Large Developing Axonal Arbors Using a Distributed and Locally-Reprogrammable Address-Event Receiver

Simeon A. Bamford, Alan F. Murray, David J. Willshaw

***Abstract*— We have designed a distributed and locally reprogrammable address event receiver. Incoming address-events are monitored simultaneously by all synapses, allowing for arbitrarily large axonal fan-out without reducing channel capacity. Synapses can change input address, allowing neurons to implement a biologically realistic learning rule locally, with both synapse formation and elimination.**

I. Introduction

Neuromorphic engineers create integrated electronic circuits which mimic neural computation in biological nervous systems, both to inform computational neuroscience and in pursuit of superior engineering solutions for classes of problems where biology currently outperforms artificial devices [1]. There is a need to form interconnects between many integrated neuron circuits to create neural networks. In many applications such as topographic map development [2], reconfigurability in the connections is essential to underpin map formation and maintenance. In a topographic map, one (typically 2D and sensor-driven) layer of neurons maps its connections to another layer such that neighbouring relationships between neurons in one layer are preserved in the other. In order for such a mapping to develop, neurons gradually change their patterns of connections according to both innate preferences and feedback induced by network input [3].

Time-division multiplexing facilitates massively-parallel connection between spiking neuron circuits across multiple chips. Specifically, spikes are treated as address-events; the unique address of a neuron within a neural array is transmitted on an address bus. This approach was first used in the theses of Sivilotti and Mahowald [4] and [5] and has since been extended and improved. Boahen [6] gives a good summary of this still-evolving technique. Within this Address-Event Representation (AER) protocol, the number of wires required to connect N neurons scales as $log(N)$, such that the number of pins and wires necessary to interconnect chips is achievable. The development of word-serial AER reduces the number of wires required still further [7]. AER exploits the large difference in frequency between the spiking behaviour of biological neurons (on the order of 10-1000Hz) and the capability of digital electronic communication (many MHz). Approximately 100,000 neurons can share a single bus [6] if biological spike rates are desired.

This work has been funded by EPSRC. Authors details: Simeon Bamford, Neuroinformatics Doctoral Training Centre, University of Edinburgh. Alan Murray, Institute of Integrated Micro and Nano Systems, University of Edinburgh. David Willshaw, Institute of Adaptive and Neural Computation, University of Edinburgh. To whom further communication should be addressed: sim.bamford@ed.ac.uk

AER was originally conceived as a point-to-point protocol. If each neuron in one neural layer has a unique connection to only one neuron in a corresponding neural layer in a topographic map arrangement, the outgoing bus can be decoded directly by a row-and-column decoder on a receiving chip, and spikes are delivered correctly to the same location on a corresponding chip (as in [4] [5]). Simplistically this type of one-to-one connectivity can be observed in some places in the nervous system, for example the connections from cone receptors to bipolar cells, at least in the fovea ([8] ch. 26). More commonly however neurons make connections to many other neurons (i.e. they have a large "fan-out") and receive large numbers of incoming connections ("fan-in"). As two examples, Xiong *et al* [9] found an average fan-out of 167 for retinal ganglion cells in the tectum of the hamster, whilst Palkovits *et al* [10] found an average fan-in of 85,000 onto the Purkinje cells of the cat. In order to implement arbitrary many-to-many network connectivity, address-events are commonly received not directly by a neural array chip but rather by a microcontroller and are then compared to a look-up table in memory in order to find out which outgoing address-events should be sent (e.g. [11]). These are then sent sequentially to one or more receiving neural arrays. This approach reduces the capacity of the bus in the presence of large fan-out. If an average fan-out of 1000 is desired for example, a bus can only support about 100 neurons.

The use of a microcontroller and a look-up table in memory has also been used to implement synaptic rewiring, where the connectivity between neurons changes with time according to a biologically inspired learning rule [12]. In the scheme of Taba and Boahen [13], information from the receiving synapse is transmitted off-chip back to the microcontroller where it is used to modify the look-up table. This is part of a trend of using the microcontroller to implement more of the neural network model. This trend has been extended by Vogelstein *et al* [14] where other synaptic variables (number of release sites, probability of release and quantal post-synaptic response — the product of these is essentially the synaptic weight) are also held in the look-up table, allowing each neuron to have a single "general purpose" synapse circuit which acts as a number of virtual synapses.

II. Proposed system

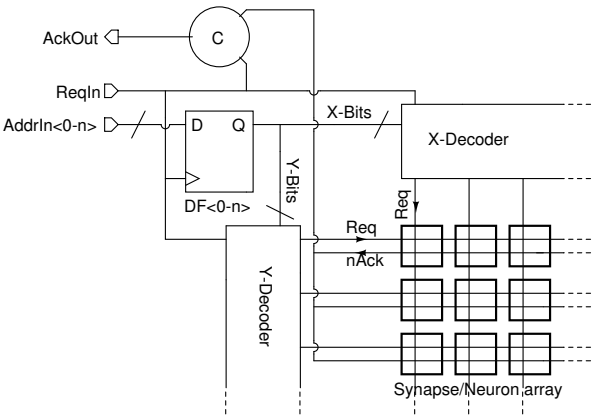
In order to overcome the bottleneck on channel-capacity as fan-out increases, we have taken an alternative approach in which more information is stored in synapse circuits within the neural array. Details of incoming connectivity are stored,

along with synaptic variables such as an analogue voltage representing synaptic weight. Address events from a sending chip are directly received by a receiving chip and broadcast across the receiving chip's neural array. Simultaneously, all synapses compare that address to a locally-stored address to establish whether the address-event was intended for it. Many synapses can store the same desired address and thus arbitrarily large axonal arbors can be implemented without reducing bus capacity. Synapses do not acknowledge receipt of an event, rather the chip-wide broadcast is timed to last long enough for all synapses to receive it. We compare our approach to the "look-up table" approach in which source neuron addresses are mapped to target synapse addresses using a look-up table, an example of which is Mitra *et al* [15]. The look up table approach allows the use of receiving circuitry as described by Boahen [6], which is shown in fig 1a. The receiving circuitry which implements our system is shown in fig 1b. In our system, to ensure that communication succeeds, each communication cycle is deliberately slower than the average cycle speed which could be achieved if the sender were allowed to proceed with the next event as soon as a synapse acknowledges, as in Fig 1a. However as average fan-out increases our solution outperforms any system which implements fan-out serially.

