BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA ELECTRÓNICA

MAESTRÍA EN INGENIERÍA ELECTRÓNICA, OPCIÓN INSTRUMENTACIÓN ELECTRÓNICA

Tesis para obtener el título de:
MAESTRO EN INGENIERÍA ELECTRÓNICA

"Sistema embebido para interconexión de dispositivos industriales de entrada/salida reconfigurables"

Presenta:

Ángel Juárez Palacios.

Asesores de Tesis

M.C. Rodrigo Lucio Maya Ramírez. (FCE-BUAP)

Dr. Gerardo Mino Aguilar. (FCE-BUAP)

Puebla, Pue. Noviembre de 2021

Índice general

Índice de figuras

Índice de cuadros o tablas

Introducción

Debido a la tendencia de una necesidad hacia sistemas reconfigurables en el entorno industrial, hoy en día, las máquinas modernas necesitan adaptarse rápidamente a los nuevos requisitos y demandas de las líneas de producción. Para implementar estrategias de fabricación cambiantes, los sistemas deben poder integrar una gran variedad de diferentes tipos de sensores.

Por el cual se tiene la necesidad de agotar el tema de sistemas embebidos en todas sus modalidades en un FPGA, de núcleos duros (arm) y suaves (MicroBlaze). Y también explorar nuevas tecnologías de programación que se conocen como barel-metal en un FPGA. Para así lograr un sistema embebido de mayor rendimiento enfocado en los procesos de interés.

Esto también provoca la necesidad de dispositivos de E/S flexibles que puedan adaptarse a los cambios de los sensores de forma más ágil. En la actualidad, los sensores y los actuadores suelen estar conectados a dispositivos de E/S que se comunican a través de un bus de entorno de interconexión (por ejemplo, PROFINET, EtherCAT, ModBus) con el controlador lógico programable (PLC) se pretende generar de alguna forma un sistema base con un enfoque industrial con los protocolos estándares del medio de comunicación.

OBJETIVO GENERAL

Diseñar e implementar un sistema embebido utilizando el procesador de núcleo suave MicroBlaze, en el kit de evaluación ZC702 de Xilinx, para la construcción de módulos especializados de E/S industriales.

.

OBJETIVOS ESPECÍFICOS

1. Diseñar el sistema multiprocesamiento asimétrico con los procesadores MicroBlaze y ARM Cortex-A9.
2. Implementar el sistema en el kit de evaluación ZC702 de Xilinx.
3. Incorporar un sistema operativo embebido basado en Linux.
4. Realizar las pruebas de funcionamiento del sistema.
5. Diseñar los módulos especializados de E/S industriales.

Capítulo 1

Celdas de PLCs con dispositivos de entrada y salida.

En este capítulo primero iniciamos con el concepto de Control Lógico Programable desde el punto de vista industrial, después una breve historia sobre ellos.
Después nos enfocaremos de forma más concreta sobre los PLCs SIEMENS de la familia s7-1200 ya que cuenta con comunicación PROFIBUS.
Y entender el funcionamiento operativo que se da entre los dispositivos de entrada y salida con los PLCs de la familia S7-1200.

1.1 Concepto de PLC.

Partiendo primero en si con el acrónimo PLC que está en inglés "Programmable Logic Control" donde se traduce como Control lógico programable, otra traducción muy común es la de autómata programable. Básicamente es una computadora, utilizada en la automatización de procesos o en la automatización industrial, para los procesos electromecánicos, tales como el control de la maquinaria de la fábrica en líneas de producción.
Existe una definición sobre el término PLC que es dado por una organización de los Estados Unidos de América llamada National Electrical Manufacturers Association que lo define como:
"Instrumento electrónico, que utiliza memoria programable para guardar instrucciones sobre la implementación de determinadas funciones, como operaciones lógicas, secuencias de acciones, especificaciones temporales, contadores y cálculos para el control mediante módulos de E/S analógicos o digitales sobre diferentes tipos de máquinas y de procesos"
En el medio industrial europeo y latinoamericano es la definición base que se toma. Por el cual nos quedamos con esa definición. Existen otras definiciones académicamente hablando, a nosotros nos interesa el aspecto industrial en este momento.

En la actualidad no podremos concebir una industria sin PLCs. Al contrario, se han ido diversificando en sus aplicaciones y en su compatibilidad con otros recursos de tipo industrial. Se han implementado una gran cantidad de protocolos y estándares de comunicación.

1.2 Historia de los PLCs

A finales de los años 60, la industria automotriz necesitaba un sistema de control económico, robusto, flexible y fácilmente modificable. En año 1968 una pequeña empresa llamada Bedford Associates y Dick Morley era uno de sus ingenieros eléctricos. Su solución propuesta se basó en las nuevas tecnologías digitales e involucró el cableado de sensores e interruptores de límite, etc. para una nueva invención que impulsaría solenoides, lámparas, actuadores, etc. El término "entradas" se usó para describir las señales del campo y las señales que van al campo se llamaba 'salidas'. Entre estas "entradas" y "salidas", el diseño del esquema de control adoptó un nuevo enfoque revolucionario. Las entradas se pueden vincular en cadenas a salidas como el método de cableado, pero ese enlace podría configurarse o programarse en lugar de cablearse manualmente en un gabinete de control.

Este nuevo diseño digital de inmediato facilitó la reconfiguración de los sistemas, mejoró la confiabilidad del sistema. En la figura 1.1 se muestra este nuevo diseño que fue el primer PLC, llamado MODICON 084.

Dick Morley y Bedford Associates habían inventado el primer controlador lógico programable (PLC). Fue llamado el controlador digital modular debido a su construcción modular y uso de tecnología digital. Dick acortó esto a MODICON y comenzó una revolución en el control industrial.

Figura 1.1 El Primer PLC de la historia, MODICON 084.

El MODICON 084 original proporcionó una enorme mejora en la rentabilidad para nuestros clientes en términos de tiempo reducido para el mercado y el tiempo de inactividad de la producción. Históricamente es considerado a Richard E. "Dick" Morley (1 de diciembre de 1932 - 17 de octubre de 2017) como “El padre del PLC”, fue un ingeniero mecánico estadounidense.

Por la parte Europea tenemos un conjunto de empresas alemanas llamadas SIEMENS Schuckert AG que fue fundada 1847 y en abril de 1958, cuando Siemens Schucker AG registró en las oficinas de patentes alemanas la marca SIMATIC, sintetizando con ello los nombres de Siemens y Automatización en un sólo concepto, lo que se gestó como un sistema de control automático revolucionario, se terminaría convirtiendo en el más innovador de los controladores PLCs, ofreciendo soluciones que revolucionarían definitivamente el concepto de automatización industrial.

Su primera generación, el SIMATIC G, se presentó en la Feria de Máquinas Herramienta de París en el año 1959. Este controlador realizaba gran cantidad de funciones, y podía utilizarse en múltiples aplicaciones que iban desde ascensores hasta el control de máquinas herramientas para realizar trabajos en una secuencia preestablecida. En las siguientes décadas, el sistema SIMATIC G fue reemplazado sucesivamente por varias generaciones de autómatas, con una cantidad de funciones en permanente aumento.

1.3 Familia de PLCs S7-1200 y sus características.

En esta parte se define una familia de PLCs a trabajar. Se elige esta familia por tener los canales de comunicación más comunes en este momento a nivel industrial. El controlador modular SIMATIC S7-1200 es la línea de productos Siemens para tareas de automatización sencillas, pero de alta precisión.

Tiene capacidad para cubrir el espectro de comunicaciones inalámbricas y remotas gracias a su interfaz Profinet o Ethernet incorporada. En este sentido, existe la posibilidad de establecer comunicaciones inalámbricas con estaciones remotas, así como establecer el control de las instalaciones a través de Internet. Esta funcionalidad permite la monitorización y su control a través de un ordenador remoto y la modificación del programa desde cualquier parte del mundo.

Existen cinco modelos de CPUs en esta familia: 1211C, 1212C, 1214C, 1215C, 1217C. En la figura 1.2 se muestra el aspecto físico de la familia de PLCs S7-1200.

Figura 1.2 PLCs S7-1200

Características:

Paneles SIMATIC HIM Basic desde las 4 pulgadas a 15 pulgadas. Sistema de alarmas, administración de recetas (este término es tomado como tecnicismo de la marca SIEMENS) y la funcionalidad de curvas.

Los HMIs son interfaces hombre-máquina, estas interfaces son conocidas de esa forma debido a sus siglas en inglés "Human Machine Interface". Básicamente, estamos hablando de medios que van a permitir a los operarios la posibilidad de interactuar con una determinada máquina a partir de un entorno entendible para la persona que se encarga de ello. Principalmente, estas interfaces son pantallas táctiles localizadas a pie de máquina o en un punto estratégico entre varias máquinas que controlan un mismo proceso.

SIMATIC STEP 7 Basic v11, sistema de ingeniería integrado. Incluye WinCC Basic para SIMATICS S7-1200 y Paneles Basic HMI.

Marco de ingeniería común para hardware y configuraciones de red, programación y diagnóstico.

Figura 1.3 Posicionamiento de la familia SIMATIC S7-1200.

Entradas y salidas digitales integradas
Entradas Digitales Tipo: Sumidero/Fuente tensión nominal: 24 VDC a 4 mA
Salidas Digitales Tipo: Relé, rango de voltaje: 5 a 30 VDC o 5 a 250 VAC Corriente (max.): 2.0 A
Tipo: Fuente de Rango de voltaje: 20.4 a 28.8 VDC Corriente (máx.): 0.5 A
Entradas analógicas integradas
2 entradas analógicas tipo: Tensión (unipolares), Rango: 0 to 10 V Resolución: 10 bits
Memoria de usuario integrada
EL CPU S7-1200 dispone de las siguientes áreas de memoria:
Memoria de carga. Un área de almacenamiento no volátil para el programa de usuario, los datos y la configuración.
Memoria de trabajo. Un área de almacenamiento volátil para algunos elementos del proyecto de usuario utilizado por el CPU mientras se ejecuta utilizado por el CPU mientras se ejecuta el programa de usuario
Memoria retentiva. Un área de almacenamiento no volátil que se utiliza para "retener" una cantidad limitada de valores de la memoria de trabajo durante una pérdida de potencia.
Memoria de usuario integrada.

Admite una frontera "flotante" entre el programa de usuario, los datos de los programas y los datos de configuración.
Almacena los comentarios del programa y símbolos de usuario.
Almacena la información de configuración general para cada dispositivo de hardware.

Rendimiento.
El CPU S7-1200 combina un microprocesador, una fuente de alimentación integrada, entradas, salidas y en una carcasa compacta para crear un controlador muy potente.
El CPU controla cambios de entradas/salidas de acuerdo a la lógica del programa de usuario, que puede incluir lógica booleana, contaje, temporizador, operaciones matemáticas, y la comunicación con otros dispositivos inteligentes.
Características adicionales
Bloques de terminales desmontables para la puesta en marcha y mantenimiento sencillo.
En todos los CPUs, existen módulos de señales y módulos de comunicaciones.
Reloj de tiempo real y calendario integrado.
Horario de verano y offset estándar de tiempo.
Un súper condensador que mantiene el reloj en funcionamiento cuando el CPU está apagado con un tiempo de retención de 10 días típicamente.
Módulos y procesadores de comunicaciones
Con los Módulos y procesadores de comunicaciones, se pueden agregar las interfaces adecuadas para satisfacer sus necesidades de comunicación.
Los módulos de comunicación RS232 y RS485 proporcionan comunicación punto a punto *USS* y *Modbus RTU*.
El módulo de comunicación PROFIBUS permite la comunicación PROFIBUS.
El procesador de comunicación GPRS proporciona soporte para la monitorización y control remoto.
Módulo de conmutación compacto (*Compact Switch Module*)
Con el Módulo de conmutación compacto CSM 1277, se puede configurar fácilmente una red de distribución uniforme o mixta en línea, en topologías de árbol o estrella y reducir al mínimo el cableado con la máxima flexibilidad en red.
Este conmutador de 4 puertos administrados le permite conectar hasta 3 dispositivos adicionales de ethernet a la estación de control S7-1200.

1.4 Comunicaciones Industriales de la Familia S7-1200

La familia S7-1200 ofrece una variedad de opciones de comunicación para satisfacer todas sus necesidades de red.
Interfaz PROFINET integrada que soporta TCP/IP en los estándares de comunicación (TCP, ISO en TCP y UDP)
La interfaz PROFINET del CPU S71200 admite las siguientes conexiones simultáneas de comunicación: 3 conexiones de comunicación HMI con CPU, 1 conexión con programadora PG, 8 conexiones para comunicación activa mediante comunicación abierta entre usuarios e instrucciones de I/O, 3 conexiones de comunicación pasiva con instrucciones de comunicación S7
PROFIBUS. Junto a los módulos de comunicación PROFIBUS maestro y esclavo, el S7-1200 CPU soporta el estándar de PROFIBUS.

Con el módulo de comunicaciones maestro profibus DP-CM 1243-5, el S7-1200 puede comunicar: Con otros CPUs, con una programadora, con dispositivos HMI y con esclavos PROFIBUS DP.
Ahora con el módulo de comunicación esclavo profibus DP-CM 1242-5, el S7-1200 se puede comunicar como un esclavo DP inteligente con cualquier maestro PROFIBUS DP.
La comunicación punto a punto (PtP). Junto a los módulos de comunicación RS232 y RS485 así como el módulo de comunicación RS485, el CPU S7-1200 soporta la comunicación punto a punto (PtP) para un conjunto de protocolos basados en caracteres.
La comunicación punto a punto permite una gran variedad de posibilidades: La capacidad de enviar información directamente a un dispositivo externo, como una impresora.
La capacidad de recibir información de otros dispositivos, tales como lectores de códigos de barras, lectores RFID, y sistemas de visión.
La capacidad de intercambio de información, envío y recepción de datos, con otros dispositivos tales como dispositivos GPS, módems de radio, y muchos otros tipos de dispositivos.
Interface serial universal (USS). Junto con las instrucciones USS, los CPUs S7-1200 puede controlar el funcionamiento de unidades que soporten la interfaz universal en serie (USS) de protocolo.
Las instrucciones USS se puede utilizar para comunicarse con múltiples unidades a través del módulo de comunicación CM 1241 RS485 o la tarjeta de comunicación CB 1241 RS485.
Modbus RTU (Modbus TCP/IP en proceso). Junto con las instrucciones Modbus S7-1200 puede comunicarse como maestro o esclavo Modbus con los dispositivos que soportan el protocolo Modbus RTU.
Las instrucciones Modbus se pueden utilizar para comunicarse con múltiples dispositivos a través de ya sea el módulo de comunicación CM 1241 RS232, el módulo de comunicación CM 1241 RS485, o el CB 1241 RS485 communication board.
Comunicación Telecontrol. Junto al procesador de comunicación GPRS, el CPU S7-1200 soporta aplicaciones sencillas de telecontrol para el monitoreo y control de estaciones distribuidas utilizando el servicio general de radio por paquetes (GPRS).
Con el procesador de comunicaciones GPRS CP 1242-7, el S7-1200 puede comunicar remotamente, con la estación de control central, con otras estaciones remotas, con dispositivos móviles (SMS), con una programadora (*Teleservice*) y con otros interlocutores que utilizan la comunicación abierta entre usuarios basado en UDP.

Capítulo 2

Planteamiento general para un sistema embebido en un FPGA con el procesador de núcleo suave.

En este capítulo se fundamentará toda la estructura teórica para el soporte de un sistema embebido en el kit evaluación ZC702 de Xilinx .
Iniciando desde la base que sustenta este sistema embebido que es el kit de evaluación, viendo todos los periféricos que están integrados en el kit, puntualizamos de forma detallada a nivel de diagrama de bloques el SoC Zynq-7000 para después observar que este SoC tiene internamente procesadores ARM Cortex A9.
Después pasamos al concepto del procesador de núcleo suave donde nos enfocamos al núcleo del procesador MicroBlaze de Xilinx, existiendo una variedad amplia de versiones de Microblaze.
Y por último se hace un bosquejo de sistemas operativos embebidos que se pueden correr en el kit de evaluación y el procesador de núcleo suave Microblaze. Por cuestiones de infraestructura de recursos nos enfocamos en sistemas de código abierto como es Linux.

2.1 FPGAs Xilinx

En la década de 1980 la industria de circuitos integrados, su objetivo principal era la de optimizar al máximo sus componentes internos.
Pero surgió una idea de un circuito integrado donde estaba conformado en bloques lógicos organizados, con conexiones que se podían configurar y reconfigurar con software. Es cuando se crea el concepto de FPGA (Field-Programmable Gate Array) .- Arreglo de compuertas programables. Con ese concepto se funda una empresa llamada Xilinx en 1984 por Jim Barnett, Ross Freeman y Bernie Vonderschmitt.
Y en 1985 sale al mercado el primer FPGA llamado XC2064 de Xilinx. De ahí se ha mantenido un desarrollo de diferentes familias de FPGAs de Xilinx y también surgieron otras empresas que también desarrollaron otros FPGAs. Se hace referencia Xilinx porque fue la creadora del primer FPGA y se considera que se ha mantenido en la innovación tecnológica de sus FPGAs, en la siguiente lista se describe la secuencia de familias de FPGAs que tiene Xilinx:.

- Spartan (bajo costo, ya está de salida)
- Artix (bajo costo)
- Kintex (rango medio)
- Virtex (alto rendimiento)
- Zynq (SoC con Procesador ARM)

Estas familias de FPGAs son implementadas en un formato de SoC que es un encapsulado que pueden ser implementadas en kits de evaluación de otras empresas y por los propios de Xilinx.

2.2 Placa de evaluación ZC702 para el System on Chip Zynq-7000 XC7Z020

La placa de evaluación ZC702 contiene un FPGA en el SoC XC7Z020 en un dispositivo Zynq® XC7Z020-1CLG484C. La placa de evaluación proporciona características comunes para sistemas empotrados, posee componentes de memoria DDR3, un puerto de ethernet que puede operar en tres diferentes velocidades 10/100/1000 MHz, también tiene puertos de entrada y salida de propósito general y dos interfaces de Transmisor-Receptor Asíncrono Universal (*Universal Asynchronous Receiver-Transmitter*). Cuenta con dos ranuras de interconexión con el estándar VITA-57 (FPGA Mezzanine Card) que nos permite agregar otras tarjetas de evaluación.

2.2.1 Características de la placa ZC702

Se pueden describir las características de forma superficial haciendo referencia al diagrama de bloques como es mostrado en la figura 2.1. A continuación se listan las características de la placa de evaluación:

- Interfaz serial de memoria flash cuádruple (Quad Serial Peripheral Interface Flash Memory)
- Modulo y Conector JTAG (Joint Test Action Group)
- 1GB de Memoria DDR3 4 x 256 Mb x 8 SDRAM
- Reloj y Reset por botones de contacto.
- JTAG Header
- USB Universal Asynchronous Receiver-Transmitter (UART)
- ARM PJTAG Header
- Switches LEDs and Pushbuttons
- I2C Real Time Clock
- I2C Multiplexer and i2C EEPROM
- XADC convertidor de analógico a digital y un módulo de conexiones.
- Señales de reloj configurables.
- High-Definition Multimedia Interface HDMI Codec and Connector
- USB 2.0 ULPI Transceptor y conector.
  ULPI - El estándar para PHY (PHYsical layer – capa física) USB de alta velocidad
- 10/100/1000 Ethernet PHY (RGMII only)
  RGMII (Reduced Gigabit Media-Independent Interface) es un estándar de transmisión.
- Conector FMC2 LPC.
- Conector FMC1 LPC.
- Conector para tarjeta SD (Secure Digital)
- CAN Bus

Nota PHY – Término usado para hacer referencia a la capa física en el modelo OSI y a nivel hardware los medios de transmisión del emisor, receptor y el canal de comunicación.

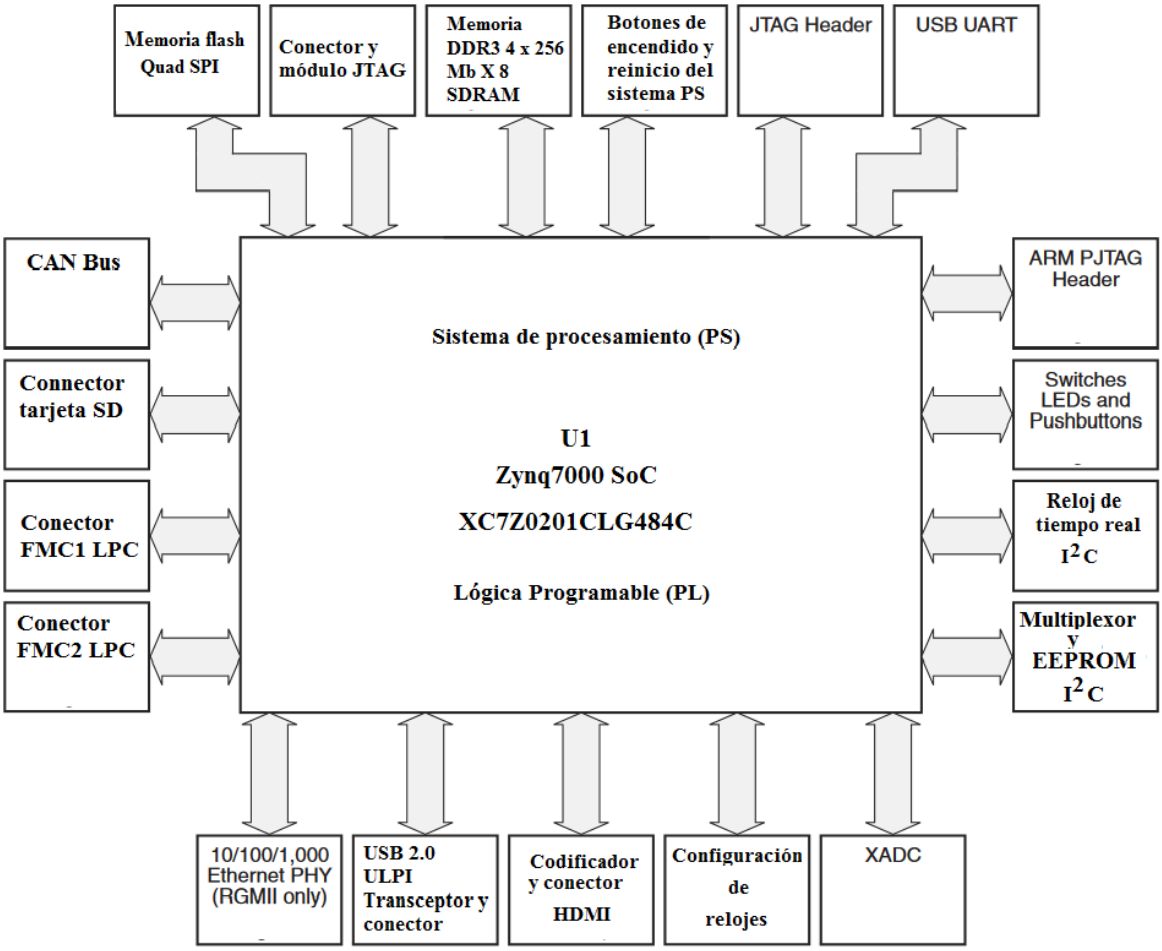


Figura 2.1 Diagrama de bloques de la Placa de evaluación ZC702.

2.2.2 System on Chip del Zynq-7000 XC7Z020

Ahora pasamos de forma más concreta ver las características del SoC Zynq-7000 XC7Z020 que es en si el FPGA. En esta parte hay dos enfoques de diseño: El sistema de procesamiento PS y la lógica programable PL, este enfoque se describe de mejor manera en la figura 2.2.

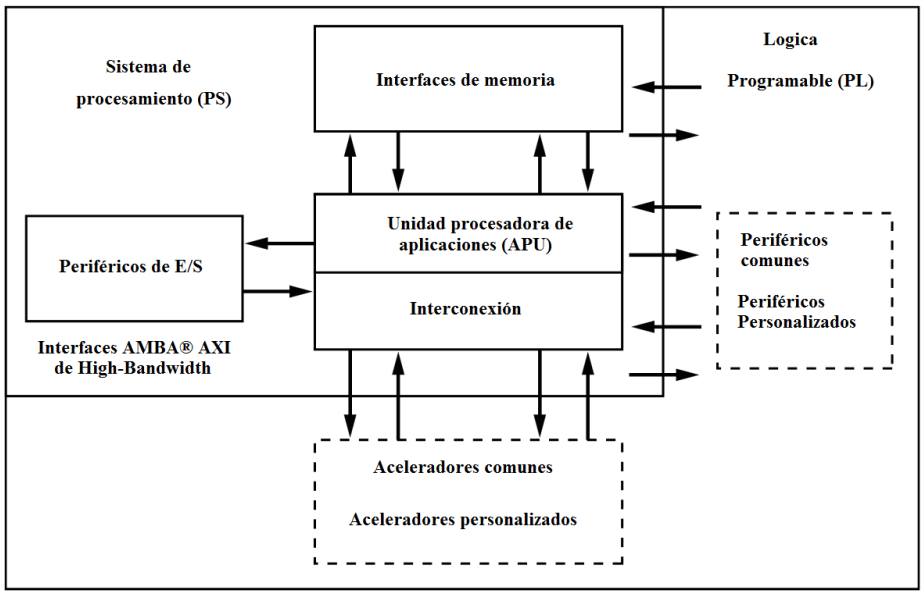


Figura 2.2. Diagrama de bloques de nivel alto del SoC.

En la parte del sistema de procesamiento se integran dos procesadores de aplicación Arm® Cortex ™ -A9 MPCore ™, también en esta parte se tienen interconexión AMBA®, memorias internas, interfaces de memoria externa y periféricos, incluidos USB, Ethernet, SPI, SD/SDIO, I2C, CAN, UART y GPIO. El PS se ejecuta independientemente del PL y se inicia al encender o reiniciar.

En la figura 2.3 se tiene los bloques funcionales del SoC Zynq-7000, que nos permiten tener una comprensión detallada a nivel bloques del funcionamiento del FPGA. Esto es sumamente importante en las etapas de diseño y configuración de nuestro sistema embebido en el FPGA. Se observa un área de suma importancia en nuestro análisis que es la parte "Unidad de procesador de aplicaciones" que está compuesta por unidades de procesamiento central (CPU) ARM Cortex-A9, cada ARM cuenta con Unidades de Punto Flotante FPU (Floating-Point Unit) con un motor NEON de medios, con doble memoria Cache de 32 KB, también cuenta MMU que es una unidad de control de memoria. Este análisis es de forma muy global.

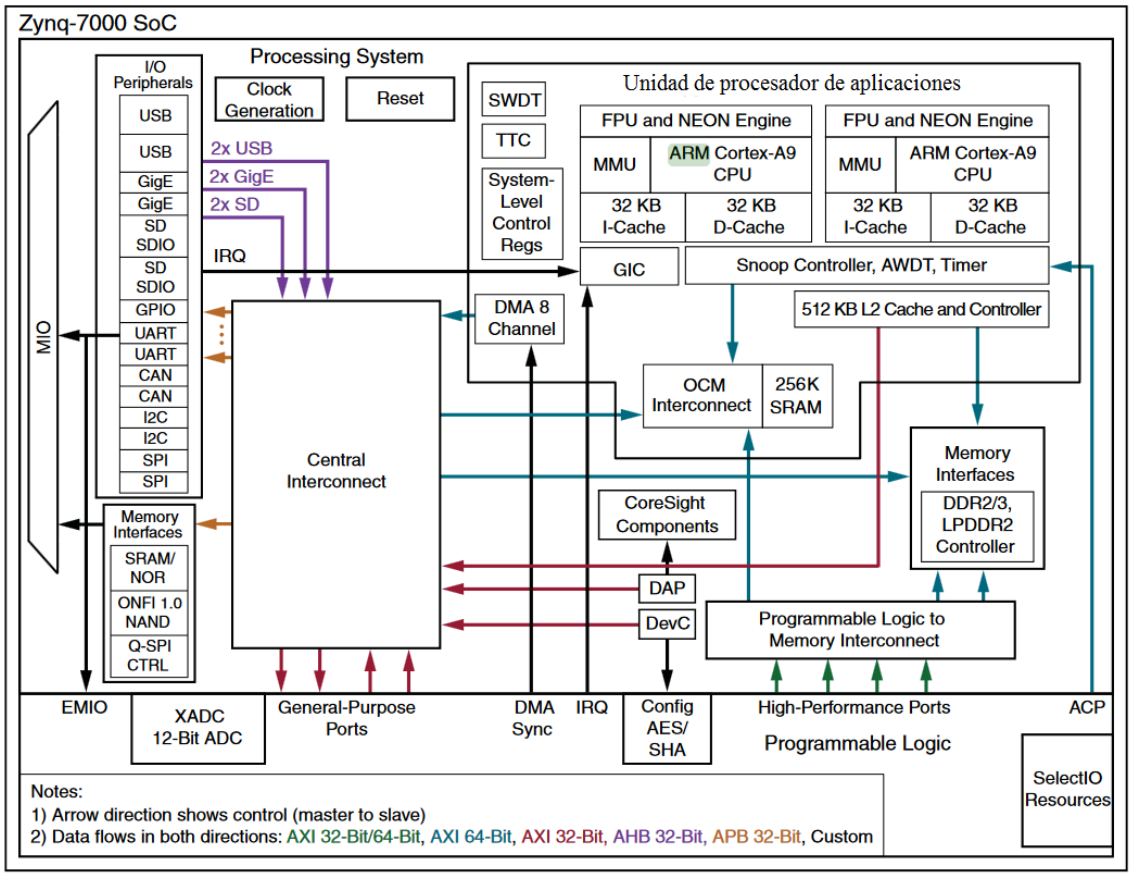


Figura 2.3 Bloques funcionales de la arquitectura Zynq-7000.

2.3 Procesador ARM Cortex-A9

El procesador Cortex-A9 es de alto rendimiento y bajo consumo de energía con un subsistema de caché L1 que proporciona capacidades de memoria virtual completa. Implementa una arquitectura ARMv7-A. y puede ejecutar instrucciones ARM de 32 bits, instrucciones *Thumb* de 16 bits y 32 bits. Las instrucciones *Thumb* se refiere de forma más precisa que admite un conjunto de instrucciones de longitud variable que proporciona instrucciones de 32 y 16 bits para mejorar la densidad del código.

