USO DE LAS MAQUINAS DE ESTADO FINITOS Y MAQUINAS DE ESTADO ALGORITMICAS EN LOS LENGUAJES DE DESCRIPCION DE HARDWARE

Autora:
Ing. Darwin Materan
Año: 2018

Email: materand@gmail.com

RESUMEN

En esta tesis se propone una metodología para el desarrollo de programas de descripción de hardware aplicando como estrategia didáctica el uso de Maquinas de estados finitos y Maquinas de estado algorítmicas como abordaje en el análisis a los problemas planteados y tratando de establecer un esquema de fases para la resolución de problemas tal como se cursó en los diseño de programas en los cursos de programación básica al establecer metodologías de programación que permitían realizar un análisis previo de los problema y luego su diseño y edición. Y así, dar soporte a los estudiantes de Arquitectura del Computador del Tercer semestre de la carrera de Ingeniería en sistemas quienes emplean los lenguajes de descripción de hardware como conocimiento transversal del curso, de ahí que se aplica un auto-aprendizaje del mismo

En primer lugar se abordan el empleo de las máquinas de estado finitos en el diseño de circuitos secuenciales y luego se traduce una máquina de estado algorítmica para así poder llegar de una manera más sencilla y didáctica a la programación en VHDL.

Por medio de Lenguajes de Descripción de hardware es posible simplificar circuitos y el uso de herramientas de análisis como FMS y ASM nos permite desarrollar circuitos secuenciales en su mínima expresión cumpliendo con las tareas encomendadas en su desarrollo.

Palabras Clave: Metodología, Hardware, Estrategia, Maquina, Estado, Finito, Algorítmica, Lenguaje ,VHDL, Circuito Diseño, Programación.

Revista Electrónica Facultad de Ingeniería UVM
DepósitoLegal:PPX200602TR2436 / ISSN:1856-6936.VolumenXII EdiciónNo2– 2018

USE OF FINITE STATE MACHINES AND ALGORITHMIC STATE MACHINES IN HARDWARE DESCRIPTION LANGUAGES

Author:
Ing. Darwin Materan

Year: 2018

SUMMARY

In this thesis a methodology for the development of hardware description programs is proposed, applying as a didactic strategy the use of Finite State Machines and Algorithmic State Machines as an approach in the analysis to the problems posed and trying to establish a phase scheme for problem solving as it was done in the design of programs in the basic programming courses by establishing programming methodologies that allowed a prior analysis of the problem and then its design and editing. And so, give support to Computer Architecture students of the third semester of the Systems Engineering degree who use the hardware description languages as transversal knowledge of the course, hence a self-learning of the same is applied

First, the use of finite state machines in the design of sequential circuits is addressed and then a Algorithmic state machine to be able to arrive in a more simple and didactic way to the programming in VHDL.

By means of hardware description languages, it is possible to simplify circuits and the use of analysis tools such as FMS and ASM allows us to develop sequential circuits in their minimum expression, fulfilling the tasks entrusted to their development.

Key Words: Methodology, Hardware, Strategy, Machine, State, Finite, Algorithmic, Language, VHDL, Design Circuit, Programming.

INTRODUCCIÓN

En la actualidad se encuentra una gama de metodologías para el análisis y desarrollo de programas, muchos de ellos para programación modular. Cuando nos enfrentamos a programas de descripción de hardware observamos la ausencia de algoritmos que permitan realizar un bosquejo del mismo a pesar de existir en la literatura herramientas que permiten abordar los mismos.

Ante esta situación se busca elaborar una metodología completa que permita atacar el problema de hardware y describirlo a través de un programa de descripción de hardware, específicamente VHDL. Para ello se realizará una investigación conceptual y aplicada que permitirá abordar toda la documentación existente y elaborar una metodología completa que permita al diseñador de hardware documentar y dar solución al problema planteado.

