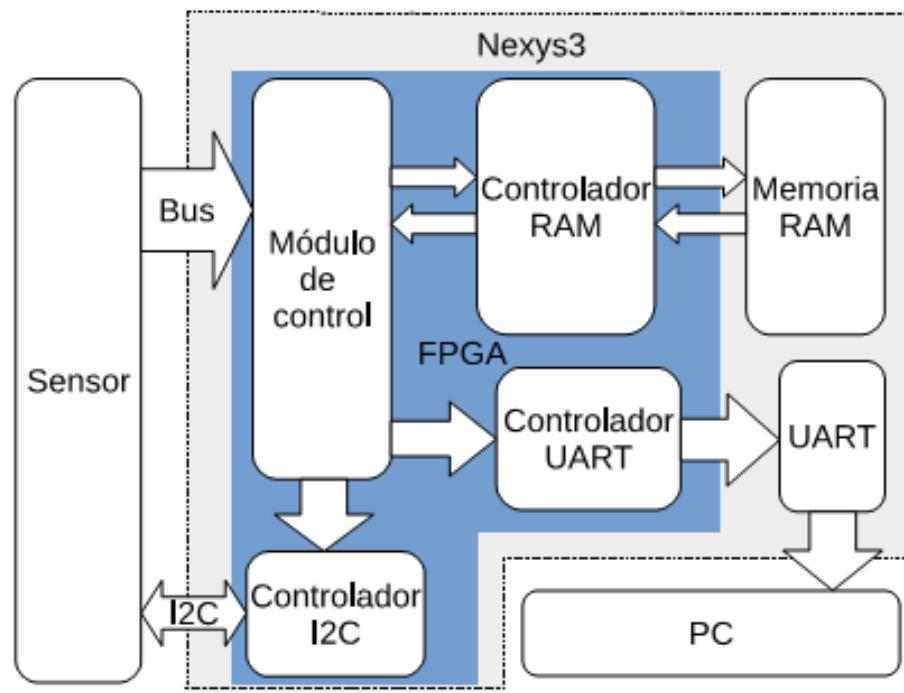
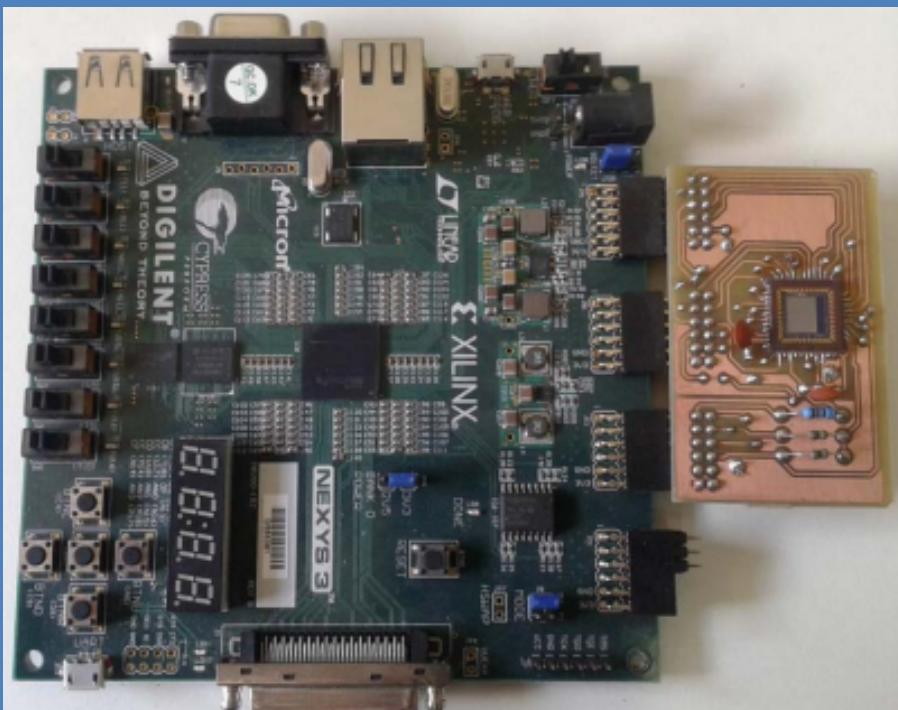


# Temas Específicos de Electrónica Digital I

Implementación de un sistema de  
comunicación USB 2.0 para aplicaciones  
científicas controladas por FPGA

Edwin Barragán  
2018

# Introducción



# Introducción

- Dotar al sistema de una comunicación con mayor ancho de banda
- Que la comunicación sea fácil de usar, accesible y robusta
- Que la comunicación sea trasladable
- Que la comunicación sea portable
- Solución propuesta: USB

# Agenda

- Objetivos
- Breve descripción del protocolo USB
- Arquitectura del sistema propuesto
- Componentes utilizados
  - CY3684 FX2LP EZ-USB DK de Cypress
  - MOJO v3
  - libusb-1.0

# Agenda

- Desarrollo
  - Configuración de la FX2LP
    - Explicación de las memorias FIFO configurables
    - Framework de Cypress
    - Firmware
    - Problemas presentados

# Agenda

- Desarrollo
  - Desarrollo del VHDL de la placa MOJO v3
    - Interfaz con la CY3684
    - Maquina de Estados
    - Test Bench's
    - Problemas presentados

# Agenda

- Desarrollo
  - Placa de conexión
    - Versión 1
    - Versión 2
    - Versión 3
- Problemas presentados del sistema en general
- Resultados y conclusiones

# Agenda

- Objetivos
- Breve descripción del protocolo USB
- Arquitectura del sistema propuesto
- Componentes utilizados
  - CY3684 FX2LP EZ-USB DK de Cypress
  - MOJO v3
  - libusb-1.0

# Objetivos

- Objetivo General
  - Realizar una efectiva comunicación entre una FPGA y una PC mediante el protocolo USB 2.0

# Objetivos

- Objetivos Particulares
  - Comprender el funcionamiento de la placa de desarrollo CY3684 y el framework provisto por Cypress
  - Realizar la Configuración del chip CY7C68014A
  - Sintetizar un circuito con VHDL que pueda manejar las memorias FIFO esclavas del integrado CY768014A

# Objetivos

- Objetivos Particulares
  - Diseñar un circuito impreso que permita la interconexión de las placas de desarrollo
  - Implementar un programa de computadoras que permita validar la conexión

# Agenda

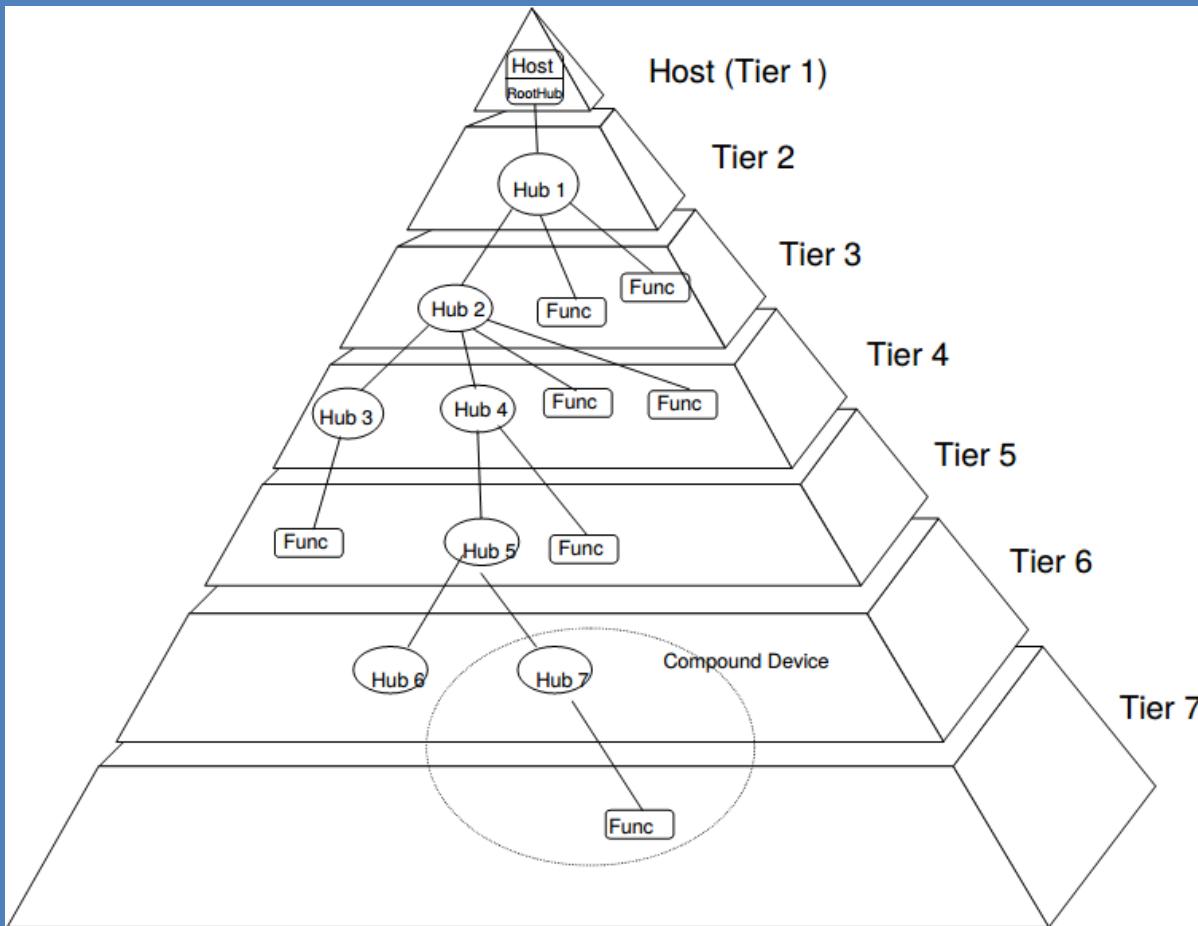
- Objetivos
- Breve descripción del protocolo USB
- Arquitectura del sistema propuesto
- Componentes utilizados
  - CY3684 FX2LP EZ-USB DK de Cypress
  - MOJO v3
  - libusb-1.0

# Norma USB revisión 2.0

- Objetivos de la norma
  - Conexión de teléfonos a la PC
  - Facilidad de uso
  - Proveer puerto de expansión

