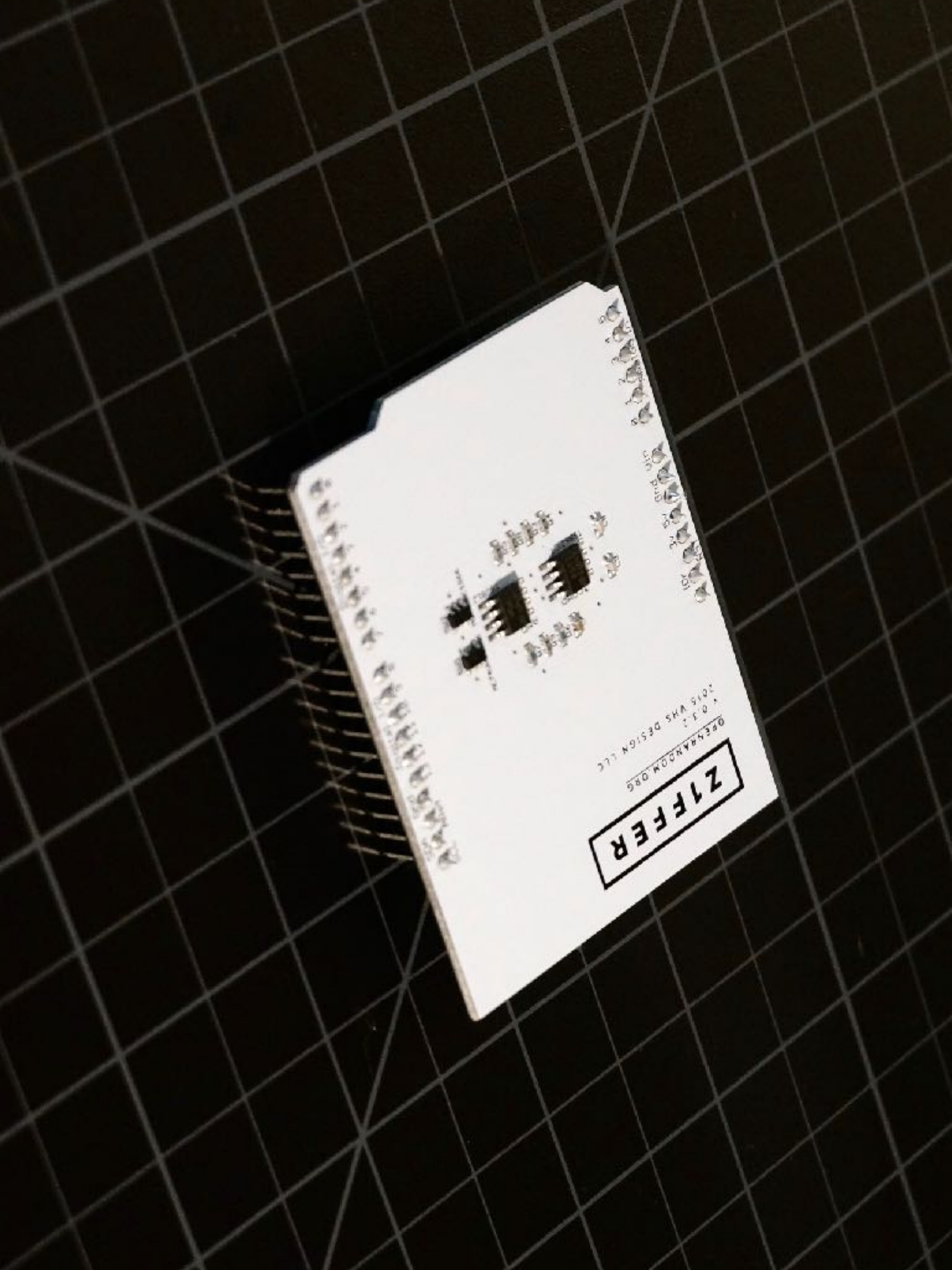
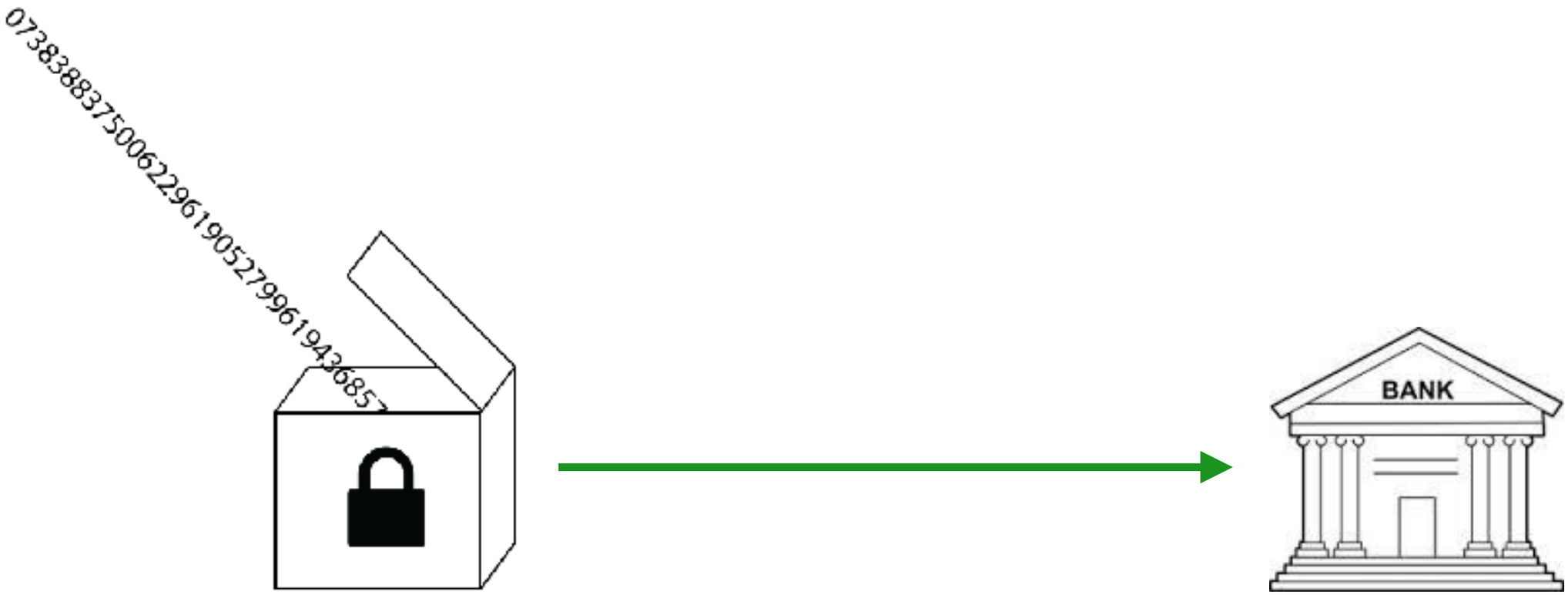
OpenRandom.org

Rob Seward


ZIFFER
OPENRANDOM DRG
V 0.1.1
2015 VHS DESIGN LLC

https://www.

07383883750062296190527996
19436857186693374046386300
67443280546436796557774892


07383883750062296190527996194368
BANK

07383883750062296190
52799619436857186693
37404638630067443280
546436796557774892

07383883750062296190
52799619436857186693
37404638630067443280
546436796557774892

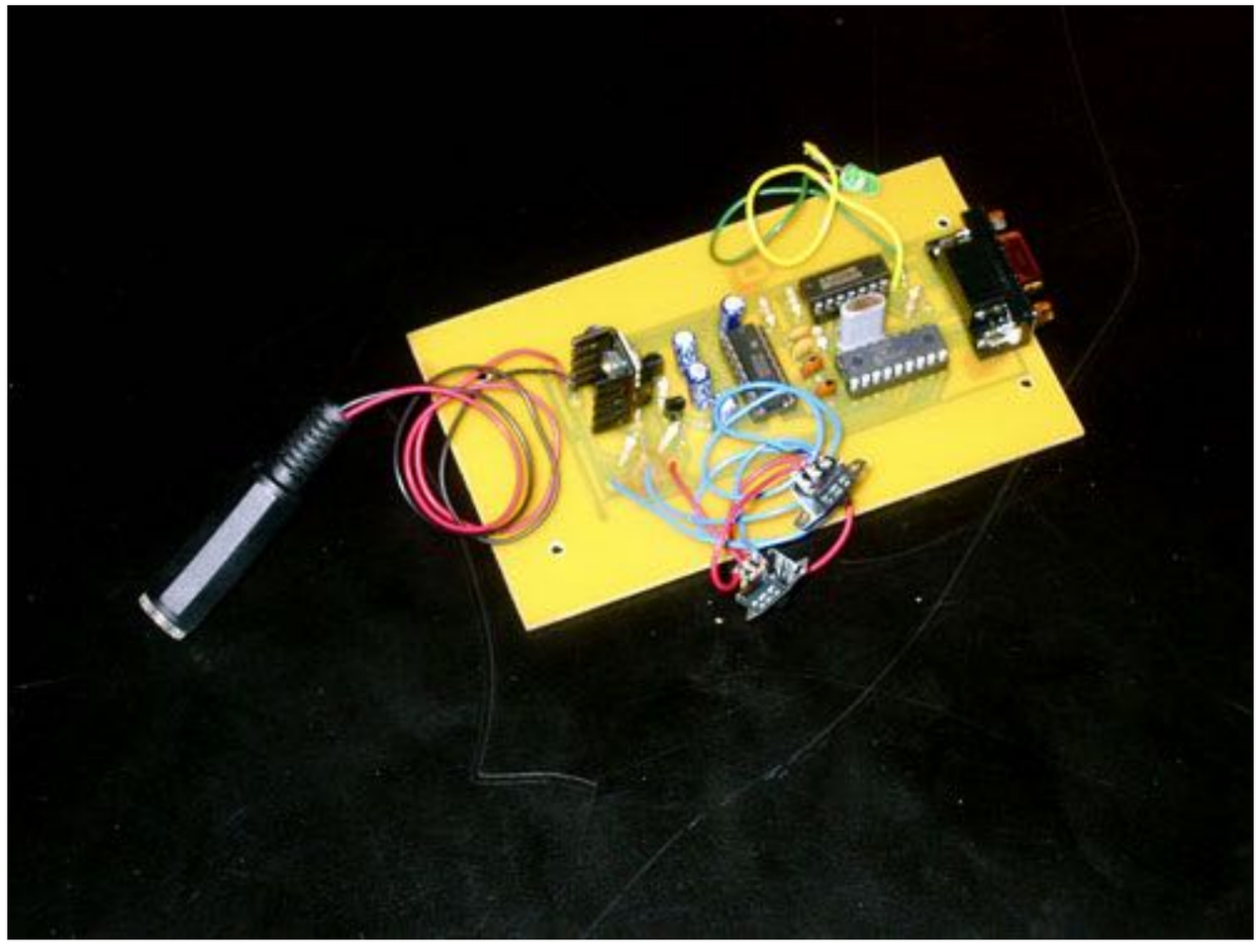
- 1994 : Netscape
- 2007: Windows XP
- 2008: Debian/OpenSSL
- 2010: Playstation 3
- 2013: Android

