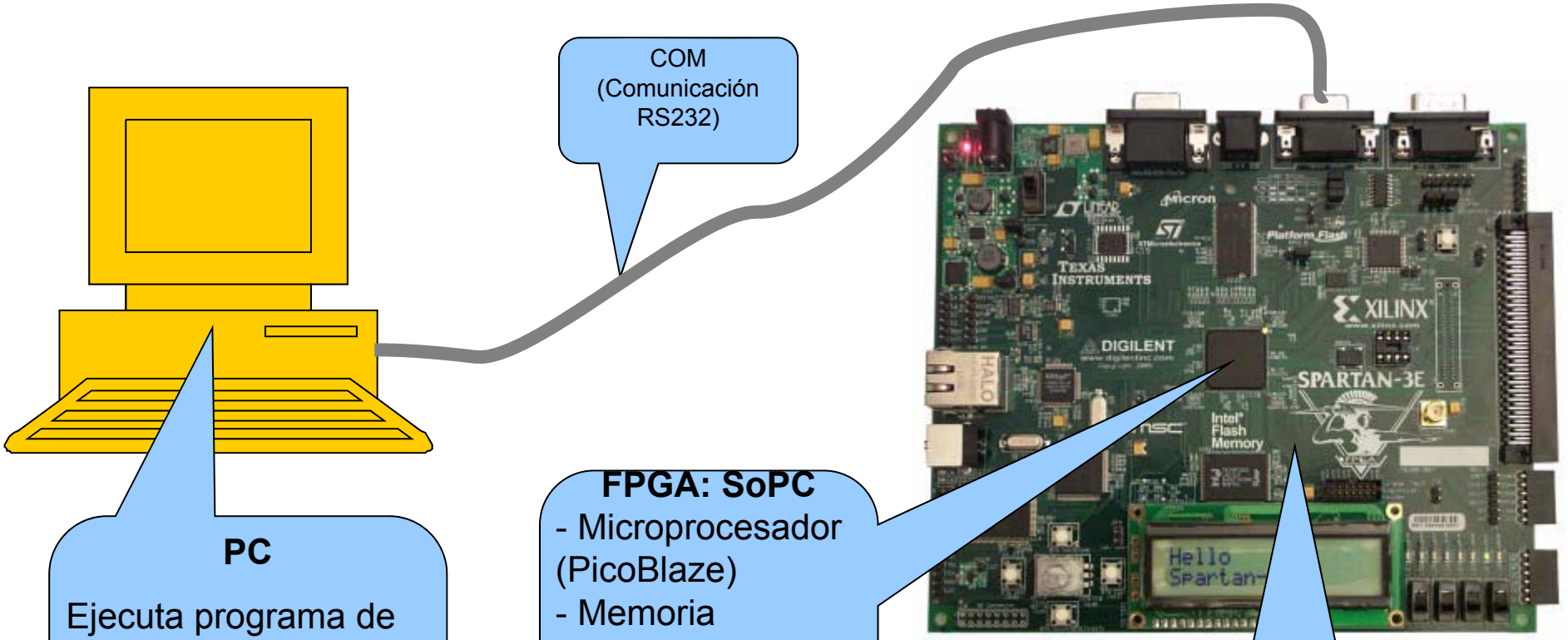
P2-Proyecto con PicoBlaze

Arquitecturas Hardware de Comunicaciones

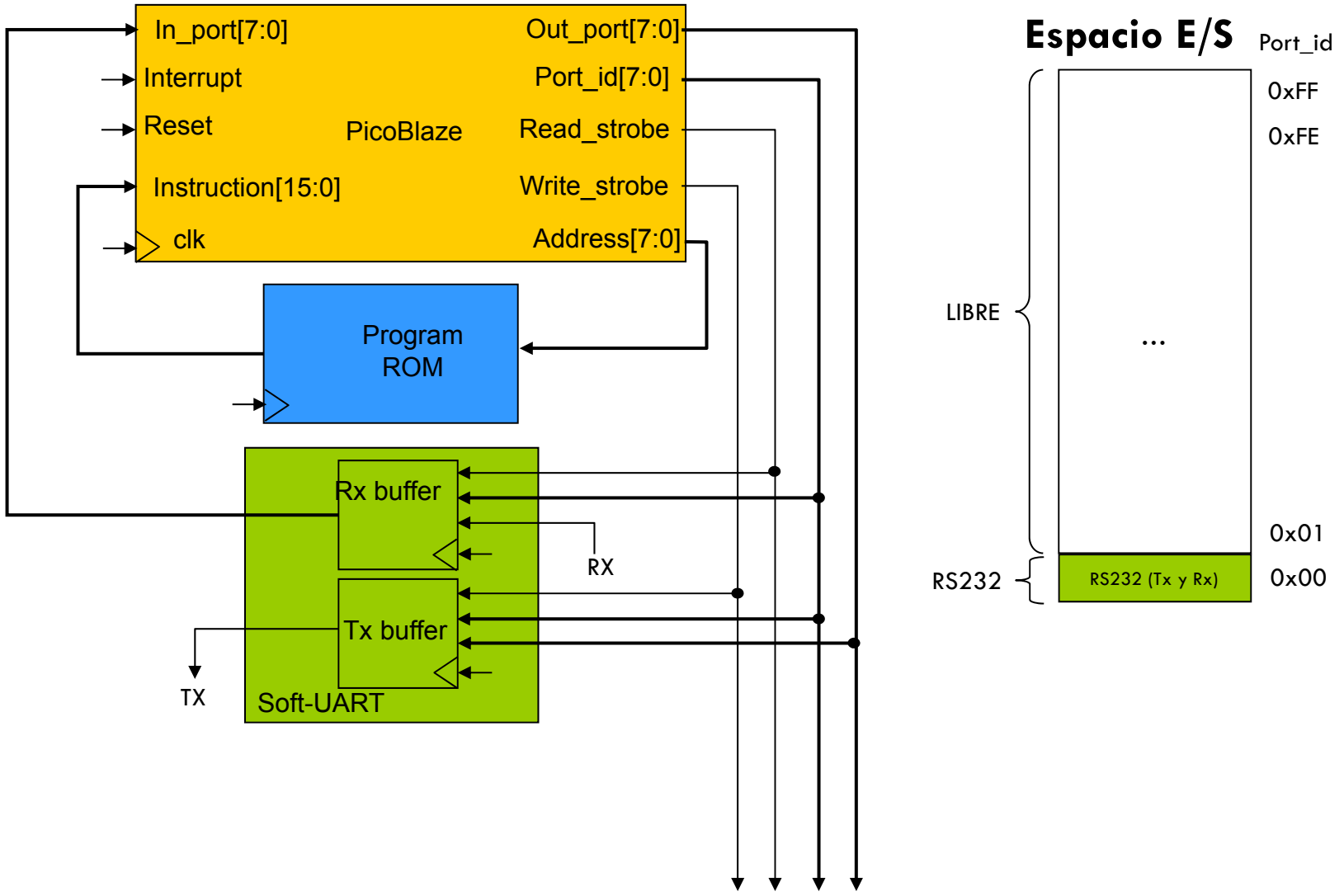
