

CONFIGURACION Y OBTENCION DE IMAGENES CON CAMARA OV7670 MEDIANTE I2C

Cesar Albeiro Parada Rincón¹, Sebastian Camilo Pira², and Julian Ricardo Corzo Camayo³

¹ceparadar@unal.edu.co

²spira@unal.edu.co

³jcorzoc@unal.edu.co

1 Introducción

En la era digital actual, la captura y procesamiento de imágenes juegan un papel fundamental en una amplia gama de aplicaciones, desde sistemas de seguridad hasta dispositivos médicos y plataformas de entretenimiento. En este contexto, la cámara OV7670, la FPGA Colorlight 75E y el lenguaje de descripción de hardware Verilog son elementos clave que permiten la implementación de sistemas avanzados de adquisición y procesamiento de imágenes.

La cámara OV7670 es conocida por su tamaño compacto y su capacidad para capturar imágenes de alta calidad. Equipada con diversas funciones como control de exposición, balance de blancos y detección de movimiento, la OV7670 se ha convertido en una elección popular para proyectos que requieren capacidades de visión por computadora. Su interfaz digital facilita la integración con sistemas embebidos y plataformas de desarrollo. Por otro lado, la FPGA Colorlight 75E se destaca como una unidad de procesamiento altamente versátil que utiliza la tecnología de puertas programables. Esta FPGA ofrece una plataforma potente para la implementación de algoritmos de procesamiento de imágenes y permite una rápida adaptación a los requisitos específicos del sistema.

En relación de lo anterior tenemos el lenguaje de descripción de hardware Verilog, el cual desempeña un papel crucial en la configuración y programación de la FPGA. Como un lenguaje dedicado para describir circuitos digitales, Verilog permite modelar y simular sistemas complejos antes de su implementación en hardware. Su sintaxis cercana al hardware facilita la representación de circuitos digitales y la definición de comportamientos específicos.

2 Generalidades

2.1 Protocolo I2C

I2C (Inter-Integrated Circuit) es un protocolo de comunicación serial síncrono que permite la transferencia de datos entre dispositivos a través de dos líneas: una para la transmisión de datos (SDA) y otra para la señal de reloj (SCL). Utiliza un enfoque maestro-esclavo, donde un maestro inicia y controla la comunicación con uno o varios esclavos.

Para el desarrollo e implementación del protocolo I2C, se debió hacer un estudio minucioso del protocolo, así como la utilización de sistemas electrónicos de medición como osciloscopio y analizadores de señales digitales, para una correcta visualización y comprobación de éxito.

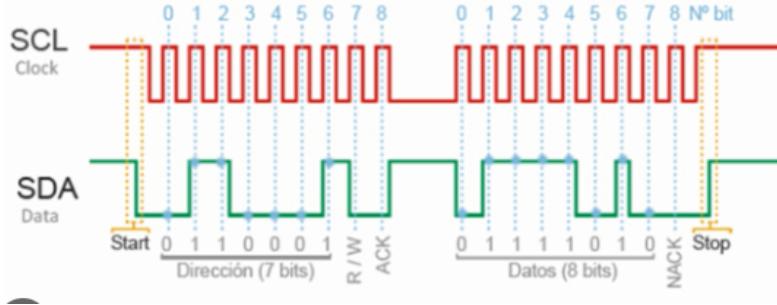


Figure 1: Señales de SDA y SCL para el inicio de comunicación.

En la figura anterior se muestra el funcionamiento general de un protocolo I2C para el inicio de la comunicación hacia un dispositivo esclavo, el cual será configurado mediante estas dos señales.

2.2 Camara OV7670

La cámara de referencia OV7670 es un dispositivo electrónico que permite la captura y procesamiento de imágenes en distintos formatos, la cual puede ser configurada mediante un protocolo I2C. Dicha cámara posee una amplia lista de configuraciones que permite variar desde el formato hasta la resolución de la imagen, además de un sin número de funciones adicionales, que para este proyecto no serán utilizadas ni tenidas en cuenta.

Es de importancia aclarar que la comunicación con este dispositivo se da a partir del correcto direccionamiento, teniendo en cuenta la dirección del mismo, debido a que se cuenta con una señal de ACK, que nos permite verificar si la comunicación se está dando de manera correcta.



Figure 2: Camara OV7670

3 Planteamiento

Para la implementación de este proyecto se tuvieron en cuenta los siguientes apartados.