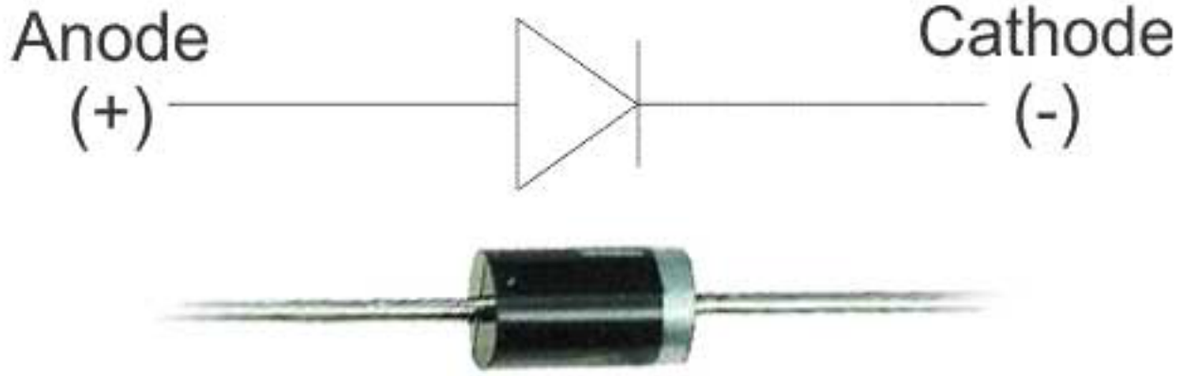
Dual_EC_DRBG

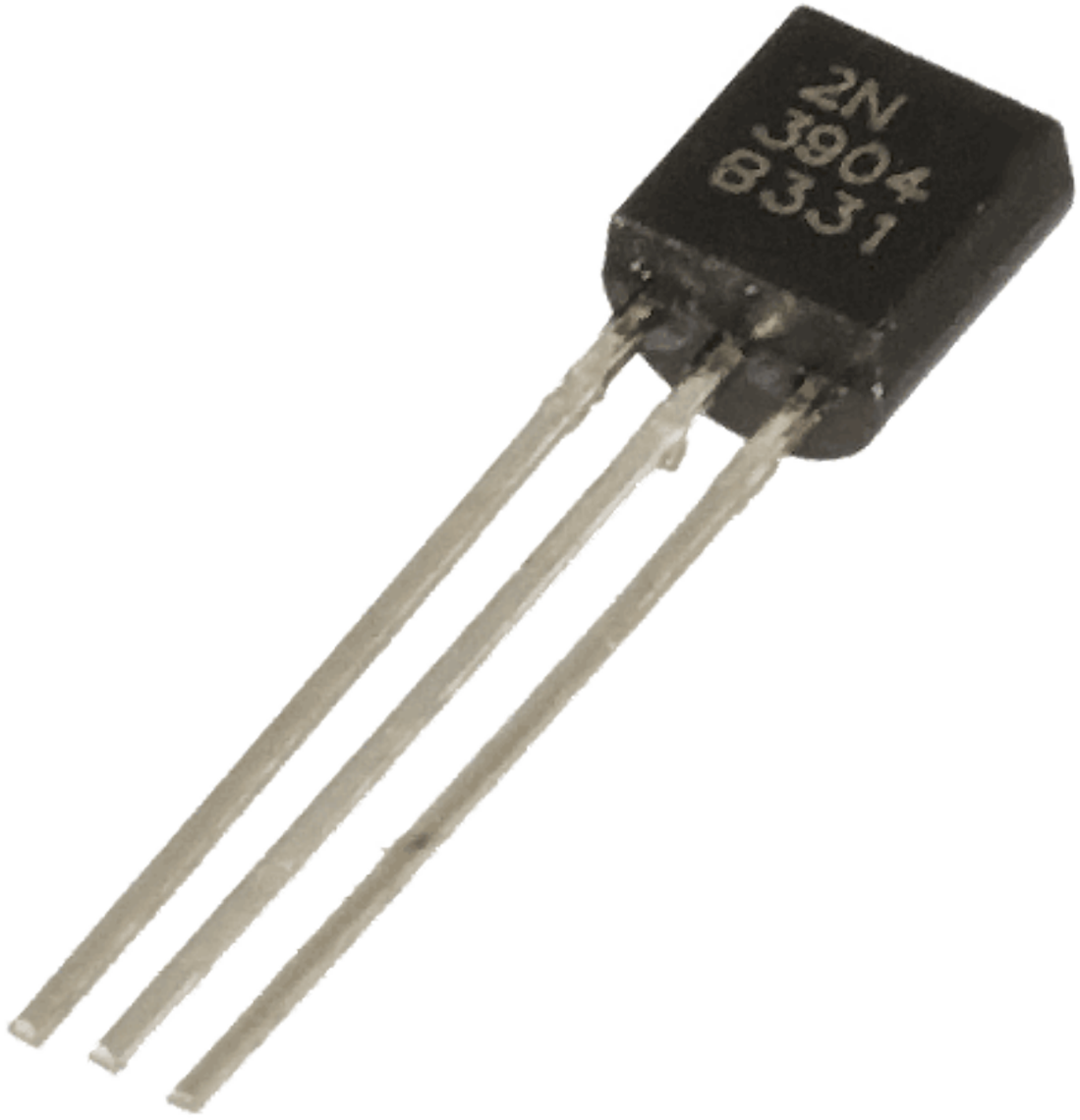

IDQ
SWISS QUANTUM

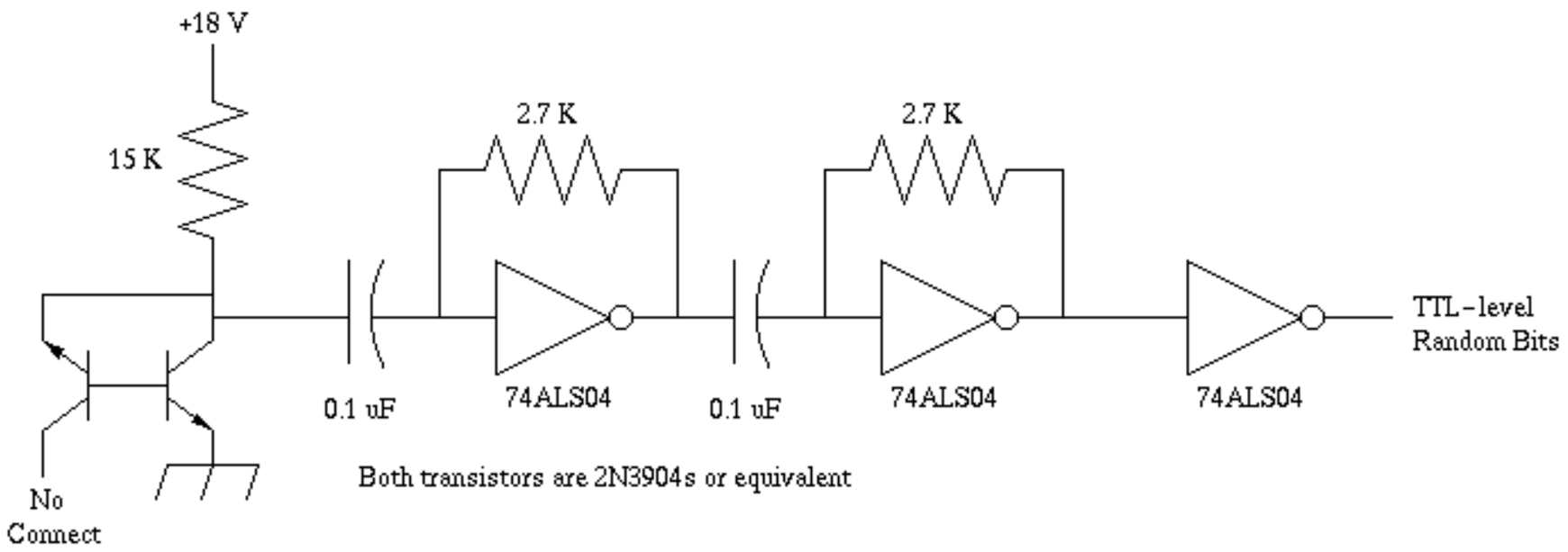
>$1000?

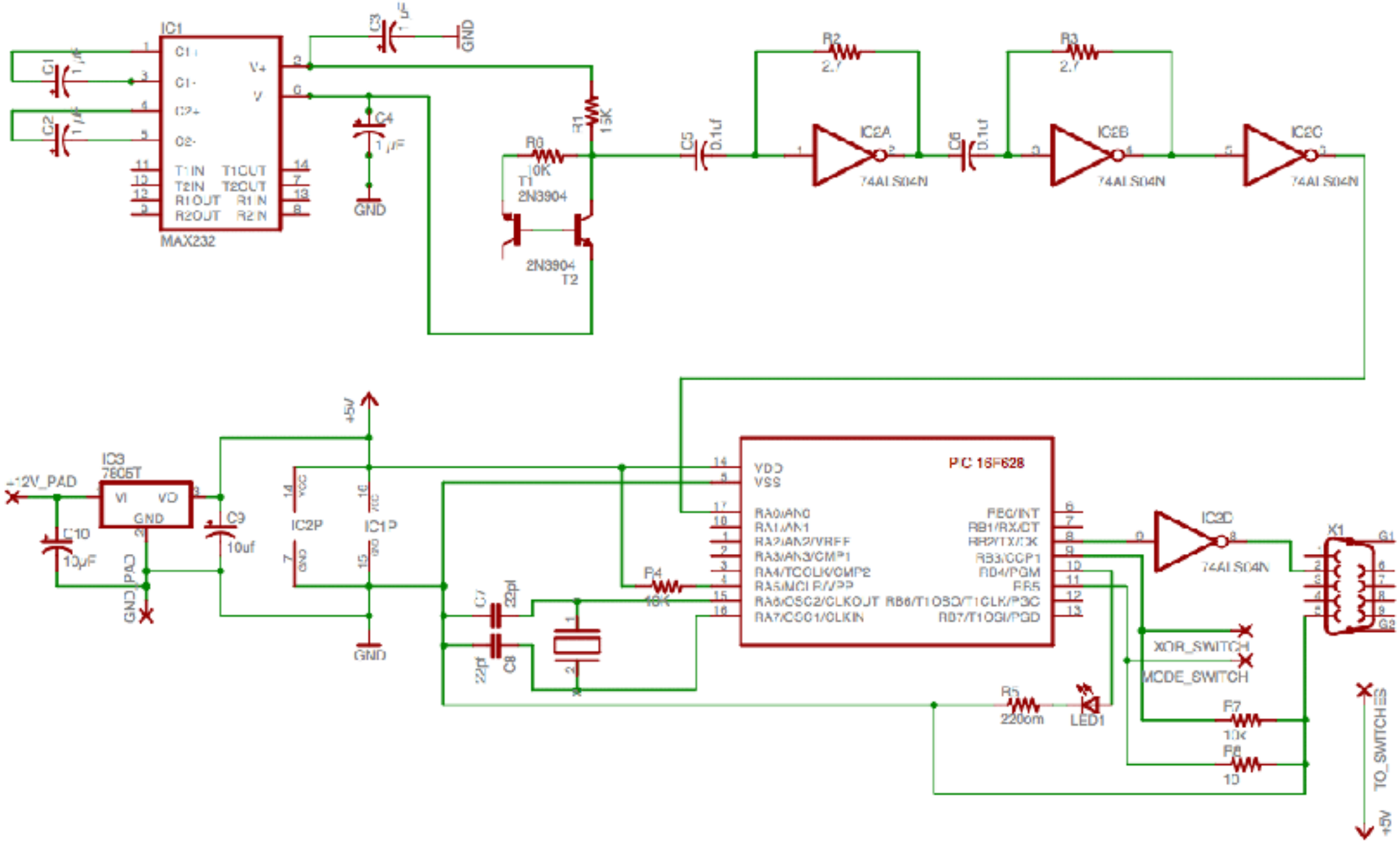
2005




Anode
(+)
Cathode
(-)


2N
3904
8331


+18 V
15 K
2.7 K
2.7 K
0.1 uF
74ALS04
0.1 uF
74ALS04
74ALS04
TTL-level
Random Bits
No
Connect
Both transistors are 2N3904s or equivalent



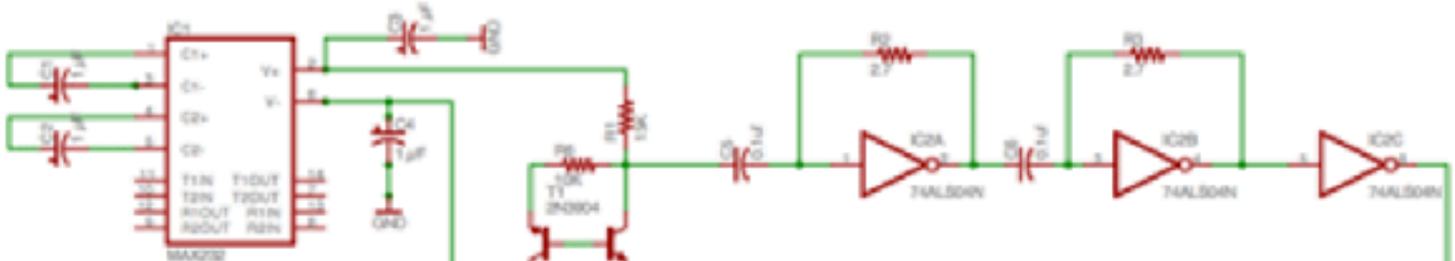
Build your own True Random Number Generator

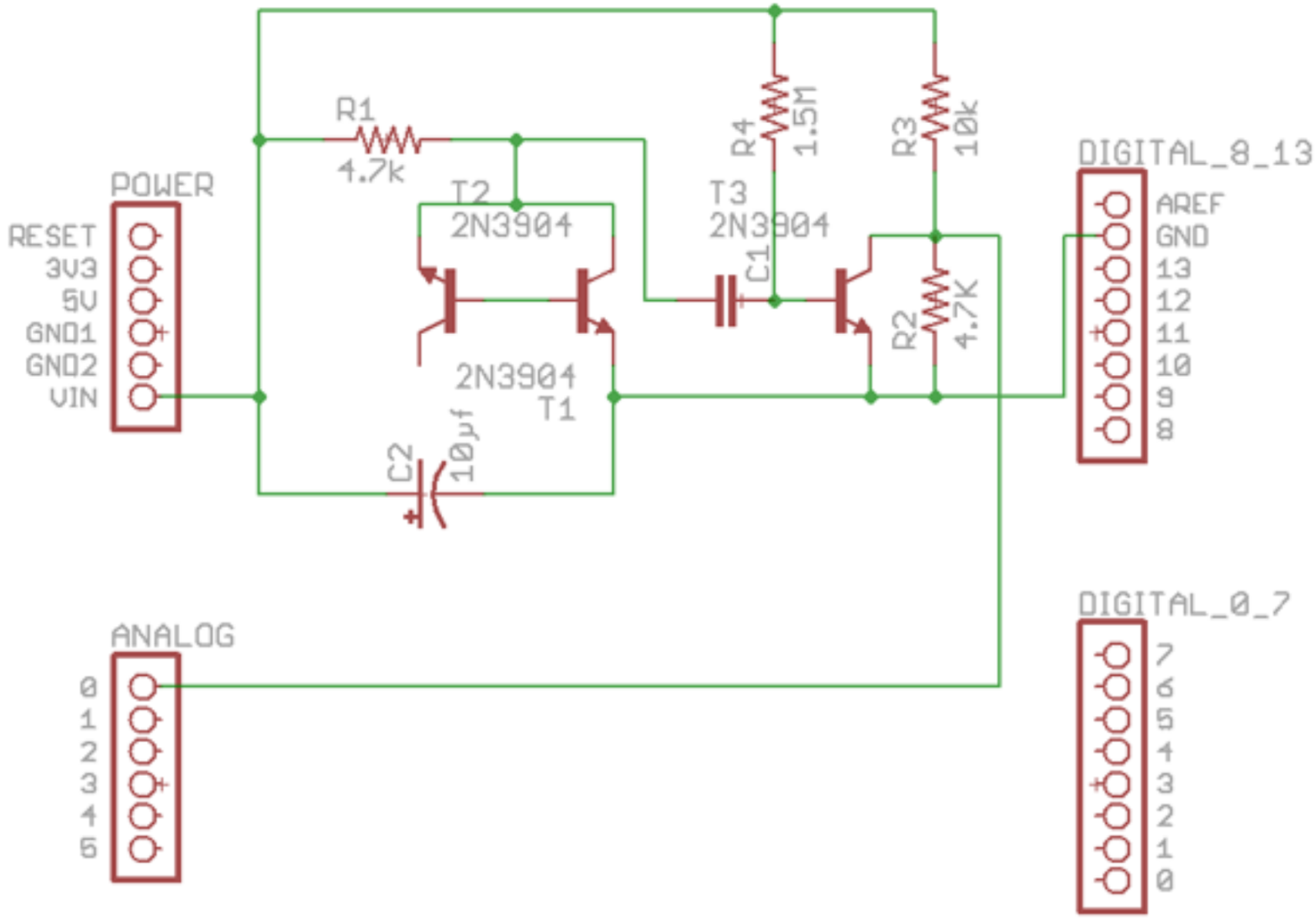
Intro

This generator uses avalanche noise, and is based on a design by Will Ware. Herein are instructions as to how you can use a pic chip to analyze a noise source and output random data serially. I've included circuit diagrams as well as links to instructions for fabricating your own board.

There are two types of random numbers: true and pseudo. Pseudo random numbers are created by an algorithm. The problem with this is that if someone knows what algorithm you use, it is theoretically possible predict what numbers you will create. True random number generators create sequences that are impossible to predict. They use random physical phenomenon as their source or randomness.

How it Works




R1
4.7k
T2
2N3904
R4
1.5M
T3
2N3904
C1
R3
10k
R2
4.7K
2N3904
T1
C2
10µf
POWER
RESET
3V3
5V
GND1
GND2
VIN
DIGITAL_8_13
AREF
GND
13
12
11
10
9
8
ANALOG
0
1
2
3
4
5
DIGITAL_0_7
7
6
5
4
3
2
1
0


10 mV
0.5 ms 1+

Build your own True Random Number Generator

ATTENTION: VERSION 2 IS OUT.

