# A standalone hardware-based learning system

Trevor CLARKSON[§] and Chi Kwong NG°

[§]Department of Electronic and Electrical Engineering, King's College, Strand, London, UK
°Department of Electronic Engineering, City Polytechnic of Hong Kong, HONG KONG

The probabilistic RAM (pRAM) is an artificial neuron. A hardware-realisable reinforcement training algorithm has been devised for the pRAM. The results presented here are obtained by training a VLSI pRAM net to perform a classification task where a hardware-based weight update procedure has been used throughout. The pRAM may be used as an embedded process controller where on-line adaptation is required and the use of a workstation is undesirable on the grounds of size or cost.

## 1. INTRODUCTION

The probabilistic RAM (pRAM) device has been recently described[1)] as an example of the VLSI implementation of an artificial neural network. Whilst the training example described in the paper[1)] above employed the on-chip weight update procedure, the environmental reward and penalty signals were supplied by a host computer. The results presented here have been obtained from a development of the above chip and additionally, the environmental signals used in training have been obtained from a pRAM. Thus the whole training operation may be conducted in hardware; no supervisory processor is required.

The pRAM neuron[2, 3)] generates an output in the form of a spike train where the probability of generating a spike is controlled by an internal weight, represented as a real-valued number $\in [0,1]$. The firing probabilities for all possible binary input vectors can be trained and $2^N$ weights are used in each pRAM, where $N$ is the number of synaptic inputs to the pRAM.

The pRAM-256 is a VLSI device which processes 256 internal pRAMs in a similar way to the earlier device[1)]. The primary difference between the two chips is in the provision of a more versatile range of reinforcement training methods in the pRAM-256, the same update algorithm being used in each. The earlier device provided only local reinforcement training where two "auxiliary pRAMs" are used to determine the appropriate reward and penalty signals for each "learning pRAM". The pRAM-256 device is no longer restricted to local learning. By allowing the reward and penalty inputs to each pRAM to be reconfigured by the use of "connection pointers", global, local and competitive methods of training can be used in this fourth generation of pRAM hardware[4)]. Each pRAM neuron now has 6 inputs.

Processing of neurons in the pRAM-256 is carried out in two passes. The first pass provides forward processing of signals through the network and updates the outputs of each of the 256 pRAMs. The second pass is only executed if training is enabled and this pass updates the pRAM weights used in pass 1 in response to the new output of the net. Pass 1 processing requires 154μs and processing both Pass 1 and Pass 2 requires 246μs in total, both timings are for a clock of 33MHz.

At the same time as the pRAMs are being processed, new external data may be shifted onto the chip so that neural processing may proceed without incurring any data transfer delay. The pRAM-256 is shown in Fig. 1.

## 2. HARDWARE LEARNING

The learning algorithm used in the pRAM-256 is:

$$\Delta\alpha_{\mathbf{u}}(t) = \rho((a-\alpha_{\mathbf{u}})r + \lambda(\bar{a}-\alpha_{\mathbf{u}})p)(t) \times \delta_{u,i}$$

where $a$ is the pRAM output, either 0 or 1, and $\alpha_{\mathbf{u}}$ is the memory content addressed by the input vector u. The reward and penalty inputs are $r$ and $p$ respectively; these are in the interval 0 to 1, but are normally binary quantities generated with a given probability by the environment. $\rho$ and $\lambda$ are learning rates $\in [0,1]$. Thus it may be seen from (1) that the weight, $\alpha_{\mathbf{u}}$, is always in the interval [0,1] so that clipping is never required.

The action of the above algorithm is to move the weight closer to the value of *a* (the pRAM output) if a reward is given and to move the weight further away from *a* if a penalty signal is given. Thus beneficial actions are made more likely to happen in the future and adverse actions are made less likely to occur, given the same circumstances.

It is the use of a hardware-realisable algorithm on-chip which makes a totally hardware-based learning system possible.

