A Trainable Sequence Learner that Learns and Recognizes Two-Input Sequence Patterns

Jan Hohenheim
University of Zurich
Zurich, Switzerland
jan@hohenheim.ch

Tommaso Stecconi
IBM Research GmbH
Zurich Research Laboratory
Work carried out at *University of Zurich*
Zurich, Switzerland
tec@zurich.ibm.com

Zhaoyu Devon Liu
CUHK BME
Chinese University of Hong Kong
Hong Kong, China
zhyliu@link.cuhk.edu.hk

Pietro Palopoli
D-ITET
ETH Zurich
Zurich, Switzerland
ppalopoli@student.ethz.ch

TENCON 2022 - 2022 IEEE Region 10 Conference (TENCON) | 978-1-6654-5095-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/TENCON55691.2022.9977663

***Abstract*—We present two designs for an analog circuit that can learn to detect a temporal sequence of two inputs. The training phase is done by feeding the circuit with the desired sequence and, after the training is completed, each time the trained sequence is encountered again the circuit will emit a signal of correct recognition. Sequences are in the order of tens of nanoseconds. The first design can reset the trained sequence on runtime but assumes very strict timing of the inputs. The second design can only be trained once but is lenient in the input's timing.**

***Index Terms*—sequence, learning, analog, design, circuit, coincidence detector**

I. Introduction

Sequence pattern recognition is both central to how our brain works and important for many modern AI applications such as speech recognition, speaker identification, automatic medical diagnosis or general classification. Since the human brain performs pattern learning and recognition with extreme energy efficiency, parallelism, and relatively good speed, the need to replicate these advantages in silica becomes apparent.

Some similar works already exist. Liu et al. [1] designed a multi-terminal transistor that can behave as a sequence detector by tuning the time delay between pulses fed into the transistor terminals. However, this system cannot train itself using certain input sequences like our brain.

As a starting point, we present an idea of how a simple sequence pattern learning and recognition chip could look like and provide two possible implementation schematics.

II. Learning Algorithm

Inputs are expected to be pulses coming from two different sources that arrive in some regular interval, determined by the design used. We will call them *Signal A* (or just A) and *Signal B* (or just B). They are instantly delayed by the same default amount of time. The goal is to make them overlap in time (learning phase) so that later the correct recognition can occur. This is done by treating A as a *reference* and shifting B's delay up or down to make it coincide with A inside the circuit.

This report was produced in collaboration with the Institute of Neuroinformatics (University of Zurich and ETH Zurich)

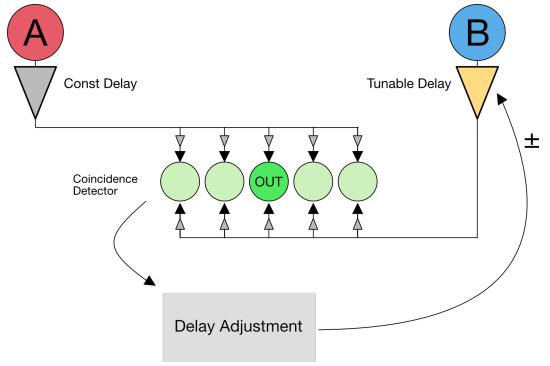

Fig. 1. Overview of the shared logic behind the circuits

To do so, It is necessary to analyze the extent of the delay between the inputs, which is done by using a *coincidence detector* inspired by the Jeffress model of sound localization [2]. This model works by letting the inputs move through a series of delaying nodes from opposite sides. They will meet at a node determined by their temporal offset. If they meet in the middle it means that they came in simultaneously. Since our inputs are assumed to always come with some offset, a meeting in the middle must mean that we successfully tweaked B's delay so that the training is finished.

Before this is the case, the inputs will meet at another node, indicating how off from our goal we are. E.g. if we wish to detect the sequence "AB", B will come in after A, so the signals will meet on one of the nodes on the right in figure 1. meaning that we must aim to *decrease* B's delay. This task is carried out by a *delay adjustment unit* specific to the design in question. In any case, the delay modified is a row of 8 *tunable delay* [3] units. Their delay is inversely proportional to a shared bias voltage.

