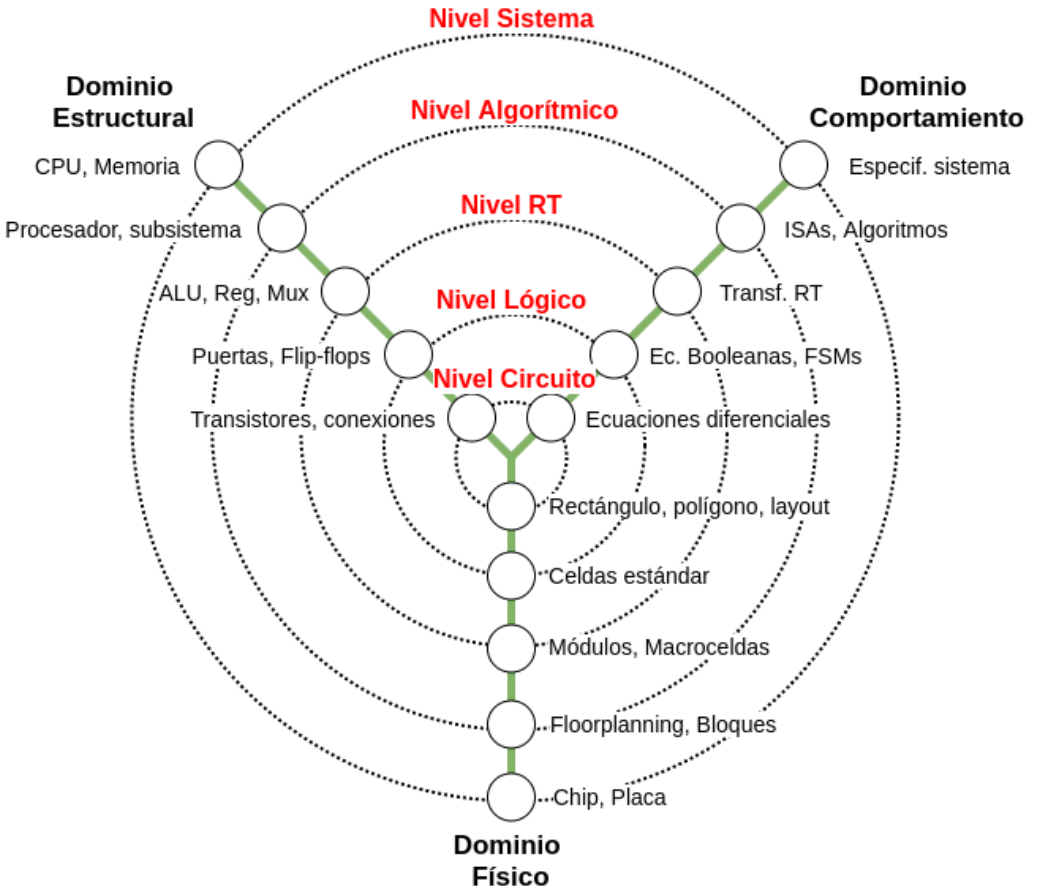
El siguiente ejemplo, muestra el abordaje de una situación problema por medio de los dominios comportamentales, estructurales y físicos.

Más allá de escribir código, simplificar ecuaciones o implimentar una síntesis en una fpga, el principal objetivo de este ejemplo, es que usted piense de manera algorítmica y contextualizada, vea diferentes opciones para afrontar situaciones reales con técnicas de diseño. Por favor, observe el siguiente diagrama y comprenda cómo está relacionado los dominios, niveles de abstracción, representaciones top-down, entre otras.



Nivel de Abstracción	Valores	Medidas
Sistema	Relaciones entre subsistemas, sincronización y protocolos.	Ancho de banda, MIPS.
Algorítmico	Estructuras abstractas. Se usan las dependencias en lugar del tiempo.	Latencia, cadencia de datos, número de módulos.
RT (Register Transfer)	Palabras con valores discretos. Control y procesamiento en tiempo discreto.	Tiempos de ciclo, márgenes y puertas equivalentes.
Lógico	Valores lógicos. Computación en tiempo continuo.	Tiempos de conmutación, Skew y áreas equivalentes.
Circuito	Valores continuos. Todo es electrónica en tiempo continuo.	Tiempos de subida, bajada y consumos de área.

0. Situación problema, especificaciones y requerimientos

En un edificio residencial de cinco pisos, los residentes de los pisos 4 y 5 han reportado problemas con la presión del agua, afectando sus actividades cotidianas. Ante esta situación, el administrador del edificio decide buscar una solución y le solicita a usted diseñar un sistema que permita garantizar el suministro de agua.

