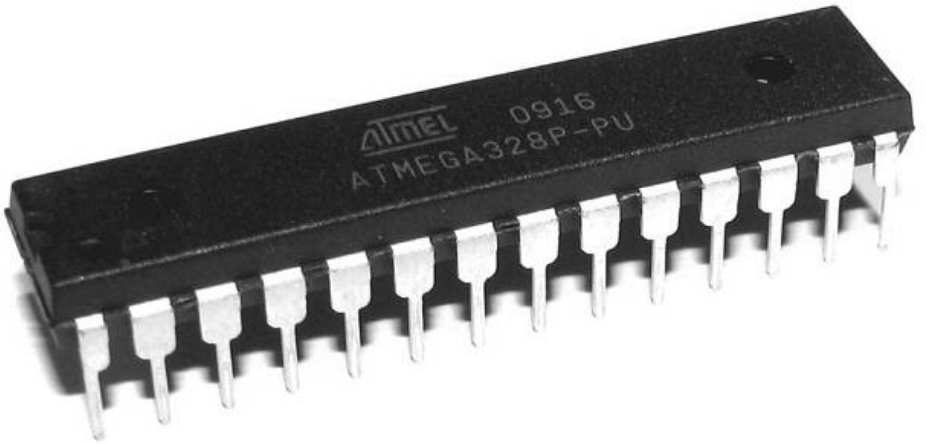
# Data types van getallen en snelheid

Richèl Bilderbeek

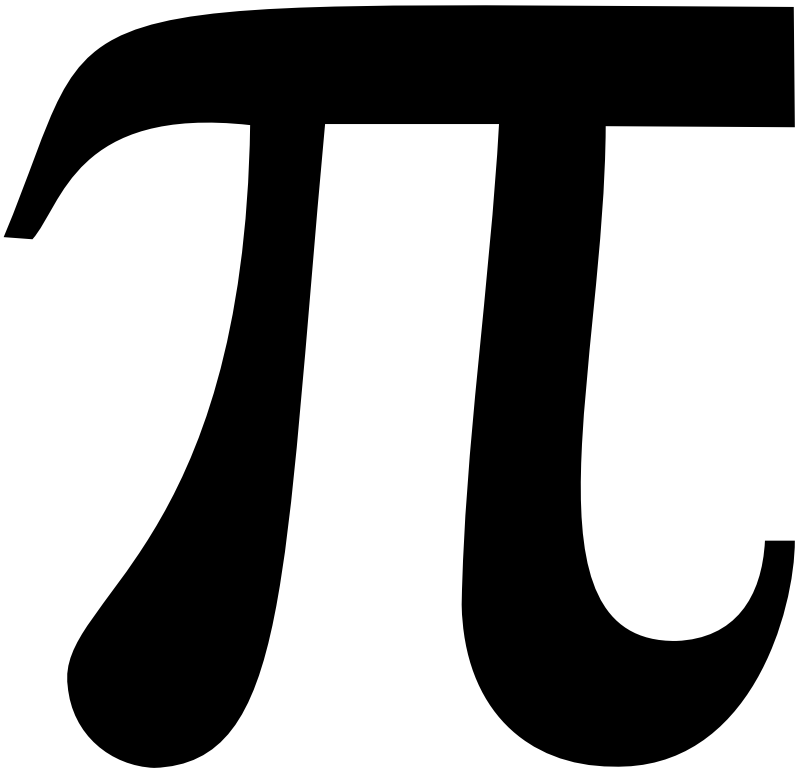


# Vragen

- Welk data type moet ik gebruiken?
  - Type
  - Bereik
  - Snelheid
- Antwoord is afhankelijk van:
  - Chip
  - C++ standaard
  - Beschikbare bibliotheken

# Soorten getallen

- Hele getallen
- Gebroken getallen

# Hele getallen

# Hele getallen

- 10 types!
- 5 basis types: char, short, int, long, long long[1]
- 2 modifiers: signed, unsigned
  - int = signed int
  - unsigned int ≠ int

- Bereikgrootte afhankelijk van het aantal bytes van het basis type
- Dit is chip afhankelijk!

