https:/doi.org/10.24275/uam/lerma/repinst/ciai2018/000172/Laguna

Propuesta de arquitectura para la realización de una unidad central de proceso a partir de una máquina de Post

Gerardo Abel Laguna Sánchez
Universidad Autónoma Metropolitana, Unidad Lerma
Área de Sistemas de Información y Ciencias Computacionales
g.laguna@correo.ler.uam.mx

Jacobo Sandoval Gutiérrez
Universidad Autónoma Metropolitana, Unidad Lerma
Área de Sistemas de Información y Ciencias Computacionales
j.sandoval@correo.ler.uam.mx

Resumen

En este artículo, se presenta una propuesta de arquitectura para la realización práctica de un CPU a partir de una variante de la máquina de Post que, dada su relativa simplicidad, permite introducir de manera clara y concreta los conceptos básicos relativos al funcionamiento de todo procesador electrónico. El diseño lógico del sistema digital se desarrolló mediante la metodología RTL, se codificó con lenguaje VHDL y se probó con los recursos de una tarjeta de desarrollo FPGA de bajo costo. Se demostró que es posible construir, con recursos relativamente limitados, un sistema micro-procesado que permite adquirir el conocimiento y la experiencia necesaria para incursionar en el ámbito del diseño y realización de un microprocesador básico mediante los recursos disponibles en los actuales sistemas digitales completamente programables.

Palabras Claves: Arquitectura de CPU, Máquina de Post, Metodología RTL, Micro-código.

Abstract

In this paper, a simple CPU architecture is proposed. The proposal is derived from a Post machine variant. This work provides insight about the basic concepts concerning the electronic micro-processor operation in a clear and straightforward way. The logic design was developed by means the RTL methodology, coded with VHDL and tested on a cheap FPGA evaluation board. It is showed that is possible to build a practical micro-processor with relatively low resources. The approach on this work contributes with an illustrative example to design and build a basic micro-processor with the resources of the current programmable digital systems.

Keywords: *CPU architecture, Micro-code, Post Machine, RTL Methodology.*

1. Introducción

El poder computacional de los actuales microprocesadores, así como de los sistemas digitales relacionados, es superior en miles de veces respecto de sus primeros antecesores, tanto en velocidad como en complejidad. Podemos reconocer que los inicios de la computación moderna se remontan a la conceptualización de la "máquina analítica" de Charles Babbage y Ada Lovelace, en el siglo XIX, así como a la formulación de las "máquinas" computadoras de Alan Turing y Emil Post, en la década de 1930. Así mismo, en forma paralela a los trabajos de Turing y Post, se construyeron las primeras computadoras, por ejemplo las de John von Neumann, en la década de 1940. Sin embargo, no obstante que el avance tecnológico ha sido inmenso, los conceptos básicos de la computación introducidos por Babbage, Lovelace, Turing y Post siguen siendo vigentes en la actualidad. Por otro lado, debido precisamente al impresionante avance de la industria de los semiconductores, es preciso reconocer que en la actualidad se cuenta con tal cantidad de recursos tecnológicos, tanto en el ámbito de los componentes digitales como en el del diseño lógico, que puede resultar abrumador introducir a las nuevas generaciones de ingenieros en los campos de los sistemas digitales y micro-procesados.

El reto implica una decisión difícil: ¿Qué debe permanecer en contenido curricular y qué debe salir? Por una parte, el avance tecnológico es rápido e inaplazable. No presentar a los alumnos las últimas tendencias tecnológicas puede resultar en una obsolescencia práctica aún antes de que egresen. Por otra parte, los conceptos básicos no pueden dejar de impartirse. Se corre el riesgo de formar ingenieros que no van más allá de ser usuarios calificados de las tecnologías, sin conocimiento de los principios que rigen su funcionamiento interno. Esta situación se presenta con particular gravedad en el caso de la impartición de los cursos de diseño lógico avanzado y sistemas micro-procesados, donde los conceptos de algoritmo y unidad central de procesos son básicos.

