Instituto Politécnico Nacional

Centro de Investigación en Computación

Secretaría de Investigación y Posgrado

Metodología para el desarrollo de funciones hardware para bibliotecas estáticas

T E S I S

QUE PARA OBTENER EL GRADO DE
MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A

EL ING. ALEJANDRO GOMEZ CONDE

DIRECTORES DE TESIS:

Dr. MARCO A. RAMÍREZ SALINAS

Dr. HUGO CESAR COYOTE ESTRADA

MÉXICO, D.F. **JUNIO 2012**

Resumen

Sistemas Empotrados. (en español) Un sistema empotrado es el conjunto de elementos hardware y software necesarios para solucionar un problema específico. . . .

Abstract

Embedded Systems. (Using English) An Embedded system is a set of individual components that are tied together to ...

Summary

This thesis has the following organization.

Chapter 1 begins with ...

Chapter 2 has ....

Chapter 3 describes ...

Chapter 4 presents ...

Chapter 5 Collect all results to ....

Thanks

I deeply appreciate the support received from CIC-IPN ...

To my family ...

Acrónimos, abreviaturas y siglas

En éste apartado se listan los acrónimos, abreviaturas y siglas usadas en éste texto. Se incluye una descripción breve, la respectiva traducción en los casos que así lo requieren y la página donde aparece citada por primera vez a fin de que el lector pueda conocer el contexto en el que fue enunciada.

ABEL Advanced Boolean Expression Language
(*Lenguaje avanzado de expresiones boleanas*), página 7

AHDL Altera HDL, página 7

FPGA Field Programable Gate Array, página 1

Hard Processor *Circuito integrado que contiene un procesador*, página 7

HDL Hardware Description Language
(*Lenguaje de descripción hardware*) , página 7

Soft Processor *Procesador sintetizado en un dispositivo reconfigurable*, página 7

Verilog Lenguaje de descripción hardware, página 7

VHDL VHSIC Hardware Description Language
(*Lenguaje de descripción hardware para circuitos integrados de muy alta velocidad*), página 7

Índice general

Índice de figuras

Índice de tablas

Índice de listados

Acrónimos, abreviaturas y siglas

En éste apartado se listan los acrónimos, abreviaturas y siglas usadas en éste texto. Se incluye una descripción breve, la respectiva traducción en los casos que así lo requieren y la página donde aparece citada por primera vez a fin de que el lector pueda conocer el contexto en el que fue enunciada.

ABEL Advanced Boolean Expression Language
(*Lenguaje avanzado de expresiones boleanas*), página 7

AHDL Altera HDL, página 7

FPGA Field Programable Gate Array, página 1

Hard Processor *Circuito integrado que contiene un procesador*, página 7

HDL Hardware Description Language
(*Lenguaje de descripción hardware*) , página 7

Soft Processor *Procesador sintetizado en un dispositivo reconfigurable*, página 7

Verilog Lenguaje de descripción hardware, página 7

VHDL VHSIC Hardware Description Language
(*Lenguaje de descripción hardware para circuitos integrados de muy alta velocidad*), página 7

Capítulo 1

Introducción

El actual entorno global en el ámbito de la tecnología insta a los desarrolladores a ...

1.1. Definiciones

Un sistema empotrado (S.E.) es ...

1.2. Antecedentes

El resultado de combinar los recursos hardware FPGA y software ...

La figura 1.1 muestra un esquema simplificado de una plataforma hardware usando la topología de bus.

1.3. Planteamiento del problema

Esta tesis pretende ...

Para ello es necesario implementar funciones hardware como las mostradas en el siguiente listado.

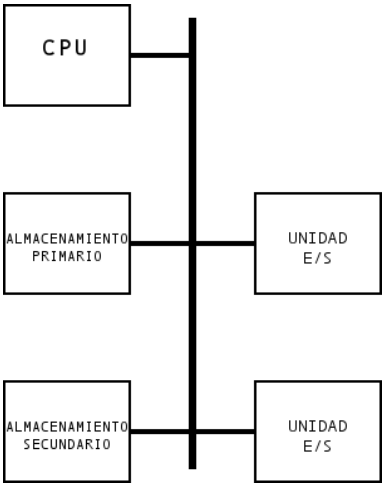

Figura 1.1: Esquema estructural de una plataforma hardware.

