TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica
Área Académica de Ingeniería en Computadores
CE 4301 — Arquitectura de Computadores I

Proyecto Grupal 1
Diseño e Implementación de un ASIP de mejora de contraste

Fecha de asignación:	21 de octubre de 2020	Fecha de entrega:	13 de noviembre de 2020
Grupos:	3-4 personas	Profesores:	Luis Chavarría Zamora Jason Leitón Jiménez

Mediante el desarrollo de este proyecto, el estudiante aplicará los conceptos de arquitectura de computadores en el diseño e implementación en hardware de un *Application Specific Instruction Set Processor* (ASIP) para nitidez en imágenes. Atributos relacionados: **A**nálisis de **P**roblemas (AP), el cual se encuentra en **A**vanzado (A).

1. Descripción General

El método de ecualización de histogramas es un método utilizado para mejorar el contraste una imagen. Este efecto se observa en la Figura 1.

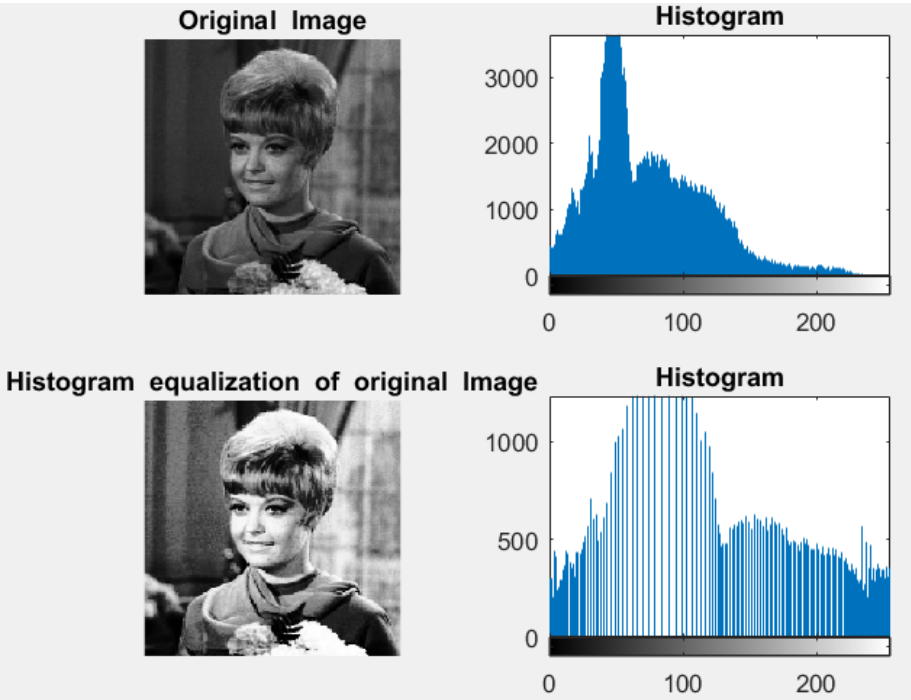

Figura 1: Ecualización de histogramas

El proceso de ecualización se muestra a continuación:

1. Se inicia con la imagen sin procesamiento:

$$I = \begin{bmatrix} 4 & 4 & 4 & 4 & 4 \\ 3 & 4 & 5 & 4 & 3 \\ 3 & 5 & 5 & 5 & 3 \\ 3 & 4 & 5 & 4 & 3 \\ 4 & 4 & 4 & 4 & 4 \end{bmatrix} \tag{1}$$

2. Se analiza (1) y se obtiene su frecuencia de distribución (número de veces que aparece un valor), esta se resume en la Tabla 1.

Tabla 1: Distribución de píxeles de la imagen (1)

I	0	1	2	3	4	5	6	7
f(I)	0	0	0	6	14	5	0	0

3. Luego se realiza la distribución acumulada, esto es sumar los píxeles de forma ascendente. Se le agrega una fila a la Tabla 1. El resultado se muestra en la Tabla 2.

Tabla 2: Frecuencia acumulada de la imagen (1)

I	0	1	2	3	4	5	6	7
f(I)	0	0	0	6	14	5	0	0
Cuf	0	0	0	6	20	25	25	25

4. Luego se distribuye uniformemente la frecuencia acumulada entre las opciones, obteniendo: $25/8 = 3{,}125$. Como se manejan píxeles enteros se le asigna 3 a 7 píexeles y 4 a uno de los píxeles para que en total se distribuya 25 ($3 \times 7 + 4 = 25$). Este valor de 4 puede ser en cualquier lugar, en este caso se coloca en el centro. El resultado se observa en la Tabla 3.

