12 ශ්‍රේණිය A/L ICT · පාඩම 4

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

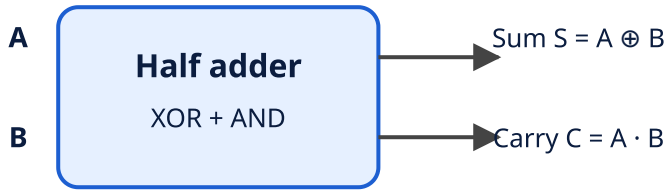
විෂය නිර්දේශය 4.4. CPU එක සංඛ්‍යා එකතු කරන්නේ **එකතුකරන්නා** පරිපථ මගිනි, මතකය බිටු ගබඩා කරන්නේ **flip-flop** මගිනි. මෙම කොටසින් **අර්ධ එකතුකරන්නා** සහ **සම්පූර්ණ එකතුකරන්නා** සඳහා සත්‍යතා වගු සහ බූලියානු ප්‍රකාශන සාදා, සම්පූර්ණ එකතුකරන්නන් ALU එකට දාමගත වන ආකාරය දකින අතර **flip-flop** එකක **ප්‍රතිපෝෂණ 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

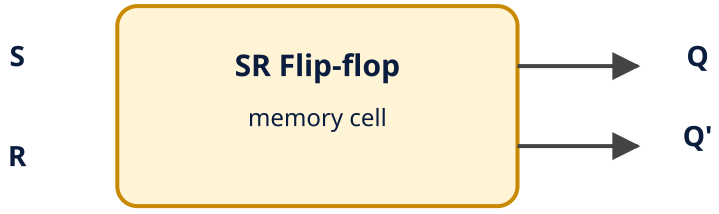
🧮 **විසඳූ උදාහරණය: A = 1, B = 0, Cin = 1 සඳහා S සහ Cout සොයන්න.**
S = 1 ⊕ 0 ⊕ 1 = **0**. Cout = බහුතරය = **1**.

🧮 **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.