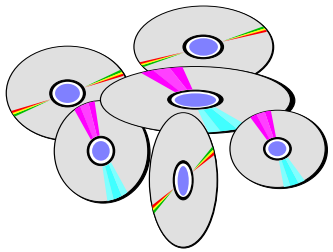
Analoog rekenen

- Gebruik makend van fysische grootheden
  - Cf Babbage
- **Analoge electronica**
  http://www.chem.uoa.gr/applets/appletopamps/appl_opamps2.html
- De rekenlat of 'slide rule' (http://www.syssrc.com/html/museum/html/sims/javaslide/), was lange tijd hèt rekeninstrument van de ingenieur

Muziek



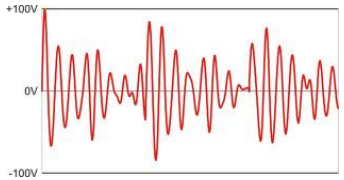
Analoog

Digitaal

informatie gecodeerd door middel van een continu-veranderlijke fysische grootheid

Muziek

Mens hoort tussen 20 en 20,000 Hz



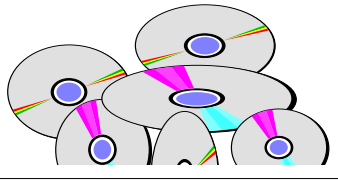

```
-096
+057
+164
+210
+219
+216
+165
-003
-117
-183
-138
-067
```

informatie gecodeerd door middel van een continu-veranderlijke fysische grootheid

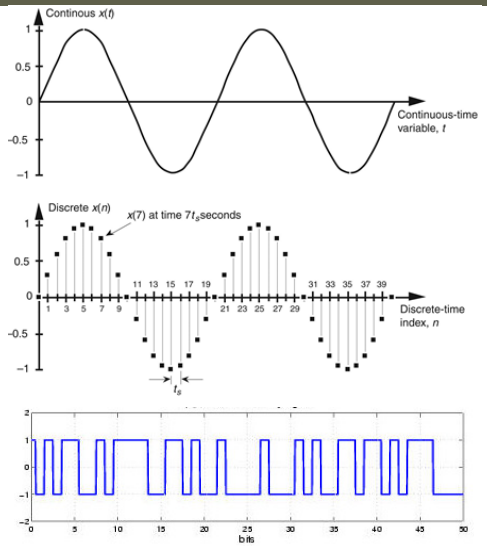
Analoog

Digitaal (CD)

(44100 metingen/s)



Digitaliseren



Analoog signaal

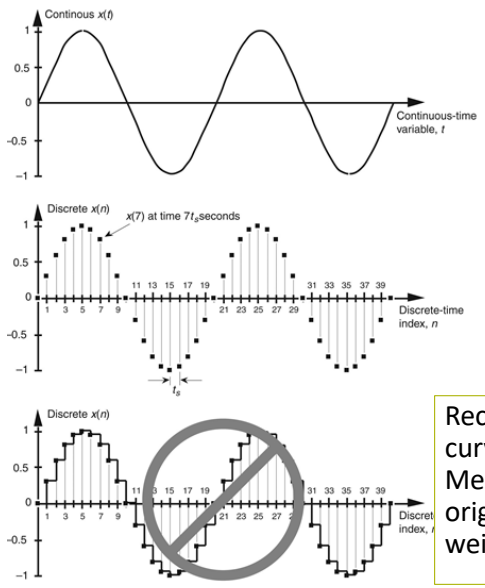
(1) Discretizeren: *samplen* in de tijd

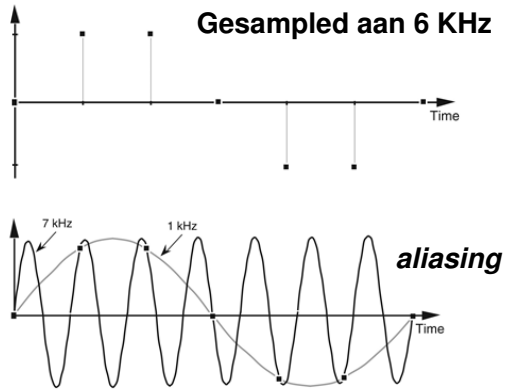
(2) Quantizatie: de amplitude in elk samplepunt voorstellen als een binair getal

Digitaal signaal

Digitaliseren van een analoog signaal (bvb muziek) gebeurt door op geregelde tijdstippen (discretizeren) de amplitude van het signaal weg te schrijven als een aantal bits (quantizeren).

