Capítulo 6

Comprobación Funcional

...

...

Resumen: En este capitulo se realizan la comprobación funcional del diseño y la exposición de los resultados obtenidos al implementar el procesador e introducir el sistema de tolerancia a fallos en el mismo.

6.1. Introducción

La herramienta software «PlanAhead» de Xilinx es capaz de sintetizar e implementar el diseño VHDL. Se utilizará esta herramienta para analizar los cambios que ocurren en el procesador al insertar los módulos de tolerancia a fallos.

Para realizar las simulaciones de ejecución de programas sobre las implementaciones del procesador se utilizará el software "ModelSim PE Student Edition 10.4". A continuación se analizan ambos diseños.

6.2. Programa de control

Para probar el funcionamiento del microprocesador se han implementado las memorias de datos e instrucciones y se han utilizado las mismas en ambos diseños del procesador:

- La **memoria de datos** se ha implementado como una memoria RAM de 128 bytes capaz de almacenar 32 palabras.

- La **memoria de instrucciones** contiene un programa sencillo de 20 líneas de código THUMB-2 en el que se realiza una multiplicación mediante un bucle de sumas.

Las pruebas consistirán en la ejecución del programa (figura 6.1) un total de 4 veces:

1. La primera ejecución se realizará sobre el procesador sin tolerancia a fallos y sin inserción de fallos. Este caso se considera de control.

2. La segunda ejecución será realizada sobre el procesador sin tolerancia a fallos pero insertando fallos.

3. La tercera prueba se ejecutará sobre el procesador con tolerancia a fallos, sin inserción de fallos. Con esto se quiere probar que se han introducido satisfactoriamente los componentes tolerantes a fallos sin modificar la funcionalidad.

4. La cuarta prueba se volverá a ejecutar en el procesador tolerante a fallos, pero insertando fallos. Así probaremos que insertando los mismos fallos que en la segunda prueba, se vuelve a ejecutar el programa satisfactoriamente.

```
1  LDR R2, R0, #5
2  MOV R1, #25
3  MOV R3, #0
4  MOV R5, #1
5  MOV R4, R2
6  NOP
7  NOP
8  NOP
9  CMP R4, R0
10 BEQ #24
11 NOP
12 NOP
13 NOP
14 ADD R3, R3, R1
15 SUB R4, R4, R5
16 B #-32
17 NOP
18 NOP
19 NOP
20 STR R0, R3, #0
```

Figura 6.1: Código Thumb-2 de programa de pruebas.

El código de la figura 6.1 es el cálculo de multiplicar 25 por el dato almacenado en la dirección 5 de memoria (en este caso 5) para después

almacenar el resultado en la dirección 0 de memoria. En primer lugar se carga el dato de memoria a un registro, $R2 = 5$. A continuación se inicializan las variables que ayudarán a ejecutar el bucle: R1 con el valor 25, R3 se inicializa a 0 y acumulará el resultado de la ejecución, R5 conserva la constante 1, R0 vale 0 y R4 contará cuantas iteraciones quedan por ejecutar el bucle. En pseudo-código estas acciones se traducen en la figura 6.2.

```
1  R1 = 25;
2  R2 = Memoria(R0+5);
3  R3 = 0;
4  R4 = R2;
5  R5 = 1;
6  Si R4 = 0:
7      saltar a "`instruccion 12"';
8  En otro caso:
9    R3 = R3 + R1;
10   R4 = R4 - R5;
11   saltar a "`instruccion 6"';
12 Memoria(R0+0) = R3;
```

Figura 6.2: Pseudo-código de programa de pruebas.

6.3. Fallos introducidos

Con la intención de probar la fiabilidad del sistema, se insertan ciertos fallos en los componentes que deseamos poner a prueba, en este caso en los registros de control que almacenan los datos entre etapas.

1. **Fallo A:** Se introduce un cambio de valor del bus del dato leído de memoria de la etapa de escritura en registro. El bit 0 de este registro se invierte resultando en 0.

2. **Fallo B:** Se activa la señal de control «MemWrite» en la etapa «Memory».

