1.

(a) True

n variables 表示有總共有 2^n 個 vertices。

1 個 literal 的 cube 包含 2^{n-1} 個 vertices；

2 個 literals 的 cube 包含 2^{n-2} 個 vertices；

3 個 literals 的 cube 包含 2^{n-3} 個 vertices；

……

……

……

k 個 literals 的 cube 包含 2^{n-k} 個 vertices。

(b) True

BDD 在給定 order 的情形下作 reduce，得到的 representation 為唯一。

(似乎有很多同學沒把整句的意思看清楚，unique 是在修飾 ordered reduced BDD representation，意思是在問 ordered 且 reduced 的 BDD representation 是否為 unique…)

(c) False

5 個 input，表示有 2^5 種 input combination。每種 input combination 的 output 可以是 0 或 1，故有2^{2^5}種可能的 functions。

(d) False

若 cover 中包含可以化簡合併的列，即 cover 不為 prime，則即使未化簡前的 cover 不為 unate，其代表的 function 仍可能為 unate。舉例來說：

X Y Z

2 0 1……(1)

2 1 1……(2)

此 cover 雖然因為 Y 行同時出現 0 和 1，不為 unate，但實際上可以把(1)、(2)兩列合併化簡，使 cover 變為：

X Y Z

2 2 1

此時 cover 為 unate，亦代表此 cover 所代表的 function 為 unate function。

(e) False

PLA 因為 two-level 的特性，結構簡單且運算快速，delay 也較容易預估，故適合用於 timing critical 的 control logic。

2.

(a)

以 f_1 來說，要測試這個 fault，t_1 或 t_3 必須有一個要被選到，即$(t_1 + t_3) = 1$。對於所有要被測試的 faults，可以用同樣的方式列出類似的式子。每個式子要同時成立，故所有式子要 and 起來，如下面所示：

$(t_1 + t_3)\ (t_2 + t_3 + t_5)\ (t_1 + t_4 + t_6)\ (t_2 + t_3 + t_6)\ (t_1 + t_4)\ (t_2 + t_5)\ (t_3 + t_6)\ (t_4 + t_5) = 1$

我們要選出一組 tests，以 cover 到所有的 faults，故此問題為一 covering 的問題。又對於每個 test，若被選取，只可能使某些 fault 從不被 cover 變成被 cover 到，而不會使任何已經被 cover 到的 fault 移出 cover，故為 unate。(意即上面的式子中，只有 t_x 項出現，沒有 t_x' 項出現)

(b)

	t_1	t_2	t_3	t_4	t_5	t_6
f_1	1	0	1	0	0	0
~~f_2~~	~~0~~	~~1~~	~~1~~	~~0~~	~~1~~	~~0~~
~~f_3~~	~~1~~	~~0~~	~~0~~	~~1~~	~~0~~	~~1~~
~~f_4~~	~~0~~	~~1~~	~~1~~	~~0~~	~~0~~	~~1~~
f_5	1	0	0	1	0	0
f_6	0	1	0	0	1	0
f_7	0	0	1	0	0	1
f_8	0	0	0	1	1	0

先檢查有沒有 essential 的 row(即整個 row 只有一個 1)，檢查後發現沒有這樣的 row，故開始利用 row/column dominate 的技巧去化簡。以 row 來看，因為 f_2 dominate f_6，表示若 f_6 被 cover，f_2 也一定會被 cover，故 f_2 可以省略。同理，f_3 dominate f_5、f_4 dominate f_7，故 f_3、f_4 也可以省略掉。

	t_1	t_2	t_3	t_4	t_5	t_6
f_1	1	0	1	0	0	0
f_5	1	0	0	1	0	0
f_6	0	1	0	0	1	0
f_7	0	0	1	0	0	1
f_8	0	0	0	1	1	0

接下來，開始以 column dominate 的技巧去化簡。觀察後發現，t_5 dominate t_2，表示 t_2 可以測到的 faults，t_5 皆可測試到，故只需留下 t_5，t_2 可以省略。同理，t_3 dominate t_6，故 t_6 可以省略。

	t_1	t_3	t_4	t_5
~~f_1~~	~~1~~	~~1~~	~~0~~	~~0~~
f_5	1	0	1	0
~~f_6~~	~~0~~	~~0~~	~~0~~	~~1~~
~~f_7~~	~~0~~	~~1~~	~~0~~	~~0~~
~~f_8~~	~~0~~	~~0~~	~~1~~	~~1~~

