

Departamento de Arquitectura y Tecnología de Computadores



UNIVERSIDAD DE SEVILLA



E.T.S. de Ingeniería Informática
Avda. Reina Mercedes, S/N.
41012 Sevilla, SPAIN



Escuela Universitaria Politécnica
C/ Virgen de África, 7.
41011 Sevilla, SPAIN

Tecnología de Microcontroladores

Sistema empotrado basado en codiseño para el procesamiento de imágenes

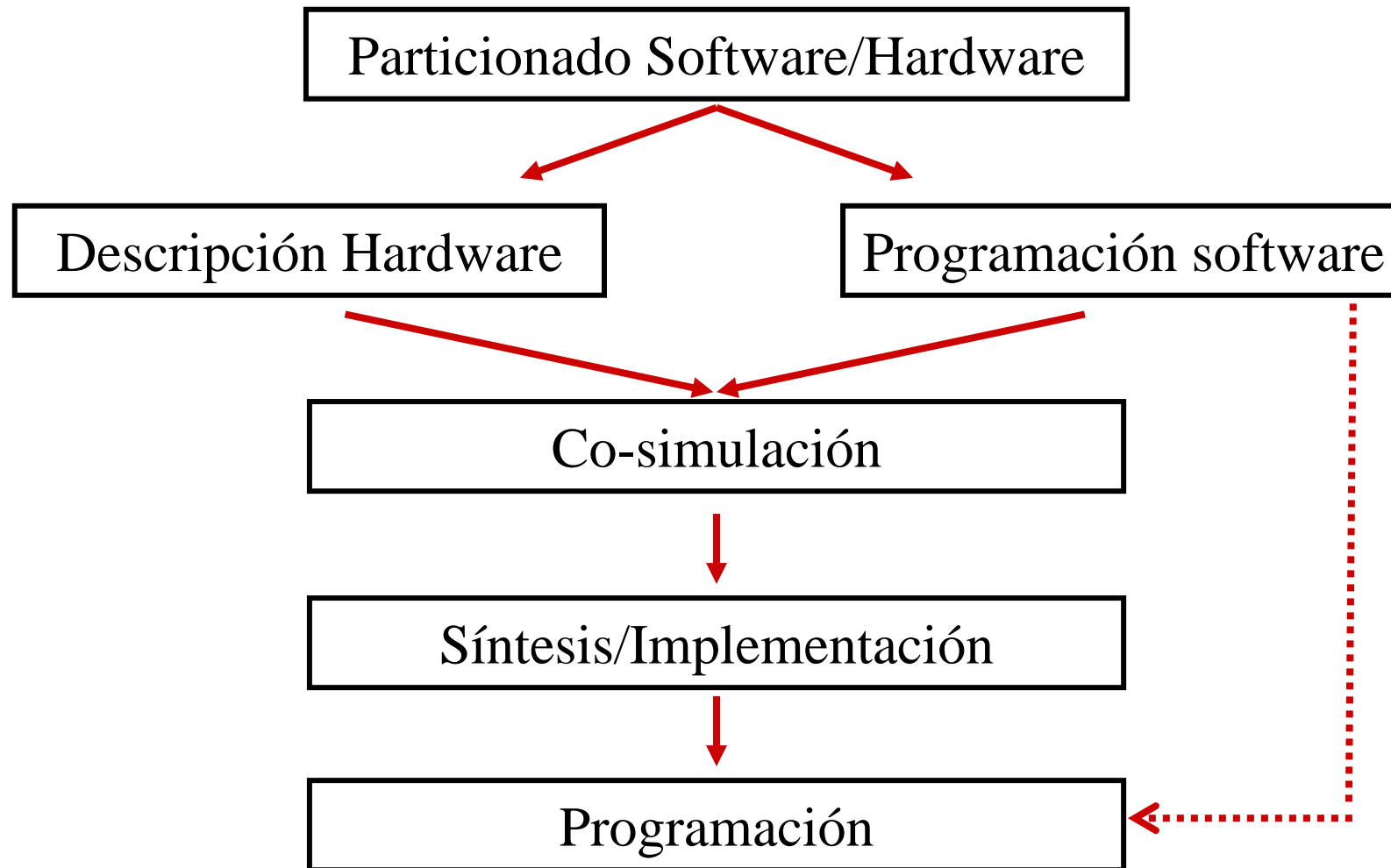
Raouf Senhadji Navarro

Índice

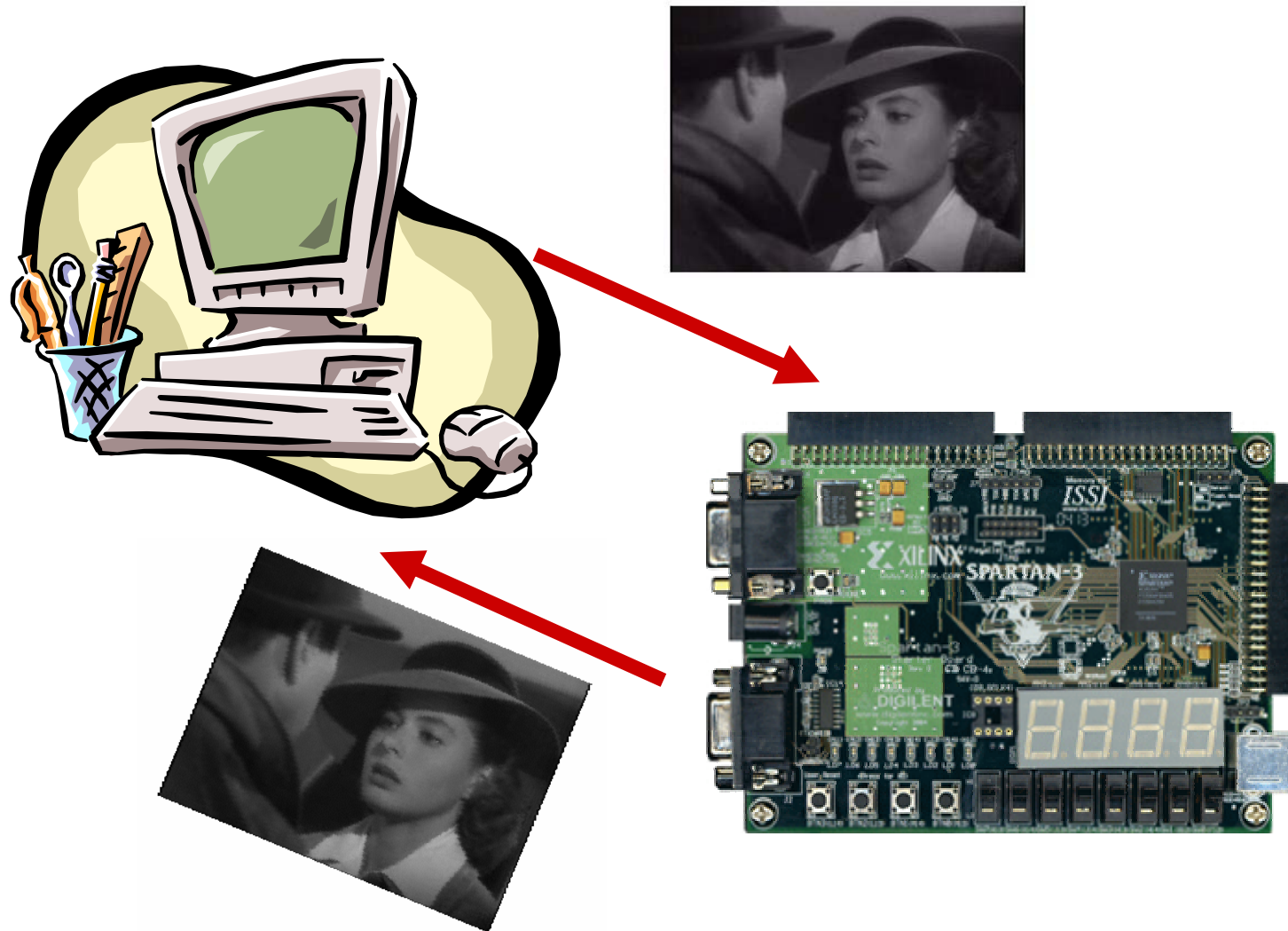
- Introducción
- La aplicación a diseñar
- El hardware
 - La tarjeta de prototipo
