nature COMMUNICATIONS

ARTICLE

DOI: 10.1038/s41467-017-01481-9 OPEN

Temporal correlation detection using computational phase-change memory

Abu Sebastian[1], Tomas Tuma[1], Nikolaos Papandreou[1], Manuel Le Gallo [1], Lukas Kull[1], Thomas Parnell[1] & Evangelos Eleftheriou[1]

Conventional computers based on the von Neumann architecture perform computation by repeatedly transferring data between their physically separated processing and memory units. As computation becomes increasingly data centric and the scalability limits in terms of performance and power are being reached, alternative computing paradigms with collocated computation and storage are actively being sought. A fascinating such approach is that of computational memory where the physics of nanoscale memory devices are used to perform certain computational tasks within the memory unit in a non-von Neumann manner. We present an experimental demonstration using one million phase change memory devices organized to perform a high-level computational primitive by exploiting the crystallization dynamics. Its result is imprinted in the conductance states of the memory devices. The results of using such a computational memory for processing real-world data sets show that this co-existence of computation and storage at the nanometer scale could enable ultra-dense, low-power, and massively-parallel computing systems.

[1] IBM Research–Zurich, Säumerstrasse 4, 8803 Rüschlikon, Switzerland. Correspondence and requests for materials should be addressed to A.S. (email: ase@zurich.ibm.com)

In today's computing systems based on the conventional von Neumann architecture (Fig. 1a), there are distinct memory and processing units. The processing unit comprises the arithmetic and logic unit (ALU), a control unit and a limited amount of cache memory. The memory unit typically comprises dynamic random-access memory (DRAM), where information is stored in the charge state of a capacitor. Performing an operation (such as an arithmetic or logic operation), f, over a set of data stored in the memory, A, to obtain the result, $f(A)$, requires a sequence of steps in which the data must be obtained from the memory, transferred to the processing unit, processed, and stored back to the memory. This results in a significant amount of data being moved back and forth between the physically separated memory and processing units. This costs time and energy, and constitutes an inherent bottleneck in performance.

To overcome this, a tantalizing prospect is that of transitioning to a hybrid architecture where certain operations, such as f, can be performed at the same physical location as where the data is stored (Fig. 1b). Such a memory unit that facilitates collocated computation is referred to as computational memory. The essential idea is not to treat memory as a passive storage entity, but to exploit the physical attributes of the memory devices to realize computation exactly at the place where the data is stored. One example of computational memory is a recent demonstration of the use of DRAM to perform bulk bit-wise operations[1] and fast row copying[2] within the DRAM chip. A new class of emerging nanocale devices, namely, resistive memory or memristive devices with their non-volatile storage capability, is particularly well suited for computational memory. In these devices, information is stored in their resistance/conductance states[3–6]. An early proposal for the use of memristive devices for in-place computing was the realization of certain logical operations using a circuit based on TiO_x-based memory devices[7]. The same memory devices were used simultaneously to store the inputs, perform the logic operation, and store the resulting output. Subsequently, more complex logic units based on this initial concept have been proposed[8–10]. In addition to performing logical operations, resistive memory devices, when arranged in a cross-bar configuration, can be used to perform matrix–vector multiplications in an analog manner. This exploits the multi-level storage capability as well as Ohm's law and Kirchhoff's law. Hardware accelerators based on this concept are now becoming an important subject of research[11–17]. However, in these applications, the cross-bar array of resistive memory devices serves as a non-von Neumann computing core and the results of the computation are not necessarily stored in the memory array.

Besides the ability to perform logical operations and matrix–vector multiplications, another tantalizing prospect of computational memory is that of realizing higher-level computational primitives by exploiting the rich dynamic behavior of its constituent devices. The dynamic evolution of the conductance levels of those devices upon application of electrical signals can be used to perform in-place computing. A schematic illustration of this concept is shown in Fig. 1c. Depending on the operation to be performed, a suitable electrical signal is applied to the memory devices. The conductance of the devices evolves in accordance with the electrical input, and the result of the computation is imprinted in the memory array. One early demonstration of this concept was that of finding factors of numbers using phase change memory (PCM) devices, a type of resistive memory devices[18–20]. However, this procedure is rather sensitive to device variabilities and thus experimental demonstrations were confined to a small number of devices. Hence, a large-scale experimental demonstration of a high-level computational primitive that exploits the memristive device dynamics and is robust to device variabilities across an array is still lacking.

In this paper, we present an algorithm to detect temporal correlations between event-based data streams using computational memory. The crystallization dynamics of PCM devices is exploited, and the result of the computation is imprinted in the very same memory devices. We demonstrate the efficacy and robustness of this scheme by presenting a large-scale experimental demonstration using an array of one million PCM devices. We also present applications of this algorithm to process real-world data sets such as weather data.