[1] long long is een C++11 data type

# Hele getallen

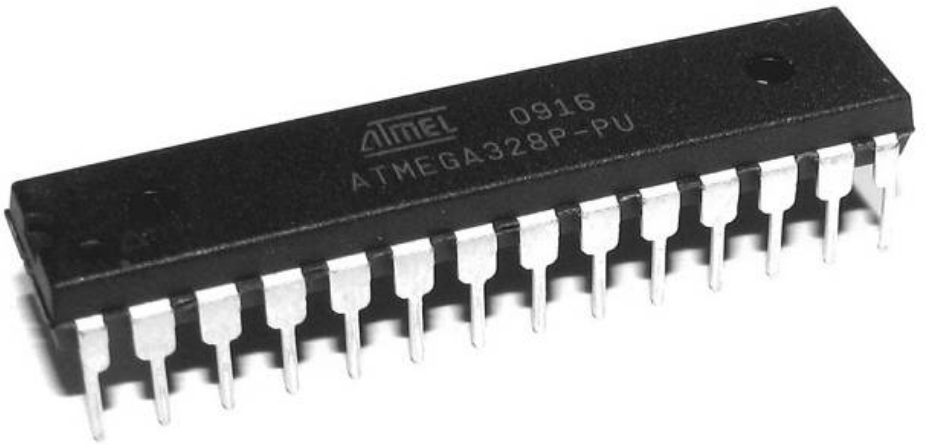
```
void setup() {
  Serial.begin(9600);
  Serial.println("Datatype sizes (in bytes)");
  Serial.print("char: "); Serial.println(sizeof(char));
  Serial.print("short: "); Serial.println(sizeof(short));
  Serial.print("int: "); Serial.println(sizeof(int));
  Serial.print("long: "); Serial.println(sizeof(long));
  Serial.print("long long: "); Serial.println(sizeof(long long));
}

void loop() {}
```

# Atmega 328P-PU

```
Datatype sizes (in bytes)
char: 1
short: 2
int: 2
long: 4
long long: 8
```

32 KB flash geheugen
8 bit



# Bereik

- Bereik van n bytes: $(2^8)^n = (256)^n$
- Unsigned: [0,b-1]
- Signed: [- ½ * b, ½ * b - 1]

- Bijvoorbeeld: char = 1 byte (per definitie)
- Bereik: $(2^8)^n = (256)^n = 256$
- unsigned char: [0,255]
- char: [-128,127]

# Atmega 328P-PU

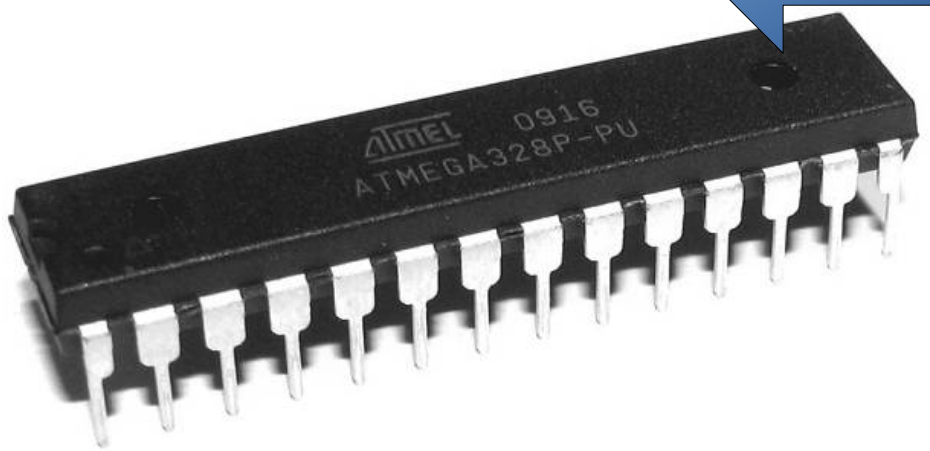
```
char: 127
short: 32767
int: 32767
long: 2147483647
long long: 9223372036854775807
```