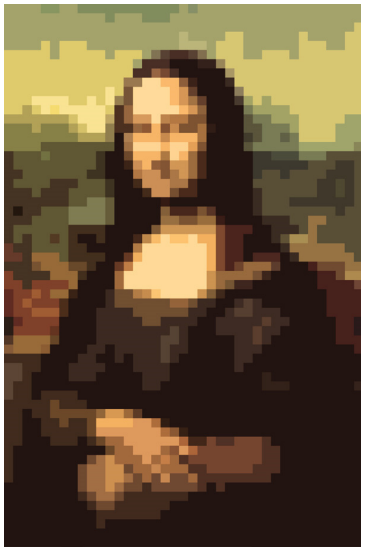
Reconstructie van analoog signaal





Reconstructie gebeurt door een 'vloeiende' curve door de digitale punten te laten lopen. Men moet wel genoeg punten hebben om het origineel te kunnen reconstrueren. Bij te weinig punten krijgt men aliasing.

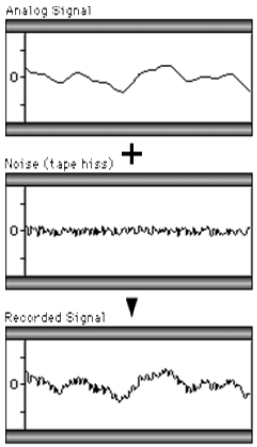
Digitalisatie van beelden

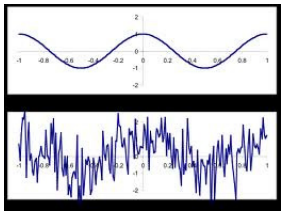


voor elk punt (pixel) de kleur opslaan
=> **bitmap**

Effect van signaal-vervormingen of ruis (noise)

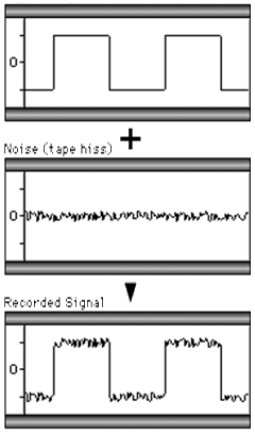
Analoog signaal





Informatieverlies

Digitaal signaal



Informatie nog te herkennen

Waarom digitaal?

- Unambiguë signalen, immuun voor ruis.
  - Als ruis onder een aanvaardbaar niveau is kan men steeds nog het origineel signaal herkennen. Het is immers een 0 of een 1.
- Perfecte copieën kunnen gemaakt worden.
- Bewerkingen zonder informatieverlies
  - *Digitale signaalverwerking is mogelijk!*
- Simpel, gemakkelijk te maken.
- ➡ Digitale componenten zijn goedkoop, klein, betrouwbaar en men kan er miljoenen op een klein gebied plaatsen.

Alles dat voorgesteld kan worden door één of ander signaal/patroon, kan voorgesteld worden door bits.

Go digital! Go binary!

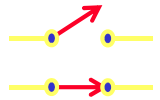
- Van analoog naar digital en binair…
- Leibniz had er al aan gedacht (zoals te zien is in zijn nota's)
- Ook was dit de basis van de computer van Atanasoff en Berry

Waarom binair?