En este trabajo se presenta una propuesta de arquitectura para un micro-procesador, de relativa simpleza conceptual y facilidad para su realización práctica, que permite conciliar de manera natural dos polos, opuestos aunque complementarios, de una misma realidad: la aplicación de los conceptos básicos de la computación, mediante la más reciente tecnología disponible, y una efectiva apropiación del conocimiento, garantizando que el alumno incursione en la construcción de su propio micro-procesador aprovechando los recursos digitales y de lógica programable con la que contamos en la actualidad.

La arquitectura propuesta en este trabajo parte del trabajo de Emil L. Post [Post, 1936], matemático estadounidense, que en la década de 1930 desarrolló, simultáneamente pero en forma independiente, un trabajo similar al del célebre matemático inglés Alan Turing [Turing, 1937] en lo referente a la conceptualización de una "máquina" computadora. En este trabajo se ha optado por la "máquina" de Post y no por la "máquina" de Turing, debido a la menor complejidad de la primera respecto de la segunda. Específicamente, el conjunto de instrucciones de la

máquina de Post es menor al de la máquina de Turing, y sin embargo ambas pueden considerarse computadoras de propósito general que pueden ejecutar programas para problemas computables.

En lo subsecuente se desarrollarán las siguientes secciones: en la sección 2 se presenta la descripción y operación básica de una máquina de Post; en la sección 3 se presenta la especificación y el diseño de la arquitectura propuesta, así como un bosquejo de su representación funcional mediante la metodología RTL (por la siglas en inglés de *Register Transfer Level*); en la sección 4 se presentan los detalles de la realización práctica de la arquitectura propuesta, así como la discusión de los resultados; finalmente, en la sección 5, se presentan las conclusiones.

2. La "máquina" de Post original y propuesta de una variante

En los trabajos originales de Emil L. Post y de Alan Turing no se especifica ninguna máquina real, se trata del desarrollo de conceptualizaciones y experimentos mentales. Es por ello que en este trabajo se usa el término máquina entre comillas cuando se refiere a las propuestas de Post y de Turing. En ambas propuestas se bosquejan los alcances y limitaciones de todo algoritmo. No obstante lo antes dicho, es perfectamente posible especificar una máquina real a partir de los trabajos de Post y Turing. De hecho, así lo hizo el matemático ruso V.A. Uspenski [Uspenski, 1983], quien específicamente vislumbró el potencial de la propuesta de Post con fines pedagógicos.

Para empezar, conviene señalar que el ejercicio que propone Post se sustenta en la manipulación de un conjunto infinito de casillas o cajas (que Uspenski representa mediante una cinta con un número infinito de celdas), ordenadas, numeradas y que pueden estar marcadas o no. También existe un "operador" que sigue una secuencia de instrucciones y, con base en estas y en el contenido de las casillas, puede modificar el estado de las casillas para, dado el caso, concluir la secuencia de instrucciones y dejar al conjunto de casillas con un estado diferente al inicial.

Ahora, se puede especificar el funcionamiento de la máquina de Post, tomado como referencia básica el trabajo de Uspenski [Uspenski, 1983]. El "hardware" de la máquina de Post estaría conformado por lo siguiente:

a) Una cinta con celdas y un carro o cabezal de lectura/escritura. Ver la figura 1.
b) Para ordenar las celdas en la cinta se usan números enteros.
c) Las celdas pueden estar marcadas o no. En la figura 1, se marcan mediante una 'x'.
d) El estado de la cinta lo constituye la información que indica cuáles celdas están marcadas y cuáles no. Esta información puede cambiar con el desarrollo del funcionamiento de la máquina.
e) Cada cierto tiempo (periodo o paso), el carro se puede mover, a la derecha o a la izquierda, y ello permite leer o escribir el estado de una celda.

f) El estado de la máquina de Post lo constituye la información que indica el estado de la cinta y la posición del carro.