 - La UART
 - El CORDIC
 - El 8051
- Las herramientas
 - ISE Foundation
 - El compilador SDCC
 - Utilidades
- Realización
 - Consejos para el diseño
 - Pasos a seguir
 - La memoria a entregar
- Referencias

Introducción al codiseño

Metodología para el desarrollo de sistemas basados en co-diseño sobre FPGAs



La aplicación a diseñar: coprocesador de imágenes



El algoritmo: La matriz de rotación



$$R = \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix}$$

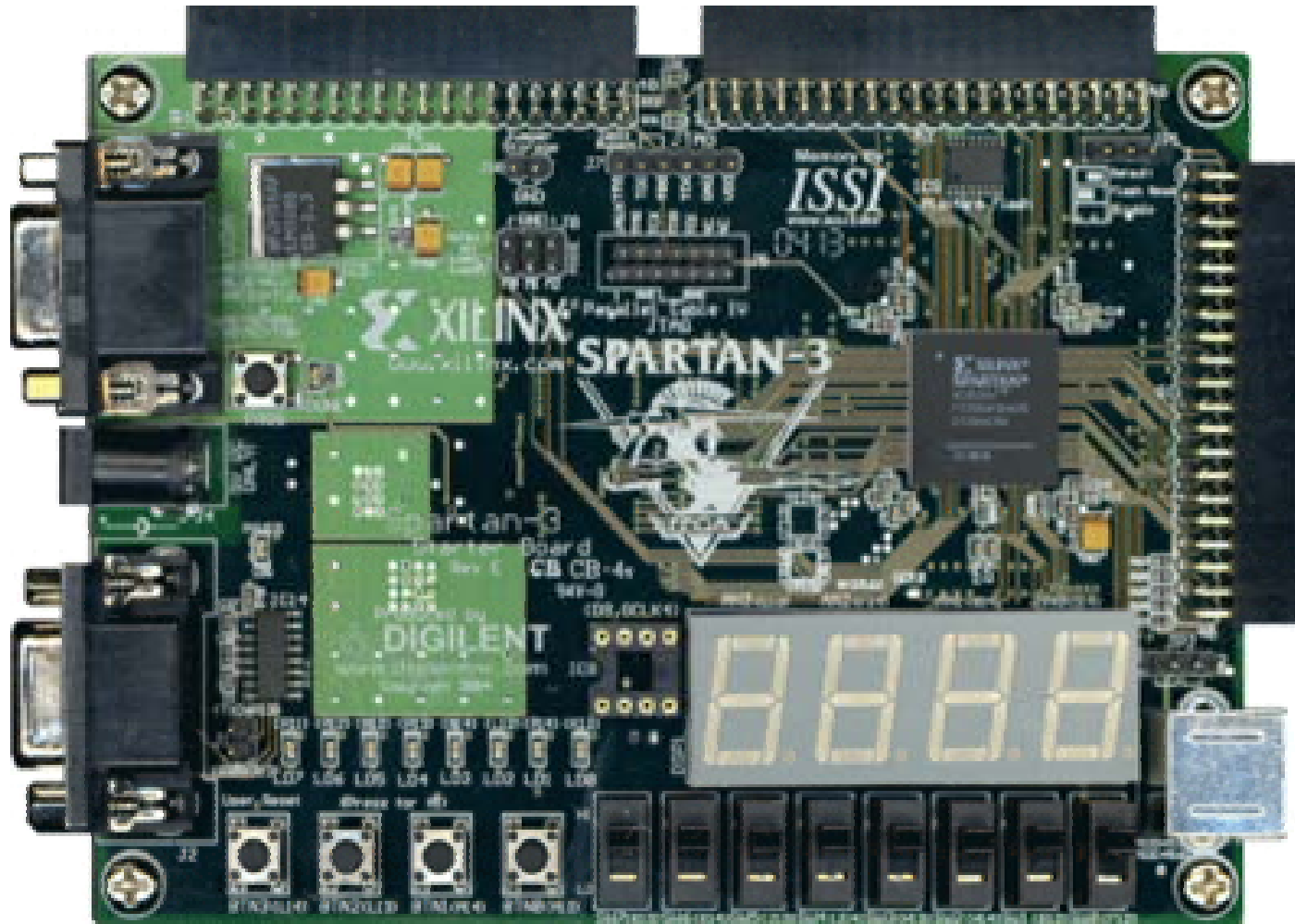
El algoritmo: La descomposición de Paeth