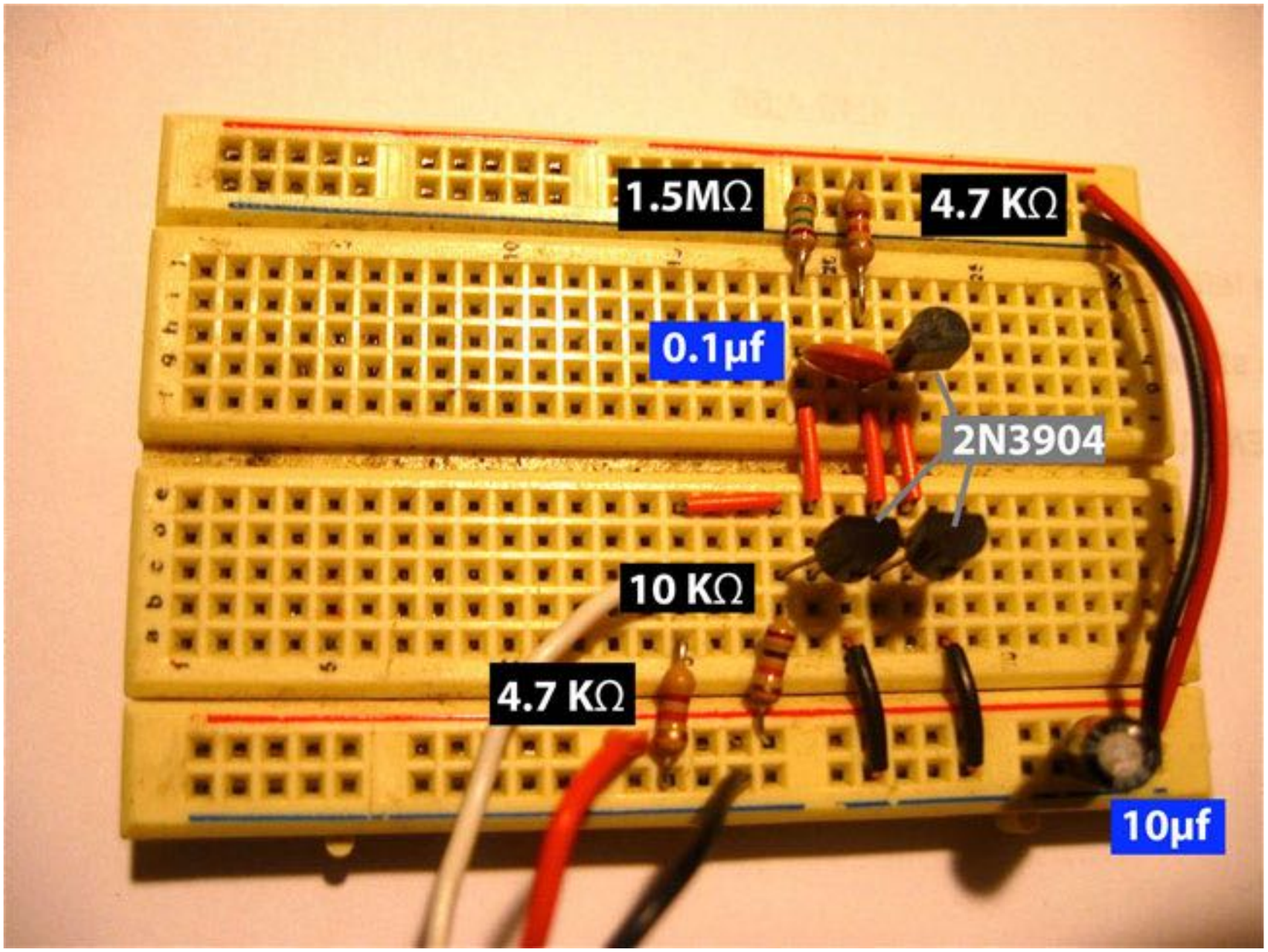
Below is version 1

Intro

This generator uses avalanche noise, and is based on a design by Will Ware. Herein are instructions as to how you can use a pic chip to analyze a noise source and output random data serially. I've included circuit diagrams as well as links to instructions for fabricating your own board.

There are two types of random numbers: true and pseudo. Pseudo random numbers are created by an algorithm. The problem with this is that if someone knows what algorithm you use, it is theoretically possible predict what numbers you will create. True random number generators create sequences that are impossible to predict. They use random physical phenomenon as their source or randomness.

How it Works


1.5MΩ
4.7 KΩ
0.1µf
2N3904
10 KΩ
4.7 KΩ
10µf

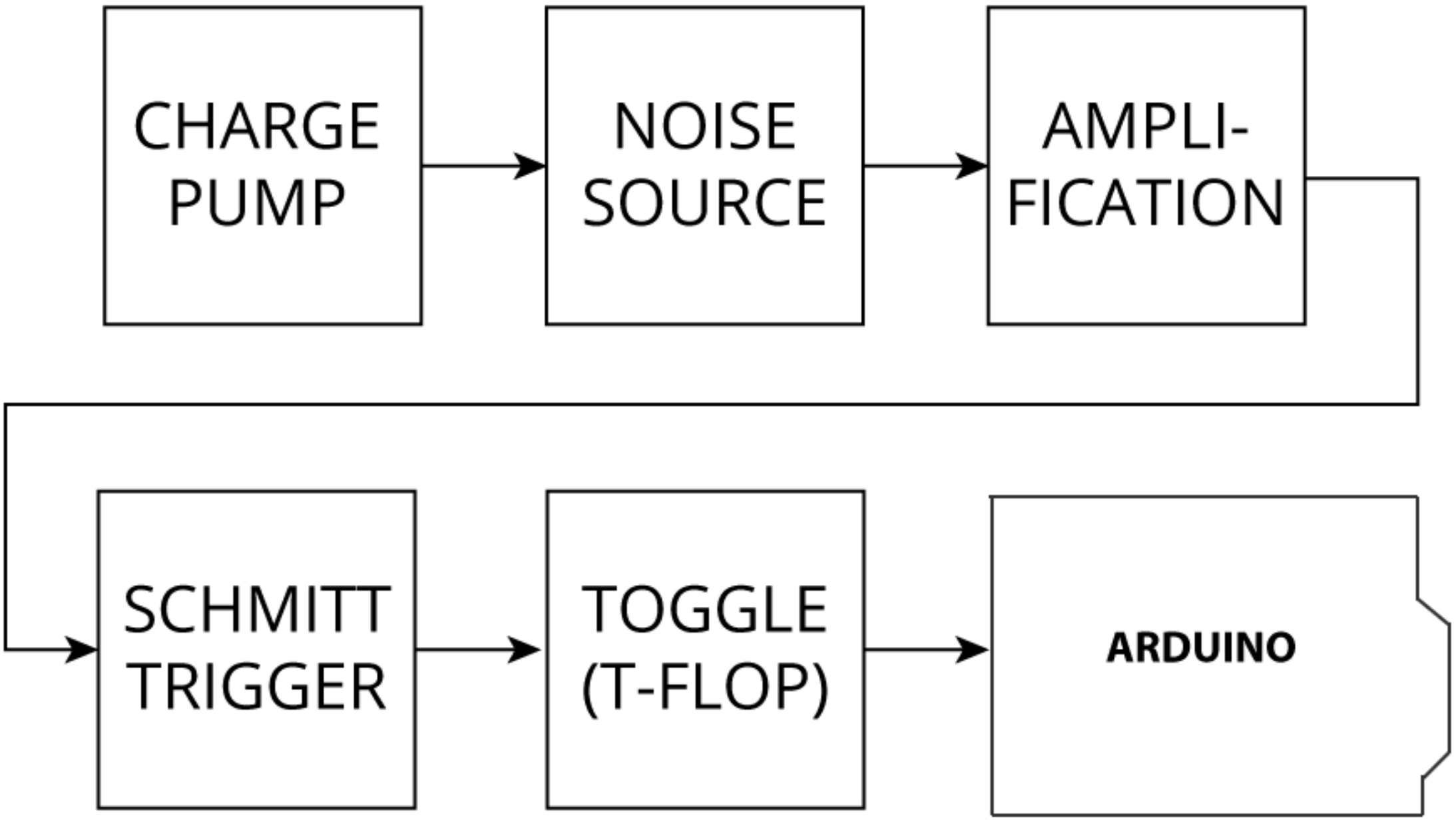
2013

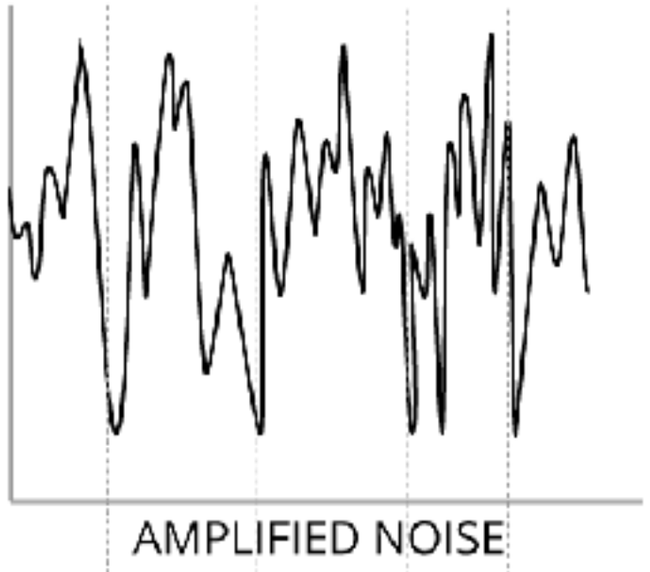
Goals

- ~~Make money~~
- ~~Lose money~~
- Spread knowledge

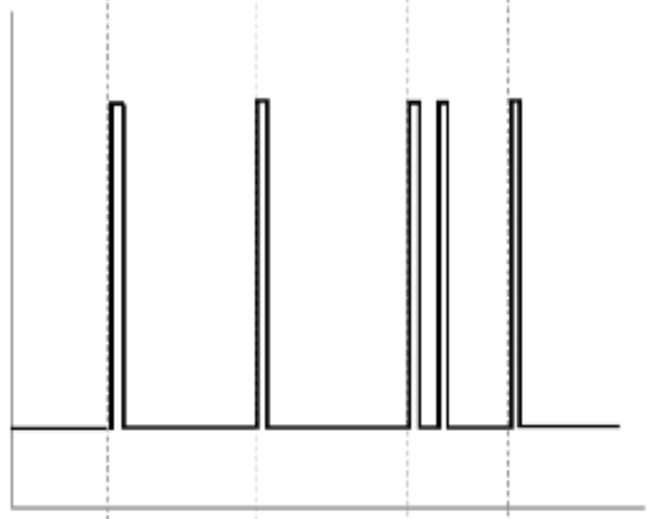
GOOD?