- *De ultieme essentie van informatie: 0 of 1 (to be or not to be)*
- In digitale toestellen gebruikt men meestal het binaire stelsel omdat binaire cijfers eenvoudig kunnen voorgesteld worden door fysische grootheden zoals een positieve of een negatieve spanning, een magnetisch veld in de ene of de andere richting of een open of gesloten schakelaar.
- Eenvoudige reconstructie: groter of kleiner dan een drempelwaarde (*threshold*) bepaalt de bitwaarde

Binaire representatie

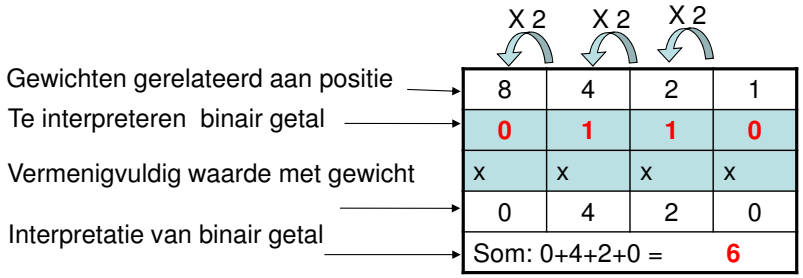
- Een binair getal (bit) kan voorgesteld worden door 2 voltages die gegeven kunnen worden door een switch:
  - Waarde 0 = 0 Volt = switch open
  - Waarde 1 = 5 Volt = switch gesloten



- Een getal van n bits kan 2^n waarden aannemen
  - 2 bits : 4 combinaties 00 01 10 11
  - 3 bits : 8 combinaties 000 001 010 011 100 101 110 111
  - 8 bits (= 1 byte) 256 combinaties
  - 16 bits: 65 536 combinaties
  - 32 bits: 4 294 967 296 combinaties

$2^{10} \approx 1000$

Binair talstelsel



128	64	32	16	8	4	2	1	Decimale Interpretatie
1	0	0	0	0	0	1	0	130
0	1	0	1	0	1	0	1	85
0	0	1	0	1	0	0	1	41
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	$\sum_{i=0}^{n} a_i \cdot 2^i$

Enkele oefeningen

128	64	32	16	8	4	2	1	Decimale Interpretatie
1	0	0	0	0	0	1	0	
0	1	0	1	0	1	0	1	
0	0	1	0	1	0	0	1	

Opgave 1: Bereken de decimale interpretatie van de bovenstaande binaire getallen.
Opgave 2: Zet om naar binair: 6, 17, 31, 255
Opgave 3: Zet uw geboortedatum om in binair formaat (dag, maand en jaar apart).
Opgave 4: Hoeveel decimale getallen kunnen er voorgesteld worden aan de hand van een 10-bit binair woord?

Grootte-ordes

		Bytes	
10^3	Kilo	kB	$\approx 2^{10} = 1024$
10^6	Mega	MB	$\approx 2^{20}$
10^9	Giga	GB	$\approx 2^{30}$
10^{12}	Tera	TB	$\approx 2^{40}$
10^{15}	Peta	PB	$\approx 2^{50}$
10^{18}	Exa	EB	$\approx 2^{60}$

Vergeet nooit meer: $2^{10} \approx 1000$

basisoperaties op bits en bytes

And

a \ b	0	1
0	0	0
1	0	1

Or

a \ b	0	1
0	0	1
1	1	1

Exclusive or

a \ b	0	1
0	0	1
1	1	0

a=60		0	0	1	1	1	1	0	0
b=13		0	0	0	0	1	1	0	1
~a	Not	1	1	0	0	0	0	1	1
a & b	And	0	0	0	0	1	1	0	0
a | b	Or	0	0	1	1	1	1	0	1
a ^ b	Exclusive or	0	0	1	1	0	0	0	1
a << 3	Shift left	1	1	1	0	0	0	0	0
a >> 3	Shift right	0	0	0	0	0	1	1	1

Zie programma **`BitwiseOperators`** in package **`voorbeelden`**

Programma van vorige

```
public class BitwiseOperators {
   public static void main(String[] args) {
      int a = 60;/* 60 = 0011 1100 */
      int b = 13;/* 13 = 0000 1101 */
      int c = 0;

      c = a & b;        /* 12 = 0000 1100  and */
      System.out.println("a & b = " + c );

      c = a | b;        /* 61 = 0011 1101  or */
      System.out.println("a | b = " + c );

      c = a ^ b;        /* 49 = 0011 0001  exclusive or (xor) */
      System.out.println("a ^ b = " + c );

      c = ~a;           /*-61 = 1100 0011  not */
      System.out.println("~a = " + c );

      c = a << 2;       /* 240 = 1111 0000  shift left */
      System.out.println("a << 2 = " + c );

      c = a >> 2;       /* 15 = 1111        shift right */
      System.out.println("a >> 2  = " + c );
   }
}
```