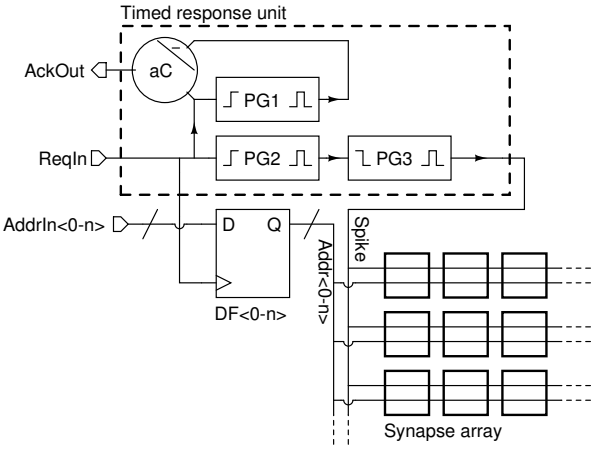
III. Scalability of proposed system

A. Area

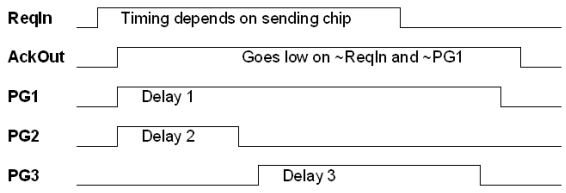
Each synapse, in order to implement its address bus monitor, must store as many bits in memory elements as the width of the incoming address bus. The total area of the monitoring circuitry across the chip (or across the system, for a multi-chip system) then scales as $S_{max}Nlog_2(N)$, where N is the number of neurons in the system and S_{max} is the maximum fan-in, i.e. number of dendritic (or incoming) synapses allowed per neuron. The $S_{max}N$ term represents the number of synapse circuits in the system and the $log_2(N)$ term represents the number of bits necessary to encode a neuron's address within each synapse. At first glance this scales poorly compared to the look-up table approach, which employs row and column decoders allowing the area of the receiving circuitry to scale as $\sqrt{S_{max}N}log_2(S_{max}N)$, where the $\sqrt{S_{max}N}$ term represents the number of row or column decoder elements necessary to decode a target synaptic address and the $log_2(S_{max}N)$ term represents the number of bits necessary to encode a synaptic address (each decoder element must store one dimension (i.e. half the bits) of the synaptic addresses it encodes for). Importantly however the look-up table approach requires that an external memory chip is used, in which area is required which scales as $S_{av}Nlog_2(S_{max}N)$, where S_{av} is average fan-out. The $S_{av}N$ term is the number of axonal (or outgoing) synapses in the system and the $log_2(S_{max}N)$ term is the number of bits necessary to encode a dendritic (or incoming) synaptic address. The costs of microcontrollers and RAM are not normally considered, whether in terms of chip area or power consumption. This is acceptable for test systems,



(a) AER receiver circuitry, functionally equivalent to that described in [6]. The incoming request "ReqIn" triggers the raising of the global acknowledge "AckOut" and the decoding of the incoming address; a synapse (or neuron) is targeted; when this acknowledges, AckOut is lowered (once ReqIn has also been lowered), allowing the next event to be transmitted.



(b) Our AER receiver. Upon ReqIn going high, AckOut is immediately driven high and also a pulse generator (PG1) is triggered, the output of which stays high for a precisely-timed (adjustable) period thereafter. AckOut stays high until ReqIn and PG1 both drop. ReqIn also triggers the local latching of the incoming address bus. Once latched the address is broadcast across the chip and all synaptic address-monitors simultaneously compare this address to their own stored address to decide whether it is correct. From the rising of ReqIn there is a short delay (implemented by PG2) to allow this to happen before a pulse ("Spike") is sent out across the chip (implemented by PG3) triggering those synapses with correct addresses to accept the event. The pulse generated by PG1 is timed to be long enough to accommodate the joint delays of PG2 and PG3 before allowing AckOut to drop and the cycle to repeat.



(c) Example timing diagram for our response unit.

Fig. 1.

TABLE I

SCALING OF AREA, ENERGY USAGE AND SPEED

System	On-chip receiver area	Off-chip memory space	Internal buffering energy per spike sent	Speed per spike sent
Ours	$S_{max}Nlog_2(N)$	none required	$S_{max}N$	unity
Look-up table	$C\sqrt{S_{max}N_{chip}}log_2(S_{max}N_{chip})$	$S_{av}Nlog_2(S_{max}N)$	$S_{av}C\sqrt{S_{max}N_{chip}}$	S_{av}
Vogelstein [14]	$C\sqrt{N_{chip}}log_2(N_{chip})$	$S_{av}Nlog_2(N)$	$S_{av}C\sqrt{N_{chip}}$	S_{av}

N_{chip} = number of neurons per chip; C = number of chips in system; N = number of neurons in system = $N_{chip}C$; S_{max} = maximum fan-in, i.e. number of dendritic synapses allowed per neuron; S_{av} = average fan-out.

however if total power budget and space are considered (for a hypothetical implantable system, for example) it can be seen that in our approach the chip space necessary to implement memory is simply being distributed throughout the neural array, rather than stored in a separate dedicated chip.

