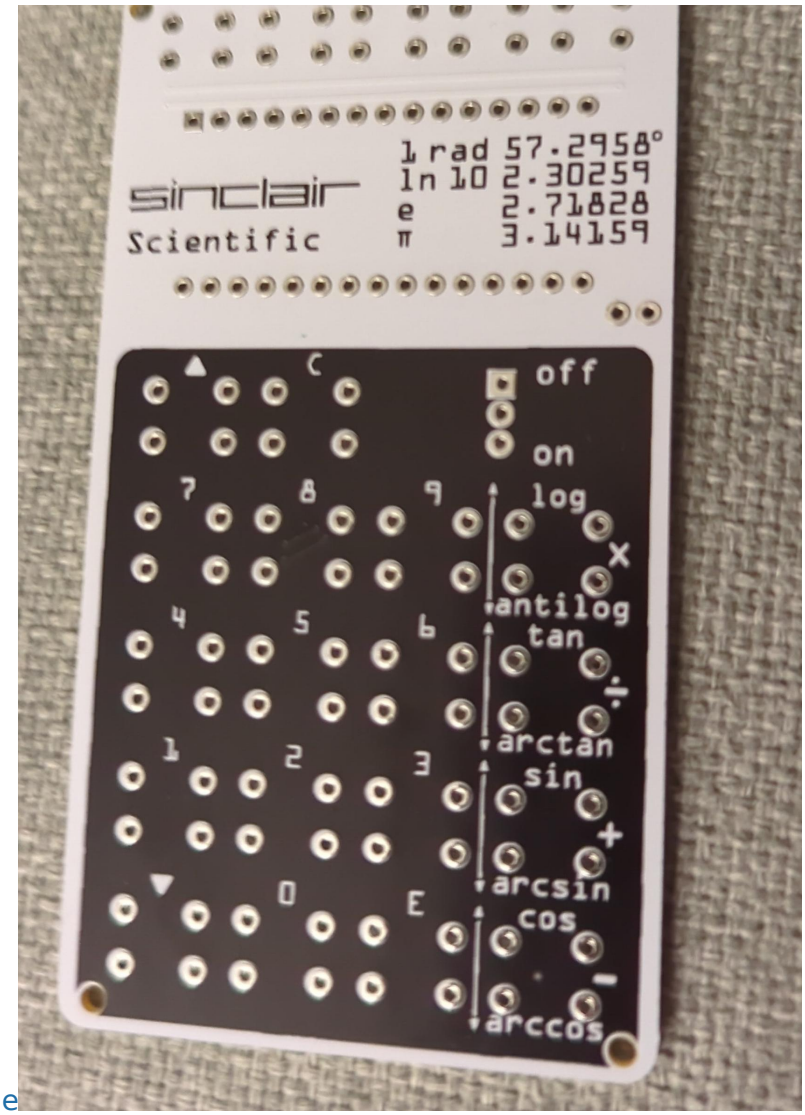
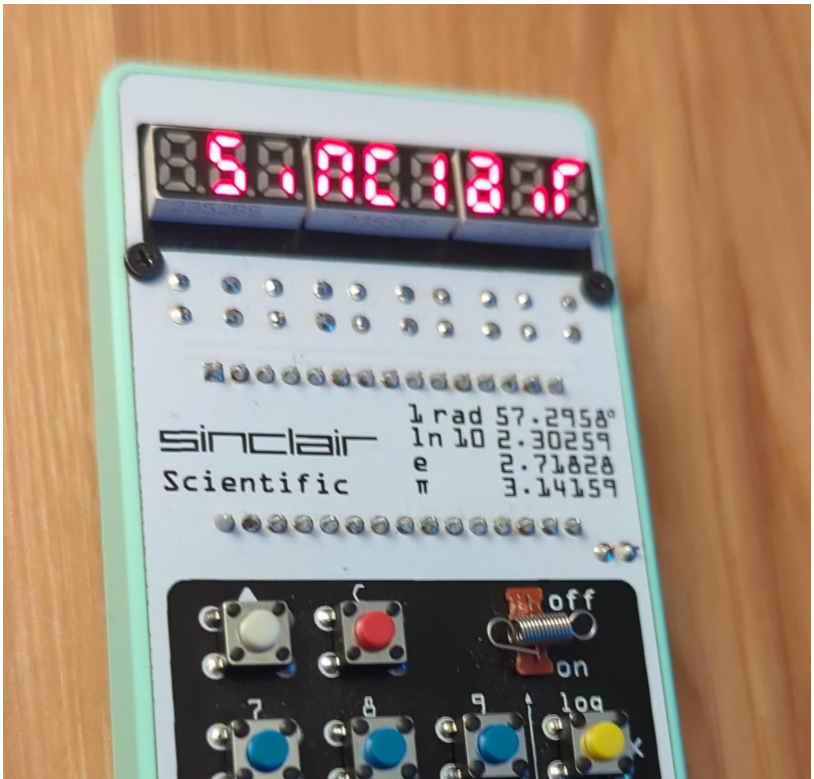
Sinclair Scientific Calculator Emulator (1974)


