
Tests de boîte blanche



Pierre Langlois

<http://creativecommons.org/licenses/by-nc-sa/2.5/ca/>

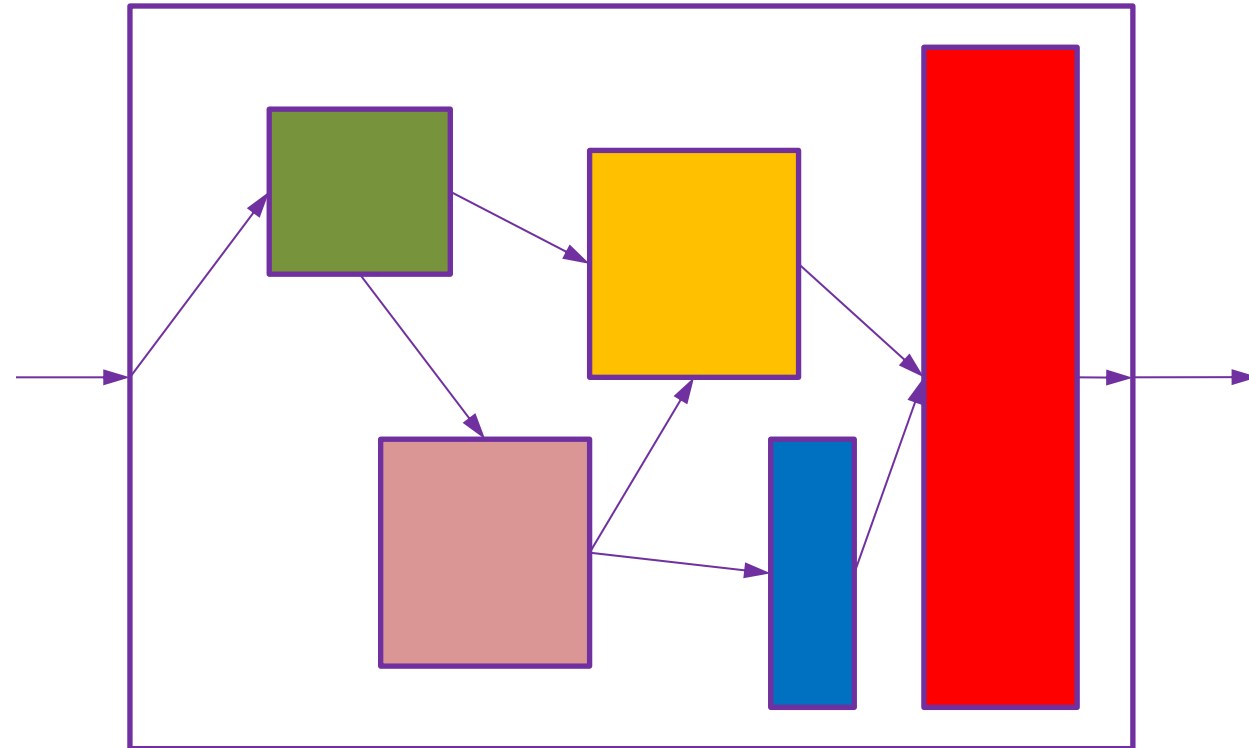
Tests de boîte blanche

Sujets de ce thème

- Définitions
- La couverture de code
- La couverture de paramètres d'opération
- La couverture fonctionnelle

Tests de boîte blanche (ou tests structurels)

- Le terme « test de boîte blanche » fait référence à un test qui nécessite de connaître le fonctionnement interne du système.
- En anglais:
white box, glass box, clear box, structural test.
- En s'appuyant sur des principes de couverture, on peut calculer des métriques permettant de déterminer à quel point le test a stimulé le système.
- Les tests de boîte blanche ne permettent pas en général de découvrir les fonctionnalités manquantes du système.