Enkele toepassingen

Zie programma **BitwiseOperators** in package **voorbeelden**

Gegeven: integers x & i

```
int x = 85, i = 4;
```

1. "Is de i-de bit van x gelijk aan 1?"

```
int y = 1 << i; // y wordt een masker genoemd
if ((x & y) > 0)
   System.out.println("De "+i+ "de bit van" +x+" is 1.");
else
    System.out.println("De "+i+ "de bit van" + x +" is 0.");
```

2. "Zet de (i+1)-de bit van x op 1"

```
y <<= 1; // is identiek aan y = y << 1;
x |= y; // is identiek aan x = x | y;
System.out.println("x = "+x+": "+(i+1)+ "de bit is nu 1.");
```

3. "Wat zijn de waarden van de 4e tot en met de 6e bit?"

```
int z = (x >> 4) & 7; // 7 is hier ook een masker: 111
System.out.println("Bits 4 tot en met 6 van "
+Integer.toBinaryString(x)+" zijn "+Integer.toBinaryString(z));
```

Hexadecimaal talstelsel

Om de notatie van binaire getallen te verkorten worden ook de ***hexadecimale notatie*** gebruikt
In een hexadecimaal cijfer worden 4 bits samengenomen.

Vb: omzetting van 1110001101

1	1	1	0	0	0	1	1	0	1
3			8			D			

Om aan te duiden dat een getal in het hexadecimaal talstelsel is (bvb tijdens programmeren), voegen we de prefix '0x' toe: 0x38D
0x11 is dus gelijk aan 17

Voorstelling van karakters

De huidige computers gebruiken gestandaardiseerde tabellen waarin de alfanumerieke tekens die op het toetsenbord staan geassocieerd worden met binaire getallen. De meestgebruikte is de ASCII tabel.

De 128 tekens van hiernaast kunnen voorgesteld worden met 7 bits (2^7=128)

TABLE 3
ASCII CHARACTER CODES (DECIMAL)

0	Ctrl-@	32	Space	64	@	96	`
1	Ctrl-A	33	!	65	A	97	a
2	Ctrl-B	34	"	66	B	98	b
3	Ctrl-C	35	#	67	C	99	c
4	Ctrl-D	36	$	68	D	100	d
5	Ctrl-E	37	%	69	E	101	e
6	Ctrl-F	38	&	70	F	102	f
7	Ctrl-G	39	'	71	G	103	g
8	Backspace	40	(	72	H	104	h
9	Tab	41	)	73	I	105	i
10	Ctrl-J	42	*	74	J	106	j
11	Ctrl-K	43	+	75	K	107	k
12	Ctrl-L	44	,	76	L	108	l
13	Return	45	-	77	M	109	m
14	Ctrl-N	46	.	78	N	110	n
15	Ctrl-O	47	/	79	O	111	o
16	Ctrl-P	48	0	80	P	112	p
17	Ctrl-Q	49	1	81	Q	113	q
18	Ctrl-R	50	2	82	R	114	r
19	Ctrl-S	51	3	83	S	115	s
20	Ctrl-T	52	4	84	T	116	t
21	Ctrl-U	53	5	85	U	117	u
22	Ctrl-V	54	6	86	V	118	v
23	Ctrl-W	55	7	87	W	119	w
24	Ctrl-X	56	8	88	X	120	x
25	Ctrl-Y	57	9	89	Y	121	y
26	Ctrl-Z	58	:	90	Z	122	z
27	Escape	59	;	91	[	123	{
28	Ctrl-\	60	<	92	\	124	|
29	Ctrl-]	61	=	93	]	125	}
30	Ctrl-^	62	>	94	^	126	~
31	Ctrl-_	63	?	95	_	127	Delet