1 rad 57.2958°
ln 10 2.30259
e 2.71828
π 3.14159
sinclair
Scientific
C
off
on
7
8
9
log
x
antilog
4
5
6
tan
÷
arctan
1
2
3
sin
+
arcsin
0
E
cos
-
arccos

e


5.AE12.F
1 rad 57.2958°
ln 10 2.30259
e 2.71828
π 3.14159
sinclair
Scientific
C
off
on
7
8
9
log

> A register level TMS0805 CPU emulator on an Arduino Nano runs the original 320 instruction calculator program. A custom PCB houses it all.

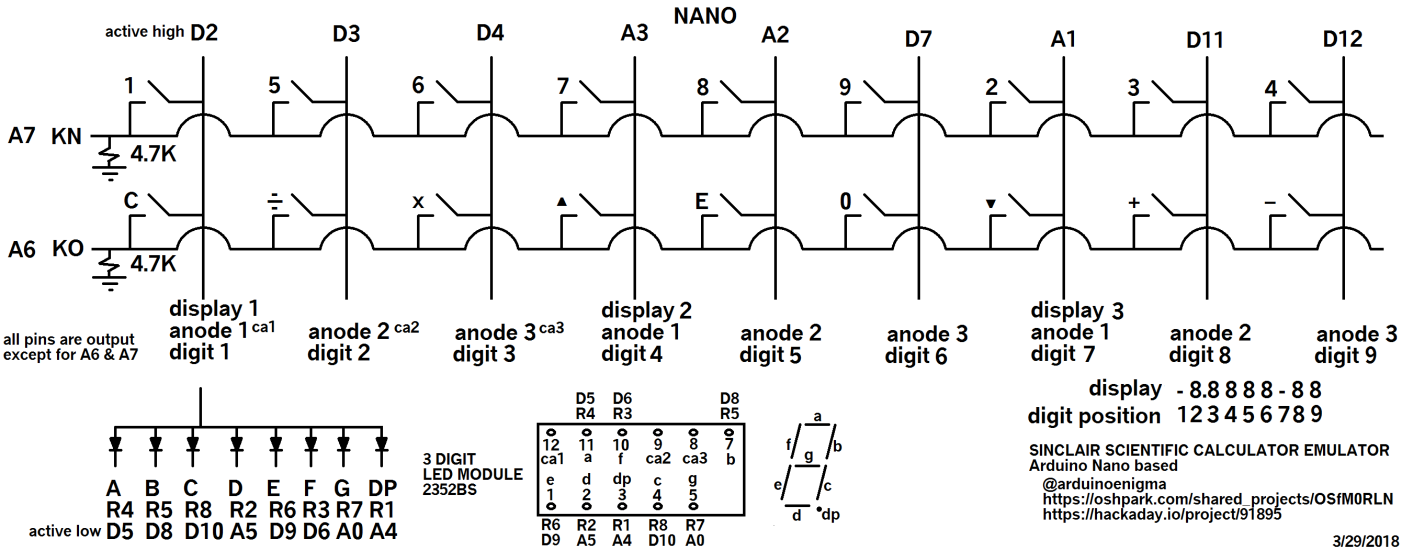
Resources

- Reversing Sinclair's amazing 1974 calculator hack

  > Now Texas Instruments offered him an inexpensive calculator chip that could barely do four-function math. Could he use this chip to build a $100 scientific calculator?
  > Texas Instruments' engineers said this was impossible - their chip only had 3 storage registers, no subroutine calls, and no storage for constants such as π. The ROM storage in the calculator held only 320 instructions, just enough for basic arithmetic. How could they possibly squeeze any scientific functions into this chip?
  > Fortunately Clive Sinclair, head of Sinclair Radionics, had a secret weapon - programming whiz and math PhD Nigel Searle. In a few days in Texas, they came up with new algorithms and wrote the code for the world's first single-chip scientific calculator, somehow programming sine, cosine, tangent, arcsine, arccos, arctan, log, and exponentiation into the chip. The engineers at Texas Instruments were amazed.