1. Random
2. Reliable
3. Fast
4. Cheap

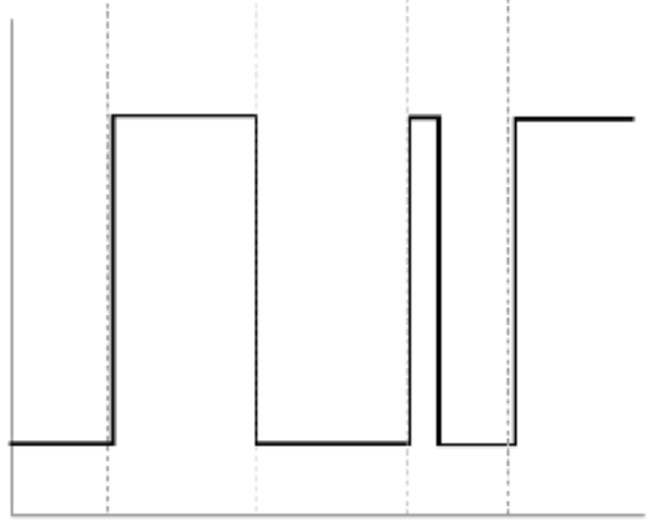

CHARGE PUMP
NOISE SOURCE
AMPLI-FICATION
SCHMITT TRIGGER
TOGGLE (T-FLOP)
ARDUINO



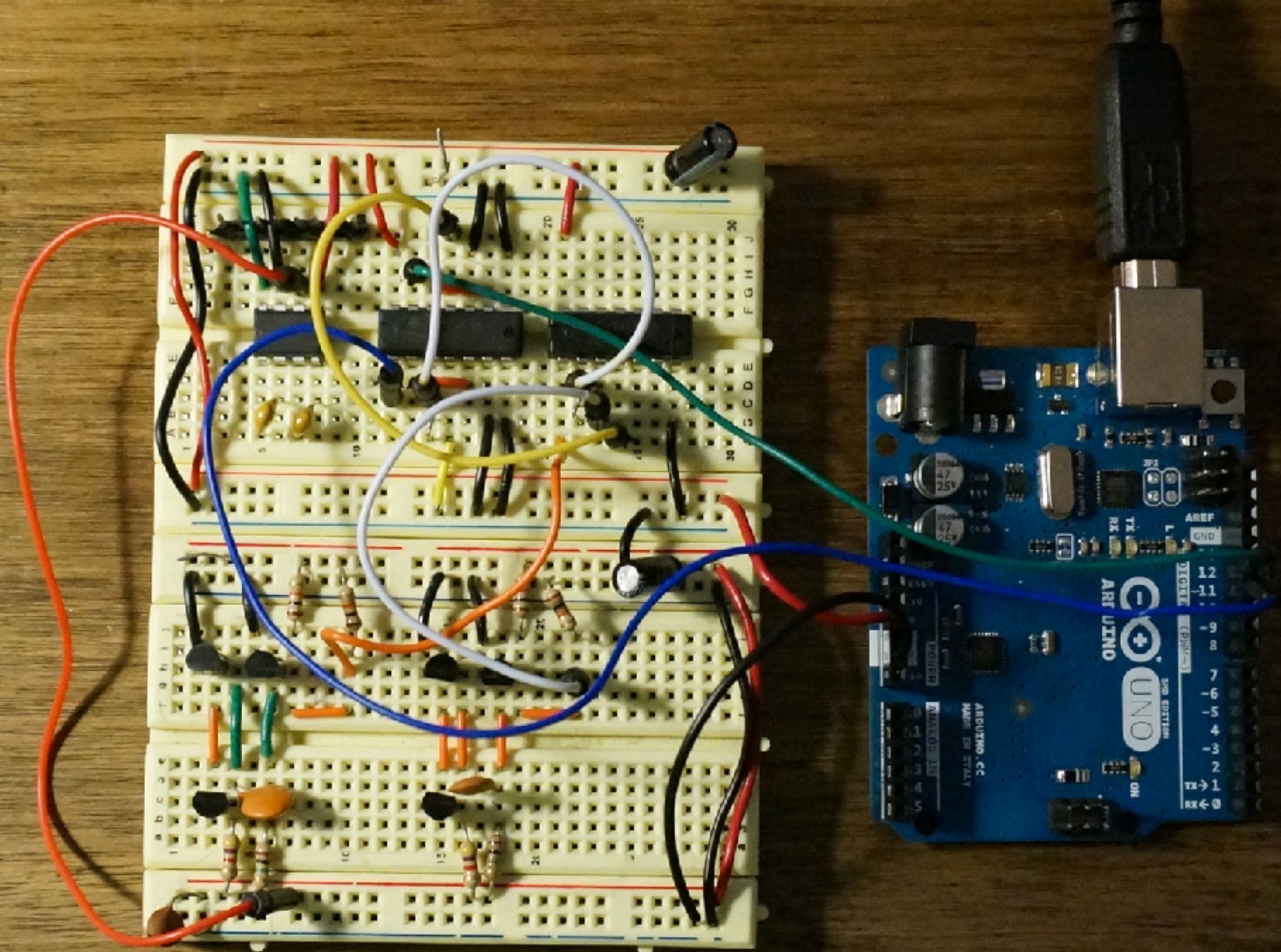
AMPLIFIED NOISE

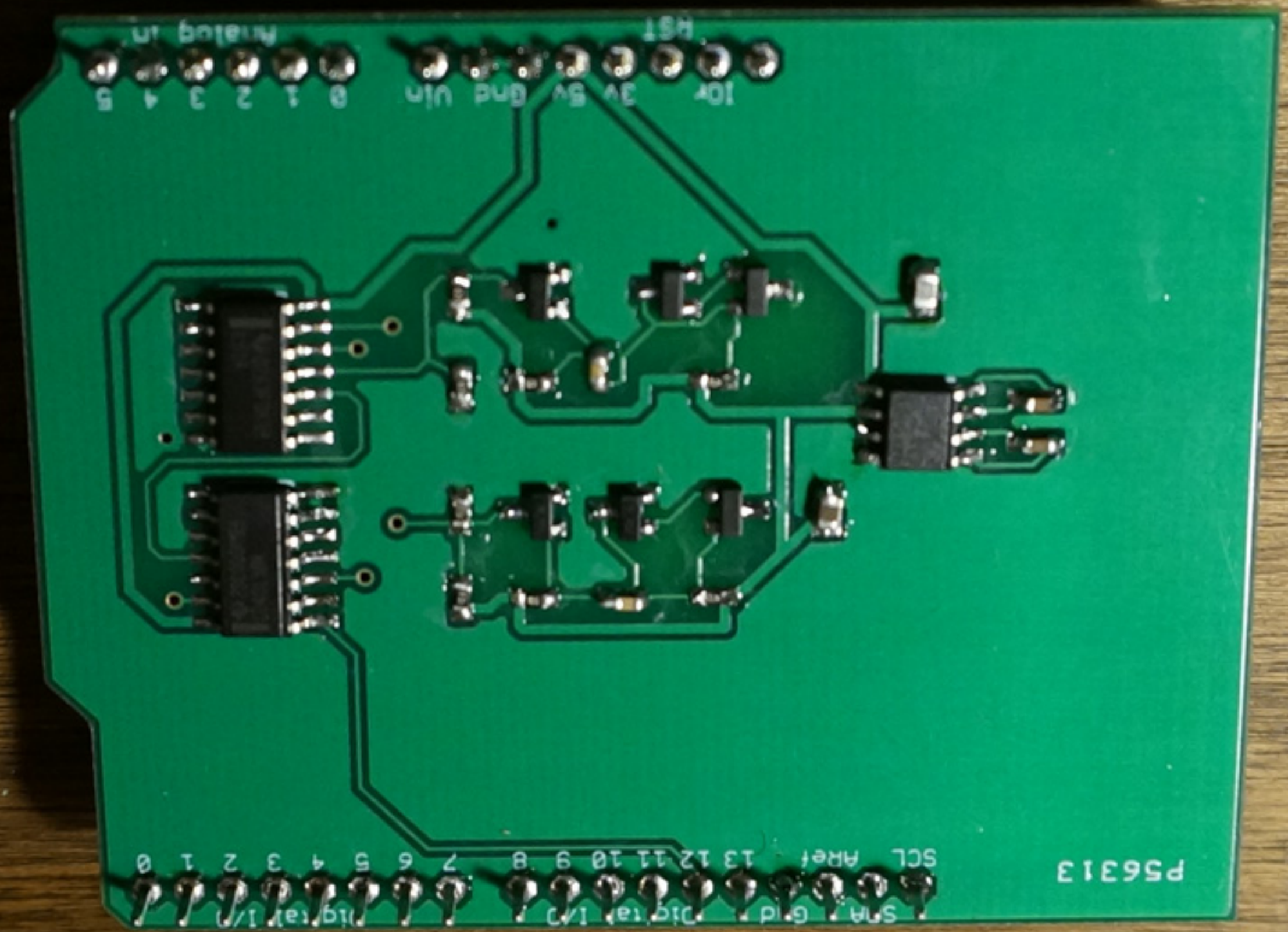


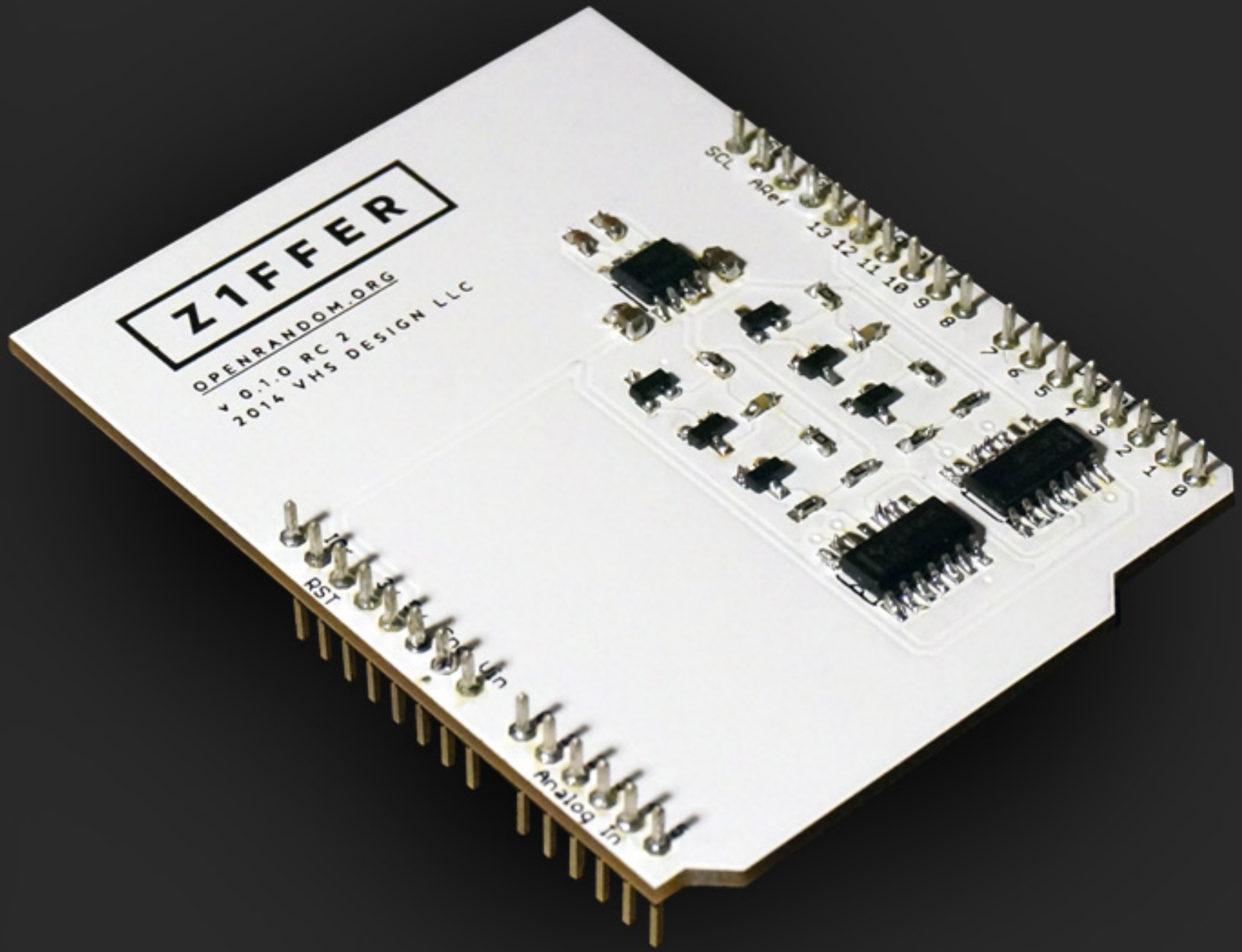
SCHMITT TRIGGER



TOGGLE (T-FLOP)


ARDUINO
UNO
ARDUINO.CC
MADE IN ITALY
DIGITAL (PWM~)
ANALOG IN
POWER
AREF
GND
ON


Analog in
0 1 2 3 4 5
RST
IOR 3V 5V Gnd Vin
SCL Aref 13 12 11 10 9 8 7 6 5 4 3 2 1 0
P56313