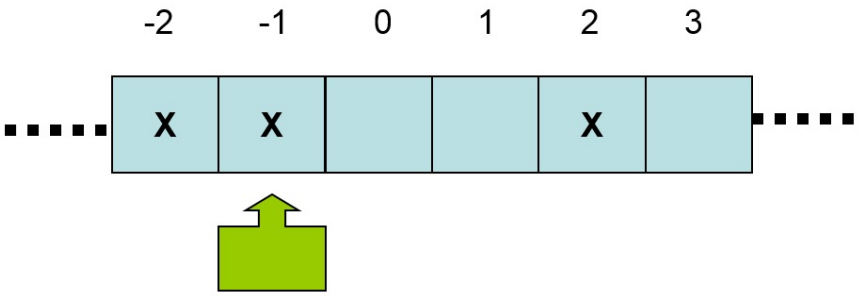

Figura 1. "Hardware" para la máquina de Post.

Por otra parte, el "software" de la máquina de Post estaría definido para operar como sigue:

a) La secuencia de instrucciones que determina el movimiento del carro, así como la lectura y escritura de las celdas, para marcarlas o borrarlas, se denomina programa.
b) Cada instrucción tiene asociado un número natural, para indicar su posición en la secuencia de instrucciones, que se denomina *número de la instrucción* i.
c) Finalmente, cada instrucción también especifica el *número de la siguiente instrucción* j. Cuando la instrucción es condicional (control de flujo) contiene dos posibles índices *j* para la siguiente instrucción: j_1 y j_2.
d) El conjunto de instrucciones contempla seis operaciones:
   - Movimiento a la derecha, que denotaremos con la sintaxis `i.rmov,j`
   - Movimiento a la izquierda, que denotaremos con la sintaxis `i.lmov,j`
   - Impresión de marca, que denotaremos con la sintaxis `i.print,j`
   - Borrado de marca, que denotaremos con la sintaxis `i.clr,j`
   - Salto condicional, que denotaremos con la sintaxis `i.ifempty j1 else j2`
   - Parada, que denotaremos con la sintaxis `i.stop`

En la Máquina de Post claramente se diferencia el espacio para código (instrucciones) del espacio para los datos (cinta), por lo tanto, se trata de una computadora con arquitectura *Harvard*. Como en toda computadora, es necesario prefijar el programa y también el estado inicial de los datos (el estado inicial de la cinta). En cuanto a la inicialización de la máquina de Post, se sugiere que el carro siempre inicie en la posición de la celda No. 0, a la vez que el apuntador de instrucciones señala a la instrucción No. 1. Se puede resumir el funcionamiento de la máquina de Post como siguiente:

- La máquina arranca en el estado inicial y ejecuta la primera instrucción, la No. 1.
- En cada paso se ejecuta una instrucción.

- Después de ejecutada la instrucción i, a continuación se ejecuta la instrucción j especificada por la misma instrucción i.
- Se repite la ejecución de instrucciones, hasta encontrar la instrucción de parada (`stop`) o una instrucción no definida.

Aunque la máquina de Post original no permite borrar celdas vacías o sobre-escribir en celdas marcadas, en este caso, se relajará esta restricción para permitirlo sin que ello implique algún problema.

A fin de construir un sistema digital para emular el funcionamiento de una máquina de Post, en este trabajo se propone realizar algunos ajustes a la especificación de Uspenski. A esta variante se le denominará *Máquina de Post Modificada* o MPM. En primer lugar, la cinta con celdas se remplaza por una memoria con direccionamiento para N localidades de 1 bit (cada localidad de datos puede contener un cero o un uno), como se puede apreciar en la figura 2. En esta nueva representación, una celda corresponde a una localidad de datos mientras que el carro corresponde al apuntador de datos, es decir al registro *data poiter* (DP) que señala la dirección de la localidad de datos en turno. En segundo lugar, a cada localidad de datos se le asocia a una dirección con un entero no negativo. Así, la primera localidad tiene la dirección 0 y la última corresponde a N-1. El número de localidades es finito, pero se solventa esta restricción diseñando el espacio de datos como un buffer circular. Cuando el DP rebasa la última dirección, regresa a la dirección 0.

