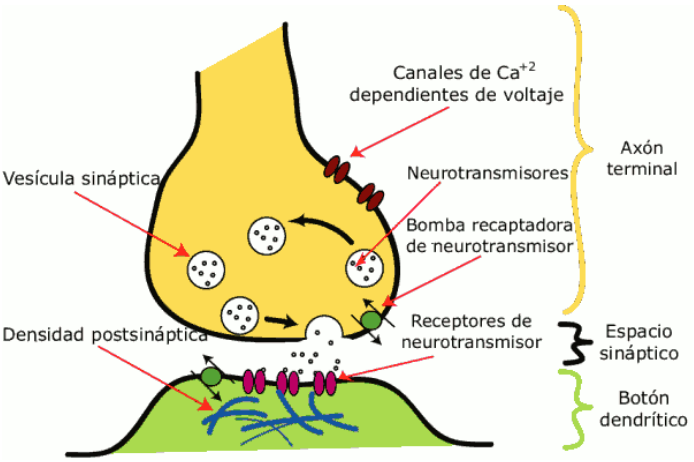


Figura 2.7 Liberación sináptica de neurotransmisores.

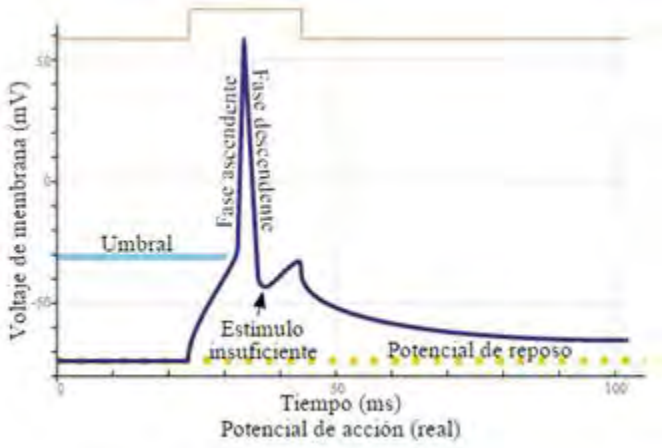


Figura 2.8 Registro real de un potencial de acción.

La liberación extrasináptica de neurotransmisores se lleva a cabo en diferentes compartimientos de la neurona ajenos a la región sináptica, pudiendo ser simultáneas estas dos clases de liberación en un mismo proceso de señalización. Cómo se produce esta liberación extrasináptica y cuál es su efecto; así como los mecanismos que la regulan y el estudio de las fuentes de calcio intracelular que la promueven; son objetos de estudio de los neurobiólogos contemporáneos. Existen muchas técnicas utilizadas para estudiar estos fenómenos biológicos en las neuronas, la microscopía de fluorescencia forma parte de esta gama de herramientas que prometen ayudar a responder a las interrogantes sobre el funcionamiento del complejo y vasto sistema nervioso. [2.2]

2.2 Microscopía de Fluorescencia

La fluorescencia es un fenómeno físico que presentan ciertos materiales o sustancias, que al ser excitadas con una fuente de luz con cierta longitud de onda (que va desde la ultravioleta, al infrarrojo) emiten luz. Según los estudios del físico británico George Stokes, la luz emitida siempre será de una longitud de onda mayor a la luz de excitación, esto debido a que la energía absorbida por la sustancia fluorescente no es disipada en su totalidad en forma de luz.

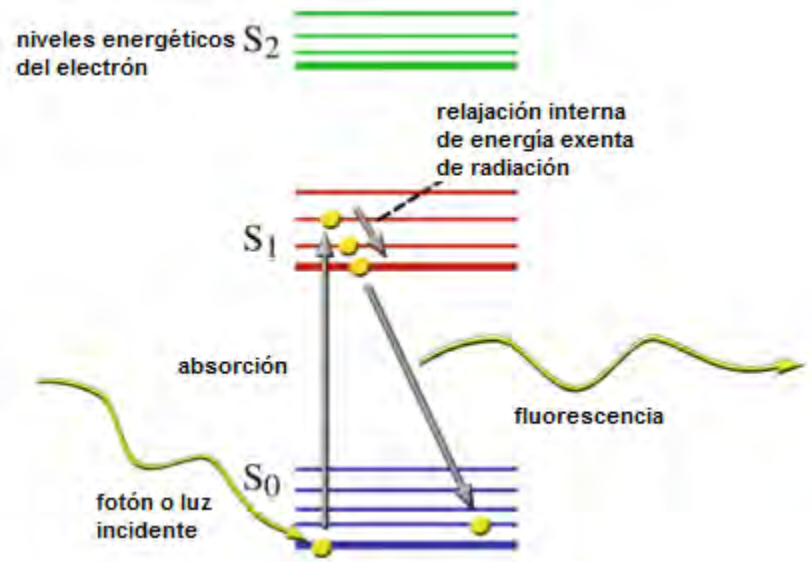
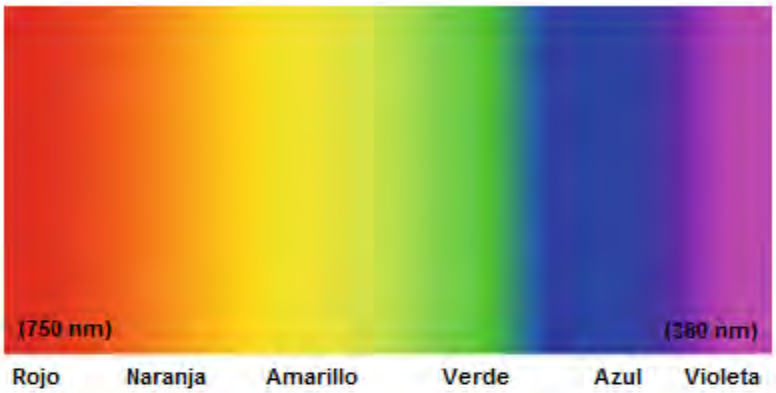


Figura 2.9 Diagrama de excitación y fluorescencia (conocido también como diagrama Jablonksi).

La fluorescencia es una propiedad presente en diversas sustancias halladas en la naturaleza; sin embargo el hombre ha fabricado multitud de tinturas y sustancias (conocidas comúnmente como fluoróforos o fluorocromos) que presentan estas propiedades fluorescentes y obedecen a longitudes de onda de excitación y emisión muy particulares y predefinidas al momento de su fabricación, utilizadas para diversos fines de investigación, así como muchas otras aplicaciones en la industria y la medicina entre otros.



Fluoróforo	Excitación máxima (nm)	Emisión máxima (nm)	Diferencia
Hoechst 33342	352	461	109
DAPI	358	461	103
Lucifer Yellow VS	430	535	105
FITC	490	523	33
Alexa Fluor® 488	495	519	24
Propidium iodide	536	617	81
Cy 3	548	562	14
DiI	549	565	16
TRITC	557	576	19
Lissamine rhodamine B	575	595	20
Texas Red	596	615	19
Cy 5	649	669	20
Alexa Fluor® 647	650	668	18

Figura 2.10 Espectro electromagnético visible junto a una tabla de relación con la longitud de onda de excitación, emisión y su diferencia de algunos fluoróforos [2.4].

La microscopía de fluorescencia aprovecha este fenómeno utilizando fluoróforos acoplados a moléculas que se unen específicamente a algún componente de la célula para visualizar su localización y medir su densidad o concentración. En la neurobiología por ejemplo, la microscopía de fluorescencia como técnica experimental ha apoyado al estudio y análisis de sistemas neuronales y fenómenos asociados como la exocitosis sináptica y extrasináptica y la concentración de calcio intracelular (véase **Sección 2.2.3**). También es común utilizar un sistema de adquisición de imágenes digitales que permita registrar eventos de fluorescencia en tiempo real con una resolución temporal muy alta, facilitando el análisis cuantitativo y cualitativo de éstos. [2.4]

2.2.1 Componentes Básicos

[2.11] En general, un equipo de microscopía de fluorescencia se diferencia de un equipo de microscopía convencional por las siguientes tres modificaciones:

- Adicional a la fuente de luz visible que se utiliza para generar las imágenes microscópicas en un equipo convencional de microscopía, un equipo de microscopía de fluorescencia cuenta con un sistema de iluminación especializada capaz de irradiar luz en un amplio rango de longitudes de onda adecuadas para excitar diversos fluoróforos. Este sistema de iluminación se conoce como **Lámpara de Fluorescencia**.
- Un mecanismo o filtro que permita seleccionar del amplio rango de longitudes de onda irradiadas por la fuente de iluminación, aquella específica para excitar únicamente al fluoróforo en cuestión. Este mecanismo o filtro se conoce como **Filtro de Excitación**.
- Un segundo mecanismo o filtro que permita formar la imagen microscópica únicamente con la luz emitida por el fluoróforo, excluyendo de la misma aquella luz empleada para excitar al fluoróforo así como cualquier otra luz que pueda degradar o alterar la fidelidad y calidad de la imagen. Este segundo mecanismo o filtro se conoce como **Filtro de Emisión**.

La lámpara de fluorescencia generalmente es una lámpara de tipo arco voltaico que utiliza bulbos de vidrio llenos de gas ionizado, comúnmente de Xenón o Mercurio. Cuando se aplica voltaje sobre el bulbo, el gas ionizado conduce una corriente de electrones entre el ánodo y el cátodo produciendo un arco voltaico que emite luz en un amplio rango de longitudes de onda, que van desde el ultravioleta hasta el infrarrojo.

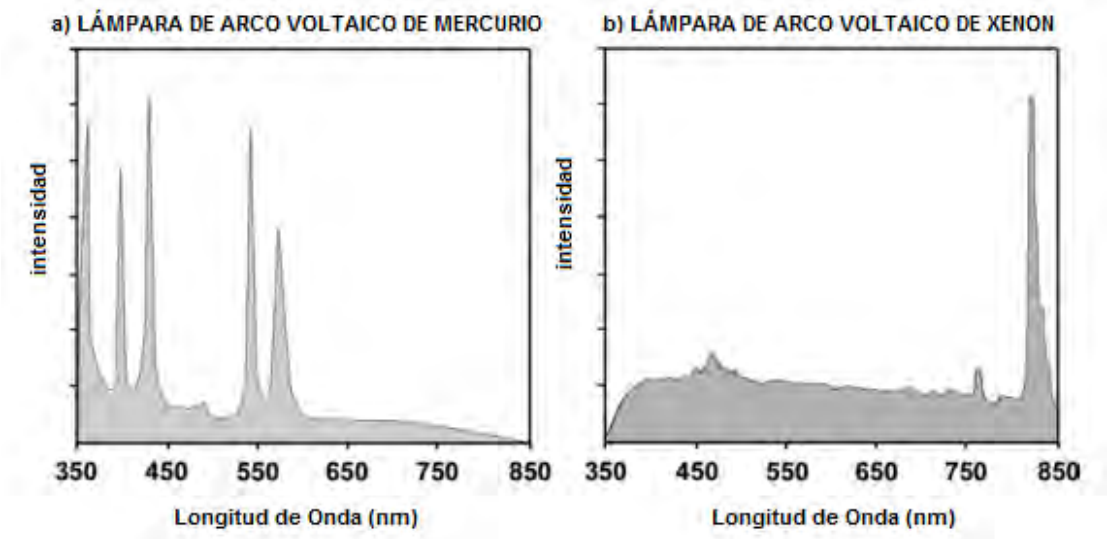


Figura 2.11 Espectro de luz emitido por una lámpara de arco voltaico de (a) Mercurio (más intensa en el ultravioleta) y (b) Xenón (más intensa en el infrarrojo).

Los filtros de excitación y de emisión, generalmente son fabricados con vidrio recubierto con material óptico especializado en el filtrado de ondas electromagnéticas; dando como

resultado un filtro óptico paso bajas, paso altas o paso bandas. En los sistemas de epi-iluminación (los más utilizados en aplicaciones de microscopía de fluorescencia) estos filtros suelen hallarse contenidos en un **Cubo Dicroico**, el cual es un arreglo cúbico de filtros y espejos que contiene un filtro de excitación de cara a la lámpara de fluorescencia, un espejo dicroico a 45° que enfoca la luz de excitación hacia la muestra o fluoróforo en cuestión y permite pasar las longitudes de onda de emisión hacia un último filtro de emisión de cara al detector del microscopio. En la **Figura 2.12** se puede apreciar un diagrama que ilustra este camino óptico de epi-iluminación comúnmente usado en los sistemas de microscopía de fluorescencia. [2.11]

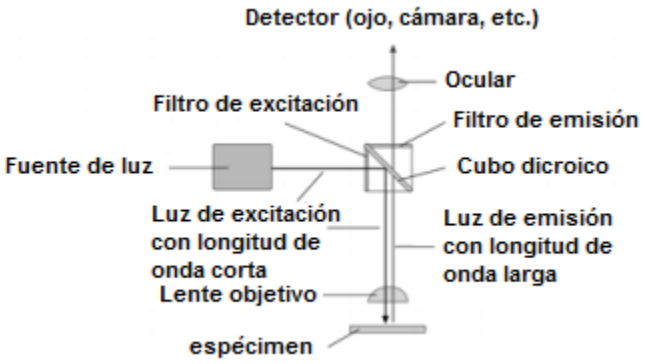


Figura 2.12 Ruta óptica de un sistema de epi-iluminación en un microscopio vertical de fluorescencia.

El microscopio utilizado puede ir desde un microscopio óptico básico o compuesto, hasta un microscopio confocal. Es bien aceptado que cualquier microscopio que permita utilizar fluorescencia para el mapeo de imágenes microscópicas, es considerado un microscopio de fluorescencia. [2.11]

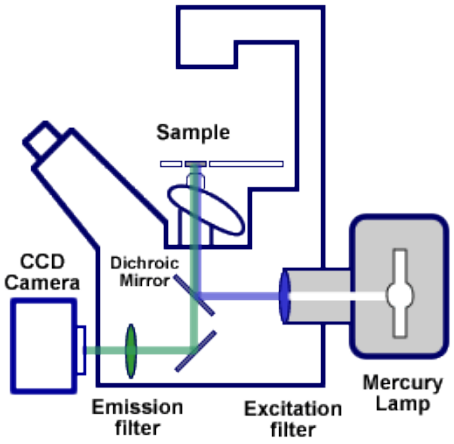


Figura 2.13 Ruta óptica en un microscopio invertido de fluorescencia.

Cuando se realizan experimentos de fluorescencia en donde la luz que excita al fluoróforo se debe presentar de manera discreta, o cuando se utilizan dos o más fluoróforos con distintas longitudes de onda de excitación a la vez en un mismo objeto de estudio, se suelen utilizar obturadores ópticos que bloquean o desbloquean el o los caminos ópticos de luz que van de la lámpara de fluorescencia al cubo dicroico o a los diferentes filtros de excitación según las necesidades particulares del experimento. Comúnmente se utilizan obturadores electrónicos controlados con señales digitales TTL (lógica transistor transistor).

Finalmente el uso de un sistema de adquisición de imágenes digitales suele ser adaptado en el microscopio de fluorescencia para almacenar imágenes digitales de fluorescencia. Generalmente se acopla al visor o detector del microscopio una cámara CCD (dispositivo de carga acoplada), la cual utiliza sensores de luz visible en un arreglo cuadricular que detectan los fotones emitidos (luz) por el fluoróforo ya excitado. Según la intensidad y geometría con la que se irradia al sensor cuadricular, éste genera una señal eléctrica de voltaje proporcional a la luz recibida, la cual viaja a través de un bus de datos analógico o digital (según las características particulares de la cámara). Esta señal viaja hasta un sistema de adquisición de imágenes digitales (comúnmente una tarjeta de adquisición de datos con interfaz USB o PCI interconectada a una computadora o estación de trabajo) que procesa la señal y la convierte en información comprensible para el usuario, es decir, en una imagen de fluorescencia del espécimen observado, desplegada por medio de algún monitor digital.

2.2.2 Metodología General

La **Figura 2.12** muestra el camino óptico de luz en un microscopio vertical, mientras que en la **Figura 2.13** se observa el camino óptico de luz en un microscopio invertido, ambos con un sistema de epi-iluminación. Aunque la arquitectura física interna, externa, y sus aplicaciones, entre otras características pueden variar, el principio de funcionamiento, cuando son utilizados como microscopios de fluorescencia, generalmente es el mismo o con pocas variaciones. Esta metodología, para la mayoría de los sistemas de microscopía de fluorescencia, es la siguiente [2.4]:

I. La lámpara de fluorescencia se enciende y emite ondas electromagnéticas en forma de luz visible, ultravioleta e infrarroja abarcando un amplio rango de longitudes de onda que pueden ir desde los 340nm hasta los 700nm aproximadamente.

II. La luz viaja a través de un camino óptico hasta pasar por un primer filtro de excitación, el cual permite pasar únicamente aquella con la longitud de onda necesaria para excitar al fluoróforo hallado en el espécimen a observar, cualquier otra onda electromagnética con longitud de onda distinta a la configuración del filtro de excitación es rechazada.

III. La luz de excitación llega hasta un espejo dicroico que la refleja y enfoca hacia el espécimen, el cual la absorbe y fluorece debido a la presencia de fluoróforos que interactúan con las ondas electromagnéticas de longitud de onda necesaria para elevar sus electrones a niveles energéticos mayores. La energía suministrada al fluoróforo por la luz de excitación es disipada por sus electrones en forma de

luz con una longitud de onda mayor a la de excitación, lo que permite que el espejo dicroico no la refleje y la filtre a un segundo camino óptico.

IV. La luz de emisión viaja por este segundo camino óptico hasta llegar a un filtro de emisión, el cual asegura que únicamente la luz emitida por el fluoróforo, que posee una longitud de onda específica y menor a la de excitación, pase hasta el detector y visor del microscopio, cualquier otra onda electromagnética con longitud de onda distinta a la configuración del filtro de emisión es rechazada.

V. La luz de emisión llega hasta los visores del microscopio formando una imagen de fluorescencia visible y acondicionada para el ojo humano, además de bañar el detector de la cámara CCD. La cámara convierte la energía en forma de luz a una señal eléctrica de voltaje la cual viaja por un bus de datos hasta un sistema de adquisición de imágenes digitales para su acondicionamiento, despliegue y almacenamiento.

2.2.3 Algunas Aplicaciones en la Neurobiología

La microscopía de fluorescencia es una técnica experimental muy utilizada en los estudios relacionados con la secreción extrasináptica de serotonina en diferentes compartimientos celulares y las fuentes de calcio que están relacionadas con dicha secreción. En las imágenes obtenidas se analiza el cambio en la fluorescencia de los diferentes compartimientos celulares asociados con la exocitosis extrasináptica en función del tiempo y en respuesta a estimulación eléctrica. A continuación se describen brevemente dos metodologías experimentales (con parámetros muy específicos) aplicadas en estudios de neurobiología:

El Fluo-4 es un fluoróforo que se inyecta en la célula por medio de un micro electrodo, éste fluorece con mayor intensidad cuando se une al calcio, es por esto que es utilizado en estudios relacionados con la concentración intracelular de calcio en la neurona. Se excita con una longitud de onda de 473 nm y emite con una longitud de onda de 503 – 543 nm. La muestra biológica se expone a la luz de excitación no más de 20 ms con un periodo de 100 ms entre cada periodo de excitación.

Por otro lado, el FM4-64 es un fluoróforo que se añade al medio extracelular de la neurona, éste fluorece con mayor intensidad cuando se adhiere a la membrana lipídica de la célula. Es utilizado en estudios relacionados con la exocitosis sináptica y extrasináptica ya que cuando ésta ocurre, las vesículas se fusionan con la membrana celular y por lo tanto queda más membrana expuesta para unirse con el fluoróforo, produciéndose así un incremento en la fluorescencia. Este fluoróforo se excita con una longitud de onda de 560 nm y emite con una longitud de onda de 600 - 700 nm. En estudios de exocitosis extrasináptica, la muestra biológica se expone a la luz de excitación no más de 50 ms con un periodo de 2000 ms entre cada periodo de excitación.

2.3 Instrumentación Virtual

La instrumentación es el área de la ingeniería encargada de diseñar, fabricar y aplicar sistemas de medición que aporten la precisión y exactitud necesarias para la asequible recolección de información provenientes de diversas fuentes; así como procesar y analizar esa información para finalmente desplegar o proporcionar un resultado. Poco avance científico se hubiera logrado hasta ahora sin el oportuno desarrollo y aplicación de la instrumentación en el área de las neurociencias; sin los potentes y modernos microscopios el hombre seguiría especulando sobre la composición y funcionamiento del complejo sistema nervioso, y de la misma manera ocurriría en muchas otras áreas del conocimiento humano.

Durante mucho tiempo, la mayoría de los instrumentos utilizados para recolectar información, procesarla y desplegar resultados utilizaron dispositivos mecánicos muy complejos que requerían años de estudio y práctica por parte del fabricante para su construcción, por lo que estos instrumentos solían ser escasos y muy caros, el mantenimiento y calibración de estos instrumentos corría a cargo del fabricante ya que no existían normas de estandarización y era prácticamente imposible que alguien ajeno al fabricante pudiera reparar, acondicionar o hallar las piezas correspondientes al instrumento. Finalmente los dispositivos mecánicos internos eran propensos a gastarse rápidamente, modificando la resolución y precisión del instrumento en un periodo de tiempo relativamente corto.

Con el descubrimiento y aplicación del electromagnetismo, se comenzaron a fabricar instrumentos de medición mucho más precisos y con una resolución mucho más potente. Sin embargo no fue hasta la invención de los tubos de vacío, comúnmente llamados bulbos, y los relés, que el análisis y procesamiento de la información comenzó a ser mucho más robusto, preciso y rápido. Después, con la aplicación de los semiconductores y la introducción de la electrónica, los sensores y transductores electrónicos comenzaron a facilitar la forma en la que se recolectaba la información, los microprocesadores aumentaron aún más el poder de procesamiento y análisis de datos, y la digitalización de la información hizo que los resultados se desplegaran de una manera mucho más comprensible para el usuario. Se comenzaron a fabricar instrumentos portátiles, con la capacidad de aplicar un análisis matemático complejo a los datos recolectados y desplegarlo en forma discreta a través de un pequeño monitor. Sin embargo el costo de estos instrumentos seguía siendo muy alto, generalmente se lanzaban al mercado instrumentos muy caros que cubrían o abarcaban necesidades de medición muy específicas y con funciones muy limitadas, con una arquitectura interna bastante rígida que le impedía al usuario aumentar sus funciones y adaptarlo a sus necesidades de instrumentación. Cuando se requería actualizar o aumentar las funciones y alcances de un instrumento, era necesario deshacerse del antiguo y adquirir uno nuevo, ya que no existía una separación como tal entre el software y el hardware del instrumento, los dos formaban parte de un todo usualmente contenido en una “caja negra”.

Finalmente el avance tecnológico dio paso al desarrollo de estaciones de trabajo y computadoras de uso general muy versátiles, con microprocesadores tan veloces y robustos, que son capaces de realizar tareas de procesamiento y análisis de datos de igual o mejor manera que los caros y rígidos instrumentos de medición convencionales, incluso de dos o más al mismo tiempo. Fue entonces cuando se comenzó a concebir la idea de una

instrumentación asistida por computadora. En el año de 1983, Truchard y Kodosky,de National Instruments, decidieron enfrentar el problema de crear un software que permitiera utilizar la computadora personal (PC) como un instrumento para realizar mediciones. De esta manera surge el concepto de instrumento virtual (IV), definido como: "un instrumento que no es real, se ejecuta en una computadora y tiene sus funciones definidas por software." [2.12]

Este tipo de metodología utiliza un hardware especializado, comúnmente llamado sistema de adquisición de datos, para la recolección de información y señales físicas (sonido, luz, calor, voltaje, corriente, etc.), la cual acondiciona y transforma en señales eléctricas discretas o digitales comprensibles para la computadora personal (PC). La PC entonces ejecuta un programa o software capaz de manipular, procesar, analizar, almacenar y realizar funciones o tareas con la información recolectada desde una misma plataforma de desarrollo o software, lo que, en conjunto con la enorme capacidad de procesamiento de las PC's modernas, le proporciona una versatilidad que jamás se había logrado con la instrumentación convencional de años anteriores. El despliegue de resultados se ve enormemente potencializado debido a la capacidad de las PC's de desplegar gráficas, estadísticas, cifras e incluso animaciones en la pantalla o monitor, lo que le permite al usuario, incluso al menos experimentado, una mejor y más sencilla comprensión y apreciación de los resultados. El usuario entonces, puede ajustar el instrumento a sus necesidades sin depender directamente de un fabricante en particular para desarrollar un instrumento determinado, dándole además la oportunidad de aumentar el alcance y funciones del mismo, sin la necesidad de reemplazar el hardware o el software en su totalidad, simplemente reprogramando las funciones del instrumento u optando por un hardware con diferentes características, disminuyendo enormemente los gastos de implementación y dándole un tiempo de vida mucho más amplio al instrumento. [2.5]

2.3.1 Componentes Básicos

[2.5] Un instrumento virtual está compuesto fundamentalmente por los siguientes elementos:

- **Computadora y Monitor**
  La computadora representa el centro de procesamiento del instrumento virtual, éste provee los requerimientos de procesamiento y memoria, para ejecutar la plataforma de desarrollo virtual (el software de instrumentación). El monitor despliega los resultados obtenidos del procesamiento, trato y análisis de datos. Los sistemas operativos basados en interfaces gráficas, han permitido a la computadora un despliegue de resultados más dinámico y gráfico que brinda mayor versatilidad y diseño a los instrumentos virtuales en comparación con la instrumentación electrónica convencional.

- **Software**
  El software es el núcleo del instrumento virtual, que le brinda la funcionalidad y personalidad a todo el sistema de instrumentación. Según la interacción del programador con el software de desarrollo, éste se clasifica en software de nivel registro, software de nivel controlador y software de alto nivel. El software de nivel registro requiere la programación de los registros individuales para cada

dispositivo o hardware utilizado en el sistema de instrumentación. El software de nivel controlador requiere la programación de la comunicación y control de los instrumentos con la computadora. El software de alto nivel es el más utilizado debido a que integra la pre-programación de los registros y controladores necesarios para utilizar y comunicar los dispositivos físicos con la PC, dejando al programador únicamente la tarea de estructurar los algoritmos de procesamiento y análisis de datos a través de un lenguaje de programación intuitivo y sencillo de utilizar, a la vez que robusto y versátil. En la sección **2.3.2** se describen las principales características del software de alto nivel LabVIEW como herramienta de desarrollo en la instrumentación virtual, aunque también existen otros paquetes de desarrollo como: LabWindows, HP VEE y Measurement Studio.

- **Interfaz de Comunicación**
  Las interfaces de comunicación más utilizadas son: la tipo Serie, Paralelo, PCI y VXI. Cada uno tiene sus protocolos de transmisión y recepción de datos, estandarizados internacionalmente, que facilitan su implementación en la instrumentación virtual. Por otro lado, los buses que comunican a la computadora con el hardware del instrumento, generalmente son cables recubiertos con material aislante y fabricados con diversos materiales que ayudan o mejoran sus propiedades de transferencia de datos, que van desde los cables y terminales de cobre, hasta la fibra óptica; aunque también existen medios de comunicación inalámbricos como el WLAN.

- **Hardware del Instrumento**
  La instrumentación virtual no elimina completamente el uso de hardware, para medir el mundo real se necesitan de sensores, transductores y dispositivos que acondicionen estas señales para su posterior manejo y despliegue.

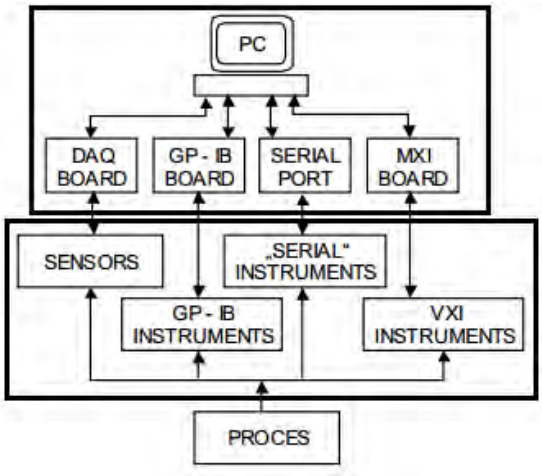


Figura 2.14 Estructura general de un instrumento virtual. El bloque superior muestra los elementos relacionados con la computadora, el monitor y el software ejecutándose en la computadora, mientras que el bloque inferior muestra el hardware del instrumento como sensores o microcontroladores. Todas las flechas bidireccionales representan la interfaz y los buses de comunicación entre el hardware de instrumentación y la PC [2.5].

2.3.2 Plataforma de Desarrollo NI LabVIEW

LabVIEW (del inglés Laboratory Virtual Instrument Engineering Workbench) es una plataforma de desarrollo virtual (software) creado en 1986 por National Instruments. Su ambiente de desarrollo es en su mayor parte gráfico, por lo que su lenguaje de programación se conoce como G. Es compatible con la mayoría de los sistemas operativos disponibles para PC como Windows, Macintosh, UNIX o Linux, convirtiéndolo en un sistema de desarrollo de instrumentación ampliamente reconocido y usado para tareas de adquisición de datos, control de instrumentos, automatización industrial, y muchas otras áreas. Los archivos o esquemas de programación creados por LabVIEW, utilizan una extensión llamada **.vi** [2.12]

Figura 2.15 Logotipo del software de desarrollo LabVIEW de National Instruments.

El lenguaje de programación de LabVIEW está basado en un entorno gráfico muy accesible, intuitivo y robusto con numerosas librerías pre programadas para la comunicación con distintos instrumentos y hardware de control. Como cualquier otro ambiente de programación, LabVIEW se sirve de una gran variedad de tipos de datos para la creación de funciones y aplicaciones (software) de alto nivel, que después serán compilados y traducidos a lenguaje máquina para su ejecución en el hardware de la PC. Sin embargo, a diferencia de lenguajes de programación textuales que utilizan prefijos como **int** o **string** para declarar los distintos tipos de datos, LabVIEW utiliza un código de colores para este fin, por ejemplo: usando el color **rosa** para el tipo de dato **string**, y **azul marino** para el tipo de dato **int**.

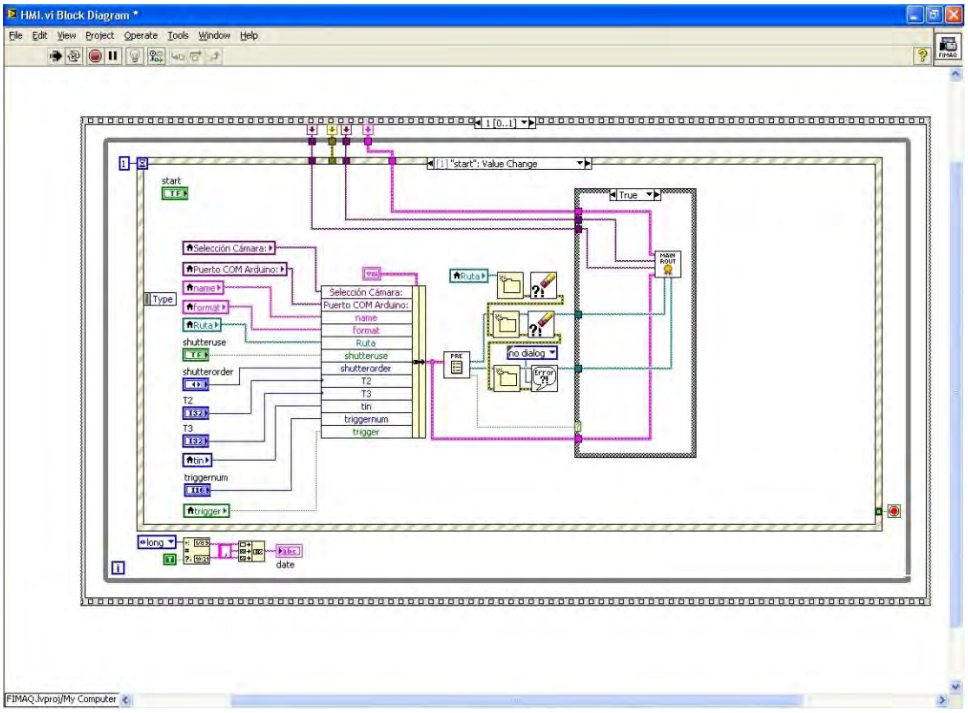


Figura 2.16 Diagrama de bloques en una aplicación estándar de LabVIEW.

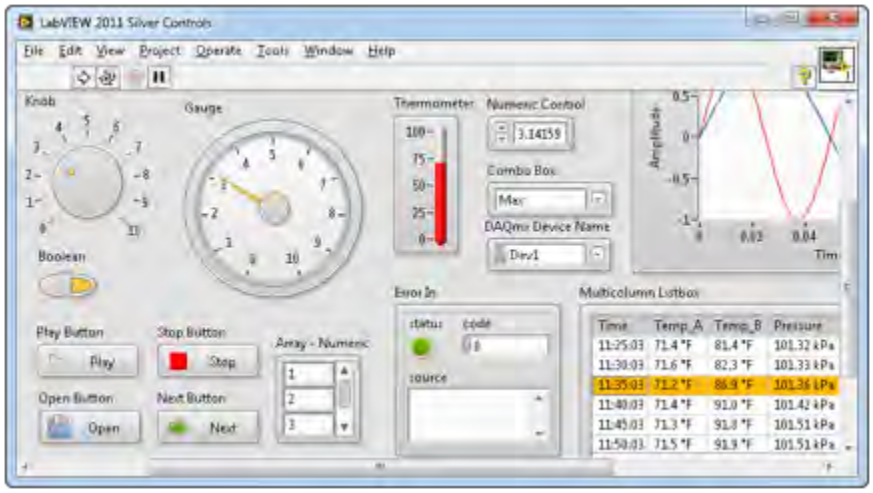


Figura 2.17 Panel de control en una aplicación estándar de LabVIEW. La interfaz de usuario se sirve de botones, perillas y demás controles gráficos o textuales operados por medio de los periféricos de la computadora para operar el instrumento virtual.

[2.12] Algunas de las principales características de LabVIEW son:

- Siendo un software de alto nivel, LabVIEW posee una arquitectura interna pre programada que le permite comunicarse y controlar gran cantidad de dispositivos a través de una comunicación serial, USB, GPIB, PCI entre otros, sin la intervención o programación directa del usuario; sin dejar a un lado la posibilidad de integrar hardware y dispositivos nativos fabricados y programados a nivel registro y controlador por el usuario. Todo esto a través del software MAX (del inglés Measurement and Automation eXplorer) de National Instruments.

- Al utilizar un lenguaje de programación gráfico o lenguaje de programación G, le da la posibilidad al programador de crear una interfaz de usuario bastante dinámica, intuitiva y sencilla de utilizar. Su estructura de programación a través de hilos o flujos de información generalmente resulta en una sintaxis de programación más sencilla e intuitiva de usar, lo que además facilita la programación de hilos paralelos de programación. Finalmente LabVIEW posee una potente herramienta de compilación que traduce el lenguaje G a código nativo para el CPU de la computadora, y en conjunto con la plataforma de ejecución LabVIEW run-time, las aplicaciones creadas en LabVIEW se convierten en códigos portátiles y adaptables a multitud de sistemas operativos (siempre y cuando tengan LabVIEW run-time).

- LabVIEW posee una enorme librería pre programada con multitud de funciones de adquisición de datos, generación de señales, matemáticas, estadísticas, acondicionamiento de señales, análisis cuantitativo y cualitativo de la información, entre muchas más, con la posibilidad de aumentar esta funcionalidad con paqueterías especializadas desarrolladas por National

Instruments o por usuarios independientes, muchas de estas gratis o a un costo de adquisición muy bajo. Además, algunas funciones incluidas en LabVIEW como MathScript, son compatibles y pueden ser migradas a otras plataformas de desarrollo como MATLAB.

- Debido a su difundido uso y aplicación en distintas áreas, es común hallar mucha asistencia técnica en comunidades de usuarios ajenos a National Instruments. Su compatibilidad con la mayoría de los sistemas operativos disponibles para las PC's modernas hacen de los instrumentos programados en LabVIEW sistemas muy versátiles y adaptables, con un tiempo de vida muy extendido y la posibilidad de dar mantenimiento y actualización a cualquier instrumento virtual siempre que se tenga conocimiento del lenguaje de programación G.

[2.12] Entre sus principales limitantes se encuentran:

- El lenguaje de programación gráfico de LabVIEW no se encuentra administrado o especificado por una agencia de estandarización internacional como ANSI, IEEE o ISO, sino por la misma National Instruments, a diferencia de algunos otros lenguajes de programación como C o FORTRAN.

- Para un programador especializado en lenguajes de programación con sintaxis escrita, migrar al ambiente de programación de LabVIEW suele ser confuso y poco intuitivo.

- Las aplicaciones creadas en LabVIEW necesariamente requieren que la PC tenga instalado el LabVIEW run-time con una versión igual o mayor a la aplicación que se intenta ejecutar, lo cual limita la portabilidad de las aplicaciones desarrolladas en LabVIEW cuando la PC destino no admite o carece del software, además del implícito pago por la licencia de LabVIEW run-time, lo cual resulta en muchos casos contraproducente económicamente.

- El uso de recursos que LabVIEW run-time demanda a la PC puede ser excesivo cuando se ejecutan aplicaciones muy básicas.

2.3.3 Metodología General

[2.5] La instrumentación virtual, aunque con sus características y componentes muy particulares, continúa dando solución al objetivo fundamental de medición establecido por la instrumentación como importante rama de la ingeniería, a través de una metodología elemental compuesta por una etapa de adquisición de la información, una de procesamiento de los datos y una de despliegue o presentación de resultados, la cual, adaptada a la instrumentación virtual, es la siguiente:

I. Las magnitudes físicas en forma de distancia, peso, luz, calor, voltaje, entre muchas otras; son recolectadas a través de un hardware de instrumentación o sistema de adquisición de datos (integrado por dispositivos mecánicos, eléctricos y semiconductores electrónicos como sensores o transductores), los cuales transforman o acondiciones estas magnitudes físicas en señales discretas o analógicas de voltaje o corriente, según las características particulares del hardware o dispositivo sensor. Las señales de voltaje o corriente, viajan a través de un medio físico alámbrico o inalámbrico llamado bus de datos hasta las terminales de la computadora, a través de un protocolo de comunicación o transferencia de datos de tipo serie, paralelo, o alguno otro. Si las señales de voltaje o corriente que llegan a las terminales de la computadora son analógicas, entonces un dispositivo se encarga de digitalizar las señales, de tal manera que el total de los datos adquiridos por el hardware de instrumentación sea transformado a información binaria, comprensible para la computadora y el software encargado de manipular la información.

II. La computadora ejecuta un software o programa previamente instalado, configurado y diseñado según las necesidades particulares del usuario o proceso a medir, con una serie de tareas y funciones que procesan, analizan, almacenan y le dan un tratamiento a los datos adquiridos. Generalmente, el usuario puede intervenir en el procesamiento y análisis de los datos a través de un panel virtual de control mostrado en el monitor y manipulado con los periféricos de la computadora, como lo son el teclado y el ratón.

III. Finalmente el software despliega el o los resultados del proceso y análisis de los datos, a través del monitor de la computadora en forma de números, palabras, gráficas, predicciones, y demás formatos gráficos; o a través del propio hardware de instrumentación o los periféricos de la computadora, en forma de señales de voltaje o corriente que generalmente implican una tarea de control específica como la impresión de un documento, la reproducción de sonidos, el control de instrumentos industriales o médicos, etc.

2.3.4 Algunas Aplicaciones en la Biomedicina

Además de la automatización y control industrial, el uso de la instrumentación virtual ha ido en aumento en diferentes áreas como las telecomunicaciones y la biomedicina. Muchos proyectos de investigación biomédica han comenzado a explorar la posibilidad de incluir aplicaciones de instrumentación virtual en diversas áreas como la examinación, monitoreo, entrenamiento, educación y rehabilitación médica.

Se ha trabajado en el desarrollado de dispositivos médicos de tipo invasivos y no invasivos como marcapasos o medidores de la presión arterial, capaces de comunicarse en tiempo real a través de una red inalámbrica de datos con el instrumento virtual. Constantemente examinan y monitorean las condiciones fisiológicas del paciente, alertando por medio de un mensaje o una notificación al médico o al paciente cuando estas condiciones rebasan los umbrales médicos aceptados. [2.6]

Muchas clínicas, instituciones médicas y universidades han comenzado a incluir instrumentos virtuales para simular condiciones médicas de pacientes en el entrenamiento del personal y los estudiantes de medicina. Muchas de estas señales son grabaciones reales de pacientes que han sufrido condiciones médicas serias como arritmias cardiacas, paros respiratorios o fallos en la presión sanguínea, por lo que el entrenamiento del personal se asemeja bastante bien a una situación real. Los estudiantes pueden interactuar con el panel virtual del instrumento tal como si estuvieran manipulando el instrumento real, y repetir una y otra vez las condiciones médicas almacenadas en la memoria de la computadora. [2.6]

Finalmente, a través de una aplicación de instrumentación virtual, el médico puede dar rehabilitación física a un paciente, obteniendo un monitoreo de signos vitales y retroalimentación de las condiciones biológicas y fisiológicas del paciente en tiempo real, almacenando toda la actividad en la memoria de la computadora para su posterior análisis, diagnóstico o futuras sesiones con el paciente, mientras éste manipula ciertos objetos, realiza actividades físicas o interactúa con el panel virtual del instrumento. Este tipo de aplicaciones han sido utilizados en el tratamiento de desórdenes psicofisiológicos de adicciones, ansiedad y depresión. [2.6]

Capítulo 3

Planteamiento del Problema

Considerando que el laboratorio ya cuenta con el equipo necesario para montar un sistema de adquisición de imágenes digitales de microscopía de fluorescencia, la interrogante fundamental a responder en este proyecto de tesis es la siguiente:

¿Qué metodología de instrumentación se debe implementar para sincronizar y controlar el sistema de adquisición de imágenes digitales de microscopía de fluorescencia con el que y se cuenta, de manera eficiente y a un bajo costo de implementación?

De primera instancia se descarta la posibilidad de desarrollar un controlador particular para cada uno de los cuatro procesos descritos en la **Sección 1.1**, ya que la sincronización de la totalidad de los controladores supone un problema mayor que el original. Además, la capacitación del usuario final en el manejo de cuatro controladores diferentes para el uso del sistema supone una curva de aprendizaje prolongada.

Tampoco se cuenta con el presupuesto para adquirir un nuevo sistema o un software de control comercial, lo cual supone un costo de implementación enorme para el laboratorio.

Por ello que se propone utilizar una metodología de instrumentación virtual sobre el equipo ya disponible, para desarrollar un software de control que se encargue de sincronizar la totalidad de los procesos en una sola plataforma de desarrollo llamada LabVIEW, capaz de cubrir las necesidades particulares de investigación, las tareas básicas de un experimento de microscopía de fluorescencia y sobre una interfaz de usuario robusta, intuitiva e integral; disminuyendo así los costos de implementación.

Capítulo 4

Objetivos

- Desarrollar el software de control, adquisición, gestión de recursos informáticos y monitoreo para un sistema de adquisición de imágenes digitales de microscopía de fluorescencia, a partir de un modelo de instrumentación virtual programado en la plataforma de desarrollo LabVIEW.

- Además, diseñar e implementar la interfaz de usuario del software de control.

Capítulo 5

Descripción del Hardware y Metodología

El sistema de adquisición de imágenes digitales de microscopía de fluorescencia lo integran cinco subsistemas de hardware principales:

- El equipo óptico y de iluminación de imágenes,
- El equipo sensor de imágenes,
- El equipo encargado de la adquisición y digitalización de imágenes,
- La unidad secundaria o periférica de control, y
- La unidad central de procesamiento de datos, control y despliegue de imágenes.

El equipo óptico y de iluminación de imágenes lo integran el sistema de iluminación de fluorescencia junto con los filtros ópticos que regulan la longitud de onda de la luz y una serie de lentes y espejos dentro del microscopio que en conjunto regulan forma en la que incide la luz en el modelo biológico y por consecuencia la creación de imágenes de fluorescencia. Como sensor principal se utilizó una cámara CCD, la cual capta la luz emitida por el modelo biológico. El equipo sensor se conectó con el equipo de adquisición de imágenes a través de un bus analógico de video, el cual se comunica con la unidad central de procesamiento, control y visualización por medio de una tarjeta de adquisición y digitalización de audio/video con interfaz USB. Una computadora de uso general conforma la unidad central de procesamiento, control y visualización; la cual ejecuta el software de control desarrollado en la plataforma LabVIEW. Finalmente, como unidad secundaria de control se utilizó un microcontrolador Arduino para el manejo y sincronización de dos obturadores electrónicos y la generación de señales TTL de control a través de sus catorce puertos de entrada/salida digitales. El microcontrolador Arduino se comunica con la unidad central de procesamiento a través de un bus de datos serial con interfaz USB.

A continuación se desglosan las especificaciones técnicas e información relevante sobre el funcionamiento y operación de cada uno de estos instrumentos y dispositivos que integran el sistema de adquisición de imágenes digitales de microscopía de fluorescencia. Finalmente se describe la metodología a utilizar para el funcionamiento de dicho sistema.

5.1 Sistema de Iluminación de Fluorescencia

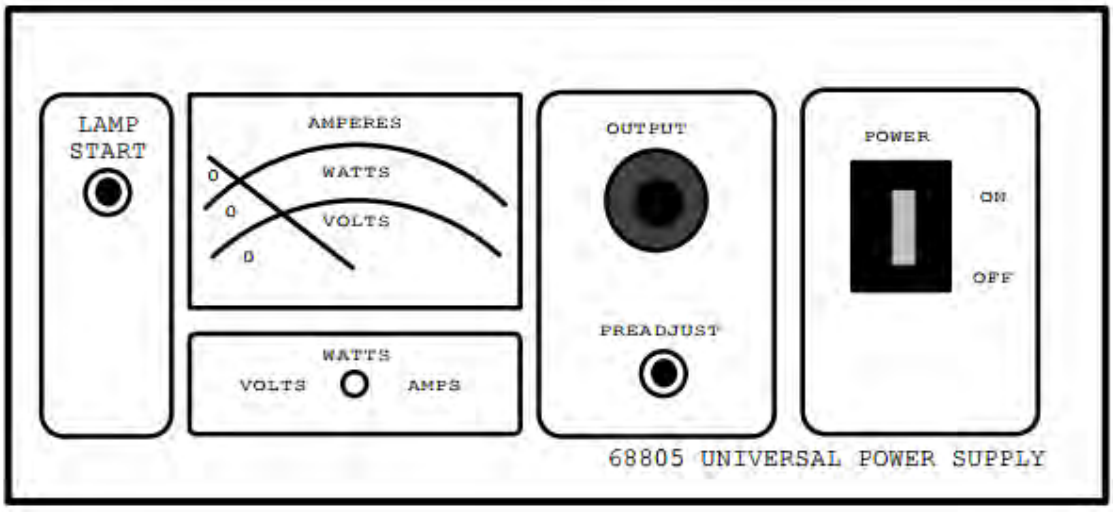
El sistema de iluminación de fluorescencia lo integran tres instrumentos principales:

- Fuente de alimentación de la lámpara de arco (40-200 W), marca **Oriel** modelo 68805,
- Chasis de la lámpara de arco, marca **Oriel** modelo 66001, y
- Lámpara de arco de Xenón (150 W), marca **Oriel** modelo 6256.

5.1.1 Fuente de Alimentación de la Lámpara de Arco

La fuente de alimentación fue diseñada para cubrir la demanda de una fuente de corriente alta, constante y regulada, necesaria para operar la lámpara de arco de fluorescencia. La fuente de alimentación es regulada por un voltaje de salida de 0 a 100 V dependiente a la impedancia del bulbo de arco alojado en la lámpara de fluorescencia, proporcionando una corriente constante y regulada de hasta 11 A. La fuente cuenta además con un limitador de potencia de 300 W y 12 A que proteja al equipo contra sobrecargas y cortos circuitos, además de un circuito reductor de descargas que minimiza la erosión causada al electrodo encendedor al momento de activar la lámpara de fluorescencia.

