RISC-V: El acceso libre como una prioridad y el cómo afectará a la innovación informática.

1st Christian Alejandro Rengifo Mejia
Escuela de ingeniería de sistemas
Universidad Industrial de Santander
2162133
Grupo: A2
Bucaramanga, Colombia
alejoreme12@gmail.com

2nd Cristian Guillermo Serrano Acevedo
Escuela de ingeniería de sistemas
Universidad Industrial de Santander
2170127
Grupo: A2
Bucaramanga, Colombia
cristiangserrano@gmail.com

3rd Edinson Jahir Rodriguez Garces
Escuela de ingeniería de sistemas
Universidad Industrial de Santander
2162887
Grupo: A2
Bucaramanga, Colombia
edinsonrg22@gmail.com

4th Laura Daniela Serrano Villanova
Escuela de ingeniería de sistemas
Universidad Industrial de Santander
2162113
Grupo: A1
Bucaramanga, Colombia
lauvillanova2@gmail.com

5th Nicolas Mauricio Ramirez Triana
Escuela de ingeniería de sistemas
Universidad Industrial de Santander
2160060
Grupo: A2
Bucaramanga, Colombia
nicolas1ramireztr@gmail.com

Abstract

Technological development has been remarkable for the last decades, appreciable in the multiple existing ways to designing and developing microcontrollers, GPUs, CPus, and more. However, these set of instructions are usually classified, disabling the posibility of being modified or studied by users, even if its with an academic purpose, blocking the creative process. The Berkeley Architecture Research designed RISC-V, a set of instruction based on RISC architecture, as a free and open license that allows its community to actively participate in the innovation, development and transformation of this architecture, looking foward to create a network of new developers focusing on the academic and research field. The research group ONCHIP located in the Industrial University of Santander, work closely with this kind of architecture developing their knowledged on multiple projects.

Resumen

Es notable el desarrollo tecnológico en las últimas décadas; apreciable en las múltiples maneras existentes para diseñar y desarrollar microcontroladores, GPUS, CPUs, entre otros. No obstante, los conjuntos de instrucciones de estos últimos suelen ser clasificados, inhabilitando la posibilidad de ser alterados por los usuarios, así sea con un motivo más académico, generando una obstrucción al proceso creativo. El grupo de arquitectura de Berkeley ha diseñado RISC-V, la cual es una licencia libre y abierta que permite a su comunidad participar activamente en la innovación, desarrollo y transformación de esta arquitectura, todo con un objetivo mayormente educativo e investigativo. El grupo de investigación ONCHIP de la Universidad Industrial de Santander maneja esta arquitectura, trabajando y desarrollando en diferentes proyectos.

Palabras clave

Arquitectura, RISC-V, ONCHIP, ARM, Linux, ALU, Microcontrolador, Microprocesador, GPIO.

Figura 1. Logo RISC-V

I. Introducción

Algo que se ha apreciado a lo largo de la historia del mundo moderno es la velocidad del desarrollo de nuevas tecnologías. Esto ha llevado a toda empresa desarrolladora de hardware y software a diseñar, procesar y distribuir nuevas tecnologías a lo largo del mundo para estar por encima de la competencia y que estén a la altura de la tecnología de vanguardia. Es más fácil decir esto que realizarlo, de hecho, no siempre esto fue así; a pesar de que le tecnología ha avanzado a pasos agigantados, hay incontables ocasiones a lo largo de la historia en donde la tecnología se ha estancado, ya sea por la falta de conocimientos y/o experiencia para aquel momento, o simplemente una falta de imaginación para la innovación. A esta historia se le pueden sumar los casos en donde diferentes mentes brillantes que podrían revolucionar el mundo con sus ideas no son capaces de llevar sus cometidos a causa de la falta de recursos para iniciar sus proyectos. Es en ese momento que entra RISC-V, una arquitectura de código abierto y gratuita, al alcance de cualquiera, la cual ayudará a escribir el futuro de la tecnología y la innovación para los próximos 50 años.

