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... i¿ia veces!?!

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 dificil?

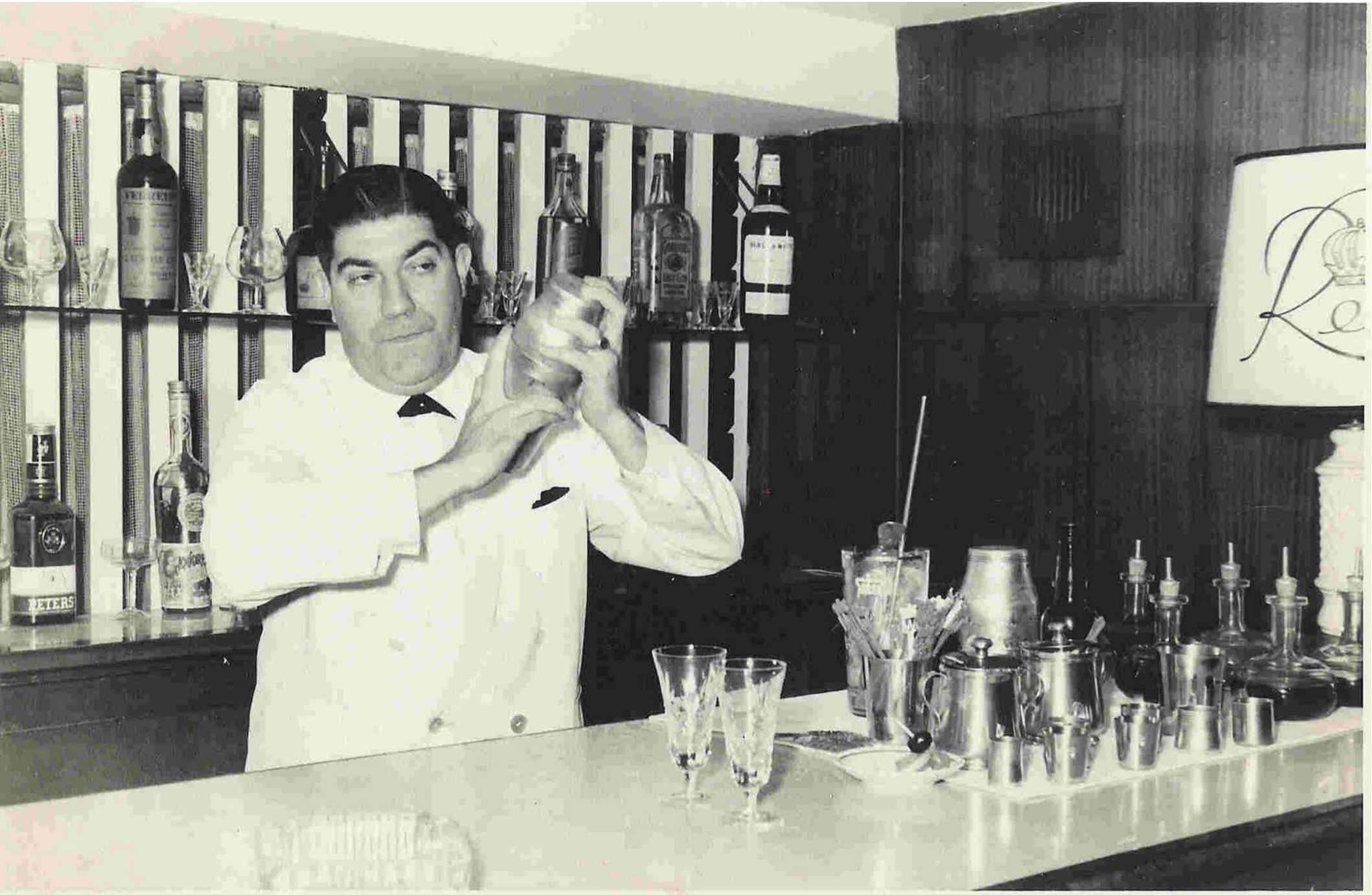
Es un problema inverso:

Datos → Parámetros del modelo

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

¿porqué es dificil?

