

Depuración de errores

Diseño Lógico 2

Instituto de Ingeniería Eléctrica

Facultad de Ingeniería

Universidad de la República



Inspirado en post de John Regehr | <http://blog.regehr.org/archives/199>

Diálogo entre estudiantes

- No funciona
- A ver, cambia esa parte
- No funciona
- Proba escribirlo así
- ¡No funciona y dejó de andar lo que andaba!
- Borra todo y arranquemos de nuevo
- Ahora funciona... ¡¿ia veces!?!



WIKIPEDIA

La depuración es un **PROCESO METÓDICO** de encontrar y reducir el número de errores o defectos, en un programa o pieza electrónica, para que **FUNCIONE COMO SE ESPERA.**

¿porqué es difícil?

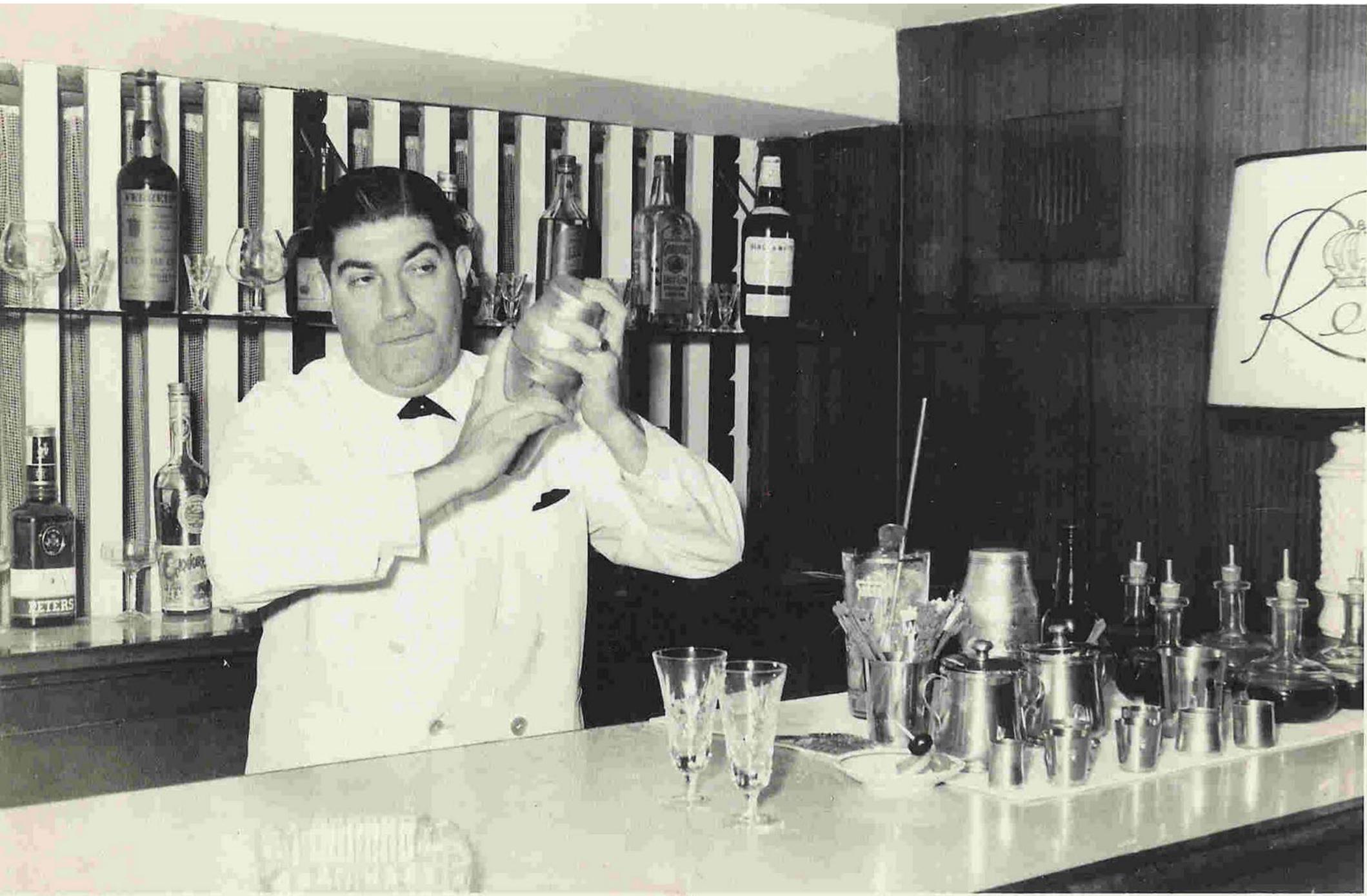
Es un problema inverso:

Datos → Parámetros del modelo

Se necesitan más entradas/estimulos para entender el problema y llegar a una solución única.