II. Marco teórico

II-A. RISC

Reduced Instruction Set Computer o Conjunto de Instrucciones Reducidas, es un tipo de arquitectura de microprocesadores y microcontroladores que utiliza un pequeño y altamente optimizado conjunto de instrucciones, diferenciandose de otras arquitecturas que suelen utilizar conjuntos de instrucciones más complejas y especializadas. Esta arquitectura hace uso de instrucciones de tamaño fijo, las cuales son presentadas en un reducido número de formatos. Además, solo las instrucciones de carga y almacenamiento acceden a la memoria de datos.

El funcionamiento de RISC-V está determinado por una serie de procesos internos. Para ellos se destacan los procesos y conceptos escenciales para entender su funcionamiento.
Un archivo de ensamble contiene comandos, macros e instrucciones en lenguaje ensamblador. Puede ser emitido por un compilador o puede ser, por el contrario, escrito directamente a mano por un desarrollador. Este archivo es aquel que se le pasará como input al ensamblador para ser ejecutado. Una vez este archivo es ejecutado, el código compilado queda contenido en un archivo objeto (Relocatable Object File). El formato de archivo más utilizado para los ejecutables de RISC-V es el formato de archivo ELF (Electronic Linker Format). En caso de querer el proceso inverso, es decir, ver el ejecutable en lenguaje ensamblador, Objdump es un programa de comandos utilizado para esta función. El ensamblador implementa una serie de reglas que permiten incluir nueva información en el archivo objeto, controlar la alineación de los comandos, opciones de compresión, etc. Además, implementa una serie de pseudo-instrucciones formadas a partir de la base ISA, pero que contienen argumentos implicitos o invertidos, lo que resultan en una semántica y función diferente. Para manejar las ubicaciones de memoria en RISC-V con direcciones absolutas (Absolute Addressing), las cuales son formadas por instrucciones LUI para cargar bits y una instrucción como addi (Add Immediate), lw (Load Word) o sw (Store Word) para

llenar los 12 bits restantes a partir de lo hecho en el primer paso. Por otro lado, para manejar direcciones relativas se usa una instrucción auipc (Add Uperr Immediate Program Counter) para cargar bits al counter del programa, el cual es un registro del procesador que contiene la locación de la instrucción siendo ejecutada en el momento actual, seguido por una instrucción addi, lw o sw.

II-B. Microcontrolador

Es un circuito integrado que en su interior contiene una unidad central de procesamiento (CPU), unidades de memoria (RAM y ROM), puertos de entrada y salida y periféricos.

II-C. Licencia de software libre

Es un documento que otorga al receptor de una pieza de software derechos extensivos para modificar y redistribuir ese software.

II-D. Microprocesador

Es el encargado de ejecutar los programas, desde el sistema operativo hasta las aplicaciones de usuario, ejecuta instrucciones programadas en lenguaje de bajo nivel, realizando operaciones aritméticas y lógicas simples.

II-E. Hardware libre

Se llama hardware libre, hardware de código abierto, electrónica libre o máquinas libres a aquellos dispositivos de hardware cuyas especificaciones y diagramas esquemáticos son de acceso público, ya sea bajo algún tipo de pago, o de forma gratuita.

II-F. ARM

Es una arquitectura para microprocesadores cuyos beneficios son reducción de los costes, calor y energía. Estas características son deseables para dispositivos que funcionan con baterías, como los teléfonos móviles, tabletas, etc.

II-G. LINUX

Es un sistema operativo cuyo código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera, bajo los términos de la licencia GPL .

II-H. ALU

Es un circuito digital que calcula operaciones aritméticas (como suma, resta, multiplicación, etc.) y operaciones lógicas (si, y, o, no), entre valores (generalmente uno o dos) de los argumentos.