- Comunicación mediante I2C (SCCB)
- Configuración de la cámara
- Recolección y análisis de datos
- Guardado de datos
- Visualización de datos(Imagen)

3.1 Comunicación mediante I2C

Para la comunicación por I2C se hizo necesario investigar acerca del protocolo, de lo cual se resume lo siguiente:

- Dirección I2C de la Cámara OV7670.
- Configuración del Bus I2C.
- Conexiones Físicas.
- Iniciar y Detener la Comunicación:
- Lectura y Escritura de Datos.

Es preciso aclarar que para este proyecto solo se realizó la escritura de datos al dispositivo, ya que solo se precisaba la configuración de la cámara a nuestras necesidades.

Nombre	Tipo	Función
SIOD	Entrada/Salida	Datos de la interfaz SCCB
SIOC	Entrada	Reloj de la interfaz SCCB
RESET	Entrada	Resetea el sensor cuando se pone a nivel bajo
PWDN	Entrada	Apaga el sensor cuando se pone a nivel alto
HREF	Salida	Señal de referencia horizontal
VSYNC	Salida	Señal de referencia vertical
XCLK	Entrada	Reloj del sistema
PCLK	Salida	Reloj para sincronización de píxel
DATA0	Salida	Bit 0 de la salida de vídeo digital
DATA1	Salida	Bit 1 de la salida de vídeo digital
DATA2	Salida	Bit 2 de la salida de vídeo digital
DATA3	Salida	Bit 3 de la salida de vídeo digital
DATA4	Salida	Bit 4 de la salida de vídeo digital
DATA5	Salida	Bit 5 de la salida de vídeo digital
DATA6	Salida	Bit 6 de la salida de vídeo digital
DATA7	Salida	Bit 7 de la salida de vídeo digital
VCC	Alimentación	Alimentación del sensor a 3.3V
GND	Tierra	Tierra

Figure 3: Pines de conexión de la cámara OV7670

Luego de esto se detalló la necesidad de utilizar una forma específica de transmisión de datos para nuestro dispositivo sugerida y diseñada por el fabricante.

El SCCB (Serial Camera Control Bus) es un bus serie de 3 hilos diseñado por OmniVision. En este caso, el dispositivo con el que trabajamos presenta una versión simplificada de 2 hilos debido a que está instalado en una placa que permite el acceso a menos pines de los realmente disponibles en el chip. Este bus se basa en el modelo maestro/esclavo (master/slave), muy similar al bus I2C, en el que un dispositivo que actúa como maestro se conecta al bus SCCB para controlar al menos un dispositivo esclavo. Los tres hilos del bus se denominan (*SCCBE*), *SIOC* y *SIOD*, de los cuales renombraremos como *SCL* y *SDA* respectivamente, dejando de lado el primero (*SCCBE*)

Partiendo de lo anterior mostraremos el ciclo completo de comunicación del *SCCB*, del cual se basara el estado entre la FPGA y la cámara.

El diagrama completo de la figura 7 se puede encontrar en: <https://github.com/petter5856a/DIGITAL-2/security>.

3.2 Configuración de la cámara

Para la configuración de la cámara fue necesario la consulta de su hoja de datos, ya que es un dispositivo con bastantes características y utilidades.

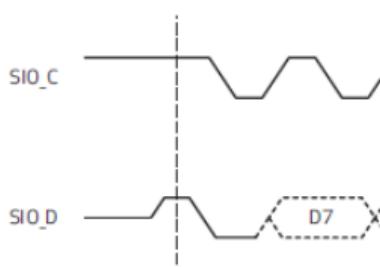


Figure 4: Inicio de la transmisión de datos.

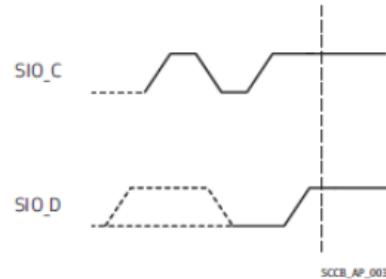


Figure 5: Finalizacición en la transmisión de datos

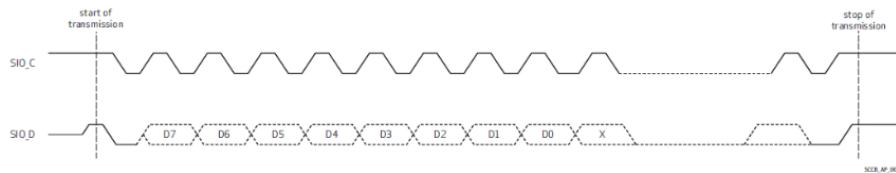


Figure 6: Ciclo completo de comunicación SCCB

La característica principal y con mayor importancia fue la dirección del dispositivo, ya que sin esta sería imposible dicha configuración debido a que al ser un dispositivo esclavo y ser llamado por el protocolo sin una dirección específica podría estar escribiéndose a cualquier otro dispositivo, teniendo en cuenta que se pueden conectar alrededor de 150 dispositivos esclavos a una misma rama de I2C.

