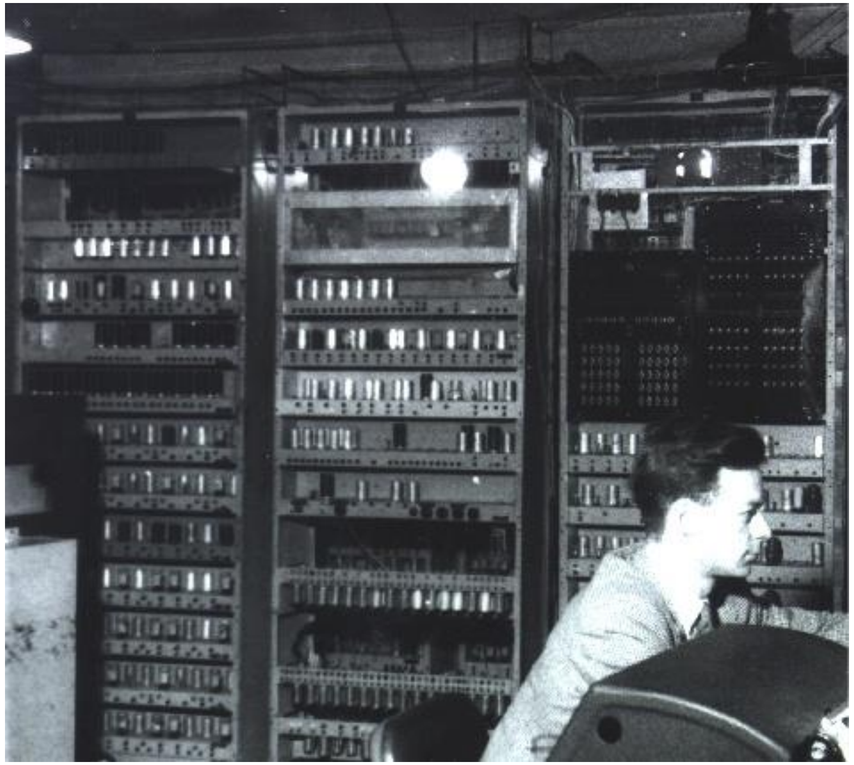
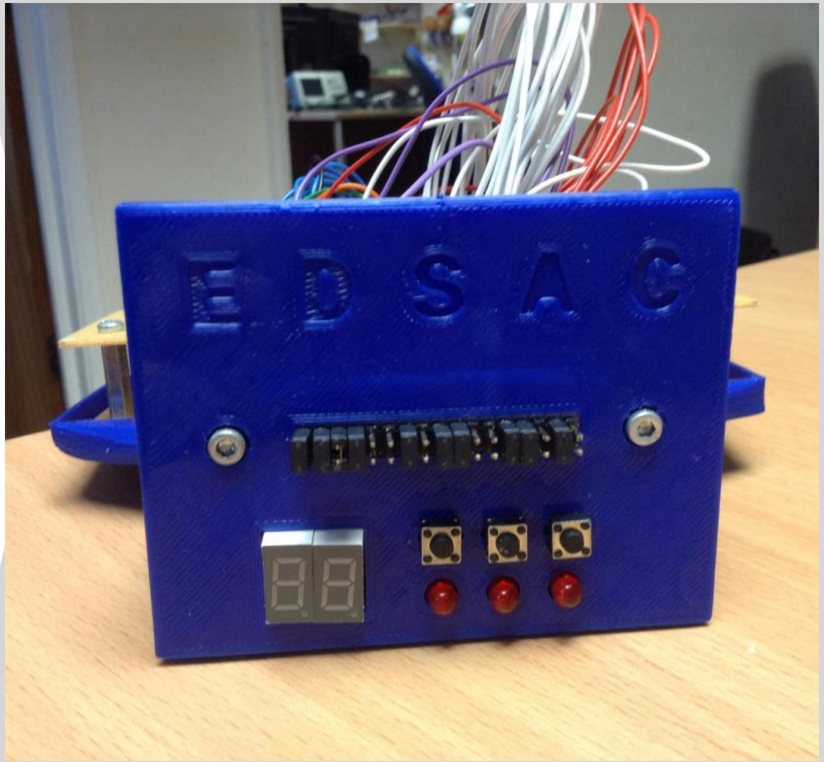
Reimagining EDSAC in Open Source

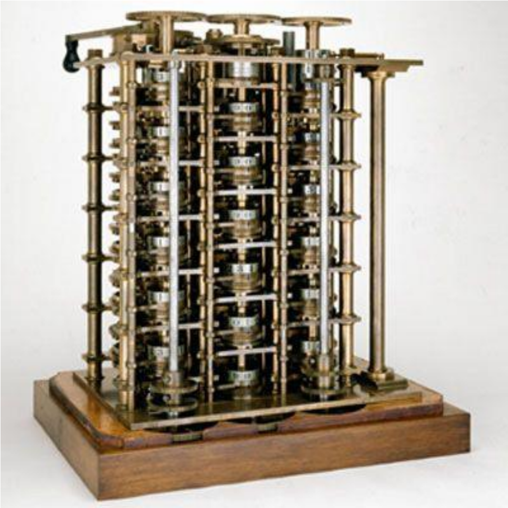
Mary Bennett

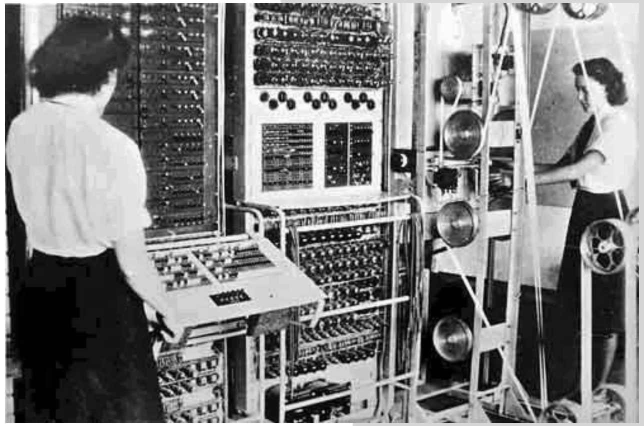
Overview

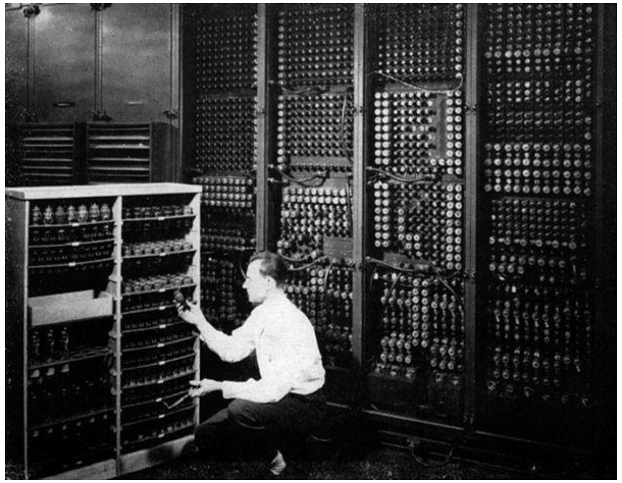


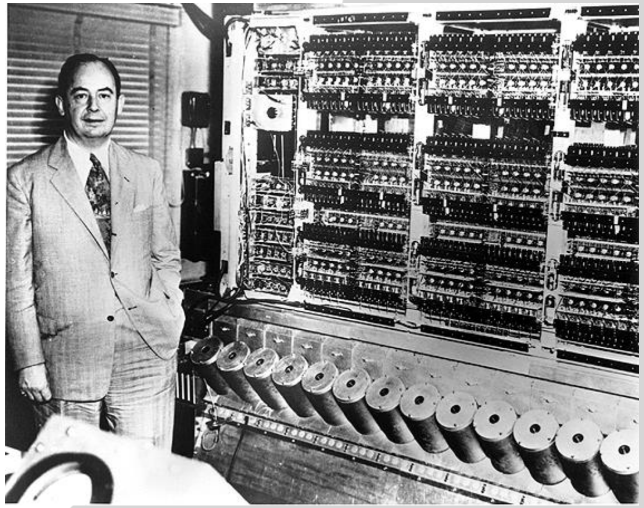


Predecessors

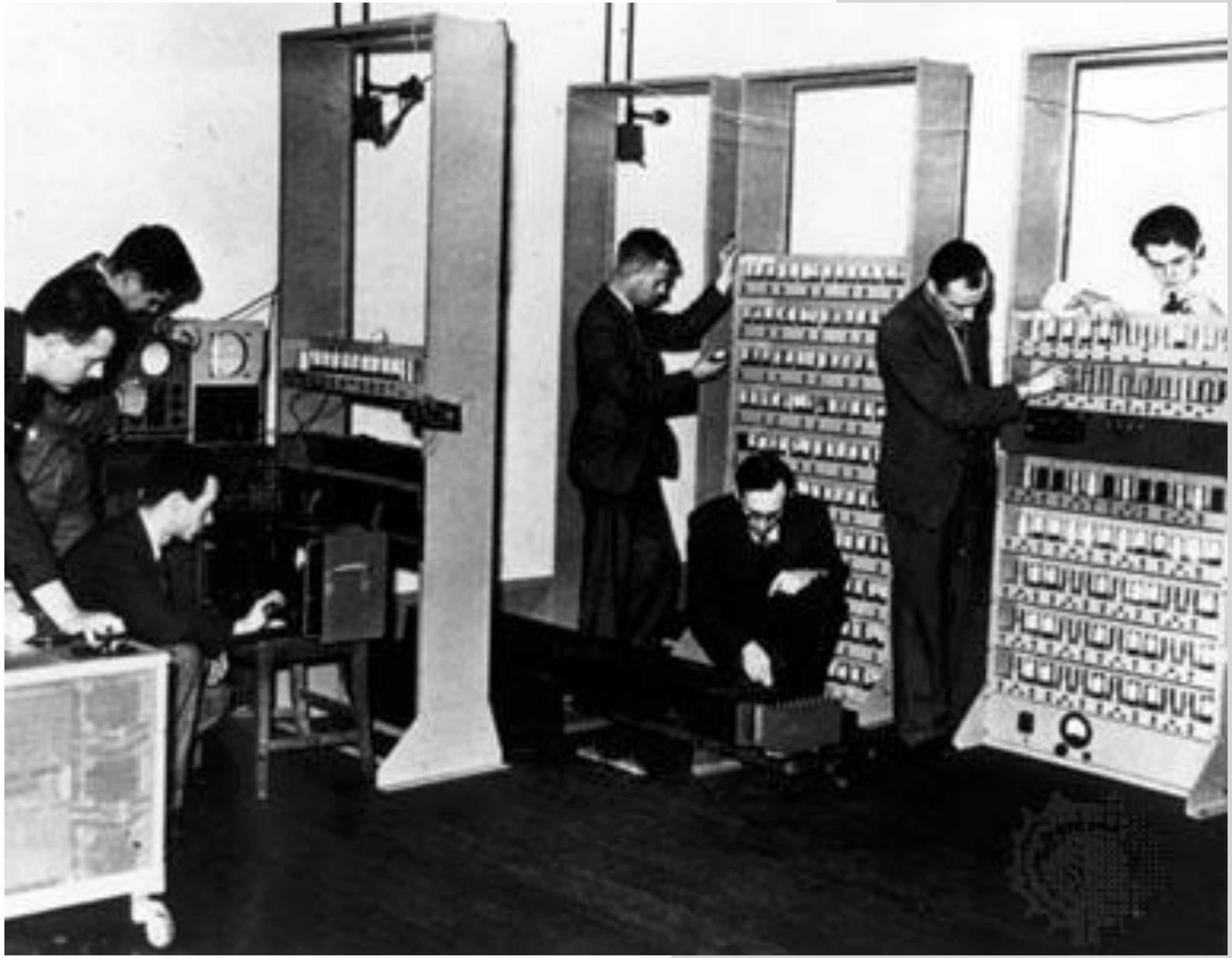




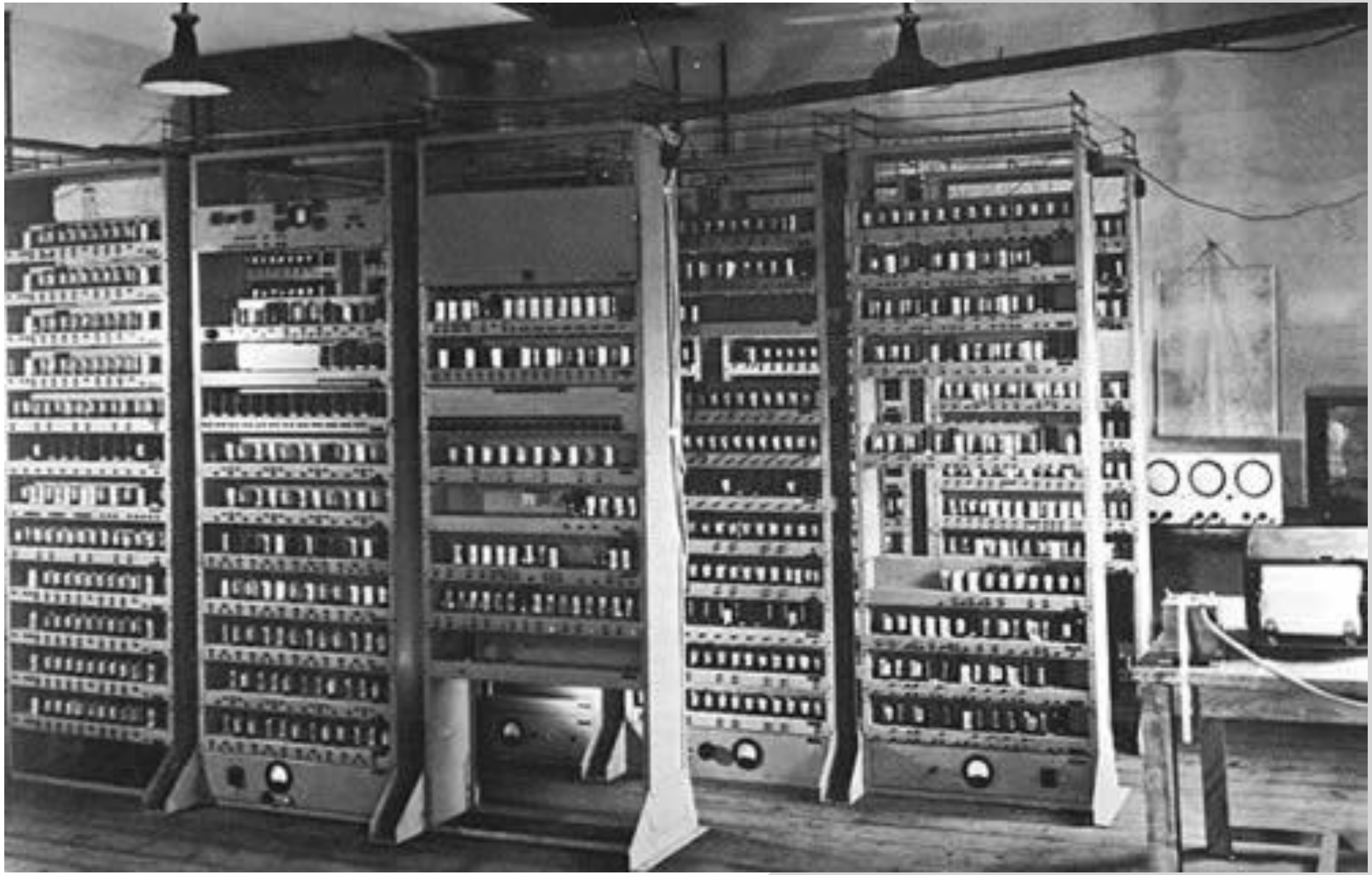




Building EDSAC

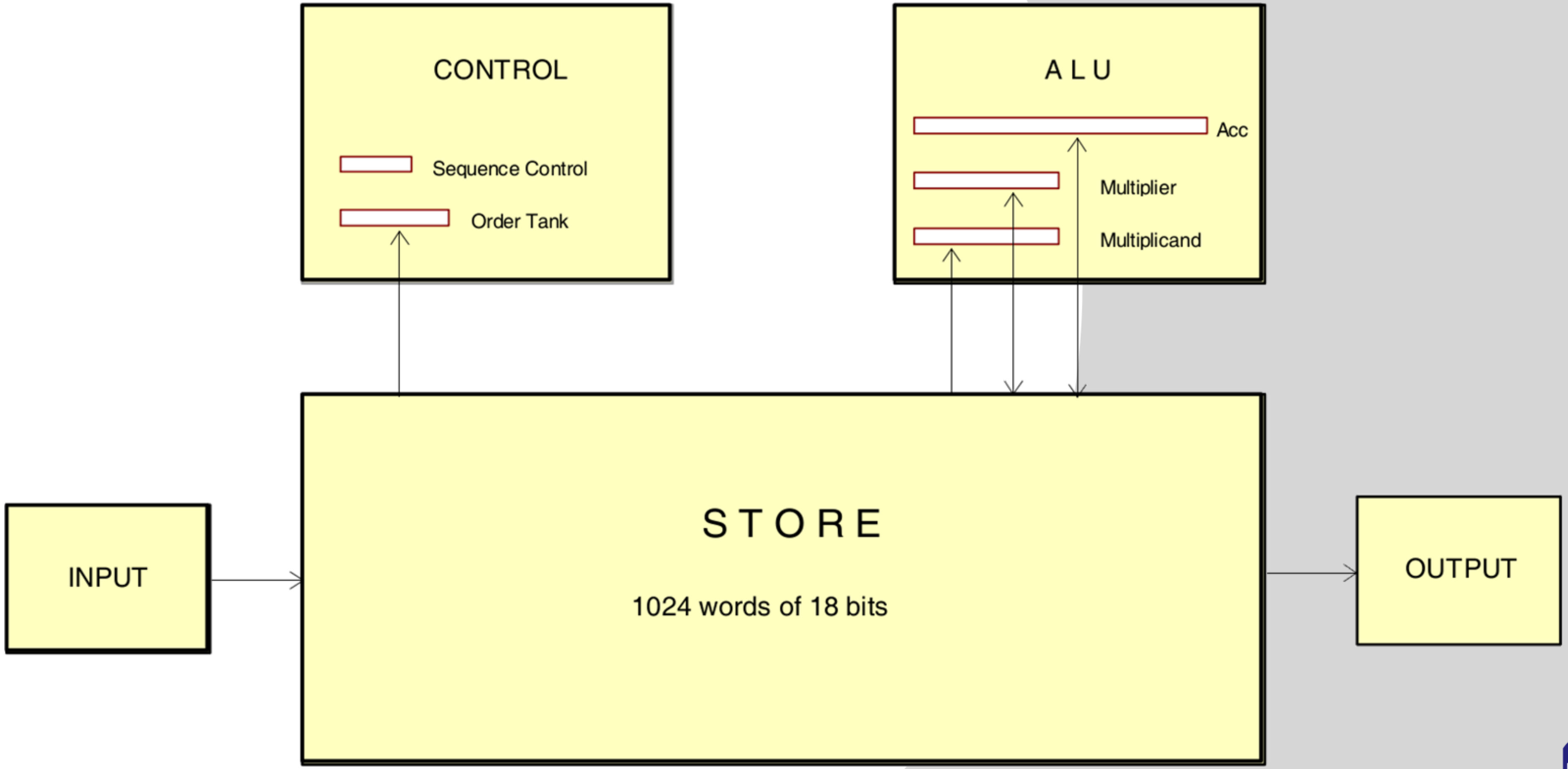


EDSAC



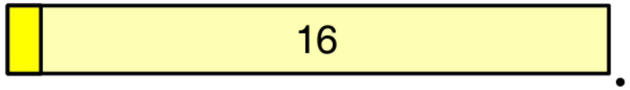
EDSAC's Memory

EDSAC Architecture

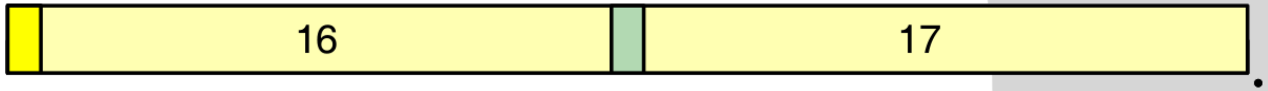


A Tutorial Guide to the EDSAC Simulator, Department of Computer Science, University of Warwick

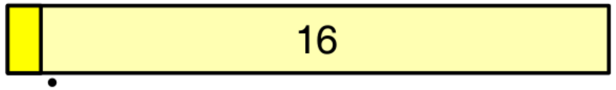
EDSAC Data



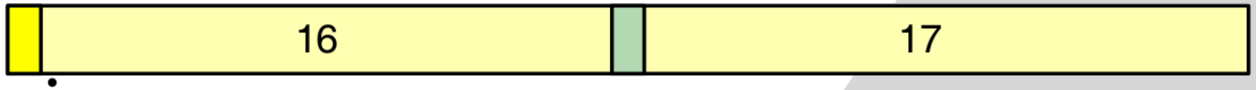
(a) short integer



(b) long integer



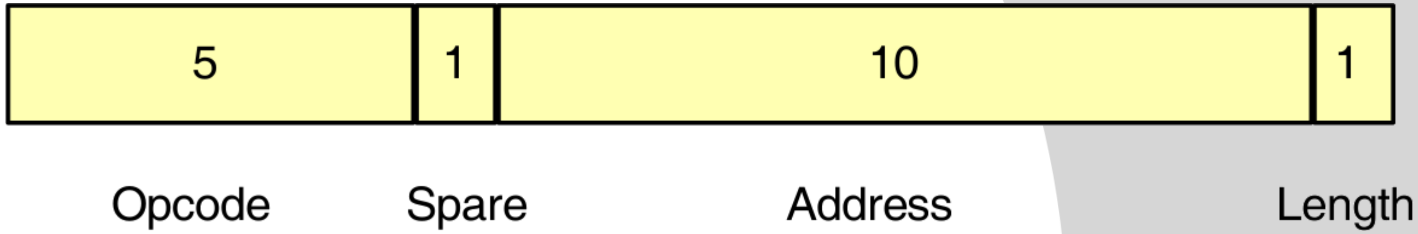
(c) short fraction



(d) long fraction

A Tutorial Guide to the EDSAC Simulator, Department of Computer Science, University of Warwick

EDSAC Instructions



A *n* Add value at storage location *n* to the accumulator

S *n* Subtract value at storage location *n* from the accumulator

...