Objetivo: Diseñar un SoPC simple



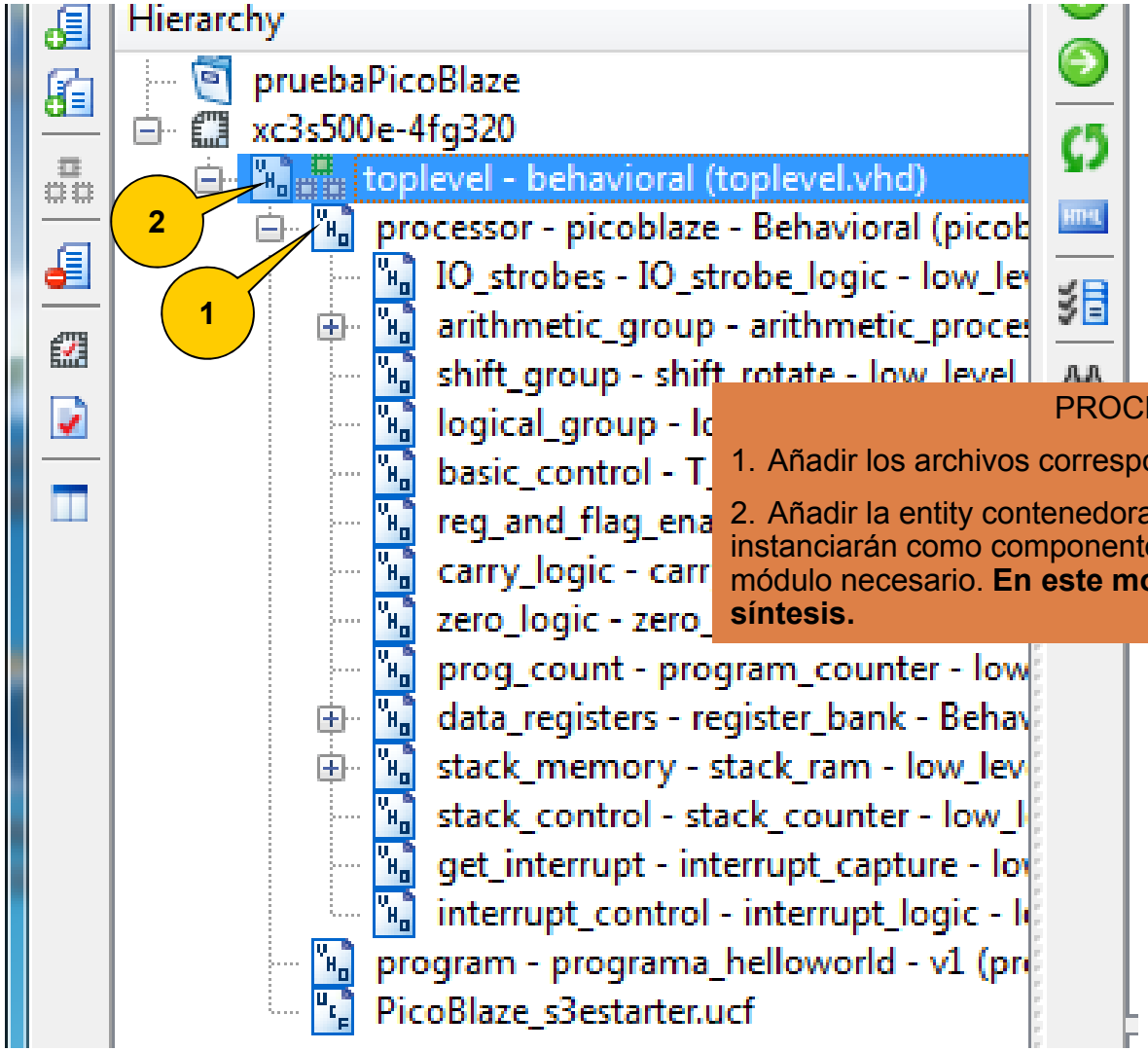
PARTE 1

PROYECTO HELLOWORLD

Sistema mínimo: "Helloworld"



