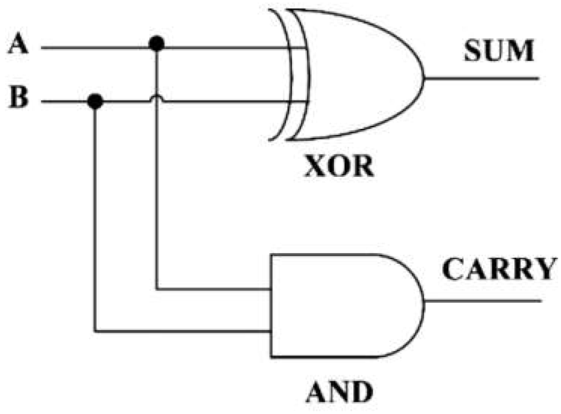
அரை கூட்டி (Half Adder)

- அரை கூட்டியின் உதவியுடன் சாதாரண கூட்டலைச் செய்யக் கூடிய திறன் கொண்ட சுற்றுக்களைத் தர்க்க வாயில்களின் உதவியுடன் வடிவமைக்கலாம்.
- ஒற்றைப் பிட்டுக்களின் (single bits) கூட்டல் பற்றி முதலில் பார்ப்போம்.பற்பயனர்
    - 0+0 = 0
    - 0+1 = 1
    - 1+0 = 1
    - 1+1 = 10

இவை குறைந்த சாத்தியமான ஒற்றை பிட் சேர்க்கைகள் ஆகும். எனினும் 1, 1 இன் விளைவு 10 ஆகும். அதனால் கூட்டுத்தொகை விளைவு மீண்டும் 2-பிட் வெளியீடுகளில் எழுத வேண்டும். ஆகவே மேலே உள்ள சமன்பாடுகள் பின்வருமாறு எழுதப்படல் வேண்டும்.

- 0+0 = 00
- 0+1 = 01
- 1+0 = 01
- 1+1 = 10

இங்கு வெளியீடு 10 இன் 1 ஆனது வெளியே எடுத்துச் (carry-out) செல்லப்படுகின்றது. கீழே உள்ள உண்மை அட்டவணையில் விளைவு காட்டப்பட்டுள்ளது. 'SUM' ஆனது சாதாரண வெளியீடு மற்றும் 'CARRY' ஆனது வெளியே எடுக்கப்படுவதையும் காட்டுகின்றது. (carry-out)



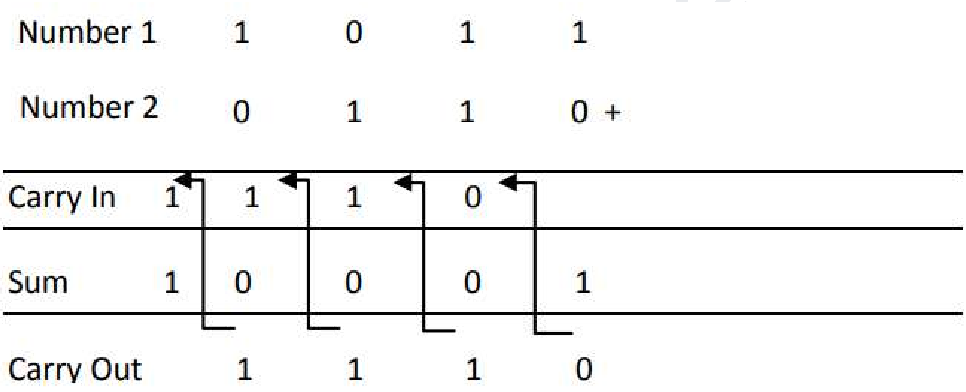
உள்ளீடுகள்		வெளியீடு	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

முழுமைக் கூட்டி (Full Adder)

முழுமைக் கூட்டிக்கும் அரைக் கூட்டிக்குமிடையிலான முக்கிய வேறுபாடு என்னவெனில் முழுமைக் கூட்டியில் மூன்று உள்ளீடுகளையும் இரண்டு வெளியீடுகளையும் கொண்டிருக்கின்றது. முதல் இரண்டு உள்ளீடுகள் A யும் B யும் மற்றும் மூன்றாவது உள்ளீடு ஏந்தி (carry) ஆனது உள்ளே செல்வதற்காக (Carry In) நியமிக்கப்பட்டுள்ளது. ஒரு முழுமைக் கூட்டி தர்க்கம் வடிவமைப்பினை மேற்கொள்வதற்கு எட்டு (string) களை ஒன்றுசேர்த்து ஒரு 8 பிட் கூட்டியையும், ஏந்தி பிட் ஒரு கூட்டியிலிருந்து மற்றைய கூட்டிக்கு செல்லும். (cascade the carry bit from one adder to the next)

ஏந்தி (carry) ஆனது ஏந்தி வெளியே கொண்டுசெல்லவும் (Carry Out) கூட்டுத்தொகையானது (Sum) கூட்டுத்தொகையினை நியமிக்கவும் வடிவமைக்கப்பட்டுள்ளது. இவற்றின் உண்மை அட்டவணை கீழே தரப்பட்டுள்ளது.