¿porqué es difícil?

- La plataforma no es determinística
- Reproducir el bug lleva mucho tiempo
- Los síntomas y el bug no son simultáneos
- Error parte de hipótesis de trabajo errónea
- El problema está fuera de mi alcance



Enfoque científico

1. Asegurarse de que existe un bug

Impresindible:

saber cual es el comportamiento esperado

Fácil: los leds deberían de encenderse

Difícil: algunas salidas están retrazadas

2. Estabilizar, aislar y minimizar

- a) intentar llegar a un sistema determinístico:
aislar la entrada (o secuencia) que genera el error

- b) acotar la búsqueda a los bloques involucrados
(focalizar la búsqueda)

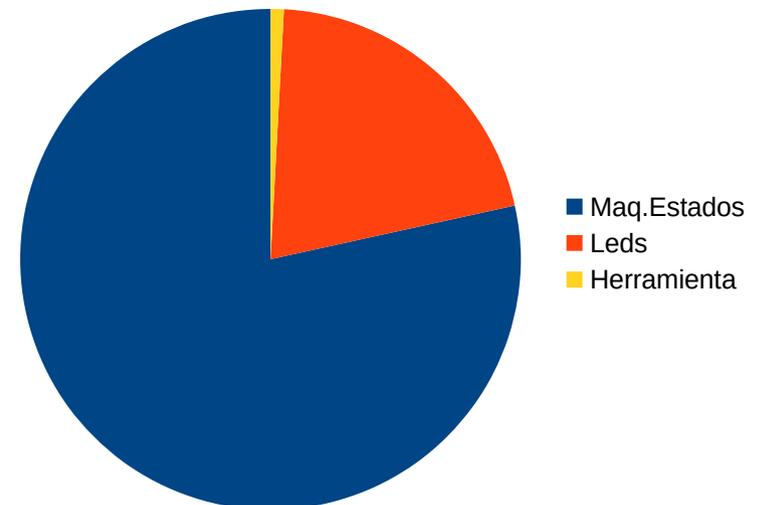
- c) acortar los tiempos necesarios para que se produzcan los síntomas

3. dist. de probabilidad de error

Hay que ADIVINAR: intentemos determinar donde es más probable que este.

Leds no se encienden:

1. Maquina estados genera mal la salidas
2. Salidas activas por nivel opuesto
3. Versión beta de herramienta



4. Experimentos

Realizar experimentos que nos permitan descartar o dividir el origen del error.

Ej:

- realizar una simulación de la FSM
- invertir las salidas
- utilizar una versión anterior de la herramienta

5. Iterar

6. Arreglar el bug

is in romper otras cosas!