Selected tests: $\{t_3, t_5\}$

在利用 row/column dominate 的技巧化簡後，觀察發現，f_6 為 essential 的 row，表示 t_5 一定要被選到才能 cover f_6。而選了 t_5 之後，f_8 也會被 cover 到，故 f_6 和 f_8 皆可在 t_5 被選到後自表中移除。同理，f_7 也是 essential 的 row，表示 t_3 一定要被選到才能 cover f_7。而選了 t_3 之後，f_1 也會被 cover 到，故 f_1 和 f_7 皆可在 t_5 被選到後自表中移除。

	t_1	t_4
f_5	1	1

最後只剩下要 f_5 考慮。此時 t_1 或 t_4 任選一個皆可。故依照標準的作法下，答案應為$\{t_1, t_3, t_5\}$或$\{t_3, t_4, t_5\}$。(其他的答案，只要是 3 個 tests，且有 cover 到所有的 faults，都會給分，不過上述的"標準作法"請詳加了解…)

3.

$f(1, x_2, \ldots, x_n)$等同於f_{x_1}；$f(0, x_2, \ldots, x_n)$等同於$f_{x_1}{}'$。

因 x_1 在 $f(x_1, x_2, \ldots, x_n)$中為 monotonic increasing，所以只會有 1 或 2(don't care)兩種情形，不會有 0 的狀況，即$f_{x_1}{}' = 0$。

故$\frac{\partial f}{\partial x_1} = f_{x_1} \cdot f_{x_1}{}' + f_{x_1}{}' \cdot f_{x_1}{}' = f_{x_1} \cdot f_{x_1}{}' + f_{x_1}{}' \cdot 0 = f_{x_1} \cdot f_{x_1}{}'$

4.