II-I. GPIO

Es un pin genérico en un chip, cuyo comportamiento (incluyendo si es un pin de entrada o salida) se puede controlar (programar) por el usuario en tiempo de ejecución.

III. Estado del arte

La ISA RISC-V está definida como una base ISA muy similar a los set de instrucciones de procesadores RISC excepto que soporta un codificador de posibles instrucciones de longitud variable. Esta base está restringida rigurosamente a un set de instrucciones mínimo que pueda proveer un funcionamiento adecuado para compiladores, ensambladores y sistemas operativos, y a su vez proveer herramientas de software que funcionen como el ¨esqueleto"sobre el cual se van a construir los procesadores. RISC-V es una familia de ISAs relacionadas. Cada ISA base está caracterizada por la longitud de sus registros y su correspondiente espacio de memoria. Actualmente, existen dos bases primordiales: RV32I y RV64I, las cuales proveen un espacio de direcciones para 32-bits y 64-bits respectivamente. Para el futuro, se visiona una variante RV128I que pueda

soportar espacios de direcciones para 128-bits.

RISC-V ha sido diseñado para soportar personalizaciones y especializaciones extensas y precisas. Cada una de las ISAs base puede ser extendidas con una o más extensiones de set de instrucciones, y a su vez, cada espacio de codificación de instrucciones está dividido en tres categorías disjuntas: estándar, reservada y personalizada. Las codificaciones estándar están definidas por la Fundación RISCV y no deberían de tener conflicto con otras extensiones estándar. La codificación reservada no están definidas actualmente, pero están reservadas para futuras extensiones estándares, mientras que la codificación personalizada está reservada para extensiones no estándar. Las extensiones de instrucciones generalmente son compartidas, sin embargo pueden llegar a proveer funcionalidades ligeramente diferentes según la ISA base que se esté trabajando. La principal ventaja de RISC-V es que, independientemente de qué modificaciones o el uso de futuras extensiones, cada versión de RISC-V continuará como una ISA independiente totalmente soportada.

RISCV-Config (RISCV Configuration Leagalizer) es un framework basado en YAML el cuál puede ser usado para validar las especificaciones de una implementación de RISC-V y generar un archivo de especificaciones estándar, comparándola con las arquitecturas privilegiadas y no privilegiadas actuales. Este funciona a partir de dos archivos YAML:

- ISA Spec: contiene las características implementadas por el usuario.
- Platform Spec: contiene las características específicas de la plataforma implementada por el usuario.

Primero, ambos archivos son procesadores para verificar que no cuenten con inconsistencias, después se generan dos archivos de especificación (uno para cada input) que contienen información de todos los campos junto con información adicional de interés; en caso de que el archivo de entrada no cuente con todos los campos especificados, el archivo se generará con valores predeterminados.

Los desafíos tecnológicos actuales están llevando a la industria de semiconductores hacia la especialización de hardware, provocando que los nuevos microchips creados estén tomandose el mercado, con un rendimiento potencialmente más alto que los de las arquitecturas tradicionales que se usan generalmente. Esta transición está siendo impulsada por una generación de arquitectura RISC-V con su ISA modular, modificable y extensible, permitiendo una muy amplia gama de diseños de procesadores y microcontroladores de bajo costo, con diversas posibles aplicaciones y ampliando la comunidad de desarrolladores que hacen uso de esta arquitectura. El Instituto de Tecnología de Georgia ha creado una unidad de procesamiento gráfica de propósito general (GPGPU) llamada Vortex compatible con OpenCL, un framework de programación de aplicaciones programado en C, basándose en la arquitectura de RISC-V. Vortex está creado a partir de SIMT, un modelo de multihilos y una extensión del set de instrucciones RISC-V que permite la ejecución de programas de OpenCL.

Vortex hace uso de librerías que garantizan que el kernel del chip pueda funcionar mientras implementan la nueva ISA, sin necesidad de modificar los compiladores RISC-V.