Crear Proyecto ISE: PicoBlaze_Helloworld

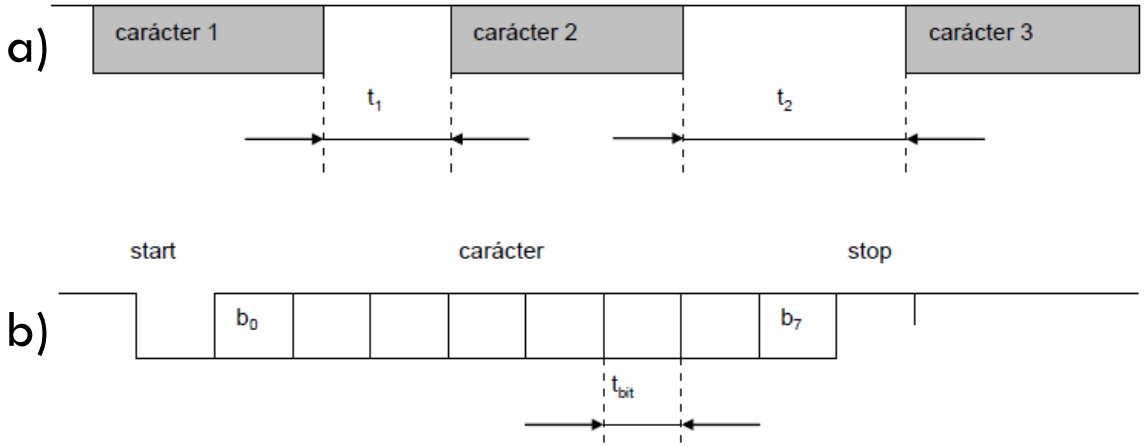


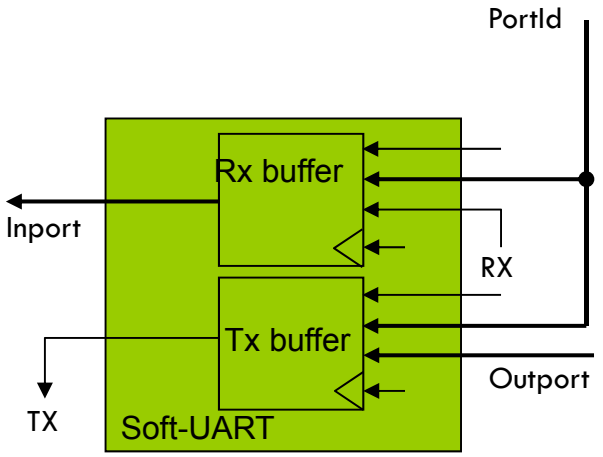
PROCEDIMIENTO:

1. Añadir los archivos correspondientes al **PicoBlaze**.
2. Añadir la entity contenedora (***toplevel***), donde se instanciarán como componentes el PicoBlaze, y cualquier otro módulo necesario. **En este momento, ya se puede realizar la síntesis.**

Transmision serie asíncrona: RS232

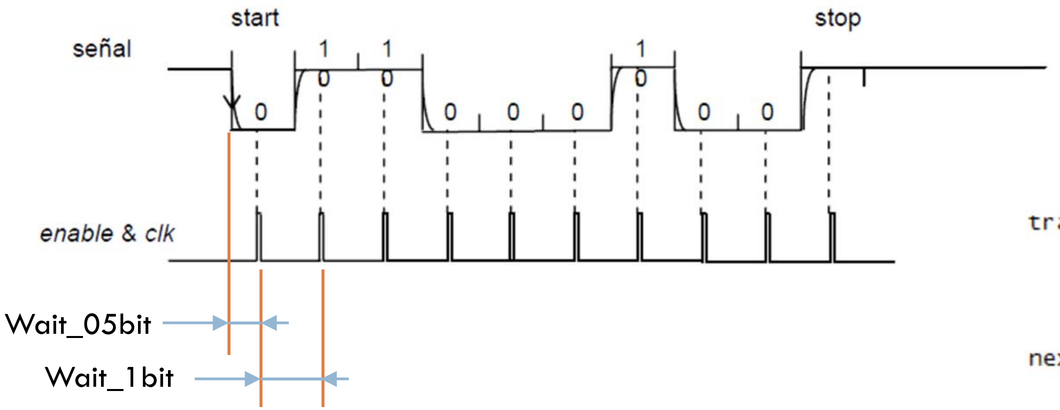
- La unidad de transmisión es el carácter
- Tiempo entre caracteres arbitrario (t_i)
- Temporización rígida dentro de cada carácter (t_{bit})
- Formato: 1 bit start – 8 bits datos – 1 bit stop
- El primer bit transmitido es el de menor peso (b_0)





Código ASM: programa_helloworld.asm

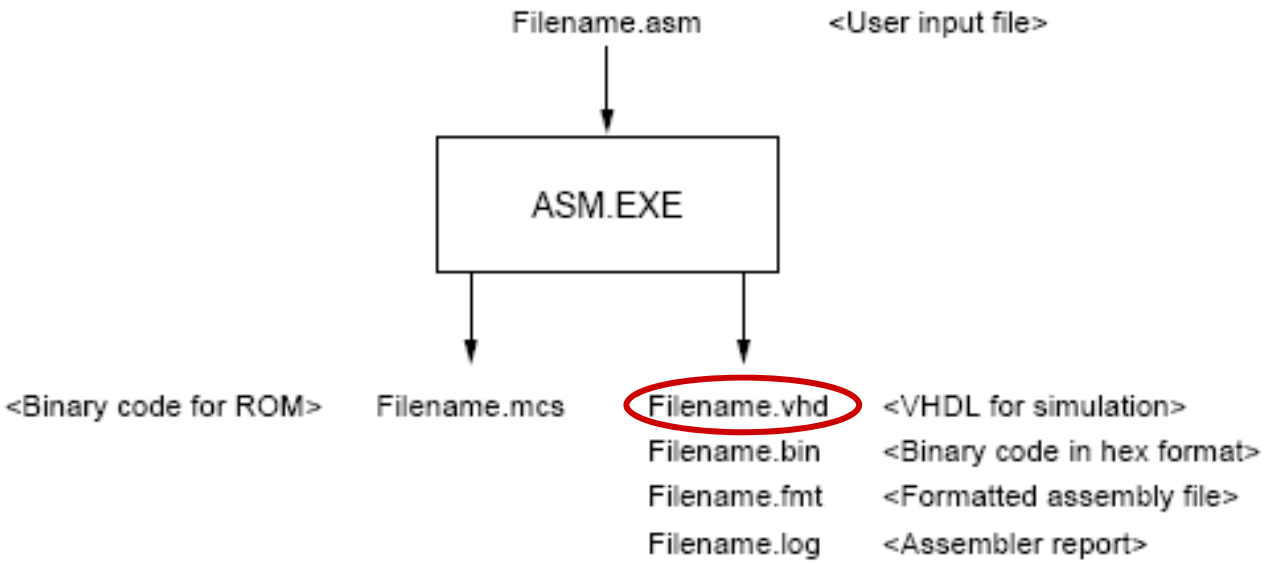
- Directivas
- Main: realiza Eco +1
- Rutinas: Recibe y Transmite
  - Rutinas de espera
    - Muestreo de datos