Las principales características del núcleo Cortex-A9 son los siguientes:

Un motor de procesamiento de medios MPE (*Media Processing Engine*) brinda soporte para operaciones vectoriales de enteros y puntos flotantes. *NEON MPE* puede acelerar el rendimiento de las aplicaciones multimedia como los gráficos en 3D y el procesamiento de imágenes.

Una unidad de punto flotante FPU (Floating-Point Unit).

Predicción de rama dinámica

La unidad de captación previa implementa la predicción de ramificación dinámica de 2 niveles con un búfer de historial global (Global History Buffer GHB), un caché de dirección de destino aBranch (BTAC) y una pila de retorno.

Las señales de reinicio del procesador Cortex-A9, nCPURESET, nNEONRESET y nDBGRESET, le permiten reiniciar diferentes partes del procesador de forma independiente.

Soporte para administración avanzada de energía con hasta tres dominios de energía.

Compatibilidad con el programa Trace Macrocell (PTM) es una herramienta de monitoreo.

Procesadores de doble núcleo.
Existen una gran cantidad de tecnicismos y de tecnología implicada en estos procesadores que para tener una visión más global hacemos referencia al diagrama de bloques de alto nivel como es mostrado en la figura 2.4.

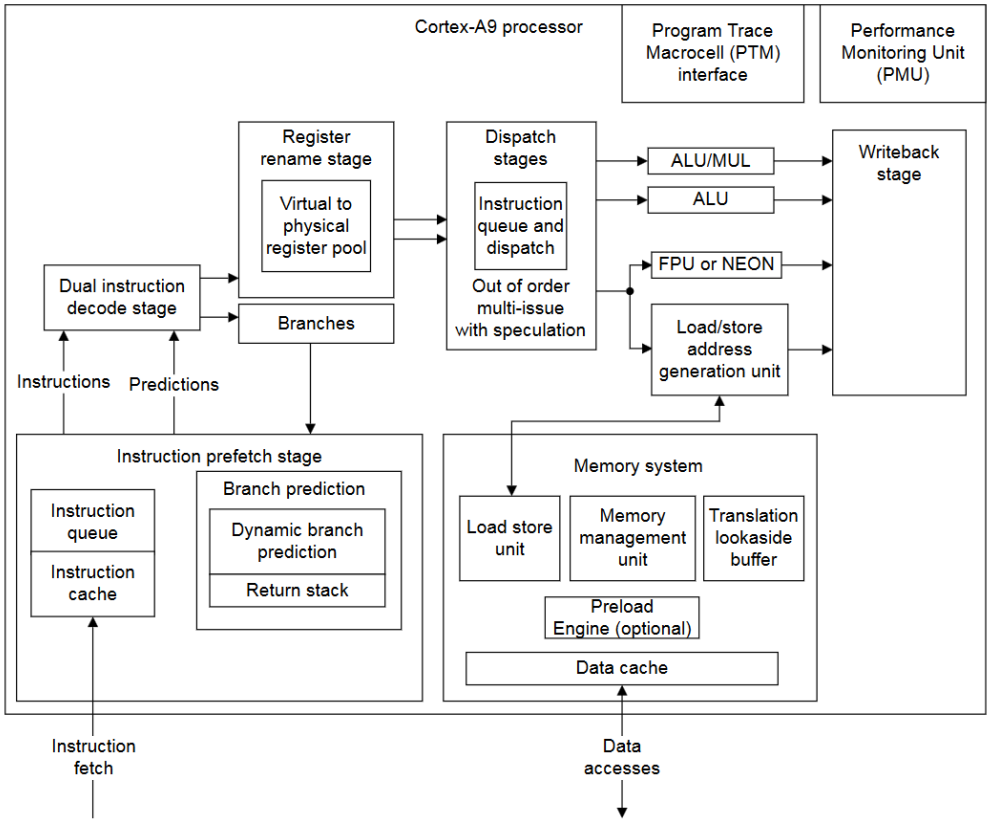


Figura 2.4 Diagrama de bloques de alto nivel del Procesador Cortex-A9

2.4 MicroBlaze procesador de núcleo flexible.

La unidad central de procesos MicroBlaze ™ es una familia de configuraciones de microprocesador RISC de 32 bits preestablecido y modificable. Esto se ilustra mejor en la figura 2.5. No se puede tener acceso a los detalles del Microblaze en VHDL. Pero si es posible trabajar con este recurso MicroBlaze. Es un producto comercial que cumple con los requisitos de muchas aplicaciones diversas, incluidos en los mercados: industriales, médicos, automotrices, de consumo y de comunicaciones.
Existen tres tipos de configuraciones pre establecidas para MicroBlaze:

- Microcontrolador (*MCS - Micro Controller System*): Que ejecuta aplicaciones simples, adecuado para ejecutar código de fuente abierto.
- Procesador en tiempo real: Procesamiento determinista en tiempo real en un sistema operativo R.T.O.S.
- Procesador de aplicaciones: Compatible con sistemas Linux empotrados.

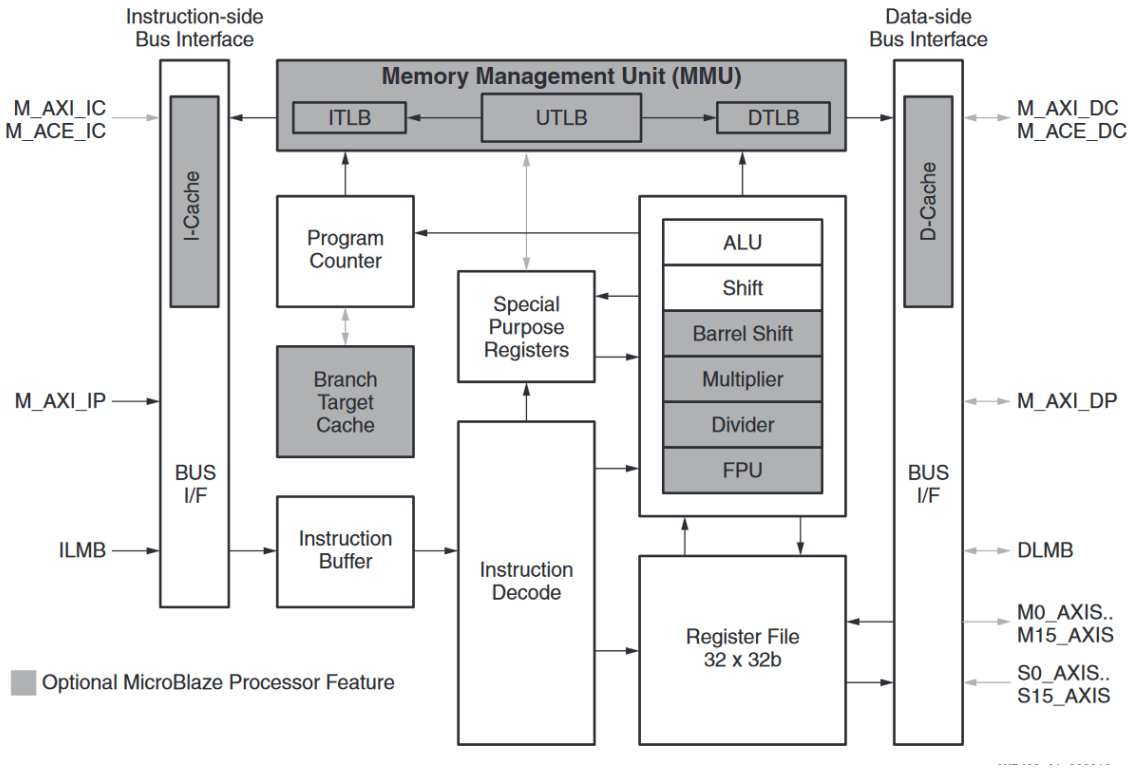


Figura 2.5 Diagrama de bloques del procesador MicroBlaze.

Características de MicroBlaze:
Más de 70 opciones configurables por el usuario.

- Tubería de 3 etapas para una huella óptima, tubería de 5 etapas para un rendimiento máximo.
- Soporta cualquier interfaz PLB o AXI.
- Soporta Big-endian or Little-endian.
- Unidad de gestión de memoria opcional (MMU).
- Unidad de punto flotante opcional (FPU)
- Instrucción y caché de datos

Una de las posibles configuraciones de MicroBlaze es de redundancia modular triple (Triple Modular Redundancy -TMR), de alta confiabilidad brinda detección, corrección y recuperación de errores flexible para los procesadores MicroBlaze que se ejecutan en FPGA de Xilinx.
Para la implementación y uso del MicroBlaze existen dos herramientas de diseño de XILINX:

- Suite de diseño ISE (Integrated Synthesis Environment).
- Suite de diseño VIVADO.

Ambas herramientas de diseño son muy parecidas, lo que cambia es el nivel de abstracción en la figura 2.6 se muestra que la herramienta ISE va de salida con las nuevas tecnologías de los FPGAs para poder aprovechar todas sus características Xilinx recomienda el uso de VIVADO.

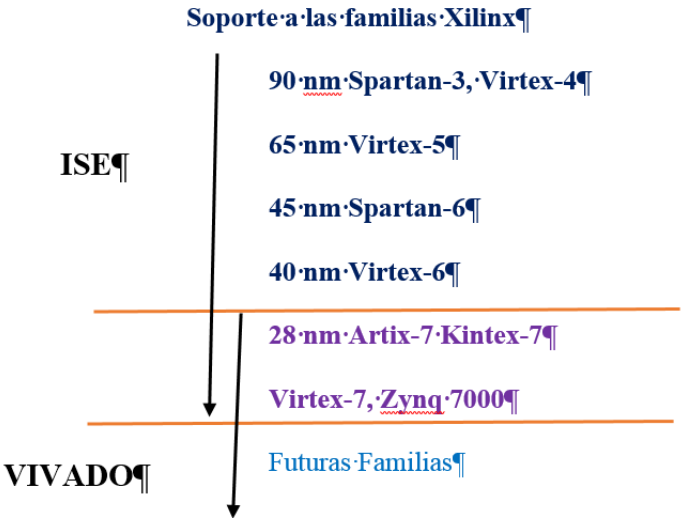


Figura 2.6 El uso de herramientas de diseño recomendadas por XILINX.

MicroBlaze se puede usar como un procesador independiente en todos los FPGA de Xilinx o como un coprocesador en un sistema SoC Zynq®-7000.
Se puede configurar una protección contra manipulaciones del sistema y protección de fallas mediante el modo de lock-step. Para el diseño de múltiples procesadores se usa el software Development kit SDK. En la figura 2.7 se muestra la herramienta SDK cuando se está cargando.

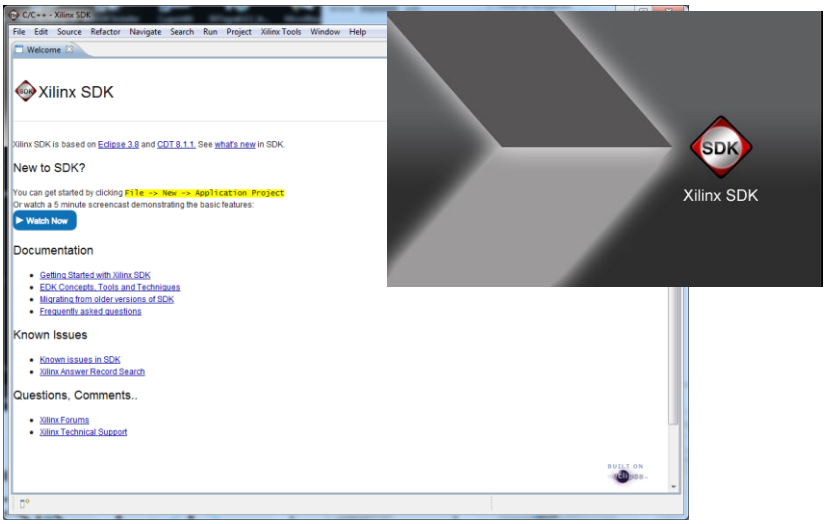


Figura 2.7. Entorno de Xilinx SDK

2.5 Software y Sistemas Operativos empotrados para la placa de evaluación ZC702 Xilinx

Existen múltiples opciones de software que incluyen Linux, Sistema Operativo en Tiempo Real (RTOS, por sus siglas en inglés), servicios básicos e incluso hipervisores y soluciones habilitadas TrustZone para la seguridad.
Tres categorías principales:

Linux soportados por MicroBlaze en un SoC Zynq-7000

- Linux, uBoot, more (Xilinx GIT)
- Yocto/OpenEmbedded support
  El Proyecto Yocto es un proyecto de colaboración de código abierto que proporciona plantillas, herramientas y métodos para ayudar a crear sistemas personalizados basados en Linux para productos integrados, independientemente de la arquitectura del hardware.

  Como proyecto de código abierto, el Proyecto Yocto opera con una estructura de gobierno jerárquica basada en la meritocracia y gestionada por su arquitecto principal, Richard Purdie, miembro de la Fundación Linux. Esto permite que el proyecto permanezca independiente de cualquiera de sus organizaciones miembros, que participan de varias maneras y proporcionan recursos para el proyecto.

- PetaLinux tools
  Las herramientas de PetaLinux
  Las herramientas ofrecen todo lo necesario para personalizar, construir e implementar soluciones de Linux embebidas en los sistemas de procesamiento Xilinx. Diseñada para acelerar la productividad del diseño, la solución funciona con las herramientas de diseño de hardware de Xilinx para facilitar el desarrollo de los sistemas Linux para Zynq® UltraScale + ™ MPSoC, Zynq®-7000 SoCs y MicroBlaze ™.

Sistemas operativos en tiempo real

- Xilinx Stand-alone (bare metal) Environment and Xilinx Software Development Kit (SDK)
  El kit de desarrollo de software de Xilinx (XSDK) es el entorno de diseño integrado para crear aplicaciones integradas en cualquiera de los microprocesadores de Xilinx: Zynq® UltraScale + MPSoC, Zynq-7000 SoCs y el microprocesador de núcleo suave MicroBlaze ™. El SDK es la primera aplicación IDE que ofrece un verdadero diseño, depuración y análisis de rendimiento de multiprocesadores homogéneos y heterogéneos.
- eCOS
  eCos es un sistema operativo gratuito de código abierto en tiempo real destinado a aplicaciones integradas. La naturaleza altamente configurable de eCos permite que el sistema operativo se personalice según los requisitos de aplicación precisos, ofreciendo el mejor rendimiento posible en tiempo de ejecución y una huella de recursos de hardware optimizada.

- Express Logic X-WARE IoT PLATFORM desarrollado por THREADX
  X-WARE IoT PLATFORM - impulsado por THREADX RTOS - posiblemente el RTOS más popular con más de 6.2 mil millones de implementaciones por investigación de VDC. X-WARE IoT PLATFORM incluye FILEX, GUIX, NETX, NETX DUO (TLS, DTLS, IPsec, MQTT, CoAP, LwM2M, Thread), USBX. Además de las ventajas de tamaño, rendimiento y facilidad de uso, ThreadX, FileX y NetX Duo cuentan con certificación previa IEC 61508 SIL 4, IEC 62304 Clase C, ISO 26262 ASIL D, UL / IEC 60730, UL / IEC 60335, UL 1998 y EN 50128 SW-SIL 4.

- SAFERTOS es un sistema operativo en tiempo real (RTOS) con certificación de seguridad y preventivo para procesadores embebidos. Ofrece un rendimiento superior y una fiabilidad precertificada, al tiempo que utiliza recursos mínimos. SAFERTOS está disponible con certificación previa IEC 61508 SIL 3 (Industrial) y ISO 26262 ASIL D (Referente a seguridad automotriz). SAFERTOS es compatible con las presentaciones de la FDA 510 (k) y la certificación IEC 62304 (Dispositivos médicos), así como la certificación de DO-178C DAL A (Aeroespacial).

Toda esta información se encuentra dada en xilinx en la parte de desarrollo embebido y solo tomamos la parte Zynq-7000 SoC.

En la tabla 2.1 los puntos donde nos enfocamos son las soluciones no comerciales, y también que sean soportadas por Microblaze como son: Linux, uBoot, more (Xilinx GIT), Yocto/OpenEmbedded support ,PetaLinux tools.

Linux

	Zynq-7000 SoC Support	**MicroBlaze Support**
No comercial		
Linux, uBoot, more (Xilinx GIT)	●	●
Yocto/OpenEmbedded support	●	●
PetaLinux tools	●	●
Arch Linux ARM	●	
Ubuntu Linux		
Android	●	
Soluciones Comerciales		
Enea Linux	●	
Mentor Embedded Linux	●	●
Star Lab Crucible	●	●
Suse Linux		●

Tabla 2.1 Distribuciones y proyectos con soporte Zynq-7000.

En la tabla 2.2 observamos el soporte de la ZYNQ-7000 y del microprocesador MicroBlaze

RTOS Bare-metal

Operating System	**Safety Certifiable**	**Zynq-7000 Support**	**MicroBlaze Support**
Xilinx Stand-alone (bare metal) Environment and Xilinx Software Development Kit (SDK)		●	●
DDC-I DEOS	●	●	●
eSol eT-kernel	●	●	●
eCOS		●	●
eForce		●	●
ENEA OSE		●	●
Etas RTA-OS	●	●	
Express Logic X-WARE IoT PLATFORM powered by THREADX	●	●	●
Green Hills Software INTEGRITY	●	●	●
LynxOS 7.0	●	Contact Partner	●
Mentor Nucleus	●	●	Contact Partner
Micrium uC/OS-II	●	●	●
Micrium uC/OS-III		●	●
RTEMS	●	●	●
Sciopta	●	●	●
Segger embOS		●	
Wittenstein OpenRTOS		●	Contact Partner
Wittenstein SafeRTOS	●	●	●

Tabla 2.2 Soporte en Tiempo real para Zynq-7000.

En esta última tabla 2.3 se hace referencia de más herramientas que son compatibles con el sistema a crear en esta tesis.

Soluciones multi-OS y Multiprocesamiento asimétrico.

Software	**Zynq-7000 Support**
Dave Embedded FreeRTOS/Linux on the Bora board	●
Express Logic X-WARE IoT PLATFORM powered by THREADX	●
General Dynamics Mission Systems OKL4	
Green Hills INTEGRITY Multivisor	
Lynx Secure Separation Kernel Hypervisor	
Mentor Hypervisor	●
Mentor Trusted Execution Environment	●
QNX Hypervisor	
Wind River Hypervisor	Contact Partner
Xen Hypervisor	

Tabla 2.3 Soporte en soluciones multi-OS y Multiprocesamiento asimétrico para Zynq-7000.

2.6 PROFINET

Se ha convertido en el estándar de Ethernet industrial líder en los mercados de control de procesos y automatización de fábricas. PROFINET satisface una amplia gama de requisitos, desde la asignación de parámetros de uso intensivo de datos hasta la transmisión de señales de E/S síncronas. La comunicación se realiza a través del mismo cable en todas las aplicaciones, desde tareas de control simples hasta aplicaciones de control de movimiento altamente exigentes. Paralelamente a la comunicación en tiempo real de PROFINET, la comunicación basada en TCP/IP también es posible al mismo tiempo.

PROFINET es el innovador estándar abierto para Industrial Ethernet. PROFINET satisface todos los requisitos de la tecnología de automatización. En la figura 2.8 se muestra el entorno operativo de una topología PROFINET de forma esquemática.

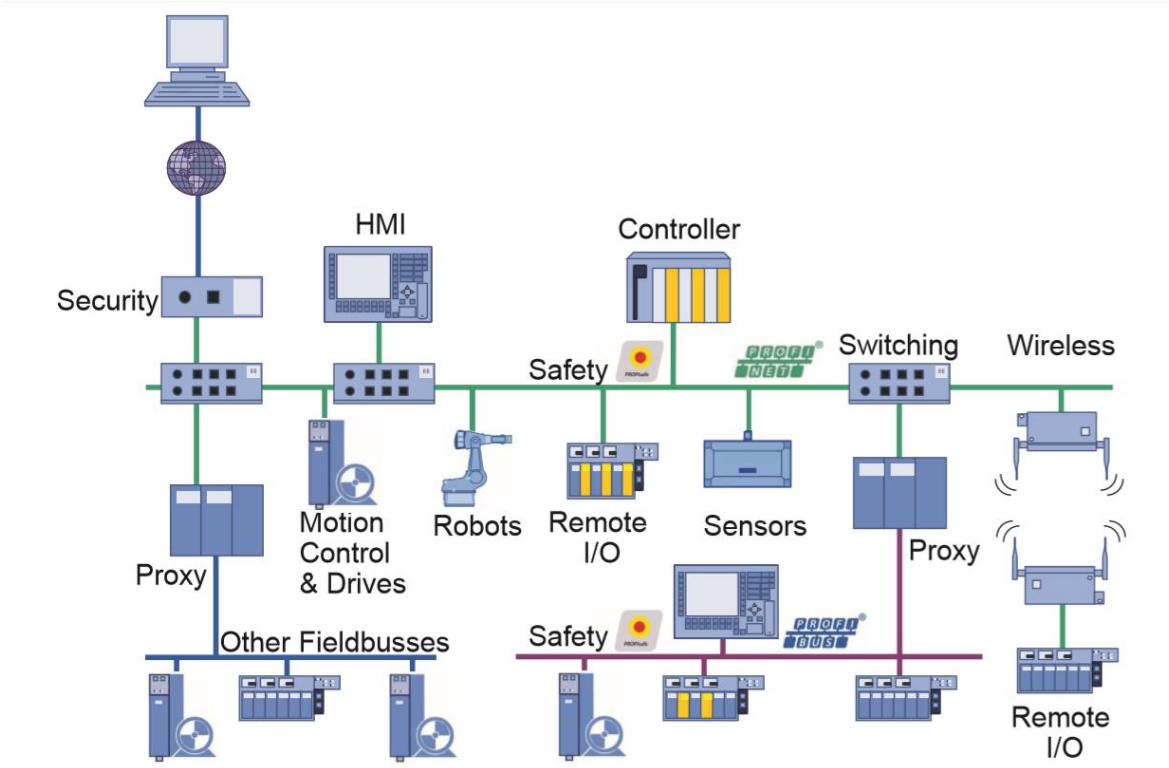


Figura 2.8 Topología PROFINET en un entorno de automatización. [12]

Ya sea que la aplicación implique automatización de producción, en procesos o unidades (con o sin seguridad funcional), PROFINET es la primera opción en todos los ámbitos. Como una tecnología estándar en la industria automotriz, ampliamente difundida en la construcción de máquinas y bien probada en las industrias de alimentos, empaques y logística, PROFINET ha encontrado su camino en todas las áreas de aplicación. Las nuevas áreas de aplicación están emergiendo constantemente, como las aplicaciones marítimas y ferroviarias o incluso las operaciones diarias, por ejemplo, en una tienda de bebidas. [12]

Y ahora: el nuevo perfil de tecnología PROFIenergy mejorará el balance energético en los procesos de producción. PROFINET está estandarizado en IEC 61158 e IEC 61784. El desarrollo continuo de PROFINET ofrece a los usuarios una visión a largo plazo para la implementación de sus tareas de automatización. Para los fabricantes de plantas y máquinas, el uso de PROFINET minimiza los costos de instalación, ingeniería y puesta en servicio. Para los propietarios de plantas, PROFINET ofrece una fácil expansión de la planta y una alta disponibilidad de la planta debido al funcionamiento autónomo de las unidades de planta y los bajos requisitos de mantenimiento. La certificación obligatoria de los dispositivos PROFINET también garantiza un estándar de alta calidad. [12]

2.6.1 Características en el uso de PROFINET

PROFINET es el estándar de comunicación para la automatización de PROFIBUS & PROFINET International (PI). La gama modular de funciones hace de PROFINET sea una solución flexible para todas las aplicaciones industriales. A continuación, se listan las ventajas o características para el uso de PROFINET

1. Facilidad de uso, minimiza los costos de instalación, ingeniería y puesta en servicio. Y logrando que la automatización es más rápida y eficiente.
2. Flexibilidad en su topología de red. PROFINET es 100% compatible con Ethernet según los estándares IEEE en soluciones de cable de cobre y fibra óptica. PROFINET y permite la comunicación inalámbrica con WLAN y Bluetooth.
3. Diagnóstico integrado. PROFINET incluye conceptos de diagnóstico inteligente para dispositivos de campo y redes.
4. Seguridad integrada. La tecnología probada de seguridad PROFIsafe de PROFIBUS también está disponible para PROFINET.
5. Alta disponibilidad. PROFINET integra automáticamente soluciones de redundancia de reacción.
6. La comunicación escalable en tiempo real esto es fundamental en líneas de producción.
   Se realiza a través del mismo cable en todas las aplicaciones, desde tareas de control simples hasta aplicaciones de control de movimiento altamente exigentes. Para tareas de control de bucle cerrado de alta precisión, es posible la transmisión determinística e isócrona de datos de proceso críticos con el tiempo con una fluctuación de menos de 1 µs.
7. Estructuras de sistemas ampliados. Además de la estructura de automatización convencional que consiste en un controlador y sus dispositivos de campo, también se pueden realizar estructuras jerárquicas con dispositivos de campo inteligentes y el uso compartido de dispositivos de campo y módulos de entrada por múltiples controladores.
8. Todo en un solo cable. Con su comunicación integrada basada en Ethernet, PROFINET satisface una amplia gama de requisitos, desde la asignación de parámetros de uso intensivo de datos hasta la transmisión de datos de E/S extremadamente rápida. PROFINET permite así la automatización en tiempo real. Además, PROFINET proporciona una interfaz directa al nivel de TI.
9. Soporte para la optimización de la energía.
   Con el perfil PROFIenergy integrado en los dispositivos PROFINET, el uso de la energía en un sistema de automatización se puede medir mediante un método estandarizado y se puede controlar activando y desactivando las funciones de forma selectiva sin hardware adicional.
10. Soporte global.
    Las especificaciones y la documentación se preparan dentro de la organización global PROFIBUS & PROFINET International (PI). La capacitación y consultoría son proporcionadas por una red global de Centros de Competencia. El establecimiento del proceso de certificación comprobado garantiza un alto nivel de calidad para los productos PROFINET y su interoperabilidad en las plantas. [12]

2.6.2 La comunicación con PROFINET

Requisitos en tiempo real. El estándar 802.3 de IEEE para Ethernet está diseñado para garantizar una comunicación sin problemas entre los dispositivos de automatización de PROFINET y entre los dispositivos de automatización de datos y otros dispositivos Ethernet estándar. Para aplicaciones con estrictos requisitos en tiempo real, PROFINET ofrece mecanismos que permiten que la comunicación estándar y en tiempo real coexista en

paralelo. La comunicación con PROFINET se puede escalar utilizando tres niveles de rendimiento que se construyen entre sí:
La transmisión de datos de ingeniería y datos no críticos en el tiempo se produce a través del Protocolo de Control de Transmisión/ Protocolo de Internet (TCP / IP). Esta comunicación estándar es posible entre todos los dispositivos de automatización.
El canal en Tiempo Real (RT) está disponible para la transmisión de datos de proceso.
Para aplicaciones isócronas como el control de movimiento, se usa comunicación isócrona en tiempo real (IRT). Esto permite una frecuencia de reloj de <1 ms y un jitter de <1 μs.
La capacidad de IRT se basa en el soporte de hardware en el dispositivo, lo que significa que para este propósito se deben usar circuito integrado de aplicación específica (Application-Specific Integrated Circuit ASIC), microcontroladores y arreglos de puertas programables (FPGA). Los ASIC de conmutador comercial sin soporte de hardware IRT son adecuados para implementar un dispositivo de automatización solo con capacidades RT. Los dispositivos con comunicación RT se pueden desarrollar en base a componentes Ethernet estándar y una pila de software PROFINET. [12]

2.6.3 Clases de conformidad PROFINET

Para cumplir con los diferentes requisitos de los sistemas de automatización, se definen tres clases de conformidad para PROFINET. Cada clase tiene un alcance funcional determinado para el área típica de aplicación. El fabricante del dispositivo debe considerar la clase de conformidad requerida antes de seleccionar la opción de complementación para la interfaz del dispositivo PROFINET porque el tipo de implementación de la interfaz afecta a la clase de conformidad que se puede lograr. Las funciones clave de las tres clases de conformidad y sus ventajas se describen a continuación:

- CC-A: Uso de Infraestructura de una red de Ethernet existente, incluida la integración de las funciones básicas de PROFINET. Todos los servicios de tecnología de la información (TI) se pueden utilizar sin restricciones. Los ejemplos de aplicaciones típicas se encuentran en la automatización de edificios y la automatización de procesos.
- CC-B: El alcance funcional de CC-B comprende las funciones de CC-A, además de que es compatible con el reemplazo de dispositivos de fácil uso sin necesidad de una herramienta de ingeniería. El protocolo simple administración de red (Simple Network Management Protocol SNMP) admite diagnósticos de dispositivos extendidos de funciones de red, como mensajes de estado de puertos. Para aumentar la fiabilidad, hay disponible como opción un protocolo de redundancia de medios adaptado al rendimiento. Todos los servicios de TI se pueden utilizar sin restricciones. Las aplicaciones típicas se pueden encontrar en sistemas de automatización con control de máquinas de nivel superior con un ciclo de datos determinístico, pero no isócrono. La gran mayoría de los dispositivos PROFNET caen en este campo.
- CC-C: El alcance funcional de CC-C abarca todas las funciones de CC-B. También es compatible con la transmisión de datos determinista y de alta precisión, incluidas las aplicaciones isócronas. La redundancia de medios opcionales integrada permite una conmutación suave del tráfico de datos de E/S si se produce una falla. Todos los servicios de TI pueden ser utilizados sin restricciones. Las aplicaciones típicas se encuentran en el campo del control de movimiento. [12]

2.7 PROFIBUS

Es el bus de campo más exitoso del mundo, con una base instalada de casi cien millones de dispositivos en automatización de fábricas y aplicaciones de la industria de procesos. Al utilizar un protocolo de comunicación único, coherente e independiente de la aplicación, PROFIBUS admite soluciones de bus de campo tanto en la automatización de fábricas como de procesos. así como en tareas de control de movimiento y seguridad. [12]

IO-Link

Es una tecnología de comunicación abierta, estandarizada en todo el mundo, que vincula sensores y actuadores (dispositivos) hasta el nivel de entrada y salida independientemente del sistema de bus de campo anterior. Ofrece ventajas para el usuario, como posibilidades de diagnóstico, asignación de parámetros a los dispositivos y beneficios de cableado. Su gran éxito ha llevado a continuas mejoras. Por ejemplo, IO-Link Safety permite aplicaciones de seguridad e IO-Link Wireless se encuentra actualmente en preparación. [12]

Capítulo 3

Implementación y diseño de un sistema embebido en un ZYNQ-7000 EPP ZC702 Evaluation kit.