- La plataforma no es deterministica
- Reproducir el bug lleva mucho tiempo
- Los sintomas y el bug no son simultaneos
- Error parte de hipótesis de trabajo erronea
- El problema esta fuera de mi alcance


PETERS

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 deterministico: aislar la entrada (o secuencia) que genera el error

b) acotar la busqueda a los bloques involucrados (focalizar la busqueda)

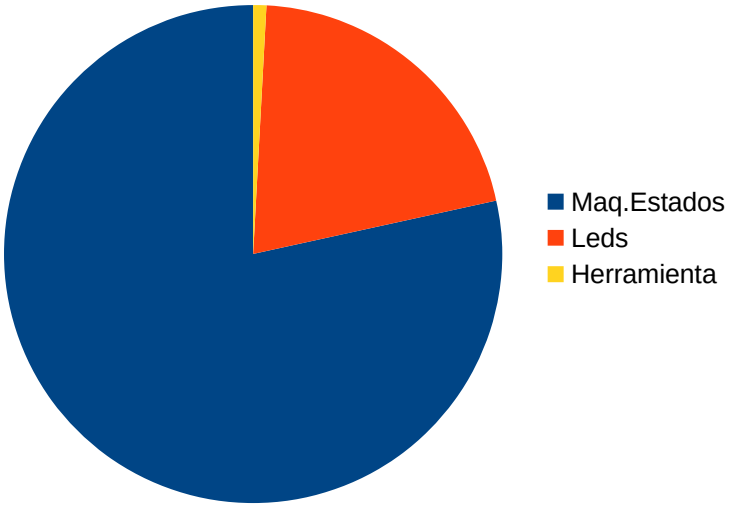
c) acortar los tiempos necesarios para que se produzcan los sintomas

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 pemitan 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

¡sin romper otras cosas!

7. Limpiar el diseño

8. Agregar un test

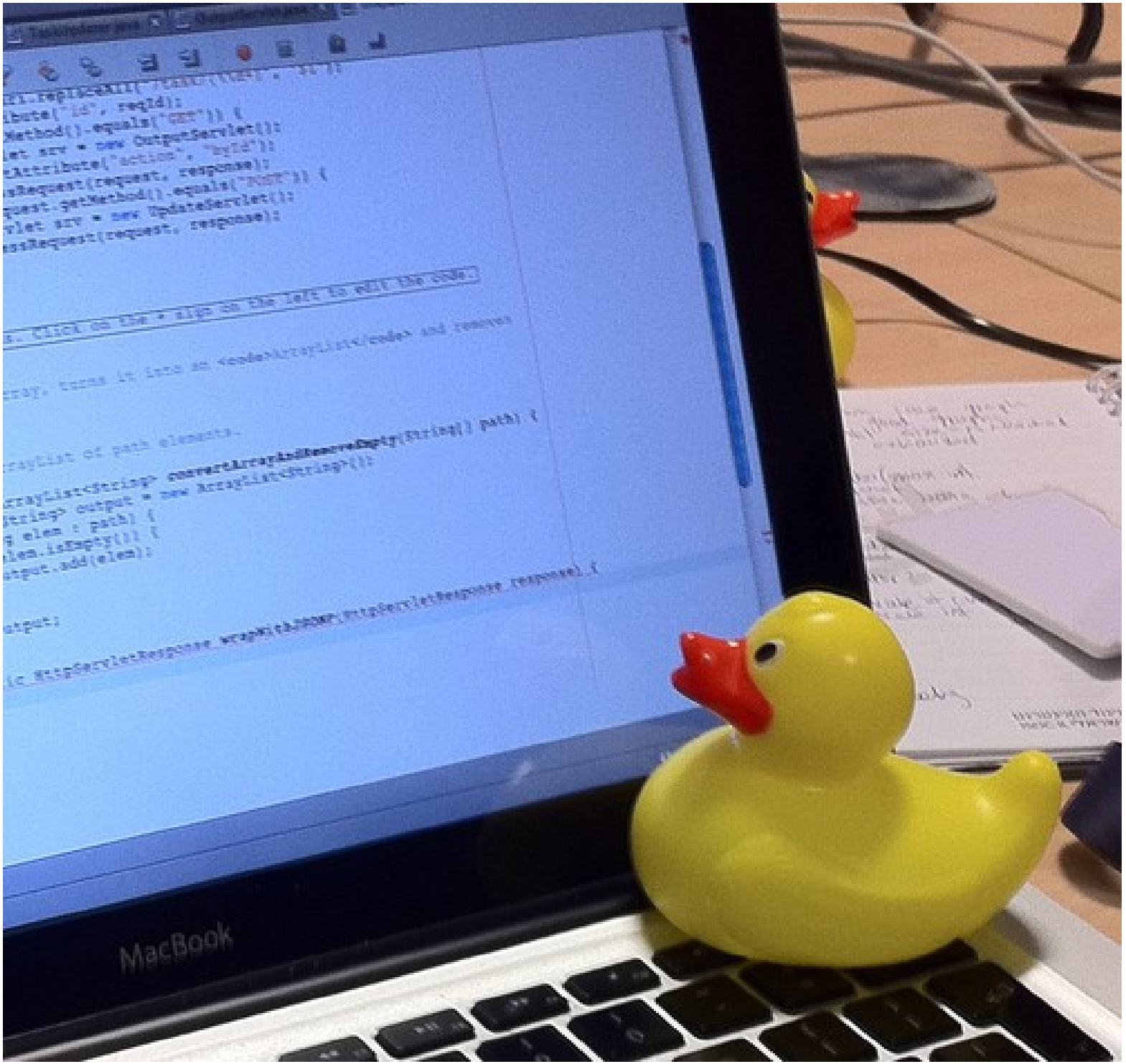
Para detectar error en futuras versiones

