12 ශ්‍රේණිය A/L ICT

පාඩම 4: සංඛ්‍යාංක පරිපථවල මූලධර්ම

- 4.1 තර්ක ද්වාර සහ සත්‍යතා වගු: NOT, AND, OR, XOR, NAND, NOR, XNOR
- 4.2 බූලියානු වීජ ගණිතය සහ Karnaugh සිතියම්: නියම, SOP, POS සහ සරල කිරීම
- 4.3 සංඛ්‍යාංක පරිපථ නිර්මාණය: එදිනෙදා අවශ්‍යතාවල සිට තර්ක ද්වාර දක්වා
- 4.4 අර්ධ එකතුකරන්නා, සම්පූර්ණ එකතුකරන්නා සහ Flip-Flop: CPU කොටස් සහ මතක කොටු

© 2026 schoolict.net (Ruwan Mangala Suraweera)
සියලු හිමිකම් ඇවිරිණි.

12 ශ්‍රේණිය A/L ICT · පාඩම 4

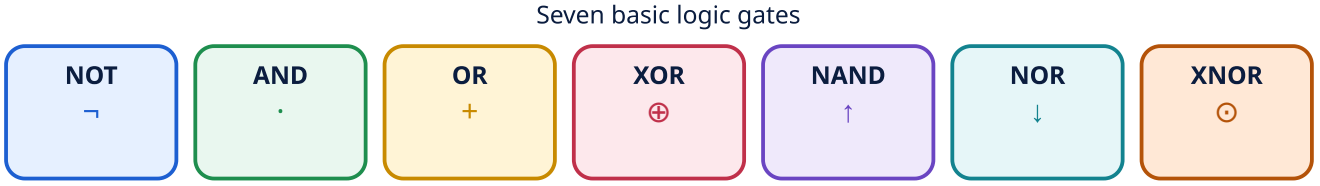
4.1 තර්ක ද්වාර සහ සත්‍යතා වගු: NOT, AND, OR, XOR, NAND, NOR, XNOR

විෂය නිර්දේශය 4.1. තර්ක ද්වාරයක් යනු එකක් හෝ වැඩි **ද්වීමය ආදාන** (0 හෝ 1) ගෙන එක් **ද්වීමය ප්‍රතිදානයක්** ලබා දෙන කුඩා පරිපථයකි. මෙම කොටසින් මූලික ද්වාර හත – **NOT, AND, OR, XOR** සහ සංයුක්ත ද්වාර **NAND, NOR, XNOR** – ඒවායේ **සංකේත ඇඳීම**, **සත්‍යතා වගු** ලිවීම, සහ වෙනත් ඕනෑම ද්වාරයක් සෑදිය හැකි **සර්ව ද්වාර** ලෙස **NAND සහ NOR** වැදගත් වන්නේ ඇයිදැයි ඉගෙන ගනී.

1. තර්ක ද්වාරයක් යනු කුමක්ද?

සෑම CPU එකක් තුළත් සෑම සංඛ්‍යාංක විපයක් තුළත් තීරණ ගන්නේ **තර්ක ද්වාර** මගිනි. සෑම ද්වාරයක්ම සරල නීතියක් අනුගමනය කරයි: ආදානවල ඇති 0 සහ 1 හි සෑම මිශ්‍රණයකටම ප්‍රතිදානය 0 හෝ 1 වේ. එම නීතිය අපි **සත්‍යතා වගුවක්** ලෙස ලියමු. **0** යනු LOW / අසත්‍ය / OFF වේ. **1** යනු HIGH / සත්‍ය / ON වේ.

> 💡 සත්‍යතා වගුවක **සෑම** ආදාන සංයෝජනයක්ම ඇත. ආදාන n කින් **පේළි 2^n** ක් ඇත. ආදාන එකක් → පේළි 2. ආදාන දෙකක් → පේළි 4. ආදාන තුනක් → පේළි 8.



මූලික තර්ක ද්වාර හත

2. මූලික ද්වාර: NOT, AND, OR, XOR

NOT (පෙරළුනය) ට **ආදාන එකක්** ඇත. එය බිටුව **පෙරළයි**: F = A′ (¬A හෝ Ā ලෙසද ලියයි). A 0 නම් F 1 වේ; A 1 නම් F 0 වේ.

A	F = NOT
0	1
1	0

AND සෑම ආදානයක්ම 1 වූ විට පමණක් 1 ලබා දෙයි. F = A · B (හෝ A ∧ B) ලෙස ලියන්න. “A **සහ** B දෙකම සත්‍ය විය යුතුය” ලෙස හිතන්න.

A	B	F = AND

0	0	0
0	1	0
1	0	0
1	1	1

OR අවම වශයෙන් එක් ආදානයක් 1 වූ විට 1 ලබා දෙයි. F = A + B (හෝ A ∨ B) ලෙස ලියන්න. “A **හෝ** B (හෝ දෙකම)”.

A	B	F = OR
0	0	0
0	1	1
1	0	1
1	1	1

XOR (විශේෂ OR) ආදාන **වෙනස්** වූ විට 1 ලබා දෙයි. F = A ⊕ B ලෙස ලියන්න. සමාන ආදාන → 0; වෙනස් → 1. අර්ධ එකතුකරන්නාගේ **එකතු බිටුව** සාදන්නේ මෙම ද්වාරයයි (පාඩම 4.4).

A	B	F = XOR
0	0	0
0	1	1
1	0	1
1	1	0

🧱 **විසඳූ උදාහරණය: F = A · B සඳහා සත්‍යතා වගුව සම්පූර්ණ කරන්න. A = 1 සහ B = 1 වූ විට F කුමක්ද?**
AND ආදාන දෙකම 1 වූ විට පමණක් 1 වේ, එබැවින් F = **1**. පේළි හතර: 00→0, 01→0, 10→0, 11→**1**.

3. සංයුක්ත ද්වාර: NAND, NOR, XNOR

සංයුක්ත ද්වාරයක් යනු ප්‍රතිදානයට **NOT** එකක් සම්බන්ධ කළ මූලික ද්වාරයකි. **NAND** = NOT-AND: F = (A · B)′. ආදාන දෙකම 1 වූ විට පමණක් 0 වේ. **NOR** = NOT-OR: F = (A + B)′. ආදාන දෙකම 0 වූ විට පමණක් 1 වේ. **XNOR** = NOT-XOR: F = (A ⊕ B)′. ආදාන **සමාන** වූ විට 1 වේ (සමානතා ද්වාරය).

A	B	F = NAND
0	0	1
0	1	1

1	0	1
1	1	0

A	B	F = NOR
0	0	1
0	1	0
1	0	0
1	1	0

A	B	F = XNOR
0	0	1
0	1	0
1	0	0
1	1	1

💡 ද්වාර සංකේතයක කුඩා **කවයක්** යනු NOT (පෙරළීම) වේ. NAND යනු කවයක් සහිත AND වැනිය; NOR යනු කවයක් සහිත OR වැනිය; XNOR යනු කවයක් සහිත XOR වැනිය.

🧱 **විසඳූ උදාහරණය: A = 1, B = 0 සඳහා AND, OR, XOR, NAND, NOR සහ XNOR හි ප්‍රතිදාන සොයන්න.**
AND = 0, OR = 1, XOR = 1, NAND = 1, NOR = 0, XNOR = 0.

4. ප්‍රකාශන සඳහා සත්‍යතා වගු (ආදාන තුනක් දක්වා)

F = A · B + A′ · C වැනි ප්‍රකාශනයකට පේළි 8 ක වගුවක් සාදන්න (A, B, C 000 සිට 111 දක්වා). එක් එක් ගුණිත පදය ගණනය කර පසුව ඒවා OR කරන්න.

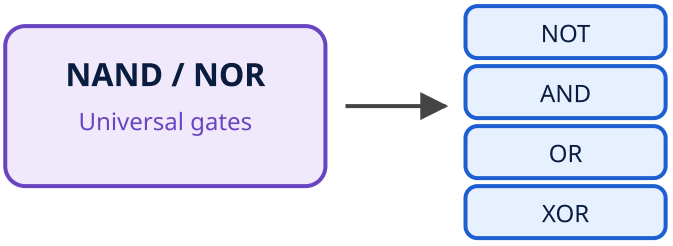
🧱 **විසඳූ උදාහරණය: F = A · B + A′ · C සඳහා සත්‍යතා වගුව සාදන්න.**
F 1 වන්නේ ABC = 001, 011, 110, 111 (දශම 1, 3, 6, 7) පේළිවලය. 000, 010, 100, 101 සඳහා F 0 වේ.

A	B	C	A·B	A′·C	F
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	0	0	0

1	0	1	0	0	0
1	1	0	1	0	1
1	1	1	1	0	1

5. සර්ව ද්වාර: NAND සහ NOR

වෙනත් **සෑම** ද්වාරයක්ම එකම ද්වාරයේ පිටපත් වලින් පමණක් සෑදිය හැකි නම් එය **සර්ව ද්වාරයකි**. **NAND** සහ **NOR** සර්ව වේ. කර්මාන්තශාලාවලදී මෙය වැදගත්ය: චිප නිර්මාණකරුට එක් ආකාරයක ද්වාර පමණක් තබාගෙන ඕනෑම පරිපථයක් සෑදිය හැක.



NAND සහ NOR මගින් වෙනත් ඕනෑම ද්වාරයක් සෑදිය හැක

NAND වලින් NOT
ආදාන දෙකම එකට සම්බන්ධ කරන්න: A NAND A = A′.

NAND වලින් AND
එක් වරක් NAND කර, ප්‍රතිදානය තවත් NAND එකකින් පෙරළන්න: (A NAND B)′ = A · B.

NAND වලින් OR
සෑම ආදානයක්ම පෙරළා NAND කරන්න: A′ NAND B′ = A + B (De Morgan).

🧱 **විසඳූ උදාහරණය: A = 0 සහ A = 1 සඳහා A NAND A = NOT A බව පෙන්වන්න.**
A = 0: 0 NAND 0 = 1 = NOT 0. A = 1: 1 NAND 1 = 0 = NOT 1. එබැවින් ආදාන එකට සම්බන්ධ කළ NAND යනු පෙරළනයකි.

💡 NOR පමණක් භාවිතයෙන්ද සියල්ල සෑදිය හැක: A NOR A = A′, ආදිය. විභාගවලදී NAND පමණක් (හෝ NOR පමණක්) භාවිතයෙන් NOT, AND හෝ OR ඇඳීමට සූදානම් වන්න.