$2^8 / 2 - 1$

$2^{16} / 2 - 1$

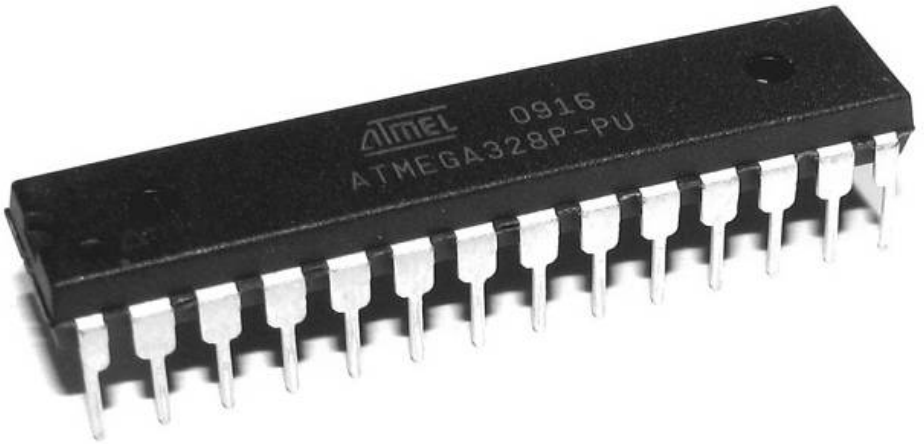
$2^{32} / 2 - 1$

$2^{64} / 2 - 1$



# Snelheid

- Snelheid is te meten met een benchmarks
- Een benchmark is moeilijk te schrijven
- Resultaten van een benchmark verschillen per chip
- De conclusies van een benchmark zijn lastig te trekken



# Benchmark

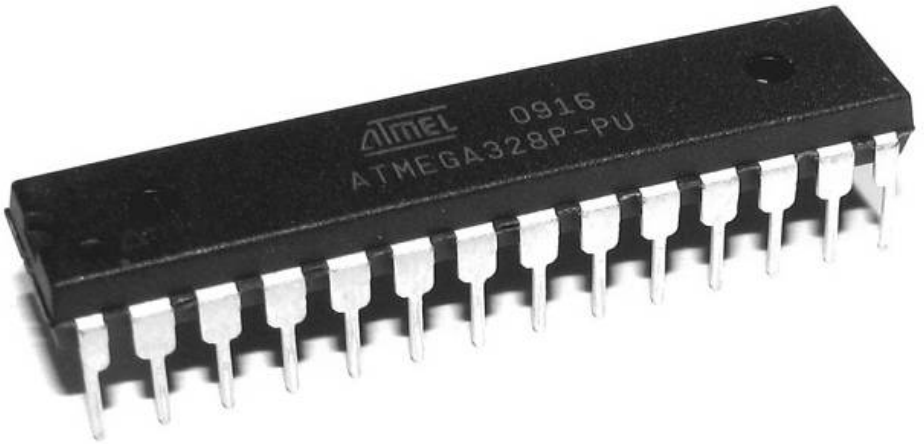
```
template <class T>
int Test() { //Return type is int, omdat T niet mag
  const int sz = 150; //Arraygrootte
  const int r = 1000; //Aantal herhalingen
  T v[sz]; //Array met willekeurige waarden
  T sum = 0;
  for (int j=0; j!=r; ++j) {
    for (int i=0; i!=sz; ++i) {
      ++v[i];
      sum += v[i];
    }
  }
  return sum; //Geef iets meetbaars terug
}
```

# Benchmark

```
void loop() {
  const double t0 = millis();
  const int s0 = Test<char>();
  const double t1 = millis();
  const double t_char = t1-t0;
  Serial.print("char: ");
  Serial.println(t_char);
  Serial.print("char (per byte): ");
  Serial.println(t_char/sizeof(char));
}
```