- Project page & build instructions: Sinclair Scientific Calculator Emulator

Schematics



> This is a custom PCB shape. A 50x100mm rectangle with 3mm radius corners.

Power

The switch is connected to VIN ping on Arduino Nano, so it goes to the 5V regulator that can handle up to 15V and has ~1.1V dropout. With MCU needing 1.8V at minimum, the board should be supplied with at least 2.9V.

- With 3xAA batteries:
  - 4.35V *VIN*, I get 3.27V on *5V* pin; so ~1.1V dropout on the regulator.
  - 4.32V directly to 5V pin will let it run longer on batteries as there is no dropout of the regulator; 3V3 pin shows 3.27V; but this would bypass the switch

Programming

The board uses Arduino Nano:

- Software: https://gitlab.com/arduinoenigma/ArduinoNanoSinclairScientificCalculator/-/tree/master?ref_type=heads
  - Use https://gitlab.com/arduinoenigma/ArduinoNanoSinclairScientificCalculator/-/blob/master/SinclairScientific7/SinclairScientific7.ino?ref_type=heads (SinclairScientific7.ino)

- Add library ZIP (**Sketch** / **Include Library** / **Add .ZIP Library...)** from: https://gitlab.com/arduinoenigma/ArduinoNanoSinclairScientificCalculator/-/blob/master/SinclairScientific1/libraries/Arduino-GPIO-master.zip?ref_type=heads (SinclairScientific7.ino)
- I have changed serial settings on line 310 to use standard baud rate but note that it may not work for your board as it is badly timed at 16MHz clock (see https://wormfood.net/avrbaudcalc.php)

```
diff --git a/SinclairScientific7/SinclairScientific7.ino
b/SinclairScientific7/SinclairScientific7.ino
index 487d8f2..ec2e143 100644
--- a/SinclairScientific7/SinclairScientific7.ino
+++ b/SinclairScientific7/SinclairScientific7.ino
@@ -307,7 +307,7 @@ void setup()
 {
   // put your setup code here, to run once:

-  Serial.begin(2000000);
+  Serial.begin(115200);

   // makes it easier to see if an arduino is programmed or not
   Serial.print(F("SINCLAIR v7 092318"));
```

After using IDE to build and upload you can connect to it via UART to get hello message:

```
hxd@morgana ~> tio /dev/ttyUSB0
[22:06:24.423] tio v2.7
[22:06:24.423] Press ctrl-t q to quit
[22:06:24.424] Connected
SINCLAIR v7 092318 -Common Anode -Aligned Right
```

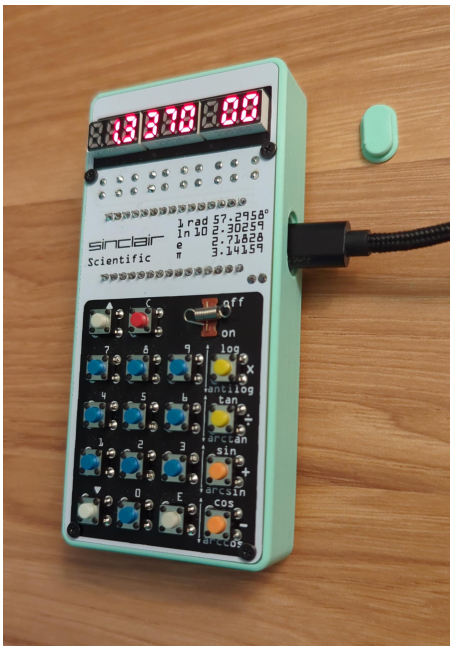
- Alternative software calculator implementation in Rust: https://gitea.hexadust.net/hxd/sinclair-sci-calc

Box

I have designed a box with battery and Arduino USB port access for easy battery replacement and programming or USB power.

>

> USB port access assumes that Arduino board was soldered with a distance from the main board - this is to avoid having to cut Arduino board header pins leaving sharp edges.