Se iniciara con la investigación de máquinas de estados finitas y algorítmicas que permitirán representar gráficamente el problema y que representa uno de los puntos ausentes en las metodologías propuestas en las literaturas actuales, con ella se evaluará y editará la solución al problema en VHDL para ello, se emplearan tres casos de estudio para depurar y establecer la metodología general que permita abordar la descripción de cualquier tipo de problema de hardware que se presente.

Los Lenguajes de descripción en hardware surgieron como herramienta de diseño que auxiliaba al ingeniero para integrar un mayor número de dispositivos en un solo circuito integrado. Una de sus principales características radica en su capacidad para describir en distintos niveles de abstracción (funcional, transferencias de registros RTL y lógico o nivel de compuerta) ciertos diseños.

En la actualidad el lenguaje de descripción en hardware más utilizado a nivel industrial es VHDL y por tal motivo no se escapa de ser utilizados en asignaturas de diseño de hardware en las diferentes carreras universitarias como informática, computación, sistemas, electrónica y carreras afines.

Los estudiantes de las carreras citadas, cursan asignaturas de programación básica donde se les enseña una metodología de programación

para abordar cualquier problema planteado. La fase más importante es la del análisis, al finalizar la misma el estudiante debe tener la capacidad de esbozar el problema a través de un algoritmo o diagrama de flujo, previo a la edición y codificación.

Ahora, cuando abordan la programación en lenguajes de descripción en hardware, se encuentra que las mayoría de las metodologías atacan el problema directamente considerando el diseño final del producto e implementando cada uno de los dispositivos bajo las plantillas existentes en la documentación, dejando atrás el aprendizaje adquirido en las asignaturas previas y trabajando por separado cada etapa del diseño, lo que dificulta la documentación del producto final.

Ante este planteamiento se requiere de una metodología integral para el diseño de hardware que agrupe las herramientas que logren describir el problema paso a paso, antes de ser programado. Dentro de esas herramientas se encuentran las Máquinas de estado finitos y las máquinas de estados algorítmicas que permiten modelar la dinámica de un circuito digital, al estilo de los algoritmos o diagramas de flujos en las asignaturas previas y luego si codificar el mismo en cualquier lenguaje de descripción en hardware.

OBJETIVOS DE LA INVESTIGACIÓN

Objetivo General

Proponer el uso de Máquinas de estados Finitos y Máquinas de estados algorítmicas como herramientas didácticas para el desarrollo de programas en Lenguajes de descripción de hardware.

Objetivos Específicos

1. Examinar las bases teóricas de las máquinas de estado finitos y máquinas de estados algorítmicas.
2. Plantear un algoritmo para representar un problema dado en máquinas de estado finitas.
3. Desarrollar un algoritmo para representar un problema dado en máquinas de estado algorítmicas.
4. Elaborar metodología que permita la codificación de la máquina diseñada en Lenguaje de Descripción de Hardware.
5. Validar la metodología de diseño de máquinas propuestas.

Tipo y Modalidad de Investigación

Este trabajo de investigación se sitúa en una perspectiva metodológica de investigación documental definida por Alfonso (1995) como proceso de construcción de conocimientos a partir de un trabajo sistemático de indagación, recolección, organización, análisis e interpretación de información en torno a un tema seleccionado y delimitado. A su vez este trabajo está apoyado de la investigación aplicada donde Lozada (2014) comenta que busca la generación de conocimiento con aplicación directa a los problemas de la sociedad o el sector productivo. Esta se basa fundamentalmente en los hallazgos tecnológicos de la investigación básica, ocupándose del proceso de enlace entre teoría y el producto.

La investigación documental es la parte esencial de un proceso de investigación científica que constituye una estrategia donde se observa y reflexiona sistemáticamente sobre realidades (teóricas o no) usando para ello diferentes tipos de documentos. Indaga, interpreta, presenta datos e informaciones sobre un tema determinado de cualquier ciencia, utilizando resultados que pudiesen ser base para el desarrollo de la creación científica.