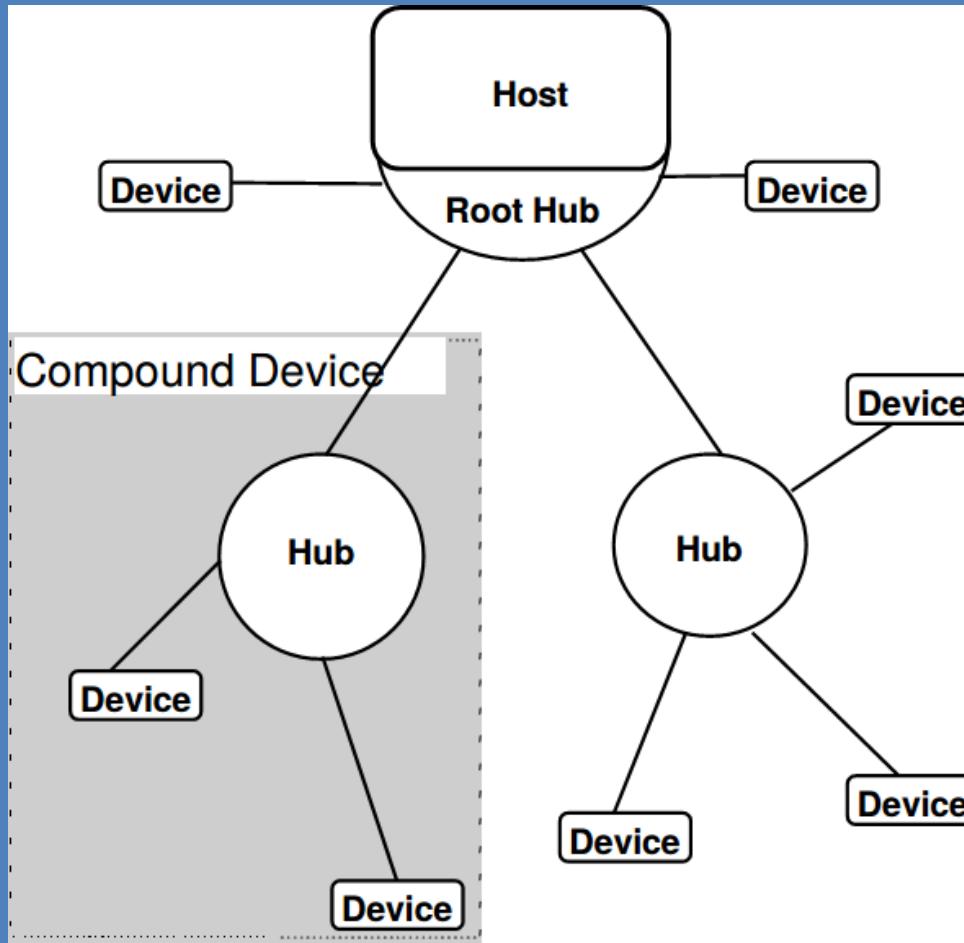
# Norma USB revisión 2.0

- Topología del USB



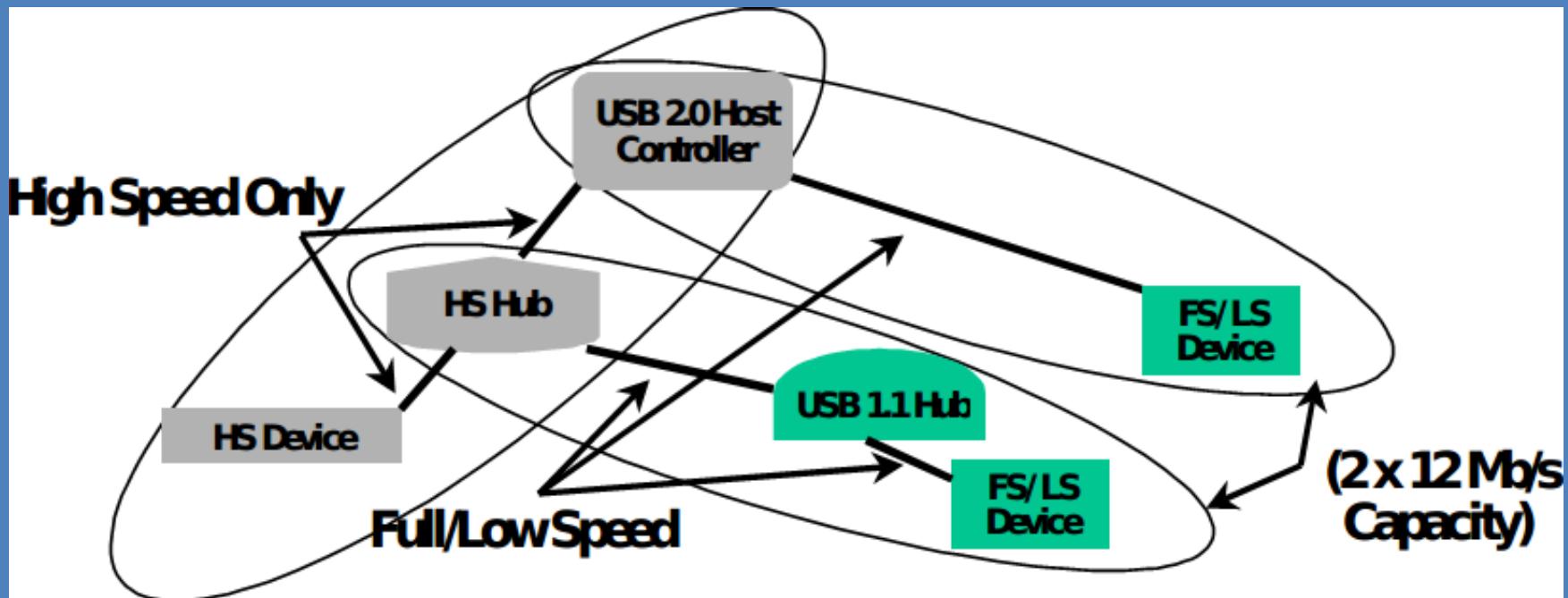
# Norma USB revisión 2.0

- Topología física del USB



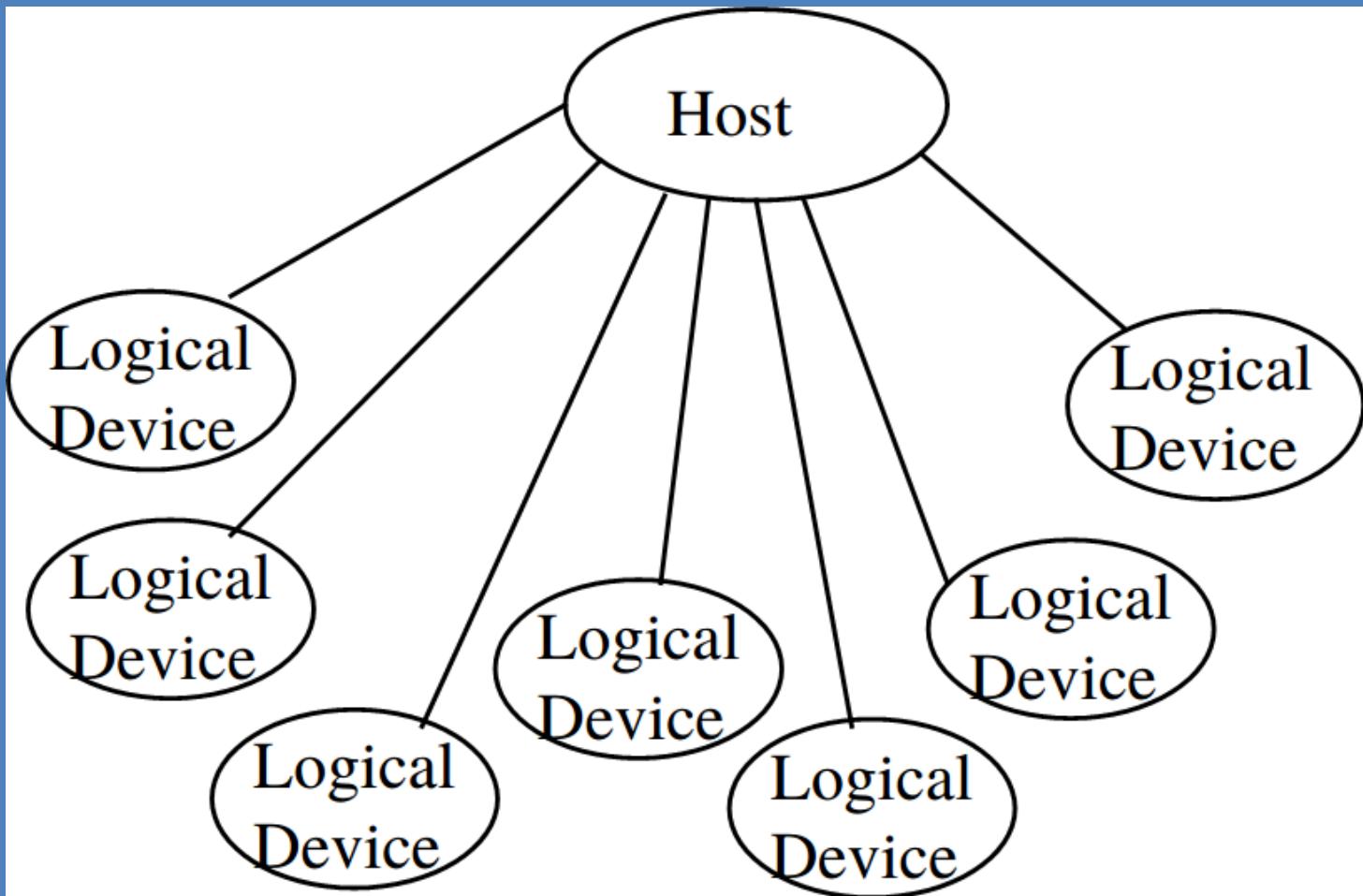
# Norma USB revisión 2.0

- Importancia del Hub



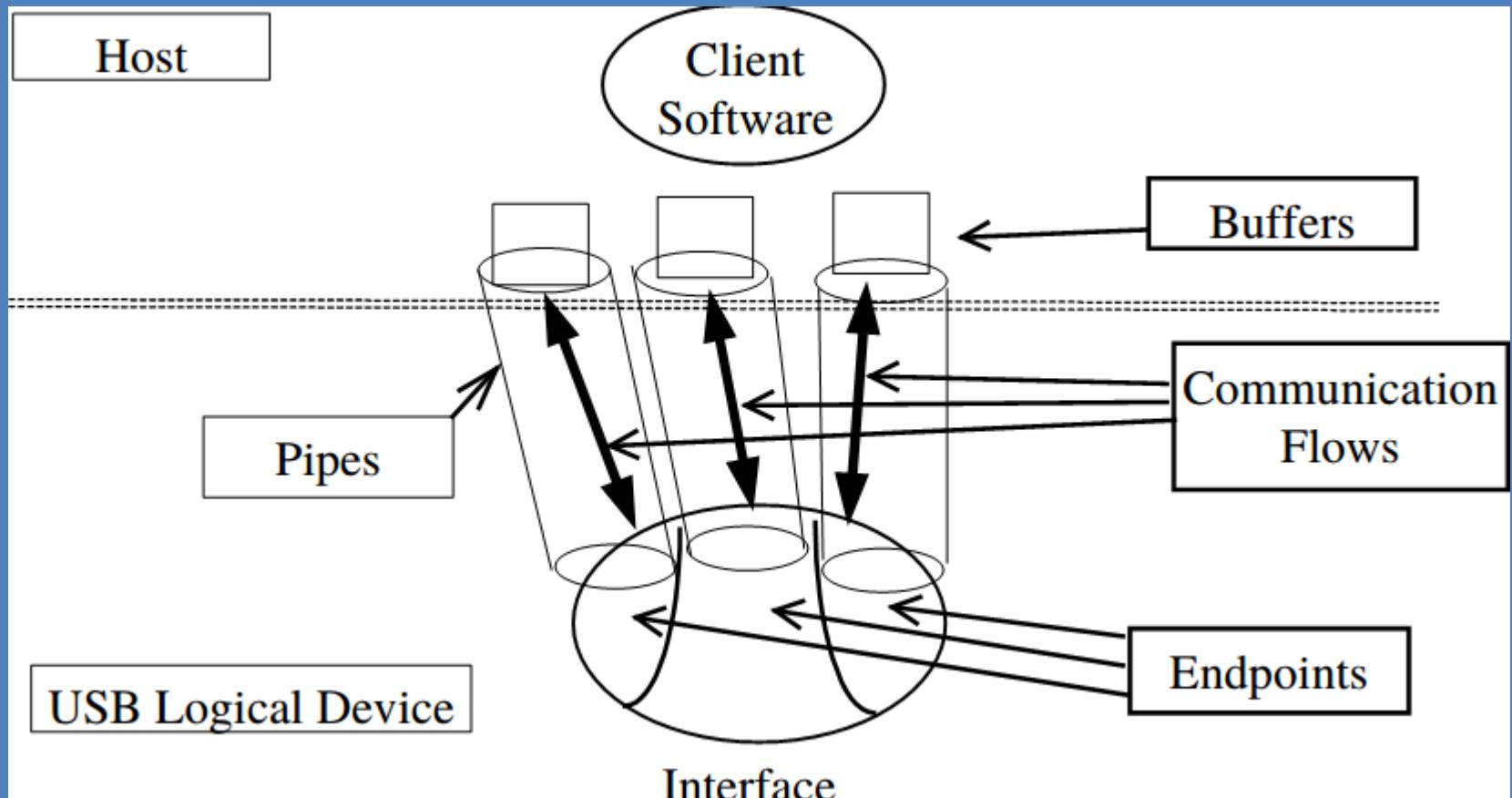
# Norma USB revisión 2.0

- Topología lógica del USB



# Norma USB revisión 2.0

- Flujo de datos



# Norma USB revisión 2.0

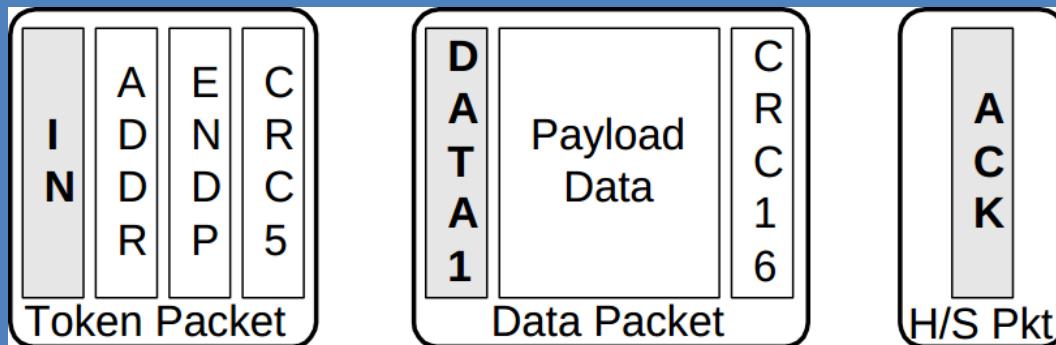
- Tipos de transferencia
  - Transferencia por bultos
  - Transferencia de Interrupciones
  - Transferencia Isocrónica
  - Transferencia de Control

# Norma USB revisión 2.0

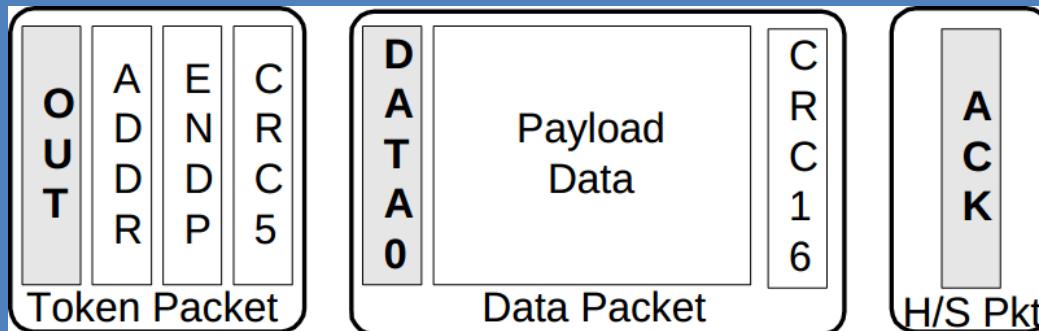
- Tipos de paquetes
  - Paquetes de token
  - Paquetes de datos
  - Paquetes de handshake
