# Introducción al uso del procesador LatticeMico32

## 1. Objetivo General

Obtener experiencia en el uso del procesador LatticeMico32, desde su programación hasta la implementación en una tarjeta de desarrollo con FPGA.

## 2. Objetivos Específicos

- Instalar las herramientas de compilación para el procesador LM32.
- Identificar las características básicas del procesador LM32.
- Realizar una aplicación con el procesador LM32.

## 3. Materiales y Herramientas Requeridas

- **1** tarjeta de desarrollo con FPGA.
- **1** computador con sistema operativo basado en el núcleo Linux.
- **1** cable serial.

## 4. Práctica

**Duración:** Para la realización de esta práctica se dispone de dos semanas.

### 4.1. Sistemas operativos basados en el núcleo Linux

En el desarrollo de este curso es **fuertemente recomendado** el uso de un sistema operativo basado en Linux. Existe un gran número de distribuciones, sin embargo, las distribuciones recomendadas son Linux Mint y Ubuntu. En todas los sistemas operativos basados en Linux es común el uso de la consola o Terminal. El uso de la consola tiene varias ventajas, como el tener mayor control sobre las instrucciones y la posibilidad de automatizar procedimientos rutinarios. Una guía para la instalación de Ubuntu esta disponible en:

`www.docentes.unal.edu.co/cepardot/docs/Ayudas/Como_Instalar_Ubuntu_1004_2.pdf`

**Actividad:** Busque la documentación (función y parámetros) de los siguientes comandos: man, sudo, apt-get, ls, cd, mkdir, wget, git, make, echo, grep.

### 4.1.1. Variables de entorno

Otro concepto importante para el trabajo con un SO basado en Linux son las variables de entorno. Una variable de entorno es un objeto designado para contener información usada por una o mas aplicaciones y afecta el comportamiento del SO. Para ver que información contiene una variable de entorno se debe hacer lo siguiente:

```
echo $PATH
```

El resultado que se debe obtener es algo de este estilo:

```
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
```

La variable de entorno $PATH lista los directorios donde el SO busca los archivos ejecutables. Esos directorios están separados por ':'

Para modificar las variables de entorno se usa el comando **export**. La forma de usarlo es la siguiente:

```
export http_proxy=http://usuario:contrasena@servidor:puerto
```

Para configurar la conexión con el servidor proxy de la Universidad, se tiene:

```
export http_proxy=http://usuario_unal:contrasena_unal@proxy.unal.edu.co:8080
```

Para verificar que el resultado ingresamos en la consola:

```
echo $http_proxy
```

La función **export** modifica las variables de entorno unicamente durante la sesión. Es por eso que para modificar durante todas las sesiones las variables de entorno se debe modificar el archivo ~/.bashrc *(El carácter ~ es un abreviación para llegar al directorio /home/user)* , escribiendo las mismas instrucciones que ingresaron en la terminal. Despues de modificado el archivo se ingresa en la consola:

source ~/.bashrc

**Actividad:** Qué información contiene la variable de entorno $SHELL? Que diferencia existe si en $SHELL esta almacenado /bin/bash o /bin/dash? Consulte cuatro variables de entorno para un SO basado en Linux y comente para que es usada.

### 4.1.2. Shell script

Con el uso de la consola ciertas tareas rutinarias pueden ser automatizadas. Esta tarea se realiza por medio de un Shell script, que no es mas que un archivo de texto plano con las instrucciones que ingresamos normalmente en consola. La extensión de un shell script es .sh y para ejecutar un script solo se debe ingresar en la consola:

```
sh nombre_del_archivo.sh
```

El siguiente es un ejemplo de un archivo que crea una carpeta y luego crea 100 archivos de texto plano.

```
#!/bin/bash
mkdir dir_prueba
cd dir_prueba
for ((  i = 1 ;  i <= 100;  i++  ))
do
        echo "Creando el archivo $i.txt"
        touch $i.txt
done
```

## 4.2. Instalación de Herramientas para desarrollo con HDL y FPGAs

Ingrese a http://wiki.linuxencaja.net/wiki/Instalación_de_Herramientas_para_desarrollo_con_HDL_y_FPGAs y siga los pasos para la instalación de las herramientas. Asegurese de revisar el directorio de instalación de Xilinx al momento de modificar la variable de entorno.