It's also worth noting that the scaling expression above for the look-up table approach only holds for a single-chip system. If the system is spread across multiple chips then the expression for the look-up table approach becomes $C\sqrt{S_{max}N_{chip}}log_2(S_{max}N_{chip})$ where C is the number of chips in the system and N_{chip} is the number of neurons per chip. Therefore as a neural network is scaled up by networking together more chips and the ratio of C/N_{chip} goes up, the on-chip area scaling advantage with respect to our system due to row and column decoding is eroded. The scaling expressions above are summarised in table I. Vogelstein's approach [14] is also included for comparison; this is included because it is a special case of the look-up table approach in which there is only one target synapse address per target neuron.

Note that we do not wish to overlook the actual difference in area requirements between these approaches. Memory on a dedicated RAM chip takes up much less space than in our design, partly because it is not integrated with decoder circuitry but rather optimised for its purpose and partly because it does not need to be implemented in a process suitable for mixed signals and can therefore benefit from smaller feature sizes. Beyond this it is also less costly simply because it is mass-produced. Our approach yields synapses of significantly larger on-chip area resulting in higher production costs for the foreseeable future. If however this increase of area can be tolerated for a given technology, then it can be tolerated equally both as miniaturisation proceeds and as the size of neural network implemented expands. Meanwhile we can expect our approach to continue to support larger neural networks with large average fan-outs long after the existing approaches run out of "bandwidth". Whilst chip area is much more expensive on trial ASICs than on mass-produced memory, this may not always be the case if neuromorphic circuitry comes into mainstream demand.

B. Energy usage

In our approach, each incoming address event must be broadcast across the neural array to each synapse. Consequently each synapse contributes a capacitive load to the on-chip buffering and therefore energy consumption will scale linearly with $S_{max}N$. This figure includes internal buffering to the neurons but does not include buffering from the pads to the peripheral latches shown in figure 1, because the look-up table approach has an equivalent cost. The chips we are fabricating each contain 2048 synapses, and based on the analysis in section VI will therefore use 1.76nJ per incoming spike for internal buffering. We expect this to be comparable to the energy necessary to transfer a spike externally between chips, though as die sizes increase we expect the energy cost of internal buffering to become increasingly dominant. In the look up table approach there is no need to broadcast the address across the chip and the spike signal can be targetted to the row and column of the correct synapse within the neural array. The energy cost of internal buffering should therefore be lower and should scale *per incoming spike* as $C\sqrt{S_{max}N_{chip}}$. In our system however, energy usage remains constant per address-event sent, whilst in the look-up table approach energy usage *per spike sent* increases linearly with axonal fan-out, as each axonal synapse requires a separate spike to be transmitted between chips and the correct synapse targetted. Bearing this in mind, scaling expressions for energy are given in Table I. This suggests that if the choice of which approach to use is to be determined by energy usage then there will be a ratio (ignoring the complexity introduced by multi-chip systems) of S_{av} / $S_{max}N$ above which our system can be expected to outperform the look-up table approach. Here we have not considered the additional energy costs of the microcontroller and RAM in the look-up table approach; how this scales depends on the implementation.

C. Speed

As noted above, in our system each communication cycle is deliberately slower than the average cycle speed which can theoretically be achieved in the look-up table approach, though we expect this difference to be no more than a small factor. However the time taken in our approach does not increase with S_{av} whereas in the look-up table approach it increases linearly (as shown in table I). Additionally even in the case where fan-out = 1, the look-up table approach introduces a small latency due to the need for the microcontroller to receive, process and send a spike, though this latency is normally considered insignificant for neural systems running on a biological time scale.

IV. Local synaptic rewiring

As the details of incoming connectivity are stored locally to the synapse, neurons can take advantage of other information stored locally at the soma and in the synapses in order to change incoming connectivity. Specifically, by also storing a binary variable at each synapse indicating whether or not the synapse exists, we use the synaptic weight (an analogue voltage stored on a capacitor) to inform the decision to disconnect. This follows Miller [16], who gives evidence that the decision whether newly sprouted synapses are stabilised or retracted is guided by changes in physiological strengths. The synapse circuit therefore becomes a circuit representing a potential synapse, part of the neuron's total synaptic capacity (a concept explored in [17]). We supplement this with a chip-wide mechanism for implementing synaptic connection, where the probability of a synapse forming with a given pre-synaptic neuron is influenced by the distance between that neuron and the post-synaptic neuron, allowing receptive fields to form according to 2D probabilistic distributions, as if the axons were guided according to some version of the chemoaffinity hypothesis [18]. The details of the neural learning algorithm we use are being published separately however a brief summary is given here:

This generalised model of map formation could equally apply to retino-tectal, geniculo-cortical, or other projections. Implemented cells are considered to be in a 2D "layer" of neurons. There are two excitatory projections to this layer, one from a simulated source layer and one a lateral projection from the cells themselves. Each location in one layer has a corresponding ideal location in the other, such that one layer maps smoothly and completely to the other; for simplicity there is no transformation from source location to ideal location; the address spaces are identical, though our implementation would allow for transformations to be inserted. Each cell in the network layer can receive a maximum number of afferent synapses (64 in our implementation). The projections compete for the synaptic capacity of the network neurons. We assume that an unspecified activity-independent process is capable of guiding the formation of new synapses so that they have a distribution around their ideal locations which is monotonically decreasing with distance. To implement this, where a network cell has less than its maximum number of synapses, the remainder are considered potential synapses. At a fixed rewiring rate a synapse is randomly chosen. If it is a potential synapse a possible pre-synaptic cell is randomly selected and, if for example we choose a Gaussian function of distance, then synapse formation occurs when:

$$r < p_{form}.e^{-\frac{\delta^2}{2{\sigma_{form}}^2}} \tag{1}$$

where r is a random number uniformly distributed in the range $(0, 1)$, p_{form} is the peak formation probability, δ is the distance of the possible pre-synaptic cell from the ideal location of the post-synaptic cell and ${\sigma_{form}}^2$ is the variance of the connection field. In other words a synapse is formed when a uniform random number falls within the area defined by a Gaussian function of distance, scaled according to the peak probability of synapse formation, (which occurs at $\delta = 0$). Lateral connections are formed by the same means as feed-forward connections though our implementation allows different parameters for equation 1 for each projection, or indeed a different function for each projection. If the selected synapse already exists it is considered for elimination. The probability of elimination should be some monotonically decreasing function of weight and is implemented in a similar manner. Weights themselves vary according to a synaptic learning rule - we have chosen a form of spike-timing-dependent plasticity. For completeness, we briefly present ideal models of the neurons and synapses we have implemented. Based on [19], we use integrate and fire neurons, where the membrane potential V_{mem} is described by:

$$\tau_{mem}\frac{\delta V_{mem}}{\delta t} = V_{rest} - V_{mem} + g_{ex}(t)(E_{ex} - V_{mem}) \tag{2}$$

where E_{ex} is the excitatory reversal potential, V_{rest} is the resting potential and τ_{mem} is the passive membrane time constant. Upon reaching a threshold V_{thr}, a spike occurs and V_{mem} is reset to V_{rest}. To simplify implementation, we use a linear approximation to membrane excitation, which is justifiable when $E_{ex} \gg V_{thr}$. Other parameters are highly modifiable. A presynaptic spike at time 0 causes a synaptic conductance $g_{ex}(t) = ge^{\frac{-t}{\tau_{ex}}}$ (where τ_{ex} is the synaptic time constant); this is cumulative for all presynaptic spikes. Spike-timing-dependent plasticity is implemented such that a presynaptic spike at time t_{pre} and a post-synaptic spike at time t_{post} modify the corresponding synaptic conductance by $g \rightarrow g + g_{max}F(\Delta t)$, where $\Delta t = t_{pre} - t_{post}$ and:

$$F(\Delta t) = \left\{ \begin{array}{ll} A_+.e^{(\frac{\Delta t}{\tau_+})}, & if \Delta t < 0 \\ -A_-.e^{(\frac{-\Delta t}{\tau_-})}, & if \Delta t \geq 0 \end{array} \right\} \tag{3}$$

where $A_{+/-}$ are magnitudes and $\tau_{+/-}$ are time constants for potentiation and depression respectively. This is cumulative for all pre- and post-synaptic spike pairs. g is bounded in the range $0 \leq g \leq g_{max}$.

V. Proposed circuit

A. Address-event receiver circuitry

Our chip-level address-event receiver is compatible with standard address-event transmitters. An incoming request is acknowledged immediately and triggers local latching of the address bus and a timed delay followed by a timed pulse to synapses. A minimum cycle time is imposed. In our circuit this is about 20ns, which also allows for the effect of parasitic capacitances extracted from layout; this could be improved if the synapse design was optimised for speed. The circuitry which implements this is shown in fig 1b and a timing diagram is given in fig 1c.

B. Synaptic address monitor circuitry

The total area of the synapse scales as the number of bits necessary to encode a neurons address in the system. It is therefore necessary to make the storage of each bit and its associated circuitry as compact as possible. We have used a static memory element with a transmission-gate implementation of an XNOR gate for comparison with the incoming address bit. The result of the comparison contributes to a NAND gate for the whole monitor, the output of which ("nAeCorrect") indicates whether or not the incoming address is correct. Additional circuits allow for overwriting and read-out (though read-out may not be necessary in a final implementation). The synaptic address monitor circuitry is shown in fig 2, omitting read-out circuitry in the interests of clarity.

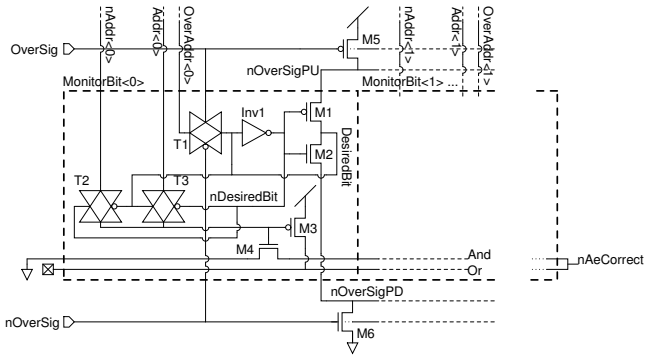


Fig. 2. Address monitor circuitry. The monitor is composed of a chain of bits; one bit is shown here (the zeroth bit). A bit of the address ("DesiredBit") is stored in a memory element composed of Inv1 and M1-2. An XNOR is continuously performed between DesiredBit and the incoming address bit ("Addr<0>") by means of T2-T3 (the incoming bit's complement "nAddr<0>" is also required). The result of the XNOR contributes to a NAND gate implemented throughout the monitor array by transistors M3-4. The result is nAeCorrect, indicating whether the full incoming address matches the full stored address. When OverSig goes high (and its complement nOverSig goes low), this is the signal for the monitor's address to be overwritten with the address on the "OverAddr" bus, a separate bus latching a recently received spike for consideration. OverSig chokes off transistors M1-2 using transistors M5-6 (these are common for all the monitor bits) while T1 opens, allowing DesiredBit to take the value of OverAddr<0>. Readout circuitry is not shown for clarity; this is an additional choked inverter with the same design as M1-2 & 5-6, opened onto a common outgoing bus during the "Compare" signal (see fig 3).