Z1FFER
OPENRANDOM.ORG
v 0.1.0 RC 2
2014 VHS DESIGN LLC
SCL
AREF
13 12 11 10 9 8 7 6 5 4 3 2 1 0
RST
Analog In

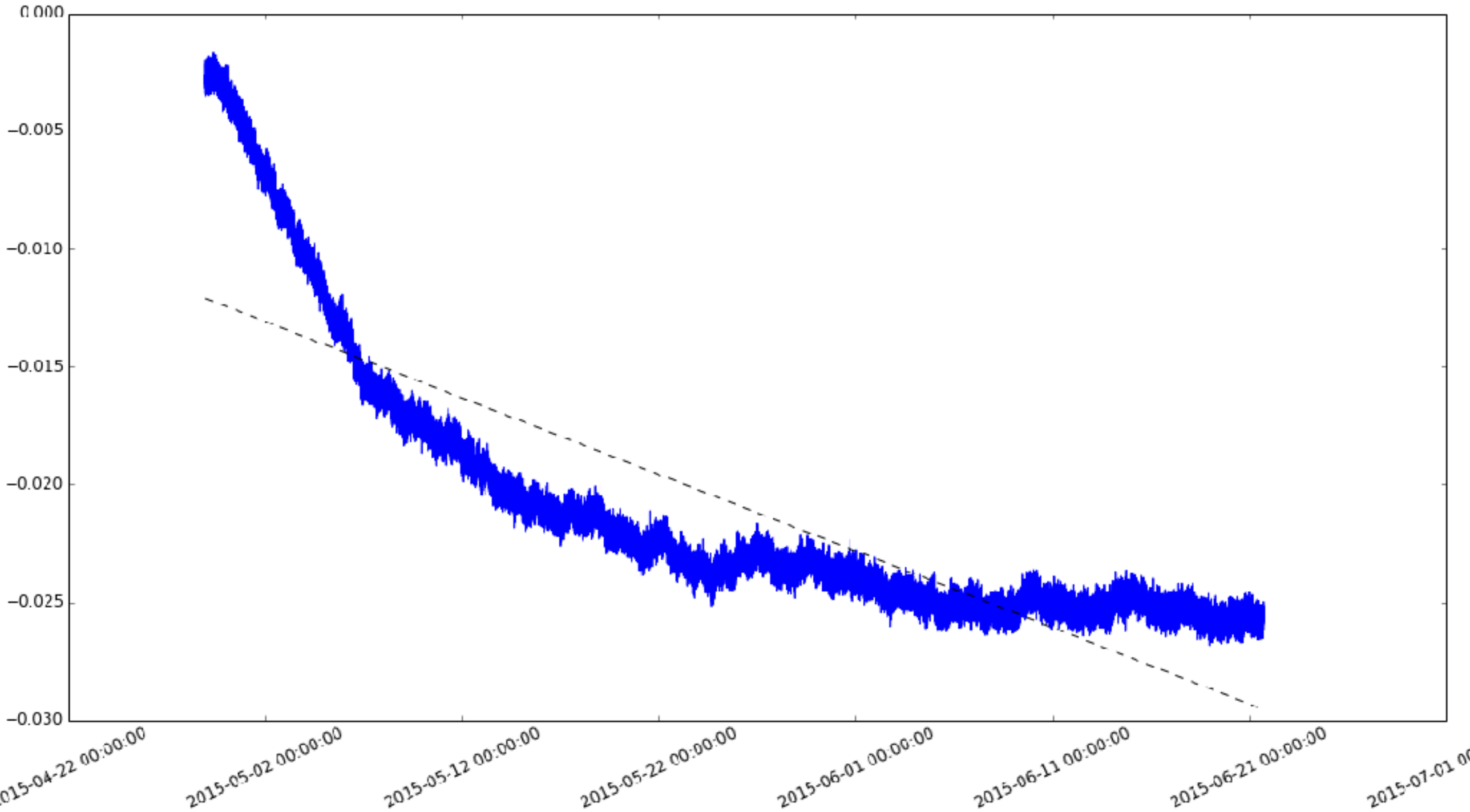
Z1FFER

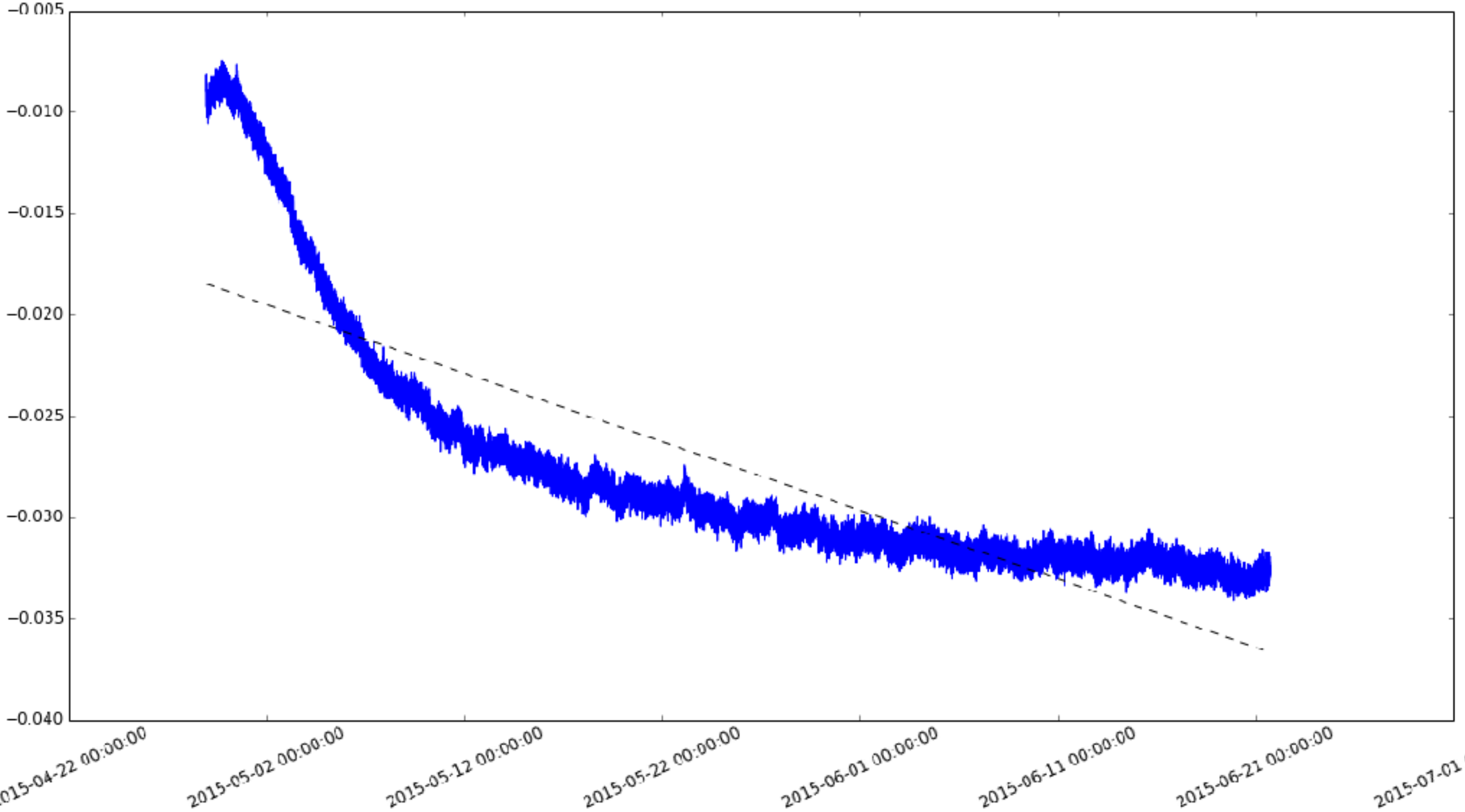
OPENRANDOM.ORG

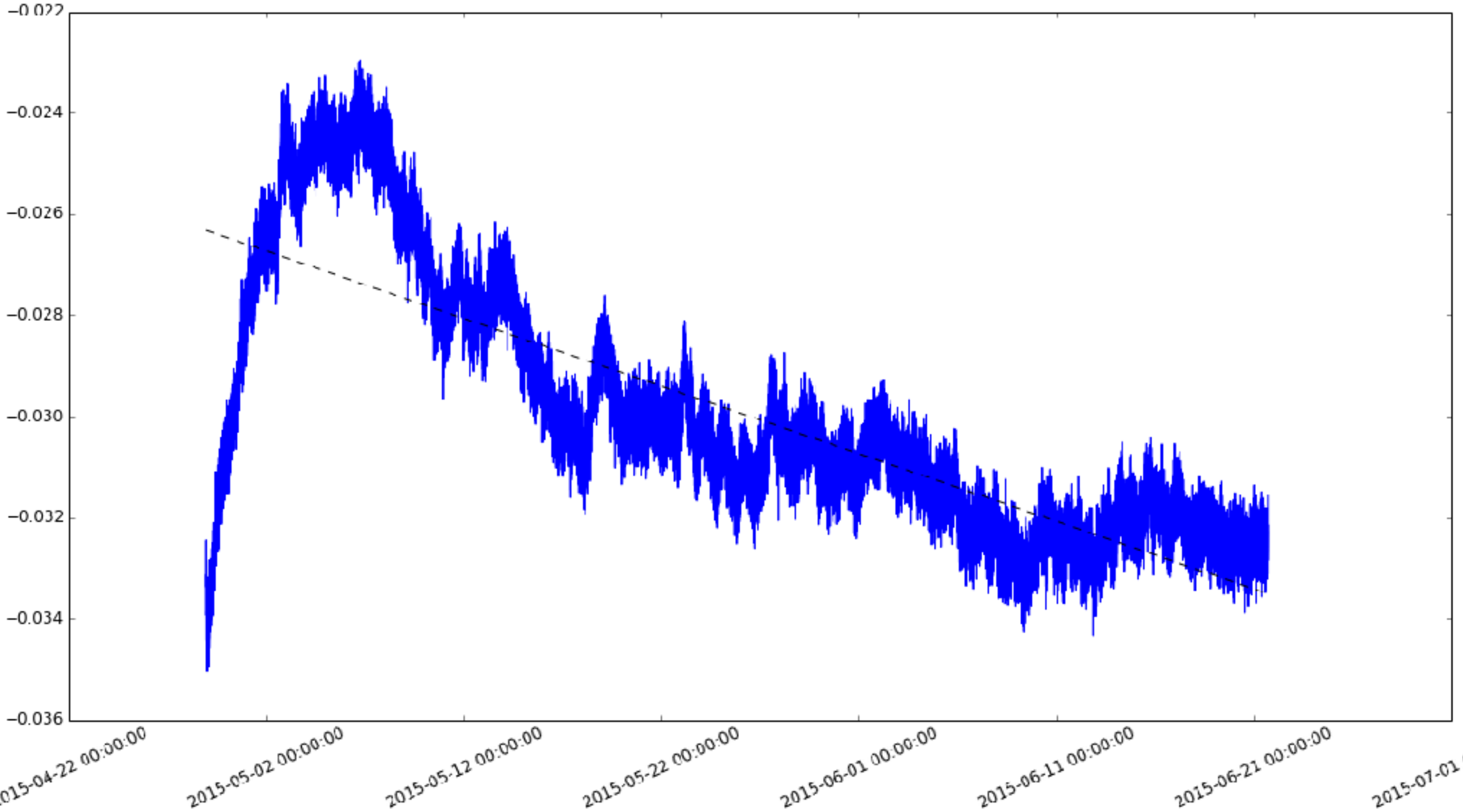
v 0.2.0 RC 1

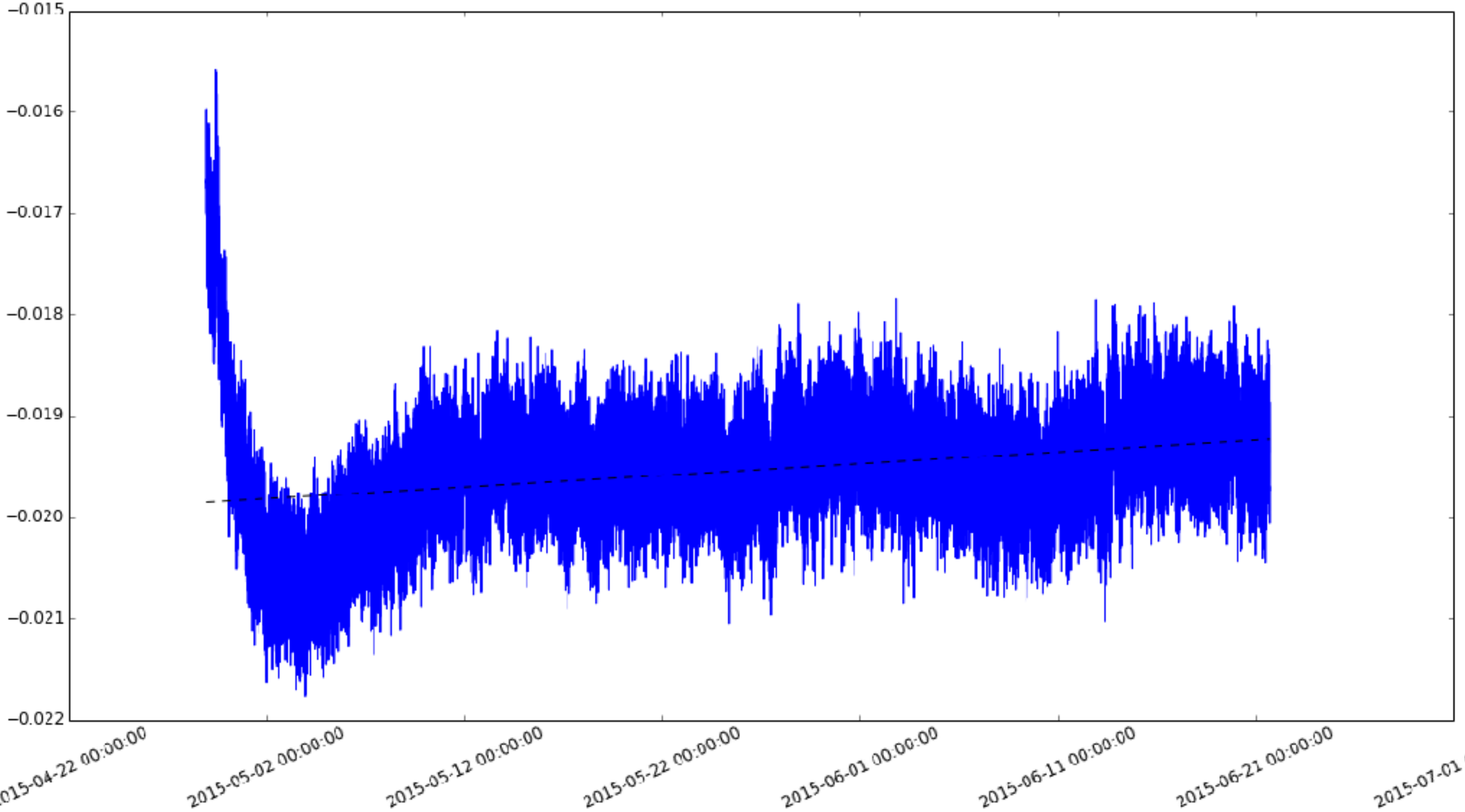
2014 VMS DESIGN LLC

Long term tests

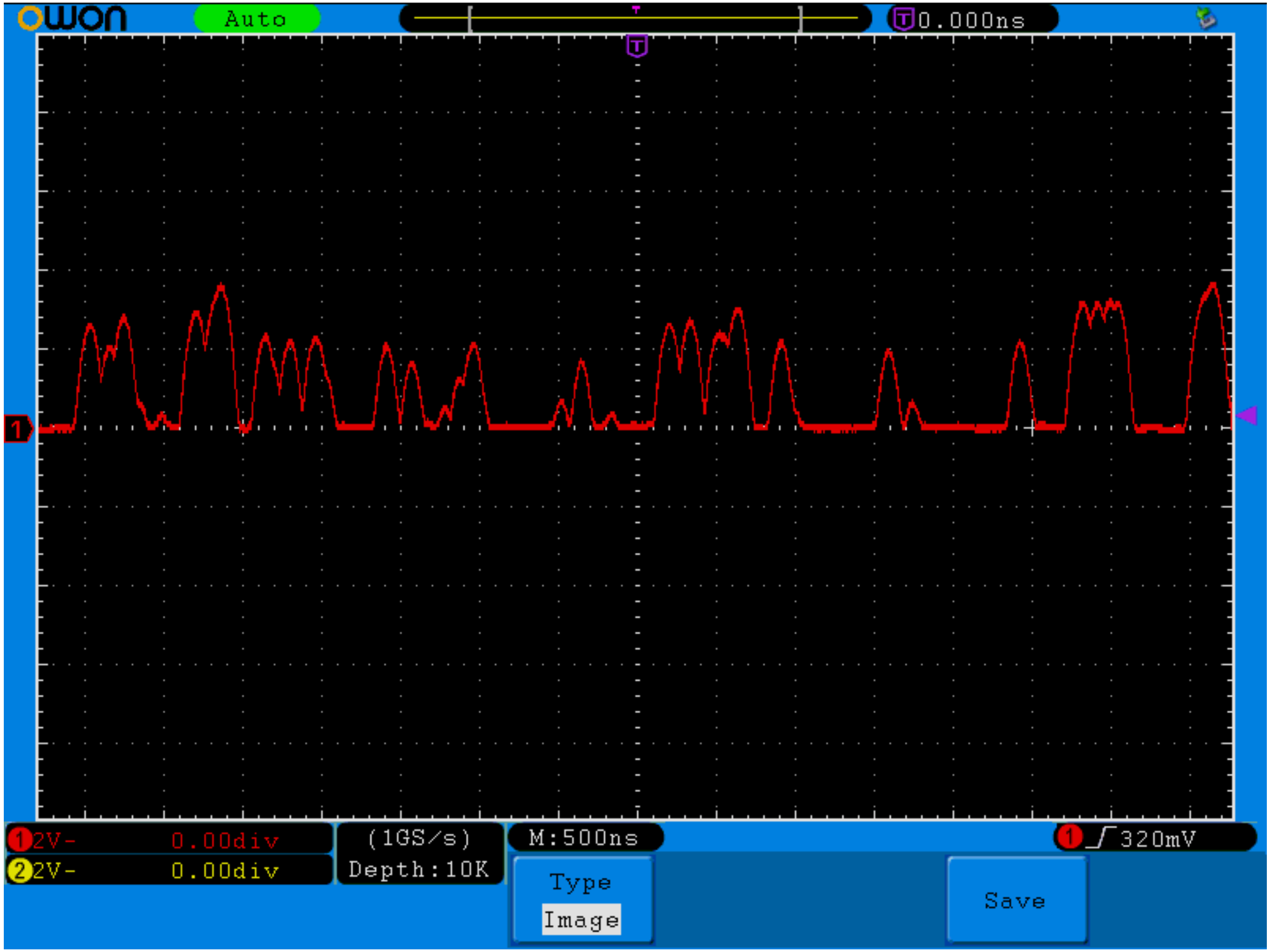

0.000
−0.005
−0.010
−0.015
−0.020
−0.025
−0.030
2015-04-22 00:00:00
2015-05-02 00:00:00
2015-05-12 00:00:00
2015-05-22 00:00:00
2015-06-01 00:00:00
2015-06-11 00:00:00
2015-06-21 00:00:00

