2023

9. Binair

R2: SCRAPY-gids

Projectnummer: **2021-1-FR01-KA220-SCH-000031617**

Co-funded by the European Union

ECAM EPMI

30/04/2023

Inhoudsopgave

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

1 Inleiding

Getallenstelsels zijn de methoden die we gebruiken om getallen weer te geven. Sinds de lagere school werken we allemaal binnen de comfortabele grenzen van een basissysteem van 10, maar er zijn er nog veel meer. Basis-2, basis-8, basis-16, basis-20, basis... je snapt het wel. Er is een oneindige verscheidenheid aan getallenstelsels, maar er zijn er maar een paar die van bijzonder belang zijn voor elektrotechniek.

De echt populaire getallenstelsels hebben zelfs een naam. Basis-10 wordt bijvoorbeeld het decimale getallenstelsel genoemd. Basis-2, waar we het vandaag over hebben, wordt ook binair genoemd. Een ander populair getallensysteem, basis-16, wordt hexadecimaal genoemd.

De basis van een getal wordt vaak weergegeven door een subscripted integer achter een waarde. Dus, in de inleiding hierboven zou de eerste afbeelding 10010 iets zijn, terwijl de tweede afbeelding 1002 iets zou zijn. Dit is een handige manier om de basis van een getal te specificeren als er kans is op dubbelzinnigheid.

2. Waarom binair?

Waarom binair vraag je je af? Waarom decimaal? We gebruiken al decimalen sinds mensenheugenis en hebben meestal voor lief genomen waarom we het basis-10 getallensysteem hebben gekozen voor onze dagelijkse getallenbehoeften. Het is omdat we 10 vingers hebben, of het is gewoon omdat de Romeinen het opdrongen aan hun oude onderdanen. Wat er ook toe geleid heeft, de trucjes die we gaandeweg geleerd hebben, hebben de plaats van basis-10 in ons hart verstevigd; iedereen kan met 10-tallen tellen. We ronden zelfs grote getallen af naar het dichtstbijzijnde veelvoud van 10. We zijn geobsedeerd door 10!

Computers en elektronica zijn beperkt in de vinger-en-teen-afdeling. Op het laagste niveau hebben ze slechts twee manieren om de toestand van iets weer te geven: AAN of UIT, hoog of laag, 1 of 0. En dus vertrouwt alle elektronica op een basis-2 getallensysteem om getallen op te slaan, te manipuleren en te rekenen.

Omdat elektronica zo afhankelijk is van binaire getallen, is het belangrijk om te weten hoe het basen-2 getallenstelsel werkt. Je zult het binaire stelsel, of zijn neefjes, zoals hexadecimaal, overal in computerprogramma's tegenkomen. Bij de analyse van digitale logische schakelingen en andere elektronica op zeer laag niveau wordt ook veel gebruik gemaakt van binaire getallen.

In deze les zul je ontdekken dat alles wat je met een decimaal getal kunt doen, ook met een binair getal kan worden gedaan. Sommige bewerkingen zijn zelfs gemakkelijker uit te voeren op een binair getal (hoewel andere pijnlijker kunnen zijn). Dat en nog veel meer komt in deze les aan bod.

3 Tellen en omrekenen

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

De basis van elk getallenstelsel wordt ook wel de radix genoemd. De radix van een decimaal getal is tien en de radix van binair is twee. De radix bepaalt hoeveel verschillende symbolen er nodig zijn om een getallenstelsel vorm te geven. In ons decimale getallenstelsel hebben we 10 cijferrepresentaties voor waarden tussen niets en tien dingen: 0, 1, 2, 3, 4, 5, 6, 7, 8 en 9. Elk van deze symbolen vertegenwoordigt een zeer specifieke, gestandaardiseerde waarde.

In het binaire stelsel mogen we maar twee symbolen gebruiken: 0 en 1. Maar met die twee symbolen kunnen we elk getal maken dat in een decimaal systeem mogelijk is.

3.1 Binair tellen

Je kunt eindeloos in decimalen tellen, zelfs in je slaap, maar hoe zou je in binair tellen? Nul en één in basis-twee zien er vrij bekend uit: 0 en 1. Vanaf daar worden de dingen uitgesproken binair.