Listado 1.1: Ejemplo de operación para el modelo de ejecución de una función hardware con paso de argumentos por valor.

```
1  #include <stdio.h>
2  ...
3  #include "hard_func.h"
4
5  //Def. del prototipo
6  int HW_FUN (int A, int B, char op);
7
8  int main() {
9  ...
10 result = HW_FUN(x, y, op);
11 ...
12 }
```

El segundo modelo se aplica a ...

El listado 1.2 muestra un ejemplo de este tipo de funciones.

Listado 1.2: Ejemplo de operación para el modelo de ejecución de una función hardware con paso de argumentos por referencia.

```
1  #include <stdio.h>
2  ...
3  #include "hard_func.h"
4
5  // Def. del prototipo
6  struct *HW_FUN (struct *A, struct *B);
7
8  int main() {
9  ...
10 data = HW_FUN(data1_ptr, data2_ptr);
11 ...
12 }
```

La figura 1.2 muestra el esquema general de interacción entre hardware y software para el uso de funciones hardware.

1.4. Objetivos

1.4.1. Objetivo general

Metodología para el desarrollo de funciones hardware para bibliotecas estáticas.

1.4.2. Objetivos específicos

- Obj uno.
- Obj dos.
- Obj tres.
- Obj cuatro.

1.5. Justificación

El uso de sistemas ...

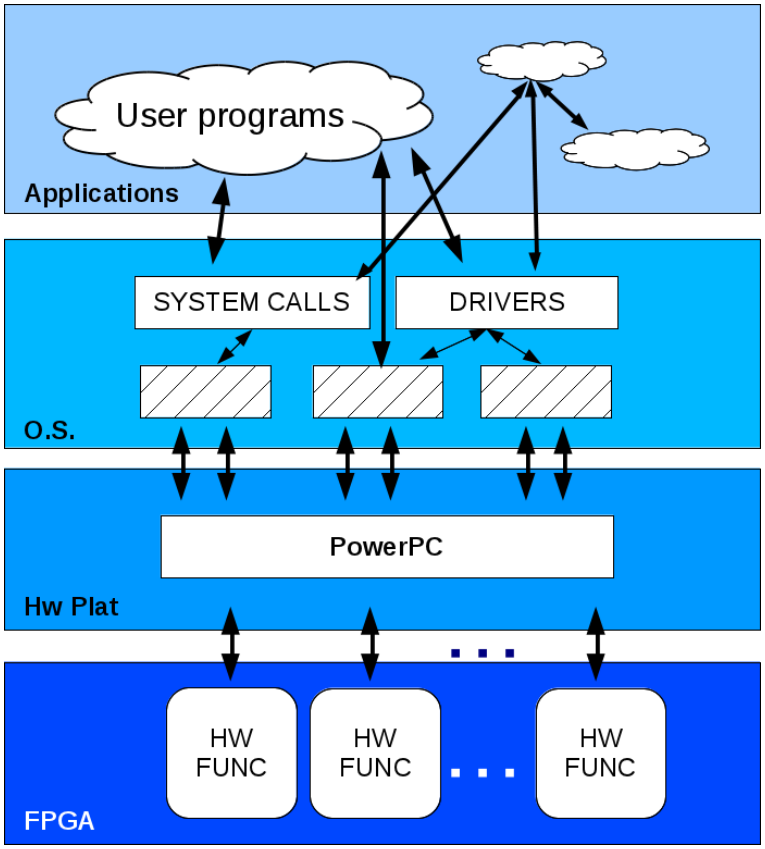

Figura 1.2: Esquema de operación hardware/software.

1.5.1. Beneficios esperados

Esta tesis pretende generar ...

1.5.2. Alcances y límites

En este trabajo se pretende ...pero sólo utiliza ...aunque es posible implementar ...

1.6. Organización de la tesis

Capítulo 1 inicia con una introducción a ...

Capítulo 2 contiene ...

Capítulo 3 describe ...

Capítulo 4 contiene...

Capítulo 5 contiene los resultados de ...

Capítulo 2

Fundamento teórico

Este capítulo presenta ...