Couverture de code

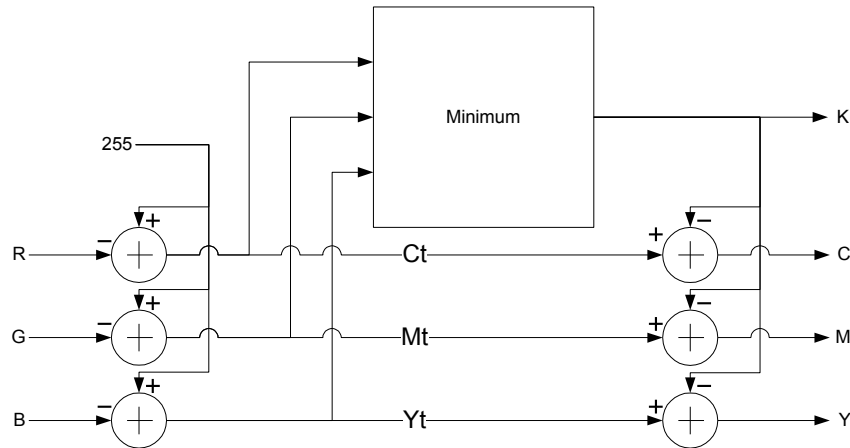
- Dans la couverture de code, on choisit des vecteurs de test pour exercer un élément particulier du design ou de sa description.
- Il y en a plusieurs :
 - Couverture d'énoncés : pourcentage des énoncés du code exécutés.
 - Couverture de blocs : pourcentage de blocs d'énoncés délimités par des structures de contrôle qui ont été exécutés. Cette métrique a l'avantage d'être plus représentative que la couverture d'énoncés, parce que les blocs comportant plusieurs énoncés n'ont pas un poids plus important que les autres.
 - Couverture de branchements : pourcentage des choix de branchement exécutés.
 - Couverture d'expressions : pourcentage des composantes des expressions booléennes qui ont affecté la valeur de ces expressions.
 - Couverture d'états : pourcentage du nombre d'états visités.
 - Couverture de transitions : pourcentage des transitions de chaque état ayant été prises.
 - Couverture de changements de signaux : pourcentage de signaux binaires ayant passé de 0 à 1 et de 1 à 0.

Couverture de code

- Pour chacune des couvertures possibles, on peut calculer une métrique qui exprime:
 - le nombre de fois où chaque situation se produit;
ou,
 - le pourcentage des situations qui se sont produites.
- Par exemple, on voudrait atteindre 100% de couverture des énoncés. Si on n'est qu'à 90%, cela signifie qu'il faut stimuler le circuit avec d'autres vecteurs de test.
- Le code doit être instrumenté pour obtenir les métriques (par un outil).
- Un outil peut présenter l'information obtenue de façon conviviale pour faciliter la sélection de vecteurs de test.
- Les différents éléments de couverture ne sont pas tous indépendants. Par exemple, la couverture d'états et la couverture de transitions sont effectivement des sous-ensembles de la couverture d'énoncés et de branchements.

Couverture de code

Exemple



A. Stodghill, Tip o'day: ask for a a refill, Green Options, 2007/06/18. Consulté le 4 septembre 2009, tiré de <http://greenoptions.com/tag/ink-cartridge>

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity convRGB2CMYK is
    port (
        rouge, vert, bleu : in unsigned(7 downto 0);
        cyan, magenta, jaune, noir : out unsigned(7 downto 0)
    );
end convRGB2CMYK;

architecture arch2 of convRGB2CMYK is
begin

    process(rouge, vert, bleu)
        variable cyant, magentat, jaunet, noirt1, noirt2 : unsigned(7 downto 0) := (others => '0');
        variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
    begin

        cyant := 255 - rouge;
        magentat := 255 - vert;
        jaunet := 255 - bleu;

        if cyant < magentat then noirt1 := cyant; else noirt1 := magentat; end if;
        if noirt1 < jaunet then noirt2 := noirt1; else noirt2 := jaunet; end if;

        cyan <= cyant - noirt2;
        magenta <= magentat - noirt2;
        jaune <= jaunet - noirt2;
        noir <= noirt2;

    end process;

end arch2;
```

Couverture de code

Exemple

- Avec les vecteurs de test: (0,0,0), (255,255,255)

The screenshot shows the Code Coverage Viewer interface for the file `convRGB2CMYK.vhd`. The Hierarchy tree on the left shows the following structure:

Hierarchy	CC [%]	CC with ch
Root : exemplesinf3500.convrgb...	90.91	
UUT : exemplesinf3500.conv...	87.50	
line_27	87.50	
line_34	90.91	

The main table displays the following data:

Line	Count	BC	EC	Source
21	None			end convRGB2CMYK;
22	None			
23	None			architecture arch2 of convRGB2CMYK is
24	None			
25	None			begin
26	None			
27	None			process rouge, vert, bleu)
28	None			variable cyant, magentat, jaunet, noirt1, noirt2 : unsigned(7 downto 0) := 10
29	None			variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
30	None			begin
31	None			
32	3			cyant := 255 - rouge;
33	3			magentat := 255 - vert;
34	3			jaunet := 255 - bleu;
35	None			
36	9*	0t 3f		// cyant < magentat then noirt1 := cyant; else noirt1 := magentat; end if;
37	9*	0t 3f		// noirt1 < jaunet then noirt2 := noirt1; else noirt2 := jaunet; end if;
38	None			
39	3			cyan <= cyant - noirt2;
40	3			magenta <= magentat - noirt2;
41	3			jaune <= jaunet - noirt2;
42	3			noir <= noirt2;
43	None			
44	3			end process;
45	None			
46	None			end arch2;
47	None			

The source code view shows the following code with highlighted lines:

```
end convRGB2CMYK;

architecture arch2 of convRGB2CMYK is
begin
  process rouge, vert, bleu)
  variable cyant, magentat, jaunet, noirt1, noirt2 : unsigned(7 downto 0) := 10
  variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
  begin
    cyant := 255 - rouge;
    magentat := 255 - vert;
    jaunet := 255 - bleu;

    // cyant < magentat then noirt1 := cyant; else noirt1 := magentat; end if;
    // noirt1 < jaunet then noirt2 := noirt1; else noirt2 := jaunet; end if;

    cyan <= cyant - noirt2;
    magenta <= magentat - noirt2;
    jaune <= jaunet - noirt2;
    noir <= noirt2;

  end process;
end arch2;
```

Couverture de code

Exemple

- Avec les vecteurs de test: (0,0,0), (255,255,255), (1,1,1), (100,100,100)

The screenshot shows the Code Coverage Viewer interface for the file `convRGB2CMYK.vhd`. The Hierarchy tree on the left shows the following coverage statistics:

Hierarchy	CC [%]	CC with ch
Root : exemplesinf3500.convrgb...	90.91	
UUT : exemplesinf3500.conv...	87.50	
line_27	87.50	
line_34	90.91	

The main window displays the source code for `convRGB2CMYK.vhd` with the following table of coverage statistics:

Line	Count	BC	EC	Source
21	None			end convRGB2CMYK;
22	None			
23	None			architecture arch2 of convRGB2CMYK is
24	None			
25	None			begin
26	None			
27	None			process rouge, vert, bleu)
28	None			variable cyant, magentat, jaunet, noit1, noit2 : unsigned(7 downto 0) := 10
29	None			variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
30	None			begin
31	None			
32	3			cyant := 255 - rouge;
33	3			magentat := 255 - vert;
34	3			jaunet := 255 - bleu;
35	None			
36	9*	0t 3f		// cyant < magentat then noit1 := cyant; else noit1 := magentat; end if;
37	9*	0t 3f		// noit1 < jaunet then noit2 := noit1; else noit2 := jaunet; end if;
38	None			
39	3			cyan <= cyant - noit2;
40	3			magenta <= magentat - noit2;
41	3			jaune <= jaunet - noit2;
42	3			noir <= noit2;
43	None			
44	3			end process;
45	None			
46	None			end arch2;
47	None			

Couverture de code

Exemple

- Avec les vecteurs de test: (0,0,0), (255,255,255), (20, 50, 0)

The screenshot shows the Code Coverage Viewer interface for the file `convRGB2CMYK.vhd`. The Hierarchy tree on the left shows the following structure:

Hierarchy	CC [%]	CC
Root : exemplesinf3500.convrgb...	90.91	
UUT : exemplesinf3500.conv...	93.75	
line_27	93.75	
line_35	90.91	

The main table displays the following data:

Line	Count	BC	EC	Source
19	None			cyan, magenta, jaune, noir : out unsigned(7 downto 0)
20	None);
21	None			end convRGB2CMYK;
22	None			architecture arch2 of convRGB2CMYK is
23	None			
24	None			begin
25	None			process rouge, vert, bleu
26	None			variable cyant, magentat, jaunet, noir1, noir2 : unsigned(7 downto 0) := (others =>
27	None			variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
28	None			begin
29	None			cyant := 255 - rouge;
30	None			magentat := 255 - vert;
31	None			jaunet := 255 - bleu;
32	4			
33	4			if cyant < magentat then noir1 := cyant; else noir1 := magentat; end if;
34	4			if noir1 < jaunet then noir2 := noir1; else noir2 := jaunet; end if;
35	None			
36	12*	0t 4f		
37	12*	1t 3f		
38	None			
39	4			cyan <= cyant - noir2;
40	4			magenta <= magentat - noir2;
41	4			jaune <= jaunet - noir2;
42	4			noir <= noir2;
43	None			
44	4			end process;
45	None			end architecture;

The source code view shows the following code with highlighted lines:

```
cyan, magenta, jaune, noir : out unsigned(7 downto 0)
);
end convRGB2CMYK;
architecture arch2 of convRGB2CMYK is
begin
  process rouge, vert, bleu
  variable cyant, magentat, jaunet, noir1, noir2 : unsigned(7 downto 0) := (others =>
  variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
  begin
  cyant := 255 - rouge;
  magentat := 255 - vert;
  jaunet := 255 - bleu;
  if cyant < magentat then noir1 := cyant; else noir1 := magentat; end if;
  if noir1 < jaunet then noir2 := noir1; else noir2 := jaunet; end if;
  cyan <= cyant - noir2;
  magenta <= magentat - noir2;
  jaune <= jaunet - noir2;
  noir <= noir2;
  end process;
```

Couverture de code

Exemple

- Avec les vecteurs de test: (0,0,0), (255,255,255), (20, 50, 0), (50, 20, 0)

Code Coverage Viewer - C:/Users/pierre/Desktop/pierre/enseignement/inf3500/wkspcINF3500/exemplesINF3500/coverage/results.ccl

File Edit View Help

Browse By: Blocks

Hierarchy	CC [%]	CC
Root : exemplesinf3500.convrgb...	90.91	
UUT : exemplesinf3500.conv...	100.00	
line_27	100.00	
line_35	90.91	

Line	Count	BC	EC	Source
19	None			cyan, magenta, jaune, noir : out unsigned(7 downto 0)
20	None);
21	None			end convRGB2CMYK;
22	None			
23	None			architecture arch2 of convRGB2CMYK is
24	None			
25	None			begin
26	None			
27	None			process rouge, vert, bleu)
28	None			variable cyant, magentat, jaunet, noirt1, noirt2 : unsigned(7 downto 0) := (others =>
29	None			variable c_ppe_m, m_ppe_j, j_ppe_c : std_logic;
30	None			begin
31	None			
32	5			cyant := 255 - rouge;
33	5			magentat := 255 - vert;
34	5			jaunet := 255 - bleu;
35	None			
36	15*	1t 4f		if cyant < magentat then noirt1 := cyant; else noirt1 := magentat; end if;
37	15*	2t 3f		if noirt1 < jaunet then noirt2 := noirt1; else noirt2 := jaunet; end if;
38	None			
39	5			cyan <= cyant - noirt2;
40	5			magenta <= magentat - noirt2;
41	5			jaune <= jaunet - noirt2;
42	5			noir <= noirt2;
43	None			
44	5			end process;
45	None			

Hierarchy Units Unused subprograms

Création de vecteurs de test selon la couverture de code

- Les métriques de couverture de code complètent les autres types de tests et donnent une certaine assurance au concepteur que le circuit est bien vérifié.
 - Un bon ensemble de vecteurs de tests produit une couverture de code de 100%.
 - Une couverture de code de 100% n'implique pas que l'ensemble de vecteurs de tests soit bon!
 - Une couverture de 100% pour un ensemble de vecteurs de test ne garantit pas que le circuit rencontre toutes ses spécifications.
- Il peut y avoir des exceptions, comme par exemple:
 - des énoncés qui sont en place pour protéger le système lors d'entrées non valides
 - des énoncés pour ramener le système dans un état valide à partir d'un état non valide

Ces deux cas nécessitent des vecteurs de test spéciaux.

Couverture de paramètres d'opération

- La couverture de code n'indique que si certaines situations ont été exercées ou non, sans égard à la fonctionnalité du système.
- Un test plus puissant consiste à identifier les paramètres d'opération du système et à vérifier la couverture des valeurs possibles de ceux-ci.
- Par exemple, pour une file d'attente, un paramètre serait le nombre d'éléments dans la file. Il est important de vérifier l'opération de la file quand celle-ci est vide, presque vide, pleine et presque pleine, ainsi qu'en situations moyennes.
- Pour obtenir la couverture des paramètres d'opération, les étapes suivantes peuvent être suivies :
 - Énumérer les paramètres d'opération du module;
 - Pour chaque paramètre, déterminer la gamme des valeurs possibles et identifier les valeurs qui doivent absolument être vérifiées et dans quelles circonstances;
 - Instrumenter le code afin de noter les valeurs de paramètre utilisées;
 - Simuler le système; et,
 - Calculer le rapport des valeurs utilisées sur le nombre de valeurs totales;
 - Établir si les valeurs à vérifier l'ont été.

Couverture fonctionnelle

- Dans ce genre de couverture, on énumère toutes les fonctions que le système doit pouvoir effectuer.
- Par exemple, dans un processeur, il doit être possible de transférer la valeur d'un registre vers un autre.
- On doit donc choisir des vecteurs de test qui exercent chacune des fonctions de la spécification.

Vous devriez maintenant être capable de ...

- Décrire quelques types de tests de boîte blanche. (B2)
- La couverture de code: décrire et utiliser. (B2, B3)
- La couverture de paramètres d'opération et la couverture fonctionnelle: décrire et utiliser. (B2, B3)

Code	Niveau (http://fr.wikipedia.org/wiki/Taxonomie_de_Bloom)
B1	Connaissance – mémoriser de l'information.
B2	Compréhension – interpréter l'information.
B3	Application – confronter les connaissances à des cas pratiques simples.
B4	Analyse – décomposer un problème, cas pratiques plus complexes.
B5	Synthèse – expression personnelle, cas pratiques plus complexes.