உதாரணம்: இலக்கங்களைக் கூட்டுவதில் அரைக் கூட்டியினதும் முழுமைக் கூட்டியினதும் பிரயோகங்கள் கீழே காட்டப்பட்டுள்ளது. மூன்று பிட்கள் கூட்டல்களும் இரண்டு பிட்கள் கூட்டல்களும் காட்டப்படுகின்றன.



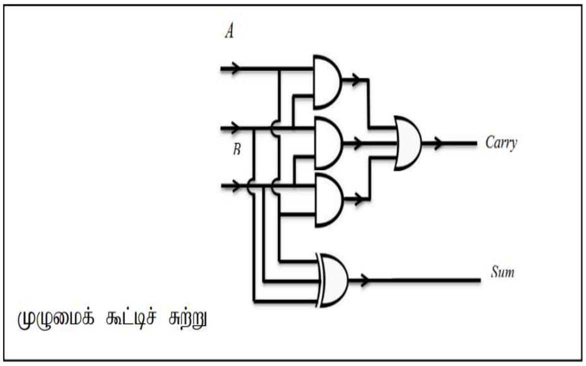
இங்கு முதல் இரண்டு பிட்கள் சேர்க்கப்படும் போது மாத்திரம் அரை கூட்டி தேவைப்படுகின்றது. ஏனெனில் உள்ளே ஏந்தி பிட் (carry in bit) கிடைக்காமையால் ஆகும். மீதமுள்ள பிட்களைக் கூட்டும் பொருட்டு, முழுமைக் கூட்டி சேர்க்கப்படுகின்றது. ஏனெனில் உள்ளே ஏந்தி பிட் (carry in bit) காணப்படுவதால் ஆகும். இரண்டு அரைக்கூட்டிகளின் (half adders) இணைப்பு முழுமைக் கூட்டியாகும் (Full adder).

உண்மை அட்டவணை மற்றும் தர்க்கச் சுற்று என்பவற்றிலிருந்து எவ்வாறு முழுமைக் கூட்டியையை (full adder) பிரதிநிதிப்படுத்துவது?

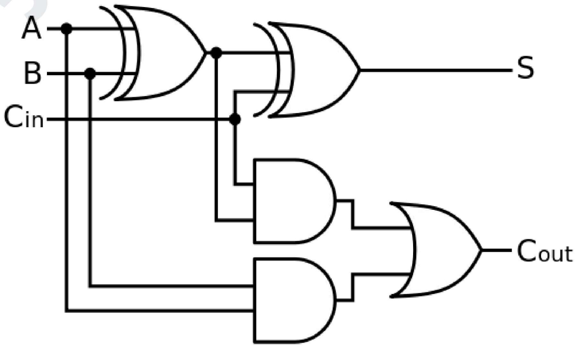
ஏந்தியானது (carry) A.B + B.C + A.C இனால் பெறப்படுகின்றது. கூட்டுத்தொகை(sum) A XOR B XOR C இனால் பெறப்படுகின்றது.

1	2	3	4	5	6	7	8	9	10	11
A	B	C	Sum	Carry	A.B	B.C	A.C	A.B+B.C+A.C	A XOR B	A XOR B XOR C
0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	0	1	1
0	1	1	0	1	0	1	0	1	1	0
1	0	0	1	0	0	0	0	0	1	1
1	0	1	0	1	0	0	1	1	1	0
1	1	0	0	1	1	0	0	1	0	0
1	1	1	1	1	1	1	1	1	0	1

மேலே உள்ள உண்மை அட்டவணை எல்லா வரிசைக்குமான உள்ளீட்டுப் பெறுமதிகளுக்குமான வெளியீட்டுப் பெறுமதிகளைக் காட்டுகின்றது. நிரல்களின் இலக்கம் 4 மற்றும் 11 என்பவை ஒத்தவையாக காணப்படுகிறது. கூட்டுத்தொகையானது **A XOR B XOR C** என்னும் பூலியன் கோவையினால் எடுத்துக்காட்டப்படுகின்றது. நிரல்களின் இலக்கம் 5, 9 என்பவை ஒத்தவையாகக் காணப்படுகின்றன. ஆகவே ஏந்தியானது (carry) ஆனது **A.B+B.C+A.C** என்னும் தர்க்கத்தினால் மேற்கொள்ளப்படுகின்றது.



முழுமைக் கூட்டிச் சுற்று



தொடர்ச்சியான சுற்று: வெளியீடானது தற்போதுள்ள உள்ளீடுகளில் மாத்திரம் தங்கியிருக்காது முன்னைய உள்ளீடுகள் மற்றும் வெளியீடுகளிலும் தங்கியிருக்கும். இவ்வகையான சுற்றுக்கள் மேலும் உள்ளீடுகள் எடுக்காமல் செயல்களைத் தொடர்ச்சியாகச் செய்வதற்குத் தேவைப்படும். நினைவக சேமிப்புக்காக (SRAM) இது பயன்படுகின்றது.