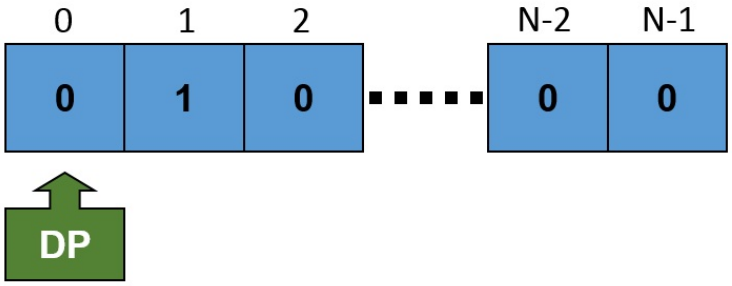

Figura 2. Hardware para la máquina de Post modificada (MPM).

Respecto de la ejecución de los programas, se tienen en cuenta las siguientes consideraciones:

a) El código lo constituye la secuencia de instrucciones que operan sobre las localidades de datos, indicadas por el registro DP, realizando la lectura o la escritura de las mismas con unos y ceros.
b) Cada instrucción tiene asociada una dirección con un número entero no negativo, indicando su posición dentro de la secuencia de instrucciones, que se denomina dirección del código i.
c) Cada instrucción se ejecuta en orden, empezando en la dirección 0 y continuando en forma ascendente, excepto cuando aparece una instrucción de salto, en cuyo caso se especifica la dirección de la siguiente instrucción

j. El apuntador que indica la instrucción por ejecutar se conoce como registro *instruction pointer* (IP).

d) En la variante propuesta, el conjunto de instrucciones contempla siete operaciones:
   - Incremento del DP, con la sintaxis `i.incdp`
   - Decremento del DP, con la sintaxis `i.decdp`
   - Escritura de uno, con la sintaxis `i.set`
   - Escritura de cero, con la sintaxis `i.clr`
   - Salto, con la sintaxis `i.jmp j`
   - Salto condicional, si la localidad contiene un cero, con la sintaxis `i.jz j`
   - Parada, con la sintaxis, `i.stop`

En esta propuesta de máquina de Post modificada, el programa opera sobre las localidades de la memoria de datos que contienen unos y ceros. Así, los valores iniciales de las localidades en la memoria constituyen los datos de entrada, mientras que los valores finales constituyen los datos de salida.

3. Arquitectura propuesta para la Máquina de Post Modificada

Dado que el diseño lógico propuesto para la MPM contempla un registro apuntador a código, o *instruction pointer* (IP), y un registro apuntador a datos, o *data pointer* (DP), la arquitectura de la máquina de Post propuesta es esencialmente del tipo Harvard. Las siete instrucciones previamente definidas para la MPM más un código de no-operación (`nop`), a fin de poder programar retardos, resulta en un total de ocho instrucciones para la arquitectura final. Dado que el conjunto de instrucciones es menor a 16 operaciones, se pueden codificar perfectamente mediante un *nibble* o 4 bits. En el caso particular de las instrucciones de salto, `jmp` y `jz`, cada código de instrucción se acompaña de dos *nibbles* adicionales con la información de direccionamiento absoluto, a saber, un primer *nibble* con los bits más significativos (MSB) y un segundo *nibble* con los bits menos significativos (LSB). En la tabla 1 se muestra un resumen para el conjunto de instrucciones especificado, así como la codificación propuesta.

Con base en la especificación del conjunto de instrucciones, es factible proponer una arquitectura específica para el diseño lógico del sistema digital que conformará a la unidad central de procesamiento (CPU) y que permitirá emular la operación de la máquina de Post modificada. Dado que el conjunto de instrucciones no incluye a ninguna operación aritmética y tan solo cuenta con una operación de comparación, para el caso del salto condicional, es por ello que el CPU esencialmente se conforma por el registro para el direccionamiento de código, el registro para el direccionamiento de datos, un registro para la información relativa a la instrucción que se va a ejecutar, una unidad lógica para la función de comparación y los puertos de entrada/salida para las señales de código, datos y control, mediante los cuales el CPU se conectará con los dispositivos de memoria que requiere. Adicionalmente, se pueden incluir puertos de salida para señalizar el estado de la máquina.