  - Paquetes especiales

# Norma USB revisión 2.0

- Entrada de datos



- Salida de datos

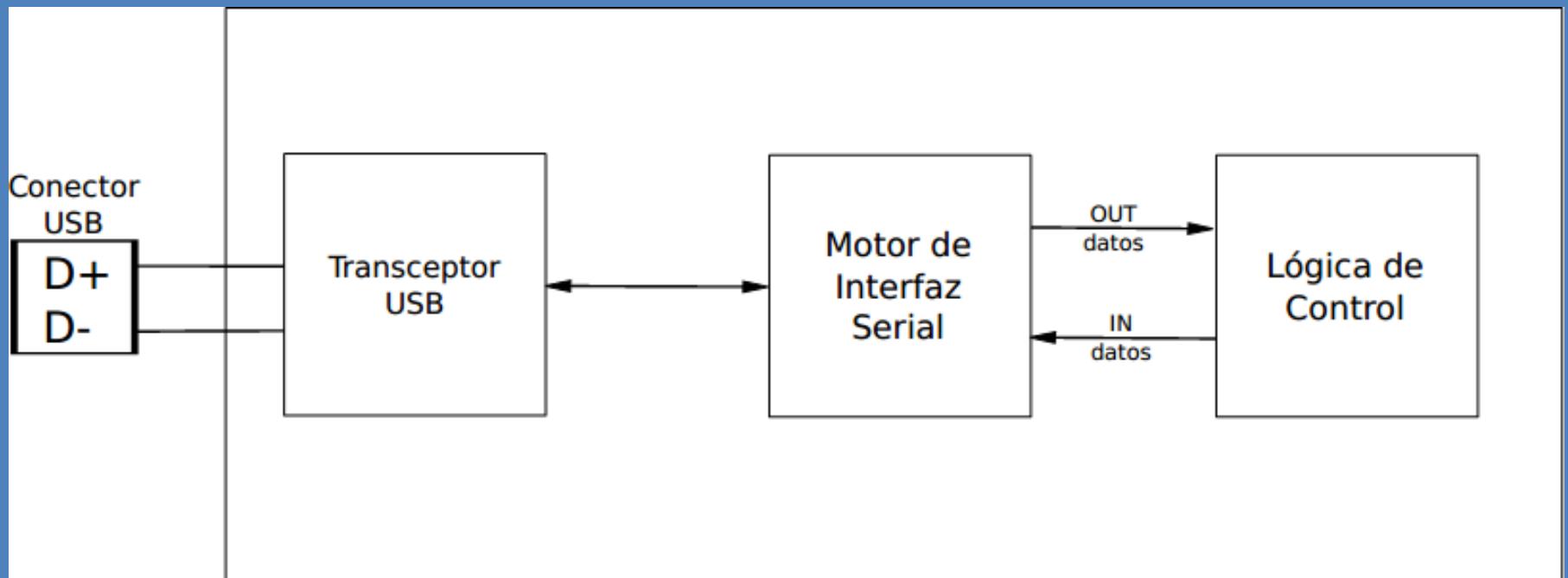


# Agenda

- Objetivos
- Breve descripción del protocolo USB
- Arquitectura del sistema propuesto
- Componentes utilizados
  - CY3684 FX2LP EZ-USB DK de Cypress
  - MOJO v3
  - libusb-1.0

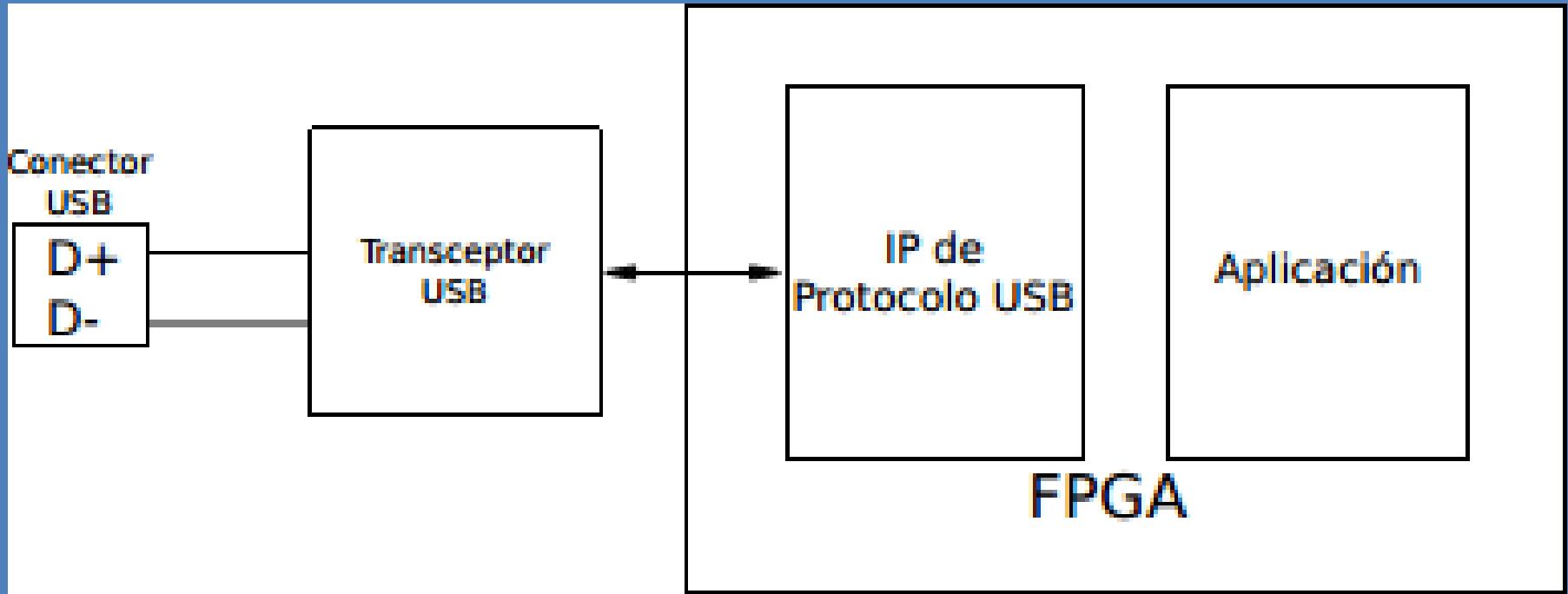
# Arquitectura del Sistema

- Requerimientos de un sistema que posee una comunicación USB



# Arquitectura del Sistema

- Hacer todo en FPGA y un transceptor externo

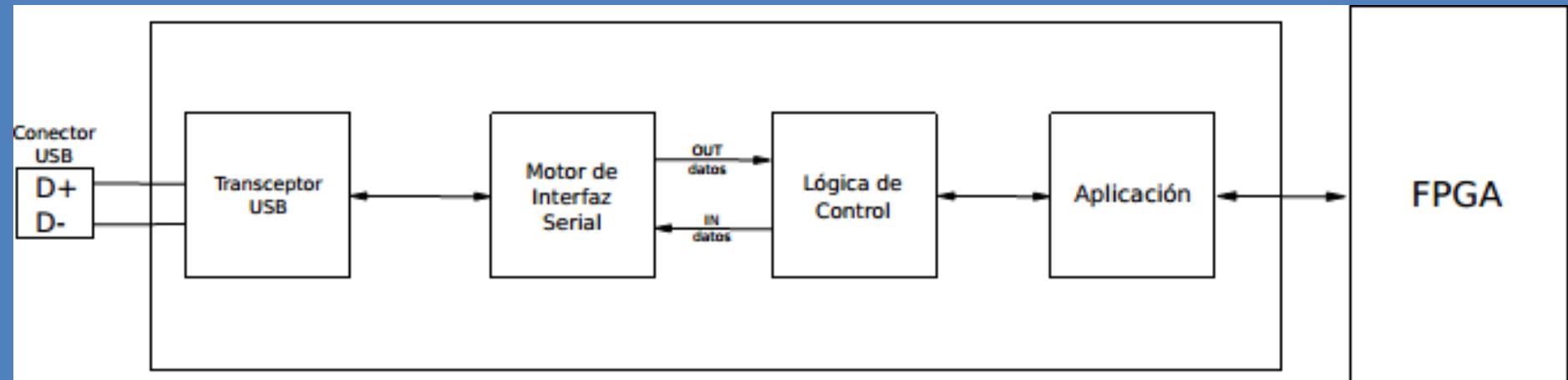


# Arquitectura del Sistema

- Hacer todo en FPGA y un transceptor externo
  - Pro
    - Requerimiento mínimo de hardware
  - Contra
    - Alto costo en desarrollo de una IP que realice esta tarea
    - Gran consumo de recursos de la FPGA misma