En este capítulo se hace el planteamiento de las herramientas de software a emplear para la creación del sistema embebido, el segunda paso es la creación de la arquitectura del sistema embebido (Hardware de núcleo duro) y como tercer paso se establece la implementación de un sistema operativo embebido (Software).

3.1 Parámetros de comunicación y configuración entre la PC y el Kit Desarrollo ZC702.

Primero se instalan los controladores en la PC del dispositivo Silicon Labs CP210x USB to UART Bridge (COM3) que habilitan la comunicación del Puente de USB a UART.

https://www.silabs.com/documents/public/software/CP210x_Windows_Drivers.zip.

Como segundo paso definimos el uso de la interface Digilent USB a JTAG interface U23. Por el cual el DIP switch SW10 queda determinado como:

SW10 → 01

También es necesario tener presente la tabla 3.1 para ver las diferentes opciones de configuración del switch SW16.

Boot Mode	SW16.1	SW16.2	SW16.3	SW16.4	SW16.5
JTAG mode(1)	0	0	0	0	0
Independent JTAG mode	1	0	0	0	0
Quad SPI mode	0	0	0	1	0
SD mode	0	0	1	1	0
MIO configuration pin	MIO2	MIO3	MIO4	MIO5	MIO6

Tabla 3.1 Opciones de configuración operativas.

Finalmente, otro dato importante es la configuración de la comunicación serial entre la PC y la tarjeta ZC702, se da conforme a la tabla 3.2

Variable	Valor
Baud rate:	115200
Data:	8
Parity:	none
Stop:	1 bit
Flow Control:	none

Tabla 3.2 Parámetros de la comunicación Serial.

3.2 Creación del sistema base con el Kit de desarrollo ZC702.

En esta parte se crea el soporte de nuestro sistema embebido con los dispositivos de trabajo el primero es **Cortex-A9** y el segundo dispositivo es el **kit de desarrollo XC7Z020**.
Con la herramienta PlanAhead 14.2 se crea un nuevo proyecto con la siguiente información:

- Nombre del proyecto CTT002
- Definiendo un directorio de trabajo para este proyecto. En este directorio de forma interna se generarán otros subdirectorios que serán su directorio de trabajo de otras herramientas de diseño de forma jerarquica.
- Tipo de Proyecto
  - **Proyecto RTL(Register Transfer Level)**
  - **Proyecto Post-Synthesis**
  - **Proyecto de E/S.**
  - **Importar el entorno ISE y los resultados de la ruta**
  - **Importar proyecto**

  Nosotros tomamos la opción "**Proyecto RTL**" Puede ver un diseño sintetizado como un esquema en un visor de **Nivel de Transferencia de Registro** (RTL). Esta vista muestra compuertas y elementos independientemente del dispositivo Xilinx® objetivo.
- Se agregan los módulos de diseño:
  - Add Sources, En esta parte damos la opción de "Agregar o crear fuentes embebidas" Es aquí donde definimos las unidades de sub-diseño embebido seleccionando los archivos fuentes XMP.

También de forma automática es creado el archivo system.xmp en C:\xup\embedded\ISE14_2\CTT002\CTT002.srcs\sources_1\edk\system\system.xmp

- Add Existing IP,
- Add Constraints

- Se especifica los datos de nuestra tarjeta **ZYNQ-7 ZC702 Evaluation**

Después usamos la herramienta XPS(Xilinx Platform Studio) ver 14.2
De forma interna es llamada otra sub-herramienta llamada "Base System Builder BSB" si se traduce al español seria generador del sistema base. La cual nos conduce a la creación de nuestro sistema. Como primer punto en BSB definimos un tipo de interconexión en nuestro caso seria "AXI System" para sacar provecho a nuestro hardware ya que contiene procesadores Dual Core ARM Cortex-A9.
AXI es un estándar de interfaz más o menos recientemente adoptado por Xilinx como la interfaz estándar utilizada para todas las versiones actuales y futuras tecnologías de IPs de Xilinx.
Dentro de esta herramienta SBS se muestra las anteriores configuraciones de nuestro sistema como son:

Datos de la tarjeta de desarrollo		
Xilinx	ZYNQ ZC702 Evaluation Plataform	C
Vendedor	Nombre de la tarjeta	Versión
Configuración de la tarjeta		
Zynq	Xc7z020	200 MHZ
Arquitectura	Dispositivo	Frecuencia de reloj
Clg484	-1	
Empaque	Grado de velocidad	

Tabla 3.3 Datos del kit desarrollo ZC702.

Lo importante de esta parte es establecer el uso de "Zynq Processing System 7" El núcleo del sistema de procesamiento actúa como una conexión lógica entre el PS y el PL mientras lo ayuda a integrar núcleos IP personalizados e integrados con el sistema de procesamiento. En este punto la herramienta crea todos los archivos que debe crear para cerrar esta sub-herramienta y retorna a la herramienta de diseño XPS. Se muestra en la figura 3.1 el estatus actual de nuestro diseño.

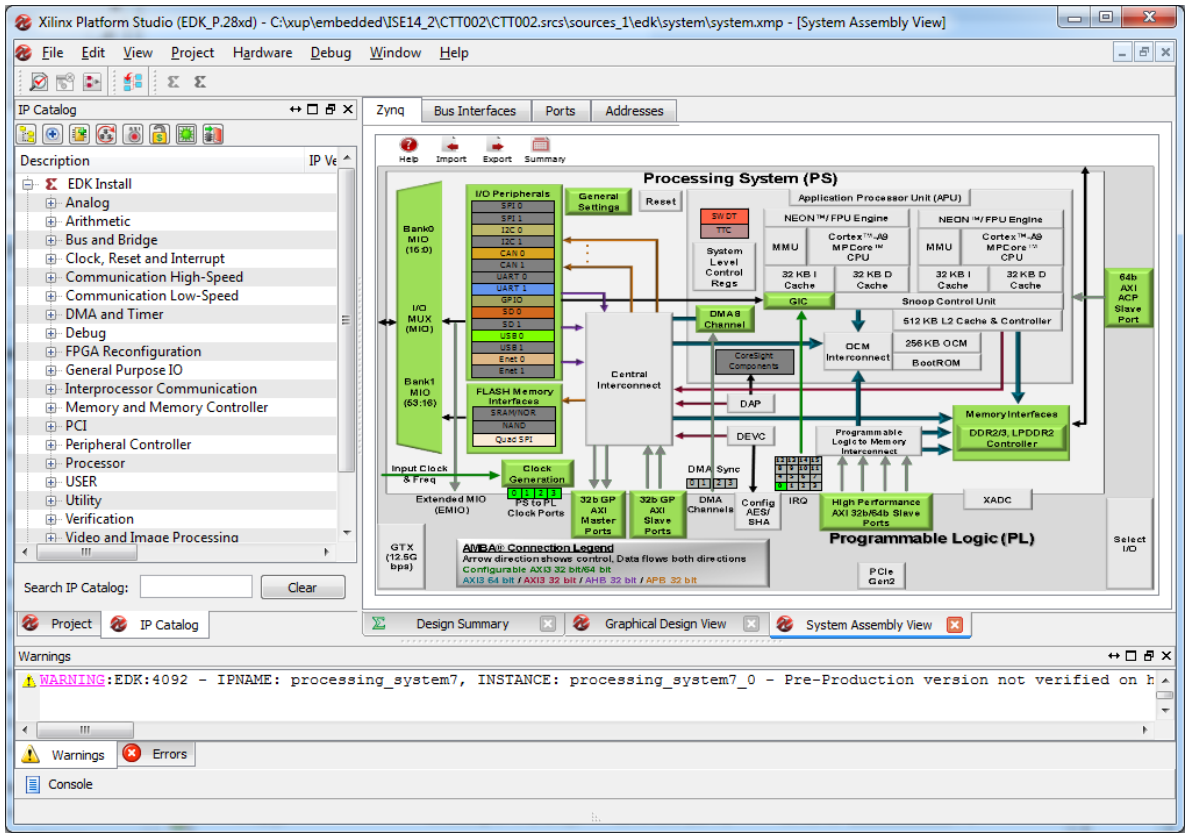


Figura 3.1 Vista de ensamblaje del sistema ZYNQ en el EDK

Como siguiente paso cerramos XPS y retornamos a la herramienta PlanAhead, de forma automática todos los cambios hechos en XPS son actualizados en PlanAhead, en la figura 3.2 se muestra de forma gráfica sobre el desarrollo de nuestro proyecto. Aún no se hecho el bitstream más adelante se realizará.

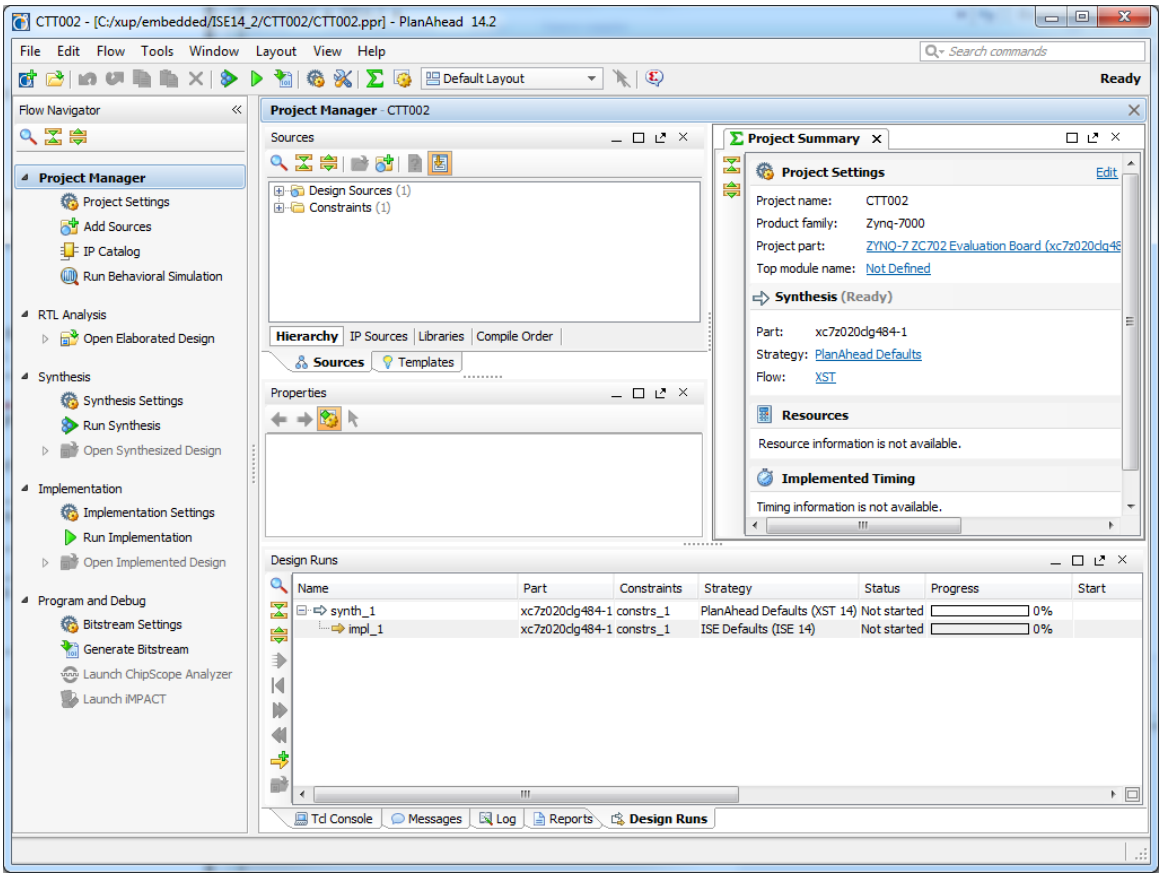


Figura 3.2 PlanAhead nos muestra el estatus de implementación del proyecto.

En este punto se debe exportar el diseño a la herramienta "Software Development Kit" (SDK), para hacer eso se requiere ir al menú principal de PlanAhead y seleccionamos las siguientes opciones que son **File > Export > Export Hardware for SDK…** Con esta acción exportamos el archivo system.xmp al SDK. En la Figura 3.3 se muestra el informe de nuestro diseño.

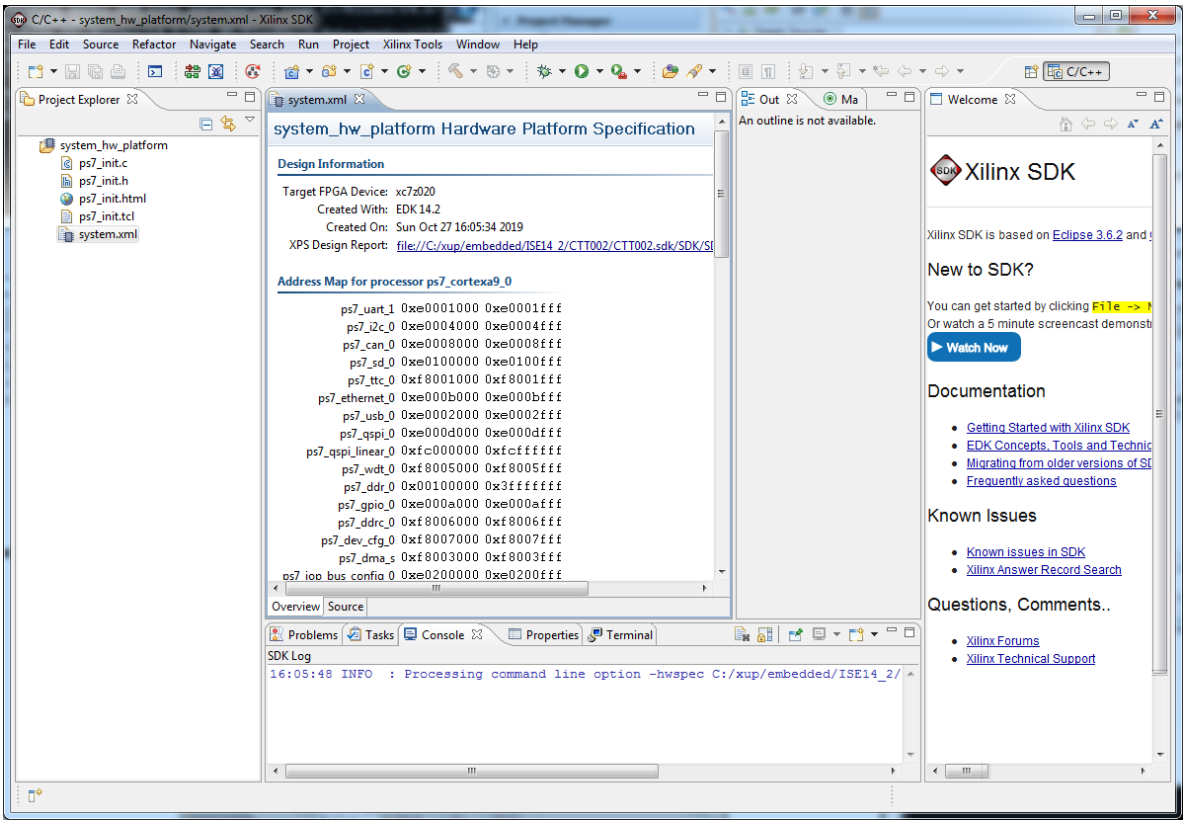


Figura 3.3 SDK Especificaciones de la plataforma ZC702.

Proyecto system_hw_plataform
En la SDK crea los archivos ps7_init.c y ps7_init.h, donde básicamente es contenido el código de inicialización del sistema de procesamiento Zynq y la configuración de inicialización para DDR(Memoria de componente DDR3 de 1 GB con cuatro dispositivos de 256 Mb x 8), relojes, PLLs y MIO.
Otro archivo que también es creado por SDK es el ps7_init..tcl, se usa en las configuraciones Debug/Run en la herramienta SDK, o desde la línea de comandos Tcl.
El ultimo archivo es ps7_init.html donde está en formato html se encuentran descritas los alcances de nuestro proyecto.

Proyecto hello_world_0
Para crear este proyecto vamos al menú principal del SDK y marcamos las opciones **File > New > Xilinx C Project**
Dentro de la herramienta SDK existen plantillas de programas fuentes, la cual tomamos uno llamado Hello World donde es definido un nuevo proyecto llamado hello_world_0 en esta parte damos de alta el procesador ps7_cortexa9_0 y declaramos el software incrustado como Standalone.

Proyecto hello_world_dsp_0
En el siguiente paso se selecciona el **Board Support Package** para este proyecto.

Ya teniendo nuestros tres proyectos:

- hello_world_0

- hello_world_dsp_0
- system_hw_plataform

se marca el proyecto hello_world_0 para correrlo como **Xilinx C/C++ ELF.** Y es creado el menú **hello_word_0 Debug.** Corremos nuestros Proyectos y ejecutamos la programación del FPGA. En la figura 3.4 se observa que ya están disponibles los dos dispositivos de la plataforma de nuestro sistema embebido.

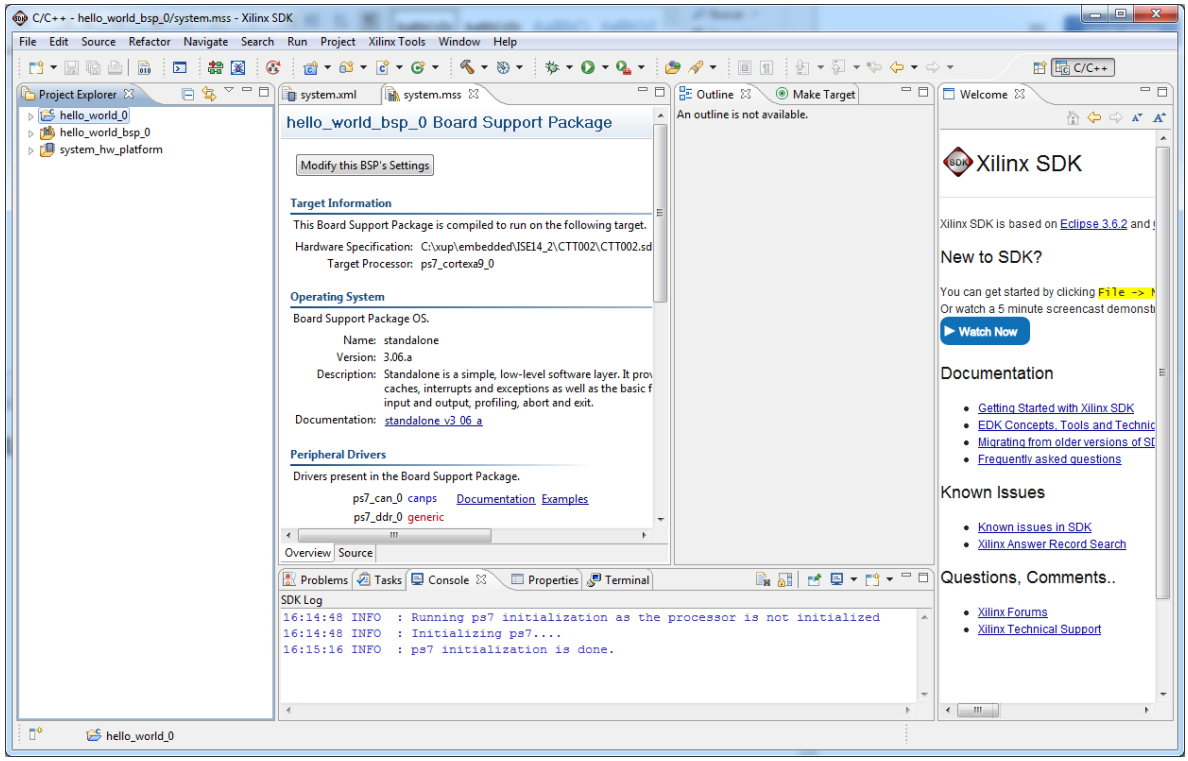


Figura 3.4 En SDK ya están inicializados los dispositivos de la plataforma.

3.3 Implementación del sistema embebido tomando un núcleo de MicroBlaze.

Desde la herramienta PlanAhead se invoca la herramienta XPS, dando doble clic en el archivo system.xmp, en esta parte son agregados al diseño todos los dispositivos que son descritos en el diagrama de bloques de la figura 3.5.

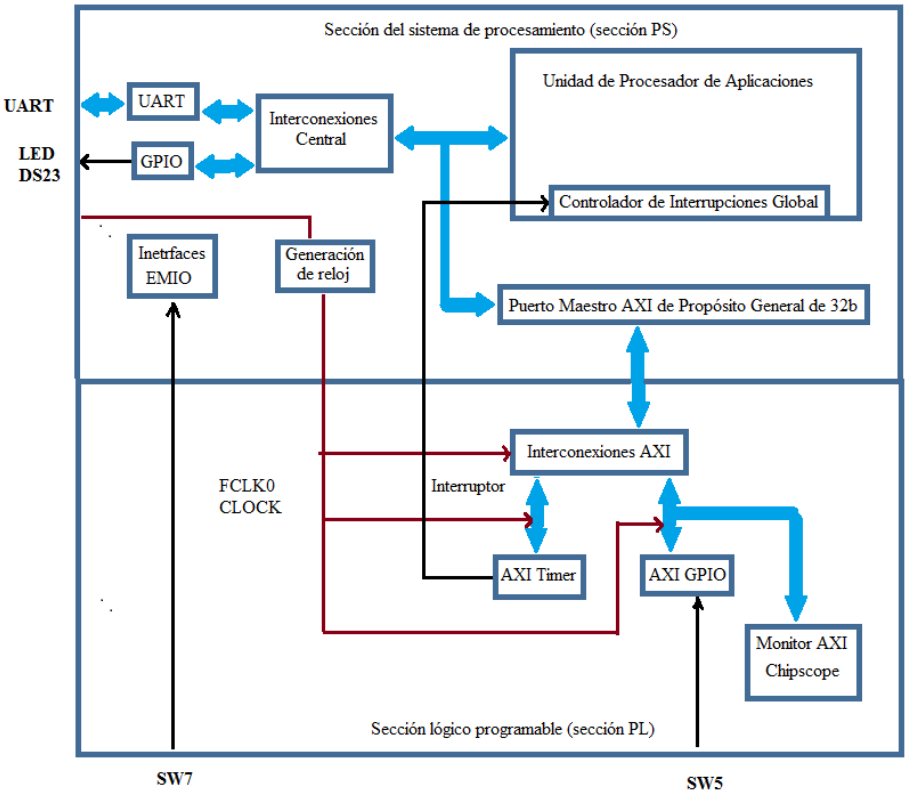


Figura 3.5 Diagrama a bloque del sistema embebido. [15]

Ya teniendo en mente el diagrama a bloques se procede a describir todos los dispositivos a trabajar como son descritos en la tabla 3.4.

IP	Puerto	Conexión
axi_interconnect_1	INTERCONNECT_ACLK	processing_system7_0 : FCLK_CLK0
	INTERCONNECT_ARESETN	processing_system7_0::FCLK_RESET0_N
axi_gpio_0	(BUS_IF) S_AXI::S_AXI_ACLK	processing_system7_0: FCLK_CLK0
	(IO_IF) gpio_0::GPIO_IO	External Port ::axi_gpio_0_GPIO_IO_pin
axi_timer_0	(BUS_IF) S_AXI_::S_AXI_ACLK	processing_system7_0 : FCLK_CLK0
Chipscope_axi_monitor_0	CHIPSCOPE_ICON_CONTROL	Chipscope_icon_0 ::control0
	(BUS_IF) MON_AXI:: MON_AXI_ACLK	processing_system7_0 : FCLK_CLK0
Chipscope_icon_0	Control0	Chipscope_axi_monitor0::CHIPSCOPE_ICON_CONTROL
Processing_system7_0	(BUS_IF) M_AXI_GP0:: M_AXI_GPO_ACLK	processing_system7_0 :: FCLK_CLK0

Tabla 3.4 IP's montados en el diseño.

Después se procede a Conectar la interrupción del temporizador en el lado del controlador de interrupción del lado PS. Esto es mostrado en la figura 3.6.

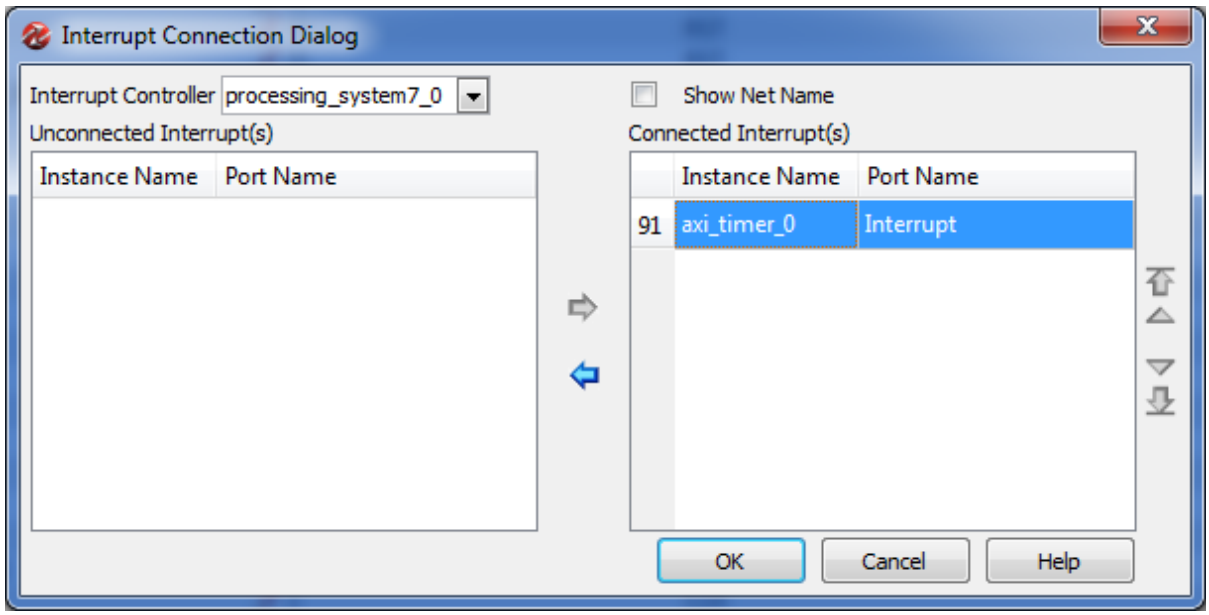


Figura 3.6 Dialogo de conexiones del interruptor.

Dentro de la herramienta XPS se checa las reglas del diseño que no exista ninguna inconsistencia y para hacer esto se va al menú principal y se ejecuta las siguientes opciones **Project → Design Rule Check.,** queda el diseño como es mostrado en la figura 3.7.

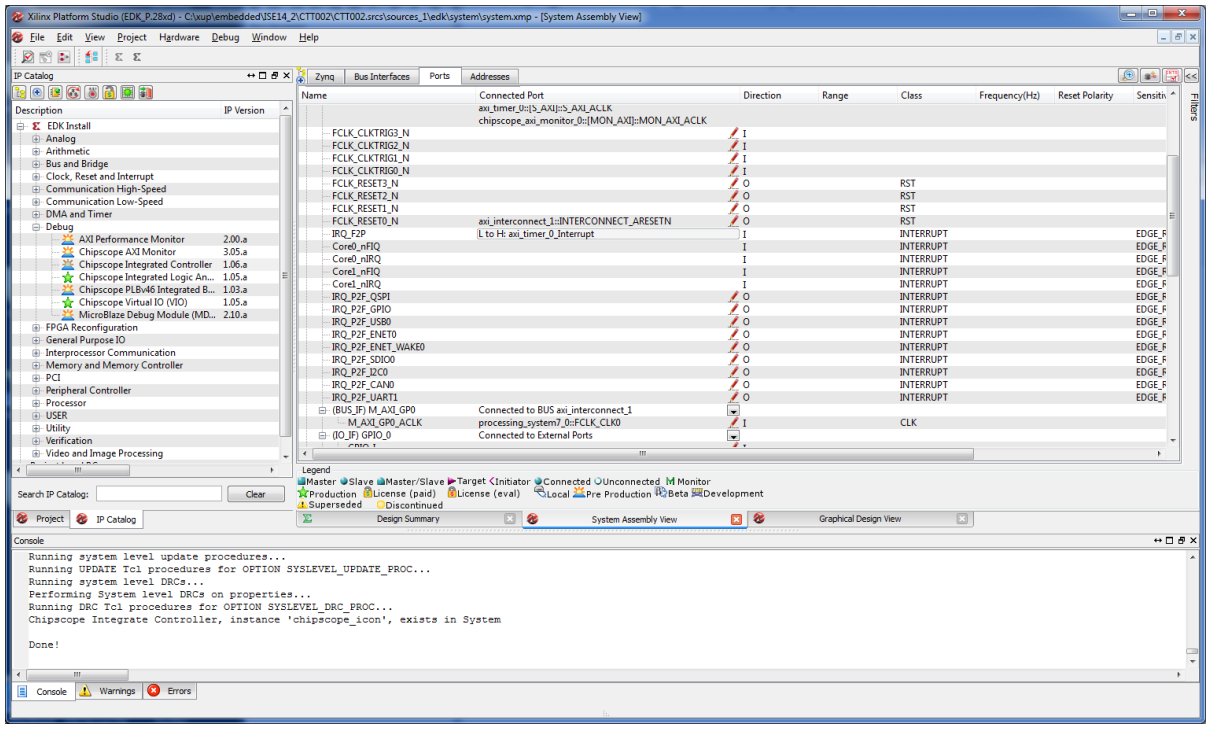


Figura 3.7 EDK después de haber checado las reglas de diseño.