3. **Fallo C:** Se modifica el valor del operando B en la instrucción de comparación entre los registros R4 y R0.

6.4. Procesador estándar

En esta sección se exponen los recursos utilizados por el procesador estándar, sin tolerancia a fallos, después de sintetizar e implementar el diseño. Como primer paso de la prueba funcional se exponen los resultados de las simulaciones de la prueba de control y la prueba de inserción de fallos.

6.4.1. Recursos

Los informes de las herramientas de síntesis e implementación de Xilinx nos proporcionan una serie de informes entre los que se encuentran los recursos consumidos por el diseño, el mapeado realizado sobre la FPGA y las limitaciones de tiempos. A continuación se presentan los datos más relevantes:

Map report

```
Design Summary
--------------

Number of errors:      0
Number of warnings:    0
Slice Logic Utilization:
  Number of Slice Registers:                 1,952 out of
     126,800    1%
    Number used as Flip Flops:               1,952
    Number used as Latches:                      0
    Number used as Latch-thrus:                  0
    Number used as AND/OR logics:                0
  Number of Slice LUTs:                      1,032 out of
     63,400    1%
    Number used as logic:                    1,015 out of
       63,400    1%
      Number using O6 output only:             873
      Number using O5 output only:              28
      Number using O5 and O6:                  114
      Number used as ROM:                        0
    Number used as Memory:                       0 out of
       19,000    0%
    Number used exclusively as route-thrus:     17
      Number with same-slice register load:     16
      Number with same-slice carry load:         1
      Number with other load:                    0

Slice Logic Distribution:
  Number of occupied Slices:                   603 out of
     15,850    3%
  Number of LUT Flip Flop pairs used:        2,362
    Number with an unused Flip Flop:           492 out of
       2,362   20%
    Number with an unused LUT:               1,330 out of
       2,362   56%
    Number of fully used LUT-FF pairs:         540 out of
       2,362   22%
    Number of unique control sets:              50
    Number of slice register sites lost
      to control set restrictions:               8 out of
         126,800    1%
```

Timing report

```
Timing summary:
---------------

Timing errors: 0  Score: 0  (Setup/Max: 0, Hold: 0)

Constraints cover 27520 paths, 0 nets, and 8309 connections

Design statistics:
   Minimum period:   6.435ns{1}   (Maximum frequency: 155.400MHz
       )
```

6.4.2. Ejecución de control

Se procede a realizar la primera prueba, o prueba de control, que consiste en ejecutar el programa explicado en la sección 6.2 sobre el procesador estándar. La simulación se ha realizado ejecutando el código adjunto en el apéndice A.

Los cálculos realizados por el procesador han sido monitorizados y comprobados. Se han controlado los registros de trabajo R0, R1, R2, R3, R4 y R5 para monitorizar que han sido actualizados en los momentos adecuados y con los valores esperados. Igualmente, se ha comprobado el resultado obtenido en la dirección 0 de la memoria de datos. En nuestro caso es 125.

6.4.3. Inserción de fallos

Realizada la prueba de control, se procede con la segunda prueba, en este caso con la inserción de fallos. Con el objetivo de comprobar si alteran el comportamiento del procesador, se han insertado 3 fallos en los registros de control entre etapas. A continuación se describen los fallos insertados y sus consecuencias. El resultado completo de la simulación se incluye en la sección C.1 del apéndice C.

Además de las señales monitorizadas en la ejecución de control se han analizado las señales en donde se han insertado los fallos, así como los registros y direcciones de memoria donde se espera que provoquen consecuencias haciendo así efectivo un error en la ejecución del programa.