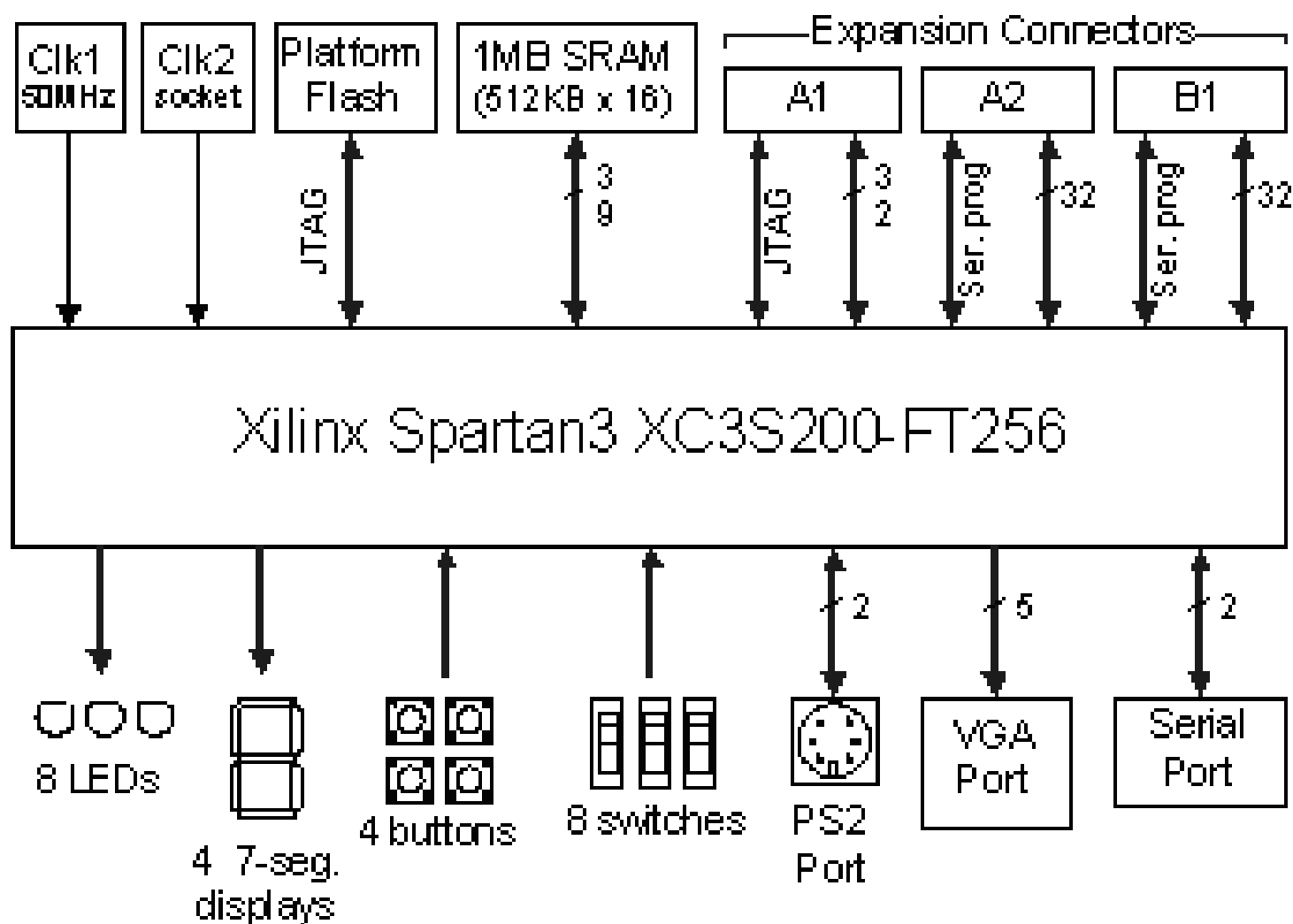
$$R = \begin{pmatrix} 1 & \tan(\frac{\phi}{2}) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\sin \phi & 1 \end{pmatrix} \begin{pmatrix} 1 & \tan(\frac{\phi}{2}) \\ 0 & 1 \end{pmatrix}$$