Tabla 3: Frecuencia distribuida de la imagen (1)

I	0	1	2	3	4	5	6	7
f(I)	0	0	0	6	14	5	0	0
Cuf	0	0	0	6	20	25	25	25
Feq	3	3	3	4	3	3	3	3

5. Luego se realiza la distribución acumulada de la frecuencia distribuida. Esto se observa en la Tabla 4.

Tabla 4: Frecuencia distribuida acumulada de la imagen (1)

I	0	1	2	3	4	5	6	7
f(I)	0	0	0	6	14	5	0	0
Cuf	0	0	0	6	20	25	25	25
Feq	3	3	3	4	3	3	3	3
CuFeq	3	6	9	13	16	19	22	25

6. Luego se realiza el remapeo de los píxeles usando las información de las filas 1, 3 y 5 como se muestra en la Tabla 5.

Tabla 5: Tabla para remapeo o ecualización de histogramas de la imagen (1)

I	0	1	2	3	4	5	6	7
Cuf	0	0	0	6	20	25	25	25
CuFeq	3	6	9	13	16	19	22	25

El remapeo o ecualización de histogramas se realiza buscando el valor de CuFeq que más se acerque a Cuf. Esto procedimiento se muestra a continuación:

a) $I = 0 \rightarrow \text{Cuf} = 0$. El valor de CuFeq que más se aproxima a 0 es 3, que es en $I = 0$. **Por esta razón el remapeo de I sería** $0 \rightarrow 0$**.**

b) $I = 1 \rightarrow \text{Cuf} = 0$. El valor de CuFeq que más se aproxima a 0 es 3, que es en $I = 0$. **Por esta razón el remapeo de I sería** $1 \rightarrow 0$**.**

c) $I = 2 \rightarrow \text{Cuf} = 0$. El valor de CuFeq que más se aproxima a 0 es 3, que es en $I = 0$. **Por esta razón el remapeo de I sería** $2 \rightarrow 0$**.**

d) $I = 3 \rightarrow \text{Cuf} = 6$. El valor de CuFeq que más se aproxima a 6 es 6, que es en $I = 1$. **Por esta razón el remapeo de I sería** $3 \rightarrow 1$**.**

e) $I = 4 \rightarrow \text{Cuf} = 20$. El valor de CuFeq que más se aproxima a 20 es 19, que es en $I = 5$. **Por esta razón el remapeo de I sería** $4 \rightarrow 5$**.**

f) $I = 5 \rightarrow \text{Cuf} = 25$. El valor de CuFeq que más se aproxima a 25 es 25, que es en $I = 7$. **Por esta razón el remapeo de I sería** $5 \rightarrow 7$**.**

g) $I = 6 \rightarrow \text{Cuf} = 25$. El valor de CuFeq que más se aproxima a 25 es 25, que es en $I = 7$. **Por esta razón el remapeo de I sería** $6 \rightarrow 7$**.**

h) $I = 7 \rightarrow \text{Cuf} = 25$. El valor de CuFeq que más se aproxima a 25 es 25, que es en $I = 7$. **Por esta razón el remapeo de I sería** $7 \rightarrow 7$**.**

El remapeo de los píxeles se observa en la Tabla 6.

Tabla 6: Remapeo final de píxeles de la imagen (1)

I	0	1	2	3	4	5	6	7
I'	0	0	0	1	5	7	7	7

7. Finalmente se contrasta la imagen original y la imagen remapeada a continuación en (2). De la Tabla 6 se convierten los píxeles: $3 \rightarrow 1$, $4 \rightarrow 5$ y $5 \rightarrow 7$.

$$I = \begin{bmatrix} 4 & 4 & 4 & 4 & 4 \\ 3 & 4 & 5 & 4 & 3 \\ 3 & 5 & 5 & 5 & 3 \\ 3 & 4 & 5 & 4 & 3 \\ 4 & 4 & 4 & 4 & 4 \end{bmatrix} \rightarrow I' = \begin{bmatrix} 5 & 5 & 5 & 5 & 5 \\ 1 & 5 & 7 & 5 & 1 \\ 1 & 7 & 7 & 7 & 1 \\ 1 & 5 & 7 & 5 & 1 \\ 5 & 5 & 5 & 5 & 5 \end{bmatrix} \tag{2}$$

Esta explicación se baso a lo mostrado en el enlace.