(A)



(B)

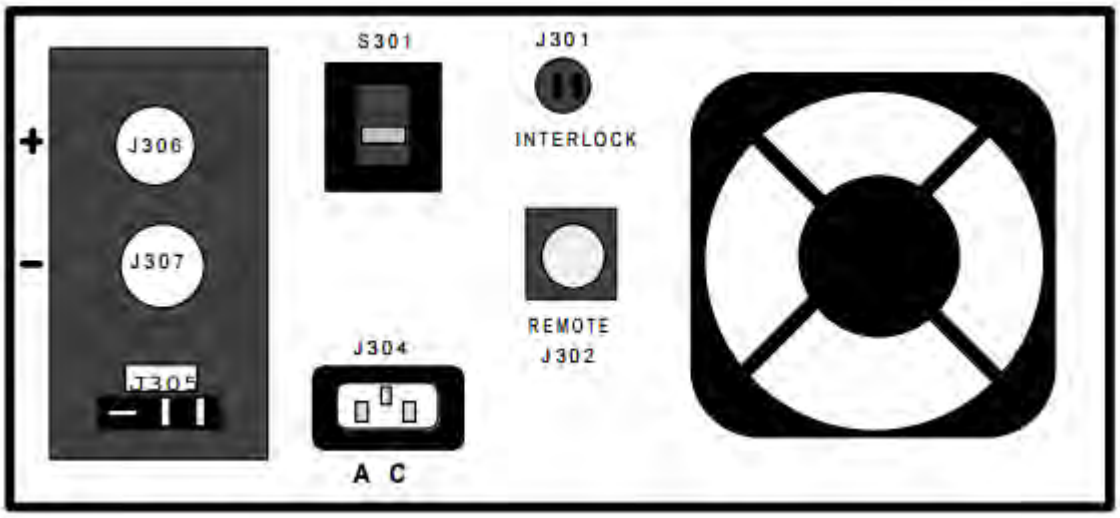


Figura 5.1 Vista frontal (A) y vista trasera (B) con nomenclatura y estructura general de la fuente de alimentación de la lámpara de arco (40-200 W), marca ***Oriel*** *modelo 68805.*

Las especificaciones técnicas de la fuente de alimentación se observan en la **Figura 5.2**

AC Mains Input:	95–135 VAC @ 8 amperes 135–270 VAC @ 4 amperes 50 or 60 Hertz, 10 ampere inrush
DC Power Output:	300 watts maximum
DC Current Output:	Adjustable from 0.5 – 11 amperes Current limit at 12 amperes
DC Voltage Output: Preignition Operating	 > 150 volts unloaded Load dependent. Total output power should not exceed 240 watts.
Light Output Ripple: Resistive Load Arc Lamp Load	 < 0.25% R.M.S. 40 Hz to 40 kHz Dependent on lamp, and operating conditions. Typically < 0.5% R.M.S.
Pre–Adjust Accuracy:	2%
Power Meter Accuracy:	2%
Line Regulation:	0.1% change in output current for a 95–135 VAC mains input or a 190–235 VAC mains input

Figura 5.2 Especificaciones técnicas de la fuente de alimentación de la lámpara de arco (40-200 W), marca ***Oriel*** *modelo 68805.*

Para mayor información acerca de la fuente de alimentación de la lámpara de arco (40-200 W), marca **Oriel** modelo 68805 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.1]** para consultar por internet la hoja de especificaciones oficial (Official Datasheet) del instrumento.

5.1.2 Chasis de la Lámpara de Arco

El chasis marca Oriel modelo 66001 corresponde a una carcasa para lámparas de arco corto de 50 hasta 250 W que cuenta con un condensador F/1.5 y un encendedor interno en la parte superior de la carcasa para activar el bulbo o lámpara de arco, reduciendo las interferencias RF y eliminando la necesidad de adquirir un encendedor aislado. El chasis cuenta con un arreglo de lentes internos para la proyección de luz UV así como un ventilador que enfría el interior de la carcasa para protegerlo contra fallas por sobrecalentamiento. El socket del chasis acepta bulbos de Mercurio (Hg) y Xenón (Xe) así como de Mercurio-Xenón [Hg (Xe)] de diferentes potencias, siendo un chasis de lámpara de arco universal. Finalmente, el chasis está diseñado para operar con una fuente de alimentación **Oriel** modelo 68805.

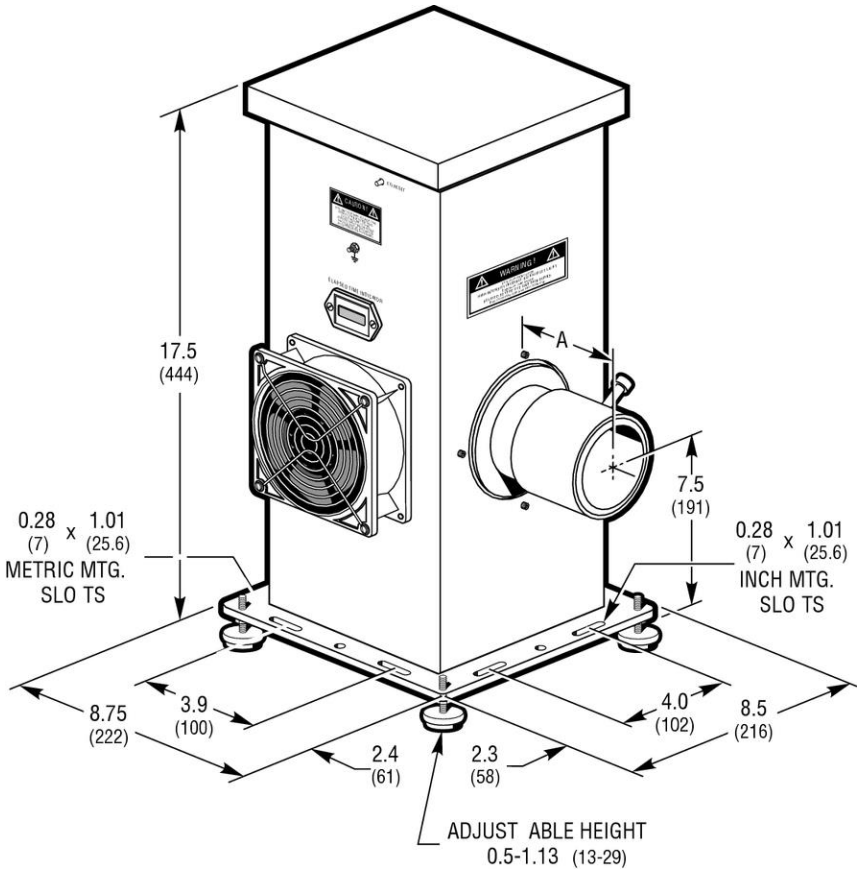


*Figura 5.3 Chasis de la lámpara de arco, marca **Oriel** modelo 66001.*

Para mayor información acerca del chasis de la lámpara de arco, marca **Oriel** modelo 66001 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.2]** para consultar por internet la hoja de especificaciones oficial (Official Datasheet) del instrumento.

5.1.3 Lámpara de Arco de Xenón 150 W

La lámpara de arco marca **Oriel** modelo 6256 se encuentra llena de xenón purificado a una presión de 5-20 bar, la cual se triplica cuando se encuentra en funcionamiento. Tiene un tamaño de arco efectivo de 0.5 x 1.5 mm e irradia luz en un espectro electromagnético de 340 nm a 700 nm, siendo más intensa en el infrarrojo. Se utiliza en aplicaciones de absorbancia y fluorescencia, aunque también se suele utilizar en simulaciones solares por su espectro parecido al solar. El bulbo se coloca en el chasis de la lámpara de arco con el ánodo en la parte superior de la carcasa. El diámetro del bulbo es de 18 mm compatible con el chasis de lámpara de arco marca **Oriel** modelo 66001.

*Figura 5.4 Bulbo o lámpara de arco de Xenón 150 W, marca **Oriel** modelo 6256.*

Las especificaciones técnicas de la lámpara de arco de Xenón de 150 W se observan en la **figura 5.5**

Model	**6256**
Type	Replacement Lamps
Lamp Type	Xenon, Ozone Free
Lamp Wattage	150 W
Lamp Current (A)	7.5 A
Lamp Voltage	20 V
Effective Arc Size	0.5 x 1.5 mm
Bulb Diameter	18 mm
Horizontal Intensity	300 cd
Average Life	1000 h
Special Features	Compact Lamp
Approximate Flux	2200 Lumens
Approximate Brightness	160 cd mm^{-2}

*Figura 5.5 Especificaciones técnicas de la lámpara de arco de Xenón 150 W, marca **Oriel** modelo 6256.*

Para mayor información acerca de la lámpara de arco de Xenón 150 W, marca **Oriel** modelo 6256 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.3]** para consultar por internet la hoja de especificaciones oficial (Official Datasheet) del instrumento.

5.2 Filtros Ópticos

El equipo óptico que integra el sistema de adquisición de imágenes digitales de microscopía de fluorescencia cuenta con hasta tres juegos de volantes de filtros con cinco posiciones manuales, marca **Oriel** modelo 77370. Estos volantes de filtros se colocan posteriores a los obturadores electrónicos y funcionan como filtros ópticos de excitación para el sistema de iluminación de fluorescencia. Tiene la capacidad de alternar hasta cinco filtros ópticos de 25.4 mm de diámetro cada uno, con un grosor de 0.06 hasta 0.5 in. En la parte superior el instrumento cuenta con una perilla numerada que permite rotar los distintos filtros e indicar el filtro actual en uso.

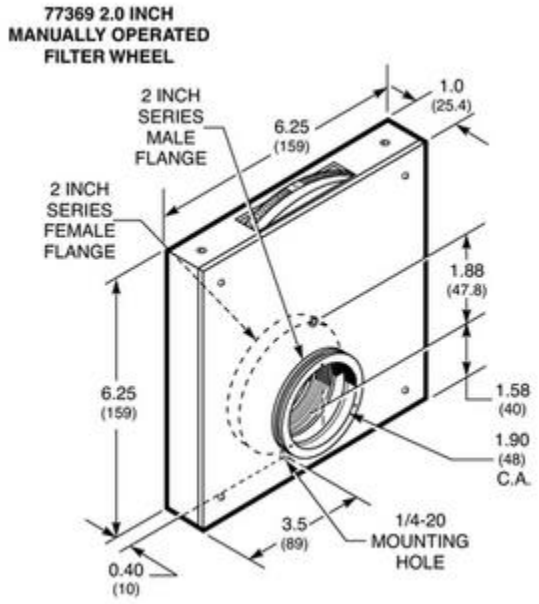


Figura 5.6 Volante de filtros con cinco posiciones manuales, marca ***Oriel*** *modelo 77370.*

Las especificaciones técnicas del volante de filtros con cinco posiciones manuales se observan en la **Figura 5.7**

Model	**77370**
Maximum Thickness	12.7 mm
Shape	Round
Number of Filters	5
Mechanism	Snap In Holder
Mounting Hole Type	1/4-20
Filter Diameter	25.4 mm
Material	Anodized Aluminum
Flange Type	1.5 inch Male and Female

*Figura 5.7 Especificaciones técnicas del volante de filtros con cinco posiciones manuales, marca **Oriel** modelo 77370.*

Finalmente se cuenta con los siguientes filtros ópticos de excitación compatibles con el volante de filtros con cinco posiciones manuales, marca **Oriel** modelo 77370:

- Filtro 495 nm NO-XB15 CHROMA.
- Filtro 495 nm (No especificado).
- Filtro 490 NO-XB13 OMEGA.
- Filtro 440 nm OMEGA.
- Filtro 435 nm CHROMA.
- Filtro 380 nm (No especificado).
- Filtro 340 nm (No especificado).

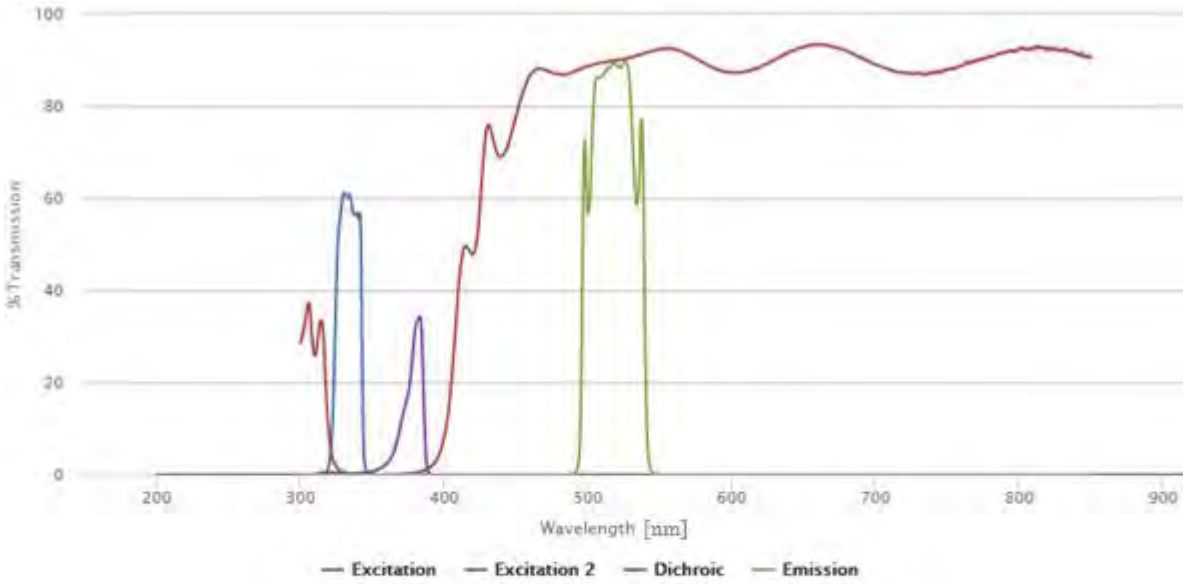
Para mayor información acerca del volante de filtros con cinco posiciones manuales, marca **Oriel** modelo 77370 así como de los diferentes filtros de excitación, véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.4], [5.5]** y **[5.6]** respectivamente para consultar por internet la página oficial de los instrumentos.

5.2.1 Cubo Dicroico

El sistema de adquisición de imágenes digitales de microscopía de fluorescencia utiliza un cubo dicroico, el cual integra tanto un filtro de excitación, un espejo dicroico y un filtro de emisión.

Se cuenta con dos juegos de cubos dicroicos ambos fabricados por OMEGA, siendo estos:

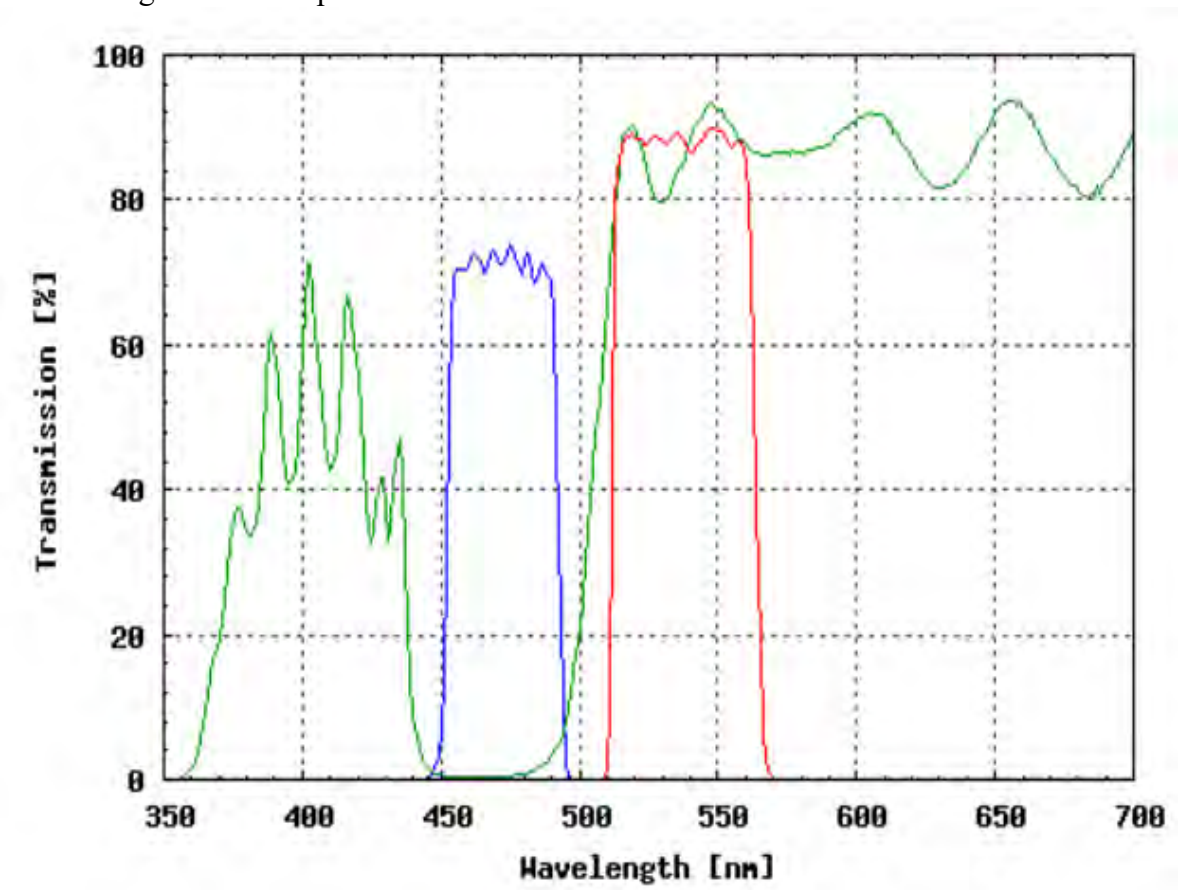
- Juego de filtros ópticos marca **OMEGA** modelo XF04-2



FILTER SET COMPONENTS	
Exciter1:	XF1093 (340AF15)
Exciter2:	XF1094 (380AF15)
Dichroic:	XF2002 (415DCLP)
Emitter:	XF3043 (510WB40)

*Figura 5.8 Especificaciones técnicas de excitación y emisión del juego de filtros ópticos (cubo dicroico) marca **OMEGA**, modelo XF04-2.*

- Juego de filtros ópticos marca **OMEGA** modelo XF100-2



FILTER SET COMPONENTS	
Exciter:	XF1073 (475AF40)
Dichroic:	XF2010 (505DRLP)
Emitter:	XF3084 (535AF45)

*Figura 5.9 Especificaciones técnicas de excitación y emisión del juego de filtros ópticos (cubo dicroico) marca **OMEGA**, modelo XF100-2.*

Para mayor información acerca de ambos juegos de filtros ópticos (cubos dicroicos), marca **OMEGA** modelo XF04-2 y XF100-2 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.6]** para consultar por internet la página oficial de los instrumentos.

5.3 Microscopio de Fluorescencia

El microscopio utilizado en el sistema de adquisición de imágenes digitales de microscopía de fluorescencia, es un microscopio invertido trinocular de luz transmitida marca **Nikon** Diaphot-TDM utilizado principalmente en estudios de cultivo de tejidos. En la **Figura 2.13** se ilustra el camino óptico de fluorescencia utilizado por este microscopio invertido.

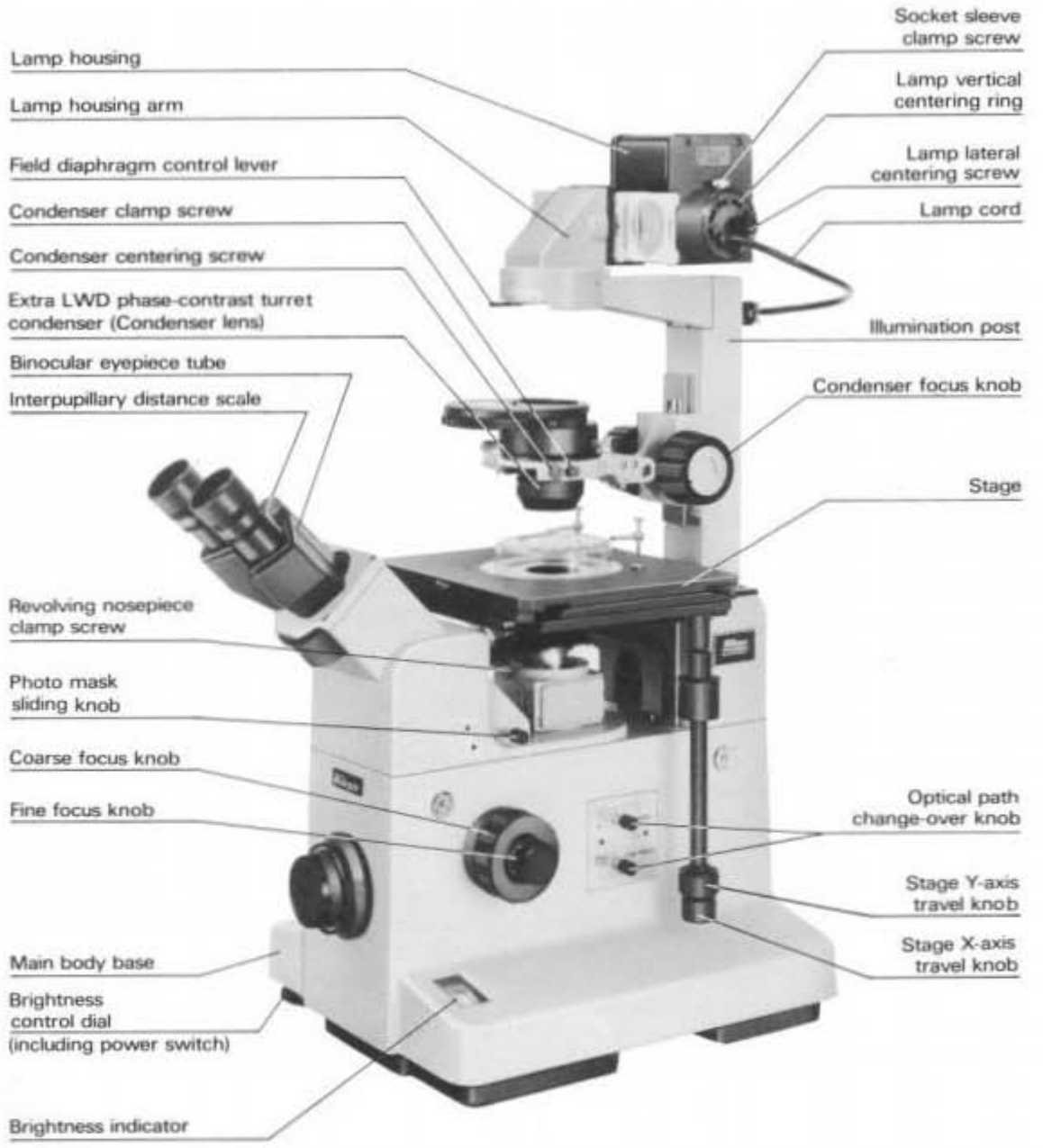
Diseñado para ser utilizado en técnicas de fotomicrografía, el microscopio posee un camino óptico interno que permite capturar imágenes microscópicas observadas en el binocular inclinado a 45° en fotografías, por acción de una cámara digital o analógica adjunta al microscopio. Con una sola superficie reflejante en el cuerpo del microscopio para reducir reflejos parásitos y destellos, se logra maximizar un contraste de imágenes de calidad.

Producido en los años 80's, los binoculares poseen un diseño Siedentopft construidos en la parte superior y posterior al cuerpo fundido del microscopio. Unos lentes de enfoque Bertrand fueron construidos en el cuerpo del binocular para una rápida alineación de los anillos de contraste de fases. Un dispositivo de diapositivas oscuras fue incorporado en la torreta para eliminar luces parásitas del sistema óptico durante la fase de cine fotomicrografía así como magnificador de enfoque de baja potencia para técnicas de fotomicrografía muy nítidas con el objetivo de 2X y 4X.

La platina se localiza ligeramente debajo de la cabeza binocular, permitiendo una fácil manipulación al mismo tiempo que se observa el espécimen. El sistema de enfoque fino y grueso controla la parte del microscopio en donde se encuentran los lentes objetivos, dejando el portaobjetos estacionario y el espécimen expuesto para su manipulación durante la observación. Los objetivos poseen una longitud tubular de 160 mm.

Finalmente el microscopio posee una fuente de iluminación visible de 12 V con un bulbo de halógeno de 50 W, pudiendo regular la intensidad de la iluminación de un 100 por ciento a un 20 por ciento con un nivelador.

(A)


Socket sleeve clamp screw
Lamp housing
Lamp vertical centering ring
Lamp housing arm
Lamp lateral centering screw
Field diaphragm control lever
Lamp cord
Condenser clamp screw
Condenser centering screw
Extra LWD phase-contrast turret condenser (Condenser lens)
Illumination post
Binocular eyepiece tube
Condenser focus knob
Interpupillary distance scale
Stage
Revolving nosepiece clamp screw
Photo mask sliding knob
Coarse focus knob
Optical path change-over knob
Fine focus knob
Stage Y-axis travel knob
Stage X-axis travel knob
Main body base
Brightness control dial (including power switch)
Brightness indicator

(B)

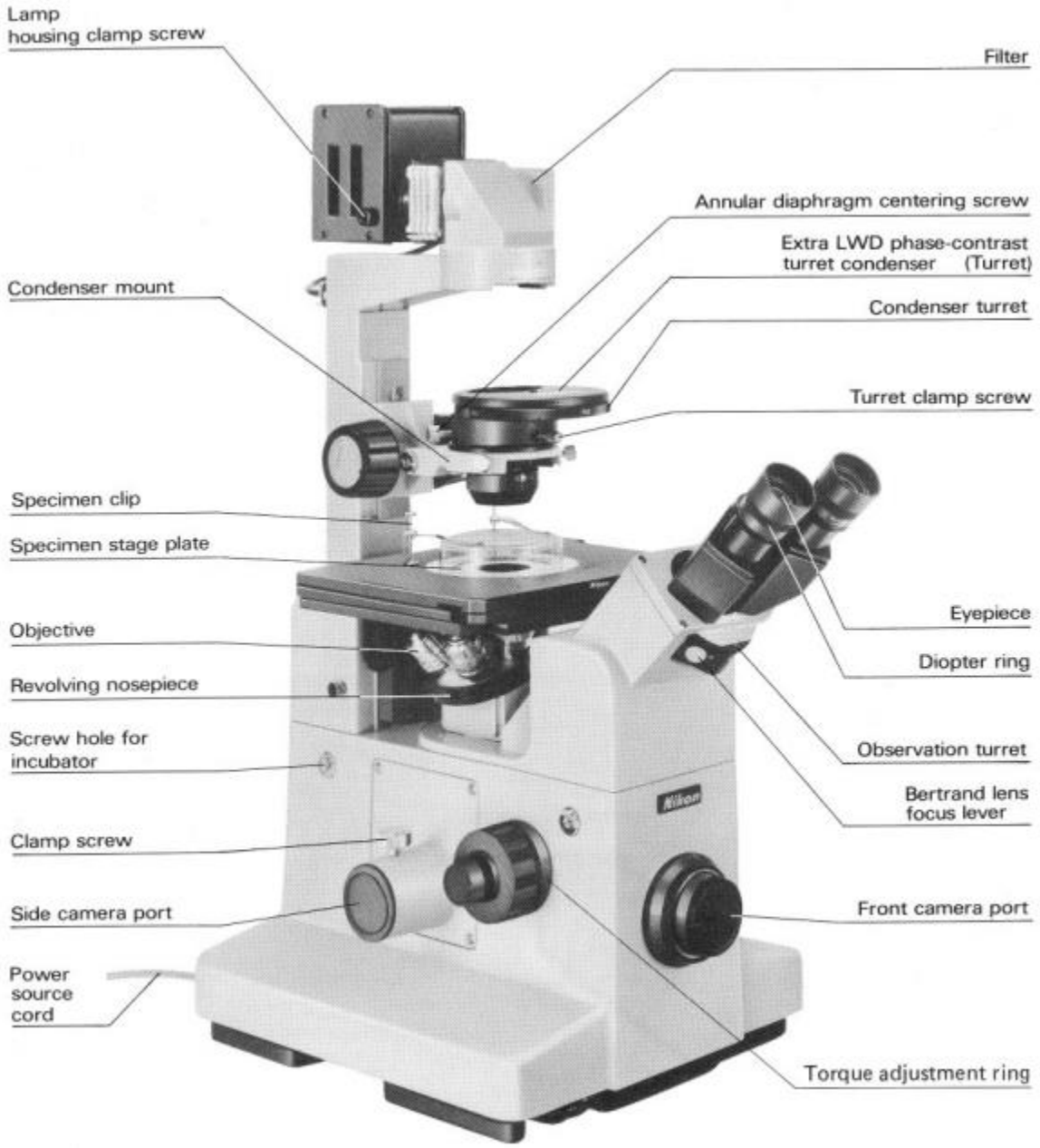


Figura 5.10 Microscopio invertido marca ***Nikon*** *modelo Diaphot-TDM trinocular de luz transmitida para cultivo de tejidos. Tanto la vista lateral (A) como la vista lateral (B) del microscopio muestran la nomenclatura y partes que integran al instrumento, así como su estructura general externa.*

Para mayor información acerca del microscopio invertido, marca **Nikon** modelo Diaphot-TDM trinocular de luz transmitida para cultivo de tejidos véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.7]** para consultar por internet el manual de operación junto con las especificaciones técnicas oficiales del instrumento.

5.4 Microcontrolador

El microcontrolador a utilizar en el sistema de adquisición de imágenes digitales de microscopía de fluorescencia es un Arduino UNO R3, encargado de realizar las tareas de control y sincronización de la apertura de los obturadores electrónicos, así como para la generación de señales TTL para la comunicación con otros instrumentos de laboratorio.

El microcontrolador Arduino UNO R3 posee un microprocesador ATMEGA328, tiene 14 pines de salidas/entradas digitales (de las cuales 6 pueden ser utilizadas como salidas de PWM), 6 pines de entradas analógicas, utiliza un resonador cerámico de 16 MHz, un conector USB para su comunicación con la PC, una entrada de alimentación de 2.1 mm con plug de centro positivo de 7 a 12 VDC, un cabezal ICSP y un botón de reset. El microcontrolador Arduino UNO puede ser programado a través del software Arduino IDE que soporta un lenguaje de programación C y C++ y utiliza el protocolo de comunicación STK500.

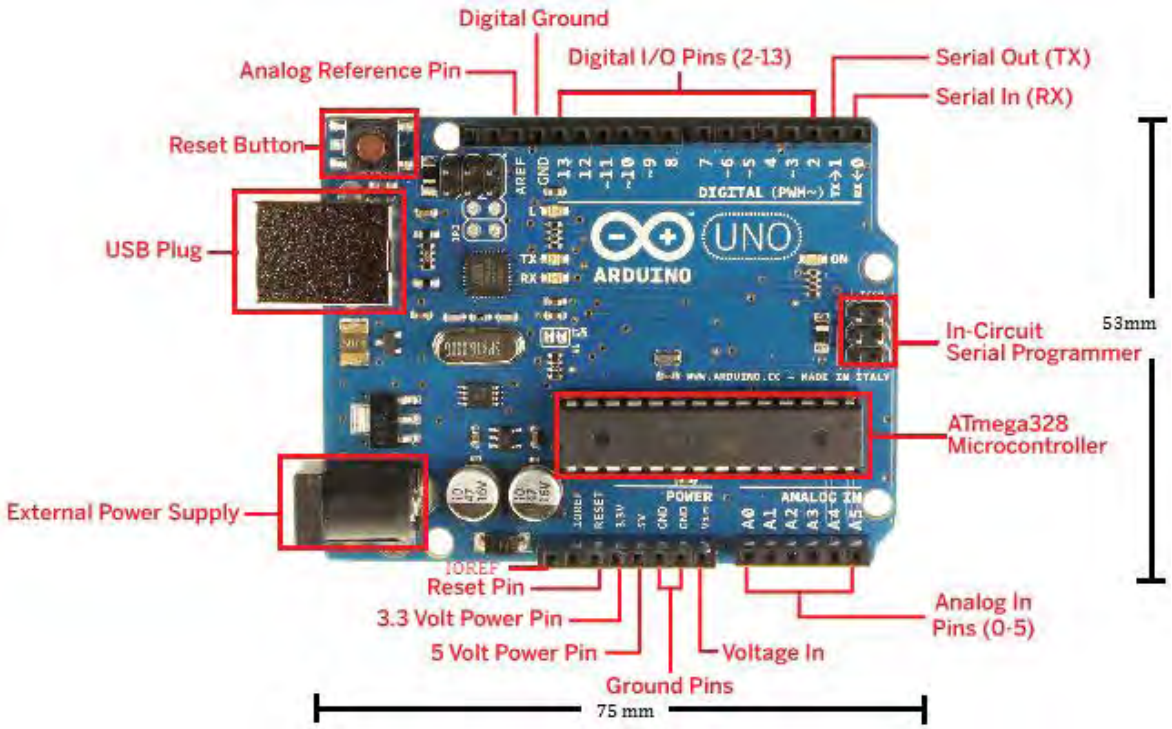


Figura 5.11 Vista frontal del microcontrolador Arduino UNO R3. Se pueden observar sus principales características y dispositivos.

Las especificaciones técnicas del Arduino UNO R3 se observan en la **Figura 5.12**

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Figura 5.12 Especificaciones técnicas del Arduino UNO R3.

Para mayor información acerca del Arduino modelo UNO R3 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.9]** para consultar por internet la página oficial del instrumento.

5.4.1 Microprocesador ATMEGA328P-PU

El microcontrolador Arduino UNO R3 utiliza como microprocesador el circuito integrado ATMEGA328P-PU de 8 bits con 32 KB de memoria en sistema programable flash, de los cuales 0.5 KB están reservados para el software de bootloader; el cual es un software pre configurado y pre quemado en el microprocesador que permite cargar nuevo código de programación (programas creados por el usuario) sin necesidad de utilizar un hardware programador externo. Cuenta además con 2 KB de SRAM y 1 KB de EEPROM. Entre sus principales características se encuentran:

- Alto rendimiento con un bajo consumo de energía (1.8 – 5.5 VDC, 0.2mA en modo activo y 0.75uA en modo de ahorro de energía).
- Arquitectura avanzada RISC (131 potentes instrucciones y 32x8 registros funcionales de propósito general).
- Segmentos de memoria no volátil de alta resistencia (retención de la información por 20 años a 85°C de operación/100 años a 25°C de operación).
- Contador de tiempo real con oscilador separado.

- 6 canales PWM.
- 14 canales digitales y 6 canales analógicos (convertidor ADC de 10 bits).
- USART serial programable.
- Temporizador Watchdog programable con oscilador separado en el chip.
- Botón de Reset (mejorado y más robusto en la revisión R3).
- Interrupciones con fuentes internas y externas.
- Oscilador calibrado interno.
- Temperatura de operación -40°C a 85°C.
- Velocidad de procesamiento de 0 – 20 MHz (1.8 – 5.5 V).

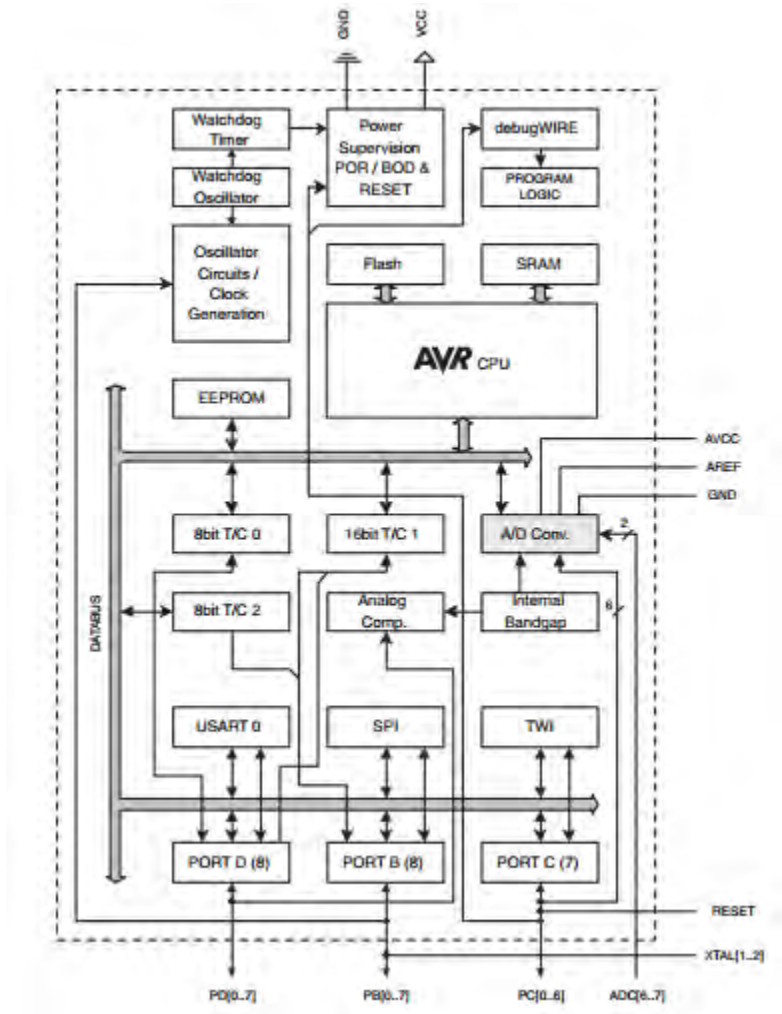


Figura 5.13 Diagrama de bloques del microprocesador ATMEGA328P-PU en el cual se puede observar su arquitectura y funcionamiento general.

Para mayor información acerca del microprocesador ATMEGA328P-PU desarrollado por Atmel AVR y utilizado en el microcontrolador Arduino UNO R3, véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.10]** para consultar por internet la hoja de especificaciones oficial (Official Datasheet) del circuito integrado.

5.4.2 Tarjeta de Desarrollo Arduino UNO R3

La tarjeta de desarrollo Arduino UNO ha recibido varias revisiones desde que fue desarrollada en el 2005, siendo la R3 la más reciente y la que se utilizó al momento de desarrollar este proyecto de tesis. Entre las características más notables de esta tarjeta de desarrollo (R3) se encuentran:

- En lugar de utilizar el FTDI USB-to-serial driver chip, el Arduino UNO R3 utiliza el Atmega16U2 programado como un convertidor USB-to-serial, el cual utiliza para la comunicación serial por interfaz USB, generalmente utilizada cando se comunica a una PC.
- Circuito RESET más robusto.
- 5 pines de energía:
  - **VIN**, utilizado para alimentar a la tarjeta Arduino UNO cuando no se está utilizando la alimentación directa de la conexión USB o la fuente de alimentación regulada del plug de 2.1 mm.
  - **5V**, pin de salida que proporciona un voltaje regulado de 5VDC extraído de la misma fuente de alimentación regulada por el plug de 2.1 mm, la conexión USB o el pin VIN.
  - **3.3V**, similar al pin 5V pero con una salida regulada de 3.3V con una corriente máxima de 50mA.
  - **GND**, pin de tierra.
  - **IOREF**, proporciona el voltaje de referencia con el cual opera el microcontrolador Arduino UNO.
- 14 pines digitales que puedes ser utilizados como salidas o entradas, operando a 5V con una corriente máxima de 40mA, con una resistencia interna pull-up de 20-50 kOhms.
- Serial pin 0 (RX) y Serial pin 1 (TX), utilizados para recibir (RX) y transmitir (TX) datos seriales TTL, a su vez conectados al convertidor serial Atmega16U2 USB-to-serial.
- Interrupciones externas en el pin 2 y pin 3.
- PWM en los pines 3, 5, 6, 9, 10 y 11.
- Comunicación SPI en los pines 10 (SS), 11 (MOSI), 12(MISO) y 13 (SCK).
- LED conectado al pin 13.
- 6 entradas analógicas con una resolución de 10 bits.
- AREF como referencia de voltaje para entradas analógicas.
- Comunicación TWI en el pin de entrada analógica A4 (SDA) y A5 (SCL).
- Oscilador cerámico de 16 MHz.

En la **Figura 5.11** se puede observar la tarjeta de desarrollo Arduino UNO R3.

Para mayor información acerca de la arquitectura, diseño y esquemático de la tarjeta de desarrollo Arduino UNO R3, véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.11]** para consultar por internet la hoja de diseño y esquemático oficial (Official Schematic & Reference Design) de la tarjeta de desarrollo.

5.4.3 Interfaz de Comunicaciones

El microcontrolador Arduino UNO R3 tiene numerosas formas de comunicarse con la PC, así como con otro Arduino y demás microcontroladores. La más utilizada es la comunicación serial proporcionada por el UART TTL (5V) del microprocesador ATMEGA328P a través de los pines digitales 0 (RX) y 1 (TX); los cuales cuentan con LEDS que encienden cada vez que se está enviando o recibiendo información serial. Además, el Arduino UNO R3 cuenta con un Atmega16U2 que funciona como convertidor Serial a USB que facilita aún más la comunicación serial con la PC, al crear un puerto virtual COM entre ellos. El firmware del Atmega16U2 utiliza el driver USB COM estándar, lo que elimina la necesidad de usar un hardware convertidor externo, permitiendo una conexión limpia entre la PC (o algún otro instrumento) y el Arduino a través de un cable USB A-B. Finalmente el software de Arduino IDE cuenta con un monitor serial para enviar y recibir información textual.

El software de control del sistema de adquisición de imágenes digitales de microscopía de fluorescencia que se desarrolló en este proyecto de tesis, utiliza la comunicación Serial previamente descrita para la transmisión de señales TTL de control hacia el microcontrolador Arduino UNO, la cual está configurada de la siguiente manera:

Baud Rate	115200
Board Type	UNO
Bytes per Packet	15
Connection Type	Serial/USB

Figura 5.14 Configuración Serial/USB del microcontrolador Arduino UNO con el software de control alojado en la PC.

Además de la comunicación Serial, el Arduino UNO R3 soporta una comunicación I2C (TWI) y SPI.

Para mayor información acerca de la comunicación del Arduino UNO R3 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.9]** para consultar por internet la página oficial del instrumento.

5.5 Obturadores Electrónicos

El sistema de adquisición de imágenes digitales de microscopía de fluorescencia cuenta con un par de obturadores electrónicos de diafragma de dos hojas marca **Vincent Associates UNIBLITZ** modelo VS25 operados a través de señales digitales TTL. Estos obturadores fueron diseñados para utilizarse en aplicaciones de microscopía, protección PMT y diversas aplicaciones de video, imágenes y fotografía en donde se requiera de una exposición repetida y precisa.

Los obturadores operan por medio de unos controladores externos, que se describen en la **Sección 5.5.1**. Estos activan los electroimanes internos de los obturadores al aplicarles una

señal eléctrica de 65 VDC sobre la bobina eléctrica de 12 Ohms, a través de un bus multipolar con una interfaz plug circular de 7 pines.

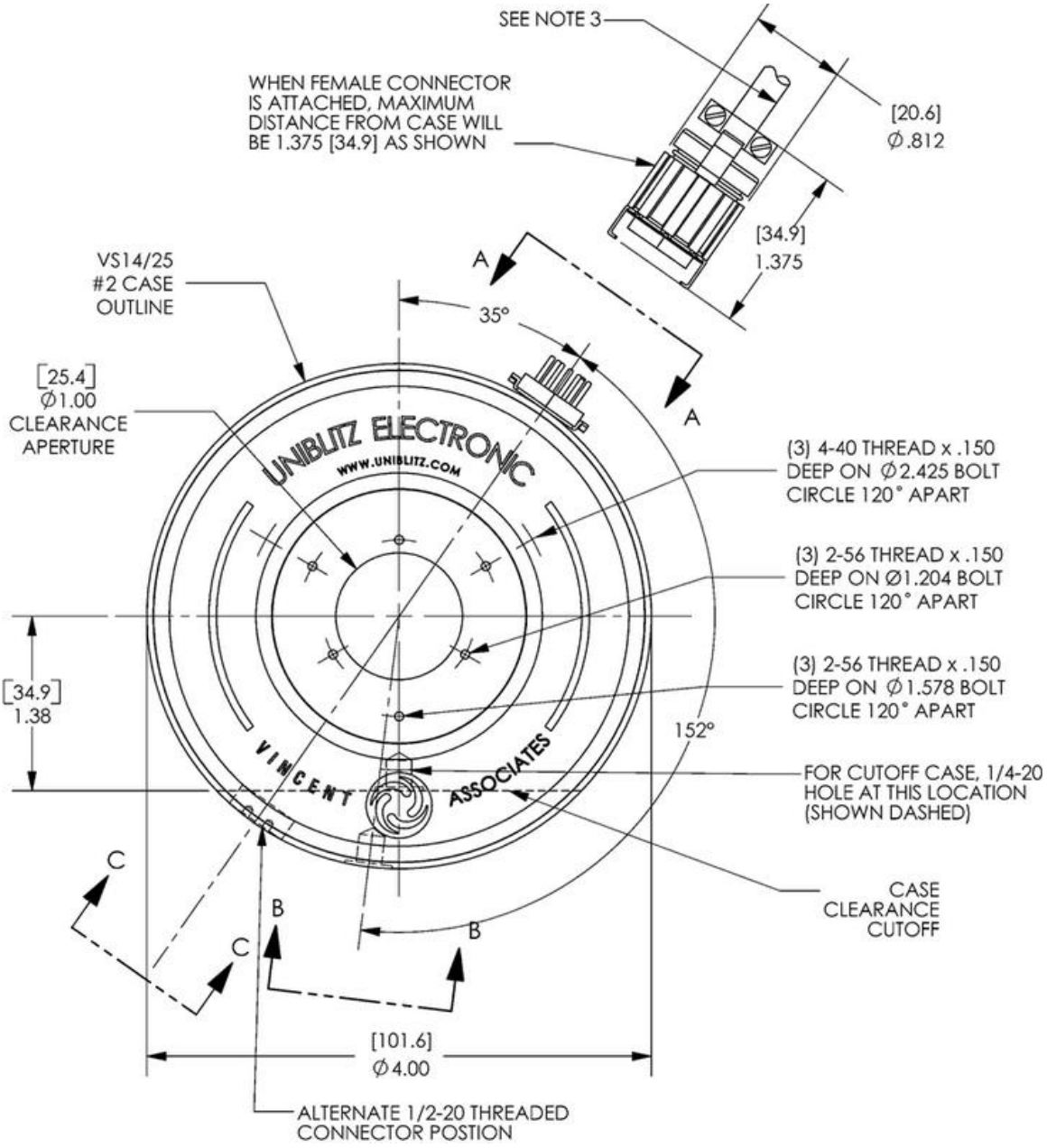
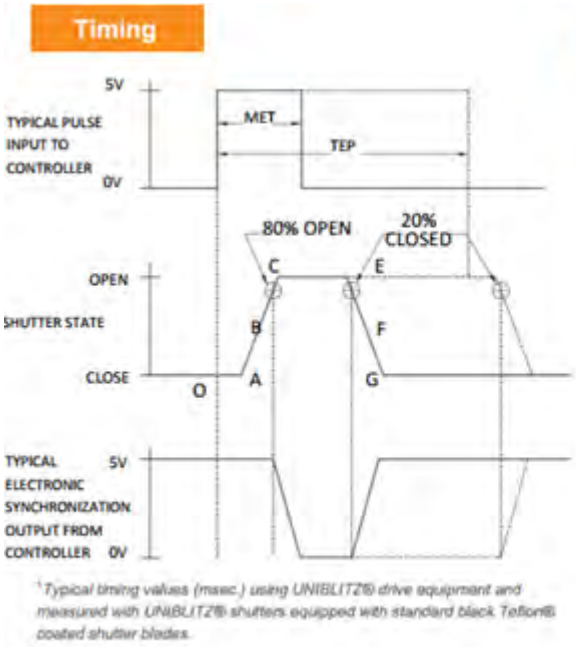


Figura 5.15 Obturador electrónico de diafragma de dos hojas, marca ***Vincent Associates UNIBLITZ*** *modelo VS25.*

En la **Figura 5.16** se muestra el diagrama de tiempos de apertura, cierre y exposición mínimos de los obturadores electrónicos, que son características importantes en este tipo de instrumentos y en especial para la aplicación que se le busca dar en el sistema de adquisición de imágenes digitales de microscopía de fluorescencia.



	VS25	Time (msec.)[1]
O-A:	Delay time on opening after current is applied	3.0
A-C:	Transfer time on opening	3.0
O-C:	Total opening time	6.0
C-E:	Min. dwell time with min. input pulse	2.0
B-F:	Min. equivalent exp. time	6.0
E-G:	Transfer time on closing	5.0
A-G:	Total window time	10.0
MET:	Min. exposure time	6.0
TEP:	Typical exposure pulse	>8.0

*Figura 5.16 Diagrama de tiempos de apertura, cierre y exposición mínimas y nominales para los obturadores marca **Vincent Associates UNIBLITZ** modelo VS25.*

Para mayor información acerca de los obturadores marca **Vincent Associates UNIBLITZ** modelo VS25 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.12]** para consultar por internet la hoja de especificaciones oficial (Official Datasheet) del instrumento.

5.5.1 Controlador de los Obturadores

El obturador electrónico es operado por medio de un controlador externo marca **Vincent Associates UNIBLITZ** modelo D122, que recibe señales TTL digitales de 5 VDC a través de su bornera frontal con 5 pines de control (**Figura 5.17 (A)**). Estos tienen las siguientes funciones:

- **CONTROL PIN 1 – Output**. Provee una señal de 8 VDC para diversas aplicaciones de control y/o comunicación a diferentes circuitos e instrumentos externos.
- **CONTROL PIN 2 – Pulse In**. Señal de entrada digital que activa los obturadores para abrir o cerrar, según la configuración establecida en el controlador D122 (NO/NC – normalmente abierto/normalmente cerrado).
- **CONTROL PIN 3 – GND**. Señal de tierra común al circuito dentro del controlador D122.
- **CONTROL PIN 4 – Input**. Funciona como un interruptor cuando el controlador D122 recibe una señal de 8 VDC y está configurado para aceptar interrupciones externas.