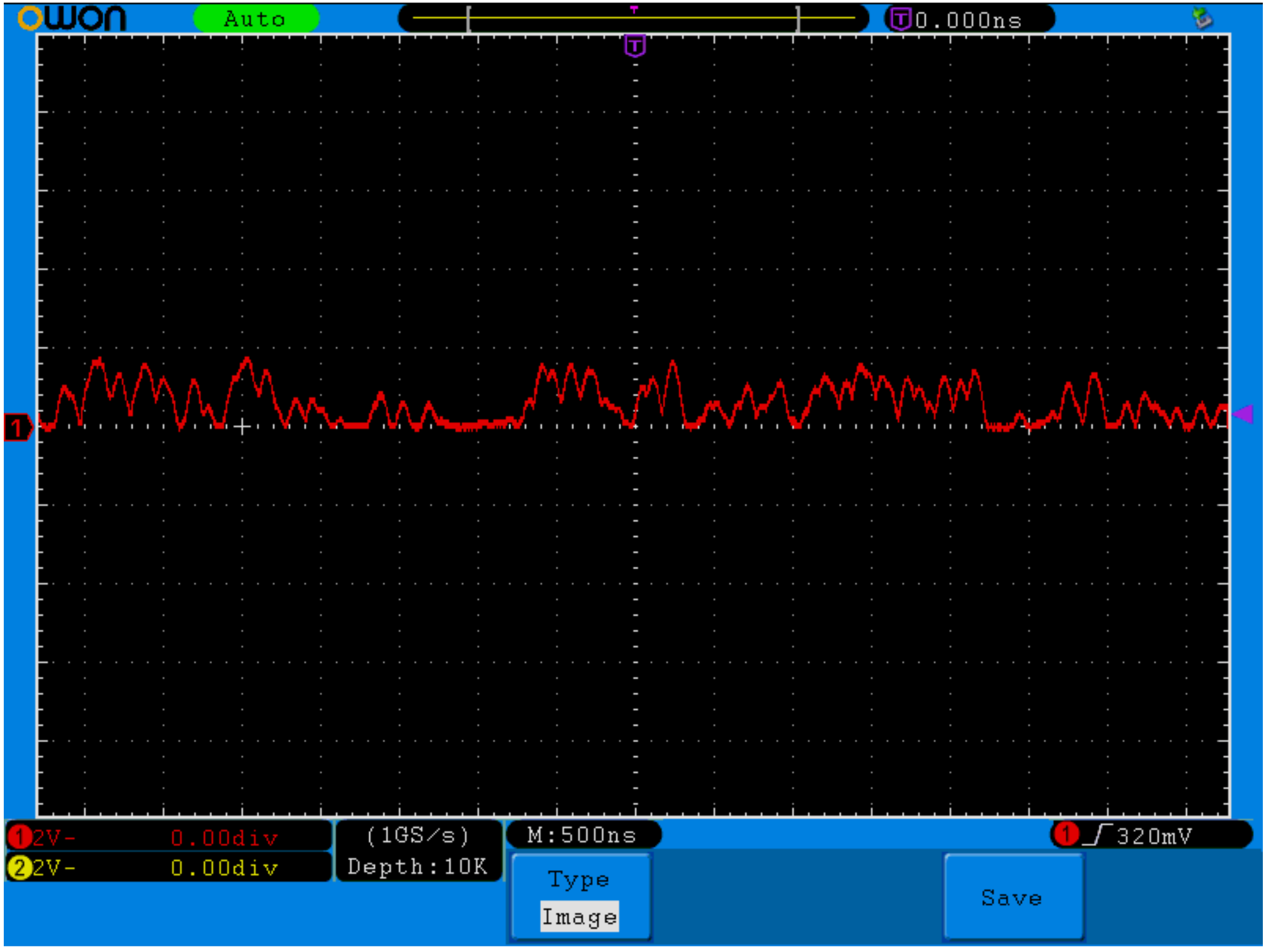

−0.005
−0.010
−0.015
−0.020
−0.025
−0.030
−0.035
−0.040
2015-04-22 00:00:00
2015-05-02 00:00:00
2015-05-12 00:00:00
2015-05-22 00:00:00
2015-06-01 00:00:00
2015-06-11 00:00:00
2015-06-21 00:00:00


−0.022
−0.024
−0.026
−0.028
−0.030
−0.032
−0.034
−0.036
2015-04-22 00:00:00
2015-05-02 00:00:00
2015-05-12 00:00:00
2015-05-22 00:00:00
2015-06-01 00:00:00
2015-06-11 00:00:00
2015-06-21 00:00:00
2015-07-01 00

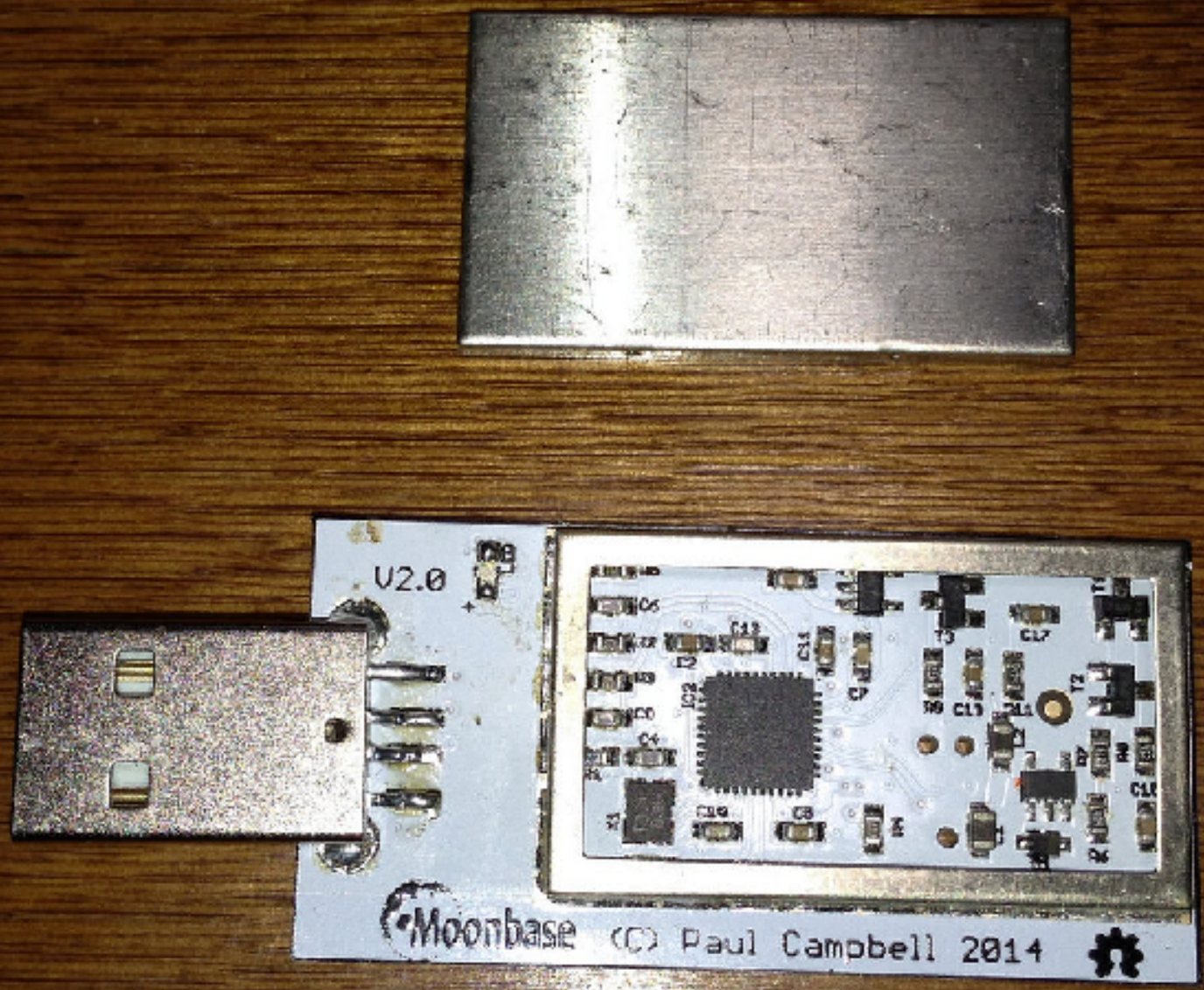

−0.015
−0.016
−0.017
−0.018
−0.019
−0.020
−0.021
−0.022
2015-04-22 00:00:00
2015-05-02 00:00:00
2015-05-12 00:00:00
2015-05-22 00:00:00
2015-06-01 00:00:00
2015-06-11 00:00:00
2015-06-21 00:00:00

Transistor Variability


OWON
Auto
0.000ns
1
1 2V- 0.00div
2 2V- 0.00div
(1GS/s)
Depth:10K
M:500ns
1 320mV
Type
Image
Save

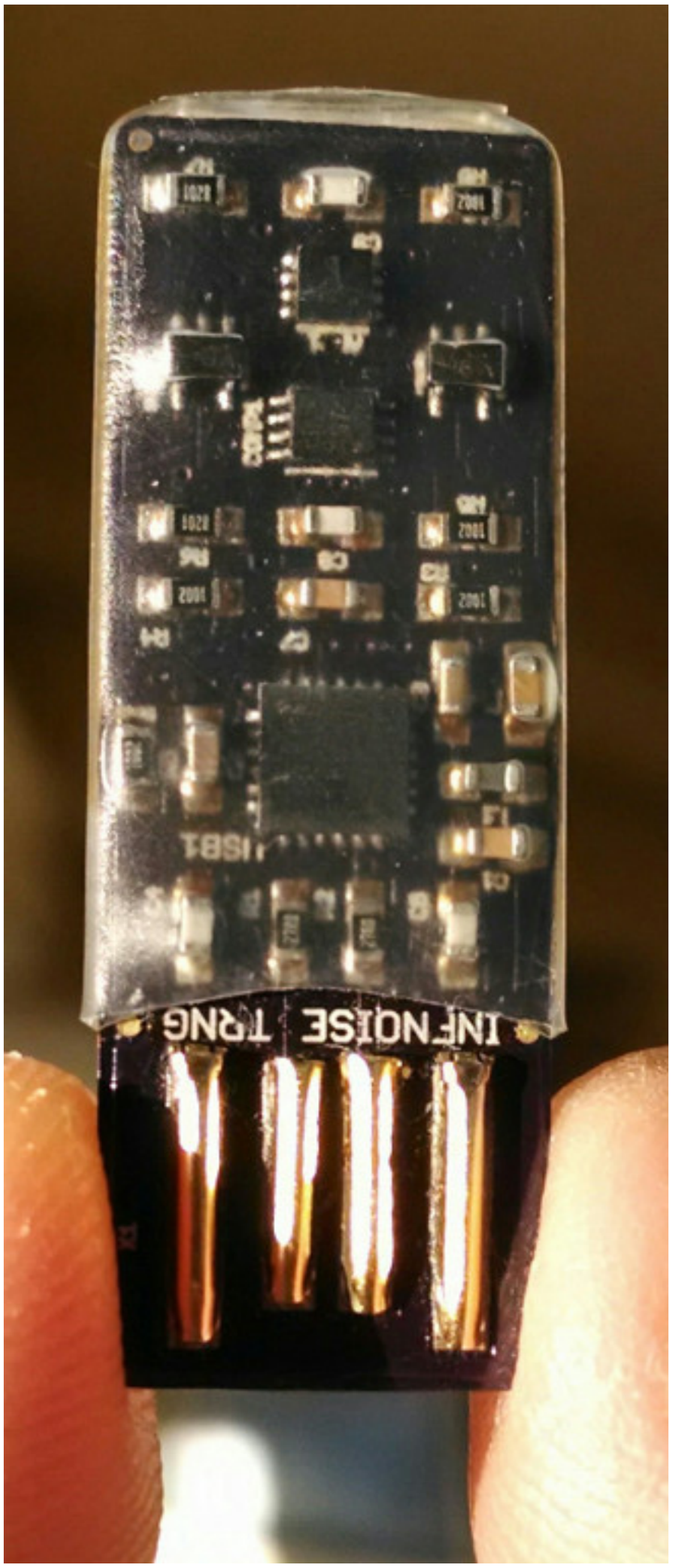

OWON
Auto
0.000ns
1
1 2V- 0.00div
2 2V- 0.00div
(1GS/s)
Depth:10K
M:500ns
1 320mV
Type
Image
Save