Después cerramos XPS y en PlanAhead es creado top del HDL, para dar lugar en la creación del archivo de restricciones system.ucf, el siguiente paso se genera el Bitstream de nuestro diseño. Para tener la pantalla de PlanAhead como es mostrado en la figura 3.8.

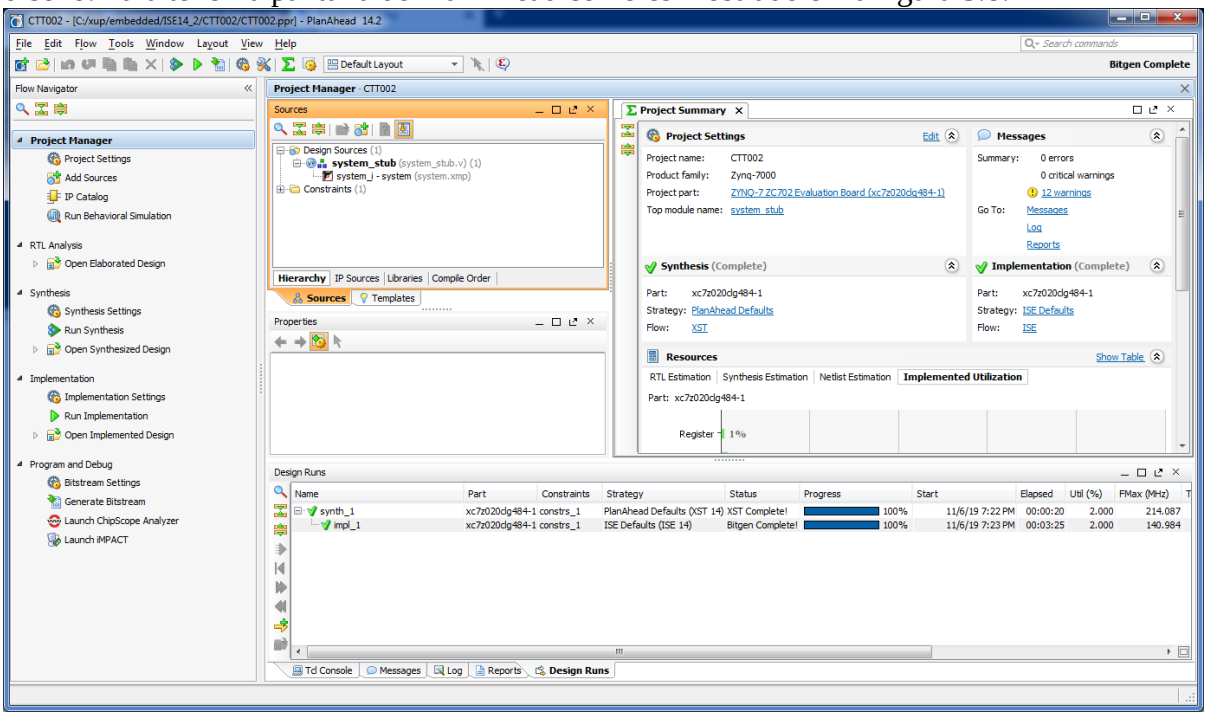


Figura 3.8 Después haber generado el Bitstream.

Se invoca la herramienta SDK para proseguir con el diseño, con la siguiente opción: **File > Export > Export Hardware for SDK..**
Se hacen todas las modificaciones pertinentes a nuestro proyecto y programamos el FPGA como es denotado en la figura 3.9

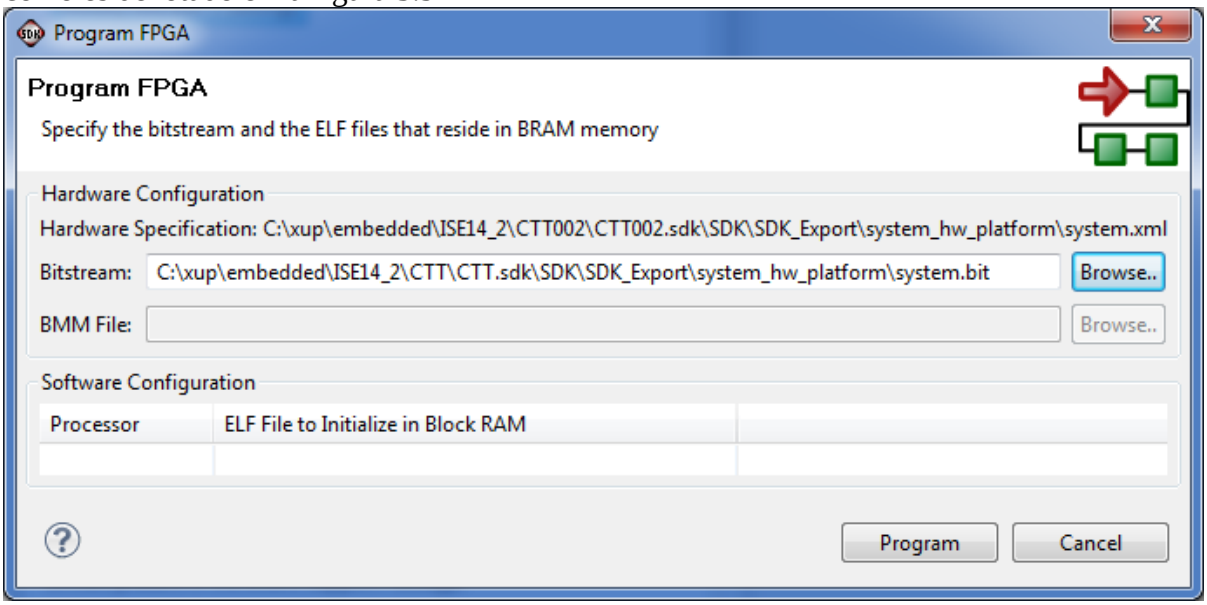


Figura 3.9. Programación del FPGA.

3.4 Instalación de Sistema Operativo Linux y depuración de aplicaciones usando SDK.

En esta parte se describe los pasos para iniciar el sistema operativo Linux en la placa Zynq ™ -7000. También proporciona información sobre la descarga de imágenes precompiladas por Linux en la memoria de destino utilizando una interfaz JTAG. Los archivos pre compilados son los siguientes:

- BOOT.bin: imagen binaria que contiene las imágenes FSBL y U-Boot producidas por bootgen.
- boot.bif: el archivo para controlar bootgen durante la creación de BOOT.BIN.
- cdma_app: software de aplicación independiente para el sistema que creará.
- devicetree.dtb: objeto grande binario del árbol de dispositivos utilizado por Linux, cargado en la memoria por U-Boot.
- helloworld.c: software de aplicación independiente para el sistema que se creó.
- linux_cdma_app: software de aplicación basado en el sistema operativo Linux.
- ramdisk8M.image.gz: imagen de Ramdisk utilizada por Linux, cargada en la memoria por U-Boot.
- README.txt: descripción de la versión.
- u-boot.elf: archivo U-Boot utilizado para crear la imagen BOOT.BIN.
- zImage: imagen del kernel de Linux, cargada en la memoria por U-Boot
- zynq_fsbl_0.elf: imagen FSBL utilizada para crear la imagen BOOT.BIN
- stub.tcl: archivo de scripts.para el arranque de Linux usando el modo JTAG.

3.4.1 Arranque de Linux usando el modo JTAG

El siguiente diagrama de flujo describe el proceso utilizado para iniciar Linux en la plataforma de destino en forma JTAG como es mostrado en la figura 3.10. Las implementaciones del booteo del sistema Linux se hacen en la herramienta SDK.

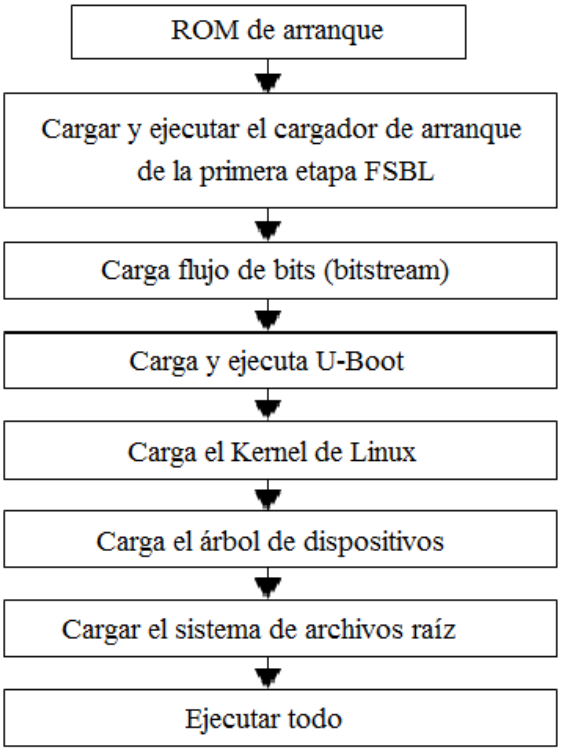


Figura 3.10 Proceso de arranque de Linux en la plataforma de destino.

Código del proceso arranque del sistema operativo Linux de modo JTAG

```
Xilinx Microprocessor Debugger (XMD) Engine
Xilinx EDK 14.2 Build EDK_P.28xd
Copyright (c) 1995-2012 Xilinx, Inc.  All rights reserved.

XMD%
XMD%
Accepted a new TCLSock connection from 127.0.0.1 on port 50308
connect arm hw
ERROR: Failed to Open JTAG Cable
       Cable target is not connected to the host

XMD% connect arm hw
ERROR: Failed to Open JTAG Cable
       Cable target is not connected to the host

XMD% connect arm hw

JTAG chain configuration
--------------------------------------------------
Device    ID Code         IR Length    Part Name
 1        4ba00477            4        Cortex-A9
 2        23727093            6        XC7Z020

CortexA9 Processor Configuration
-------------------------------------
```

```
Version..............................0x00000003
User ID..............................0x00000000
No of PC Breakpoints.................6
No of Addr/Data Watchpoints.........1

Connected to "arm" target. id = 64
Starting GDB server for "arm" target (id = 64) at TCP port no 1234
XMD% source
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/ps7_init.tc
l
XMD% ps7_init
XMD% source
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/s
tub.tcl

CortexA9 Processor Configuration
-------------------------------------
Version..............................0x00000003
User ID..............................0x00000000
No of PC Breakpoints.................6
No of Addr/Data Watchpoints.........1

Connected to "arm" target. id = 65
Starting GDB server for "arm" target (id = 65) at TCP port no 1235

RUNNING> XMD% target 64
-----------------------------------------------
System(0 ) - Hardware System on FPGA(Device 1) Targets:
-----------------------------------------------
       Target(64) - Cortex-A9(1) Hardware Debug Target*
       Target(65) - Cortex-A9(2) Hardware Debug Target
       Target(352) - Cortex-A9 CoreSight Debug Target


XMD% dow
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/u
-boot.elf
warning: FPGA is NOT Configured. Accessing FPGA Address space can
cause the System to hang. Use "fpga" command to Configure the FPGA
Processor started. Type "stop" to stop processor
Downloading Program --
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/u
-boot.elf
       section, .text: 0x04000000-0x040205ef
       section, .rodata: 0x040205f0-0x04026d99
       section, .hash: 0x04026d9c-0x04026ddb
       section, .data: 0x04026ddc-0x04027b63
       section, .got.plt: 0x04027b64-0x04027b6f
       section, .u_boot_cmd: 0x04027b70-0x0402801f
       section, .rel.dyn: 0x04028020-0x0402c8a7
       section, .dynsym: 0x0402c8a8-0x0402c957
       section, .bss: 0x04028020-0x0406acff
Setting PC with Program Start Address 0x04000000
XMD% con
```

```
RUNNING> XMD% stop
Processor stopped

XMD%
XMD% dow -data
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/z
Image 0x8000
warning: FPGA is NOT Configured. Accessing FPGA Address space can
cause the System to hang. Use "fpga" command to Configure the FPGA
Processor started. Type "stop" to stop processor
User Interrupt, Processor Stopped at 0x0ffa35c8
Downloading Data File --
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/z
Image at 0x00008000
Progress
..........................................................................
..........................................................................
.....................................................................Don
e
XMD% dow -data
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/r
amdisk8M.image.gz 0x800000
warning: FPGA is NOT Configured. Accessing FPGA Address space can
cause the System to hang. Use "fpga" command to Configure the FPGA
Downloading Data File --
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/r
amdisk8M.image.gz at 0x00800000
Progress
..........................................................................
..........................................................................
..........................................................................
.............Done
XMD% dow -data
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/d
evicetree.dtb 0x1000000
warning: FPGA is NOT Configured. Accessing FPGA Address space can
cause the System to hang. Use "fpga" command to Configure the FPGA
Downloading Data File --
C:/xup/embedded/ISE14_2/CTT/CTT.sdk/SDK/SDK_Export/system_hw_platform/directory/d
evicetree.dtb at 0x01000000
Progress .....Done
XMD% con

RUNNING> XMD%
```

3.5 Herramientas de desarrollo para el kernel del Sistema embebido.

Xilinx crea un árbol Git, bajo los parámetros de código abierto de forma pública para todos sus proyectos y desarrollos.

Git es un concepto desarrollado por Linus Torvalds, pensando en el control de la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones. Donde se lleva un registro

de los cambios realizados en archivos. Esto sirve para coordinar el enfoque sobre el trabajo que varias personas realizan sobre archivos compartidos.
Git es un software libre distribuible bajo los términos de la versión 2 de la Licencia Pública General de GNU. En pocas palabras se trata de una coordinación sobre objetivos y metas que se deben tomar en cuenta para el desarrollo del software.
La plataforma en la que se trabaja es linux con la distribución Fedora 22, en donde hace uso del comando yum, que es un gestor de paquetes de instalación, desarrollado por la Universidad de Duke University para mejorar y facilitar la instalación de los RPMs del sistema.
A continuación, se tiene descrita la instrucción y su salida en el sistema:

```
yum install git git-email

[root@localhost ~]# yum install git git-email
Yum command has been deprecated, redirecting to '/usr/bin/dnf install git git-
email'.
See 'man dnf' and 'man yum2dnf' for more information.
To transfer transaction metadata from yum to DNF, run:
'dnf install python-dnf-plugins-extras-migrate && dnf-2 migrate'
Última comprobación de caducidad de metadatos hecha hace 0:00:00, el Wed May  6
00:00:15                                                                  2020.
El paquete git-2.4.11-1.fc22.x86_64 ya se encuentra instalado, omitiendo.
Dependencias resueltas.
================================================================
Package          Arquitectura                  Versión                        R
epositorio                      Tamaño
================================================================================
Instalando:
 git-email           noarch                                              2.4.11-
1.fc22                          updates                           67 k
 perl-Authen-SASL                 noarch                                   2.16-
4.fc22                            fedora                            62 k
 perl-GSSAPI        x86_64                                              0.28-
13.fc22                           fedora                            64 k
 perl-Net-SMTP-SSL noarch                                              1.01-
18.fc22                           fedora                            14 k
Resumen de la transacción
================================================================================
Instalar  4 Packages
Tamaño total de la descarga: 206 k
Tamaño instalado: 348 k
¿Está de acuerdo [s/N]?:
```

Otra de las herramientas instaladas en el sistema es Petalinux para realizar la instalación debe hacerse como súper usuario como es mostrado en las siguientes líneas:

```
[root@localhost pkg]# ./petalinux-v2014.2-final-installer.run
-bash: ./petalinux-v2014.2-final-installer.run: Permiso denegado
[root@localhost pkg]# ls -lisa
total 1181996
1453519       4 drwxr-xr-x. 3 root    root          4096 may 13 01:12 .
1048577       4 drwxr-xr-x. 3 root    root          4096 may 12 23:10 ..
1453530 1181984 -rw-r--r--. 1 usuario usuario 1210347134 may 11 00:39 petalinux-
v2014.2-final-installer.run
1453456       4 drwxr-xr-x. 2 usuario usuario       4096 may 13 01:12 tools
```

```
[root@localhost pkg]# chmod 700 petalinux-v2014.2-final-installer.run
[root@localhost pkg]# ./petalinux-v2014.2-final-installer.run
INFO: Checking installer checksum...
INFO: Extracting PetaLinux installer...
INFO: Installing PetaLinux...
./petalinux-v2014.2-final-installer.run: /tmp/tmp.3CsOPQhg84/petalinux-v2014.2-
final/tools/common/petalinux/utils/petalinux-install: /lib/ld-linux.so.2: bad ELF
interpreter: No existe el fichero o el directorio

Please refer to the PetaLinux Tools Installation Guide.

Check the troubleshooting guide at the end of that manual, and if you are
unable to resolve the issue please contact customer support with file:
   /opt/pkg/petalinux_installation_log

[root@localhost pkg]#
```

Se instalan unas librerías necesarias para las herramientas a emplear.

```
[root@localhost pkg]# yum -y install glibc.i686
Yum command has been deprecated, redirecting to '/usr/bin/dnf -y install
glibc.i686'.
See 'man dnf' and 'man yum2dnf' for more information.
To transfer transaction metadata from yum to DNF, run:
'dnf install python-dnf-plugins-extras-migrate && dnf-2 migrate'

Última comprobación de caducidad de metadatos hecha hace 2:39:07, el Tue May 12
22:45:11 2020.
Dependencias resueltas.
================================================================================
====================
 Package                        Arquitectura    Versión
Repositorio        Tamaño
================================================================================
====================
Instalando:
 glibc                          i686            2.21-13.fc22
updates            4.2 M
 nss-softokn-freebl             i686            3.23.0-1.0.fc22
updates            200 k

Resumen de la transacción
================================================================================
====================
Instalar  2 Packages

Tamaño total de la descarga: 4.4 M
Tamaño instalado: 15 M
Descargando paquetes:
(1/2): nss-softokn-freebl-3.23.0-1.0.fc22.i686.rpm                 322 kB/s |
200 kB     00:00
(2/2): glibc-2.21-13.fc22.i686.rpm                                 1.0 MB/s |
4.2 MB     00:04
```

```
--------------------------------------------------------------------------------
------------------
Total                                                           906 kB/s |
4.4 MB     00:04
Ejecutando verificación de operación
Verificación de operación exitosa.
Ejecutando prueba de operaciones
Prueba de operación exitosa.
Ejecutando operación
  Instalando              : nss-softokn-freebl-3.23.0-1.0.fc22.i686
1/2
  Instalando              : glibc-2.21-13.fc22.i686
2/2
  Verificando             : glibc-2.21-13.fc22.i686
1/2
  Verificando             : nss-softokn-freebl-3.23.0-1.0.fc22.i686
2/2

Instalado:
  glibc.i686 2.21-13.fc22                 nss-softokn-freebl.i686 3.23.0-
1.0.fc22

¡Listo!
[root@localhost pkg]#

[root@localhost pkg]# ./petalinux-v2014.2-final-installer.run /opt/PetaLinux14_2
INFO: Checking installer checksum...
INFO: Extracting PetaLinux installer...
INFO: Installing PetaLinux...
INFO: Checking PetaLinux installer integrity...
INFO: Extracting Installation files...

LICENSE AGREEMENTS

PetaLinux SDK contains software from a number of sources.  Please review
the following licenses and indicate your acceptance of each to continue.

You do not have to accept the licenses, however if you do not then you may
not use PetaLinux SDK.

Use PgUp/PgDn to navigate the license viewer, and press 'q' to close

Press Enter to display the license agreements
Do you accept this license? [y/N] > y
Do you accept this license? [y/N] > y
INFO: Checking installation environment requirements...
INFO: Checking free disk space
INFO: Checking installed tools
INFO: Checking installed development libraries
INFO: Checking network and other services
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
```

```
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
warning, got bogus l2cap line.
WARNING: No tftp server found - please refer to "PetaLinux SDK Installation
Guide" for its impact and solution
INFO: Installing PetaLinux SDK to "/opt/PetaLinux14_2/petalinux-v2014.2-final"
INFO: PetaLinux SDK has been installed to /opt/PetaLinux14_2/petalinux-v2014.2-
final
[root@localhost pkg]#

[root@localhost pkg]# ./petalinux-v2014.2-final-installer.run /opt
INFO: Checking installer checksum...
INFO: Extracting PetaLinux installer...
INFO: Installing PetaLinux...
INFO: Checking PetaLinux installer integrity...
INFO: Extracting Installation files...

LICENSE AGREEMENTS

PetaLinux SDK contains software from a number of sources.  Please review
the following licenses and indicate your acceptance of each to continue.

You do not have to accept the licenses, however if you do not then you may
not use PetaLinux SDK.

Use PgUp/PgDn to navigate the license viewer, and press 'q' to close

Press Enter to display the license agreements
```

En el sistema se deben de cargar las dos variables de entorno que a continuación son denotadas:

```
[usuario@localhost ~]$ export CROSS_COMPILE=arm-xilinx-linux-gnueabi-
[usuario@localhost ~]$ export PATH=/path/to/cross/compiler/bin:$PATH
```

Es creado el directorio de trabajo del proyecto ZYNQ:

```
[usuario@localhost ~]$ mkdir ZYNQ_TRD_HOME
[usuario@localhost ~]$ cd  ZYNQ_TRD_HOME
```

Se descarga el código fuente de U-Boot, un cargador de arranque para placas integradas basado en ARM, también se encuentran herramientas que permiten modificar las características del U-boot.

```
[usuario@localhost ZYNQ_TRD_HOME]$ git clone git://github.com/Xilinx/u-boot-
xlnx.git
Cloning into 'u-boot-xlnx'...
remote: Enumerating objects: 378, done.
remote: Counting objects: 100% (378/378), done.
remote: Compressing objects: 100% (251/251), done.
remote: Total 691159 (delta 202), reused 221 (delta 127), pack-reused 690781
Receiving objects: 100% (691159/691159), 160.89 MiB | 659.00 KiB/s, done.
Resolving deltas: 100% (565469/565469), done.
Checking connectivity... done.
```

Dentro del directorio ZYNQ_TRD_HOME

```
[usuario@localhost ZYNQ_TRD_HOME]$ cd u-boot-xlnx
[usuario@localhost u-boot-xlnx]$ ls
api   board  common     configs  doc      dts  examples  include  Kconfig
Licenses     Makefile  post    scripts  tools
arch  cmd    config.mk  disk     drivers  env  fs        Kbuild   lib
MAINTAINERS  net       README  test
[usuario@localhost u-boot-xlnx]$ git checkout -b zynq_base_trd_14_2 xilinx-14.2-
build1-trd
Switched to a new branch 'zynq_base_trd_14_2'

[usuario@localhost u-boot-xlnx]$ du -ah |egrep zynq_base_trd_14_2
4.0K    ./.git/refs/heads/zynq_base_trd_14_2
4.0K    ./.git/logs/refs/heads/zynq_base_trd_14_2
```

Construcción u-boot personalizado de la placa ZC702

```
[usuario@localhost u-boot-xlnx]$ make zynq_zc702_config
awk '(NF && $1 !~ /^#/) { print $1 ": " $1 "_config; $(MAKE)" }' boards.cfg >
.boards.depend
Configuring for zynq_zc702 board...


usuario@localhost u-boot-xlnx]$ make
make: arm-xilinx-linux-gnueabi-gcc: No se encontró el programa
/bin/sh: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
dirname: falta un operando
Pruebe 'dirname --help' para más información.
Generating include/autoconf.mk
/bin/sh: línea 3: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
/bin/sh: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
dirname: falta un operando
Pruebe 'dirname --help' para más información.
Generating include/autoconf.mk.dep
/bin/sh: línea 3: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
make: arm-xilinx-linux-gnueabi-gcc: No se encontró el programa
/bin/sh: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
dirname: falta un operando
Pruebe 'dirname --help' para más información.
/bin/sh: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
/bin/sh: arm-xilinx-linux-gnueabi-ld: no se encontró la orden
```

```
/bin/sh: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
dirname: falta un operando
Pruebe 'dirname --help' para más información.
/bin/sh: arm-xilinx-linux-gnueabi-gcc: no se encontró la orden
dirname: falta un operando
Pruebe 'dirname --help' para más información.
arm-xilinx-linux-gnueabi-gcc -DDO_DEPS_ONLY \
        -g  -Os   -fno-common -ffixed-r8 -msoft-float   -fno-strict-aliasing  -
fno-common -ffixed-r8 -msoft-float  -D__KERNEL__ -I/home/usuario/ZYNQ_TRD_HOME/u-
boot-xlnx/include -fno-builtin -ffreestanding -nostdinc -isystem  -pipe  -
DCONFIG_ARM -D__ARM__          -march=armv5 -march=armv7-a -
I/home/usuario/ZYNQ_TRD_HOME/u-boot-xlnx/board/xilinx/common -
I/home/usuario/ZYNQ_TRD_HOME/u-boot-xlnx/board/xilinx/zynq_common -Wall -Wstrict-
prototypes -mno-unaligned-access    \
        -o lib/asm-offsets.s lib/asm-offsets.c -c -S
make: arm-xilinx-linux-gnueabi-gcc: No se encontró el programa
Makefile:491: recipe for target 'lib/asm-offsets.s' failed
make: *** [lib/asm-offsets.s] Error 127


Correccion del error
============================================================
dnf install qemu-user gcc-c++-arm-linux-gnu
```

Descripción del sistema operativo donde se trabaja la parte de Linux es una distribución de fedora core 22 de 64 bits.

```
[root@localhost ~]# uname -a
Linux localhost.localdomain 4.4.14-200.fc22.x86_64 #1 SMP Fri Jun 24 21:19:33 UTC
2016 x86_64 x86_64 x86_64 GNU/Linux
[root@localhost ~]# dnf install qemu-user gcc-c++-arm-linux-gnu
Última comprobación de caducidad de metadatos hecha hace 1:18:02, el Fri May  8
22:35:02 2020.
Dependencias resueltas.
================================================================================
 Package                    Arquitectura
                                       Versión                Repositorio Tamaño
================================================================================
Instalando:
 binutils-arm-linux-gnu     x86_64     2.26-8.fc22            updates     1.9 M
 cross-binutils-common      noarch     2.26-8.fc22            updates     1.7 M
 cross-gcc-common           noarch     5.3.1-2.fc22           updates     1.9 M
 gcc-arm-linux-gnu          x86_64     5.3.1-2.fc22           updates      12 M
 gcc-c++-arm-linux-gnu      x86_64     5.3.1-2.fc22           updates     6.2 M
 qemu-user                  x86_64     2:2.3.1-16.fc22        updates     9.1 M

Resumen de la transacción
================================================================================
Instalar  6 Packages

Tamaño total de la descarga: 32 M
Tamaño instalado: 146 M
```

```
¿Está de acuerdo [s/N]?: s
Descargando paquetes:
(1/6): gcc-c++-arm-linux-gnu-5.3.1-2.fc22.x86_64.rpm
630 kB/s | 6.2 MB     00:10
(2/6): binutils-arm-linux-gnu-2.26-8.fc22.x86_64.rpm
660 kB/s | 1.9 MB     00:02
(3/6): cross-gcc-common-5.3.1-2.fc22.noarch.rpm
651 kB/s | 1.9 MB     00:03
(4/6): qemu-user-2.3.1-16.fc22.x86_64.rpm
491 kB/s | 9.1 MB     00:18
(5/6): cross-binutils-common-2.26-8.fc22.noarch.rpm
452 kB/s | 1.7 MB     00:03
(6/6): gcc-arm-linux-gnu-5.3.1-2.fc22.x86_64.rpm
470 kB/s |  12 MB     00:25
--------------------------------------------------------------------------------
--------------------------------------------
Total
1.3 MB/s |  32 MB     00:25
Ejecutando verificación de operación
Verificación de operación exitosa.
Ejecutando prueba de operaciones
Prueba de operación exitosa.
Ejecutando operación
  Instalando              : cross-binutils-common-2.26-8.fc22.noarch
1/6
  Instalando              : binutils-arm-linux-gnu-2.26-8.fc22.x86_64
2/6
  Instalando              : cross-gcc-common-5.3.1-2.fc22.noarch
3/6
  Instalando              : gcc-arm-linux-gnu-5.3.1-2.fc22.x86_64
4/6
  Instalando              : gcc-c++-arm-linux-gnu-5.3.1-2.fc22.x86_64
5/6
  Instalando              : qemu-user-2:2.3.1-16.fc22.x86_64
6/6
  Verificando             : qemu-user-2:2.3.1-16.fc22.x86_64
1/6
  Verificando             : gcc-c++-arm-linux-gnu-5.3.1-2.fc22.x86_64
2/6
  Verificando             : gcc-arm-linux-gnu-5.3.1-2.fc22.x86_64
3/6
  Verificando             : binutils-arm-linux-gnu-2.26-8.fc22.x86_64
4/6
  Verificando             : cross-gcc-common-5.3.1-2.fc22.noarch
5/6
  Verificando             : cross-binutils-common-2.26-8.fc22.noarch
6/6

Instalado:
  binutils-arm-linux-gnu.x86_64 2.26-8.fc22  cross-binutils-common.noarch 2.26-
8.fc22   cross-gcc-common.noarch 5.3.1-2.fc22
  gcc-arm-linux-gnu.x86_64 5.3.1-2.fc22      gcc-c++-arm-linux-gnu.x86_64 5.3.1-
2.fc22  qemu-user.x86_64 2:2.3.1-16.fc22
```

```
¡Listo!
[root@localhost ~]#

[usuario@localhost u-boot-xlnx]$ mkdir -p ~/ldbox/fedora-armhfp

cd ~/ldbox/targets/fedora-armhfp
lb-init -L --sysroot=/ -C --sysroot=/ -c /usr/bin/qemu-arm -m obs-rpm-build -n -N
-t / -A armv7hl -M armv7hl -P /usr/arm-linux-gnu/sys-root/ -P /usr/bin/arm-linux-
gnu- -P /usr/libexec/gcc/arm-linux-gnueabi/ fedora-armhfp /usr/bin/arm-linux-gnu-
gcc
```

Se concluye con la instalación de las librerías para los procesadores arm que es usado por el compilador gcc. Para la placa ZC702.