Diseño de la Investigación

El diseño metodológico se ajusta a las fase del proceso de la investigación documental: preparatoria, descriptiva, interpretativa y construcción del documento final. A continuación se definen cada una de las etapas estipulando las actividades desarrolladas.

Fase preparatoria:

Esta fase inicial da respuesta a tres intenciones concretas: construir un marco teórico que permita contextualizar la investigación desarrollada, tomar las decisiones en torno al diseño de instrumentos adecuados a los objetivos y problema planteados y reflexionar en torno a la información obtenida.

- Lectura de bibliografía especializada relacionada con el objeto de estudio. Revisión de libros, artículos, investigaciones, etc. Con la finalidad de identificar temas clave de iniciar la construcción de un marco teórico para el informe final de la investigación.
- Conceptualización y elaboración de una base que permita la fundamentación de la investigación.
- Elaboración del primer borrador del marco teórico y del diseño de la investigación.
- Definición del diseño de la investigación, objetivos del estudio, técnicas de obtención de la información, cronología y metodología a utilizar.

En esta fase se abordará específicamente los conceptos de máquinas de estados finitas y algorítmicas desde el punto de vista algorítmica para luego plasmar en la metodología de desarrollo de hardware y permitir evaluar de forma gráfica los cambios en el tiempo que se ejecutan en la misma. Una vez solventada esa documentación se estudiaran diferentes metodologías de desarrollo de hardware existente para producir una metodología que aborde los problemas de manera general a lo particular.

Fase Descriptiva:

Comprende el trabajo de campo que se realiza con el fin de dar cuenta de los diferentes tipos de estudio que se han efectuado sobre el tema y subtemas,

cuáles son sus referentes disciplinares y teóricos. Dentro de éste se establecieron los siguientes pasos: lectura exploratoria de cada uno de los artículos, libros y cualquier otro material obtenido en Internet.

Una vez establecida una metodología de descripción de hardware general se implementaran tres casos de estudio para evaluar las salidas obtenidas y la similitud en las mismas al ser ejecutadas por al menos tres programadores y de esa manera lograr depurar la misma y establecer los cambios necesarios para describir de forma expedita una propuesta idónea para el programador.

Fase Interpretativa:

Permite ampliar el horizonte de estudio por unidad de análisis donde se integraran las máquinas de estados finitos y algorítmicos a la metodología propuesta que cumplirá con el objetivo de dar uso de dichas máquinas en el proceso de descripción de hardware a través de VHDL.

Fase de construcción del documento final.

En esta fase se integrara la documentación, el análisis, las metodologías abordadas en la construcción de la propuesta y se dará difusión de la misma.

Grafico 1. Esquema del diseño de la metodología. Fuente: Materan. (2018)

DESARROLLO DE LA PROPUESTA

Metodología para el desarrollo de código VDHL a través del uso de FSM y ASM

La siguiente metodología es una propuesta a través de la cual planteado un problema que requiere el diseño de circuitos secuencial que pueda generar el código VHDL a través del uso de Máquinas de Estados Finitos y Maquinas de

Estados Algorítmica de manera directa para ello es necesario seguir una serie de etapas los cuales se indican a continuación:

Etapa Nº 1: Análisis del problema planteado

El primer paso es de gran importancia a la hora de diseñar cualquier tipo de circuito, ya que gracias a ese análisis entenderemos y especificaremos como debe ser el funcionamiento del circuito generando así ideas claras sobre lo que verdad se requiere ya que es primordial tener claro cómo debe funcionar nuestro circuito.

Etapa Nº 2: Identificar entradas y salidas:

Después de realizado el análisis se puede determinar cuáles son las señales externas que generan cambios en el funcionamiento del sistema es decir serán nuestras entradas al circuito. De igual manera identificaremos cuáles serán las salidas de nuestro circuito las cuales serían las señales que después de ser procesadas las entradas variaran según los requerimientos de funcionamiento del sistema.