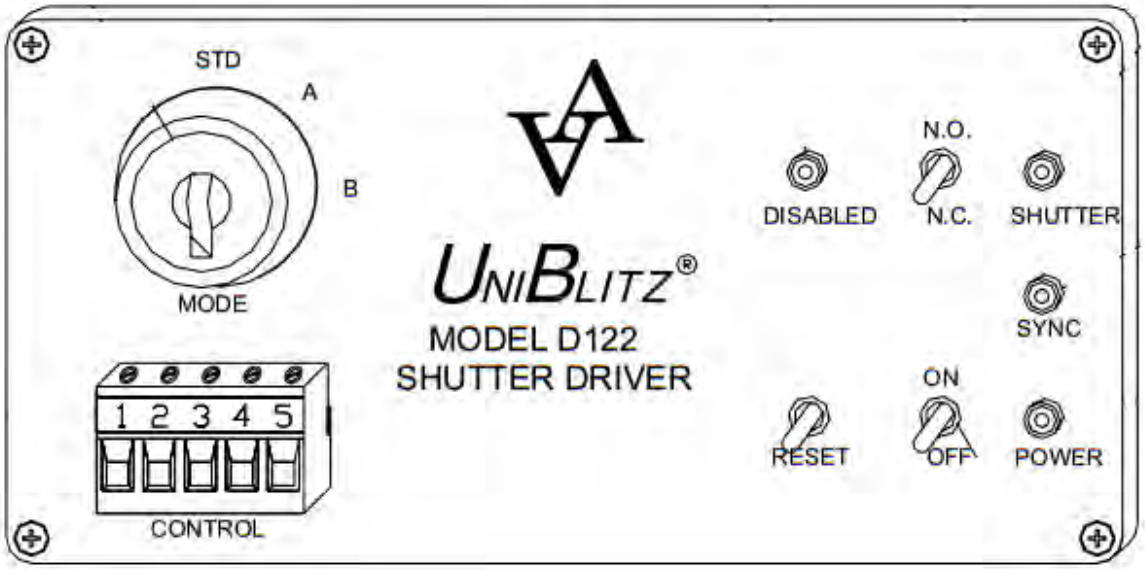
- **CONTROL PIN 5 – Output**. Cuando una interrupción está activada, este pin provee una señal digital alta hasta que la interrupción cesa.

Los controladores se comunican con los obturadores a través de un conector circular de 7 pines localizado en la parte trasera del controlador (**Figura 5.17 (B)**), transmitiendo las señales de apertura y cierre según sea el caso:

- PIN 1 – DC Input.
- PIN 2 – GND.
- PIN 3 – Pulse In.
- PIN 4, 5, 6, 7 – NC.

Finalmente poseen la capacidad de comunicarse con otros controladores para crear una red sincronizada de obturadores a través de sus diferentes conectores BNC: Pulse Input, Pulse Output, Pulse O/C Input, etc.

(A)



(B)

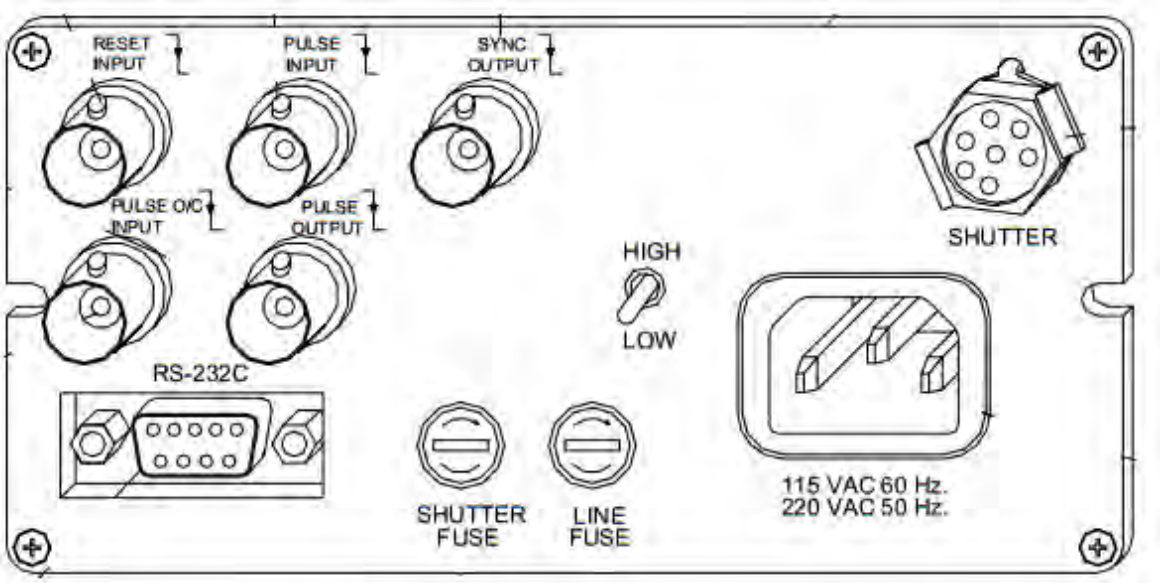


Figura 5.17 Vista frontal (A) y vista trasera (B) del controlador de obturadores marca ***Vincent Associates UNIBLITZ*** *modelo D122. En la vista frontal (A) se pueden observar los pines de control 1 al 5, mientras que en la vista trasera (B) se puede observar el conector circular de 7 pines que comunica al controlador con el obturador y los diferentes conectores BNC para sincronizar dos o más controladores, y por consiguiente obturadores, en cadena.*

Para mayor información acerca de los controladores de obturador marca **Vincent Associates UNIBLITZ** modelo D122 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.13]** para consultar por internet el manual de operación junto con las especificaciones técnicas oficiales del instrumento.

5.6 Cámara Digital

Como sensor principal en el sistema de adquisición de imágenes digitales de microscopía de fluorescencia, se utilizó una cámara digital CCD (charge-coupled device/dispositivo de carga acoplada) compacta y ligera, marca **Hamamatsu** modelo C2400-77, la cual se acopló a la ranura lateral del microscopio invertido.

La cámara CCD C2400-77 fue originalmente diseñada por **Hamamatsu** para su uso en aplicaciones de campo claro, contraste de fases y contraste de interferencia diferencial principalmente relacionadas al área médica y biológica, adecuada para la visualización y análisis de imágenes de células, tejidos y demás especímenes dinámicos y fenómenos con una cinemática de acción veloz, eliminando retrasos y distorsión al momento de registrar imágenes, al poseer un detector o sensor lineal. El sensor CCD mide 2/3 in y utiliza una transferencia interlineal de 768x494 pixeles. Finalmente la cámara tiene una tasa de muestreo de 30 Hz nominal.

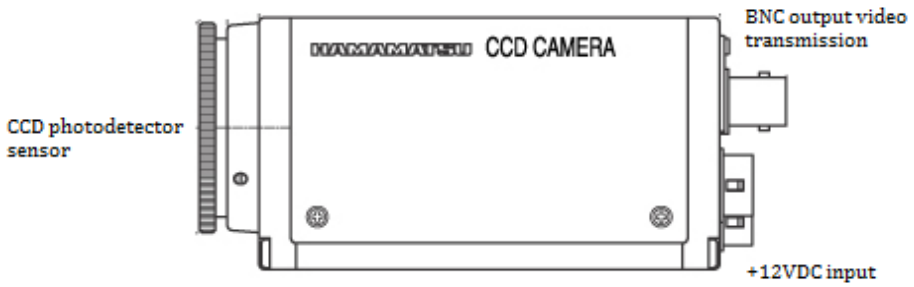


*Figura 5.18 Cámara digital CCD, marca **Hamamatsu** modelo C2400-77.*

La cámara digital CCD es alimentada con 12 VDC a través de un controlador externo descrito en la **Sección 5.6.1**, y transmite las imágenes registradas como señales eléctricas de voltaje codificadas como video compuesto, a través de un bus analógico coaxial con interfaz BNC-RCA, hasta el controlador externo o directamente hacia la tarjeta de adquisición de imágenes descrita en la **Sección 5.7**.

Al momento de desarrollar este proyecto de tesis, **Hamamatsu** dejó de dar soporte a la cámara digital CCD modelo C2400-77 en su página oficial por internet, sin embargo puede dirigirse a la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.14]** para consultar por internet una página con especificaciones técnicas generales de un distribuidor independiente.

5.6.1 Controlador de la Cámara Digital

El controlador de la cámara digital CCD, marca **Hamamatsu** modelo C2400 tiene como función principal la de proporcionar a la cámara con 12 VDC de alimentación para el funcionamiento de ésta, así como proporcionar una primera etapa de acondicionamiento y procesamiento de las imágenes registradas por la cámara cuando el bus de datos coaxial de video compuesto se conecta al controlador de la cámara y no a la tarjeta de adquisición de imágenes directamente. Algunas de las etapas de acondicionamiento y características que aporta el controlador de la cámara digital CCD al registro de imágenes son:

- Realce del contraste.
- Corrector de sombreado.
- Amplificador de video.
- Inversor de video.
- Realce de nitidez y calidad en la imagen.
- Control de ganancia automático.
- Indicador de nivel de video.
- Entrada de video compuesto.
- Sincronización externa.
- Control externo con una PC u otro instrumento a través de una interfaz RS-232C.

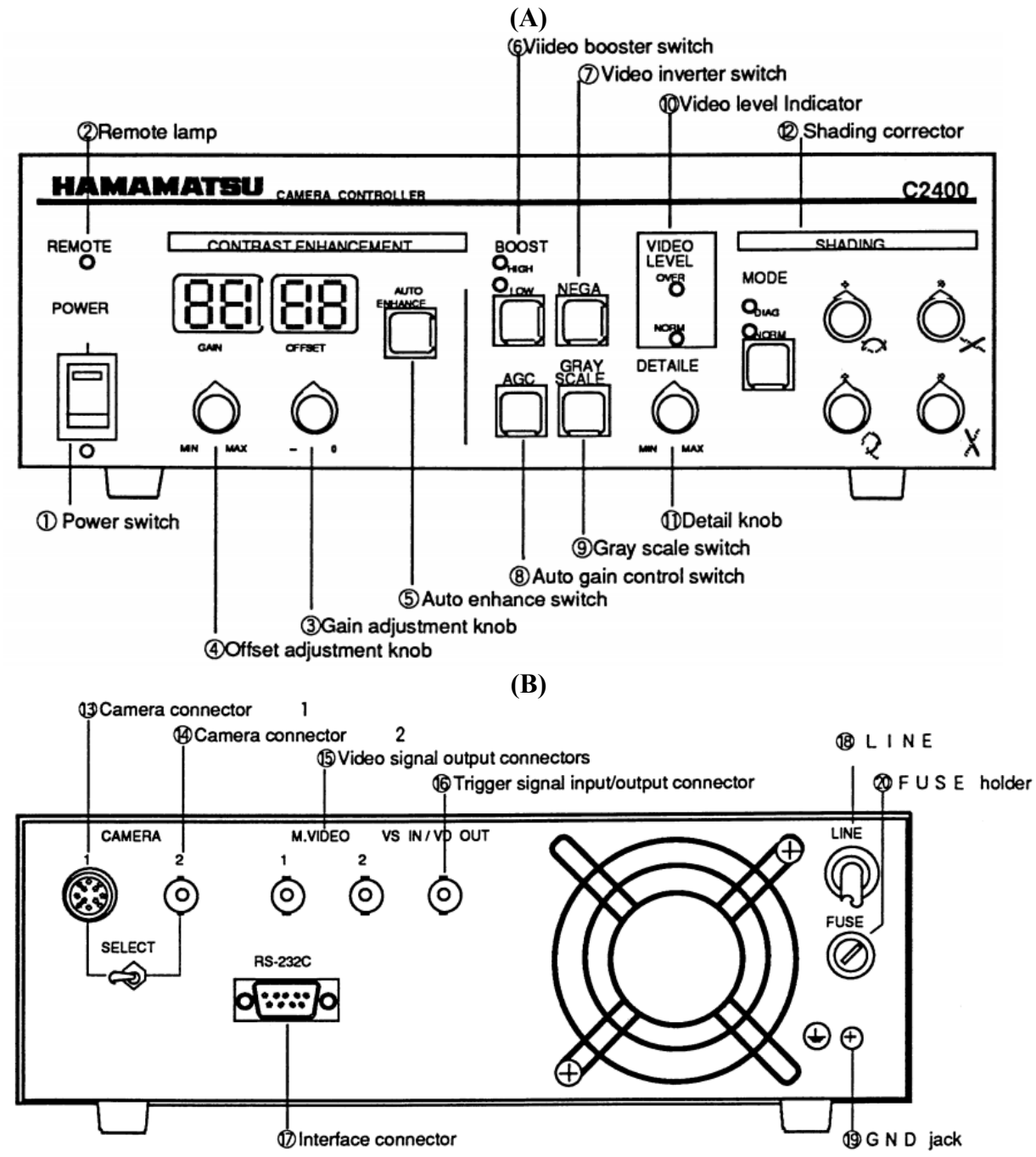


Figura 5.19 Vista frontal (A) y vista trasera (B) con nomenclatura y estructura general del controlador de la cámara digital CCD, marca ***Hamamatsu*** *modelo C2400.*

Para mayor información acerca del controlador de la cámara digital CCD marca **Hamamatsu** modelo C2400 véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.15]** para consultar por internet el manual de operación junto con las especificaciones técnicas oficiales del instrumento.

5.7 Tarjeta de Adquisición de Imágenes

El sistema de adquisición de imágenes digitales de microscopía de fluorescencia utiliza una tarjeta USB 2.0 AV Grabber, marca **Sabrent** modelo USB-ECPT para adquirir y acondicionar digitalmente las imágenes registradas por la cámara digital. La tarjeta USB 2.0 AV Grabber está diseñada para capturar video MPEG de hasta 720x576 pixeles de resolución a través de sus puertos AV y S-Video para su almacenamiento en el disco duro de la PC, también admite la adquisición de video analógico, Vidlcon, VCR y DVD a través de su puerto DV/Digital 8. Entre las principales características de la tarjeta de adquisición se encuentran:

- Transferencia de audio/video con interfaz USB 2.0 Plug & Play (conecta y utiliza).
- Driver de comunicación con la PC y Software de control y adquisición incluido en el CD de operación.
- Codificación MPEG 4/2/1 en tiempo real.

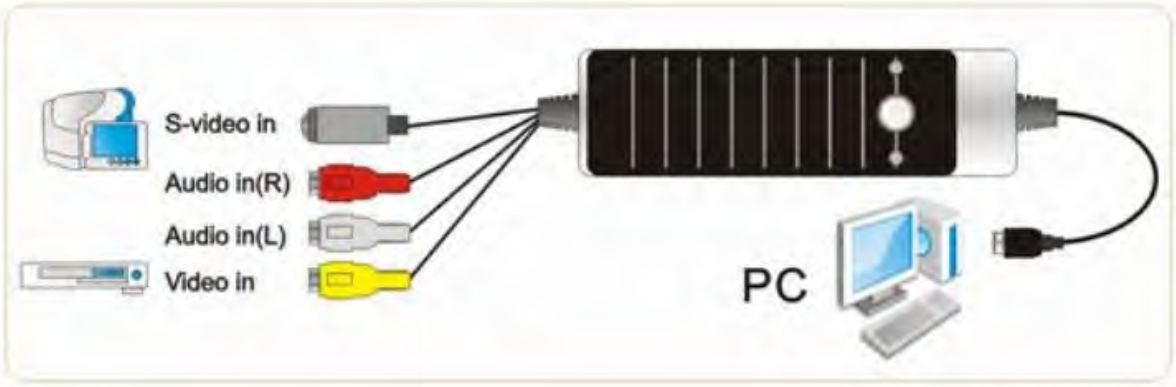


*Figura 5.20 Tarjeta de adquisición de imágenes USB 2.0 AV Grabber, marca **Sabrent** modelo USB-ECPT.*

Los requerimientos mínimos del sistema para la operación de la tarjeta de adquisición de imágenes USB 2.0 AV Grabber son:

- Procesador Pentium 4 - 1.6GHZ.
- 256 MB de memoria RAM.
- Un puerto USB (preferentemente USB2.0) disponible.
- Hardware de gráficos que soporten DirectX 9.0c.
- Hardware de sonido compatible con códec AC97 de audio.
- 1 GB de espacio en disco duro.
- SO Microsoft Windows XP / Vista.

Para mayor información acerca de la tarjeta de adquisición de imágenes USB 2.0 AV Grabber, marca **Sabrent** modelo USB-ECPT véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.16]** para consultar por internet el manual de operación junto con las especificaciones técnicas oficiales del instrumento.

5.8 Unidad de Procesamiento Central (PC)

La PC representa el sostén del sistema de adquisición de imágenes digitales de microscopía de fluorescencia, aunque ésta representa la unidad de procesamiento y control de todo el sistema, únicamente opera y se comunica de manera directa con la tarjeta de adquisición de imágenes USB 2.0 AV Grabber y el microcontrolador Arduino UNO R3 a través de dos puertos USB, los demás instrumentos se sincronizan de manera indirecta en cuanto a operación y comunicación se refiere a través de los dos instrumentos antes mencionados y del software de control desarrollado en la plataforma NI LabVIEW e instalado en la PC, por lo que el funcionamiento y rendimiento de ésta última son de vital importancia para que el sistema sea operado y sincronizado correctamente. Los requerimientos mínimos que debe satisfacer la PC son los siguientes:

- Procesador Intel Core2/ AMD Phenom de cuatro núcleos a 2.2GHz o superior.
- 2 GB de memoria RAM.
- Dos puertos USB (preferentemente USB2.0) disponibles.
- Hardware de gráficos que soporten DirectX 9.0c.
- Hardware de sonido compatible con códec AC97 de audio.
- 5 GB de espacio libre en disco duro.
- SO Windows XP Pro SP3 o superior.
- Monitor con resolución 1024x768 pixeles o superior.
- Periféricos de la computadora: teclado y mouse.

Marcas y modelos de los dispositivos que cumplan con las especificaciones mínimas descritas anteriormente, permanecen a criterio del usuario.

5.8.1 Plataforma NI LabVIEW y Paquetes de Desarrollo

Adicional al Software de Arduino IDE y su driver de comunicación por interfaz USB con la PC que simula una comunicación serial tipo COM; y el driver de comunicación de la tarjeta de adquisición de imágenes digitales USB 2.0 AV Grabber por interfaz USB con la PC; se requiere la instalación del software NI LabVIEW así como distintos paquetes de desarrollo que integran las librerías y drivers necesarios para la comunicación con los diferentes dispositivos y así operar en su totalidad el sistema de adquisición de imágenes digitales de microscopía de fluorescencia. Estos paquetes de desarrollo son:

- **NI LabVIEW 2014 f1 Professional Development System**, el cual incluye:
  - NI LabVIEW Run-Time 2014 f1
  - NI LabVIEW Application Builder 2014
  - Measurement & Automation Explorer 14.0.1

Como ya se mencionó anteriormente, NI LabVIEW es la plataforma de desarrollo utilizada para programar el software de control para el sistema de adquisición de imágenes digitales de microscopía de fluorescencia; y es la base para los siguientes paquetes de desarrollo:

- **NI Vision Acquisition Software 2014 f1**, el cual incluye:
  - NI IMAQ 14.0: Development Support 14.0.0 & Run-Time 14.0.0
  - NI IMAQdx 14.0: Development Support 14.0.0 & Run-Time 14.0.0

NI Vision Acquisition Software, es un paquete de drivers y librerías básicas y avanzadas que permiten comunicar cámaras e instrumentos de adquisición de imágenes tanto desarrolladas por NI como por algún otro vendedor con el software de desarrollo LabVIEW, a través de los diferentes puertos de comunicación de la PC.

- **NI VISA 14.0.1**, el cual incluye:
  - NI VISA Run-Time 14.0.1

NI VISA, es un paquete de drivers y librerías básicas y avanzadas que permiten comunicar instrumentos con comunicación tipo serial tanto desarrollados por NI como por algún otro vendedor con el software de desarrollo LabVIEW, a través de los diferentes puertos seriales de la PC.

- **NI LabVIEW (2014) Interface for Arduino v2.2.0.79**

NI LabVIEW Interface for Arduino, es un paquete de librerías básicas y avanzadas que permiten comunicar y controlar con mayor facilidad el microcontrolador Arduino UNO con el software de desarrollo LabVIEW, a través de los puertos USB y/o seriales de la PC. Es necesario contar con NI VISA instalado para el correcto uso de éste paquete. También proporciona el programa en leguaje C (firmware) para ser grabado en la memoria del microprocesador ATMEGA328P del Arduino UNO, y el cual necesario para comunicarlo con el software LabVIEW.

Los requisitos mínimos del sistema mencionados en la **Sección 5.8** son suficientes para instalar y ejecutar tanto la plataforma de desarrollo NI LabVIEW como los paquetes de desarrollo. Para información detallada acerca de los requisitos mínimos del sistema, soporte técnico especializado así como las características particulares de cada uno de los paquetes de desarrollo, véase la sección **Referencias Bibliográficas**, **Capítulo 5 ítem [5.17]** y **[5.18]** para consultar por internet la página oficial de National Instruments y sus diferentes productos.

5.9 Metodología Propuesta

La metodología a utilizar se basa en el concepto básico de la instrumentación virtual revisada en la **Sección 2.3**, basada en la sincronización, control y monitoreo de los instrumentos y demás hardware del sistema a través de un software ejecutado desde una unidad de procesamiento y control central (PC), esta metodología permite utilizar al sistema hasta dos caminos ópticos, cada uno con diferentes longitudes de onda y ambas partiendo de una misma fuente de luz, alternando las distintas longitudes de onda por medio de dos obturadores electrónicos, controlando el periodo de apertura, tiempo de exposición, adquisición de imágenes por la cámara, su almacenamiento y despliegue en la computadora a través del ya mencionado software de control desarrollado e instalado en la PC. La automatización de este procedimiento ha permitido crear un **protocolo de experimentación** que permite estudiar la liberación de neurotransmisores a partir de diferentes compartimientos celulares y sus mecanismos en neuronas identificadas. A continuación se muestra el diagrama de bloques del instrumento virtual y el diagrama gráfico que ilustra el funcionamiento general del sistema de adquisición de imágenes digitales de microscopía de fluorescencia.

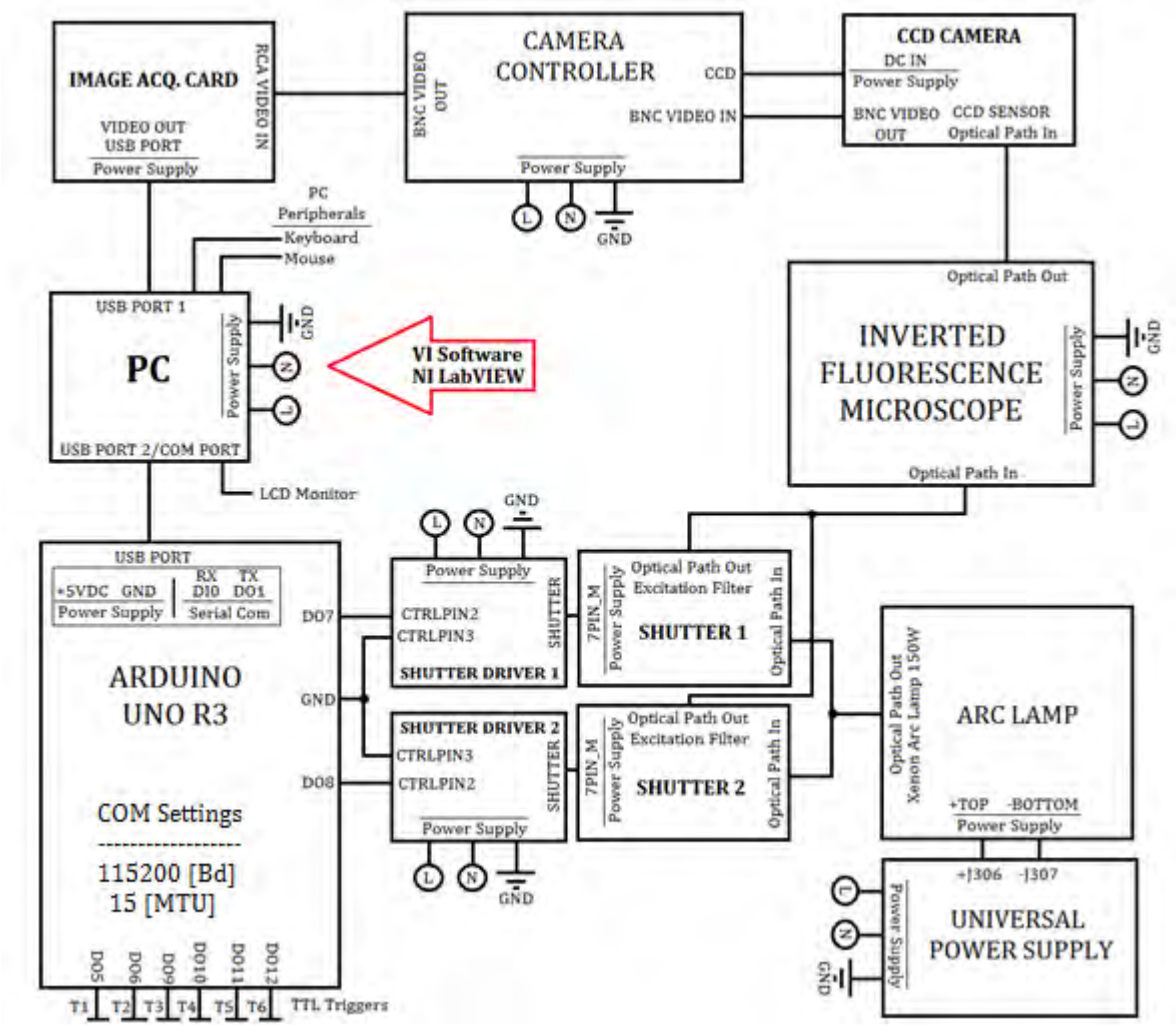


Figura 5.21 Diagrama de bloques del instrumento virtual.

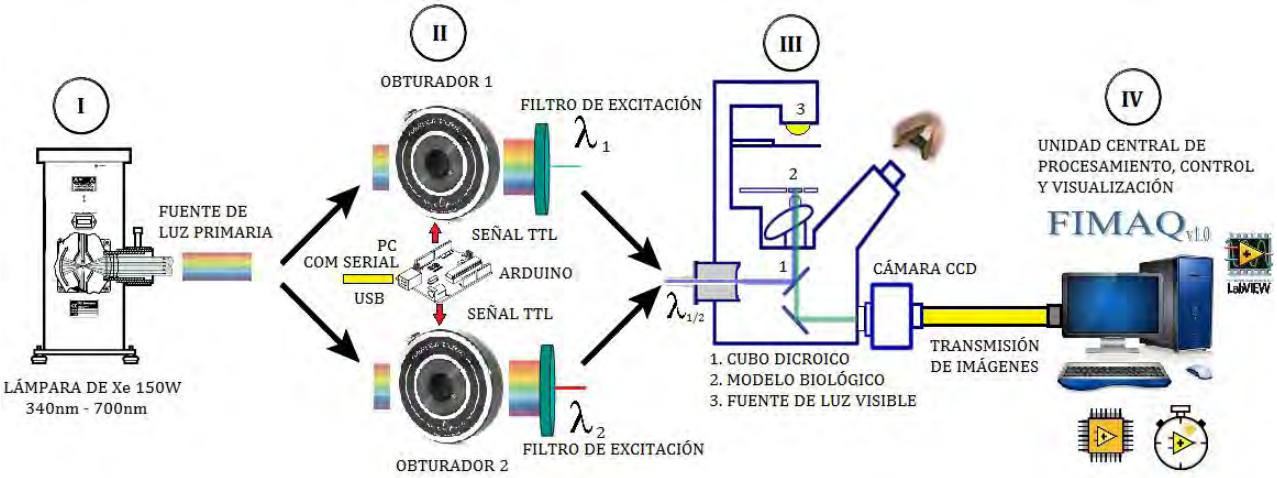


Figura 5.22 Diagrama gráfico que ilustra el funcionamiento del sistema de adquisición de imágenes digitales de microscopía de fluorescencia.

Apoyado en el diagrama de la **Figura 5.22**, la metodología propuesta es la siguiente:

I. La lámpara de fluorescencia, alojando un bulbo de Xenón de 150 W, emite luz con una longitud de onda dentro de un rango de 340nm – 700nm a través de dos conductos ópticos aislados y separados entre sí, que dirigen la luz hasta los obturadores electrónicos, los cuales cuentan con un filtro de excitación que selecciona una longitud de onda específica, dentro del amplio rango de longitudes de onda que emite la lámpara de fluorescencia.

II. El microcontrolador Arduino UNO, a través de un bus de datos tipo Serial con interfaz USB, se comunica con el instrumento virtual ejecutándose en la PC para bloquear o desbloquear automáticamente estos caminos ópticos por medio señales TTL enviadas a los controladores de los obturadores electrónicos, a un periodo y tiempo de exposición específica para cada uno de éstos, lo que permite enviar una u otra longitud de onda de excitación al modelo biológico sin riesgo de colisiones entre ellas.

III. La longitud de onda de excitación en turno, entra por el camino óptico interno del microscopio de fluorescencia invertido y es reflejada por el cubo dicroico directamente sobre el modelo biológico, provocando la fluorescencia. La luz emitida por el fluoróforo presente en el modelo biológico y la luz que no fue absorbida por éste, así como cualquier otra luz parásita dentro del microscopio, pasan a través del cubo dicroico, filtrando únicamente la luz emitida por el fluoróforo del modelo biológico la cual viaja hasta los binoculares del microscopio y hacia el detector de la cámara digital CCD acoplada al mismo.

IV. La cámara CCD codifica la luz que recibe en una señal analógica de voltaje de video compuesto, la cual es transmitida por medio de un bus coaxial de audio/video hasta la tarjeta de adquisición de imágenes por medio de una interfaz

RCA la cual está acoplada a su vez a la PC a través de una interfaz USB. La tarjeta de adquisición se comunica con el instrumento virtual para digitalizar la señal y enviarla a la computadora para su procesamiento en información comprensible para el usuario, almacenamiento en memoria, despliegue en pantalla y posterior análisis.

El investigador controla el desarrollo del experimento a través del software de control que se ejecuta en la computadora, desde la ruta y formato de almacenamiento para las imágenes adquiridas, hasta el número total y orden de adquisición de éstas. El software integra un sistema de gestión de recursos informáticos y de imágenes adquiridas, así como una interfaz de usuario gráfica muy amigable, intuitiva y completa en sus funciones, dedicada al despliegue y monitoreo de resultados, manipulación del periodo de apertura y tiempo de exposición, visualización y adquisición de imágenes a bajas y altas frecuencias, sincronización de señales TTL, configuración avanzada de hardware y programación de plantillas experimentales de microscopía de fluorescencia.

Capítulo 6

Descripción del Software de Control

El software de control, llamado **FIMAQ** por sus siglas en inglés **Fluorescence Image Acquisition**, se desarrolló a través de un paradigma de programación modular, compuesto por una rutina, módulo o programa principal llamado ***HMI.vi*** (Human Machine Interface), que además funge como la interfaz de usuario principal el cual divide al software en dos modos de operación: visualización y adquisición; y una librería llamada **IMAQ.lvlib** que contiene 23 módulos o subrutinas de programación (sub .vi's), que en conjunto se encargan de ejecutar las cuatro funciones principales mencionadas en la **Sección 1.1**, a saber:

- Control automático de los obturadores electrónicos,
- Adquisición de imágenes digitales,
- Almacenamiento de imágenes y gestión de recursos informáticos, y
- Despliegue de imágenes y resultados.

Las tareas que realiza el software de control son las siguientes:

- Configuración automática de ambos puertos de comunicación USB: tarjeta Arduino UNO y tarjeta de adquisición de imágenes AV Grabber.
- Creación de archivos de configuración.
- Visualización y manipulación de imágenes digitales.
- Adquisición manual y automática de imágenes y secuencias de imágenes digitales.
- Creación y administración de rutas de trabajo y almacenamiento.
- Visualización en tiempo real de imágenes de microscopía.
- Operación individual o simultánea de hasta 2 obturadores electrónicos.
- Control sobre los tiempos de exposición de los obturadores electrónicos.
- Control sobre el periodo de apertura de los obturadores electrónicos (véase **Anexo C: Pruebas de Tiempo** en la **Sección Anexos** para mayor información acerca de los tiempos de operación de los obturadores, las señales de control y la adquisición de imágenes).
- Configuración y ejecución de sesiones experimentales de microscopía de fluorescencia con control automático de obturadores y adquisición de secuencias de imágenes (véase **Sección 5.9** y **Sección 6.2.10** para mayor información sobre la metodología y algoritmo de programación respectivamente sobre este procedimiento experimental).

- Control de hasta 6 señales digitales para su uso como Triggers TTL durante las sesiones experimentales de microscopía de fluorescencia.
- Monitoreo en tiempo real de parámetros de adquisición e imágenes digitales en sesiones experimentales de microscopía de fluorescencia.
- Creación de archivos de registro de las sesiones experimentales.
- Generación de reportes escritos de las sesiones experimentales.

La rutina principal del software de control se ejecuta dentro de un ciclo *while* recurrente, llamando a las funciones y subrutinas por medio de una *estructura de eventos* para realizar las tareas y comandos que el usuario ejecuta desde la interfaz de usuario en cualquiera de sus dos modos de operación.

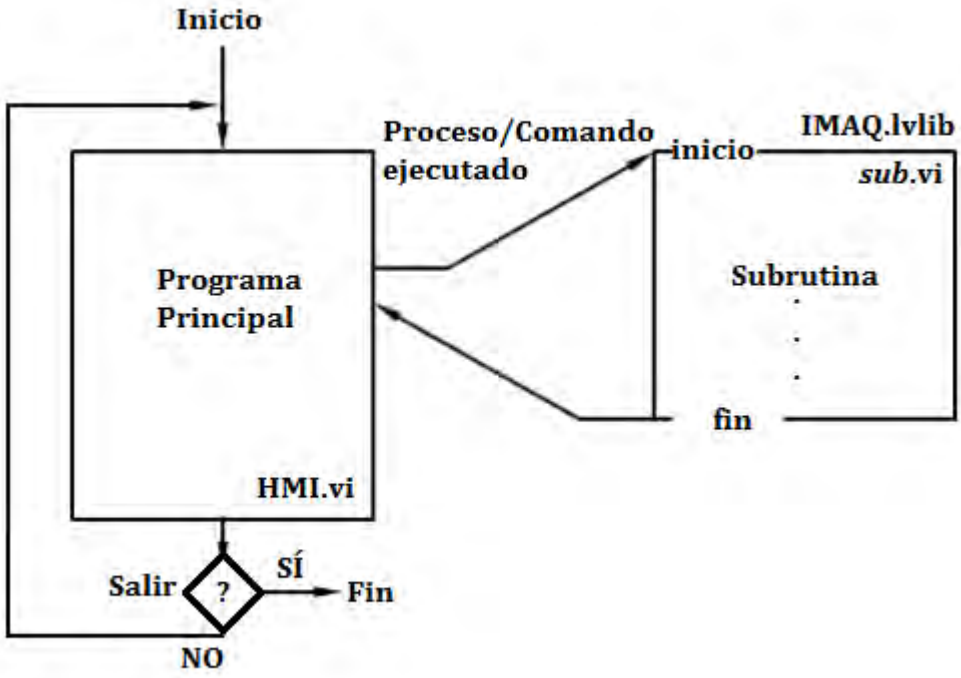


Figura 6.1 Estructura del software de control.

El software de control también utiliza una librería interna llamada **vi.lib** con más de 60 sub .vi's pre programados y 3 librearías .dll internas pre programadas llamadas **niimaqdx.dll, nivision.dll** y **nivissvc.dll** incluidas en la arquitectura base de LabVIEW y sus distintos paquetes de desarrollo, necesarias para la ejecución de funciones básicas como sumar, restar, construir gráficas y demás tareas avanzadas de procesamiento de datos, asegurando el correcto funcionamiento del software de control. Finalmente se construyó un archivo ejecutable .exe llamado **FIMAQ.exe** para acceder al software de control como una aplicación desde un acceso directo del escritorio de Windows o cualquier otra ubicación, que ejecuta la rutina principal de programación *HMI.vi*. Tanto los .vi's, como las librerías y el archivo ejecutable, se encuentran dentro de un administrador de proyectos llamado **FIMAQ.lvproj**, desde donde se manipula la programación y estructura general e interna del software.

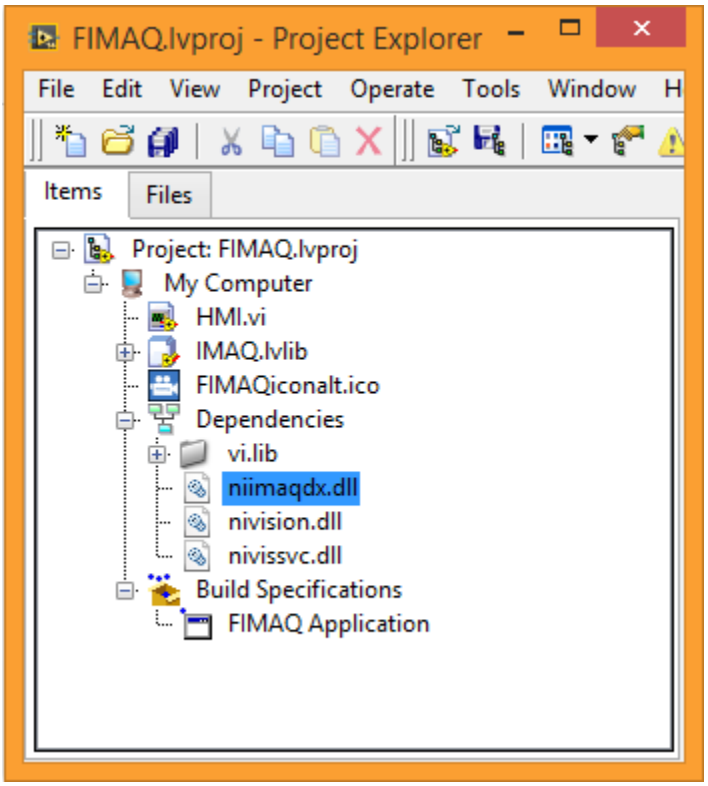


Figura 6.2 Proyecto FIMAQ.lvproj que engloba todos los archivos de programación (.vi), librerías internas programadas y pre programadas y el archivo ejecutable del software de control FIMAQ.

El presente capítulo describe el algoritmo de programación, función y características de cada uno de los 24 módulos programados que integran el software de control FIMAQ desarrollado en este proyecto de tesis, divididos en una rutina principal descrita en la **Sección 6.1** y 23 subrutinas descritas en la **Sección 6.2** sin un orden específico. En la **Sección 6.3** se describe la estructura, uso y características de la interfaz de usuario del software de control. Véase **Anexo A: FIMAQ Manual de Usuario** y **Anexo B: FIMAQ Código Fuente** en la **Sección Anexos** para consultar tanto el manual de usuario y acceder al código fuente completo del software de control FIMAQ.

Finalmente el presente capítulo no ofrece de ninguna manera una guía o tutorial de programación, manual de usuario o información detallada sobre el funcionamiento, librearías y/o uso general o específico del software NI LabVIEW ni sus paquetes de desarrollo, para obtener información sobre estos temas, véase la sección **Referencias Bibliográficas**, **Capítulo 6 ítem [6.1]** para consultar por internet la página oficial de National Instruments en donde encontrará soporte técnico especializado, guías de ayuda, y podrá revisar una variedad de manuales y tutoriales sobre el uso general y específico de NI LabVIEW y sus diferentes productos y paquetes de desarrollo, incluidos aquellos mencionados en la **Sección 5.8.1**

6.1 Descripción del Programa Principal: HMI.vi

El módulo *HMI.vi* es el programa principal de ejecución e interfaz de usuario del software de control, por lo que su estructura de programación es similar a la de un *int main (void)* en lenguaje C. Su ejecución inicializa el software de control y su cierre lo finaliza, por lo que el programa no puede ser llamado en más de una ocasión durante una sesión de ejecución del software de control (a diferencia de las subrutinas que pueden ser llamadas varias veces durante una misma sesión de ejecución). Cuenta con un ciclo de programación recurrente *while* interno que lo mantiene activo y en espera de las tareas y comandos del usuario; manejando una serie de variables y eventos internos, y llamando cuando se requiere a las subrutinas y funciones programadas para la ejecución de las tareas mencionadas al comienzo del presente capítulo.

Entre sus funciones se encuentran la de inicializar la configuración de los puertos físicos, librerías de programación y variables internas; proporciona al usuario los comandos de configuración, visualización, adquisición e inicio de experimento necesarios para operar el software de control desde la interfaz de usuario del panel frontal del instrumento virtual (véase **Sección 6.3**); funge como visualizador principal de imágenes almacenadas en la PC e imágenes de microscopía en tiempo real; administra las rutas de trabajo y de almacenamiento; y en el cierre del software de control, finaliza el uso de los puertos físicos, librerías de programación y libera el espacio en memoria de las variables internas.

Finalmente el programa estructura la interfaz de usuario en dos modos de operación accesibles desde el panel frontal del instrumento virtual: visualización y adquisición, descritos en la **Sección 6.3.3 y Sección 6.3.4** respectivamente.

6.1.1 Descripción del algoritmo

El algoritmo de programación para el módulo principal HMI.vi es el siguiente:

1. El programa inicializa las variables internas del software de control, asignándoles valores por defecto y limpiando cualquier registró previo en memoria de las mismas.
2. Inicializa los elementos visibles en el panel frontal del instrumento virtual, ocultando aquellos sin relevancia o prohibidos para el usuario.
3. Carga/Crea la carpeta y ruta de trabajo raíz en el disco duro de la PC (*C:\FIMAQ*).
4. Si es la primera vez que se ejecuta el software de control en la PC, establece con ayuda del usuario los puertos USB por defecto para el microcontrolador Arduino UNO y la tarjeta de adquisición de imágenes AV Grabber y los guarda en un archivo de configuración en la ubicación raíz *(C:\FIMAQ\DATA\SETUP*). De otra manera, el programa carga el archivo de configuración desde la ubicación raíz.
5. Se configura e inicia la adquisición de imágenes con la tarjeta AV Grabber.
6. Se configura e inicia la comunicación serial con el microcontrolador Arduino UNO.

7. Se inicia un ciclo *while* recurrente que contiene la *estructura principal de eventos* con 22 comandos y un Timeout de 1[ms].
8. Los 22 comandos programados son llamados a través de los controles y botones del panel frontal de la interfaz de usuario principal, y están asociados a las siguientes funciones:

[01] – Iniciar experimento.
[02] – Ajuste numérico en el *tiempo de exposición* (te [ms]) y *periodo de apertura* (Ta [ms]) del obturador 1.
[03] – Ajuste numérico en el *tiempo de exposición* (te [ms]) y *periodo de apertura* (Ta [ms]) del obturador 2.
[04] – Guardar imagen adquirida en la ruta de trabajo raíz.
[05] – Guardar imagen adquirida en la ruta especificada por el usuario.
[06] – Cargar archivo de registro de experimento en la ventana 1 y/o 2 del modo visualización.
[07] – Respaldar reporte escrito y/o secuencia de imágenes del archivo de registro de experimento cargado en la ventana 1 y/o 2 del modo visualización.
[08] – Visualizar imágenes almacenada en la PC o en el archivo de registro de experimento en la ventana 1 del modo visualización.
[09] – Visualizar imágenes almacenada en la PC o en el archivo de registro de experimento en la ventana 2 del modo visualización.
[10] – Navegación de las imágenes desplegadas en la ventana 1 y/o 2 del modo visualización.
[11] – Iniciar visualización continua del total de imágenes desplegadas en la ventana 1 y/o 2 del modo visualización.
[12] – Modifica ruta de trabajo raíz.
[13] – Modifica nombre por defecto de las imágenes adquiridas.
[14] – Modifica formato de guardado de las imágenes adquiridas.
[15] – Activar opción de autonombrar imágenes adquiridas.
[16] – Cargar/Establecer puertos físicos USB a utilizar por defecto.
[17] – Guardar nueva configuración de puertos físicos USB a utilizar por defecto.
[18] – Activa el uso del Trigger TTL en el *puerto* y *número de imagen* especificado en la configuración del experimento.
[19] – Muestra el estado (activo/no activo) del Trigger TTL seleccionado.
[20] – Ajuste numérico en la configuración *número de imagen* al activar el uso de los Triggers TTL.
[21] – Activa/desactiva el uso de ambos obturadores durante el experimento, ocultando/mostrando los controles e indicadores de configuración correspondientes al obturador 2.
[22] – Finalizar software de control.

Estos eventos se pueden agrupar según sus funciones generales de la siguiente manera:

[16,17] - Configuración de los puertos USB.

[2,3] - Configuración del periodo de apertura y tiempo de exposición de los obturadores.
[4,5,6,8,9] - Guardado y carga de imágenes.
[10,11] - Manipulación de las imágenes cargadas.
[1,12,13,14,15,20,21] - Configuración e inicio de experimento.
[7] - Generación de reportes escritos y archivos de registro de experimentos.
[18,19] - Configuración de los Triggers TTL durante el experimento.
[22] - Cierre del software.

9. El Timeout es un evento especial dentro de la *estructura de eventos* el cual se ejecuta automáticamente cada 1 [ms] sin que el usuario tenga que presionar algún botón o control desde el panel frontal, cuya función es mantener al software en modo de espera. Éste contiene una *estructura de casos* con dos funciones programadas:

[Caso 0] – Visualización: oculta los controles e indicadores que operan la adquisición de imágenes y configuración e inicio de experimentos y habilita los controles e indicadores que operan la visualización y manipulación de imágenes y registros de experimentos. Despliega las imágenes seleccionadas en pantalla.
[Caso 1] – Adquisición: oculta los controles e indicadores que operan la visualización y manipulación de imágenes y registros de experimentos y habilita los controles e indicadores que operan la adquisición de imágenes y configuración e inicio de experimentos. Despliega en tiempos la real la adquisición de imágenes de microscopía.

10. Desde el comando [22] se finaliza la *estructura de eventos* y el ciclo *while*.
11. Se finaliza la sesión de adquisición y se cierra la comunicación con la tarjeta AV Grabber.
12. Se finaliza la comunicación serial con el microcontrolador Arduino UNO.
13. Se vacía la memoria asignada a las variables internas.
14. Se finaliza el software de control junto con cualquier función o subrutina en segundo plano y se cierra LabVIEW.

6.1.2 Diagrama de Flujo

En la **Figura 6.3** se muestra el diagrama de flujo que ilustra el algoritmo de programación, estructura y funcionamiento general del módulo principal de programación HMI.vi:

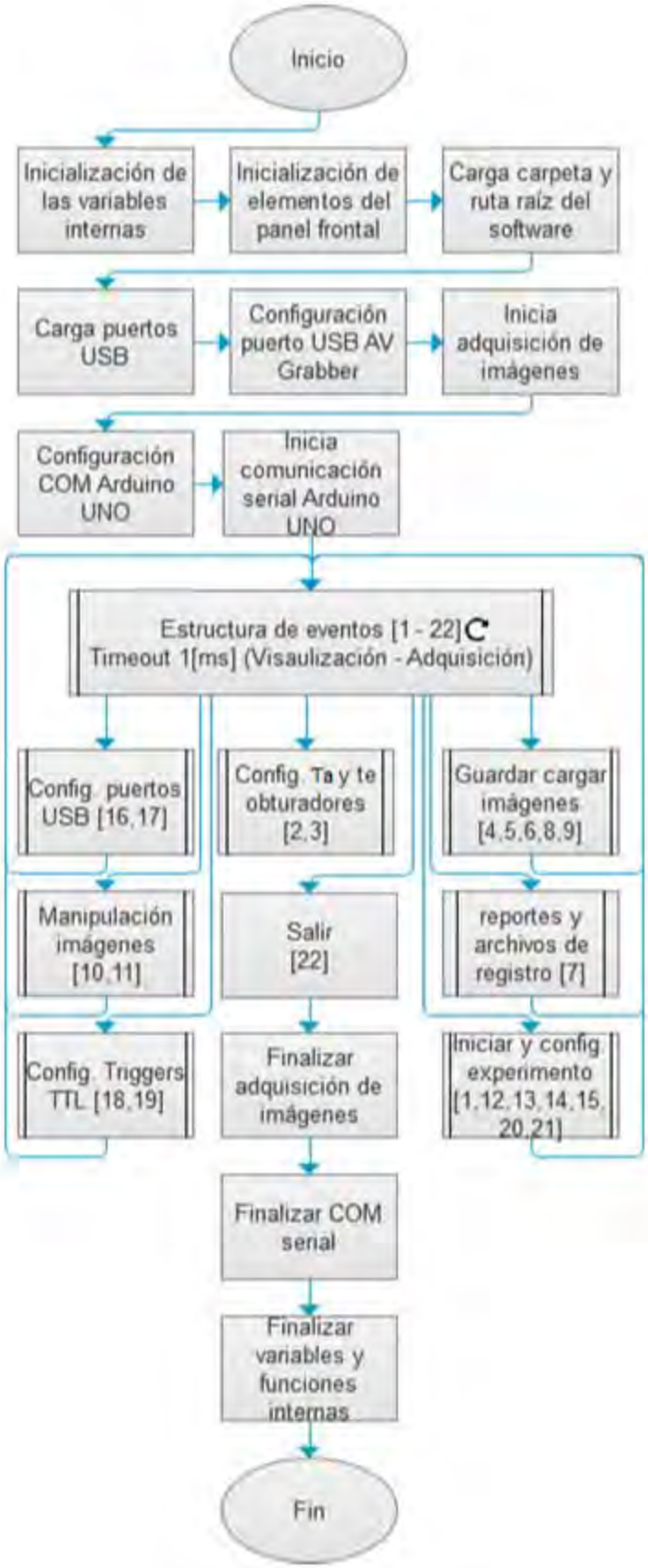


Figura 6.3 Diagrama de flujo del programa principal HMI.vi

6.1.3 Parámetros de Entrada y Salida

El archivo de programación no recibe parámetros como si se tratara de una función o subrutina, y tampoco devuelve un dato como tal, simplemente se encarga de ejecutar las tareas que el usuario opere durante la sesión de uso del software de control, llamado funciones y subrutinas en el caso en que sean necesarias, una vez que el usuario finaliza su operación éste termina cualquier subrutina o función y la ejecución del software de control finaliza sin regresar algún dato o resultado al usuario.