External Noise

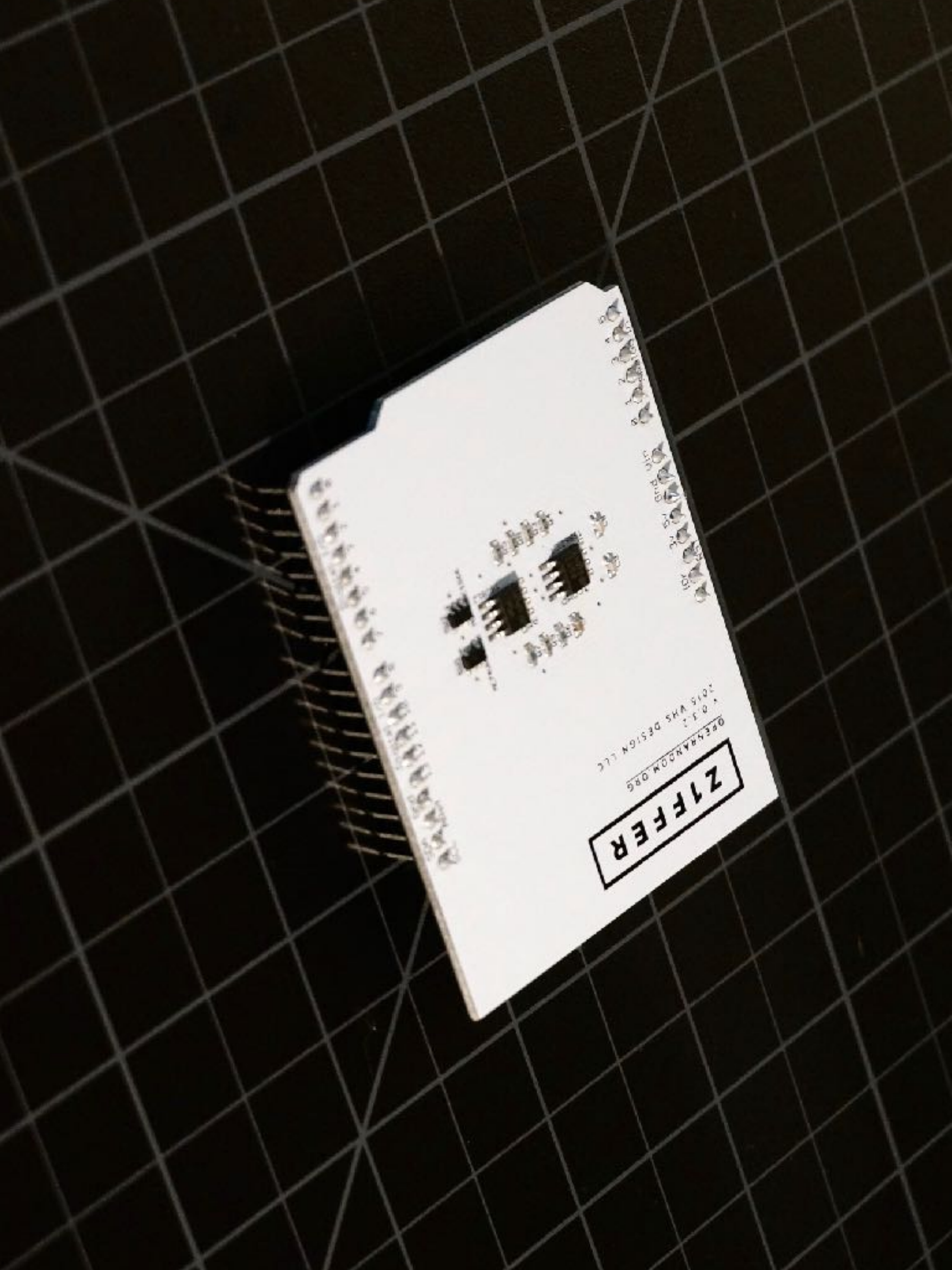

V2.0
Moonbase (C) Paul Campbell 2014

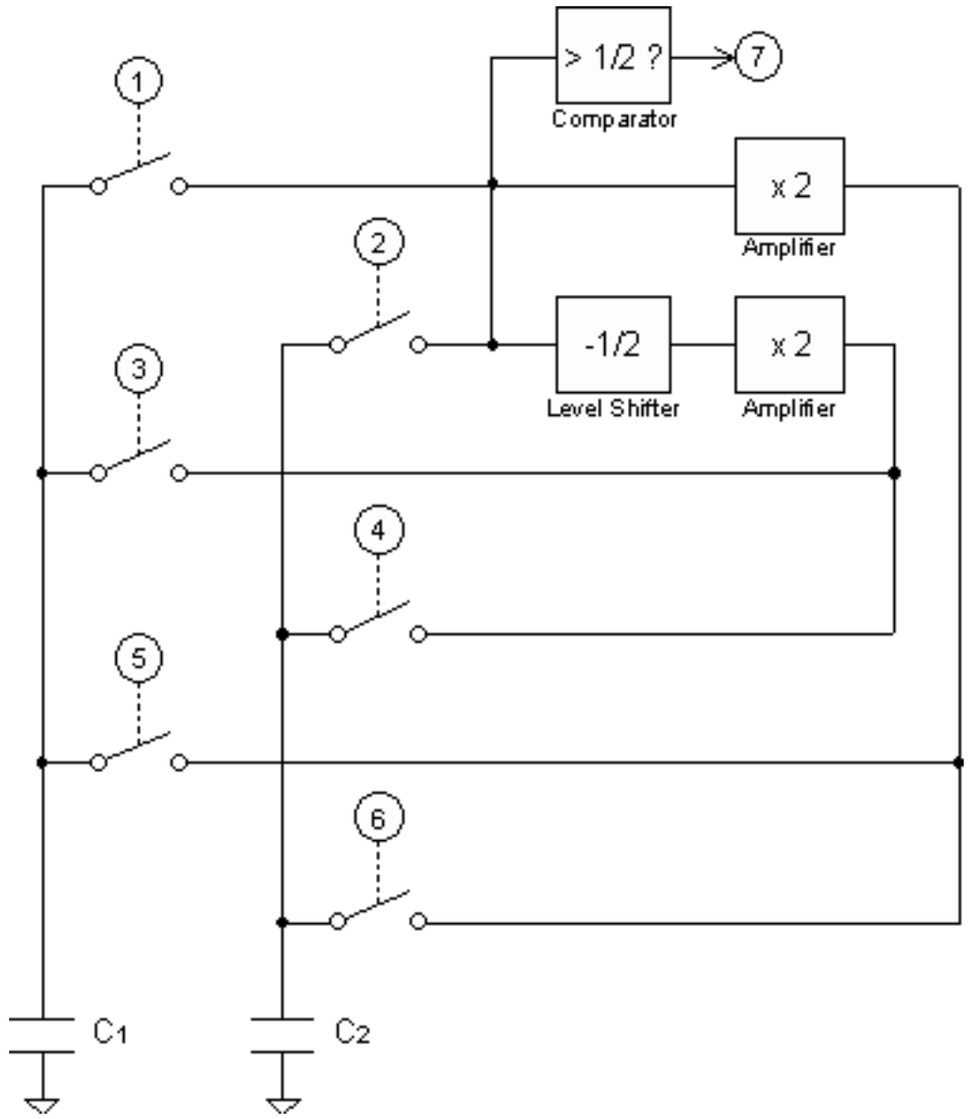
GOOD?

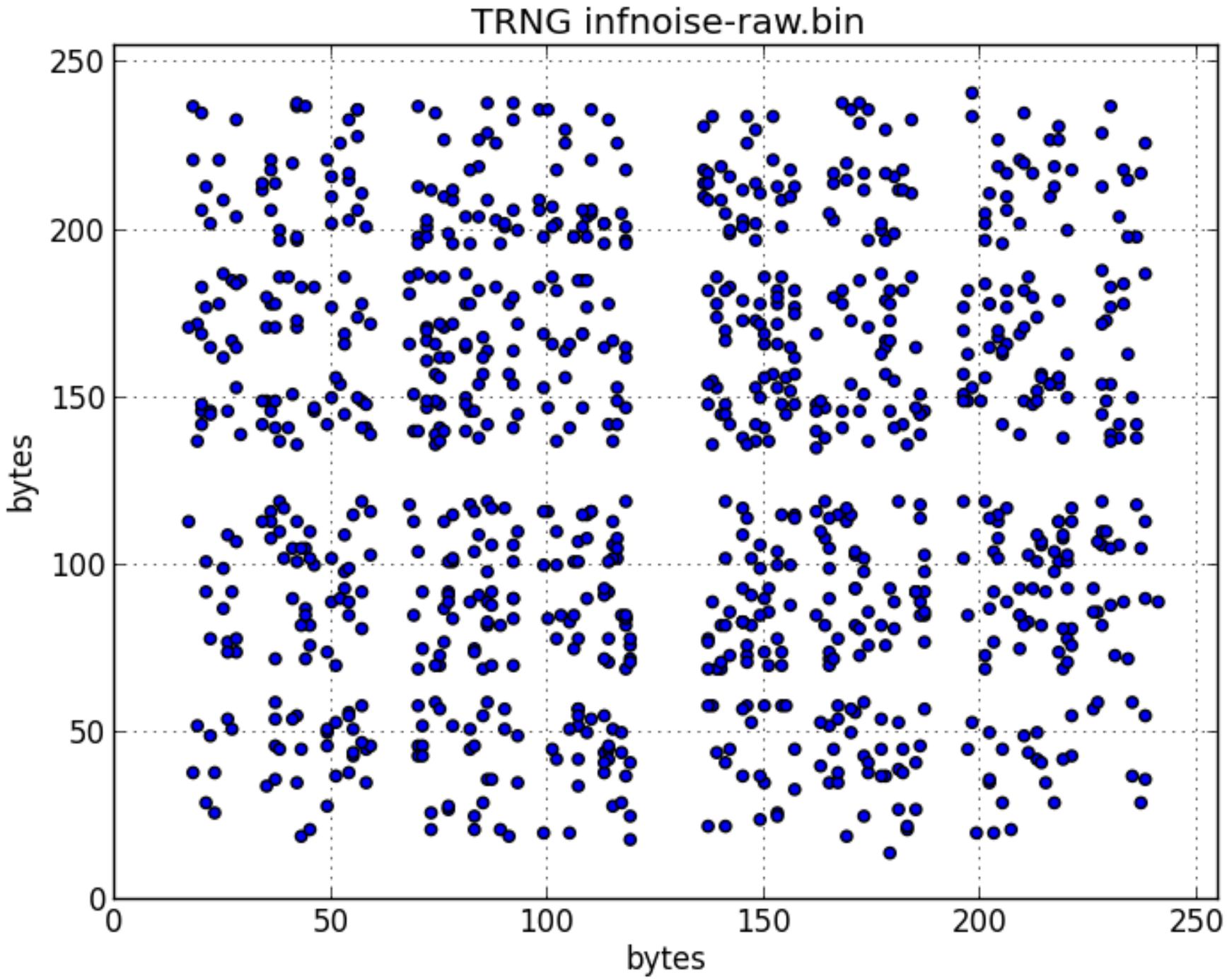
1. Random ✅
2. Reliable ??
3. Fast ✅
4. Cheap ✅


INFNOISE TRNG

“The easiest TRNG to get right”


ZIFFER
OPENRANDOM DRG
V 0.1.1
2015 VHS DESIGN LLC


> 1/2 ?
7
Comparator
1
x 2
Amplifier
2
-1/2
Level Shifter
x 2
Amplifier
3
4
5
6
C1
C2

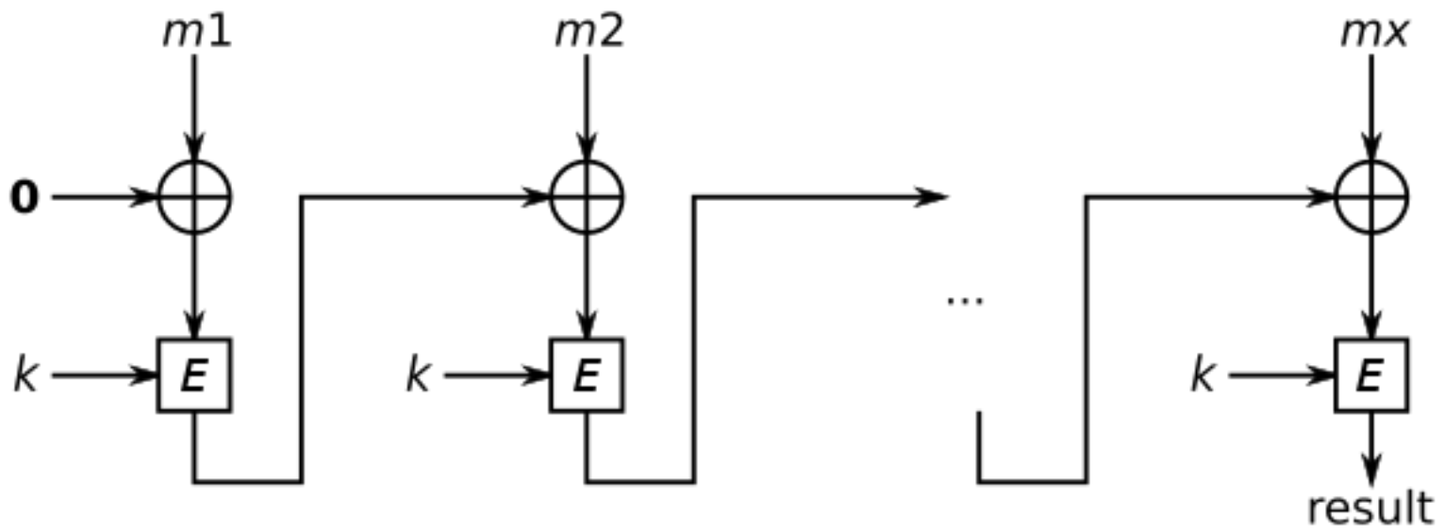

TRNG infnoise-raw.bin
250
200
150
100
50
0
bytes
0
50
100
150
200
250
bytes

Conditioning

NIST 800-90B

“Recommendation for the Entropy Sources Used for Random Bit Generation”