III. Sequence Learner Designs

A. Design A

1) Usage:

Main inputs	V_{in1} and V_{in2}
Auxiliary inputs	V_{reset} to reset the learned sequence
Constant inputs	V_{delay} at 1.18 V

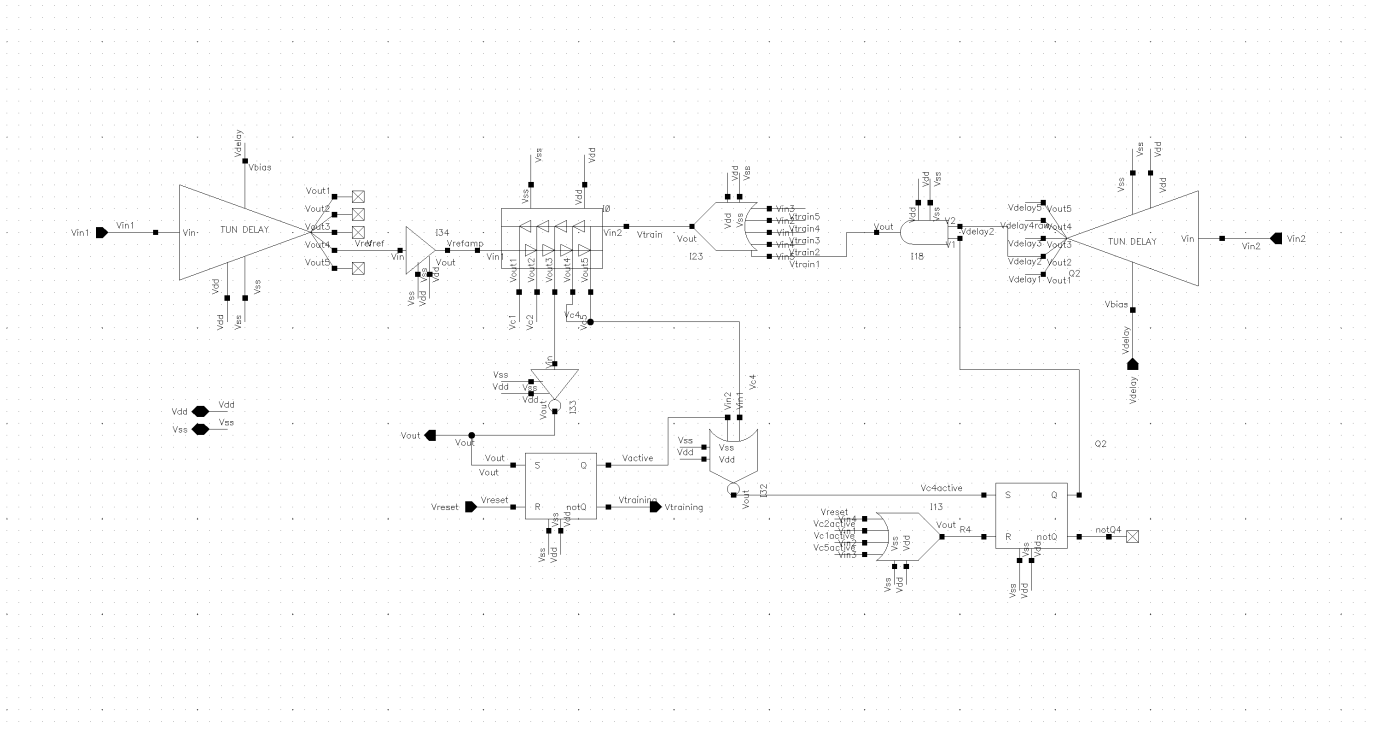

Fig. 2. A simplified view of the circuit's schematic. For clarity's sake, only the connections coming from the coincidence detector nodes in the middle and one to the right of the middle are included. In the full view, all nodes have equivalent connections

Main outputs	V_{out} for the detection
Auxiliary outputs	V_{training} to tell a user that we are currently in training mode
Pulse length	10 - 12 ns
Pulse delay	either 10 - 11 ns (so right after each other) or 20 - 21 ns
Pattern delay	at least 15 ns

Pattern delay is defined as the amount of time that needs to pass after a sequence finished until another one can be detected.

2) Working principles: The first version of the sequence learner uses the common design elements introduced above and SR latches as delay adjustment units (figure 2).

Two pulses with a default delay are fed into two tunable delay lines. The signals are labeled as a reference signal and a trained signal, where the reference signal has a constant delay (figure 4) while the trained signal will have its delay shifted (figure 3).

In this design, the tunable delay line has a constant bias of 1.18 V, which makes every subelement have a delay of 5 ns, where one subelement consists of two tunable delay units. The delay shifting is not done by changing this bias (see the design B for that), but by reading the signal at different points along the delay line. This is done by ANDing the signals with SR latches (figure 3). Only one of them is on at any time, ensuring that only one delay is selected. The outputs of all ANDs are ORd together to get the active signal. The reference signal instead gets its constant delay by always picking up the signal after the third subunit (figure 4).