Tras analizar el problema, usted propone instalar un tanque de agua en la terraza del edificio, el cual será llenado automáticamente por una bomba eléctrica. Durante la conversación con el administrador, se identifican las siguientes necesidades:

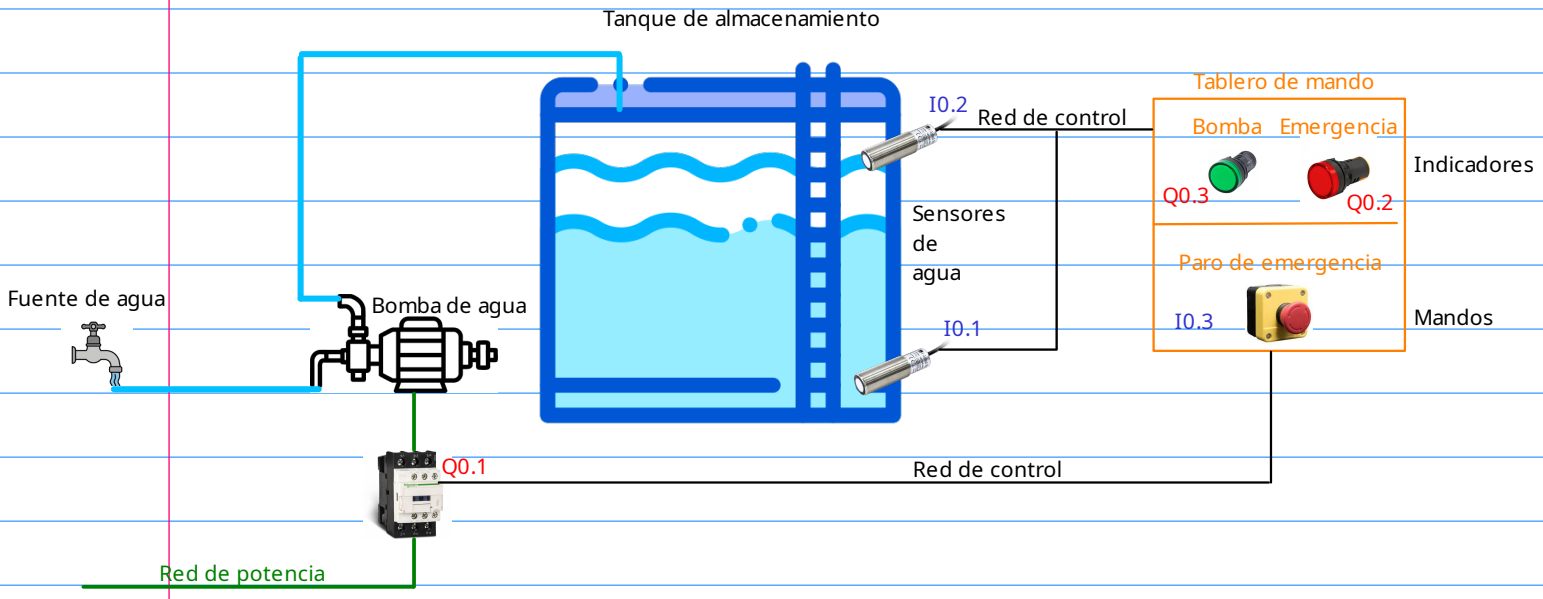
1. La bomba no debe permanecer encendida todo el tiempo para evitar un aumento excesivo en el consumo eléctrico del edificio.
2. El sistema debe contar con la capacidad de ser detenido en caso de emergencia o durante procedimientos de mantenimiento.

Con base en estas necesidades, usted diseña una solución que incluye:

1. Sensores para automatizar el llenado del tanque:
   * Uno colocado en la parte superior para detectar cuando el agua alcanza el nivel máximo y detener la bomba, evitando desbordamientos.
   * Otro en la parte inferior para detectar cuando el agua alcanza el nivel mínimo y activar la bomba automáticamente.
2. Un botón de paro de emergencia, que permitirá detener el funcionamiento del sistema tanto en emergencias como en situaciones de mantenimiento.
3. Indicadores luminosos:
   * Una luz verde para indicar cuando la bomba está operando.
   * Una luz roja para señalar que el sistema está detenido por el botón de paro.

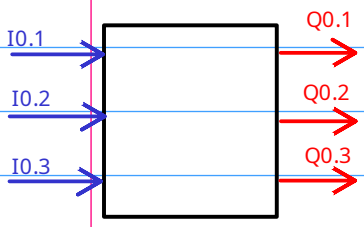
El administrador del edificio aprueba la idea y lo contrata para desarrollar e implementar la solución. Ahora es su responsabilidad diseñar el sistema que cumpla con estas especificaciones, garantizando un funcionamiento eficiente y seguro.

Representación esquemática de los sensores y actuadores a controlar en la solución a plantear:



1. Dominio comportamental (especificación y algoritmo)

1.1 Sistema de caja negra



I0.1 Sensor de nivel de agua bajo

I0.2 Sensor de nivel de agua alto

I0.3 Mando de paro de emergencia

Q0.1 Relé de bomba de agua

Q0.2 Indicador paro de emergencia

Q0.3 Indicador de bomba encendido

Nota: I0.1 e I0.2 son sensores que se activan al detectar agua en ese nivel.
I0.3 es un pulsador que al ser activado (accionado por el usuario) abre el circuito.
Q0.1, Q0.2 y Q0.3 son actuadores que se encienden o se apagan.

En este sistema se representa
las entradas y salidas de información

I0.1 e I0.2 : Activo -> Interruptor normalmente cerrado (NC) se abre
Inactivo -> Estado natural (NC).

I0.3 -> Activo: El contacto normalmente cerrado (NC) se abre.
Inactivo -> Estado natural (NC).