La tarjeta de prototipo



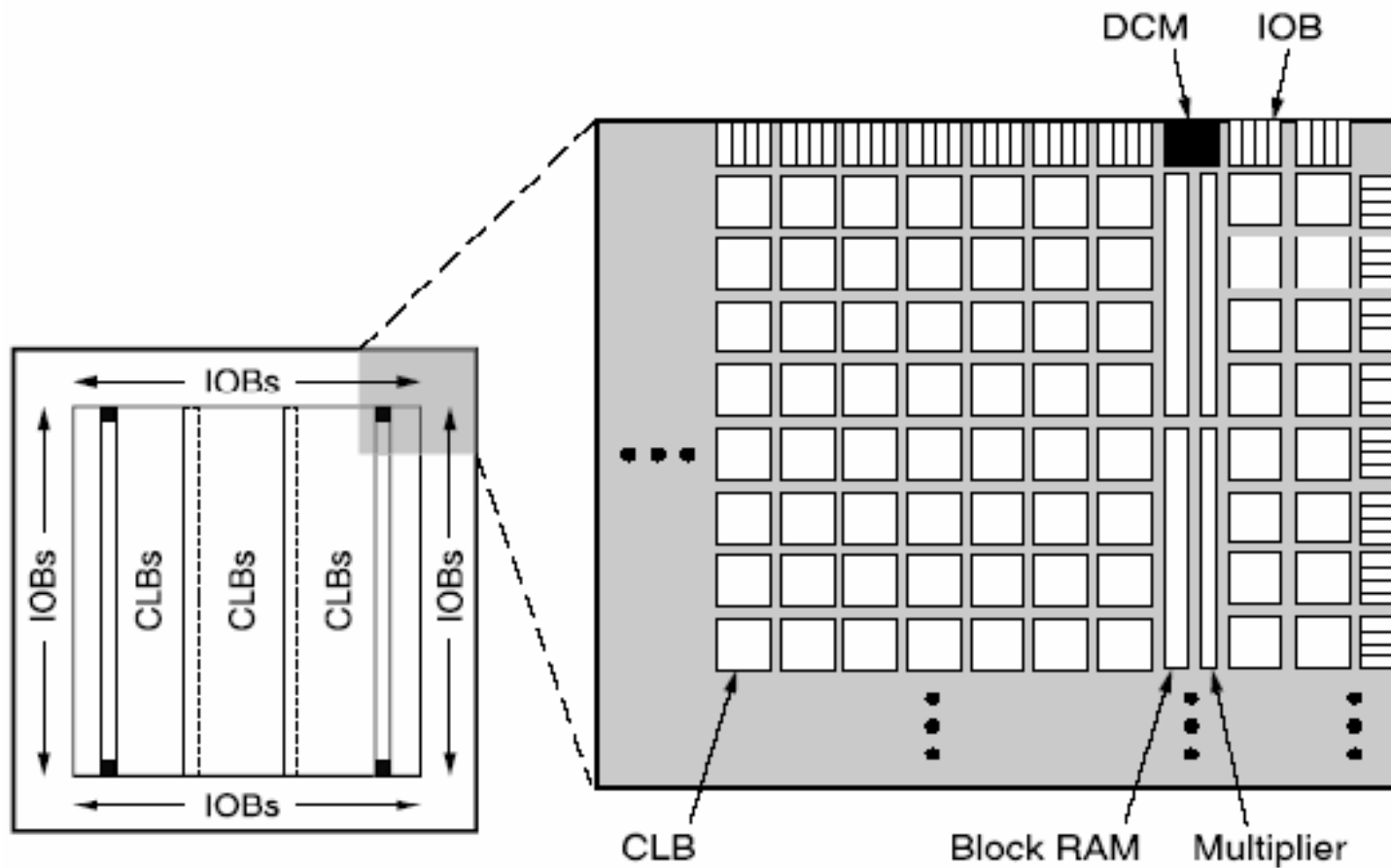
La tarjeta de prototipo



La tarjeta de prototipo: señales a utilizar

Nombre	Descripción	Tipo
AN1	Habilitación del dígito más significativo del <i>display</i>	1 bit de entrada
AN0	Habilitación del dígito menos significativo del <i>display</i>	1 bit de entrada
A B C D E F G	Display <pre> A ***** F * * B * G * ***** E * * C * * ***** D </pre>	1 bit de entrada
BTN3	Botón de RESET	1 bit de salida
BTN2	Botón disponible	1 bit de salida
BTN1	Botón disponible	1 bit de salida
BTN0	Botón disponible	1 bit de salida
LD7	LED disponible	1 bit de entrada

La FPGA Spartan-3



La FPGA Spartan-3

Device	System Gates	Equivalent Logic Cells	CLB Array (One CLB = Four Slices)			Distributed RAM (bits ¹)	Block RAM (bits ¹)	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differential I/O Pairs
			Rows	Columns	Total CLBs						
XC3S50 ²	50K	1,728	16	12	192	12K	72K	4	2	124	56
XC3S200 ²	200K	4,320	24	20	480	30K	216K	12	4	173	76
XC3S400 ²	400K	8,064	32	28	896	56K	288K	16	4	264	116
XC3S1000 ^{2, 3}	1M	17,280	48	40	1,920	120K	432K	24	4	391	175
XC3S1500 ³	1.5M	29,952	64	52	3,328	208K	576K	32	4	487	221
XC3S2000	2M	46,080	80	64	5,120	320K	720K	40	4	565	270
XC3S4000 ³	4M	62,208	96	72	6,912	432K	1,728K	96	4	712	312
XC3S5000	5M	74,880	104	80	8,320	520K	1,872K	104	4	784	344

Modelo VHDL de la UART: señales

Nombre	Descripción	Tipo
Bus paralelo		
WB_CLK_I	Reloj	1 bit de entrada
WB_RST_I	Reset	1 bit de entrada
WB_ADR_I	Dirección	2 bits de entrada
WB_DAT_I	Bus de datos de entrada	8 bits de entrada
WB_DAT_O	Bus de datos de salida	8 bits de salida
WB_WE_I	Escritura/Lectura	1 bit de entrada
WB_STB_I	Inicio de transferencia	1 bit de entrada
WB_ACK_O	Fin de transferencia	1 bit de salida
Interfaz serie		
IntTx_O	Interrupción Buffer de Transmisión Libre (no usada)	1 bit de salida
IntRx_O	Interrupción Dato recibido (no usada)	1 bit de salida
BR_Clk_I	Reloj	1 bit de entrada
TxD_PAD_O	Línea de transmisión RS232	1 bit de salida
RxD_PAD_I	Línea de recepción RS232	1 bit de entrada

Modelo VHDL de la UART: mapa de registros

Nombre	Dirección	Acceso
Buffer de recepción	0	Lectura
Buffer de transmisión	0	Escritura
Estado	1	Lectura
Reservado	2	
Reservado	3	