En resumen

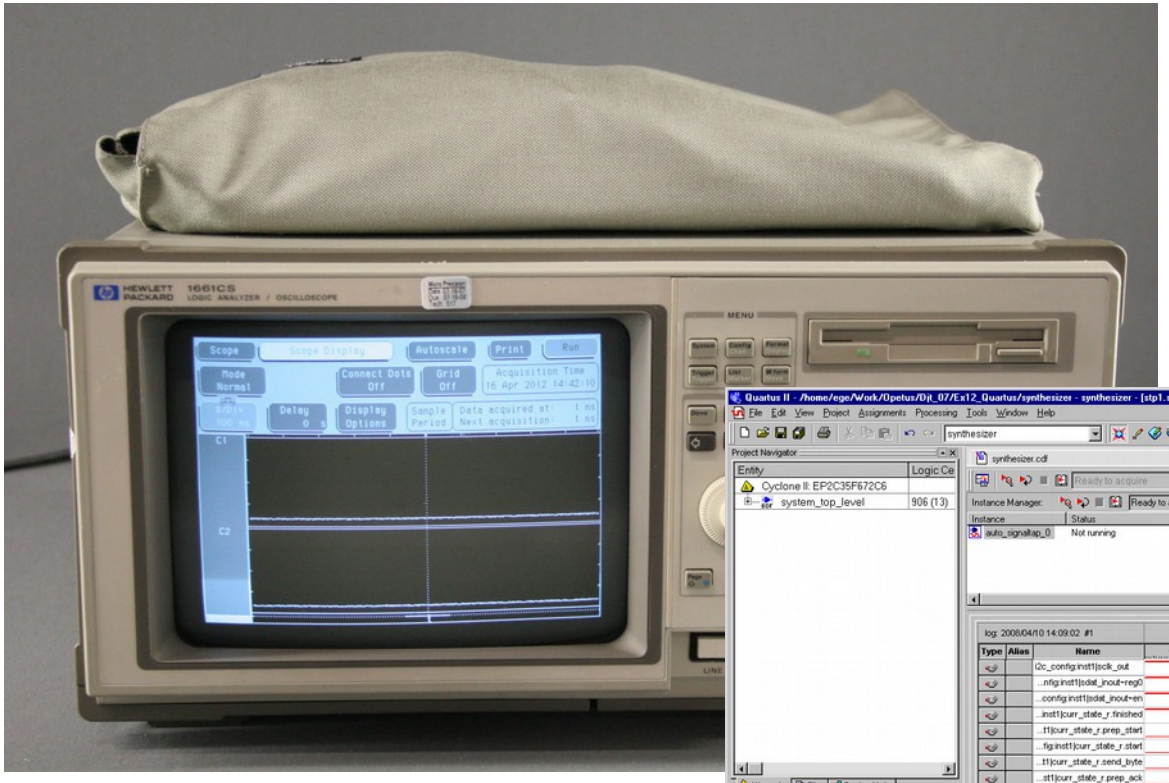
1. Verificar existencia de bug
2. Estabilizar aislar y minimizar
3. Distribución de probabilidad de error
4. Experimentos
5. Iterar 3 y 4
6. Arreglar error
7. Dejar todo en orden
8. Guardar test