$w'xy$ 有 w、x、y 三個方向可以 expand。

測試往 w 方向 expand，即把 w 設為 don't care，expand 為 xy 的情形：

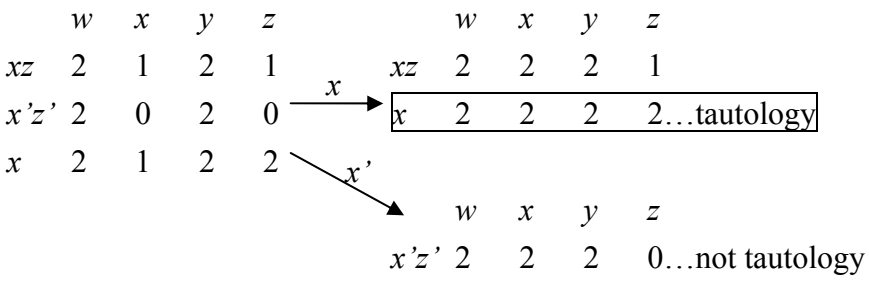
$G_{xy} = z + w'$

	w	x	y	z
z	2	2	2	1
w'	0	2	2	2

not tautology，故不可 expand 為 xy。

測試往 x 方向 expand，即把 x 設為 don't care，expand 為 $w'y$ 的情形：

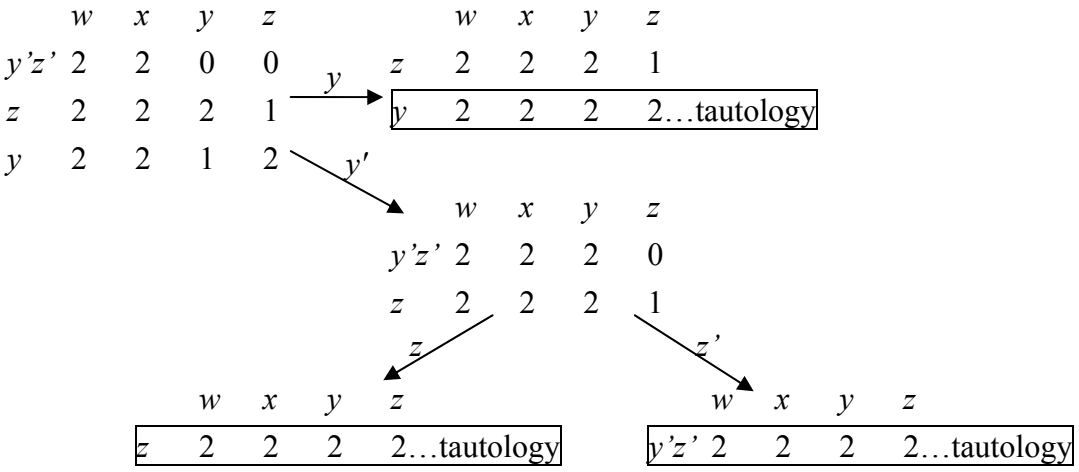
$G_{w'y} = xz + x'z' + x$



not tautology，故不可 expand 為 $w'y$。

測試往 y 方向 expand，即把 y 設為 don't care，expand 為 $w'x$ 的情形：

$G_{w'x} = y'z' + z + y$



全部都 tautology，表示 $G_{w'x}$ 為 tautology，故 $w'xy$ 可 expand 為 $w'x$，讓 G' = $w'y'z' + xz + x'yz' + w'x$。又在先前的測試中，已知 $w'xy$ 不可往 w 或 x 方向 expand，而 $w'x$ 包含 $w'xy$，比 $w'xy$ 更大，故更無法往這兩方向 expand，可推得 $w'x$ 為 prime。

5.

(a)

a: cd + e# (cube free，*cd + e* 沒有重複出現的 literal)

b: cd + e# (cube free，*cd + e* 沒有重複出現的 literal)

c: ad + de'f + bd (not cube free，*d* 重複出現)

cd: a + e'f + b# (cube free，*a + e'f + b* 沒有重複出現的 literal)

d: ac + ce'f + bc (not cube free，*c* 重複出現，但 *c* 在 *d* 前面，已處理過)

e: f' + a + b# (cube free，*f' + a + b* 沒有重複出現的 literal)

其他沒列出的 literal，表示在原式中沒有重複出現過。

co-kernel	**Kernel**	**Level**
a	*cd + e*	0
b	*cd + e*	0
cd	*a + e'f + b*	0
e	*f' + a + b*	0
1	*(a + b)(cd + e) + ef' + cde'f*	1

(b)

$x = acd + ef' + ae + cde'f + bcd + be$

$y = cd + e$

$U = cd + e + e + cd + cd + e$

$V = a + f' + a + e'f + b + b$

$U_{cd} = a + e'f + b$

$U_e = f' + a + b$

$q = U_{cd} \cap U_e = a + b$

$r = x - yq$

$= (acd + ef' + ae + cde'f + bcd + be) - (cd + e)(a + b)$

$= acd + ef' + ae + cde'f + bcd + be - acd - ae - bcd - be$

$= ef' + cde'f$

$x = yq + r$

$= y(a + b) + ef' + cde'f$

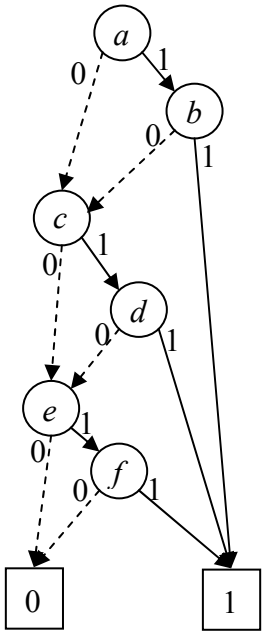
6.

(a)

$f(a, b, c, d, e, f) = ab + cd + ef$

給定 order $a < b < c < d < e < f$，要建立 OBDD：

直接建立完整的 BDD 再作化簡，過程過度繁瑣，且結構會相當龐大。分析給定的 order 以及 $f(a, b, c, d, e, f)$的內容可發現：a、b 會先被判斷，若 a、b 皆為 1，可得知結果為 1，不必考慮 c、d、e、f 的值；若 a、b 不同時為 1，則 c、d 會接著被判斷，若 c、d 皆為 1，可得知結果為 1，不必考慮 e、f 的值；若 a、b 不同時為 1 且 c、d 不同時為 1，才須透過 e、f 的值作判斷。利用這些觀察可以直接建立如右圖所示的 OBDD。



(b)

(i)

*abc**def*	000	001	010	011	100	101	110	111	class
000	0	0	0	1	0	0	0	1	(1)
001	0	0	0	1	1	1	1	1	(2)
010	0	0	0	1	0	0	0	1	(1)
011	0	0	0	1	1	1	1	1	(2)
100	0	0	0	1	0	0	0	1	(1)
101	0	0	0	1	1	1	1	1	(2)
110	1	1	1	1	1	1	1	1	(3)
111	1	1	1	1	1	1	1	1	(3)

