Shift-add-only exponential for the next generation SpiNNaker system

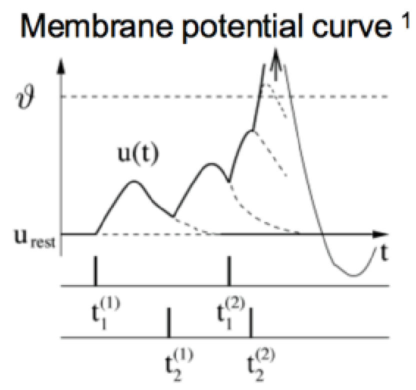
Mantas Mikaitis
mikaitis@cs.man.ac.uk

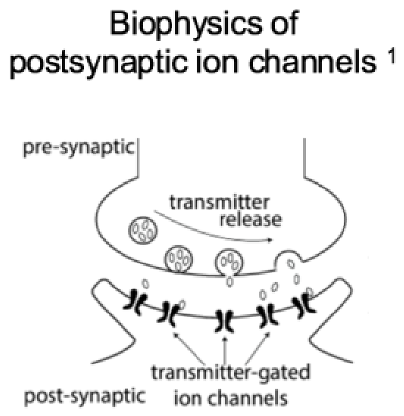
Supervisor Dr David R. Lester
david.r.lester@manchester.ac.uk

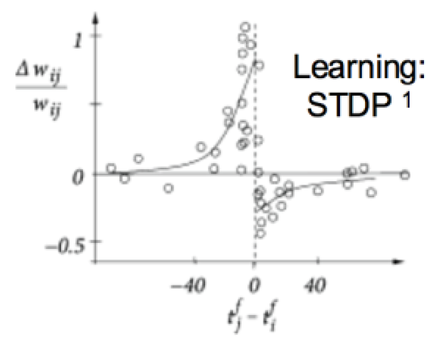
Co-Supervisor Prof. Steve Furber
steve.furber@manchester.ac.uk

1 Exponential in simulations of Spiking-Neural-Networks

Exponentially decaying values can be met in many parts of biological systems: Neuron membrane potential, biophysics of synapses, Spike-Timing-Dependent-Plasticity and more. As a result, for neuromorphic simulator designers, exponential is an important operation to have in order to simulate these phenomena accurately. In the first SpiNNaker system, exponential was designed in software which provided an easy to use but relatively slow and limited exponential. In the following sections, I demonstrate a proposal to build a fast exponential unit in hardware that would be incorporated into the next generation SpiNNaker-2 system.







2 Shift-add algorithm

The main algorithm is based on the convergence algorithm presented in Section 8 by Muller [2]. From Theorem 16 in the book, the sequences t_n and d_n defined as

$$t_0 = 0$$
$$t_{n+1} = t_n + d_n ln(1+2^{-n})$$
$$d_n = \begin{cases} 1 & \text{if } t_n + ln(1+2^{-n}) \le t \\ 0 & \text{otherwise} \end{cases}$$