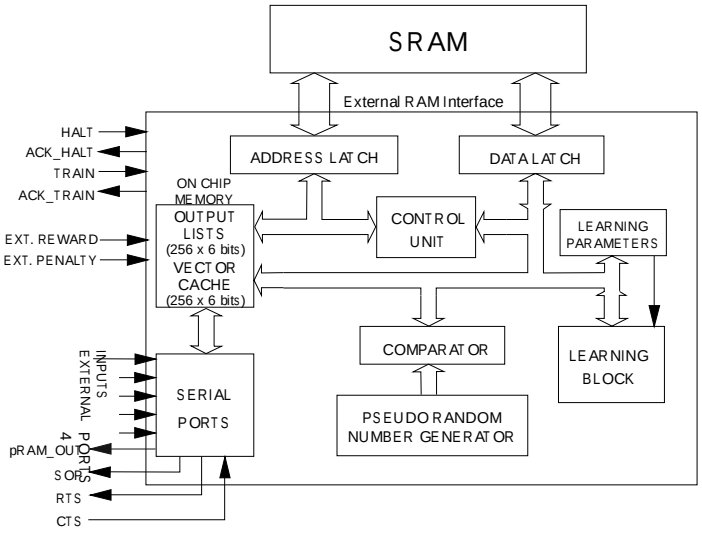


Fig. 1. pRAM-256 Architecture

## 9. TRAINING TASK

A pRAM net was trained to classify patterns. In this example, four patterns of 6 x 6 pixels in size were used which were encoded as 2-bit numbers, 00, 01, 10 and 11. As each pattern was presented to the net in a randomised order, the labelled pattern class was compared to the output of the network by a supervisor and a reward was given to the net if the output of the net agreed with the class of the pattern presented; a penalty signal was sent from the supervisor to the net otherwise (Fig. 2). The supervisor therefore acts as a predefined lookup table. The appropriate reward and penalty signals are sent to all pRAMs in the net, so that global reinforcement took place. As training proceeds, so the pRAM weights are modified; correct outputs from the net are rewarded and incorrect outputs are penalised.

Receiving a reward signal makes a pRAM more likely to produce the same action in the future in similar circumstances (e.g. it will be more likely to output a 1 later on, if it output a 1 and received a reward); and vice-versa for a penalty signal.

The output of the net is monitored and may be displayed graphically. To observe the overall trend, output spikes from the net may be integrated over 256 spike periods to obtain the mean firing frequency. In this way, as the net trains, it is possible to observe the qualitative results produced by the net. The weights, representing the probability of firing, are initialised to 0.5, which means that each pRAM is equally likely to output a '1' as it is to output a '0'.

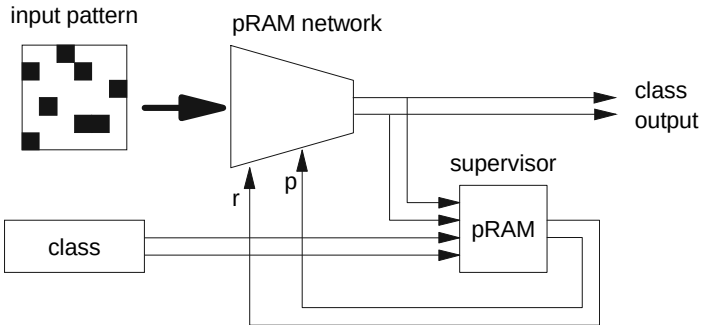


Fig. 2. Training the pRAM network.

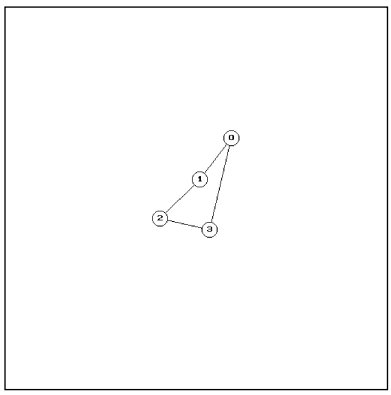


Fig. 3. The mean firing frequencies of the pRAM net are represented graphically where the edge of the area represents mean firing rates of the two pRAM outputs of 0% or 100%. The centre of the area denotes a mean firing rate of 50% from each pRAM neuron. The network has been run for 12 iterations.

At the start of training, the mean firing frequencies of the pRAMs are very close to 0.5 - no one class is clearly identified. After only 12 iterations however, some structure emerges (Fig. 3). After (typically) 60 iterations, all input patterns are correctly classified and the patterns are all correctly identified.

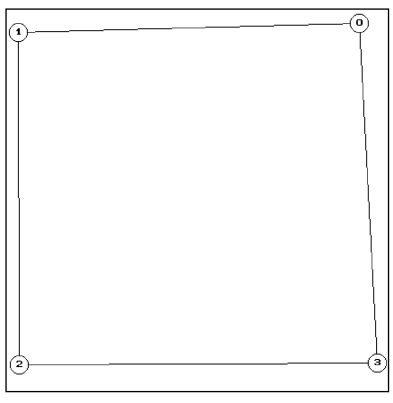


Fig. 4. The mean firing frequencies of the pRAM net after 60 iterations.