7. Limpiar el diseño

8. Agregar un test

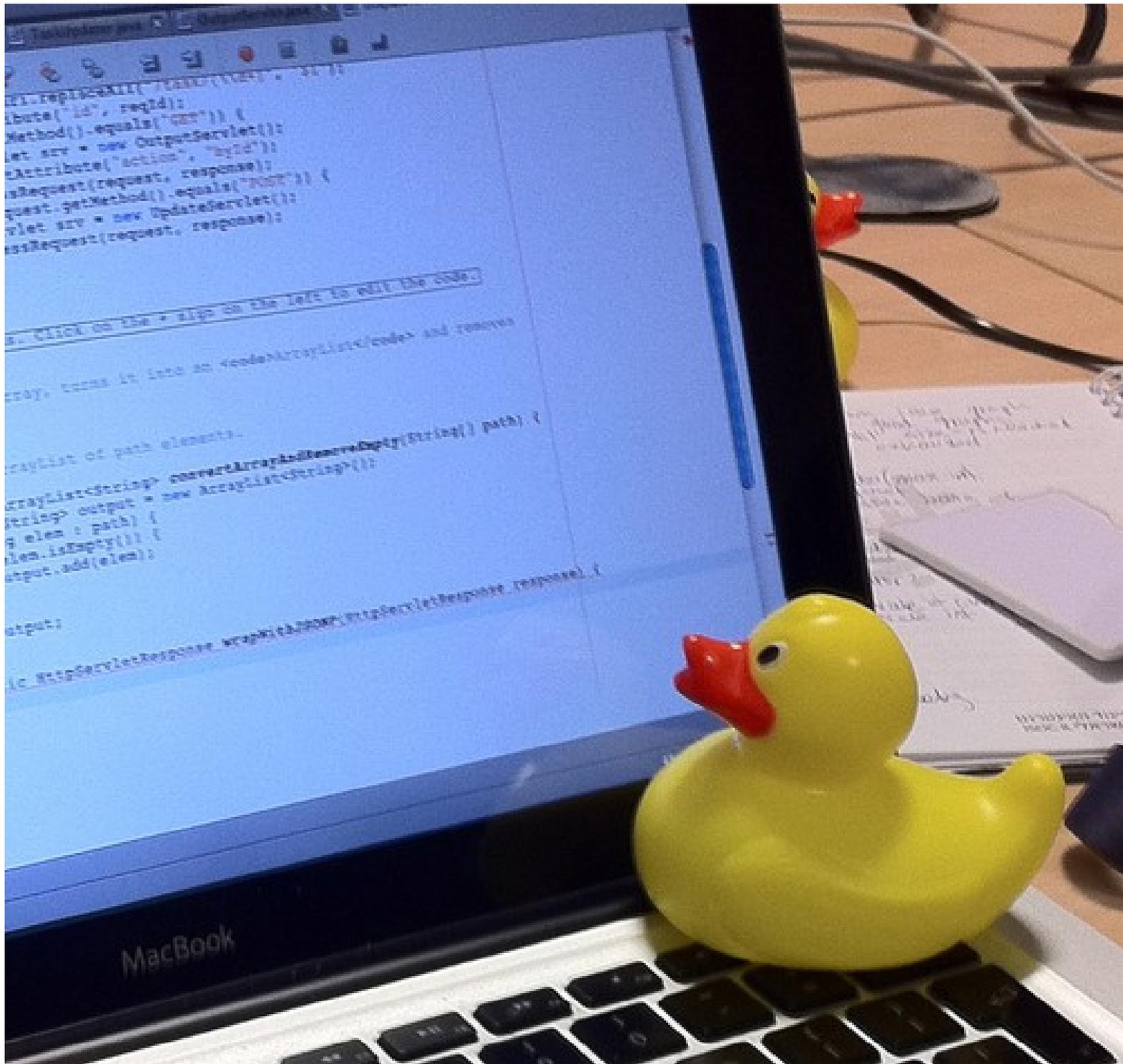
Para detectar error en futuras versiones

En resumen

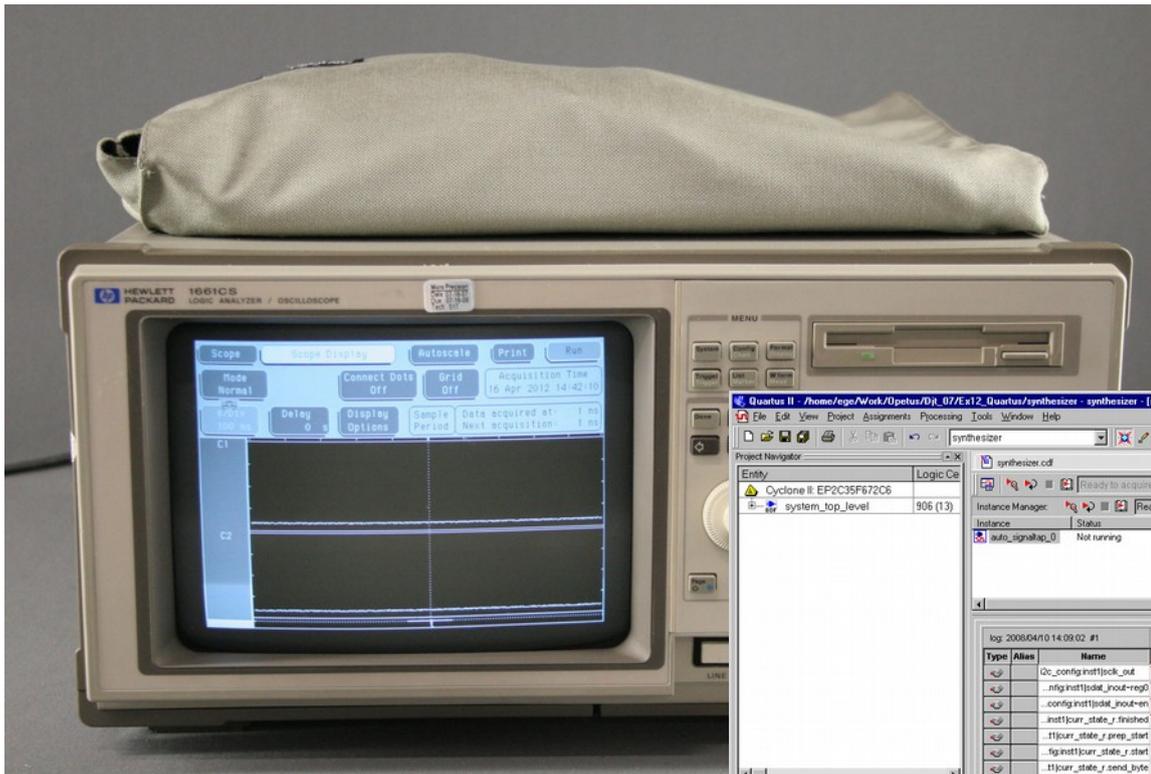
1. Verificar existencia de bug
2. Estabilizar aislar y minimizar
3. Distribución de probabilidad de error
4. Experimentos
5. Iterar 3 y 4
6. Arreglar error
7. Dejar todo en orden
8. Guardar test