**Actividad:** Obtener la documentación de la tarjeta de desarrollo e instalar los drivers necesarios para poder grabar la FPGA desde el SO basado en Linux. Instalar la ultima versión de KiCAD (Usar la compilación oficial).

## 4.3. Cadena de herramientas *(toolchain)* para el procesador LM32

Ingrese a la pagina web http://wiki.linuxencaja.net/wiki/Academico/Digital2/ y siga los pasos para la instalación de las herramientas. En la sección *Instrucciones Para Compillar la Cadena de Herramientas para el Procesador LM32* cree un Shell Script.

## 4.4. Archivos del LM32 y el proceso para sintetizar

### 4.4.1. Directorios y ficheros

En la carpeta de trabajo SIE/SoftCore/lm32/sie estan los siguientes archivos:

**system.ucf** Almacena la información de asociación de pines físicos de la FPGA y las señales del procesador y sus periféricos.

**system.v** Archivo Verilog que tiene el diseño de alto nivel del procesador y sus periféricos. En este archivo se define la ruta del archivo .ram, que es el archivo resultado de la compilación del código para el procesador.

**system.xst** Archivo usado por Xilinx que tiene la referencia de la FPGA y opciones para sintetizar.

**system_config.v** Archivo Verilog que tiene parámetros de configuración del procesador.

**system_tb.v** Archivo Verilog usado para una simulación (test bench).

**Makefile** Es un archivo de texto plano necesario para sintetizar el procesador desde consola. Este archivo es usado por el comando make y permite automatizar el proceso.

En el mismo directorio encuentran las carpetas:

**Cores** Contiene tres periféricos descritos en Verilog.

**firmware** Contiene ejemplos para programar el procesador LM32.

**rtl** Contiene archivos Verilog con la descripción de hardware del procesador y sus periféricos.

**sim** Contiene archivos verilog para la simulación.

**tools** Herramientas de para pruebas desarrolladas en C y Ruby.

### 4.4.2. Proceso para sintetizar y compilar

Para sintetizar o simular el ejemplo se debe:

1. Ingresar a la consola y ubicarnos en SIE/SoftCore/lm32/sie.
2. Asegurarse de tener un archivo .ram compilado. Para eso se ingresa a la carpeta firmware y se realiza el make correspondiente.

3. Escribir en consola make syn (para sintetizar).
4. Escribir en consola make sim (para simular).
5. Escribir en consola make view (para ver los resultados de la simulación en gtkwave).

En la figura 1 se ilustra el flujo de diseño detallado (Camargo, 2012). En este diagrama se ve cada paso de la cadena de herramientas del procesador LM32 y se detalla las tareas de desarrollo de software y hardware. En la figura 2 se muestra el flujo de diseño simplificado para hacer uso del procesador LM32.