	RX dato	TX libre
--	---------	----------

Estado

Core del CORDIC: configuración

CORDIC

Parameters Core Overview Contact Web Links

LogiCORE

CORDIC

Component Name

Functional Selection

☒ Rotate ☐ Sin and Cos ☐ Arc Tan ☐ Square Root

☐ Translate ☐ Sinh and Cosh ☐ Arc Tanh

Architectural Configuration

☒ Word Serial ☐ Parallel

Pipelining Mode

☐ No Pipelining ☐ Optimal ☒ Maximum

☐ Display Core Footprint

< Back Next > Page 1 of 4

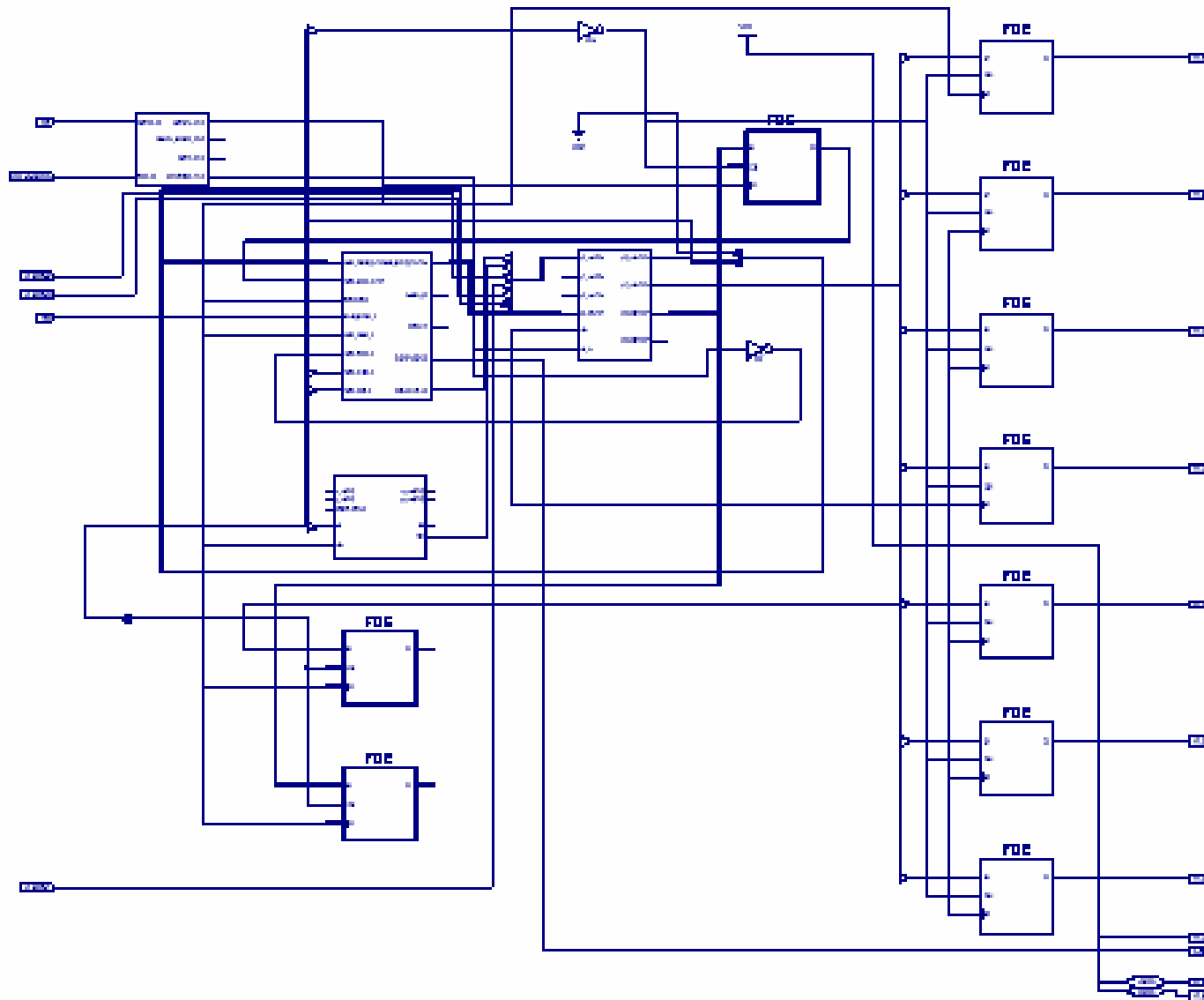
Core del CORDIC: señales

Nombre	Descripción	Tipo
x_in	Coordenada x de entrada	8 bits de entrada
y_in	Coordenada y de entrada	8 bits de entrada
phase_in	Ángulo de rotación	8 bits de entrada
nd	Cuando está a 1 indica nuevo dato de entrada	1 bit de entrada
x_out	Coordenada x de salida	8 bits de salida
y_out	Coordenada y de salida	8 bits de salida
rdy	Cuando está a 1 indica que el dato de salida está listo	1 bit de salida
clk	Reloj	1 bit de entrada