Onthoud dat we alleen deze twee cijfers hebben, dus zoals we doen in decimale getallen als de symbolen op zijn, moeten we een kolom naar links verschuiven, een 1 toevoegen en alle cijfers naar rechts op 0 zetten. Dus na 1 krijgen we 10, dan 11, dan 100. Laten we beginnen met tellen...

Decimaal	Binair	... Decimaal	Binair
0	0	16	10000
1	1	17	10001
2	10	18	10010
3	11	19	10011
4	100	20	10100
5	101	21	10101
6	110	22	10110
7	111	23	10111
8	1000	24	11000
9	1001	25	11001
10	1010	26	11010
11	1011	27	11011
12	1100	28	11100
13	1101	29	11101
14	1110	30	11110
15	1111	31	11111

Begint dat het plaatje te schetsen? Laten we eens kijken hoe we deze binaire getallen kunnen omrekenen naar decimalen.

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

3.2 Binair naar decimaal omrekenen

Er is niet één manier om binair naar decimaal om te zetten. We zullen hieronder twee methoden schetsen, de meer "wiskundige" methode en een andere die meer visueel is. We zullen beide behandelen, maar als de eerste te veel lelijke terminologie gebruikt, ga dan naar de tweede.

Methode 1

Er is een handige functie die we kunnen gebruiken om elk binair getal om te zetten naar decimaal:

$$a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_1 2^1 + a_0 2^0$$

Er zijn vier belangrijke elementen in die vergelijking:

an, an-1, a1, enzovoort, zijn de cijfers van een getal. Dit zijn de 0's en 1's die je kent, maar in binair kunnen ze alleen 0 of 1 zijn.

De positie van een cijfer is ook belangrijk om in de gaten te houden. De positie begint bij 0, op het meest rechtse cijfer; deze 1 of 0 is het minst significant. Elk cijfer dat je naar links verschuift, neemt toe in betekenis en verhoogt ook de positie met 1.

De lengte van een binair getal wordt gegeven door de waarde van n, eigenlijk is het n+1. Bijvoorbeeld, een binair getal als 101 heeft een lengte van 3, en iets groters als 10011110 heeft een lengte van 8.

Elk cijfer wordt vermenigvuldigd met een gewicht: de 2n, 2n-1, 21, enz. Het meest rechtse gewicht - 20 is gelijk aan 1, verplaats een cijfer naar links en het gewicht wordt 2, dan 4, 8, 16, 32, 64, 128, 256,... en ga maar door. Machten van twee zijn van groot belang voor binair, ze worden snel erg bekend.

Laten we die n's en exponenten weglaten en onze binaire positienotatievergelijking op acht posities uitvoeren:

$$a_7 \cdot 128 + a_6 \cdot 64 + a_5 \cdot 32 + a_4 \cdot 16 + a_3 \cdot 8 + a_2 \cdot 4 + a_1 \cdot 2 + a_0 \cdot 1$$

Laten we dat verder uitwerken door wat waarden voor de cijfers in te voeren. Wat als je een binair getal als 10011011 had? Dat zou betekenen dat (een) waarde van:

1	0	0	1	1	0	1	1
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

Methode 2

Een andere, meer visuele manier om binaire getallen om te zetten naar decimalen is door te beginnen met het sorteren van elke 1 en 0 in een bin. Elke bak heeft een opeenvolgende macht van twee gewichten, de 1, 2, 4, 8, 16,... die we gewend zijn. Uitvoeren tot acht plaatsen zou er ongeveer zo uitzien:

1286432168421

Dus als we ons binaire getal 10011011 in die bakken sorteren, ziet het er zo uit:

1286432168421

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

10011011

Voor elke bin met een binaire 0-waarde kun je deze doorhalen en verwijderen.

1286432168421
10011011

En tel dan alle overgebleven gewichten op om je getal te krijgen!

