Primitive processor by Eric Shelbogashev https://github.com/EricShelbogashev and Michael Kulakov

Used instruction set

Binary code	Short name	Register	Description
001	st	A	Write to [A] from [Temp]
010	ld	A	Load from [A] to [Temp]
011	add	A	Add [A] to [Temp]
100	cmp	A	Add 1 to counter if [A] > [Temp]; Add 2 to counter if [A] <= [Temp]
101	jmp	A	Replace value in [the Instruction Pointer] by [A]
110	halt		Stop all operations

Inx	Short Name	Reg	OP Description	Glob Description
0	ld	30	Load from 30 to [Temp]	Summary of [30] and [29]
1	add	29	Add 29 to [Temp]	
2	st	31	Write to 31 from [Temp]	Store sum result to [31]
3	ld	30	Load from 30 to [Temp]	Move second value to first place
4	st	29	Write from [Temp] to 29	
5	ld	31	Load from 31 to [Temp]	Move sum result to second place
6	st	30	Write from [Temp] to 30	
7	ld	28	Load from 28 to [Temp]	Incrementing the iteration counter
8	add	27	Add “1” from 27 to [Temp]	
9	st	28	Write from 28 to [Temp]	
10	cmp	32	Go to next if [Temp] < 32; Go to next to next if [Temp] <= 32	Compare [the iteration counter] with [N]
11	jmp	0	Replace value in [the Instruction Pointer] by 0	Jump to 0 if compare is [True]; Halt if compare is [False];
12	ld	30	Load from 30 to [Temp]	
13	st	20	Write from [Temp] to 20	
14	halt		Stop all operations	
…	…			
27	1			_Const
28	i=0			_IterationCounter
29	xpp=1			_FirstValue
30	xp=1			_SecondValue
31	x			_SumResult
32	N=10			[Input data] – Fibonacci’s numbers count
…	…			
20	y			Peripheral module memory

Format	Description
5-bit value	[Instruction Pointer]
8-bit value	[Temp]