Imperas Software Ltd. desarrolló un modelo de referencia con encapsulación UVM para verificación RISC-V; estudia el entorno de manera completa y detallada para admitir la verificación del procesador, con los entornos de verificación de diseño de hardware SystemVerilog proporcionados por Cadence, Mentor, Synopsys y demás herramientas basadas en la nube de Metrics. Tradicionalmente, al analizar el diseño de SoC se estima que, aproximadamente la mitad del tiempo y recursos son dirigidos a tareas de verificación. Ahora con RISC-V, los desarrolladores pueden disfrutar de diseños adicionales y explorar opciones

para instrucciones y extensiones personalizadas de la ISA. Este modelo está disponible de manera libre para la comunidad de diseñadores y desarrolladores interesados.

IV. RISC-V

RISC-V (pronunciado Risk-five) es una arquitectura de conjunto de instrucciones (ISA) libre y gratuita, la cual ha permitido una nueva era de innovación en los procesadores por medio de la colaboración estándar abierta. Fundada en 2015, la fundación de RISC-V comprende a más de 325 miembros desarrollando la primera comunidad abierta y colaborativa de innovadores de software y hardware que impulsan la innovación de vanguardia. Originado en la academia y la investigación, el ISA RISC-V ofrece un nuevo nivel de libertad, con software y hardware libres de arquitectura flexible, dejando una huella para los próximos 50 años de diseño e innovación informática.

La fundación de RISC-V, una corporación sin fines de lucro controlada por sus propios miembros, dirige el futuro desarrollo e impulsa la aprobación del ISA RISC-V. Los miembros de la fundación de RISC-V tienen acceso y participación en el desarrollo de las especificaciones del ISA RISC-V relacionadas con los sistemas de hardware y software. La fundación posee una junta de directores compuesta de por siete representantes de Bluespec, Inc.; Google; Microsemi; Nvidia; NXP; la Universidad de Californica, Berkeley; y Western Digital. En noviembre de 2018, la fundación de RISC-V anunció una colaboración conjunta con la fundación de Linux. Como parte de esta colaboración, la fundación de Linux también proporcionará una afluencia de recursos para la organización de RISC-V, como programas de entrenamiento, herramientas de infraestructura, así como alcance comunitario, mercadeo y experiencia legal.

Cada año, la organización de RISC-V patrocina eventos mundiales para traer una expansión conjunta al sistema y discutir proyectos e implementaciones actuales y a futuro de RISC-V, así como también impulsar colectivamente la evolución futura de la arquitectura de conjunto de instrucciones (ISA). Las sesiones de los eventos cuentan con compañías tecnológicas líderes e institutos de investigación debatiendo la arquitectura de RISC-V, implementaciones de código abierto y comerciales, software y silicio, vectores y seguridad, aplicaciones y aceleradores, simulación de infraestructura y mucho más.