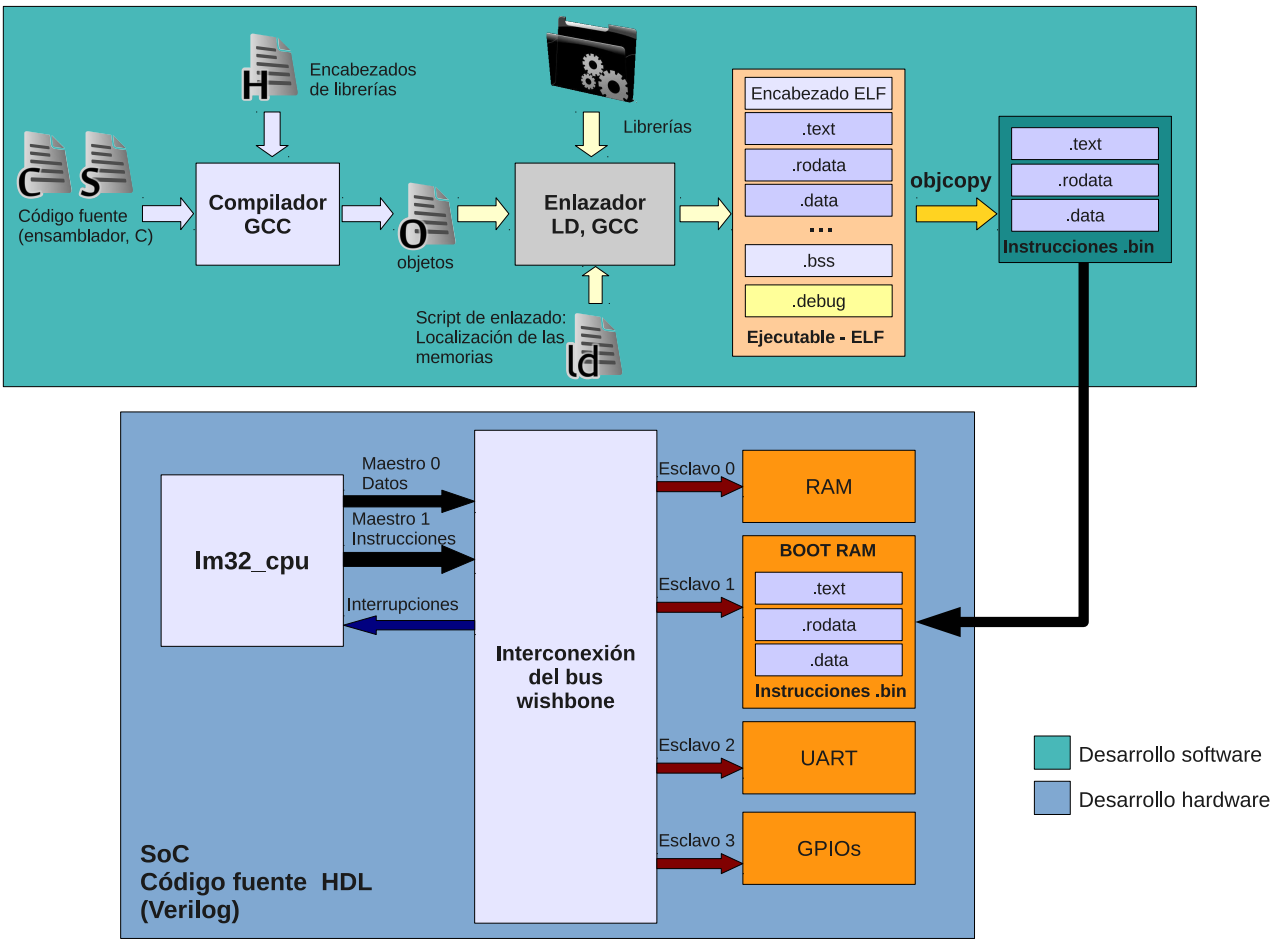


Figura 1: Flujo de diseño Hardware / Software al utilizar el procesador *softcore* LM32 (Camargo, 2012).

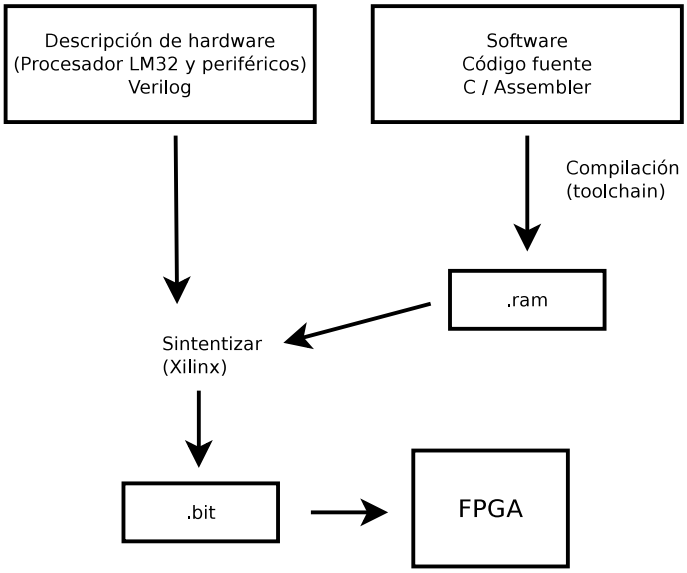


Figura 2: Flujo de diseño Hardware / Software reducido.