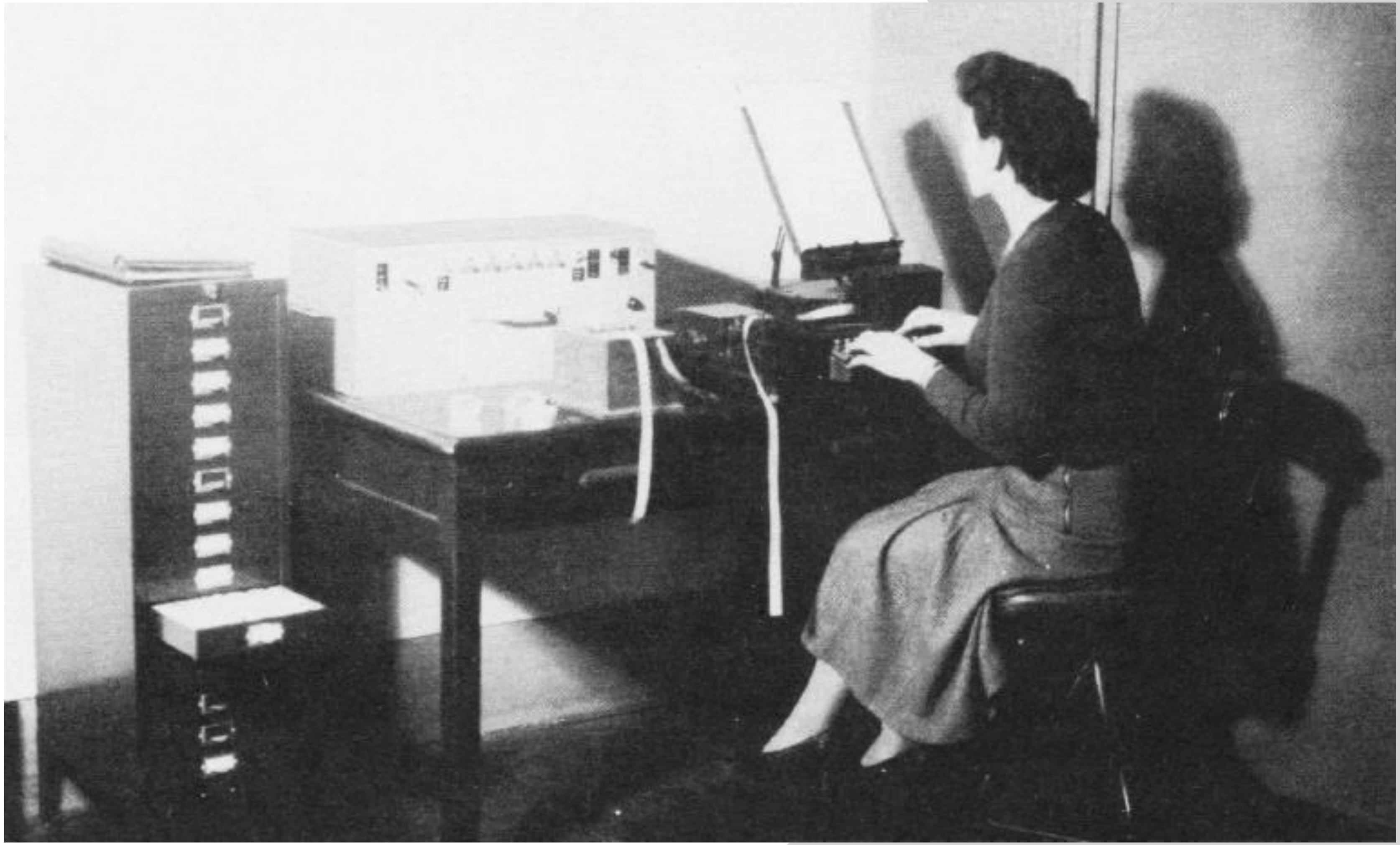
Y Round the accumulator to 34 bits

Z Stop the machine

Total of 18 instructions

A Tutorial Guide to the EDSAC Simulator, Department of Computer Science, University of Warwick

Program Input

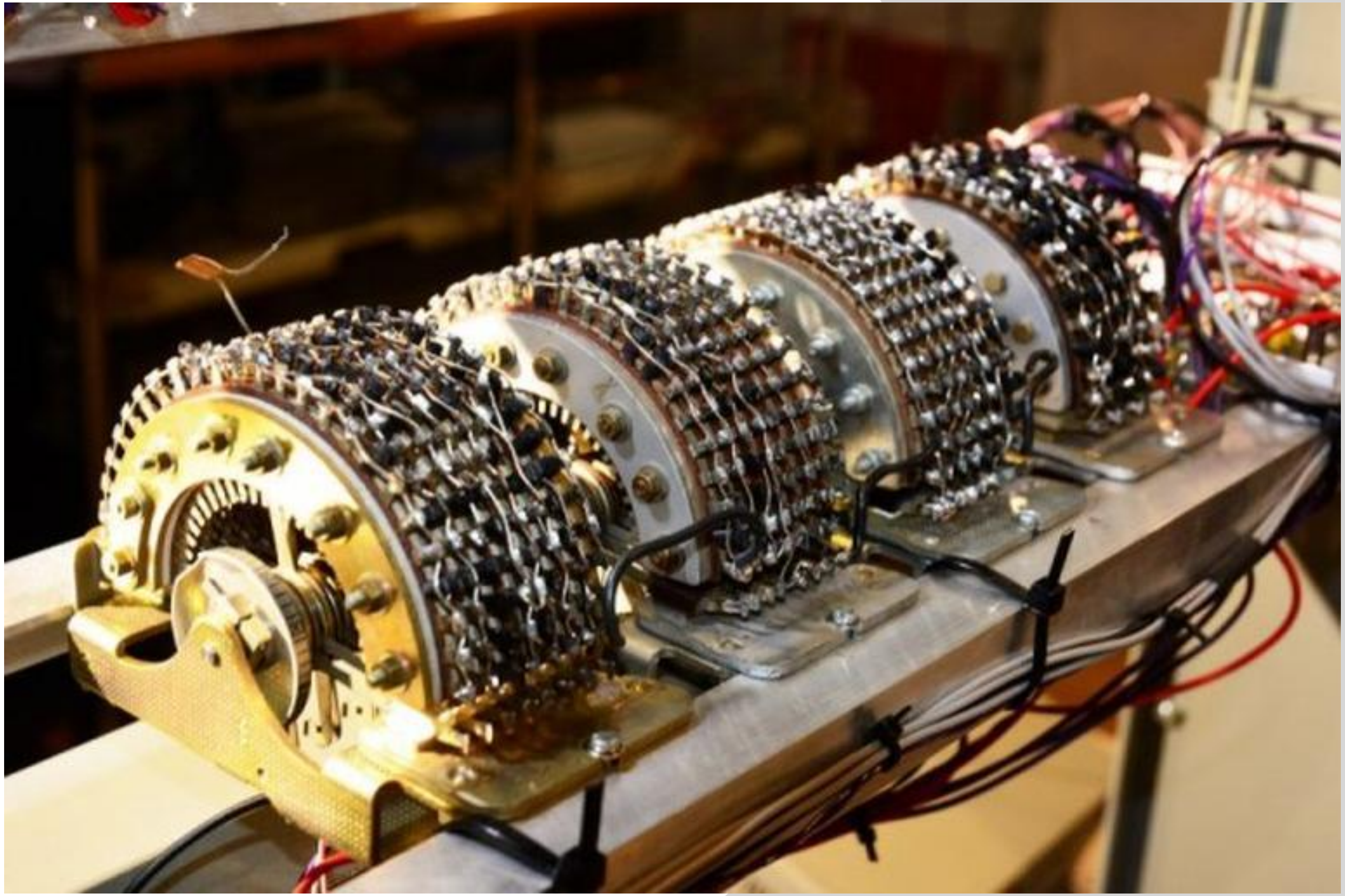


Job Queue

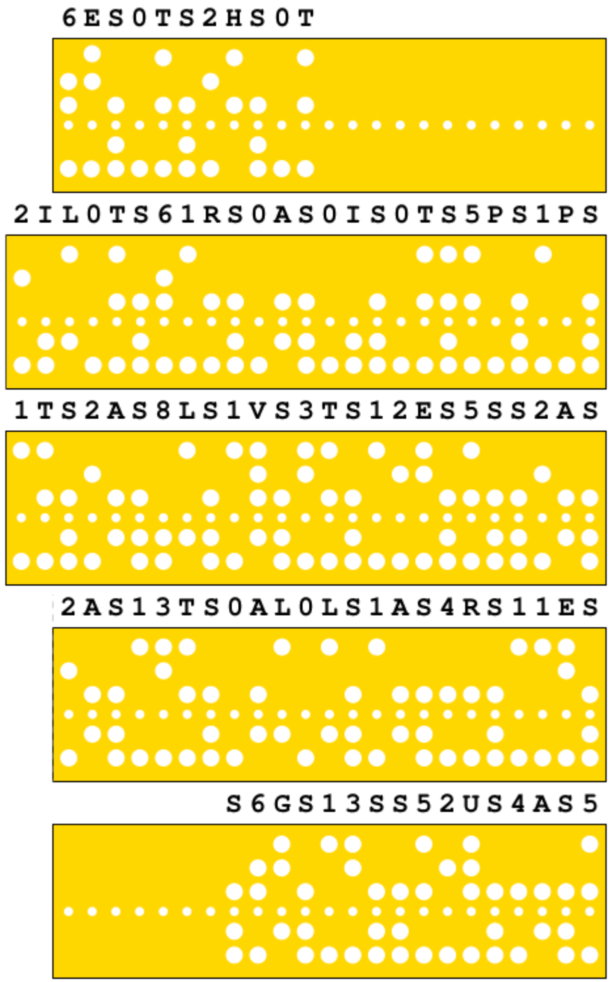
Teleprinter

EMBECOSM®

Bootstrap



EDSAC Initial Orders



Order bit pattern	Loc	Order	Meaning	Comment
00101 0 0000000000 0	0:	T0S	m[0]=A; ABC=0	
10101 0 0000000010 0	1:	H2S	R=m[2]	Put 10<<11 in R
00101 0 0000000000 0	2:	T0S	m[0]=A; ABC=0	
00011 0 0000000110 0	3:	E6S	goto 6	Jump to main loop
00000 0 0000000001 0	4:	P1S	data 2	The constant 2
00000 0 0000000101 0	5:	P5S	data 10	The constant 10
00101 0 0000000000 0	6:	T0S	m[0]=A; ABC=0	Start of the main loop
01000 0 0000000000 0	7:	I0S	m[0]=rdch()	Get operation code
11100 0 0000000000 0	8:	A0S	A+=m[0]	Put it in A
00100 0 0000010000 0	9:	R16S	ABC>>=6	Shift and store it
00101 0 0000000000 1	10:	T0L	w[0]=AB; ABC=0	so that it becomes the senior 5 bits of m[0] m[1] is now zero
01000 0 0000000010 0	11:	I2S	m[2]=rdch()	Put next ch in m[2]
11100 0 0000000010 0	12:	A2S	A+=m[2]	Put ch in A
01100 0 0000000101 0	13:	S5S	A-=m[5]	A=ch-10
00011 0 0000010101 0	14:	E21S	if A>=0 goto 21	Jump to 21, if ch>=10
00101 0 0000000011 0	15:	T3S	m[3]=A; ABC=0	Clear A, m[3] is junk
11111 0 0000000001 0	16:	V1S	AB+=m[1]*R	A = m[1]*(10<<11)
11001 0 0000001000 0	17:	L8S	A<<=5	Shift 5 more places
11100 0 0000000010 0	18:	A2S	A+=m[2]	Add the new digit
00101 0 0000000001 0	19:	T1S	m[1]=A; ABC=0	Store back in m[1]
00011 0 0000001011 0	20:	E11S	goto 11	Repeat from 11
00100 0 0000000100 0	21:	R4S	ABC>>=4	A=2, if ch='S'(=12) A=15, if ch='L'(=25) lenbit=0, if ch='S' lenbit=1, if ch='L'
11100 0 0000000001 0	22:	A1S	A+=m[1]	Add in the address
11001 0 0000000000 1	23:	L0L	ABC<<=1	Shift to correct position
11100 0 0000000000 0	24:	A0S	A+=m[0]	Add in the operation field
00101 0 0000011111 0	25:	T31S	m[31]= A; ABC=0	Store the order in next location
11100 0 0000011001 0	26:	A25S	A+=m[25]	Increment the address field of m[25]
11100 0 0000000100 0	27:	A4S	A+=m[4]	m[4] holds 2
00111 0 0000011001 0	28:	U25S	m[25]=A	Update m[25]
01100 0 0000011111 0	29:	S31S	A-=m[31]	Jump to 6, if there are
11011 0 0000000110 0	30:	G6S	if A<0 goto 6	more orders to load

EMBECOSM®

EDSAC Initial Orders and Squares Program, Martin Richards, University of Cambridge Computer Laboratory

Wheeler Jumps

Location	*Order*	*Notes*	
p - 1	T0S	Move acc to loc 0 and zero acc	
p	ApS	Store current loc in acc	Main program
p + 1	EqS	Jump to loc q if acc positive	
.			
.			
.			
q	A3S	Loc 3 contains E2S, creates Ep+2S	Prologue
q + 1	Tq+rS	Overwrite link order	
.			
.			
.			
q + r - 1	T0S	Zero accumulator	Epilogue
q + r	E2S	Link order which will be modified	

First Book on Programming

THE PREPARATION OF
PROGRAMS
FOR AN ELECTRONIC
DIGITAL COMPUTER

With special reference to
the EDSAC
and the use of a
library of subroutines

by

MAURICE V. WILKES
Director of the Mathematical Laboratory of the
University of Cambridge

DAVID J. WHEELER

and

STANLEY GILL

1951

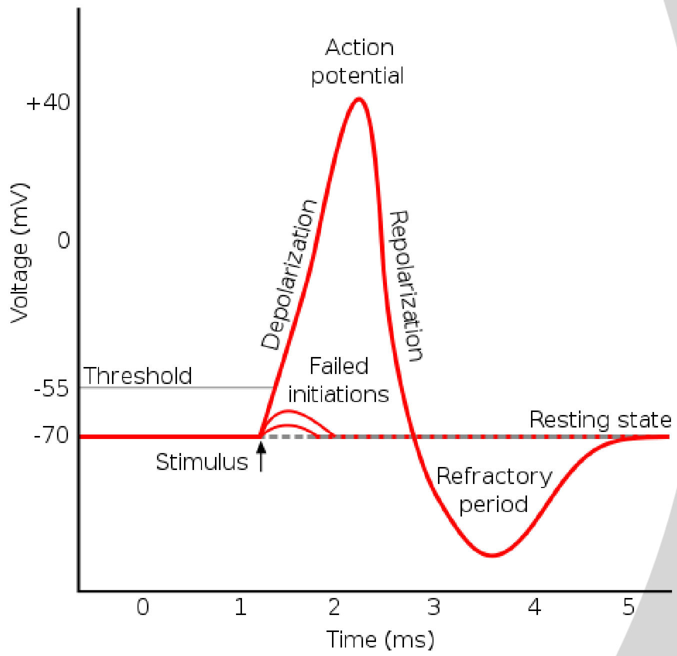
Famous Programs

$$180(2^{127} - 1)^2 + 1$$

TABLE II

STANDARDIZED DEVIATES (LEGITS) FOR GIVEN GENE PERCENTAGES

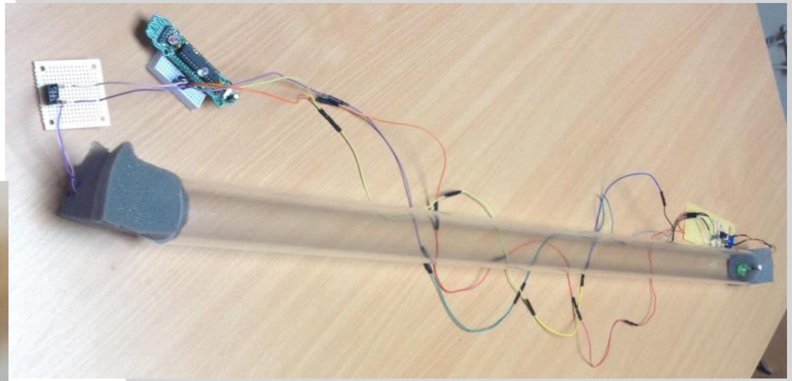
Gene percentage	Deviate	Gene percentage	Deviate	Gene percentage	Deviate
50	.00000	20.0	.64827	5.0	1.30643
49	.01908	19.5	.66247	4.8	1.32331
48	.03817	19.0	.67691	4.6	1.34080
47	.05729	18.5	.69161	4.4	1.35896
46	.07645	18.0	.70658	4.2	1.37784
45	.09566	17.5	.72184	4.0	1.39752
44	.11493	17.0	.73740	3.8	1.41807
43	.13429	16.5	.75329	3.6	1.43958
42	.15375	16.0	.76952	3.4	1.46216
41	.17332	15.5	.78612	3.2	1.48594
40	.19302	15.0	.80311	3.0	1.51106
39	.21286	14.5	.82052	2.8	1.53771
38	.23286	14.0	.83837	2.6	1.56609
37	.25304	13.5	.85669	2.4	1.59648
36	.27341	13.0	.87553	2.2	1.62922
35	.29400	12.5	.89492	2.0	1.66474
34	.31483	12.0	.91492	1.8	1.70360
33	.33592	11.5	.93556	1.6	1.74656
32	.35729	11.0	.95691	1.4	1.79468
31	.37897	10.5	.97902	1.2	1.84951
30	.40099	10.0	1.00198	1.0	1.91340
29	.42338	9.5	1.02586	0.9	1.94988
28	.44617	9.0	1.05076	0.8	1.99029
27	.46940	8.5	1.07680	0.7	2.03565
26	.49311	8.0	1.10411	0.6	2.08745
25	.51734	7.5	1.13285	0.5	2.14795
24	.54214	7.0	1.16321	0.4	2.22095
23	.56757	6.5	1.19543	0.3	2.31345
22	.59369	6.0	1.22978	0.2	2.44101
21	.62056	5.5	1.26662	0.1	2.65223

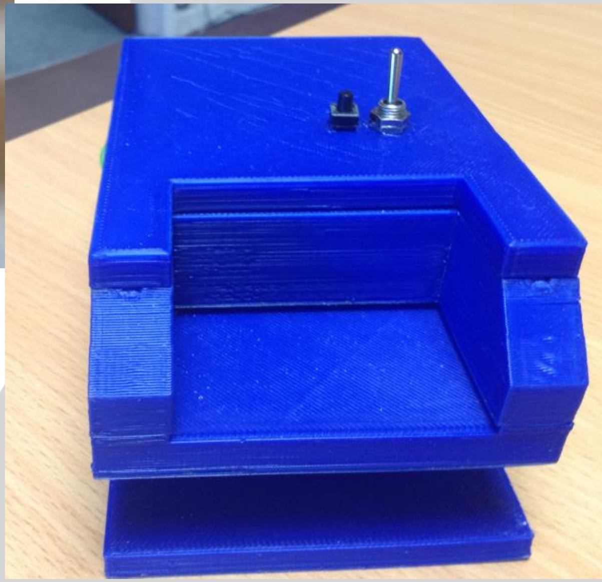


Popular Appreciation

The “brain” [computer] may one day come down to our level [of the common people] and help with our income-tax and book-keeping calculations. But this is speculation and there is no sign of it so far.