2.1. Elemento teórico A

Un sistema A ...
Los elementos de ... son:

- Unidad de procesamiento
- Elemento de almacenamiento de datos
- Dispositivos de entrada y salida

La figura 2.1 muestra un esquema detallado con las características comúnmente observadas en estos sistemas. En esta figura se observa la unidad central de procesamiento, la memoria como elemento de almacenamiento y los periféricos como dispositivos de entrada/salida. También se puede observar el modelo híbrido para una arquitectura tipo Harvard dónde los buses para la memoria, las unidades de co-procesamiento, y los periféricos están separados.

2.1.1. Teoria A.i

Las unidades de procesamiento usadas en... estos lenguajes son usados para describir modelos de comportamiento y son mostrados en la tabla 2.1.

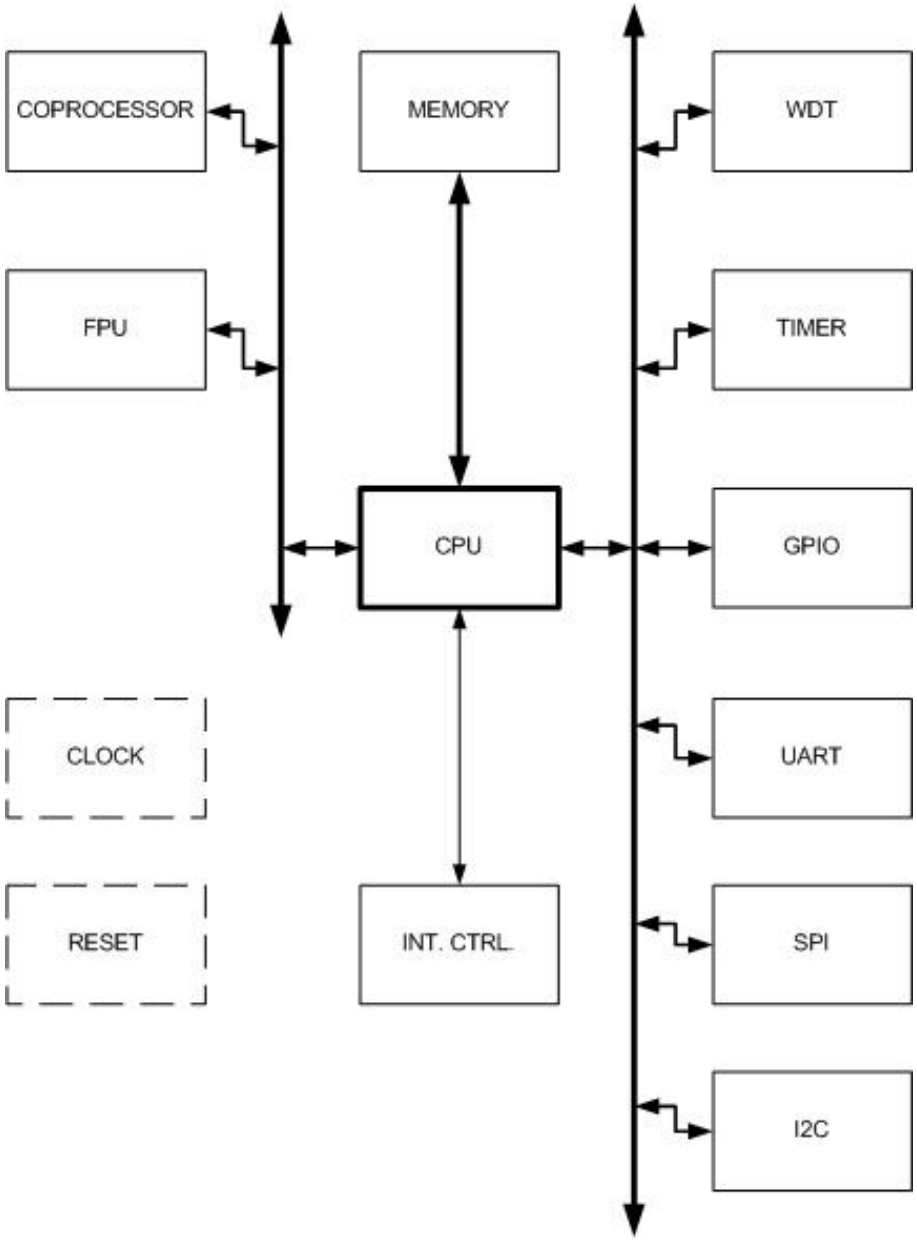

Figura 2.1: Esquema general de una plataforma hardware.