# Arquitectura del Sistema

- Tener un control completo en un integrado específico y usar el FPGA con una comunicación mínima



# Arquitectura del Sistema

- Tener un control completo en un integrado específico y usar el FPGA con una comunicación mínima
  - Pro
    - Una implementación rápida y más económica
    - Los recursos del FPGA liberados para el desarrollo de la aplicación específica
  - Contra
    - Espacio necesario en un impreso para incorporar el hardware extra

# Agenda

- Objetivos
- Breve descripción del protocolo USB
- Arquitectura del sistema propuesto
- Componentes utilizados
  - CY3684 FX2LP EZ-USB DK de Cypress
  - MOJO v3
  - libusb-1.0

# Arquitectura del Sistema



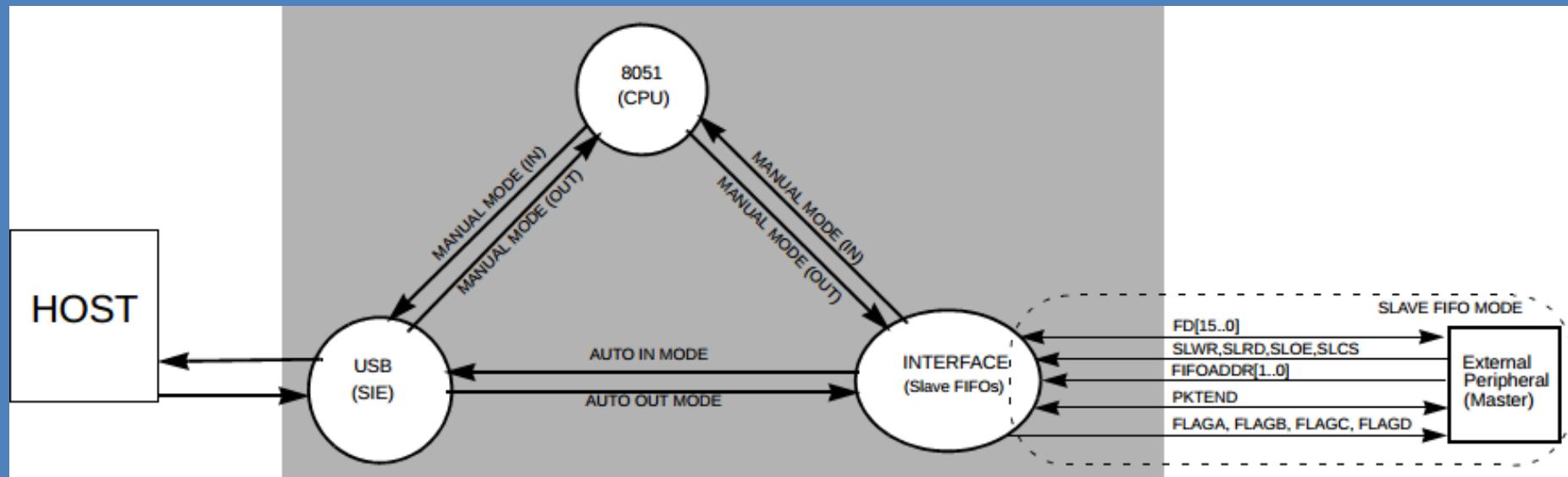
# CY3684 FX2LP EZ-Development Kit

## CY7C68013A

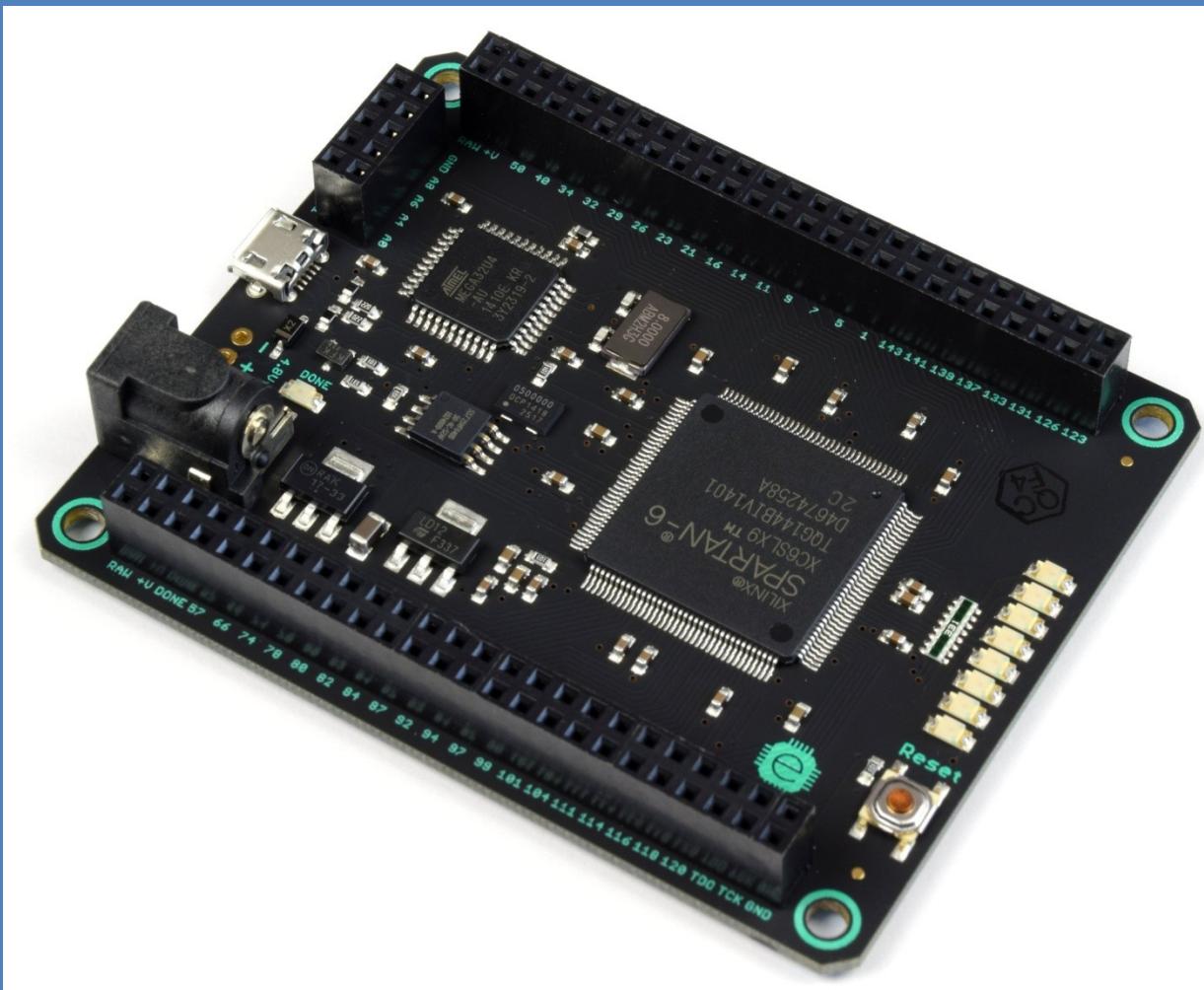


# CY3684 FX2LP EZ-Development Kit

## CY7C68013A



# MOJO v3



# MOJO v3

- FPGA Spartan 6 XC6SLX9 de Xilinx
- 84 pines IO digitales
- 8 entradas analógicas
- 8 LEDs de propósito general
- 1 botón pulsador de propósito general
- Regulador de voltaje de entrada de 4.8V - 12V
- ATmega32U4 que sirve para configurar la FPGA y leer los pines analógicos
- Bootloader compatible con Arduino
- Memoria flash para almacenar la configuración de la FPGA (programación persistente)

# libusb-1.0

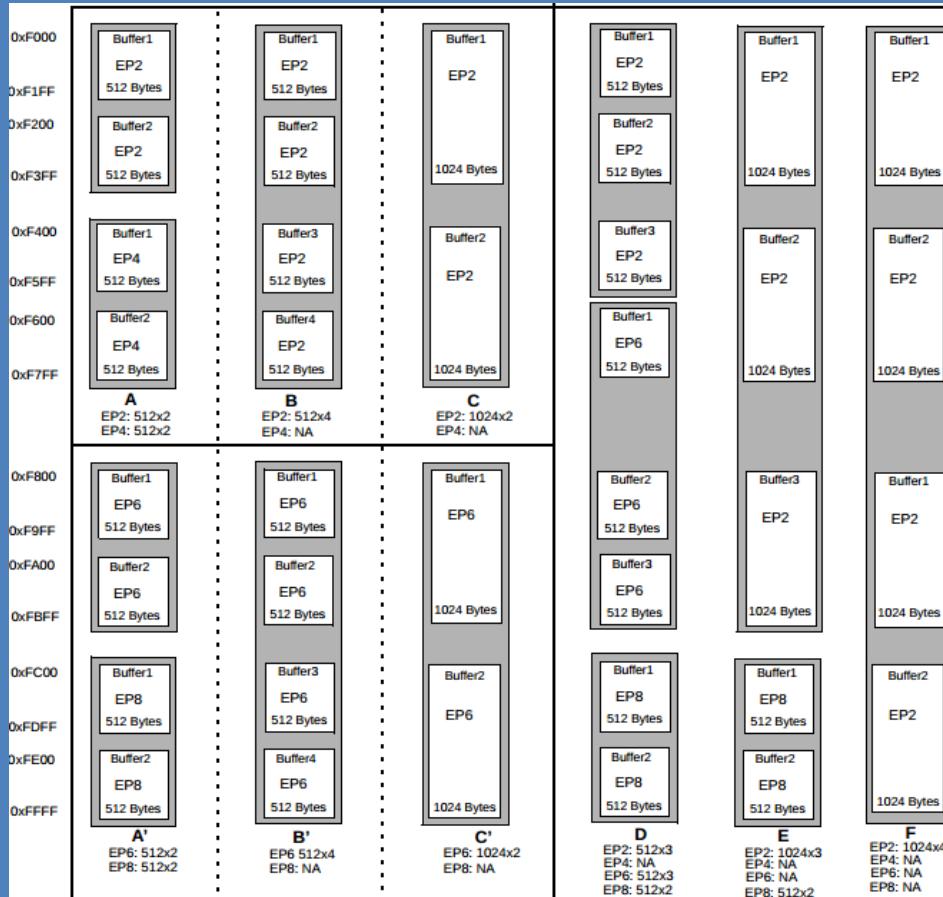
- Biblioteca de C/C++ genérica para manejar puertos USB
- Código abierto y muy bien documentado
- Es compatible con cualquier sistema operativo
- Es compatible con todos las versiones de USB disponibles