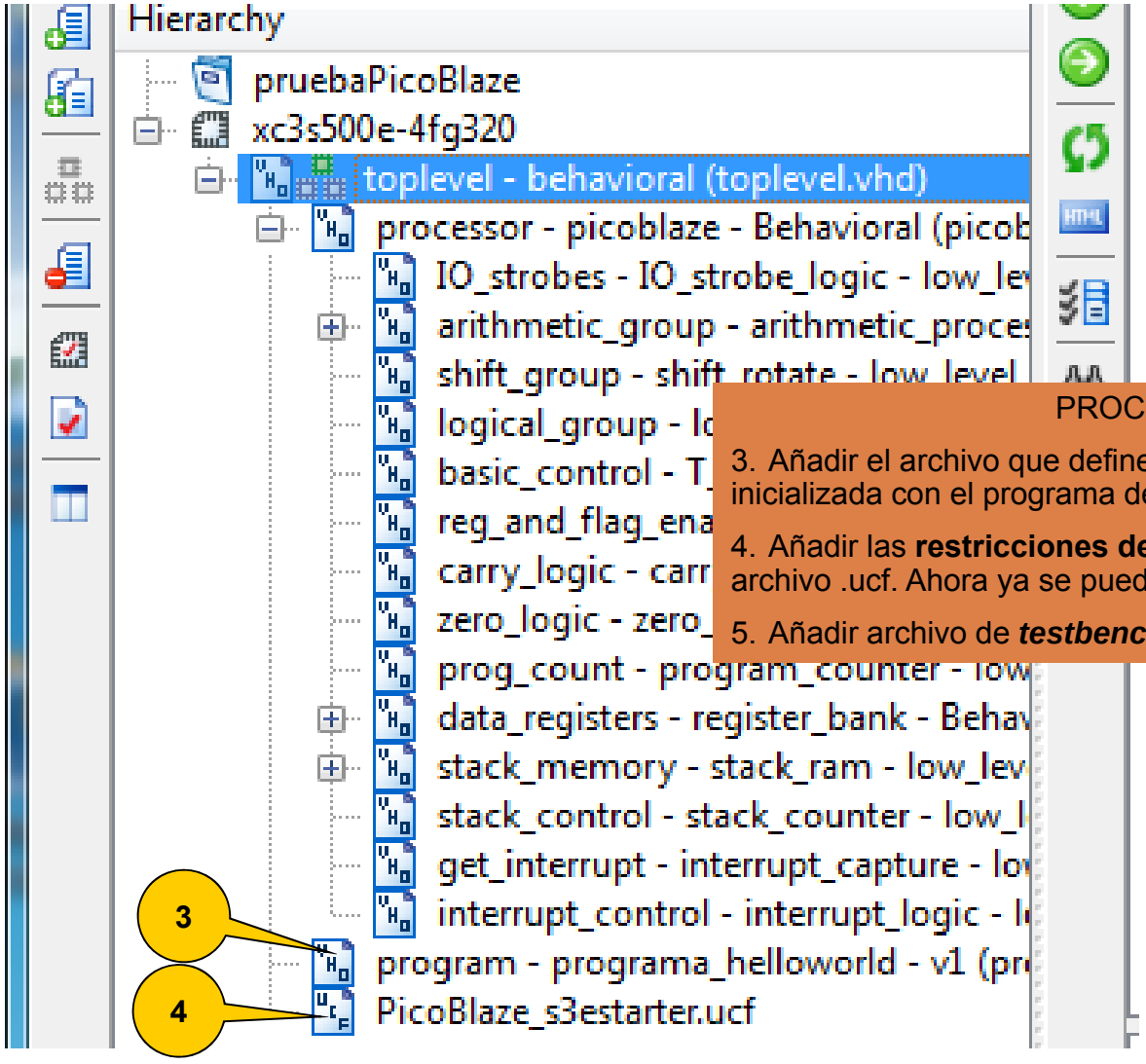

```
            ADDRESS         00
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ;Inicio del programa
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
start:      ;esperamos a recibir un caracter
            CALL            recibe
            ;copiamos el caracter recibido al buffer de
            LOAD            txreg, rxreg
            ADD             txreg, 01
            ;hacemos el eco del caracter recibido
            CALL            transmite
            JUMP            start
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ;Rutina de recepcion de caracteres
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
recibe:     ;esperamos a que se reciba un bit de inicio
            INPUT           rxreg, rs232
            AND             rxreg, 80
            JUMP            NZ, recibe
            CALL            wait_05bit
            ;almacenamos los 8 bits de datos
            LOAD            contbit,09
next_rx_bit: CALL           wait_1bit
            SR0             rxreg
            INPUT           s0, rs232
            AND             s0, 80
            OR              rxreg, s0
            SUB             contbit, 01
            JUMP            NZ, next_rx_bit
            RETURN
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ;Rutina de transmision de caracteres
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
transmite:  ;enviamos un bit de inicio
            LOAD            s0, 00
            OUTPUT          s0, rs232
            CALL            wait_1bit
            ;enviamos los 8 bits de datos
            LOAD            contbit, 08
next_tx_bit: OUTPUT         txreg, rs232
            CALL            wait_1bit
            SR0             txreg
            SUB             contbit, 01
            JUMP            NZ, next_tx_bit
            ;enviamos un bit de parada
            LOAD            s0, FF
            OUTPUT          s0, rs232
            CALL            wait_1bit
            RETURN
```

Ensamblador: asm.exe

- Aplicación para MS-DOS desarrollada en C. Uso:
  - C:\>asm.exe programa_helloworld.asm