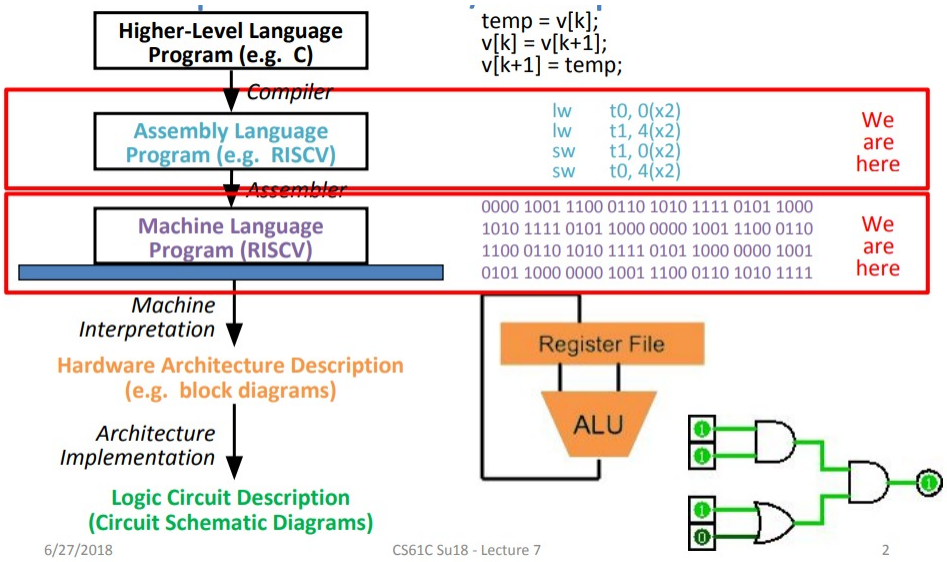

Figura 2. Arquitectura RISC-V

V. ¿POR QUÉ USAR RISC-V?

V-A. Costo

El arquitecto usualmente desea conservar el ISA simple para reducir el tamaño del procesador que lo implementa. El ISA del RISC-V es mucho más sencillo que el ISA de ARM-32. Como un ejemplo concreto del impacto de la simplicidad, comparando un procesador RISC-V Rocket con un ARM-32 Cortex-A5, ambos utilizando la misma tecnología (TSMC40GPLUS) y el mismo tamaño de cache (16 KiB). El chip de RISCV es 0.27 mm^2 vs 0.53 mm^2 para ARM-32. Dado que es casi el doble del área, el costo del chip del ARM-32 Cortex-A5 es aproximadamente 4 veces mayor que el del RISC-V Rocket. Aun una reducción del 10 % del tamaño del chip, reduce el costo en un factor de 1.2.

V-B. Simplicidad

Dada la correlación costo-complejidad, los arquitectos prefieren un ISA simple para reducir el tamaño del chip. La simplicidad también reduce el tiempo requerido para diseñar y validar, lo cual es un porcentaje importante del costo de desarrollo de un chip. Dichos costos deben ser agregados al costo del procesador siendo este incremento dependiente de la cantidad de procesadores vendidos. La simplicidad también reduce el costo de la documentación y la dificultad de hacer que los clientes entiendan cómo usar el ISA. Un procesador simple es beneficioso en aplicaciones embebidas dado que es más fácil predecir el tiempo de ejecución. Los programadores de lenguaje ensamblador para micro controladores muchas veces quieren mantener tiempos precisos, por lo que confían en que su código se ejecutará en una cantidad de ciclos predecible, la cual pueda ser calculada con antelación.

V-C. Rendimiento.

A excepción de chips muy pequeños utilizados para aplicaciones embebidas, los arquitectos se preocupan tanto por rendimiento como costo. Tres factores que afectan el rendimiento son:

$$\frac{Instrucciones}{programa} * \frac{ciclospromedio}{instruccion} * \frac{Instrucciones}{programa} = \frac{tiempo}{programa} \tag{1}$$

Aun cuando un ISA más sencillo ejecute más instrucciones que un ISA complejo, este programa puede ejecutarse más rápido si la frecuencia de reloj es más rápida o si promedian menos CPI (Ciclos Por Instrucción).

El último factor es el inverso de la frecuencia de reloj, así que una frecuencia de 1 GHz significa que el período es de 1 ns (1/109). Por ejemplo, para la prueba de rendimiento CoreMark (100,000 iteraciones), el rendimiento para el ARM-32 Cortex-A9 es:

$$\frac{32{,}27BInstrucciones}{programa} * \frac{0{,}79CiclosdeReloj}{instruccion} * \frac{0{,}71ns}{ciclosdereloj} = \frac{18{,}15segs}{programa} \tag{2}$$

Para la implementación de BOOM en RISC-V, la ecuación es:

$$\frac{29{,}51BInstrucciones}{programa} * \frac{0{,}72CiclosdeReloj}{instruccion} * \frac{0{,}67ns}{ciclosdereloj} = \frac{14{,}26segs}{programa} \tag{3}$$

El procesador ARM no necesariamente ejecuta menos instrucciones que RISC-V. Las instrucciones simples son además las más populares, así que un ISA más simple puede ganar en todas las métricas. Para este programa, el procesador RISC-V aventaja cerca de un 10 % en cada uno de los tres factores, lo cual resulta en un rendimiento casi 30 % más rápido. Si un ISA más simple además resulta en un chip más pequeño, su relación costo rendimiento será muy buena.