Tabla 1. Instrucciones y códigos para la MPM.

Instrucción	Nemónico	Nibble 1 (código)	Nibble 2 MSB	Nibble 3 LSB
No operación	`nop`	0		
Incremento del DP	`incdp`	1		
Decremento del DP	`decdp`	2		
Escritura de uno	`set`	3		
Escritura de cero	`clr`	4		
Salto	`jmp`	5	Add 7-4	Add 3-0
Salto condicional	`jz`	6	Add 7-4	Add 3-0
Parada	`stop`	7		

En el diagrama de la figura 3 del Anexo, se presenta una propuesta específica de arquitectura considerando que el CPU se sintetizará y construirá empleando un FPGA Artix-7 de marca Xilinx y que se conectará con bloques IP Xilinx (por las siglas de *Intelectual Property*) de memoria RAM y ROM síncronos. Específicamente, los bloques de memoria son del tipo nativo, en modo de operación *write first*, y son generados por la herramienta *Block Memory Generator* de Xilinx [Xilinx, 2017].

Partiendo de la arquitectura propuesta en la figura 3 del Anexo, de la especificación de la operación y funcionamiento de la máquina de Post modificada, así como de los diagramas de tiempo para la operación de los bloques IP de memoria síncrona, se procedió al desarrollo del diseño lógico mediante la metodología de diseño RTL [Chu, 2008] y diagramas ASMD (acrónimo de *Algorithmic State Machine with a Data path*), del circuito secuencial que realiza el procesamiento y la ejecución del micro-código, que soporta al ciclo *fetch-decode-execute* del CPU, así como el control de las señales para la lectura/escritura del código y de los datos. En la figura 4 del Anexo, se presenta un fragmento del diagrama ASMD, donde se puede apreciar la secuencia de micro-código para la instrucción `incdp`. Finalmente, se muestra la codificación del módulo CPU, usando el lenguaje de descripción de hardware VHDL, así como a la creación de un proyecto HDL para la síntesis, construcción y prueba del mismo en una tarjeta de desarrollo.

4. Realización práctica y discusión de los resultados

Para la síntesis, implementación y prueba de la arquitectura propuesta se empleó el entorno Vivado de Xilinx y una tarjeta de desarrollo Basys 3, de marca Digilent, que contiene un FPGA Artix-7 de Xilinx (XC7A35T-1CPG236C) que, a su vez, cuenta con 33,280 celdas lógicas, distribuidas en 5200 secciones, cada una

con cuatro tablas LTU de 6 entradas y 8 *flip-flops* (FF) [Digilent, 2017]. Adicionalmente, el FPGA incluye 1,800 Kbits, en un bloque de RAM rápida, cinco módulos para la gestión de las señales de reloj y servicios PLL, así como 90 secciones para DSP, entre otros recursos. Por otra parte, la tarjeta de desarrollo Basys 3 incluye una señal de reloj de 100MHz, 16 interruptores (SW), cinco botones (BTN), 16 LED y un módulo de despliegue LED, con cuatro caracteres de 7 segmentos, entre otros recursos de memoria y dispositivos de entrada/salida (IO) que no se han empleado en este proyecto.

A fin de poder acceder al contenido de los datos, antes y después de la ejecución de los programas para la MPM, se realizaron ligeras adiciones a la arquitectura presentada en la figura 3 del Anexo. En esencia, se *multiplexaron* tanto las señales de datos como las de control para contar con dos modos de operación: uno de ejecución y otro para acceder manualmente al contenido de la memoria RAM. Para la carga del código y de los datos, respectivamente en la memoria ROM y en la memoria RAM, se emplearon archivos de inicialización **.coe* que se incorporaron durante la etapa de implementación de los bloques IP correspondientes.