🚀 **විෂය නිර්දේශයෙන් ඔබ්බට (2026 යාවත්කාලීනය)** *විභාගයට අවශ්‍ය නැත – දැනුම සඳහා කියවන්න.* 2026 දී CPU තවමත් කුඩා NAND සහ NOR ද්වාර ලෙස හැසිරෙන CMOS ට්‍රාන්සිස්ටර භාවිත කරයි. නවීන දුරකථන SoC එකක එවැනි ද්වාර බිලියන ගණනක් ඇත. ක්වොන්ටම් පරිගණක බිටු නොව qubit මත ක්‍රියා කරන වෙනස් “ද්වාර” (Hadamard, CNOT) භාවිත කරයි – නමුත් කුඩා සර්ව ද්වාර කට්ටලයකින් විශාල පරිපථ සෑදීමේ අදහස සමානය.

ප්‍රධාන පද

තර්ක ද්වාරය – ද්වීමය ආදාන ගෙන එක් ද්වීමය ප්‍රතිදානයක් ලබා දෙන පරිපථයකි.

සත්‍යතා වගුව – සෑම ආදාන සංයෝජනයක්ම සහ ඊට ගැළපෙන ප්‍රතිදානය ලැයිස්තුගත කරන වගුවකි.

NOT – පෙරළුනය: ප්‍රතිදානය එකම ආදානයේ විරුද්ධයයි. F = A′.

AND – සෑම ආදානයක්ම 1 නම් පමණක් ප්‍රතිදානය 1 වේ. F = A · B.

OR – අවම වශයෙන් එක් ආදානයක් 1 නම් ප්‍රතිදානය 1 වේ. F = A + B.

XOR – විශේෂ OR: ආදාන වෙනස් වූ විට ප්‍රතිදානය 1 වේ. F = A ⊕ B.

NAND – NOT-AND: F = (A · B)′. සර්ව ද්වාරයකි.

NOR – NOT-OR: F = (A + B)′. සර්ව ද්වාරයකි.

XNOR – NOT-XOR: ආදාන සමාන වූ විට ප්‍රතිදානය 1 වේ.

සර්ව ද්වාරය – වෙනත් ඕනෑම ද්වාරයක් තනිවම සෑදීමට භාවිත කළ හැකි ද්වාරයකි (NAND, NOR).

පෙරළුම් කවය – ද්වාර සංකේතයේ NOT අදහස් කරන කුඩා කවය.

සංයුක්ත ද්වාරය – මූලික ද්වාරයකට NOT එකතු කර සෑදූ ද්වාරයකි (NAND, NOR, XNOR).

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. තර්ක ද්වාරයක් යනු කුමක්ද?

පිළිතුර: නිශ්චිත නීතියකට අනුව එකක් හෝ වැඩි ද්වීමය ආදාන ගෙන එක් ද්වීමය ප්‍රතිදානයක් ලබා දෙන කුඩා සංඛ්‍යාංක පරිපථයකි.

2. ආදාන 2 ක AND ද්වාරයක සත්‍යතා වගුව ලියන්න.

පිළිතුර: AB: 00→0, 01→0, 10→0, 11→1.

3. XOR, OR ගෙන් වෙනස් වන්නේ කෙසේද?

පිළිතුර: ඕනෑම ආදානයක් 1 වූ විට (දෙකම ඇතුළුව) OR 1 වේ. XOR ආදාන වෙනස් වූ විට පමණක් 1 වේ – දෙකම 1 නම් 0 වේ.

4. NAND සහ NOR සර්ව ද්වාර ලෙස හඳුන්වන්නේ ඇයි?

පිළිතුර: NAND පමණක් හෝ NOR පමණක් භාවිතයෙන් වෙනත් ඕනෑම ද්වාරයක් (NOT, AND, OR, XOR, ...) සෑදිය හැකි නිසා.

5. NAND ද්වාරයකින් NOT ද්වාරයක් සාදන්නේ කෙසේද?

පිළිතුර: NAND ආදාන දෙකම එකම සංඥාව A වෙත සම්බන්ධ කරන්න. එවිට A NAND A = A′.

6. ආදාන තුනක් සඳහා සත්‍යතා වගුවක පේළි කීයක් තිබේද?

> **පිළිතුර:** 2^3 = පේළි 8 ක් (000 සිට 111 දක්වා).

බහුවරණ ප්‍රශ්න

1. සෑම ආදානයක්ම 1 වූ විට පමණක් 1 ලබා දෙන ද්වාරය කුමක්ද?

A. XOR
B. NOR
C. AND
D. OR
E. NOT

2. ආදානය 1 වූ විට NOT ද්වාරයක ප්‍රතිදානය

A. unchanged
B. 2
C. 1
D. undefined
E. 0

3. විශේෂ OR ලෙසද හඳුන්වන ද්වාරය කුමක්ද?

A. XOR
B. OR
C. NAND
D. AND
E. XNOR

4. NAND සමාන වන්නේ

A. NOR followed by NOT
B. OR followed by NOT
C. XOR followed by NOT
D. AND followed by NOT
E. NOT followed by AND

5. සර්ව වන ද්වාර දෙක කුමක්ද?

A. OR and XOR
B. AND and OR
C. NAND and NOR
D. XOR and XNOR
E. NOT and AND

6. A = 1, B = 1 සඳහා XOR ප්‍රතිදානය

A. B
B. 0
C. 1
D. A
E. 2

7. A = 0, B = 0 සඳහා NOR ප්‍රතිදානය

A. 1
B. undefined
C. A
D. B
E. 0

8. ද්වාර සංකේතයක කුඩා කවයකින් අදහස් වන්නේ

A. AND
B. power supply
C. inversion (NOT)
D. ground
E. clock

9. ආදාන 3 ක් සහිත සත්‍යතා වගුවක පේළි කීයක්ද?

A. 9
B. 3
C. 6
D. 4
E. 8

10. XNOR ප්‍රතිදානය 1 වන්නේ ආදාන

A. both 0 only
B. equal
C. undefined
D. both 1 only
E. different

11. A NAND B = A′ වන්නේ

A. B = 1
B. A = 1
C. B = 0
D. A = 0
E. B is tied to A (B = A)

12. OR වන ප්‍රකාශනය කුමක්ද?

A. A ⊕ B
B. (A · B)′
C. A · B
D. A + B
E. A′

13. “ද්වීමය ආදාන ගෙන එක් ද්වීමය ප්‍රතිදානයක් ලබා දෙන පරිපථයකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. NOR
B. පෙරළුම් කවය
C. XOR
D. තර්ක ද්වාරය
E. NOT

14. “සෑම ආදාන සංයෝජනයක්ම සහ ඊට ගැළපෙන ප්‍රතිදානය ලැයිස්තුගත කරන වගුවකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. සත්‍යතා වගුව
B. NAND
C. AND
D. XNOR
E. සංයුක්ත ද්වාරය

15. “පෙරළුනය: ප්‍රතිදානය එකම ආදානයේ විරුද්ධයයි. F = A′” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. NOT
B. තර්ක ද්වාරය
C. සර්ව ද්වාරය
D. OR
E. NOR

16. “සෑම ආදානයක්ම 1 නම් පමණක් ප්‍රතිදානය 1 වේ. F = A · B” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. සත්‍යතා වගුව
B. පෙරළුම් කවය
C. XOR
D. XNOR
E. AND

17. “අවම වශයෙන් එක් ආදානයක් 1 නම් ප්‍රතිදානය 1 වේ. F = A + B” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. NAND
B. OR
C. සර්ව ද්වාරය
D. සංයුක්ත ද්වාරය
E. NOT

18. “විශේෂ OR: ආදාන වෙනස් වූ විට ප්‍රතිදානය 1 වේ. F = A ⊕ B” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. පෙරළුම් කවය
B. AND
C. XOR
D. NOR
E. තර්ක ද්වාරය

19. “NOT-AND: F = (A · B)′. සර්ව ද්වාරයකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. XNOR
B. NAND
C. සත්‍යතා වගුව
D. OR
E. සංයුක්ත ද්වාරය

20. “NOT-OR: F = (A + B)′. සර්ව ද්වාරයකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. XOR
B. තර්ක ද්වාරය
C. සර්ව ද්වාරය
D. NOR
E. NOT

ව්‍යුහගත ප්‍රශ්න

1. ආදාන 2 ක OR ද්වාරයක සත්‍යතා වගුව අඳින්න සහ ප්‍රතිදානය 0 වන විට කියන්න. *(ලකුණු 2)*

> පේළි: 00→0, 01→1, 10→1, 11→1 (එක බැගින්); ආදාන දෙකම 0 වූ විට පමණක් ප්‍රතිදානය 0 (1).

2. සත්‍යතා වගු භාවිතයෙන් AND සහ NAND අතර වෙනස පැහැදිලි කරන්න. *(ලකුණු 3)*

> AND: 11→1, අනෙක් පේළි 0 (1); NAND යනු AND හි විරුද්ධයයි: 11→0, අනෙක් පේළි 1 (1); NAND = (A·B)′ (1).

3. සර්ව ද්වාර ප්‍රයෝජනවත් වන්නේ ඇයි? සර්ව ද්වාර දෙක නම් කරන්න. *(ලකුණු 2)*

> එක් ද්වාර වර්ගයකින් අනෙක් සියල්ල සෑදිය හැක / චිප නිර්මාණය සරල කරයි (1); NAND සහ NOR (1).

4. NAND ද්වාර පමණක් භාවිතයෙන් NOT සහ AND ලබා ගන්නා ආකාරය පෙන්වන්න. *(ලකුණු 2)*

> NOT: A NAND A = A′ (1); AND: (A NAND B) පසුව ප්‍රතිඵලය එයා සමඟම NAND කර පෙරළන්න → A·B (2).

5. F = A · B + A′ · C සඳහා ABC = 011 සහ ABC = 101 වූ විට F සොයන්න. *(ලකුණු 2)*

> 011: A=0,B=1,C=1 → A·B=0, A′·C=1 → F=1 (1); 101: A=1,B=0,C=1 → A·B=0, A′·C=0 → F=0 (1).