- FreeCAD model (see *Spreadsheet* for tuning): Sinclair_Scientific.FCStd
- Exported models:
  - USB-C:
    - Sinclair_Scientific-Body-USB-C.stl
    - Sinclair_Scientific-USB Door-USB-C.stl
  - USB mini-B:
    - Sinclair_Scientific-Body-USB-mini-B.stl
    - Sinclair_Scientific-USB-mini-B.3mf
  - Common:

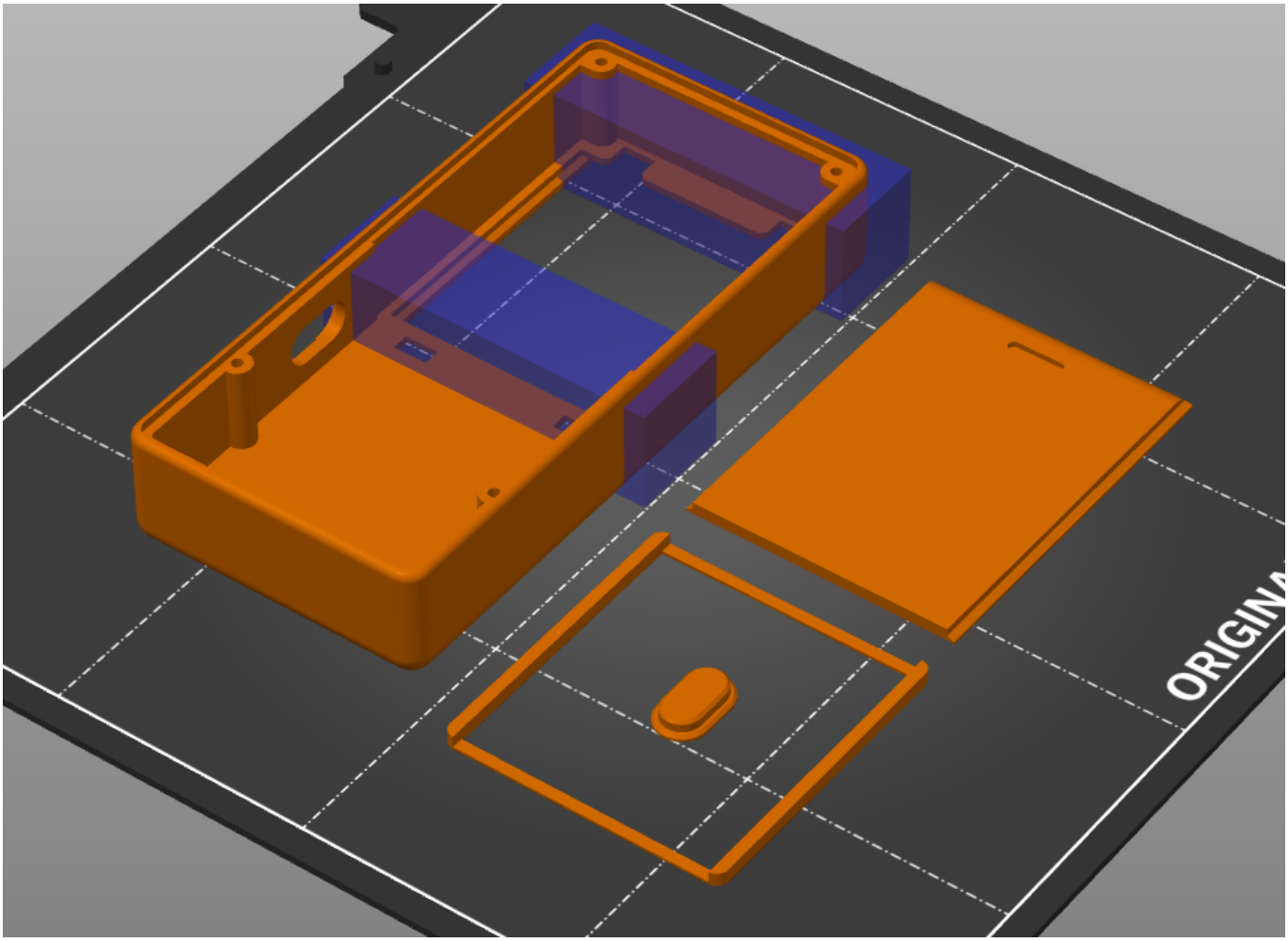
    - Sinclair_Scientific-Battery Door.stl
    - Sinclair_Scientific-Battery pack holder.stl
- PrusaSlicer project:
    - USB-C: Sinclair_Scientific-USB-C.3mf
    - USB min-B: Sinclair_Scientific-USB-mini-B.3mf

Printing

Make sure that main body has supports printed under the battery door area. The battery pack holder goes inside the box on top of the battery pack to prevent it getting pushed in when replacing the batteries. Before assembly use lubricant on sides and on the latches of the battery doors for smooth operation.