C. Synaptic rewiring circuitry

Synapses can be individually targeted for rewiring by an additional chip-wide mechanism, employing row and column decoders in the periphery. This allows both for the explicit setting and read-out of synaptic variables from an off-chip control mechanism for the purpose of testing the circuit, and for ongoing probabilistic rewiring, where synapses are randomly selected at a given rate as candidates for rewiring. The randomly chosen synapse addresses come from off-chip in our test implementation but could come from an on-chip random-number generator in a mature implementation.

When a synapse is selected as a candidate for rewiring its behaviour depends on its state of connectedness, stored in a static memory element. If it is connected then it is considered for disconnection. Its analogue weight value is compared to a voltage randomly chosen according to a probabilistic distribution. If the weight is below the random value then the synapse is disconnected. The random value is common for all the synapses on the chip but is only used at one synapse at a time and changes between each usage, avoiding the possibility of correlation between synapses. In our implementation the voltage is produced off-chip, but could be produced on chip by a random number generator and a DAC in a mature implementation. It is also possible to generate analogue noise for use in this way [20] which could then be profiled to match the probability of adaptation.

If the synapse is disconnected and it is selected as a candidate for rewiring then the possibility of it taking a new pre-synaptic partner is considered. The pre-synaptic partner considered is the last address to have arrived on the incoming bus. This is latched separately by the chip and also broadcast across the chip at the point that a rewiring consideration takes place. This allows a chip-wide calculation to take place providing a value, available at each neuron, of the geometric proximity of that neuron to the incoming address. The synapse under consideration then compares this proximity value to a random value, similar to the random value for disconnection but separate, created according to a probabilistic distribution for synapse formation. If the proximity value is higher than the random value then the synapse becomes connected and it adopts the incoming address in consideration as its new stored address. The circuitry which implements the connection and disconnection algorithm is shown in fig 3.

Regarding the proximity value, the incoming address may be from a neuron in the same neural layer, even a recurrent spike from the neuron itself, or it may be from a neuron in an afferent layer. We are considering a model in which there is a strong topographic mapping between successive neural layers, but this assumption is not essential to the system we describe. The effect of the proximity on the probability of rewiring can be eliminated altogether if it is not required, by reducing the probabilistic distribution to a binary choice between an extremely high value (where the synapse will not connect no matter how high the proximity) and an extremely low value (where the synapse will definitely connect regardless of proximity). The circuitry for creating the proximity value will be published separately. Briefly however it is an analogue current-mode circuit capable of operating across a neural layer composed of multiple chips and capable of delivering proximity values based on either toroidal (wrap-around) or non-toroidal spaces. It creates voltage gradients along both dimensions of the area upwards from the ideal location of the pre-synaptic partner and then allows a Euclidean distance to be created at a chosen node based on circuitry fundamentally similar to that described in [21].

Whilst it is possible to impose an arbitrary network topology by external programming, it is also possible to allow a probabilistic topology to form and, if desired, to continue to

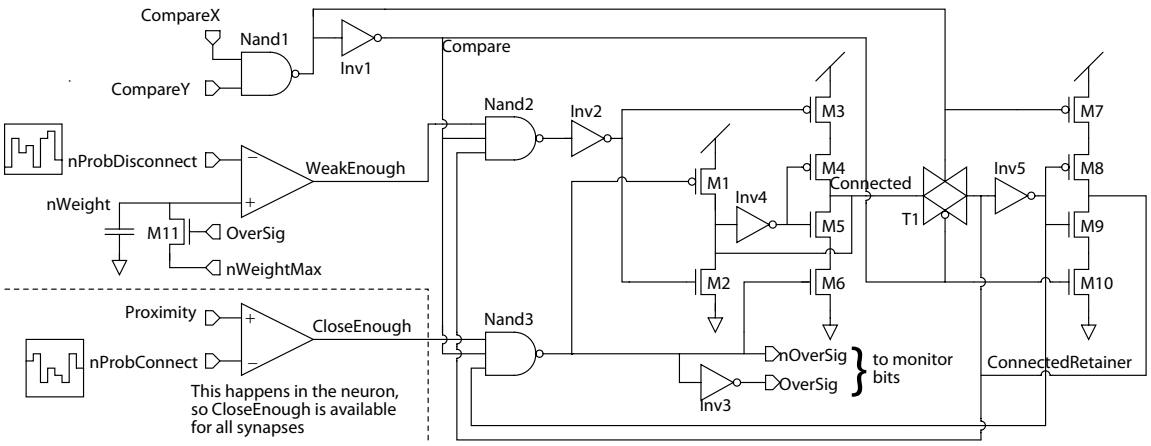


Fig. 3. Circuitry for synaptic rewiring. The synapse's "Connected" state is stored in a memory element composed of Inv4 and M4-5. This state can be overridden by a disconnection signal from Nand2 and Inv2, using M2-3, or by a connection signal from Nand3 using M1 and M6. "Compare" is driven high by the targeted conjunction of the CompareX and CompareY signal from row and column decoders, to indicate that rewiring is under consideration. While Compare is high, the Connected state is latched in a separate memory element "ConnectedRetainer" (Inv 5 and M8-9). This ensures that only connection or disconnection can occur, avoiding oscillations during the Compare signal. On connection, the override signal "Oversig" and its complement are sent to the address monitor, allowing the address under consideration to override the monitor's stored address; nWeight is also set to its strongest value (M11).

develop within the system according to biologically realistic principles, without any details of the topology being made available off-chip. In other words this system allows a black-box approach to network wiring at the level of individual synapses, allowing a system designer to concentrate on higher-level building blocks. Rewiring probabilities can be made arbitrarily low, even achieving biologically-realistic rates of synapse formation and elimination, i.e. hours, days or months between events [22].

VI. Simulation results and layout

A simulation demonstrating the ability of a neuron to rewire one of its synapses is shown in fig 4.