# Agenda

- Desarrollo
  - Configuración de la FX2LP
    - Explicación de las memorias FIFO configurables
    - Framework de Cypress
    - Firmware
    - Problemas presentados

# Configuración del FX2LP

- Memorias FIFO



# Configuración del FX2LP

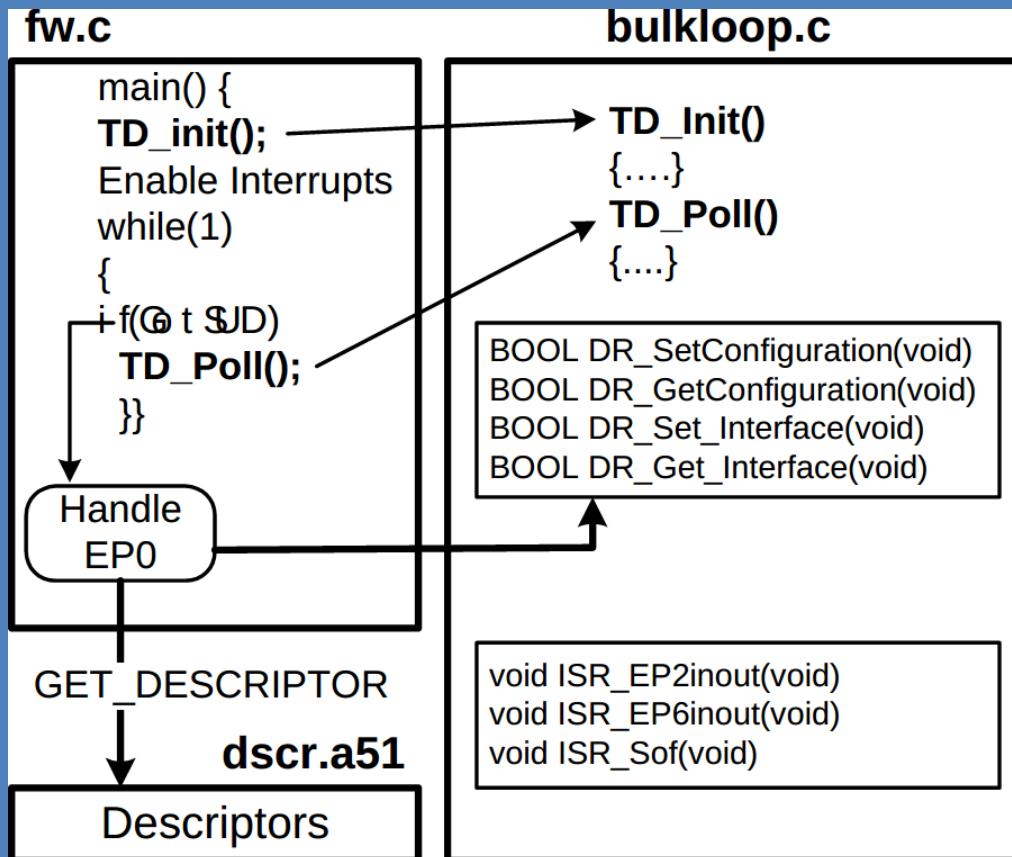
- Memorias FIFO
  - Selección de solo dos extremos
    - EP2:
      - Entrada
      - Doble Buffer
      - 512 B/buffer
      - Transmisión Isocrónica

# Configuración del FX2LP

- Memorias FIFO
  - Selección de solo dos extremos
    - EP8:
      - Salida
      - Doble Buffer
      - 512 B/buffer
      - Transmisión por bultos

# Configuración del FX2LP

- Framework



# Configuración del FX2LP

- Firmware

```
void TD_Init(void)          // Called once at startup
{ [...]
 // set the CPU clock to 48MHz
 //CPUCS = ((CPUCS & ~bmCLKSPD) | bmCLKSPD1); // 48 MHz //commented 'cos are configured by Serial_Init
 FX2LPSerial_Init();
 FX2LPSerial_XmitString("Serial Port Initialized");
 FX2LPSerial_XmitChar('\n');
 FX2LPSerial_XmitChar('\n');

 // set the slave FIFO interface to 48MHz
 //set s-fifo to internal clk, no_output, no_inverted clk, no_async, no_gstate, slavefifo(11) = 0xC3
 IFCONFIG = 0xcb; /*/0xCB; *///para hacerlo async. Error de diseño. Corregir en VHDL previamente

SYNCDELAY;
REVCTL = 0x00; /*Chip Revision Control Register: poniendo esto en 0x01 me deja manipular los paquetes, pero debo mover paquete por paquete cada buffer... no logro tomar ningún tipo de flag y mucho menos pude mover el buffer a la memoria fifo. Por otro lado, poniendo el bit 0x02 logro desactivar el auto-armado de los endopoints y puedo cambiar de modo autoout a manualout sin necesidad de perder datos.*/
SYNCDELAY;
```

# Configuración del FX2LP

- Firmware

```
void TD_Init(void)      // Called once at startup
{ [...]
    //Pin Flags Configuration
    PINFLAGSAB = 0xBC; // FLAGA <- EP2 Full Flag
    // FLAGD <- EP2 Empty Flag
    SYNCDELAY;
    PINFLAGSCD = 0x8F; // FLAGC <- EP8 Full Flag
    // FLAGB <- EP8 Empty Flag

    EP1OUTCFG = 0xA0;
    SYNCDELAY;
    EP1INCFG = 0xA0;
    SYNCDELAY;
    EP4CFG &= 0x7F;
    SYNCDELAY;
    EP6CFG &= 0x7F;
    SYNCDELAY;
    EP8CFG = 0xA0; //EP8 is DIR=OUT, TYPE=BULK, SIZE=512, BUF=2x
    SYNCDELAY;
    EP2CFG = 0xD2; // EP2 is DIR=IN, TYPE=ISOC, SIZE=1024, BUF=3x EP2CFG@e612 --> size = 512, buf = 2x
    SYNCDELAY;
```

# Configuración del FX2LP

- Firmware

```
void TD_Init(void)          // Called once at startup
{ [...]
    //setting on auto mode. rising edge is necessary
    EP8FIFO CFG = 0x11; // & ~bmAUTOOUT; //at the end, auto mode is setted off
    SYNCDELAY;
    EP2FIFO CFG = 0x0D;
    SYNCDELAY;
    [...]
    // We want to get SOF interrupts
    USBIE |= bmSOF;
    EIEX4 = 1;
    INTSETUP |= (INT4IN | bmAV4EN);
    EXIF &= ~0x40;
    EP8FIFO IE = 0x03;
    [...]}
```

# Configuración del FX2LP

- Firmware

```
void TD_Poll(void)      // Called repeatedly while the device is idle
{
    BYTE dum;
    if(EP8FIFOFLGS & bmBIT1)//ep8 fifo empty
        dum = D4ON;
    else
        dum = D4OFF;
    if(EP8FIFOFLGS & bmBIT0)//ep8 fifo full//(EP2468STAT & bmBIT6)//ep8 empty
        dum = D3ON;
    else
        dum = D3OFF;
    if(EP2FIFOFLGS & bmBIT1)//ep2 fifo empty
        dum = D2ON;
    else
        dum = D2OFF;
}
```

# Configuración del FX2LP

- Firmware

```
void ISR_Ep8eflag(void) interrupt 0
{
    EXIF &= ~bmBIT6;
    EP8FIFOIRQ = 0x02;
    FX2LPSerial_XmitHex2(EP8BCH);
    FX2LPSerial_XmitHex2(EP8BCL);
    FX2LPSerial_XmitString("\nEP8: Estoy vacío\n");
}

void ISR_Ep8fflag(void) interrupt 0
{
    EXIF &= ~0x40;
    EP8FIFOIRQ = 0x01;
    FX2LPSerial_XmitString("EP8: Estoy lleno\n");
}
```

# Configuración del FX2LP

- Problemas presentados
  - Problemas de sincronismo:
    - El código provisto por el fabricante aleatoria y frecuentemente se colgaba cuando el firmware era cargado al chip
    - Falla detectada en una reconexión del chip.
    - Para debug se implementó una comunicación serie vía UART.
    - La falla no volvió a aparecer con un mensaje de comunicación entre la etapa de desconexión y reconexión. Sin esto no se logró que el fallo no se presente.

# Configuración del FX2LP

- Problemas presentados
  - Problemas de sincronismo:

```
void main()
{[...]
 // unconditionally re-connect. If we loaded from eeprom we are
 // disconnected and need to connect. If we just renumerated this
 // is not necessary but doesn't hurt anything
 USBCS &= ~bmDISCON;

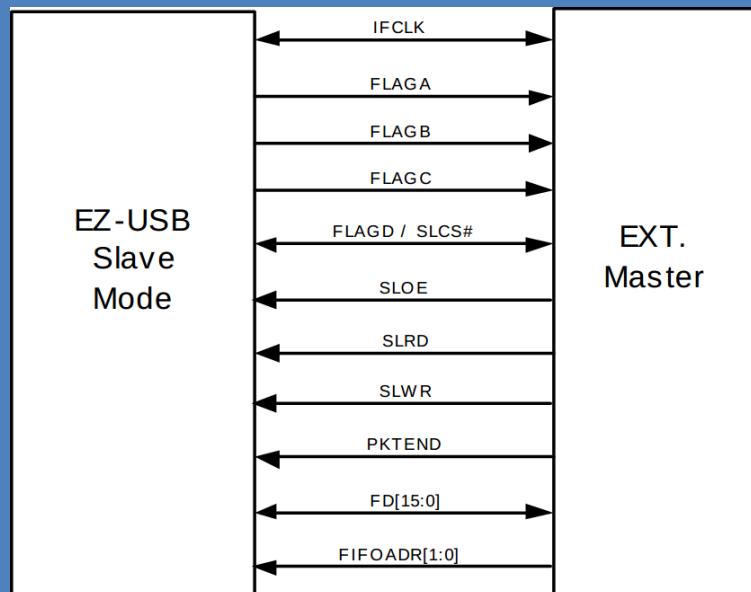
 FX2LPSerial_XmitString("Reconecting...\n\n");
 [...]
}
```

# Agenda

- Desarrollo
  - Desarrollo del VHDL de la placa MOJO v3
    - Interfaz con la CY3684
    - Maquina de Estados
    - Test Bench's
    - Problemas presentados