සාරාංශය

- තර්ක ද්වාරයක් ද්වීමය ආදාන එක් ද්වීමය ප්‍රතිදානයකට සිතියම්ගත කරයි; සත්‍යතා වගුව සෑම අවස්ථාවක්ම ලැයිස්තුගත කරයි.
- NOT පෙරළයි; AND ට සියලු 1 අවශ්‍යයි; OR ට ඕනෑම 1 අවශ්‍යයි; XOR ට වෙනස් ආදාන අවශ්‍යයි.
- NAND, NOR සහ XNOR යනු AND, OR සහ XOR හි පෙරළුණු ආකාර වේ.
- NAND සහ NOR සර්ව වේ – එකක් පමණක්කොට අනෙක් සෑම ද්වාරයක්ම සෑදිය හැක.
- ආදාන n කින් සත්‍යතා වගුවක පේළි 2^n ක් ඇත; ආදාන තුනක් → පේළි 8.

MCQ පිළිතුරු – 4.1

1. **C** – AND සියලු 1 පේළිය සඳහා පමණක් 1 වේ.
2. **E** – NOT 1 න් 0 ට පෙරළයි.
3. **A** – XOR = විශේෂ OR.
4. **D** – NAND = (A · B)′.
5. **C** – තනිවම අනෙක් සියලු ද්වාර සෑදිය හැක්කේ NAND සහ NOR ට පමණි.
6. **B** – සමාන ආදාන → XOR = 0.
7. **A** – ආදාන දෙකම 0 වූ විට පමණක් NOR 1 වේ.
8. **C** – කවය පෙරළීමයි.
9. **E** – $2^3 = 8$.
10. **B** – XNOR සමානතා ද්වාරයයි.
11. **E** – A NAND A = A′.
12. **D** – බූලියානු වීජ ගණිතයේ + යනු OR වේ.
13. **D** – තර්ක ද්වාරය – ද්වීමය ආදාන ගෙන එක් ද්වීමය ප්‍රතිදානයක් ලබා දෙන පරිපථයකි.
14. **A** – සත්‍යතා වගුව – සෑම ආදාන සංයෝජනයක්ම සහ ඊට ගැළපෙන ප්‍රතිදානය ලැයිස්තුගත කරන වගුවකි.
15. **A** – NOT – පෙරළනය: ප්‍රතිදානය එකම ආදානයේ විරුද්ධයයි. F = A′.
16. **E** – AND – සෑම ආදානයක්ම 1 නම් පමණක් ප්‍රතිදානය 1 වේ. F = A · B.

17. **B** – OR – අවම වශයෙන් එක් ආදානයක් 1 නම් ප්‍රතිදානය 1 වේ. F = A + B.
18. **C** – XOR – විශේෂ OR: ආදාන වෙනස් වූ විට ප්‍රතිදානය 1 වේ. F = A ⊕ B.
19. **B** – NAND – NOT-AND: F = (A · B)′. සර්ව ද්වාරයකි.
20. **D** – NOR – NOT-OR: F = (A + B)′. සර්ව ද්වාරයකි.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

12 ශ්‍රේණිය A/L ICT · පාඩම 4

4.2 බූලියානු වීජ ගණිතය සහ Karnaugh සිතියම්: නියම, SOP, POS සහ සරල කිරීම

විෂය නිර්දේශය 4.2. සංඛ්‍යාංක පරිපථ **බූලියානු වීජ ගණිතය** – 0 සහ 1 හි ගණිතය – අනුගමනය කරයි. මෙම කොටසින් **පූර්විකා සහ නියම** (විනිමය, සහායක, ව්‍යාප්ත, අනන්‍යතා, අතිරික්ත, De Morgan), **ගුණිතවල එකතුව (SOP)** සහ **එකතුවල ගුණිතය (POS)** ආකාරයෙන් ප්‍රකාශන ලිවීම, ප්‍රමේය මගින් **සරල කිරීම**, සහ **Karnaugh සිතියම (K-map)** මගින් 1 කාණ්ඩ කර ඇසින් සරල ප්‍රකාශනයක් කියවීම ඉගෙන ගනී.

1. බූලියානු ප්‍රකාශන සරල කරන්නේ ඇයි?

සරල ප්‍රකාශනයකට **අඩු ද්වාර** අවශ්‍ය වේ, එබැවින් පරිපථය **ලාභදායක, කුඩා, සිසිල් සහ වේගවත්** වේ. විභාගය බොහෝ විට නියම මගින් හෝ K-map මගින් සරල කිරීමට ඉල්ලයි.

2. පූර්විකා සහ නියම

බූලියානු වීජ ගණිතය **0 සහ 1** අගයන් පමණක් භාවිත කරයි. මූලික මෙහෙයුම් **AND (·)**, **OR (+)** සහ **NOT (′)** වේ. ප්‍රයෝජනවත් නියම:



බූලියානු වීජ ගණිත නියම

නියමය	AND ආකාරය	OR ආකාරය
විනිමය	A · B = B · A	A + B = B + A
සහායක	(A · B) · C = A · (B · C)	(A + B) + C = A + (B + C)
ව්‍යාප්ත	A · (B + C) = A·B + A·C	A + (B · C) = (A+B)·(A+C)
අනන්‍යතා	A · 1 = A	A + 0 = A
ශූන්‍ය	A · 0 = 0	A + 1 = 1
සමාන	A · A = A	A + A = A
අනුපූරක	A · A′ = 0	A + A′ = 1
අතිරික්ත	A + A·B = A	A · (A + B) = A
De Morgan	(A · B)′ = A′ + B′	(A + B)′ = A′ · B′

💡 **De Morgan වචන වලින්:** තීරුව කඩා ලකුණ වෙනස් කරන්න – AND තීරුව අනුපූරකවල OR වේ; OR තීරුව අනුපූරකවල AND වේ.

🧮 **විසඳූ උදාහරණය: A = 1, B = 0 සඳහා De Morgan පරීක්ෂා කරන්න: (A · B)′ = A′ + B′ ද?**
(1 · 0)′ = 0′ = 1. A′ + B′ = 0 + 1 = 1. සමානයි – De Morgan සත්‍යය.

3. සම්මත ආකාර: SOP සහ POS

ගුණිතවල එකතුව (SOP) යනු AND පදවල OR එකකි, උදා: A·B + A′·C. සෑම විචල්‍යයක්ම (හෝ එහි අනුපූරකය) එක් වරක් ඇති AND පදයක් **මිනිටරම්** වේ. **එකතුවල ගුණිතය (POS)** යනු OR පදවල AND එකකි, උදා: (A+B)·(A′+C). සෑම විචල්‍යයක්ම ඇති OR පදයක් **මැක්ස්ටරම්** වේ.

සත්‍යතා වගුවකින්: F = 1 වන සෑම පේළියකටම එක් මිනිටරමක් ලියා ඒවා OR කරන්න → **SOP**. F = 0 වන සෑම පේළියකටම එක් මැක්ස්ටරමක් ලියා ඒවා AND කරන්න → **POS**. එකම ශ්‍රිතයේ SOP සහ POS සමානය.

🧮 **විසඳූ උදාහරණය: F(A,B,C) මිනිටරම් 1, 3, 5, 7 සඳහා 1 වේ. SOP ලියන්න.**
m1=A′B′C, m3=A′BC, m5=AB′C, m7=ABC. SOP = A′B′C + A′BC + AB′C + ABC. මෙය **C** දක්වා සරල වේ.

4. බූලියානු ප්‍රමේය මගින් සරල කිරීම

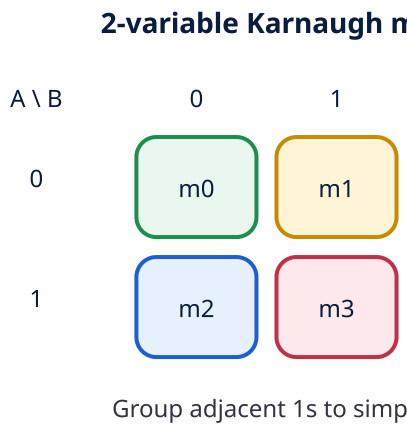
🧮 **විසඳූ උදාහරණය: F = A · B + A · B′ සරල කරන්න.**
A පොදු කරන්න: F = A · (B + B′) = A · 1 = **A**.

🧮 **විසඳූ උදාහරණය: F = A + A′ · B සරල කරන්න.**
A + A′·B = **A + B** (අතිරික්ත / ව්‍යාප්ත නියම).

🧮 **විසඳූ උදාහරණය: F = (A + B) · (A + B′) සරල කරන්න.**
ව්‍යාප්ත: A + (B · B′) = A + 0 = **A**.

5. Karnaugh සිතියම්

Karnaugh සිතියම යනු **අසල්වැසි කොටු බිටු එකකින් පමණක් වෙනස්** වන සේ නැවත ඇඳි සත්‍යතා වගුවකි. 1, 2, 4 හෝ 8 එකතුවක කාණ්ඩ (දෙකේ බල) වට කරන්න. සෑම කාණ්ඩයක්ම නොවෙනස් විචල්‍ය සහිත එක් ගුණිත පදයක් වේ. විශාල කාණ්ඩ → අඩු ලිතරල් → සරල පරිපථය.



විචල්‍ය 2 ක Karnaugh සිතියම

විචල්‍ය 2 K-map: පේළි A=0,1; තීරු B=0,1. කොටු m0..m3. **විචල්‍ය 3 K-map:** සාමාන්‍යයෙන් පේළි A, තීරු BC Gray කේත අනුපිළිවෙල 00, 01, 11, 10 ලෙස තබයි එවිට අසල්වැසියන් තවමත් බිටු එකකින් වෙනස් වේ. දාර වටා එකතු වේ.

🧮 **විසඳූ උදාහරණය: K-map එකකින් F(A,B) = Σm(1,2,3) සරල කරන්න.**
කොටු m1,m2,m3 = 1. m1+m3 කාණ්ඩය (B=1) → B. m2+m3 කාණ්ඩය (A=1) → A. ප්‍රතිඵලය: **A + B**.

🧮 **විසඳූ උදාහරණය: K-map එකකින් F(A,B,C) = Σm(0,1,2,3,6) සරල කරන්න.**
A=0 සහිත කොටු හතර (m0–m3) → **A′**. m6 සහ m2 → **B · C′**. ප්‍රතිඵලය: **A′ + B·C′**.

💡 කාණ්ඩ 3 ක් හෝ 5 ක් සාදන්න එපා. 1, 2, 4, 8, … පමණි. සෑම 1 ක්ම අවම වශයෙන් එක් වරක් ආවරණය කරන්න; අතිච්ඡාදනයට ඉඩ ඇත; 0 වට නොකරන්න.