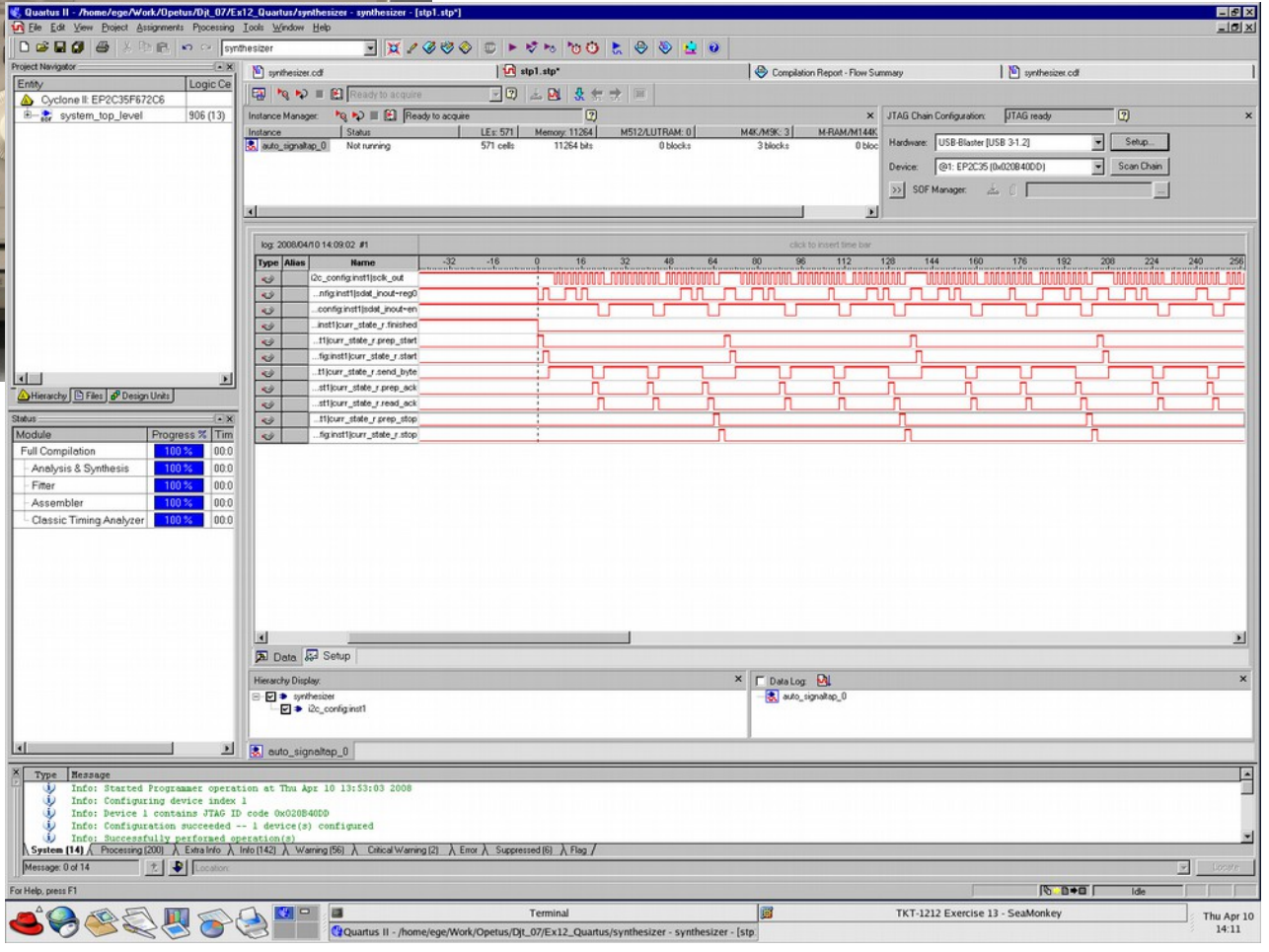
¿Y si no se por donde empezar?