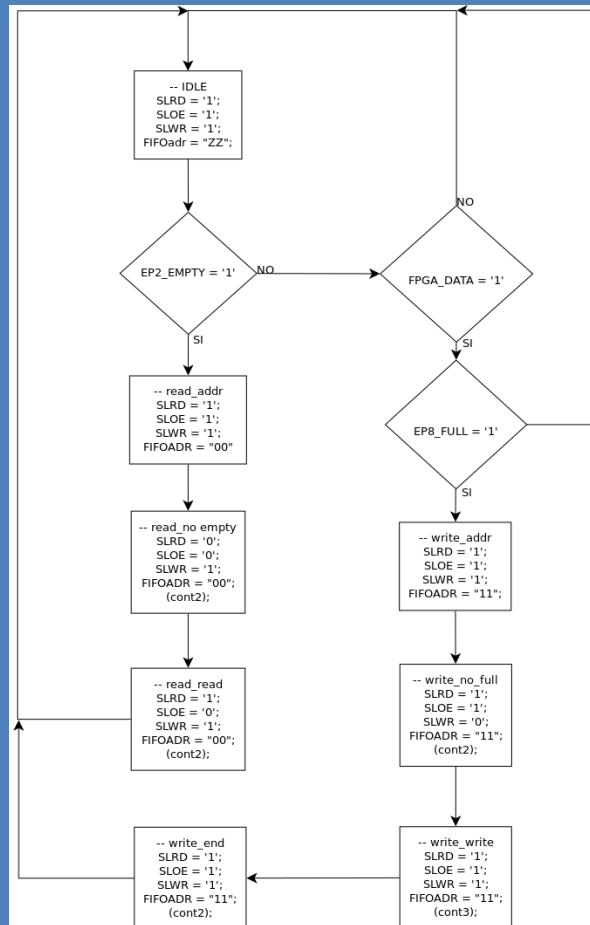
# Desarrollo del VHDL

- Interfaz con la CY3684
  - El integrado CY7C68013A se configuró con sus memorias FIFO en modo esclavo, las cuales presentan la siguiente interfaz



# Desarrollo del VHDL

- Maquina de estados finitos



# Desarrollo del VHDL

- Maquina de estados finitos
  - La realización de esta pequeña maquina de estado consume 36 de los 11440 Slices y 51 de las 5720 LUT's del Spartan6

Slice Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Registers	36	11,440	1%	
Number used as Flip Flops	35			
Number used as Latches	1			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	51	5,720	1%	
Number used as logic	51	5,720	1%	
Number using O6 output only	25			
Number using O5 output only	1			
Number using O5 and O6	25			
Number used as ROM	0			
Number used as Memory	0	1,440	0%	
Number of occupied Slices	22	1,430	1%	
Number of MUXCYs used	12	2,860	1%	
Number of LUT Flip Flop pairs used	51			
Number with an unused Flip Flop	16	51	31%	
Number with an unused LUT	0	51	0%	
Number of fully used LUT-FF pairs	35	51	68%	
Number of unique control sets	6			
Number of slice register sites lost to control set restrictions	28	11,440	1%	
Number of bonded IOBs	37	102	36%	
Number of LOCed IOBs	36	37	97%	
IOB Flip Flops	1			
IOB Latches	16			
Number of RAM16BWERs	0	32	0%	
Number of RAM8BWERs	0	64	0%	

# Desarrollo del VHDL

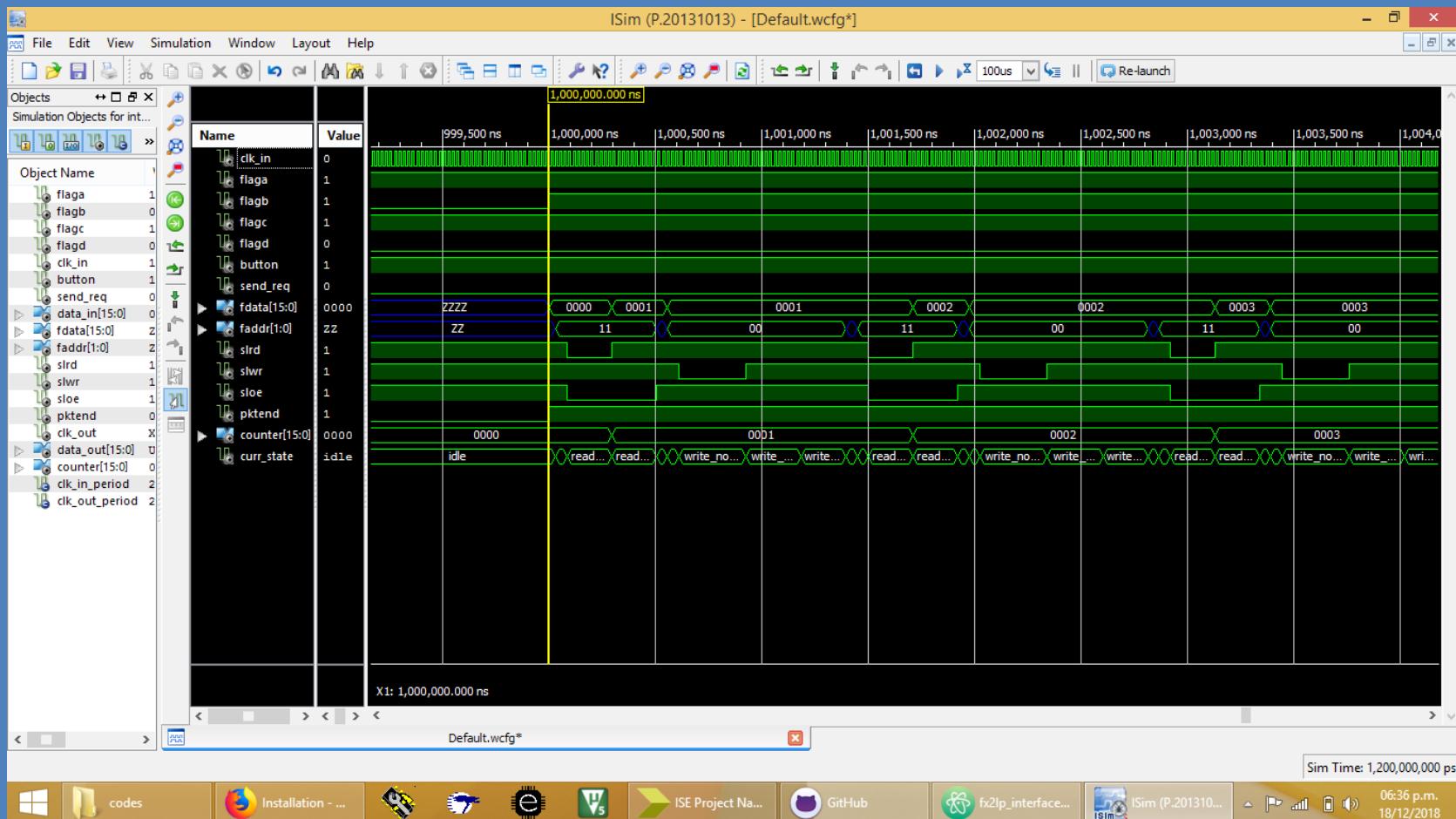
- Maquina de estados finitos
  - Se observa que la utilización del PLL consume mucho recurso, ya que solo posee 4. Sin embargo esto será salvado con la versión 3 del circuito impreso que se explica más adelante.

# Desarrollo del VHDL

- Test Bench's
  - Se probó el sistema en diferentes situaciones
    - Inicio de la comunicación cuando se presentan datos a leer
    - Final de la comunicación cuando se acaban los datos a transmitir
    - Final de la comunicación cuando se llena la memoria externa
    - Se observó en detalle el correcto funcionamiento de la MEF