1.2 Tabla de verdad

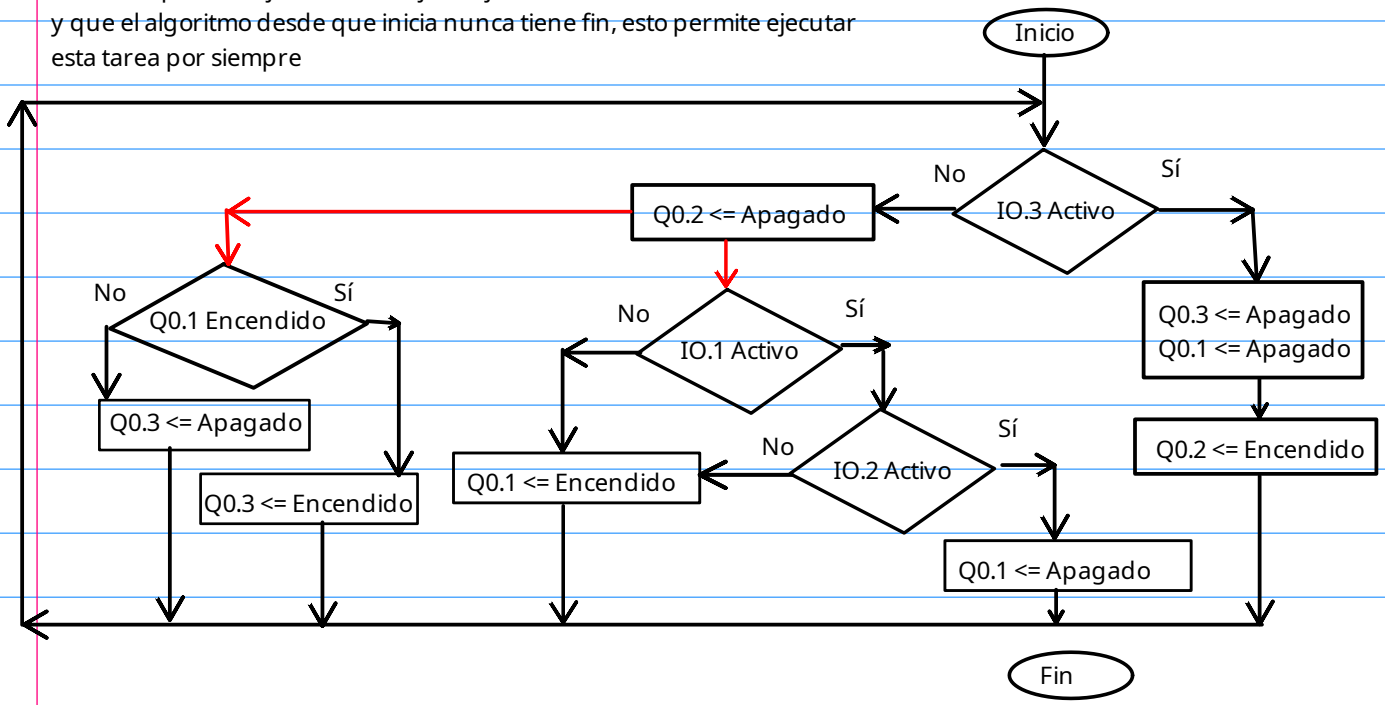
IO.3	I0.2	I0.1	Q0.3	Q0.2	Q0.1	Observaciones
0	0	0	1	0	1	- Indicador de bomba y bomba encendidos
0	0	1	1	0	1	- Indicador de bomba y bomba encendidos
0	1	0	1	0	1	- Indicador de bomba y bomba encendidos (hay un error en los sensores)
0	1	1	0	0	0	- Tanque lleno, indicador de bomba y bomba apagados
1	x	x	0	1	0	- Estado de emergencia o de mantenimiento, circuito de trabajo abierto, indicador de estado de emergencia o mantenimiento activo.

Nota: las x representan una condición que no importa

Nota: aunque Q0.3 es iguala a Q0.1 es importante indicar que Q0.3 dependa del estado de Q0.1, de tal manera que se pueda detectar errores físicos, por ejemplo, cuando el relé de Q0.1 está activo, Q0.3 también lo está pero el motor no funciona, se puede indicar que el error está en el motor y no en la lógica de los sensores.

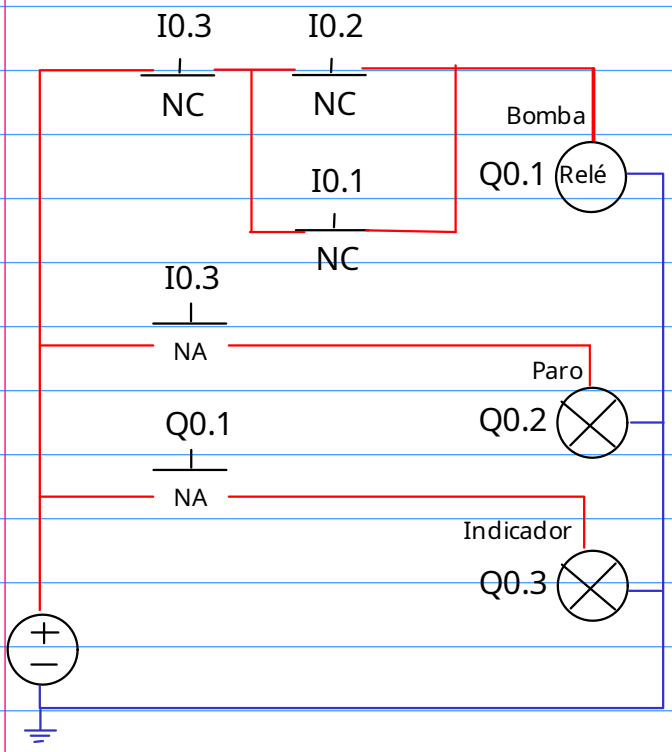
1.3 Algoritmo representado en diagrama de flujo