Una vez concluida la síntesis y construcción del proyecto VHDL, el reporte post-implementación dentro del entorno Vivado arrojó las siguientes estadísticas de aprovechamiento de los recursos del FPGA XC7A35T-1CPG236C: 1% de la LUT, 1% de los FF, 2% de la BRAM, 44% de los puertos de IO, y 3% de los BUFG, con lo que se demuestra que el proyecto pudo realizarse con un mínimo de recursos. Se probó el funcionamiento del CPU MPM precargando la ROM con diversos programas que, a su vez, fueron ejecutados con diferentes configuraciones de datos precargados en la RAM. Se pudo comprobar, en forma práctica, que el funcionamiento de la arquitectura propuesta para el CPU, así como del sistema resultante, es el esperado conforme a la especificación para la máquina de Post modificada.

En cuanto a la aplicación del sistema obtenido en el proceso de enseñanza-aprendizaje, vale la pena recordar que, tradicionalmente, se inicia un curso de sistemas micro-procesados con la descripción de una arquitectura de referencia que puede o no estar relacionada con la arquitectura de un dispositivo comercial específico. Sin embargo, al momento de desarrollar la componente práctica del curso, se hace necesario adoptar alguna familia de dispositivos comerciales, a fin de aplicar los conceptos adquiridos en una plataforma de hardware concreta. Es ahí cuando surge la necesidad de manejar un lenguaje de programación, que puede ir desde lenguajes de bajo nivel, como lo es el ensamblador, hasta el uso de lenguajes de mediano o alto nivel, como por ejemplo C y sus variantes. Desafortunadamente, mientras mayor es el nivel del lenguaje de programación, mayor el distanciamiento entre el programador y el hardware subyacente. Por ello, la propuesta que en este trabajo proporciona, en forma natural, un eslabón que da continuidad a un curso de diseño lógico avanzado y uno de sistemas micro-procesados. Con ello se proporciona una herramienta para el diseño de un CPU

que, aunque elemental, sirve como punto de partida para comprender arquitecturas mucho más sofisticadas. La versión de la MPM presentada en este trabajo ya ha sido incluida como uno de los bancos de prueba para las prácticas de laboratorio de la UEA Diseño Lógico Avanzado, del Plan de la licenciatura en Ingeniería y Computación de la Unidad Lerma y se espera que pronto se cuente con evidencia de las ventajas que la propuesta pudiera aportar al proceso de enseñanza aprendizaje.

5. Conclusiones

En este trabajo, se ha bosquejado una arquitectura mínima para la realización práctica de un CPU con base en la especificación de una máquina de Post modificada. La máquina de Post es similar a la de Turing, pero con un set de instrucciones más compacto, lo que permite disponer de una computadora de propósito general para la ejecución de programas para todo problema computable. Si bien es cierto que un conjunto de instrucciones más reducido implica códigos más largos, también es cierto que lo más simple es muy apropiado para ilustrar conceptos fundamentales dentro de un proceso de enseñanza-aprendizaje. La arquitectura propuesta fue desarrollada mediante diagramas ASMD y descrita mediante código VHDL. Finalmente, la propuesta fue sintetizada e implementada mediante la plataforma Vivado de Xilinx y una tarjeta de desarrollo Basys 3, que cuenta con un FPGA de la familia Artix-7. El sistema obtenido fue probado y su desempeño fue conforme con lo especificado. Con ello, ahora se cuenta con un banco de prueba para que los alumnos de la licenciatura de Ingeniería y Computación de la Unidad Lerma apliquen los conceptos del diseño lógico avanzado e incursionen exitosamente en el diseño y aplicación de sistemas micro-procesados.

6. Bibliografía y Referencias

[Chu, 2008] Chu, Pong P. FPGA Prototyping by VHDL Examples. Wiley-Interscience. EUA, 2008.

[Digilent, 2017] Digilent. Basys 3 FPGA Board Reference Manual (DOC# 502-183). EUA, 2017.

[Post, 1936] Post, E.L. Finite combinatory processes – formulation 1. The Journal of Symbolic Logic. Vol. 1, No. 3. 1936.

[Turing, 1937] Turing, A.M. On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society. Vol. S2-42, No. 1, pp. 230-265. 1937.

[Uspenski, 1983] Uspenski, V.A. Máquina de Post. Lecciones populares de matemáticas. Editorial MIR. Moscú, 1983.