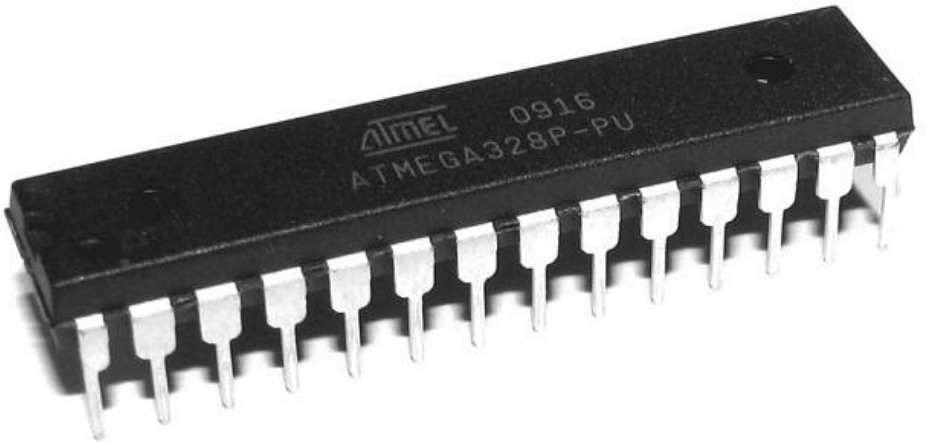
# Atmega 328P-PU

```
char: 95.00
short: 151.00
int: 152.00
long: 264.00
long long: 982.00
```



# Atmega 328P-PU

```
char (per byte): 94.00
short (per byte): 75.50
int (per byte): 76.00
long (per byte): 66.00
long long (per byte): 122.75
```



# Conclusies Atmega 328P-PU

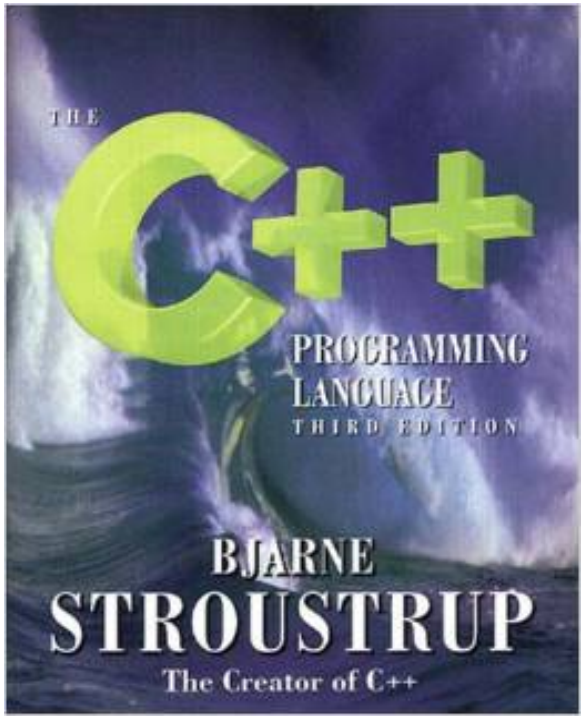
- short en int zijn hetzelfde
- Snelheid is niet van belang: het bereik varieert meer dan snelheid:
  - Bereik long long is 2000000000x groter dan long
  - Snelheid long long is 2x hoger per byte dan long

# Discussie

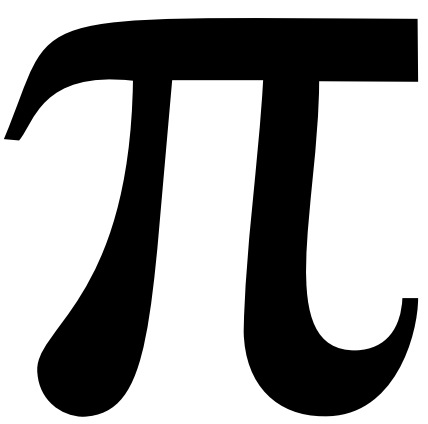
- Hele getallen met groter bereik nodig?
  - Gebruik dan een bibliotheek
  - (Google op 'big integer library Arduino')
  - Toepassing?
- Het bereik van een long long is misschien onnodig groot:
  - Meer dan het aantal zandkorrels op alle stranden ter wereld
  - Meer dan het aantal sterren binnen het zichtbaar universum

# Advies

- Kies het kleinste data type dat past bij het bereik
- Kies liever te groot dan te klein
- In de regel: int
- Gebruik geen unsigned (Stroustrup, 1997, 2005):
  - Onverwachte uitkomsten bij berekeningen
  - Moeilijker debuggen