Observe que los flujos de color rojo se ejecutan simulataneamente
y que el algoritmo desde que inicia nunca tiene fin, esto permite ejecutar
esta tarea por siempre

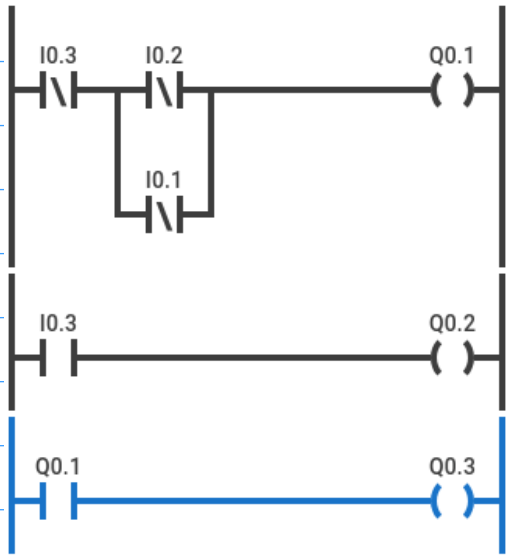


2. Dominio físico inicial (circuito eléctrico)

Respresentación del algoritmo a modo de circuito eléctrico de comportamiento ON/OFF



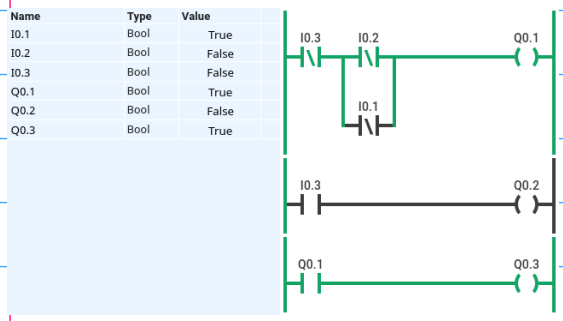
Circuito representado en lenguaje ladder



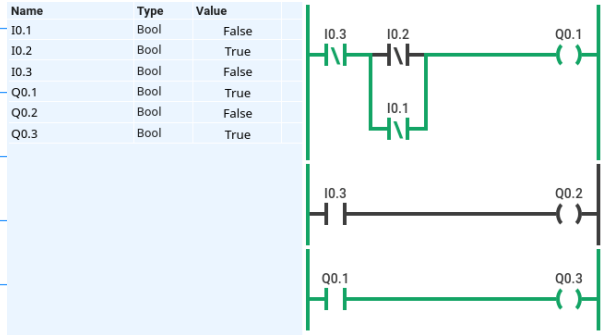
Haciendo uso de los conocimientos de circuitos eléctricos, se elabora el circuito, tenga presente que usted puede ir armando el circuito e ir comprobando si se cumplen las condiciones propuestas en el dominio comportamental. Cuando se ha logrado, traducir al lenguaje escalera consiste en verificar el tipo de contacto a usar (NA o NC) junto a las etiquetas a usar. Las etiquetas son importantes, de hecho, observe que por algunas etiquetas se asocian contactos dobles, eso quiere decir que físicamente al accionar una de estas etiquetas, por ejemplo un pulsador o un relé, varios contactos pueden cambiar de su estado normal a su estado complementario al mismo tiempo.

3. Simulación en lenguaje ladder

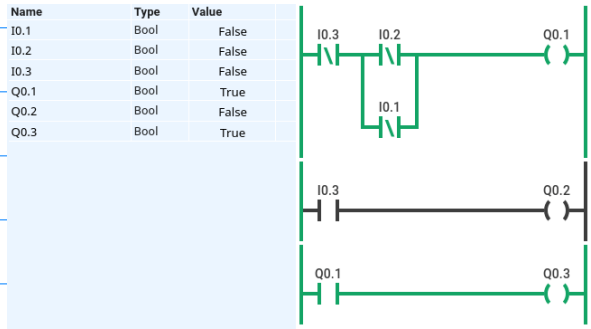
Bomba activa por sensor I0.2 (nivel alto) el cual no detecta agua



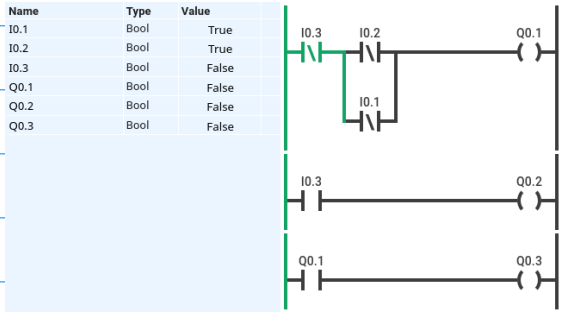
Bomba activa por sensor I0.1 que no detecta agua, observar que I0.2 sí detecta agua, un estado no deseado que posiblemente se trate de un error (hallazgo)



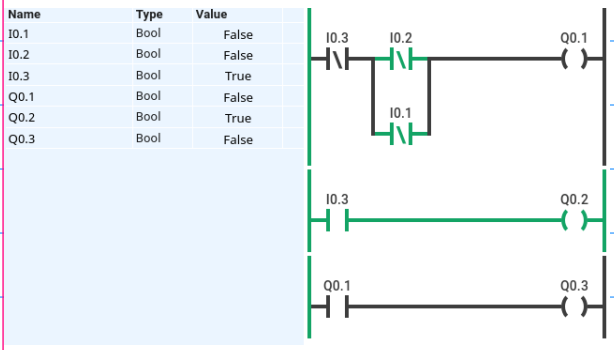
Bomba Q0.1 está encendida porque el tanque está totalmente vacío



Bomba inactiva debido a que el tanque está totalmente lleno.