¿Y si no se por donde empezar?

- Usar herramientas que me permitan mirar más adentro
- Revisar logs y warning con cuidado
- Contarle el problema a un tercero



Herramientas



Project Navigator

- Entity: Cyclone II EP2C35F672C6
- Logic Cell: system_top_level (906 (13))

Instance Manager

Instance	Status	LEs	Memory	M512/LUTRAM	M4K/M9K	M-RAM/M144K	JTAG Chain Configuration
auto_signalap_0	Not running	571 cells	11264 bits	0 blocks	3 blocks	0 bloc	JTAG ready

Status

Module	Progress %	Tim
Full Compilation	100%	00.0
- Analysis & Synthesis	100%	00.0
- Filter	100%	00.0
- Assembler	100%	00.0
- Classic Timing Analyzer	100%	00.0

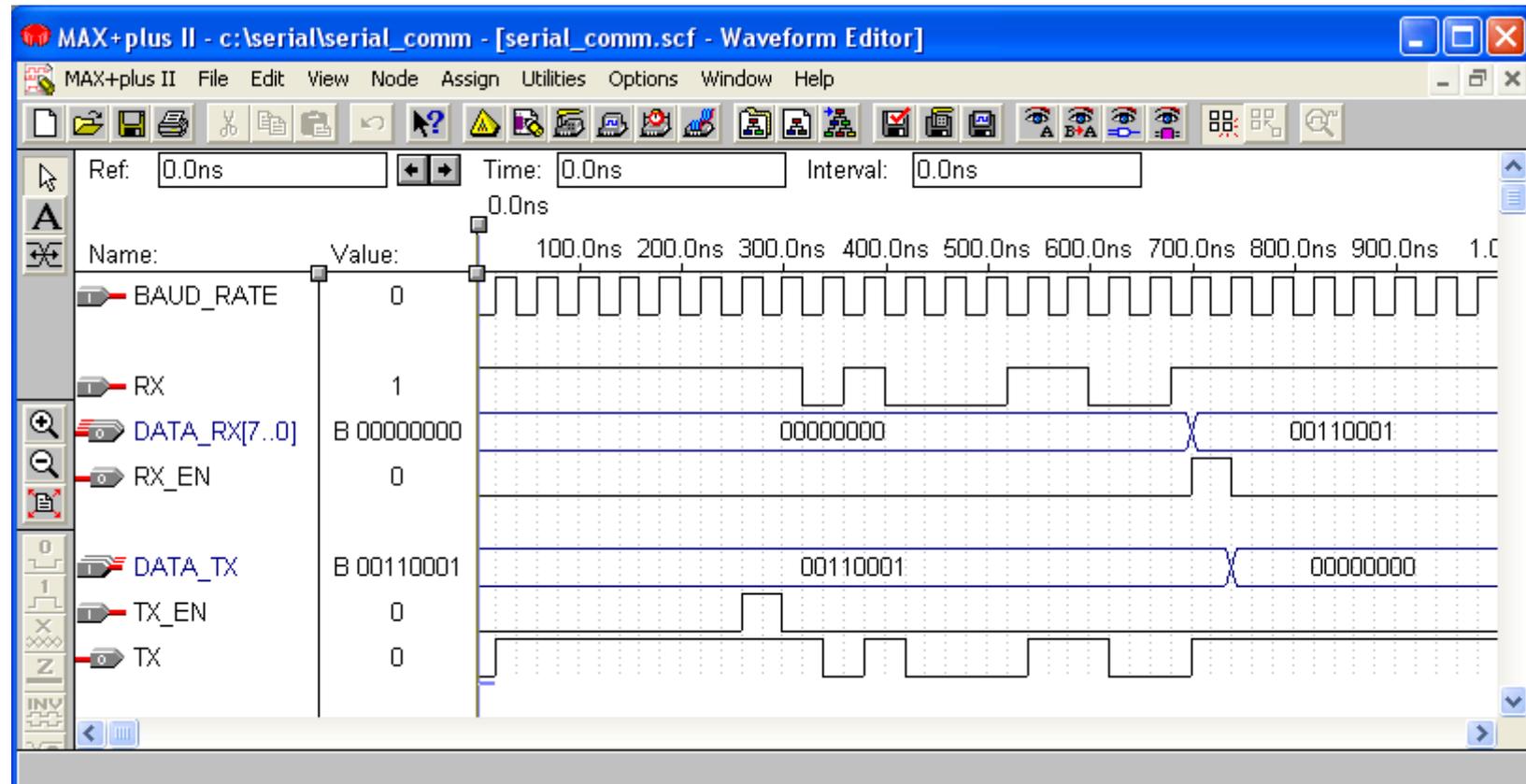
Timing Diagram

Type	Alias	Name
Signal	ic_config_inst[lock_out	ic_config_inst[lock_out
Signal	..nig_inst[tdid_inout-reg0	..nig_inst[tdid_inout-reg0
Signal	config_inst[tdid_inout-en	config_inst[tdid_inout-en
Signal	inst[our_state_j_finished	inst[our_state_j_finished
Signal	..t[our_state_j_prep_start	..t[our_state_j_prep_start
Signal	..fg_inst[our_state_j_start	..fg_inst[our_state_j_start
Signal	..t[our_state_j_send_byte	..t[our_state_j_send_byte
Signal	..st[our_state_j_prep_ack	..st[our_state_j_prep_ack
Signal	..st[our_state_j_read_ack	..st[our_state_j_read_ack
Signal	..t[our_state_j_prep_stop	..t[our_state_j_prep_stop
Signal	..fg_inst[our_state_j_stop	..fg_inst[our_state_j_stop

Message Window

```
Info: Started Programmer operation at Thu Apr 10 13:53:03 2008
Info: Configuring device index 1
Info: Device 1 contains JTAG ID code 0x020B40D
Info: Configuration succeeded -- 1 device(s) configured
Info: Successfully performed operation(s)
```

Herramientas



If - elsif

```
16 architecture behav of case_incompleto is
17 begin
18
19 gen_salida: process (entrada)
20 begin
21     if (entrada = 0) then
22         salida <= '1';
23     elsif (entrada = 1) then
24         salida <= '0';
25     end if;
26 end process;
27
28 end behav;
29
```

Messages

Type	Message
Info	Info: Command: quartus_map --read_settings_files=on --write_settings_files=off case_incompleto -c case_incompleto
Info	Info: Found 2 design units, including 1 entities, in source file //VBOXSVR/sebfer/Documents/00_work/facultad/dl2/curso/2012/clases/debug/case_incompleto
Info	Info: Elaborating entity "case_incompleto" for the top level hierarchy
Warning	Warning (10631): VHDL Process Statement warning at case_incompleto.vhd(19): inferring latch(es) for signal or variable "salida", which holds its previous value
Info	Info (10041): Inferred latch for "salida" at case_incompleto.vhd(19)
Info	Info: Implemented 14 device resources after synthesis - the final resource count might be different
Info	Info: Quartus II Analysis & Synthesis was successful. 0 errors, 1 warning