V-D. Facilidad de programar, compilar y linkear

Dado que el acceso de datos en registros es mucho más rápido que en memoria, es crucial que los compiladores hagan una buena asignación de registros. Dicha tarea es más fácil entre más registros se tenga. Bajo esa perspectiva, ARM-32 tiene 16 registros y x86-32 únicamente 8. La mayoría de ISAs modernos, incluyendo RISC-V, tienen una cantidad relativamente generosa de 32 registros enteros. Más registros hacen la tarea más fácil para programadores de ensamblador y compiladores. Otra complicación para programadores de ensamblador y compiladores es entender la velocidad de ejecución. Las instrucciones de RISC-V normalmente se ejecutan en un ciclo de reloj (ignorando cache misses), pero como vimos antes, tanto ARM-32 como x86-32 tienen instrucciones que llevan varios ciclos de reloj aun sin cache misses. Es más, a diferencia de ARM-32 y RISC-V, las instrucciones aritméticas del x86-32 pueden tener operandos en memoria en vez de solo registros. Instrucciones complejas y operandos en memoria incrementan la dificultad para los diseñadores de procesadores en proveer estimaciones de desempeño. Es útil si el ISA soporta PIC (Position Independent Code: Código independiente de su posición), dado que soporta dynamic linking porque el código de la librería compartida puede estar en diferentes direcciones en distintos programas. Saltos relativos al PC y direccionamiento de datos son una ventaja para PIC. Mientras casi todos los ISAs proveen saltos relativos al PC, x86-32 y MIPS-32 omiten el direccionamiento de datos relativo al PC.

V-E. Tamaño del Programa

Entre más pequeño el programa, menor el área del chip que ocupará la memoria, lo cual puede ser un costo significativo para sistemas embebidos. Por esto, los arquitectos de ARM agregaron instrucciones pequeñas en los ISAs Thumb y Thumb2. Programas más pequeños además implican menos perdidas de instrucciones de caché, lo cual ahorra energía porque accesos a memoria DRAM externa consumen mucho más energía que accesos a SRAM en el chip. Generar código pequeño puede ser un objetivo del arquitecto del ISA. El ISA x86-32 tiene instrucciones que van desde 1 byte hasta 15 bytes. Uno esperaría que el tamaño variable en instrucciones del x86 condujera a código más pequeño que ISAs donde todas las instrucciones son de 32 bits, como ARM-32 o RISC-V. Lógicamente, instrucciones variables de 8 bits deberían ser más pequeñas que ISAs que únicamente ofrecen instrucciones de 16 ó 32 bits, así como Thumb-2 y RISC-V usando extensiones RV32C, mientras que el código de ARM-32 y RISC-V es del 6 % al 9 % mayor que el código del x86-32 cuando todas las instrucciones son de 32 bits, sorprendentemente, x86-32 es 26 % mayor que las versiones compresas (RV32C y Thumb-2) que ofrecen instrucciones de 16 y 32 bits. Un ejemplo de instrucción de x86- 32 de 15 bytes es $lock\ add\ dword\ ptr\ ds : [esi + ecx * 4 + 0x12345678]$, $0xefcdab89$. Ensambla en hexadecimal a: $\#66f03e81848e\ 78563412\ 89abcdef$. Los últimos 8 bytes son 2 direcciones y los primeros 7 bytes especifican operaciones de memoria atómicas, la operación suma, datos de 32-bits, el registro del segmento de datos, los dos registros de direcciones y el modo de direccionamiento escalado indexado. Un ejemplo de instrucción de 1-byte es $inceax$, que ensambla a 40. A pesar de que probablemente un nuevo ISA de instrucciones variables de 8 bits llevaría a código más compacto que RV32C y Thumb-2, los arquitectos del ISA x86 en los 70s tenían otras preocupaciones. Además, dado el requerimiento de retrocompatibilidad binaria de un ISA incremental, los cientos de instrucciones nuevas son mayores de lo esperado, dado que quedó muy poco espacio para el opcode en el x86 original.