3.3 Omzetten van decimaal naar binair

Net zoals er meer dan één manier is om van binair naar decimaal te gaan, is er ook meer dan één manier om decimaal naar binair te converteren. De eerste gebruikt deling en resten, en de tweede gebruikt aftrekking. Probeer beide en blijf bij de manier waar jij je goed bij voelt!

Methode 1

Het is niet zo eenvoudig om een decimaal getal om te zetten naar binair. Deze conversie vereist het herhaaldelijk delen van het decimale getal door 2, totdat je het tot nul hebt gereduceerd. Elke keer dat je deelt, wordt de rest van de deling een cijfer in het binaire getal dat je maakt.

Weet je niet meer hoe je restsommen moet maken? Als het al een tijdje geleden is, onthoud dan dat, aangezien we delen door twee, als het dividend even is, de rest 0 is; een oneven dividend betekent een rest van 1.

Om bijvoorbeeld 155 om te zetten naar binair, doorloop je dit proces:

155 ÷ 2 = 77 R 1 (Dat is het meest rechtse cijfer, 1e positie)
77 ÷ 2 = 38 R 1 (2e positie)
38 ÷ 2 = 19 R 0 (3e positie)
19 ÷ 2 = 9 R 1
9 ÷ 2 = 4 R 1
4 ÷ 2 = 2 R 0
2 ÷ 2 = 1 R 0
1 ÷ 2 = 0 R 1 (8e positie)

De eerste rest is het minst significante (meest rechtse) cijfer, dus lees van boven naar beneden om ons binaire getal van rechts naar links uit te werken: 10011011. Vergelijk het met het voorbeeld hierboven... dat is een bingo!

Methode 2

Als delen en het vinden van resten niet jouw ding is, is er misschien een eenvoudigere methode om decimale getallen om te zetten naar binaire getallen. Begin met het vinden van de grootste macht van twee die nog steeds kleiner is dan je decimale getal en trek deze af van het decimale getal. Ga dan verder met het aftrekken van de grootst mogelijke macht van twee tot je bij nul komt. Elke gewichtspositie die werd afgetrokken, krijgt een binair 1-cijfer; cijfers die niet werden afgetrokken, krijgen een 0.

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

Doorgaan met ons voorbeeld: 155 kan worden afgetrokken van 128, wat 27 oplevert:

155 - 128 = 27

1286432168421
1

Ons nieuwe getal, 27, kan niet worden afgetrokken van 64 of 32. Beide posities krijgen een 0. We kunnen wel aftrekken van 16, wat 11 oplevert. Beide posities krijgen een 0. We kunnen aftrekken door 16, wat 11 oplevert.

27 - 16 = 11

1286432168421
1001

En 8 trekt af van 11, wat 3 oplevert. Daarna geen geluk met 4.

11 - 8 = 3

1286432168421
100110

Onze 3 kan worden afgetrokken door 2, wat 1 oplevert. En tenslotte wordt de 1 afgetrokken door 1 om 0 te maken.

3 - 2 = 1
1 - 1 = 0

1286432168421
10011011

We hebben een binair getal!

Bits, nibbles en bytes

Bij het bespreken van het merk van een binair getal, hebben we kort de lengte van het getal besproken. De lengte van een binair getal is het aantal 1'en en 0'en dat het getal heeft.

4 Gemeenschappelijke binaire getallenlengtes

Binaire waarden worden vaak gegroepeerd in een gemeenschappelijke lengte van 1's en 0's, dit aantal cijfers wordt de **lengte** van een getal genoemd. Veel voorkomende bitlengtes van binaire getallen zijn bits, nibbles en bytes (nog honger?). Elke 1 of 0 in een binair getal wordt een **bit genoemd**. Vervolgens wordt een groep van 4 bits een **nibble genoemd** en 8 bits vormen een **byte**.

Bytes zijn een veelgebruikt modewoord bij het werken in binary. Processors zijn allemaal gebouwd om te werken met een vaste lengte bits, die meestal een veelvoud is van een byte: 8, 16, 32, 64, enzovoort.

Samengevat:

Lengte	Naam	voorbeeld

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

1Bit0
4Nibble1011
8Byte10110101

Word is een ander modewoord dat af en toe de ronde doet. Word klinkt veel minder lekker en is veel dubbelzinniger. De lengte van een woord is meestal afhankelijk van de architectuur van een processor. Het kan 16 bits, 32, 64 of zelfs meer zijn.