satisfy

$$\lim_{n\to\infty} t_n = t = \sum_{n=0}^{\infty} d_n ln(1+2^{-n}).$$

Now a sequence E_n is defined such that at any step n of the algorithm,

$$E_n = exp(t_n).$$

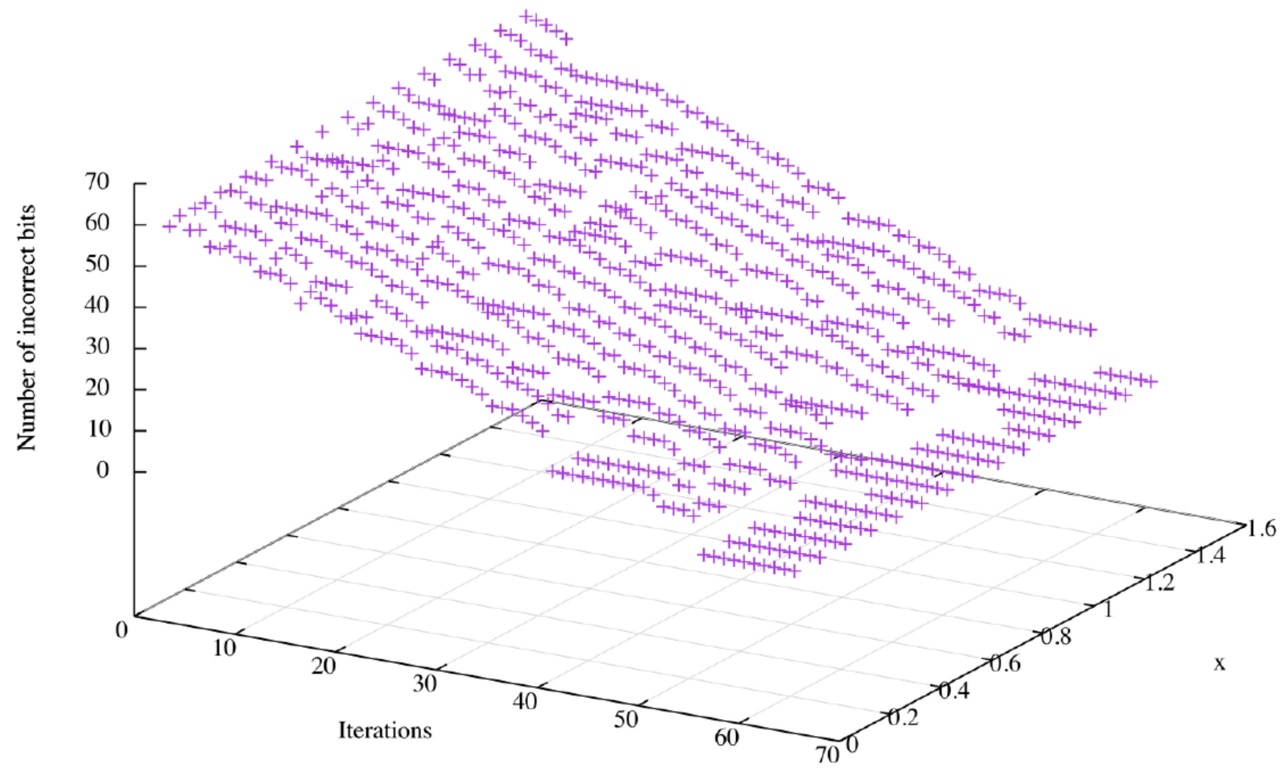
Since $t_0 = 0$, E_0 is initialised as 1. When $d = 1$, we add $ln(1+2^{-n})$ to t_n. As a result $E_{n+1} = exp(t_{n+1}) = exp(t_n + ln(1+2^{-n})) = exp(t_n)exp(ln(1+2^{-n})) = E_n exp(ln(1+2^{-n})) = E_n(1+2^{-n}) = E_n + E_n 2^{-n}$. Then as a series dependent on d we can write

$$E_n = E_n + d_n E_n 2^{-n}$$

which requires only adder and shifter, same as t_n.

3 Accuracy

In the figure below, demonstrated is a number of incorrect bits by stopping at each step from 2 to 64. I find that the average number of iterations needed for a full 64-bit accuracy is 53.769120.



4 Natural logarithm using exp hardware

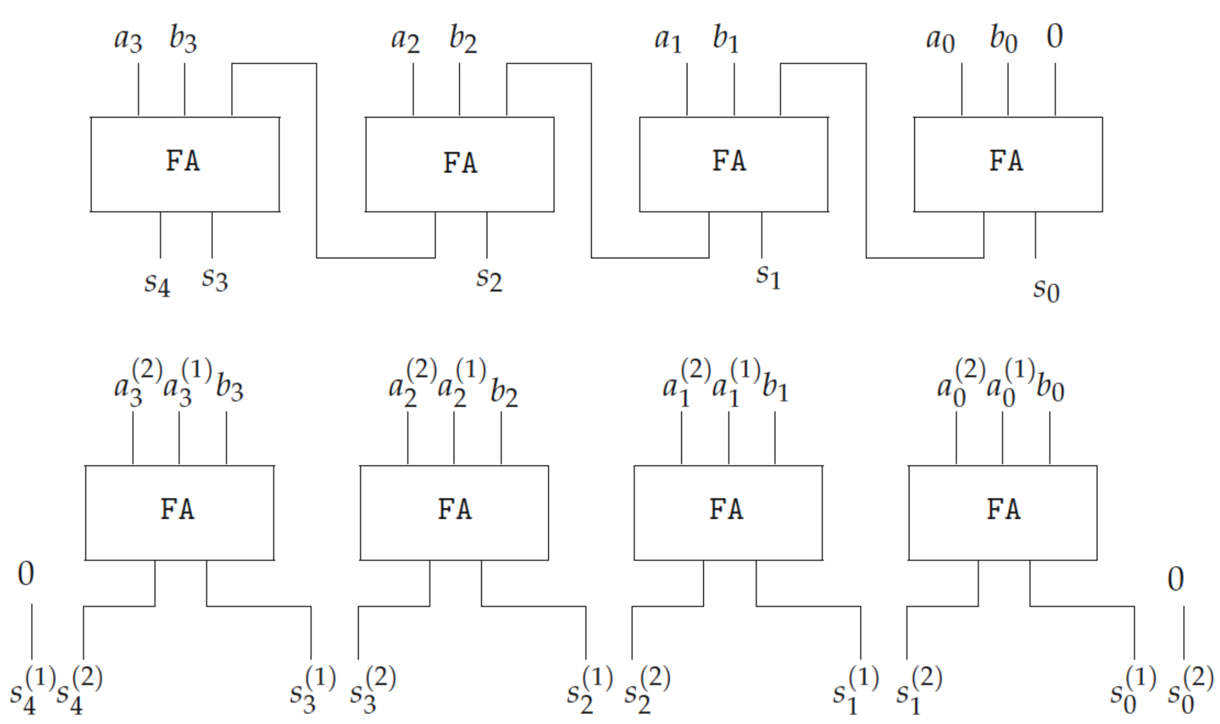
We can also obtain a similar algorithm for $ln(x)$ by modifying a choice of value d in the previous algorithm, to remove $t = ln(x)$, which indeed we want to calculate:

$$d_n = \begin{cases} 1 & \text{if } E_n(1+2^{-n}) \le x \\ 0 & \text{otherwise} \end{cases}$$