Crear Proyecto ISE: PicoBlaze_Helloworld



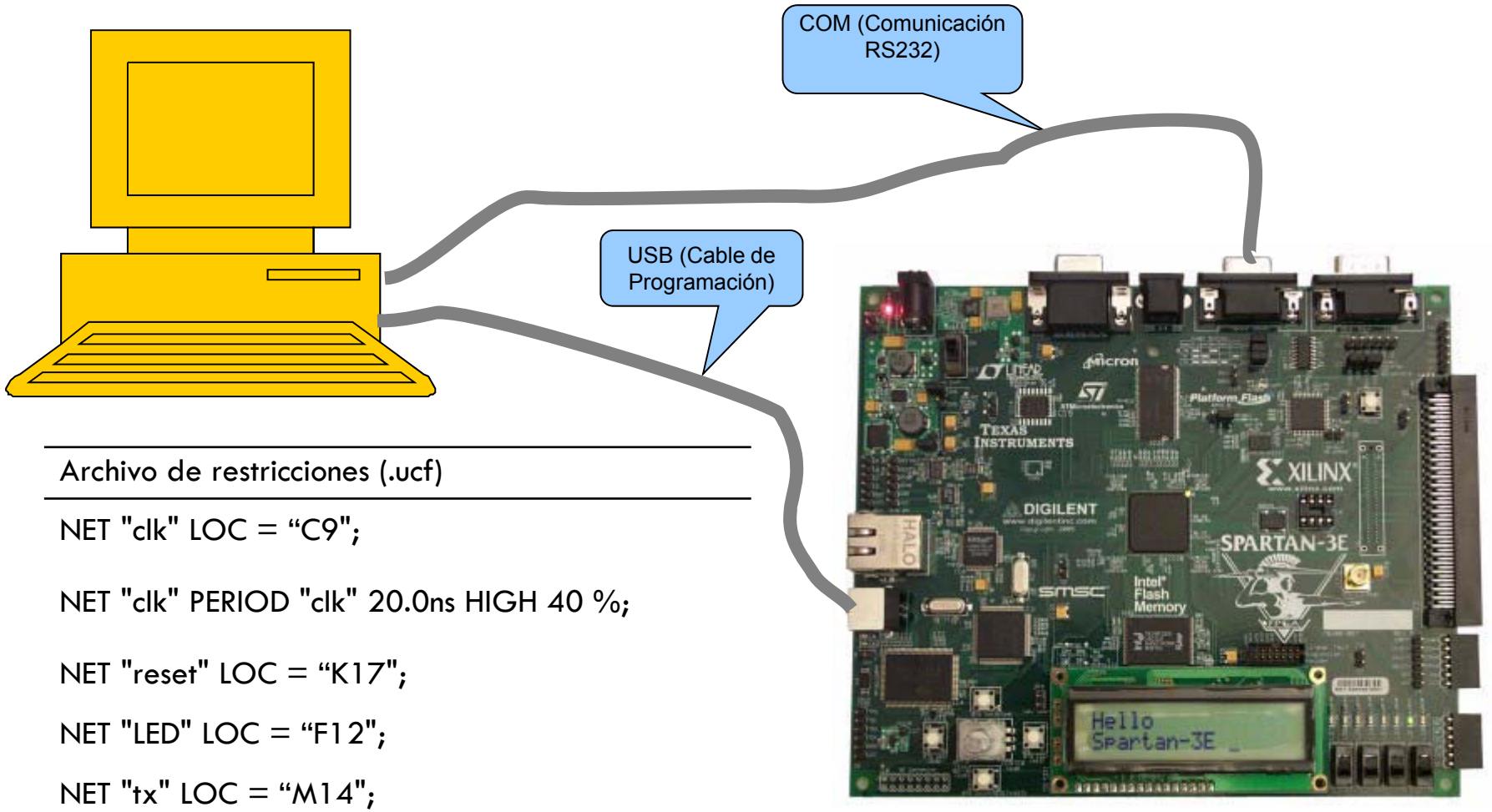
PROCEDIMIENTO:

3. Añadir el archivo que define la **ROM de programa**, inicializada con el programa de nuestra aplicación.

4. Añadir las **restricciones de pines** y tiempos mediante un archivo .ucf. Ahora ya se puede realizar la implementación

5. Añadir archivo de ***testbench*** para poder simular el sistema

Configuración para conexionado a PC



Archivo de restricciones (.ucf)

```
NET "clk" LOC = “C9";

NET "clk" PERIOD "clk" 20.0ns HIGH 40 %;

NET "reset" LOC = “K17";

NET "LED" LOC = “F12";

NET "tx" LOC = “M14";

NET "rx" LOC = “R7";
```

Programar la FPGA con iMPACT

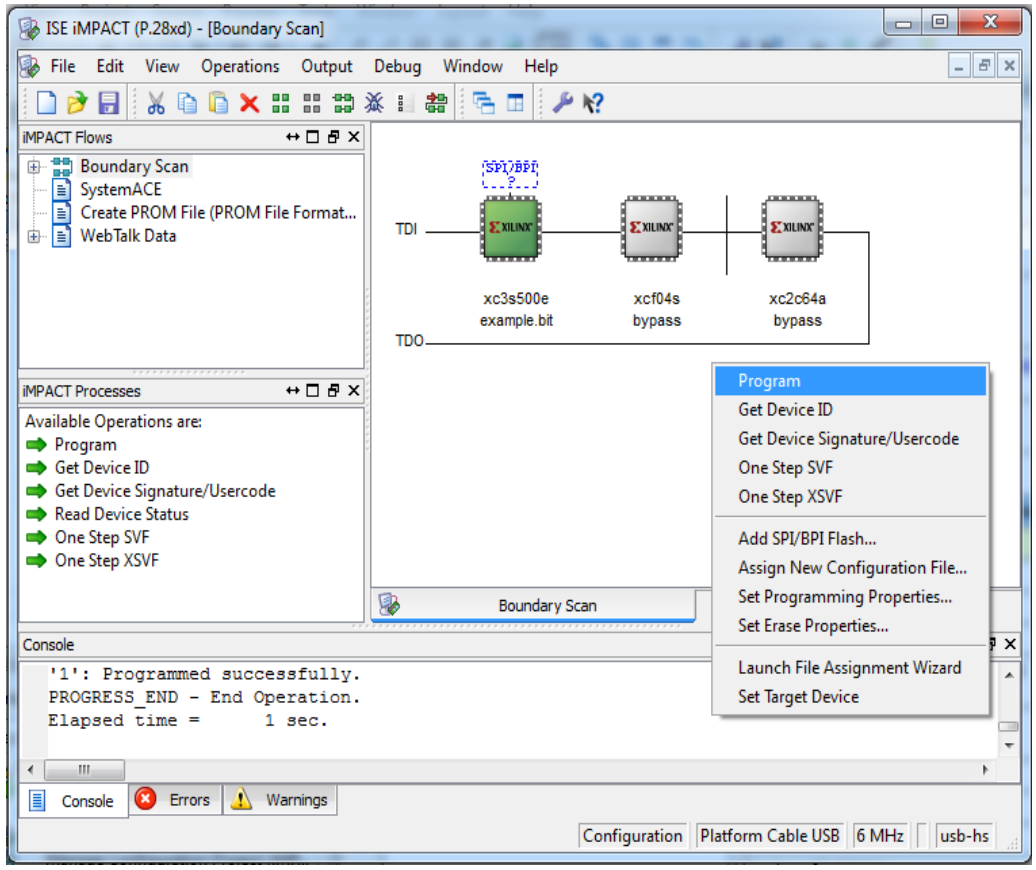
1. Lanzar el wizard de configuración: Menú→Edit→*Launch Configuration Wizard*.

2. Configurar los siguiente parámetros en el asistente de configuración del iMPACT:

   - Configurar dispositivo *vía: Boundary Scan mode*.
   - *Automatically connect to cable and identify the Boundary Scan Chain*.