Luego de solucionar el problema de la dirección se hizo necesario buscar acerca de los formatos entregados por la cámara, ya que al momento de guardar datos en la RAM de la FPGA podríamos tener problemas con el almacenamiento debido a formatos de gran volumen por lo tanto se buscaron y encontraron los siguientes formatos:

- CIF
- QVGA
- QCIF
- RGB

De los formatos presentados anteriormente se decidió optar por el formato QCIF, ya que poseía la menor resolución, solucionando el problema de almacenamiento, con dicho formato se obtiene una resolución de 176 x 144 píxeles. Por último se realizó la configuración y correcta elección del reloj de sistema, con una frecuencia de $1.92MHz$

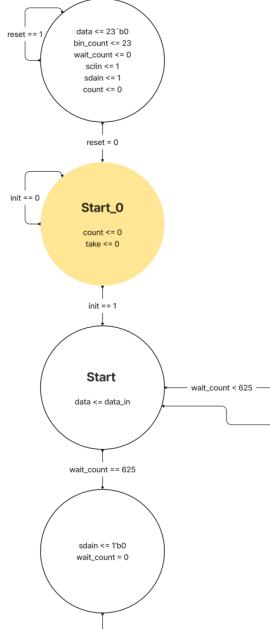


Figure 7: Sección de diagrama del protocolo SCCB.

Por lo tanto se requirió del envío de información mediante la siguiente estructura.

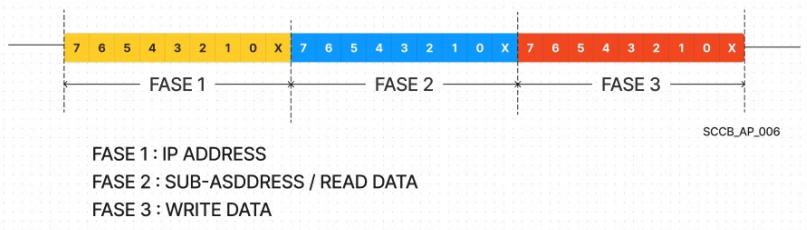


Figure 8: Esquema de transmisión de un byte

3.3 Recolección y análisis de datos

Para la recolección de datos se hizo necesario liberar algunos de los pines de la FPGA para adaptarlos como entradas, a lo cual se procedió a desoldar 3 chips de los 23 que se encontraban soldados por la parte inferior de la tarjeta y habilitándolos mediante la unión entre algunos de sus pines.

Por otra parte se consultó el manual de la tarjeta con el fin de poder asignar de manera correcta los pines recién liberados, para los cuales se asignarían



Figure 9: FPGA Colorlight
75E



Figure 10: Chips re-
movidos y soldados

posteriormente las siguientes entradas:

Pin FPGA	Pin OV7670
C4	SDA
D4	SCL
J4	VSYNC
E4	HREF
J5	PCLK
G3	XCLK
D3	D0
F4	D1
F3	D2
F5	D3
H3	D4
H4	D5
G5	D6
H5	D7
3.3V	RESET
GND	PWDN

Tabla 1. Asignación de pines de entrada a la FPGA

Algo a tener en cuenta es el formato de color que se manejará, ya que en función de esto se tendrá el guardado de los datos de los píxeles de la imagen, por ende se aclara que el formato de color a utilizar será el *RGB565* el cual consiste en el envío de información por los pines de salida de D_0 a D_7 en dos ciclos diferentes, los cuales corresponden a 16 bits, 7 por trama para la composición

de un píxel organizados de la siguiente manera:

Dato 7	Dato 6	Dato 5	Dato 4	Dato 3	Dato 2	Dato 1	Dato 0
R_4	R_3	R_2	R_1	R_0	G_5	G_4	G_3

Table 1: Secuencia de transmisión $RGB - 565$ (Primer ciclo)

Dato 7	Dato 6	Dato 5	Dato 4	Dato 3	Dato 2	Dato 1	Dato 0
G_2	G_1	G_0	B_4	B_3	B_2	B_1	B_0

Table 2: Secuencia de transmisión $RGB - 565$ (Segundo ciclo)

Como se observa en las Tablas 1 y 2, el color de cada píxel viene dado por dos tramas de 8 bits cada una, la cual posee una especificación para cada uno de los tres colores principales, dando espacio para 5 bits del color rojo, 6 del color verde y 5 del azul, siendo el verde el color con más peso de los 3.