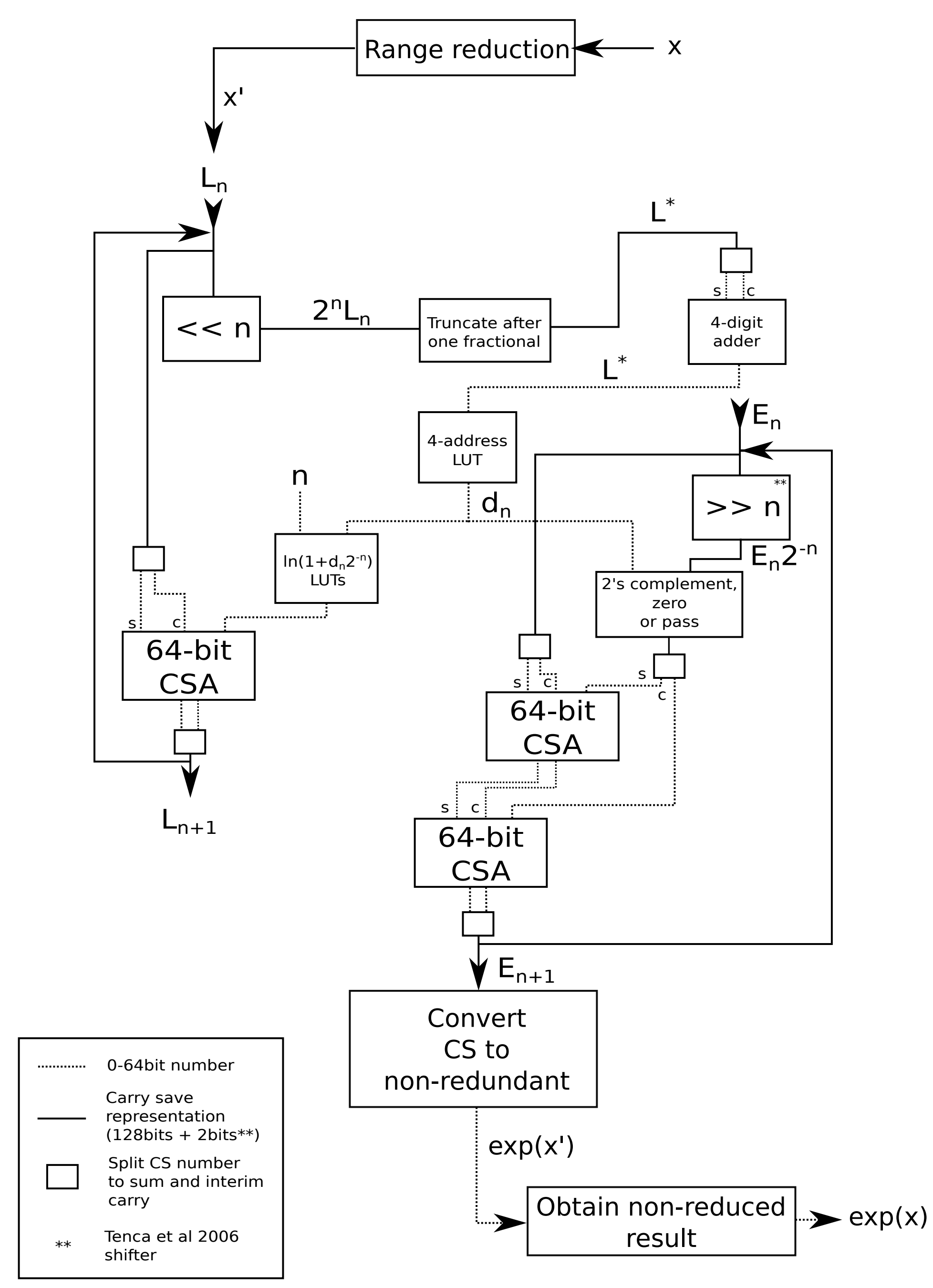
Which gives the same convergence as in the previous algorithm and hence allows us to find an unknown $ln(x)$ because t_n converges to it.

5 Carry save adders

We use carry-save adders in order to avoid ripple-carry overhead. Only when the last iteration finishes, a usual adder is used to get the final result.



6 Hardware Design



References

[1] W. Gerstner, W. M. Kistler, R. Naud, L. Paninski *Neuronal Dynamics* 2014

[2] Jean-Michel Muller *Elementary Functions - Algorithms and Implementation* 3rd Ed. 2016

[3] A. F. Tenca and S. Park and L. A. Tawalbeh *Carry-save representation is shift-unsafe: the problem and its solution* IEEE Transactions on Computers, 2006