A high level neural network simulation implemented in C++/Matlab has shown the ability of a system with these capabilities and parameters to be capable of performing biologically realistic topographic map formation, even when mismatch ranges taken from Monte Carlo simulations of circuits are applied to the simulation (results not shown here).

The chip is being fabricated in AMS 0.35u 4-metal 2-poly process. The area of the synaptic address monitor bit is $11.1\mu m \times 15.95\mu m = 177\mu m^2$. We are creating a test system with 512 neurons (spread across multiple chips), therefore each synapse has a 9-bit receiver. This takes up 56% of the total synapse area, which is $11.1\mu m \times 255.95\mu m = 2841\mu m^2$. The remaining area is dedicated to: storing the additional synaptic variables; implementing the connection and disconnection circuitry; creating an increase in the neuron's level of synaptic current when a spike arrives; and implementing the synaptic weight change algorithm (spike-timing-dependent plasticity). Each neuron has 64 potential synapses, and the synaptic array takes up 98.6% of the total area of the neuron ($740.275\mu m \times 255.95\mu m = 0.189mm^2$), where the remaining area is dedicated to the storage of the neuron's variables, its central (integrate and fire) functions and its sending circuitry (the neuron circuit is novel, using a switched capacitor approach; this will be described in a separate publication). The layout of the synaptic address-monitor bit is shown in fig 5, excluding upper metal signal and power rails for clarity.

The need to buffer the address out to all the synapses as well as the spike signal requires that a significant capacitive load is overcome. Buffers which achieve this within a reasonable timescale (<5ns) are placed in the periphery of the chip. The ratio of address buffer to synaptic monitor area is $\approx 0.5\%$ and we would expect this ratio to remain approximately constant as neural network size is scaled up within the same technology.

Regarding power consumption, based on simulation with extracted capacitances, the synapse consumes per spike: 227fJ for delivery of the spike signal; 69.6fJ per address bit (assuming that each incoming address bit makes a transition with 50% probability each spike); and 5fJ pumped through the NAND gate that determines whether the correct address has been received. Our 9-bit synapses therefore consume 859fJ each per spike. This figure includes internal buffering to the neurons but does not include buffering from the pads to the peripheral latches shown in figure 1 (this is not included because the look-up table approach would be expected to have an equivalent cost).

VII. Discussion

The address event receiver we have implemented redefines synapse circuits as potential synapses and, in a straightforward manner, shifts the burden of decoding and receiving spike events into them. Pursuing this design choice we have moved other functions into the synapse, namely the ability to implement a developmental model which involves synaptic rewiring. Thus the circuit we have created highlights an alternative pole on a spectrum of design choices regarding the amount of functionality implemented within synapses. Hybrid approaches are clearly possible and may