El simulador nos facilita la siguiente información alertando que los datos almacenados en los registros no son los esperados después de ejecutar las instrucciones:

```
1 # ** Error: ERROR: LDR R2, R0, #5
2 #    Time: 150 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
3 # ** Error: ERROR: MOV R4, R2
4 #    Time: 190 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
5 # ** Error: ERROR: SUB R4, R4, R5(4)
6 #    Time: 300 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
```

```
# ** Error: ERROR: SUB R4, R4, R5(3)
#    Time: 410 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: SUB R4, R4, R5(2)
#    Time: 520 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR:PC 204
#    Time: 620 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: ADD R3, R3, R1(100)
#    Time: 620 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: PC 208
#    Time: 630 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: PC 248
#    Time: 730 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: ADD R3, R3, R1(125)
#    Time: 730 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: PC 252
#    Time: 740 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: SUB R4, R4, R5(0)
#    Time: 740 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: PC 296
#    Time: 860 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
# ** Error: ERROR: STR R3, R0, #0
#    Time: 860 ns  Iteration: 0  Instance: /tb_ejecucion_fallos
```

6.5. Procesador tolerante a fallos

Siguiendo el mismo proceso anterior, en esta sección se exponen los recursos utilizados por el procesador tolerante a fallos después de sintetizar e implementar el diseño. De igual modo se exponen los resultados obtenidos al realizar las simulaciones de la prueba de control y la prueba de inserción de fallos sobre este microprocesador.

6.5.1. Recursos

Los informes de las herramientas de síntesis e implementación de Xilinx nos proporcionan una serie de informes entre los que se encuentran los recursos consumidos por el diseño, el mapeado realizado sobre la FPGA y las limitaciones de tiempos. A continuación se presentan los datos más relevantes:

Map report

```
Design Summary
--------------
Number of errors:      0
Number of warnings:    6
Slice Logic Utilization:
  Number of Slice Registers:                  2,798 out of
     126,800    2%
```

```
7     Number used as Flip Flops:                2,798
8     Number used as Latches:                       0
9     Number used as Latch-thrus:                   0
10    Number used as AND/OR logics:                 0
11  Number of Slice LUTs:                       1,539 out of
        63,400    2%
12    Number used as logic:                     1,464 out of
          63,400    2%
13      Number using O6 output only:            1,328
14      Number using O5 output only:               28
15      Number using O5 and O6:                   108
16      Number used as ROM:                         0
17    Number used as Memory:                        0 out of
          19,000    0%
18    Number used exclusively as route-thrus:      75
19      Number with same-slice register load:      74
20      Number with same-slice carry load:          1
21      Number with other load:                     0
22
23  Slice Logic Distribution:
24  Number of occupied Slices:                    858 out of
        15,850    5%
25  Number of LUT Flip Flop pairs used:         3,146
26    Number with an unused Flip Flop:            844 out of
          3,146   26%
27    Number with an unused LUT:                1,607 out of
          3,146   51%
28    Number of fully used LUT-FF pairs:          695 out of
          3,146   22%
29    Number of unique control sets:               71
30    Number of slice register sites lost
31      to control set restrictions:               42 out of
            126,800    1%
```

Timing report

```
1  Timing summary:
2  ---------------
3
4  Timing errors: 0  Score: 0  (Setup/Max: 0, Hold: 0)
5
6  Constraints cover 163797 paths, 0 nets, and 10410 connections
7
8  Design statistics:
9     Minimum period:   6.819ns{1}   (Maximum frequency: 146.649MHz
       )
```

6.5.2. Ejecución de control

Se ha ejecutado el programa de control sobre el procesador tolerante a fallos monitorizandose los registros relevantes para la ejecución del programa

así como las direcciones de memoria accedidas por el mismo.

Durante la ejecución se comprueba que la evolución de los registros es la esperada y al finalizar la ejecución del programa se vuelve a comprobar el valor almacenado en la dirección 0 de memoria. Los resultados de la simulación se incluyen en el apartado C.3 del apéndice C.

6.5.3. Inserción de fallos

Se ha ejecutado la simulación con inserción de fallos, esta simulación se puede consultar en la sección C.4 del apéndice C.

Junto a los registros monitorizados en la prueba de control se presentan los registros donde se insertan los fallos y así como los registros y las direcciones de memoria que se espera se verán afectadas por los fallos si estos no se toleran.

Bibliografía

Y así, del mucho leer y del poco dormir,
se le secó el celebro de manera que vino
a perder el juicio.