Luego de la asignación anterior de pines en el archivo adecuado(SOC.plf), se procedió a la realización del código en assembler para el direccionamiento de memoria necesario.

3.4 Guardado de datos

Luego de la adquisición de los datos provenientes de la cámara y el direccionamiento a la memoria RAM mediante un programa en assembler, se vio la necesidad de investigar como deberían empaquetarse y guardarse los datos, ya que en función de esto dependería la correcta visualización de la imagen tomada, por tanto se debió analizar como se guardaban los bits en una imagen, esto mediante el uso de programas en Python que descomponían una imagen en su archivo más básico, de esta manera se entendería la forma en la que debían guardarse los datos. A continuación se muestra un código sencillo en Python que toma una imagen y la descompone en datos básicos(bits) y los guarda en una variable para ser mostrados.

Listing 1: Código para descomponer imagen

```
import os
from PIL import Image

def image_to_binary(image_path):
    with open(image_path, "rb") as image_file:
        binary_data = image_file.read()
    return binary_data
image_path = os.path.join(os.path.dirname(__file__), "cesar.jpeg")
```

```
binary_data = image_to_binary(image_path)
```



Figure 11: Imagen de prueba

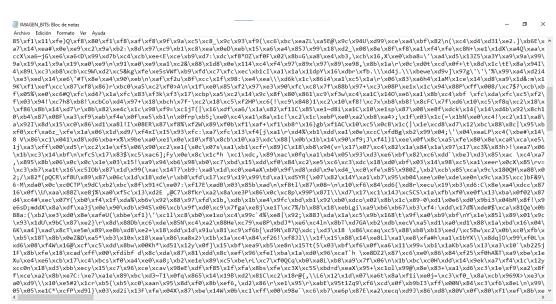


Figure 12: Sección de datos en hexadecimal de la imagen de prueba

Para la lectura de datos se optó por la creación de un segundo periférico el cual controlaría el flujo de datos desde la cámara hacia la FPGA, que posteriormente sería redireccionada hacia la memoria del procesador para su posterior procesamiento.

Para cada uno de los apartados anteriores se realizaron esquemas, diagramas de flujo y casos de uso, los cuales nos brindaron mayor facilidad al momento de implementar instrucciones en código.

A continuación se muestran los esquemas planteados por el grupo para abordar cada uno de los ítems anteriormente mencionados.

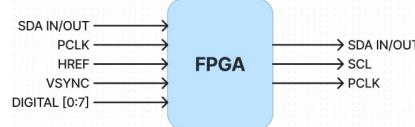


Figure 13: Diagrama de señales para la FPGA

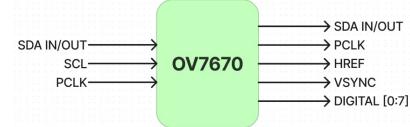


Figure 14: Diagrama de señales para la cámara OV7670

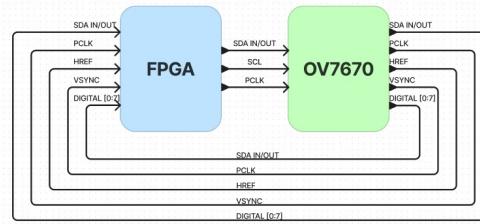


Figure 15: Diagrama de señales entre la FPGA y la cámara OV7670

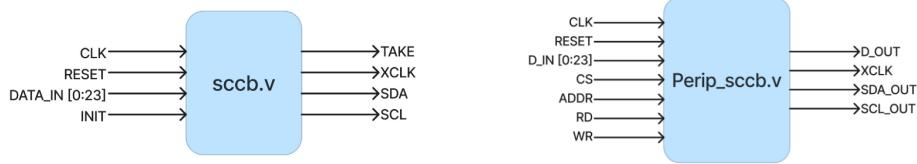


Figure 16: Diagrama de señales para SCCB

Figure 17: Diagrama de señales para el periférico de SCCB

3.5 Visualización de datos(Imagen)

4 Resultados

4.1 Comunicación I2C

Al realizar la construcción y ejecución del código diseñado en Verilog y assembler para la ejecución del protocolo I2C se obtuvieron los siguientes resultados, mostrados a través del osciloscopio.