Nombre	Descripción
ABEL	Advanced Boolean Expression Language, adquirido por Xilinx
AHDL	Altera HDL. Lenguaje propietario desarrollado por Altera Corporation
Handel-C	HDL compatible con C
MyHDL	HDL basado en Python
SystemC	Conjunto estandarizado de bibliotecas de clases para C++ para diseño hardware
SystemVerilog	Un superconjunto de Verilog destinado a mejorar el proceso de diseño y verificación de sistemas
Verilog	HDL ampliamente utilizado en el desarrollo de sistemas
VHDL	VHSIC HDL

Tabla 2.1: Lenguajes HDL para descripción comportamental de sistemas.

Teoría A.i.1

Los procesadores ...

Texto énfasis. Este texto es un ejemplo para el contenido de un párrafo

Capítulo 3

Análisis y diseño

El diseño del sistema[3] ...

3.1. Generar α.

Pretende desarrollar una ...

3.1.1. Obtención de requerimientos

Los req....

3.1.2. Análisis de requerimientos

Se requiere ...

3.1.3. Diseño

Diseñar una

3.1.4. Implementación

La implementación de

3.2. Generación $\alpha 2$

Usar las herramientas de ...

3.2.1. Obtención de requerimientos

Configurar, integrar, compilar, generar e instalar ... de α ...

3.3. Desarrollo de β

Estos modelos ...

3.3.1. Análisis

Para ejemplificar el proceso de ...

3.3.2. Comunicación a nivel kernel para β

Esta sección diseña una ...

Capítulo 4

Pruebas y resultados

Las pruebas se realizaron para ...

4.1. Primera etapa

La primera etapa generó ...
El resultado obtenido en la consola serial fue:

```
-- Entering main() --

Running GpioOutputExample() for LEDs_8Bit...
GpioOutputExample PASSED.

Running GpioOutputExample() for LEDs_Positions...
GpioOutputExample PASSED.

Running GpioInputExample() for Push_Buttons_5Bit...
GpioInputExample PASSED. Read data:0x0

Running GpioInputExample() for DIP_Switches_8Bit...
GpioInputExample PASSED. Read data:0x2

-- Exiting main() --
```

4.2. Segunda etapa

La segunda etapa en el proceso ...

4.3. Tercera etapa

La tercera etapa genera ...

Capítulo 5

Conclusiones y trabajo futuro

En base a los resultados obtenidos en XX, YY y ZZ podemos concluir AA y BB pero no CC por tanto DD.

Bibliografía

[1] Aylor, James H. *et al.* 1996. *The codesign of embedded systems: a unified hardware/software representation* Primera edición. Kluwer Academic Publishers. AH Dordrecht. The Netherlands. ISBN 0-7923-9636-7. Capítulo 2. Páginas 11 y 12.

[2] Weiwu Hu. *et al.* 4 de Mayo del 2006. *Microarchitecture and Performance Analysis of Godson-2 SMT Processor.* IEEE. http://www.cecs.uci.edu/~papers/iccd2006/papers/paper_38.pdf

[3] Christopher Mims. 22 de Octubre de 2010. *Chinese Chip Closes in on Intel, AMD.* Technology Review. ACM. http://technews.acm.org/archives.cfm?searchterm=godson&fo=2010-10-oct/oct-22-2010.html

Apéndice A

Tarjeta ML507

Los elementos de la tarjeta de desarrollo ML507 que fueron utilizados en la implementación del S.E. se listan a continuación.

- FPGA XC5VFX70T
  - PowerPC 440
  - 11,200 Slices
- DDR2 SODIMM (256 MB)
- ZBT SRAM (1 MB)
- GTX transceivers
  - GTX: SFP (1000Base-X)
  - GTX: SMA (RX and TX differential pairs)
  - GTX: SGMII
  - GTX: PCIe™
  - GTX: SATA (dual host connections)
  - GTX clock synthesis chips
- Header for second serial port
- Soft touch port

Apéndice B

Hardware Platform Reference Design

This appendix shows step by step how to integrate a hardware platform for an embedded system using the ML507 Board.