🚀 **විෂය නිර්දේශයෙන් ඔබ්බට (2026 යාවත්කාලීනය)** *විභාගයට අවශ්‍ය නැත – දැනුම සඳහා කියවන්න.* 2026 දී තර්ක සංශ්ලේෂණ මෙවලම් (Synopsys, Cadence, විවෘත මූලාශ්‍ර Yosys) තවමත් පසුබිමේ බූලියානු වීජ ගණිතය සහ Espresso වැනි අවම කිරීම භාවිත කර, ප්‍රතිඵලය සම්මත-කොටස් ද්වාර මිලියන ගණනකට සිතියම්ගත කරයි. අතින් K-map ඉගෙනීම එම මෙවලම් කරන දේ තේරුම් ගැනීමට උදව් වේ.

ප්‍රධාන පද

බූලියානු වීජ ගණිතය – AND, OR සහ NOT සමඟ 0 සහ 1 අගයන්ගේ වීජ ගණිතය.

විනිමය නියමය – අනුපිළිවෙල වැදගත් නැත: A·B = B·A සහ A+B = B+A.

සහායක නියමය – AND හෝ OR සඳහා කාණ්ඩ කිරීම වැදගත් නැත.

ව්‍යාප්ත නියමය – A·(B+C) = A·B+A·C සහ A+(B·C) = (A+B)·(A+C).

De Morgan නියම – (A·B)′ = A′+B′ සහ (A+B)′ = A′·B′.

SOP – ගුණිතවල එකතුව – AND පදවල OR එකකි.

POS – එකතුවල ගුණිතය – OR පදවල AND එකකි.

මිනිටරම් – සෑම විචල්‍යයක්ම එක් වරක් (සත්‍ය හෝ අනුපූරක) ඇතුළත් ගුණිත පදයකි.

මැක්ස්ටරම් – සෑම විචල්‍යයක්ම එක් වරක් ඇතුළත් එකතු පදයකි.

Karnaugh සිතියම – බූලියානු ශ්‍රිතයක් සරල කිරීමට අසල්වැසි 1 කාණ්ඩ කරන ජාලයකි.

ලිතරල් – විචල්‍යයක් හෝ එහි අනුපූරකය (A හෝ A′).

අතිරික්ත නියමය – A + A·B = A (සහ ද්විත්ව A·(A+B) = A).

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. De Morgan ගේ නියම දෙක ප්‍රකාශ කරන්න.

පිළිතුර: (A · B)′ = A′ + B′ සහ (A + B)′ = A′ · B′.

2. SOP ආකාරය යනු කුමක්ද?

පිළිතුර: ගුණිතවල එකතුව: ශ්‍රිතය AND පදවල (මිනිටරම්) OR එකක් ලෙස ලියයි.

3. බූලියානු ප්‍රකාශන සරල කරන්නේ ඇයි?

පිළිතුර: අඩු ද්වාර → කුඩා, ලාභදායක, සිසිල් සහ බොහෝ විට වේගවත් පරිපථ.

4. A · B + A · B′ සරල කරන්න.

පිළිතුර: A · (B + B′) = A · 1 = A.

5. K-map එකක අනුමත කාණ්ඩ ප්‍රමාණ මොනවාද?

පිළිතුර: දෙකේ බල: 1, 2, 4, 8, … (3 හෝ 5 කිසි විටෙකත් නැත).

6. සත්‍යතා වගුවකින් POS ප්‍රකාශනයක් ලබා ගන්නේ කෙසේද?

පිළිතුර: F = 0 වන සෑම පේළියකටම මැක්ස්ටරමක් ලියා ඒවා AND කරන්න.

බහුවරණ ප්‍රශ්න

1. De Morgan: (A · B)′ සමාන වන්නේ

A. A′ + B′
B. A + B
C. A · B
D. A′ · B′
E. (A + B)′

2. SOP යනු

A. Signed Octal Product
B. Sum of Products
C. Product of Sums
D. Sum of Powers
E. Single Output Process

3. POS යනු

A. Product of Sums
B. Product of Squares
C. Sum of Products
D. Positive Output Signal
E. Parallel Output Sum

4. A · 1 සමාන වන්නේ

A. 0
B. A · A
C. A
D. 1
E. A′

5. A + A′ සමාන වන්නේ

A. 0
B. 1
C. A
D. A′
E. A · A′

6. A · B + A · B′ සරල කරන්න

A. B
B. A · B
C. 0
D. A
E. A + B

7. K-map කාණ්ඩයක 1 කීයක් තිබිය හැකිද?

A. 1, 2, 4 or 8
B. Any odd number
C. Only 4
D. 3 or 5
E. Only 2

8. අසල්වැසි K-map කොටු වෙනස් වන්නේ

A. all bits
B. three bits
C. two bits
D. no bits
E. exactly one bit

9. A + A · B සරල වන්නේ

A. A
B. B
C. 1
D. A · B
E. A + B

10. විචල්‍ය දෙකක් සඳහා F = Σm(1,2,3) සරල වන්නේ

A. A′
B. A + B
C. B′
D. A · B
E. 1

11. (A + B)′ සමාන වන්නේ

A. A′
B. A + B
C. A′ + B′
D. A′ · B′
E. A · B

12. විචල්‍ය තුනක් සඳහා මිනිටරමක ලිතරල් කීයක්ද?

A. 1
B. 8
C. 4
D. 2
E. 3

13. “AND, OR සහ NOT සමඟ 0 සහ 1 අගයන්ගේ වීජ ගණිතය” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. මිනිටරම්
B. සහායක නියමය
C. ලිතරල්
D. SOP
E. බූලියානු වීජ ගණිතය

14. “අනුපිළිවෙල වැදගත් නැත: A·B = B·A සහ A+B = B+A” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. මැක්ස්ටරම්
B. විනිමය නියමය
C. අතිරික්ත නියමය
D. ව්‍යාප්ත නියමය
E. POS

15. “AND හෝ OR සඳහා කාණ්ඩ කිරීම වැදගත් නැත” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. බූලියානු වීජ ගණිතය
B. De Morgan නියම
C. මිනිටරම්
D. සහායක නියමය
E. Karnaugh සිතියම

16. “A·(B+C) = A·B+A·C සහ A+(B·C) = (A+B)·(A+C)” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. විනිමය නියමය
B. මැක්ස්ටරම්
C. ව්‍යාප්ත නියමය
D. ලිතරල්
E. SOP

17. “(A·B)′ = A′+B′ සහ (A+B)′ = A′·B′” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. POS
B. අතිරික්ත නියමය
C. සහායක නියමය
D. De Morgan නියම
E. Karnaugh සිතියම

18. “ගුණිතවල එකතුව – AND පදවල OR එකකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. ව්‍යාප්ත නියමය
B. මිනිටරම්
C. SOP
D. ලිතරල්
E. බූලියානු වීජ ගණිතය

19. “එකතුවල ගුණිතය – OR පදවල AND එකකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. මැක්ස්ටරම්
B. විනිමය නියමය
C. POS
D. De Morgan නියම
E. අතිරික්ත නියමය

20. “සෑම විචල්‍යයක්ම එක් වරක් (සත්‍ය හෝ අනුපූරක) ඇතුළත් ගුණිත පදයකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. Karnaugh සිතියම
B. බූලියානු වීජ ගණිතය
C. සහායක නියමය
D. SOP
E. මිනිටරම්

ව්‍යුහගත ප්‍රශ්න

1. AND සහ OR සඳහා විනිමය, සහායක සහ ව්‍යාප්ත නියම ප්‍රකාශ කරන්න. *(ලකුණු 3)*

> විනිමය: A·B=B·A, A+B=B+A (1); සහායක: (A·B)·C=A·(B·C), + සඳහාද (1); ව්‍යාප්ත: A·(B+C)=A·B+A·C සහ A+(B·C)=(A+B)·(A+C) (1).

2. SOP සහ POS පැහැදිලි කර සත්‍යතා වගුවකින් එකක් ලබා ගන්නා ආකාරය කියන්න. *(ලකුණු 3)*

> SOP = F=1 පේළිවල මිනිටරම්වල OR (1); POS = F=0 පේළිවල මැක්ස්ටරම්වල AND (1); දෙකම එකම ශ්‍රිතය නිරූපණය කරයි (1).

3. බූලියානු නියම මගින් F = A + A′ · B සරල කරන්න. පියවර පෙන්වන්න. *(ලකුණු 2)*

> A + A′·B → A + B (පියවර සහිතව) (2); අවසාන පිළිතුර A + B (1).

4. විචල්‍ය 2 K-map භාවිතයෙන් F = Σm(1,2,3) සරල කරන්න. *(ලකුණු 3)*

> m1,m2,m3 හි 1 තබන්න (1); කාණ්ඩ → A සහ B (1); F = A + B (1).

5. De Morgan යොදා (A · B′)′ සරල කරන්න. *(ලකුණු 2)*

> (A · B′)′ = A′ + (B′)′ (1); (B′)′ = B එබැවින් A′ + B (1).

සාරාංශය

- බූලියානු නියම (විනිමය, සහායක, ව්‍යාප්ත, De Morgan, අනන්‍යතා, අතිරික්ත) ප්‍රකාශන නැවත ලිවීමට ඉඩ දෙයි.
- SOP යනු ගුණිතවල OR; POS යනු එකතුවල AND – දෙකම සත්‍යතා වගුවෙන් පැමිණේ.
- සරල කිරීම ද්වාර ගණන අඩු කරයි, එබැවින් පරිපථ ලාභදායක සහ වේගවත් වේ.
- K-map අසල්වැසි 1 කාණ්ඩ 1, 2, 4, 8 ප්‍රමාණයෙන් කාණ්ඩ කර අවම SOP කියවයි.
- De Morgan: තීරුව කඩා මෙහෙයුම්කරු වෙනස් කරන්න.

MCQ පිළිතුරු – 4.2

1. **A** – තීරුව කඩා · න් + ට වෙනස් කරන්න.