Figura 6.4 Parámetros E/S del programa principal HMI.vi, al ser el módulo principal de ejecución y de interfaz de usuario, es un módulo sin entradas ni salidas de parámetros. El programa maneja una serie de variables y comandos internos durante la ejecución del software de control, pero jamás recibe ni requiere que el usuario proporcione parámetros de entrada al momento de ejecutar el programa, de la misma manera no arroja resultados ni datos alguno al momento de finalizar.

6.2 Descripción de las Subrutinas de Programación

La programación modular es un paradigma de programación que consiste en dividir un programa en módulos o subprogramas con el fin de hacerlo más legible y manejable, dividiendo un problema de programación complejo en varios sub problemas más simples, y estos a su vez en otros sub problemas más simples si es necesario, facilitando así la solución del problema de programación.

Las subrutinas representan módulos secundarios de programación del software de control, las cuales se encuentran inmersas y distribuidas a lo largo del código fuente del módulo principal *HMI.vi* y son llamadas una o más veces de manera individual o conjuntamente por éste para cumplir funciones y tareas específicas del software de control o para apoyar al procesamiento y/o configuración de datos. En total se programaron 23 módulos secundarios contenidos en la librería **IMAQ.lvlib**.

La descripción de las subrutinas que integran el software de control se realiza sin un orden específico y su estructura es de la siguiente manera: descripción general de sus funciones y características, descripción del algoritmo de programación, diagrama de flujo y descripción de sus parámetros de entrada y salida.

6.2.1 Subrutina: *LoadConfig.vi*

La subrutina *LoadConfig.vi* carga el archivo de configuración *SETUP* de la carpeta raíz del software de control *C:\FIMAQ* para obtener la información referente a los puertos físicos USB por defecto que se utilizarán para configurar la comunicación serial con el microcontrolador Arduino UNO y con la tarjeta de adquisición de imágenes AV Grabber. Si

es la primera vez que se ejecuta el software de control o el archivo de configuración y/o la carpeta raíz fueron borrados o se movieron de su ubicación, se crea la carpeta raíz (si es necesario) y se llamada a la subrutina *SetUp.vi* (**Sección 6.2.2**) para establecer los puertos USB por defecto con ayuda del usuario, después los guarda en un archivo de configuración el cual almacena en la carpeta raíz. El algoritmo de programación es el siguiente:

1. Se verifica la existencia de la carpeta raíz del software de control. Caso afirmativo se accede a ella, caso contrario se crea.
2. Se verifica la existencia del archivo de configuración. Caso afirmativo se accede a él, casi contrario se crea.
3. Se extrae la información referente a los puertos físicos USB por defecto a utilizar del archivo de configuración existente. Caso contrario se llama la subrutina *SetUp.vi* y se establecen con ayuda del usuario.
4. Se cierra el archivo de configuración. En caso de haber sido creado se guarda en la carpeta raíz y se cierra el archivo.
5. La subrutina finaliza.

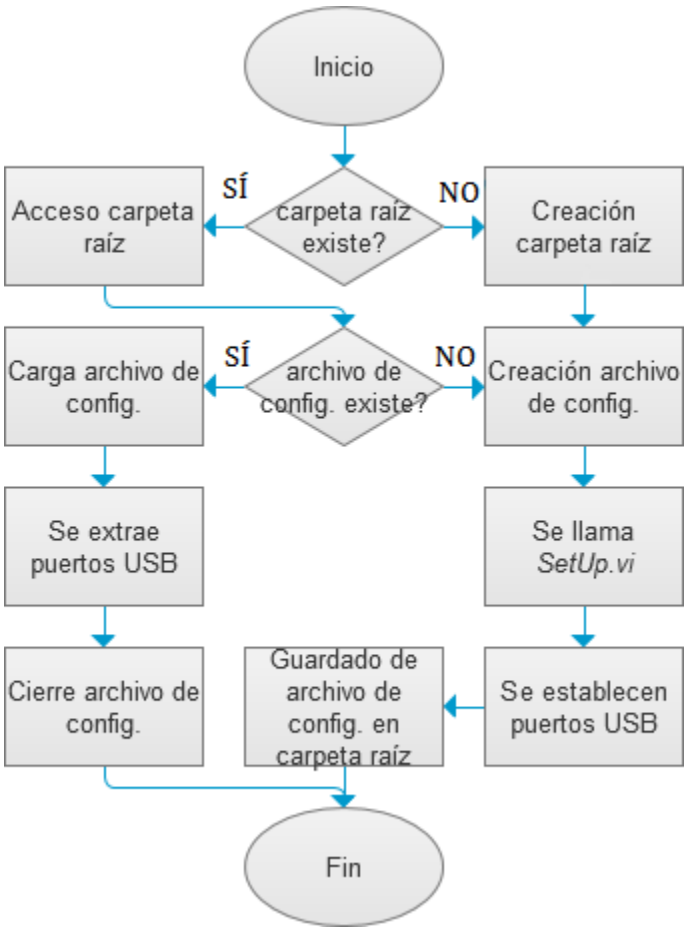


Figura 6.5 Diagrama de flujo de la subrutina LoadConfig.vi

La subrutina *LoadConfig.vi* no posee parámetros de entrada y cuenta con 2 parámetros salida. Estos son:

Parámetros de Salida

Nombre	Tipo de dato	Descripción
cam	typedef IMAQdx.ctl	Referencia al puerto USB de la tarjeta de adquisición de imágenes
arduino	typedef IMAQdx.ctl	Referencia al puerto USB del microcontrolador Arduino UNO

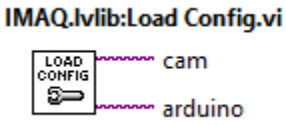


Figura 6.6 Parámetros E/S de la subrutina LoadConfig.vi

6.2.2 Subrutina: *SetUp.vi*

La subrutina *SetUp.vi* permite al usuario seleccionar los 2 puertos físicos USB predeterminados correspondientes a la tarjeta de adquisición de imágenes AV Grabber y el microcontrolador Arduino UNO, para ser almacenados en el archivo de configuración al inicializar el software de control por primera vez o si éstos fueron eliminados o movidos de su ubicación dentro de la carpeta raíz del software de control. El algoritmo de programación es el siguiente:

1. Se inicializan los elementos del panel frontal.
2. Se introducen los puertos físicos USB a utiliza por defecto, a través de la interfaz de usuario emergente.
3. Se verifica que la información ingresada sea válida.
4. En caso de ser información válida la subrutina finaliza. Caso contrario, el algoritmo se repite desde el punto **2**.

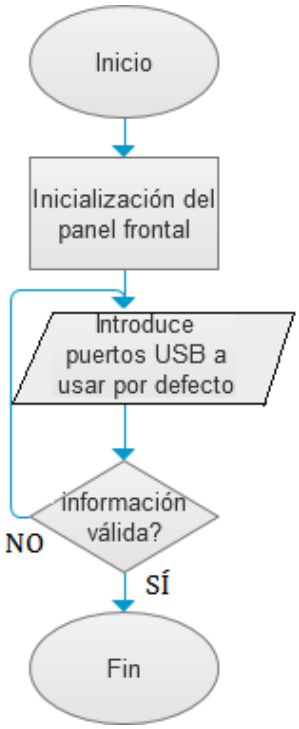


Figura 6.7 Diagrama de flujo de la subrutina SetUp.vi

La subrutina *SetUp.vi* no posee parámetros de entrada y cuenta con 2 parámetros salida. Estos son:

Parámetros de Salida

Nombre	Tipo de dato	Descripción
cam	typedef IMAQdx.ctl	Referencia al puerto USB de la tarjeta de adquisición de imágenes
arduino	typedef IMAQdx.ctl	Referencia al puerto USB del microcontrolador Arduino UNO

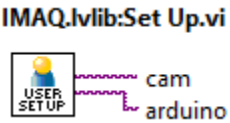


Figura 6.8 Parámetros E/S de la subrutina SetUp.vi

6.2.3 Subrutina: *Start.vi*

La subrutina *Start.vi* configura la comunicación a nivel hardware entre la tarjeta de adquisición de imágenes AV Grabber y el software de control, conectada a la PC por medio de un puerto USB, creando una instancia que hace referencia a esta comunicación. La subrutina utiliza la instancia de comunicación para configurar e inicializar una sesión de adquisición de imágenes de tipo *continua* (secuencia de imágenes o video). Finalmente asigna a una instancia el espacio en memoria suficiente para el almacenamiento temporal de las imágenes adquiridas. Si resulta imposible establecer comunicación con la tarjeta de adquisición de imágenes, será posible continuar pero el modo **Adquisición** no estará disponible desde la interfaz de usuario del software de control. El algoritmo de programación es el siguiente:

1. Configuración de la comunicación entre la tarjeta de adquisición de imágenes y el software de control.
2. Configuración de la sesión de adquisición tipo *continua*.
3. Inicialización de la sesión de adquisición tipo *continua*. Si la sesión de adquisición falla, se detiene la sesión y se elimina la instancia de comunicación, el algoritmo se repite desde el puto **1** y se activa un timeout para evitar loop infinito.
4. Si la comunicación y sesión de adquisición se estableció con éxito, se crea una instancia de las imágenes adquiridas. Caso contrario y el timeout finaliza, se deshabilita el modo **Adquisición** de la interfaz de usuario del software de control.
5. La subrutina finaliza.

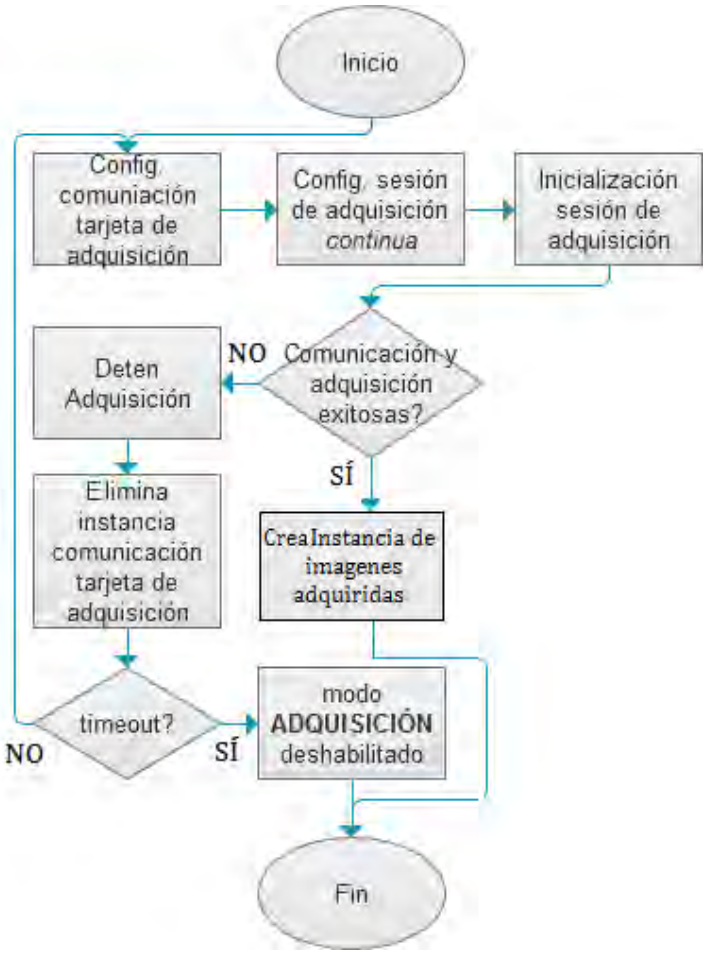


Figura 6.9 Diagrama de flujo de la subrutina Start.vi

La subrutina *Start.vi* cuenta con 2 parámetros de entrada y 3 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
cam	typedef IMAQdx.ctl	Referencia al puerto USB de la tarjeta de adquisición de imágenes
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Session Out	typedef IMAQdx.ctl	Referencia a la comunicación y sesión de adquisición de la tarjeta de adquisición de imágenes conectada a la PC por USB.
image	typedef IMAQ Image.ctl	Referencia a la imagen actual adquirida
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

Figura 6.10 Parámetros E/S de la subrutina Start.vi

6.2.4 Subrutina: *Init.vi*

La subrutina *Init.vi* verifica que se haya establecido una comunicación y una sesión de adquisición exitosa con la tarjeta de adquisición de imágenes para inicializar la comunicación serial con el microcontrolador Arduino UNO, si esta no se estableció correctamente, la subrutina finaliza inmediatamente. La subrutina configura la comunicación a nivel hardware entre el Arduino UNO y el software de control creando una instancia que hace referencia a esta comunicación, finalmente configura los puertos digitales 5 – 12 del microcontrolador Arduino como salidas (Digital Output). Si resulta imposible establecer comunicación con el microcontrolador Arduino UNO, será posible continuar pero no se podrán iniciar experimentos en el modo **Adquisición** desde la interfaz de usuario del software de control. El algoritmo de programación es el siguiente:

1. Verifica comunicación y sesión de adquisición exitosas, pasa al punto **2**. Caso contrario pasa al punto **5**.
2. Configuración de la comunicación serial con el microcontrolador Arduino UNO:

Baud Rate	115200
Board Type	UNO
Bytes per Packet	15
Connection Type	Serial/USB

3. Si la comunicación se estableció con éxito, se configuran los puertos 5 – 12 del microcontrolador Arduino UNO como salidas digitales y se pasa al punto **5**. Caso contrario, se elimina la instancia de comunicación serial con el microcontrolador Arduino UNO y se pasa al punto **4**.
4. Reintentar comunicación serial con el microcontrolador Arduino UNO pasar al punto **2**. Caso contrario, se deshabilita el uso de experimentos en el modo **Adquisición** desde la interfaz de usuario del software de control.
5. La subrutina finaliza.

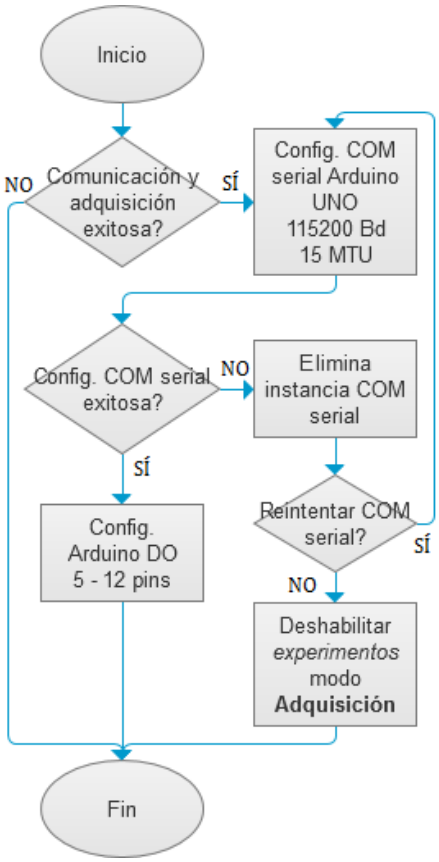


Figura 6.11 Diagrama de flujo de la subrutina Init.vi

La subrutina *Init.vi* cuenta con 2 parámetros de entrada y 4 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Arduino COM port	VISA session	Referencia al puerto USB del microcontrolador Arduino UNO
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Arduino Session Out	typedef ArduinoResource.ctl	Referencia a la COM serial del microcontrolador Arduino UNO conectado a la PC por USB
experiment enable	int	Indica si el uso de *experimentos* estará disponible
mode	1D Array [string]	Indica si el modo **Adquisición** estará disponible
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

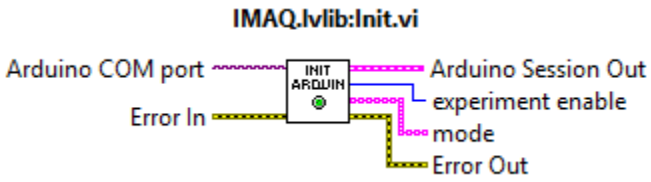


Figura 6.12 Parámetros E/S de la subrutina Init.vi

6.2.5 Subrutina: *Preview.vi*

La subrutina *Preview.vi* se encarga de recopilar todos los datos e información relevante de la sesión de experimentación que el usuario configuró y está a punto de ejecutar, mostrándole:

- El puerto físico USB que se está utilizando para la comunicación y adquisición con la tarjeta de adquisición de imágenes AV Grabber,
- El puerto físico USB que se está utilizando para la comunicación serial del microcontrolador Arduino UNO,
- El formato de guardado de las imágenes,
- La ruta de trabajo raíz,
- Si se utilizarán ambos obturadores o solo uno de ellos,
- El obturador a utilizar en caso de utilizar uno solo, o el obturador que comenzará a operar en caso de utilizar ambos,
- El tiempo de exposición te [ms], periodo de apertura Ta [ms] y número total de imágenes a adquirir respecto al obturador a utilizar en caso de utilizar uno solo, o respecto al obturador 1 y de la misma manera con respecto al obturador 2 en caso de utilizar ambos.

También computa el tiempo estimado de la sesión de experimentación según los tiempos de exposición, periodo de apertura y número total de imágenes a adquirir. Finalmente permite elegir o modificar las rutas de almacenamiento para las imágenes a adquirir así como los nombres genéricos de éstas últimas, verificando que ni las rutas ni los nombres asignados entren en conflicto con archivos importantes del sistema, sesiones de experimentación anteriores, o rutas conflictivas con el sistema operativo de la PC (nombres de rutas y archivos con caracteres especiales). El algoritmo de programación tiene una estructura similar a la presentada en la **Sección 6.1.1**, la cual es la siguiente:

1. Se inicializan los elementos del panel frontal.
2. Se inicia un ciclo *while* recurrente que contiene la *estructura principal de eventos* con 4 comandos y un Timeout de 1[ms].
3. Los 4 comandos programados son llamados a través de los controles y botones del panel frontal de la interfaz de usuario emergente, y están asociados a las siguientes funciones:

   [1] – Activa el uso de la misma ruta de almacenamiento para la adquisición de las imágenes asociadas tanto al obturador 1 como al obturador 2.
   [2] – Se verifica que las rutas de trabajo no causen conflicto en el sistema, con sesiones experimentales previas y/o entre ellos y comienza la sesión de experimentación con los parámetros configurados previamente. También cancela la sesión de experimentación por acción del usuario.
   [3] – Verifica que los nombres de las rutas de almacenamiento no causen conflicto en el sistema, entre ellas o entre sesiones experimentales previas.
   [4] – Verifica que los nombres de las imágenes no causen conflicto en el sistema, entre ellos o entre sesiones experimentales previas.

4. El Timeout es un evento especial dentro de la estructura de eventos el cual se ejecuta automáticamente cada 1 [ms] sin que el usuario tenga que presionar algún botón o control desde el panel frontal, cuya función es mantener al software en modo de espera. Éste se encarga de dar formato a la información concerniente a la configuración de la sesión experimental para presentarla en pantalla al usuario a manera de resumen previo a la ejecución de la sesión experimental.
5. El comando [**2]** inicia la verificación de las rutas de trabajo, si no existe conflicto la subrutina regresa una variable *bool* TRUE y se procede al punto **7**. Caso contrario se manda una señal de alerta al usuario sobre posibles conflictos.
6. Si el usuario decide continuar la subrutina regresa una variable *bool* TRUE, caso contrario regresa una variable *bool* FALSE.
7. La subrutina finaliza.

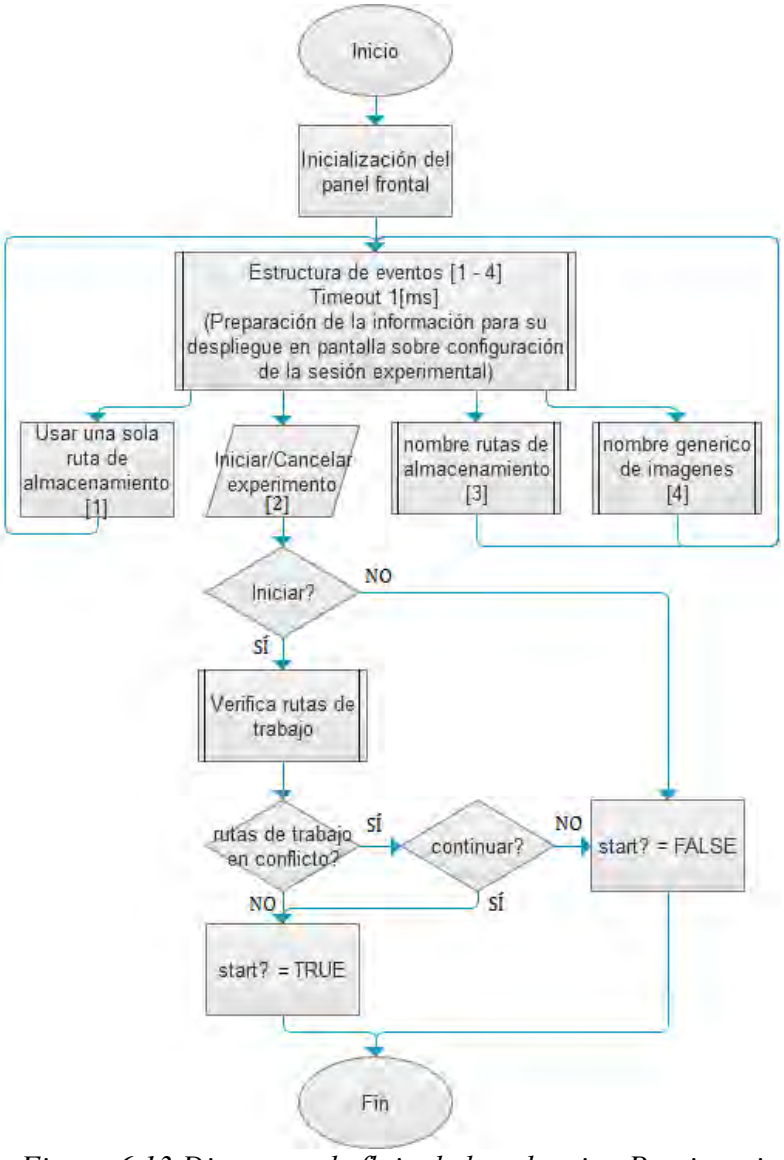


Figura 6.13 Diagrama de flujo de la subrutina Preview.vi

La subrutina *Preview.vi* cuenta con 1 parámetro de entrada y 5 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Pcluster	Cluster [16 elementos]	Contiene la configuración e información referente a la sesión de experimentación: puertos USB, formato de imagen, ruta raíz, uso de obturadores, orden de operación, te, Ta y total de imágenes.

Parámetros de Salida

Nombre	Tipo de dato	Descripción
name1	string	Nombre genérico de las imágenes adquiridas por el obturador 1
Shutterdir1	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 1
Shutterdir2	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 2
start?	bool	Indica la ejecución de la sesión experimental
name2	string	Nombre genérico de las imágenes adquiridas por el obturador 2

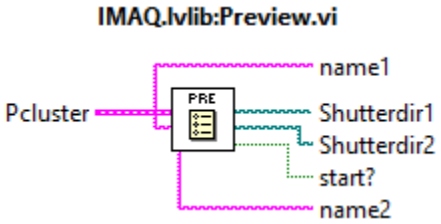


Figura 6.14 Parámetros E/S de la subrutina Preview.vi

6.2.6 Subrutina: *TriggerPreview.vi*

La subrutina *TriggerPreview.vi* se encargar de dar formato a dos variables internas del software de control que forman parte de la plantilla de configuración del experimento: *triggernums* (1D Array *int*) y *triggers (*1D Array *bool)*, a modo de ser desplegadas como información comprensible para el usuario en el resumen previo a iniciar el experimento. La subrutina toma estos dos arreglos y revisa su contenido para indicarle al usuario si alguno de los 6 Triggers está activado y en qué imagen exactamente se activará y a cual obturador está asociada la imagen en caso de uso simultaneo de éstos. El algoritmo de programación es el siguiente:

1. Se inicializa un ciclo *for (i=0; i==5; i++)*
2. Se verifica:
   *triggers [i] == TRUE && (triggernums [i*2] =! 0 || triggernums [i*2 + 1] =! 0)*
   Caso afirmativo se verifica uso múltiple o individual de obturadores. Caso contrario subrutina regresa:
   'TRIGGER i DESACTIVADO'
   Y se procede al punto **4**.
3. Si el uso es individual subrutina regresa:
   *'TRIGGER i ACTIVADO en' + 'imagen triggernums [i*2]'*
   Si el uso es múltiple subrutina regresa:

*'TRIGGER i ACTIVADO en imagen' + (if (triggernums [i*2] =! 0) 'triggernums [i*2]' else '-') + 'y' + (if (triggernums [i*2 + 1] =! 0) 'triggernums [i*2 + 1]' else '-')])*

4. Si *i==5* la subrutina finaliza. Caso contrario *i++* y se procede al punto **2**.

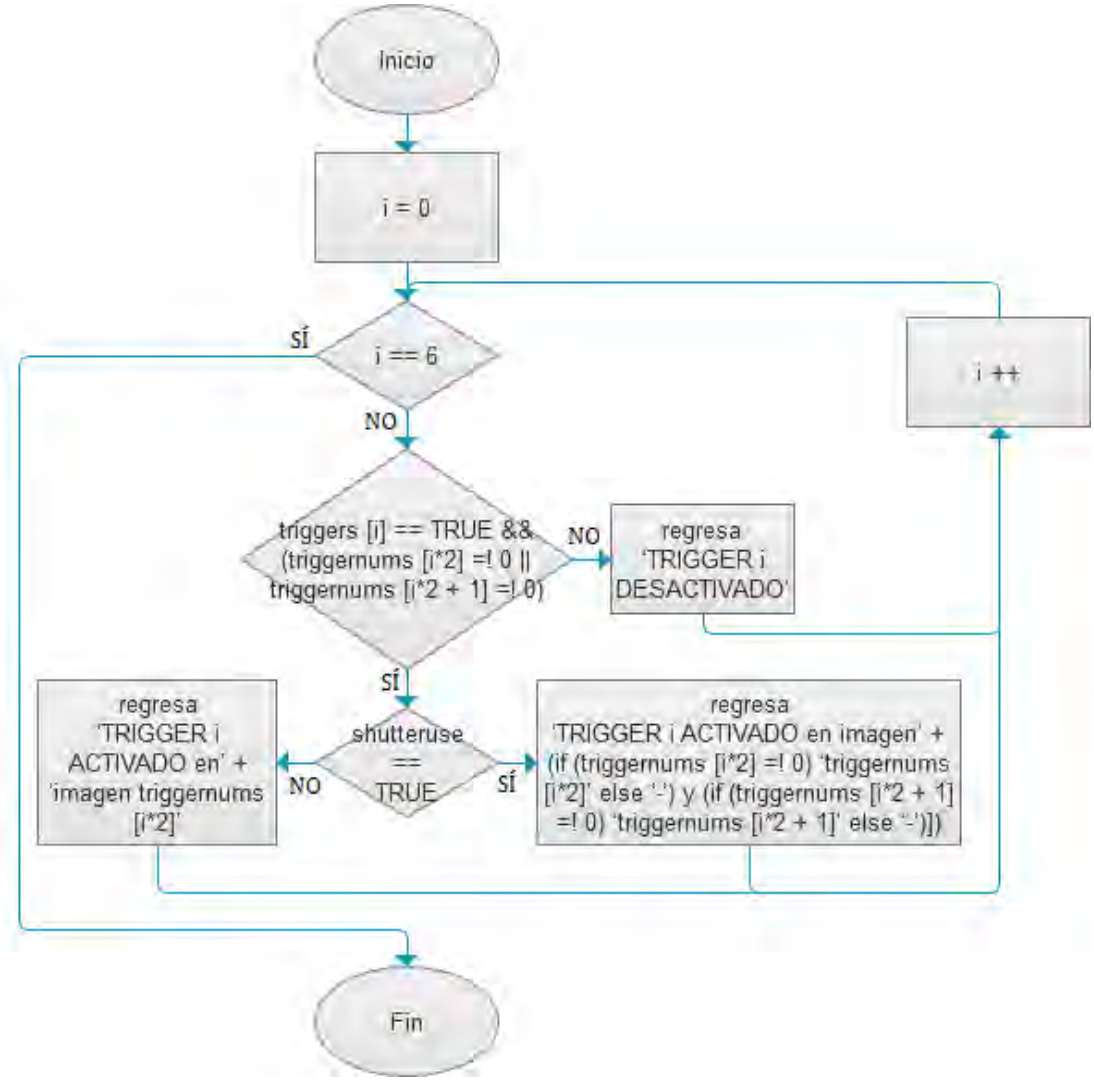


Figura 6.15 Diagrama de flujo de la subrutina TriggerPreview.vi

La subrutina *TriggerPreview.vi* cuenta con 3 parámetros de entrada y 1 parámetro de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
triggernums	1D Array [int]	Indica en qué imagen se activará el Trigger y a qué obturador está asociado
triggers	1D Array [bool]	Indica que Triggers se activarán durante el experimento
shutteruse	bool	Indica si se usaran ambos obturadores durante el experimento

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Triggers:	string	Indica cuales de los Triggers se encuentran activos y en qué imagen se activarán

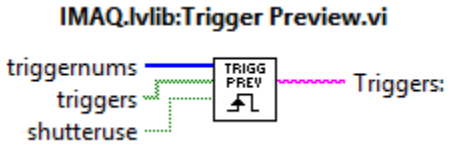


Figura 6.16 Parámetros E/S de la subrutina TriggerPreview.vi

6.2.7 Subrutina: *TotalTime.vi*

La subrutina *TotalTime.vi* se encarga de calcular el tiempo total estimado que durará una sesión experimental en segundos [s], basado únicamente en el periodo de apertura (Ta_1, Ta_2) [ms] y el número total de imágenes a adquirir (tin_1, tin_2) [1] de ambos obturadores. La subrutina calcula el tiempo estimado de duración de ambos obturadores (tt_1, tt_2) [s] y luego los suma ($tt = tt_1 + tt_2$) [s], obteniendo el tiempo total por obturador y en conjunto.

Basado en el hecho de que un ciclo del periodo de apertura equivale a un ciclo o etapa de exposición del modelo biológico a la luz de excitación, en donde se adquiere una imagen de fluorescencia, basta con multiplicar el total de las imágenes a adquirir por el periodo de apertura del obturador para obtener una estimación bastante acertada del tiempo total que durará el experimento y luego ajustarla a las unidades deseadas. Sin embargo cuando se utilizan los dos obturadores de manera simultánea esta estimación puede no ser muy acertada, debido a que la adquisición de ambos obturadores no es alternada u ordenada, sino que es independiente la una de la otra, a pesar de esto el tiempo total calculado por la subrutina resulta una estimación aceptable. Para mayor información sobre los tiempos de operación de los obturadores, las señales de control y la adquisición de imágenes, véase **Anexo C: Pruebas de Tiempo** en la sección **Anexos**. El algoritmo de programación es el siguiente:

1. $tt_1 = (Ta_1 * tin_1) / 1000$
2. $tt_2 = (Ta_2 * tin_2) / 1000$
3. $tt = tt_1 + tt_2$
4. La subrutina finaliza.

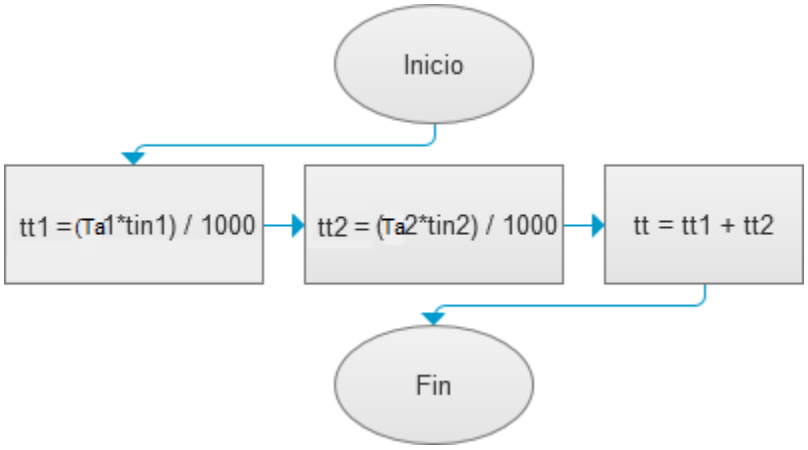


Figura 6.17 Diagrama de flujo de la subrutina TotalTime.vi

La subrutina *TotalTime.vi* cuenta con 4 parámetros de entrada y 3 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
T3	int	Periodo de apertura del obturador 1
tin	int	Total de imágenes a adquirir asociadas al obturador 1
T3 2	int	Periodo de apertura del obturador 2
tin 2	int	Total de imágenes a adquirir asociadas al obturador 2

Parámetros de Salida

Nombre	Tipo de dato	Descripción
tt1	string	Tiempo aproximado que durará la adquisición de las imágenes asociadas al obturador 1
tt2	string	Tiempo aproximado que durará la adquisición de las imágenes asociadas al obturador 2
tt	string	Tiempo aproximado que durará el experimento

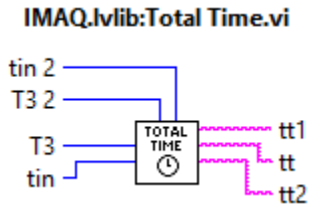


Figura 6.18 Parámetros E/S de la subrutina TotalTime.vi

6.2.8 Subrutina: *NameChanged.vi*

La subrutina *NameChanged.vi* verifica que el nombre seleccionado, ya sea para las imágenes adquiridas, las rutas de trabajo o las rutas de almacenamiento, sean válidas y no causen conflicto con el sistema operativo de la PC, verificando que no contengan caracteres especiales. El algoritmo de programación es el siguiente:

1. Se crea un archivo temporal en la carpeta raíz del software de control con el nombre establecido por el usuario.
2. Si el archivo es creado sin causar conflictos con el sistema operativo de la PC, se elimina y la subrutina regresa el nombre ingresado por el usuario. Caso contrario se notifica al usuario sobre nombre inválido y la subrutina regresa nombre por defecto.
3. La subrutina finaliza.

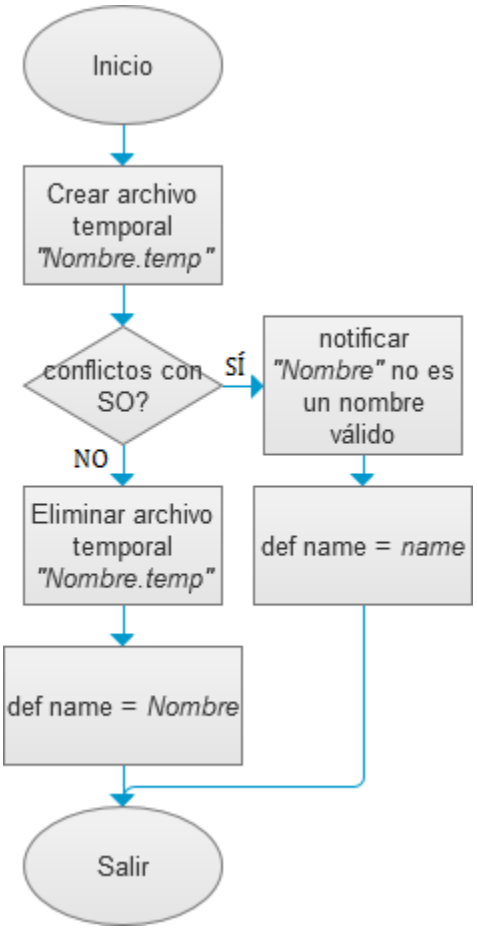


Figura 6.19 Diagrama de flujo de la subrutina NameChanged.vi

La subrutina *NameChanged.vi* cuenta con 1 parámetro de entrada y 1 parámetro de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Nombre	string	Nombre ingresado por el usuario

Parámetros de Salida

Nombre	Tipo de dato	Descripción
def name	string	Nombre validado por la subrutina

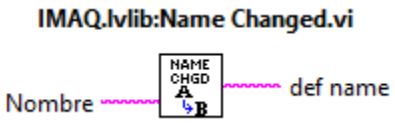


Figura 6.20 Parámetros E/S de la subrutina NameChanged.vi

6.2.9 Subrutina: *ShutterPath.vi*

La subrutina *ShutterPath.vi* verifica que las rutas de almacenamiento elegidas para las secuencias de imágenes que serán adquiridas durante el experimento, no causen conflicto en el sistema, entre ellas mismas o entre sesiones experimentales previas. El algoritmo de programación es el siguiente:

1. Verifica si se utilizarán ambos obturadores en el experimento.
2. Para el uso de obturadores individuales: verifica la existencia de carpetas de almacenamiento con el mismo nombre en la ruta de trabajo raíz. Caso afirmativo le pregunta al usuario si desea continuar y se procede al punto **4**. Caso contrario se procede al punto **5**.
3. Para uso de obturadores múltiple: verifica si se utilizará una sola carpeta para el almacenamiento de imágenes y verifica la existencia de carpetas con el mismo nombre en la ruta de trabajo raíz, si no es así, verifica que ambas carpetas no tengan el mismo nombre ni existan carpetas con el mismo nombre en la ruta de trabajo raíz. En todas las opciones si el caso es afirmativo se pregunta al usuario si desea continuar y se procede al punto **4**. Caso contrario en todas las opciones se procede al punto **5**.
4. Si el usuario no desea continuar la subrutina regresa una variable bool FALSE y finaliza. Caso contrario se procede al punto **5**.
5. La subrutina regresa una variable bool TRUE y finaliza.

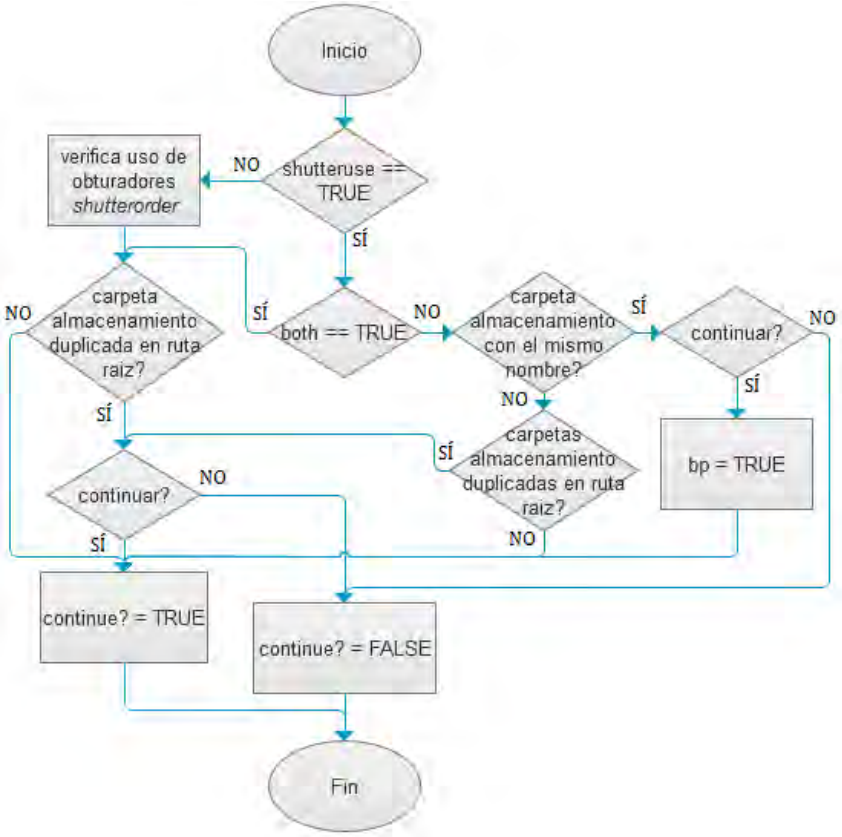


Figura 6.21 Diagrama de flujo de la subrutina ShutterPath.vi

La subrutina *ShutterPath.vi* cuenta con 7 parámetros de entrada y 2 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
shutterorder	int	Indica el obturador a usar y el orden de operación
Error Shutter 1	Cluster [3 elementos]	Indica si existe una carpeta duplicada para las imágenes asociadas al obturador 1
shutter path 1	string	Nombre de la carpeta de almacenamiento para las imágenes asociadas al obturador 1
shutter path 2	string	Nombre de la carpeta de almacenamiento para las imágenes asociadas al obturador 2
Error Shutter 2	Cluster [3 elementos]	Indica si existe una carpeta duplicada para las imágenes asociadas al obturador 2
both	bool	Indica si se usará la misma carpeta para almacenar ambas secuencias de imágenes
shutteruse	bool	Indica si se usarán ambos obturadores en el experimento

Parámetros de Salida

Nombre	Tipo de dato	Descripción
continue?	bool	Continuar utilizando los nombres elegidos
bp	bool	Indica si los nombres de las carpetas de almacenamiento asociadas con los obturadores son iguales

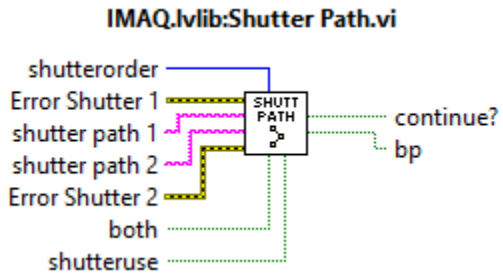


Figura 6.22 Parámetros E/S de la subrutina ShutterPath.vi

6.2.10 Subrutina: *MainRoutine.vi*

La subrutina *MainRoutine.vi* debe su nombre a que representa el núcleo del software, al encargarse de realizar las principales tareas de control y adquisición durante la ejecución de los experimentos de microscopía de fluorescencia, más no se debe confundir con el programa o módulo principal del software de control *HMI.vi*. La subrutina toma como entradas las opciones que el usuario estableció en la plantilla de configuración del experimento y, previamente verificado que no existan conflictos con estas opciones y configuraciones, comienza la ejecución del experimento.

Cabe mencionar, que la subrutina utiliza una estructura de programación por hilos independientes, es decir que la ejecución de cada uno de estos hilos (que representan funciones, tareas o comandos), se realiza en segundo plano a la par de otros hilos, procesos o funciones que se estén ejecutado en ese momento, aumentando las funciones, alcance y productividad del programa pero también aumentando el consumo de recursos de la PC, en

especial los recursos de procesamiento y memoria volátil (RAM). En LabVIEW, estos hilos de programación independientes se representan con loops *while* sin ninguna conexión de flujo de datos uniéndolos o relacionándolos, ya que esto impediría su ejecución simultánea en el diagrama de bloques. LabVIEW asigna recursos de procesamiento y memoria volátil a cada uno de los loops *while* de forma arbitraria, dándole un tanto de prioridad a aquellos con funciones complejas. La ejecución de estos hilos también es arbitraria, es decir, que LabVIEW los ejecutará sin ningún orden específico.

Según la configuración establecida por el usuario, la subrutina opera bajo dos modalidades:

- **Operación con un obturador (obturador1/obturador2)**
  La subrutina ejecuta cuatro hilos de programación independientes o paralelos entre sí, estos hilos son:
  - **Cronometraje de la sesión experimental.** Este hilo de programación corre un contador para medir el tiempo del experimento transcurrido desde su inicio hasta su fin, mostrándolo al usuario en la interfaz de usuario emergente.

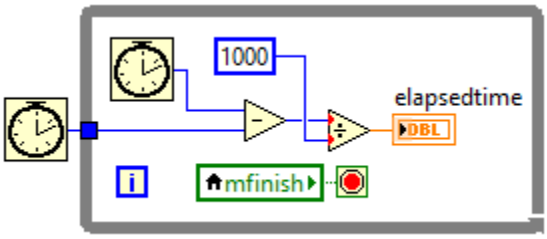


Figura 6.23 Cronometraje de la sesión experimental.

  - **Adquisición y despliegue en tiempo real de imágenes de microscopía.** Este hilo de programación se encarga de mantener la comunicación activa con la tarjeta de adquisición de imágenes AV Grabber, proporcionando un flujo o adquisición de imágenes continua que serán desplegadas en tiempo real sobre la interfaz de usuario emergente y posteriormente almacenadas en la PC.

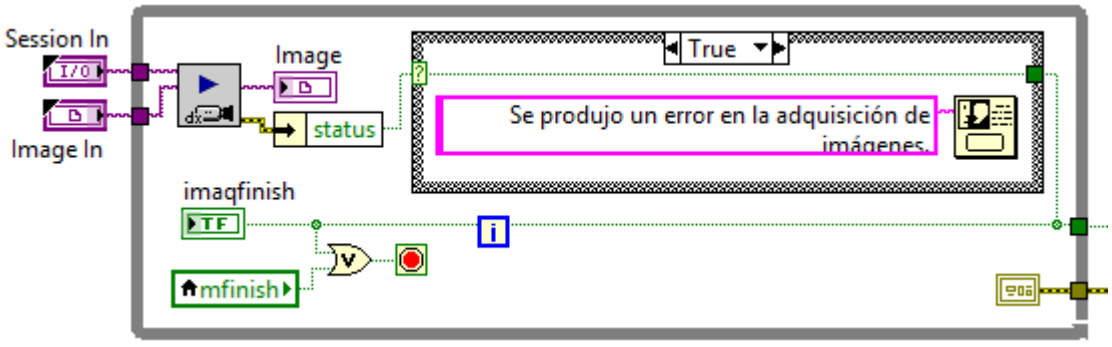


Figura 6.24 Adquisición y despliegue en tiempo real de imágenes de microscopía.

- **Condicional *Abortar Sesión*.** La subrutina mantiene un comando independiente a cualquier función, tarea o proceso que le permite al usuario cancelar el experimento en cualquier momento de este, no importando el proceso que se esté ejecutando.

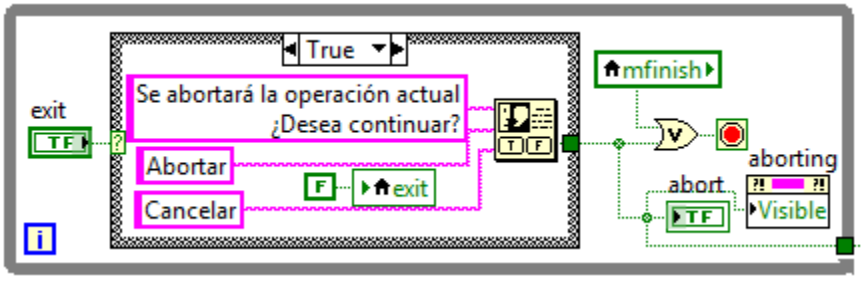


Figura 6.25 Condicional Abortar Sesión.

- **Control del obturador y almacenamiento en memoria de imágenes.** Este procedimiento representa el corazón de la subrutina y del experimento, ya que es el encargado de controlar las señales digitales de salida del microcontrolador Arduino UNO para la apertura y cierre de los obturadores, sincronizando así la exposición del modelo biológico a la luz de excitación con el almacenamiento de las imágenes de fluorescencia adquiridas por la tarjeta AV Grabber en el momento de la fluorescencia del mismo modelo biológico. Debido a la extensión del hilo de programación se omiten las imágenes que ilustren el procedimiento, véase **Anexo B: FIMAQ Código Fuente** para visualizar el código de programación desde el diagrama de bloques de la subrutina. El algoritmo de programación es el siguiente:

1. Se inicializan los elementos del panel frontal correspondientes a la interfaz de usuario emergente.
2. Se inicializa y depura el método de compresión y almacenamiento de imágenes.
3. Se inicia el hilo cronometraje de la sesión experimental.
4. Se inicia el hilo adquisición y despliegue en tiempo real de imágenes de microscopía.
5. Se inicia el hilo condicional *abortar sesión*.
6. Se inicia el hilo control del obturador y almacenamiento en memoria de imágenes con la configuración establecida por el usuario (obturador a utilizar, uso de Triggers, *te* [ms], *Ta* [ms], número total de imágenes a adquirir).
7. Se verifica uso de Triggers para la imagen actual. Caso afirmativo se activa junto con la apertura del obturador. Caso contrario se activa únicamente la señal de control para la apertura del obturador.
8. Subrutina espera el tiempo establecido en *te* [ms].
9. Se desactivan las señales de control.
10. Compresión y almacenamiento en memoria de la imagen actual adquirida.
11. Subrutina espera el tiempo establecido por *Ta* [ms].
12. Verifica errores en la adquisición de imágenes, compresión y almacenamiento en memoria de imágenes, comunicación con los instrumentos y comando *abortar*

sesión. Caso afirmativo en cualquier de los eventos procede al punto **14**. Caso contrario procede al punto **13**.