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

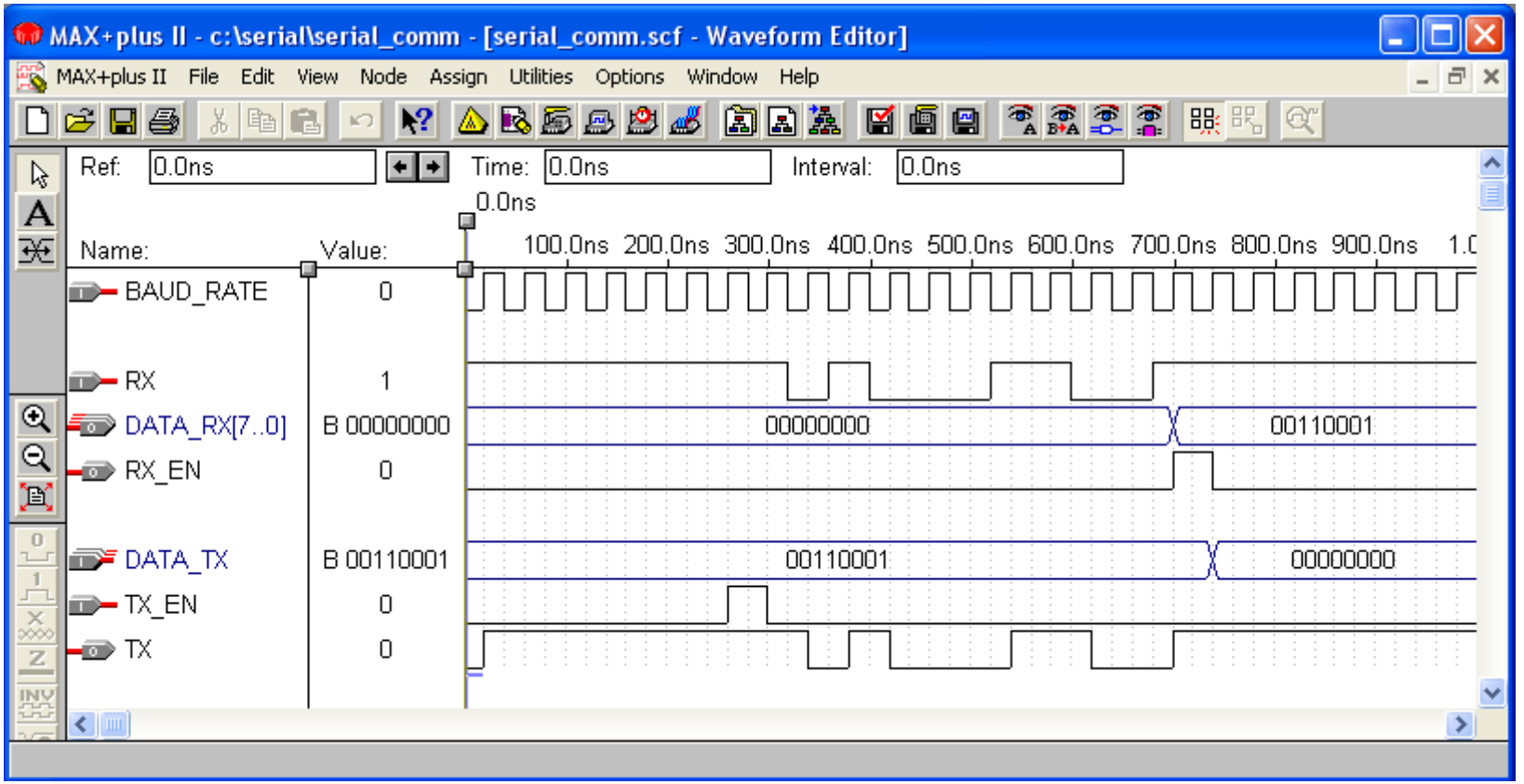

MacBook

Herramientas