Paro de emergencia activo, indicador del paro de emergencia encendido, no importa el estado de los sensores.

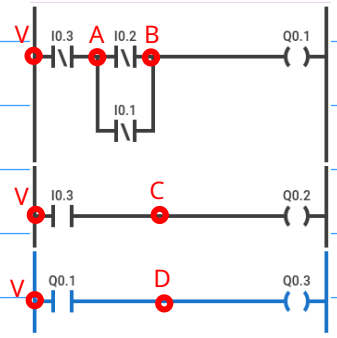


Las simulaciones fueron realizadas con el software: https://app.plcsimulator.online/

Según lo planteado en el dominio comportamental en comparación con las simulaciones, la solución del circuito eléctrico (dominio físico) es válido teniendo presente los hallazgos en la simulación; por tanto, la solución podría implementarse a través de los relés, contactos e indicadores que han sido considerados, es decir, podría ofrecerle esta implementación como solución al administrador del edificio.

Por otro lado, si tomamos la solución física y la trasladamos al dominio estructural, podría desde allí traducir la solución estrutural a otras implementaciones en el dominio físico. En este ejemplo, al nivel de abstracción al que queremos llegar del dominio estructural es el de las compuertas lógicas y verificación de su comportamiento a través de otras simulaciones.

4. Dominio estructural (red de compuertas lógicas)

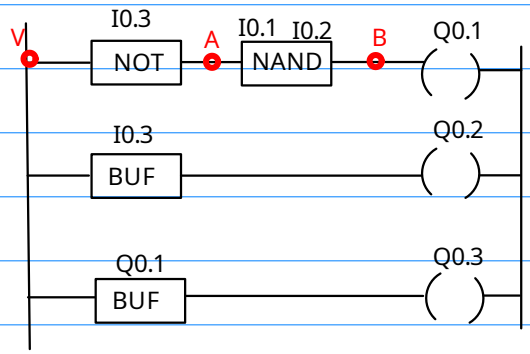


Para construir una red de compuertas hay que tener clara la relación que existe entre la lógica cableada y la tabla de verdad, la cual puede corresponer a una función lógica como la not, xnor, nand, and, xor, or y nor.
Como estrategia, se plantea para este caso (en otros casos se pueden sustituir compuertas compleajas como la xnor o la xor, entre otras), realizar una simplificación de aquellos nodos que se encuentren en paralelo y sabiendo la compuerta que representa sustituir su diagrama por una caja negra.

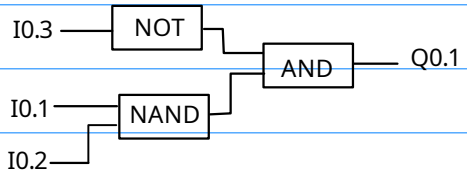
Por ejemplo, los elementos conectados entre los nodos:

V-A: Se puede representar por una compuerta NOT
A-B: Se pueden representar por una compuerta NAND
V-C: Se representa por un Buffer (seguidor)
V-D: Se representa por un Buffer (seguidor)

Ya con la información obtenida, se puede redibujar el diagrama y representar el contenido de estos nodos por cajas negras, es decir, conservar los nodos, e indicar exclusivamente la función que representa, sin indicar el cómo.

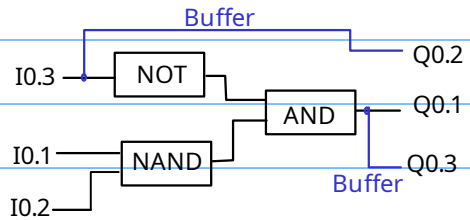


Observe ahora que en los nodos V y B hay dos cajas que están conectadas en serie, es decir, NOT y NAND comparten un nodo en común (A) y los nodos V y B son los nodos terminales. A esta configuración en serie, se tiene un equivalente a una compuerta AND. El resultado en compuertas para este par de nodos se puede representar de la siguiente manera:



Tenga presente que se está cambiando de dominio físico a estructural, por tanto, las líneas que representaban cables de conexión electrica y contactos NA y NC ya no están representados en este dominio. Las líneas en el dominio estructural están representando la comunicación de un estado lógico de alguna de las variables de entrada o de salida a una caja que cumple una función lógica. Tener claro este proceso permitirá volver a un dominio físico de otra tecnología como pueden ser transistores y representar un circuito que cumpla las mismas funciones lógicas.

Finalmente, en el dominio físico los contactos normalmente abiertos son elementos que en el dominio estructural son representados como seguidores, buffers que se usan para regenerar una señal y garantizar un valor, en el dominio estructural serán representados por cables, por tanto el circuito final queda como sigue:



En este punto es importante identificar si se cumple las condiciones de operación propuestas en el dominio comportamental una manera de hacerlo es a través de una simulación que permita verificar su tabla de verdad.
En el simulador de Digital, se simula el diagrama digital propuesto.