2. **B** – AND පදවල OR.
3. **A** – OR පදවල AND.
4. **C** – අනන්‍යතා නියමය.
5. **B** – අනුපූරක නියමය.
6. **D** – A(B+B′)=A.
7. **A** – දෙකේ බල පමණි.
8. **E** – Gray කේත අසල්වාසිත්වය.
9. **A** – අතිරික්ත නියමය.
10. **B** – K-map කාණ්ඩ A + B දෙයි.
11. **D** – OR සඳහා De Morgan.
12. **E** – විචල්‍යයකට එක් ලිතරලයක්.
13. **E** – බූලියානු වීජ ගණිතය – AND, OR සහ NOT සමඟ 0 සහ 1 අගයන්ගේ වීජ ගණිතය.
14. **B** – විනිමය නියමය – අනුපිළිවෙල වැදගත් නැත: A·B = B·A සහ A+B = B+A.
15. **D** – සහායක නියමය – AND හෝ OR සඳහා කාණ්ඩ කිරීම වැදගත් නැත.
16. **C** – ව්‍යාප්ත නියමය – A·(B+C) = A·B+A·C සහ A+(B·C) = (A+B)·(A+C).
17. **D** – De Morgan නියම – (A·B)′ = A′+B′ සහ (A+B)′ = A′·B′.
18. **C** – SOP – ගුණිතවල එකතුව – AND පදවල OR එකකි.
19. **C** – POS – එකතුවල ගුණිතය – OR පදවල AND එකකි.
20. **E** – මිනිටරම් – සෑම විචල්‍යයක්ම එක් වරක් (සත්‍ය හෝ අනුපූරක) ඇතුළත් ගුණිත පදයකි.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

12 ශ්‍රේණිය A/L ICT · පාඩම 4

4.3 සංඛ්‍යාංක පරිපථ නිර්මාණය: එදිනෙදා අවශ්‍යතාවල සිට තර්ක ද්වාර දක්වා

විෂය නිර්දේශය 4.3. දැන් ඔබ කුඩා පරිපථ **නිර්මාණය** කරයි. සැබෑ අවස්ථාවක් තෝරා (දොර එලාමයක්, බහුතර ඡන්දයක්, මෝටර් රථ seatbelt එළියක්), එහි **සත්‍යතා වගුව** ලියා (ආදාන තුනක් දක්වා), වගුව **බූලියානු ප්‍රකාශනයකට** හරවා, සරල කර, අවසානයේ **ද්වාර පරිපථය අඳින්න**. ඉංජිනේරුවන් සෑම සංඛ්‍යාංක නිර්මාණයක්ම අරඹන්නේ මෙසේය.

1. නිර්මාණ පියවර

අවශ්‍යතාවක සිට ක්‍රියාකාරී පරිපථයක් දක්වා පැහැදිලි පියවර හතරක් ඇත:



අවශ්‍යතාව → සත්‍යතා වගුව → ප්‍රකාශනය → පරිපථය

1. අවශ්‍යතාව හඳුනාගන්න
එදිනෙදා වචන වලින් ආදාන සහ එක් ප්‍රතිදානය නම් කරන්න.

2. සත්‍යතා වගුව
සෑම ආදාන මිශ්‍රණයක්ම (පේළි 2^n) සහ අවශ්‍ය ප්‍රතිදානය ලැයිස්තුගත කරන්න.

3. ප්‍රකාශනය
SOP (හෝ POS) ලියා නියම හෝ K-map මගින් සරල කරන්න.

4. පරිපථය අඳින්න
සෑම · AND ලෙසද, සෑම + OR ලෙසද, සෑම ′ NOT ලෙසද ප්‍රතිස්ථාපනය කරන්න.

2. උදාහරණය A – රාත්‍රියේ දොර එලාමය

නිවසක එලාමය **දොර විවෘත (D = 1)** සහ **රාත්‍රිය (N = 1)** වූ විට නාද විය යුතුය. දවසේ හෝ දොර වැසී ඇති විට එලාමය නිහඬව තිබේ.

🧱 විසඳූ උදාහරණය: දොර එලාමය සඳහා සත්‍යතා වගුව සහ බූලියානු ප්‍රකාශනය ලියන්න.
DN: 00→0, 01→0, 10→0, 11→1. එබැවින් F = **D · N**. පරිපථය එක් **AND** ද්වාරයකි.

D (දොර)	N (රාත්‍රිය)	එලාම F
0	0	0
0	1	0

1	0	0
1	1	1

3. උදාහරණය B – ඡන්ද තුනක බහුතරය

පුද්ගලයින් තිදෙනෙක් A, B, C ඔව් (1) හෝ නැත (0) ඡන්දය දෙති. අවම වශයෙන් දෙදෙනෙක් ඔව් කිවූ විට තීරණය F ඔව් වේ – **බහුතර** ශ්‍රිතය.

> 🧱 **විසඳූ උදාහරණය: A, B, C හි බහුතරය සඳහා F ලියා F = 1 වන විට ලැයිස්තුගත කරන්න.**
> F = A·B + B·C + A·C. F = 1 වන්නේ ABC = 011, 101, 110, 111 (1 දෙකක් හෝ තුනක් ඇති පේළි).

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

> 💡 බහුතර පරිපථය **AND ද්වාර තුනක්** (යුගලයකට එකක්) සහ ඒවායේ ප්‍රතිදාන එකතු කරන **OR ද්වාර එකක්** භාවිත කරයි. එය සම්පූර්ණ එකතුකරන්නා තුළ carry තර්කයද වේ (4.4).

4. උදාහරණය C – Seatbelt අනතුරු ඇඟිලිය

මෝටර් රථ අනතුරු ඇඟිලිය **එන්ජිම ක්‍රියාත්මක (E = 1)** සහ **seatbelt අමුණා නැත (S = 0)** වූ විට දැල්වේ. එබැවින් F = E · S′.

> 🧱 **විසඳූ උදාහරණය: F = E · S′ සඳහා සත්‍යතා වගුව අඳින්න සහ අවශ්‍ය ද්වාර නම් කරන්න.**
> ES: 00→0, 01→0, 10→1, 11→0. ද්වාර: S මත **NOT** එකක්, පසුව E සමඟ **AND** එකක්.

5. ප්‍රකාශනයක් පරිපථයක් බවට හරවන ආකාරය

ප්‍රකාශනයක් වමේ සිට දකුණට කියවීම: සෑම ′ NOT කවයක් (හෝ NOT ද්වාරයක්) වේ, සෑම · AND ද්වාරයක් වේ, සෑම + OR ද්වාරයක් වේ. ආදාන වමේද, ප්‍රතිදානය දකුණේද අඳින්න. එකම පෙරළුණු සංඥාව දෙවරක් භාවිත වේ නම් NOT ද්වාර බෙදා ගන්න.

🧱 **විසඳූ උදාහරණය: F = A·B + A′·C සඳහා සරල කිරීම අවශ්‍ය නොවන විට අවශ්‍ය ද්වාර ලැයිස්තුගත කරන්න.**
NOT එකක් (A′ සඳහා), AND ද්වාර දෙකක් (A·B සහ A′·C), ඒවා එකතු කරන OR ද්වාර එකක් – මුළු **ද්වාර 4**.

💡 ඇඳීමට පෙර සැමවිටම සරල කරන්න. F = A·B + A·B′ AND දෙකක් සහ OR එකක් ලෙස පෙනේ (ද්වාර 3), නමුත් එය හුදෙක් A වේ.

🚀 **විෂය නිර්දේශයෙන් ඔබ්බට (2026 යාවත්කාලීනය)** *විභාගයට අවශ්‍ය නැත – දැනුම සඳහා කියවන්න.* සැබෑ නිෂ්පාදන මෙම කුඩා නිර්මාණ HDL කේතයට (Verilog / VHDL) එකතු කරයි. 2026 දී ශිෂ්‍යයින් සහ නිර්මාණකරුවන් අඩු මිල FPGA පුවරු (උදා: Lattice iCE40, Xilinx / AMD Spartan) භාවිතයෙන් සත්‍යතා වගුවක් download කර කඩදාසියේ නිර්මාණය කළ පරිපථයට LED අනුගමනය කරන ආකාරය නරඹති.

ප්‍රධාන පද

සංඛ්‍යාංක පරිපථ නිර්මාණය – සැබෑ අවශ්‍යතාවක් සත්‍යතා වගුවක්, ප්‍රකාශනයක් සහ ද්වාර පරිපථයක් බවට හැරවීම.

ආදාන විචල්‍යය – පරිපථයකට පෝෂණය වන ද්වීමය සංඥාවක් (දොර, රාත්‍රිය, ඡන්දය, ...).

ප්‍රතිදාන ශ්‍රිතය – පරිපථය නිපදවිය යුතු එකම ද්වීමය ප්‍රතිඵලය.

බහුතර ශ්‍රිතය – ආදාන තුනෙන් අවම වශයෙන් දෙකක් 1 වූ විට ප්‍රතිදානය 1 වේ.

එලාම පරිපථය – උදා: F = D · N (දොර විවෘත සහ රාත්‍රිය).

ද්වාර-මට්ටමේ පරිපථය – සෑම AND, OR සහ NOT ද්වාරයක්ම පෙන්වන ඇඳීමකි.

ඇඳීමට පෙර සරල කිරීම – අඩු ද්වාර භාවිත වන සේ ප්‍රකාශනය අඩු කරන්න.

ආදාන තුනේ නිර්මාණය – විෂය නිර්දේශ සීමාව: ආදාන තුනක් දක්වා සත්‍යතා වගු (පේළි 8).

Seatbelt අනතුරු ඇඟවීම – F = E · S′ – එන්ජිම ක්‍රියාත්මක සහ පටිය අමුණා නැත.

බෙදාගත් පෙරළුනය – එකම පෙරළුණු සංඥාව දෙවරක් අවශ්‍ය වූ විට නැවත භාවිත කරන NOT ද්වාරයක්.

යෙදුම – තර්ක පරිපථයකට පාලනය කළ හැකි සැබෑ ජීවිත අවස්ථාවක්.

බූලියානුවෙන් ද්වාර වෙත – · → AND, + → OR, ′ → NOT.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. සංඛ්‍යාංක පරිපථ නිර්මාණයේ පියවර හතර ලැයිස්තුගත කරන්න.

> **පිළිතුර:** අවශ්‍යතාව → සත්‍යතා වගුව → බූලියානු ප්‍රකාශනය (සරල කළ) → ද්වාර අඳින්න.

2. රාත්‍රියේ දොර විවෘත එලාමයේ බූලියානු ප්‍රකාශනය කුමක්ද?

> **පිළිතුර:** F = D · N.

3. ආදාන තුනක බහුතර ශ්‍රිතය ලියන්න.

> **පිළිතුර:** F = A·B + B·C + A·C.

4. ආදාන 3 ක නිර්මාණ සත්‍යතා වගුවක පේළි කීයක්ද?