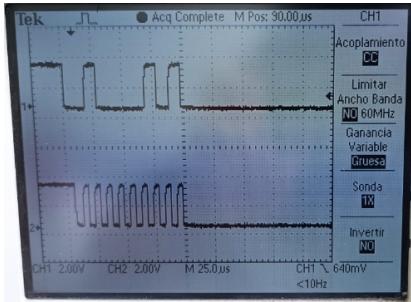


Figure 18: Señales de SDA y SCl con una dirección de camara incorrecta en osciloscopio

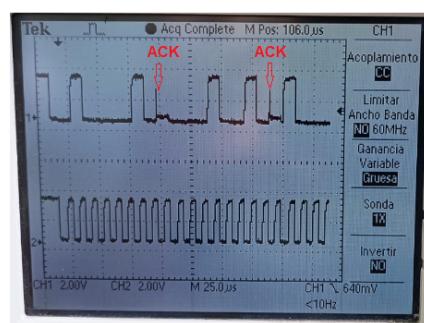


Figure 19: Señales de SDA y SCl con ACK en osciloscopio.

Como se observa en la figura 8, se evidencia de manera clara la correcta ejecución del protocolo I2C, donde nuestro dispositivo esclavo envía un bit de confirmación, la cual nos permite corroborar el correcto direccionamiento y envío de información, ya que como se mencionó en el apartado de configuración, la cámara poseía una dirección específica a la cual se debía comunicar de manera precisa, de lo contrario no sería reconocida por el protocolo.

En esta imagen se observa de manera más claro como la cámara reconoce la dirección que se está enviando mediante el protocolo y devuelve el bit de confirmación.

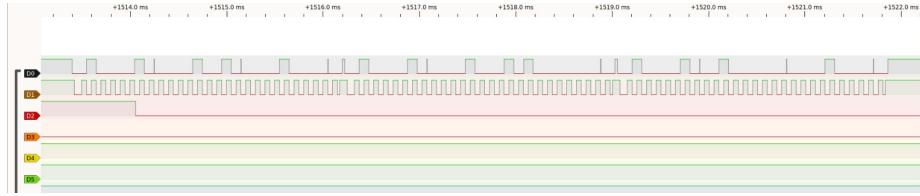


Figure 20: Señales de SDa y SCL con ACK en analizador lógico

4.2 Configuración de la cámara

Al realizar la correcta comunicación mediante el protocolo, procedimos a enviar los datos de configuración de la cámara, teniendo en cuenta lo mencionado en la sección del planteamiento respecto a los formatos, además de la comprobación de la configuración del reloj del sistema, ya que al realizar pruebas con distintas frecuencias, se observó que para valores de frecuencia inferiores a $10MHz$ la configuración fallaba debido a que no existía bits de confirmación, en principio esto se hizo con el fin comprobar a qué frecuencia podría recibir la cámara los bits de comunicación.

A continuación se muestran los resultados gráficos obtenidos haciendo uso de un analizador lógico.

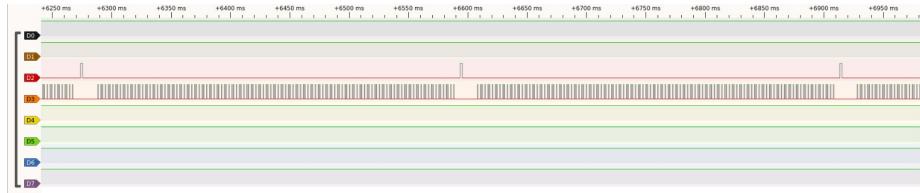


Figure 21: Configuración de Formato a QCIF



Figure 22: Configuración de Formato a VGA

Como se observa en las dos figuras anteriores no es difícil ver que el formato que posee menor cantidad de datos es el QCIF en comparación con el formato VGA, lo cual era de esperarse según lo investigado. Por otro lado se evidencia el funcionamiento de la configuración de formato.

Adicionalmente se muestra como cambia el formato de envío de información por parte de la cámara al realizar el envío de datos de configuración.