Miguel de Cervantes Saavedra

[1] R. Brinkgreve, W. Swolfs, and E. Engin. *ARM Architecture Reference Manual Thumb-2 Supplement.* 2011.

[2] S. Brown and J. Rose. Architecture of FPGAs and CPLDs: A tutorial. *IEEE Design and Test of Computers*, 13(2):42–57, 1996.

[3] C. T. Bustillos. Simulador arm en el ámbito docente. 2012.

[4] I. N. de Estadistica. Penetración de ordenador en hogares. 2014.

[5] S. Flash. Nexys4 FPGA Board Reference Manual Ethernet connector. pages 1–29, 2013.

[6] J. Gaisler. A portable and fault-tolerant microprocessor based on the SPARC V8 architecture. *Proceedings of the 2002 International Conference on Dependable Systems and Networks*, pages 409–415, 2002.

[7] J. C. González Salas. *Filtro adaptativo tolerante a fallos.* PhD thesis, 2014.

[8] S. Habinc. Functional Triple Modular Redundancy (FTMR). *Design and Assessment Report, Gaisler Research*, pages 1–56, 2002.

[9] J. L. Hennessy and D. A. Patterson. *Arquitectura de Computadores: Un enfoque cuantitativo.* Mcgraw Hill Editorial, 1993.

[10] J. L. Hennessy and D. a. Patterson. *Computer Architecture, Fourth Edition: A Quantitative Approach.* Number 0. 2006.

[11] A. C. Hu and S. Zain. NSEU Mitigation in Avionics Applications. 1073:1–12, 2010.

[12] O. Ieee-std. LEON3 7-Stage Integer Pipeline. (March), 2010.

[13] A. O. Investigation. ATSB TRANSPORT SAFETY REPORT Aviation Occurrence Investigation AO-2008-070 Final. (October), 2008.

[14] Jedec. Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray Induced Soft Error in Semiconductor Devices: JESD89A. *JEDEC Sold State Technology Association*, pages 1–85, 2006.

[15] A. Kadav, M. J. Renzelmann, and M. M. Swift. Fine-grained fault tolerance using device checkpoints. *Proceedings of the eighteenth international conference on Architectural support for programming languages and operating systems - ASPLOS '13*, page 473, 2013.

[16] H. Kirrmann. Fault Tolerant Computing in Industrial Automation. *Lecture notes ABB Corporate ResearchETH*, 2005.

[17] I. Kuon, R. Tessier, and J. Rose. FPGA Architecture: Survey and Challenges. *Foundations and TrendsÂ® in Electronic Design Automation*, 2(2):135–253, 2007.

[18] A. R. M. Limited. ARM7TDMI-S. (Rev 3), 2000.

[19] a. R. M. Limited. ARM Architecture Reference Manual. pages 1–1138, 2007.

[20] W. K. Melis. *Reconstruction of High-energy Neutrino-induced Particle Showers in KM3NeT .* PhD thesis, 2014.

[21] C. Mobile. Streaming 4K Ultra HD video at home and on the go. pages 0–1.

[22] J. Rose, A. E. Gamal, and A. Sangiovanni-Vincentelli. Architecture of Field-Programmable Gate Arrays.

[23] E. Rotenberg. AR-SMT: a microarchitectural approach to fault tolerance in microprocessors. *Digest of Papers. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing (Cat. No.99CB36352)*, 1999.

[24] D. J. Sorin and S. Ozev. Fault Tolerant Microprocessors for Space Missions. *Memory*, pages 1–4.

[25] U. States. Reduce Cost and Board Space. 374:1–8, 2011.

[26] I. S. Summary, T. C. Field, M. Long, S. D. Transfer, U. Instruction, and I. S. Examples. ARM Instruction Set. pages 1–60.

[27] J. M. Torrecillas. RAID - Tolerancia a Fallos.

[28] C. Weaver and T. Austin. A fault tolerant approach to microprocessor design. *Proceedings of the International Conference on Dependable Systems and Networks*, (July):411–420, 2001.

[29] Xilinx. Xilinx Artix-7 Fpgas: a New Performance Standard for Power-Limited, Cost-Sensitive Markets.