13. Verifica *número total de imágenes a adquirir == número de imágenes adquiridas actualmente*. Caso afirmativo procede al punto **14**. Caso contrario procede al punto **7**.

14. Finaliza el hilo control del obturador y almacenamiento en memoria de imágenes.

15. Finaliza el hilo cronometraje de la sesión experimental.

16. Finaliza el hilo adquisición y despliegue en tiempo real de imágenes de microscopía.

17. Finaliza el hilo condicional *abortar sesión*.

18. Inicia subrutina *ReportGenerator.vi* (**Sección 6.2.12**) para la generación de archivo de registro del experimento y generación de reporte escrito del experimento.

19. La subrutina finaliza.

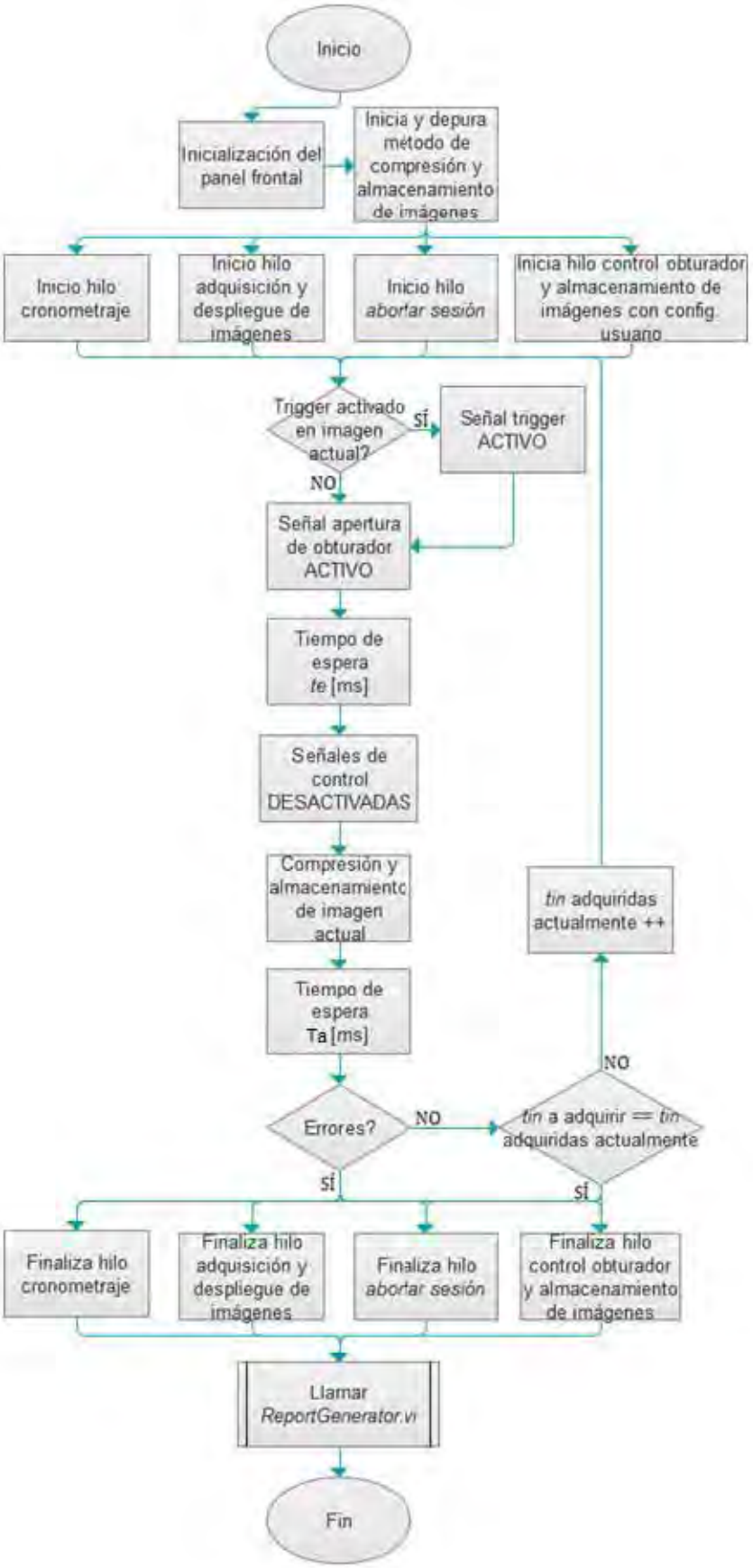


Figura 6.26 Diagrama de flujo de la subrutina MainRoutine.vi (modo de operación con un solo obturador).

- **Operación con ambos obturadores (obturador1 y obturador2)**
  La subrutina ejecuta 6 hilos de programación independientes o paralelos entre sí, estos hilos son:
  - **Cronometraje de la sesión experimental.** Ya descrito en el modo de operación con un solo obturador.
  - **Adquisición y despliegue en tiempo real de imágenes de microscopía.** Ya descrito en el modo de operación con un solo obturador.
  - **Condicional *Abortar Sesión*.** Ya descrito en el modo de operación con un solo obturador.
  - **Periodo de apertura *Ta* [ms] del obturador 1** y **Periodo de apertura *Ta* [ms] del obturador 2.** Ya que la adquisición de imágenes no obedece a un patrón ordenado o en serie cuando se operan ambos obturadores en el experimento, el periodo de apertura de cada obturador se debe cronometrar de forma independiente, esto debido a especificaciones muy particulares de investigación, por lo tanto es necesario contar con dos hilos más de programación independiente a cargo del conteo del periodo de apertura, uno para cada obturador.

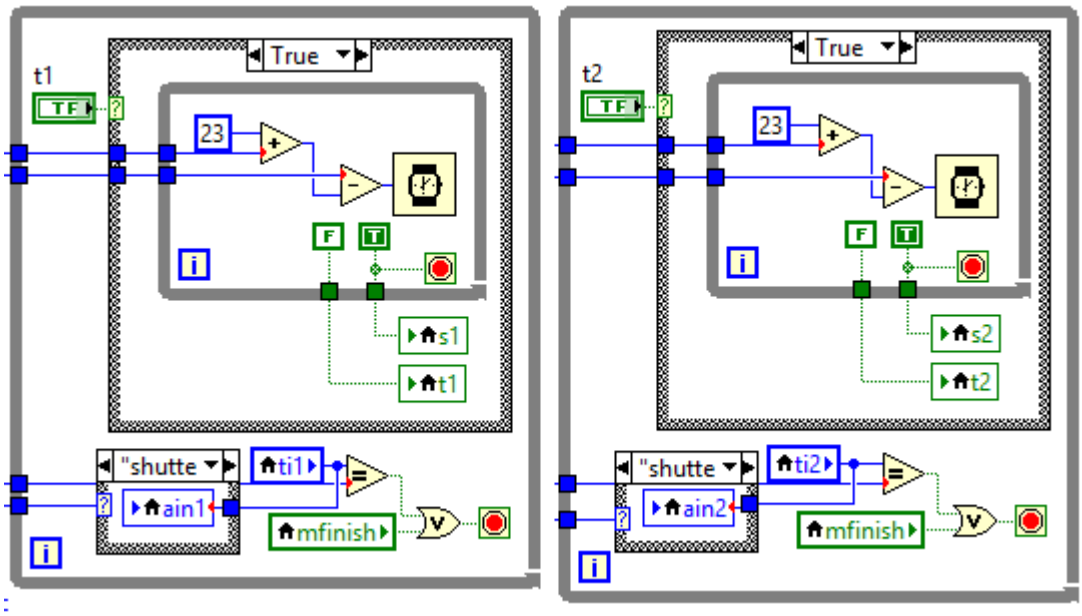


Figura 4.27 Conteo del periodo de apertura Ta [ms], uno para cada obturador.

  - **Control del obturador y almacenamiento en memoria de imágenes.** La función es muy similar a aquella descrita en el modo de operación con un solo obturador, el cambio más significativo radica en la inclusión de un conteo del periodo de apertura independiente y separado del hilo de control y almacenamiento en memoria de imágenes, que obliga al hilo de control y almacenamiento a verificar el contador de *Ta* [ms] antes de comenzar las operaciones de control y almacenamiento para cada uno de los obturadores. Debido a la extensión del hilo de programación se omiten las imágenes que ilustren el procedimiento, véase **Anexo B: FIMAQ Código Fuente** para visualizar el código de programación

desde el diagrama de bloques de la subrutina. El algoritmo de programación es el siguiente:

1. Se inicializan los elementos del panel frontal correspondientes a la interfaz de usuario emergente.
2. Se inicializa y depura el método de compresión y almacenamiento de imágenes.
3. Se inicia el hilo cronometraje de la sesión experimental.
4. Se inicia el hilo adquisición y despliegue en tiempo real de imágenes de microscopía.
5. Se inicia el hilo condicional *abortar sesión*.
6. Se inicia el hilo control del obturador y almacenamiento en memoria de imágenes con la configuración establecida por el usuario (primer obturador a operar, uso de Triggers en cada obturador, *te* [ms], *Ta* [ms] y número total de imágenes a adquirir para cada obturador).
7. Se inicia el hilo contador *Ta* [ms] para el obturador 1 (se inicializa como *contador terminó*).
8. Se inicia el hilo contador *Ta* [ms] para el obturador 2 (se inicializa como *contador terminó*).
9. Se verifica *contador terminó* para obturador 1 desde el hilo contador *Ta* [ms]. Caso afirmativo proceder al punto **11**. Caso contrario proceder al punto **10**.
10. Se verifica *contador terminó* para obturador 2 desde el hilo contador *Ta* [ms]. Caso afirmativo proceder al punto **11**. Caso contrario proceder al punto **9**.
11. Se verifica uso de Triggers para la imagen y obturador actual. Caso afirmativo se activa junto con la apertura del obturador en turno. Caso contrario se activa únicamente la señal de control para la apertura del obturador en turno.
12. Subrutina espera el tiempo establecido en *te* [ms] para obturador actual.
13. Se desactivan las señales de control.
14. Compresión y almacenamiento en memoria de la imagen actual adquirida asociada al obturador actual.
15. Se reinicia el contador *Ta* [ms] para el obturador en turno.
16. Verifica errores en la adquisición de imágenes, compresión y almacenamiento en memoria de imágenes, comunicación con los instrumentos y comando *abortar sesión*. Caso afirmativo en cualquier de los eventos finaliza hilos contador *Ta* [ms] para ambos obturadores y procede al punto **19**. Caso contrario procede al punto **17**.
17. Verifica *número total de imágenes a adquirir == número de imágenes adquiridas actualmente* para el obturador en turno. Caso afirmativo finaliza hilo contador *Ta* [ms] para el obturador en tuno (representado como un contador que nunca termina) y procede al punto **18**. Caso contrario procede al punto **9**.
18. Verifica hilo contador *Ta* [ms] finalizado para ambos obturadores. Caso afirmativo procede al punto **19**. Caso contrario procede al punto **9**.
19. Finaliza el hilo control del obturador y almacenamiento en memoria de imágenes.
20. Finaliza el hilo cronometraje de la sesión experimental.
21. Finaliza el hilo adquisición y despliegue en tiempo real de imágenes de microscopía.
22. Finaliza el hilo condicional *abortar sesión*.

23. Inicia subrutina *ReportGenerator.vi* (**Sección 6.2.12**) para la generación de archivo de registro del experimento y generación de reporte escrito del experimento.

24. La subrutina finaliza.

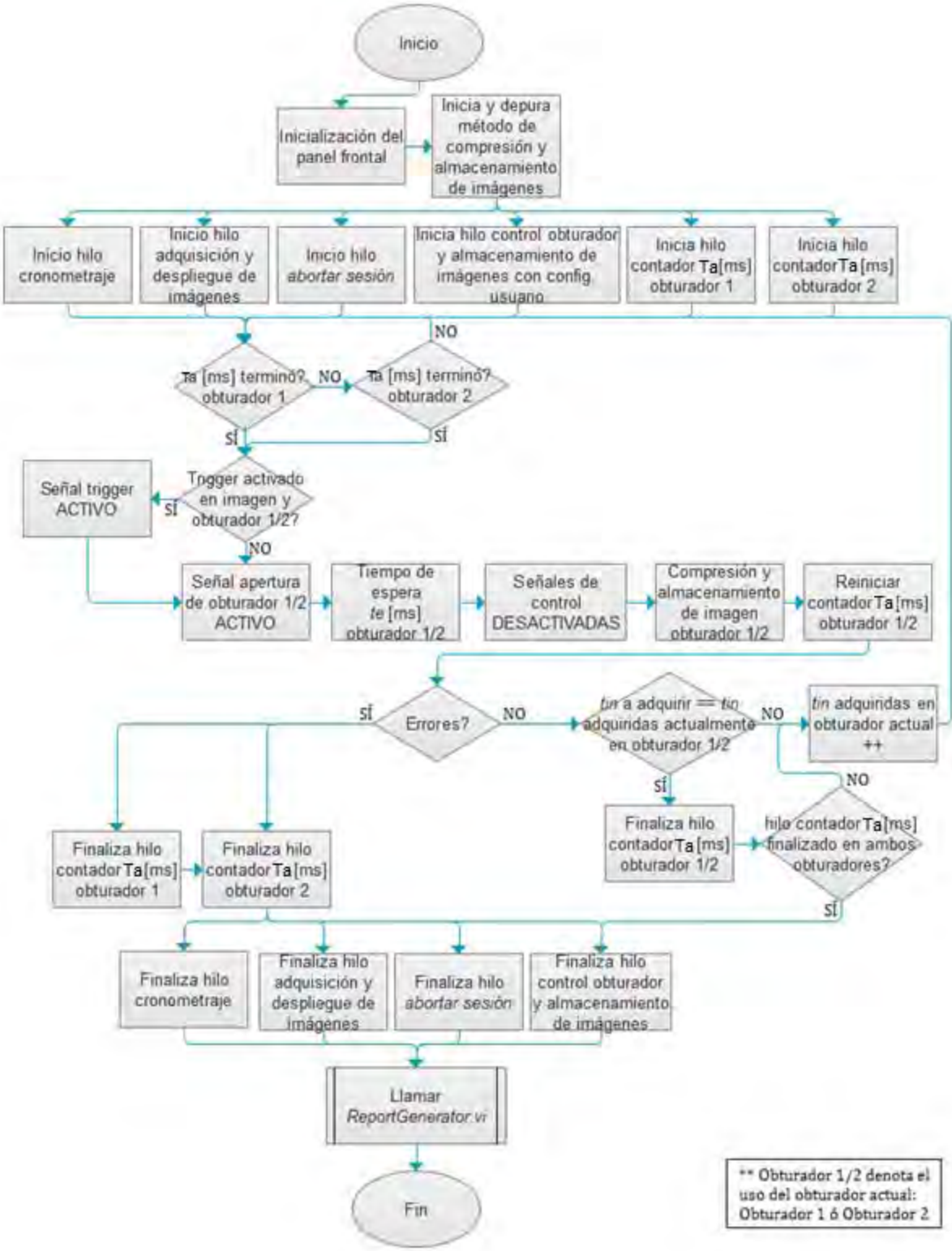


Figura 6.28 Diagrama de flujo de la subrutina MainRoutine.vi (modo de operación con ambos obturador).

La subrutina *MainRoutine.vi* cuenta con 8 parámetros de entrada y no posee parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
name1	string	Nombre genérico de las imágenes adquiridas por el obturador 1
name2	string	Nombre genérico de las imágenes adquiridas por el obturador 2
Arduino Resource	typedef ArduinoResource.ctl	Referencia a la COM serial del microcontrolador Arduino UNO conectado a la PC por USB
Image In	typedef IMAQ Image.ctl	Referencia a la imagen actual adquirida
Session In	typedef IMAQdx.ctl	Referencia a la comunicación y sesión de adquisición de la tarjeta de adquisición de imágenes conectada a la PC por USB.
Cluster	Cluster [16 elementos]	Contiene la configuración e información referente a la sesión de experimentación: puertos USB, formato de imagen, ruta raíz, uso de obturadores, orden de operación, te, Ta y total de imágenes.
shutterdir1	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 1
Shutterdir2	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 2

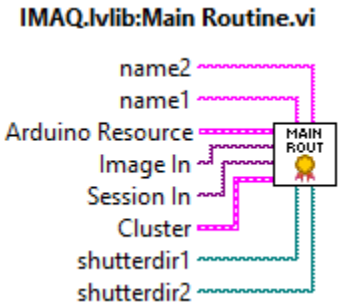


Figura 6.29 Parámetros E/S de la subrutina MainRoutine.vi

6.2.11 Subrutina: *MainTrigger.vi*

La subrutina *MainTrigger.vi* actúa durante la ejecución del experimento y comprueba si se solicitó la activación de algún Trigger por parte del usuario (a través de la plantilla de configuración de experimento en la interfaz de usuario principal) en la imagen de adquisición actual, de ser así, se dispara junto con la señal de control de apertura del obturador actual. El algoritmo de programación es el siguiente:

1. Se verifica que se haya configurado por lo menos un Trigger de los 6 disponibles como ACTIVADO:
   for (i=0; i==6; i++) if (Trigger [i] == TRUE) break, else (exit)
   De no ser así se procede al punto **3**.
2. Ciclo *for (i=0; i==6; i++)* verifica que el/los Trigger(s) ACTIVADO(S) dentro del arreglo de 6 elementos coincida con la imagen actual adquirida:
   if (Trigger [i] && triggernums [i] == imac) Control[triggerpin] = TRUE, else Control[triggerpin] = FALSE

El puerto del Trigger ACTIVADO se almacena en un arreglo bool junto con la señal de control de apertura del obturador actual. Si *i==6* se procede al punto **3**. Contrario ciclo *for* continua.

3. La subrutina envía el comando con los Triggers y la señal de control de apertura del obturador actual y finaliza.

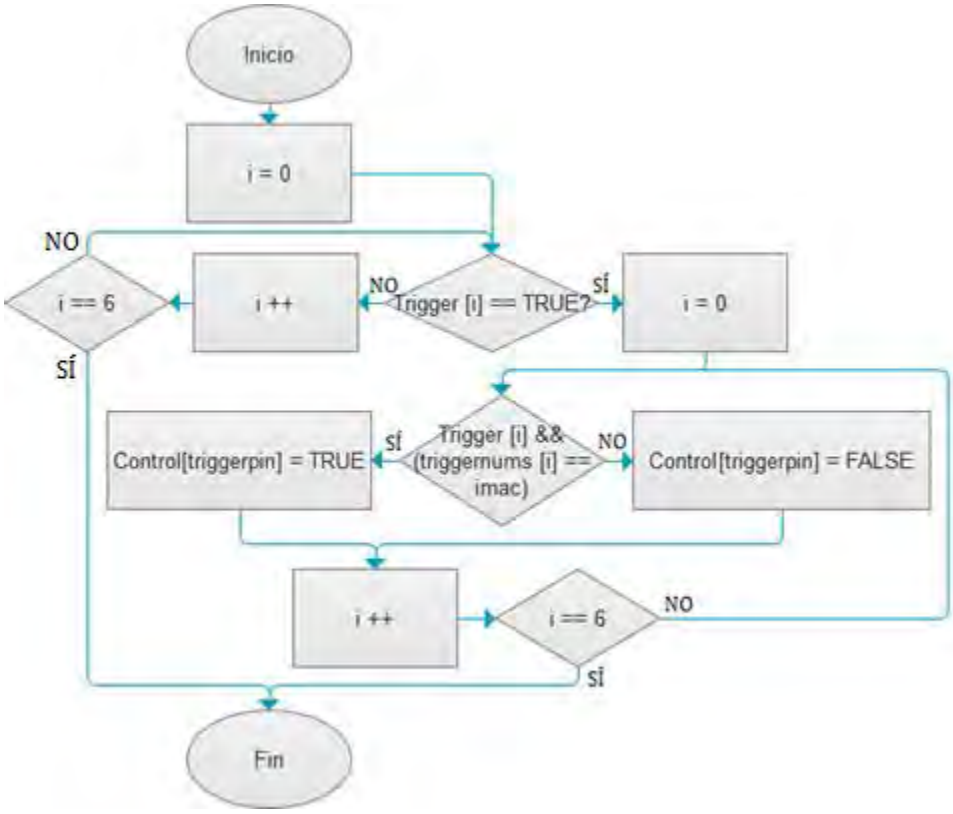


Figura 6.30 Diagrama de flujo de la subrutina MainTrigger.vi

La subrutina *MainTrigger.vi* cuenta con 6 parámetros de entrada y 2 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
pin	int	Indica el pin de salida que controla al obturador en turno
Arduino Resource In	typedef ArduinoResource.ctl	Referencia a la COM serial del microcontrolador Arduino UNO conectado a la PC por USB
triggernums	1D Array [int]	Indica en qué imagen se activará el Trigger y a qué obturador está asociado
Boolean Array	1D Array [bool]	Indica que Triggers se activarán durante el experimento
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa
imac	int	Indica la imagen actual adquirida

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Arduino Resource Out	typedef ArduinoResource.ctl	Referencia a la COM serial del microcontrolador Arduino UNO conectado a la PC por USB
error out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

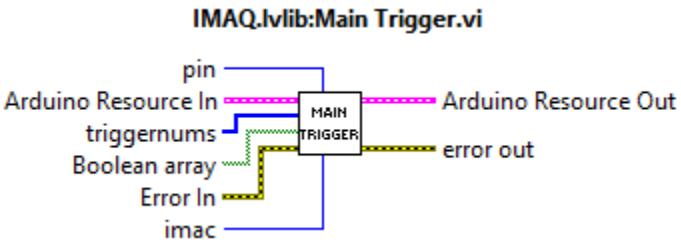


Figura 6.31 Parámetros E/S de la subrutina MainTrigger.vi

6.2.12 Subrutina: *ReportGenerator.vi*

En conjunto con otras subrutinas, *ReportGenerator.vi* se encarga de dar formato a los datos establecidos por el usuario en la plantilla de configuración del experimento desde la interfaz de usuario principal así como información recopilada durante la sesión experimental de tal manera que puedan ser comprensibles para el usuario y para el software de control a través de un archivo de registro del experimento (*RegisterGenerator.vi*, **Sección 6.2.15**) y un reporte escrito del experimento (*TXTGenerator.vi*, **Sección 6.2.16**), lo que permite acceder a esta información en sesiones posteriores desde el panel de **Visualización** de la interfaz de usuario principal. Esta información comprende:

- El puerto físico USB que se utilizó para la comunicación y adquisición con la tarjeta de adquisición de imágenes AV Grabber,
- El puerto físico USB que se utilizó para la comunicación serial del microcontrolador Arduino UNO,
- El formato de guardado de las imágenes,
- Nombre genérico de las imágenes adquiridas,
- Las secuencias de imágenes adquiridas durante el experimento asociadas al obturador 1, al obturador 2, o ambas.
- La ruta de trabajo raíz,
- Las rutas de almacenamiento de las imágenes adquiridas,
- Los triggers que se utilizaron durante el experimento, en qué imagen se activaron y a qué obturador estaban asociados,
- Si se utilizaron ambos obturadores o solo uno de ellos,
- El obturador que se utilizó en caso de haberse utilizado uno solo, o el obturador que comenzó a operar en caso de haberse utilizado ambos,
- El tiempo de exposición te [ms], periodo de apertura Ta [ms] y número total de imágenes adquiridas respecto al obturador que se utilizó en caso de haberse utilizado uno solo, o respecto al obturador 1 y de la misma manera con respecto al obturador 2 en caso de haberse utilizado ambos,
- Nombre del experimento,
- Información relevantes y conclusiones sobre el experimento,
- Fecha y hora en que se ejecutó y terminó el experimento,
- Duración total del experimento.

Finalmente se encarga de llamar a la subrutina *MainSave.vi* (**Sección 6.2.14**) encargada de almacenar en el disco duro de la PC la(s) secuencia(s) de imagen(es) adquirida(s) y

almacenada(s) en memoria durante el experimento. El algoritmo de programación es el siguiente:

1. Verifica que la sesión experimental haya concluido satisfactoriamente sin haber sido abortada o sin haber presentado errores. Caso contrario procede al punto **10**.
2. Llama a la subrutina *Comments.vi* el cual permite al usuario incluir unos últimos comentarios al experimento y nombrarlo.
3. Llama a la subrutina *MainSave.vi* para el almacenamiento de imágenes en el disco duro de la PC.
4. Del *cluster* principal se extrae la configuración del experimento y se da formato a sus 16 elementos para ser presentados como datos *string*.
5. De la sesión experimental se toma la hora y fecha de inicio y el tiempo total transcurrido y se da formato para ser presentada como datos *string*.
6. Llama a la subrutina *RegisterGenerator.vi* para integrar todos los datos *string* e imágenes del experimento en un *cluster* maestro de 39 elementos y en un dato *string* maestro.
7. Llama a la subrutina *TXTGenerator.vi* para crear un reporte escrito con la información y configuración del experimento a partir del dato *string* maestro.
8. Crea un archivo de registro con la información, configuración e imágenes adquiridas del experimento a partir del cluster maestro de 39 elementos.
9. La subrutina verifica errores en el proceso. Caso afirmativo notifica al usuario.
10. La subrutina finaliza.

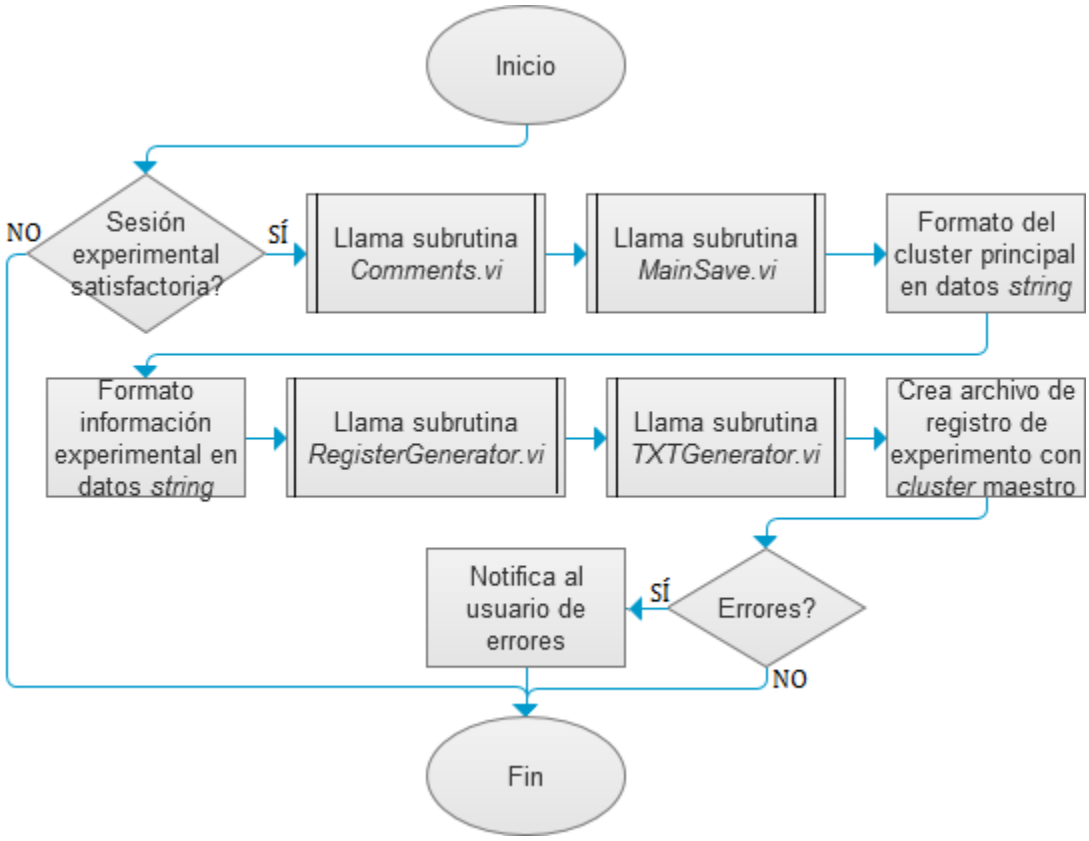


Figura 6.32 Diagrama de flujo de la subrutina ReportGenerator.vi

La subrutina *ReportGenerator.vi* cuenta con 11 parámetros de entrada y 1 parámetro de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
total time	double	Indica el tiempo total transcurrido durante el experimento
Image Array 1	1D Array [string]	Contiene las imágenes adquiridas asociadas al obturador 1
Image Array 2	1D Array [string]	Contiene las imágenes adquiridas asociadas al obturador 2
datei	string	Indica la fecha y hora de inicio del experimento
Cluster	Cluster [16 elementos]	Contiene la configuración e información referente a la sesión de experimentación: puertos USB, formato de imagen, ruta raíz, uso de obturadores, orden de operación, te, Ta y total de imágenes.
OK?	bool	Indica una sesión experimental satisfactoria
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa
name1	string	Nombre genérico de las imágenes adquiridas por el obturador 1
shutterdir1	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 1
name2	string	Nombre genérico de las imágenes adquiridas por el obturador 2
shutterdir2	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 2

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

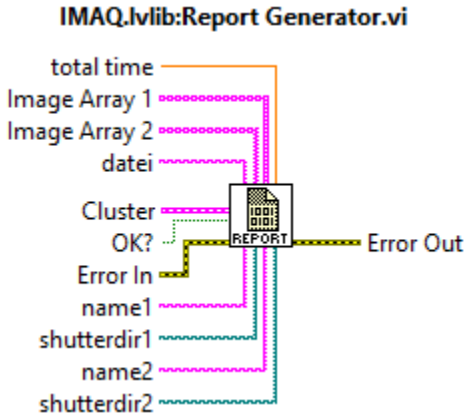


Figura 6.33 Parámetros E/S de la subrutina ReportGenerator.vi

6.2.13 Subrutina: *Comments.vi*

La subrutina *Comments.vi* permite al usuario ingresar un comentario final y nombrar al experimento. Mientras que el comentario es opcional a consideración del usuario, el nombre del experimento es obligatorio ya que servirá para nombrar tanto al archivo de registro del experimento como el reporte escrito del mismo, teniendo en cuenta esto, la subrutina verifica que no se ingresen caracteres especiales ni nombres que causen conflicto con el sistema operativo del sistema en que se esté ejecutando el software de control. El algoritmo de programación es el siguiente:

1. Inicializa los elementos del panel frontal.

2. Se solicita al usuario ingresar el nombre y comentarios del experimento a través de la interfaz de usuario emergente.
3. Se verifica que el nombre del experimento sea válido, no contenga caracteres especiales o cause conflicto con el sistema operativo de la PC. En caso de que no sea válido procede al punto **2**. Caso contrario la subrutina finaliza.

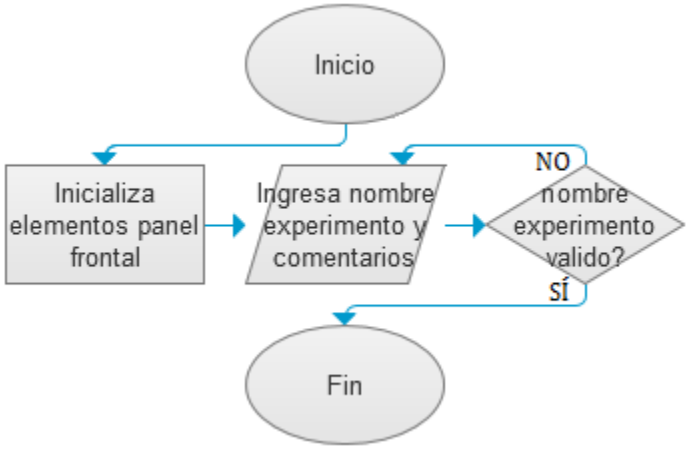


Figura 6.34 Diagrama de flujo de la subrutina Comments.vi

La subrutina *Comments.vi* cuenta con 1 parámetro de entrada y 3 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa

Parámetros de Salida

Nombre	Tipo de dato	Descripción
expname	string	Nombre del experimento
comments	string	Comentarios finales del experimento
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

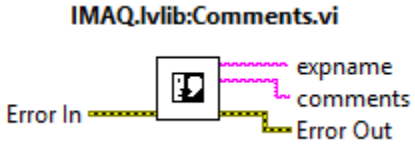


Figura 6.35 Parámetros E/S de la subrutina Comments.vi

6.2.14 Subrutina: *MainSave.vi*

La subrutina *MainSave.vi* se encarga de almacenar en el disco duro de la PC la totalidad de las imágenes adquiridas durante el experimento, asignándoles un nombre, numeración y una ruta de almacenamiento. Durante el experimento, las imágenes adquiridas son comprimidas y procesadas a través de una función llamada *FlattenToString.vi* la cual recibe como entrada un valor de cualquier tipo de dato y la convierte en una cadena binara tipo *string,* conservando todas sus propiedades originales con una perdida en calidad de imagen

mínima (imperceptible para el ojo humano). Esto debido a que manejar, tratar y procesar un dato *string* es mucho más sencillo y rápido que manejar un dato *IMAQ Image.ctl.* Una vez que la imagen es procesada, esta se aloja en la memoria RAM de la PC dentro de un arreglo unidimensional tipo *string*, con tantos elementos disponibles como se requieran para alojar la totalidad de las imágenes a adquirir, utilizándose un arreglo por obturador.

Cabe mencionar que la memoria RAM que puede ser utilizada para alojar las imágenes, representadas como cadenas *string*, es limitada y es administrada arbitrariamente por LabVIEW, ni el programador ni el usuario tienen acceso a la distribución o administración de la memoria en ningún modulo o sub modulo del programa, sin embargo esta distribución es lo suficientemente grande como para alojar hasta 5000 imágenes repartidas entre las series de imágenes asociadas a los dos obturadores, es decir 2 secuencias de imágenes de hasta 2500 imágenes cada una, más allá de eso es posible experimentar problemas de memoria durante la ejecución y no se asegura un buen funcionamiento del software de control.

La subrutina *MainSave.vi* utiliza el mismo método de compresión y procesamiento pero en su modo inverso *UnflattenFromString.vi* para convertir la cadena binaria *string* nuevamente a un imagen, asignándole un nombre, una numeración y almacenándola en la carpeta correspondiente de la ruta de trabajo raíz dentro de la memoria rígida de la PC, es decir, en el disco duro, liberando así el espacio de alojamiento en la memoria RAM. El algoritmo de programación es el siguiente:

1. Se crean carpetas de almacenamiento dentro de la ruta asignada por el usuario como ruta de trabajo raíz.
2. Se inicializa *n = 1*.
3. Se inicializa un ciclo *for(i=0; i==tin_n; i++)*
4. Se toma el elemento *ImageArray_n [i]* y se procesa la cadena *string* a imagen.
5. Se nombra la imagen y se numera *name_i*
6. La imagen se almacena en el disco duro de la PC en la ruta de almacenamiento asociada al obturador_*n*.
7. Se verifican errores durante el procedimiento. Caso afirmativo se notifica al usuario y se procede al punto **10**.
8. Se verifica que *i==tin_n*. Caso afirmativo se procede al punto **9**. Caso contrario *i++* y se procede al punto **4**.
9. Se verifica uso de obturadores. En caso de usar un solo obturador se procede al punto **10**. En caso de usar ambos se verifica *n==2*, caso afirmativo se procede al punto **10**, caso contrario *n++* y se procede al punto **3**.
10. Subrutina finaliza

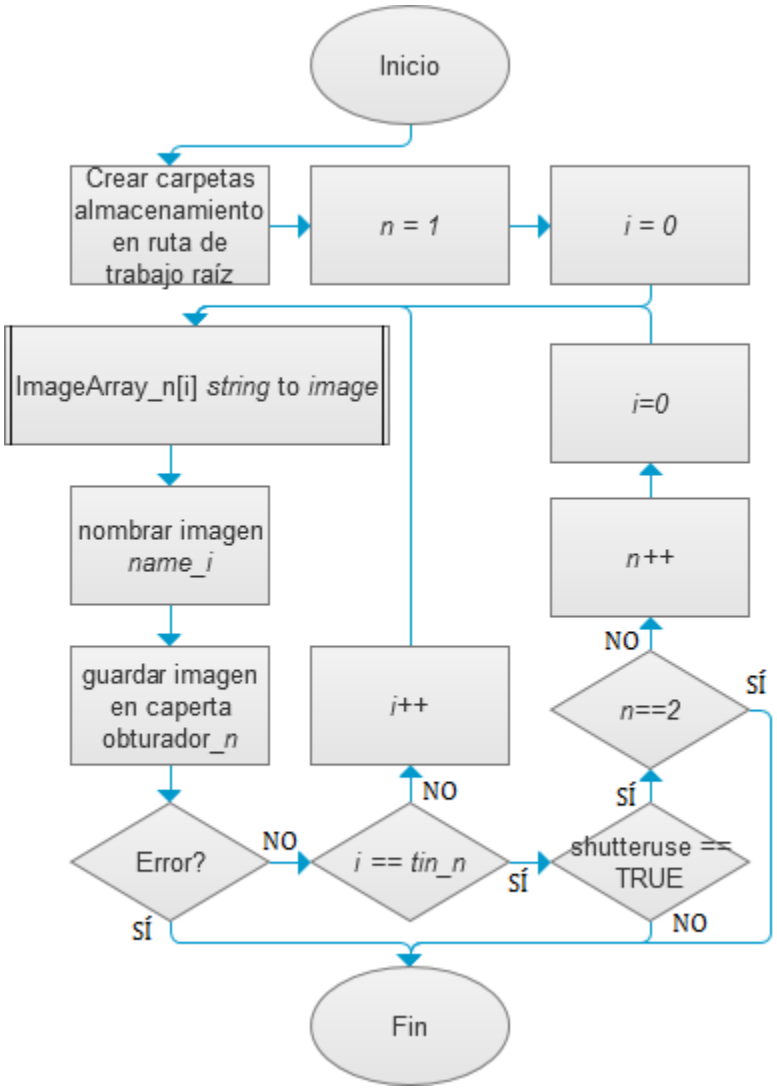


Figura 6.36 Diagrama de flujo de la subrutina MainSave.vi

La subrutina *MainSave.vi* cuenta con 10 parámetros de entrada y 1 parámetro de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
format	string	Indica el formato de guardado para las imágenes
Image Array 1	1D Array [string]	Contiene las imágenes adquiridas asociadas al obturador 1
Image Array 2	1D Array [string]	Contiene las imágenes adquiridas asociadas al obturador 2
shutterorder	int	Indica el obturador a usar y el orden de operación
shutteruse	bool	Indica si se usaran ambos obturadores durante el experimento
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa
name1	string	Nombre genérico de las imágenes adquiridas por el obturador 1
shutterdir1	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 1
name2	string	Nombre genérico de las imágenes adquiridas por el obturador 2
shutterdir2	file path	Ruta de almacenamiento para las imágenes adquiridas por el obturador 2

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

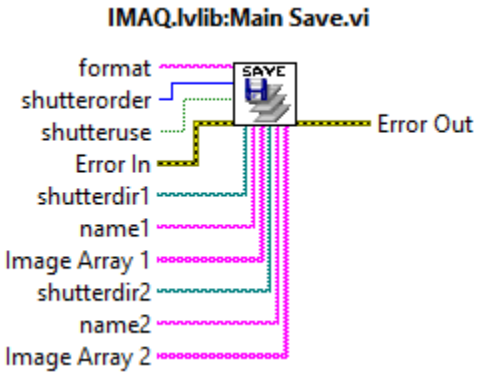


Figura 6.37 Parámetros E/S de la subrutina MainSave.vi

6.2.15 Subrutina: *RegisterGenerator.vi*

La subrutina *RegisterGenerator.vi* reúne la información contenida en la plantilla de configuración del experimento y la información recolectada durante el experimento y, elemento por elemento, les da un formato y convierte a datos tipo *string*, de tal manera que puedan ser integrados en un dato *string* maestro que a su vez pueda ser manejado y procesado fácilmente en un reporte escrito del experimento en formato *.txt* por la subrutina *TXTGenerator.vi* (**Sección 6.2.16**). De igual manera reúne en un solo *cluster* maestro de 39 elementos tanto el dato *string* maestro **txt** como el **Cluster Principal** correspondiente a la plantilla de configuración del experimento, además de un **Cluster String** con todos los elementos que integran el dato *string* maestro de manera individual y un **Cluster Imágenes** que puede contener uno o dos arreglos unidimensionales con las imágenes adquiridas durante el experimento en formato *string*, según el modo de operación seleccionado. Este **Cluster Maestro** representa el archivo de registro del experimento, el cual es creado por la subrutina *ReportGenerator.vi* (**Sección 6.2.12**) y que contiene no solamente toda la información del experimento, sino también los datos e imágenes del mismo. En la **Sección 6.2.12** se describen los datos e información que integran tanto el dato *string* maestro **txt** como el **Register Cluster**. El algoritmo de programación es el siguiente:

1. Se da formato a todos los elementos del *cluster* principal **Main Cluster** en datos *string* con su respectiva descripción.
2. Se da formato a todos los datos informativos del experimento en datos *string* con su respectiva descripción.
3. Se concatenan los datos *string* individuales en un dato *string* maestro **txt**.
4. Se crea un *cluster* **String Cluster** con todos los elementos del *cluster* principal convertidos a datos string.
5. Se crea un *cluster* maestro **Register Cluster** con el *cluster* **Main Cluster**, el *cluster* **Image Cluster**, el *cluster* **String Cluster** y el dato *string* maestro **txt**.
6. La subrutina regresa el dato *string* maestro **txt** y el *cluster* maestro **Register Cluster** y finaliza.

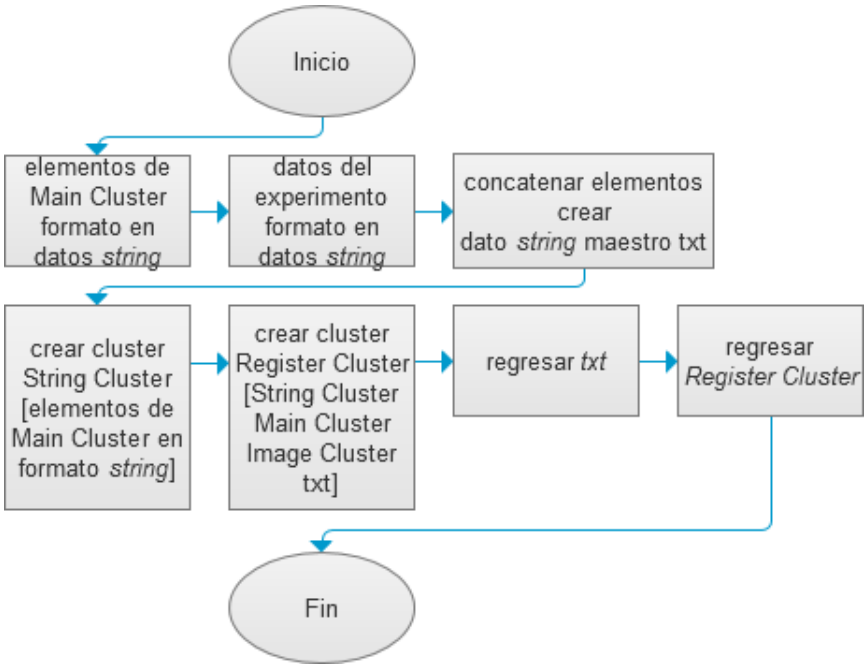


Figura 6.38 Diagrama de flujo de la subrutina RegisterGenerator.vi

La subrutina *RegisterGenerator.vi* cuenta con 18 parámetros de entrada y 2 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	**Tipo de dato**	**Descripción**
Ruta	string	Ruta de trabajo raíz del experimento
Main Cluster	Cluster [16 elementos]	Contiene la configuración e información referente a la sesión de experimentación: puertos USB, formato de imagen, ruta raíz, uso de obturadores, orden de operación, te, Ta y total de imágenes.
Formato	string	Indica el formato de guardado para las imágenes
Triggers	string	Indica cuales de los Triggers se encuentran activos y en qué imagen se activarán
label	string	Información relevante del experimento ingresada por el usuario
tin	string	Total de imágenes a adquirir asociadas al obturador 1 y al obturador 2
T2	string	Tiempo de exposición del obturador 1 y del obturador 2
T3	string	Periodo de apertura del obturador 1 y del obturador 2
Obturadores	string	Ruta de almacenamiento para las imágenes adquiridas por el obturador 1 y el obturador 2
Nombres	string	Nombre genérico de las imágenes adquiridas por el obturador 1 y el obturador 2
nombre	string	Nombre del experimento
comments	string	Comentarios finales del experimento
Images Cluster	Cluster [6 elementos]	Contiene las imágenes adquiridas asociadas al obturador 1 y al obturador 2, así como sus nombres y rutas de almacenamiento.
total time	string	Indica el tiempo total transcurrido durante el experimento
datei	string	Indica la fecha y hora de inicio del experimento
datef	string	Indica la fecha y hora de finalización del experimento
COM cam	string	Referencia a la comunicación y sesión de adquisición de la tarjeta de adquisición de imágenes conectada a la PC por USB.
COM Arduino	string	Referencia a la COM serial del microcontrolador Arduino UNO conectado a la PC por USB

Parámetros de Salida

Nombre	Tipo de dato	Descripción
txt	string	Contiene toda la información en formato *string* del experimento
Register Cluster	Cluster [39 elementos]	Contiene toda la información en formato *string*, datos y configuración en sus respectivos formatos e imágenes en formato *string* del experimento

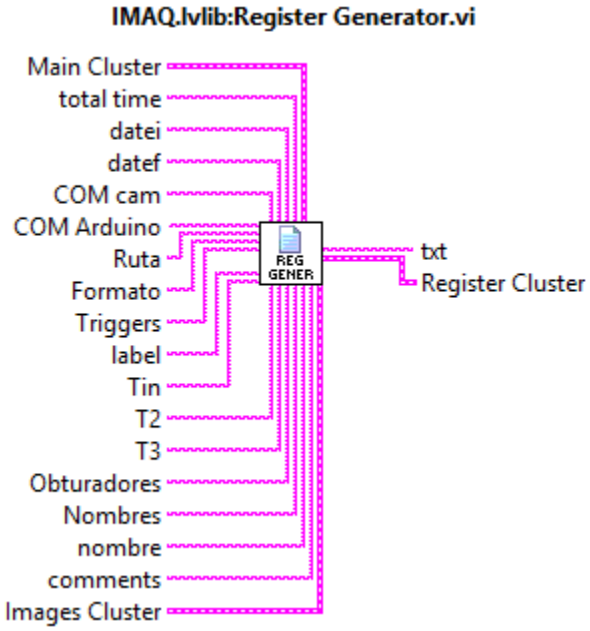


Figura 6.39 Parámetros E/S de la subrutina RegisterGenerator.vi

6.2.16 Subrutina: *TXTGenerator.vi*

La subrutina *TXTGenerator.vi* se encarga de generar el reporte escrito del experimento, incluyendo todos los datos relevantes del mismo para su registro y posterior consulta. Utiliza el dato *string* maestro **text** creado por la subrutina *RegisterGenerator.vi* (descrito en la **Sección 6.2.15**) y escribe su contenido en un archivo *.txt*, el cual es almacenado con el nombre del experimento en la ruta de trabajo raíz en el disco duro de la PC. El algoritmo de programación es el siguiente:

1. Se crea un documento .txt con el nombre del experimento asignado por el usuario en la ruta de trabajo raíz del experimento.
2. Se utiliza la función *WriteToTextFile.vi* para escribir el contenido del dato *string* maestro **text** en el documento *.txt* creado.
3. El documento *.txt* se cierra. La subrutina finaliza.