5. Opvullen met voorloopnullen

Je ziet soms binaire waarden weergegeven in bytes (of meer), zelfs als je om een getal 8-bits lang te maken voorloopnullen moet toevoegen. Voorloopnullen zijn één of meer 0's die links van het meest significante 1-bit in een getal worden toegevoegd. Je ziet meestal geen voorloopnullen in een decimaal getal: 007 zegt niets meer over de waarde van het getal 7 (het zegt misschien iets anders).

Voorloopnullen zijn niet verplicht bij binaire waarden, maar ze helpen wel om informatie te geven over de bit-lengte van een getal. Je ziet bijvoorbeeld het getal 1 afgedrukt als 00000001, om aan te geven dat we binnen het bereik van een byte werken. Beide getallen vertegenwoordigen dezelfde waarde, maar het getal met zeven 0's ervoor voegt informatie toe over de bitlengte van een waarde.

6 Bitwise Operatoren

Er zijn verschillende manieren om binaire waarden te manipuleren. Net zoals je dat kunt met decimale getallen, kun je standaard wiskundige bewerkingen - optellen, aftrekken, vermenigvuldigen en delen - uitvoeren op binaire waarden (die we op de volgende pagina zullen behandelen). Je kunt ook individuele bits van een binaire waarde manipuleren met behulp van bitwise operatoren.

Bitwise operatoren voeren bit-voor-bit functies uit op één of twee volledige binaire getallen. Ze maken gebruik van booleaanse logica die werkt op een groep binaire symbolen. Deze bitwise operatoren worden veel gebruikt in zowel elektronica als programmeren.

7 Aanvulling (NIET)

Het complement van een binaire waarde is zoiets als het vinden van het exacte tegenovergestelde van alles aan die waarde. De complement-functie kijkt naar een getal en verandert elke 1 in een 0 en elke 0 in een 1. De complement-operator wordt ook wel NOT genoemd.

Bijvoorbeeld, om het complement van 10110101 te vinden:

NIET 10110101 (decimaal 181)
-------- =
01001010 (decimaal 74)

NOT is de enige bitwise operator die alleen werkt op een enkele binaire waarde.

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

8 OR

OR neemt twee getallen en produceert de vereniging ervan. Dit is het proces om twee binaire getallen samen te OR-en: zet elk getal op een rij zodat de bits overeenkomen en vergelijk vervolgens elk van hun bits die een positie delen. Voor elke bitvergelijking geldt dat als een of beide bits 1 zijn, de waarde van het resultaat op die bitpositie 1 is. Als beide waarden een 0 hebben op die positie, krijgt het resultaat ook een 0 op die positie.

De vier mogelijke OF-combinaties en hun uitkomst zijn:

- 0 OF 0 = 0
- 0 OF 1 = 1
- 1 OF 0 = 1
- 1 OF 1 = 1

Om bijvoorbeeld de 10011010 OF 01000110 te vinden, zet je alle getallen bit voor bit op een rij. Als één of beide getallen een 1 hebben in een kolom, dan heeft de resultaatwaarde daar ook een 1:

10011010
OR 01000110
-------- =
11011110

Zie de OR-bewerking als binair optellen, zonder carry-over. 0 plus 0 is 0, maar 1 plus iets is 1.

9 EN

AND neemt twee getallen en produceert de conjunctie ervan. AND produceert alleen een 1 als beide waarden waarop het werkt ook 1 zijn.

Het proces om twee binaire waarden samen te AND'en is vergelijkbaar met dat van OR. Zet elk getal op een rij zodat de bits overeenkomen en vergelijk vervolgens alle bits die een positie delen. Voor elke bitvergelijking geldt dat als een of beide bits 0 zijn, de waarde van het resultaat op die bitpositie 0 is. Als beide waarden een 1 hebben op die positie, krijgt het resultaat ook een 1 op die positie.