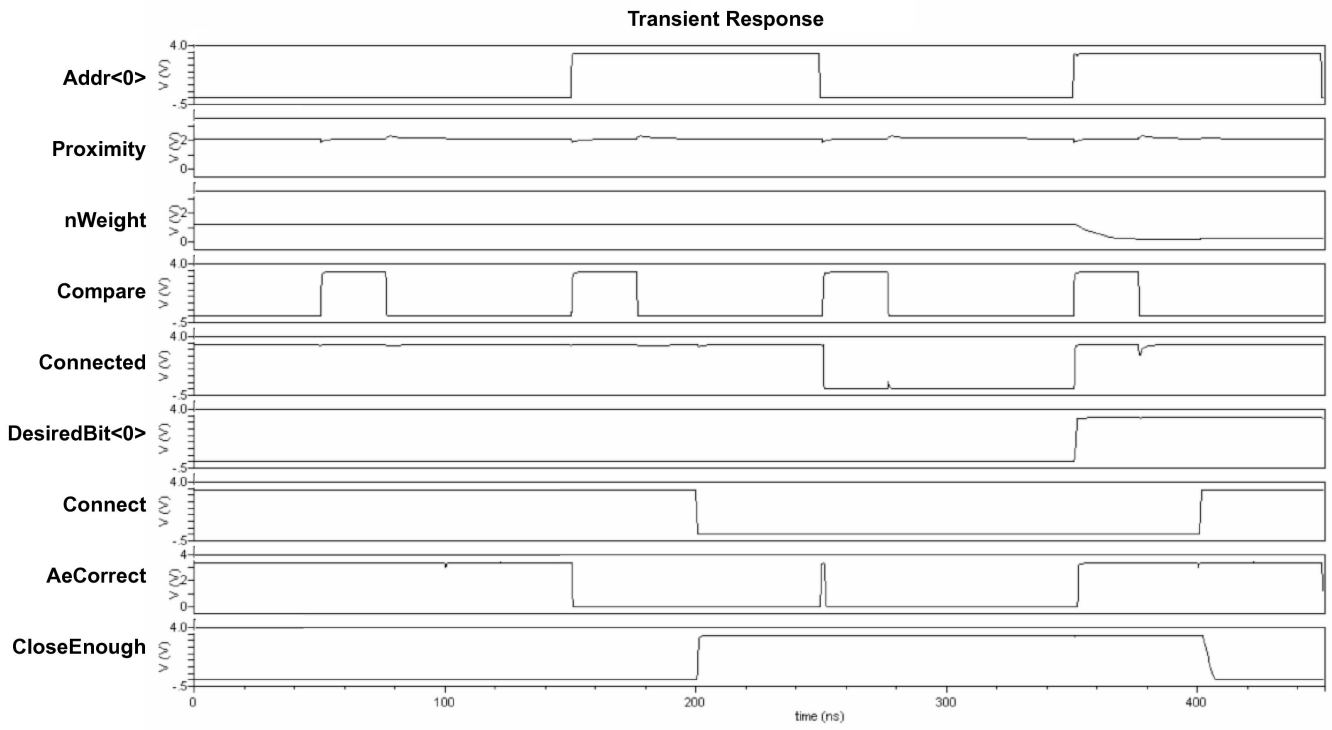


Fig. 4. Trace showing the rewiring of a synapse. The first synapse of a neuron is initially connected("Connected"=true=vdd) to pre-synaptic address 000000000 (only the least significant bit is shown: "DesiredBit¡0¿"). The incoming address starts as 000000000, switches to 000000001 at 150ns, and then switches back and so on every 100ns thereafter (only the least significant bit is shown: "Addr<0>"). "AeCorrect" is the (inverted) output of the NAND gate composed of all monitor bits and this initially indicates that the incoming address is correct, until 150ns at which point the incoming address changes. The random value for connection is initially lower than the "Proximity" value (i.e. nProbConnect is higher) thus "CloseEnough" is false (= 0), until it they switch to respectively high values at 200ns. The random value for disconnection and the corresponding thresholded value "WeakEnough" happen to mirror the aforementioned values (they are not shown here). nProbDisconnect is compared to "nWeight". The two rewiring consideration ("Compare") events at 50ns and 150ns therefore fail to disconnect the neuron because WeakEnough is low. Once WeakEnough goes high the next Compare event at 250ns causes disconnection. Now, although the incoming address matches the stored address, AeCorrect is false, thus the synapse will not accept a spike. At the following Compare event at 350ns, CloseEnough is true and the disconnected synapse is free to connect to the currently latched incoming address, 000000001. Thus DesiredBit goes high and AeCorrect now indicates that the incoming address 000000001 is correct. nWeight is also driven to its minimum (= strong synapse) — a feature of the learning rule we have implemented.

prove beneficial. For example, the adoption of word-serial AER would limit the number of decoder elements in the synapse and promote the adoption of a more standard choice for a repeating memory element. If standard 6-transistor S-RAM elements were used then the size of the repeating memory element would be $\approx 30\mu m^2$ for the same technology and additional read-out circuitry would not be required. Alternatively a DRAM architecture could reduce the size of the repeating unit to as little as $\approx 2\mu m^2$, though noise and power consumption issues may prove prohibitive. As a further alternative, whilst floating gate technology is not best suited to storing synaptic weights because the high frequency of changes usually required by synaptic learning rules would lead to eventual dielectric breakdown, the low rates of synaptic rewiring in natural systems make storage of pre-synaptic addresses on floating gates an attractive option. Analogue storage of many address bits on a single gate could be explored for a possible space saving [23].

Rewiring functions could be centralised to a single circuit on the periphery of each chip. This would remove about 20% of the area of our synapse design, with the expense that the synapse would have to buffer its analogue weight value out to the periphery and some additional signal rails would be required. Note that in our present design, there are two sets of row and column decoders necessary for synaptic rewiring. One targets a synapse for rewiring whilst the other creates a reference location for the proximity calculation. The area required for these scales as $C\sqrt{S_{max}N_{chip}}log_2(S_{max}N_{chip})$ and $C\sqrt{N_{chip}}log_2(N_{chip})$ respectively. This area requirement would not change with the aforementioned proposal.

VIII. Conclusion

We have designed a distributed and locally reprogrammable address event receiver, which allows for arbitrarily large axonal fan-out without reducing channel capacity. Our approach has been mooted before e.g. [11]:

> "Ideally each node should recognise its relevant source events, but our present multi-neuron chips use a DSP chip and lookup table to implement the fan-out from source address to the individual target synaptic addresses."

To our knowledge, however, no such system has been implemented. There is a precedent for simultaneous receipt of events by multiple neurons, in which the same spike was

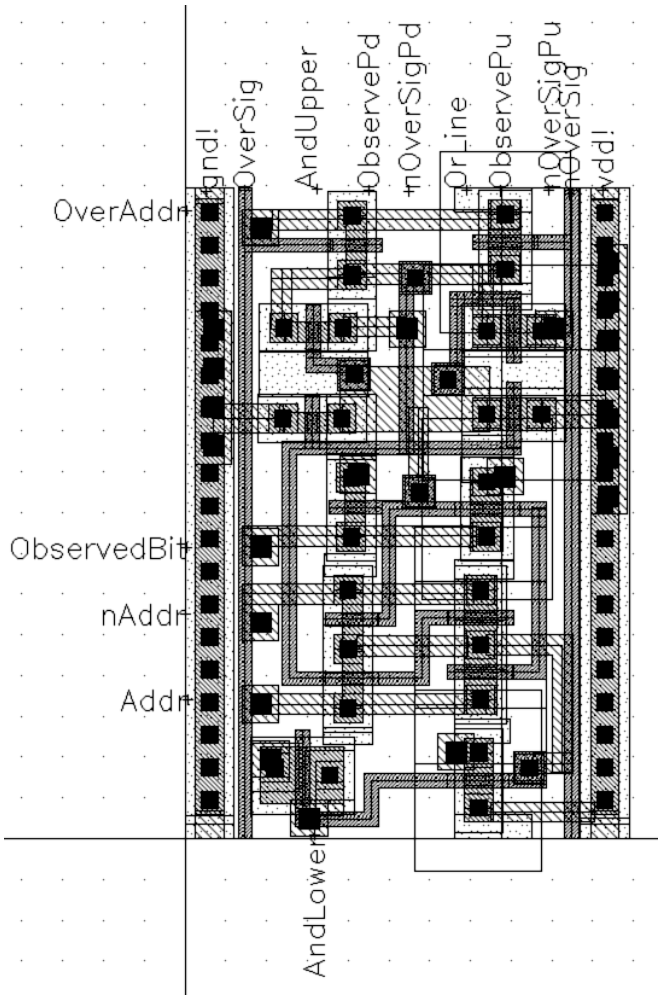


Fig. 5. Layout of synaptic address-monitor bit, in AMS 0.35μ 4-metal 2-poly. Two intermeshing signal layers M2 and M3, and the power layer, M4, have been removed for clarity, though their pin labels and contacts downwards to M1 (larger black squares) are shown. Signal names broadly follow those in fig 2.

delivered to each neuron within a defined area on a chip, implementing a geometrical projective field [24], but this connectivity pattern is fixed and therefore cannot contribute to learning. Our approach also allows for locally implemented probabilistic synaptic rewiring according to a biologically realistic learning rule. Future work will be on demonstrating the abilities of the fabricated chip. Information-theoretic analyses considering constraints of space and power consumption are also anticipated.