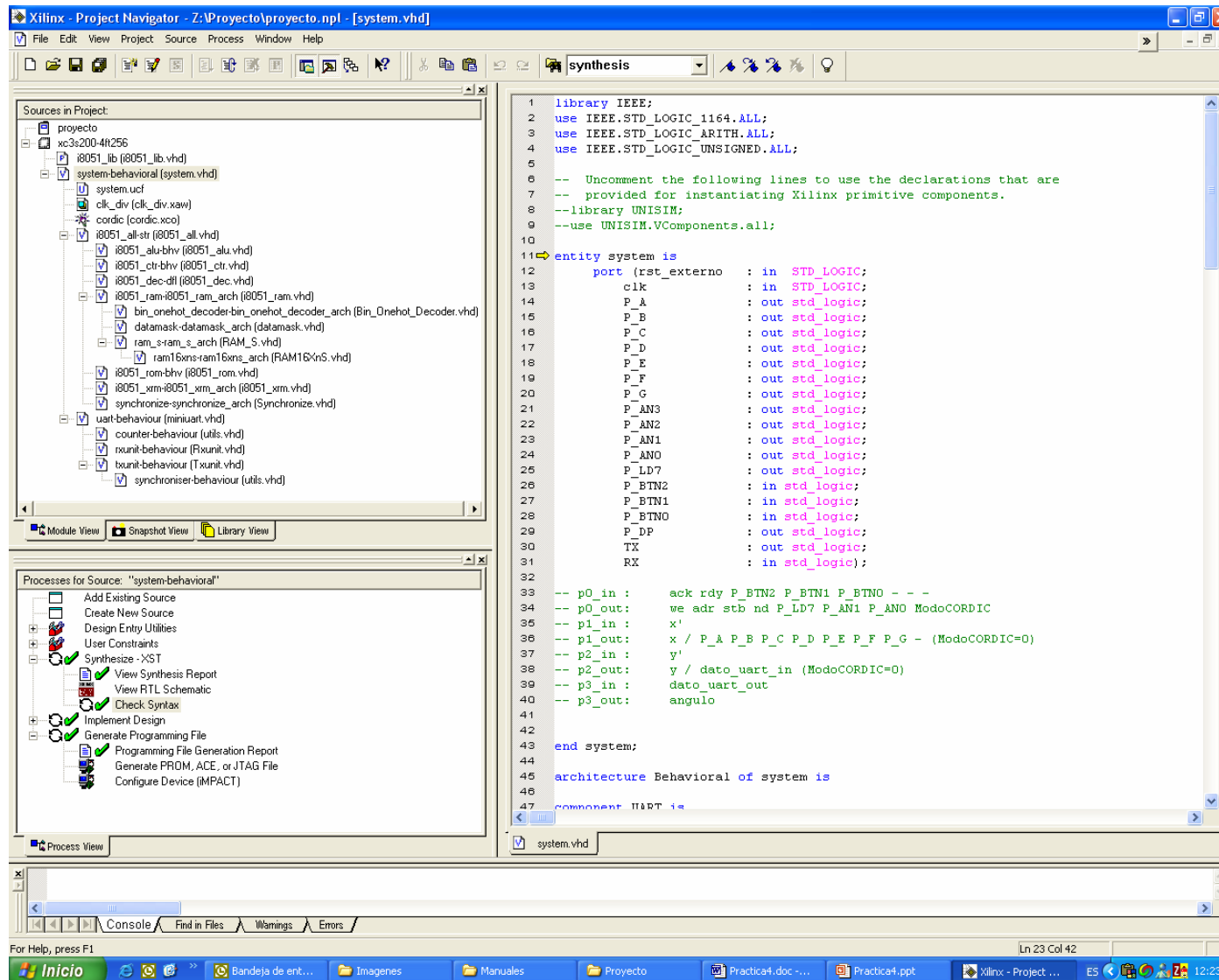
Modelo VHDL del microcontrolador 8051: puertos

Puertos	Bits (7:0)							
Puerto 0 Entrada	WB_ACK_O	rdy	BTN2	BTN1	BTN0	-	-	-
Puerto 1 Entrada	x_out							
Puerto 2 Entrada	y_out							
Puerto 3 Entrada	WB_DAT_O							
Puerto 0 Salida	WB_WE_I	WB_ADR_I(0)	WB_STB_I	nd	LD7	AN1	AN0	MODO_C
Puerto 1 Salida (MODO_C)=0	A	B	C	D	E	F	G	-
Puerto 1 Salida (MODO_C)=1	x_in							
Puerto 2 Salida (MODO_C)=0	WB_DAT_I							
Puerto 2 Salida (MODO_C)=1	y_in							
Puerto 3 Salida	phase_in							