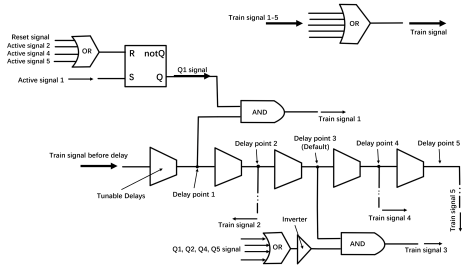

Fig. 3. A trained tunable delay unit. For simplicity, only the connections from the first subunit are included. If this SR latch is the only one active, the delay was successfully reduced.

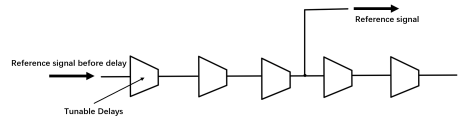

Fig. 4. The constant delay for the reference signal takes the input after it went through three tunable delay subunits.

The latches determining the delay for the train signal get set by the coincidence detector. Its output is a NAND between every pair of nodes, so an output of 0 means that a coincidence was found[1]. If this did not happen in the middle, the training is either not finished or it is finished and we didn't find the right sequence. To differentiate between these, there is an SR latch controlling whether or not we are in *training* or in *active*

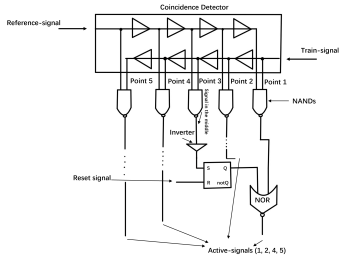

Fig. 5. The diagram of the coincidence detector fed with delayed Train-Signal and Reference-Signal. The ellipsis in the diagram indicates the same way of connection as between point 1 and point 3.

mode via its output V_{active}. Its inverse is V_{training} and it can be reset via V_{reset}. It is set when we detect the first coincidence in the middle of the detector.

To ensure we are only updating the trained delay while actually training, we NOR the output of the coincidence detector with V_{active} (figure 5). This output then is only V_{dd} when both V_{active} and the coincidence detector's output are 0. This output will both set the SR latch that allows reading the correct delay and reset all other latches.

B. Design B

1) Usage:

Main inputs	V_{in1} , V_{in2}
Constant inputs	$V_{\text{default delay}}$ and V_{clock} (values described later)
Main outputs	V_{out} for the detection
Pulse length	10 - 15 ns
Pulse delay	10 - 50 ns
Pattern delay	at least 15 ns

2) Working principles: The second version of the sequence learner extends this functionality to flexible delays ranging between +- 50 ns (A with respect to B). Moreover, it internally provides the analog voltages required to tune the delays of the tunable delay element. To extend the range of detectable delays we modified the coincidence detector block by including a pair of transmission gates working in antiphase between each stage of the coincidence detector (see figure 6).

The output of the coincidence detector can either activate a default delay of 60 ns, in case A and B are overlapped or 40 ns or 20 ns. In the latter two cases, a delay control logic is activated to disconnect the default input voltage of the tunable delay element of $\approx$ 700 mV. The delay of 40 ns is given by an input voltage of $\approx$ 950 mV, which can be provided by an NMOS in transdiode configuration, mirroring its gate voltage, as shown in figure 7.

This solution can be used to set even longer delays; however, $\sim$ 700 mV correspond to a 60 ns delay and is already the threshold voltage for strong inversion regime. Therefore, for longer delays the input N-FET of the tunable delay must be biased in weak inversion, making the system highly sensitive to PVT and mismatches.

[1]An earlier design kept the analog nature of the coincidence detector by using a bump-antibump [4] instead of NANDs. This led to multiple coincidences being detected at the same time, which we tried to remedy using a hysteretic winner-takes-all [5]. All of this ended up producing roughly the same results as just using digital NANDs, so we use them now instead.

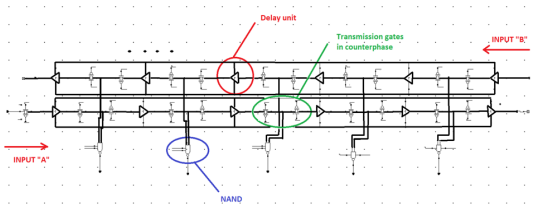

Fig. 6. The second version of the coincidence detector includes a sample and hold system made of two transmission gates in series working in antiphase.

