CPSC 1000: Introduction to Computer Science

Reading the voltage of an analog signal

Robert Benkoczi, C556
robert.benkoczi@uleth.ca

25-Sep-2018
(Week 3)

Objectives

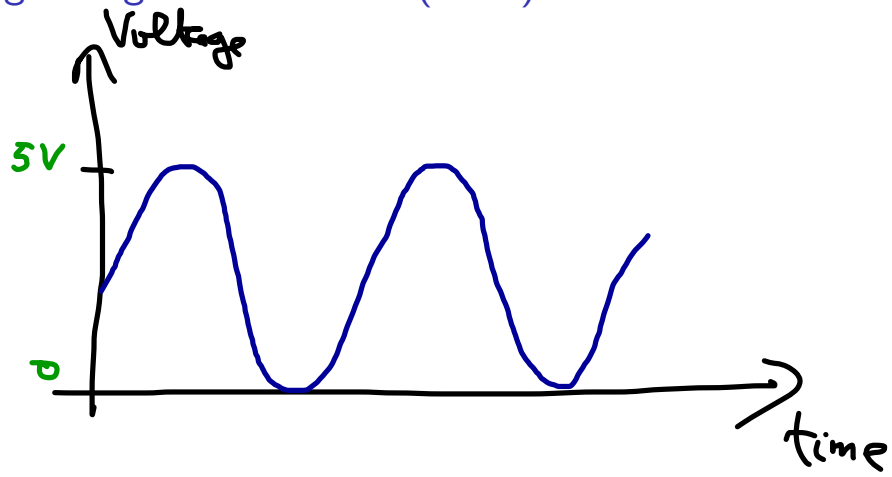
- Using the Lab Manual and the sample code available on Moodle, students will correctly connect a device generating an analog signal to the Arduino.
- Students will write code to measure the analog signal.
- Students will output information from the Arduino board to the serial monitor window on the workstation.

Motivation

- Many inexpensive sensors transmit their information through an analog signal. The code we use in this activity can also be used to read these sensors (ex: light, sound, temperature, etc.)
- A potentiometer can also be used as a form of input for an Arduino project (ex: to configure the device).

Reading an analog signal

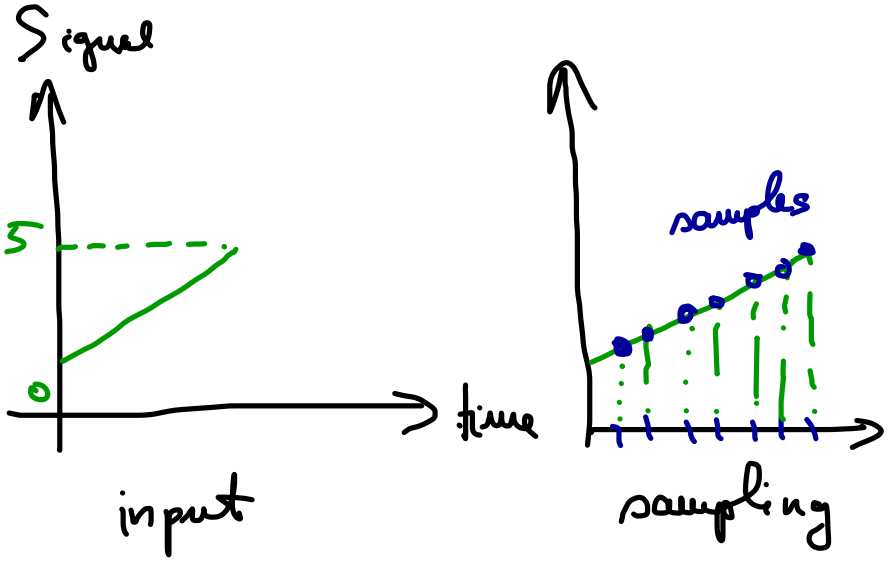
Analog to digital conversion (ADC)