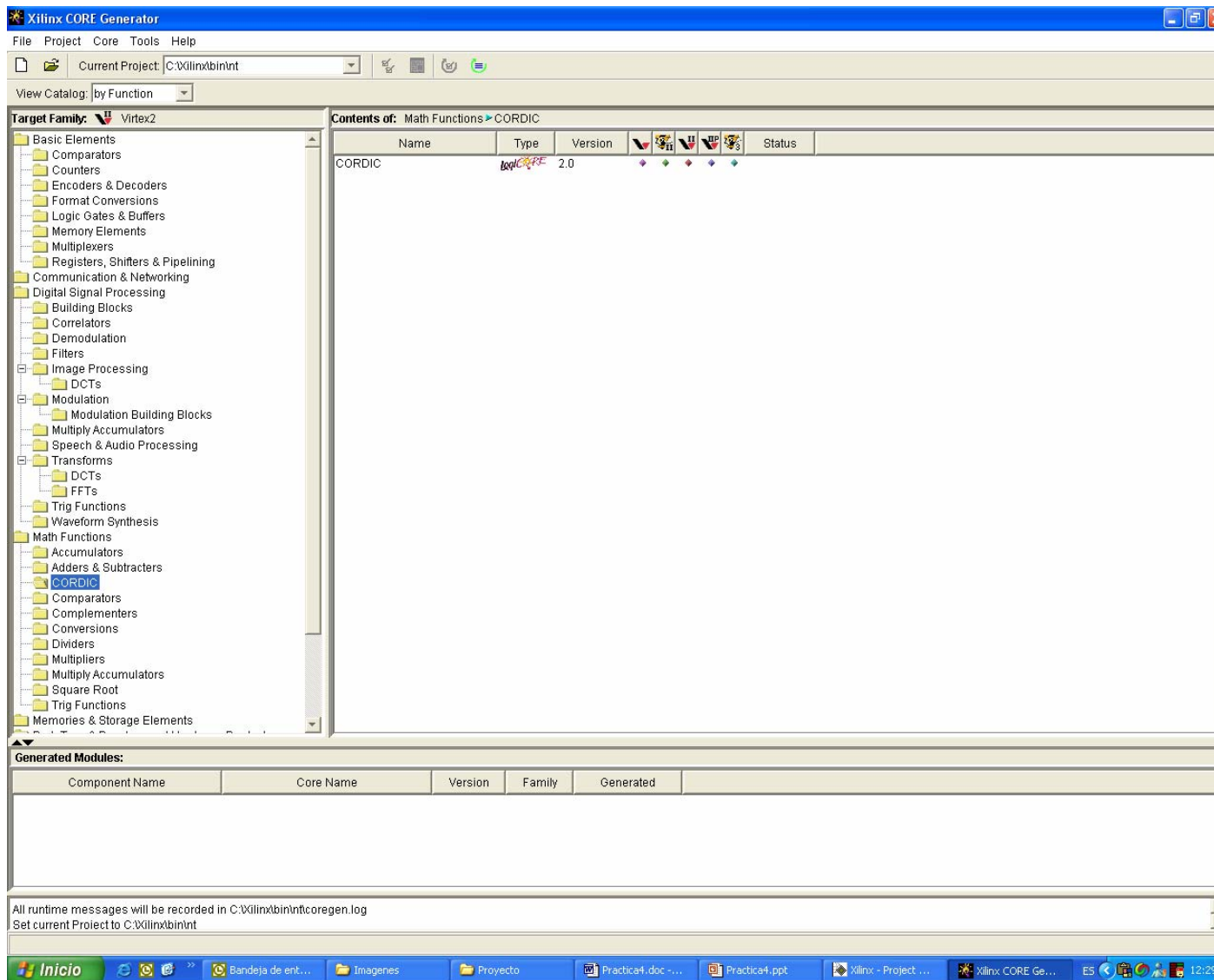
Esquemático del sistema completo



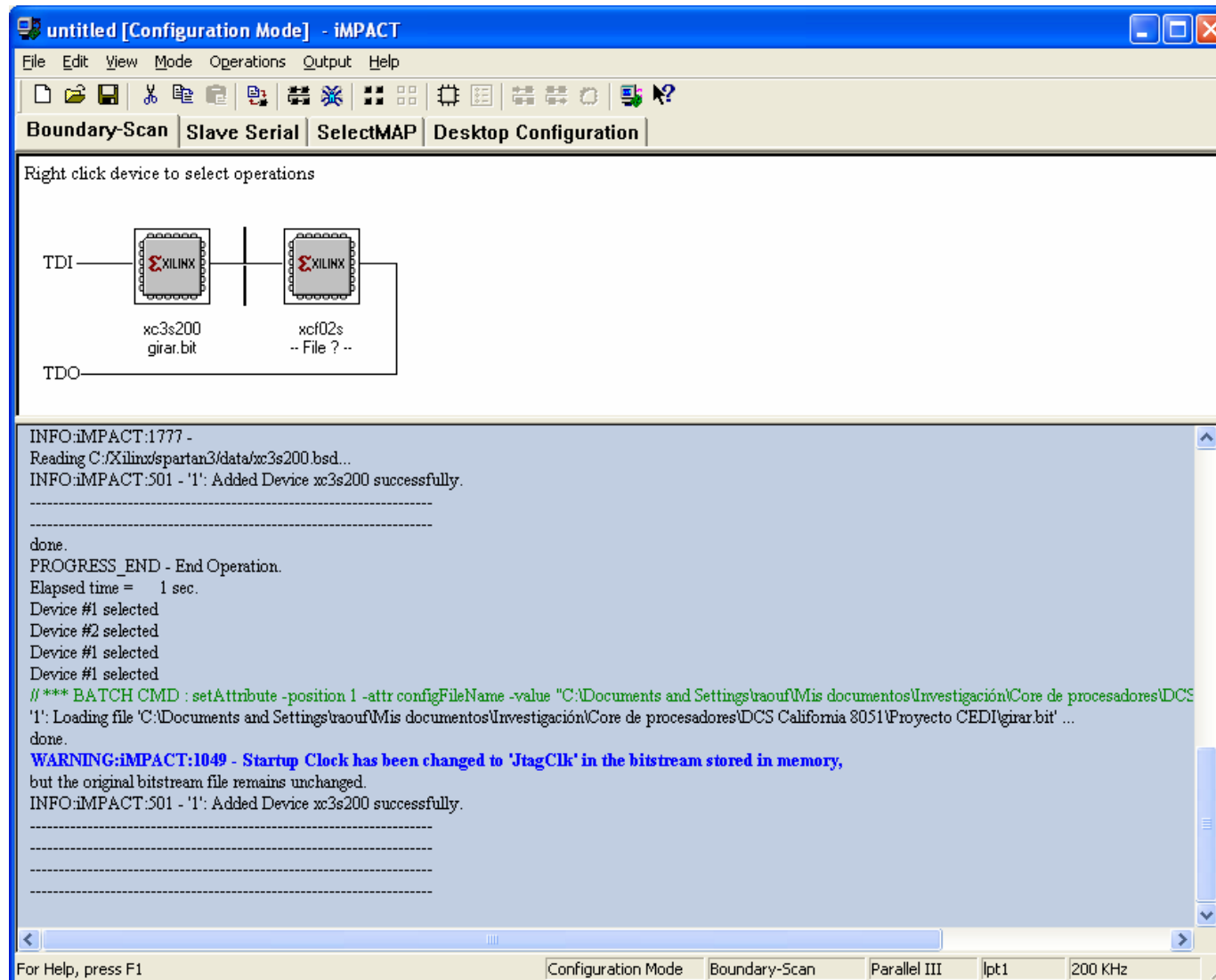
Xilinx ISE Foundation: Project Navigator



Xilinx ISE Foundation: CORE Generator



Xilinx ISE Foundation: IMPACT



El compilador SDCC

- Definición de variables asociadas a los puertos:

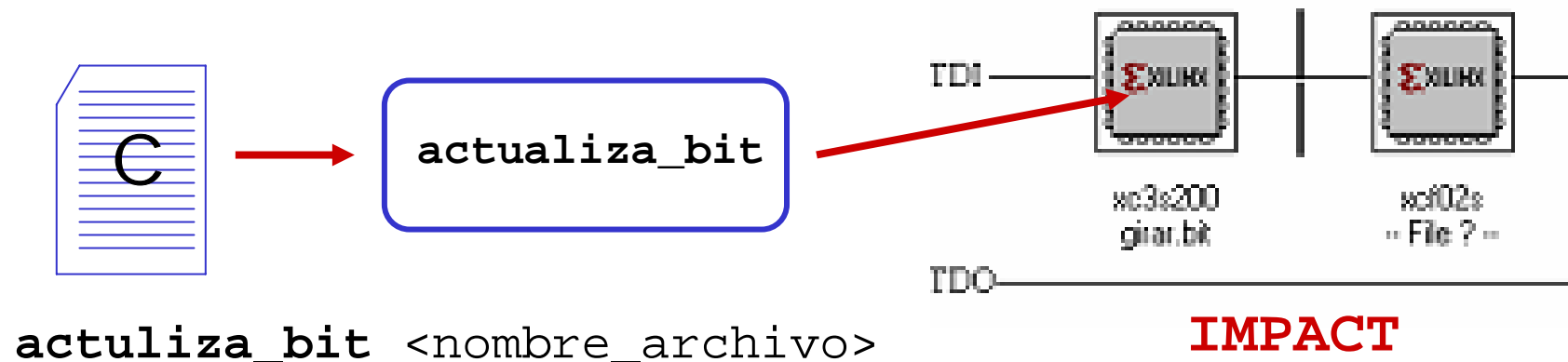
```
bit at 0x83 P0_3; // Bit 3 del puerto 0  
sfr at 0x90 P1; // Puerto 1
```

- Definición de variables en memoria externa:

```
xdata unsigned char dato;
```

Archivo de comandos *actualiza_bit*

```
sdcc --code-size 2048 --xram-size 2560 %1.c  
makebin -s 2048 < %1.ihx > %1.bin  
bin2asciihex %1.bin > %1.mem  
data2mem -bt system.bit -bm rom.bmm -bd %1.mem  
-o b %1.bit
```



`actualiza_bit <nombre_archivo>`

Otros programas

`crea_imagen_bin` <angulo> <nombre_archivo>

`crea_imagen_bmp` <nombre_archivo>



Para que los programas funcionen hay que instalar previamente las librerías adecuadas (mglinstaller)

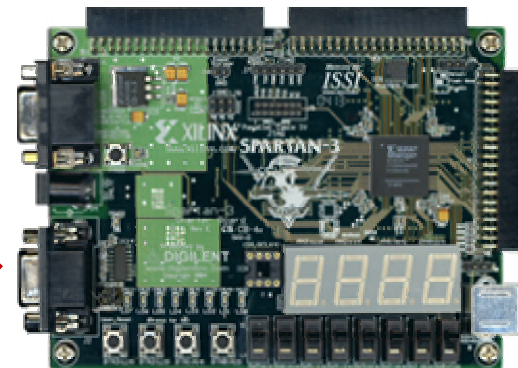


θ

`crea_imagen_bin`

Hyper Terminal
(Windows)

`crea_imagen_bmp`



Consejos para la programación de la aplicación

1. Definir variables asociadas a todos los puertos (puede haber más de una variable asociada al mismo puerto: por ejemplo *BTN2* y *nd* para el bit 5 del puerto 0 o *WB_DATA_I* y *y_in* en el puerto 2)
2. Definir alguna macro para mostrar valores en el *display* para realizar la depuración
3. Para probar la comunicación serie puede enviar caracteres hacia la tarjeta mostrando su código ASCII (de dos dígitos) en el *display* y recibir un carácter de una secuencia predefinida (A,B,C, ...) cada vez que pulse un botón (BTN0).
4. Para probar el funcionamiento del módulo CORDIC puede fijar unos valores de entrada y mostrar el resultado en el *display*.
5. Por restricciones de tamaño de la memoria interna del 8051 la imagen debe almacenarse en la memoria externa. Para evitar el uso de dos *arrays* del tamaño de la imagen hay que ir rotando las coordenadas conforme se van recibiendo los datos de la imagen.

Pasos para la realización de la práctica

1. Cree un proyecto, añada todas las especificaciones hardware y genere el archivo *system.bit*. Puede editar los archivos VHDL, el CORDIC (.xco), el divisor de reloj (.xaw) y el archivo de restricciones (.ucf)
2. Pruebe la tarjeta de prototipo, los programas para enviar y recibir la imagen y la comunicación serie programando la FPGA con el archivo *girar.bit* (configuración de la FPGA con el programa correcto)
3. Instale el compilador SDCC
4. Utilice el archivo de comandos *actualiza_bit*, el *sistem.bit* generado (también se suministra), los programas C que implemente y el IMPACT para modificar la configuración de la FPGA
5. Realice los programas que se sugieren en el apartado de “consejos para la programación de la aplicación”

La memoria a entregar

- Describir el proceso seguido e incorporar los programas diseñados.
- Contestar al siguiente cuestionario:
 1. ¿Cuál es la frecuencia de reloj a la que funciona el 8051?
 2. Indicar qué pasos deberíamos seguir si quisiéramos cambiar la velocidad a la que se transmiten y reciben datos por el puerto serie.
 3. ¿Si quisiéramos añadir al diseño alguno de los conmutadores de los que dispone la placa (por ejemplo el SW7) que deberíamos hacer? ¿Y para otro LED (por ejemplo el LD6)?
 4. ¿Por qué el archivo de restricciones incluye directivas de colocación para la memoria ROM?
 5. ¿Cuál es el tamaño máximo de código que permite el 8051 implementado y qué valor tiene por defecto? Como puede modificarse y hasta que valor puede alcanzarse en la FPGA utilizada. Hacer los mismos cálculos para la memoria externa.
 6. Explica las diferencias entre el código ensamblador generado para el acceso a variables internas y externas.
 7. Describir el procedimiento a seguir si en vez de utilizar la matriz de rotación utilizáramos la descomposición de Paeth para evitar el efecto de la discretización (píxeles sin rellenar)

Referencias

1. Hardware/Software CO-Design: Principles and Practice, Wayne Wolf and Jorgen Staunstrup, Kluwer Academic Publishers
2. <http://www.digilentinc.com>
3. <http://www.xilinx.com>
4. <http://www.cs.ucr.edu/~dalton/i8051/i8051syn>
5. <http://www1.mmu.edu.my/~khkoay/8051core>
6. <http://www.opencore.org>
7. A.W. Paeth, A fast algorithm for general raster rotation. In Graphics Interface'86, pp.77-81,196