ACKNOWLEDGEMENTS

Although not described here, the chip being fabricated contains sections of circuitry acquired from Giacomo Indiveri, Tobi Delbrück and others at INI Zurich. AER Sender schematics were reworked for Cadence by Vasin Boonsobhak. We are grateful to Katherine Cameron for her help and to many people for helpful discussions at the Telluride Neuromorphic Engineering Workshop.

REFERENCES

[1] R. Sarpeshkar, "Borrowing from biology makes for low-power computing," *IEEE Spectrum*, vol. 43, pp. 24–29, May 2006.

[2] D. Willshaw and D. Price, *Modelling Neural Development*. MIT Press, 2003, ch. Models for topographic map formation, pp. 213–244.

[3] H. Cline, "Sperry and Hebb: oil and vinegar?" *Trends in Neurosciences*, vol. 26, pp. 655–661, Dec 2003.

[4] M. Sivilotti, "Wiring considerations in analog VLSI systems, with application to field-programmable networks," Ph.D. dissertation, California Institute of Technology, 1991.

[5] M. Mahowald, "VLSI analogs of neuronal visual processing: A synthesis of form and function," Ph.D. dissertation, California Institute of Technology, 1992.

[6] K. Boahen, "Point-to-point connectivity between neuromorphic chips using address- events," *Circuits and Systems, IEEE Transactions on*, vol. 47, pp. 416–434, 2000.

[7] ——, "A burst-mode word-serial address-event link i: Transmitter design," *Circuits and Systems, IEEE Transactions on*, vol. 51, pp. 1269–1280, 2004.

[8] E. Kandel, J. Schwartz, and T. Jessel, *Principles of Neural Science; 4th Edition*. McGraw-Hill Medical, 2000.

[9] M. Xiong, S. Pallas, S. Lim, and B. Finlay, "Regulation of retinal ganglion cell axon arbor size by target availability: Mechanisms of compression and expansion of the retinotectal projection," *Journal of Comparative Neurology*, vol. 344, pp. 581–597, Oct 1994.

[10] M. Palkovits, P. Magyar, and J. Szentagothai, "Quantitative histological analysis of the cerebellar cortex in the cat," *Brain Res.*, vol. 34, pp. 1–18, 1971.

[11] S. Deiss, R. Douglas, and A. Whatley, *Pulsed Neural Networks*, 1999, ch. A pulse-coded communications infrastructure for neuromorphic systems, pp. 157–178.

[12] D. Chklovskii, B. Mel, and K. Svoboda, "Cortical rewiring and information storage," *Nature*, vol. 431, pp. 782–788, 2004.

[13] B. Taba and K. , Boahen, "Topographic map formation by silicon growth cones," in *Neural Information Processing Systems, Proceedings of,*, 2002.

[14] R. Vogelstein, U. Mallik, J. Vogelstein, and G. Cauwenberghs, "Dynamically reconfigurable silicon array of spiking neurons with conductance-based synapses," *IEEE Transactions on Neural Networks*, vol. 18, pp. 253–265, 2007.

[15] S. Mitra, S. Fusi, and G. Indiveri, "A VLSI spike-driven dynamic synapse which learns only when necessary," in *Proc. IEEE International Symposium on Circuits and Systems*, 2006, pp. 2777–2780.

[16] K. Miller, "Equivalence of a sprouting-and-retraction model and correlation-based plasticity models of neural development," *Neural Computation*, vol. 10, pp. 529–547, 1998.

[17] J. Bougeois and P. Rakic, "Changes of synaptic density in the primary visual cortex of the macaque monkey from fetal to adult stage," *Journal of Neuroscience*, vol. 13, pp. 2801–2820, 1993.

[18] R. Sperry, "Chemoaffinity in the orderly growth of nerve fiber patterns and connections," *Proc. Natl. Acad. Sci. USA*, vol. 50, pp. 703–709, 1963.

[19] S. Song and L. Abbott, "Cortical development and remapping through spike timing- dependent plasticity," *Neuron*, vol. 32, pp. 339–350, Oct 2001.

[20] J. Alspector, R. Allen, V. Hu, and S. Satyanarayana, "Stochastic learning networks and their electronic implementation," in *Neural information processing systems; Proceedings of the First IEEE Conference*, 1988, pp. 9–21.

[21] U. Cilingiroglu and D. Aksin, "A 4-transistor euclidean distance cell for analog classifiers," in *IEEE International Symposium on Circuits and Systems*, 1998, pp. 84–87.

[22] J. Trachtenberg, B. Chen, G. Knott, G. Feng, J. Sanes, E. Welker, and K. Svoboda, "Long-term in vivo imaging of experience-dependent synaptic plasticity in adult cortex," *Nature*, vol. 420, pp. 788–794, 2002.

[23] M. Holler, S. Tam, H. Castro, and R. Benson, "An electrically trainable artificial neural network (etann) with 10240 floating gatešynapses," pp. 50–55, 1990.

[24] T. Serrano-Gotarredona, A. Andreou, and B. Linares-Barranco, "Aer image filtering architecture for vision-processing systems," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 46, pp. 1064–1071, Sept 1999.