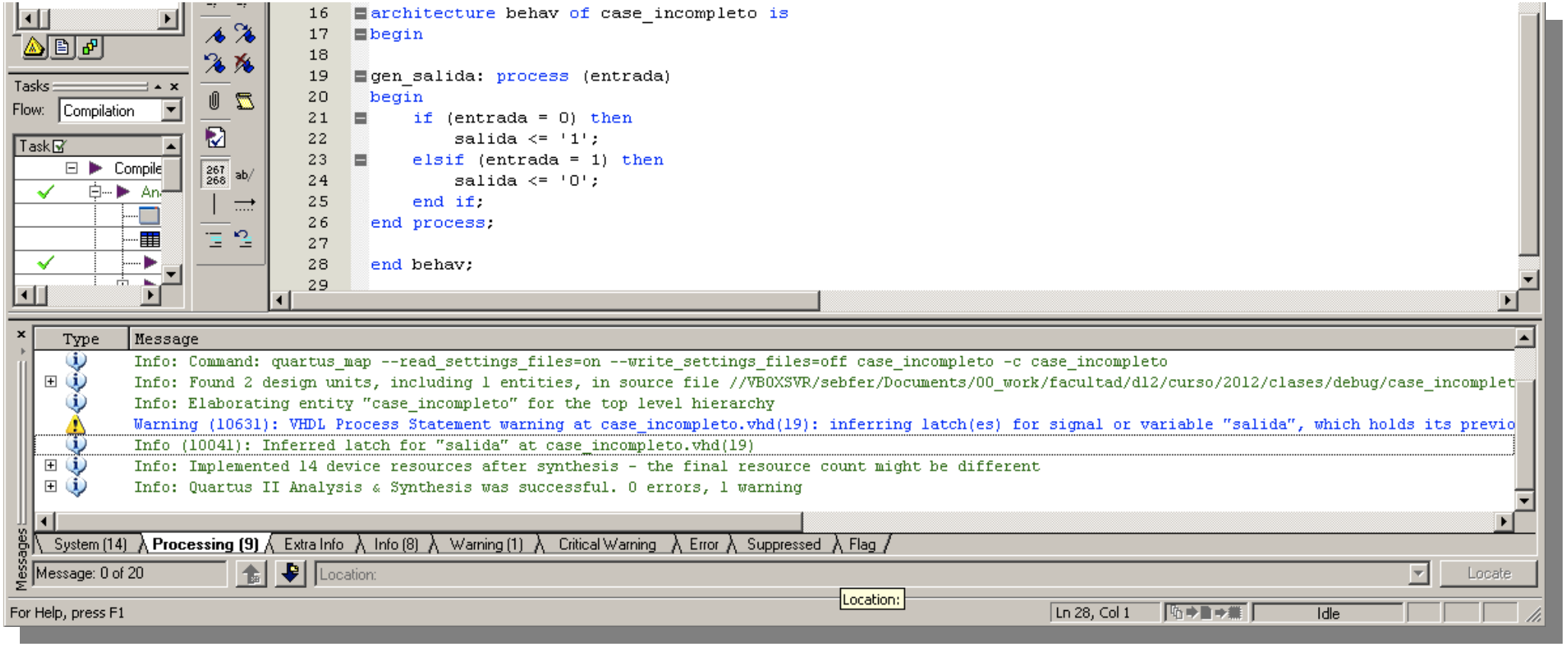




Herramientas

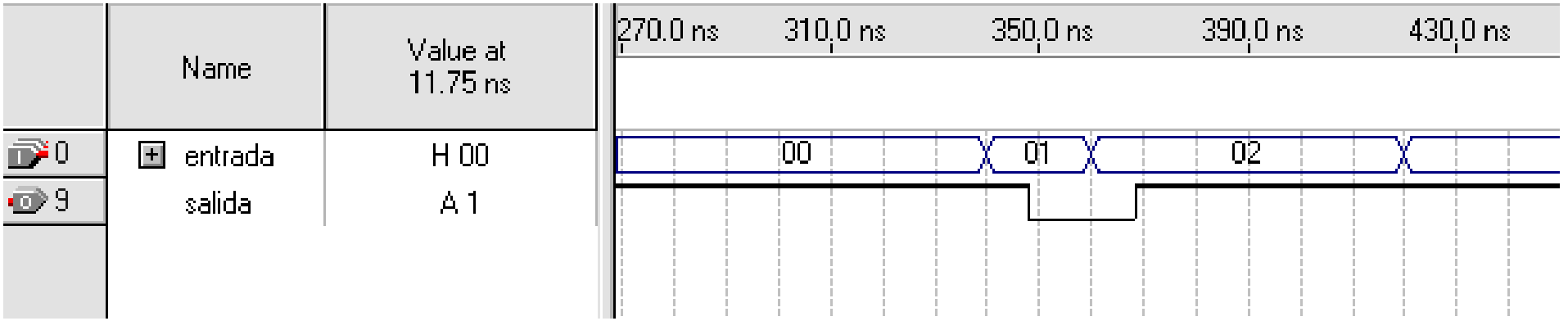


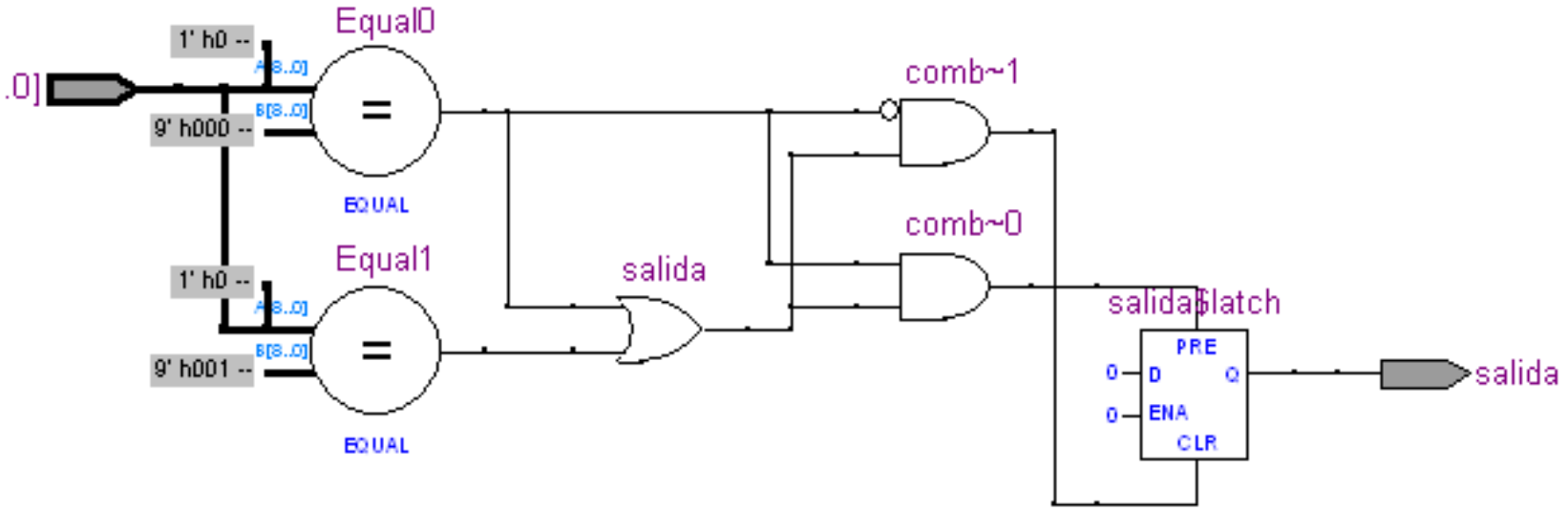
