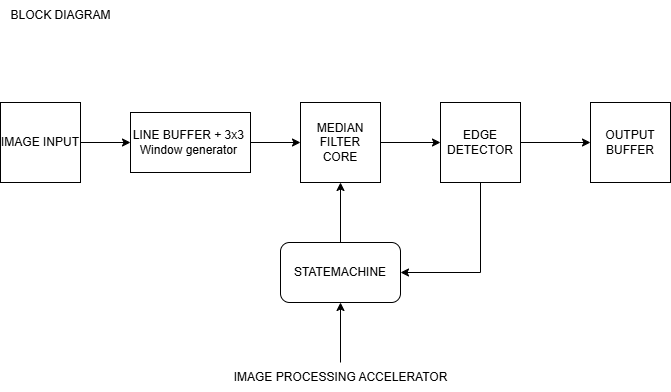
****

**1. Image Input**

* **Purpose:** Supplies the pixel data to the accelerator.
* **Details:** This could be an image stored in memory or streamed pixel-by-pixel into the system.
* **Output:** Sequential pixel values.

**🟦 2. Line Buffer + 3×3 Window Generator**

* **Purpose:** Creates a 3×3 pixel window centered on the current pixel.
* **Details:** Uses line buffers (shift registers or RAM) to hold rows of the image. As pixels are streamed in, the window "slides" across the image.
* **Output:** A 3×3 array of pixel values used for filtering.

**🟦 3. Median Filter Core**

* **Purpose:** Performs **noise reduction** by applying a median filter.
* **Details:**
  + Sorts the 9 pixel values from the 3×3 window.
  + Selects the middle (median) value as the output.
  + Helps eliminate salt-and-pepper noise while preserving edges better than averaging filters.
* **Output:** A filtered version of the center pixel.

**🟦 4. Edge Detector**

* **Purpose:** Detects edges in the filtered image.
* **Details:**
  + Applies a convolution filter (e.g., **Sobel**, **Prewitt**, or **Roberts**).
  + Calculates gradient magnitude or direction to highlight edges.
  + Typically uses neighboring pixels to compute intensity changes.
* **Output:** Edge-enhanced pixel value (e.g., grayscale with stronger values at edges).

**🟦 5. Output Buffer**

* **Purpose:** Stores or sends out the processed image.
* **Details:** Could be written to external memory, VGA display, or UART interface depending on use case.

**🟦 State Machine**

* **Purpose:** Controls the operation of all blocks.
* **Details:**
  + Manages states like IDLE, LOAD, PROCESS, STORE, DONE.
  + Synchronizes timing between buffering, filtering, and edge detection.
  + Ensures proper sequencing and data validity.

**🔁 Overall Flow Summary**

1. Pixels are read from memory/input.
2. Line buffers collect them into a 3×3 window.
3. The median filter cleans up noise.
4. The edge detector highlights edges.
5. Results are written to an output buffer.