Capítulo 4

Construcción de un sistema de MultiProcesamiento Asimétrico AMP, en un kit de desarrollo ZC702.

En esta parte del trabajo el Zynq-7000 System on Chip totalmente programable contiene dos procesadores Cortex-A9, hacen uso de la memoria y periféricos comunes, que pueden ser compartidos a la vez desde un procesador MicroBlaze. Es implementado en la parte lógica programable (PL).
El concepto de MultiProcesamiento Asimétrico es un mecanismo que permite que varios procesadores ejecuten sus propios sistemas o aplicaciones completas, con la posibilidad de acoplar libremente esas aplicaciones a través de recursos compartidos.
En este caso al usar el kit de desarrollo ZC702, los procesadores involucrados son Cortex-A9 y los Procesadores MicroBlaze.
En un acotamiento de recursos se hace uso de un procesador Cortex-A9 CPU0 y un procesador Microblaze MB0.
Cada CPU ejecuta una aplicación "bare-metal" dentro de su propio entorno independiente. Se debe de tener cuidado con los conflictos con los recursos de hardware compartidos de los dos procesadores en juego (CPU0 y MB0).
El término "bare-metal" se determina como metal desnudo o expuesto. A lo que se refiere que cada dispositivo procesamiento duro o blando ejecuta una aplicación en concreto e independiente entre ellos.

4.1 Características de bajo nivel del sistema AMP

En la parte del diseño donde se usa la herramienta SDK, se implementa una depuración múltiple de procesadores simultáneos.
Acceso del procesador MicroBlaze a la memoria DDR3 del Zynq SoC a través de los puertos de alto rendimiento (HP0 y HP1). Pero antes pasan por una interconexion_AXI_mb.
El acceso del procesador MicroBlaze a los periféricos del sistema de procesamiento (PS) Zynq SoC y la memoria en el chip (OCM), se hace a través del puerto esclavo de propósito general S_GP0. Se logra con una interconexion_AXI independiente llamada axi4lite_mb_0.
Algo importante se hace uso de entradas y salida virtuales (VIO) y la medición de la latencia de interrupción mediante el analizador lógico integrado (ILA) de la herramienta analizador ChipScope ™ para controlar la lógica.

4.2 Implementación de dispositivos del sistema AMP

processing_system7_0

En esta parte se tiene una interconexión fuerte de todos los periféricos y los IP de los que se hacen uso para implementar este sistema.

GPIO_O[0:63] conecta al dispositivo util_bus_split_0 puerto de entrada Sig[63:0]
processing_system7_0_DDR
MIO[0:53] Processing_system7_0_MIO[0:53]
S_AXI_HP0 axi_interconnect_mb microblaze_0
S_AXI_HP1axi_interconnect_mb microblaze_0
S_AXI_GP0 axi4lite_mb_0 microblaze_0

clock_generator_0 S_AXI_GP0_ACLK
S_AXI_HP0_ACLK
S_AXI_HP1_ACLK
processing_system7_0_PS_SRSTB
processing_system7_0_PS_CLK
processing_system7_0_PS_PORB

clock_generator_0

Este dispositivo generador de señales de reloj sus entradas están dadas por las señales de Reset, CLK_N y CLK_P. Las señales de entrada están fijadas a una frecuencia de 200 MHz. Las señales de salidas son de una frecuencia de 100 Mhz. La cual es la señal principal de reloj en todo el diseño.

util_bus_split_0

La operación Bus Split está diseñada para usarse con Xilinx Platform Studio para dividir un bus en buses más pequeños. También se incluye un archivo de definición de periférico de microprocesador (MPD) asociado con este módulo.

Se tiene una entrada de 64 bits que viene del GPIO del processing_system7_0. Su salida está dirigida al proc_sys_reset_0.

DDR3

La memoria DDR está conectada directamente al processing_system7_0

debug_module

El MicroBlaze Debug Module (MDM) está vinculado directamente con microblaze_0, el modulo proc_sys_reset_0 y axi4lite_mb_0.

Las características que destacan del MDM en este sistema AMP, son las siguientes:

El soporte para herramientas de depuración de software basadas en JTAG
Soporte para depurar hasta 32 procesadores MicroBlaze.
El control sincronización de múltiples procesadores MicroBlaze.
En un UART basado en JTAG con una interfaz configurable AXI4-Lite

proc_sys_reset_0

Módulo de reinicio del sistema del procesador

El diseño del módulo de reinicio del sistema del procesador Xilinx permite al cliente adaptar el diseño para que se adapte a su aplicación mediante el establecimiento de ciertos parámetros para habilitar y deshabilitar funciones.

En este módulo se tienen comunicación de entrada de la señal de reloj también una señal de bloqueo. Del módulo de depuración del MB también se recibe una señal de reinicio, se agrega una señal de reinicio auxiliar del módulo util_bus_split_0.
Para que finalmente se tenga un reinicio en :

- axi_interconnect_mb
- axi4lite_mb_0
- microblaze_0_intc
- microblaze_0

irq_gen_0
Es un registro de control que tiene una señal de reloj del módulo clock_generator_0, lo importante de este dispositivo genera interrupciones a los siguientes dispositivos:

- chipscope_vio_0
- chipscope_ila_0
- microblaze_0_intc

Que de alguna manera tienen que ver con las herramientas de monitoreo del chipscope.

Chipscope_icon_0
Controlador integrado ChipScope (ICON) de IP en la herramienta ChipScope existe una interfaz a nivel hardware, controla los puertos virtuales de entrada y salida(vio). También tiene el control del analizador lógico integrado(ila).
axi_interconnect_mb
La interconexión AXI conecta uno o más dispositivos maestros asignados en memoria AXI a uno o más dispositivos esclavos asignados en memoria. Está diseñado para transferencias asignadas en memoria únicamente.
Que sirve como intermediario entre microblaze_0 y el processing_system7_0 para gestionar la memoria DDR. También existe comunicación de coordinación con el dispositivo axi4lite_mb_0.
axi4lite_mb_0
Es un subconjunto de interfaces de registro de control AXI4-Lite que opera entre microblaze_0 y modulo processing_system7_0. Pero esta interfaz tiene conectado los siguientes modulos:

- irq_gen_0
- debug_module
- microblaze_0_intc
- LEDs_4Bits
- Proc_sys_reset_0
- Axi_interconnect_mb

microblaze_0_intc
Controlador de interrupciones AXI. El núcleo del controlador de interrupciones LogiCORE ™ IP AXI (AXI INTC) concentra múltiples entradas de interrupción de dispositivos periféricos en una única salida de interrupción al procesador del sistema. En este caso el microprocesador del sistema es microblaze_0.

Todas las implementaciones de los dispositivos del sistema de AMP son mostrados de forma gráfica en la figura 4.1

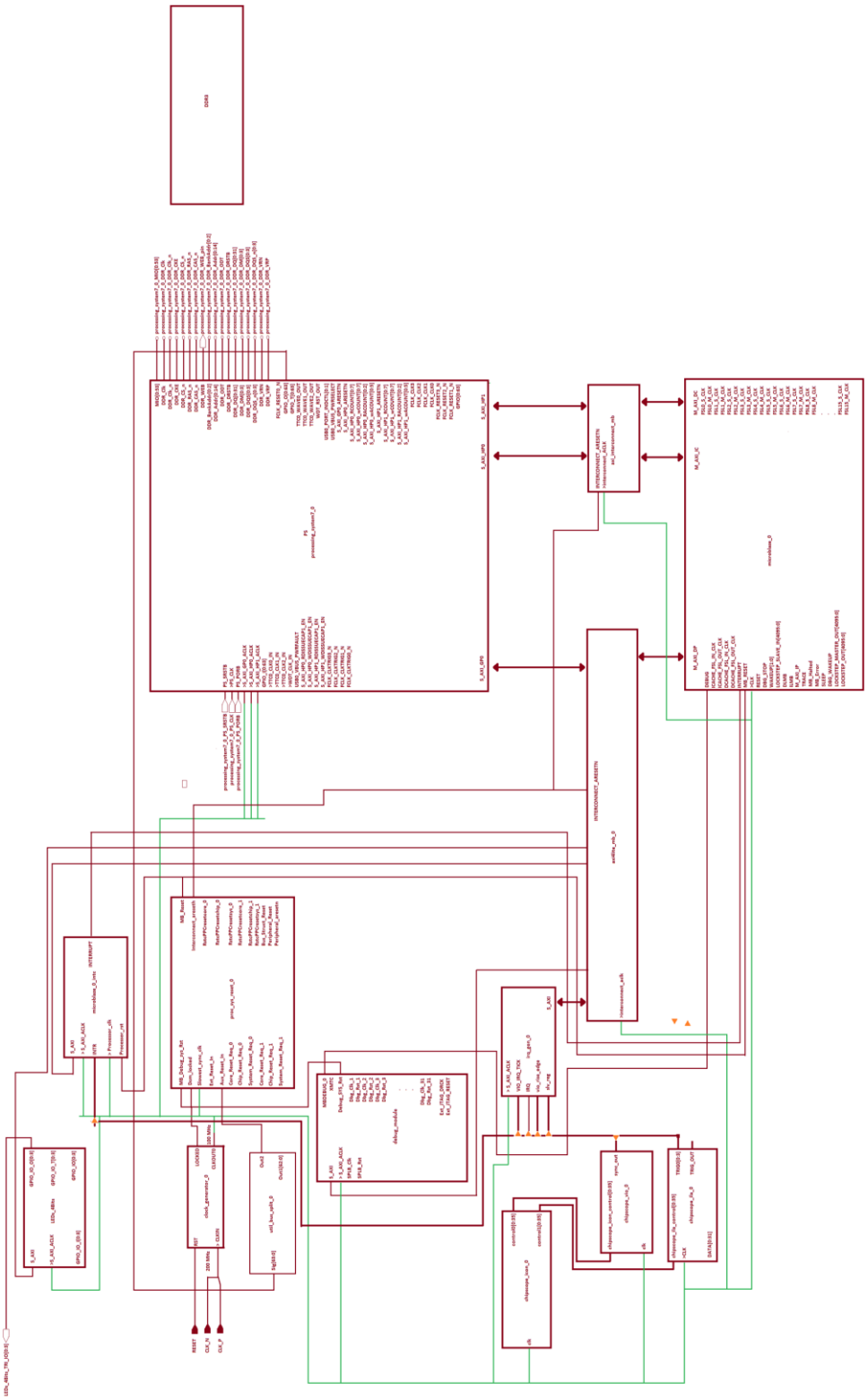


Figura 4.1 Diagrama de bloques del sistema de MultiProcesamiento Asimétrico AMP.

4.3 Mapeo de direcciones.

Es importante tener mapeo de direccionamiento en función del sistema de procesamiento7.0 y del MicroBlaze. Como es mostrada en la tabla 4.1.

Instancia	Nombre de la Base	Rango de Direcciones	Tamaño	Interfaces de Bus	Nombre del Bus
processing_system7_0's Address Map					
processing_system7_0	C_DDR_RAM_BASEADDR	0x00000000 0x3FFFFFFF	1G		
processing_system7_0	C_S_AXI_GP0_BASEADDR	0xE0000000 0xFFFFFFFF	512M	S_AXI_GP0	axi4lite_mb_0
processing_system7_0	C_UART1_BASEADDR	0xE0001000 0xE0001FFF	4K		
processing_system7_0	C_USB0_BASEADDR	0xE0002000 0xE0002FFF	4K		
processing_system7_0	C_I2C0_BASEADDR	0xE0004000 0xE0004FFF	4K		
processing_system7_0	C_CAN0_BASEADDR	0xE0008000 0xE0008FFF	4K		
processing_system7_0	C_GPIO_BASEADDR	0xE000A000 0xE000AFFF	4K		
processing_system7_0	C_ENET0_BASEADDR	0xE000B000 0xE000BFFF	4K		
processing_system7_0	C_SDIO0_BASEADDR	0xE0100000 0xE0100FFF	4K		
processing_system7_0	C_TTC0_BASEADDR	0xF8001000 0xF8001FFF	4K		
microblaze_0's Address Map					
processing_system7_0	C_S_AXI_HP0_BASEADDR	0x30000000 0x3FFFFFFF	256M	S_AXI_HP0	axi_interconnect_mb
processing_system7_0	C_S_AXI_HP1_BASEADDR	0x30000000 0x3FFFFFFF	256M	S_AXI_HP1	axi_interconnect_mb
LEDs_4Bits	C_BASEADDR	0x40000000 0x4000FFFF	64K	s_AXI	axi4lite_mb_0
microblaze_0_intc	C_BASEADDR	0x41000000 0x41000FFF	4K	S_AXI	axi4lite_mb_0
debug_module	C_BASEADDR	0x42000000 0x4200FFFF	64K	S_AXI	axi4lite_mb_0
irq_gen_0	C_BASEADDR	0x50000000 0x5000FFFF	64K	S_AXI	axi4lite_mb_0
processing_system7_0	C_S_AXI_GP0_BASEADDR	0xE0000000 0xFFFFFFFF	512M	S_AXI_GP0	axi4lite_mb_0

Tabla 4.1 Mapeo general del sistema de AMP

Pero como los dos sistemas de procesamiento comparten los recursos de memoria y periféricos, es muy importante visualizar los recursos compartidos mostrados en la tabla 4.2.

processing_system7_0's Address Map		Rango de Direcciones	microblaze_0's Address Map	
Instancia	Nombre de la Base		Nombre de la Base	Instancia
processing_system7_0	C_DDR_RAM_BASEADDR	0x00000000		
		0x30000000	C_S_AXI_HP0_BASEADDR C_S_AXI_HP1_BASEADDR	processing_system7_0
		0x3FFFFFFF	C_S_AXI_HP0_BASEADDR C_S_AXI_HP1_BASEADDR	
		0x3FFFFFFF		

Tabla 4.2 Mapeo compartido sistema de procesamiento 7 y el MicroBlaze.

El mapeo de algunos recursos que se hacen el Microblaze se detallan en la tabla 4.3, Se observa 4 instancias de IP que se toman en la parte PL:

- 0x40000000–0x4000FFFF GPIO para acceso a LED de 4 bits.
- 0x41000000–0x4100FFFF Controlador de interrupciones MicroBlaze.
- 0x42000000–0x4200FFFF Modulo de Depuración MicroBlaze.
- 0x50000000–0x5000FFFF Núcleo personalizado IRQ_GEN.
- 0xE0001000–0xE0001FFF PS Uart1.
- 0xFFFF0000–0xFFFF0000 PS OCM a través del puerto PS S_GP0.

Instancia	Nombre de la Base	Rango de Direcciones	Tamaño	Interfaces de Bus	Nombre del Bus
LEDs_4Bits	C_BASEADDR	0x40000000 0x4000FFFF	64K	s_AXI	axi4lite_mb_0
microblaze_0_intc	C_BASEADDR	0x41000000 0x41000FFF	4K	S_AXI	axi4lite_mb_0
debug_module	C_BASEADDR	0x42000000 0x4200FFFF	64K	S_AXI	axi4lite_mb_0
irq_gen_0	C_BASEADDR	0x50000000 0x5000FFFF	64K	S_AXI	axi4lite_mb_0
processing_system7_0	C_S_AXI_GP0_BASEADDR	0xE0000000 0xFFFFFFFF	512M	S_AXI_GP0	axi4lite_mb_0

Tabla 4.3 Mapeo las instancias de la parte PL.

También hay que destacar el control del registro del irq_gen_0 se determina en la tabla 4.4

Bit	Aceso	Descripción
[31:1]	R/W	No usado. El valor escrito se puede leer.
[0]	R/W	IRQ acertado: 0: IRQ no se afirma hacia el PS. 1: IRQ se afirma hacia el controlador de interrupción MB0. Si el pin VIO_IRQ_TICK está afirmado (por el VIO), este bit se establece. Además, el MB0 puede establecer este bit. Solo MB0 puede escribir este bit para borrarlo.

Tabla 4.4 Control del registro del irq_gen.

La siguiente tabla se hace mediante el direccionamiento del sistema de procesamiento 7 que también incrusta ciertos recursos de hardware mostrados en la tabla 4.5 que son accedidos

por los puertos de MIO que está limitado en los pines de conexión. El software programa el enrutamiento de las señales de E/S a los pines MIO. Las señales de los periféricos de E/S también se pueden direccionar a la parte de la lógica de programación (PL).

Instancia	Nombre de la Base	Rango de Direcciones	Nombre de la Base	Instancia
processing_system7_0's Address Map				
processing_system7_0	C_S_AXI_GP0_BASEADDR	0xE0000000		
		0xE0001000 0xE0001FFF	C_UART1_BASEADDR	processing_system7_0
		0xE0002000 0xE0002FFF	C_USB0_BASEADDR	
		0xE0004000 0xE0004FFF	C_I2C0_BASEADDR	
		0xE0008000 0xE0008FFF	C_CAN0_BASEADDR	
		0xE000A000 0xE000AFFF	C_GPIO_BASEADDR	
		0xE000B000 0xE000BFFF	C_ENET0_BASEADDR	
		0xE0100000 0xE0100FFF	C_SDIO0_BASEADDR	
		0xF8001000 0xF8001FFF	C_TTC0_BASEADDR	
		0xFFFFFFFF		

Tabla 4.5 Traslape de C_S_AXI_GP0_BASEADDR con UART1, USB0, I2C0, etc.

4.4 Módulo de depuración de MicroBlaze MDM ver 2.10.a

Este módulo de depuración MicroBlaze permite la depuración basada en JTAG de uno a 32 procesadores MicroBlaze como máximo. Con tal capacidad se pone de manifiesto el soporte del control de múltiples procesadores blandos. Admite la conexión al núcleo ChipScope Pro ICON. Se muestra el diagrama a bloques del Módulo de Depuración del MicroBlaze, en la figura 4.2.

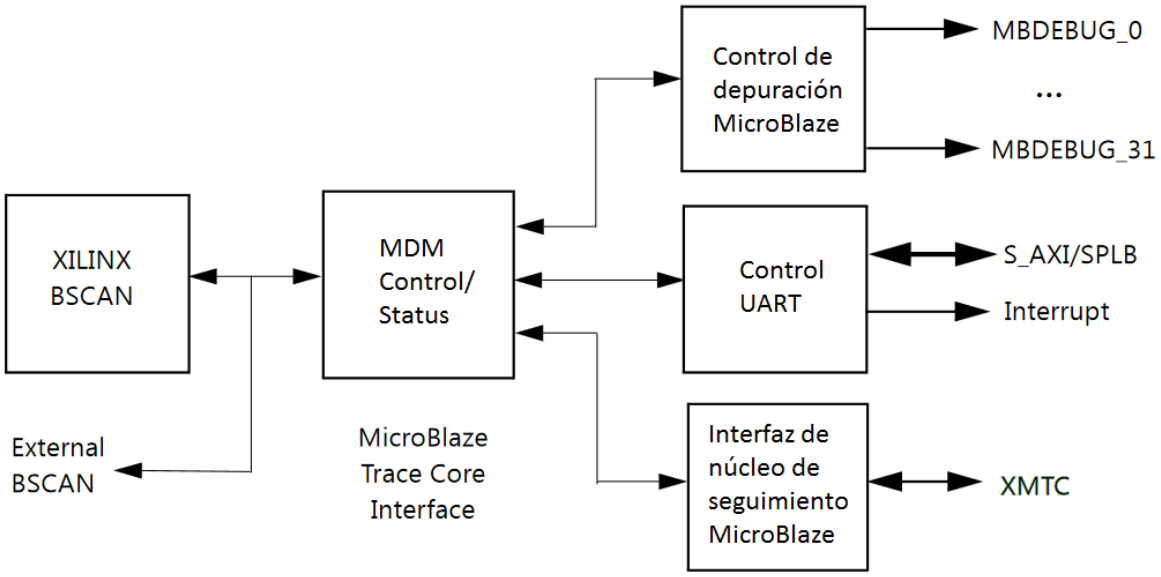


Figura 4.2 Diagrama de bloques del módulo de depuración del MicroBlaze**.**

Las señales de interfaz del núcleo de seguimiento del MicroBlaze XMTC, manejadas son:

- Ext_JTAG_DRCK
- Ext_JTAG_RESET
- Ext_JTAG_SELX
- Ext_JTAG_CAPTURE
- Ext_JTAG_SHIFT
- Ext_JTAG_UPDATE
- Ext_JTAG_TDI
- Ext_JTAG_TDO

4.5 Recursos compartidos entre CPU0 y MB0.

Básicamente el CPU0 tiene el control de todos los recursos, en el momento que el procesador MB0 requiere hacer uso de los recursos de hardware se envía la solicitud de usos de dichos recursos al procesador CPU0. La aplicación Bare-Metal que se ejecuta en el procesador MB0 limita el acceso a los recursos para evitar interferencias.

Los recursos compartidos de mayor importancia son:

El Distribuidor de Control de Interrupciones (ICD), la memoria DDR, la memoria en chip OCM On-Chip Memory, en el temporizador global no se tiene mayor problema porque es el proporciona la señal de reloj y por ultimo tenemos la Unidad de Control de monitoreo (SCU) en combinación de la memoria caché L2.

En la tabla 4.6 tenemos la asignación del direccionamiento de la memoria DDR para CPU0 y para MB0. En la parte del MB0 se usa para la aplicación Bare-Metal de este mismo procesador.

Instancia	Nombre de la Base	Rango de Direcciones	Tamaño	Interfaces de Bus	Nombre del Bus
processing_system7_0's Address Map					
processing_system7_0	C_DDR_RAM_BASEADDR	0x00000000 **0x00100000** **0x3FFFFFFF**	1G		
microblaze_0's Address Map					
processing_system7_0	C_S_AXI_HP0_BASEADDR	0x30000000 0x3FFFFFFF	256M	S_AXI_HP0	axi_interconnect_mb

Tabla 4.6 Direccionamiento de la memoria DDR para el CPU0 y el MB0.

En este diseño los accesos a OCM son manejados con cautela para cada CPU. El CPU0 inicializa la bandera 0 antes de inicializa el MB0, cuando la bandera es 0, el CPU0 posee el control de UART. Y cuando la bandera es diferente de cero MB0 pose el control del UART. Control de la bandera para cada CPU:

- CPU0 define la bandera
- MB0 Borra la bandera

El uso del núcleo VIO del analizador ChipScope permite generar una interrupción de forma manual al procesador MB0. Esto nos permite medir la latencia de las interrupciones. En un diseño real este módulo no existiría. Pero la interrupción se originaria en la parte lógica programable como un motor de Acceso Directo a Memoria DMA.

4.6 Lógica secuencial de la ejecución del Software.

El software está conformado en tres secciones:

- Cargador de arranque de primera etapa (FSBL)
  Al reiniciar PS se ejecuta FSBL en el CPU0 y se programa PL, y se copia los archivos ELF a la memoria DDR. Después se cargan aplicaciones en la memoria DDR y se ejecutan las aplicaciones.
- Aplicación bare-metal para CPU0
  Es responsable de inicializar los recursos compartidos y de iniciar MB0 eliminando el reinicio de PL.
- Aplicación bare-metal para MB0

En la figura 4.3 se representa la forma lógica de la ejecución del software en el sistema de AMP. Es una estructura sólida donde la capa inferior le da sustento a la siguiente capa de forma escalonada.

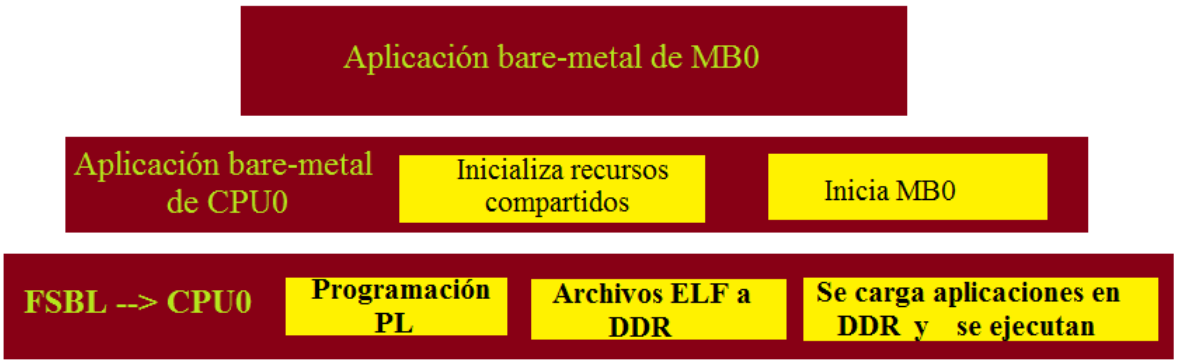


Figura 4.3 Secciones de ejecución del software.

4.6.1 Aplicación bare–metal del CPU0.

En la dirección de la memoria 0x00100000 se encuentra el script enlazador (linker) se usa para establecer la dirección de inicio.

1. Se re-mapea por completo la parte de la OCM que tiene un espacio de 256 KB su direccionamiento está comprendido entre la dirección 0xFFFC0000 a la dirección 0xFFFFFFFF, como se muestra en la figura 4.4. El espacio de 256 KB está formado por cuatro bloques de 64 KB.
   En el código fuente(main.c) en el SDK se determina como: **MY_REMAP();**
   El re-mapeo esta descrito en el archivo: remap_ocm.h

```
// Remap all 4 64KB OCM to top of memory starting at 0xFFFC0000
// Open address filtering to include DDR at 0x00000000
#define MY_REMAP()   __asm__ __volatile__(\
    "mov  r5, #0x03                                          \n"\
    "mov  r6, #0                                             \n"\
    "LDR  r7, =0xF8000000  /* SLCR base address    */        \n"\
    "LDR  r8, =0xF8F00000  /* MPCORE base address  */        \n"\
    "LDR  r9, =0x0000767B  /* SLCR lock key        */        \n"\
    "mov  r10,#0x1F                                          \n"\
    "LDR  r11,=0x0000DF0D  /* SLCR unlock key               \n"\
    "dsb                                                     \n"\
    "isb                   /* make sure it completes */      \n"\
    "pli  do_remap     /* preload the instruction cache */   \n"\
    "pli  do_remap+32                                        \n"\
    "pli  do_remap+64                                        \n"\
    "pli  do_remap+96                                        \n"\
    "pli  do_remap+128                                       \n"\
    "pli  do_remap+160                                       \n"\
    "pli  do_remap+192                                       \n"\
    "isb                    /* make sure it completes */     \n"\
    "b    do_remap                                           \n"\
    ".align 5, 0xFF         /* forces the next block to a cache line alignment
*/ \n"\
    "do_remap:              /* Unlock SLCR                        */ \n"\
    "str  r11, [r7, #0x8]   /* Configuring OCM remap value        */ \n"\
    "str  r10, [r7, #0x910] /* Lock SLCR                          */ \n"\
    "str  r9,  [r7, #0x4]   /* Disable SCU & address filtering    */ \n"\
    "str  r6,  [r8, #0x0]   /* Set filter start addr to 0x00000000 */ \n"\
    "str  r6,  [r8, #0x40]  /* Enable SCU & address filtering     */ \n"\
    "str  r5,  [r8, #0x0]                                       \n"\
    "dmb                                                        \n"\
         );
```

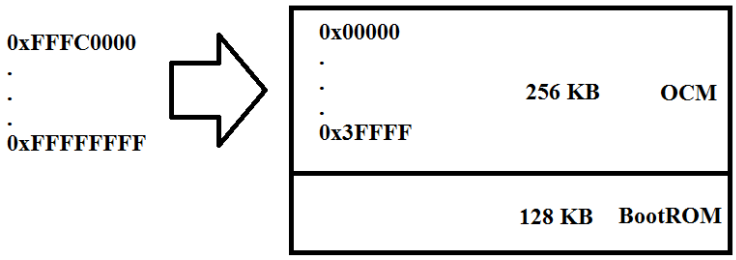


Figura 4.4 Re-mapeo de la Memoria on Chip (OCM).

Después se deshabilita el filtrado de memoria DDR para habilitar la memoria DDR en la parte inferior del mapa de direcciones de CPU0.

2. Se configura la Unidad controladora de Memoria MMU para deshabilitar la caché para accesos a la OCM en el rango de direcciones de 0xFFFC0000 a 0xFFFFFFFF.

```
// Disable L1 cache for OCM
Xil_SetTlbAttributes(0xFFFC0000,0x04de2); / S=b0 TEX=b100 AP=b11, Domain=b1111, C=b0, B=b0
```

En esta parte tomamos en cuenta la tabla 4.7 como parte descriptiva de esta línea de código del archivo fuente. Que es tomada de "Zynq-7000 SoC Technical Reference Manual UG585 (v1.12.2)"

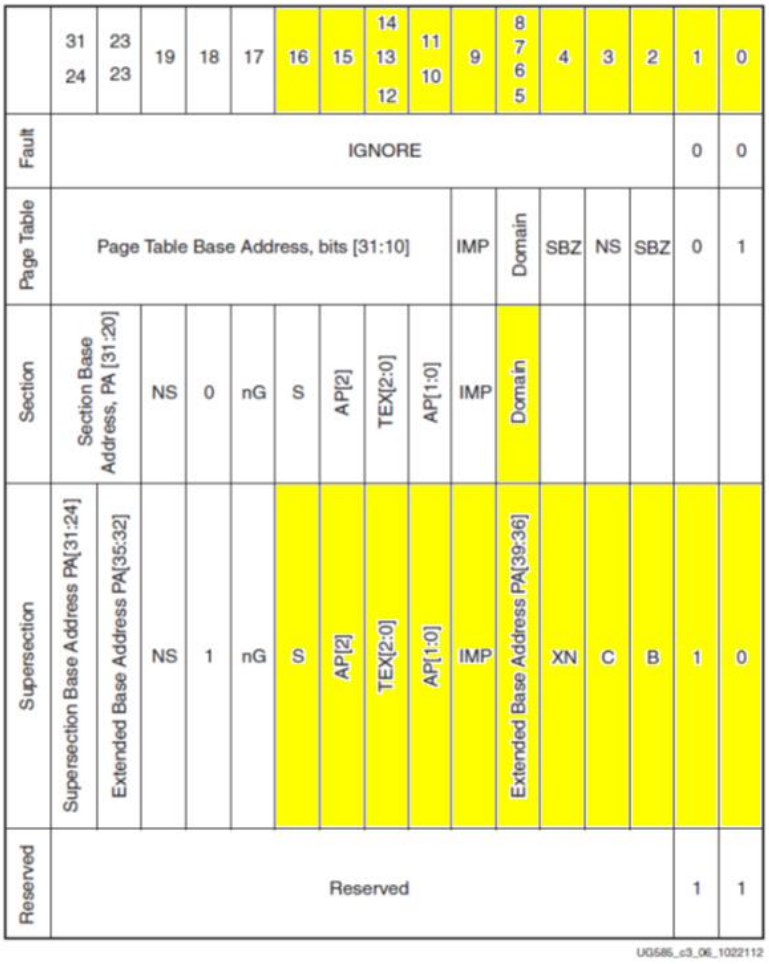


Tabla 4.7 L1 Formato de entrada de la tabla de páginas

En la tabla 4.8 se observa la descomposición de la parte de atributos de la instrucción Xil_SetTlbAttributes de cómo está conformada.