The configuration of the pRAM network is defined by a "connection pointer" table where one entry exists

for each input of each pRAM. This table is held in static RAM with the pRAM weight memory (Fig. 1). Thus many architectures can be built with the same hardware; reconfiguring the network only requires updating a connectivity table.

## 10. INTERFACING TO EXTERNAL HARDWARE

To interface the pRAM-256 device to external hardware, Lattice in-system programmable (*isp*) FPGA devices are currently used. These FPGAs have many registers, some of which may be used to count the spikes coming from the serial outputs of individual pRAMs to generate a mean firing frequency. Combinational circuitry within the FPGA may be used for the supervisor, to generate the reward and penalty signals during training. In this way, the pRAM supervisor of Fig. 2 may be replaced by FPGA functions. Since the logic of the *isp* FPGA devices may be reconfigured in-situ, a very versatile system results where both the neural network, and its interface can be reused for a different purpose with no changes to the hardware circuitry.

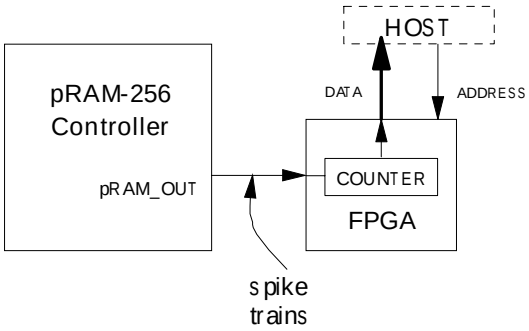


Fig. 5. Example showing the use of a pRAM-256 with a host computer. An FPGA counter is used to convert a pRAM spike train into a mean firing frequency which may be read from a register by a host computer. Similar registers within the FPGA containing the outputs of other neurons may also be addressed and read by the host.

The pRAM-256 and FPGA hardware shown in Fig. 5, can be reconfigured by changes in the firmware because the functions of many signals are fixed. For example, the pRAM_OUT pin of the pRAM-256 always conveys pulses which represent the states of all 256 pRAM neurons. Another output, SOP (not shown), indicates the start of a processing pass and CTS (not shown in Fig. 5) is a clock for the pRAM_OUT signal. In this way, these three signals will normally be assigned to three inputs of the FPGA. The FPGA may be configured to count pRAM pulses and form a mean firing frequency or, alternatively, the FPGA may be reconfigured as a shift register to convey the states of a number of pRAM neurons to the host computer.

The FPGA to host connection can be similarly predetermined, requiring a unidirectional address bus, a bidirectional data bus and a read/write line. In this way data or status registers may be read as shown in Fig. 5, or a control register can be configured within the FPGA which may be written to by the host computer. The control register may be used to set the state of the pRAM-256 control lines, such as TRAIN and HALT which enable training and halt the pRAM-256 respectively. The HALT mode tri-states all pRAM-256 lines and, in this case, would allow the host computer to modify, read or store the neurons' weights or update the connection table to reconfigure the network.

External data may be applied directly to the pRAM-256 or such data may be buffered by the FPGA. The FPGA is used to take the neural network output and to directly control the system to which it is connected.

## 11. CONCLUSION

The probabilistic RAM (pRAM) is a VLSI device with on-chip learning. It has been shown how an autonomous learning system may be constructed using this device. Networks containing in excess of 1500 neurons may be built by the use of multiple pRAM-256 devices. The use of CMOS digital hardware allows high-speed processing to be combined with a low-power consumption and a small size. This makes the pRAM-256 applicable to a wide range of applications.

## 12. REFERENCES

1) T G Clarkson, C K Ng and Y Guan, IEEE Transactions on Neural Networks, 4 (1993) 408.

2) D Gorse and J G Taylor, Physica D, 34 (1989) 90.

3) T G Clarkson, D Gorse, J G Taylor and C K Ng, IEEE Transactions on Computers, 41 (1992) 1552.

4) T G Clarkson and C K Ng, Proc. Microelectronics for Neural Networks, Edinburgh, Scotland, 1993, (UnivEd, Edinburgh, 1993), p 233.