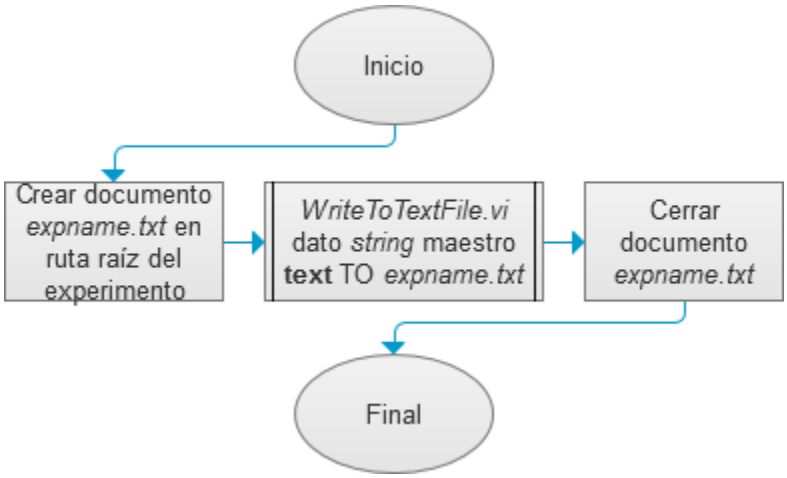


Figura 6.40 Diagrama de flujo de la subrutina TXTGenerator.vi

La subrutina *TXTGenerator.vi* cuenta con 4 parámetros de entrada y 1 parámetro de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa
Ruta Raíz	file path	Ruta de trabajo raíz del experimento
exp name	string	Nombre del experimento
text	string	Contiene toda la información en formato *string* del experimento

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

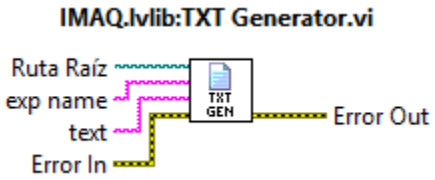


Figura 6.41 Parámetros E/S de la subrutina TXTGenerator.vi

6.2.17 Subrutina: *Save.vi*

La subrutina *Save.vi* automatiza el proceso de guardado de imágenes individuales, facilitando al usuario la adquisición y almacenamiento manual de imágenes de microscopía. La subrutina se encarga de tomar la imagen actualmente adquirida por la tarjeta de adquisición de imágenes AV Grabber (desplegada en pantalla en el modo **Adquisición** de la interfaz de usuario principal) y almacenarla (guardarla) automáticamente en la ruta de trabajo raíz, asignándoles un nombre y numeración automática o un nombre genérico elegido por el usuario, así como un formato de imagen (*jpeg*, *tiff*, *png* y *bmp*).El algoritmo de programación es el siguiente:

1. Se verifica numeración automática de imágenes. Caso afirmativo explorar ruta de trabajo raíz para verificar último número asignado *imagename_n*, *while(imagename_n exists) n++*. Caso contrario se asigna nombre *imagename*.
2. Se verifica nombre automático de imágenes. Caso afirmativo se asigna nombre: *Imagen _%d%m%y_%H%M%S*. Caso contrario se asigna nombre *imagename || imagename_n*.
3. Se crea un archivo en la ruta de trabajo raíz con el nombre y formato asignados. Si existe error se notifica al usuario y se procede al punto **6**.
4. Se almacena en disco duro la imagen adquirida en el archivo creado.
5. Se cierra el archivo.
6. La subrutina finaliza.

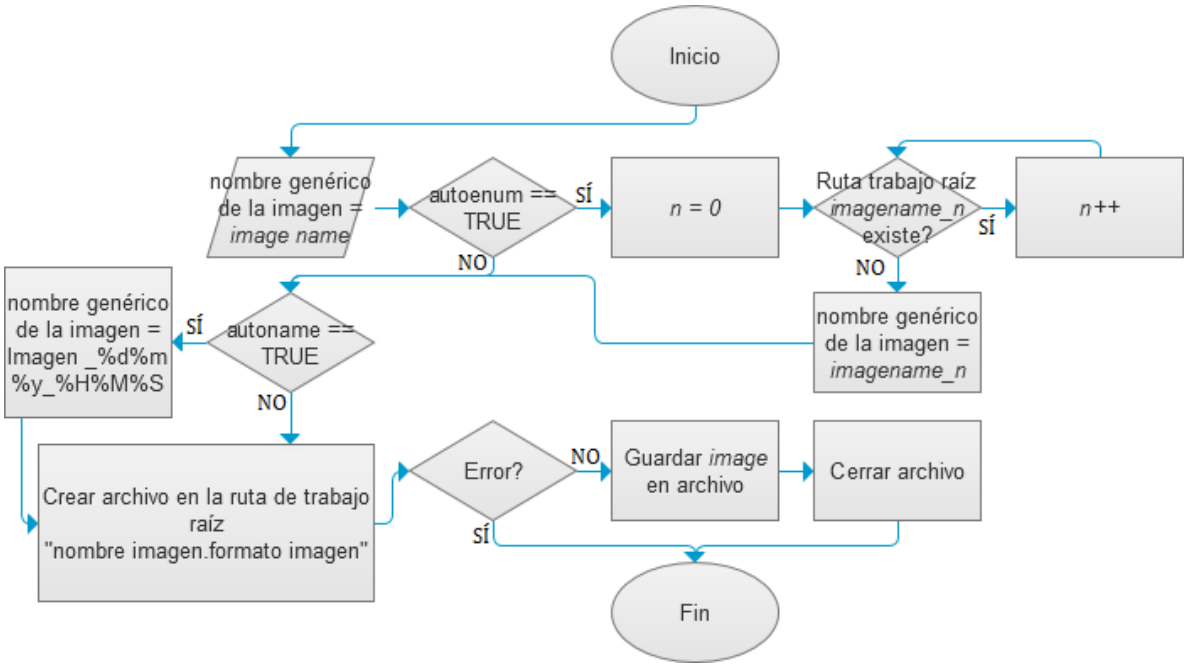


Figura 6.42 Diagrama de flujo de la subrutina Save.vi

La subrutina Save.*vi* cuenta con 6 parámetros de entrada y no posee parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
image path	file path	Ruta de trabajo raíz
image name	string	Nombre genéricos de las imágenes adquiridas
autoname	bool	Indica si se auto nombrarán las imágenes en el proceso de guardado
autoenum	bool	Indica si se auto enumerarán las imágenes en el proceso de guardado
image	typedef IMAQ Image.ctl	Referencia a la imagen actual adquirida
image format	string	Indica el formato de guardado de las imágenes

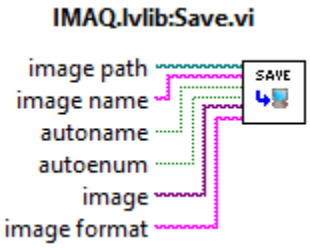


Figura 6.43 Parámetros E/S de la subrutina Save.vi

6.2.18 Subrutina: *SaveAs.vi*

La subrutina *SaveAs.vi* toma la imagen actualmente adquirida por la tarjeta de adquisición de imágenes AV Grabber (desplegada en pantalla en el modo **Adquisición** de la interfaz de usuario principal) y a través de una ventana emergente llamada desde la función *FileDialog.vi*, le permite al usuario explorar los archivos y carpetas del sistema para guardarla con un nombre, formato y en la ruta de trabajo de su elección. La subrutina funciona como cualquier comando estándar "*Guardar Como…*". El algoritmo de programación es el siguiente:

1. Se llama a la función *FileDialog.vi*
2. Si el usuario cancela el guardado de imagen desde la ventana emergente se procede al punto **6**. Caso contrario la función *FileDialog.vi* regresa la ruta de almacenamiento elegida por el usuario junto con el nombre y formato de la imagen a guardar.
3. Se crea un archivo en la ruta de almacenamiento, nombre y formato elegidos por el usuario. Si existe error se notifica al usuario y se procede al punto **6**.
4. Se almacena en disco duro la imagen adquirida en el archivo creado.
5. Se cierra el archivo.
6. La subrutina finaliza.

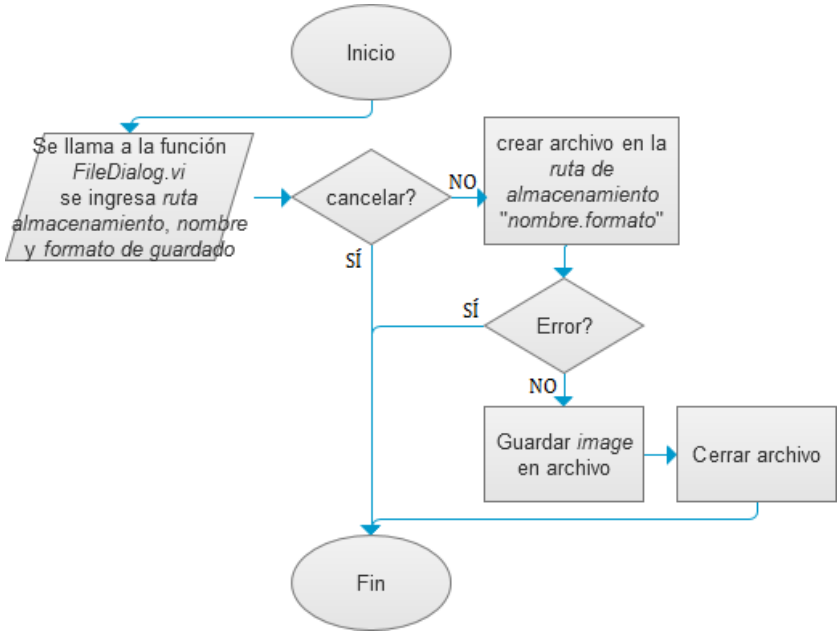


Figura 6.44 Diagrama de flujo de la subrutina SaveAs.vi

La subrutina SaveAs.*vi* cuenta con 3 parámetros de entrada y no posee parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
start path	file path	Ruta de trabajo raíz
image	typedef IMAQ Image.ctl	Referencia a la imagen actual adquirida
image format	string	Indica el formato de guardado de las imágenes

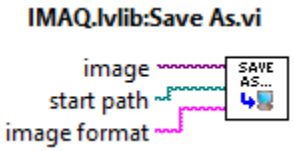


Figura 6.45 Parámetros E/S de la subrutina SaveAs.vi

6.2.19 Subrutina: *LoadSession.vi*

La subrutina *LoasSession.vi* carga en la memoria del software de control el *cluster* maestro **Register Cluster** contenido en los distintos archivos de registro de los experimentos previamente ejecutados para la revisión y análisis de su contenido: revisión y respaldo del reporte escrito del experimento, visualización y respaldo de las secuencias de imágenes y revisión de la plantilla de configuración del experimento. La subrutina llama a la función *FileDialog.vi* el cual abre un explorador de carpetas y archivos del sistema que le permite explorar y elegir al usuario un archivo de registro de experimento a cargar, todo a través de una ventana emergente. El algoritmo de programación es el siguiente:

1. Se llama a la función *FileDialog.vi*
2. Si el usuario cancela la carga del archivo de registro desde la ventana emergente se procede al punto **6**. Caso contrario la función *FileDialog.vi* regresa la ruta del archivo de registro elegida por el usuario.
3. Se abre el archivo de registro.
4. Se carga el *cluster* maestro **Register Cluster** que contiene: dato *string* maestro **txt**, *cluster* **Main Cluster**, *cluster* **Images Cluster** y *cluster* **String Cluster**. Si se produce algún error al momento de la carga se notifica al usuario y se procede al punto **6**.
5. Se cierra el archivo de registro.
6. La subrutina finaliza.

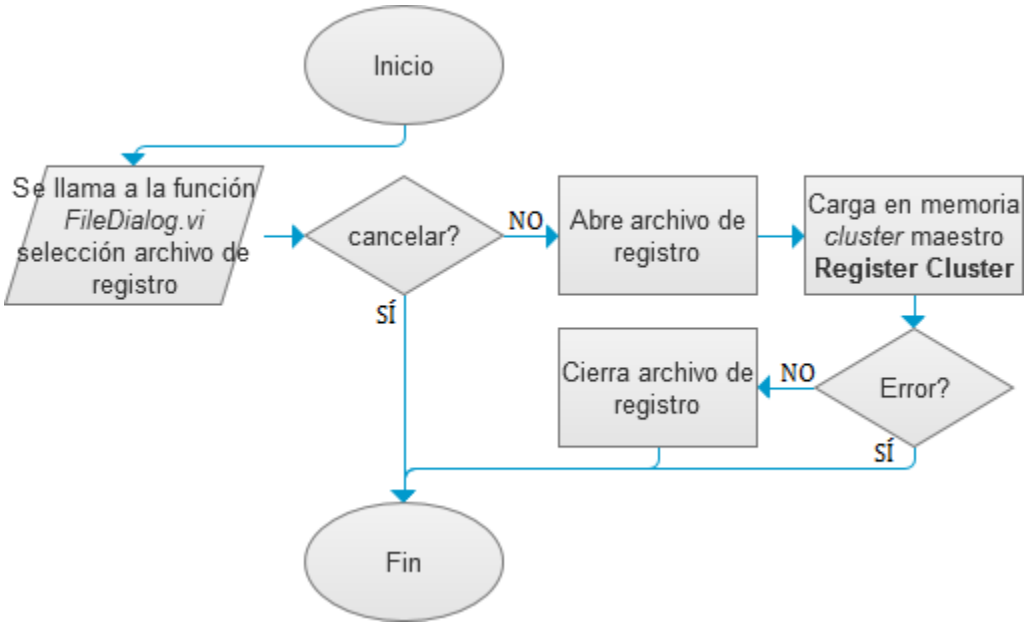


Figura 6.46 Diagrama de flujo de la subrutina LoadSession.vi

La subrutina *LoadSession.vi* no posee parámetros de entrada y cuenta con 7 parámetros de salida. Estos son:

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Main Cluster	Cluster [16 elementos]	Contiene la configuración e información referente a la sesión de experimentación: puertos USB, formato de imagen, ruta raíz, uso de obturadores, orden de operación, te, Ta y total de imágenes.
display	1D Array [int]	Indica a que obturador están asociadas las secuencias de imágenes contenidas en Images Cluster
Enable/Disable	int	Habilita los controles de respaldo de secuencias de imágenes y respaldo de reporte escrito del experimento
OK?	bool	Indica que la carga del archivo de registro se realizó satisfactoriamente
Images Cluster	Cluster [6 elementos]	Contiene las imágenes adquiridas asociadas al obturador 1 y al obturador 2, así como sus nombres y rutas de almacenamiento.
String Cluster	Cluster [16 elementos]	Contiene los datos *string* individuales que integran el dato *string* maestro **txt**
info	string	Contiene toda la información en formato *string* del experimento

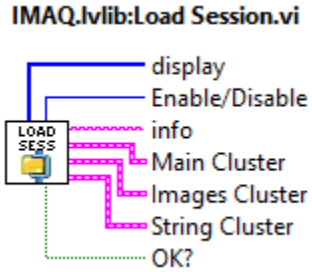


Figura 6.47 Parámetros E/S de la subrutina LoadSession.vi

6.2.20 Subrutina: *BackUp.vi*

La subrutina *BackUp.vi* se encarga de respaldar la información contenida en el *cluster* maestro **Register Cluster** extraído del archivo de registro del experimento. Específicamente, la información que se respalda es el reporte escrito del experimento con ayuda de la subrutina *TXTGenerator.vi* (**Sección 6.2.16**) y las secuencias de imágenes asociadas al obturador 1 y/o al obturador 2 con ayuda de la subrutina *MainSave.vi* (**Sección 6.2.14**). La subrutina llama a la función *FileDialog.vi* el cual abre un explorador de carpetas y archivos del sistema que le permite explorar y elegir al usuario una ruta de trabajo raíz en donde almacenar el respaldo tanto del reporte escrito del experimento como de las secuencias de imágenes, todo a través de una ventana emergente. El algoritmo de programación es el siguiente:

1. Se llama a la función *FileDialog.vi*
2. Si el usuario cancela la elección de una ruta de trabajo desde la ventana emergente se procede al punto **5**. Caso contrario la función *FileDialog.vi* regresa la ruta de trabajo elegida por el usuario.
3. Se verifica el tipo de respaldo solicitado por el usuario. Si *i == TRUE* se llama a la subrutina *MainSave.vi* con la ruta de trabajo elegida por el usuario. Si *txt == TRUE* se llama a la subrutina *TXTGenerator.vi* con la ruta de trabajo elegida por el usuario.
4. Se verifican errores durante el respaldo. En caso afirmativo se notifica al usuario.
5. La subrutina finaliza.

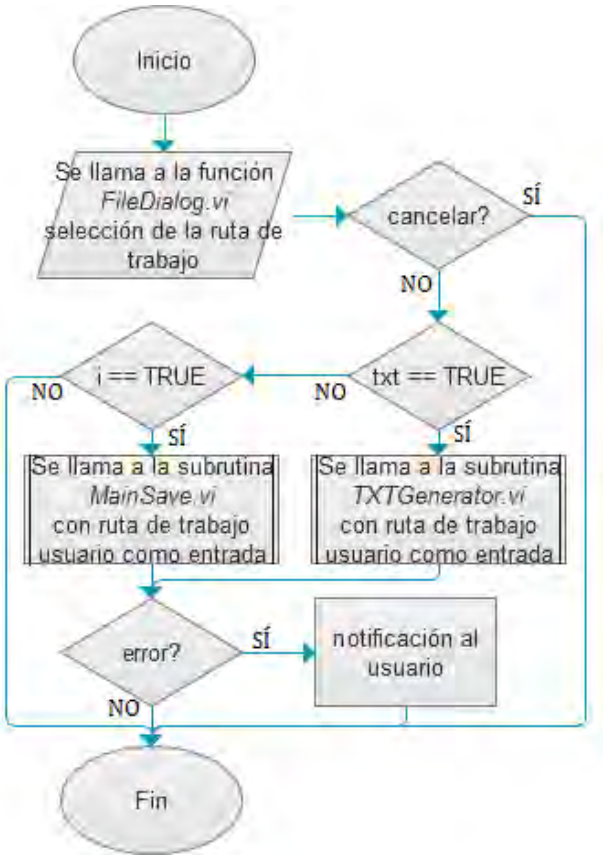


Figura 6.48 Diagrama de flujo de la subrutina BackUp.vi

La subrutina *BackUp.vi* cuenta con 7 parámetros de entrada y 1 parámetro de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Main Cluster	Cluster [16 elementos]	Contiene la configuración e información referente a la sesión de experimentación: puertos USB, formato de imagen, ruta raíz, uso de obturadores, orden de operación, te, Ta y total de imágenes.
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa
i?	bool	Indica si se hará un respaldo de las secuencias de imágenes
txt?	bool	Indica si se hará un respaldo del reporte escrito del experimento
Images Cluster	Cluster [6 elementos]	Contiene las imágenes adquiridas asociadas al obturador 1 y al obturador 2, así como sus nombres y rutas de almacenamiento.
String Cluster	Cluster [16 elementos]	Contiene los datos *string* individuales que integran el dato *string* maestro **txt**
info	string	Contiene toda la información en formato *string* del experimento

Parámetros de Salida

Nombre	Tipo de dato	Descripción
Error Out	Cluster [3 elementos]	Indica error en la ejecución de la función/vi actual

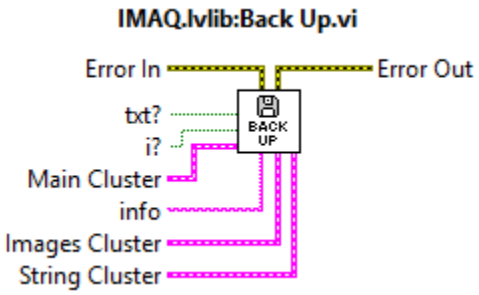


Figura 6.49 Parámetros E/S de la subrutina BackUp.vi

6.2.21 Subrutina: *PathChanged.vi*

La subrutina *PathChanged.vi* se encarga de modificar la ruta de trabajo raíz del software de control, aquella en donde se almacenan las imágenes adquiridas de forma manual, así como las secuencias de imágenes adquiridas de manera automática en las sesiones experimentales, junto con el reporte escrito y el archivo de registro del mismo. La subrutina llama a la función *FileDialog.vi* el cual abre un explorador de carpetas y archivos del sistema que le permite explorar y elegir al usuario una ruta de trabajo raíz nueva, todo a través de una ventana emergente. El algoritmo de programación es el siguiente:

1. Se llama a la función *FileDialog.vi*
2. Si el usuario cancela la elección de una ruta de trabajo raíz desde la ventana emergente la subrutina regresa error == TRUE. Caso contrario la función *FileDialog.vi* regresa la ruta de trabajo raíz elegida por el usuario y la subrutina regresa error == FALSE.
3. La subrutina finaliza.

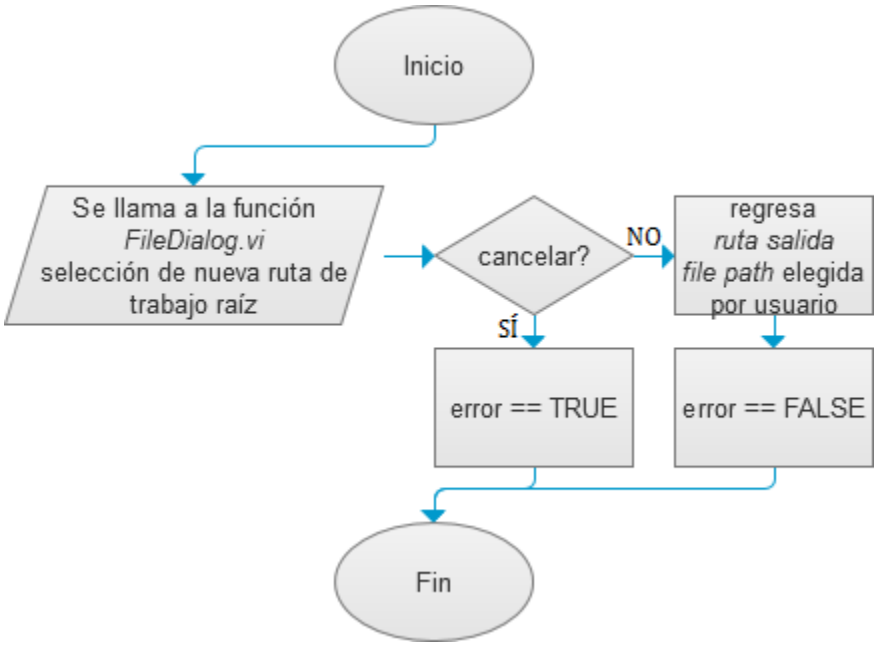


Figura 6.50 Diagrama de flujo de la subrutina PathChanged.vi

La subrutina *PathChanged.vi* cuenta con 1 parámetro de entrada y 2 parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
start path	file path	Indica la ruta de trabajo raíz actual

Parámetros de Salida

Nombre	Tipo de dato	Descripción
ruta salida	file path	Indica la ruta de trabajo raíz nueva elegida por el usuario
error	bool	Indica error o cancelación en la ejecución de la subrutina

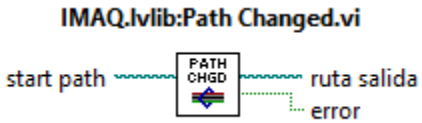


Figura 6.51 Parámetros E/S de la subrutina PathChanged.vi

6.2.22 Subrutina: *SaveConfig.vi*

La subrutina *SaveConfig.vi* se encarga de guardar en el archivo de configuración ya existente o nuevo, según sea el caso, las modificaciones realizadas en la selección de los puertos USB físicos a utiliza por defecto para la comunicación con la tarjeta de adquisición de imágenes AV Grabber y el microcontrolador Arduino UNO, establecidos por el usuario desde el **Panel de Opciones y Configuración** (**Sección 6.3.2**) de la interfaz de usuario principal. El algoritmo de programación es el siguiente:

1. Verifica que exista la carpeta raíz en la ruta raíz del software de control. Si no existe la crea.
2. Verifica que exista el archivo de configuración. Si no existe lo crea, caso contrario abre el archivo de configuración.
3. Guarda la nueva configuración establecida por el usuario en el archivo de configuración.
4. Cierra el archivo de configuración.
5. La subrutina finaliza.

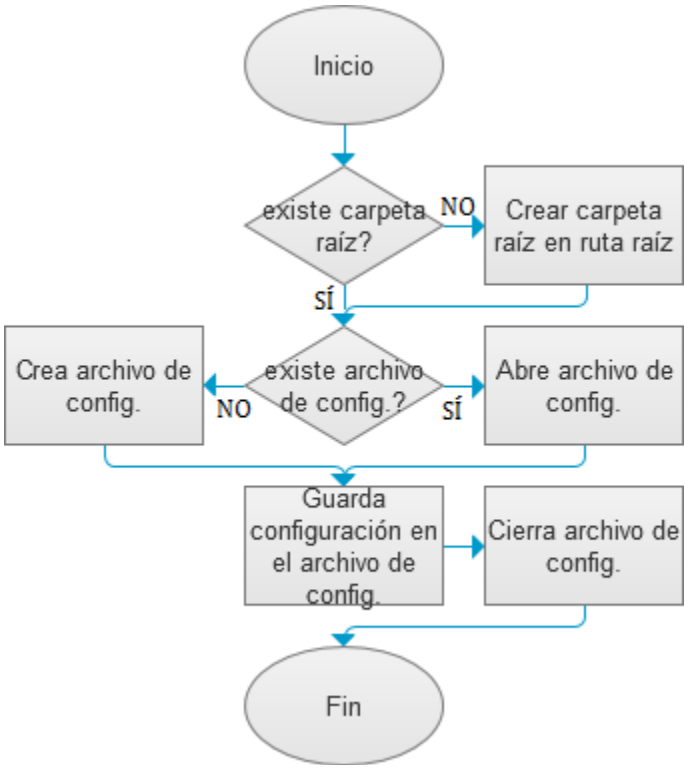


Figura 6.52 Diagrama de flujo de la subrutina SaveConfig.vi

La subrutina *SaveConfig.vi* cuenta con 2 parámetros de entrada y no posee parámetros salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
cam	typedef IMAQdx.ctl	Referencia al puerto USB de la tarjeta de adquisición de imágenes
arduino	typedef IMAQdx.ctl	Referencia al puerto USB del microcontrolador Arduino UNO

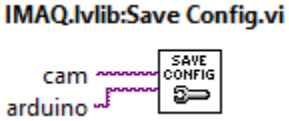


Figura 6.53 Parámetros E/S de la subrutina SaveConfig.vi

6.2.23 Subrutina: *Close.vi*

La subrutina *Close.vi* finaliza la comunicación a nivel hardware entre la tarjeta de adquisición de imágenes AV Grabber y el software de control, una vez que esta se detiene elimina la instancia que hace referencia a esta comunicación y de la misma manera finaliza la sesión de adquisición de imágenes de tipo *continua*. Finalmente vacía y elimina la instancia utilizada para el almacenamiento temporal de las imágenes adquiridas. El algoritmo de programación es el siguiente:

1. Detiene la sesión de adquisición tipo *continua*.
2. Detiene la comunicación entre la tarjeta de adquisición de imágenes y el software de control.
3. Elimina la instancia que hace referencia a la comunicación con la tarjeta de adquisición de imágenes AV Grabber.
4. Vacía y elimina la instancia para el almacenamiento temporal de las imágenes adquiridas.
5. La subrutina finaliza.

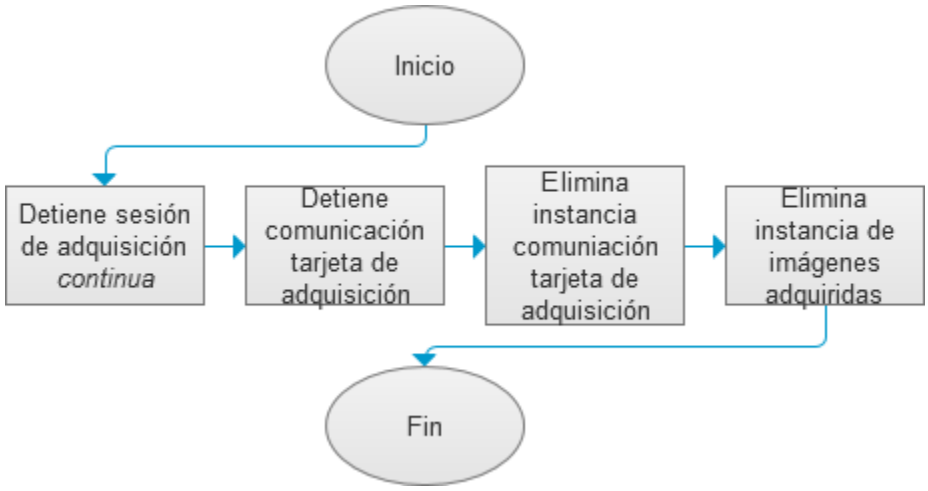


Figura 6.54 Diagrama de flujo de la subrutina Close.vi

La subrutina *Close.vi* cuenta con 3 parámetros de entrada y no posee parámetros de salida. Estos son:

Parámetros de Entrada

Nombre	Tipo de dato	Descripción
Session In	typedef IMAQdx.ctl	Referencia al puerto USB de la tarjeta de adquisición de imágenes
Error In	Cluster [3 elementos]	Indica error en la ejecución de la función/vi previa
Image	typedef IMAQ Image.ctl	Referencia a la imagen actual adquirida

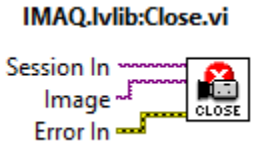


Figura 6.55 Parámetros E/S de la subrutina Close.vi

6.3 Descripción de la Interfaz de Usuario

La interfaz de usuario del software de control **FIMAQ** son todos aquellos controles, indicadores, imágenes o elementos gráficos desplegados en pantalla que el usuario utiliza para controlar o recibir información del software de control. Toda aplicación o programa elaborado en LabVIEW, cuenta con un panel frontal y un diagrama de bloques. Mientras que el primero despliega los elementos visibles y accesibles del instrumento virtual, el diagrama de bloques representa los elementos no visibles y la programación del instrumento virtual, emulando así un instrumento físico real, en donde el operador tiene una parte física que manipula, referente a los controles, perillas e indicadores del instrumento y una parte abstracta que no controla, referente a los circuitos integrados y la programación de estos para realizar las funciones de medición del instrumento.

El panel frontal del archivo de programación principal *HMI.vi* representa en gran medida la interfaz de usuario del software de control, sin embargo también existen algunas subrutinas que utilizan ventanas emergentes importantes para el usuario durante la operación y manejo del software de control, como la ventana de resumen de experimento (*Preview.vi*) y la ventana de experimento (*MainRoutine.vi*). Finalmente también existen funciones que utilizan ventanas de exploración de archivos y carpetas, pero que no se consideran como tal parte de la interfaz de usuario del software de control.

A continuación se describen las características, funciones y usos tanto del panel frontal del programa principal *HMI.vi* como del panel frontal de aquellas subrutinas *Preview.vi* y *MainRoutine.vi* que utilizan ventanas emergentes importantes, y que en conjunto, integran la interfaz de usuario del software de control FIMAQ.

6.3.1 Inicialización del Software

La inicialización del software se lleva a cabo en la rutina principal de programación *HMI.vi*. En la **Figura 6.56** se puede observar el diagrama de bloques encargado de inicializar la mayoría de los elementos desplegados en el panel frontal de la interfaz de usuario principal. Durante esta etapa de inicialización, el usuario no tiene control sobre el software de control y el cursor del mouse es bloqueado durante unos segundos, que generalmente es el tiempo requerido para cargar la interfaz de usuario y demás funciones del software de control. Las tareas que se realizan durante la inicialización son las siguientes:

- Se bloquea el cursor del mouse para evitar interferencias por parte del usuario al momento de inicializar el panel frontal del instrumento virtual, el cual representa la interfaz de usuario del software de control.
- La ventana de la interfaz de usuario se ajusta a la resolución establecida por el sistema operativo de la PC y se ocultan los controles y opciones avanzadas del panel frontal.
- Se cargan las librerías internas del software de control: **IMAQ.lvlib**, **vi.lib**, **niimaqdx.dll, nivision.dll** y **nivissvc.dll**.
- Se cargan los elementos gráficos del panel de control: imágenes, controles e indicadores.

- Utilizando nodos de propiedad, se ocultan las variables internas del software de control, representadas como controles e indicadores en el panel frontal, evitando así que el usuario tenga acceso directo a ellas.
- Utilizando variables locales y referencias, se fijan los valores por defecto de las variables controladas directamente por el usuario, representadas como controles e indicadores en el panel frontal.

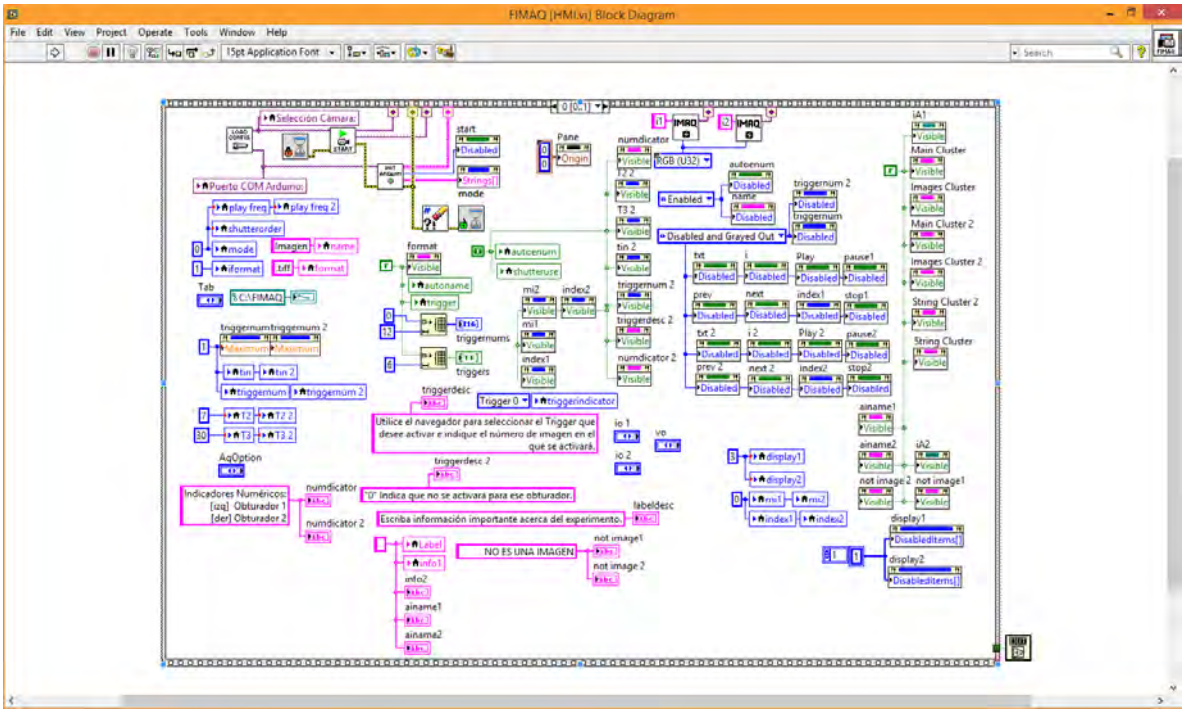


Figura 6.56 Diagrama de bloques que inicializa el panel frontal del instrumento virtual, el cual representa la interfaz de usuario principal del programa.

La inicialización termina cuando el cursor del mouse se desbloquea y se despliega en pantalla la interfaz de usuario mostrada en la **Figura 6.57**, permitiendo al usuario comenzar a operar y utilizar las funciones del software de control. A continuación se describen los elementos de la interfaz de usuario principal:

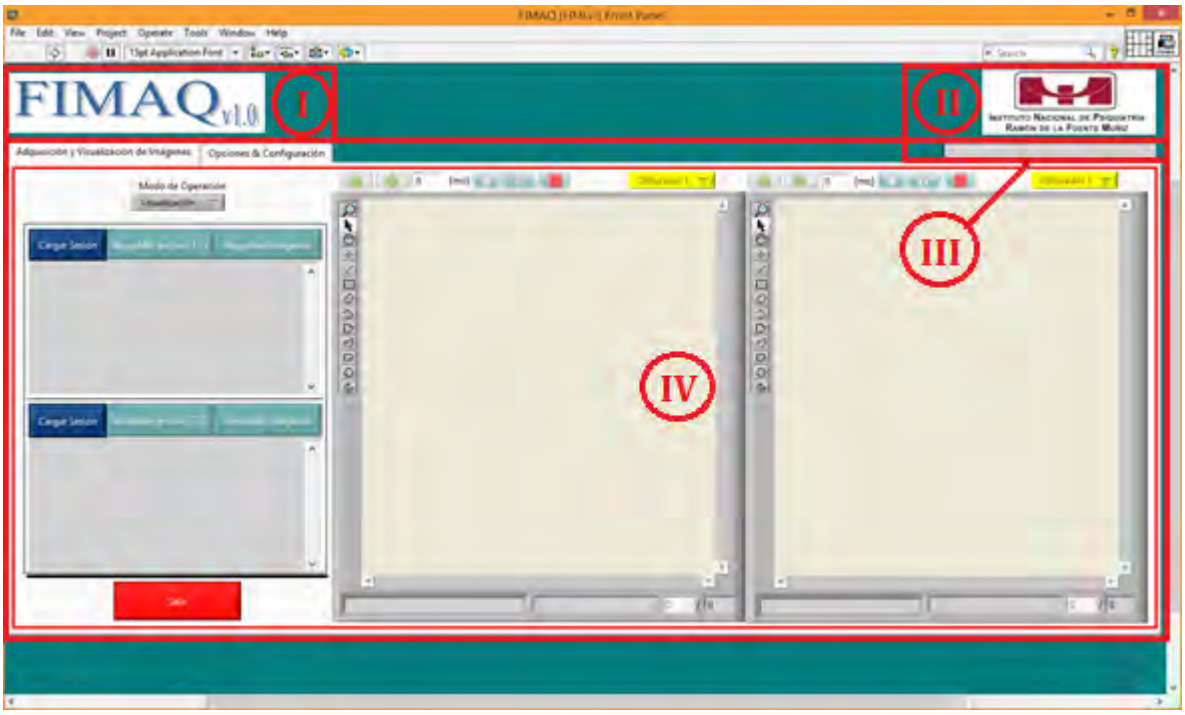


Figura 6.57 Interfaz de usuario principal del software de control ***FIMAQ****.*

I. Despliega el nombre comercial y versión del software de control.
II. Despliega el logotipo del instituto de investigación en donde se desarrolló del software de control.
III. Despliega la fecha y hora en tiempo real.
IV. Contiene los dos paneles de operación principales del software de control: el **Panel de Opciones y Configuración** (**Sección 6.3.2**) y **Panel de Adquisición** (**Sección 6.3.4**) y **Visualización** (**Sección 6.3.3**) de imágenes.

6.3.2 Panel de Opciones y Configuración

El panel de opciones y configuración es accesible desde el panel frontal del programa principal *HMI.vi* y le permite al usuario controlar la configuración de los puertos físicos USB. Desde este panel el usuario puede realizar las siguientes tareas:

- Revisar la configuración actual de los puertos físicos USB.
- Modificar la configuración actual de los puertos físicos USB.
- Guardar una nueva configuración de los puertos físicos USB.
- Reestablecer (cargar) la configuración de los puertos físicos USB.

A continuación se describen cada uno de los elementos presentes en la pantalla de **Opciones y Configuración**:

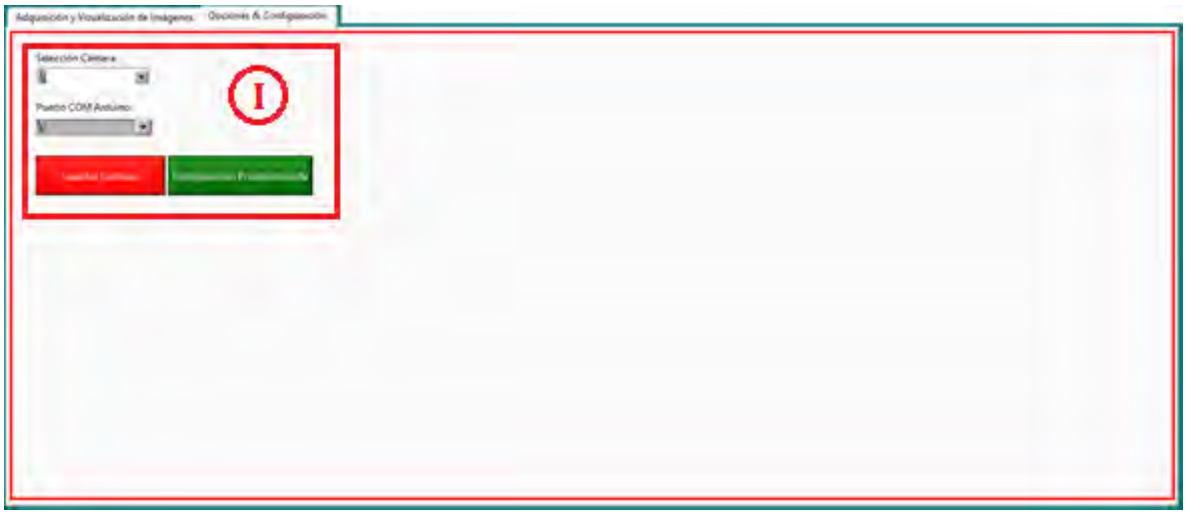


Figura 6.58 Interfaz de usuario del Panel de Opciones y Configuración.

I. **Selección Cámara.** Permite revisar y modificar el puerto físico USB utilizado actualmente para establecer comunicación con la tarjeta de adquisición de imágenes AV Grabber.
Puerto COM Arduino. Permite revisar y modificar el puerto físico USB utilizado actualmente para establecer comunicación con el microcontrolador Arduino UNO.
Guardar Cambios. Guarda los cambios realizados en la configuración de los puertos físicos USB.
Configuración Predeterminada. Carga la configuración predeterminada de los puertos físicos USB.

6.3.3 Panel de Visualización

El panel de visualización es accesible desde el panel frontal del programa principal *HMI.vi* y le permite al usuario visualizar imágenes almacenadas tanto en la PC como en los archivos de registro experimentales. Desde este panel el usuario puede realizar las siguientes tareas:

- Cargar y visualizar imágenes almacenadas en el disco duro de la PC.
- Cargar archivos de registro de experimentos.
- Visualizar las secuencias de imágenes almacenadas en los archivos de registro experimentales.
- Manipular la visualización de las imágenes y secuencias de imágenes.
- Respaldar las secuencias de imágenes almacenadas en los archivos de registro experimentales.
- Respaldar la información almacenada en los archivos de registro experimentales en un reporte escrito *.txt*

A continuación se describen cada uno de los elementos presentes en la pantalla de **Visualización:**

Figura 6.59 Interfaz de usuario del Panel de Visualización.

I. **Modo de Operación:** Permite seleccionar entre dos modos de operación del software de control: 1) visualización de imágenes o 2) adquisición de imágenes en tiempo real.
Visualización. Accede al panel de Visualización.
Adquisición. Accede al panel de adquisición.

II. **Anillo de selección de imágenes:** Permite seleccionar las imágenes a visualizar.
Obturador 1. Seleccionar la secuencia de imágenes asociada al obturador 1 dentro del archivo de registro del experimento.
Obturador2. Seleccionar la secuencia de imágenes asociada al obturador 2 dentro del archivo de registro del experimento.
Imagen. Selecciona la o las imágenes almacenadas en las carpetas del disco duro de la PC.
------- (clear). Limpiar el visualizador de imágenes.

III. **Visualizador de imágenes.** Permite visualizar imágenes almacenadas en la PC o en los archivos de registro de experimento. El visualizador tiene una paleta de comandos para crear regiones de interés (**ROI**'s por sus siglas en inglés Region of Interest) y realizar acercamientos. En la parte inferior se visualiza el nombre de la imagen, ubicación y numeración dentro de su ubicación.

IV. **Previous.** Despliega la imagen anterior dentro de la carpeta de imágenes del disco duro o dentro de la secuencia de imágenes del archivo de registro.
Next. Despliega la siguiente imagen dentro de la carpeta de imágenes del disco duro o dentro de la secuencia de imágenes del archivo de registro.
Timer [ms]. Establece el periodo de tiempo en que las imagines contenidas en la carpeta del disco duro o en la secuencia de imágenes del archivo de registro se desplegarán automáticamente.
Play. Despliega automáticamente las imágenes contenidas en la carpeta del disco duro o en la secuencia de imágenes del archivo de registro.
Pause. Pausa el despliegue automático de las imagines contenidas en la carpeta del disco duro o en la secuencia de imágenes del archivo de registro.
Stop. Detiene el despliegue automático las imagines contenidas en la carpeta del disco duro o en la secuencia de imágenes del archivo de registro.

V. **Cargar Sesión.** Permite cargar un archivo de registro experimental.
Respaldar archivo TXT. Respalda la información contenida en el archivo de registro experimental (Véase **Sección 6.2.12**) a través de un reporte escrito *.txt*.
Respaldar Imágenes. Respalda las secuencias de imágenes contenidas en el archivo de registro experimental dentro de la ruta de trabajo raíz del software de control (*C:\FIMAQ*).

VI. **Despliegue de información.** Muestra la información contenida en el archivo de registro experimental.

6.3.4 Panel de Adquisición

El panel de adquisición es accesible desde el panel frontal del programa principal *HMI.vi* y le permite al usuario adquirir imágenes de microscopía en tiempo real y almacenarlas en el disco duro de la PC, así como configurar y ejecutar experimentos de adquisición de secuencias de imágenes de microscopía de fluorescencia. Desde este panel el usuario puede realizar las siguientes tareas:

- Visualización de imágenes de microscopía en tiempo real.
- Guardar imágenes en el disco duro de la PC dentro de una ruta de trabajo preestablecida o una ruta establecida por el usuario al momento de guardar la imagen.
- Establecer el nombre y formato de guardado para las imágenes almacenadas en el disco duro de la PC.
- Establecer la ruta de trabajo para el guardado de imágenes y las sesiones experimentales.
- Establecer el uso y orden de operación de los obturadores durante la sesión experimental.
- Establecer los tiempos de exposición, periodo de apertura y total de imágenes a adquirir para cada obturador.
- Programa el uso de señales TTL (Triggers) para activar otros dispositivos durante las sesiones experimentales.
- Agregar etiquetas y comentarios sobre el experimento.
- Iniciar la sesión experimental.

A continuación se describen cada uno de los elementos presentes en la pantalla de **Adquisición:**

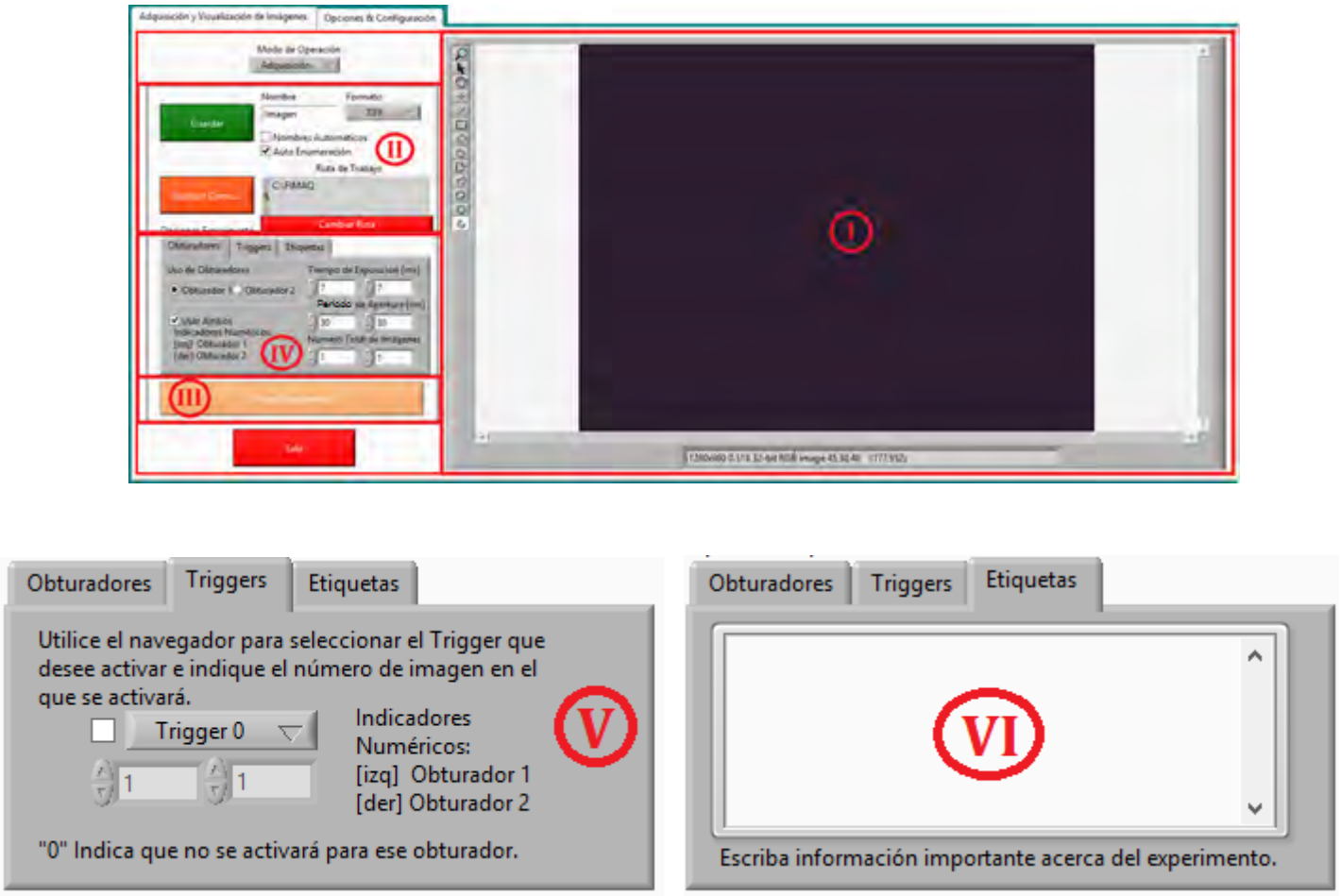


Figura 6.60 Interfaz de usuario del Panel de Adquisición.