0				4				d				e				2			
0	0	0	0	0	1	0	0	1	1	0	1	1	1	1	0	0	0	1	0
20	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			S	AP[2]	TEXT[2:0]			AP[1:0]	IMP	DOMAIN					XN	C	B		

Tabla 4.8 Atributos de la función Xil_SetTlbAttributes.

En los campos de las variables “C” y “B” en la tabla 4.9 se describe sus diferentes codificaciones posibles existentes.

Bits de codificación		Atributos de la cache
C	B	
0	0	No Cache
0	1	Write-Back, Write-Allocate Reescribir, Escribir asignar
1	0	write-through, no write-allocate escritura simultánea, sin asignación de escritura
1	1	write-back, no write-allocate reescritura, sin asignación de escritura

Tabla 4.9 Bits de codificación C y B.

Siguiendo con los demás campos de configuración tenemos:

Bit XN(Execute Never) → bit de no ejecución.

Dominio (Domain) → Un dominio es una colección de regiones de memoria. Los dominios solo son válidos para L1.

IMP → El valor es determinado como"0".

AP Permisos de Acceso de Memoria (Memory Access Permissions) → La codificación de los permisos de acceso a la memoria están descritos en la tabla 4.10.

APX	AP1	AP0	Privilegiado	Sin Privilegios	Descripción
0	0	0	Sin acceso	Sin acceso	Permisos por defecto
0	0	1	Leer/Escribir	Sin acceso	Solo acceso privilegiado
0	1	0	Leer/Escribir	Leer	Sin escritura en modo de usuario
0	1	1	Leer/Escribir	Leer/Escribir	Acceso completo
1	0	0	~~~	~~~	Reservado
1	0	1	Leer	Sin acceso	Privilegiado Solo lectura
1	1	0	Leer	Leer	Solo lectura
1	1	1	~~~	~~~	Reservado

Tabla 4.10 Codificaciones de permisos de acceso (AP y APX)

En la tabla 4.11 el punto a resaltar es la parte del campo TEX[2:0] con las variables "C" y "B".

TEX[2:0]	C	B	Descripción	Tipode Memoria
0	0	0	Fuertemente ordenado	Fuertemente ordenado
0	0	1	Dispisitivo compartido	Dispositivo
0	1	0	Escritura externa e interna, sin asignación en escritura	Normal
0	1	1	Respuesta externa e interna, sin asignar en escritura (Outer and Inner write back, no allocate on write)	Normal
1	0	0	Externo e interno no cacheable (Cache)	Normal
1	1	1	Caché externo e interno	Normal
10	0	0	Dispositivo que no se puede compartir	Dispositivo
10	~	~	Reservado	~
11	~	~	Reservado	~
1XX	Y	X	Memoria caché XX - Política exterior YY - Política interna	Normal

Tabla 4.11 Codificaciones de atributos de memoria TEX[2:0]

S Bit de compartir. Este bit determina si la memoria es compartida. S = 0, la ubicación de la memoria es no compartible, y S = 1, es compartible.

La función, Xil_SetTlbAttributes, que permite modificar los atributos de memoria para una entrada de la tabla MMU.

El prototipo de la función se proporciona en el archivo fuente xil_mmu.h del ARM BSP que se encuentra en C:\Xilinx\14.5\ISE_DS\EDK\sw\lib\bsp\standalone_v3_09_a\src\cortexa9.

3. Se libera el reinicio la parte de la PL.
   La Memoria en Chip OCM es usada para la comunicación con CPU0.
   `#define COMM_VAL (*(volatile u32 *)(0xFFFF0000))`
   Para indicar al compilador que se trata de una constante, usaremos la directiva #define y su sintaxis queda declarada como: `#define <identificador> <valor>`
   `#define < COMM_VAL> <(*(volatile u32 *)(0xFFFF0000))>`
   volatile →La palabra clave volátil le dice al compilador que el valor de la variable puede cambiar en cualquier momento como resultado de condiciones externas, no solo como resultado del flujo del programa. El compilador no optimizará nada que tenga que ver con la variable volátil.
   `COMM_VAL = 0;`
   Asignación el valor cero a la variable COMM_VAL.
   El comentario del programa fuente determina los siguiente:

   "mb ciclo de arranque(bootloop)
   Si no usa FSBL para descargar Microblaze elf a DDR el vector de reinicio del Microblaze debe contener una 'rama a uno mismo' para proporcionar el Microblaze código válido para ejecutar antes de que el depurador pueda detenerse el Microblaze y descargue la aplicación que coloca un código válido en el vector de reinicio del Microblaze

   Si usa SDK/XMD para descargar la aplicación Microblaze descomenta las siguientes dos líneas de código
   //Xil_Out32(0x30000000, 0xb8000000);
   //Xil_DCacheFlush();"

   La liberación del reinicio se determina con las siguientes líneas de código:
   //Enable emio bit 0
   Xil_Out32(GPIO_DIRM, 0x1);
   //Set emio[0] to release PL reset
   Xil_Out32(GPIO_MASK_DATA, 0xfffe0001);

   #define GPIO_DIRM (0xE000A000+0x0284) → *Xil_Out32(0xE000A000+0x0284, 0x1);*
   #define GPIO_MASK_DATA (0xE000A000+0x0010) →
   Xil_Out32(0xE000A000+0x0010, 0xfffe0001);
4. Imprimir el mensaje "CPU0: Hello World" en la UART.
   xil_printf("CPU0: Hello World\n\r");

5. Espera a que se vacíe el UART TX FIFO.

```
        while ((Xil_In32(STDOUT_BASEADDRESS + 0x2C) & 0x08) != 0x08);
        COMM_VAL = 1;
        while(COMM_VAL == 1);
    }
```

Los bits de estado para las interrupciones FIFO se ilustran en la figura 4.5. Estos bits de estado de interrupción están en los registros Channel Status (uart.Channel_sts_reg0) y Estatus de interrupción del canal (uart.Chnl_int_sts_reg0) con la excepción de que los bits [TOVR] y [ROVR] no forman parte del registro uart.Channel_sts_reg0.

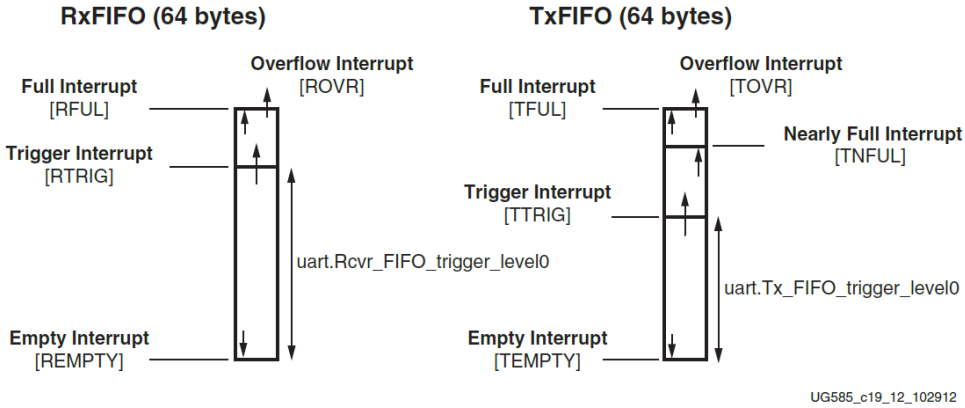


Figura 4.5 Interrupción UART RxFIFO y TxFIFO.

Los niveles de disparo FIFO están controlados por los siguientes campos de bits:

- uart.Rcvr_FIFO_trigger_level0 [RTRIG], un campo de 6 bits
- uart.Tx_FIFO_trigger_level0 [TTRIG], un campo de 6 bits

6. Se define una ubicación de memoria en la OCM que es utiliza como bandera de interrupción.
7. Espera a que se borre la ubicación de la memoria en OCM que se utiliza como bandera de semáforo.

Después de que el PS se enciende y la ROM de arranque interno completa la ejecución, CPU1 se re-direcciona a una pequeña parte de código en la OCM en 0xFFFFFE00. Esta pieza de código es un ciclo continuo que espera un evento, verifica la ubicación de la dirección 0xFFFFFFF0 para un valor específico y luego continúa el ciclo. Si 0xFFFFFFF0 no contiene el valor específico, el CPU1 salta a la dirección obtenida. En esta aplicación, el CPU1 no se utiliza, por lo que continúa ejecutando el ciclo de espera de eventos de forma indefinida.
El CPU0 se mantiene entre el punto 4 al punto 7 de forma indeterminada.
El procesador MicroBlaze usa el parámetro "C_BASE_VECTORS = 0x30000000" que indica a MB0 que use la ubicación de memoria 0x30000000 como vector de reinicio. Cuando CPU0 libera el reinicio de PL, MB0 comienza a ejecutar el código en 0x30000000.
El FSBL es responsable de colocar el MB0 ELF en 0x30000000.

4.6.2 Aplicación bare–metal del MB0.

En la dirección de memoria 0x30000000 se encuentra la aplicación del MB0. El FSBL coloca el ELF del MB0 y el script enlazador (linker) es usado para definir dirección de inicio.

La aplicación del MB0 realiza los siguientes procesos:

1. Inicializa el controlador de interrupciones y el subsistema de interrupciones.
   `init_platform();` → Inicio de la plataforma MB0

   ```
   irq_count = 0;

   IrqGenInstancePtr.Config.DeviceId = IRQ_GEN_ID;
   IrqGenInstancePtr.Config.BaseAddress = IRQ_PCORE_GEN_BASE;
   IrqGenInstancePtr.IsReady = XIL_COMPONENT_IS_READY;
   IrqGenInstancePtr.IsStarted = 0;
   ```

   Se inicia el controlador para la instancia IrqGen, en tres fases. La primer es identificar el dispositivo, la segunda fase es el direccionamiento y la última fase es cuando la instancia se encuentra en operación.
   En el código fuente también es considerado si existe un error en estas tres fases. Y está contemplado en las siguientes líneas de código:

   ```
   Status = SetupInterruptSystem();
   if (Status != XST_SUCCESS) {
   return XST_FAILURE;}
   ```

2. Espera a que se establezca una ubicación de memoria en OCM que se utiliza como bandera de semáforo.
   `COMM_VAL == 0`
3. Imprime "MB0: Esperando IRQ" en la UART

   ```
   while(COMM_VAL == 0);
   xil_printf("MB0 : Waiting for IRQ\n\r");
   ```

4. Espera a que la rutina del servicio de interrupción incremente la variable irq_count.
   `while(irq_count == 0);`
5. Imprime "MB0: IRQ reconocido" en la UART y borra la variable irq_count.

   ```
   xil_printf("MB0 : IRQ Acknowledged\n\r");
   irq_count = 0;
   ```

   La variable irq_count se le asigna un valor de cero.
6. Espera a que se vacíe el UART TX FIFO. Que en forma de código queda como:

   ```
   while ((Xil_In32(UART_BASE + 0x2C) & 0x08) != 0x08);
   ```

7. Borra la ubicación de la memoria en OCM que se utiliza como bandera de semáforo.

   ```
   COMM_VAL = 0;
   ```

   En esta parte se pasa el control al CPU0 de la parte del Sistema de Procesamiento PS. También se hace una depuración de la plataforma MB0 que está sustentado con la siguiente línea: `cleanup_platform();`
   A si logrando un estado inicial en el siguiente ciclo.

4.7 Ejecución del diseño.

El directorio de trabajo del diseño está ubicado en la dirección:
C:\xup\embedded\ISE14_5\design
El sistema operativo: Windows 7 Professional 64 bits.
Procesador: Intel Core i7-4510U 2Ghz.
Memoria ram: 8 GB.
Placa de evaluación: HW-Z7ZC702 REV 1.1

Cable USB tipo A a USB mini-B (para comunicaciones UART).
La terminal se usa la del entorno de desarrollo del SDK.
Se instalan los controladores de **CP210x_Windows_Drivers.**

En el kit de desarrollo de sistemas embebidos (EDK) de xilinx se tiene el Estudio de Plataforma de Xilinx (XPS) y el Kit de desarrollo de software (SDK). En la figura 4.6 se muestra las interfaces de bus.

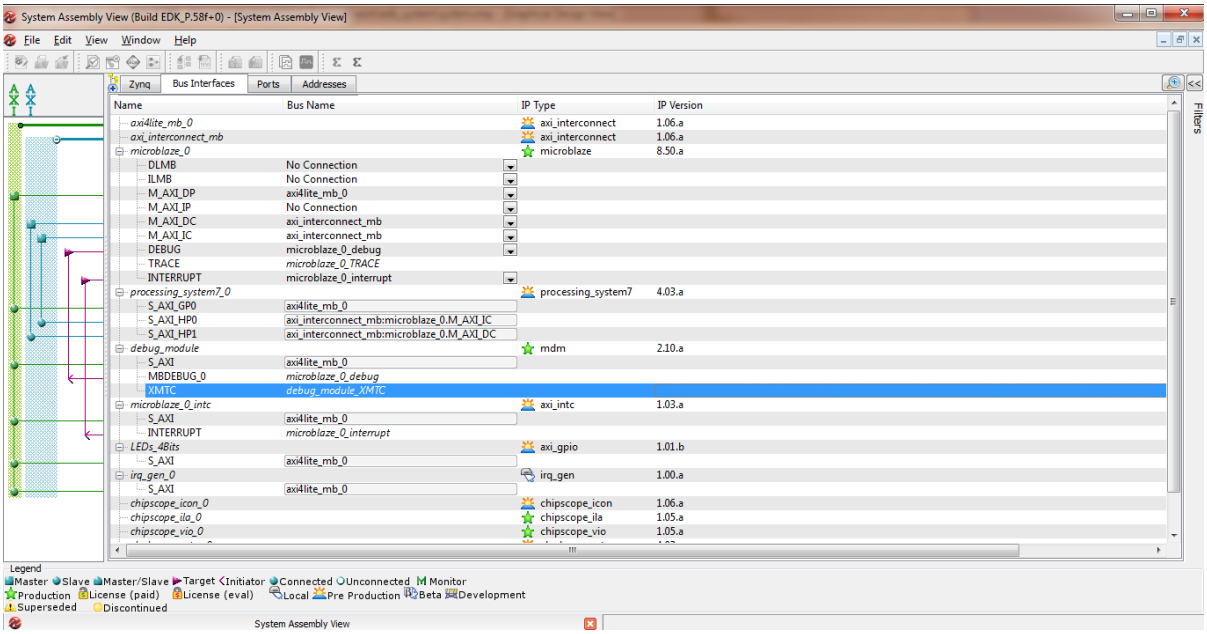


Figura 4.6 Estudio de plataforma de xilinx del Multi Procesamiento Asimétrico.

De ahí se hace una exportación del proyecto a la herramienta del Kit de desarrollo de software de xilinx. En la figura 4.7 se muestra el entorno del SDK del proyecto del Multi Procesamiento Asimétrico.

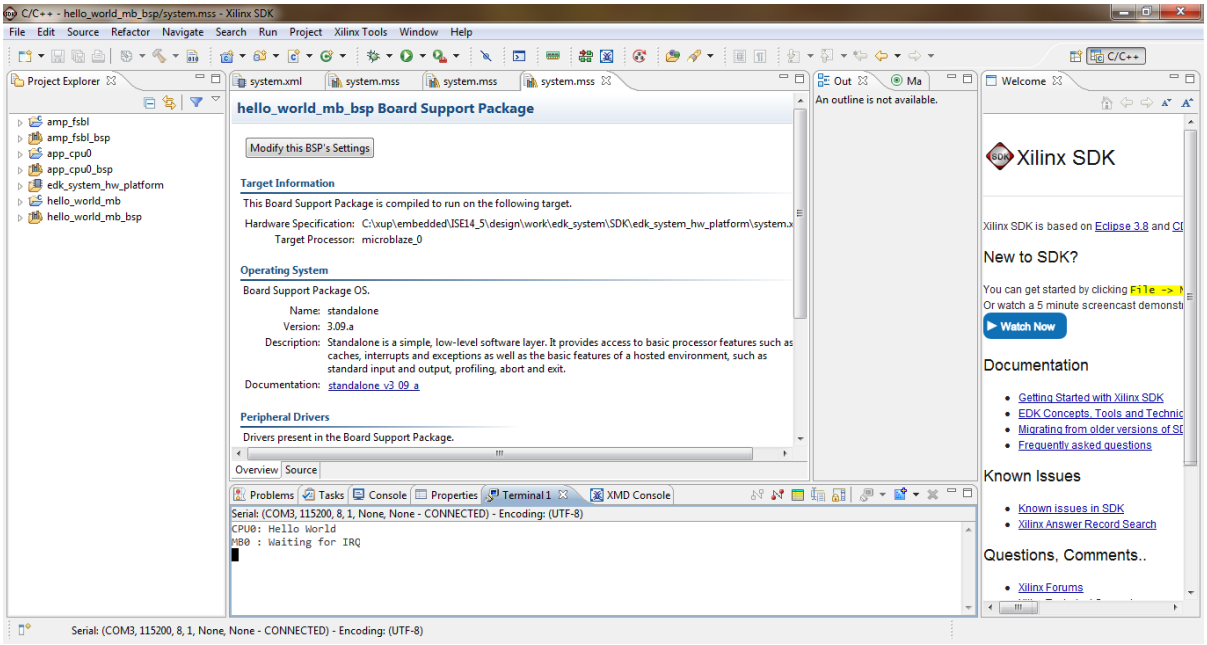


Figura 4.7 SDK del Proyecto MPA

Dentro del SDK se observa 7 proyectos en su conjunto que tienen que ver en la parte del FSBL, la aplicación bare metal del CPU0, La aplicación bare metal del MB0, la plataforma del hardware del EDK.

Para la ejecución de las pruebas se hacen con la herramienta del analizador ChipScope de Xilinx. Mientras está corriendo la aplicación en SDK, se ejecuta el ChipScope y se monitorea dos dispositivos: ARM_DAP y XC7Z020. Estos dispositivos son el CPU0 y el MB0. En la figura 4.8 se observa el entorno de la herramienta ChipScope.

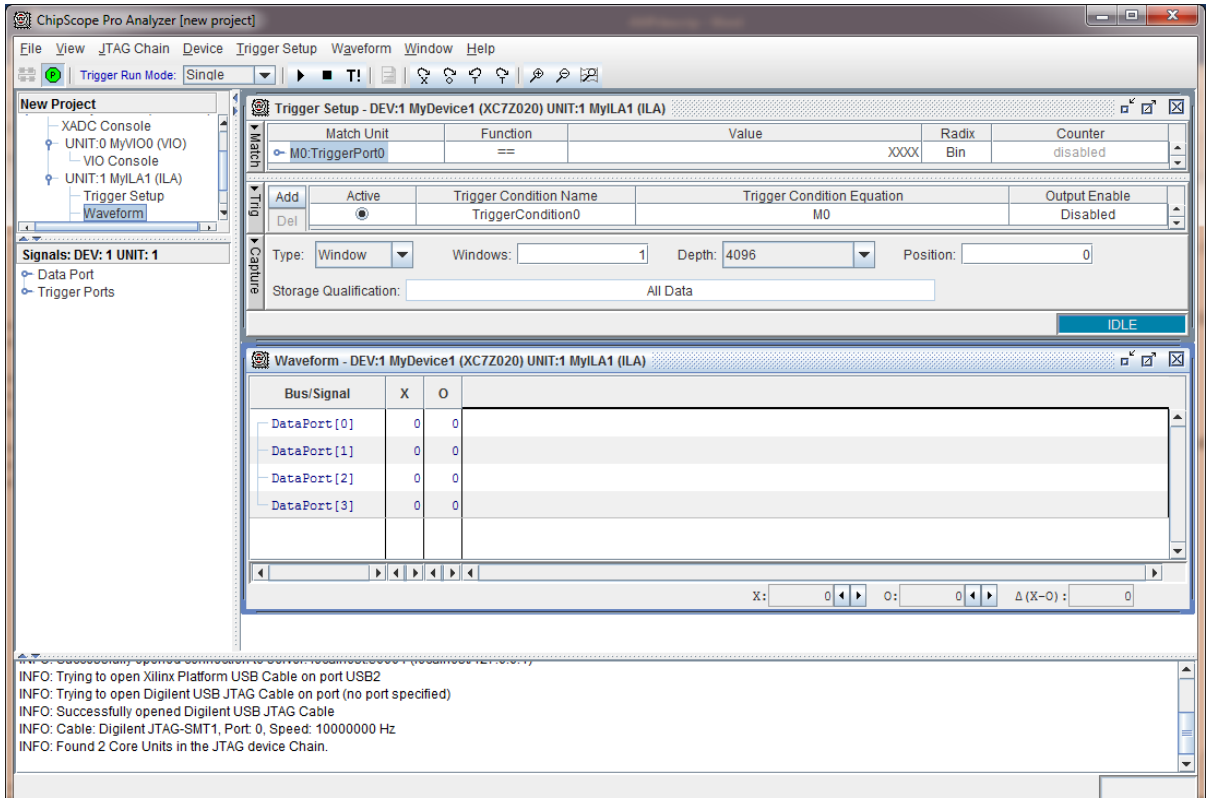


Figura 4.8 ChipScope para el análisis del sistema MPA

Con la herramienta ChipScope se tiene a disposición el dispositivo gen_irq, con el simple hecho de oprimir el botón se crea una interrupción al sistema y se puede monitorear sus efectos. Esto es mostrado en la figura 4.9.

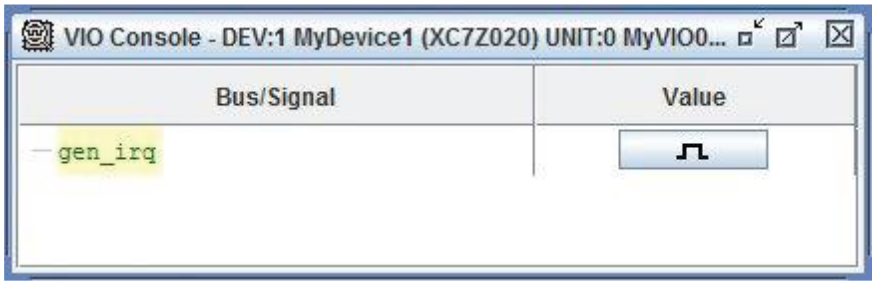


Figura 4.9 Generación de interrupción por medio gen_irq.

Después de haber ejecutado la interrupción se genera una gráfica de ondas de señales como son el vio_sync_out, vio_rise_edge, irq_gen_IRQ y slv_reg. De una forma gráfica se visualiza el comportamiento del sistema de MPA entre el CPU0 y MB0. En la figura 4.10 se muestra todo el entorno del ChipScope en este proyecto.

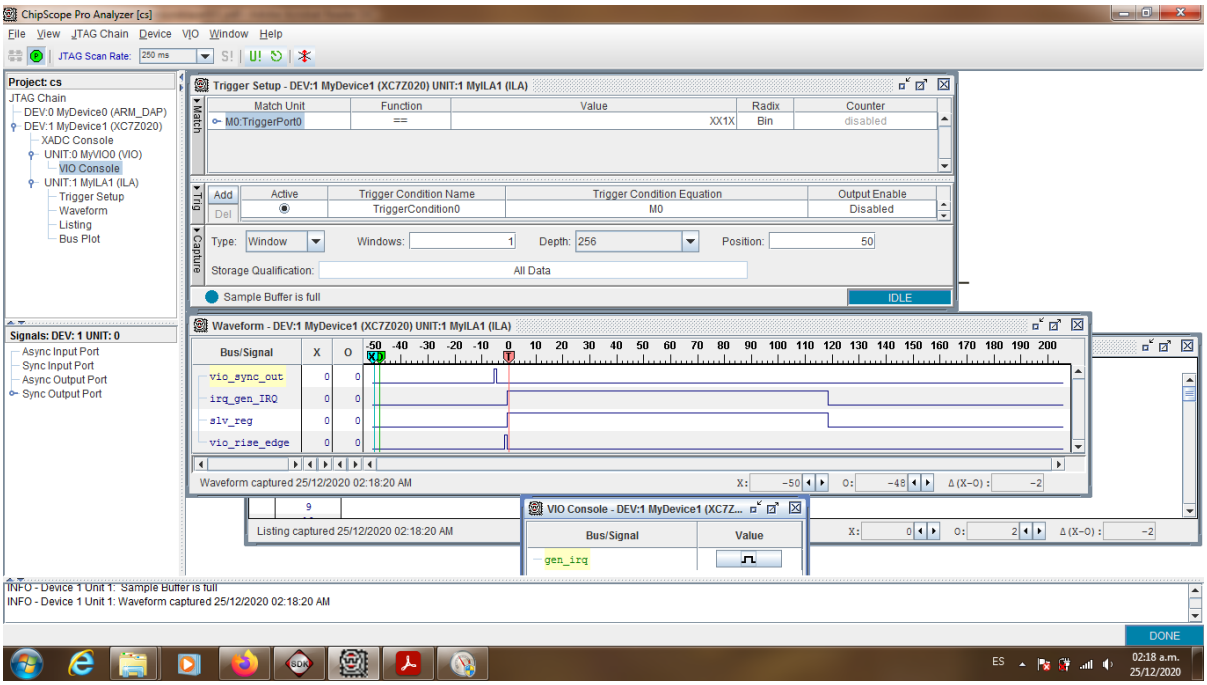


Figura 4.10 ChipScope en la ejecución del MPA.

Mientras la ejecución del ChipScope en el kit de desarrollo de software SDK se muestra en la ventana de **terminal 1** la secuencia de mensajes emitidos por CPU0 y MB0. Como es mostrado en la figura 4.11.

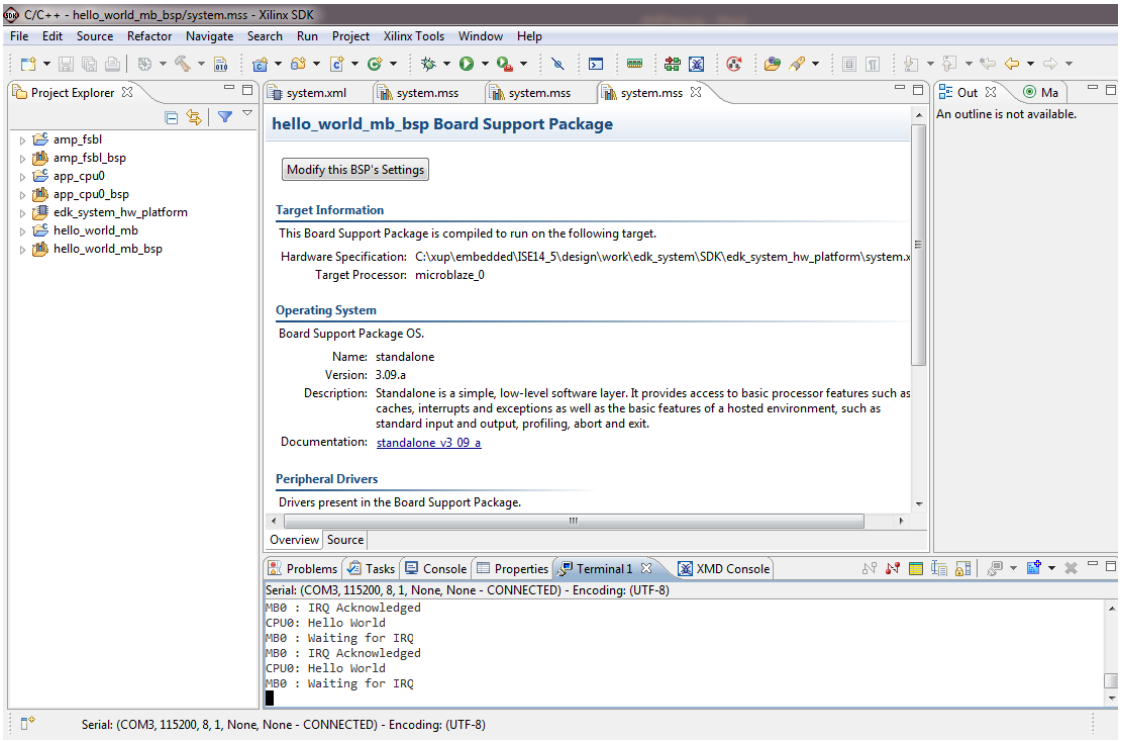


Figura 4.11 Ventanas del SDK en la ejecución del ChipScope en proyecto de MPA.

Trabajos futuros

Como trabajo futuro en la placa de evaluación ZC702 y tomando en cuenta que se tienen dos procesadores ARM Cortex A9, se tendría que evaluar el rendimiento de los procesos suaves y duros en función la densidad recomendable de los procesadores suaves. Como lo que pasa en los procesadores Intel Pentium con la tecnología Hyper-Threading que está basada en múltiples hilos, y mejora la reacción y el tiempo de respuesta.

Otro trabajo a futuro seria montar un escenario de comunicación mínimo de un switch (cisco) de internet para garantizar la compatibilidad con las redes actuales. En un extremo estaría la placa de evaluación y en el otro extremo un sistema mínimo para activar una secuencia de conteo. Mediante el puerto de comunicación Ethernet PHY RGMI de la placa. Y el monitoreo del sistema mínimo hacerlo con un procesador de núcleo suave. Como primer paso.
Como segundo paso incrementar los sistemas mínimos en función de los procesadores de núcleo suave.