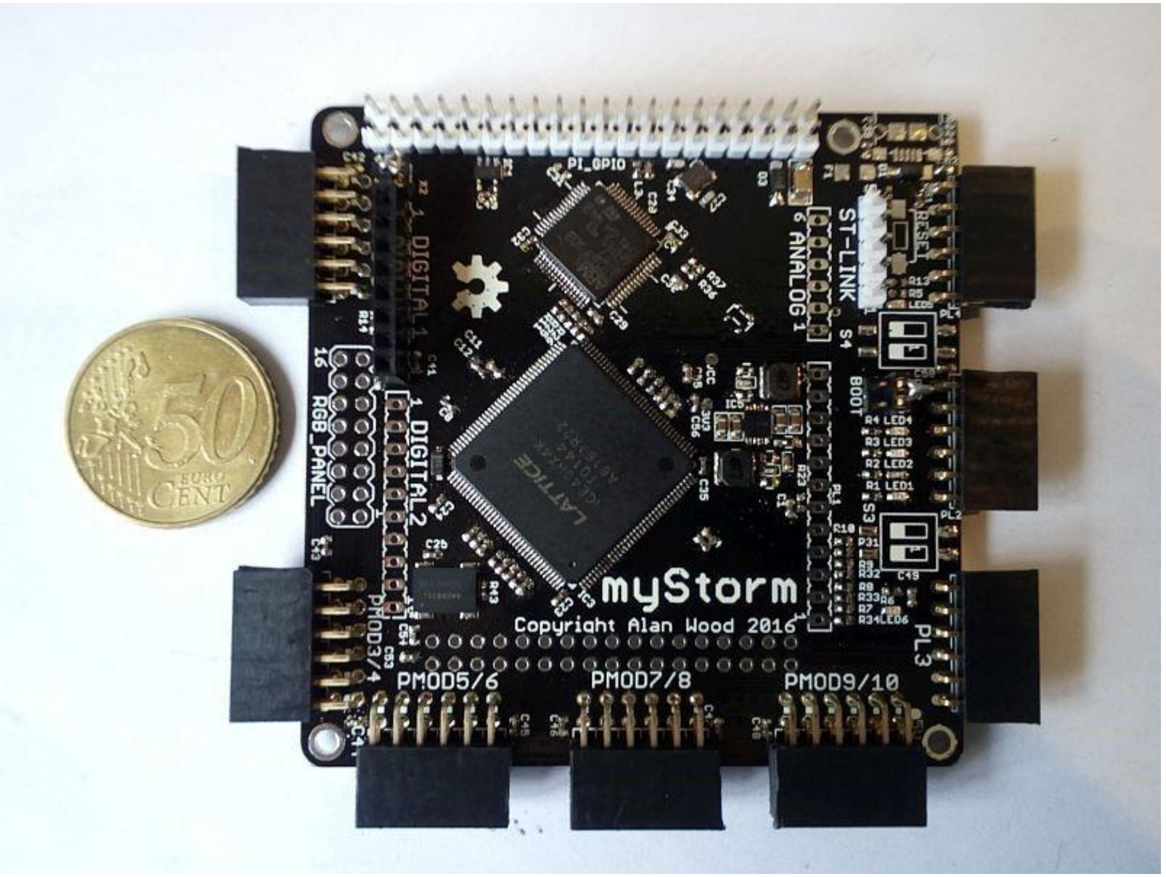
Reimagined EDSAC

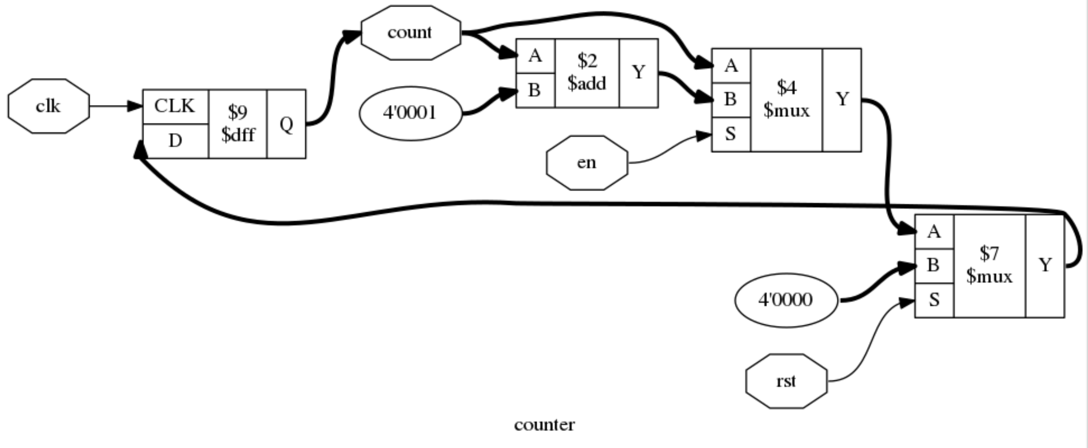


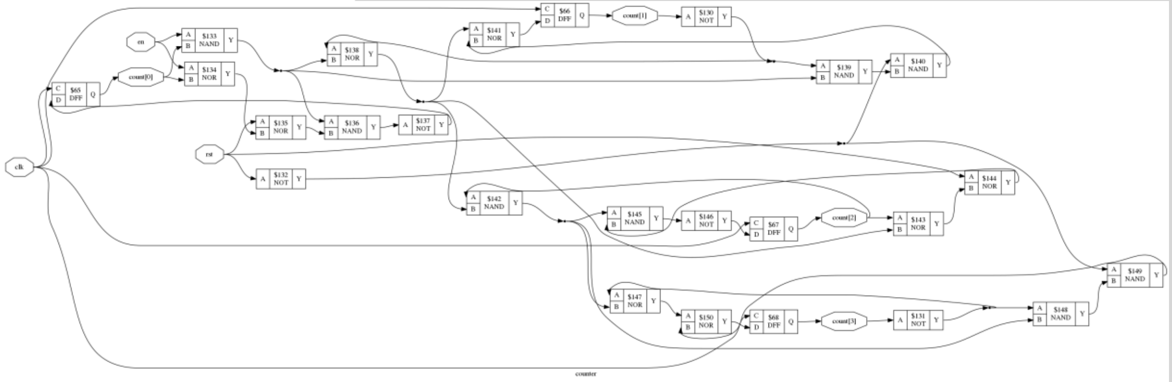


EMBECOSM®

Field Programmable Gate Array (FPGA)