# Gebroken getallen

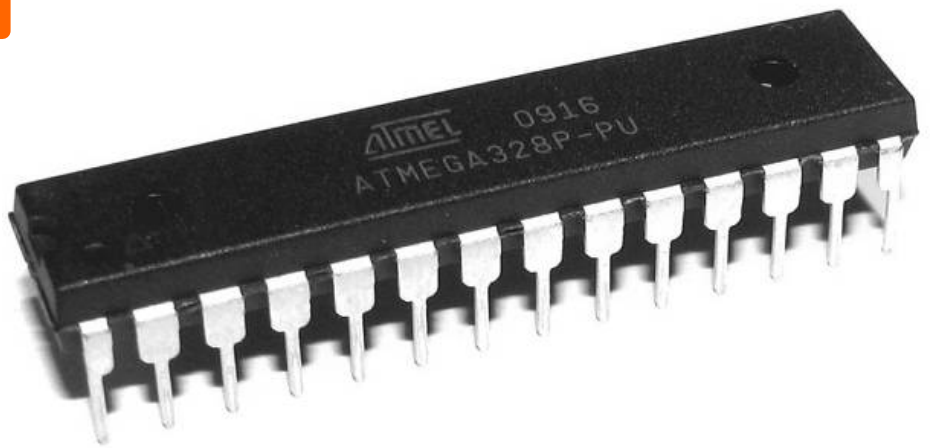


# Gebroken getallen

- 3 types: float, double en long double
- Geen modifiers

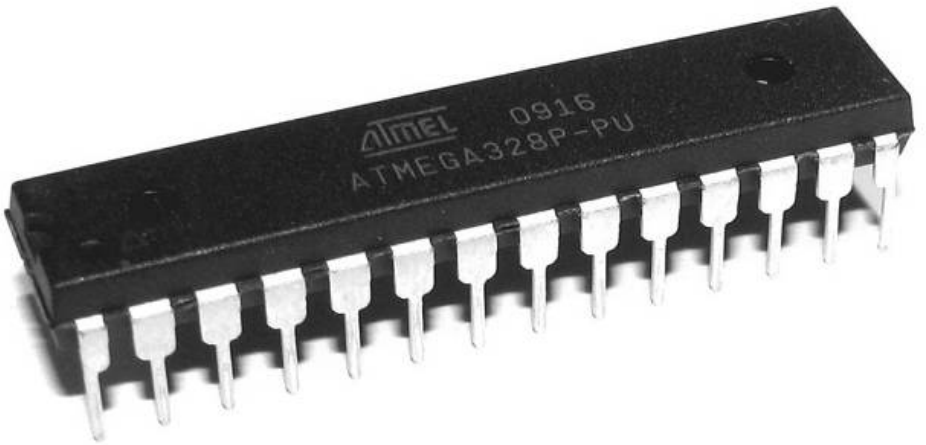
# Grootte

```
Datatype sizes (in bytes)
Gebroken getallen:
double: 4
float: 4
long double: 4
```



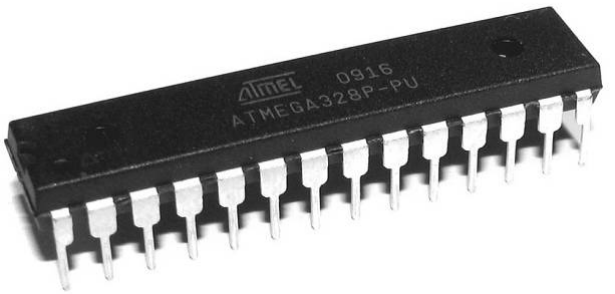
# Bereik

- Van $-3.4 * 10^{38}$ tot $3.4 * 10^{38}$
- 6-7 decimalen precisie
- Laagste waarde boven nul: 1.17549e-38



# Snelheid

```
float: 2173.00
double: 2177.00
long double: 2186.00
float (per byte): 543.25
double (per byte): 544.25
long double (per byte): 546.50
```



# Conclusie

- float en double en long double zijn hetzelfde op de Atmega 328P-PU

# Discussie

- Nauwkeurigere gebroken getallen nodig? Gebruik dan een bibliotheek (Google op 'arbitrary precision floating point library Arduino)
  - Toepassing?
- De meeste sensoren hebben veel minder dan 6-7 decimalen precisie

# Eindconclusie

- Hele getallen:
  - Gebruik geen unsigned
  - Kies een type met het juiste bereik
- Gebroken getallen:
  - Type maakt niet uit
- Snelheid maakt geen verschil