4.1 Simulación en el dominio estructural

Para tal fin, se crea un proyecto en Digital en una carpeta con el nombre del proyecto, pero el archivo a guardar tendrá el nombre de top.dig, esto permitirá que en un paso más adelante se pueda sintetizar esta solución en un PLD (dispositivo lógico programable). Para obtener los diferentes resultados y diagramas haga uso de la herramienta de análisis de Digital.

Representación en compuertas de la solución

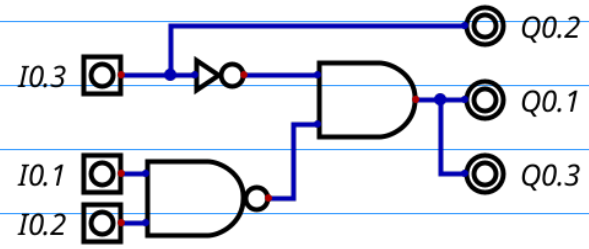
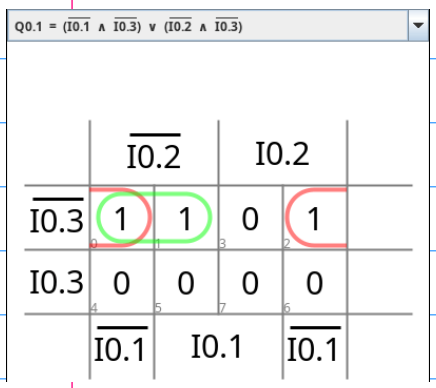


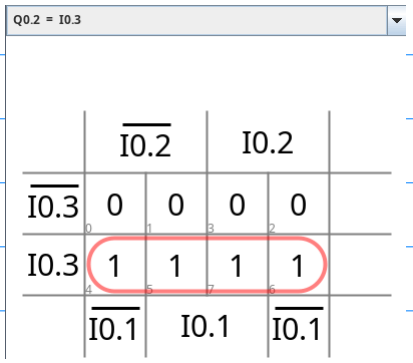
Tabla de verdad del comportamiento de la red de compuertas, comprueba el resultado con la tabla de verdad del dominio comportamental

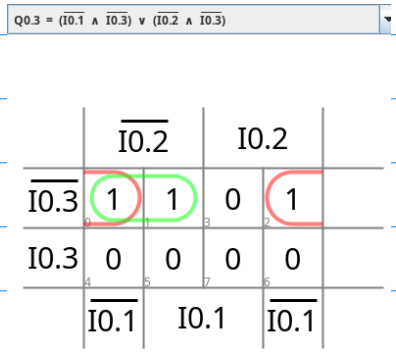
I0.3	I0.2	I0.1	Q0.3	Q0.2	Q0.1
0	0	0	1	0	1
0	0	1	1	0	1
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	0	1	0

Herramienta Digital: https://github.com/hneemann/Digital

Digital resuelve las ecuaciones de las funciones lógicas que representan a cada una de las salidas. Como puede apreciar se representa su reducción a través de los mapas de karnaugh

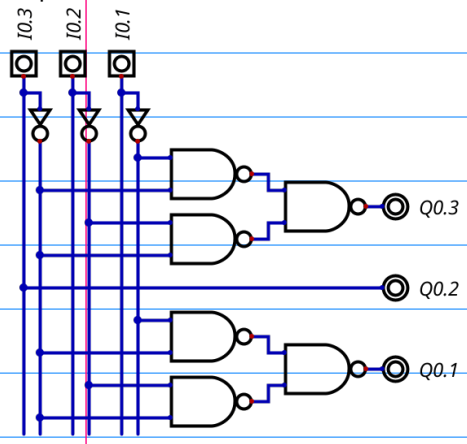




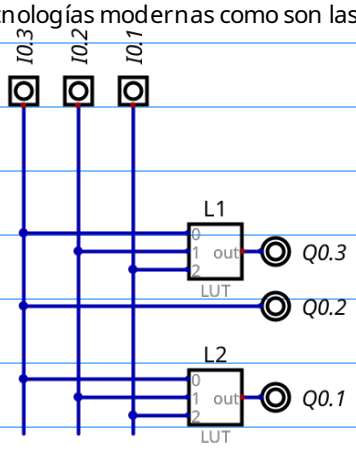


Digital propone circuitos digitales equivalentes que se puedan implementar en alguna tecnología.

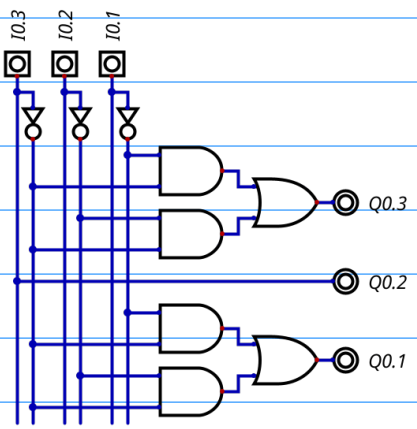
Solución equivalente representada por la compuerta universal NAND.