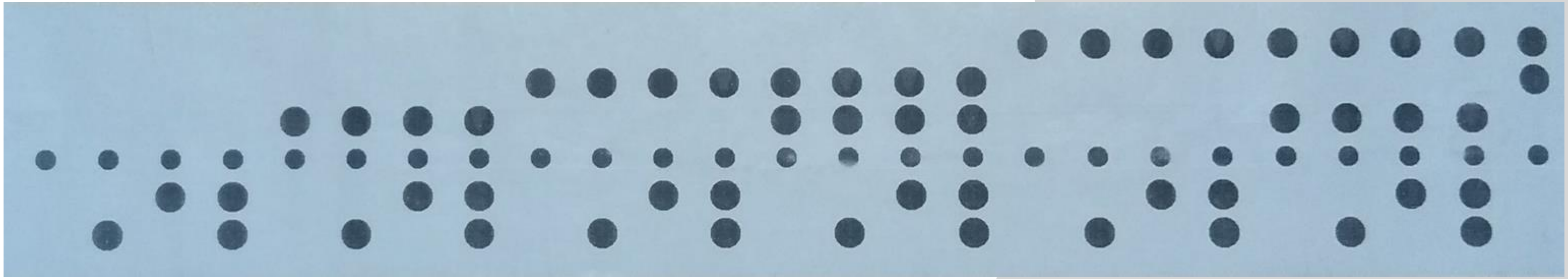




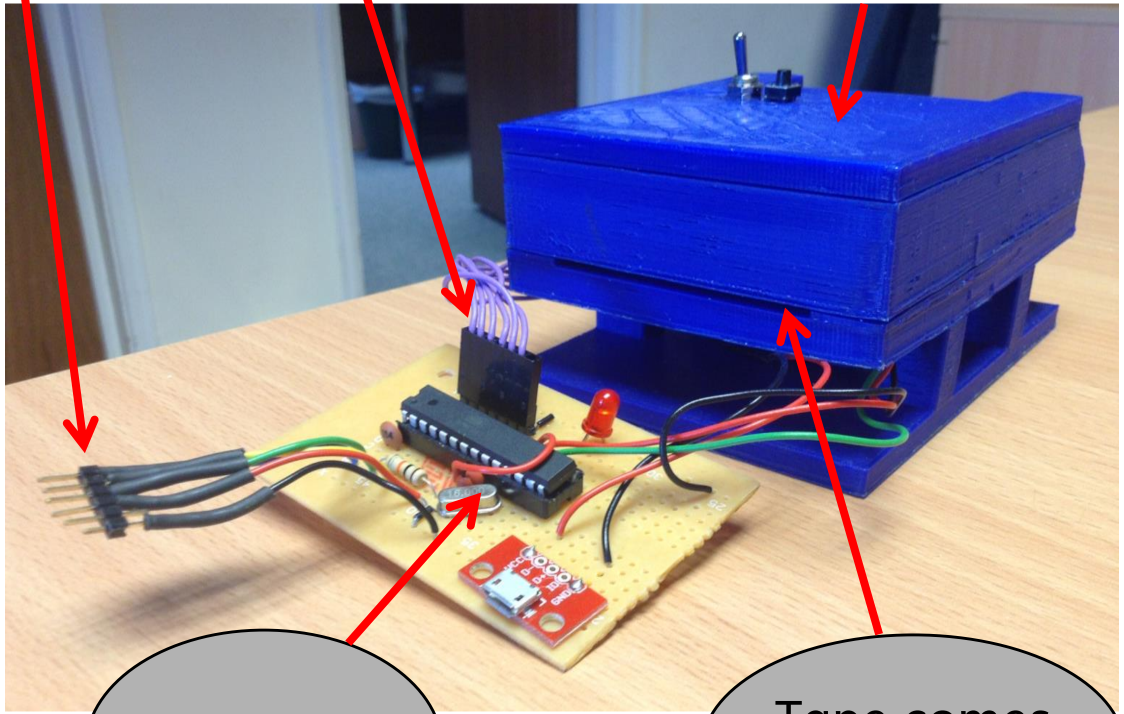
EMBECOSM®

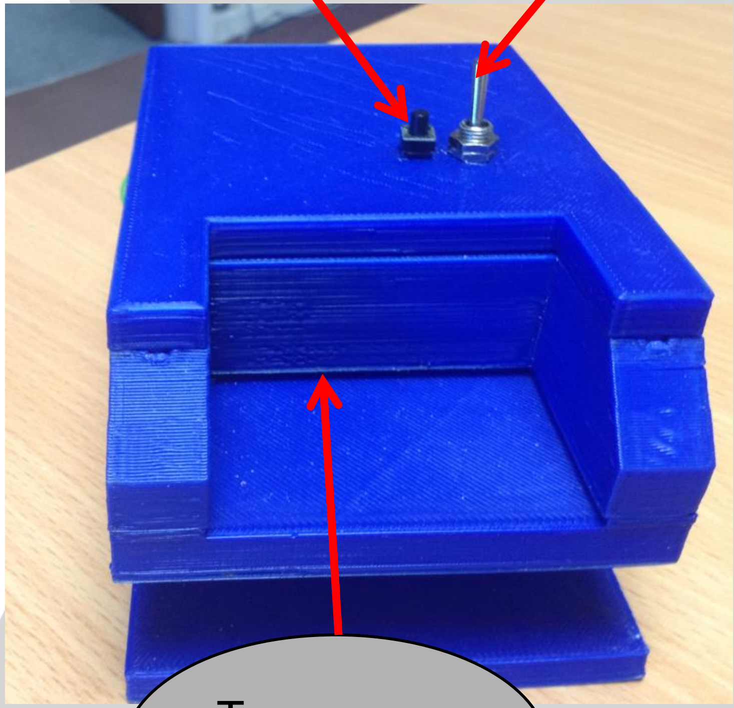
EDSAC Verilog

Reimagined Paper Tape



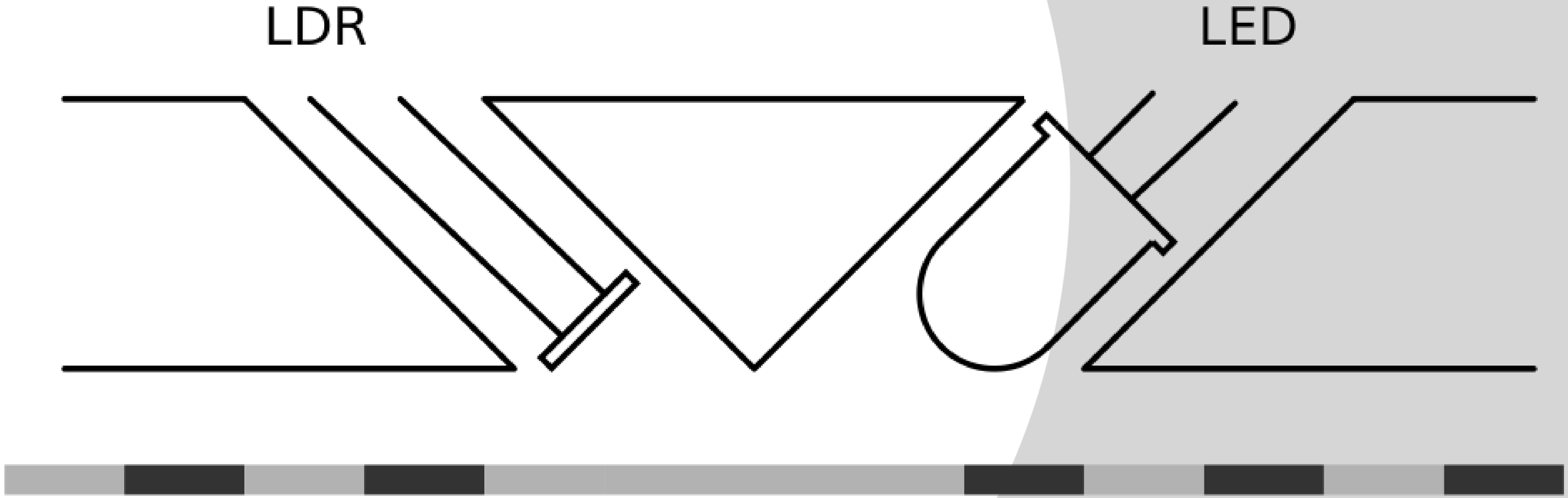
Reimagined Paper Tape Reader



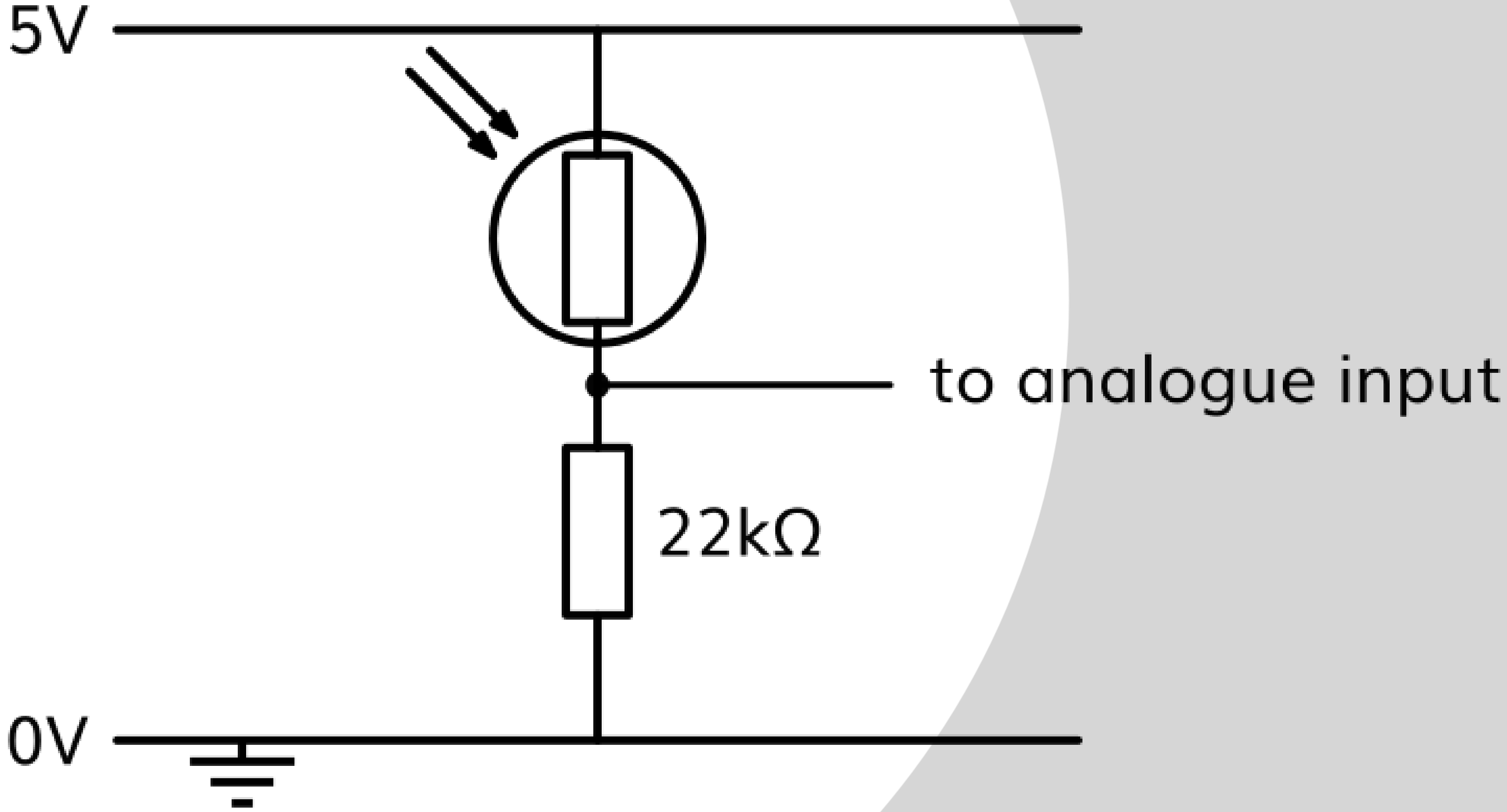


EMBECOSM®

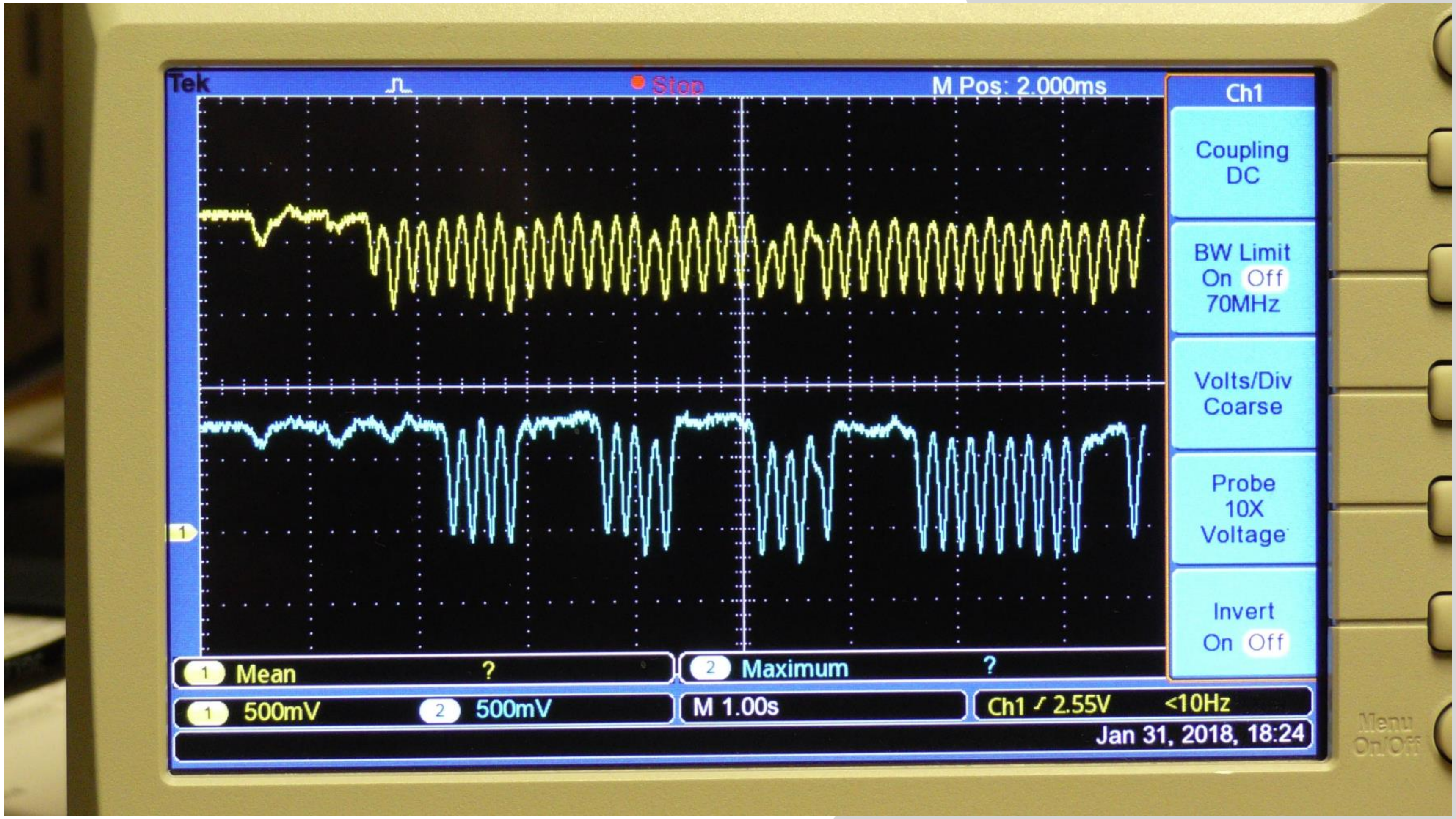
Sensor Design



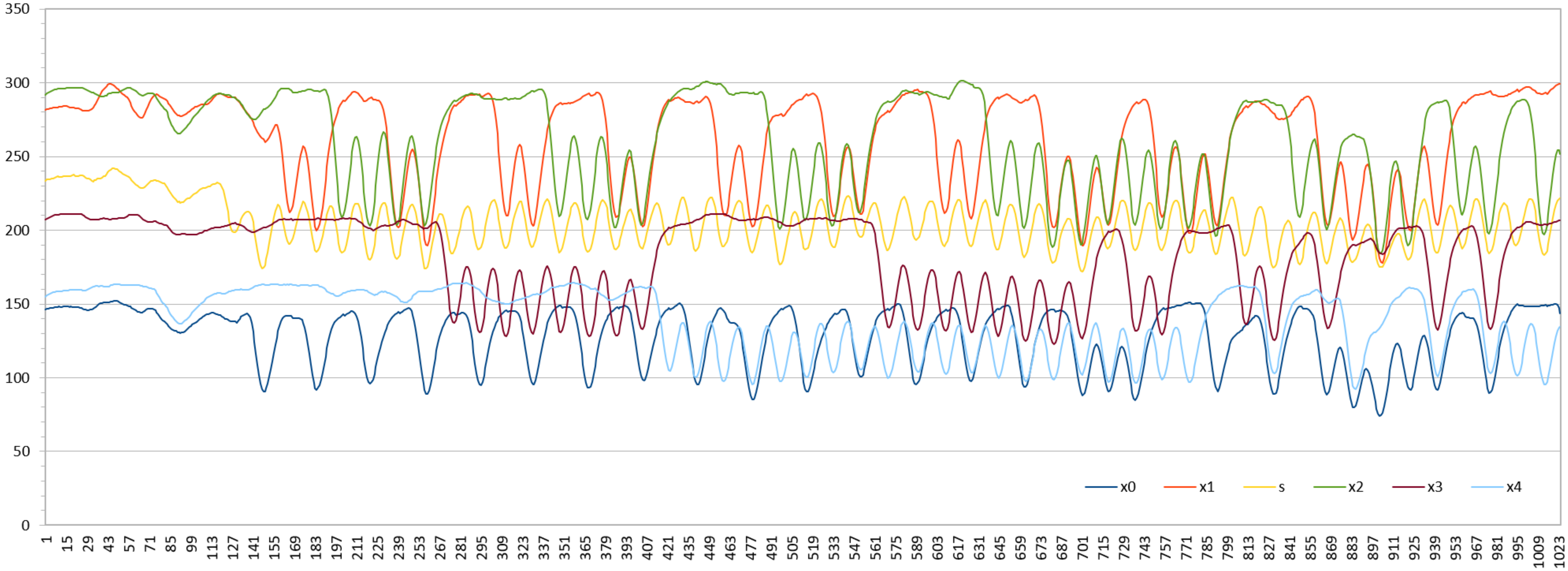
Sensor Circuit



Paper Tape Reader Output

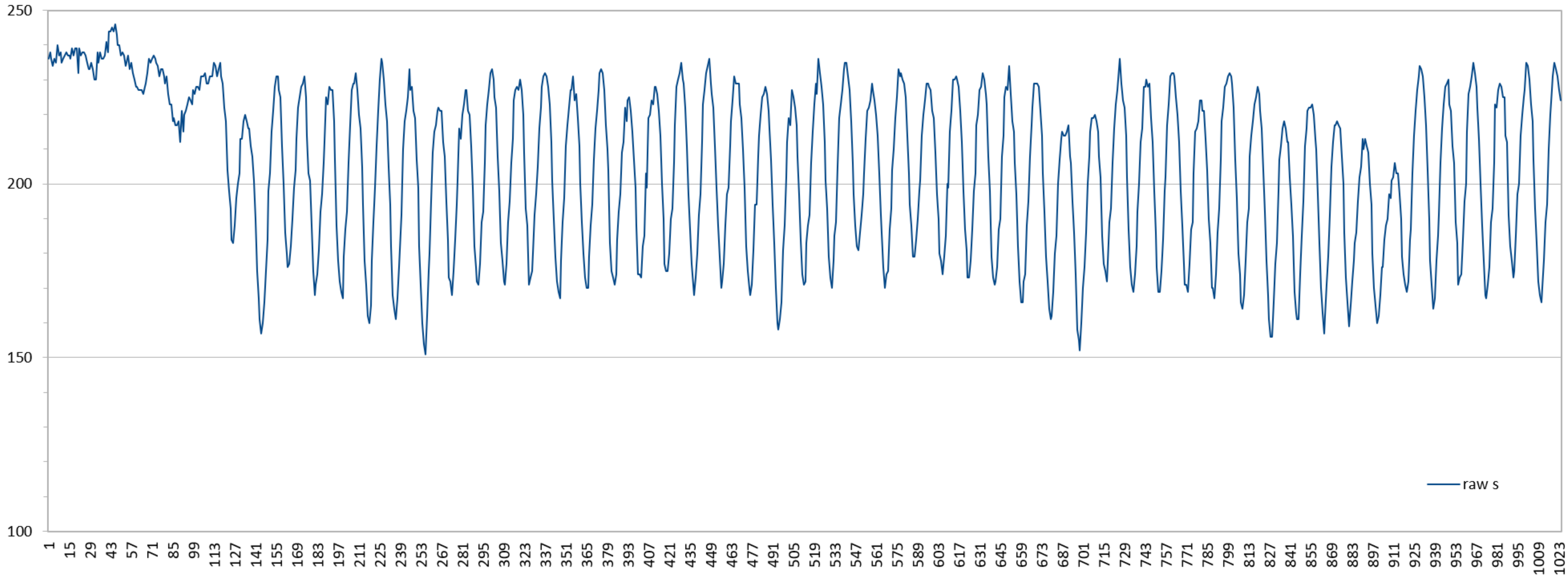


Raw Sensor Data



EMBECOSM®

Sprocket Hole



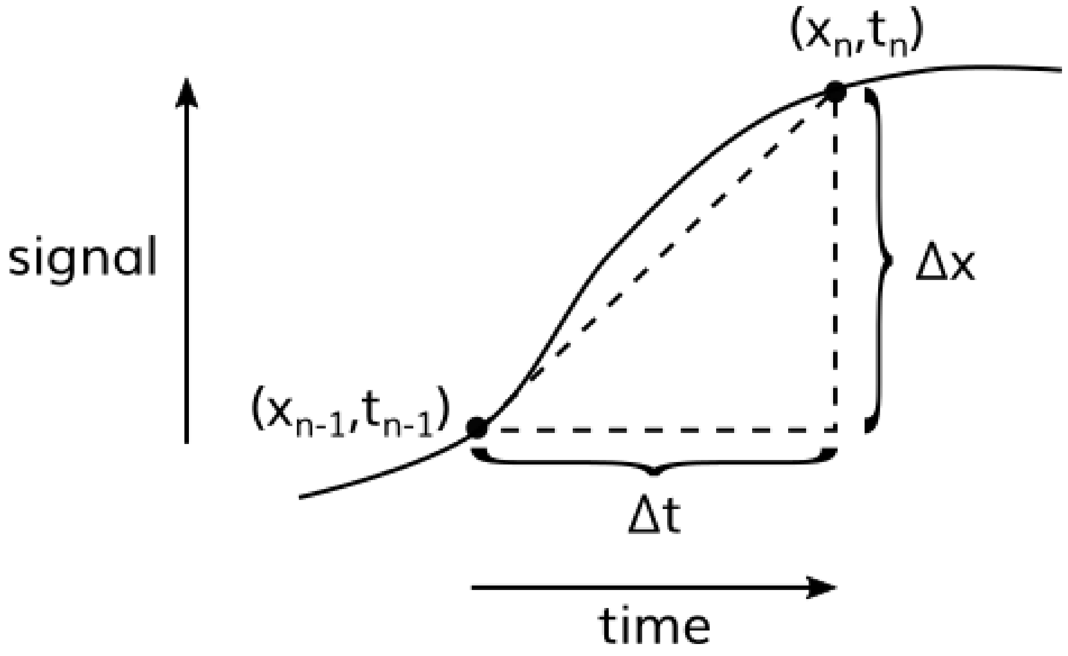
Calculating Rate of Signal Change

$$slope = \frac{\mathrm{d}x}{\mathrm{d}t}$$

Calculating Rate of Signal Change

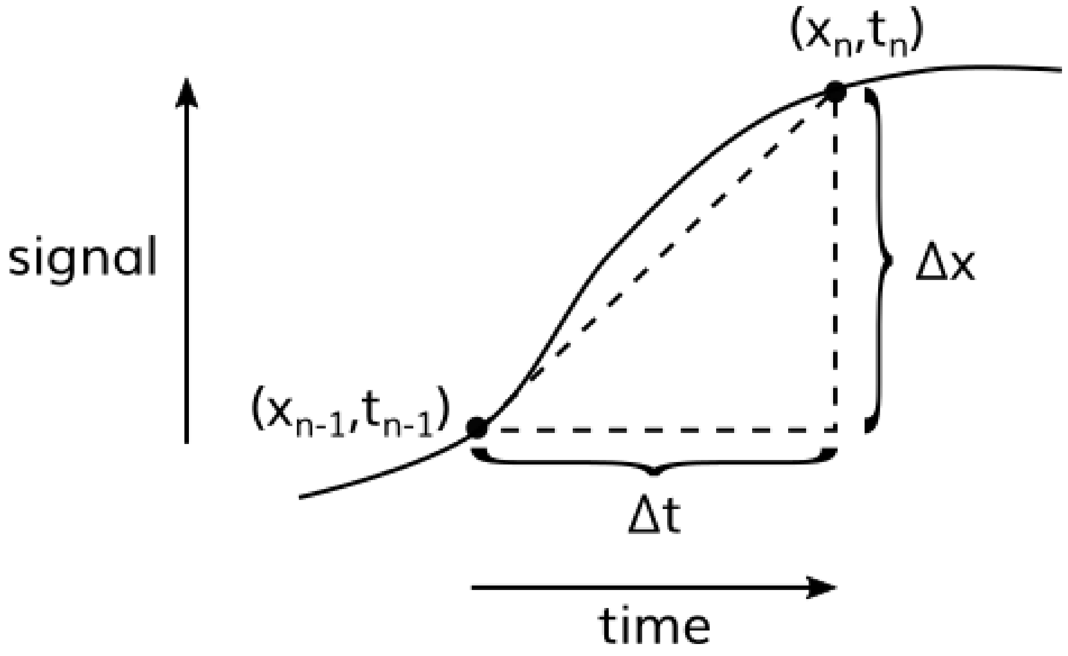
$$slope = \frac{\mathrm{d}x}{\mathrm{d}t}$$

$$slope = \frac{\Delta x}{\Delta t}$$



Calculating Rate of Signal Change

$$slope = \frac{\mathrm{d}x}{\mathrm{d}t}$$

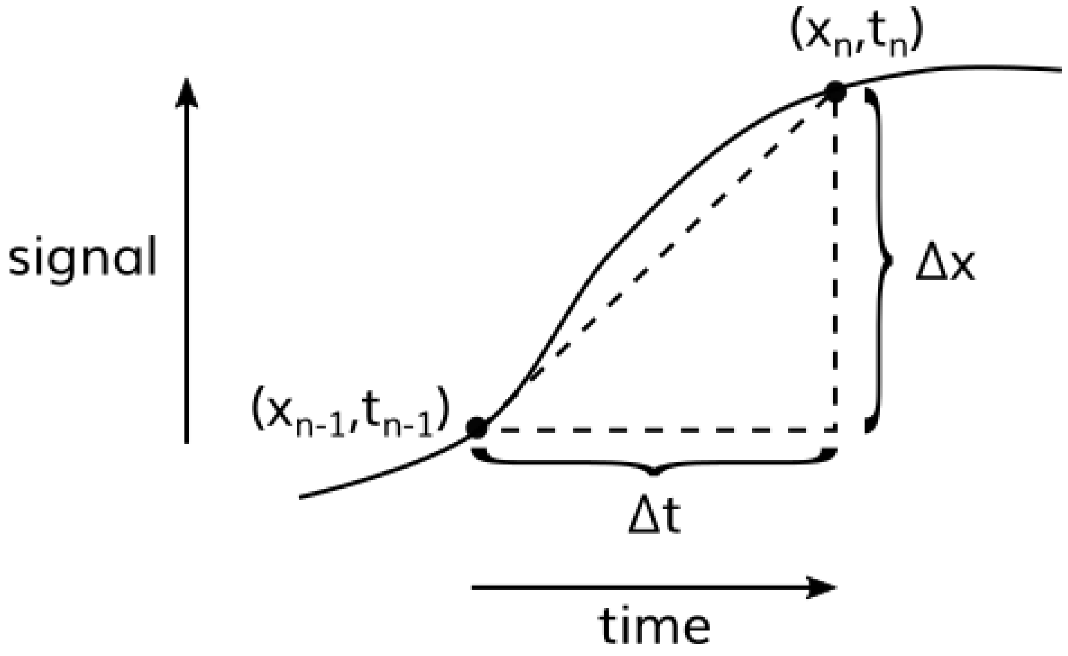


$$slope = \frac{\Delta x}{\Delta t}$$

$$slope_n = \frac{x_\mathrm{n} - x_{n-1}}{t_n - t_{n-1}}$$

Calculating Rate of Signal Change

$$slope = \frac{\mathrm{d}x}{\mathrm{d}t}$$

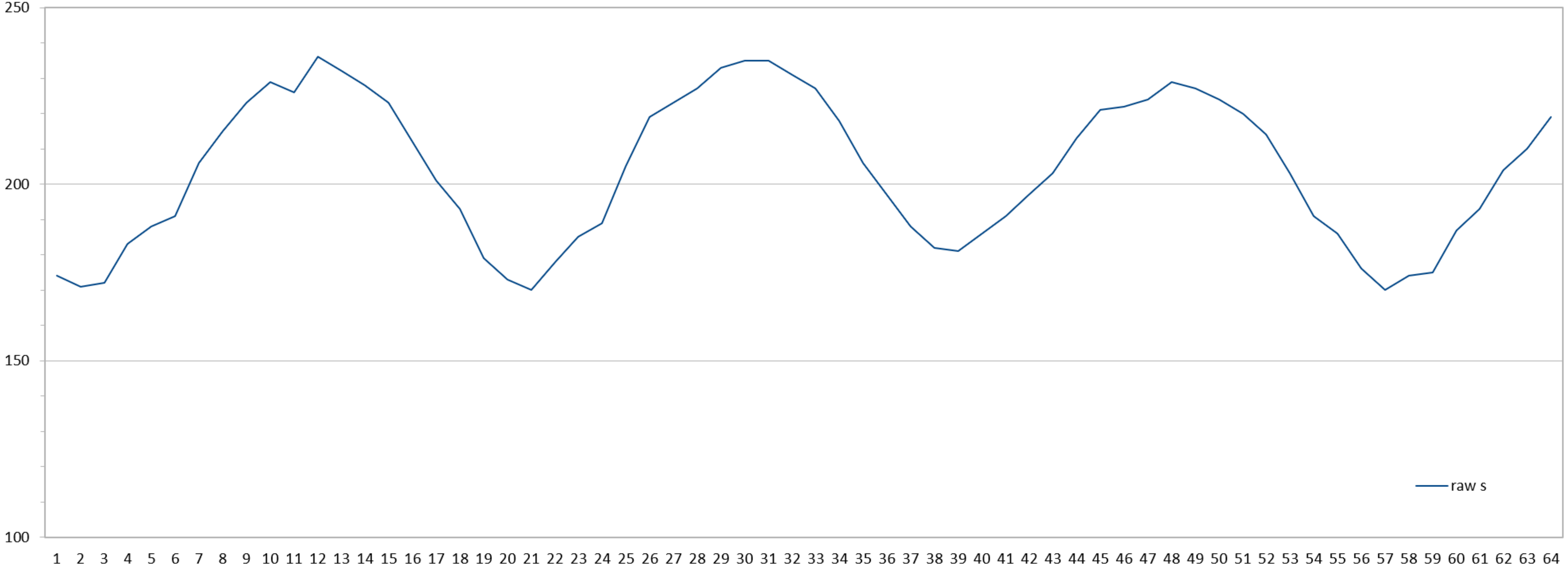


$$slope = \frac{\Delta x}{\Delta t}$$

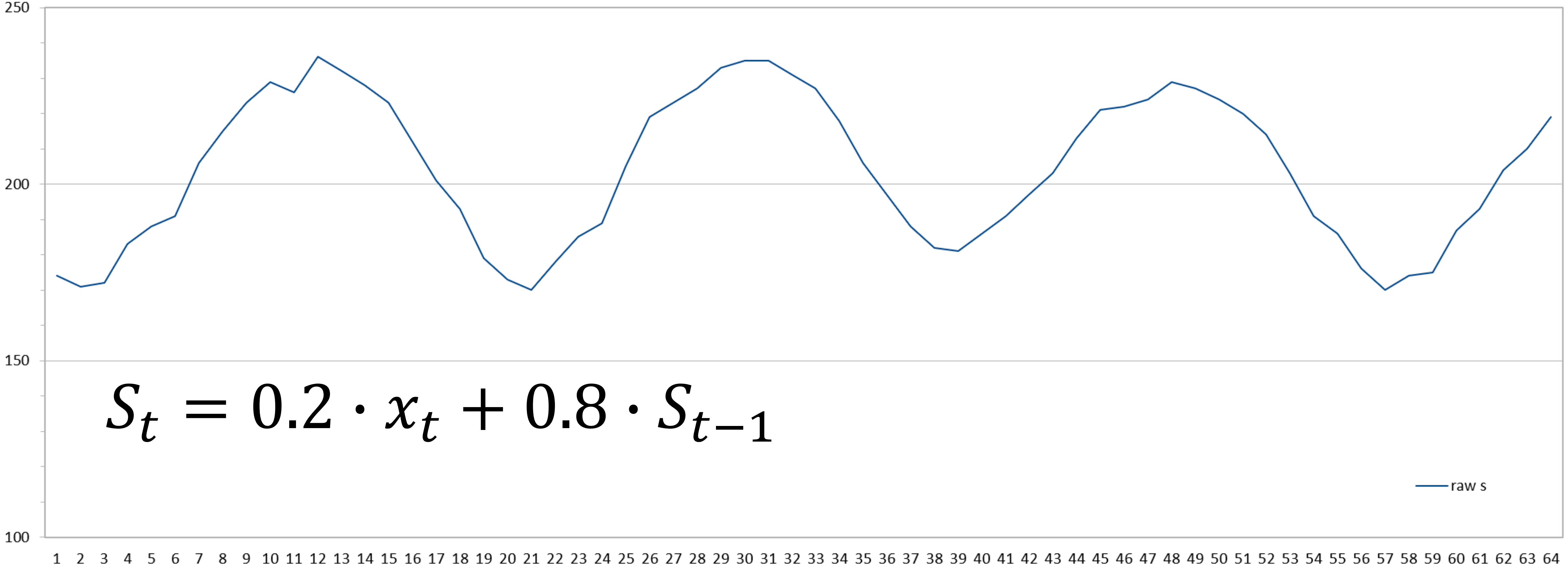
$$slope_n = \frac{x_\mathrm{n} - x_{n-1}}{t_n - t_{n-1}}$$