[Xilinx, 2017] Xilinx. Block Memory Generator v8.3: LogiCORE IP Product Guide (PG058). EUA, 2017.

Anexo

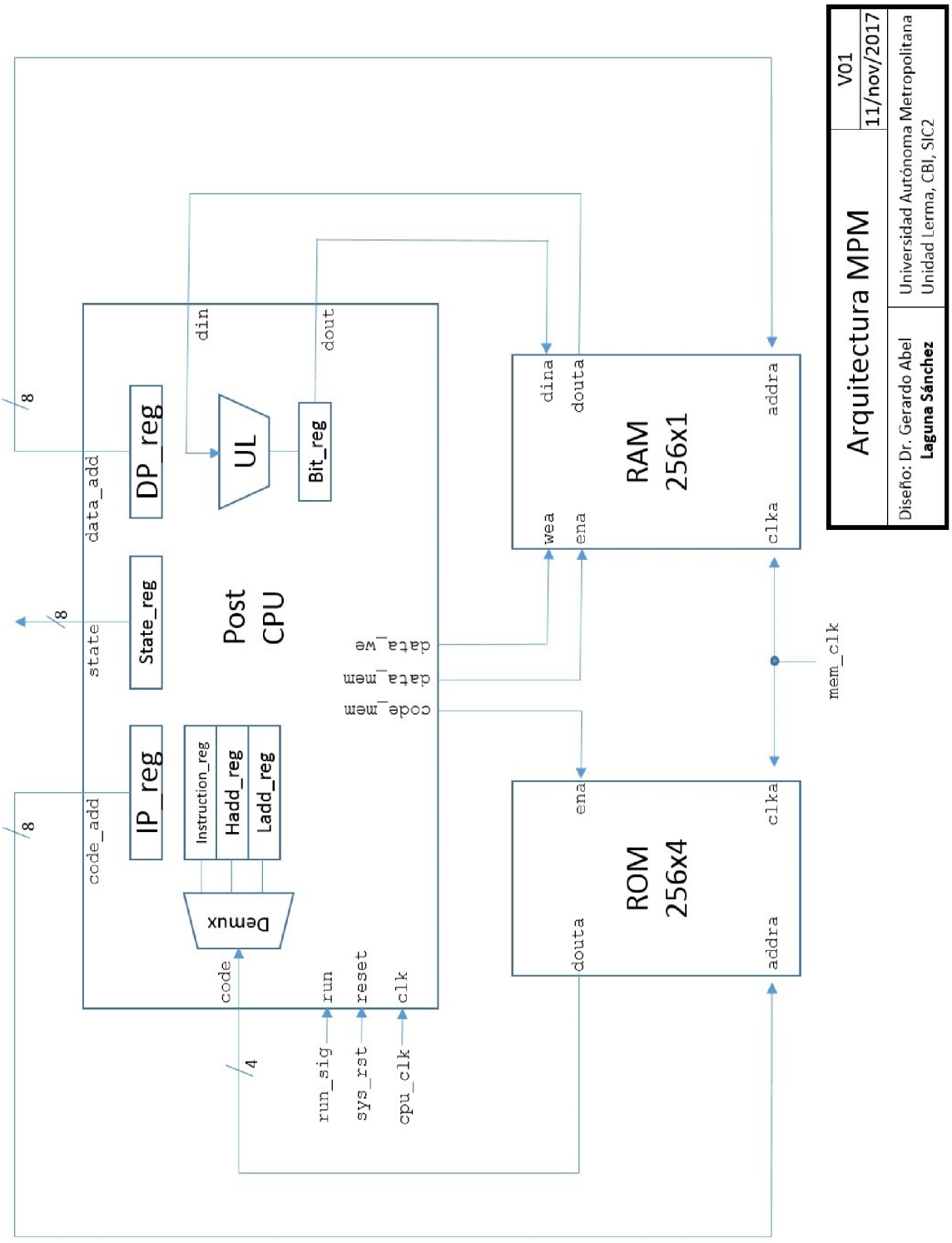

Figura 3. Diagrama a bloques para la arquitectura propuesta para una la realización práctica de la máquina de Post modificada (MPM).