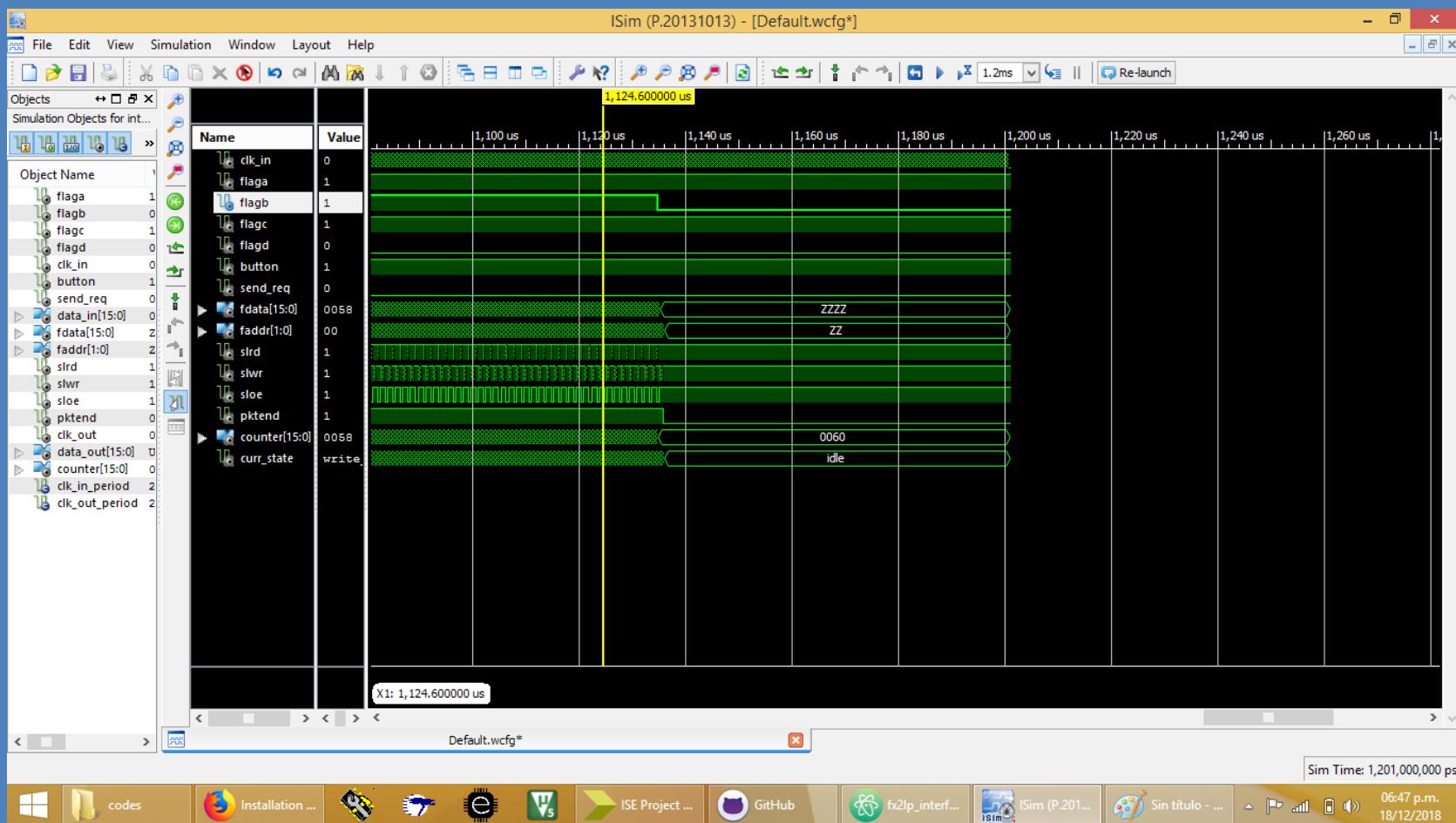
# Desarrollo del VHDL

- Test Bench's



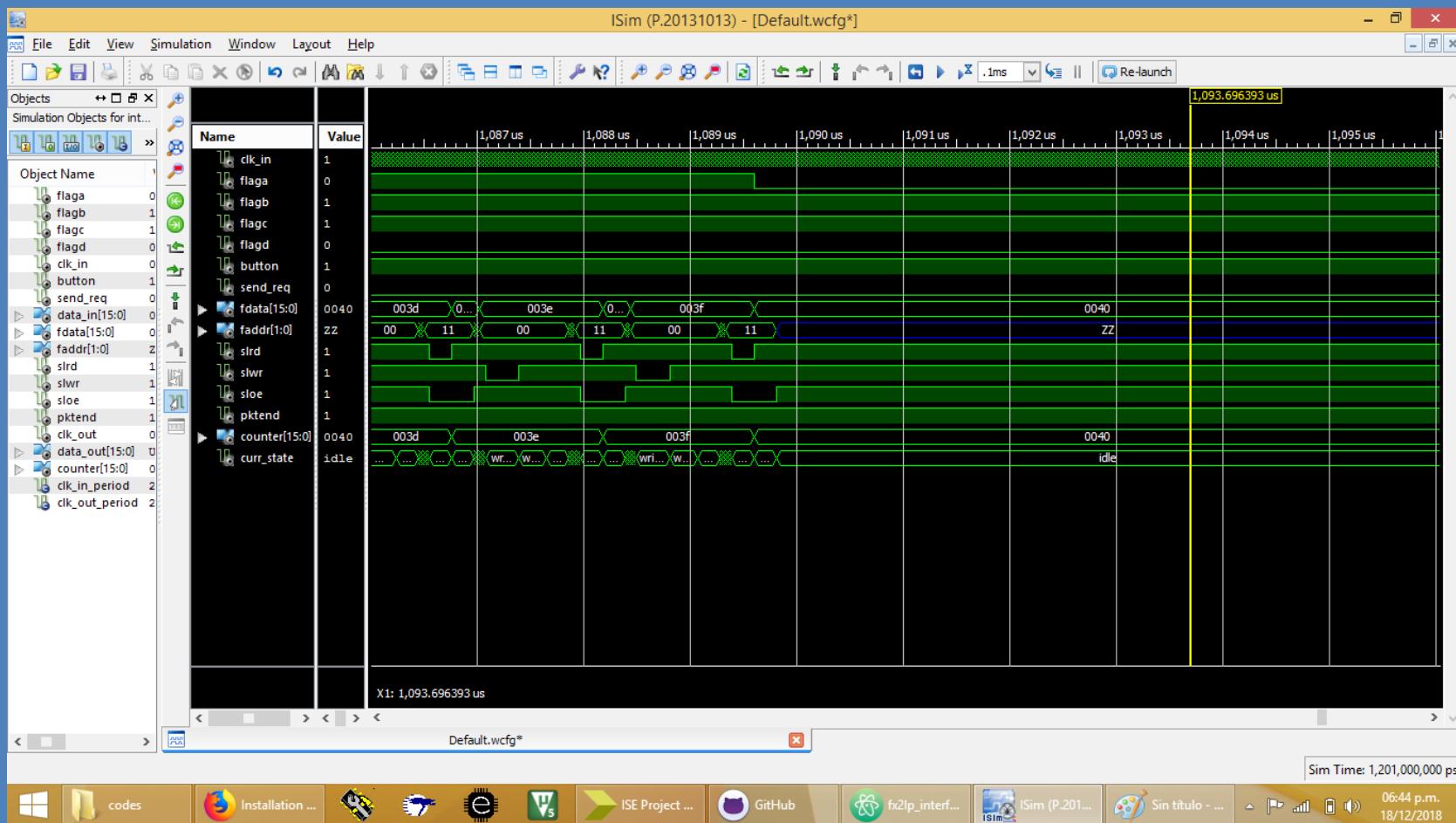
# Desarrollo del VHDL

- Test Bench's



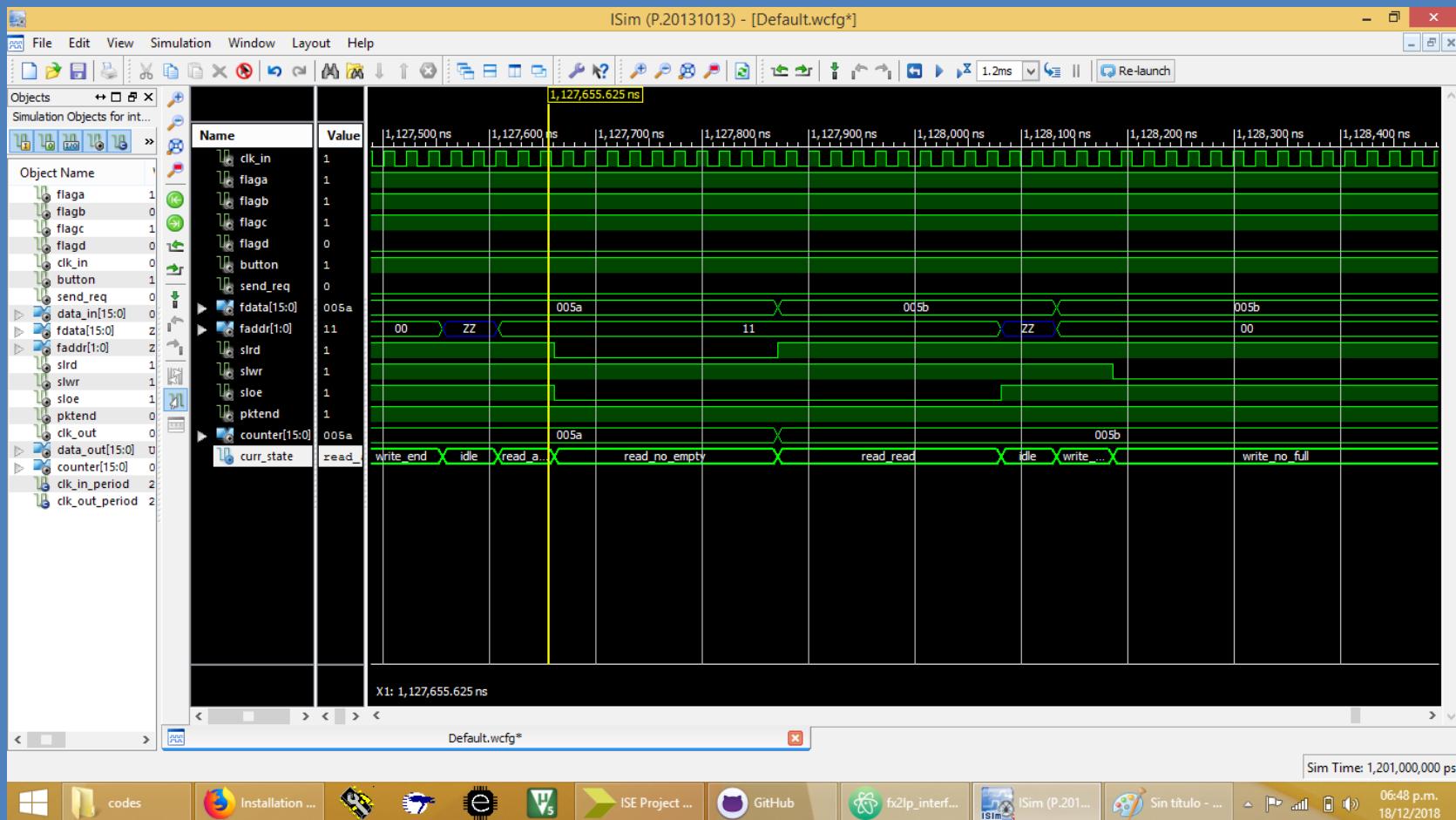
# Desarrollo del VHDL

- Test Bench's



# Desarrollo del VHDL

- Test Bench's



# Desarrollo del VHDL

- Problemas Presentados
  - Se halló un error de sincronismo debido a una leve diferencia en los relojes de los sistemas. Para solucionarlo se bajó la frecuencia de trabajo.
    - Queda por resolver para trabajos futuros.
  - Error por perdida de datos. Se encontró que el archivo de asignación de pines no estaba correctamente escrito.

# Agenda

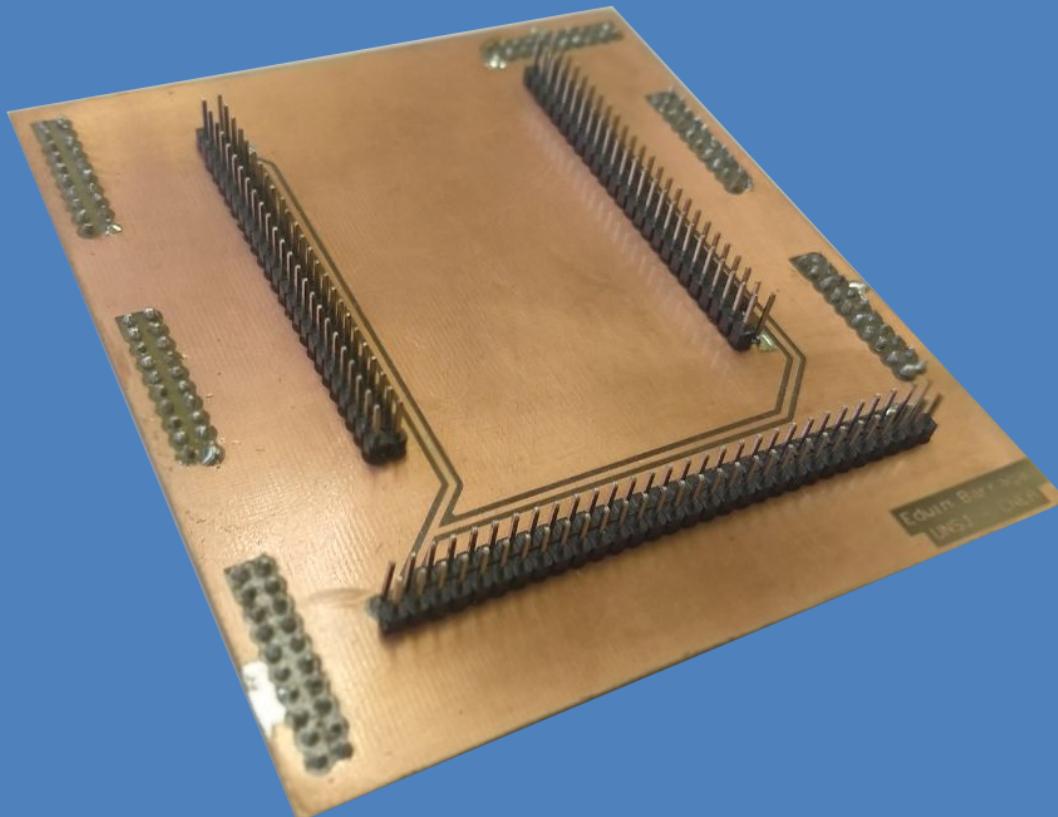
- Desarrollo
  - Placa de conexión
    - Versión 1
    - Versión 2
    - Versión 3
- Problemas presentados del sistema en general
- Resultados y conclusiones

# Placa de interconexión

Para garantizar la correcta conexión eléctrica entre las dos placas, se debió realizar 3 circuitos impresos, ya que las mismas presentaron diferentes fallas, tanto de fabricación como de diseño.