$$slope'_n = x_n - x_{n-1}$$

Exponential Smoothing



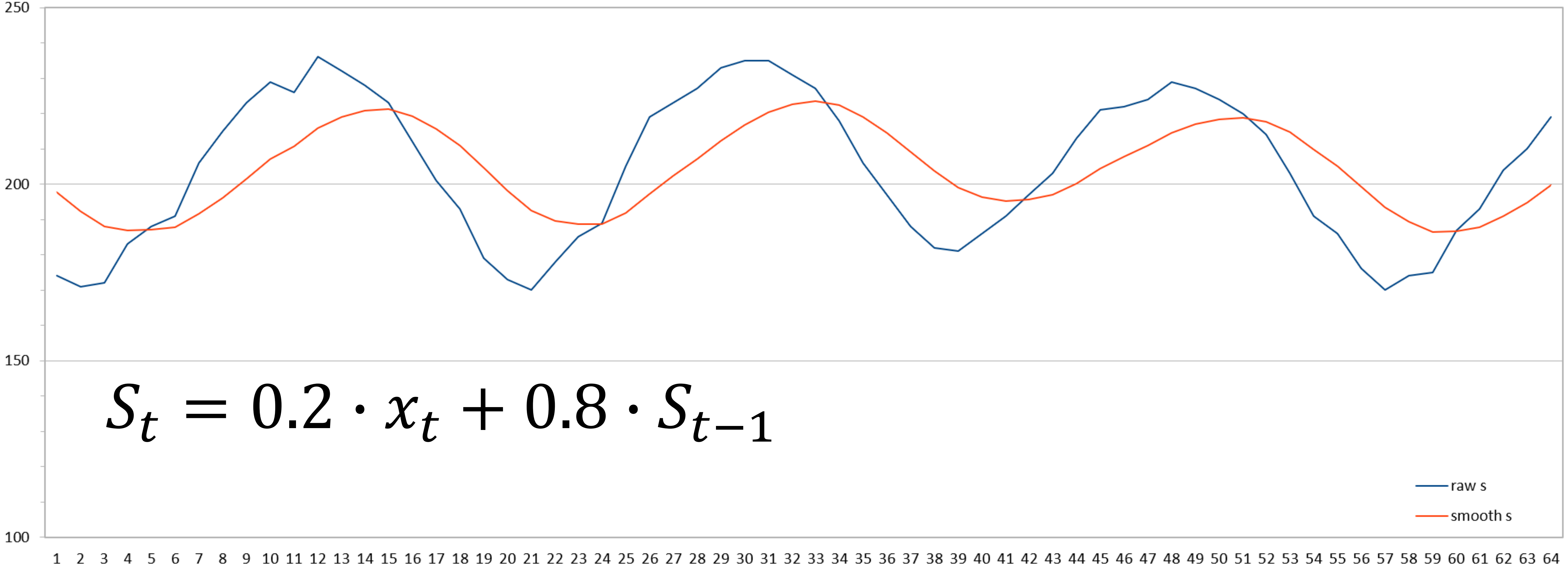
Exponential Smoothing



$$S_t = 0.2 \cdot x_t + 0.8 \cdot S_{t-1}$$

EMBECOSM®

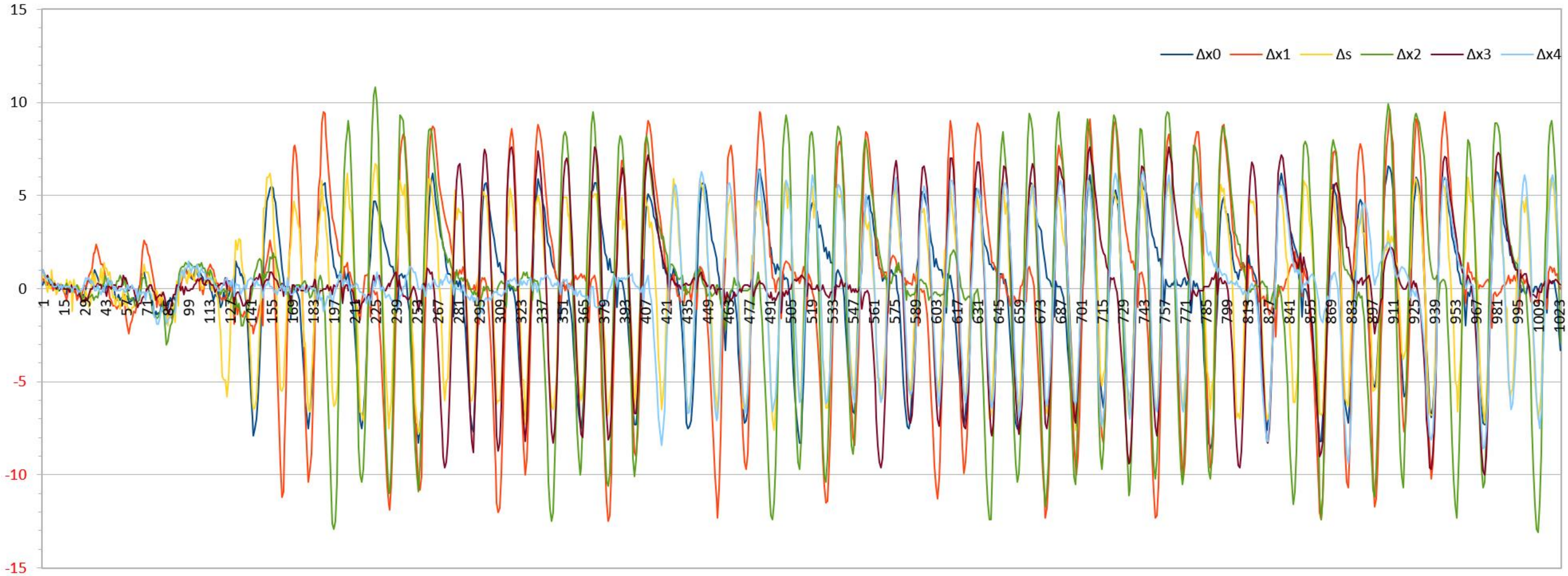
Exponential Smoothing



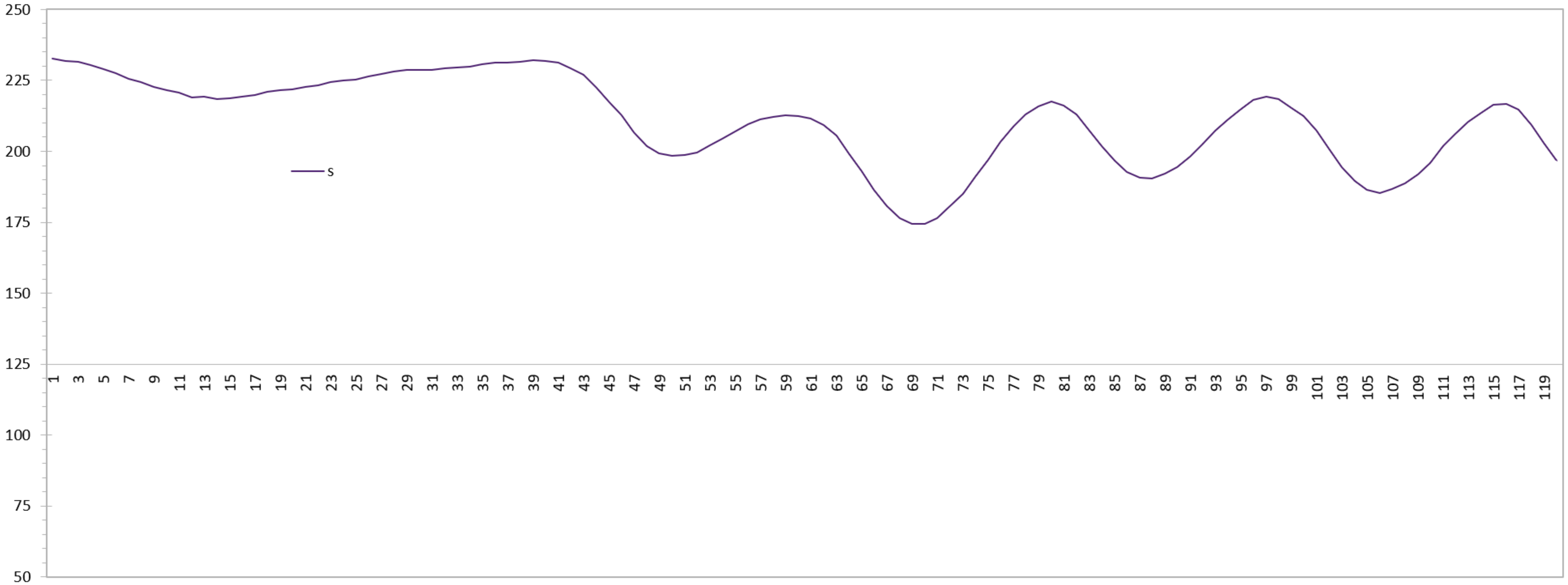
$$S_t = 0.2 \cdot x_t + 0.8 \cdot S_{t-1}$$

EMBECOSM®

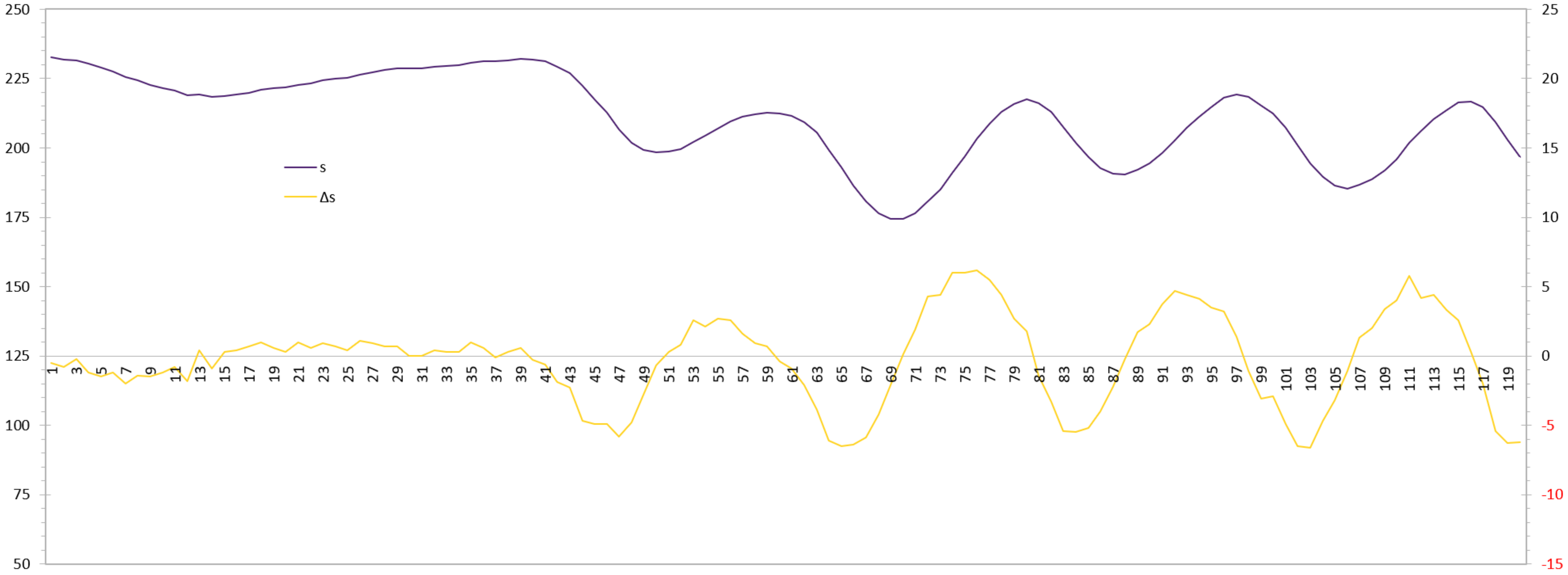
Deltas


Δx0
Δx1
Δs
Δx2
Δx3
Δx4
15
10
5
0
-5
-10
-15
1 15 29 43 57 71 85 99 113 127 141 155 169 183 197 211 225 239 253 267 281 295 309 323 337 351 365 379 393 407 421 435 449 463 477 491 505 519 533 547 561 575 589 603 617 631 645 659 673 687 701 715 729 743 757 771 785 799 813 827 841 855 869 883 897 911 925 939 953 967 981 995 1009 1023

Sprocket

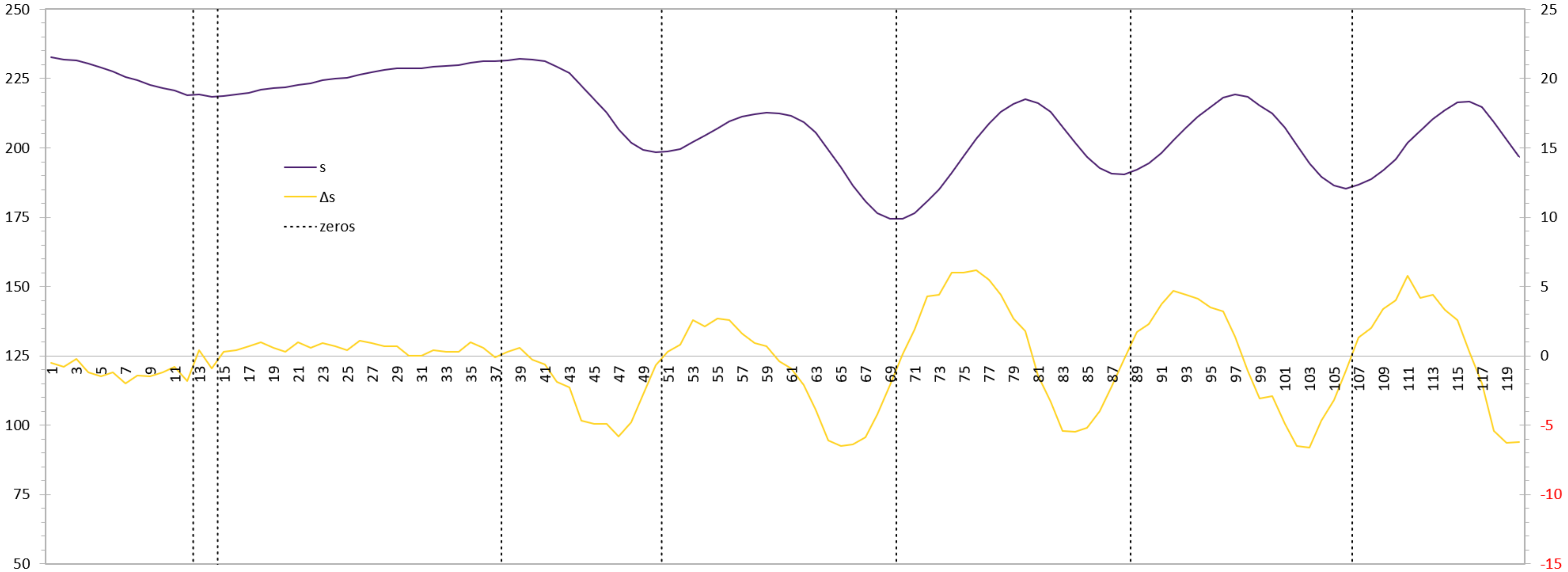


Sprocket

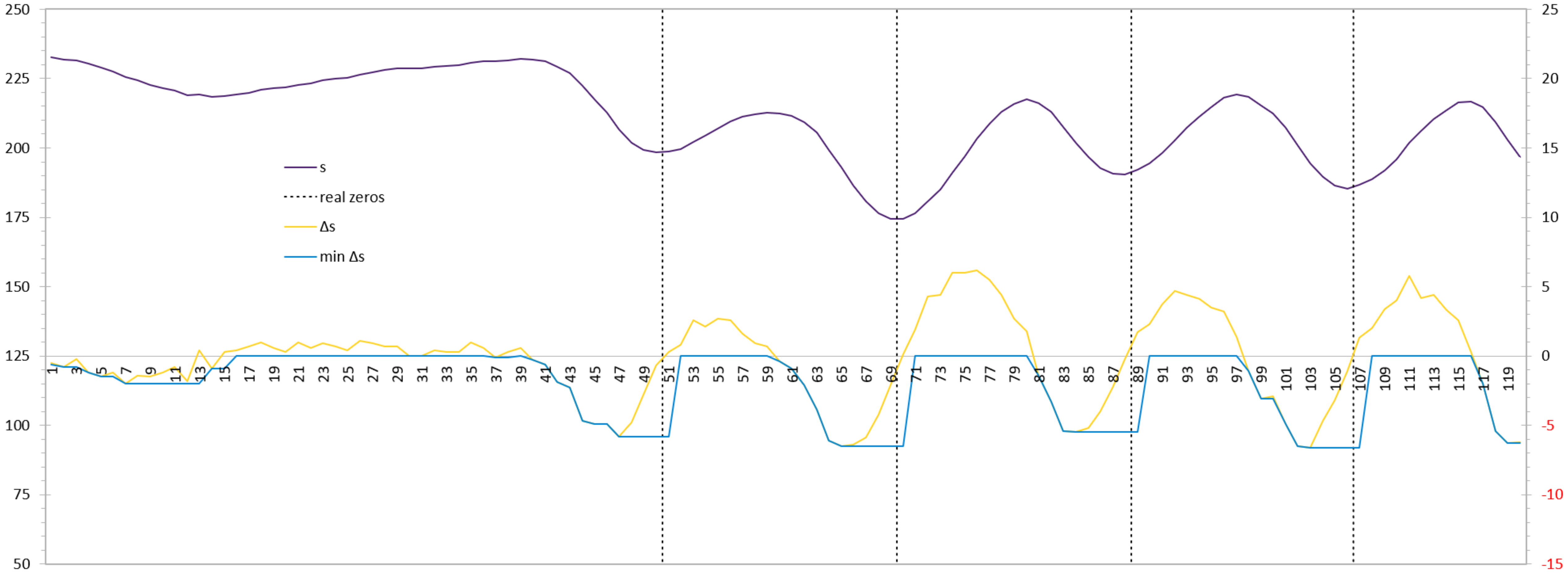


250
225
200
175
150
125
100
75
50
25
20
15
10
5
0
-5
-10
-15
s
Δs
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
53
55
57
59
61
63
65
67
69
71
73
75
77
79
81
83
85
87
89
91
93
95
97
99
101
103
105
107
109
111
113
115
117
119