எழு விழுவிகள் (Flip Flops)

தர்க்கச் சுற்றுக்களின் பிரயோகமாக எழு விழுவிகள் காணப்படுகின்றன. பூலியன் தர்க்கத்தின் உதவியுடன் இவற்றினைப் பயன்படுத்தி நினைவகத்தை உருவாக்க முடியும். எழு விழுவிகள் மிகவும் அடிப்படை யோசனையான (idea) எழுமாற்று அணுகல் நினைவகமாகவும் (Random-Access Memory) கருதப்பட முடியும். குறித்த உள்ளீட்டுப் பெறுமதிகள் கொடுக்கப்படும் போது அவை அவற்றை நினைவில் வைத்திருக்கும்.

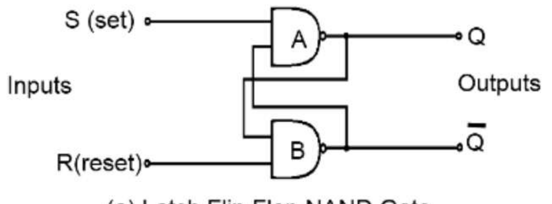
எழு விழுவிகளின் ஒரு வடிவம் துல்லியமாகக் கணினியின் மின் ஆளியைப் போல் செயற்படுகின்றது: முதன் முறை அதன் உள்ளீடு 1 ஆகிறது, எழு விழுவிகள் வெளியீட்டை நிறுத்துகின்றது. அதாவது எழுவிழுவிகள் உள்ளீடு 1 கிடைக்கப் பெறுவதன் காரணமாக வெளியீட்டைத் தற்போதைய நிலையிலிருந்து அதற்கு எதிராக மாற்றுகின்றது. தள்ளு பொத்தானைப் போல ஆளியானது மின்னைக் கட்டுப்படுத்தப் பயன்படுகின்றது, எழுவிழுவிகள் தொடர்ச்சியான உள்ளீட்டுக்குப் பொறுப்பல்ல - உள்ளீடானது பெறுமதி 1 இற்கு முன்னால் 0 இற்கு கட்டாயமாகத் திரும்புதல் ஆனது எழு விழுவிகள் நிலையை மாற்றுவதற்குக் காரணமாக அமையும். கீழே உள்ள படமானது தொடர்ச்சியான உள்ளீடுகள் மற்றும் விளைவான வெளியீடு என்பவற்றைக் காட்டுகின்றது.

One form of flip-flop acts exactly like the power switch on a computer: the first time its input becomes 1, the flip flop turns on the output, and the second time its input becomes 1, the flip-flop turns off the output. That is, receiving an input of 1 causes the flip-flop to change the output from the current state to the opposite. Like a push-button switch used to control power, a flip-flop does not respond to a continuous input — the input must return to 0 before a value of 1 will cause the flip-flop to change state. Below figure shows a sequence of inputs and the resulting output.

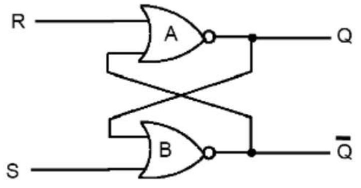
மடிப்பு எழு விழுவிகள் (Latch Flip-flop)

R-S (Reset Set) எழு விழுவியானது எல்லாவற்றிலும் எளிமையானதும் விளங்குவதற்கு இலகுவானதுமாகும். இது அடிப்படையில் ஒரு உபகரணமாகும். இது இரண்டு வெளியீடுகளையும் ஒரு வெளியீடு மற்றையதின் நேர்மாறாக அல்லது நிரப்பியாகவும் இரண்டு உள்ளீடுகளையும் கொண்டது. உள்ளீடுகளில் ஒன்றின் மீதான துடிப்பு குறிப்பிட்ட தர்க்க நிலையை எடுத்துக் கொள்ளும். மற்றைய உள்ளீட்டுக்கும் இதே மாதிரியான துடிப்பைப் பிரயோகிக்கும் வரை பின்னர் வெளியீடுகள் இதே நிலையில் இருக்கும். இந்த இரண்டு உள்ளீடுகளும் Set மற்றும் Reset என அழைக்கப்படும். (சில நேரங்களில் preset மற்றும் clear inputs என அழைக்கப்படும்)

இவ்வகையான எழுவிழுவிகள் வெறுமனே இரண்டு குறுக்கு நேர்மாற்று வாயில்கள் NAND அல்லது NOR வாயில் பயன்படுத்தி உருவாக்கப்பட முடியும். உருவம் 1(a) ஆனது NAND வாயிலைப் பயன்படுத்தி RS எழு விழுவியையும் மற்றும் உருவம் (b) NOR வாயிலைப் பயன்படுத்தி RS எழு விழுவியையும் காட்டுகின்றது.



(a) Latch Flip Flop NAND Gate



(b) RS Latch Flip Flop NOR Gate

paulrajeevan.com