> **පිළිතුර:** පේළි 8 ක්.

5. පරිපථය ඇඳීමට පෙර සරල කරන්නේ ඇයි?

> **පිළිතුර:** සරල ප්‍රකාශනයකට අඩු ද්වාර අවශ්‍ය වේ, එබැවින් පරිපථය ලාභදායක සහ කුඩා වේ.

6. F = E · S′ සඳහා අවශ්‍ය ද්වාර මොනවාද?

> **පිළිතුර:** S මත NOT එකක් සහ E සමඟ AND එකක්.

බහුවරණ ප්‍රශ්න

1. සංඛ්‍යාංක පරිපථයක් නිර්මාණය කිරීමේ පළමු පියවර

A. write POS only
B. identify the need and name inputs/output
C. buy an FPGA
D. draw random gates
E. skip the truth table

2. දොර එලාම F = D · N භාවිත කරන ද්වාරය

A. XOR
B. NOR
C. AND
D. NOT
E. OR

3. A,B,C හි බහුතරය සමාන වන්නේ

A. A + B + C
B. A′B′C′
C. A ⊕ B ⊕ C
D. AB + BC + AC
E. A · B · C

4. ආදාන 3 ක සත්‍යතා වගුවක

A. 4 rows
B. 8 rows
C. 6 rows
D. 3 rows
E. 9 rows

5. F = E · S′ ට අවශ්‍ය

A. XOR only
B. OR only
C. NOR only
D. NAND only
E. NOT and AND

6. බූලියානු ප්‍රකාශනයක · යනු කුමන ද්වාරයද?

A. OR
B. NOT
C. NOR
D. XOR
E. AND

7. බූලියානු ප්‍රකාශනයක + යනු කුමන ද්වාරයද?

A. OR
B. NOT
C. XOR
D. AND
E. NAND

8. එලාමය සඳහා F 1 වන්නේ

A. always
B. door shut
C. door open and night
D. never
E. daytime

9. බහුතර පරිපථය භාවිත වන්නේ

A. carry logic of a full adder
B. a clock
C. an antenna
D. a NOT gate
E. a memory cell

10. ඇඳීමට පෙර F = A·B + A·B′ සරල කිරීමෙන් ඉතිරි වන්නේ

A. the truth table
B. gates (it becomes just A)
C. nothing
D. inputs
E. De Morgan

11. විෂය නිර්දේශ නිර්මාණ ආදාන කීයකට සීමාද?

A. three
B. five
C. ten
D. eight
E. one

12. බෙදාගත් පෙරළන භාවිත වන්නේ

A. you have spare OR gates
B. there is no truth table
C. POS is forbidden
D. F = 1 always
E. the same inverted signal is needed more than once

13. “සැබෑ අවශ්‍යතාවක් සත්‍යතා වගුවක්, ප්‍රකාශනයක් සහ ද්වාර පරිපථයක් බවට හැරවීම” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. ප්‍රතිදාන ශ්‍රිතය
B. යෙදුම
C. ආදාන තුනේ නිර්මාණය
D. සංඛ්‍යාංක පරිපථ නිර්මාණය
E. ද්වාර-මට්ටමේ පරිපථය

14. “පරිපථයකට පෝෂණය වන ද්වීමය සංඥාවක් (දොර, රාත්‍රිය, ඡන්දය, ...)” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. Seatbelt අනතුරු ඇඟවීම
B. බහුතර ශ්‍රිතය
C. බූලියානුවෙන් ද්වාර වෙත
D. ආදාන විචල්‍යය
E. ඇඳීමට පෙර සරල කිරීම

15. “පරිපථය නිපදවිය යුතු එකම ද්වීමය ප්‍රතිඵලය” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. ආදාන තුනේ නිර්මාණය
B. බෙදාගත් පෙරළුනය
C. ප්‍රතිදාන ශ්‍රිතය
D. සංඛ්‍යාංක පරිපථ නිර්මාණය
E. ඵලාම පරිපථය

16. “ආදාන තුනෙන් අවම වශයෙන් දෙකක් 1 වූ විට ප්‍රතිදානය 1 වේ” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. බහුතර ශ්‍රිතය
B. යෙදුම
C. ද්වාර-මට්ටමේ පරිපථය
D. Seatbelt අනතුරු ඇඟවීම
E. ආදාන විචල්‍යය

17. “උදා: F = D · N (දොර විවෘත සහ රාත්‍රිය)” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. බෙදාගත් පෙරළුනය
B. ප්‍රතිදාන ශ්‍රිතය
C. ඵලාම පරිපථය
D. බූලියානුවෙන් ද්වාර වෙත
E. ඇඳීමට පෙර සරල කිරීම

18. “සෑම AND, OR සහ NOT ද්වාරයක්ම පෙන්වන ඇඳීමකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. බහුතර ශ්‍රිතය
B. ආදාන තුනේ නිර්මාණය
C. යෙදුම
D. ද්වාර-මට්ටමේ පරිපථය
E. සංඛ්‍යාංක පරිපථ නිර්මාණය

19. “අඩු ද්වාර භාවිත වන සේ ප්‍රකාශනය අඩු කරන්න” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. Seatbelt අනතුරු ඇඟවීම
B. ඇඳීමට පෙර සරල කිරීම
C. ඵලාම පරිපථය
D. ආදාන විචල්‍යය
E. බූලියානුවෙන් ද්වාර වෙත

20. “විෂය නිර්දේශ සීමාව: ආදාන තුනක් දක්වා සත්‍යතා වගු (පේළි 8)” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. සංඛ්‍යාංක පරිපථ නිර්මාණය
B. ද්වාර-මට්ටමේ පරිපථය
C. බෙදාගත් පෙරළනය
D. ප්‍රතිදාන ශ්‍රිතය
E. ආදාන තුනේ නිර්මාණය

ව්‍යුහගත ප්‍රශ්න

1. කෙටි උදාහරණයක් සමඟ නිර්මාණ පියවර හතර විස්තර කරන්න. *(ලකුණු 4)*

> අවශ්‍යතාව (උදා: රාත්‍රි දොර ඵලාමය) (1); සත්‍යතා වගුව (1); ප්‍රකාශනය F=D·N (1); AND ද්වාරය අඳින්න (1).

2. ස්විච A හෝ B (හෝ දෙකම) ON වූ විට එළිය ON වන සත්‍යතා වගුව සහ ප්‍රකාශනය සාදන්න. *(ලකුණු 3)*

> AB: 00→0, 01→1, 10→1, 11→1 (1); F = A + B (1); පරිපථය = OR ද්වාරය (1).

3. බහුතර ශ්‍රිතය සඳහා F ප්‍රකාශ කර ABC = 101 සහ 100 න් කුමක් සඳහා ප්‍රතිදානය 1 දැයි කියන්න. *(ලකුණු 3)*

> F = AB+BC+AC (1); 101 → 1 (1); 100 → 0 (1).

4. F = E · S′ (seatbelt අනතුරු ඇඟවීම) නිර්මාණය කරන්න: සත්‍යතා වගුව, ද්වාර. *(ලකුණු 2)*

> වගුව ES: 00→0, 01→0, 10→1, 11→0 (2); ද්වාර NOT + AND (1).

5. ඇඳීමට පෙර A·B + A·B′ සරල කළ යුත්තේ ඇයිදැයි පැහැදිලි කර සරල ආකාරය කියන්න. *(ලකුණු 3)*

> සරල නොකළ විට AND 2 + OR 1 (1); සරල කළ විට A – කම්බියක් පමණි (1); ද්වාර/පිරිවැය ඉතිරි වේ (1).

සාරාංශය

- නිර්මාණ ප්‍රවාහය: අවශ්‍යතාව → සත්‍යතා වගුව → ප්‍රකාශනය සරල කරන්න → ද්වාර අඳින්න.
- දොර ඵලාමය: F = D · N (AND එකක්).
- බහුතරය: F = AB + BC + AC (සම්පූර්ණ එකතුකරන්නාගේ carry ද).
- Seatbelt අනතුරු ඇඟවීම: F = E · S′ (NOT පසු AND).
- විෂය නිර්දේශ නිර්මාණ ආදාන තුනක් දක්වා භාවිත කරයි (පේළි 8 වගු).

MCQ පිළිතුරු – 4.3

1. **B** – සැබෑ ලෝක අවශ්‍යතාවෙන් අරඹන්න.
2. **C** – දොර සහ රාත්‍රියේ AND.
3. **D** – අවම වශයෙන් ආදාන දෙකක් 1.
4. **B** – $2^3 = 8$.
5. **E** – S පෙරළා E සමඟ AND.
6. **E** – · යනු AND.
7. **A** – + යනු OR.
8. **C** – D සහ N දෙකම 1 විය යුතුය.
9. **A** – Cout = බහුතර(A,B,Cin).
10. **B** – $A{\cdot}B + A{\cdot}B' = A$.
11. **A** – ආදාන තුනක් දක්වා.
12. **E** – NOT එකක් නැවත භාවිත කරන්න.
13. **D** – සංඛ්‍යාංක පරිපථ නිර්මාණය – සැබෑ අවශ්‍යතාවක් සත්‍යතා වගුවක්, ප්‍රකාශනයක් සහ ද්වාර පරිපථයක් බවට හැරවීම.
14. **D** – ආදාන විචල්‍යය – පරිපථයකට පෝෂණය වන ද්වීමය සංඥාවක් (දොර, රාත්‍රිය, ඡන්දය, ...).
15. **C** – ප්‍රතිදාන ශ්‍රිතය – පරිපථය නිපදවිය යුතු එකම ද්වීමය ප්‍රතිඵලය.
16. **A** – බහුතර ශ්‍රිතය – ආදාන තුනෙන් අවම වශයෙන් දෙකක් 1 වූ විට ප්‍රතිදානය 1 වේ.
17. **C** – එලාම පරිපථය – උදා: F = D · N (දොර විවෘත සහ රාත්‍රිය).
18. **D** – ද්වාර-මට්ටමේ පරිපථය – සෑම AND, OR සහ NOT ද්වාරයක්ම පෙන්වන ඇඳීමකි.
19. **B** – ඇඳීමට පෙර සරල කිරීම – අඩු ද්වාර භාවිත වන සේ ප්‍රකාශනය අඩු කරන්න.
20. **E** – ආදාන තුනේ නිර්මාණය – විෂය නිර්දේශ සීමාව: ආදාන තුනක් දක්වා සත්‍යතා වගු (පේළි 8).