Etapa Nº 3: Elaboración de FSM

El siguiente paso es desarrollar la máquina de estados finitos representándolo mediante un diagrama de estados para visualizar de manera más clara el funcionamiento del sistema y las transiciones entre estados que ocurrirán de acuerdo a los valores que poseen las entradas en un determinado instante para ello es necesario primero seguir los siguientes pasos:

3. aDefinir el número de estados posibles para dicho sistema

En este paso se definen los estados en los que se puede encontrar el sistema de acuerdo a cada una de las entradas posibles y representarlos con palabras claras y sencillas. Por ejemplo podemos identificar los estados de la siguiente manera S0, S1, S2, S3.

3. b Representar las transiciones de estado a través de un diagrama de estado:

Los diagramas de estado son una manera gráfica de representar las FSM lo cual es de gran ayuda a la hora de identificar las transiciones entre los estados del sistema

Los diagramas de estados se componen por círculos los cuales representan dichos estados del sistema y flechas que provienen de un estado a otro indicando la transición entre ellos según los valores de las entradas.

3. c Determinar el número de biestables:

Los biestables son dispositivos de memoria y nos permitirán representar los estados de la maquina en determinado instante el número de biestables que debemos implementar se determina según la cantidad de estados posibles que posea el sistema para lo cual se debe cumplir que

$2^n >=$ N estados

Donde n es el número de biestables y N es el número de estados posibles del sistema, es decir si necesitamos representar 4 estados la ecuación seria $2^n >= 4$ de tal manera n=2 ya que $2^2 = 4$ satisfaciendo la ecuación $2^2 >= 4$

3. d Codificación de los estados:

Para el diseño del circuito necesitamos representar los estados del sistema a través de señales lógicas 0 y 1 para ello se realiza la codificación de estados. Se le asigna a cada uno de los estados una combinación única de bits los cuales serán los valores posibles que podrán tomar los biestables implementados siguiendo el ejemplo anterior la codificación podría ser la siguiente:

Contamos con 4 estados S0, S1, S2, S3 e implementaremos 2 Flip-flop

S0 => 00
S1 => 01
S2 => 10
S3 => 11

3. e Realizar la tabla de estados, entradas y salidas

De donde surge esta tabla bueno proviene del diagrama de estados anteriormente elaborado en ella se representaran cuáles serán los estados siguientes y salidas de acuerdo a las entradas al sistema a partir de cualquiera de los estados posibles, es decir si estamos en el estado S0 cuál será el estado siguiente para cada una de las posibles entradas y así sucesivamente con cada uno de los estados. En dicha tabla se representaran los estados con las

combinaciones de bits anteriormente fijadas para cada uno de ellos para así poder seguir con el siguiente paso.

3. f Tabla de excitación de los biestables

Sabemos que los valores de nuestros biestables son quienes representaran nuestros estados por ello es necesaria saber cuáles deben ser las entradas a los flip-flop para poder pasar de un estado determinado a otro según lo indique nuestra tabla de estados para ello se utilizan las tablas de excitación de los flip-flop.

- Primeros debemos elegir el tipo de biestables a utilizar J-K, D, S-R
- Luego de esto utilizamos las tablas de excitación ya conocidas según el biestable de nuestra elección. Con esta tabla determinaríamos los valores de las entradas de los biestables necesarios para que ocurra un determinado cambio de estado para cada uno de los biestables que utilizaríamos.

3. g Obtención de funciones lógicas a través de mapas k

A partir de la tabla anterior generamos las funciones lógicas tanto para las salidas como para las entradas de los biestables a través de mapas de Karnaught obteniendo así lo necesario para el diseño del circuito requerido.