De vier mogelijke EN combinaties en hun uitkomst zijn:

- 0 EN 0 = 0
- 0 EN 1 = 0
- 1 EN 0 = 0
- 1 EN 1 = 1

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

Om bijvoorbeeld de waarde van 10011010 EN 01000110 te vinden, begin je met elke waarde op een rij te zetten. Het resultaat van elke bit-positie zal alleen 1 zijn als beide bits in die kolom ook 1 zijn.

10011010
EN 01000110
-------- =
00000010

Zie AND als vermenigvuldigen. Wanneer je vermenigvuldigt met 0 zal het resultaat ook 0 zijn.

10 XOR

XOR is de exclusieve OF. XOR gedraagt zich als een gewone OR, behalve dat het alleen een 1 produceert als een van beide getallen een 1 heeft in die bit-positie.

De vier mogelijke XOR-combinaties en hun uitkomst zijn:

- 0 XOR 0 = 0
- 0 XOR 1 = 1
- 1 XOR 0 = 1
- 1 XOR 1 = 0

Bijvoorbeeld, om het resultaat te vinden van 10011010 XOR 01000110:

10011010
XOR 01000110
-------- =
11011100

Let op het 2e bit, een 0 die het resultaat is van twee XOR'en van 1.

11 Bitverschuivingen

Bitverschuivingen zijn niet noodzakelijkerwijs een bitwise operator zoals de hierboven genoemde, maar ze zijn een handig hulpmiddel om een enkele binaire waarde te manipuleren.

Een bitverschuiving bestaat uit twee componenten: de richting en het aantal bits dat verschoven moet worden. Je kunt een getal naar links of rechts verschuiven en je kunt één bit of veel bits verschuiven.

Bij het verschuiven naar rechts worden een of meer van de minst significante bits (aan de rechterkant van het getal) gewoon afgekapt en verschoven naar het oneindige niets. Voorloopnullen kunnen worden toegevoegd om de bitlengte hetzelfde te houden.

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.

Bijvoorbeeld 10011010 twee bits naar rechts verschuiven:

RIGHT-SHIFT-2 10011010 (decimaal 154)
-------- =
00100110 (decimaal 38)

Door naar links te verschuiven, worden alle bits naar de meest significante kant (de linkerkant) van het getal geduwd. Voor elke verschuiving wordt een nul toegevoegd op de positie van het minst significante bit.

Bijvoorbeeld 10011010 één bit naar links verschuiven:

LEFT-SHIFT-1 10011010 (decimaal 154)
-------- =
100110100 (decimaal 308)

Die eenvoudige bitverschuiving voert een ingewikkelde wiskundige functie uit. Een verschuiving naar links van n bits vermenigvuldigt een getal met 2n (zie je hoe het laatste voorbeeld de invoer met twee vermenigvuldigde?), terwijl een verschuiving in bits naar rechts een geheel getal deelt door 2n. Naar rechts verschuiven om te delen kan vreemd worden - alle fracties die worden geproduceerd door de verschoven deling zullen worden afgekapt, daarom is 154 tweemaal naar rechts verschoven gelijk aan 38 in plaats van 154/4=38.5. Bitverschuivingen kunnen een krachtige snelle manier zijn om te delen of te vermenigvuldigen met 2, 4, 8, enz.

12 Conclusie

Binair is de bouwsteen van alle berekeningen en bewerkingen in de elektronica. Er zijn dus veel plaatsen om naartoe te gaan.

Nu je kunt converteren tussen decimaal en binair, kun je die kennis toepassen om te begrijpen hoe tekens universeel worden gecodeerd: ASCII

Of je kunt je glimmende nieuwe kennis toepassen op low-level schakelingen en IC's:

- Digitale logica
- Shift-registers

Je kunt ook bekijken hoe binair een belangrijke rol speelt in deze communicatieprotocollen:

- Seriële communicatie
- Seriële perifere interface
- I2C

Dit project is gefinancierd met steun van de Europese Commissie. De verantwoordelijkheid voor deze publicatie [mededeling] ligt uitsluitend bij de auteur; de Commissie kan niet aansprakelijk worden gesteld voor het gebruik van de informatie die erin is vervat.