If - elsif



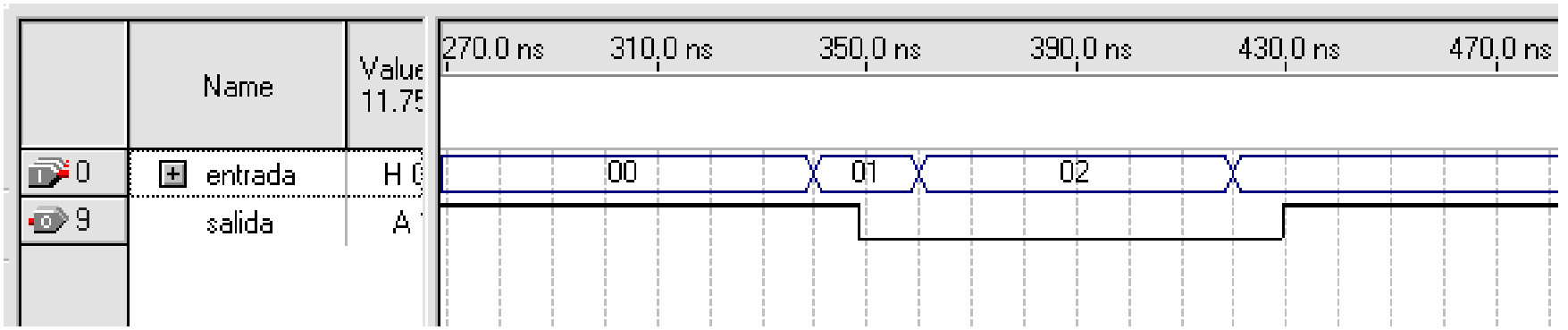
Warning: ... Infering latch(es) for signal or variable “salida”