3. h Diseño del circuito

El último paso para la elaboración de la FSM es el diseño del circuito como tal el cual se genera gracias a las funciones lógicas anteriormente generadas. **Etapa Nº 4: Pasar de la FSM a una ASM:**

Las ASM o máquina de estados algorítmico es otra manera de representar las FSM de manera más clara y precisa ya que define de mejor manera las condiciones necesarias para pasar de un estado a otro y de igual manera representa las acciones que se realizan en cada estado acercándonos así más a nuestro propósito el cual es generar el código en VHDl.

Como ya hemos elaborado nuestro diagrama de estados es más sencillo pasar a ASM solo debemos tomar en cuenta los siguientes aspectos:

- Los estados en FSM son representados con círculos y en ASM se representan con rectángulos los cuales son llamados cajas de estados
- Las transiciones entre estados son especificadas de manera más precisa estableciendo condiciones las cuales se representan con rombos

- Las salidas de tipo Mealy son representadas con rectángulos con bordes redondeados
- Los bloques ASM se ejecutan en un ciclo de reloj.

Aclarado esto podemos proceder a ejecutar los siguientes pasos los cuales nos ayudaran a definir nuestra ASM:

4. a Realizar la descripción del funcionamiento del circuito a través de un Pseudocódigo

En este paso se describe el funcionamiento del circuito a través de un pseudocódigo de manera secuencial ya que gracias a que el tipo de lenguaje de Pseudocódigo es más claro y comprensible hace que sea más fácil la representación de condiciones y pasos lo cual ayudara a la hora de la representación en el diagrama ASM

4. b Identificar condiciones necesarias para las diferentes transiciones entre estados:

A partir del diagrama de estados podemos identificar qué condiciones se deben cumplir para poder pasar de un estado a otro, esto se refleja en las transiciones entre los mismos ya que para pasar de un estado a otro los valores de las entradas deben poseer un valor especifico a esto se le llama condiciones de transición.

4. c Determinar acciones:

Tenemos claro que en cada uno de los estados las salida deben tener un valor determinado sabiendo esto debemos tener en cuenta que al pasar de un estado a otro estos valores varían, al cambio de estos valores se les denomina acciones las cuales se representan en las cajas de estados. En otras palabras las acciones son las asignaciones de valores a las variables que nos representan las salidas del sistema.

4. d Elaboración del diagrama ASM

Teniendo claro las transiciones entre estados, las condiciones y acciones para cada una de los estados podemos realizar el diagrama ASM a partir del diagrama de estados de la FSM.

En el diagrama ASM se representa la ruta a seguir para pasar de un estado a otro según se cumplan o no las condiciones anteriormente descritas.

Ejemplo para pasar del estado S0 a S1 la entrada debe ser X=1

Figura 32. FSM llevada a ASM. Fuente: Materan. (2018)

Etapa Nº 5: Declaración de la entidad en VHDL

En el lenguaje de descripción de hardware VHDL se definen dos componentes primordiales uno de ellos es la entidad en la cual se establecen o declaran las entradas y salidas del sistema anteriormente definidas en el paso Nº 2

```
ENTITY nombre-entidad IS

  PORT  (nombre_del_puerto-1:  direccion_de_señal  tipo_de_dato;
         nombre_del_puerto-2:  direccion_de_señal  tipo_de_dato;
         nombre_del_puerto-n: direccion_de_señal tipo_de_dato;)

END nombre-entidad;
```

El nombre entidad no tiene mucho que ver con la FSM que se trabaja, y no necesariamente tiene que estar relacionado con ella.

El nombre del puerto es el identificativo para las entradas y salidas.

Cuando todas las entradas y salidas son bits individuales se puede usar el tipo de dato **std_logic**.

La dirección de señal puede ser **IN** o **OUT**, dependiendo si se trata de una entrada o una salida.