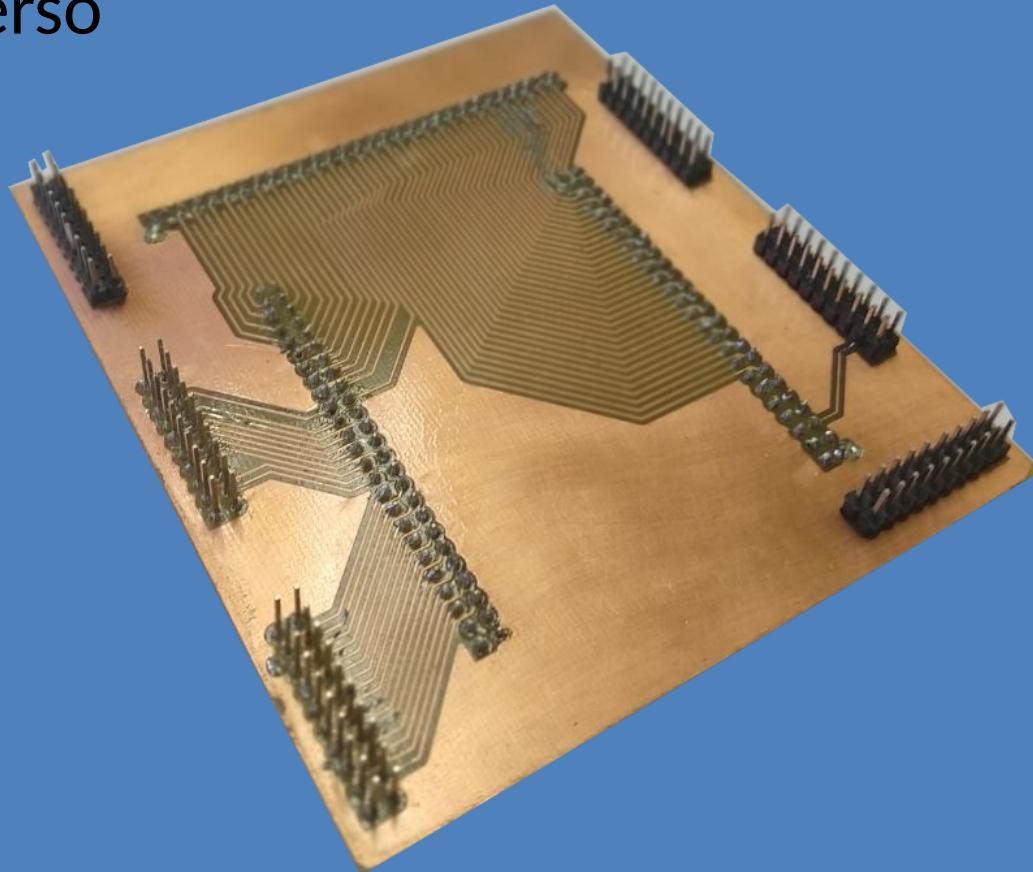
# Placa de interconexión

- Versión 1
  - Anverso



# Placa de interconexión

- Versión 1
  - Reverso

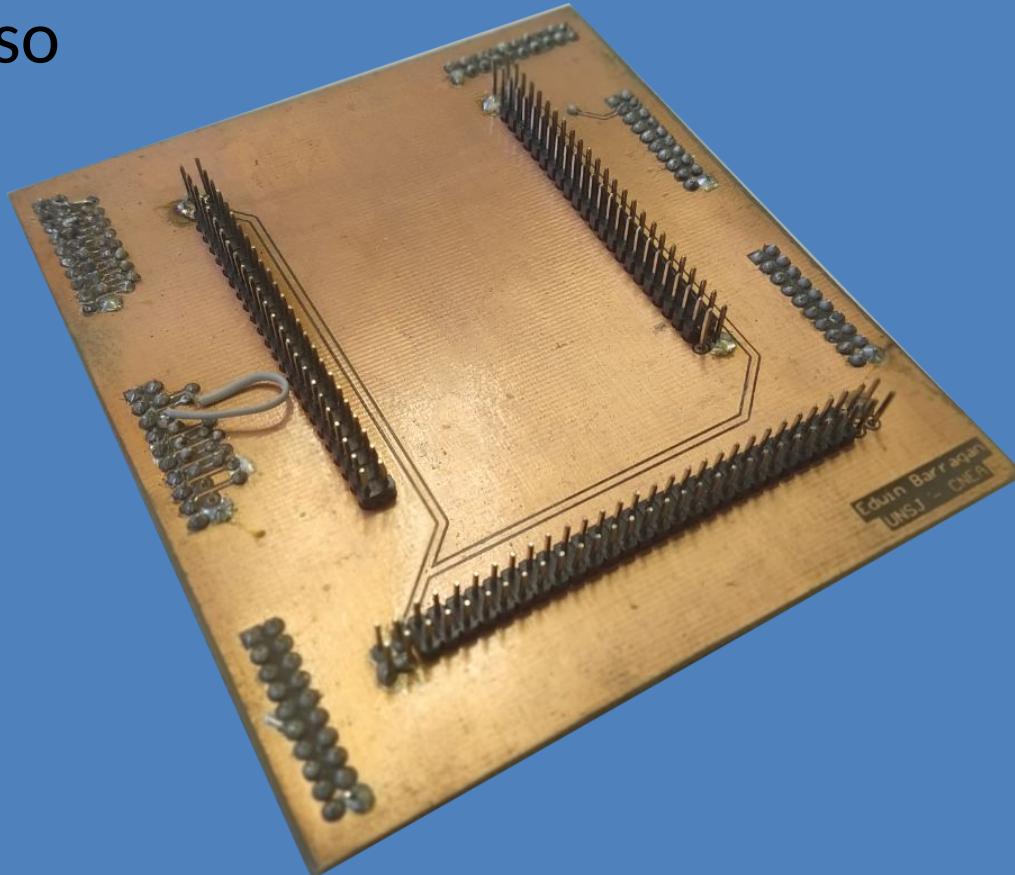


# Placa de interconexión

- Versión 1
  - Error de fabricación
    - La fabricación se preveía que debía tener agujeros metalizados y esto no fue así.
  - Error de ensamblaje
    - En la foto del reverso se observa que los pines conectados, que debían ser hembras, son machos. Esto podía ser salvado, sin embargo el error de fabricación impedía soldar correctamente pines hembra en la capa correcta, por lo que se descartó el impresor.

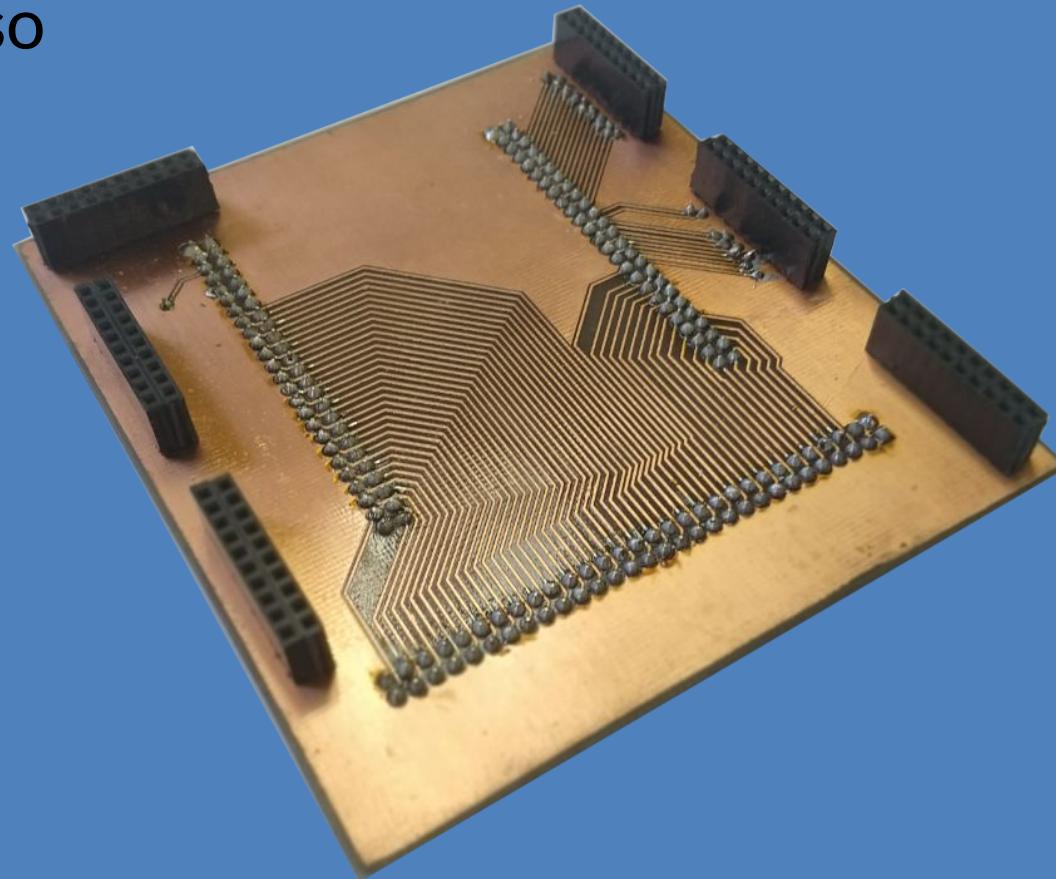
# Placa de interconexión

- Versión 2
  - Anverso



# Placa de interconexión

- Versión 2
  - Reverso



# Placa de interconexión

- Versión 3
  - Error de diseño
    - El pin que debe servir de reloj terminó conectado en un punto ciego del microcontrolador que posee la placa MOJO v3.
  - Error de fabricación
    - Un pin de conexión se rompió durante el proceso de taladrado por lo que hubo que colocar un cable para salvar el detalle.

# Placa de interconexión

- Versión 3
  - Este impreso se envió a realizar a china con un proceso de mayor calidad.
    - Presenta agujeros metalizados, lo que permitió volver a un diseño sin vías pasantes innecesarias.
    - Se corrigió el error de diseño de la versión 2 y se agregó conexiones adicionales en caso de ser necesarias en futuros trabajos.
  - Aun no montada

# Agenda

- Desarrollo
  - Placa de conexión
    - Versión 1
    - Versión 2
    - Versión 3
- Problemas presentados del sistema en general
- Resultados y conclusiones

# Problemas Generales

- Debido al error de diseño del impreso Versión 2, la comunicación intercircuitos, que se pensaba hacer sincrónica debió ser asíncrona.
- Debido a este problema de sincronismo, no se puede aprovechar al máximo el ancho de banda de la comunicación USB

# Agenda

- Desarrollo
  - Placa de conexión
    - Versión 1
    - Versión 2
    - Versión 3
- Problemas presentados del sistema en general
- Resultados y conclusiones

# Resultados

- Se escribió un código programa que envía mensajes aleatorios desde la PC y recibe ecos desde el FPGA.
  - Este programa no prueba el ancho de banda aunque si la robustez.
  - Se dejó correr mas de 24 horas sin obtener ninguna pérdida de información.

# Conclusiones

- Se logró realizar un sistema que envía y recibe datos desde una PC a un FPGA y viceversa de forma satisfactoria.
- Se logró comprender el funcionamiento del protocolo de comunicación USB.
- Se implementó y sintetizó un circuito descripto en VHDL que consume menos del 1% de los recursos de un Spartan6
- Se logró configurar un integrado comercial y comprender como es su funcionamiento.

# Conclusiones

- Se aprendió la importancia de la revisión exhaustiva de los trabajos mandados a fabricar y de la correcta especificación de los trabajos solicitados.