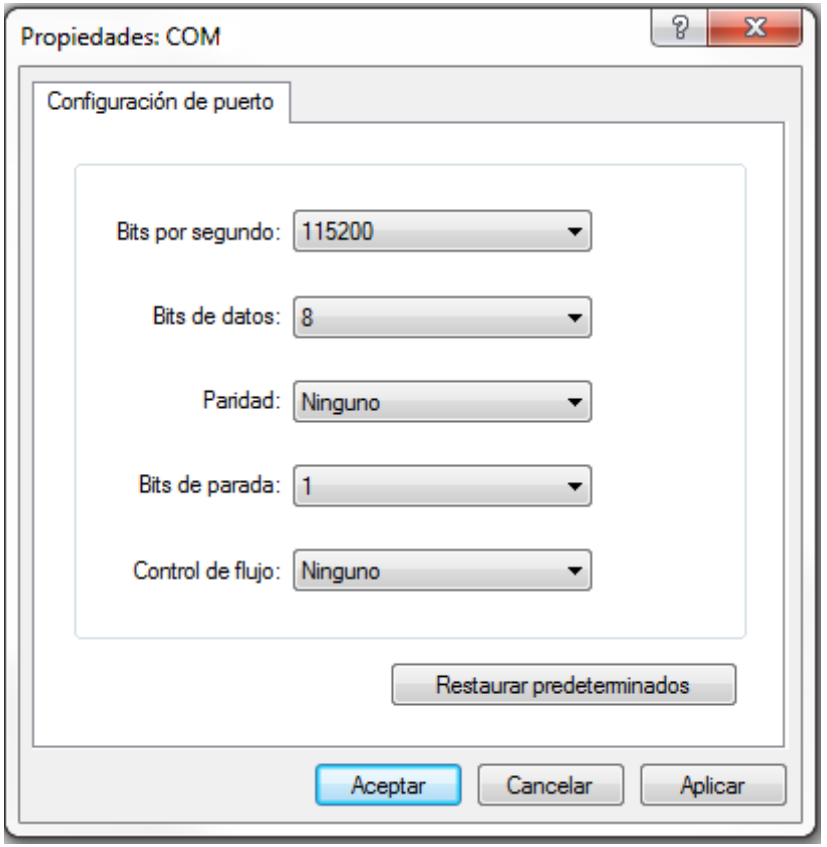
3. Asignar el archivo .bit a la FPGA tipo Spartan (xc3s500e) y dejar las otras dos en bypass.

4. Seleccionar la FPGA xc3s500e y en las opciones del boton secundario del ratón elegir “Program…” para programar la FPGA.



Configuración para comunicación con PC

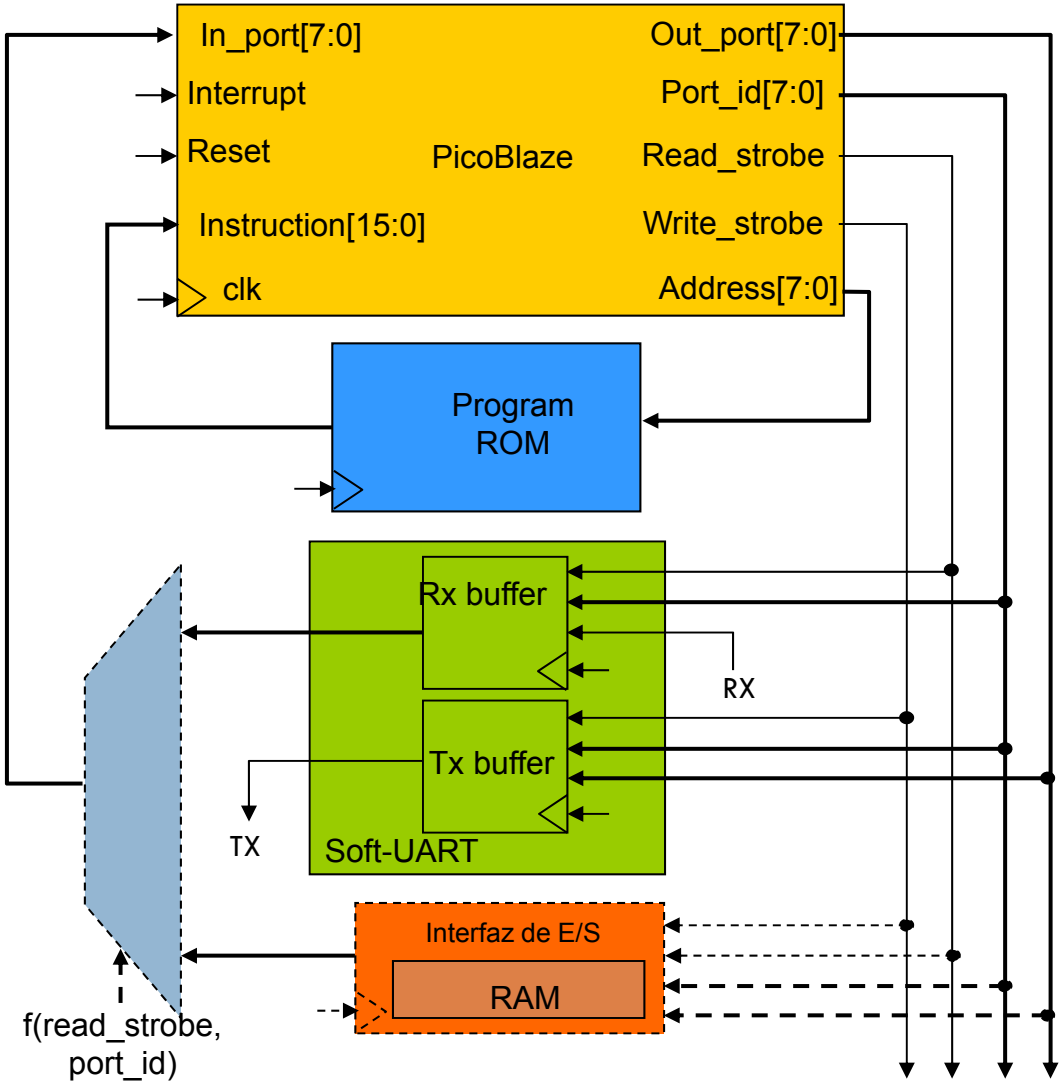
- Configuración del HyperTerminal de W'XP/W7 para funcionar con PicoBlaze en su configuración por defecto:
  - Bits por segundo: 115200
  - Bits de datos: 8
  - Paridad: Ninguno
  - Bits de Parada: 1
  - Control de Flujo: Ninguno
  - Emulación: Automática