Cuando se es dependiente de un ciclo de reloj, se define este puerto de la siguiente forma:

Clock: IN STD_LOGIC;

Para retornar a un estado inicial, se define el puerto de la siguiente forma:

reset: IN STD_LOGIC;

Etapa Nº 6:Declaración de la arquitectura

Este es el segundo componente en VHDL en el se declaran los estados de nuestra ASM además de variables y señales internas que podamos utilizar para definir el funcionamiento esperado lo cual es la segunda parte de la arquitectura del circuito como tal la cual está conformada por varios procesos o sentencias concurrentes según sean necesarios para el funcionamiento del mismo puede contener tantos procesos como sean necesarios en esta parte

ARCHITECTURE nombre de la arquitectura OF nombre-entidad IS

[declaraciones]

BEGING

[sentencias concurrentes]

END [nombre de la arquitectura];

Las declaraciones de estados: aquí se definen los estados y se crean las señales que tendrá un estado definido como su valor además se podrán declarar señales internas necesarias

Ejemplo:

```
SIGNAL señales internas a utilizar

TYPE Tipo_De_Estado IS (A, B, C, D);

SIGNAL Estado: Tipo_De_Estado;
```

Donde A, B, C y D representan los estados que contiene el diagrama.

Etapa Nº 7: Definición de los procesos o sentencias concurrentes

Los procesos o sentencias concurrentes son la parte más importante del código VHDL ya que en ellos es que se especifica detalladamente las operaciones, acciones y condiciones a evaluar para el funcionamiento de nuestro circuito nuestra metodología genera los procesos a partir de nuestra ASM ya que en ella se encuentran especificadas las acciones y condiciones necesarias para los cambios de estado que en si sería el funcionamiento del circuito

Se realiza siguiendo el siguiente formato:

```
PROCESS (lista_de_sensibilidad)
BEGIN
        [Configurar_estado_incial]
        [Transiciones_de_estados]
END PROCESS;
 [Instrucciones_de_salida]
```

En la arquitectura, Cuando un proceso VHDL se configura con señales de reloj y reinicio, este se plantea de la siguiente forma:

```
PROCESS (clock, reset)
 BEGIN
   IF (reset = '1') THEN
        State <= A;
```

Configuración de estado inicial

ELSIF elsif(clk'event and clk='1')THEN

Configuración del reloj

La Transiciones de estados se realiza por medio del siguiente además en ellas se puede realizar la asignación de las salidas dependiendo de los estados en los que se encuentre en un determinado instante para ello se utiliza el siguiente formato:

CASE Estado IS La sentencia siguiente dependerá del estado actual

WHEN Estado_n =>

```
IF Entrada='1' THEN
Asignacion de salidas
Estado <= Estado_siguiente1;
ELSE
Asignacion de salidas
Estado <= Estado_siguiente2;
END IF;
```

Esta sentencia se coloca para los n números de estados que se tengan en el diagrama ASM, Tomando en cuenta su Transición a los siguientes estados con respecto a su entrad

WHEN others =>

```
Asignacion de salidas
Estado <= Estado incial;
```

Esta sentencia se utiliza para restablecer el valor de A en caso de que el estado no obtenga ningún valor anterior

END CASE;

```
ENTITYnombre-entidadIS

PORT (nombre_del_puerto-1: direccion_de_señaltipo_de_dato;
        nombre_del_puerto-2: direccion_de_señaltipo_de_dato;
        nombre_del_puerto-n: direccion_de_señaltipo_de_dato;)
ENDnombre-entidad;

ARCHITECTUREnombre de la arquitectura OFnombre-entidadIS

SIGNAL señales interneas a utilizar

TYPE Tipo_De_Estado IS (A, B, C, D);

SIGNAL Estado: Tipo_De_Estado;
BEGING
        PROCESS (lista_de_sensibilidad)

BEGIN
 IF(reset = '1')THEN
        State <= A;
        ELSIF elsif(clk'event and clk='1')THEN

CASE Estado IS  La sentencia siguiente dependerá del estado actual


WHEN Estado_n =>


                IF Entrada='1' THEN
                                Asignacion de salidas
                        Estado <= Estado_siguiente1;
                ELSE
                        Asignacion de salidas
                        Estado <= Estado_siguiente2;
                END IF;
WHEN others =>
                        Asignacion de salidas
                        Estado <= Estado inicial;


END CASE;
END PROCESS;
END [nombre de la arquitectura];

END CASE;
END PROCESS;
END [nombre de la arquitectura];
```

Figura 33.Código para definir componentes en VHDL: Fuente Materan. (2018)

Grafico 2. Esquema general de la metodología para programar en lenguajes de descripción de hardware (VHDL) a través del uso de las FSM y ASM. Fuente: Materan, 2018.