Representación en lookup tables que son las unidades básicas de tecnologías modernas como son las FPGAs



Sumas de productos



5. Descripción en lenguaje HDL (Hardware Description Language)

Como ha podido observar, las abstracciones se han representado a través de métodos gráficos. Para los casos de circuitos electrónicos digitales y análogos se puede representa a través de texto, como puede ser por medio de modelos spices. Para el caso del dominio estrutural y comportamental, estas abtracciones se puden representar por medio de lenguajes de descripción de hardware, como puede ser el caso de Verilog, SystemVerilog, VHDL, Migen, entre otros.
Estos lenguajes permiten traducir tales representaciones a alguna tecnología moderna, como puede ser FPGAs, a este proceso se le llama síntesis (dentro de lo cual, hay un flujo de diseño que se requiere seguir según cada tecnología).

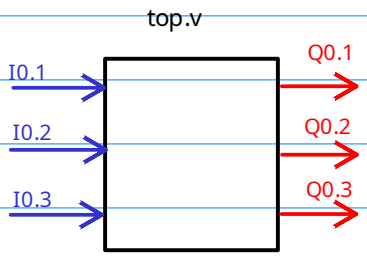
Mientras que se aprende estos leguajes, se puede hacer uso de Digital, el cual puede traducir estos circuitos gráficos a un lenguaje HDL como es el caso de Verilog o VHDL. Este proceso lo puede realizar desde la barra de herramientas haciendo uso de export, por ejemplo "Export to Verilog", genera un archivo top.v que contiene el siguiente resultado:

```
/*
 * Generated by Digital. Don't modify this file!
 * Any changes will be lost if this file is regenerated.
 */

module top (
  input \I0.1 ,
  input \I0.2 ,
  input \I0.3 ,
  output \Q0.1 ,
  output \Q0.2 ,
  output \Q0.3
);
  wire \Q0.3_temp ;
  assign \Q0.3_temp  = (~ \I0.3  & ~ (\I0.1  & \I0.2 ));
  assign \Q0.1  = \Q0.3_temp ;
  assign \Q0.2  = \I0.3 ;
  assign \Q0.3  = \Q0.3_temp ;
endmodule
```

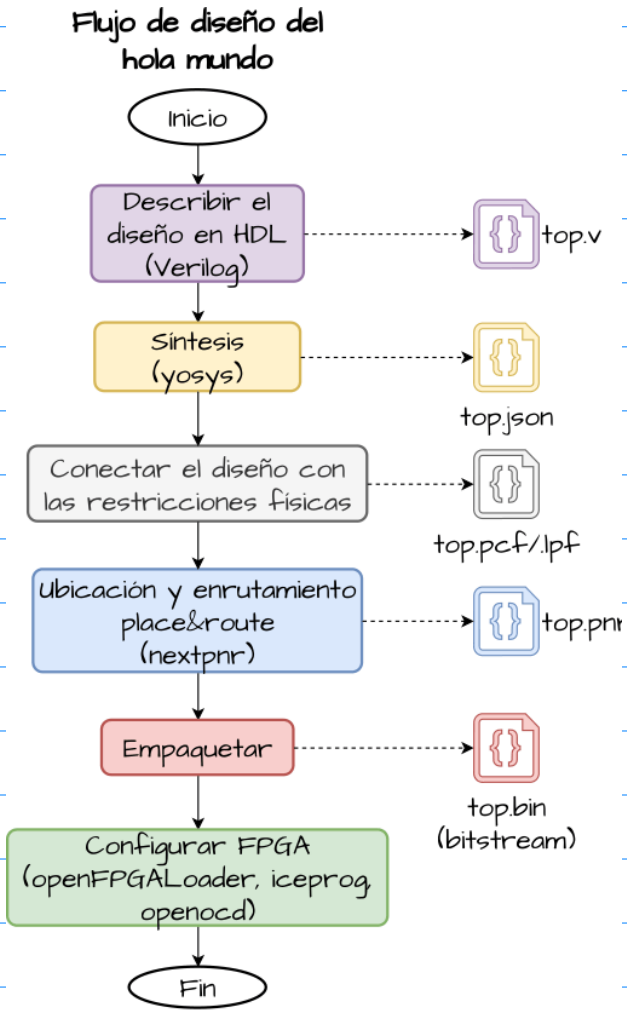
Observe lo siguiente en el resultado:

- Nombre del módulo
- Numero de entradas y salidas
- Las asignaciones existentes
- Los operadores ~ (not), & (and), = (asignación)
- Observe comparativamente la caja de abajo (caja negra) con el código, identifique los puertos que se han llamado input y ouput
- El uso de \I0.1 no necesariamente es obligatorio, cuando conozca el lenguaje podrá realizar modificaciones