**Actividad:** ¿Cómo se realiza un archivo Makefile? Indique por medio de un diagrama los pasos que realiza el Makefile ubicado en *SIE/SoftCore/lm32/sie* y en *SIE/SoftCore/lm32/sie/firmware/boot0-serial*.

#### 4.4.3. RTL y camino de datos

Para identificar la arquitectura del procesador LM32 existen dos opciones:

- Usar ISE de Xilinx para ver el RTL (Register transfer Level).
  1. Estando ubicado en el directorio SIE/SoftCore/lm32/sie ejecute en consola make system.ngc
  2. Abrir ISE y crear un proyecto.
  3. Agregar un archivo sintetizable (Puede usar este rtl/wb_sram/wb_sram32.v).
  4. Sintetice este proyecto y luego de click en ver RTL.
  5. Ahora de clic en abrir y busque en la carpeta SIE/SoftCore/lm32/sie/build el archivo system.ngr
  6. Ese es el RTL del LM32.
- Revisando los archivos de Verilog buscando las señales por medio del comando **grep**.
  1. Ingresar a la consola en el directorio SIE/SoftCore/lm32/sie/.
  2. Ingresar en la consola el comando **grep** de la siguiente forma:

```
grep -IR palabra_que_busca *
```

**Actividad:** Realice un diagrama del camino de datos del procesador. Según el diagrama que obtiene y la documentación disponible que puede afirmar de la arquitectura.

## 4.5. Ejemplo con el LM32

El ejemplo de aplicación esta disponible en el directorio *SIE/Softcore/lm32/sie/firmware/boot0-serial*. En esta carpeta encontramos los archivos:

- crt0ram.S
- image.ram
- linker.ld
- main.c
- Makefile
- soc-hw.c
- soc-hw.h

En la figura 3 aparece el diagrama de hardware simplificado.

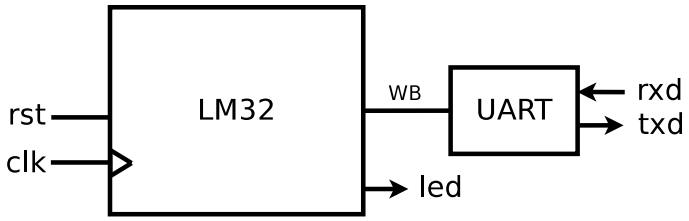


Figura 3: Diagrama de hardware del ejemplo boot0-serial

Este ejemplo implementa una comunicación serial a 57600 baudios. El led indica si el procesador LM32 esta trasmitiendo información.

**Actividad:** Leer los archivos de código fuente ubicados en la carpeta *SIE/SoftCore/lm32/sie/firmware/boot0-serial*.

## 4.6. Aplicación con el LM32

Basados en el ejemplo boot0-serial, se debe modificar el código fuente para recibir dos números, sumarlos y transmitir el resultado por el puerto serial. La figura 4 muestra un diagrama de la aplicación que debe ser implementada en una tarjeta de desarrollo.

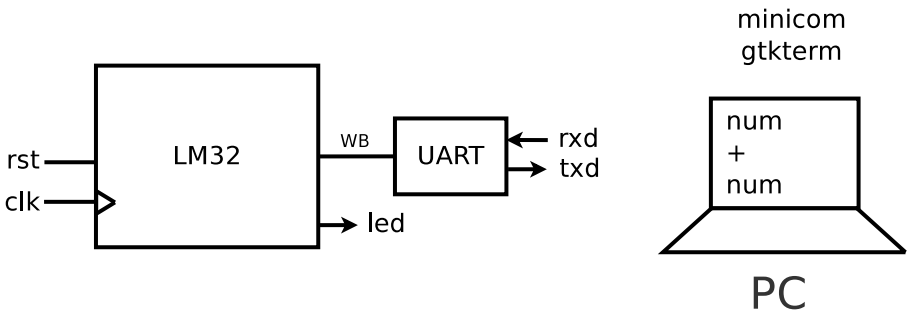


Figura 4: Aplicación conel procesador LM32

#### 4.6.1. Especificaciones de diseño

La aplicación debe cumplir con las siguientes especificaciones de diseño:

- Sumador de dos números de 32 bits.
- Una vez realizada la operación el usuario debe poder realizar otra suma.
- La captura e impresión de los datos debe ser en formado decimal. Implementaciones en código ASCII o en codificación Hex. no serán recibidas.
- La captura de los datos debe ser de la siguiente forma: `Número_A [Carácter +] Número_B [Enter]`.
- La captura debe detectar cuando se ingresa un numero o un carácter y tener control de errores (mensajes de alerta al usuario, reinicializar el programa).

#### 4.6.2. Bono

La aplicación debe tener las operaciones de suma, resta, multiplicación y división. La captura de los datos debe ser de la siguiente forma: `Número_A [Carácter + , - , x , /] Número_B [Enter]`. Las operaciones deben ser definidas para números enteros positivos, sin embargo debe tener control de errores como división por cero o el mal ingreso de datos.

## 5. Evaluación

En la evaluación serán considerados los siguientes elementos:

**25 %** Asistencia por medio de un quiz.

**20 %** Trabajo en el laboratorio: funcionamiento de la aplicación, instalación de las herramientas, desarrollo de las actividades.

**30 %** Informe: El informe debe estar presentado en IEEE elaborado en LaTeX con las siguientes secciones:

- Título.
- Abstract.
- Index terms.
- Introducción.
- *Estructura definida por el estudiante.*
- Conclusiones.
- Referencias.

En el informe se debe incluir el desarrollo de la guía desde el punto 4.4. En la actividad relacionada con el camino de datos se debe hacer un análisis de la arquitectura e incluir un diagrama simplificado del procesador. Se debe anexar el codigo fuente de la sección 4.6

**25 %** Sustentación.

## Referencias


- Camargo, Carlos. Plan de Estudios Para la Enseñanza / Aprendizaje de Sistemas Embebidos. 2012. Disponible en: http://wiki.linuxencaja.net/images/c/c7/CDIO.pdf
- G, Mike. Programación en BASH. Disponible en *http://linux-cd.com.ar/manuales/howtos/programacion-bash/Bash-Prog-Intro-COMO.html*
- Harris, David & Harriss, Sarah. "Digital desing and computer architecture". Prentice Hall. 2003.
- Instrucciones de instalación de la cadena de herramientas del procesador LM32. Disponible en: www.linuxencaja.net
- Instrucciones de instalación de herramientas para desarrollo con HDL y FPGAs. Disponible en: www.linuxencaja.net
- LatticeMico32 Processor Reference Manual. Disponible en http://www.latticesemi.com (requiere registrarse).
- Manuales Gentoo Linux/x86. Variables de entorno. Disponible en http://www.gentoo.org/doc/es/handbook/handbook-x86.xml?style=printable&part=2&chap=5
- Núñez, Antonio. Tutorial Verilog. Disponible en: http://www.iuma.ulpgc.es/~nunez/clases-FdC/verilog/Verilog%20Tutorial%20v1.pdf
- Ospina, Francisco. Cómo instalar Ubuntu 10.04 (Lucid Lynx). Julio 2010. Disponible en: www.docentes.unal.edu.co/cepardot/docs/Ayudas/Como_Instalar_Ubuntu_1004_2.pdf
- Smith, Bob. FPGAs at the Command Line. Disponible en: http://www.demandperipherals.com/docs/CmdLineFPGA.pdf
- Variable de entorno [en línea]. Wikipedia, La enciclopedia libre. Disponible en http://es.wikipedia.org/w/index.php?title=Variable_de_entorno&oldid=52409104.
- XilinxISE. Ubuntu documentation. Disponible en https://help.ubuntu.com/community/XilinxISE