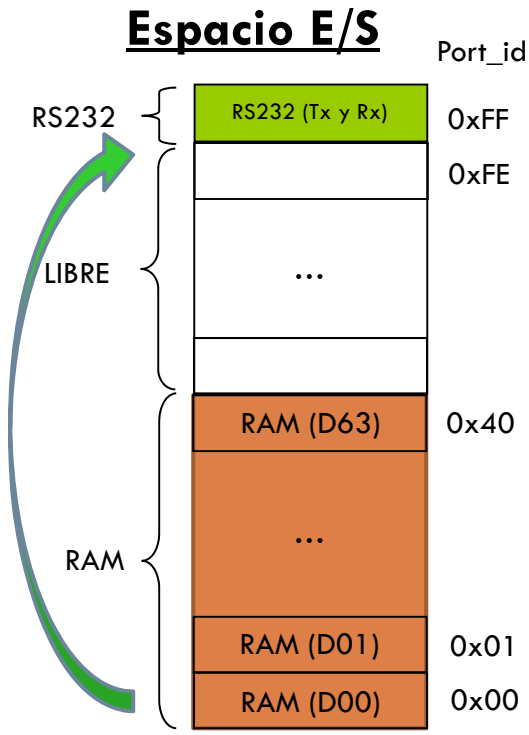


PARTE 2

MODIFICACIÓN DEL PROYECTO HELLOWORLD

Modificación-1ª: añadir RAM

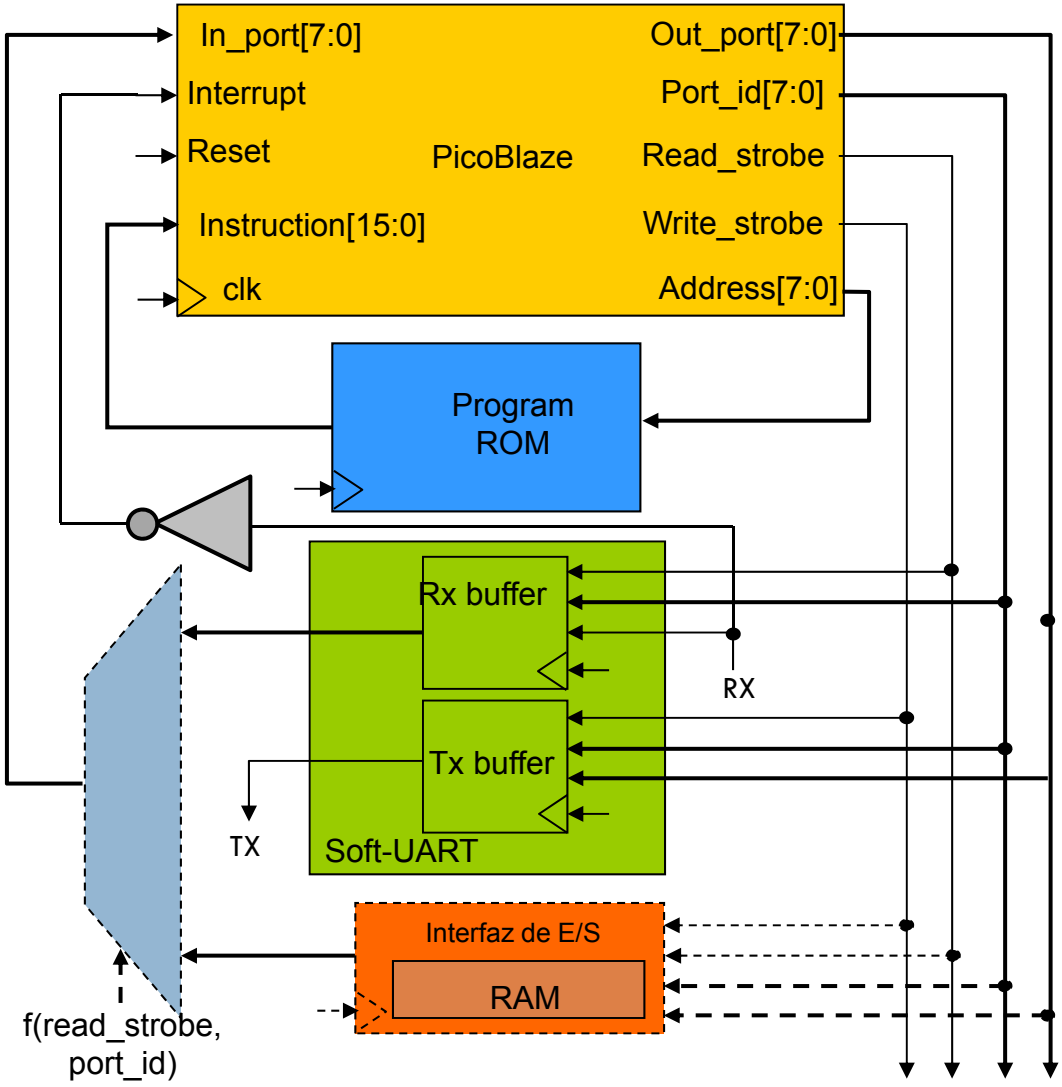




Main

```
        DISABLE INTERRUPT
start:  CALL            recibe
        ;Instrucciones para la parte1
        LOAD            S7,00
parte1: INPUT           txreg,S7
        ADD             txreg,00
        JUMP Z          parte2
        CALL            transmite
        ADD             S7,01
        JUMP            parte1
```

Modificación-2ª: añadir Interrupción



Main

```
parte2:   ENABLE INTERRUPT
bucle1:   LOAD             S6,09
bucle2:   SUB              S6,01
          JUMP NZ          bucle2
          LOAD             S6,09
          JUMP             bucle2
```