Sprocket

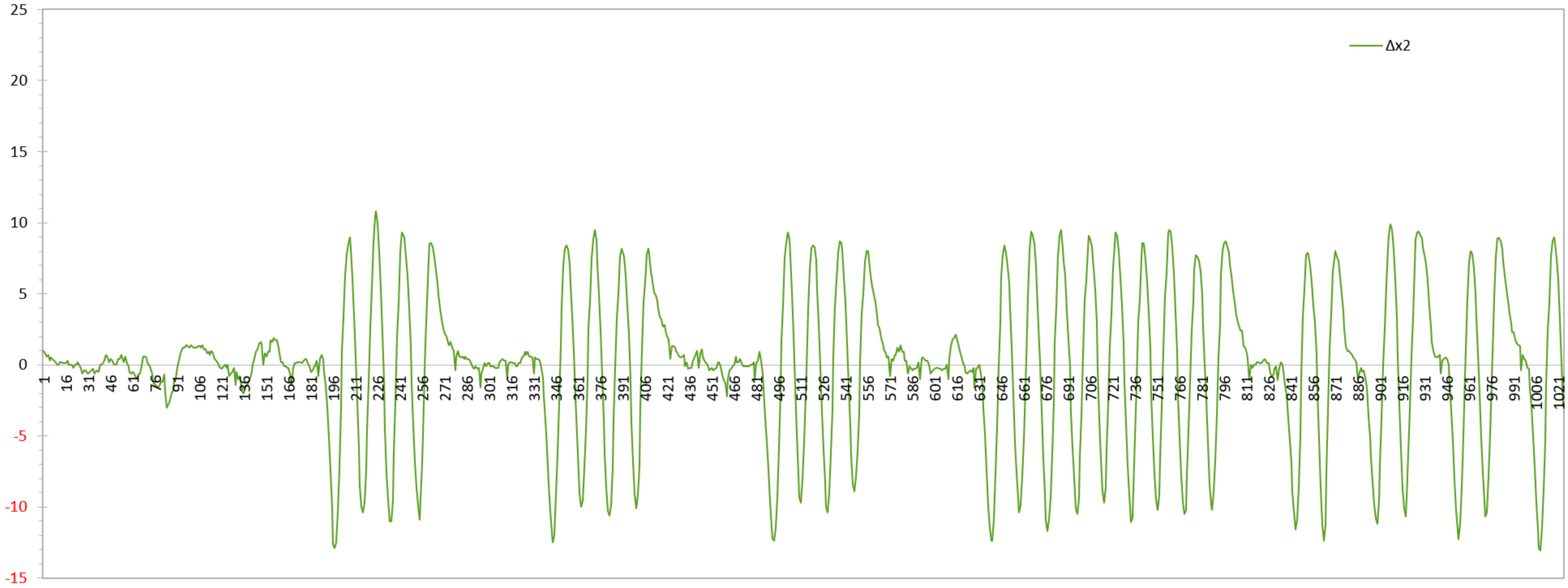


Sprocket



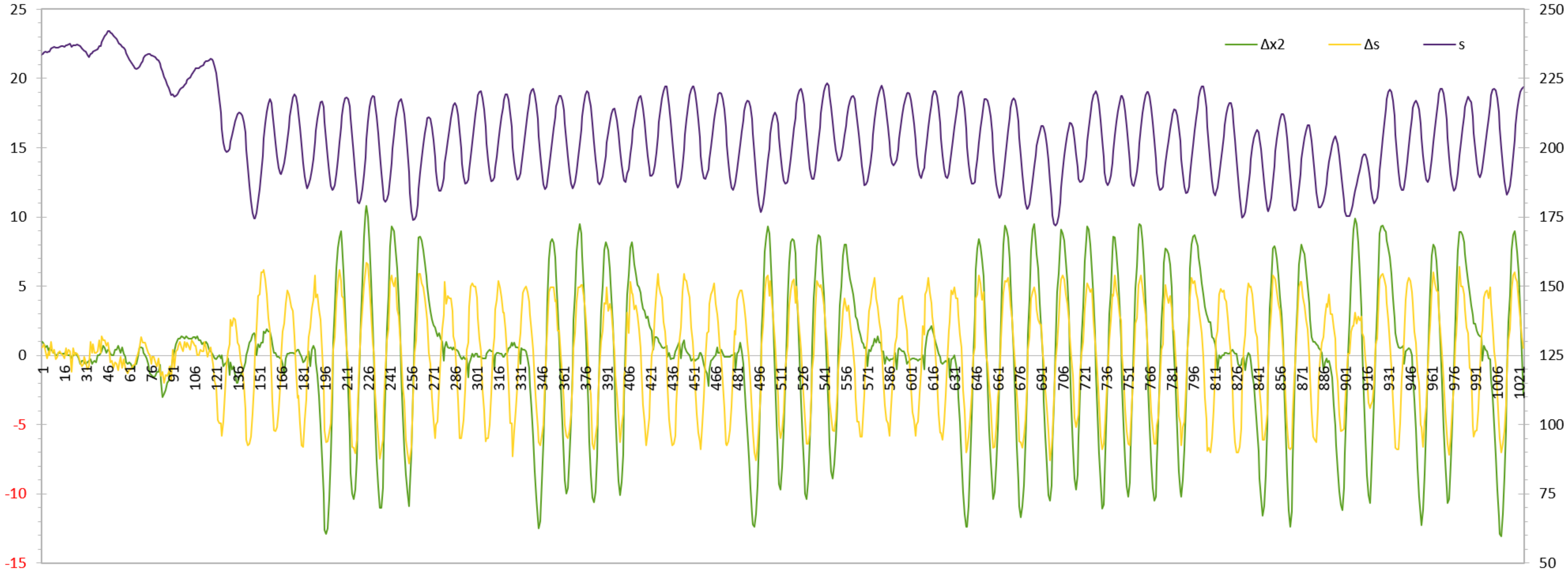
EMBECOSM®

Data Bit 2



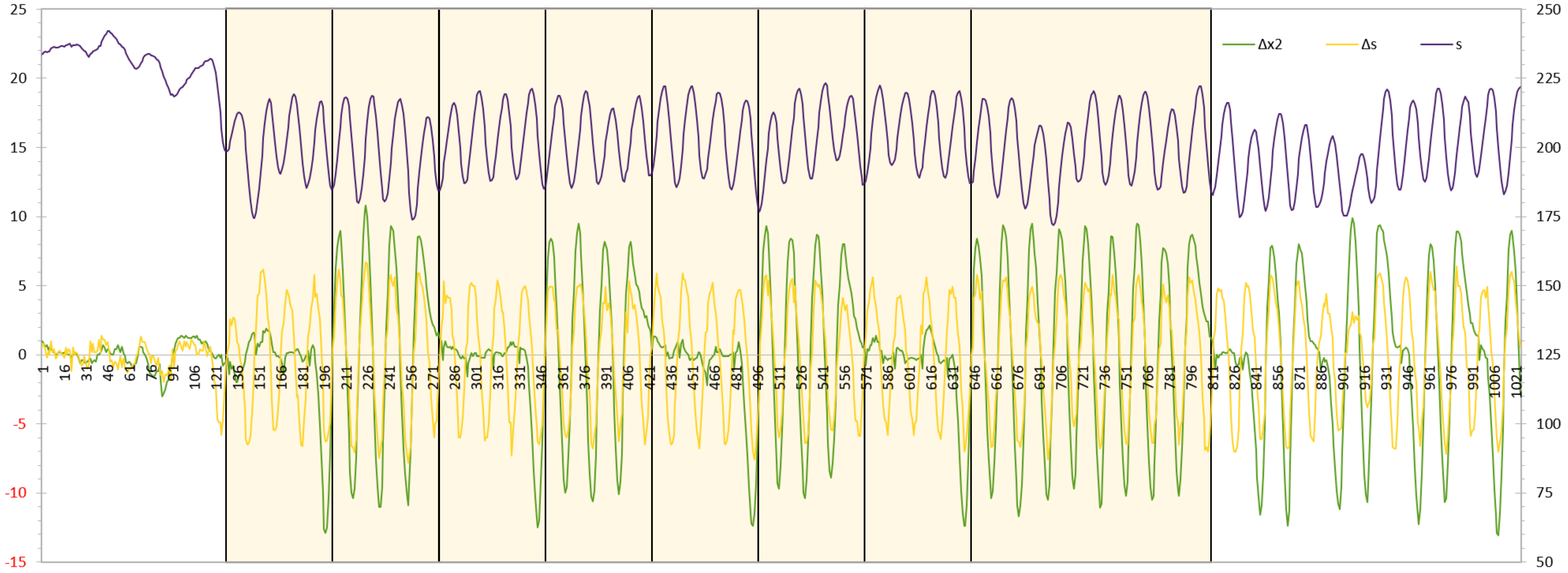
EMBECOSM®

Data Bit 2



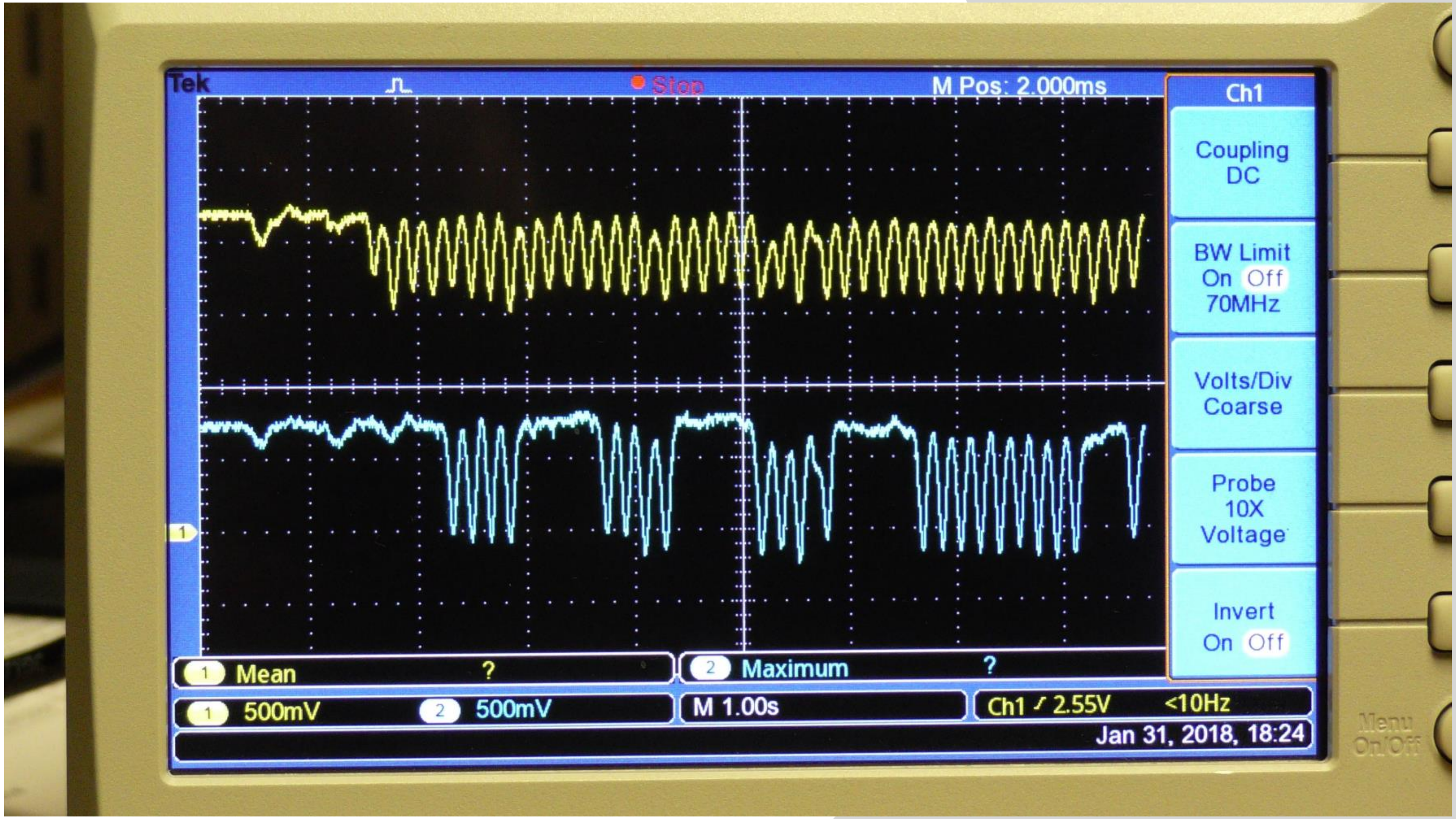
EMBECOSM®

Data Bit 2



EMBECOSM®

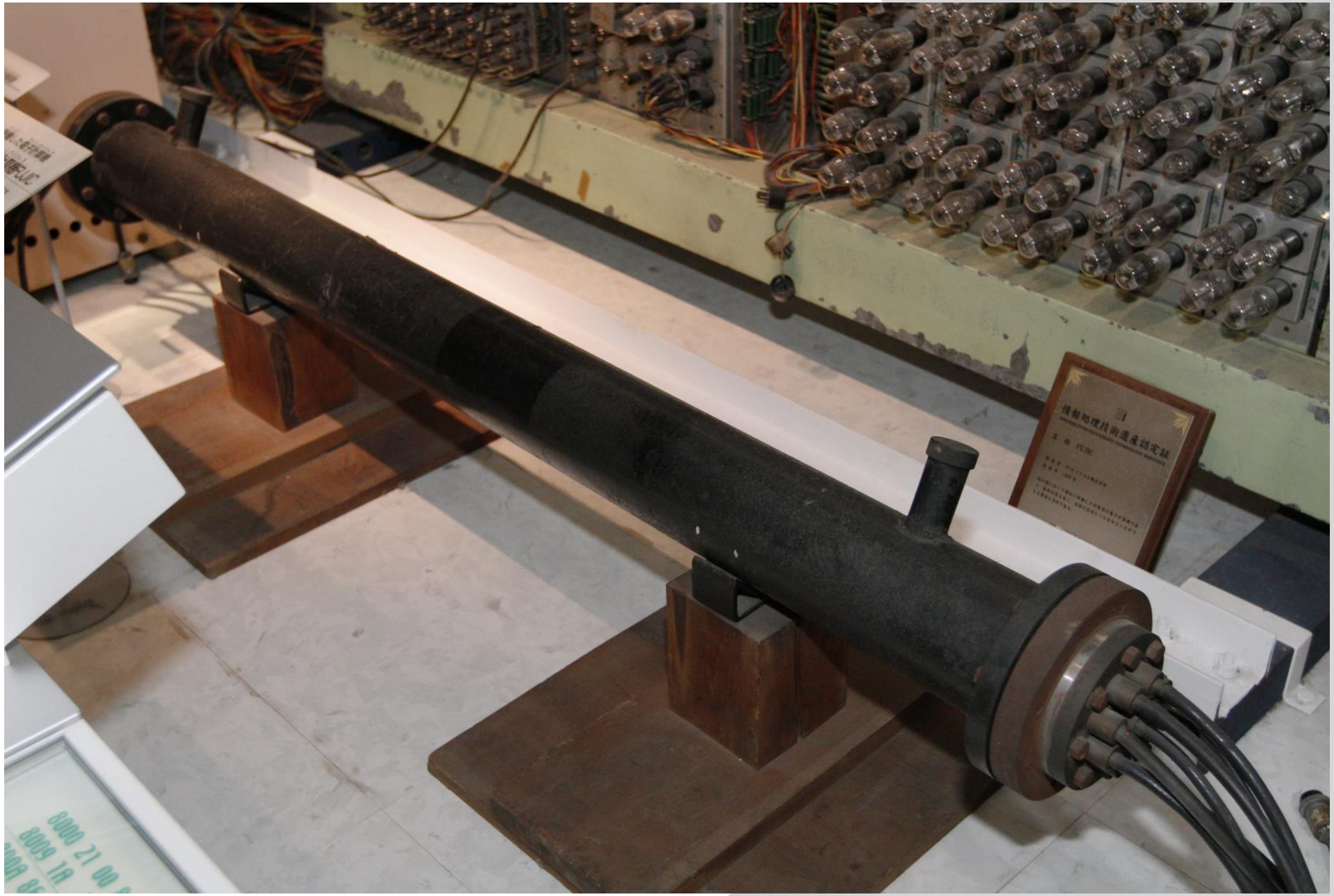
Paper Tape Reader Output



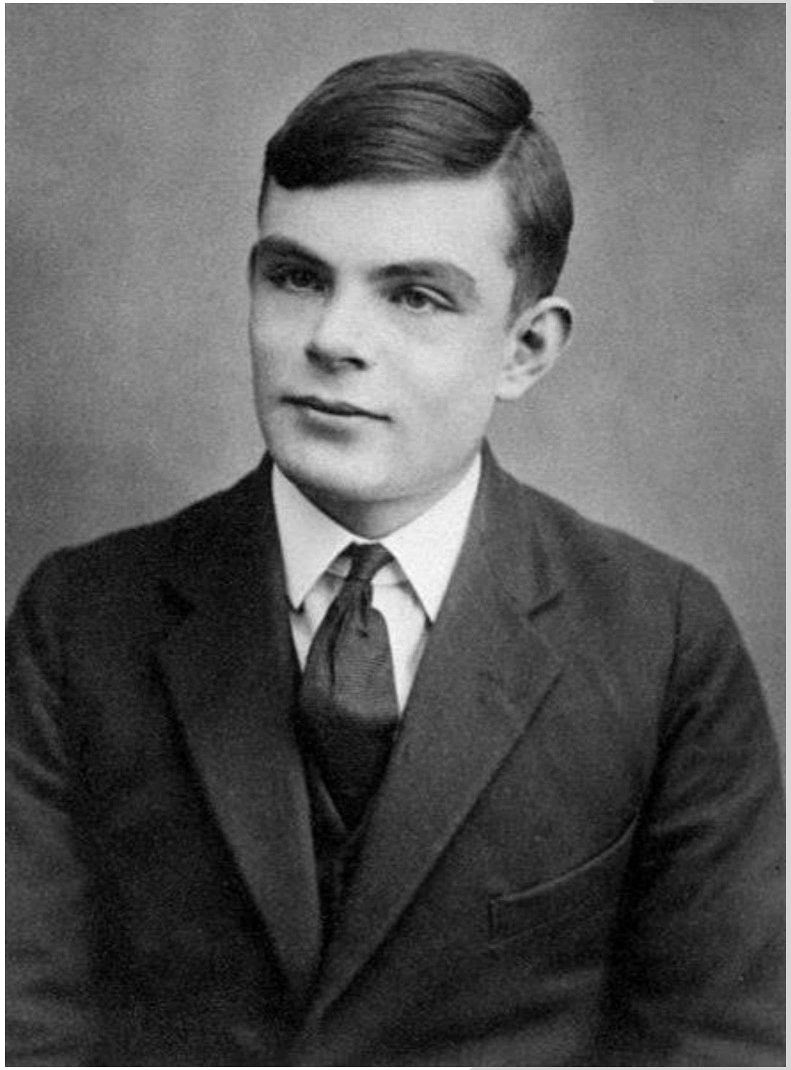
EMBECOSM®

Paper Tape Reader Demo

Mercury Delay Line



Delay Line Fluids



Delay Line Fluids

EMBECOSM®

Delay Line Fluids