2. Especificación

Se le solicita desarrollar una **arquitectura** y una **microarquitectura** que realice el proceso de ecualización de histogramas con una imagen libre con dimensión mínima de 390×390.

Se deben seguir los siguientes requisitos generales de funcionalidad:

1. El diseño completo debe poder ser sintetizable en una tarjeta de desarrollo Terasic DE1-SoC-M TL2 (debe caber todo ahí, inclusive la imagen de entrada y el producto de ecualización por histograma).

2. Las imágenes de entrada y salida deben ser almacenadas en memoria (se recomienda usar el bloque IP de la biblioteca de Quartus).

3. El sistema debe permitir la interacción con el usuario, para poder escoger la imagen mediante algún periférico (e.g., botones, switches). Esta imagen se refleja en 8 pines GPIO con un reloj asociado.

4. El procesamiento de ecualización de histogramas se realizará una vez se seleccione mostrar la imagen de salida **por primera vez solamente**. Las próximas veces que se seleccione la imagen de salida no debe realizar el procesamiento para ahorrar tiempo.

5. La imagen mostrada debe ser escrita en un `.txt` que será leído e interpretado por un software de alto nivel libre.

6. El ISA debe ser eficiente y congruente, con criterios de diseño definidos. Es importante hacer reuniones con el profesor para guía.

7. El formato de las imágenes será en escala de grises con píxeles con valores entre [0, 255].

Requisitos de Arquitectura ISA:

1. Debe diseñar un conjunto de instrucciones y arquitectura que permita solucionar el problema planteado, considerando detalles como:

   a) Modos de direccionamiento.

   b) Tamaño y tipo de datos.

   c) Tipo y sintaxis de las instrucciones.

   d) Registros disponibles y sus nombres.

   e) Codificación y descripción funcional de las instrucciones

   Tome en cuenta que estos detalles deben ser justificados desde el punto de vista de diseño (complejidad, costo, área, recursos disponibles).

2. Las instrucciones a desarrollar son libres así como el tipo de datos. Aunque no hay un límite en cuanto la cantidad de instrucciones, es importante que provea al menos instrucciones para control de flujo, operaciones aritméticas-lógicas, acceso a memoria.

3. Los productos finales de esta etapa son el *instruction reference sheet* o *green sheet*.

4. El ISA debe ser personalizado y realizado por los estudiantes, no se aceptarán ISAs ya diseñados (e.g., ARM, x86, RISC-V, otros). Debe justificar cada característica del mismo.

Requisitos de Microarquitectura

1. La implementación diseñada debe ser correcta respecto a las reglas definidas por la arquitectura, esto quiere decir que el procesador debe ser capaz de ejecutar todas las instrucciones definidas y su especificación respecto a errores y excepciones.

2. El procesador diseñado debe emplear pipelining. Tenga en cuenta las implicaciones respecto a riesgos de dicha técnica, el uso de registros y unidades de ejecución. **No se revisará si no tiene pipeline.**

3. Debe ser implementado usando SystemVerilog.

4. No se permite realizar módulos especializados de hardware. Es un curso de Arquitectura de Computadores, no de Diseño de Sistemas Digitales.

5. El procesador debe tener capacidad de segmentación de memoria en datos e instrucciones además debe ser capaz de acceder los dispositivos de entrada y salida del sistema (GPIO, volcado de memoria, switches, etc).

6. Cada unidad funcional del sistema debe ser debidamente probada en simulación, para verificar su funcionamiento correcto (unit tests). Además debe incluir pruebas de integración y sistema. Se le solicita un plan de pruebas donde especifique los objetivos y descripción de las pruebas junto con sus resultados.

7. Los resultados finales de esta etapa son:

   a) El código fuente (SystemVerilog) y el bitstream para programar la tarjeta de desarrollo.

   b) Un diagrama de bloques de la microarquitectura y descripción de las interacciones entre ellos.

   c) Simulaciones de las pruebas unitarias y de integración.

8. Reporte de consumo de recursos del FPGA para el modelo.

Requisitos de Software:

1. Crear una aplicación (software) empleando la arquitectura diseñada, con el fin de implementar la ecualización de histogramas.

2. Debe realizar un programa ('compilador') que permita traducir las instrucciones del ISA a binario, con la finalidad de ejecutarlo en el procesador. No es necesario que realice análisis léxico, sintáctico y semántico (este curso no es de Compiladores).

El proceso de diseño debe incluir propuestas y comparación de viabilidad de las mismas.