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

12 ශ්‍රේණිය A/L ICT · පාඩම 4

4.4 අර්ධ එකතුකරන්නා, සම්පූර්ණ එකතුකරන්නා සහ Flip-Flop: CPU කොටස් සහ මතක කොටු

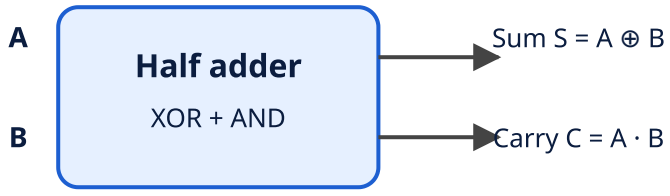
විෂය නිර්දේශය 4.4. CPU එක සංඛ්‍යා එකතු කරන්නේ **එකතුකරන්නා** පරිපථ මගිනි, මතකය බිටු ගබඩා කරන්නේ **flip-flop** මගිනි. මෙම කොටසින් **අර්ධ එකතුකරන්නා** සහ **සම්පූර්ණ එකතුකරන්නා** සඳහා සත්‍යතා වගු සහ බූලියානු ප්‍රකාශන සාදා, සම්පූර්ණ එකතුකරන්නන් ALU එකට දාමගත වන ආකාරය දකින අතර **flip-flop** එකක **ප්‍රතිපෝෂණ වළල්ලක්** බිටු එකක් මතක තබා ගන්නා ආකාරය – රෙජිස්තර සහ RAM හි ගොඩනැගිලි කොටස – ඉගෙන ගනී.

1. CPU හි ගොඩනැගිලි කොටස්

ALU තුළ අංක ගණිතය බිටු දෙකක් එකතු කිරීමෙන් අරඹයි. **අර්ධ එකතුකරන්නා** බිටු දෙකක් එකතු කර එකතුවක් සහ carry එකක් සාදයි. **සම්පූර්ණ එකතුකරන්නා** පෙර තීරුවෙන් **carry-in** ද ගනී, එබැවින් බිටු 8 ක සංඛ්‍යා දෙකක් එකතු කිරීමට සම්පූර්ණ එකතුකරන්නන් අටක් දාමගත කළ හැක.

2. අර්ධ එකතුකරන්නා

ආදාන: A, B. ප්‍රතිදාන: **එකතු S** සහ **Carry C**. ද්වීමය එකතු කිරීම: 0+0=00, 0+1=01, 1+0=01, 1+1=10. එබැවින් ආදාන වෙනස් වූ විට S 1 වේ → **S = A ⊕ B**. දෙකම 1 වූ විට පමණක් C 1 වේ → **C = A · B**.



අර්ධ එකතුකරන්නා: Sum සඳහා XOR, Carry සඳහා AND

A	B	එකතු S	Carry C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

🧮 **විසඳූ උදාහරණය: A = 1, B = 1 සඳහා අර්ධ එකතුකරන්නාගේ S සහ C සොයන්න.**
S = 1 ⊕ 1 = **0**. C = 1 · 1 = **1**. (ද්විමය 1 + 1 = 10_2.)

💡 අර්ධ එකතුකරන්නාට **Cin නැත**, එබැවින් බහු-බිටු සංඛ්‍යාවක අඩුම වෙසෙසි බිටුව පමණක් එකතු කළ හැක. ඉහළ බිටු සෑම එකකටම සම්පූර්ණ එකතුකරන්නාක් අවශ්‍යයි.

3. සම්පූර්ණ එකතුකරන්නා

ආදාන: A, B, **Cin**. ප්‍රතිදාන: **එකතු S** සහ **Cout**. ආදාන තුනෙන් **ඔත්තේ** ගණනක් 1 වූ විට S 1 වේ → **S = A ⊕ B ⊕ Cin**. අවම වශයෙන් **දෙකක්** 1 වූ විට Cout 1 වේ → **Cout = A·B + B·Cin + A·Cin** (4.3 හි බහුතර ශ්‍රිතය).



S = A ⊕ B ⊕ Cin Cout = majority(A,B,Cin)

Cin සහ Cout සහිත සම්පූර්ණ එකතුකරන්නා

A	B	Cin	එකතු S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

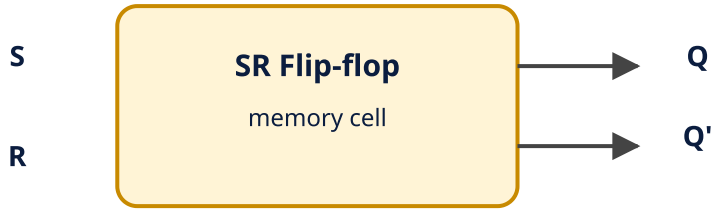
🧮 **විසඳූ උදාහරණය: A = 1, B = 0, Cin = 1 සඳහා S සහ Cout සොයන්න.**
S = 1 ⊕ 0 ⊕ 1 = **0**. Cout = බහුතරය = **1**.

🧮 **විසඳූ උදාහරණය: A = 1, B = 1, Cin = 1 සඳහා S සහ Cout සොයන්න.**
S = 1 ⊕ 1 ⊕ 1 = **1**. Cout = **1**.

සම්පූර්ණ එකතුකරන්නා **අර්ධ එකතුකරන්නන් දෙදෙනෙකු සහ OR එකකින්** සෑදිය හැක: පළමු HA A සහ B එකතු කරයි → S1, C1; දෙවන HA S1 සහ Cin එකතු කරයි → S, C2; පසුව Cout = C1 + C2.

4. බිටු ගබඩා කිරීම – ප්‍රතිපෝෂණ සහ flip-flop

සංයෝජන පරිපථයක ප්‍රතිදානය රඳා පවතින්නේ **වත්මන් ආදාන මත පමණි**. බිටුවක් **මතක තබා ගැනීමට ප්‍රතිපෝෂණ වළල්ලක්** අවශ්‍යයි: ප්‍රතිදානය ආදානයකට ආපසු යවා පරිපථයට තම තත්ත්වය තබා ගත හැකි වේ. **Flip-flop** යනු **බිටු එකක්** ගබඩා කරන මූලික මතක කොටුවයි.



SR flip-flop ප්‍රතිපෝෂණයෙන් බිටු එකක් ගබඩා කරයි

සරල **SR (Set-Reset) flip-flop** එකට ආදාන S සහ R සහ ප්‍රතිදාන Q සහ Q′ ඇත:

- S = 1, R = 0 → **Set**: Q 1 වේ.
- S = 0, R = 1 → **Reset**: Q 0 වේ.
- S = 0, R = 0 → **Hold**: Q තිබුණා විදිහටම තිබේ (මතකය!).
- S = 1, R = 1 → මූලික SR latch එකක **තහනම්** (ප්‍රතිදාන දෙකම ගැටෙයි).

💡 රෙජිස්තර, කෑෂ් සහ static RAM යනු flip-flop එකතුවකි. Dynamic RAM වෙනුවට කුඩා ධාරිත්‍රක භාවිත කරයි, නමුත් අදහස – කොටුවකට බිටු එකක් – සමානය.

🧱 **විසඳූ උදාහරණය: Flip-flop එකක් දැන් Q = 0 තබාගෙන ඇත. (a) එය 1 ට සැකසීමට, (b) 0 ලෙස තබා ගැනීමට S සහ R කුමක්ද?**
(a) S = 1, R = 0 (set). (b) S = 0, R = 0 (hold) – නැත්නම් S = 0, R = 1 මගින් reset.

🚀 **විෂය නිර්දේශයෙන් ඔබ්බට (2026 යාවත්කාලීනය)** *විභාගයට අවශ්‍ය නැත – දැනුම සඳහා කියවන්න.* 2026 දී දුරකථන CPU එකක් තවමත් එහි ALU සහ SIMD ඒකකවල CMOS සම්පූර්ණ-එකතුකරන්නා කොටස් බිලියන ගණනක්ද, ට්‍රාන්සිස්ටර 6 ක flip-flop වැනි කොටු වලින් සෑදූ SRAM කෑෂ්ද භාවිත කරයි. නව MRAM සහ ReRAM සමහර DRAM ප්‍රතිස්ථාපනයට උත්සාහ කරයි, නමුත් විෂය නිර්දේශ අදහස නොවෙනස්ය: එකතුකරන්නන් ගණනය කරයි, flip-flop මතක තබයි.

ප්‍රධාන පද

අර්ධ එකතුකරන්නා – බිටු දෙකක් එකතු කරයි: S = A ⊕ B, C = A · B. Cin නැත.

සම්පූර්ණ එකතුකරන්නා – A, B සහ Cin එකතු කරයි: S = A⊕B⊕Cin, Cout = බහුතරය.

එකතු බිටුව – එකතුකරන්නා තීරුවක ප්‍රතිඵල බිටුව (ආදානවල XOR).

Carry – එකතුව ≥ 2 වූ විට ඊළඟ තීරුවට යන අමතර බිටුව.

Cin / Cout – පෙර තීරුවෙන් carry ඇතුළු වීම; ඊළඟට carry පිටවීම.

Flip-flop – ප්‍රතිපෝෂණයෙන් බිටු එකක් ගබඩා කරන මතක කොටුවකි.

ප්‍රතිපෝෂණ වළල්ල – පරිපථයට තත්ත්වය තබා ගත හැකි වන සේ ප්‍රතිදානය ආදානයකට ආපසු සම්බන්ධ කිරීම.

SR flip-flop – Set-Reset latch: S මගින් Q=1, R මගින් Q=0, 00 තබා ගනී.

Set – Q = 1 කරන්න (S = 1, R = 0).

Reset – Q = 0 කරන්න (S = 0, R = 1).

Hold / මතකය – S = R = 0 පෙර Q තබා ගනී.

රෙජිස්තරය – CPU තුළ බහු-බිටු වචනයක් ගබඩා කරන flip-flop කාණ්ඩයකි.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. අර්ධ එකතුකරන්නා සඳහා බූලියානු ප්‍රකාශන ලියන්න.

> **පිළිතුර:** S = A ⊕ B, C = A · B.

2. බහු-බිටු එකතුකරන්නාකට අර්ධ එකතුකරන්නන් පමණක් නොව සම්පූර්ණ එකතුකරන්නන් අවශ්‍ය වන්නේ ඇයි?