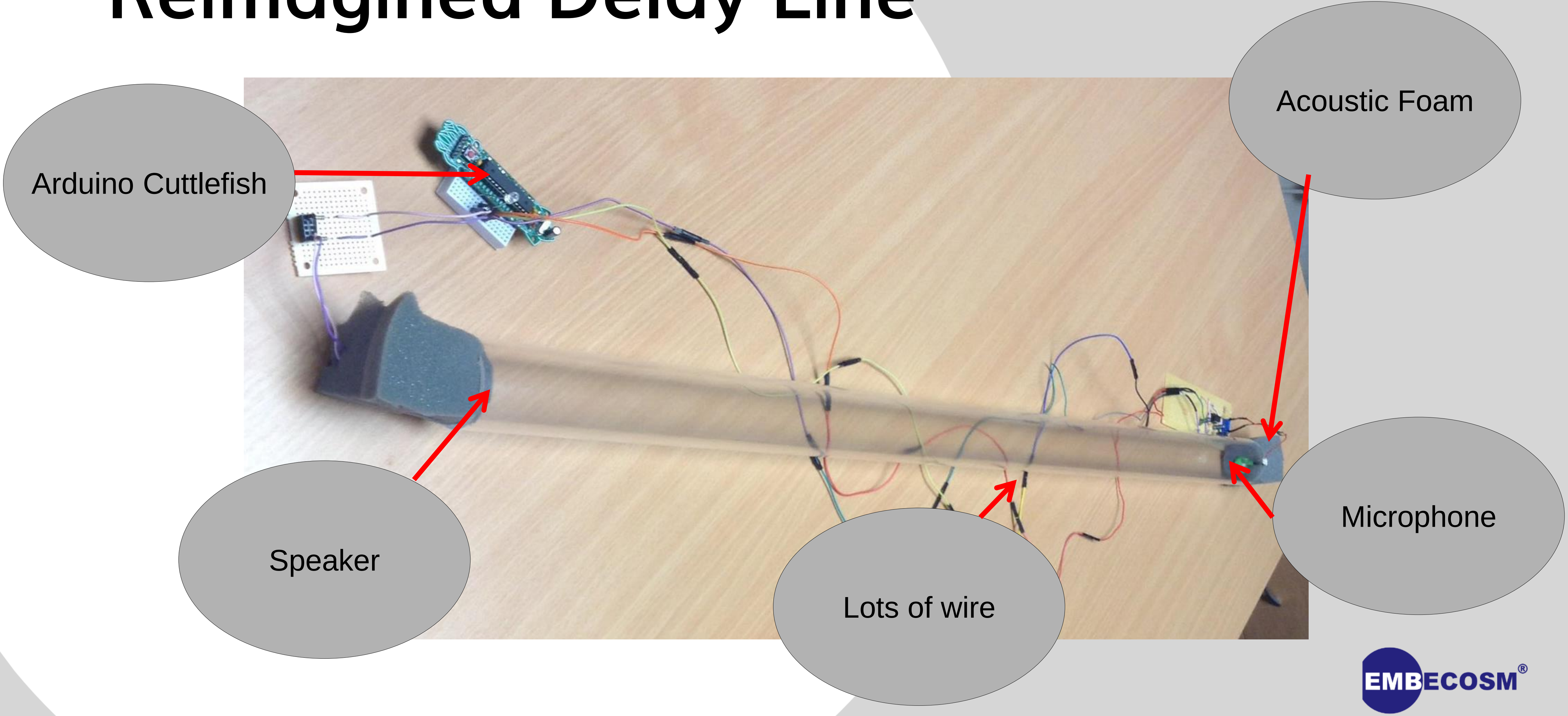
Delay Line Fluids

Delay Line Fluids

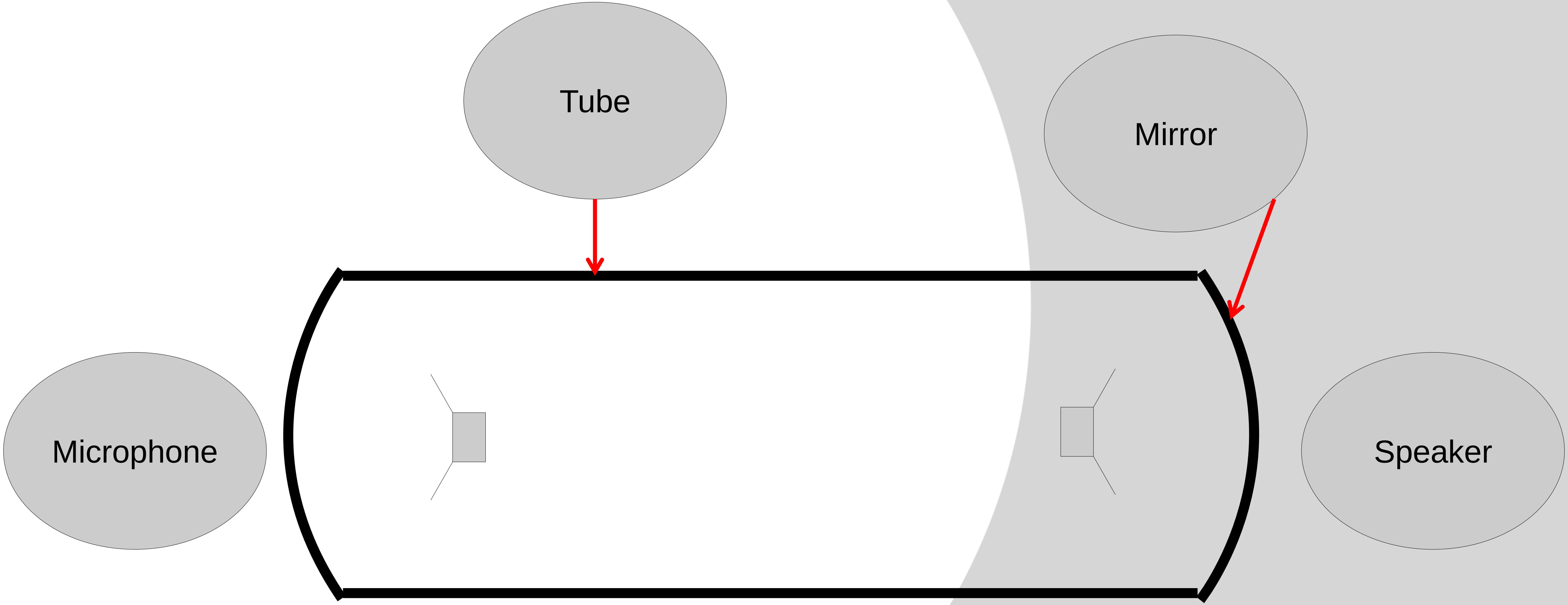
EMBECOSM®

Delay Line Fluids

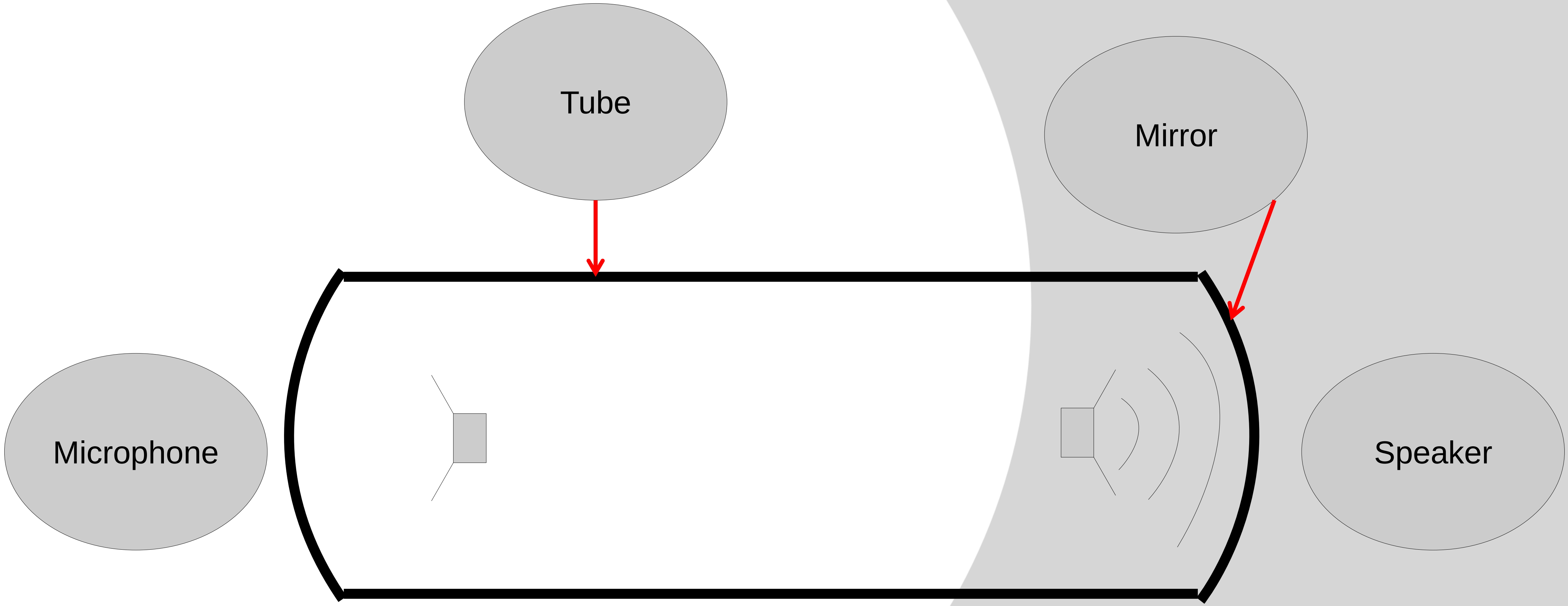
Reimagined Delay Line



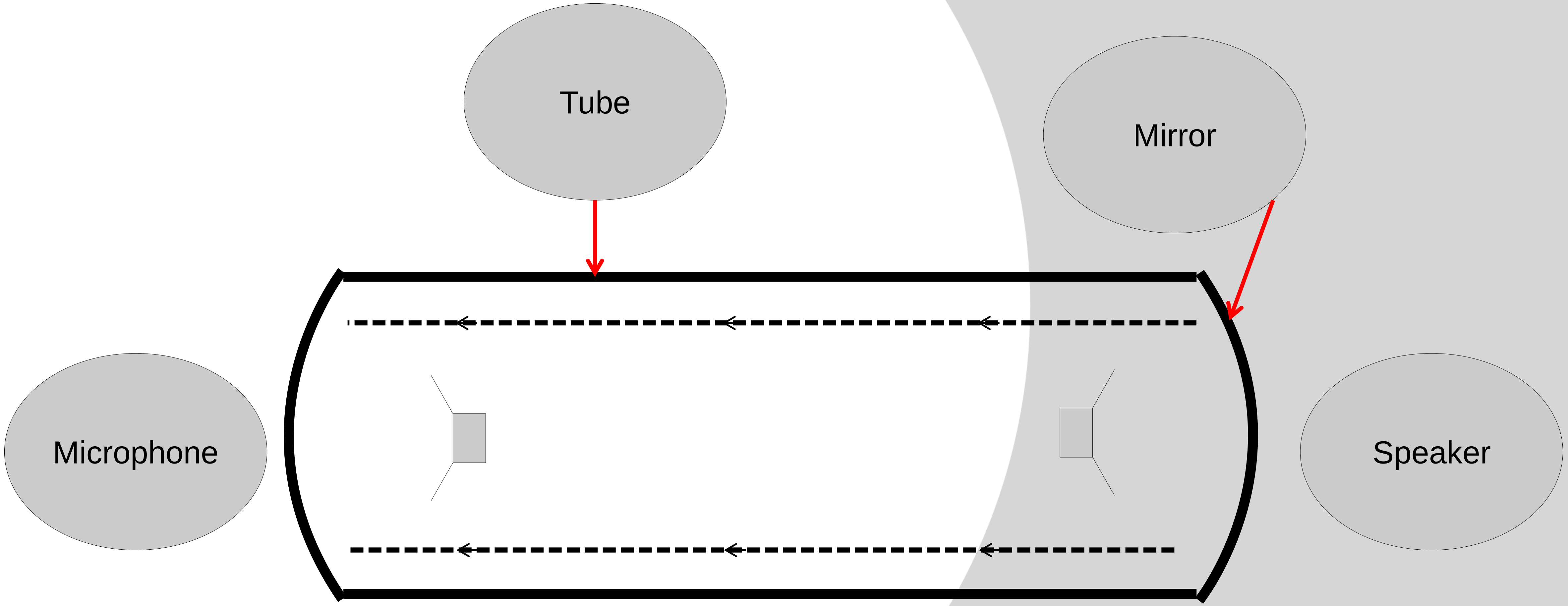
Mirrors



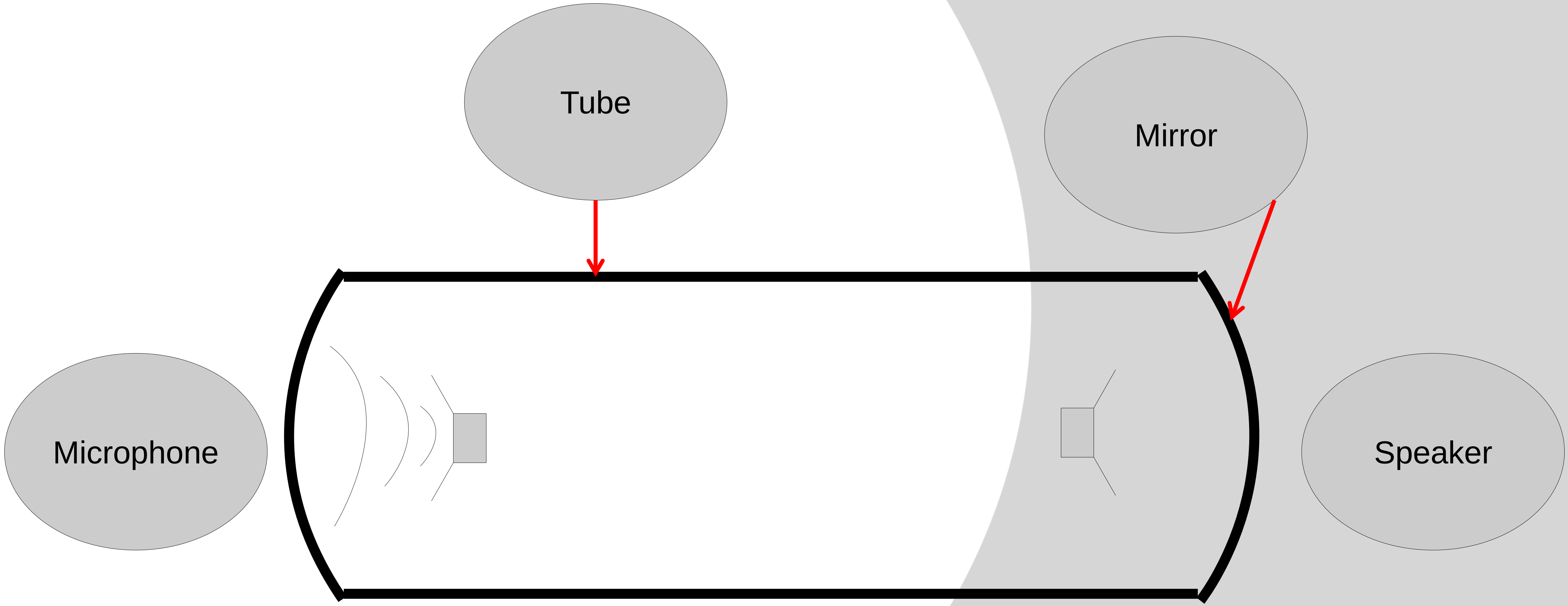
Mirrors



Mirrors

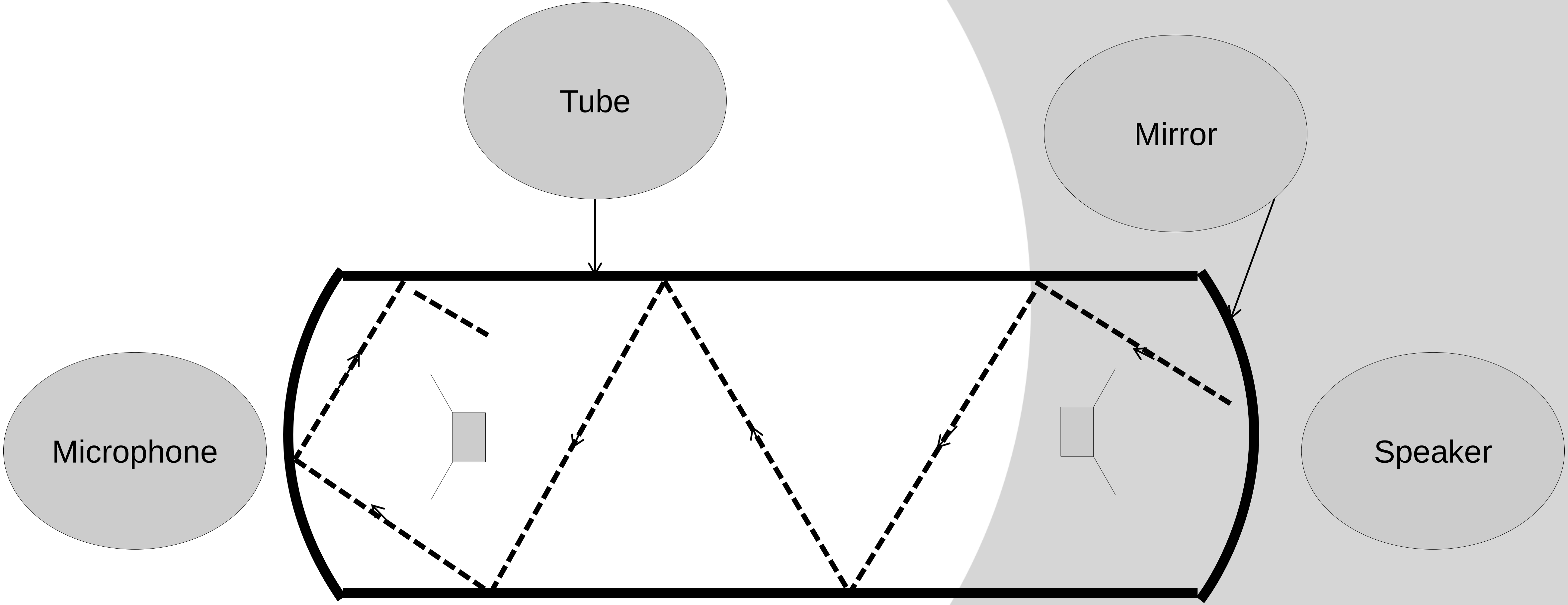


Mirrors

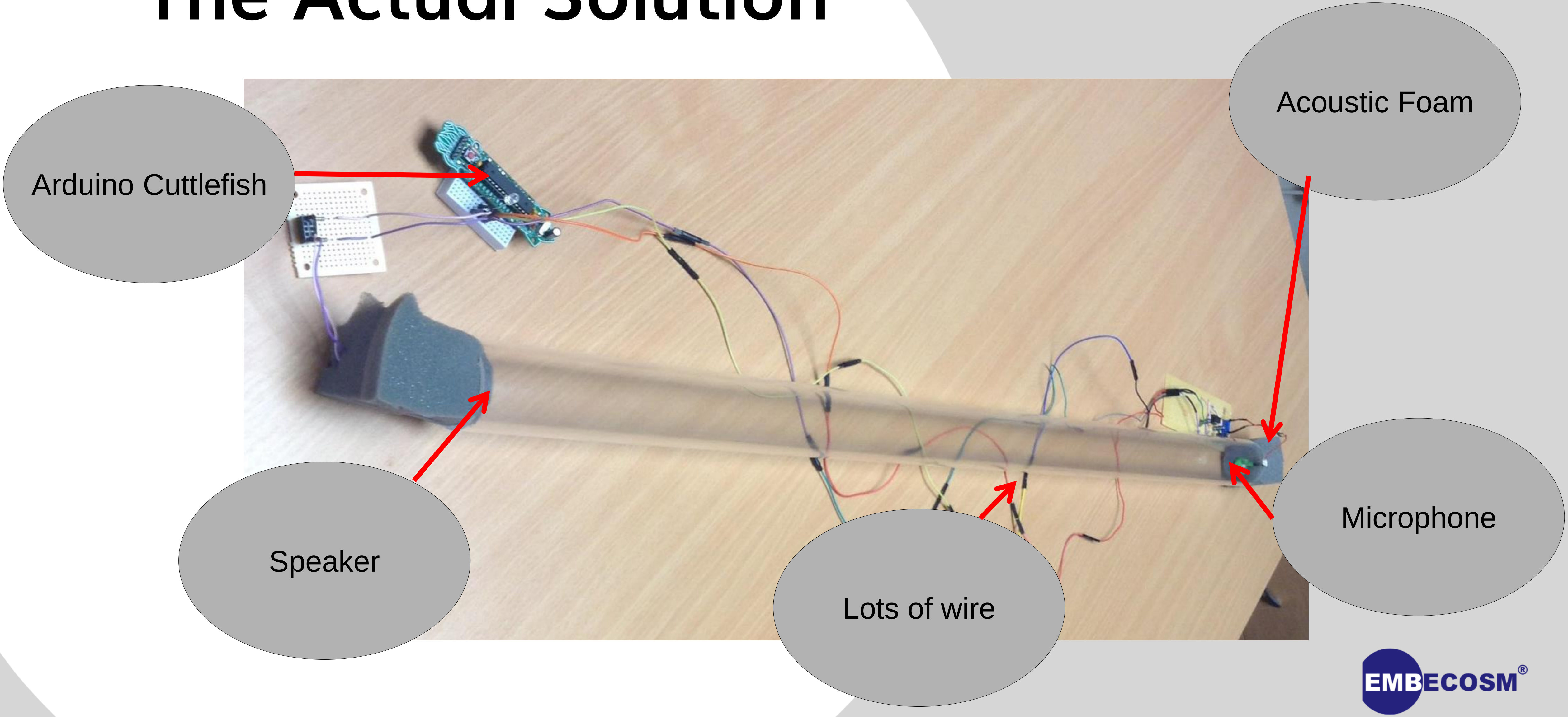


EMBECOSM®

Mirrors

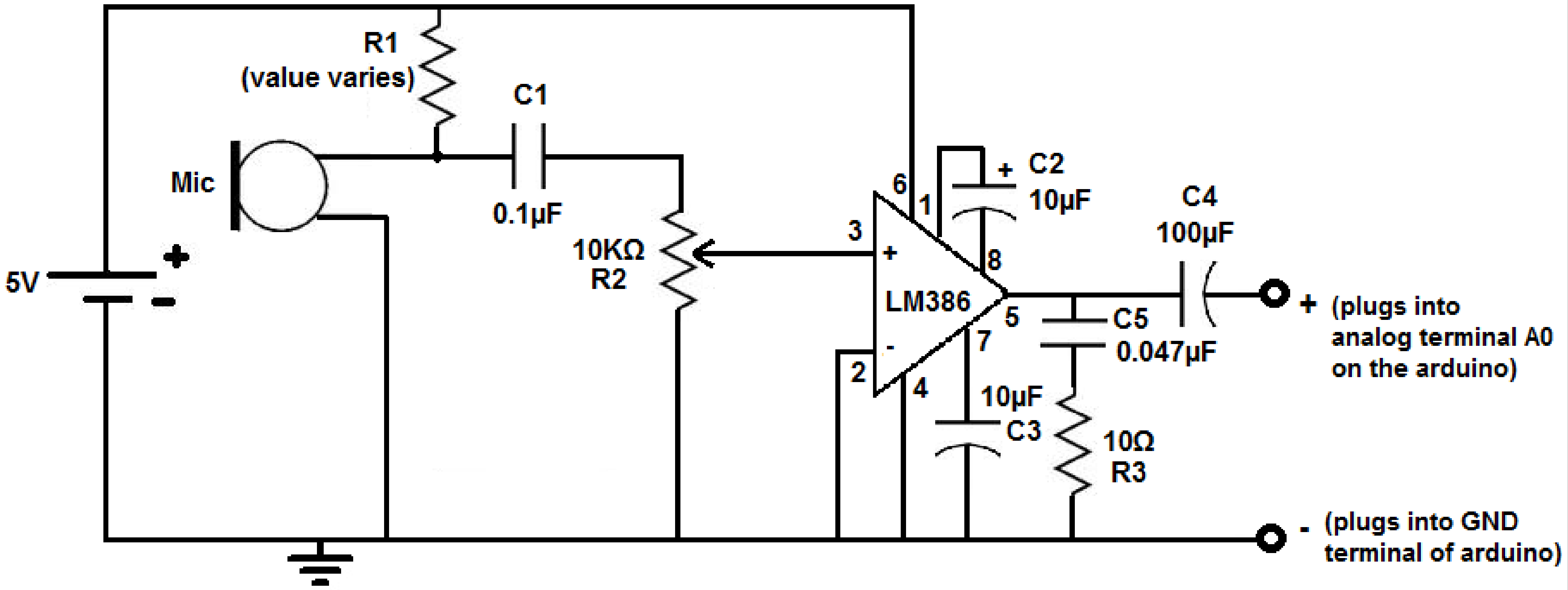


"The Actual Solution"