VI. ONCHIP

La Universidad Industrial de Santander, cuenta con el grupo de investigación ONCHIP enfocado en sistemas integrados en la escuela de Electrónica, Eléctrica y Comunicaciones. Dirigido por Elkim Roa, se basa en la arquitectura RISC-V para el

desarrollo de microcontroladores para ser usados en diferentes áreas del conocimiento.

Open-V es un microcontrolador de 32-bits de código abierto creado por ONCHIP, el primer microcontrolador en Colombia y el primero en el mundo en usar esta arquitectura. Todas las comunicaciones del microcontrolador de 2x2mm son realizadas a través de AXI4-Lite, un protocolo libre diseñado por ARM (Arquitectura RISC de 32 bits) usado para interconectar buses en este tipo de arquitectura. Open-V está diseñado para adaptarse a diferentes aplicaciones que la industria espere y requiera de un microcontrolador. Usa buses SPI para manejar la comunicación del microcontrolador, primero a través de un maestro que permite programar y depurar las operaciones y un esclavo que recibe y captura la información. El microcontrolador también cuenta con 8 GPIOs con capacidad hasta el 20mA; por último, este cuenta con interfaces análogas-digitales que ayudan a verificar todas las funciones. El chip usa SAR ADC para convertir señales análogas continuas en representación digital discreta y un convertidor digital R-2R para manejar salidas de audio. Open-V pretende ser el equivalente a microcontroladores comerciales implementados con la arquitectura RISC ARM y aspira ser un camino para futuras implementaciones.

El grupo de investigación ONCHIP, pretende además apoyar la investigación, educación y una comunidad de desarrolladores que emerge a raíz del hardware de código abierto, por lo que no solo comparte Open-V con la comunidad informática actual, sino que intenta acercar a nuevas poblaciones a esta, como es en el caso de niños y adolescentes de primaria y bachillerato en Colombia. Por esta razón, se diseñó un set de rompecabezas que incorpora en sus piezas el microcontrolador de 32 bits para su funcionamiento. Según la manera en que el niño ubique las piezas se encenderán luces LED en diferentes patrones, a través de sensores activados por los dedos del niño. De esta manera, los niños pueden desarrollar su imaginación y expandir su creatividad, pues son ellos mismos los encargados de crear y descubrir patrones.

VII. Contribución desde Arquitectura de Computadores y la EISI

VII-A. Arquitectura de Computadores

Por parte de la materia de Arquitectura de Computadores, el dar a conocer de información sobre qué es RISC-V ayuda a expandir el conocimiento su funcionamiento y arquitectura por lo que futuros estudiantes conocerán de esta, no de una manera de investigación sino de enseñanza, que permita que los mismos estudiantes interesados puedan abrirse al mundo laboral desde un sector que, en pleno desarrollo, ha generado mucho interés. Aunque RISC-V tiene pocos años de vida, lentamente está tomando una relevancia que tomará una gran parte del sector de procesadores, principalmente en los sistemas embebidos donde más brilla esta arquitectura. El devenir de los cursos futuros de esta materia es incierto, la tecnología avanza de una manera tan rápida que es necesario acelerar el ritmo de enseñanza, ser capaces de preveer esta situación da tiempo a prepararnos para los cambios más radicales posibles.