Object scaling (may vary from printer to printer and printing settings):

- For battery bay doors use 99% scale for sides axis (y), 99.8% for length (x) and 98% for thickness (z).
- For USB doors use 99% scale for all axis.



Usage

As per User Manual:

- Enter firs number followed by `+` or `-` for negative number (`0+<number>`)
- Use `E` key to start entering exponent - 2 numbers can be entered, further presses overwrite entered numbers; press `-` before entering numbers for negative exponent
- Enter second number
- Select operation (press up or down arrow followed by operation for alternative operation)

Calculation	**Key Sequence**	**Result Display**
Basic input		
`592`	`592E2+`	`5.9200 00`
`4.29`	`429+`	`4.9200 00`
`0.0037`	`037E-2+`	`3.7000-03`
`0.5673*10^-12`	`05673E-12+`	`5.6730-13`
`6.7*10^-3` (`0.0067`)	`067E-2+`	`6.7000-03`
Reverse Polish notation		
`18*((4.5-3.2)/7)`	`45+32-7%18E1x`	`3.3427 00`
`(0.326-0.583)*1.48*10^7`	`0326+0583-148E7x`	`-3.8936 06`
Logarithm ($\log_{10}$)		
`log 1`	`1□x`	`0.0000 00`
`log 3.6`	`36□x`	`5.5634-01`
`log 71000`	`71E4□x`	`4.8512 00`
`log 10`	`1E1□x`	`1.0000 00`
Natural logarithm ($\log_e$) - Multiply by $\log_e 10$ (`ln10 2.30259`)		
$\log_e 5$	`5□x23026x`	`1.6095 00`

Anti-logarithm (10^x) - input from `0.0` to `99.999`, error aprox `0.001`		
`10^0`	`0␣x`	`1.0000 00`
`sqrt 10`	`05␣x`	`3.1621 00`
`10^1.5`	`15␣x`	`3.1621 01`
`10^67.5`	`675E1␣x`	`3.1621 67`
Exponential function (e^x) - Divide by $\log_e 10$ (`ln10 2.30259`)		
`sqrt(e)*(e^0.5)`	`05+23026%␣x`	`1.6486 00`
Sine, Cosine, Tangent - Angle between `0` and `PI/2` radians (`90º`), error less than `0.001`		
`sin 0.3966`	`03966␣+`	`3.8629-01`
`cos 0.66`	`066␣-`	`7.8994-01`
`tan 0.1322`	`01322␣%`	`1.3330-01`
Sine, Cosine, Tangent in degree - Divide by conversion factor (`1rad 57.2958º`)		
`sin 45º`	`45+573%␣+`	`7.0729-01`
`cos 60º`	`6+573%␣-`	`5.0008-01`
`tan 75º`	`3+573%␣%`	`3.7197 00`
Arcsine, Arccosine, Arctangent - Result in radians, input from `0.0` to `9.9995`, error max `0.001`		
`asin 0.9994`	`09994␣+`	`1.5350 00`
`acos 0.3`	`03␣-`	`1.2660 00`
`atan 3`	`3␣%`	`1.2500 00`
Arcsine, Arccosine, Arctangent in degree - Multiply result by conversion factor (`1rad 57.2958º`)		
`asin 0.5`	`05␣+573E1x`	`2.9967 01`
`acos 0.5`	`05␣-573E1x`	`6.0050 01`
`atan 1`	`1␣%573E1x`	`4.5038 01`
Roots		
`sqrt 6` (base 2)	`6␣x2%␣x`	`2.4495 00`
root base 3 of `47.6/1.7`	`476E1+17%␣x3%␣x`	`3.0367 00`

Revision #41
Created 2024-12-17 22:03:27 GMT by hxd
Updated 2025-07-18 21:17:58 IST by hxd