Otra aplicación a futuro de mayor complejidad tener sistemas mínimos en un puerto serial y en el puerto de comunicación Ethernet PHY RGMI, y también poder almacenar los reportes de los eventos de cada uno de los sistemas mínimos en una unidad de almacenamiento que esté conectada de forma serial. La tarea control de cada sistema mínimo sea asignada a un procesador de núcleo suave.

Después de los anteriores trabajos ahora recrear los escenarios en una red de tipo industrial que podría ser bajo una topología PROFINET. Y en lugar de sistemas mínimos serian módulos industriales.

Todos estos trabajos a futuro se explotarían todas las ventajas que ofrece la plataforma de la placa de evaluación ZC702, pero también explotar al máximo los sistemas embebidos con dispositivos duros y blandos. En otras palabras, se explotarían los beneficios del procesamiento asimétrico en un FPGA Zynq-7000 XC7Z020. Con recursos de aplicaciones bare-metal.

Conclusiones

En este trabajo el punto más importante es el de trabajar con dos tipos de microprocesadores (Cortex-A9 y MicroBlaze), que cada uno tiene sus propias cualidades y característica. Desde el aspecto del hardware y el software tienen características muy dispares esto es compensado con la flexibilidad de reconfiguración del sistema del FPGA.

El reto es de poder compartir recursos entre ellos, es muy importante señalar que el concepto *Bare-Metal* es fundamental para la coexistencia en un entorno operativo de microprocesadores de diferente naturaleza. A todo esto, se encierra un concepto de *multiprocesamiento asimétrico*.

Otro punto que se debe comentar el lenguaje C embebido de xilinx que no tiene una estandarización ANSI c. El lenguaje C embebido de xilinx obedece más a las características de manejo del hardware y de los diferentes dispositivos IPs de xilinx en un aspecto de programación de bajo nivel de su direccionamiento, para manejar configuraciones internas del hardware que es contenido en un FPGA de xilinx. Tanto en la parte PL y como el PS.

Se probó el funcionamiento del diseño con la herramienta ChipScope para accionar una interrupción para ver la conmutación operativa de los dos microprocesadores. Y en la herramienta SDK se observan las salidas de forma secuencial de la operación de ambos microprocesadores.

Apéndice A

Técnicas y herramientas para el diseño de un sistema embebido-hardware, en un FPGA.

En esta parte nos enfocamos en la creación y diseño de un sistema embebido utilizando el procesador de núcleo suave MicroBlaze versión 8.20.a de 32 bits, conceptualizando todo su entorno y sus interconexiones existentes. Para una implementación en un FPGA.

A.1 Kit de desarrollo embebido

(Xilinx® Embedded Development Kit EDK)

Xilinx tiene un kit de desarrollo para los sistemas embebidos la cual simplifica mucho el diseño de un procesador embebido. Aunque en realidad sería muy complejo hacerlo sin esta herramienta. El EDK viene integrado dentro de una Suite de diseño ISE edición embebida, que está conformada por las siguientes herramientas:

- Entorno de software integrado (ISE)
- Software de análisis de diseño PlanAhead ™
- ChipScope ™ Pro (que es útil para la depuración en chip de diseños FPGA)
- Kit de desarrollo embebido (EDK). EDK también está disponible con ISE Design Suite: System Edition, que incluye herramientas para el diseño DSP.

El Kid de Desarrollo Embebido está conformado por dos grandes herramientas que son:

- Xilinx Estudio de plataforma (Xilinx Platform Studio XPS)
- Kit de desarrollo de software (Software Development Kit SDK)

Xilinx Platform Studio (XPS) es el entorno de desarrollo utilizado para diseñar la parte de hardware del sistema del procesador embebido. Puede ejecutar XPS en modo por lotes o utilizando la Interfaz Gráfica de Usuario GUI.

En la segunda parte del EDK está conformado por El Kit de desarrollo de software (Software Development Kit SDK) es un entorno de desarrollo integrado, complementario a XPS, que se utiliza para la creación y verificación de aplicaciones de software integradas C/C ++. SDK está basado en el marco de código abierto de Eclipse.

En la figura A.1 se muestra el proceso de diseño de un sistema embebido usando las herramientas XPS y SDK.

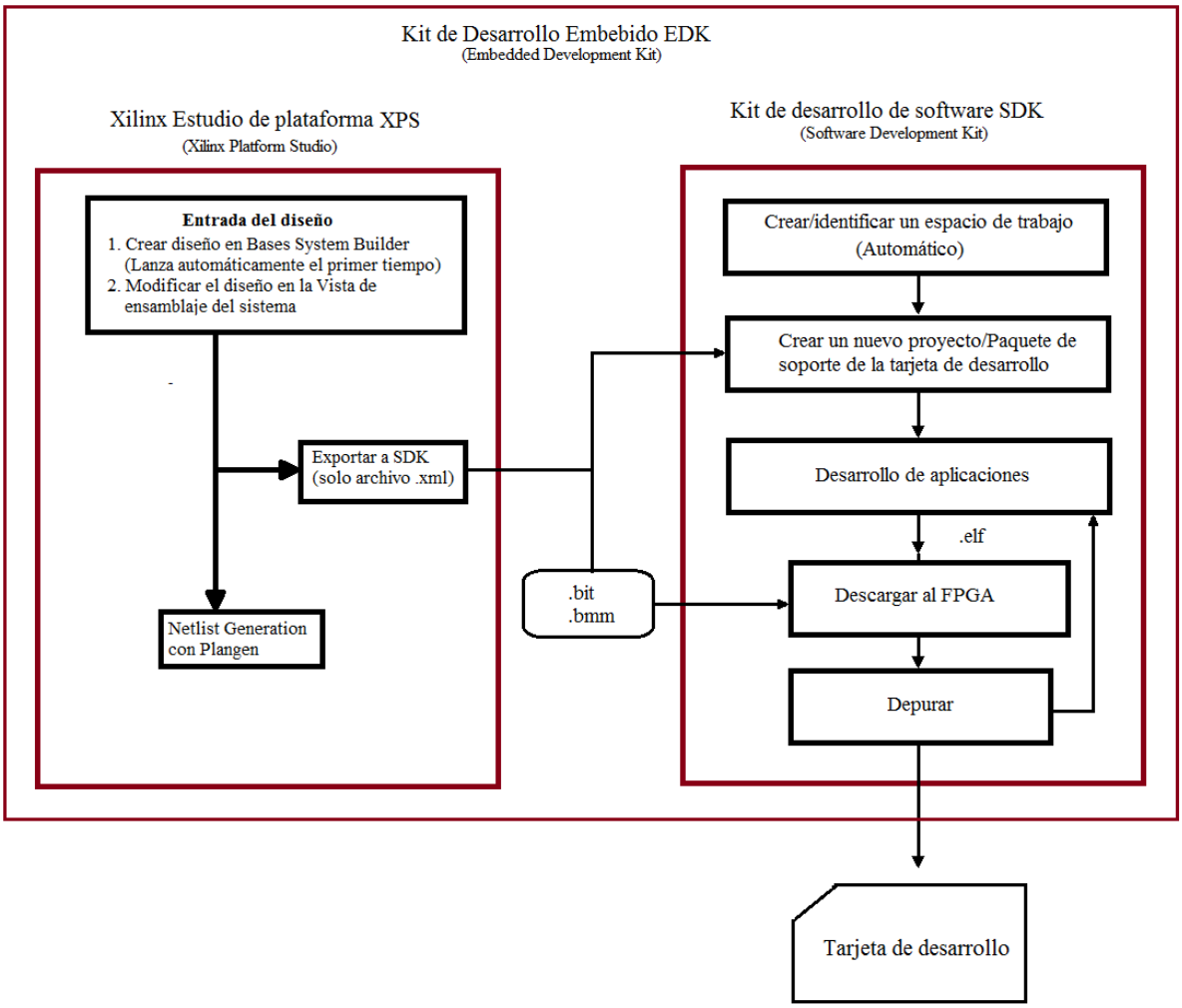
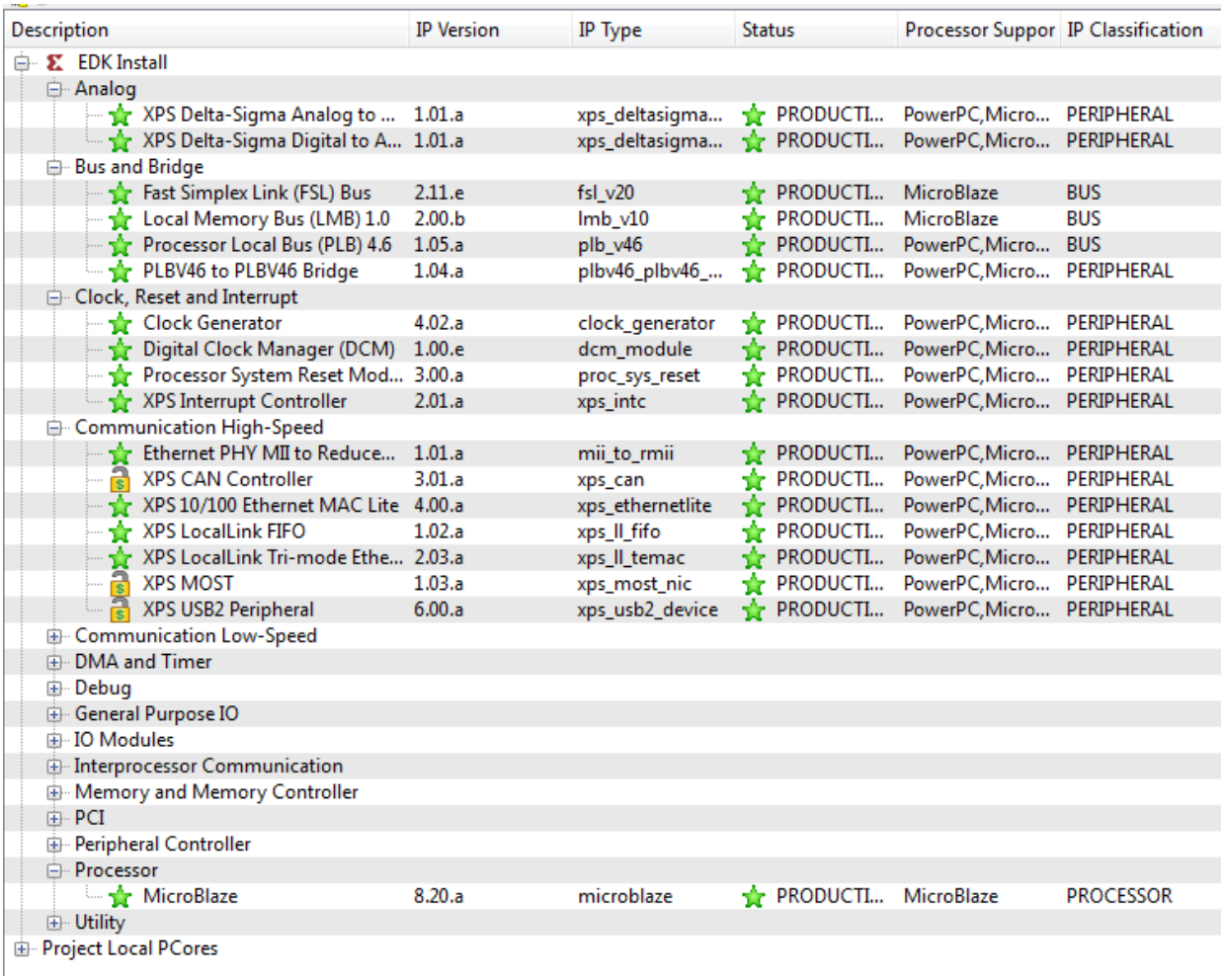


Figura A.1 Flujo del proceso de diseño embebido básico

También EDK posee una rica biblioteca de Propiedad Intelectual (IP), que son dispositivos que nos permiten trabajar en nuestros diseños personalizados de nuestro sistema embebido a desarrollar. En la Figura A.2 se muestra el Catalogo de IP´s de los dispositivos que existen.



Description	IP Version	IP Type	Status	Processor Suppor	IP Classification
EDK Install					
Analog					
XPS Delta-Sigma Analog to ...	1.01.a	xps_deltasigma...	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS Delta-Sigma Digital to A...	1.01.a	xps_deltasigma...	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
Bus and Bridge					
Fast Simplex Link (FSL) Bus	2.11.e	fsl_v20	PRODUCTI...	MicroBlaze	BUS
Local Memory Bus (LMB) 1.0	2.00.b	lmb_v10	PRODUCTI...	MicroBlaze	BUS
Processor Local Bus (PLB) 4.6	1.05.a	plb_v46	PRODUCTI...	PowerPC,Micro...	BUS
PLBV46 to PLBV46 Bridge	1.04.a	plbv46_plbv46_...	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
Clock, Reset and Interrupt					
Clock Generator	4.02.a	clock_generator	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
Digital Clock Manager (DCM)	1.00.e	dcm_module	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
Processor System Reset Mod...	3.00.a	proc_sys_reset	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS Interrupt Controller	2.01.a	xps_intc	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
Communication High-Speed					
Ethernet PHY MII to Reduce...	1.01.a	mii_to_rmii	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS CAN Controller	3.01.a	xps_can	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS 10/100 Ethernet MAC Lite	4.00.a	xps_ethernetlite	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS LocalLink FIFO	1.02.a	xps_ll_fifo	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS LocalLink Tri-mode Ethe...	2.03.a	xps_ll_temac	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS MOST	1.03.a	xps_most_nic	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
XPS USB2 Peripheral	6.00.a	xps_usb2_device	PRODUCTI...	PowerPC,Micro...	PERIPHERAL
Communication Low-Speed					
DMA and Timer					
Debug					
General Purpose IO					
IO Modules					
Interprocessor Communication					
Memory and Memory Controller					
PCI					
Peripheral Controller					
Processor					
MicroBlaze	8.20.a	microblaze	PRODUCTI...	MicroBlaze	PROCESSOR
Utility					
Project Local PCores					

Figura A.2 Catalogo de IP´s del EDK 13.2

A.2 Creación de un nuevo proyecto de un sistema embebido.

Para la creación de un proyecto según la figura A.1. Primero nos adentramos en la herramienta llamada Xilinx Estudio de plataforma (Xilinx Platform Studio XPS) donde se inicia con una ventana llamada Base System Builder la que nos guía en la construcción de un sistema base. Una vez agotada esta herramienta pasamos a Xilinx Estudio de plataforma (Xilinx Platform Studio XPS) se define un nuevo proyecto donde se establece una estructura de archivos. Una vez ya definido el proyecto, definimos el tipo de interconexiones que se van usar en nuestro proyecto. Existen dos opciones:

- Sistema AXI
  AXI es un estándar de interfaz que es usado en las versiones actuales y futuras de Xilinx IP y flujos de herramientas. Y más aun si se cuenta con microprocesadores ARM con esta configuración se le saca provecho a sus características a los ARM.
- Sistema PLB
  PLB es el estándar de bus heredado utilizado por Xilinx que admite las familias FPGA no tan actuales, incluidas Spartan6 y Virtex6. PLB no admitirá nuevas familias FPGA.

Después de definir el tipo de interconexión a usar, se genera un archive .xmp (system.xmp) que es determinado por Xilinx Platform Studio XPS y así se da paso a la construcción del sistema base con la aplicación Base System Builder, que esta conformado por 7 pasos:

1. Welcome
   En esta parte se da la opción de crear un nuevo diseño o retomar un diseño existente a partir de un archivo .bsb.
2. Borad.
   Aquí se define el vendedor, el nombre y la revisión de la tarjeta de desarrollo donde se va implementar nuestro diseño.
3. System
   Definimos si nuestro diseño si es de un procesador o si es dos procesadores.
4. Processor
   Establecemos la frecuencia de referencia del reloj, el tipo de procesador, el tamaño de la memoria local.
5. Peripheral
   Se adiciona en esta parte los periféricos con los que se van a trabajar en nuestro diseño, así como también se configuran sus variables.
6. Cache
   Definimos la memoria cache qe se maneja en nuestro diseño.
7. Summary
   En este último punto se genera un resumen del direccionamiento de los periféricos y de los archivos generados por nuestro diseño. Como se muestra en la figura A.3.

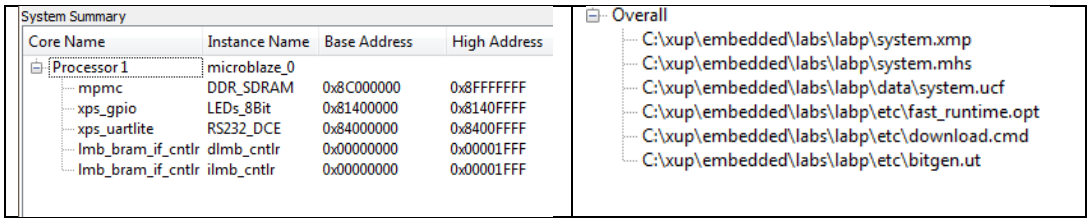


Figura A.3 Resumen del Base System Builder.

Regresando a la herramienta Xilinx Estudio de plataforma (Xilinx Platform Studio XPS) se define más detalladamente nuestro diseño con respecto a sus variables de configuración como es el direccionamiento, conexiones de nuestros periféricos.
En XPS se puede manipular los buses, configurar los puertos y el direccionamiento de los periféricos. Cuenta con una herramienta donde genera la visualización de diagrama de bloques de nuestro proyecto.

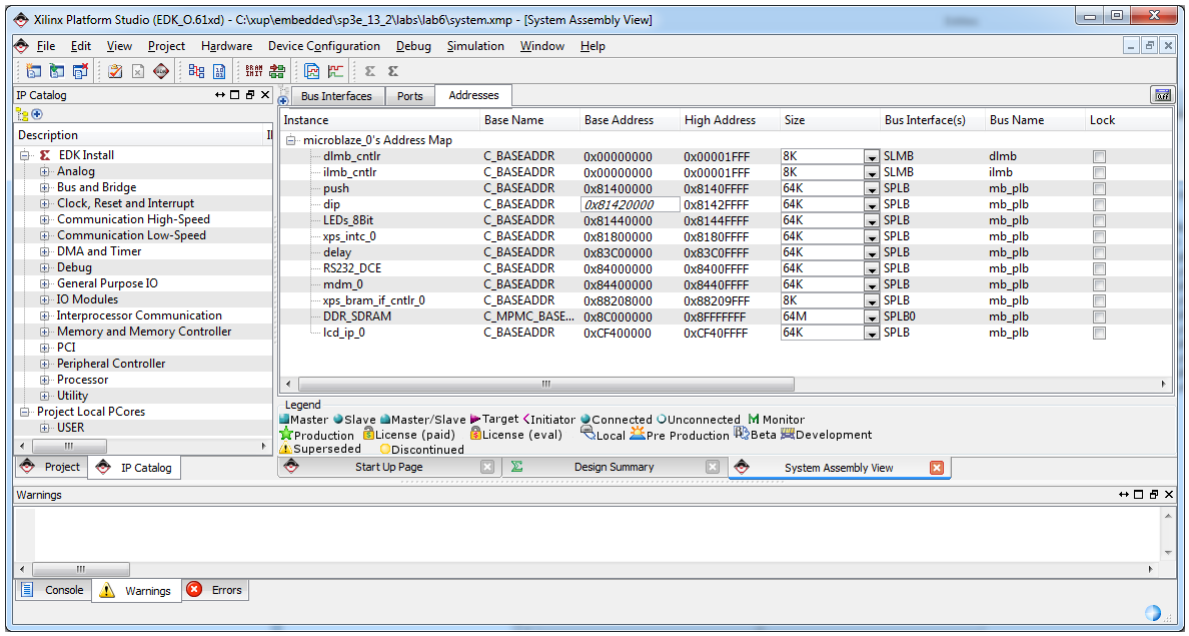


Figura A.4 Entorno de Xilinx Estudio de plataforma (Xilinx Platform Studio XPS)

En Xilinx Estudio de plataforma (Xilinx Platform Studio XPS) existe una Herramienta de generación de plataforma de hardware. La herramienta de generación de plataforma de hardware (Platgen) personaliza y genera el sistema de procesador embebido, en forma de listas de redes de hardware (archivos HDL).
De manera predeterminada, Platgen sintetiza cada instancia de núcleo de IP de procesador que se encuentra en su diseño de hardware embebido utilizando el compilador Xilinx® Synthesis Technology XST. Platgen también genera el archivo HDL a nivel de sistema que interconecta todos los núcleos IP, para ser sintetizado posteriormente como parte del flujo general de implementación de Xilinx® ISE®. Este proceso si es tardado. Cuando es realizado este proceso satisfactoria mente en la Consola tenemos un mensaje parecido al que se muestra a continuación:

```
Rebuilding cache ...

Total run time: 589.00 seconds
"Running synthesis..."
cd synthesis & synthesis.cmd
"xst -ifn "system_xst.scr" -intstyle silent"
"Running XST synthesis ..."
PMSPEC -- Overriding Xilinx file
<C:/Xilinx/13.2/ISE_DS/EDK/spartan3/data/spartan3.acd> with local file
<C:/Xilinx/13.2/ISE_DS/ISE/spartan3/data/spartan3.acd>
"XST completed"
Done!
```

En este punto ya somos capaces de exportar nuestro proyecto a la herramienta del Kit de desarrollo de software (SDK).
Pero existen tres archivos que ya se generaron en su versión final que son: .mhs, .bit y el .bmm.

El archivo **.mhs** (microprocessor hardware specification) es creado por Base System Builder.
Un archivo **.bmm** (Block Memory Map) es un archivo de texto que tiene descripciones sintácticas de la RAM.
Archivo .bit Un flujo de bits (bitstream) FPGA es un archivo que contiene la información de programación para un FPGA. Un dispositivo FPGA Xilinx debe programarse utilizando un flujo de bits específico para que se comporte como una plataforma de hardware embebida. Este flujo de bits generalmente lo proporciona el diseñador de hardware que crea la plataforma embebida [13].
La programación de un FPGA es el proceso de cargar un flujo de bits en el FPGA. Durante la fase de desarrollo, el dispositivo FPGA se programa utilizando utilidades como Vivado® o las opciones de menú en SDK. Estas herramientas transfieren el flujo de bits al FPGA a bordo. En el hardware de producción, el flujo de bits generalmente se coloca en una memoria no volátil, y el hardware está configurado para programar el FPGA cuando se enciende [13].

El kit de desarrollo de software (SDK) Xilinx® facilita el desarrollo de proyectos de aplicaciones de software embebido. SDK es una herramienta complementaria a XPS. Utiliza SDK para desarrollar el software que se utiliza en la plataforma embebida en XPS. SDK se basa en el conjunto de herramientas de código abierto Eclipse.
Algunos términos comunes utilizados al describir la operación del SDK incluyen:

- Espacio de trabajo
  Un espacio de trabajo es una ubicación de directorio utilizada por SDK para almacenar datos y metadatos del proyecto. Se puede crear múltiples espacios de trabajo para administrar más fácilmente múltiples versiones de software.
- Proyecto de software
  Un proyecto de software contiene uno o más archivos de origen, junto con los archivos de encabezado necesarios, para permitir la compilación y la generación de un archivo de salida binaria (.elf).
- plataforma de hardware
  Debe tener una plataforma de hardware para su diseño. La plataforma de hardware es el diseño de hardware embebido que se crea en XPS. La plataforma de hardware incluye el archivo de descripción de hardware basado en XML, el archivo de flujo de bits y el archivo BMM. Cuando importa el archivo XML en SDK, importa la plataforma de hardware. Pueden existir múltiples plataformas de hardware en un solo espacio de trabajo.

- Paquete de soporte de la tarjeta
  Un paquete de soporte de tarjeta (Board Support Package BSP) es una colección de bibliotecas y controladores que forman la capa más baja de la pila de software de su aplicación. Sus aplicaciones de software deben vincularse o ejecutarse sobre un paquete de soporte de tarjeta dado, utilizando las Interfaces de Programa de Aplicación (Application Program Interfaces API) proporcionadas.
- Perspectivas
- Puntos de vista

Al exportar el diseño del hardware a SDK, se define un área de trabajo y se predefinen dos subdirectorios: SDK y SDK_Export. Como consta en la figura A.5.

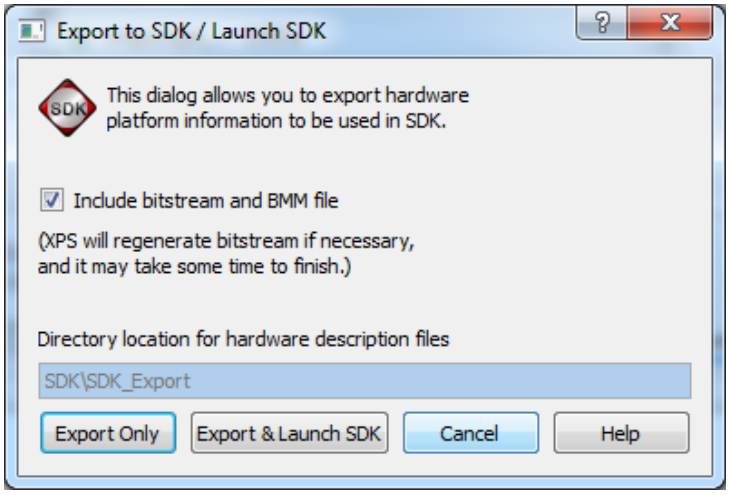


Figura A.5 De XPS se exporta el diseño del hardware a SDK.

En SDK, seleccione Nuevo proyecto Xilinx C, nos va a solicitar un nombre del proyecto y que seleccionemos un template o uno propuesto como diseñador del sistema embebido. Esto está ilustrado en la Figura A.6.

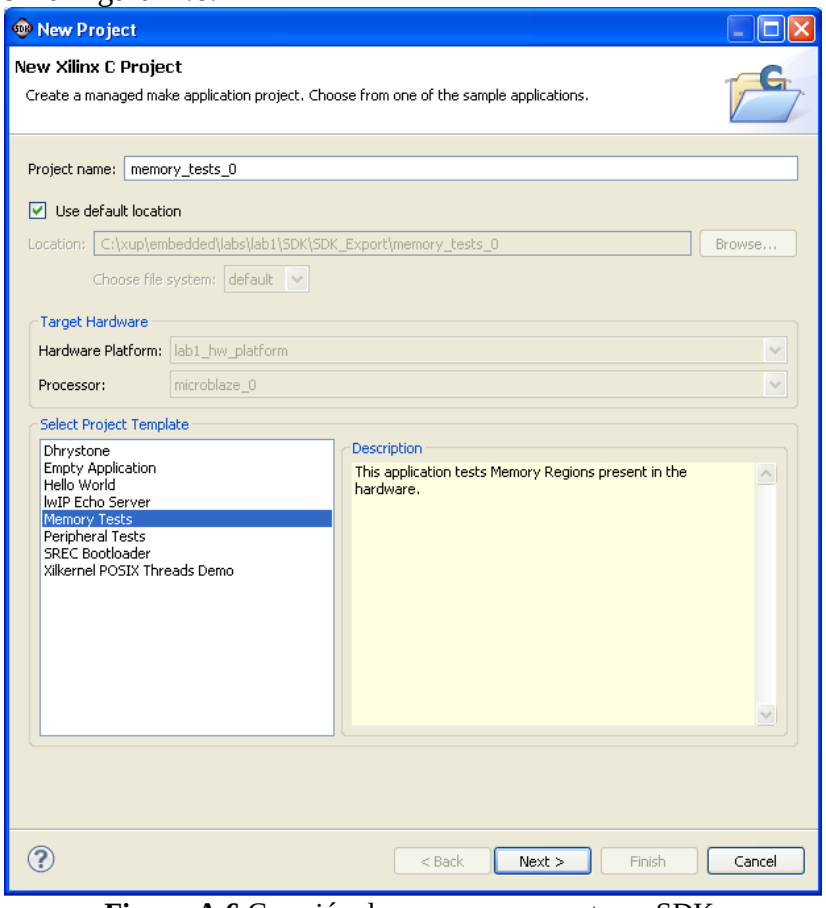


Figura A.6 Creación de un nuevo proyecto en SDK.

EL SDK tiene las siguientes características:

- Admite el desarrollo de aplicaciones de software en sistemas de procesador único o multiprocesador.
- Admite el desarrollo de aplicaciones de software en un entorno de equipo.
- Capacidad para crear y configurar paquetes de soporte de placa (BSP) para sistemas operativos de terceros.
- Proporciona proyectos de software de muestra disponibles para probar la funcionalidad del hardware y el software.
- Tiene una interfaz GUI fácil para generar scripts de enlazador para aplicaciones de software, programar dispositivos FPGA y programar memoria flash paralela.
- Tiene un editor de código C / C ++ rico en funciones y un entorno de compilación.
- Proporciona gestión de proyectos • Configura las compilaciones de aplicaciones y automatiza la generación de archivos de creación.
- Suministros de navegación de error.
- Proporciona un entorno bien integrado para depurar y perfilar sin problemas los objetivos integrados.
- Tiene una consola Xilinx Microprocessor Debugger (XMD), XMD proporciona una interfaz de lenguaje de comando de herramienta (Tool Command Language Tcl). Esta interfaz se puede utilizar para el control de línea de comandos, así como para ejecutar scripts de prueba de verificación para probar un sistema completo.

Todas estas caracteristicas estan contenidas en su interfaz gráfica, es mostrada en la figura A.7.