Apéndice C

Complete source code

The listing C.1 show the C code for the memory-base test on the hardware platform.

Listado C.1: Prueba de la plataforma hardware base donde se ejercita el módulo DDR y el microprocesador principalmente.

```
1  #include "xparameters.h"
2  #include "xcache_l.h"
3  #include "stdio.h"
4  #include "xutil.h"
5  #include "xuartns550_l.h"
6
7  int main (void) {
8      XCache_EnableICache(0xc0000000);
9      XCache_EnableDCache(0xc0000000);
10
11     /* Initialize RS232_Uart_1 - Set baudrate and number of stop
           bits */
12     XUartNs550_SetBaud(XPAR_RS232_UART_1_BASEADDR,
           XPAR_XUARTNS550_CLOCK_HZ, 9600);
13     XUartNs550_mSetLineControlReg(XPAR_RS232_UART_1_BASEADDR,
           XUN_LCR_8_DATA_BITS);
14     print("-- Entering main() --\r\n");
15
16     /*
17      * MemoryTest routine will not be run for the memory at
18      * 0xffff0000 (xps_bram_if_cntlr_1)
```

```
19       * because it is being used to hold a part of this
             application program
20       */
21
22
23      /*
24       * MemoryTest routine will not be run for the memory at
25       * 0xfc000000 (FLASH)
26       * because it is a read-only memory
27       */
28
29
30      /* Testing Memory (DDR2_SDRAM)*/
31      {
32          XStatus status;
33
34          print("Starting MemoryTest for DDR2_SDRAM:\r\n");
35          print("  Running 32-bit test...");
36          status =
                XUtil_MemoryTest32((Xuint32*)XPAR_DDR2_SDRAM_MEM_BASEADDR,
                1024, 0xAAAA5555, XUT_ALLMEMTESTS);
37          if (status == XST_SUCCESS) {
38              print("PASSED!\r\n");
39          }
40          else {
41              print("FAILED!\r\n");
42          }
43          print("  Running 16-bit test...");
44          status =
                XUtil_MemoryTest16((Xuint16*)XPAR_DDR2_SDRAM_MEM_BASEADDR,
                2048, 0xAA55, XUT_ALLMEMTESTS);
45          if (status == XST_SUCCESS) {
46              print("PASSED!\r\n");
47          }
48          else {
49              print("FAILED!\r\n");
50          }
51          print("  Running 8-bit test...");
52          status =
                XUtil_MemoryTest8((Xuint8*)XPAR_DDR2_SDRAM_MEM_BASEADDR,
                4096, 0xA5, XUT_ALLMEMTESTS);
53          if (status == XST_SUCCESS) {
54              print("PASSED!\r\n");
55          }
56          else {
```

```
57              print("FAILED!\r\n");
58          }
59      }
60
61      print("--␣Exiting␣main()␣--\r\n");
62      XCache_DisableDCache();
63      XCache_DisableICache();
64      return 0;
65  }
```

The listing C.2 show the C code for the peripheral-based test on the hardware platform.

Listado C.2: Prueba de la plataforma hardware base donde se ejercitan los periféricos en especial los Leds y los Dip-Switches.

```
1 a
```

Apéndice D

Thesis recipes for Poky Framework

This appendix contains all the developed recipes that were used to generate a Dynamic Harware Execution Embeded System.

Listing ** shows a recipe for compiling a kernel module.

```
1  DESCRIPTION = "hello-world-mod tests the module.bbclass
       mechanism."
2  LICENSE = "GPLv2"
3  LIC_FILES_CHKSUM =
       "file://COPYING;md5=12f884d2ae1ff87c09e5b7ccc2c4ca7e"
4
5  inherit module
6
7  PR = r0
8  PV = "0.1"
9
10 SRC_URI = "file://Makefile \
11 ----------file://hello.c \
12 ----------file://hello.h \
13 ----------file://COPYING \
14 ----------"
15
16 S = "${WORKDIR}"
17
18 do_install() {
19         install -d ${D}
20         install -m 0644 hello.ko  ${D}
21 }
```