共有 3 種 equivalence classes，$g(B)$需要 2 個 bit 作 encode。

(ii)

在不考慮 $f(a, b, c, d, e, f)$功能特性的情形下，會用 00、01、10 作 encode，並把 11 當作 don't care。但若把 $f(a, b, c, d, e, f)$的功能特性考慮進去，應用 1 個 bit 來反應 $ab = 1$ 是否發生(因為此時已知結果為 1)，再用另一個 bit 來反應 c 的值，在 ab 不為 1 時，給 $f_t(g(B), F)$繼續判斷結果。故去較好的 encode 方式，應該為 00、01、1x，如下所示：

Truth table of $g(B)$

a	b	C	$g(B)_2$	$g(B)_1$
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	x(0)
1	1	1	1	x(1)

此時，經過化簡，可得 $g(B)_2 = ab$，$g(B)_1 = c$，畫出下圖的 OBDDs。

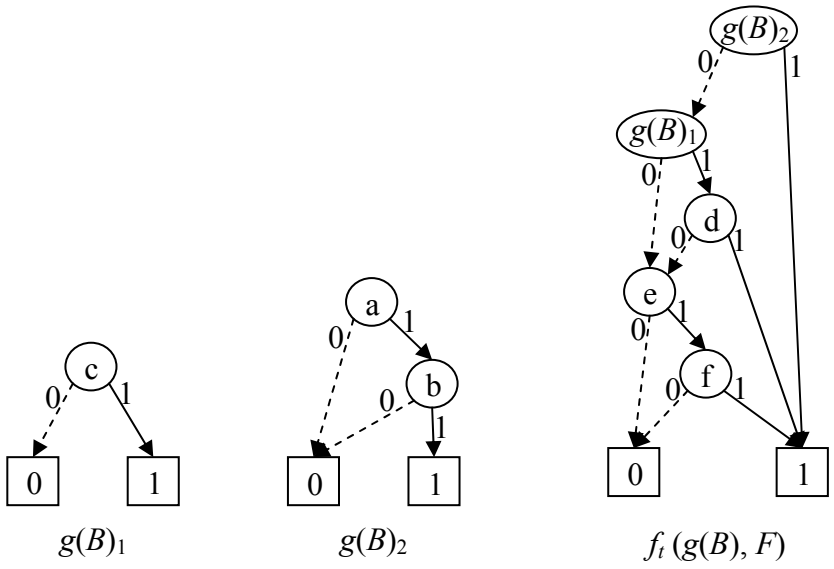
而 $f_t(g(B), F)$的部分，可由式子推導：

$f(a, b, c, d, e, f) = f_t(g(B), F) = f_t(g(B)_1, g(B)_2, c, d, e)$

$= ab + cd + ef = g(B)_2 + g(B)_1 d + ef$

可得 OBBD 如下圖所示。

(若不考慮功能特性，直接用 00、01、10 去作 encode，所得到的 OBBDs 會稍微複雜一點，但整體架構和功能相同)



7.

(a)

$y_4 = ac'$

SDC$y_4 = y_4 \oplus ac'$

$= y_4' \cdot (ac') + y_4 \cdot (ac')'$

$= y_4'ac' + y_4 \cdot (a' + (c')')$

$= y_4'ac' + y_4a' + y_4 c$

(b)

$y_1 = y_2 + y_3 = y_2 + y_4y_5$

$y_2 = ac$

$y_5 = (ab)'$

ODC$y_4 = (\frac{\partial y_1}{\partial y_4})' = (y_{1_{y^4}} \oplus y_{1_{y^{4'}}})'$

$= (y_{1_{y^4}} \cdot y_{1_{y^{4'}}} + y_{1_{y^4}}' \cdot y_{1_{y^{4'}}}')$

$= (y_2 + y_5) \cdot y_2 + (y_2 + y_5)' \cdot y_2'$

$= y_2 + (y_2' y_5') \cdot y_2'$

$= y_2 + y_2' y_5'$

$= ac + (ac)' ((ab)')'$

$= ac + (a' + c') ab$

$= ac + a'ab + abc'$

$= ac + abc'$

(c)

原來 $y_4 = ac'$的 k-map，如下圖所示：

$a\backslash c$	0	1
0	0	0
1	1	0

考慮前面所算出的 don't care，將對應的欄位換成 x，得到下面的 k-map：

$a\backslash c$	0	1
0	0	0
1	1	x

使用 don't care 化簡後，只剩下 a 一個 literal。