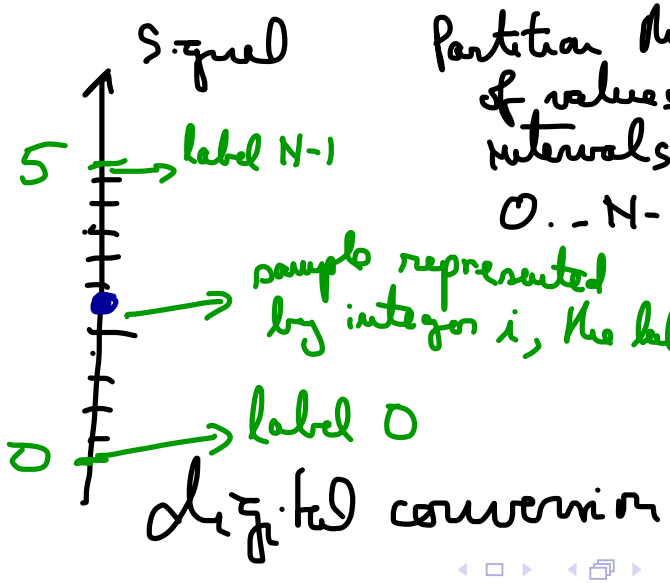


Analog signal : signal whose parameters (voltage) varies continuously in time.

Min value = 0V
Max value = 5V
} for Arduino.

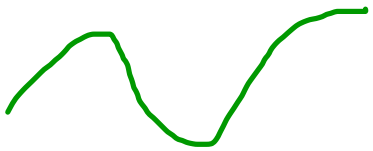
Analog to digital conversion (ADC) ... reading the analog signal...





Partition the range of values in N equally spaced intervals; Assign integers 0..N-1 to each interval

Analog to digital conversion summary:



input

$\longrightarrow$

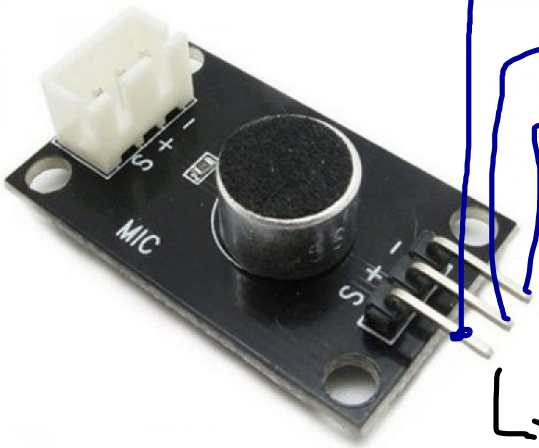
$x_1, x_2, x_3, \ldots \in \{0, 1, \ldots, N-1\}$