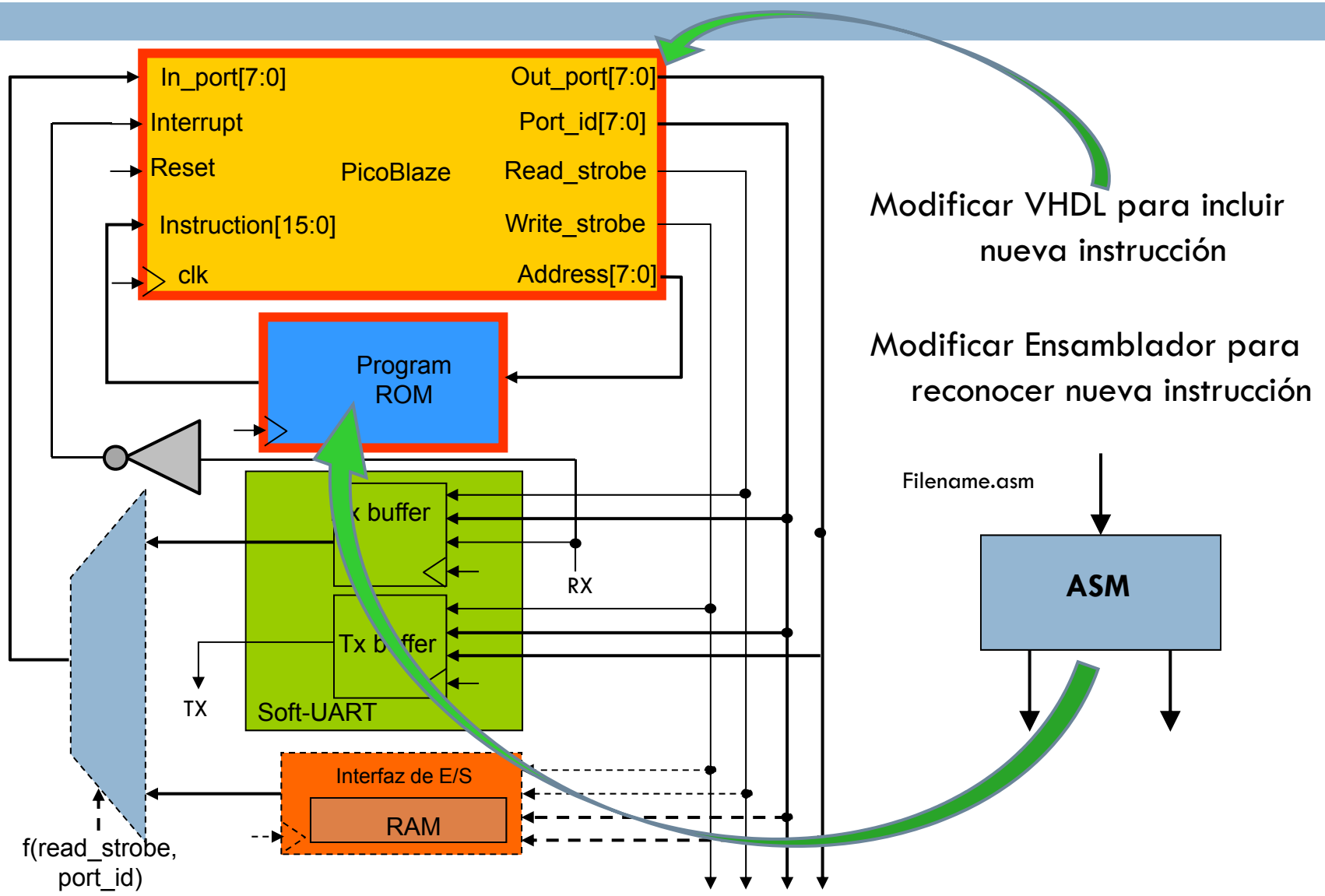
Rutina de interrupción

```
interrup: DISABLE          INTERRUPT
          CALL             recibe
          LOAD             txreg,rxreg
          CALL             transmite
          ADD              S6,30
          LOAD             txreg,S6
          CALL             transmite
          RETURNI          ENABLE
```

Dirreción de salto a Interr.

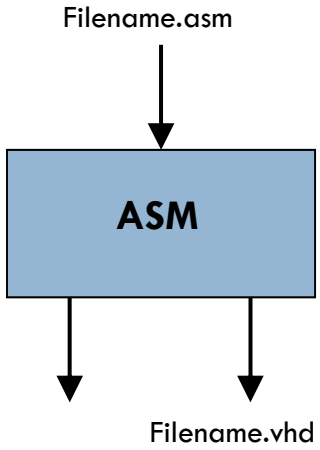
```
          ADDRESS          FF
          JUMP             interrup
```

Modificación-3ª: añadir Instrucción FLIP



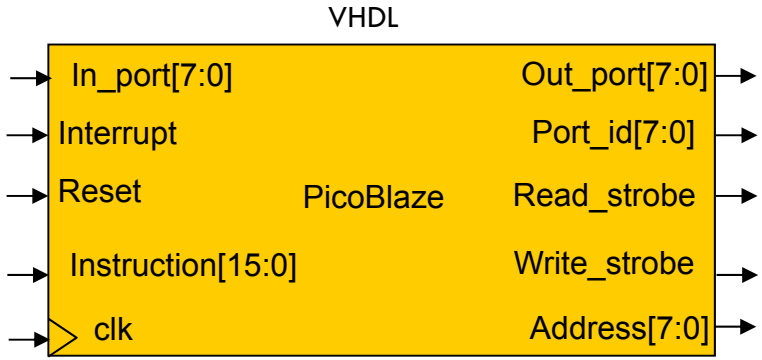
Modificación-3ª: añadir Instrucción FLIP

- Modificar herramienta ensambladora (asm.ccp):
  - Añadir nuevo código de operación.
  - Añadir nuevo nemónico al juego de instrucciones.
  - Incrementar el número de instrucciones reconocibles
  - Añadir instrucción en el parser de chequeo de sintaxis.
  - Añadir instrucción en el parser de decodificación de Código Máquina



Modificación-3ª: añadir Instrucción FLIP

- Modificar código VHDL de PicoBlaze:
  - Añadir código de operación (flip_id = '11111')
  - Declarar el componente y definir el comportamiento de la instrucción
  - Declarar nuevas señales de control (i_flip)
  - Habilitar señales de habilitación de flags del Banco de Registros.
  - Añadir al decodificar de instrucciones de la UC la nueva instrucción.
  - Añadir a la ALU la salida de la nueva instrucción.



Modificación-3ª: añadir Instrucción FLIP

Modificar VHDL

1- Añadir cod. Operación

2- Declarar/instanciar Componente.

3- Añadir señal “i_flip”

4- Habilitar Flag y puertos en “register_and_flag_enable”

5- Añadir decodificador UC.

6- Añadir salida ALU

Rutina de interrupción

```
interrup:  DISABLE          INTERRUPT
           CALL             recibe
           FLIP             rxreg
           LOAD             txreg,rxreg
           CALL             transmite
           ADD              S6,30
           LOAD             txreg,S6
           CALL             transmite
           RETURNI          ENABLE
```