The transmission gates are controlled by a clock signal of period 10 ns, introducing a hold time >5 ns. This solution improves the robustness of the coincidence detection and allows to work with ranges of delays rather than fixed ones.

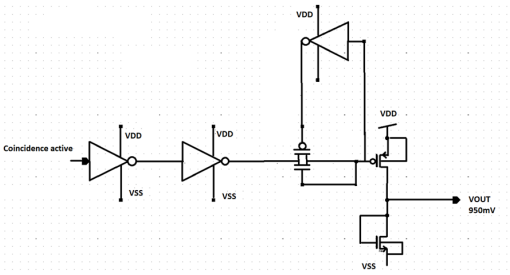

Fig. 7. The schematic for the circuit that converts the 20 ns delay output of the coincidence detector into the corresponding analog voltage driving the tunable delay element.

We tested multiple sequence learning transient simulations, to evaluate the correct functionality of the circuit. The system can adjust the correct delay of 20 ns, for the range of delay 12 $\div$ 30 ns, and 40 ns for the range 32 $\div$ 50 ns, making the signals A and B overlap following the initial training. Only the specific minor ranges of 10 $\div$ 12 ns and 30 $\div$ 32 ns don't succeed, due to the signal distortion occurring in the tunable delay element (a 10 ns long input pulse gets distorted into an 8 ns long delayed output).
In figure 8 we show the transient simulation for a delay of B of 45 ns. The other two main units implemented in this configuration include logic for feeding the tunable delay with 1.8 V (as shown in figure 9) and logic to deactivate the default voltage value in input to the tunable delay (which in our case is $\approx$ 700 mV) - shown in figure 9. It is important to note that the latter circuit has two branches. The main branch connects directly to the tunable delay and provides 1.8 V . The second branch is used to deactivate the detection of 20 ns delay. This solution has been adopted because when a 40 ns delay detection occurs, an additional undesired 20 ns

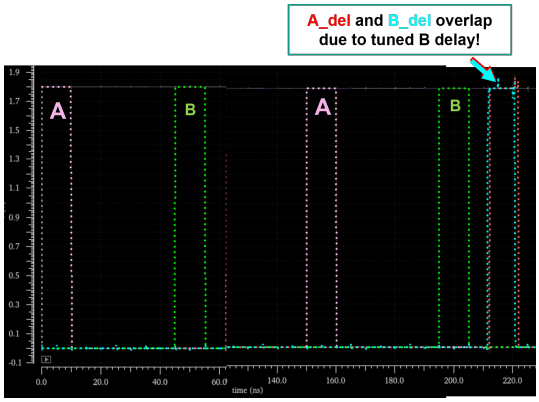

Fig. 8. Transient simulation of the sequence learner, when B is delayed by 45 ns.

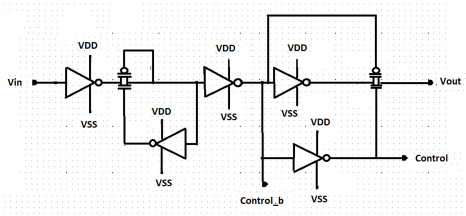

Fig. 9. Circuit schematics converting the 40 ns delay output of the coincidence detector into the corresponding 1.8 V driving the tunable delay element.

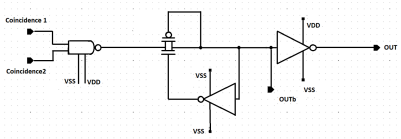

Fig. 10. Circuit schematics used to deactivate the pre-training default voltage in input to the tunable delay.

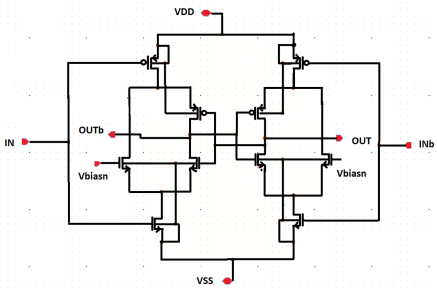

Fig. 11. Single unit of tunable delay obtained from [2]. 8 of these units in series build the tunable delay used in the sequence learner implementation.

delay detection appears which must be suppressed.

IV. Figures of Merit

A. Design A

Power consumption	static: negligible, dynamic: 241 μW
Operations	$1.3 \times 10^7\ s^{-1}$
Area	$2.1 \times 1.2\ mm$

An operation is defined as either training a sequence, detecting a sequence or resetting the circuit. The reported operations assume that it is impractical to let the inputs and outputs overlap. Since the output always comes with a fixed delay of 65 ns after the last input, a careful setup can feed as many inputs to the circuit as possible and let the inputs and outputs overlap, resulting in about 2.5 times more operations ($3.1 \times 10^7\ s^{-1}$).