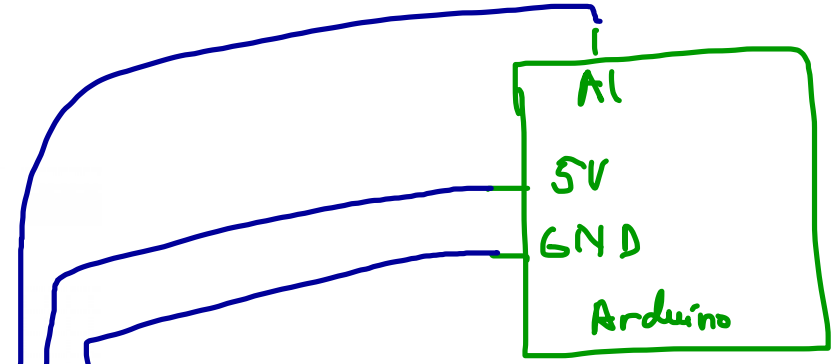
output

$N = 2^c$, For arduino $N = 2^{10} = 1024$

Applications

Recording sound





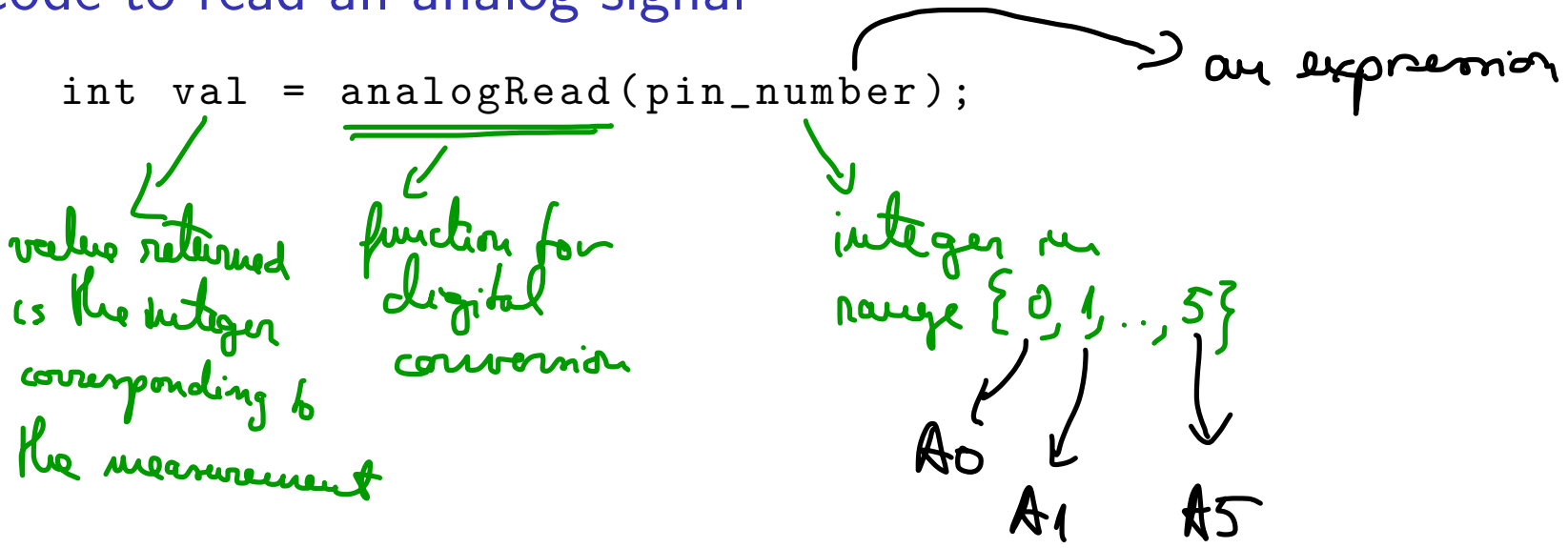
} power to the sensor

↳ analog signal generated (0V – 5V range)

→ we can use the power provided by the Arduino

Code to read an analog signal

```
int val = analogRead(pin_number);
```



Q: why not calling the function:

analogRead(A0);?

=

↓

expressions

(something that has a value)

Kind 1: formula (uses +, /, ±, -, ())

Kind 2: variable

Rules for variable names:

1) Must start with a letter or _ (A..Z, a..z, _)

2) Any # of letters, digits, or _ can follow:

ex: A0, city, birth_date ...

Kind 3: values.

Rules for numbers: like math: -2.3

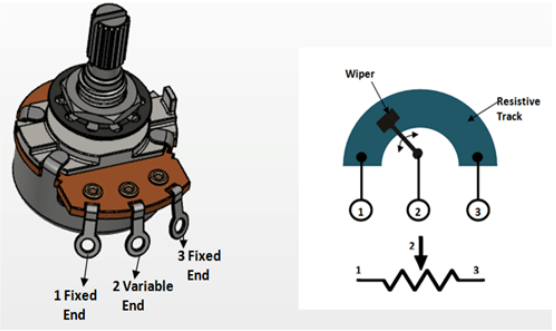
Rules for text: " ... "