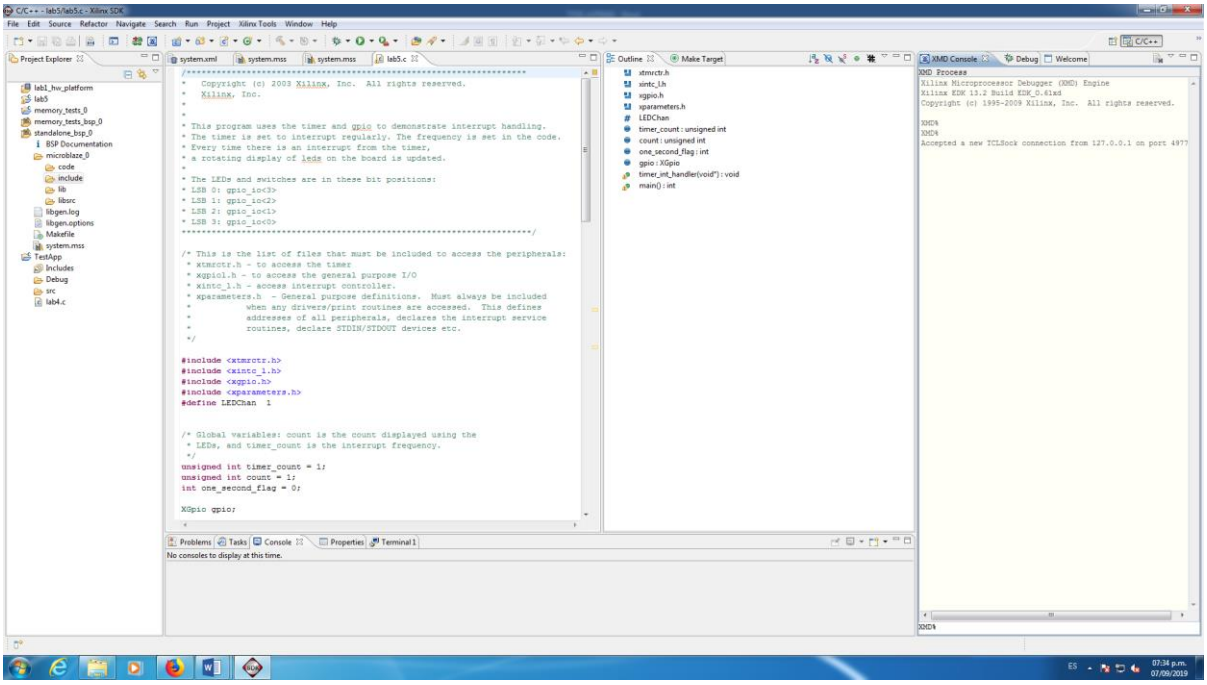


Figura A.7 Entorno gráfico del Kit de desarrollo de software (Software Development Kit SDK)

Finalmente ya teniendo nuestro proyecto terminado y sin errores proseguimos a la programación del FPGA, donde es de gran importancia el archivo .ELF. en la figura A.8 se muestra cuando el SDK nos muestra una ventana cuando programamos el FPGA.

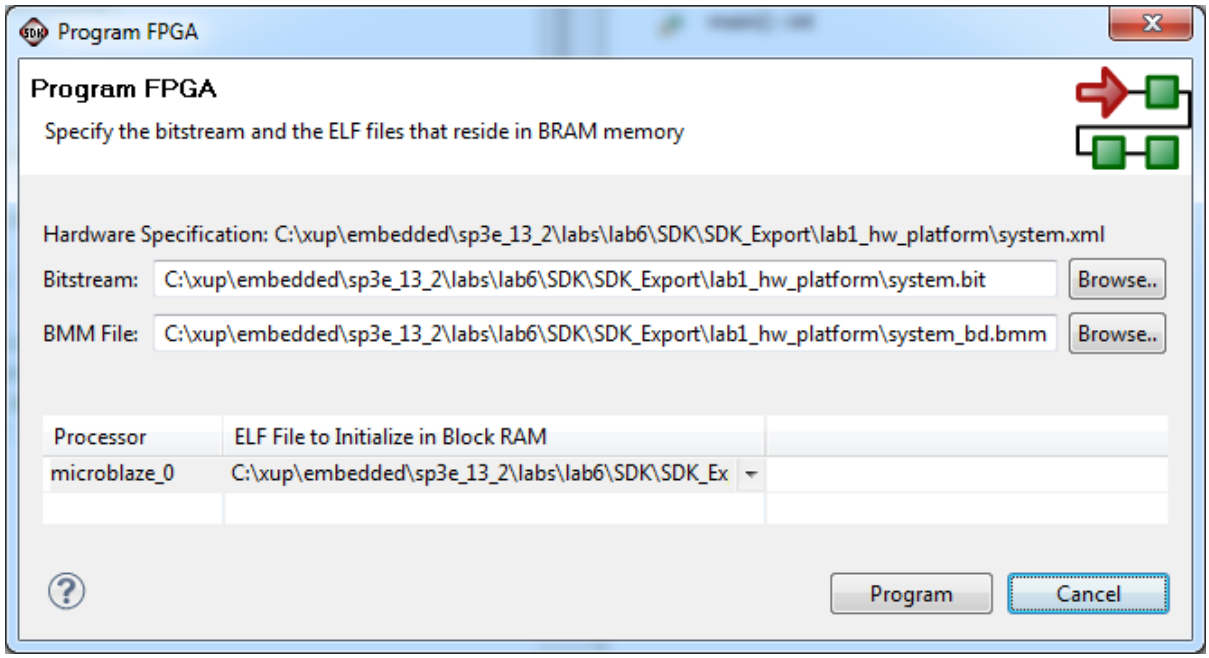


Figura A.8 Programación del FPGA.

Archivo .ELF es un Archivo de formato ejecutable y enlazable (Linkable)

A.3 Topología del SoC Zynq-7000.

El SoC Zynq-7000 está conformado en dos secciones, la primera sección es el Sistema de Procesamiento PS y la segunda sección es la Lógica Programable de Xilinx PL. Las dos secciones PS y PL se encuentran en dominios separados de energía. Por lo cual se podría apagar la sección de la PL. Ya teniendo claro estos conceptos ahora pasamos con los bloques funcionales de ambas secciones:

- Sistema de Procesamiento – PS
    - Unidad de Procesador de Aplicaciones – APU
    - Interfaces de Memoria
    - Periféricos de E/S
    - Interconexiones
- Lógica Programable - PL

En la figura A.3.1 Se muestran los bloques funcionales del Sistema de Procesamiento y la Lógica Programable del SoC Zynq-7000.

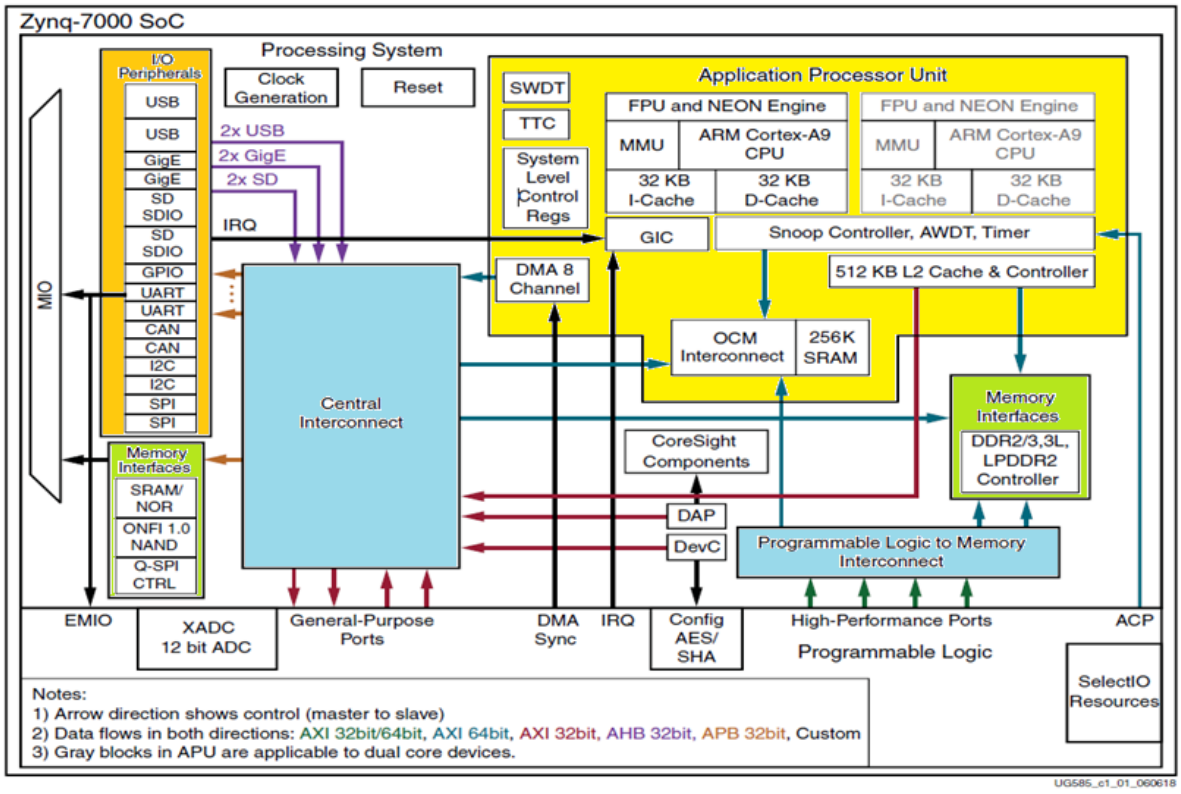


Figura A.9 Diagrama de bloques SoC Zynq-7000

En el SoC Zynq-700 existen dispositivos de terceros que a continuación se listan:

- **Cortex-A9 MPCore**.- Micro Procesador ARM.
- **AMBA Level 2 Cache Controller (PL310)**.- Controlador de caché de nivel 2, ARM.
- **PrimeCell Static Memory Controller (PL353)**.- Soporta dos interfaces de memoria, interface 0 SRAM, interface 1 NAND. ARM
- **PrimeCell DMA Controller (PL330)**.-
  El DMAC es un periférico PrimeCell compatible con la arquitectura de bus de microcontrolador avanzado (AMBA) desarrollado, probado y licenciado por ARM. El DMAC proporciona una interfaz AXI para realizar las transferencias DMA y dos interfaces APB que controlan su funcionamiento. El DMAC implementa la tecnología segura TrustZone®.
- **Generic Interrupt Controller GIC (PL390).-**
  El GIC es una arquitectura de bus de microcontrolador avanzado (AMBA) y un periférico de sistema en chip (SoC) compatible con la arquitectura ARM. Es un controlador de interrupción de alto rendimiento y área optimizada con interfaces de bus AMBA en chip que, según la configuración, se ajustan al protocolo AMBA Advanced eXtensible Interface (AXI) o al protocolo AMBA AHB-Lite.
- **CoreLink Network Interconnect (NIC-301)**.- Es un componente altamente configurable que le permite crear una infraestructura de red completa de alto rendimiento, optimizada y compatible con AMBA. Las posibles configuraciones para CoreLink Network Interconnect pueden variar desde un solo componente de puente, por ejemplo, un puente de protocolo AHB a AXI, hasta una infraestructura compleja

que consta de hasta 128 maestros y 64 esclavos de una combinación de diferentes protocolos AMBA.

- **DesignWare Cores IntelliDDR Multi Protocol Memory Controller**.-
  El Controlador de memoria DDR universal (uMCTL) es un controlador de memoria de múltiples puertos que acepta solicitudes de acceso a memoria de hasta 32 puertos host del lado de la aplicación. Las interfaces del lado de la aplicación pueden conectarse al uMCTL a través de las interfaces de bus AMBA AXI / AHB estándar o a través de la interfaz nativa extendida (ENIF) Synopsys personalizada. Los registros de configuración para el uMCTL se programan a través de la interfaz de software AMBA 2.0 APB.
- **USB 2.0 High Speed Atlantic Controller**.- Provedor Synopsys.

- **Watchdog Timer**.-
  La IP del temporizador de vigilancia Cadence® (WDT) se usa para evitar el bloqueo del sistema si el software queda atrapado en un punto muerto. En condiciones normales de funcionamiento, el usuario reinicia el watchdog a intervalos regulares antes de que el temporizador llegue a cero. Si el temporizador llega a cero y el watchdog está habilitado, se genera una a una combinación de las siguientes señales: un reinicio del sistema, una interrupción o una señal externa.
- Inter Intergrated Circuit.-
  Compatible con la especificación de Circuito Inter-Integrado (I2C), el Controlador Cadence® I2C ofrece a los usuarios un controlador Maestro/Esclavo compatible con el bus ARM® AMBA® APB de 32 bits fácil de integrar que admite modos I2C Estándar y Rápido (Fast).
- **Gigabit Ethernet MAC**.-
  En sistemas industriales se buscan velocidades de Ethernet más rápidas para resolver las mayores demandas de ancho de banda. Cadence está a la vanguardia en la corriente principal de Ethernet IP. Cumple con el estándar IEEE 802.3, el Cadence®IP para Gigabit Ethernet MAC es altamente personalizable con soporte para un 1000BASE-X integrado PCS, un DMA de alto rendimiento con funciones avanzadas de descarga AXI y almacenamiento en caché de descriptores, QoS, 1588 y TSN / AVB para admitir cualquier aplicación. Admite una serie de otras características, incluyendo IEEE 802.3az Ethernet de bajo consumo de energía (EEE), VLAN, descarga TCP/IP y monitoreo remoto de red (RMON) .La MAC IP está diseñada para integrarse rápida y fácilmente en cualquier SoC y para conectarse sin problemas a un PHY a través de interfaces estándar independientes de medios como MII, RMII, GMII, RGMII, SGMII y TBI.
- **Serial Peripheral Interface SPI**.-
  Conéctese sin problemas a Cadence o dispositivos maestros de bus compatibles con APB de terceros y periféricos SPI. La interfaz periférica en serie (SPI) de Cadence® proporciona comunicación full-duplex, síncrona y serial entre maestro y esclavo u otros dispositivos periféricos.
- **Universal Asynchronous Receiver Transmitter.-**
  Cadence® IP Factory ofrece IP personalizado y sintetizable para cumplir con los requisitos de diseño específicos. El IP del transmisor receptor asíncrono universal cumple con la especificación AMBA® 2. El IP del transmisor receptor asíncrono

universal proporciona un funcionamiento dúplex completo con FIFO configurables tanto en transmisión como en recepción.

- **Triple Timer Counter**
  Cadence® ofrece una IP personalizada y sintetizable para cumplir con los requisitos de diseño específicos. La IP del contador de temporizador triple de Cadence cumple con la especificación AMBA® 2. La IP del contador de temporizador triple de Cadence contiene tres temporizadores/contadores de 16 bits programables de forma independiente con 16 bits preescaladores.
- **SD2.0/SDIO2.0/MMC3.31 AHB Host Controller**.-
  El controlador SD / SDIO es compatible con la especificación estándar del controlador de host SD Versión 2.0 Parte A2 con SDMA (DMA de operación única), ADMA1 (DMA limitado por límite de 4 KB) y ADMA2 (ADMA2 permite transferir datos de cualquier ubicación y cualquier tamaño) en una memoria de sistema de 32 bits (compatibilidad con dispersión-reunión DMA). El núcleo también admite hasta siete funciones en SD1, SD4, pero no admite el modo SPI. Es compatible con los estándares de tarjeta SD de alta velocidad (SDHS) y SD de alta capacidad (SDHC).

Se observa que todos los dispositivos tienen compatibilidades entre sí. por lo que se debe considerar todos los detalles en sus especificaciones.

La PL de Xilinx se deriva de la tecnología FPGA de la serie 7 Artix®-7 y Kintex®-7, se utiliza para ampliar la funcionalidad para cumplir con los requisitos específicos de la aplicación.

La PL de Xilinx incluyen muchos recursos como son:

- Bloques lógicos configurables –CBL
- BRAM configurable el ancho y el puerto.
- Segmentos de DSP con un multiplicador de 25 * 18 (Procesador Digital de Señales)
- Acumulador de 48 bits y pre-sumador (DSP48E1),
- Un usuario configurable convertidor analógico a digital (XADC),
- Mosaicos de gestión de reloj (CMT),
- Un bloque de configuración con 256 AES para descifrado (Advanced Encryption Standard)
- SHA para autenticación, tecnología SelectIO ™ configurable (Secure Hash Algorithm)
- Y opcionalmente, transceptores multi-gigabit GTP o GTX y un PCI Express® integrado (PCIe) bloquear.

El concepto de Clock Management Blocks CMB, esta aprovechado en los FPGA de hoy incorporan poderosos bloques de administración de reloj para facilitar el proceso de diseño y reducir costos. Estos son referidos de diferente manera dependiendo de la empresa:

- Xilinx → usa mosaico de administración de reloj (Clock Management Tile CMT) o administrador de reloj digital (DCM),
- Intel → usa el conocido término de bucle de bloqueo de fase (PLL).
- Microsemi → usa circuitos de acondicionamiento de reloj.

Para obtener más información sobre los recursos de PL, consulte las siguientes Guías de usuario de FPGA de la serie Xilinx 7:

- *UG471, 7 Series FPGAs SelectIO Resources User Guide.*

En esta guía se describe los aceleradores de hardware en la transmisión de datos y la disposición de bancos de E/S de alto rendimiento (High-Performance - HP) y de alto rango (High-Range - HR). En los FPGAs de la serie 7. Los bancos de E/S HP están diseñados para cumplir con los requisitos de rendimiento de la memoria de alta velocidad y otras interfaces de chip a chip con voltajes de hasta 1.8V. Los bancos de E/S de alto rango HR están diseñados para admitir una gama más amplia de estándares de E/S con voltajes de hasta 3.3V.

- *UG472, 7 Series FPGAs Clocking Resources User Guide*
  En este manual se proporciona una descripción general de la sincronización de los FPGA de la serie 7, una comparación entre la sincronización de los FPGA de la serie 7 y las generaciones anteriores de FPGA, y de forma resumida la conectividad de sincronización dentro de los FPGA de la serie 7.
- *UG473, 7 Series FPGAs Memory Resources User Guide*
  En esta guía se describe el uso de la RAM en los FPGA de la serie Xilinx® 7 almacena hasta 36 Kbits de datos y se puede configurar como dos RAM independientes de 18 Kb o una RAM de 36 Kb. Y otras posibilidades más.
- *UG474, 7 Series FPGAs Configurable Logic Block User Guide*
  El bloque lógico configurable (CLB) de la serie 7 proporciona una lógica FPGA avanzada y de alto rendimiento: Tecnología de tabla de búsqueda real (LUT) de 6 entradas, Opción LUT5 dual (LUT de 5 entradas), Capacidad lógica de registro distribuido de memoria y desplazamiento, Lógica de transporte de alta velocidad dedicada para funciones aritméticas • Amplios multiplexores para una utilización eficiente los CLB son los principales recursos lógicos para implementar circuitos secuenciales y combinatorios. Cada elemento CLB está conectado a una matriz de conmutación para acceder a la matriz de enrutamiento general.
- UG476, *7 Series FPGAs GTX Transceiver User Guide*
  Descripción general y características de los FPGAs de la serie 7 en los transceptores GTX y GTH de la serie 7 son transceptores de bajo consumo de energía, que admiten velocidades de línea de 500 Mb/s a 12.5 Gb/s para transceptores GTX y 13.1 Gb/s para transceptores GTH. El transceptor GTX/GTH es altamente configurable y está estrechamente integrado con los recursos lógicos programables del FPGA. Esta parte es importante para la implementación de las comunicaciones industriales actuales y sus protocolos, en sus diferentes aspectos de seguridad y la calidad del servicio.
- UG482, *7 Series FPGAs GTP Transceiver User Guide*
  El transceptor GTP FPGAs serie 7 es un transceptor de bajo consumo de energía, que admite velocidades de línea entre 500 Mb/s y 6.6 Gb/s. El transceptor GTP es altamente configurable y está estrechamente integrado con los recursos lógicos programables del FPGA.
- *PG054, 7 Series FPGAs Integrated Block for PCI Express LogiCORE IP Product Guide*
  El bloque integrado FPGA de la serie 7 para el núcleo PCI Express® es un bloque de construcción de interconexión en serie escalable, de alto ancho de banda y confiable para usar con Xilinx® Zynq®-7000 SoC y familias de FPGA de la serie 7. La solución 7 Series Integrated Block para PCI Express (PCIe®) admite configuraciones de punto final y puerto raíz de 1, 2 o 4 líneas y 8 líneas a velocidades de hasta 5 Gb/s (Gen2), todo lo cual cumplen con la especificación de base PCI Express, rev. 2.1. Esta solución es compatible con la interfaz AMBA® AXI4-Stream para la interfaz

de usuario del cliente. Con un mayor ancho de banda por pin, baja carga, baja latencia, problemas de integridad de señal reducida y arquitectura CDR, el bloque integrado de la serie 7 para PCIe establece el estándar de la industria para un Solución de E/S de alto rendimiento, rentable y de tercera generación.

- *UG479, 7 Series FPGAs DSP48E1 User Guide*
  Los FPGA son eficientes para aplicaciones de procesamiento de señal digital (DSP) porque pueden implementar algoritmos personalizados totalmente paralelos. Las aplicaciones DSP utilizan muchos multiplicadores y acumuladores binarios que se implementan mejor en segmentos DSP dedicados. Todos los FPGA de la serie 7 tienen muchos cortes DSP dedicados, totalmente personalizados y de baja potencia, que combinan alta velocidad con pequeño tamaño mientras conservan la flexibilidad de diseño del sistema. Los segmentos DSP mejoran la velocidad y la eficiencia de muchas aplicaciones más allá del procesamiento de señal digital, como los cambiadores de bus dinámico amplio, generadores de direcciones de memoria, multiplexores de bus ancho y registros de E/S mapeados en memoria.
- *UG480, 7 Series FPGAs XADC User Guide*
  En esta guía de usuario proporciona una breve descripción de la funcionalidad XADC de FPGA de la serie Xilinx 7. El XADC está disponible en todos los dispositivos ArtCo®-7, Kintex®-7, Virtex®-7 y Zynq®-7000 SoC. El XADC también está disponible en muchos, pero no en todos los dispositivos Spartan®-7. Para identificar dispositivos específicos que admitan el bloque XADC, consulte la descripción general de la familia Spartan-7 en DS180, Descripción general de los FPGA de la serie 7. El XADC es el bloque de construcción básico que permite la funcionalidad de señal mixta analógica (AMS) que es nuevo en 7 Serie de FPGAs. Al combinar bloques analógicos de alta calidad con la flexibilidad de la lógica programable, es posible crear interfaces analógicas personalizados para una amplia gama de aplicaciones.

El acoplamiento entre las partes PS y PL existen más de 3000 interconexiones de forma ajustada y flexibles utilizando múltiples interfaces. Esto permite la integración de aceleradores de hardware en la transmisión de datos por el usuario y otras funciones en la Lógica Programable de Xilinx, que son accesibles para los procesadores y también se puede acceder a los recursos de memoria en el sistema de procesamiento.

Los periféricos de E/S del PS y las interfaces de memoria estática/flash, comparten unas E/S multiplexada (MIO), hasta 54 pines.
Los dispositivos SoC Zynq-7000 incluyen la capacidad de usar las E/S que forman parte del dominio PL para muchos de los periféricos de E/S de PS.
Esto se realiza a través de una interfaz de E/S multiplexada extendida (EMIO).
La parte de la PL debe estar encendido para usar funciones de seguridad, prueba y depuración. Por el cual la desencriptación AES y los bloques de autenticación SHA como funciones de seguridad de la PL de Xilinx,

Glosario

Application Programming Interface (API). Es un conjunto de rutinas que provee acceso a funciones de un determinado software.

Bare-Metal.- Es un concepto que hace referencia a una máquina desnuda, que significa cuando no hay un kernel o sistema operativo instalado en el hardware. Suelen ser sistemas sencillos.

Board Support Package (BSP).- Es un paquete de soporte de placa (BSP) es una colección de controladores personalizados según la descripción de hardware proporcionada. Cada aplicación debe estar asociada a un BSP. Pueden existir varios BSP en un espacio de trabajo y admitir una única descripción de hardware.

BSCAN.- Es un bloque interno **MDM (**Microblaze Debug Module**)** Basado en escaneo limitado (Based on Boundary Scan).

BSP (Board Support Package).- El Paquete de soporte de placa es una colección de controladores personalizados, según la descripción de hardware proporcionada. Cada aplicación debe estar asociada a un BSP. Pueden existir varios BSP en un espacio de trabajo y admitir una única descripción de hardware.

First Stage Bootloader (FSBL).- Cargador de arranque de primera etapa.

JTAG (Joint Test Action Group).- Norma de la IEEE 1149.1 titulada **Standard Test Access Port and Boundary-Scan Architecture** para probar los puertos de acceso de un PCBs.

MMU (Memory Management Unit).- Unidad controladora de memoria**.**

OCM (On-Chip Memory).- Memoria en Chip es un módulo interno del Sistema de procesamiento (PS).

PCB (Printed Circuit Board).- Placa del circuito impreso.

PS (Processing System).- Sistema de Procesamiento.

SCU (Snoop control unit).- Unidad de control de monitoreo de forma furtivo (snoop fisgoneo).

UART(Universal Asynchronous Receiver-Transmitter).- Se usa para el control de los dispositivos seriales.

XMTC (**Xilinx MicroBlaze Trace Core**)**.-** Señales de interfaz del núcleo de seguimiento de MicroBlaze

BIBLIOGRAFÍA

[1] IEEE SPECTRUM (30 Jun 2017 | 17:00 GMT) Chip Hall of Fame: Xilinx XC2064 FPGA https://spectrum.ieee.org/tech-history/silicon-revolution/chip-hall-of-fame-xilinx-xc2064-fpga. 11marzo 2019.

[2] FUNDINGUNIVERSE, http://www.fundinguniverse.com/company-histories/xilinx-inc-history/ ,Xilinx, Inc. History 09 marzo 2019.

[3] XILINX https://www.xilinx.com/products/silicon-devices/fpga.html , FPGA Leadership across Multiple Process Nodes, 10 marzo 2019

[4] XILINX https://www.xilinx.com/publications/prod_mktg/zynq-7000-kit-product-brief.pdf, ZYNQ-7000 EPP ZC702 EVALUATION KIT, 10 marzo 2019

[5] XILINX https://www.xilinx.com/products/design-tools/microblaze.html, MicroBlaze Soft Processor Core, 11 marzo 2019

[6] Crockett H L., A. Elliot R, A. Enderwitz M., W. Stewart R. (Julio2014) The Zynq Book Embedded Processing with the ARM® Cortex®-A9 on the Xilinx® Zynq®-7000 All Programmable SoC First Edition, Strathclyde Academic Media, 2014. University of StrathclydeGlasgow, Scotland, UK

[7] researchgate.net https://www.researchgate.net/profile/Juergen_Jasperneite/publication/270160091_An_Architectural_Approach_for_Reconfigurable_Industrial_IO_Devices/links/54b39be60cf28ebe92e2eed2/An-Architectural-Approach-for-Reconfigurable-Industrial-I-O-Devices.pdf , An Architectural Approach forReconfigurable Industrial I/O Devices, 10 marzo 2019.

[8] E. Monmasson and M. Cirstea, "FPGA design methodology for industrial control systems – a review," IEEE Trans. on Industrial Electronics, vol. 54, no. 4, pp. 1824–1842, Aug 2007.

[9] J. O. Hamblen and T. S. Hall, "Using system-on-a-programmable-chip technology to design embedded systems," Int. Journal of Computers and Their Applications, vol. 13, no. 3, pp. 142–152, 2006.

[10] H. Flatt, J. Jasperneite, D. Dennstedt, and T. Hung, "Mapping of PRP/HSR redundancy protocols onto a configurable FPGA/CPU based architecture," in Int. Conf. on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS XIII). IEEE, 2013, pp. 121–128.

[11] Xilinx. https://www.xilinx.com/products/design-tools/software-zone/embedded-computing.html#os (2018). Embedded Development. 2019, de XILINX Sitio web: Embedded Development

[12] Literature from PI . (Version October 2014). PROFINET System Description Technology and Application. PROFIBUS Nutzerorganisation e. V. (PNO) : Haid-und-Neu-Str. 7 · 76131 Karlsruhe · Germany .

[13] https://www.xilinx.com/html_docs/xilinx2018_1/SDK_Doc/SDK_concepts/concept_fpgabitstream.html.

[14] EDK Concepts, Tools, and Techniques, A Hands-On Guide to Effective Embedded System Design, UG683 (v13.2) July 6, 2011, Xilinx.

[15] Zynq-7000 All Programmable SoC: Concepts, Tools, and Techniques (CTT), UG873 (v14.2) July 27, 2012, Xilinx.
[16] Bootgen User Guide, UG1283 (v2018.2) September 28, 2018, Xilinx.
[17] 7 Series FPGAs SelectIO Resources User Guide, UG471 (v1.10) May 8, 2018, Xilinx
[18] 7 Series FPGAs Clocking ResourcesUser Guide UG472 (v1.14) July 30, 2018, Xilinx
[19] 7 Series FPGAs Memory Resources User Guide, UG473 (v1.14) July 3, 2019, Xilinx
[20] 7 Series FPGAs Configurable Logic BlockUser Guide, UG474 (v1.8) September 27, 2016, Xilinx
[21] 7 Series FPGAs GTX/GTH Transceivers User Guide, UG476 (v1.12.1) August 14, 2018, Xilinx
[22] 7 Series FPGAs GTP Transceivers User Guide, UG482 (v1.9) December 19, 2016, Xilinx
[23] 7 Series FPGAs Integrated Block for PCI Express v3.3 LogiCORE IP Product Guide Vivado Design Suite, PG054 December 5, 2018, Xilinx
[24] 7 Series DSP48E1 Slice User Guide, UG479 (v1.10) March 27, 2018, Xilinx
[25] 7 Series FPGAs and Zynq-7000 SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter User Guide, UG480 (v1.10.1) July 23, 2018, Xilinx
[26] Zynq-7000 SoC Technical Reference, Manual UG585 (v1.12.2) July 1, 2018, Xilinx
[27] Simple AMP: Zynq SoC Cortex-A9 Bare-Metal System with MicroBlaze Processor, Author: John McDougall, XAPP1093 (v1.0.1) January 24, 2014, Xilinx
[28] PetaLinux Tools Documentation Reference Guide, UG1144 (v2016.2) June 8, 2016, Xilinx.
[29] GitHub, 2020, https://github.com/Xilinx/u-boot-xlnx, Repositorios.