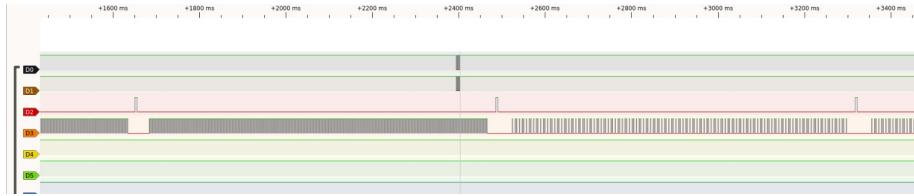


Figure 23: Cambio de la configuración por defecto

En la siguiente figura se observa como a una frecuencia inferior a $420kHz$ la cámara no es capaz de enviar el bit de confirmación, por lo que no es posible realizar la configuración.



Figure 24: Señales de SDA y SCL sin ACK en analizador lógico

4.3 Lectura de Datos

Al realizar el diseño de la lectura de datos de obtuvo es siguiente mapa de memoria y lectura de pixel.

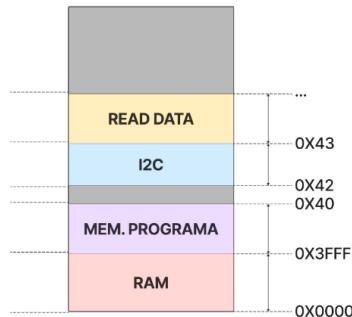


Figure 25: Mapa de memoria

Por otra parte se presenta el esquema general de comunicación del proyecto.

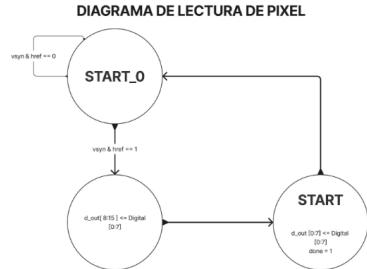


Figure 26: Lectura de pixel

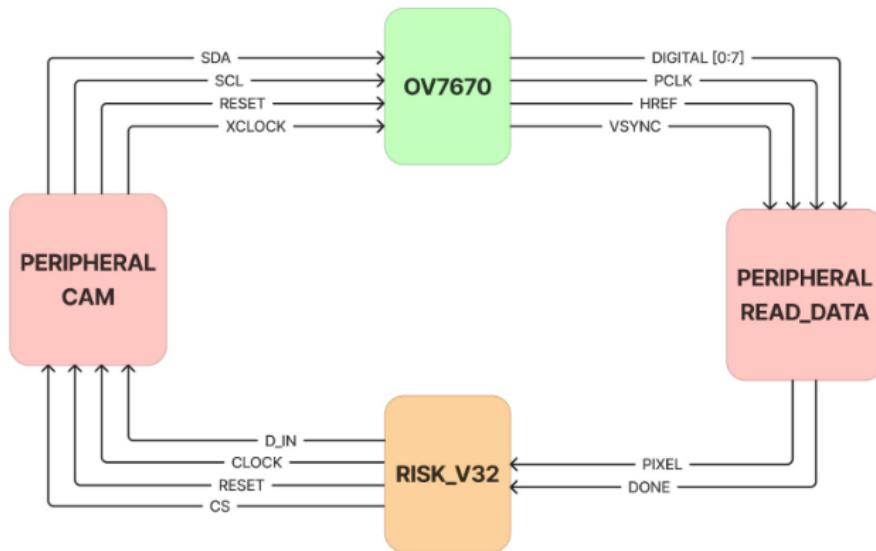


Figure 27: Esquema general de comunicación

5 Conclusiones

A lo largo del proyecto se comprendieron los conceptos del diseño digital así como la implementación de algunos de ellos mediante el planteamiento esquemático y su posterior implementación, adicionalmente mediante una "variante" del protocolo I2C se logró visualizar de manera correcta la configuración y envió de datos de la cámara OV7670, lo cual pudo ser observado mediante un analizador lógico.

Por otra parte se hizo uso del lenguaje ensamblador donde se realizó la interconexión de los diferentes módulos, tanto de hardware y software, permitiendo el correcto funcionamiento de la comunicación, además de un direccionamiento de memoria hacia el procesador para el guardado de los píxeles.

References

- [1] Camargo C.(2023).Síntesis de Sistemas Digitales. Universidad Nacional de Colombia.
- [2] Cánovas S.(2014).Sistema hardware de adquisición de vídeo basado en el sensor de imagen OV7670. Universidad Politécnica de Cartagena.