3. Evaluación y entregables

La defensa será el mismo día de la entrega y todos los archivos (incluyendo código fuente) serán entregados a las 11:59 pm ese mismo día (realícenlo progresivamente y no lo deje para el final). La evaluación del proyecto se da bajos los siguientes rubros contra rúbrica correspondiente:

- Presentación proyecto 100 % funcional (65 %): La defensa se realizará de la siguiente manera: Debido a la situación actual (COVID-19) no se puede tener acceso a hardware u otros instrumentos y medios que requieran presencia y contacto físico tanto entre el profesor como l@s estudiantes. Por esta razón para la defensa se debe presentar lo siguiente en una hora:

1. Todo el diseño debe ser sintetizable en una tarjeta: **Terasic DE1-SoC-M TL2**. Es decir, debe llegar hasta la generación de un **.sof**. Se garantiza que la tarjeta tenga suficientes recursos para almacenar y ejecutar el diseño según el reporte.
2. Debe reservar los espacios de memoria para la imagen de salida.
3. Mediante ModelSim debe crear un *testbench* de los mismos archivos de SystemVerilog que se usaron para sintetizar. Con este *testbench* debe escribir un archivo:
   - Imagen de salida (.img).

   Este archivo es una representación que contiene
4. Debe crear un script de alto nivel para visualizar la imagen de entrada y la de salida.

Los entregables adicionales que se revisarán en la defensa son los siguientes:

1. Arquitectura:
   a) *Instruction reference sheet* o *green sheet*.
2. Microarquitectura:
   a) Diagrama de bloques de la microarquitectura.
   b) Reporte de consumo de recursos del FPGA.
3. Software
   a) Programa de software.
   b) Compilador usado.

- Articulo científico tipo *paper* (17.5 %): El paper a realizar deberá tener una extensión no mayor a 4 páginas completas (incluyendo bibliografías), deberá ser realizado con LaTeX, siguiendo un formato establecido (IEEE Transactions o ACM, por ejemplo). Se les provee un ejemplo de paper en el enlace. En general el *paper* deberá contar con las siguientes secciones:

  1. Abstract (en inglés): Un buen abstract tiene las siguientes características:
     a) Un abstract permite a los lectores obtener la esencia o esencia de su artículo o artículo rápidamente, para decidir si leer el artículo completo.
     b) Un abstract prepara a los lectores para seguir la información detallada, los análisis y los argumentos en su artículo completo.
     c) Un abstract ayuda a los lectores a recordar puntos clave de su paper.
     d) Un abstract es de entre 150 y 250 palabras.
  2. Palabras clave significativas (a lo sumo 6).

3. Introducción: Una buena introducción muestra el contexto del problema o lo que se va a solucionar, introduce el tema al lector. Al final de la introducción se indica la organización del documento (primero se muestra el algoritmo, luego....).
4. Algoritmo desarrollado.
5. Resultados.
6. Conclusiones escritas en prosa.
7. Bibliografía, en formato IEEE y referenciadas en el texto (usar cite). Referencia bien para evitar problemas de plagio. Una documento no referenciado en el texto no existe.

- Documentación de diseño (17.5 %): Este documento se encuentra directamente ligado con el atributo AP. La documentación del diseño deberá contener las siguientes secciones:

1. Listado de requerimientos del sistema: Cada estudiante deberá determinar los requerimientos de ingeniería del problema planteado, considerando partes involucradas, estado del arte, estándares, normas, entre otros.
2. Elaboración de opciones de solución al problema: Para el problema planteado deberán documentarse al menos dos opciones de solución. Cada solución deberá ser acompañada de algún tipo de diagrama.
3. Comparación de opciones de solución: Se deberán comparar explícitamente las opciones de solución, de acuerdo con los requerimientos y otros aspectos aplicables de salud, seguridad, ambientales, económicos, culturales, sociales y de estándares.
4. Selección de la propuesta final: Se deberá evaluar de forma objetiva, válida y precisa las soluciones planteadas al problema y escoger una solución final.
5. Archivo tipo README donde especifiquen las herramientas que usaron. **Es un documento README.MD aparte.**

Se seguirán los siguientes lineamientos:

1. Los documentos serán sometidos a control de plagios para eliminar cualquier intento de plagio con trabajos de semestres anteriores, actual o copias textuales, tendrán nota de cero los datos detectados. Se prohíbe el uso de referencias hacia sitios no confiables.

2. No coloque código fuente en los documentos, quita espacio y aporta poco. Mejor explique el código, páselo a pseudocódigo o use un diagrama.