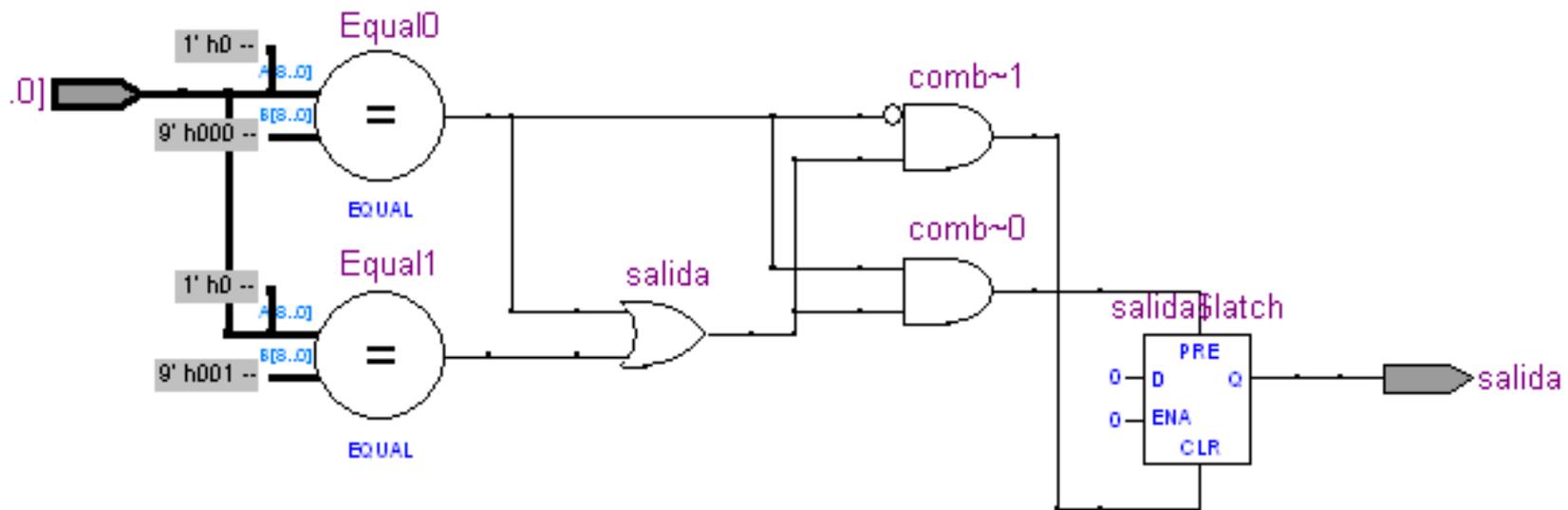
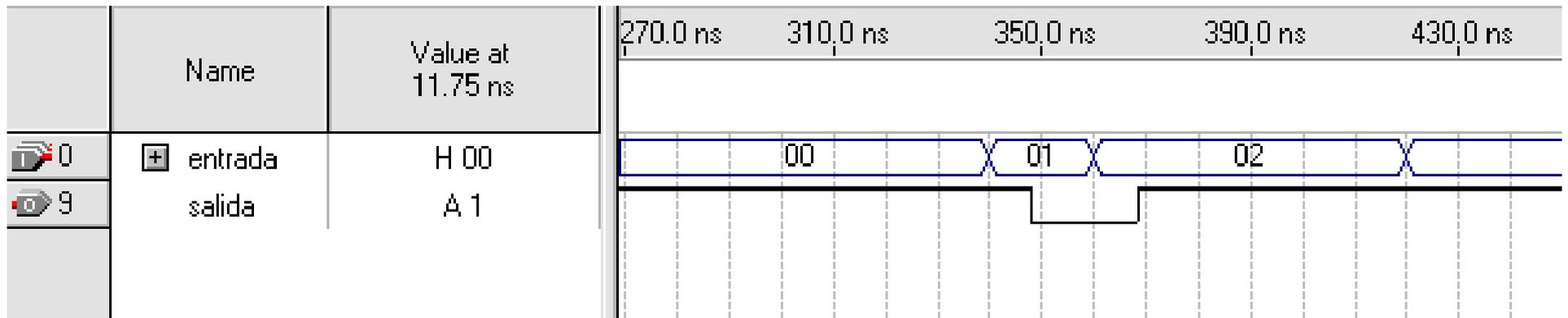
System (14) Processing (9) Extra Info Info (8) Warning (1) Critical Warning Error Suppressed Flag

Message: 0 of 20 Location: Locate

For Help, press F1 Location: Ln 28, Col 1 Idle

Warning: ... Inferring latch(es) for signal or variable "salida"

If - elsif



If - else

