Embedded pool

Day 07 : Analog

contact@42chips.fr

Summary: Analog is not automatic

Chapter I

Introduction

Rosin is the solid residue obtained after distilling turpentine.
It is collected from resinous trees, particularly from the genus Pinus, through a process called "gumming."

The name comes from Kolophon, an ancient Greek city in Asia Minor where this substance was produced.
Rosin is solid and brittle at room temperature, with colors ranging from very light yellow to almost black depending on the distillation process.

Its color, or grade, is defined by a letter scale ranging from D for the darkest to X for the lightest.

Rosin does not melt but softens when heated, with its softening point varying from 90 to 110 °C.
It is composed of 90% of a mixture of organic acids from the diterpene family called resin acids, which are isomers.
Rosin can be used as a flux in soft soldering.

Fluxes are used to reduce the surface tension of the molten solder and enable it to flow more easily to quickly and effectively cover the surfaces of the parts being soldered.

Since oxide layers continuously form on surfaces when heated, fluxes are used to dissolve and eliminate them. To make it easier to use, hollow soft solder wires are manufactured that contain one or more flux cores.

Chapter II

General instructions

Unless explicitly stated otherwise, the following instructions will be valid for all assignments.

- The language used for this project is C.
- It is not necessary to code according to the 42 norm.
- The exercises are ordered very precisely from the simplest to the most complex. Under no circumstances will we consider or evaluate a complex exercise if a simpler one is not perfectly successful.
- You must not leave any files other than those explicitly specified by the exercise instructions in your directory during peer evaluation.
- All technical answers to your questions can be found in the `datasheets` or on the Internet. It is up to you to use and abuse these resources to understand how to complete your exercise.
- You must use the datasheet of the microcontroller provided to you and comment on the important parts of your program by indicating where you found the clues in the document, and if necessary, explaining your approach. Don't write long blocks of text, keep it clear.
- Do you have a question? Ask your neighbor to the right or left. You can ask in the dedicated channel on the Piscine's Discord, or as a last resort, ask a staff member.

Chapter III

v=p7YXXieghto

	Exercise 00
The nozzle is initializing	
Turn-in directory : *ex*00/	
Files to turn in : `Makefile, *.c, *.h`	
Allowed functions : `avr/io.h, util/delay.h, avr/interrupt.h`	
Notes : `n/a`	

Read the value of the linear potentiometer RV1 using the ADC peripheral.

- ADC must be configured with an 8-bit resolution and AVCC as a reference.
- Then display its value in hexadecimal format every 20ms on the console.

```
...
00
a1
...
```

	Exercise 01
Analog Reading V2	
Turn-in directory : *ex*01/	
Files to turn in : `Makefile, *.c, *.h`	
Allowed functions : `avr/io.h, util/delay.h, avr/interrupt.h`	
Notes : `n/a`	

- Read the potentiometer RV1 + the LDR (R14) + the NTC (R20)
- After that, you must display the values in hexadecimal format every 20ms on the console.

```
...
00, ef, ff
00, ef, ff
...
```

Chapter IV

10 bit funk

	Exercise 02
Isn't that a bit much ?	
Turn-in directory : *ex*02/	
Files to turn in : `Makefile, *.c, *.h`	
Allowed functions : `avr/io.h, util/delay.h, avr/interrupt.h`	
Notes : `n/a`	

- Read the potentiometer RV1 + the LDR (R14) + the NTC (R20)
- But this time, you must use the ADC in 10bit mode
- After that, you must display the values in decimal format every 20ms on the console.

```
...
0, 128, 1023
0, 128, 1023
...
```

	Exercise 03
Temperature Reading	
Turn-in directory : *ex*03/	
Files to turn in : `Makefile, *.c, *.h`	
Allowed functions : `avr/io.h, util/delay.h, avr/interrupt.h`	
Notes : `n/a`	

The Atmega328P is able to read its internal temperature. It's not ultra precise but we will read it anyway.

- Read the internal temperature sensor value
- Then display it on the console and convert it to celcius every 20ms.

```
...
20
22
25
...
```

Chapter V

You can fly !

	Exercise 04
Analog Color	
Turn-in directory : *ex*04/	
Files to turn in : `Makefile, *.c, *.h`	
Allowed functions : `avr/io.h, util/delay.h, avr/interrupt.h`	
Notes : `n/a`	

Read the RV1 value with your ADC.

- RV1 must be able to change the color of D5 with the Wheel function.
- Also the LEDs D1-D4 must display the value of RV1 as a digital gauge.
    - LED D1: 25%
    - LED D2: 50%
    - LED D3: 75%
    - LED D4: 100%

`With a touch of music, it's always better !`