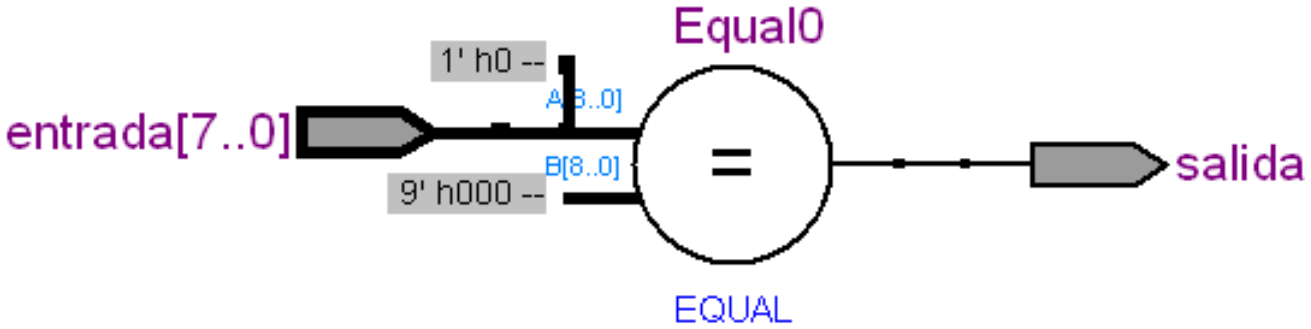
If - elsif

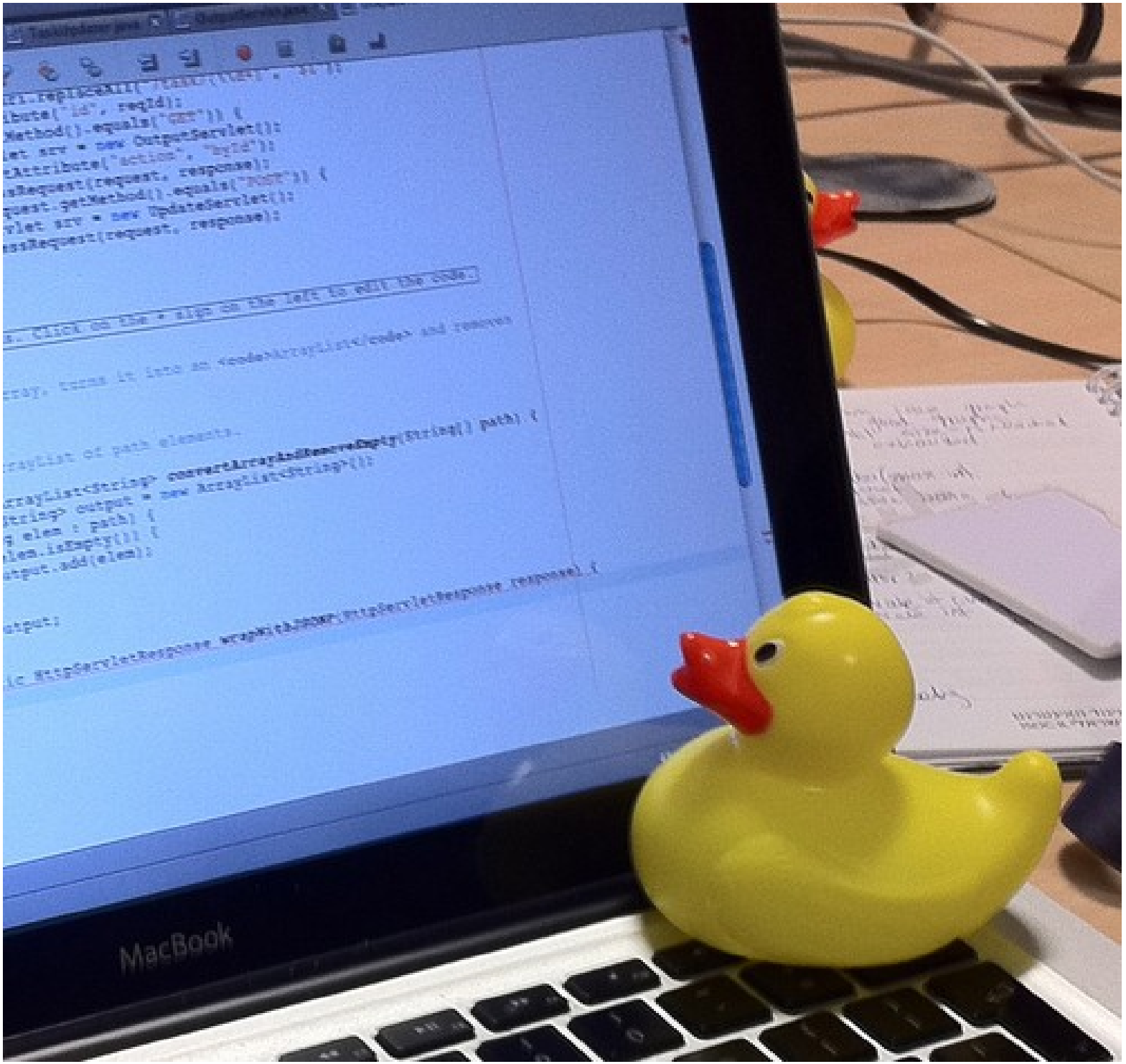

Name
Value at
11.75 ns
270.0 ns
310.0 ns
350.0 ns
390.0 ns
430.0 ns
entrada
H 00
00
01
02
salida
A 1


Equal0
1' h0 --
9' h000 --
EQUAL
Equal1
1' h0 --
9' h001 --
EQUAL
comb~1
comb~0
salida
salida$latch
PRE
D
Q
ENA
CLR
salida

If - else


Name
Value
270.0 ns
310.0 ns
350.0 ns
390.0 ns
430.0 ns
470.0 ns
0
entrada
9
salida
00
01
02


Equal0
1' h0 --
entrada[7..0]
A[8..0]
B[8..0]
9' h000 --
=
salida
EQUAL


MacBook