I. **Visualizador de imágenes.** Permite visualizar las imágenes de microscopía en tiempo real. El visualizador tiene una paleta de comandos para crear ROI's y realizar acercamientos.

II. **Nombre.** Permite establecer el nombre genérico de las imágenes que serán guardadas en el disco duro de la PC.
Formato. Permite establecer el formato de guardado de las imágenes *(.tiff*, *.jpeg*, *.bmp*, *.png*) que serán almacenadas en el disco duro de la PC.
Nombre Automáticos. Permite al software de control autonombrar las imágenes que serán guardadas en el disco duro de la PC. El software nombra la imagen a partir de la fecha y hora de guardado.
Auto Enumeración. Permite al software de control auto enumerar las imágenes que serán guardadas en el disco duro de la PC a partir de un nombre genérico establecido por el usuario.
Ruta de Trabajo. Permite establecer la ruta de trabajo raíz para el guardado de imágenes y del experimento.
Guardar. Guarda la imagen actual desplegada en el visualizador en el disco duro de la PC, en la ruta de trabajo establecida y según el nombre y formato de guardado establecidos por el usuario.

Guardar Como. Abre una ventana de exploración de archivos que le permite al usuario elegir la ruta de almacenamiento así como nombre y formato de guardado para la imagen actual desplegada en el visualizador.

III. **Iniciar Experimento.** Inicia la sesión de experimentación con las opciones y configuraciones establecidas.

IV. **Uso de Obturadores.** Si se utilizará un solo obturador durante el experimento, permite establecer cuál de los dos se utilizará. Si se utilizarán ambos obturadores durante el experimento, permite establecer cuál de los dos comenzará a operar primero.
Usar Ambos. Estable si se utilizarán ambos obturadores durante el experimento.
Tiempo de Exposición [ms]. Establece el tiempo en milisegundos de exposición para cada uno de los obturadores. Esto se traduce como el tiempo que el obturador permanecerá abierto.
Periodo de Apertura [ms]. Establece el periodo de apertura en milisegundos para cada uno de los obturadores. Esto se traduce como el intervalo de tiempo en [ms] ente una instancia de apertura y otra del obturador.
Número Total de Imágenes. Establece el número total de imágenes a adquirir asociadas al obturador 1 y al obturador 2. Esto se traduce como el número de imágenes de microscopía que se adquieren cuando el obturador está abierto dejando pasar la luz de excitación al modelo biológico.

V. **Triggers.** Establece que Triggers se activarán durante el experimento y en qué imágenes (asociada al obturador 1 o al obturador 2) se activarán.

VI. **Etiquetas.** Permite al usuario agregar comentarios, etiquetas e información relevante sobre el experimento, como fluoróforos a utilizar, filtros, composición del modelo biológico, etc. Esta información estará presente en el reporte escrito del experimento.

6.3.5 Ventana Resumen de Experimento

La ventana de resumen de experimento es accesible desde el panel frontal de la subrutina *Preview.vi* y le permite al usuario visualizar un resumen general con todas las opciones de configuración del experimento antes de iniciar la ejecución del mismo. Desde esta ventana el usuario puede realizar las siguientes tareas:

- Visualizar los puertos físicos USB activos utilizados para la comunicación serial con el microcontrolador Arduino UNO y la comunicación con la tarjeta de adquisición de imágenes AV Grabber.
- Modificar los nombres genéricos de las imágenes adquiridas durante el experimento.
- Visualizar el formato de guardado de las imágenes adquiridas durante el experimento.
- Visualizar la ruta de trabajo raíz del experimento.
- Modificar los nombres de las carpetas de almacenamiento.
- Visualizar los Tiempos de Exposición (*te* [ms]), Periodo de Apertura (*Ta* [ms]) y Total de Imágenes a adquirir asociados a ambos obturadores.

- Visualizar el uso de Triggers durante el experimento, qué pines se usarán y en qué imagen se activarán.
- Visualizar la duración total aproximada del experimento.

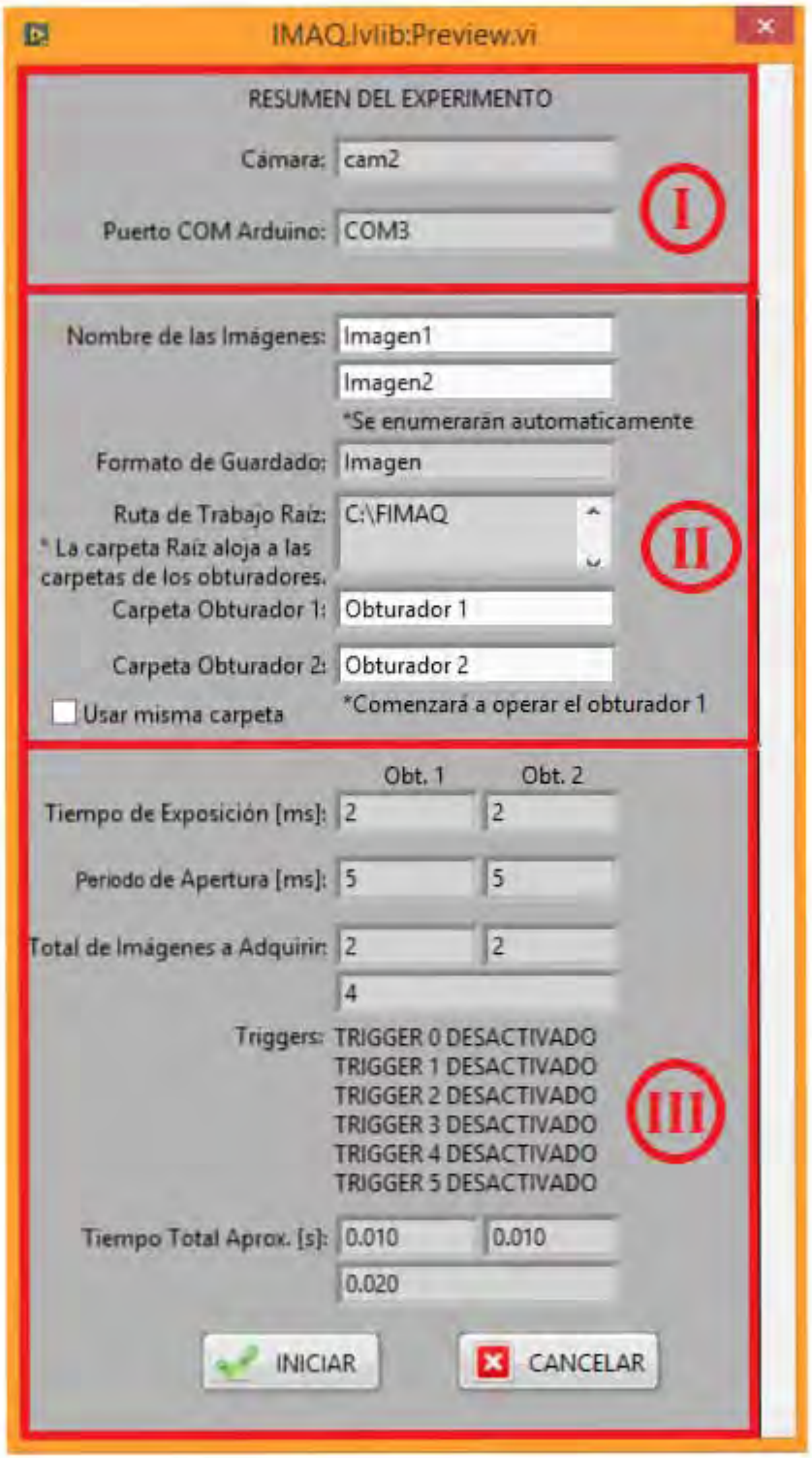


Figura 6.61 Interfaz de usuario de la Ventana Resumen de Experimento.

I. **Cámara.** Indica el puerto físico USB utilizado para establecer la comunicación con la tarjeta de adquisición de imágenes AV Grabber.
Puerto COM Arduino. Indica el puerto físico USB utilizado para establecer la comunicación serial con el microcontrolador Arduino UNO.

II. **Nombre de las Imágenes.** Permite establecer el nombre genérico de las imágenes adquiridas durante el experimento, tanto para aquellas asociadas al obturador 1 como aquellas asociadas al obturador 2.
Formato de Guardado. Indica el formato de guardado para las imágenes adquiridas durante el experimento.
Ruta de Trabajo Raíz. Indica la ruta de trabajo raíz del experimento, en donde se almacenarán tanto las imágenes adquiridas como el reporte escrito y el archivo de registro del experimento.
Carpeta Obturador 1. Permite establecer el nombre de la carpeta de almacenamiento para la secuencia de imágenes adquiridas asociadas al obturador 1.
Carpeta Obturador 2. Permite establecer el nombre de la carpeta de almacenamiento para la secuencia de imágenes adquiridas asociadas al obturador 2.
Usar misma carpeta. Permite utilizar una sola carpeta de almacenamiento para ambas secuencias de imágenes adquiridas.

III. **Tiempo de Exposición [ms].** Indica el tiempo de exposición en [ms] establecido para cada uno de los obturadores durante el experimento.
Periodo de Apertura [ms]. Indica el periodo de apertura en [ms] establecido para cada uno de los obturadores durante el experimento.
Total de Imágenes a Adquirir. Indica el total de imágenes a adquirir asociadas al obturador 1, al obturador 2 y el total de ambos.
Triggers. Indica si alguno de los Triggers se activará durante el experimento, así como la imagen asociada en la que se activará.
Tiempo total aprox. [s]. Indica el tiempo aproximado que durará la sesión de adquisición asociada al obturador 1, al obturador 2 y la duración aproximada total del experimento.
Iniciar. Inicia el experimento.
Cancelar. Cancela el experimento y regresa a la pantalla de **Adquisición**.

6.3.6 Ventana de Experimento

La ventana de experimento es accesible desde el panel frontal de la subrutina *MainRoutine.vi* y le permite al usuario monitorear en tiempo real los parámetros del experimento así como el modelo biológico. Desde esta ventana el usuario puede realizar las siguientes tareas:

- Visualizar en tiempo real el modelo biológico durante la sesión experimental.
- Visualizar la hora de inicio del experimento.
- Visualizar la hora actual.
- Visualizar la ruta de trabajo raíz en donde se almacenarán las secuencias de imágenes, el reporte escrito y el archivo de registro de la sesión experimental.

- Monitorear los siguientes parámetros fundamentales del experimento: Total de imágenes adquiridas y por adquirir, tiempo de exposición y periodo de apertura tanto del obturador 1 como del obturador 2.
- Visualizar el uso de los TRIGGERS durante el experimento.
- Visualizar el tiempo transcurrido del experimento.
- Abortar el experimento en curso.

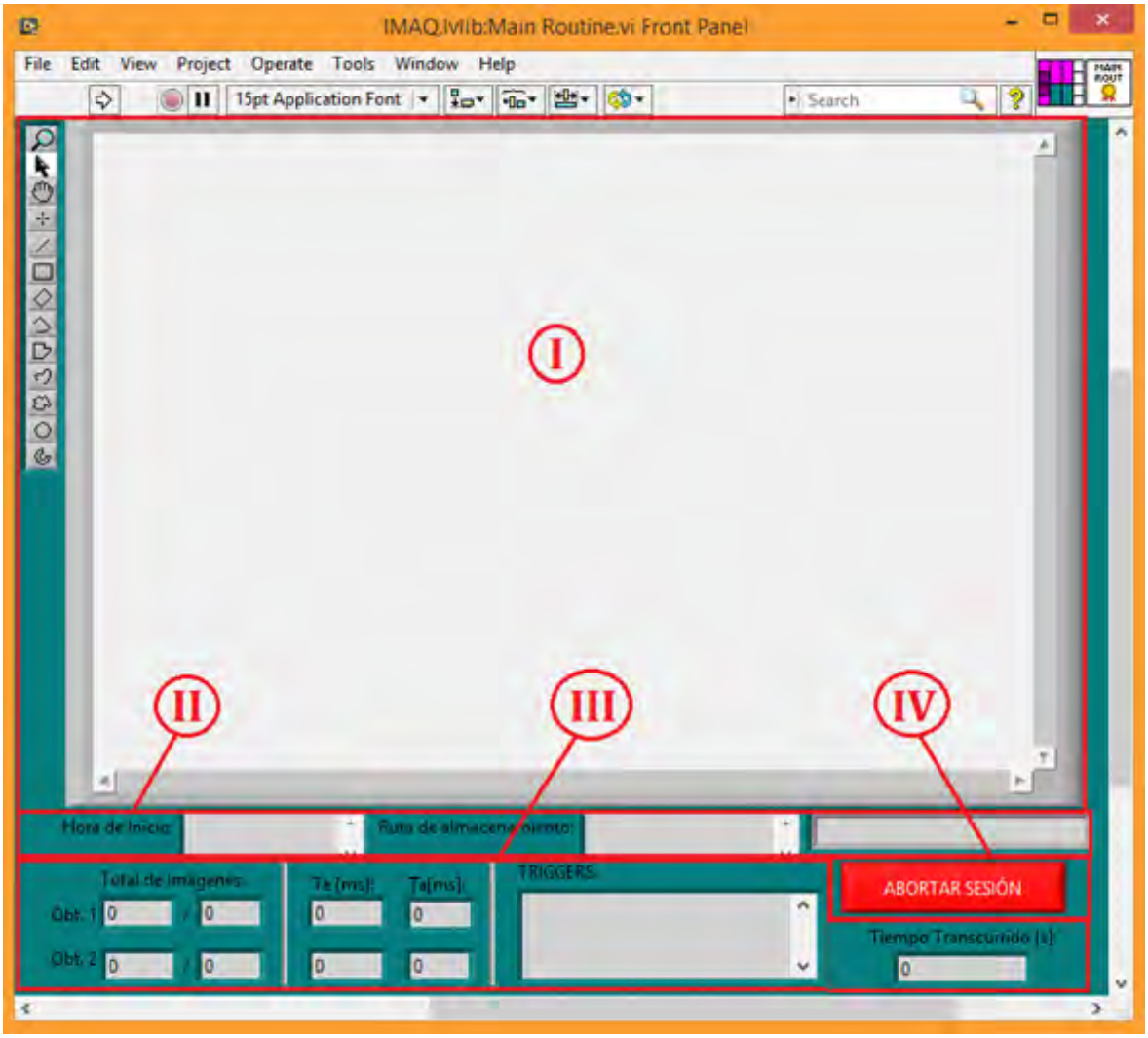


Figura 6.62 Interfaz de usuario de la Ventana de Experimento.

I. **Visualizador de imágenes.** Permite visualizar en tiempo real el modelo biológico durante el transcurso del experimento. El visualizador tiene una paleta de comandos para crear ROI's y realizar acercamientos.

II. **Hora de Inicio.** Indica la hora de inicio del experimento.
Ruta de almacenamiento. Indica la ruta de trabajo raíz del experimento.
Hora actual. Indica la hora actual.

III. **Total de Imágenes.** Indica el número total de imágenes adquiridas hasta ahora y el número total de imágenes a adquirir asociadas tanto al obturador 1 como al obturador 2.

Te [ms]. Indica el tiempo de exposición del obturador 1 y obturador 2 en milisegundos.
Ta [ms]. Indica el periodo de apertura del obturador 1 y el obturador 2 en milisegundos.
TRIGGERS. Indica si se utilizarán Triggers durante el experimento y en qué imagen o imágenes.
Tiempo Transcurrido [s]. Indica el tiempo transcurrido hasta ahora desde que se inició el experimento en segundos.

IV. **Abortar Sesión.** Permite abortar la sesión experimental, descartando todas las imágenes adquiridas hasta el momento.

6.3.7 Cierre del Software

Ya sea que se esté operando el software de control desde el panel de adquisición o en el panel de visualización, la interfaz de usuario cuenta con un comando que finaliza su ejecución. El control de cierre finaliza las comunicaciones físicas con los instrumentos conectados a la PC por medio de los puertos USB, termina la sesión de adquisición y vacía de la memoria RAM cualquier variable almacenada. Finalmente, el comando no solamente finaliza la ejecución del programa principal *HMV.vi*, sino que también cierra cualquier ejecución en segundo plano y la aplicación LabVIEW a través de la función *QuitLabVIEW.vi*.

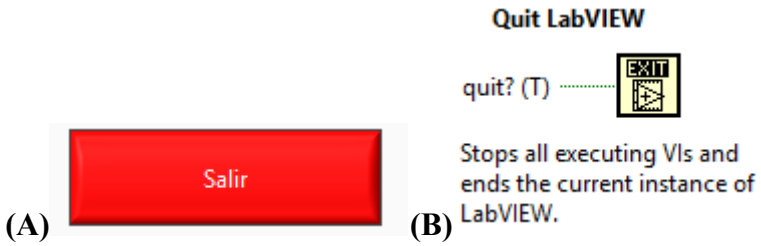


Figura 6.63 (A) Comando de cierre de software presente en el panel de Visualización y el panel de Adquisición. (B) Función QuitLabVIEW.vi que finaliza cualquier .vi ejecutándose así como la instancia actual de LabVIEW.

Capítulo 7

Resultados

El presente capítulo describe los resultados obtenidos al momento de ejecutar el software de control FIMAQ y probar las cuatro funciones principales mencionadas en la **Sección 1.1** a través de la configuración y puesta en marcha de sesiones experimentales. Estas funciones son:

- Control automático de obturadores electrónicos,
- Adquisición de imágenes digitales,
- Gestión de datos y archivos informáticos, y
- Despliegue de resultados.

El capítulo finaliza con la discusión y análisis sobre el alcance y desempeño de las funciones mencionadas anteriormente.

7.1 Pruebas

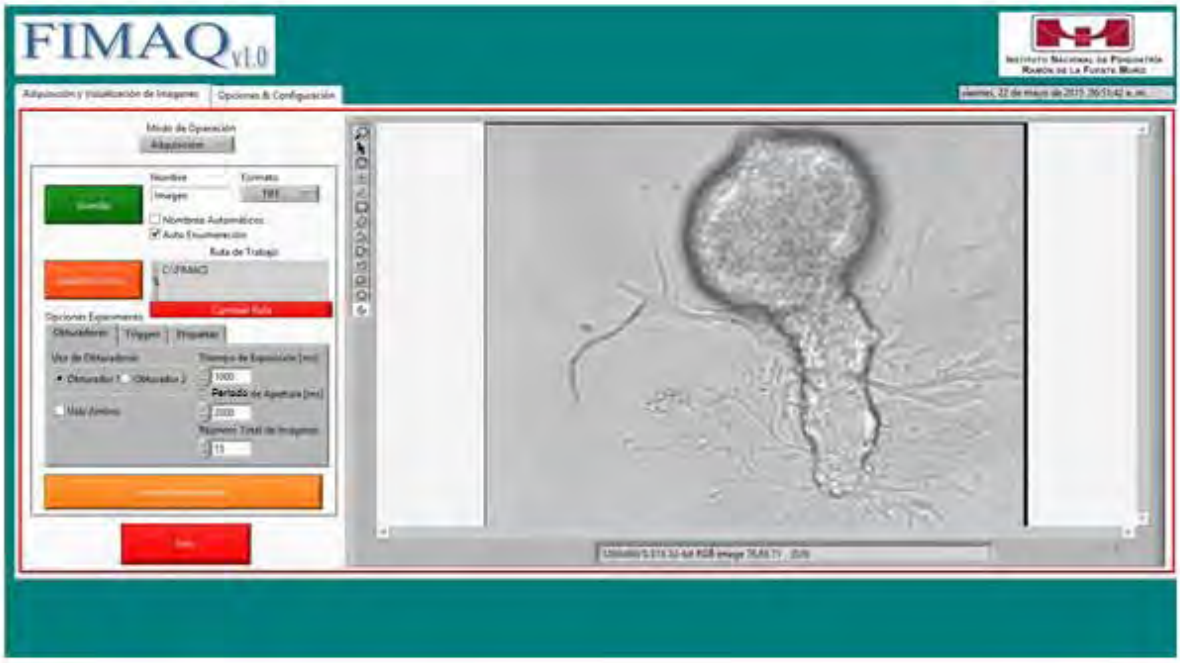
Cada una de las pruebas de laboratorio se presenta y describe de la siguiente manera: configuración utilizada en el experimento, acciones realizadas sobre la muestra biológica, resultados y archivos obtenidos del experimento, e imágenes demostrativas de la prueba de laboratorio.

7.1.1 Prueba de Laboratorio #1

Las especificaciones de la prueba de laboratorio #1 a la que fue sometido el software de control FIMAQ para analizar el desempeño en las cuatro funciones principales mencionadas en la **Sección 1.1** son las siguientes:

- Uso de obturadores: Obturador 1
- Tiempo de Exposición: 1000 [ms]
- Periodo de Apertura: 2000 [ms]
- Total de Imágenes adquiridas: 15
- Trigger 0 ACTIVADO en imagen 7
  Trigger 5 ACTIVADO en imagen 15
  Trigger 1-2-3-4 DESACTIVADOS
- Formato de las imágenes: .TIFF
- Tiempo estimado del experimento: 30 [s]

Como muestra biológica se utilizó una neurona viva aislada del sistema nervioso central de una sanguijuela *Hirudo medicinalis*. Durante la prueba de laboratorio, la muestra se sometió a constantes cambios de enfoque óptico.



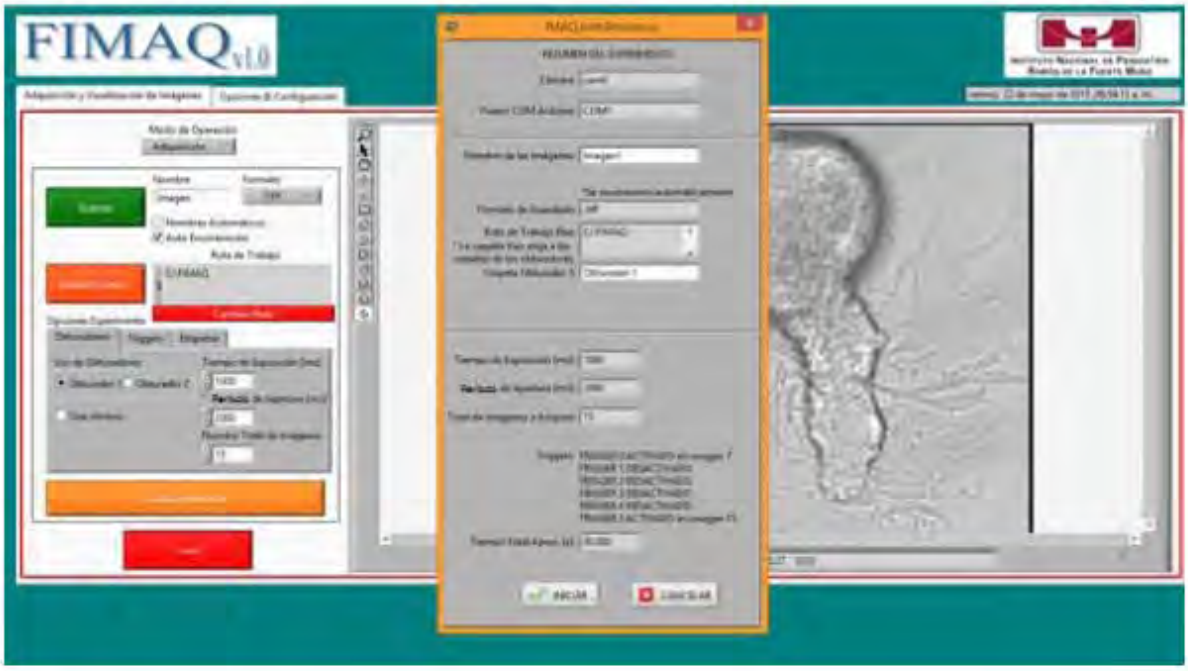
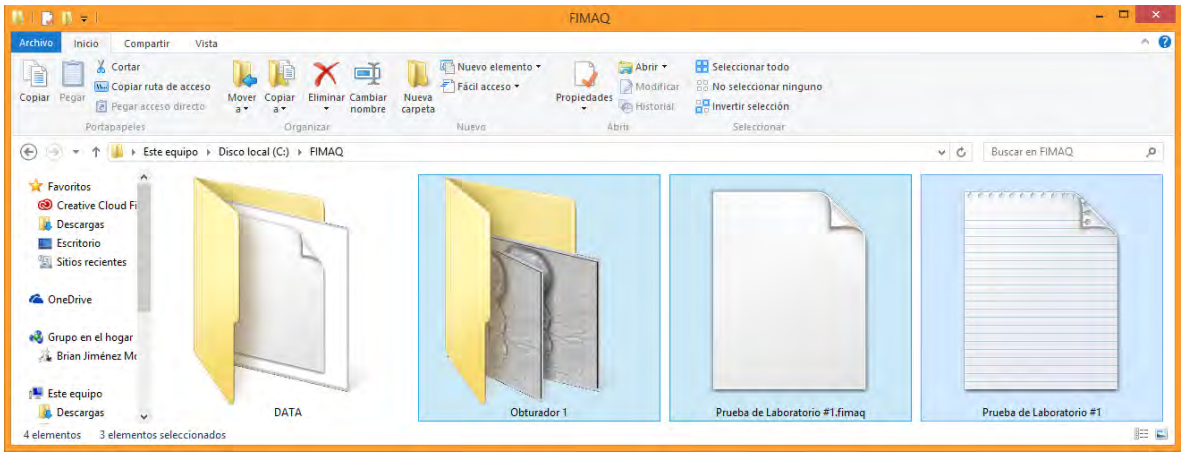


Figura 7.1 Imágenes de la Prueba de Laboratorio #1

El tiempo final transcurrido durante el experimento fue de 30 [s]. Los archivos generados por el experimento fueron los siguientes:

- Archivo de registro: *Prueba de Laboratorio #1.fimaq* (750 KB)
- Reporte escrito: *Prueba de Laboratorio #1.txt* (2 KB)
- Secuencia de 15 imágenes formato *.tiff* en *C:\FIMAQ\Obturador 1* (745 KB)
- Total de espacio en disco duro: 1.497 MB



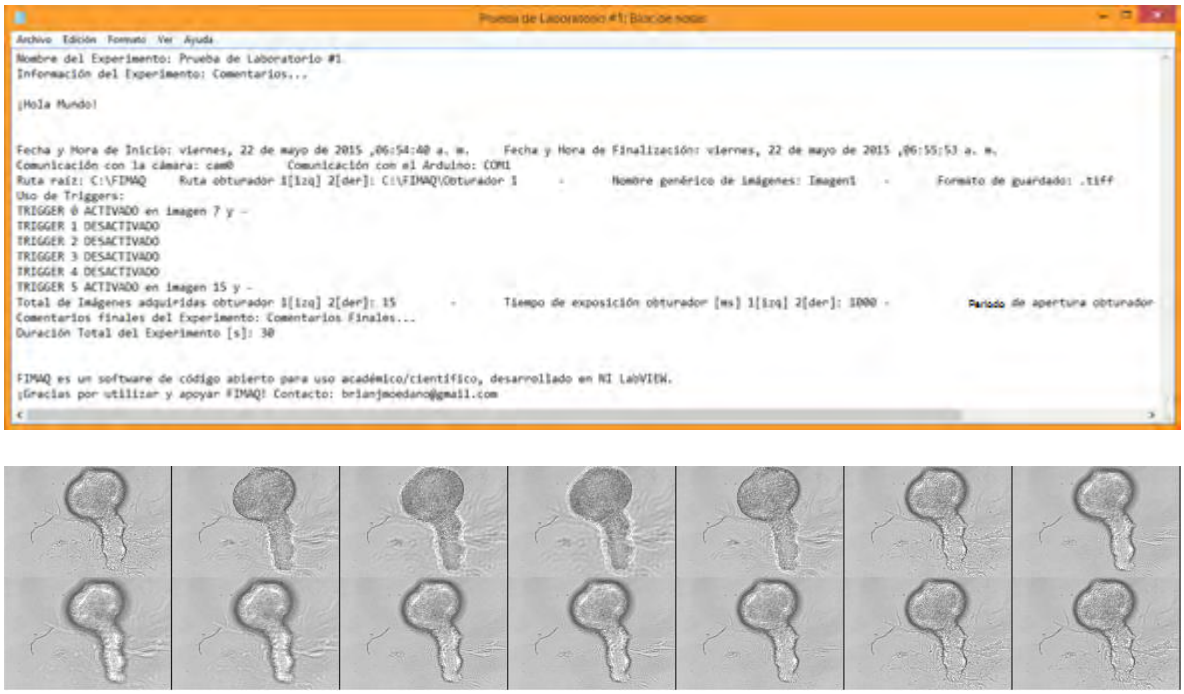


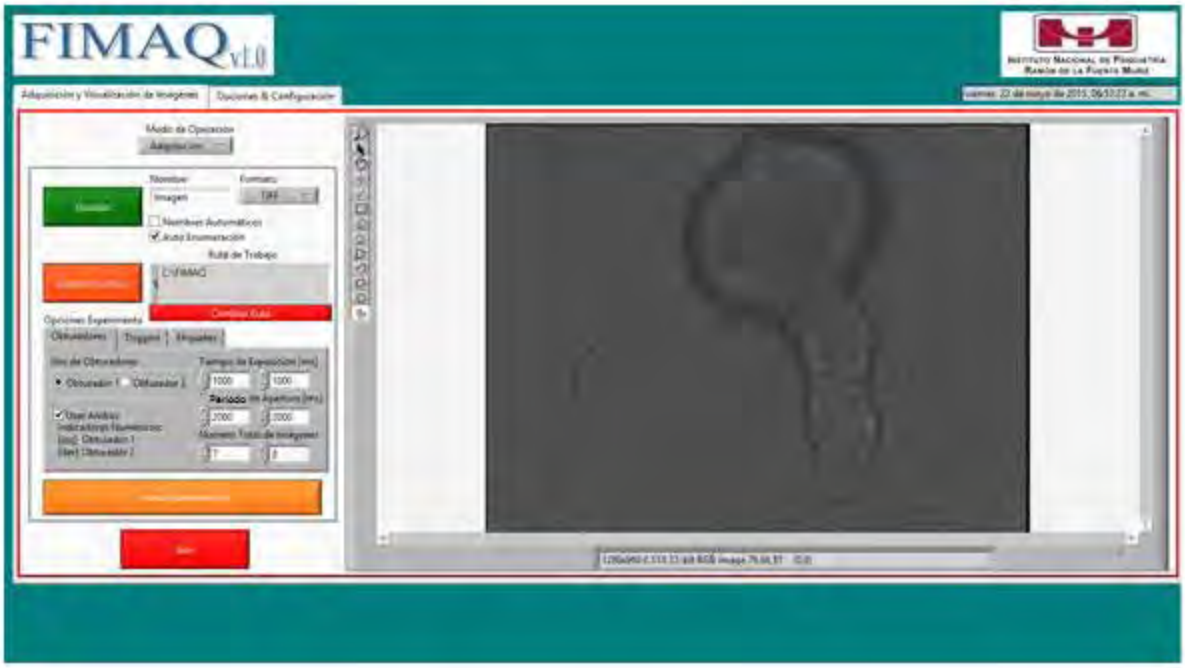
Figura 7.2 Archivos generados por la sesión experimental de la Prueba de Laboratorio #1

7.1.2 Prueba de Laboratorio #2

Las especificaciones de la prueba de laboratorio #2 a la que fue sometido el software de control FIMAQ para analizar el desempeño en las cuatro funciones principales mencionadas en la **Sección 1.1** son las siguientes:

- Uso de obturadores: Obturador 1 [inicia] y Obturador 2
- Tiempo de Exposición (Obturador 1): 1000 [ms]
  Tiempo de Exposición (Obturador 2): 1000 [ms]
- Periodo de Apertura (Obturador 1): 2000 [ms]
  Periodo de Apertura (Obturador 2): 2000 [ms]
- Total de Imágenes adquiridas (Obturador 1): 7
  Total de Imágenes adquiridas (Obturador 2): 8
  Total de Imágenes adquiridas (Total): 15
- Trigger 1 ACTIVADO en imagen 7 y –
  Trigger 2 ACTIVADO en imagen - y 7
  Trigger 3 ACTIVADO en imagen 5 y 5
  Trigger 0-4-5 DESACTIVADOS
- Formato de las imágenes: .TIFF
- Tiempo estimado del experimento (Obturador 1): 14 [s]
  Tiempo estimado del experimento (Obturador 2): 16 [s]
  Tiempo estimado del experimento (Total): 30 [s]

Como muestra biológica se utilizó una neurona viva aislada del sistema nervioso central de una sanguijuela *Hirudo medicinalis*. Durante la prueba de laboratorio la muestra se sometió a constantes cambios de iluminación y contraste óptico.



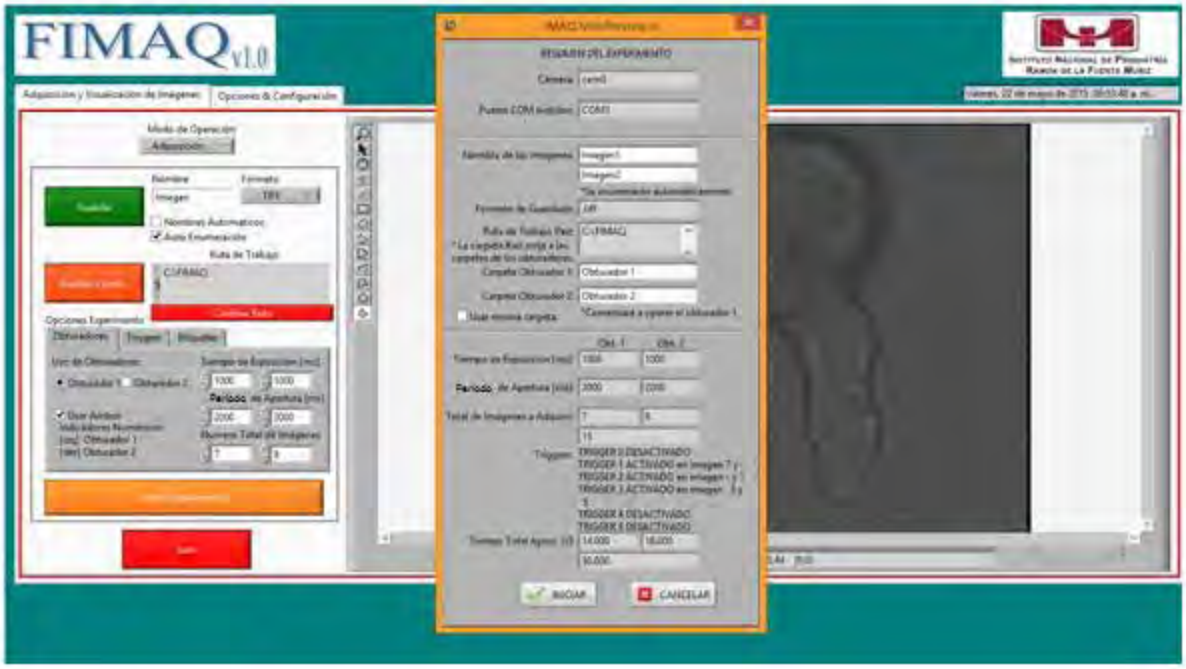


Figura 7.3 Imágenes de la Prueba de Laboratorio #2

El tiempo final transcurrido durante el experimento fue de 16 [s]. Los archivos generados por el experimento fueron los siguientes:

- Archivo de registro: *Prueba de Laboratorio #2.fimaq* (760 KB)
- Reporte escrito: *Prueba de Laboratorio #2.txt* (2 KB)
- Secuencia de 7 imágenes formato *.tiff* en *C:\FIMAQ\Obturador 1* (404 KB)
- Secuencia de 8 imágenes formato *.tiff* en *C:\FIMAQ\Obturador 2* (430 KB)
- Total de espacio en disco duro: 1.596 MB

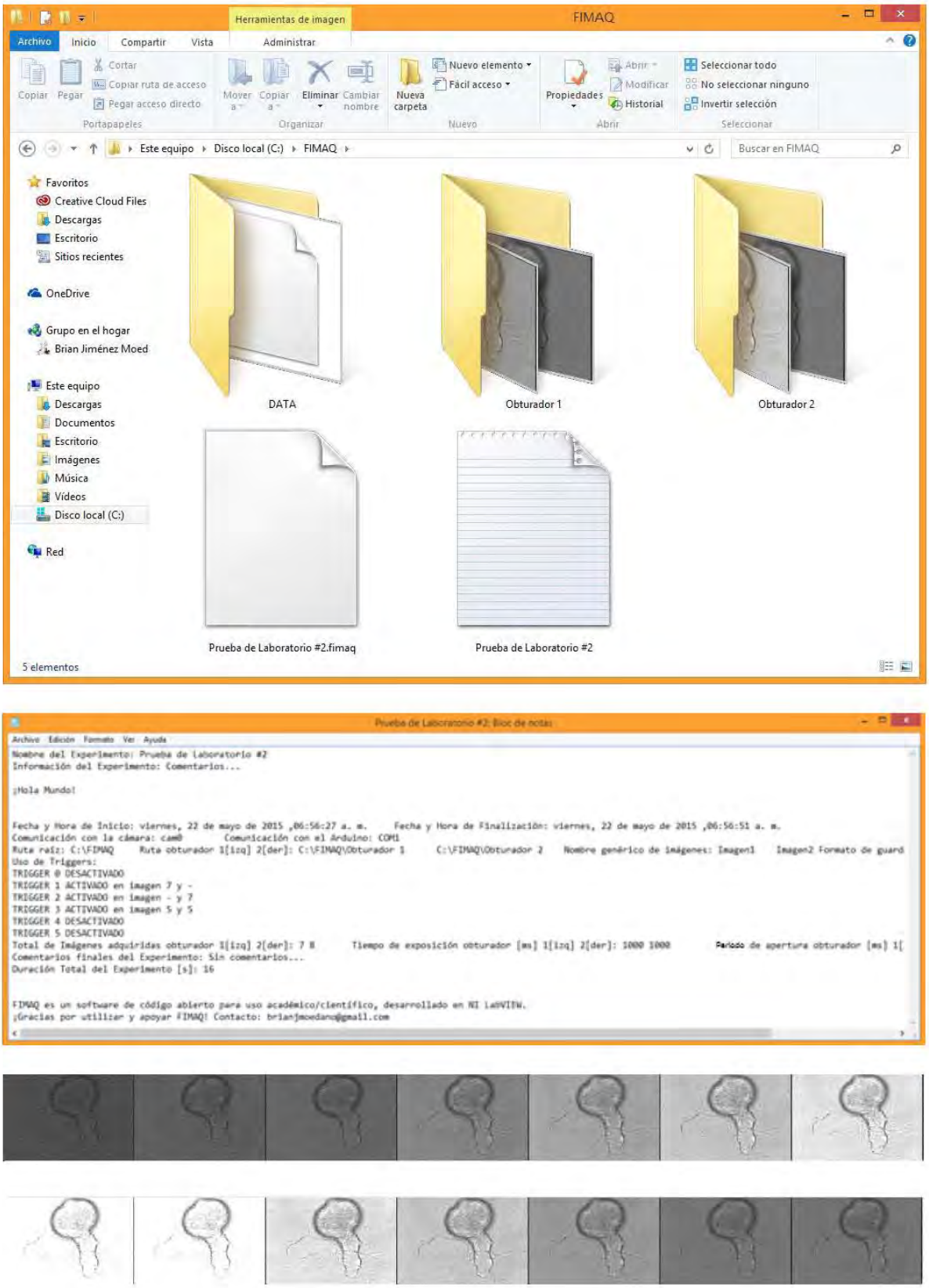


Figura 7.4 Archivos generados por la sesión experimental de la Prueba de Laboratorio #2

7.2 Discusión de Resultados

En la prueba de laboratorio #1 la duración estimada del experimento coincidió con la duración real, mientras que en la prueba de laboratorio #2 la duración estimada del experimento no coincidió con la duración real. Esto se debe al algoritmo del cálculo del tiempo aproximado del experimento **cuando se utilizan los dos obturadores**; ya que este se hace considerando un caso ideal en el que el uso de los obturadores es consecutivo (serie), cuando en la práctica esto no es necesariamente así, pudiendo ser que el control de los obturadores sea tanto consecutivo como independiente uno del otro (paralelo). Por lo tanto, mientras el obturador 1 permanecía cerrado y en espera de volver a abrir, el obturador 2 estaba abriendo; resultando en una duración del experimento menor al estimado. Aunque esto no afecta de manera importante la funcionalidad del software, queda abierta la posibilidad de implementar un mejor algoritmo para calcular el tiempo aproximado del experimento cuando se utilizan los dos obturadores. Véase la **Sección 6.2.10** para mayor información sobre el algoritmo del cálculo del tiempo aproximado del experimento, y la sección **Anexo C: Pruebas de Tiempo** para consultar los tiempos mínimos de operación de los obturadores.

El experimento finalizó sin errores de ejecución por lo que se deduce que en ningún momento se presentaron errores de comunicación con el microcontrolador Arduino UNO, lo que indica que el control de los obturadores resultó adecuado y funcional tanto en tiempos de ejecución como en operación, así como en el uso de Triggers durante la sesión. Todas las señales digitales de control se midieron utilizando multímetros, lo cuales registraron el potencial de los pines digitales de salida del microcontrolador Arduino UNO, verificándose que estos se activaran en la imagen correspondiente y se desactivaran posteriormente.

La adquisición de las imágenes de microscopía resultó en imágenes de baja resolución y un mínimo de pérdidas de calidad al momento de convertirlas en cadenas de datos *string*. Aunque la baja resolución se debe principalmente a la cámara digital CCD (véase **Sección 5.6**), al comparar las imágenes procesadas por FIMAQ con las obtenidas originalmente por la cámara, estas presentaban una pequeña perdida en la calidad, la cual no es importante, pasando de un peso original aproximado de 34 KB a uno de 29 KB. Se desconoce si la pérdida de calidad se produce al momento de convertir la imagen a cadena binaria, o al momento de su reconstrucción; así como también se desconoce en donde repercute exactamente esta pérdida en la calidad, en la imagen ya reconstruida.

Los hilos de programación paralelos (contadores de tiempo e imágenes adquiridas), no interfirieron en la ejecución de la rutina principal del programa (adquisición y procesamiento de imágenes), ni la retrasaron de algún modo, pudiéndose comprobar esto último en los tiempos finales de duración del experimento. Véase la **Sección 6.2.14** para mayor información sobre el algoritmo de almacenamiento y guardado de secuencias de imágenes, y la sección **Anexo C: Pruebas de Tiempo** para consultar los tiempos mínimos de adquisición y procesamiento de imágenes.

Como se observa en la **Figura 7.2** y **Figura 7.4** el experimento arrojó en ambas pruebas los tres archivos esperados: un **archivo de registro** con extensión .fimaq el cual contiene los parámetros del experimento e imágenes en formato *string* del experimento, leíbles para el

propio software FIMAQ; un **reporte escrito** en formato .txt con los parámetros, comentarios y resultados del experimento; y las **secuencias de imágenes** en formato (.tiff, .jpeg, .bmp, .png) según se haya especificado por el usuario. Véase la **Sección 6.2.12** para mayor información sobre el archivo de registro, reporte escrito y secuencias de imágenes.

Por último las ventanas de visualización cumplieron sus respectivas tareas al presentar imágenes de microscopía en tiempo real al momento de ejecutar los experimentos (esto sin interferir en el tiempo de ejecución ni en las funciones del experimento); y como se muestra en la **Figura 7.5,** también para la consulta y análisis posterior de las secuencias de imágenes adquiridas.

Figura 7.5 Archivos de registro de la prueba de laboratorio #1 y #2 desplegados en pantalla desde el panel de Visualización del software de control.

Capítulo 8

Conclusiones

La metodología de desarrollo utilizada, dio como resultado un sistema de bajo costo comparado con aquellos sistemas comerciales que realizan funciones de adquisición de imágenes de microscopía. En la **Figura 8.1** se observa una tabla comparativa entre el costo de implementar el software de control a través de la plataforma de desarrollo NI LabVIEW, y el costo de implementarlo con alguna otra solución comercial conocida para el control de sistemas de adquisición de imágenes digitales de microscopía de fluorescencia; esto sin considerar además, el ahorro monetario que implicó haber aprovechado mucho del hardware e instrumentos ya disponibles en el laboratorio de investigación.

El costo inicial de adquirir una licencia de uso para la plataforma de desarrollo NI LabVIEW, se ve compensado con los enormes beneficios que ésta trae a largo plazo; ya que puede ser utilizada para implementar numerosos sistemas de instrumentación y demás aplicaciones informáticas (como bases de datos y redes de trabajo en línea), los cuales resultan fácilmente adaptables y compatibles en prácticamente cualquier computadora moderna y con cualquier instrumento de laboratorio que utilicen los actuales protocolos estandarizados de comunicación.

Figura 8.1 Tabla comparativa del costo de la licencia de uso de algunos software de adquisición de imágenes de microscopía (verde y naranja) e Instrumentación Virtual (azul). Véase la sección ***Referencias Bibliográficas****,* ***Capítulo 8 ítem [8.2] y [8.3]*** *para mayor información sobre características, usos y costos.*

Además del bajo costo de implementación, otra de las virtudes de FIMAQ (el software de control desarrollado en este proyecto de tesis), es su fácil adaptabilidad a cualquier equipo de adquisición de imágenes digitales de microscopía (sea de fluorescencia o no), y a cualquier computadora que cumpla con los requisitos mínimos del sistema (véase **Sección 5.8**); debido a la flexibilidad con la que fue diseñado y programado, y gracias a la portabilidad que ofrece NI LabVIEW para ejecutar aplicaciones e instrumentos virtuales entre una amplia gama de sistemas operativos y hardware de control/adquisición (véase **Sección 2.3.2**).

Finalmente el mantenimiento que FIMAQ pudiera necesitar en el futuro, puede ser fácilmente proporcionado por cualquier profesional o estudiante con conocimientos de programación en LabVIEW; reduciendo así los costos en comparación con el mantenimiento de agencias privadas a sistemas comerciales. Esto también abre la posibilidad de actualizar o ampliar las funciones operativas de FIMAQ según las necesidades del laboratorio de investigación vayan cambiando; a diferencia de un sistema comercial, que por lo general se actualiza según los tiempos y alcances particulares de los desarrolladores del producto, y el soporte técnico se encuentra limitado a la versión y tiempo de garantía del proveedor.

Apoyado en la discusión de resultados presentada en la **Sección 7.2**, el software de control que se desarrolló (**FIMAQ**) cubre las funciones principales de control, adquisición, gestión de recursos informáticos y despliegue de resultados necesarias para concluir que se ha cumplido satisfactoriamente con los alcances del proyecto y sus objetivos propuestos en el **Capítulo 4**.

Referencias Bibliográficas

CAPÍTULO 1. INTRODUCCIÓN

[1.1] Eco, U. (2001). *Cómo se hace una tesis*. Editorial GEDISA, S. A., México.

[1.2] Hernández, R.; Fernández, C.; Baptista, P. (2003). *Metodologías de la Investigación*. Editorial Mc Graw Hill, México.

[1.3] *¿Cómo se escribe una tesis?* http://www.tanialu.co/2012/11/29/como-se-escribe-una-tesis-parte-1/

CAPÍTULO 2. MARCO TEÓRICO

[2.1] Purves, D. (2004). *Neuroscience*. Editorial Sinauer Associates, Inc., U.S.A.

[2.2] Nicholls, J. (2001). *From Neuron to Brain*. Editorial Sinauer Associates, Inc., U.S.A.

[2.3] Blackshaw, S. E.; Nicholls, J. G. (1995). *Neurobiology and Development of the Leech*. Journal of Neurobiology, Vol. 27, No. 3, Editorial John Wiley & Sons, Inc., U.S.A.

[2.4] Price, R. L.; Jerome, W. G. (2011). *Basic Confocal Microscopy*. Editorial Springer, U.S.A.

[2.5] Smiesko, V.; Kovac K. (2004). *Virtual Instrumentation and Distributed Measurement Systems*. Journal of Electrical Engineering, Vol. 55, U.S.A.

[2.6] Rihan M.; Agarwal A. (2010). *Virtual Instrumentation For Bio-Medical Applications*. Proceedings of the 4th National Conference; INDIACom, India.