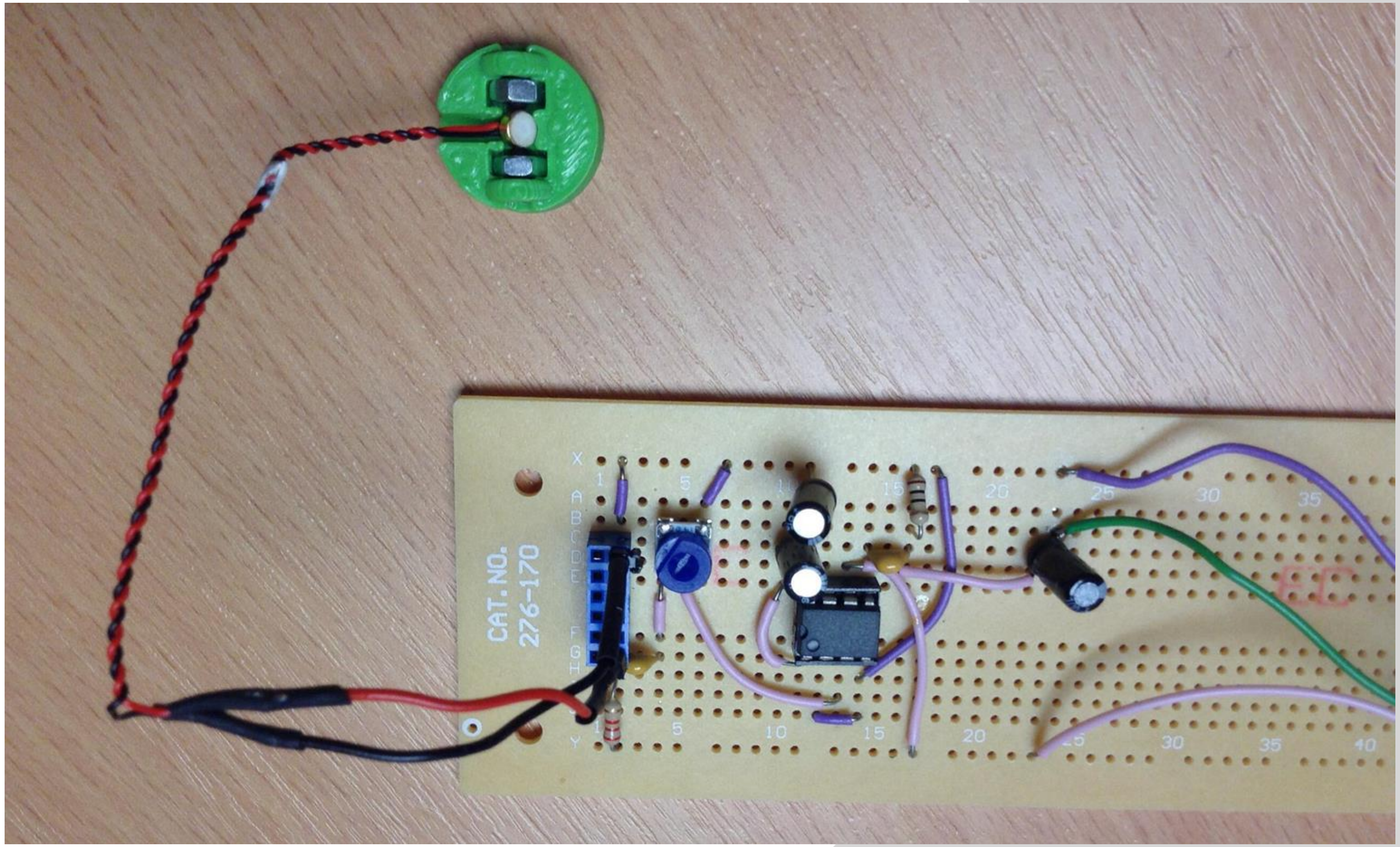


EMBECOSM®

Microphone Circuit

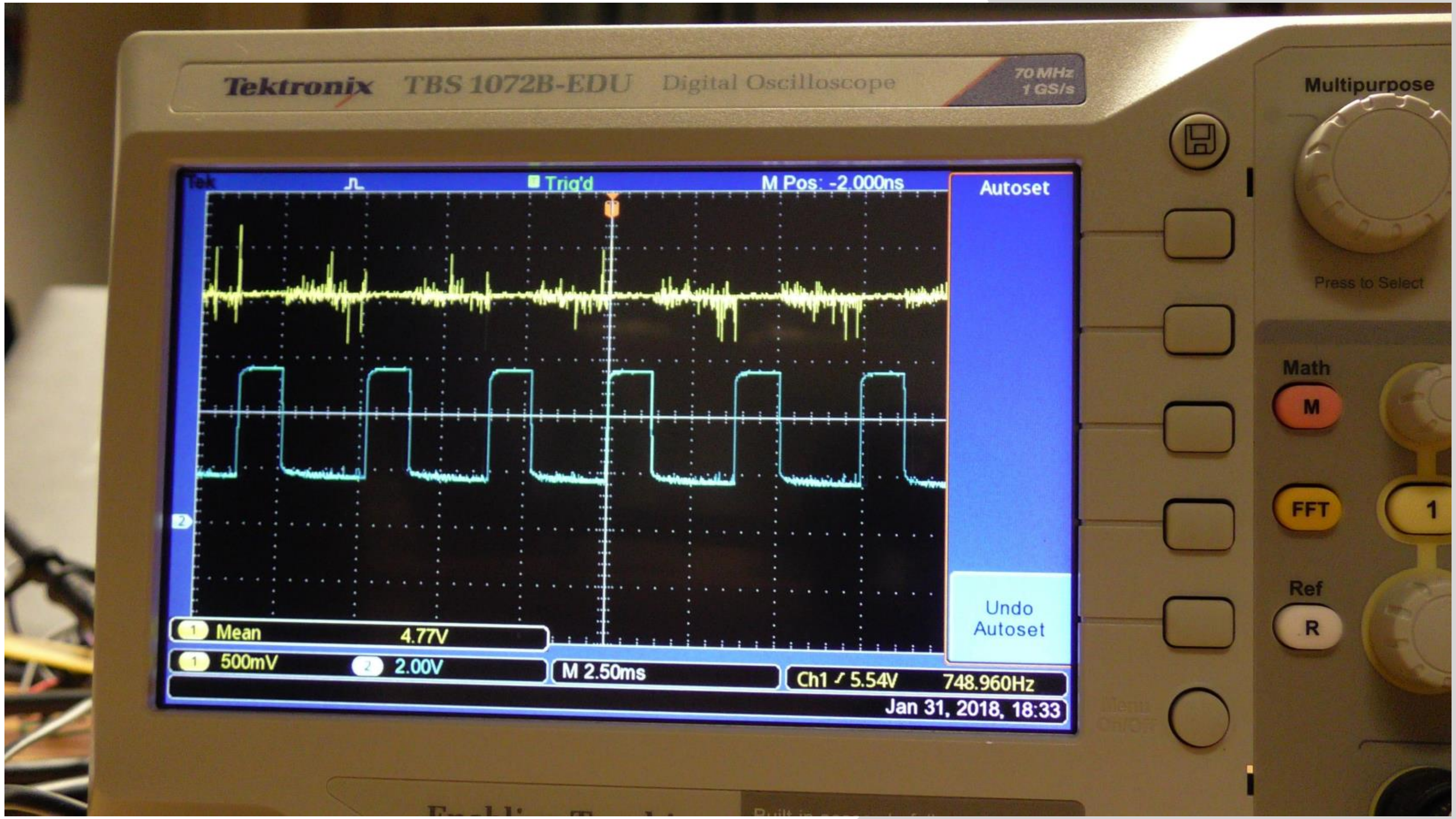


Microphone Circuit

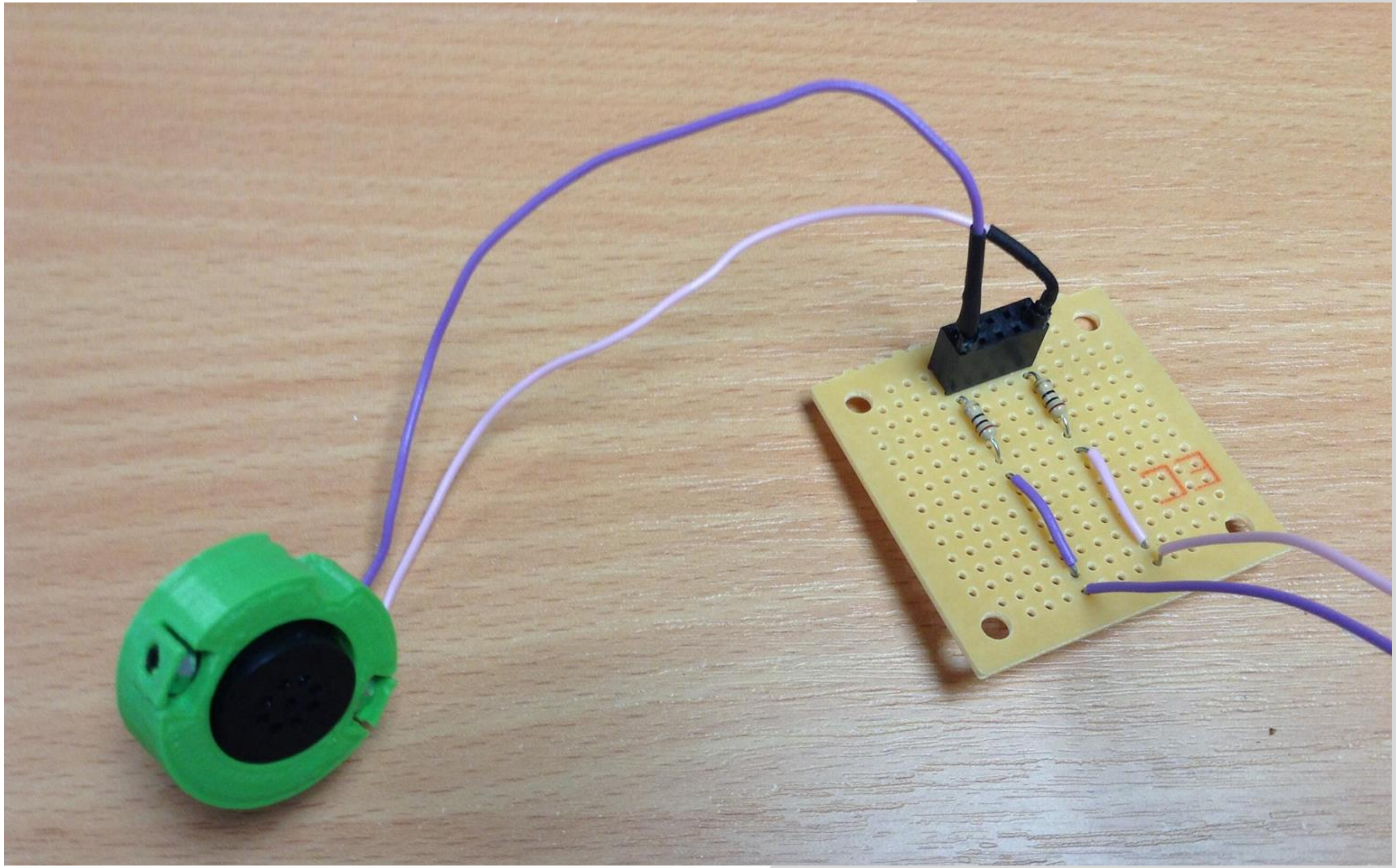


EMBECOSM®

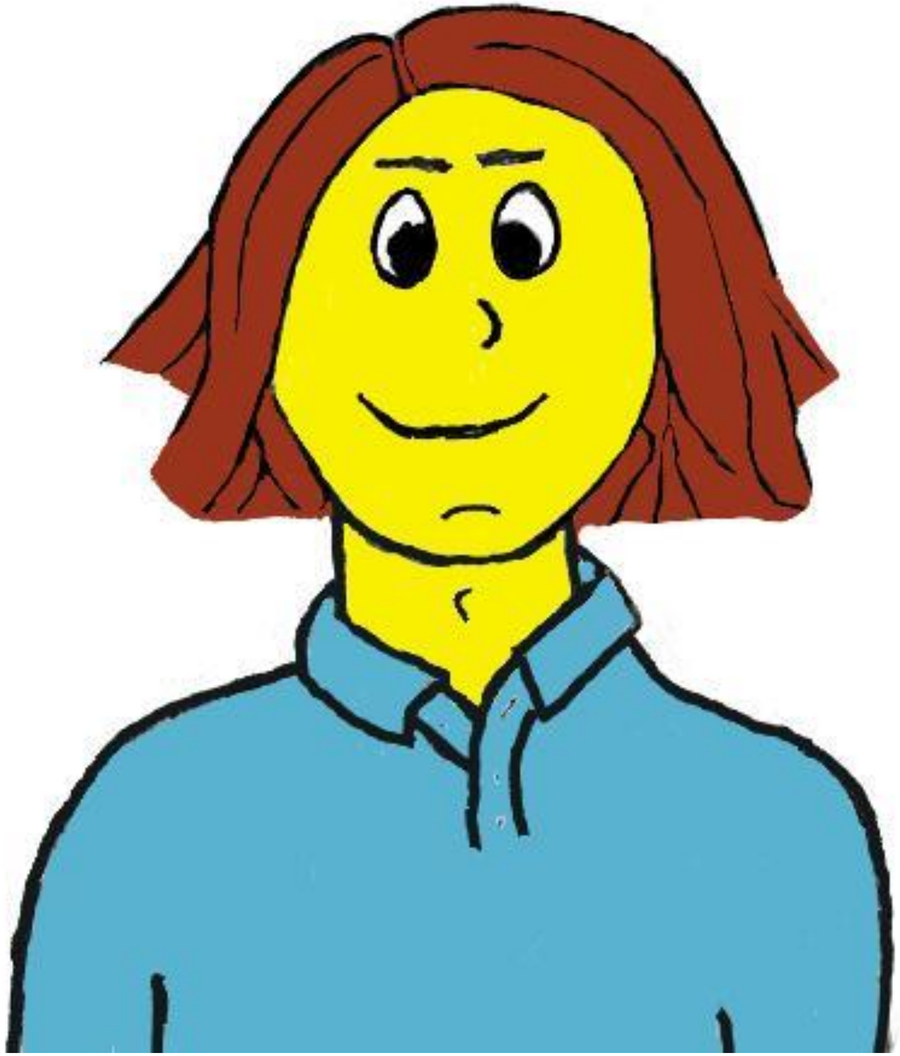
Testing the Microphone Circuit



Speaker Circuit



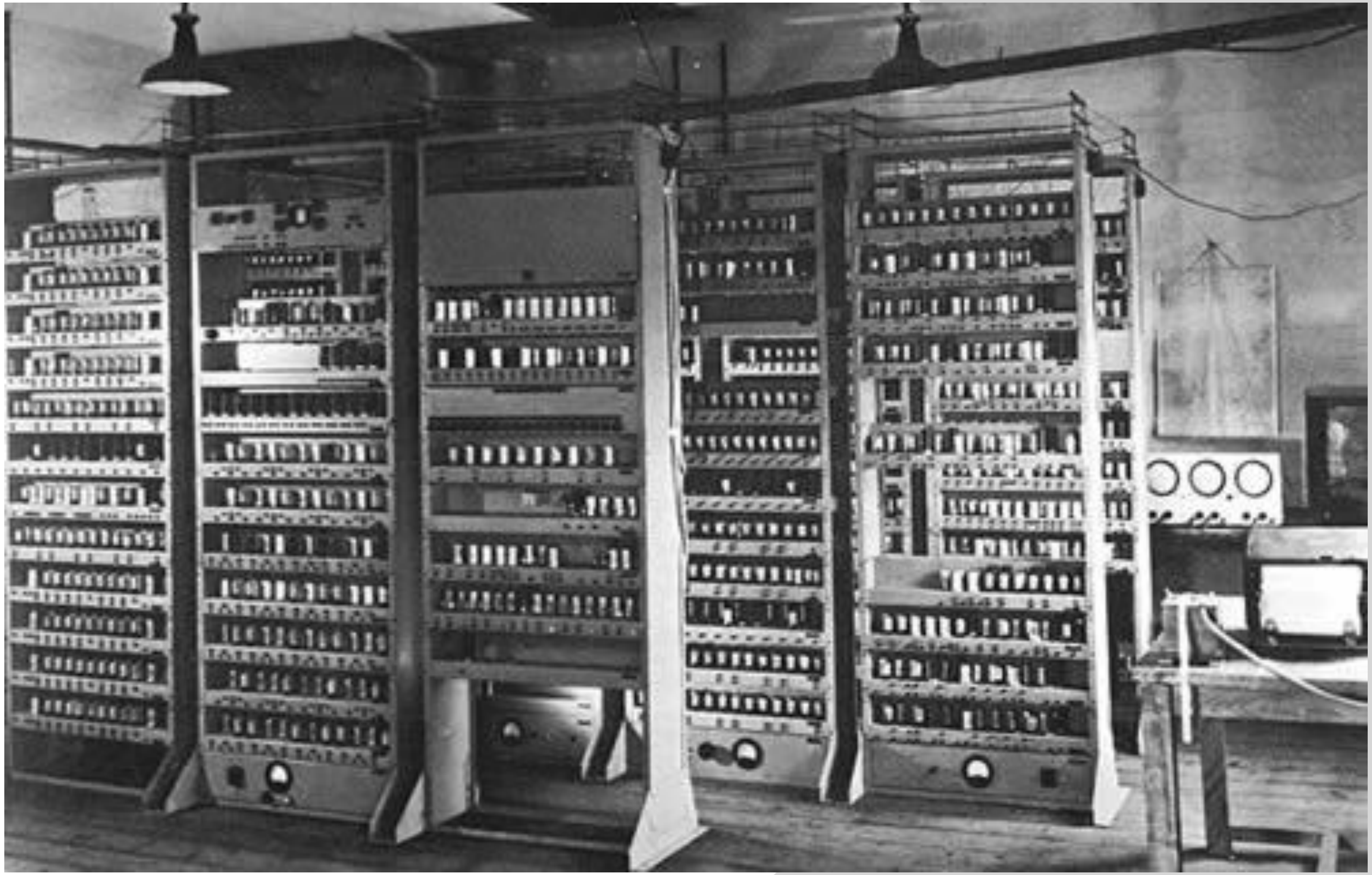
Air Delay Line Demo



Chip Hack

EMBECOSM®

Summary



Thank You

Contact

- Make your own EDSAC peripherals:
  - github.com/embecosm/edsac-peripherals
- Hatim Kanchwala Verilog EDSAC:
  - github.com/librecores/gsoc-museum-edsac
- Chip Hack repository:
  - github.com/embecosm/chiphack
- More information on the myStorm:
  - mystorm.uk

Questions?

www.embecosm.com