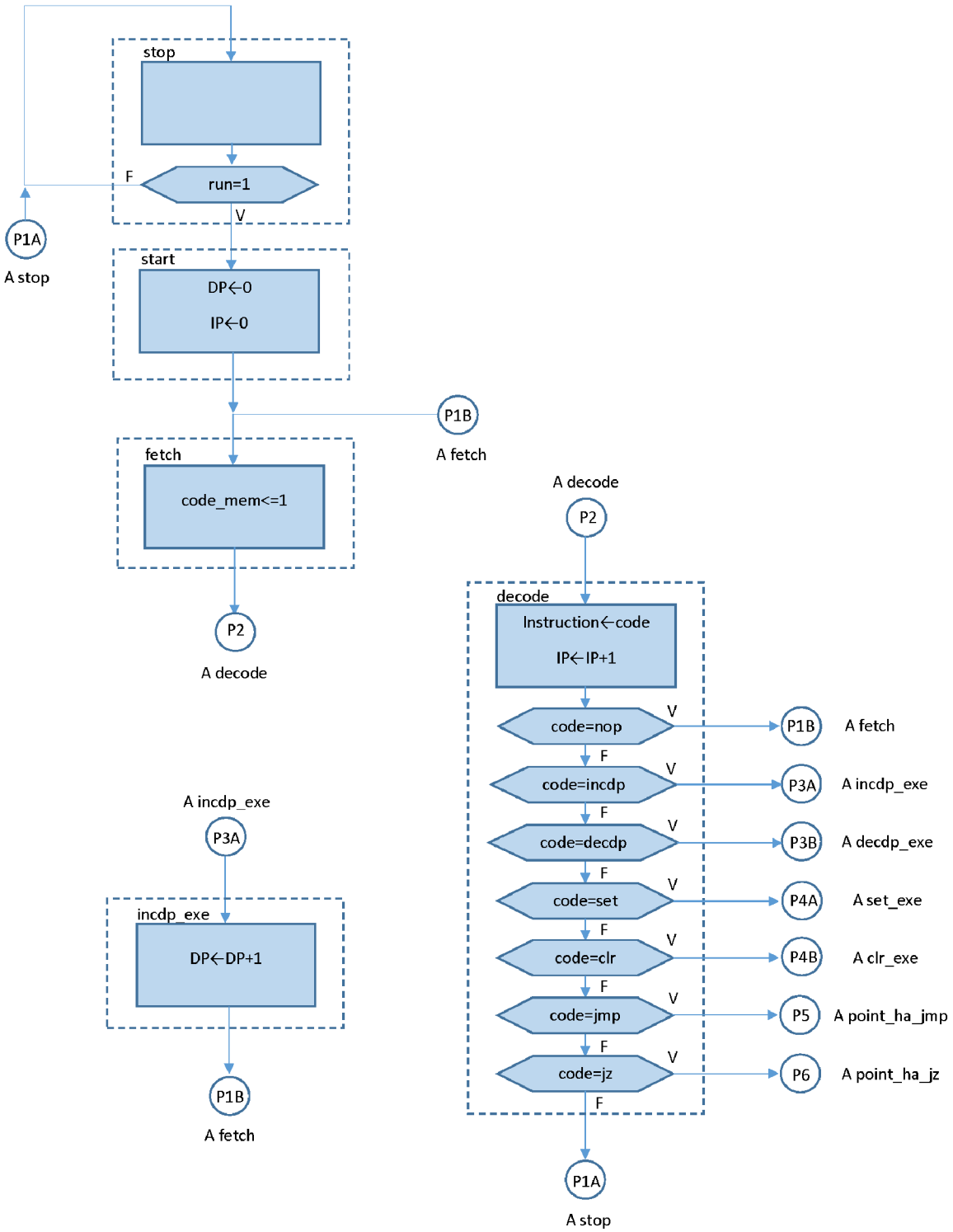

Figura 4. Diagrama a ASMD con la secuencia de micro-código para la instrucción `incdp` de la arquitectura propuesta para la máquina de Post modificada (MPM).