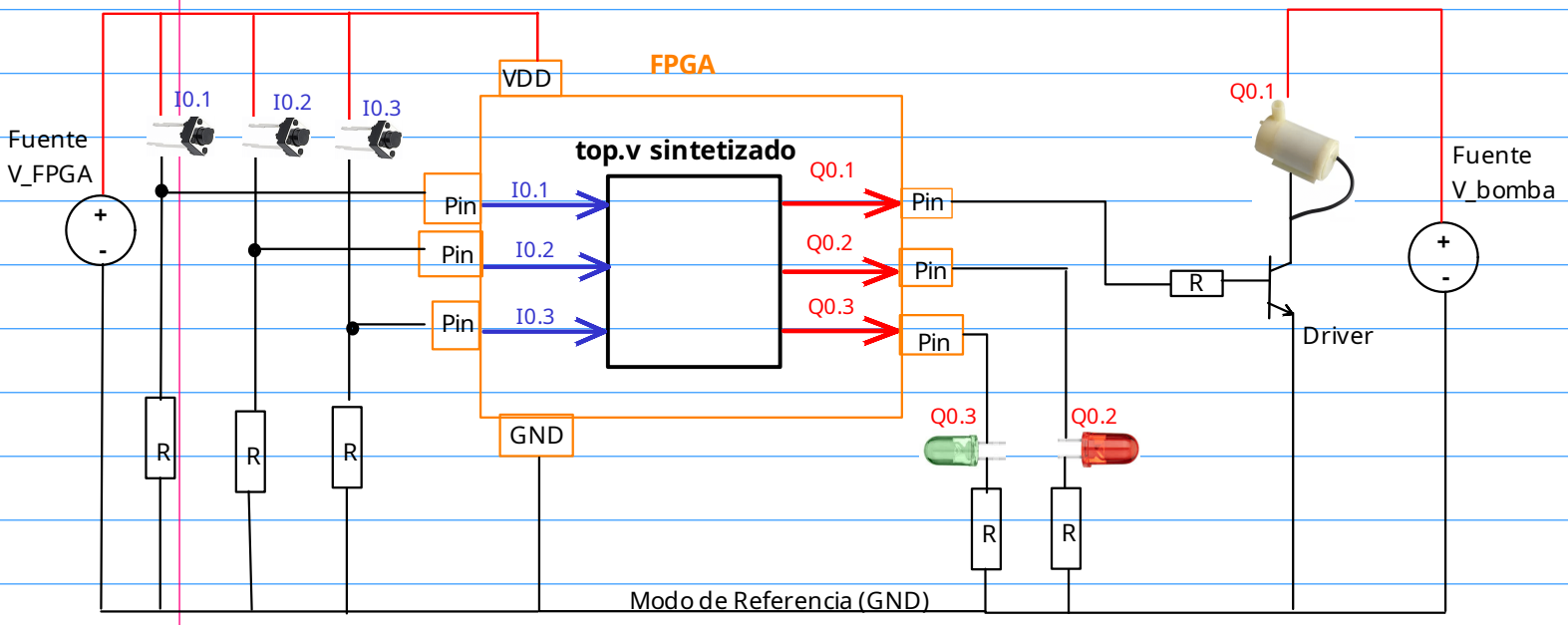


6. Síntesis en FPGA (dominio físico final)

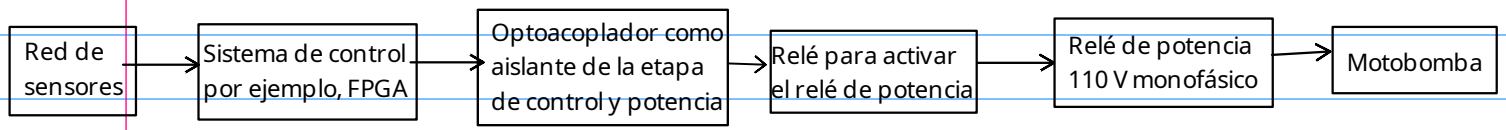
En el anterior apartado representamos la solución a través de un modelo estrutural de manera gráfica (red de compuertas) como también en un lenguaje que describe el hardware de esa solución. Podríamos entonces con esa información volver al dominio físico y por ejemplo plantear un circuito con transistores que cumplan los planteamientos propuestos en el dominio comportamental. También existen herramientas digitales capaces de recibir como entrada un HDL y traducirlo a un chip de silicio (como puede ser QFlow de opencircuit) y generar un diseño que pueda ser fabricado en una fabless.
Para continuar con el ejemplo, al tener este HDL, podemos implementar la solución configurando una FPGA, esta FPGA tiene bloques lógicos programables que son compuestos por LookUp Tables (que representan tablas de verdad, generalmente de 4 entradas) y flip-flops (unidades de almacenamiento de bits).
Según el fabricante de la FPGA, hay unas herramientas que permiten el flujo de diseño de sintesis, generación del archivo de configuración (bitstream) y configuración del mismo en la FPGA (subir el bitstream a la FPGA).
A continuación se muestra el flujo de diseño para una FPGA que funciona con herramientas de desarrollo puramente opensource (yosys, nextpnr, openFPGALoader, entre otros), este flujo de diseño generalmente es compartido por herramientas privativas de otros fabricantes.



Suponiendo que se ha realizado el proceso de configuración de una FPGA lo que tendremos ahora es una FPGA que contiene en su interior, la información del comportamiento de las salidas en función de sus entradas, sin embargo, las restricciones físicas de la FPGA deben ser consideradas para garantizar los niveles lógicos de tensión-corriente para estimular las entradas y los drivers necesarios para hacer funcionar los actuadores.
Aunque no es el caso, supongamos que se desea realizar una maqueta del funcionamiento del diseño de la solución para mostrar al administrador, por tanto, a través de pulsadores, leds y una minimotobomba, se desea dar a conocer el funcionamiento de la solución, veamos entonces cómo se conectan los elementos a la FPGA.



Finalmente, es de aclarar, que la solución a plantear al administrador se puede plantear con componentes electrónicos y relés, es decir, en el diagrama de arriba se planteó el uso de pulsadores y una minibomba, sin embargo, con las adaptaciones relevantes se puede acondicionar para hacer funcionar los actuadores del problema; basta con crear unas etapas de aislamiento entre el control y Relé, por ejemplo, a través de optoacopladores y Relés:



Regards,
Johnny Cubides