6.4.2.1.2 CBC-MAC Conditioning Function


m1
m2
mx
0
k
E
k
E
...
k
E
result

AES-CBC-MAC

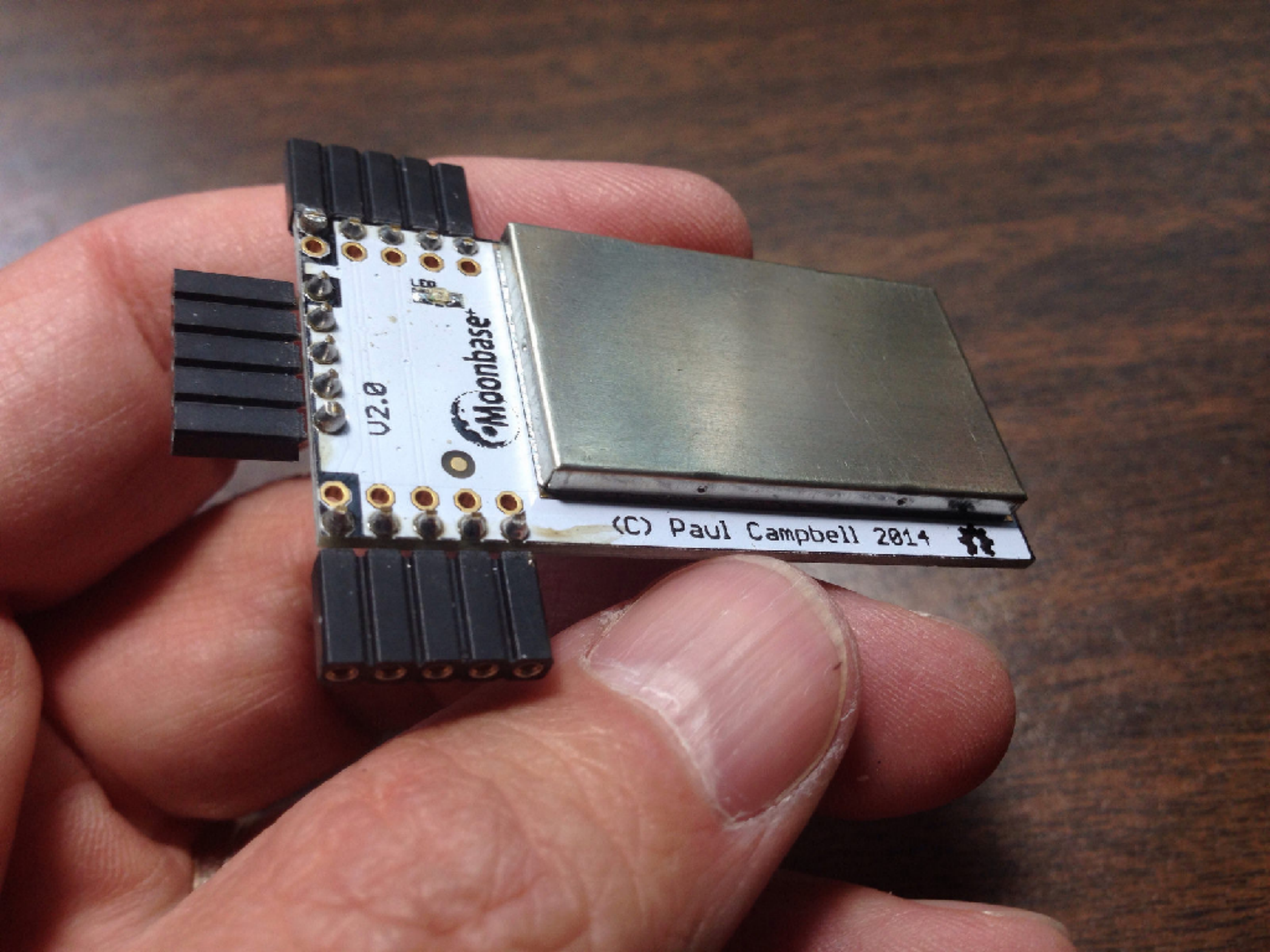
Manufacturing

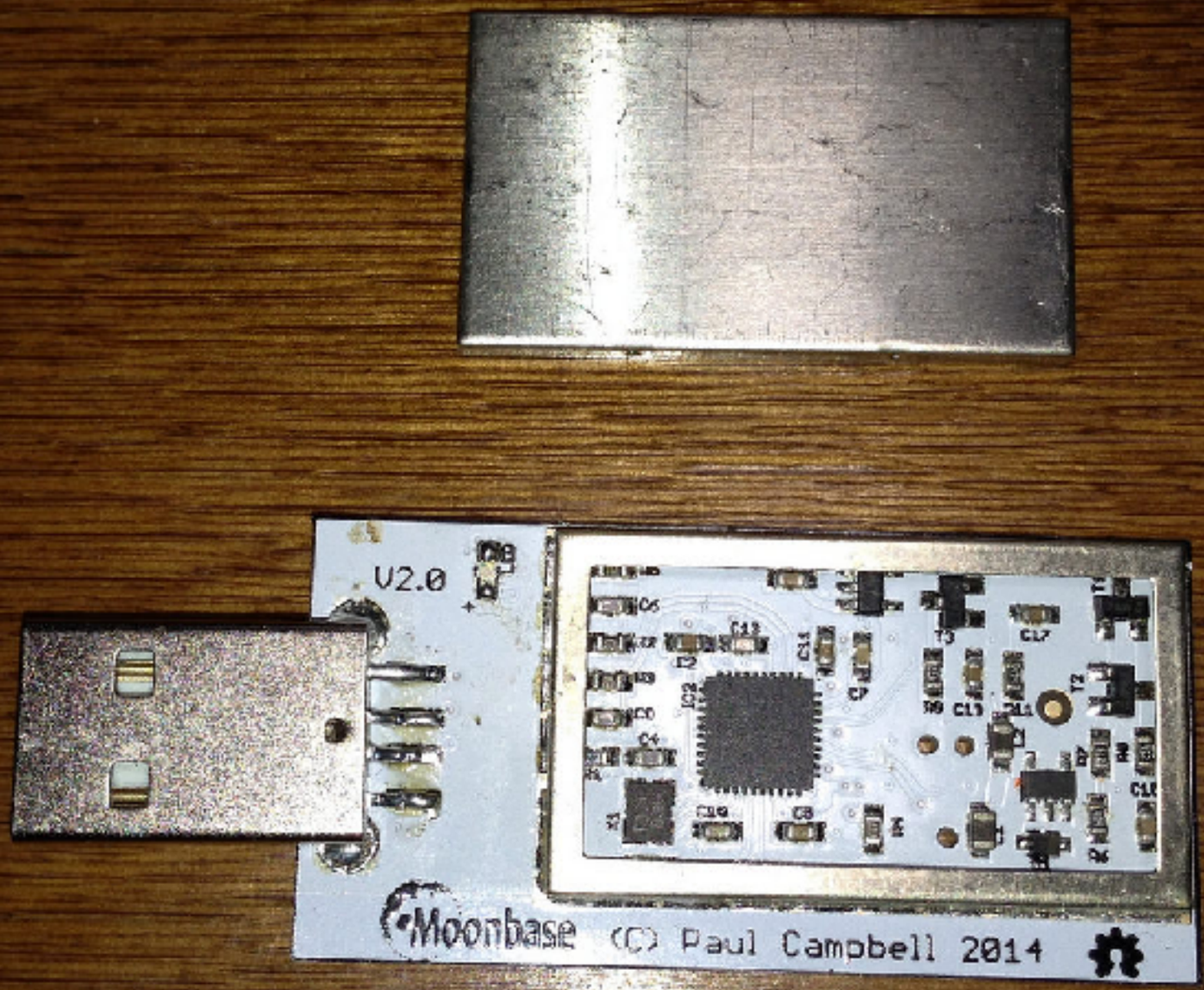
- CircuitHub
- Whiplash Merch


DEATH
DEATH DEATH DEATH
DEATH DEATH DEATH

/dev/random

STRTIME	UID	PID	COMM	FD	PATH
2013 Oct 8 17:40:52	1493659202	1429	quicklookd	15	/dev/random
2013 Oct 8 17:40:52	1493659202	1429	quicklookd	15	/dev/random
2013 Oct 8 17:40:57	1493659202	1432	head	3	/dev/random
2013 Oct 8 17:41:01	1493659202	1433	head	3	/dev/random
2013 Oct 8 17:41:11	1493659202	1434	CVMCompiler	4	/dev/random
2013 Oct 8 17:41:11	1493659202	1434	CVMCompiler	4	/dev/random
2013 Oct 8 17:41:48	0	1436	kcm	5	/dev/random
2013 Oct 8 17:41:48	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:41:49	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:41:49	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:42:19	1493659202	1437	findNames	4	/dev/random
2013 Oct 8 17:42:19	1493659202	1437	findNames	4	/dev/random
2013 Oct 8 17:46:45	0	1440	kcm	5	/dev/random
2013 Oct 8 17:46:45	1493659202	589	CalendarAgent	7	/dev/random
2013 Oct 8 17:46:48	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:46:48	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:46:48	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:46:48	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:51:48	0	1467	kcm	5	/dev/random
2013 Oct 8 17:51:48	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:51:49	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:51:49	1493659202	738	Mail	35	/dev/random
2013 Oct 8 17:52:02	1493659202	1468	dd	4	/dev/random
2013 Oct 8 17:52:50	1493659202	1469	Google Chrome H	18	/dev/random
2013 Oct 8 17:52:50	1493659202	1469	Google Chrome H	19	/dev/random


V2.0
Moonbase
(C) Paul Campbell 2014


V2.0
Moonbase (C) Paul Campbell 2014


TrueRNG
2
ubld.it


1
Bit
Babbler
1
Bit
Babbler

QUANTIS
QUANTUM RANDOM NUMBER GENERATOR
IDQ
FROM VISION TO TECHNOLOGY
www.idquantique.com
MADE IN SWITZERLAND
TRUERNG2

Popular noise sources

- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise

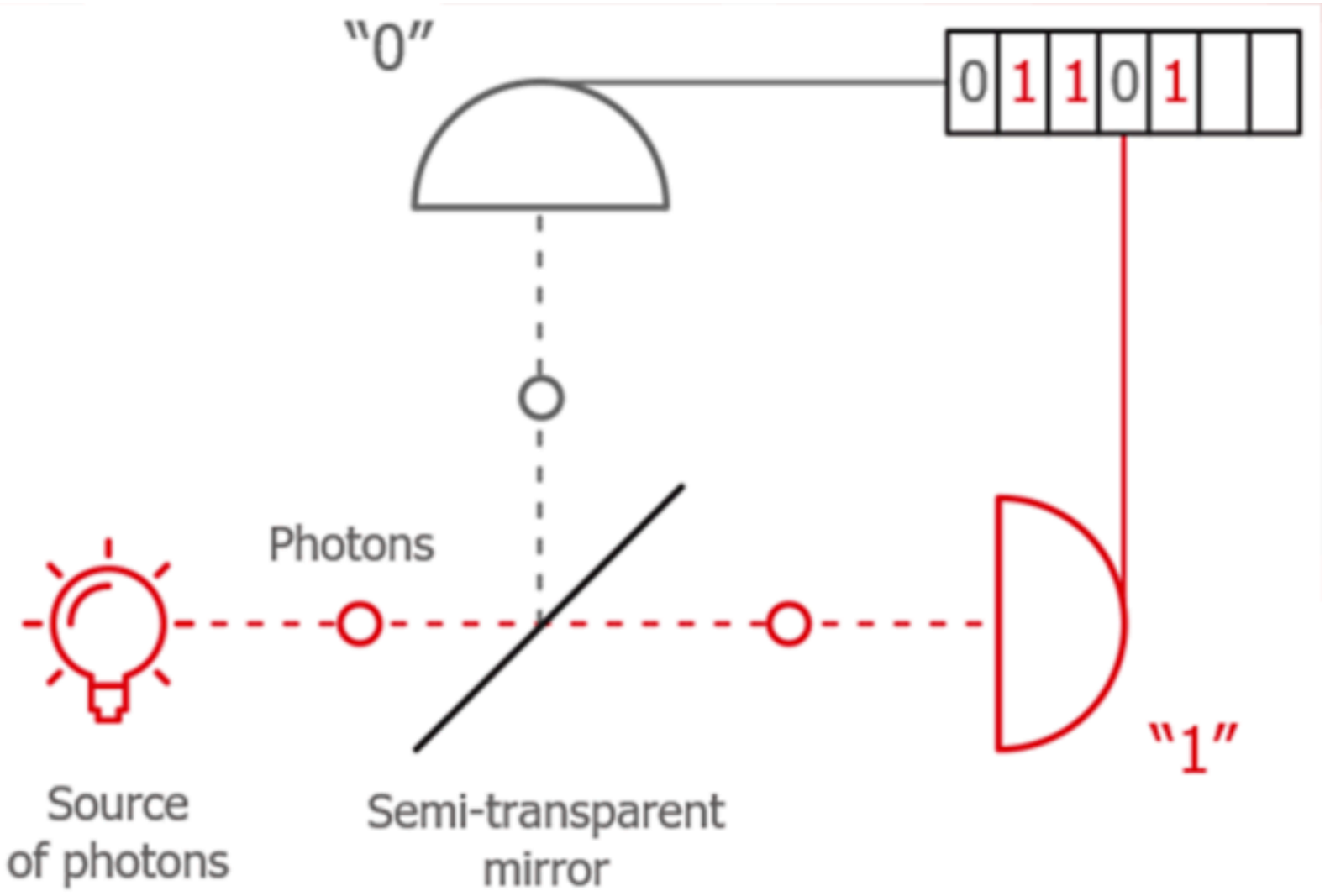
Popular noise sources

- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise


Cs-137
Gamma
AWARE
Electronics
Model RM-80
Computer Interface

Popular noise sources

- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise


"0"
0 1 1 0 1
Photons
Source
of photons
Semi-transparent
mirror
"1"

Popular noise sources

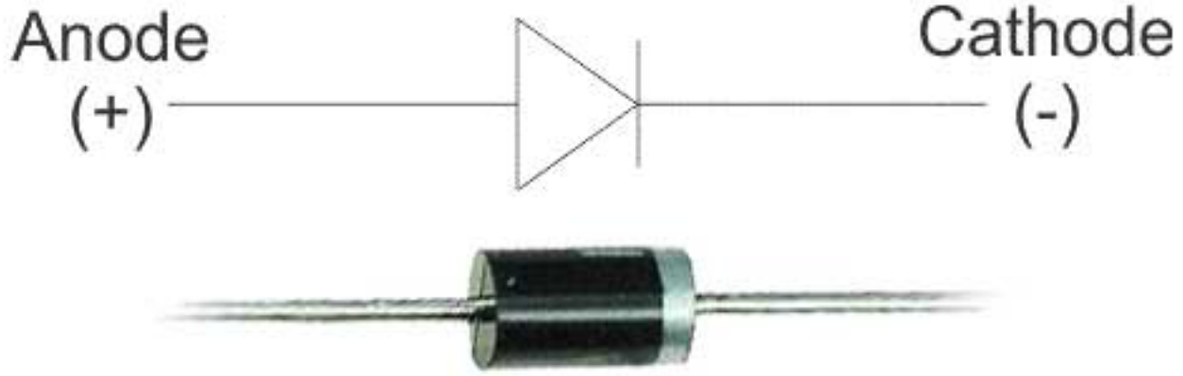
- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise

Popular noise sources

- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise

Popular noise sources

- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise


Anode
(+)
Cathode
(-)

Popular noise sources

- Radioactive decay
- Photons
- Radio noise
- Lava lamp
- Avalanche noise
- Thermal noise

```
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is <speeed_whitened_1.dat>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
 14  17   5  11  10   6   5   9  12  11  0.129620     97/100     Frequency
 22  14  12  15  11   6   5   5   6   4  0.000320     98/100     BlockFrequency
 15   9  15   9   6   6  12  13   7   8  0.275709     97/100     CumulativeSums
 16  15  13  11   7   6  10   7   9   6  0.202268     97/100     CumulativeSums
 91   4   1   1   0   1   1   1   0   0  0.000000 *   24/100  *  Runs
 12   9   7  14  12   9   8   9   5  15  0.437274     99/100     LongestRun
  7  14   7  10   8   7  13  13  15   6  0.304126     99/100     Rank
 13   9   8  12  12  10  10   3  10  13  0.534146     99/100     FFT
```

…non overlapping templates all pass….

```
 14  16  15  11  13   9   4   6   4   8  0.035174     97/100     NonOverlappingTemplate
 16  14  12  11   7   8  11   5  11   5  0.202268     98/100     OverlappingTemplate
  8   6  14  12  15   5   8   7  10  15  0.171867     99/100     Universal
 28  14  15   8   5   6   7   7   4   6  0.000000 *   96/100     ApproximateEntropy
  8   5   5   4   6   3   8   7   5   6  0.759756     57/57      RandomExcursions
```

… everything else passes...

```
  4   8   4   4   3   7  11   7   2   7  0.102526     57/57      RandomExcursionsVariant
 16   5   8  11   6  12  15   8  11   8  0.213309     99/100     Serial
 12   6   9   9   9  11   9  12  14   9  0.867692     98/100     Serial
 10  12   3  11   9  13  12  10   9  11  0.637119     99/100     LinearComplexity
```