The layout area was not prioritized at all and can massively be optimized without changing the circuit design.

B. Design B

Power consumption	static: 55 μW, dynamic: 100 μW
Operations	$1.1 \times 10^7\ s^{-1}$
Area	Not layouted

V. Applications and Modifications

As is, the circuit can essentially just process a single bit: The incoming sequence is either AB or BA. The presented implementation can however be extended easily to support recognizing different delays between the signals.

This way, it could be used as a trainable directional detector for binaural auditory systems working in extremely high frequencies, e.g. learning that a mobile sound emitter is slightly to the left of the recording device and then detecting whenever the sound emitter returns to that location. The first design assumes pulse lengths and delays of just 10-20 ns. We can imagine this being of use for industrial robots that need to detect the position of other fast-moving machinery.

By altering the system a bit, one could in principle extend the operating range to support signals in the millisecond range, which would make it compatible with signals seen in biology. This comes closer to the original usage of the Jeffres model embedded within the circuits. This way, a circuit might be trained to detect each time a studied animal population visits a certain spot in conditions where a camera setup is not feasible, e.g. small mice in tall grass.

Another possible usage is to keep the circuit as a binary recognition machine that converts a signal coming from two sources into a single binary output. A sequence learner trained on “AB” would then convert the sequence “ABABBAAB” to “1101”. In this interpretation, we reach a rate of 10 Mbit s^{-1}. It is however not clear when such a thing would be useful.

The input range can also be extended to include any number of signals by adding nodes “perpendicular” to the coincidence detector. This could be used as a generic pattern detector. For example, every input could represent an individual letter and the sequence learner could recognize entire words.

Furthermore, detecting signals with a time-scale of nanoseconds is essential in some fields. R.Kamrla et. al [6]

introduced the concept of 'delayed coincidence' and used 'delayed coincidence circuit' to distinguish between single photon spectra and pair electrons excited by a single photon. Anita et al. [7] introduced an anti-coincidence circuit with a refractory period to reduce the noise when detecting gamma-rays released by boron neutron. By implementing our circuit, it is possible to achieve the functions above.

ACKNOWLEDGMENT

Our thanks to Prof. Shih-Chii Liu for her guidance and help during the long brainstorming sessions that went into the design of these circuits; Prof. Tobi for his advice and suggestions along the way; and the teaching assistants, classmates for their patient help and companionship in the course Neuromorphic Engineering by Institute of Neuroinformatics.

All four authors contributed equally to this paper.

REFERENCES

[1] R. Liu, Y. He, S. Jiang, L. Wang, and Q. Wan, "Synaptic plasticity modulation and coincidence detection emulated in multi-terminal neuromorphic transistors," *Organic Electronics*, vol. 92, p. 106125, 2021.

[2] L. A. Jeffress, "A place theory of sound localization," *J. Comp. Physiol. Psychol.*, vol. 41, pp. 35–39, Feb. 1948.

[3] I. Lee, D. Sylvester, and D. Blaauw, "A Constant Energy-Per-Cycle Ring Oscillator Over a Wide Frequency Range for Wireless Sensor Nodes," *IEEE J. Solid-State Circuits*, vol. 51, pp. 697–711, Feb. 2016.

[4] T. Delbruck, "'Bump' circuits for computing similarity and dissimilarity of analog voltages," in *IJCNN-91-Seattle International Joint Conference on Neural Networks*, vol. i, pp. 475–479vol.1, IEEE, July 1991.

[5] G. Indiveri, "A Current-Mode Hysteretic Winner-take-all Network, with Excitatory and Inhibitory Coupling," *Analog Integr. Circuits Signal Process.*, vol. 28, pp. 279–291, Sept. 2001.

[6] R. Kamrla, W. Widdra, C.-T. Chiang, and F. Schumann, "Electron pair emission from surfaces: Some general experimental considerations," *Journal of Electron Spectroscopy and Related Phenomena*, vol. 257, p. 147185, 2022.

[7] A. Caracciolo, L. Buonanno, D. Di Vita, I. D'Adda, A. Chacon, M. Kielly, M. Carminati, M. Safavi-Naeini, and C. Fiorini, "Benedicte (boron neutron capture): a versatile gamma-ray detection module for boron neutron capture therapy," *IEEE Transactions on Radiation and Plasma Medical Sciences*, pp. 1–1, 2022.