[2.7] http://www.sciencedaily.com

[2.8] http://www.livescience.com

[2.9] *La Neurona* http://www.psicologia-online.com/ebooks/general/neuronas.htm

[2.10] *Neurotransmisor* http://es.wikipedia.org/wiki/Neurotransmisor

[2.11] *Capítulo 6. Técnicas Especiales de Microscopía*
http://www.medic.ula.ve/histologia/anexos/microscopweb/MONOWEB/capitulo6_5.htm

[2.12] ***LabVIEW*** http://en.wikipedia.org/wiki/LabVIEW

CAPÍTULO 5. DESCRIPCIÓN DEL HARDWARE Y METODOLOGÍA

[5.1] ***Fuente de alimentación de la lámpara de arco (40-200 W), marca Oriel, modelo 68805*****. Manual de usuario y Hoja de especificaciones técnicas**
http://www.artisantg.com/info/PDF__4F7269656C5F36363830355F4D616E75616C.pdf

[5.2] ***Chasis de la lámpara de arco, marca Oriel, modelo 66001*****. Manual de usuario y Hoja de especificaciones técnicas**
http://mmrc.caltech.edu/Oriel/Oriel%20Lamp%20housing%20660XX.pdf

[5.3] ***Bulbo o lámpara de arco de Xenón 150 W, marca Oriel, modelo 6256*****. Manual de usuario y Hoja de especificaciones técnicas**
http://assets.newport.com/webDocuments-EN/images/Light_Sources.pdf

[5.4] http://www.newport.com

[5.5] https://www.chroma.com

[5.6] http://www.omegafilters.com

[5.7] ***Microscopio invertido, marca Nikon, modelo Diaphot-TDM, trinocular de luz transmitida para cultivo de tejidos*****. Manual de usuario y Hoja de especificaciones técnicas**
https://physics.ucsd.edu/neurophysics/Manuals/Nikon/Diaphot-TMD.pdf

[5.8] ***Diaphot Inverted Tissue Culture Microscope***
https://www.microscopyu.com/museum/diaphot.html

[5.9] ***Microcontrolador Arduino UNO R3***
http://www.arduino.cc/en/Main/ArduinoBoardUno

[5.10] ***Microprocesador ATMEGA328P-PU.*** **Hoja de especificaciones técnicas**
http://www.atmel.com/Images/doc8161.pdf

[5.11] ***Microcontrolador Arduino UNO R3.*** **Hoja de diseño y Esquemático**
http://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf

[5.12] ***Obturador electrónico de diafragma de dos hojas, marca Vincent Associates UNIBLITZ, modelo VS25.*** **Manual de usuario y hoja de especificaciones técnicas**
https://www.uniblitz.com/resources_filelibrary/datasheets/vs25datasheet.pdf

[5.13] ***Controlador de obturadores, marca Vincent Associates UNIBLITZ, modelo D122.*** **Manual de usuario y Hoja de especificaciones técnicas**
http://www.uniblitz.com/resources_filelibrary/t132_d122_user_manual_1_3.pdf

[5.14] ***Cámara digital CCD, marca Hamamatsu, modelo C2400-77***

http://www.sao.ru/drabek/CCDP/Hamamatsu/USA/CCD-CAMERAS/CCD-cam/index.html

[5.15] ***Controlador de la cámara digital CCD, marca Hamamatsu, modelo C2400.*** **Manual de usuario y hoja de especificaciones técnicas**
https://physics.ucsd.edu/neurophysics/Manuals/Hamamatsu/HAMAMATSU%20CAMERA%20CONTROL%20UNIT.pdf

[5.16] ***Tarjeta de adquisición de imágenes USB 2.0 AV Grabber, marca Sabrent, modelo USB-ECPT.*** **Manual de usuario y hoja de especificaciones técnicas**
http://downloads.monoprice.com/files/manuals/5616_Manual_100120.pdf

[5.17] http://www.ni.com/labview/esa

[5.18] https://decibel.ni.com/content/groups/labview-interface-for-arduino

CAPÍTULO 6. DESCRIPCIÓN DEL SOFTWARE DE CONTROL

[6.1] ***LabVIEW 2014 Help Manual***
http://digital.ni.com/manuals.nsf/websearch/C40964A57AC3E76A86257CE100677D96

[6.2] Diagramas de Flujo diseñados con *Axure* http://www.axure.com/is/flowchart-software

CAPÍTULO 7. RESULTADOS

[7.1] Capturas de pantalla realizados con *Mirillis*
https://mirillis.com/en/products/action.html

CAPÍTULO 8. CONCLUSIONES

[8.1] ***Discusión y conclusiones*** http://es.slideshare.net/cristiandiazv/redaccin-de-discusin-y-conclusiones

[8.2] Precio *LabVIEW* http://www.ni.com/labview/buy

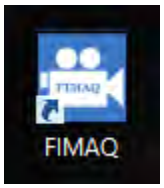
[8.3] Precio comparativo *Image-Pro Plus MC 6.0* y *AxioVision 4.4 w/AIA & w/EF*
http://www.meyerinst.com/html/mediacy/zeiss.htm

Anexos

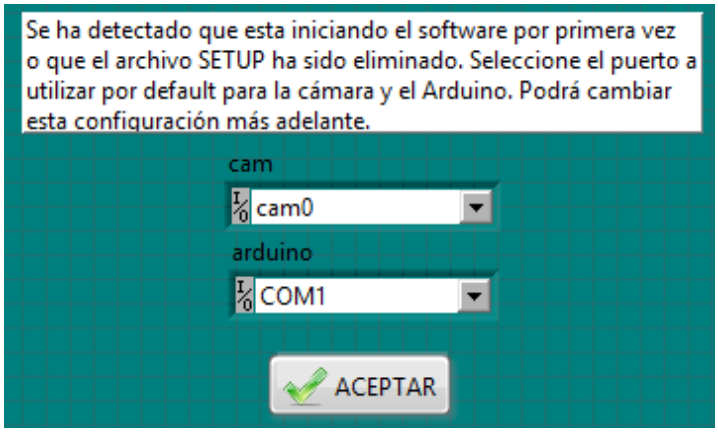
Anexo A: FIMAQ Manual de Usuario

Para consultar en línea y descargar una copia del manual de usuario del software de control **FIMAQ**, acceda a la siguiente liga electrónica:
https://drive.google.com/file/d/0B4yskx0h_hr7Q3N4c2drd2tpNE0/view?usp=sharing

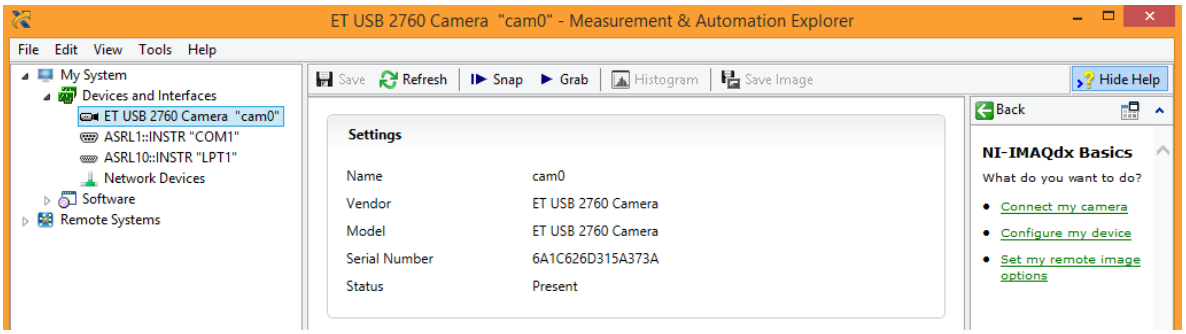
1. Acceda al software de control **FIMAQ** a través del acceso directo disponible en el escritorio de la PC.



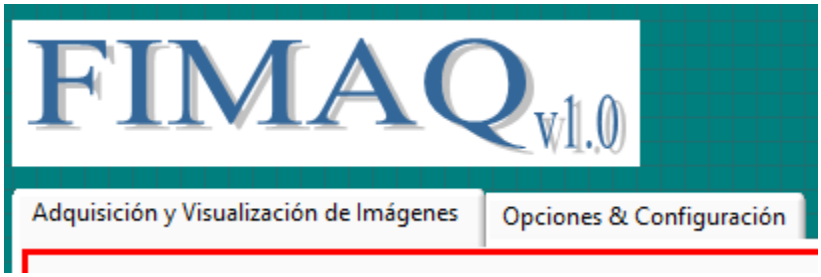
2. Elija los puertos USB que se utilizarán para la comunicación con la tarjeta de adquisición de imágenes AV Grabber y el microcontrolador Arduino UNO respectivamente.



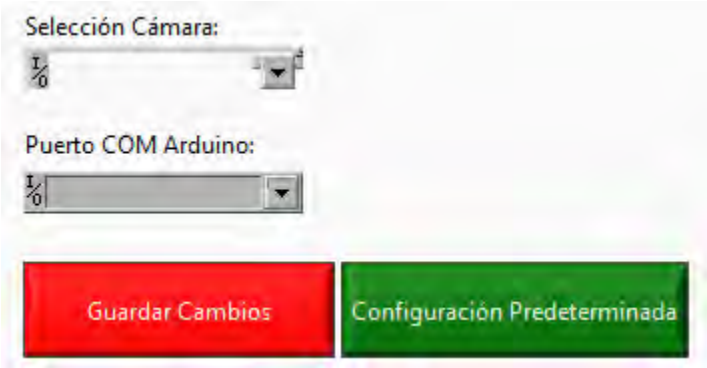
Puede consultar los puertos USB disponibles para cada uno de los instrumentos desde la aplicación NI MAX hallada en el escritorio de la PC.



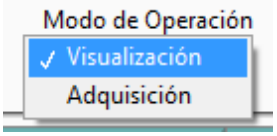
3. Desde la interfaz de usuario principal, utilice las pestañas superiores para navegar entre el modo de Adquisición y Visualización de Imágenes y el modo de Opciones & Configuración.



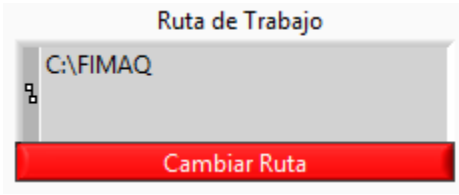
4. Desde el modo de Opciones & Configuración elija nuevos puertos USB para la comunicación con los instrumentos de adquisición y control respectivamente, guarde los cambios y reinicie el software de control para que surjan efecto, o cargue la configuración predeterminada para suprimir cualquier cambio o modificación.



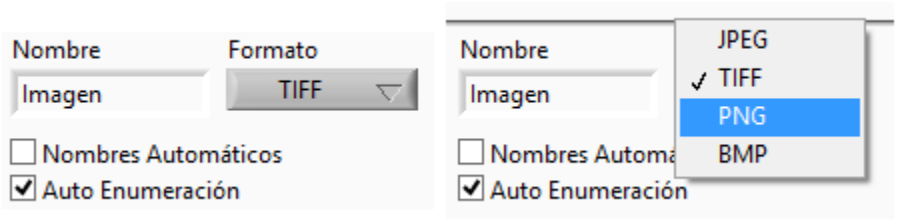
5. Desde el modo de Adquisición y Visualización de Imágenes utilice el anillo Modo de Operación para elegir el modo de operación del software.



6. Desde el modo de **Adquisición** consulte y establezca la ruta de trabajo raíz para el almacenamiento de imágenes y archivos de registro de experimento. Por defecto la ruta de trabajo raíz se establece como la carpeta raíz del software de control.

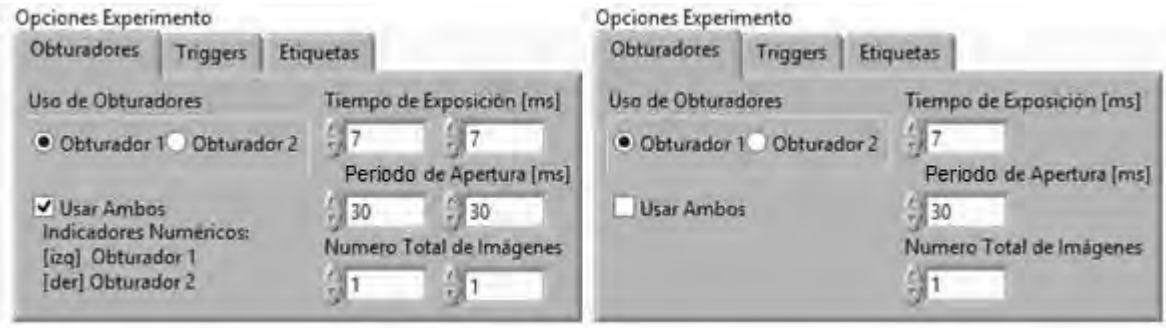


7. Modifique el nombre genérico con el que se almacenarán las imágenes adquiridas, así como el formato de guardado. Habilite la opción de autonombrado y auto enumeración para facilitar el guardado de imágenes.

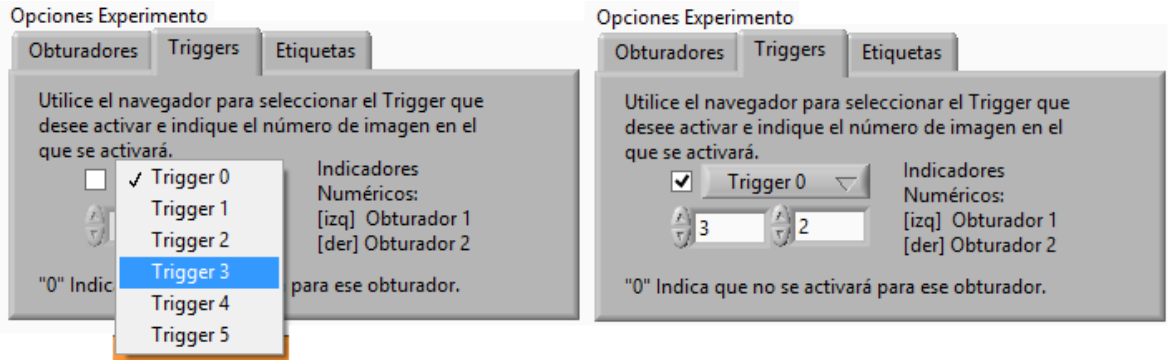


8. Guarde imágenes directamente en la ruta de trabajo Raíz con el comando *Guardar* o utilice el explorador de archivos con el comando *Guardar Como…* para almacenar las imágenes en alguna ruta específica.

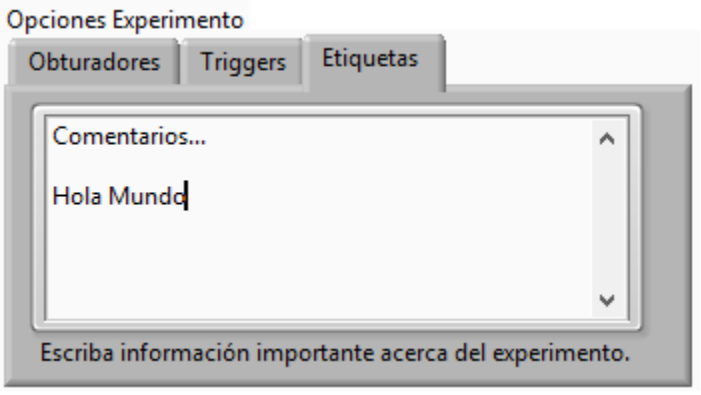
9. Configure el experimento eligiendo usar un solo o ambos obturadores durante el experimento. En caso de elegir usar un solo obturador, decida cual se utilizará, en caso de elegir usar ambos obturadores, decida cual comenzará a operar. El tiempo de exposición representa el tiempo que el obturador permanecerá abierto exponiendo al modelo biológico a la luz de excitación. El periodo de apertura representa el tiempo que necesita transcurrir para que el obturador abra y cierre. Establezca estos tiempos junto con el número total de imágenes a adquirir asociados al o los obturadores a utilizar.



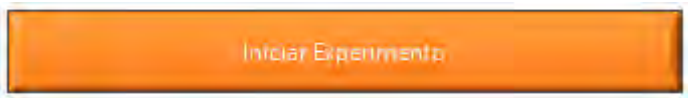
10. Active alguno de los 6 Triggers disponibles a utilizar durante el experimento. Cada Trigger tiene que estar asociado a un número de imagen en el cual se activará. Puede elegir activar todos en un mismo experimento.



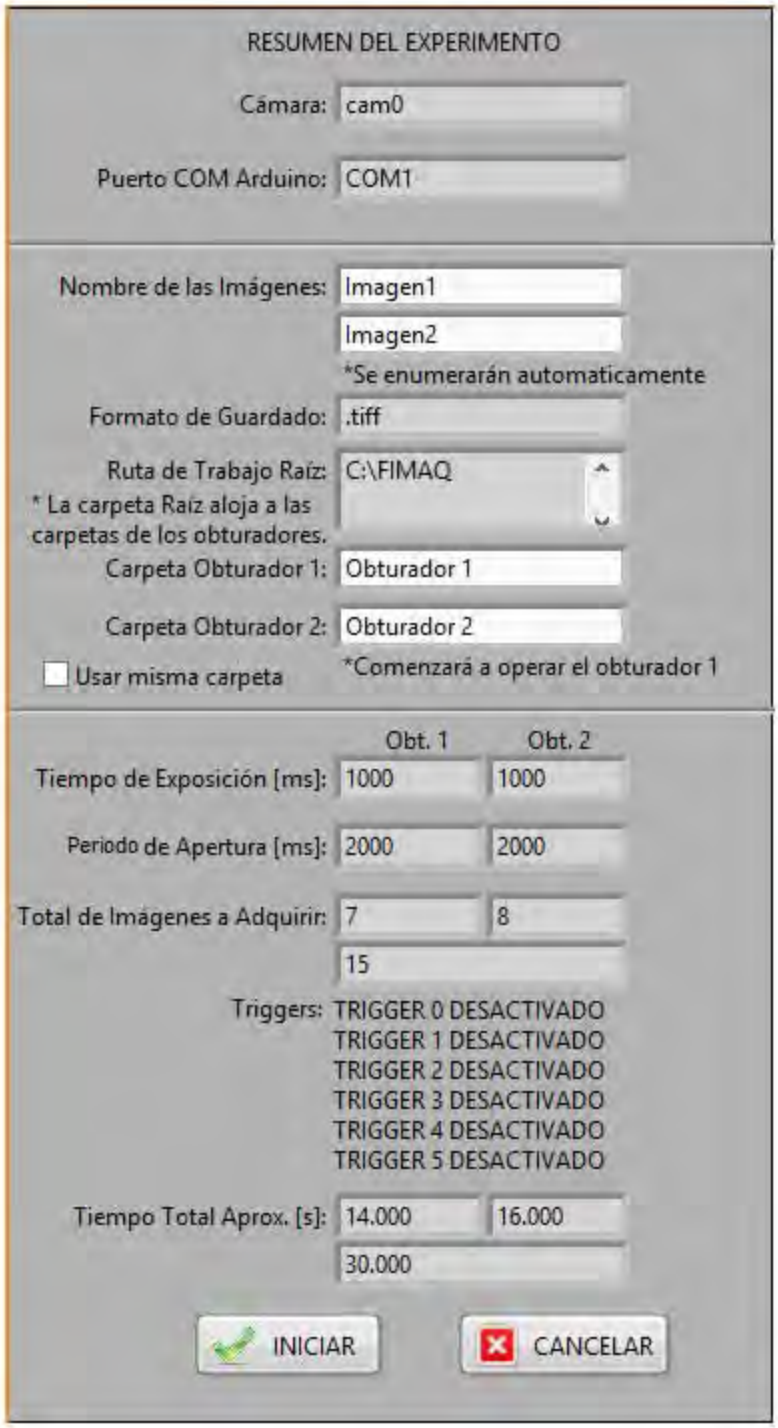
11. Coloque información relevante sobre el experimento en la sección *etiquetas* de las opciones de configuración del experimento.



12. Inicie el experimento.

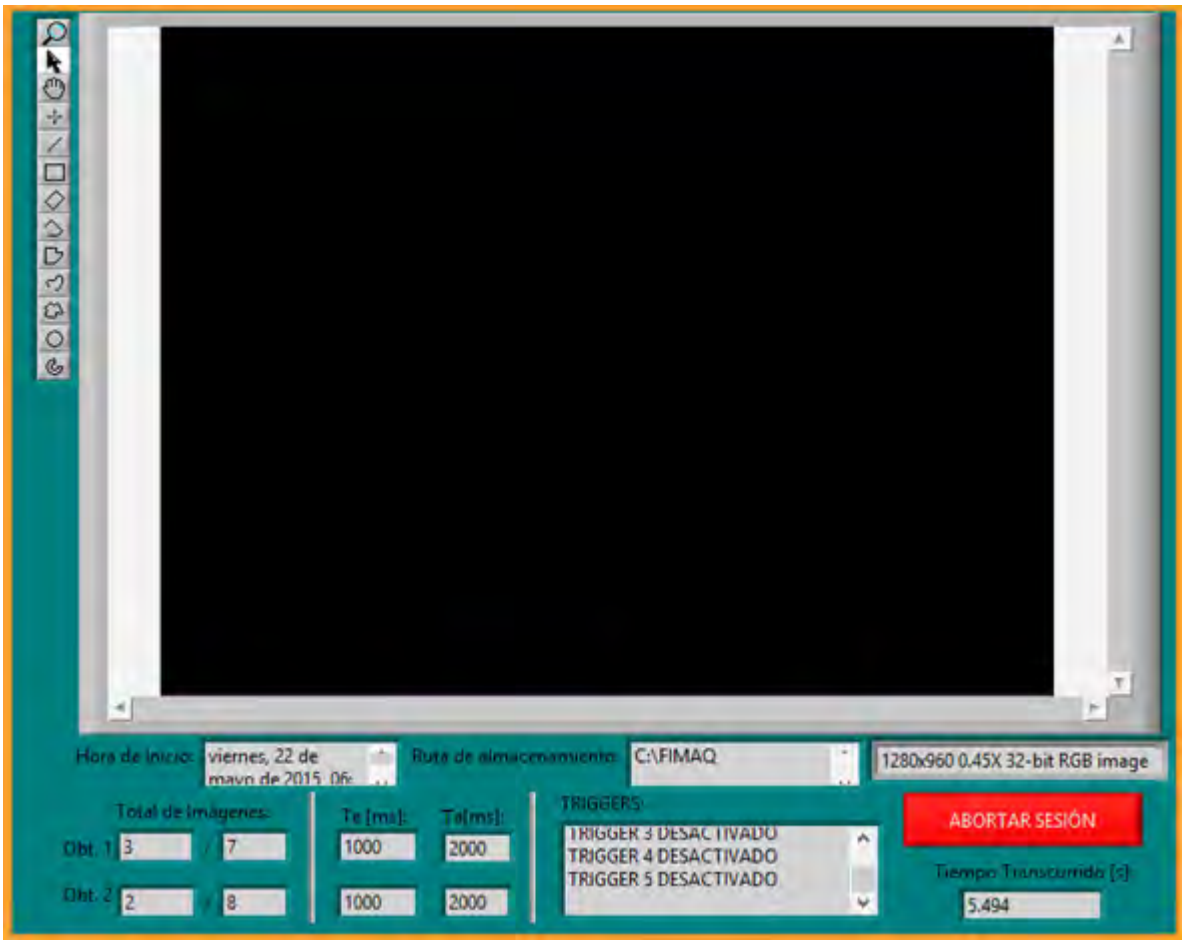


13. Verifique las opciones, configuración e información relacionada con el experimento en la ventana de resumen. Continúe con el experimento o cancele para realizar los cambios pertinentes.

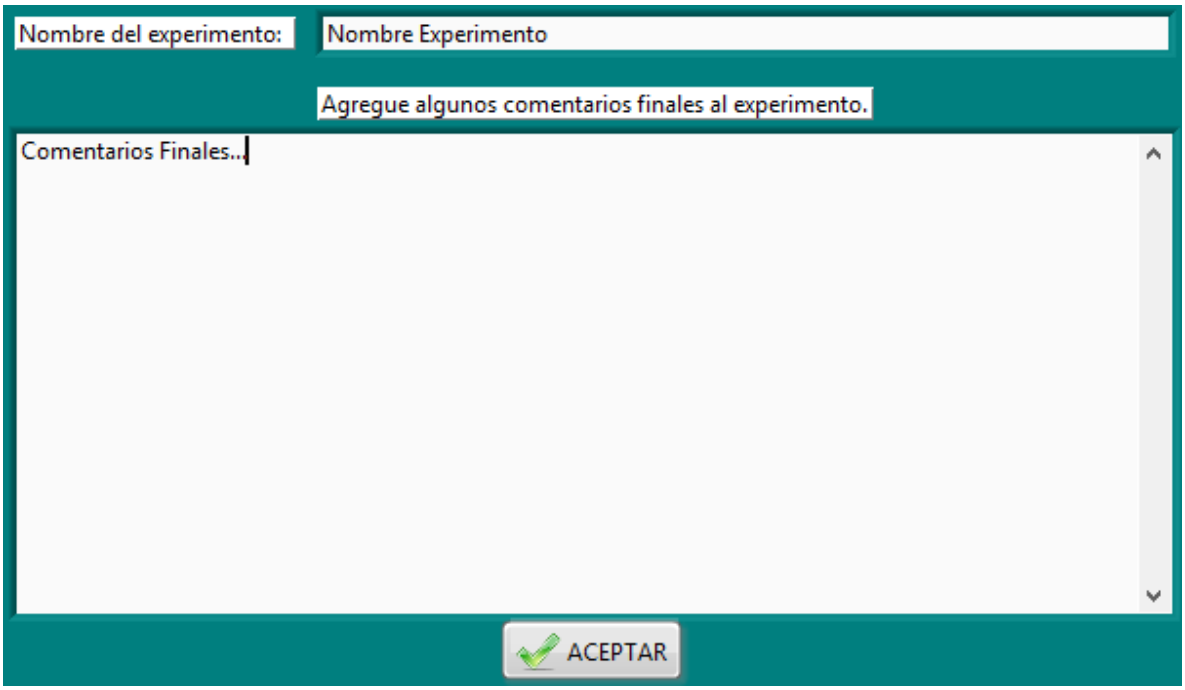


14. Durante el experimento, puede monitorear el estado del modelo biológico en tiempo real desde la ventana de experimento. También puede observar el tiempo transcurrido y los parámetros del experimento relevantes como total de imágenes

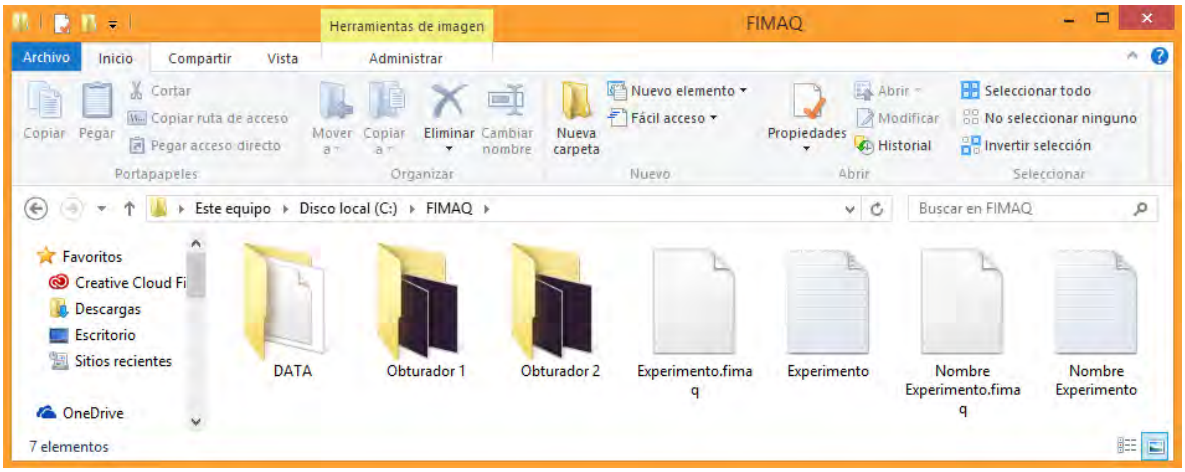
adquiridas y por adquirir, tiempos de exposición, periodo de apertura y uso de Triggers. En cualquier momento puede abortar el experimento (las imágenes adquiridas al momento de abortar se descartarán).



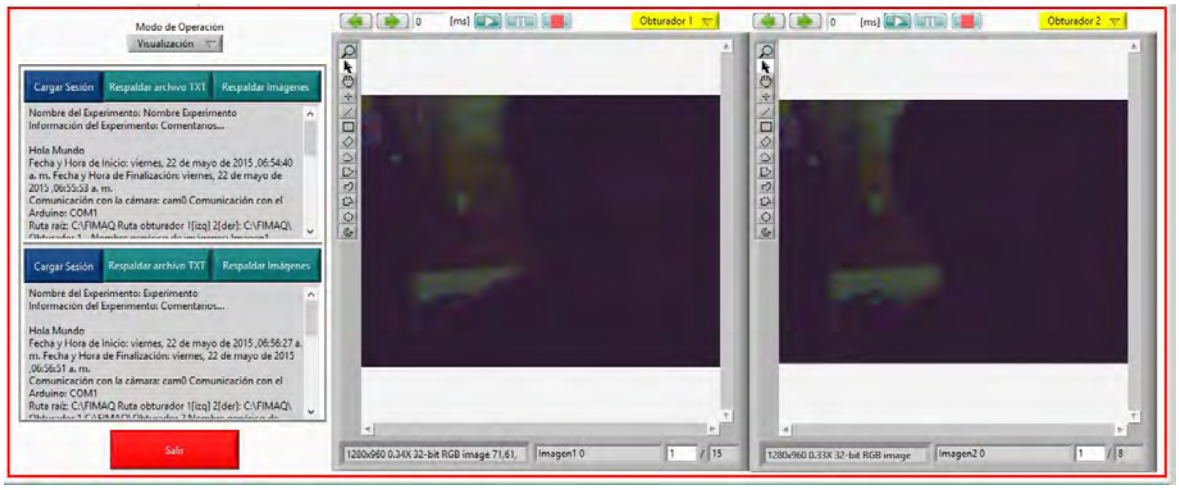
15. Finalmente nombre el archivo de registro que se creará y adjunte algunos comentarios finales al reporte escrito del experimento.



16. Localice el archivo de registro junto con las imágenes adquiridas y el reporte escrito del experimento en la ruta de trabajo raíz.

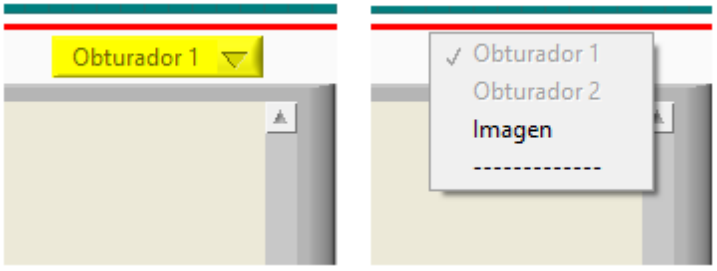


Visualice y respalde en cualquier momento el contenido del archivo de registro desde el modo Visualización.



17. Desde el modo de Visualización cargue sesiones experimentales almacenadas en archivos de registro *.fimaq* con el control *Cargar Sesión*. Utilice el control *Respaldar archivo TXT* para guardar una copia del reporte escrito de la sesión experimental cargada y el control *Respaldar Imágenes* para guardar una copia de las secuencias de imágenes de la sesión experimental cargada; estos controles solamente estarán disponibles al momento de cargar una sesión experimental exitosamente.

18. Visualice imágenes almacenadas en la PC utilizando el control *Imagen*. Limpie el visualizador utilizando el control ------. Visualice imágenes almacenadas en archivos de registro con el control *Obturador 1* (para visualizar la secuencia de imágenes asociada al obturador 1) y el control *Obturador 2* (para visualizar la secuencia de imágenes asociada al obturador 2); estos dos últimos controles solamente estarán disponibles al momento de cargar una sesión experimental exitosamente.



19. Navegue por las imágenes cargadas utilizando los controles de desplazamiento o directamente el indicador numérico de la imagen. Utilice los controles *Play, Pause, Stop* para visualizar una pequeña animación de todas las imágenes cargadas.

20. Finalice el software de control **FIMAQ**.

Anexo B: FIMAQ Código Fuente

Debido a la naturaleza gráfica del lenguaje de programación (G) que utiliza LabVIEW para la creación de instrumentos virtuales, sumado a la extensión de los archivos de programación que se utilizaron para la creación del software de control **FIMAQ**, resulta complicado documentar por escrito la totalidad del código fuente, por lo que se proporciona una liga electrónica en donde puede descargar la totalidad del código fuente así como otros archivos y librerías importantes que integran el software de control **FIMAQ**.

Código fuente del software de control FIMAQ
https://drive.google.com/file/d/0B4yskx0h_hr7dlBPdWFuT25jTUU/view?usp=sharing

La liga electrónica tiene como fin proporcionar al lector la documentación completa del software de control desarrollado en este proyecto de tesis para su consulta. Aunque no protegido oficialmente por ninguna licencia nacional o internacional de derechos de autor, **FIMAQ** se considera por su autor como un software de código abierto y licencia pública común, por lo que cualquier uso del código fuente ajeno a la de su consulta se considera permisible dentro del marco legal anteriormente establecido, acreditando responsabilidad y participación a quien corresponda.

Anexo C: Pruebas de Tiempo

Las siguientes pruebas de tiempo tienen como propósito presentar los tiempos de operación mínimo tanto de los obturadores electrónicos como de la adquisición de imágenes digitales, siendo estos fundamentalmente importantes en la ejecución de las sesiones experimentales de adquisición dentro del software de control, estableciendo los tiempos mínimos de exposición (*te* [ms]) y los tiempos mínimos de apertura (*Ta* [ms]) que el usuario establece dentro de la interfaz de usuario del software de control (véase **Sección 6.3.4**). Finalmente se presenta un diagrama con los tiempos de operación mínimos de una sesión experimental estándar.

Todos los .vi's aquí presentados se encuentran disponibles para su consulta dentro del código fuente del software de control, el cual se puede adquirir desde la liga electrónica proporcionada en el **Anexo B: FIMAQ Código Fuente**.

- **Pruebas de tiempos para la operación de los obturadores electrónicos.**

Existen dos medidas de tiempo importantes que determinan el tiempo de operación mínimo de los obturadores electrónicos:

1. El tiempo mínimo que tarda el software de control en enviar la señal de apertura/cierra al microcontrolador Arduino UNO.
2. El tiempo mínimo que tardan los obturadores físicamente en abrir/cerrar el diafragma de doble hoja.

Para obtener el tiempo mínimo que tarda el software de control en enviar la señal de apertura/cierre al microcontrolador Arduino UNO, se utilizó la subrutina *ShutterOCTime.vi* la cual fue diseñada para medir el tiempo que el software de control tarda en ejecutar la función *DigitalWritePort.vi* encargada de enviar las señales digitales de control al microcontrolador Arduino UNO a través de la conexión serial establecida por medio del puerto USB de la PC.

La subrutina comienza configurando la comunicación con el microcontrolador Arduino UNO y estableciendo el pin 8 como salida digital. Utilizando contadores internos, se toma el tiempo que tarda la instancia de LabVIEW en ejecutar la función *DigitalWritePort.vi* tanto para activar como desactivar el pin de salida digital del microcontrolador Arduino UNO, este dato se almacena dentro de una suma numérica interna. El ciclo *while* repite esta medición una y otra vez almacenando cada vez más valores que indican el tiempo de ejecución de la función *DigitalWritePort.vi.* Finalmente, después de un tiempo considerable (en este caso se tomó un tiempo de prueba de 10 minutos aproximadamente), se detiene el ciclo *while* y se promedian todos los valores almacenados para obtener un tiempo mínimo estimado para la ejecución de la función *DigitalWritePort.vi* tanto para activar como para desactivar la señal digital de salida del microcontrolador Arduino UNO. La desviación estándar en cada caso descarta la posibilidad de una dispersión muy alta de los valores reales obtenidos en torno al valor promedio final.

De esta prueba se obtiene que el tiempo mínimo que tarda el software de control en enviar las señales de apertura al microcontrolador Arduino UNO es de:

$$to_A = 7 \text{ [ms]}$$

Y el tiempo mínimo que tarda el software de control en enviar las señales de cierre al microcontrolador Arduino UNO es de:

$$tc_A = 7 \text{ [ms]}$$

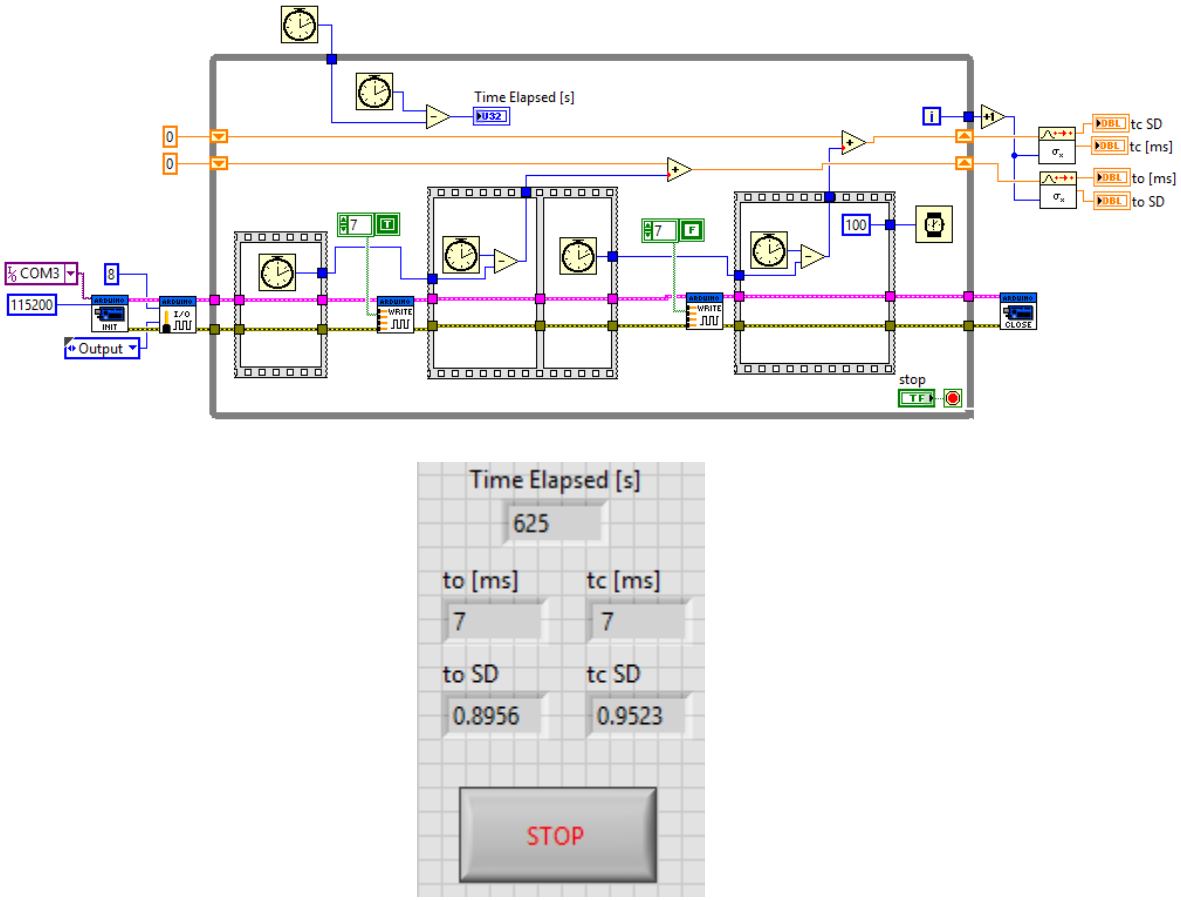


Figura C.1 La subrutina ShutterOCTime.vi mide el tiempo en que el software de control, es decir, cualquier instancia de ejecución de LabVIEW, tarda en ejecutar la función DigitalWritePort.vi la cual envía los comandos necesarios al microcontrolador Arduino UNO para que éste active o desactive las señales de control a través de sus puertos digitales.

Para obtener el tiempo mínimo que tardan los obturadores físicamente en abrir/cerrar el diafragma de doble hoja, se utilizó la hoja de especificaciones técnicas proporcionadas por el distribuidor oficial del instrumento, el cual se puede consultar en la **Sección 5.5** específicamente en la **Figura 5.16**, o directamente en la sección **Referencias Bibliográficas, Capítulo 5 ítem [5.12]**.

De la hoja de especificaciones se obtiene que el tiempo mínimo que tardan los obturadores en recibir la señal eléctrica (TTL) de control y abrir el diafragma es de:

$$\mathbf{to_S = 6\ [ms]}$$

Y el tiempo mínimo que tardan los obturadores electrónicos en dejar de recibir la señal eléctrica (TTL) de control y cerrar el diafragma es de:

$$\mathbf{tc_S = 5\ [ms]}$$

- **Pruebas de tiempo para la adquisición de imágenes digitales.**

Existen dos medidas de tiempo importantes que determinan el tiempo de adquisición mínimo de las imágenes digitales:

1. El tiempo mínimo que tarda el software de control en adquirir una imagen a través de la tarjeta de adquisición de imágenes AV Grabber.
2. El tiempo mínimo que tarda en convertir una imagen en una cadena binaria *string*.

Para obtener el tiempo mínimo que tarda el software de control en adquirir una imagen a través de la tarjeta de adquisición de imágenes AV Grabber, se utilizó la subrutina *ImageAFTime.vi* la cual fue diseñada para medir el tiempo que el software de control tarda en ejecutar tanto la función *IMAQdxGrab.vi* como la función *IMAQFlattenImageToString.vi* encargadas de adquirir imágenes digitales a través de la tarjeta AV Grabber conectada a la PC por medio de un puerto USB y de convertir imágenes en cadenas binarias *string* respectivamente.

La subrutina comienza configurando la comunicación y adquisición de imágenes con la tarjeta de adquisición de imágenes AV Grabber y creando un espacio en memoria para alojar las imágenes adquiridas. Utilizando contadores internos, se toma el tiempo que tarda la instancia de LabVIEW en ejecutar la función *IMAQdxGrab.vi* y la función *IMAQFlattenImageToString.vi* tanto para adquirir una imagen como para convertirla en una cadena *string*, estos datos se almacenan dentro de una suma numérica interna. El ciclo *while* repite esta medición una y otra vez almacenando cada vez más valores que indican el tiempo de ejecución de las funciones *IMAQdxGrab.vi* y *IMAQFlattenImageToString.vi.* Finalmente, después de un tiempo considerable (en este caso se tomó un tiempo de prueba de 10 minutos aproximadamente), se detiene el ciclo *while* y se promedian todos los valores almacenados para obtener un tiempo mínimo estimado para la ejecución de la función *IMAQdxGrab.vi* y *IMAQFlattenImageToString.vi* respectivamente. La desviación estándar en cada caso descartar la posibilidad de una dispersión muy alta de los valores reales obtenidos en torno al valor promedio final.

De esta prueba se obtiene que el tiempo mínimo que tarda el software de control en adquirir una imagen a través de la tarjeta de adquisición de imágenes AV Grabber es de:

$$ti_A = 0 \text{ [ms]}$$

Y el tiempo mínimo que tarda el software de control en convertir una imagen a una cadena binaria *string* es de:

$$ti_F = 10 \text{ [ms]}$$

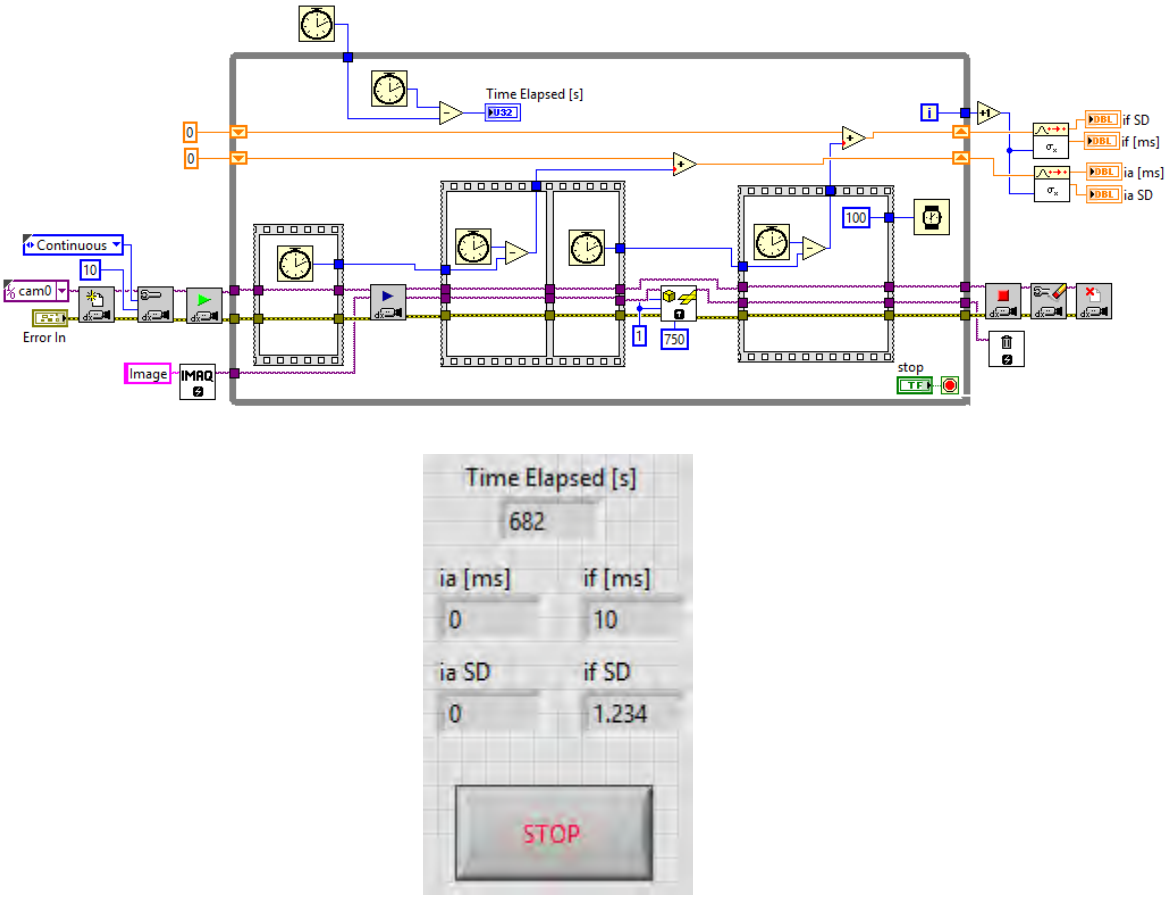


Figura C.2 La subrutina ImageAFTime.vi mide el tiempo en que el software de control, es decir, cualquier instancia de ejecución de LabVIEW, tarda en ejecutar las funciones IMAQdxGrab.vi y IMAQFlattenImageToString.vi las cuales adquieren imágenes a través de la tarjeta AV Grabber y convierten las imágenes en cadenas binarias string respectivamente.

- **Diagrama de tiempos mínimos de una sesión experimental.**

A partir de los tiempos obtenidos con las pruebas anteriores, el diagrama de tiempos mínimos para una sesión experimental estándar es el siguiente:

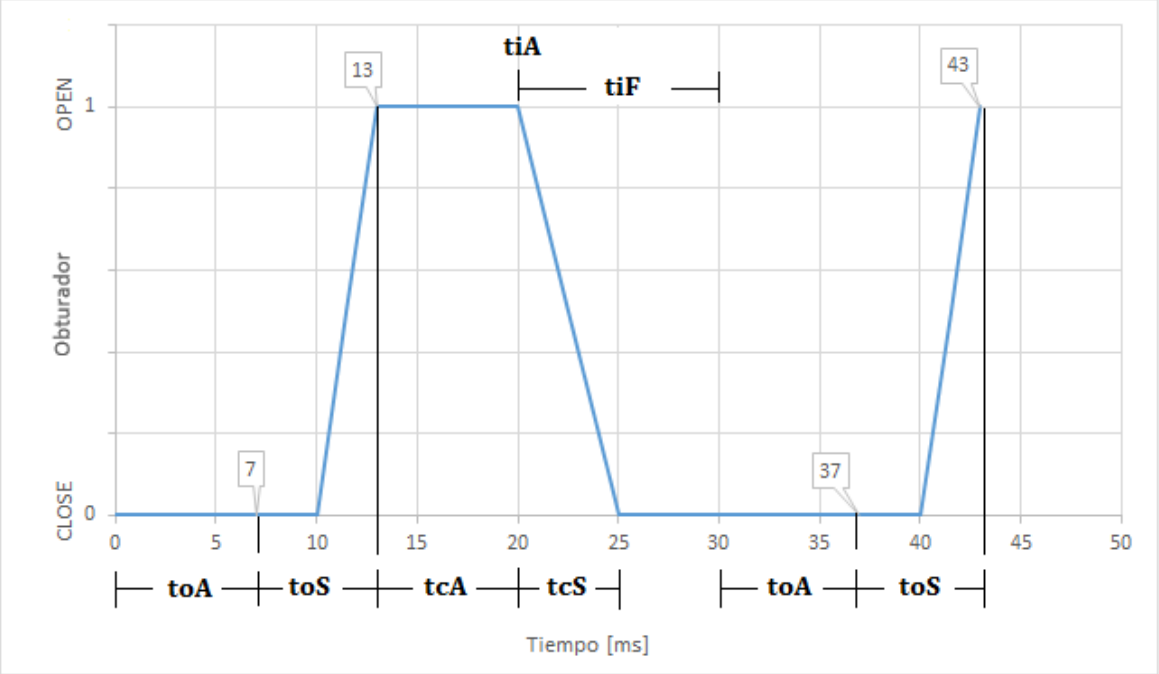


Figura C.3 Diagrama de tiempos mínimos para una sesión experimental estándar.

De este diagrama de tiempos mínimos se observa que:

El tiempo de exposición mínimo es de:

$$\mathbf{te_{min} = 7\ [ms]}$$

El periodo de apertura mínimo es de:

$$\mathbf{Ta_{min} = 30\ [ms]}$$