first text last

↙

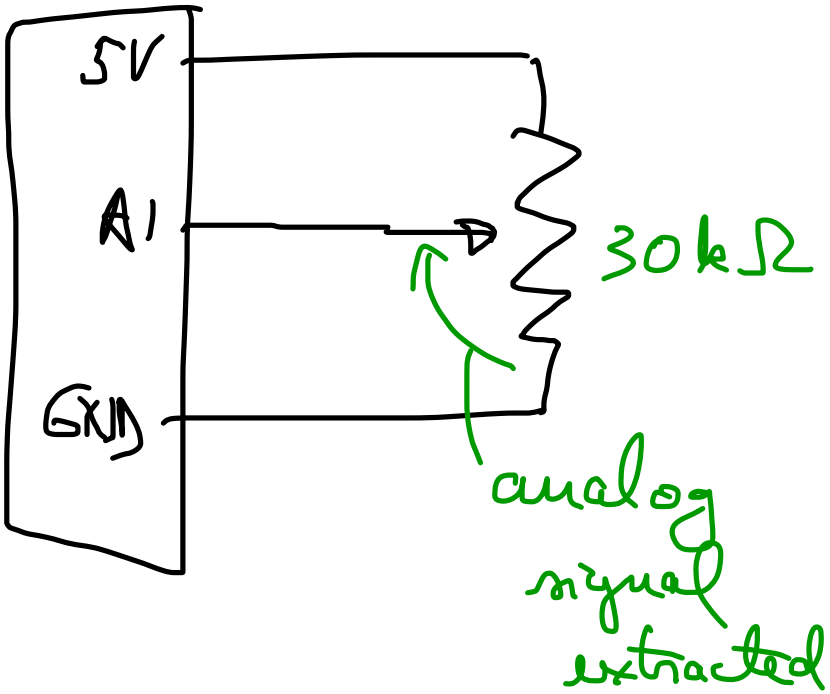
A: you can use analogRead(A0) provided you declare variable A0 & initialize it.

Simulating an analog signal with potentiometer



Homework: what is the amount of current flowing between GND & 5V pins for a 30 kΩ potentiometer?

Use $U = R \cdot I$ ← current
voltage, resistance



Q2: is our choice safe?
$I \leq 40\,mA$?

Application : reading a light sensor

Sensor → search for "light sensor brick".

Connections (see sound sensor page) →
- V : 5V on Arduino
- G : GND
- S : A0.

Code : identical to Activity 2 (potentiometer).

```
...
void loop() {
  Serial.println(analogRead(0));   ⇐
  delay(..);
}
```

How do we "process" information from a sensor?

Ex : detect if lights are ON or OFF.

Q : what does it mean to detect that lights are OFF/ON ?

→ use a threshold for the sample read from the sensor

Ex

400 ≥ value from sensor → Lights ON

value from sensor > 400 → Lights OFF

Task : → print message ON (light on)

↘ — " — OFF (light off)

Arduino program :

" If the value from the sensor is less than or equal to 400, write the message "ON".
Otherwise, write the message OFF. "

(English version)

```
void loop() {
  int value = analogRead(0);
  if (value <= 400) {
    Serial.println("ON");
  } else {
    Serial.println("OFF");
  }
  delay(500);
}
```

(Arduino version)

If statement syntax.

```
if ( boolean_expression ) {
        Instruction 1 ;
        Instruction 2 ;
        :
} else {
        Instruction A ;
        Instruction B ;
        :
}
```

Execution :

- evaluate the boolean expression
- execute Instruction 1, 2, ... if the boolean expression was true.
  DONE
- execute Instructions A, B, ... if the boolean expression was false. DONE.

Note: only one of the sets of instructions (1, 2...) or (A, B, ..) is executed.

Boolean expressions

→ an expression that can be either true or false.

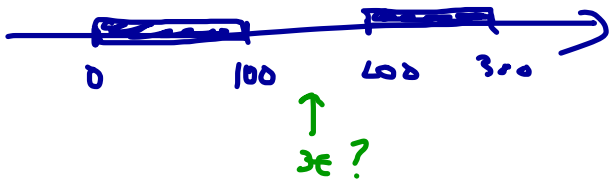
Examples

- boolean values: true, false
- integer values: { 0 : false
  anything else : true
- logical operators: and (&&), or (||), not (!)
  - English
  - Arduino
- comparison: ≤ (<=), ≥ (>=), = (==), < (<), > (>), ≠ (!=)
  - math
  - Arduino

Examples of tests

int x, y, z; (suppose they are initialized)

	(boolean expression)
a) x is equal to 100	x == 100
x is greater than y	x > y
$x \in [0, 100] \cup [200, 300]$ (intervals)	x inside [0, 100] OR x inside [200, 300] ... (to be translated further.)