Metodología para programar en lenguajes de descripción de hardware (VHDL) mediante del uso de las FSM y ASM.

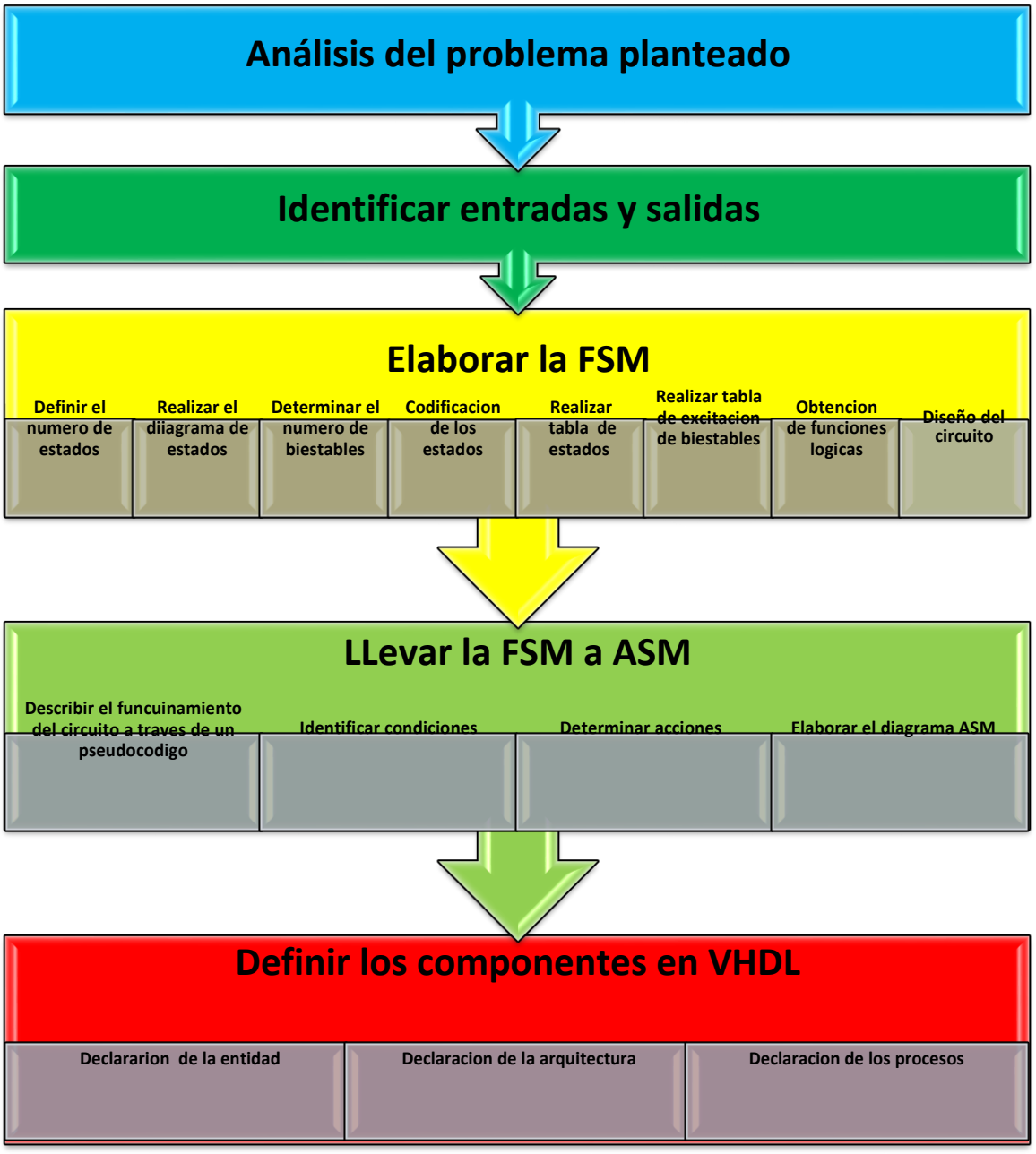

Grafico3. Metodología para programar en lenguajes de descripción de hardware (VHDL) mediante el uso de las FSM y ASM. Fuente: Materan. 2018

CONCLUSIONES

Las instituciones requieren de metodologías didácticas que le brinden herramientas al estudiante a la hora de resolver cualquier tipo de problema planteado para soluciones exitosas, preparándolos así para su desarrollo profesional en el ámbito laboral.

Para finalizar con el presente trabajo de investigación, se concluye principalmente dando respuesta al objetivo número uno (1), dos (2) y tres (3), enmarcado en la investigación; que es posible diseñar algoritmos que faciliten resolver cualquier tipo de problema mediante esta herramienta de carácter didáctico.

Todas las fases de estudio permitieron desarrollar esta metodología para los estudiantes de la asignatura arquitectura al computador en la carrera de ingeniería en sistemas de la Universidad de los Andes (NURR), facilitando el desarrollo de código en los lenguajes de descripción de hardware, es de importancia señalar que la metodología desarrollada se puede adaptar a cualquier tipo de problema planteado.

Culminado este proceso se han cumplido con todos los objetivos planteados en esta investigación cuyo resultado fue una metodología de diseño que representa una verdadera herramienta didáctica que le facilite al estudiante la tarea a la hora de enfrentarse con cualquier tipo de problema.

:

REFERENCIAS BIBLIOGRÁFICAS