VII-B. EISI

Desde la EISI, podemos apoyar los proyectos en ONCHIP a través del análisis de los distintos procesos que son llevados a cabo en este grupo, el estudio de sus funcionalidades, mediciones en tiempo de ejecución, complejidad de algoritmos y su proceso de desarrollo. Con esto podemos desarrollar un diseño estructurado para la creación de micro controladores, con el fin de que estos ayuden a las distintas aplicaciones que se estén realizando, optimizar ciertos procesos que retrasan la ejecución del mismo y además, brindar apoyo en el desarrollo de software para que este soporte los procesos que se llevarán a cabo a través del hardware.

Dando una visión desde la parte del curso de Arquitectura de computadores, se presenta una documentación que valida los estudios realizados y el conocimiento adquirido, para sentar una base de donde partir para la realización de actualizaciones o, en su mismo caso, optimizaciones de diferentes arquitecturas, organizando la estructura que se presente en estos sistemas con el fin de dejar presente el análisis, diseño y posibles ejecuciones del mismo.

Sabiendo que desde una materia se puede generar tanto desarrollo, es igual de importante que desde la Escuela de Ingeniería de Sistemas e Informática (EISI) de apoyo al desarrollo de RISC-V, esto de una manera más directa, al fomentar estudios y proyectos relacionados con RISC-V. Se espera que en el futuro esta arquitectura este implementada en objetos de uso diario y que muchas de las aplicaciones hallan sido desarrolladas por ingenieros de la EISI o de otra facultad de la Universidad Industrial de Santander.

VIII. Conclusiones

1. RISC-V puede ser de mucha utilidad a la escuela de ingeniería de sistemas para aquellos interesados en estudiar y trabajar con esta temática, y necesiten una arquitectura abierta para comenzar con el desarrollo de sus ideas.
2. La flexibilidad de RISC-V le permite estar presente en una rica variedad de proyectos de desarrollo de hardware y software, lo cual compensa el defecto principal de RISC-V, el cual viene siendo que no es compatible con lenguajes de alto nivel que son populares en el mercado.
3. Muchas son las ideas que se quedan en el papel por falta de que los autores no tienen los suficientes recursos para empezar a desarrollarlas, por lo tanto, tecnologías de código abierto como RISC-V son necesarias para que el desarrollo de nuevas tecnologías no se vea estancado.
4. La fundación de RISC-V quiere evitar que el proyecto salga de su concepción inicial, quieren que siga siendo un ISA de código abierto y gratuito. La razón de esto es porque quieren mantener a RISC-V disponible para todos.

Referencias

[1] D. Paerson y A. Waterman, Guía prácca de RISC-V El Atlas de una Arquitectura abierta, (1er edición), San Francisco. CA. USA: Strawberry Canyon LLC. 2018. [online] Disponible en: hp://riscvbook.com/spanish/guia-pracca-de-risc-v-1.0.5.pdf

[2] ONCHIP Group (2016). Onchip Uis (1er edición) [online]. Disponible en: hp://www.onchipuis.io/index.html

[3] Á. Sampayo (2016). RISC-V MCU grown in Colombia (1er edición) [online]. Disponible en: hps://ieeexplore.ieee.org/document/7451073

[4] C. Duran, D. Rueda y otros (2016). A 32-bit RISC-V AXI4-lite bus-based microcontroller with 10-bit SAR ADC (1er edición) [online]. Disponible en: hps://www.eenewseurope.com/news/risc-v-mcu-grown-colombia

[5] Fundación RISC-V. About the RISC-V Foundation (1er edición) [online]. Disponible en: https://riscv.org/risc-v-foundation/

[6] Guía práctica de RISC-V (1er edición) [online]. Disponible en: http://riscvbook.com/spanish/guia-practica-de-risc-v-1.0.5.pdf

[7] Great Ideas in Computer Architecture RISC-V Instruction Formats https://inst.eecs.berkeley.edu/ cs61c/resources/su18_lec/Lecture7.pdf