$x \in [0, 100]$ ← set notation

Math: $0 \leq x \leq 100$ ← inequalities

in program ↓

Attempt 1: `0<=x<=100` ← incorrect because of the way expressions are evaluated:

Ex: `2+3<7*8`

2nd `5` `56` — first

`61` last

`0<=x<=100`

`True` (represented by `1`)

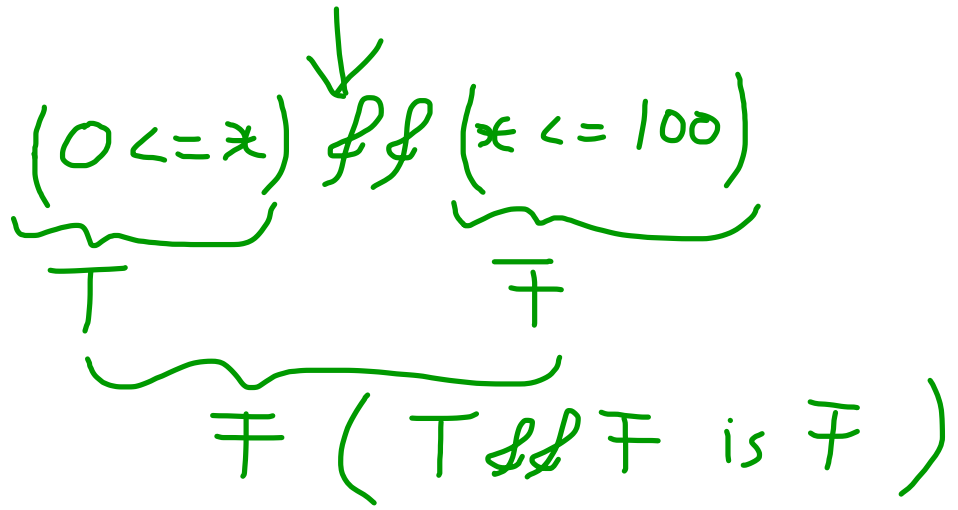
`1<=100`

`True`

Suppose `x=200`

We translate $0 \leq x \leq 100$ by: $0 \leq x$ AND $x \leq 100$

$x = 200$

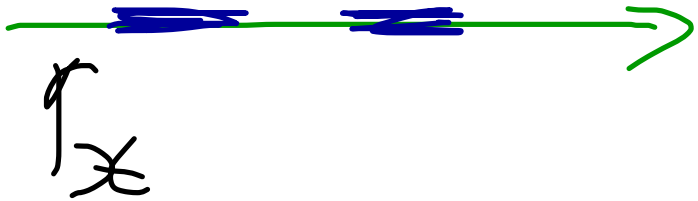


And: (both arguments must be true for the expression to be true)

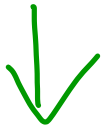
Or: (at least one argument must be true for the expression to be true.

Ex. 2: test for $x \in [0,100] \cup [200,300]$.

Step 1: write the "test" in English.



"x in the range $[0..100]$ OR
in the range $[200, 300]$."



```
((0<=x) && (x<=100)) || ((200<=x) && (300>=x))
```

Example 3:

Suppose $x \in \{1, 2, 3\}$. Write a message as follows:

x	Message
1	"one"
2	"two"
3	"three"

Strategy: - write instructions "telling the computer" what to do to accomplish the task. (The value of x is in a box. We don't see it, but the computer can read it).

Solution 1:

"Dear computer, if x is 1 then write "one". If x is 2 then write two. If x is 3 then write three.

↓

```
if(x==1) {
    Serial.println("one");
}
if(x==2){
    Serial.println("two");
}
if(x==3) {
    Serial.println("three");
}
```

Note: we can omit the else{} in an if statement

Solution 2:

Dear computer, if x is 1 then write one, otherwise if x is 2 write 2, otherwise write 3.

```
if (x==1) {
    Serial.println("one");
} else if (x==2) {
    Serial.println("two");
} else {
    Serial.println("three");
}
```

Example three

Suppose x is a fractional variable (float x) in the range 0...100 (percentage). Write the following messages depending on the value of x:

x	Message
$[0, 20]$	low
$(20, 80)$	normal
$[80, 100]$	high

Solution

Step 1: instruction sheet: Dear computer,

if $x \leq 20$ then write low, otherwise if $x < 80$ then write normal, otherwise write high.

Testing

$x = 75$ ... x not less than 20, so we test $x < 80$? (75) → yes "Normal"

OK.

A more interesting case: $x = 80$ (boundary condition). Homework: verify

Homework: write the code.

Homework 2: write a different set of instructions for the same problem, where the cases appear in the order: normal, high, low.