Arias, F. (2004). El proyecto de Investigación. 4ta Edición. Carcas, Venezuela.

Cáceres, S., De Pablo, S., Cebrián, J.A., Sanz, F., Berrocal, M. (2015). Los diagramas ASM++ como herramienta aplicada en la enseñanza de la electrónica digital. Departamento de Tecnología Electrónica, Universidad de Valladolid, Valladolid, España.

Cassanovas, M. (2014). Máquinas de estados finitas. Centro Universitario de
Desarrollo de Automatización y Robótica. Córdoba, Argentina.

Gutiérrez, J. (2008). Máquinas de estados finitos. Escuela Superior de Computo. México

Holguín, M. Orozco, A., Escobar, A., (2011). Metodología para al diseño de autómatas finitos con salidas en lenguaje ladder bajo el estándar IEC611131-3. Universidad Tecnológica de Pereira, Colombia.

Lara, E. (2002). Diseño de Sistemas Digitales con Lógica Programable.
Universidad Autónoma de Nuevo León. México.

Lozada, J. (2014) Investigación Aplicada Definición, Propiedad Intelectual e Industria. Centro de investigación en Mecatronica y Sistemas interactivos, Universidad tecnológica Indoamericana. Quito, Ecuador.

Martin, B. (2006). Maquinas Algorítmicas como opción didáctica de sistemas digitales complejos. Universidad de Zaragoza.

Maxinez, D. y Alcalá J. (2002). El arte de programar sistemas digitales. Instituto tecnológico y de Estudios Superiores de Monterrey Campus, Estado de México.

Morales, L.; (2014). Propuesta de ejemplos integradores para la asignatura Digital VLSI, Universidad Central “Marta Abreu de las Villas, Santa Clara, Cuba.

Morris M, (1987). Diseño Digital, Primera Edición. Prentice Hall, México.

Muñoz, G (2002). Método simple para pasar de un algoritmo a un m