> **පිළිතුර:** ඉහළ බිටු තීරුවලට පෙර තීරුවෙන් carry-in ලබා ගත යුතුය; අර්ධ එකතුකරන්නාට Cin නැත.

3. සම්පූර්ණ එකතුකරන්නා සඳහා S සහ Cout ලියන්න.

> **පිළිතුර:** S = A ⊕ B ⊕ Cin; Cout = A·B + B·Cin + A·Cin.

4. Flip-flop එකක් ගබඩා කරන්නේ කුමක්ද?

> **පිළිතුර:** ප්‍රතිපෝෂණ වළල්ලකින් තබා ගන්නා බිටු එකක්.

5. SR flip-flop එකක S = 1, R = 0 සහ S = R = 0 කරන්නේ කුමක්ද?

> **පිළිතුර:** S=1,R=0 මගින් Q 1 ට සැකසේ; S=R=0 පෙර අගය තබා ගනී.

6. අර්ධ එකතුකරන්නන්ගෙන් සම්පූර්ණ එකතුකරන්නාක් සෑදීමේ ආකාරය කුමක්ද?

> **පිළිතුර:** Carry ප්‍රතිදාන දෙක සඳහා අර්ධ එකතුකරන්නන් දෙදෙනෙක් සහ OR ද්වාරයක්.

බහුවරණ ප්‍රශ්න

1. අර්ධ එකතුකරන්නාගේ එකතුව සමාන වන්නේ

A. A ⊕ B ⊕ Cin
B. A · B
C. A ⊕ B
D. A + B
E. A′

2. අර්ධ එකතුකරන්නාගේ Carry සමාන වන්නේ

A. A · B
B. A + B
C. A ⊕ B ⊕ Cin
D. A′ · B′
E. A ⊕ B

3. සම්පූර්ණ එකතුකරන්නාකට ආදාන කීයක්ද?

A. 1
B. 8
C. 4
D. 2
E. 3 (A, B, Cin)

4. සම්පූර්ණ එකතුකරන්නාගේ එකතුව සමාන වන්නේ

A. A · B
B. A ⊕ B ⊕ Cin
C. A ⊕ B
D. A + B
E. majority

5. සම්පූර්ණ එකතුකරන්නාගේ Cout සමාන වන්නේ

A. A + B + Cin only as OR of three
B. A ⊕ B ⊕ Cin
C. majority of A, B, Cin
D. A′
E. A ⊕ B

6. අර්ධ එකතුකරන්නා මත A=1,B=1 සඳහා S සහ C

A. S=1, C=1
B. S=0, C=0
C. S=1, C=0
D. S=0, C=1
E. S=1, C=undefined

7. Flip-flop එකක් ගබඩා කරන්නේ

A. no data
B. an analog voltage only
C. one word only
D. one bit
E. one byte

8. Flip-flop එකක ප්‍රතිපෝෂණය භාවිත වන්නේ

A. add two numbers
B. hold / remember the state
C. invert a bit only
D. create Cin
E. delete memory

9. SR flip-flop: S=1, R=0 යනු

A. Hold
B. Toggle randomly
C. Forbidden
D. Reset Q = 0
E. Set Q = 1

10. SR flip-flop: S=R=0 යනු

A. Hold previous Q
B. Reset
C. Set
D. Q becomes 1 always
E. Forbidden

11. මූලික SR latch එකක තහනම් වන්නේ

A. S = 1, R = 0
B. S = 1 and R = 1
C. S = 0, R = 1
D. Q = 1
E. S = 0, R = 0

12. බිටු 8 එකතුකරන්නාකට සම්පූර්ණ එකතුකරන්නන් කීයක් (ආසන්න වශයෙන්)ද?

A. 1
B. 16
C. 8
D. 4
E. 2

13. “බිටු දෙකක් එකතු කරයි: S = A ⊕ B, C = A · B. Cin නැත” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. එකතු බිටුව
B. Hold / මතකය
C. Flip-flop
D. SR flip-flop
E. අර්ධ එකතුකරන්නා

14. “A, B සහ Cin එකතු කරයි: S = A⊕B⊕Cin, Cout = බහුතරය” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. සම්පූර්ණ එකතුකරන්නා
B. රෙජිස්තරය
C. Set
D. ප්‍රතිපෝෂණ වළල්ල
E. Carry

15. “එකතුකරන්නා තීරුවක ප්‍රතිඵල බිටුව (ආදානවල XOR)” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. අර්ධ එකතුකරන්නා
B. Reset
C. SR flip-flop
D. එකතු බිටුව
E. Cin / Cout

16. “එකතුව ≥ 2 වූ විට ඊළඟ තීරුවට යන අමතර බිටුව” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. සම්පූර්ණ එකතුකරන්නා
B. Flip-flop
C. Hold / මතකය
D. Carry
E. Set

17. “පෙර තීරුවෙන් carry ඇතුළු වීම; ඊළඟට carry පිටවීම” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. ප්‍රතිපෝෂණ වළල්ල
B. රෙජිස්තරය
C. එකතු බිටුව
D. Reset
E. Cin / Cout

18. “ප්‍රතිපෝෂණයෙන් බිටු එකක් ගබඩා කරන මතක කොටුවකි” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. SR flip-flop
B. Flip-flop
C. Hold / මතකය
D. අර්ධ එකතුකරන්නා
E. Carry

19. “පරිපථයට තත්ත්වය තබා ගත හැකි වන සේ ප්‍රතිදානය ආදානයකට ආපසු සම්බන්ධ කිරීම” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. රෙජිස්තරය
B. සම්පූර්ණ එකතුකරන්නා
C. ප්‍රතිපෝෂණ වළල්ල
D. Set
E. Cin / Cout

20. “Set-Reset latch: S මගින් Q=1, R මගින් Q=0, 00 තබා ගනී” යන්නෙන් අදහස් වන පදය කුමක්ද?

A. SR flip-flop
B. අර්ධ එකතුකරන්නා
C. Reset
D. එකතු බිටුව
E. Flip-flop

ව්‍යුහගත ප්‍රශ්න

1. අර්ධ එකතුකරන්නාගේ සත්‍යතා වගුව අඳින්න සහ S සහ C බූලියානු ප්‍රකාශන ලෙස ලියන්න. *(ලකුණු 2)*

> වගුව AB→SC: 00→00, 01→10, 10→10, 11→01 (2); S=A⊕B, C=A·B (2).

2. LSB ට ඉහළ බිටු සඳහා සම්පූර්ණ එකතුකරන්නාක් අවශ්‍ය වන්නේ ඇයිදැයි පැහැදිලි කරන්න. *(ලකුණු 3)*

> එම තීරුවලට දකුණෙන් carry ලැබේ (1); අර්ධ එකතුකරන්නාට Cin නැත (1); සම්පූර්ණ එකතුකරන්නා Cin ගෙන Cout නිපදවයි (1).

3. A=1, B=1, Cin=0 සඳහා සම්පූර්ණ එකතුකරන්නාගේ S සහ Cout සොයන්න. ක්‍රියාවලිය පෙන්වන්න. *(ලකුණු 3)*

> S=1⊕1⊕0=0 (1); Cout=1 (1); පරීක්ෂාව $1+1+0=10_2$ (1).

4. Flip-flop එකක් බිටුවක් ගබඩා කරන ආකාරය සහ SR flip-flop එකක S=R=0 අදහස් කරන්නේ කුමක්දැයි විස්තර කරන්න. *(ලකුණු 3)*

> ප්‍රතිපෝෂණ වළල්ල තත්ත්වය තබා ගනී (1); බිටු එකක් ගබඩා කරයි (1); S=R=0 යනු පෙර Q හි hold / මතකය (1).

5. SR ආදාන සංයෝජන හතර සහ ඒවායේ බලපෑම් ප්‍රකාශ කරන්න. *(ලකුණු 4)*

> 10 set Q=1 (1); 01 reset Q=0 (1); 00 hold (1); 11 තහනම් (1).

සාරාංශය

- අර්ධ එකතුකරන්නා: S = A ⊕ B, C = A · B (Cin නැත).
- සම්පූර්ණ එකතුකරන්නා: S = A ⊕ B ⊕ Cin, Cout = බහුතරය.
- ALU තුළ බහු-බිටු සංඛ්‍යා එකතු කිරීමට සම්පූර්ණ එකතුකරන්නන් දාමගත කරන්න.
- Flip-flop එකක් ප්‍රතිපෝෂණයෙන් බිටු එකක් ගබඩා කරයි.
- SR: 10 set, 01 reset, 00 hold, 11 තහනම්.

MCQ පිළිතුරු – 4.4

1. **C** – බිටු දෙකේ XOR.
2. **A** – බිටු දෙකේ AND.
3. **E** – A, B සහ carry-in.
4. **B** – තුනේම XOR.
5. **C** – AB + BCin + ACin.
6. **D** – $1+1 = 10_2$.
7. **D** – මූලික මතක කොටුව = බිටු 1.
8. **B** – ප්‍රතිදානය ආපසු යැවීමෙන් Q තබයි.
9. **E** – Set ආදානය සක්‍රියයි.
10. **A** – මතක / hold තත්ත්වය.
11. **B** – එකවර set සහ reset.
12. **C** – බිටු තීරුවකට එකක්.
13. **E** – අර්ධ එකතුකරන්නා – බිටු දෙකක් එකතු කරයි: S = A ⊕ B, C = A · B. Cin නැත.
14. **A** – සම්පූර්ණ එකතුකරන්නා – A, B සහ Cin එකතු කරයි: S = A⊕B⊕Cin, Cout = බහුතරය.
15. **D** – එකතු බිටුව – එකතුකරන්නා තීරුවක ප්‍රතිඵල බිටුව (ආදානවල XOR).
16. **D** – Carry – එකතුව ≥ 2 වූ විට ඊළඟ තීරුවට යන අමතර බිටුව.
17. **E** – Cin / Cout – පෙර තීරුවෙන් carry ඇතුළු වීම; ඊළඟට carry පිටවීම.
18. **B** – Flip-flop – ප්‍රතිපෝෂණයෙන් බිටු එකක් ගබඩා කරන මතක කොටුවකි.
19. **C** – ප්‍රතිපෝෂණ වළල්ල – පරිපථයට තත්ත්වය තබා ගත හැකි වන සේ ප්‍රතිදානය ආදානයකට ආපසු සම්බන්ධ කිරීම.
20. **A** – SR flip-flop – Set-Reset latch: S මගින් Q=1, R මගින් Q=0, 00 තබා ගනී.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.