Results

Dynamics of phase change memory devices. A PCM device consists of a nanometric volume of phase change material sandwiched between two electrodes. A schematic illustration of a PCM device with mushroom-type device geometry is shown in Fig. 2a)[21]. In an as-fabricated device, the material is in the crystalline phase. When a current pulse of sufficiently high amplitude is applied to the PCM device (typically referred to as the RESET pulse), a significant portion of the phase change material melts owing to Joule heating. When the pulse is stopped abruptly, the molten material quenches into the amorphous phase because of the glass transition. In the resulting RESET state, the device will be in the low conductance state as the amorphous region blocks the bottom electrode. The size of the amorphous region is captured by the notion of an effective thickness, u_a that also accounts for the asymmetric device geometry[22]. PCM devices exhibit a rich dynamic behavior with an interplay of electrical, thermal and structural dynamics that forms the basis for their application as computational memory. The electrical transport exhibits a strong

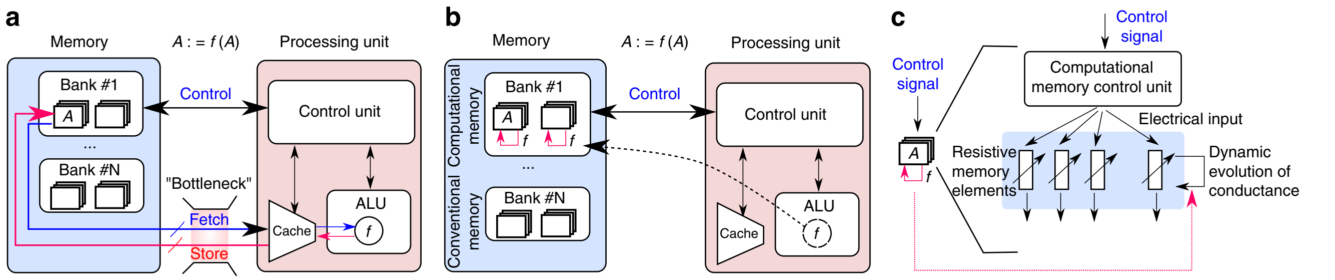

Fig. 1 The concept of computational memory. **a** Schematic of the von Neumann computer architecture, where the memory and computing units are physically separated. A denotes information stored in a memory location. To perform a computational operation, $f(A)$, and to store the result in the same memory location, data is shuttled back and forth between the memory and the processing unit. **b** An alternative architecture where $f(A)$ is performed in place in the same memory location. **c** One way to realize computational memory is by relying on the state dynamics of a large collection of memristive devices. Depending on the operation to be performed, a suitable electrical signal is applied to the memory devices. The conductance of the devices evolves in accordance with the electrical input, and the result of the operation can be retrieved by reading the conductance at an appropriate time instance

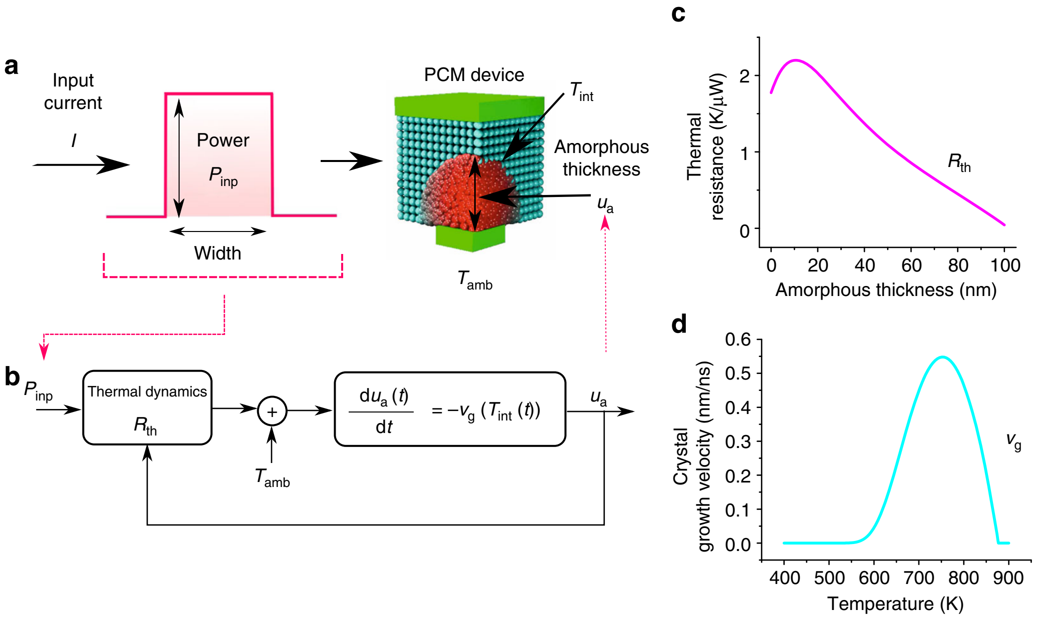

Fig. 2 Crystallization dynamics. **a** Schematic of a mushroom-type phase change memory device showing the phase configurations. **b** Illustration of the crystallization dynamics. When an electrical signal with power P_{inp} is applied to a PCM device, significant Joule heating occurs. The resulting temperature distribution across the device is determined by the thermal environment, in particular the effective thermal resistance, R_{th}. The effective thickness of the amorphous region, u_a, evolves in accordance with the temperature at the amorphous–crystalline interface, T_{int}, and with the temperature dependence of crystal growth, v_g. Experimental estimates of **c** R_{th} and **d** v_g

field and temperature dependence[23]. Joule heating and the thermal transport pathways ensure that there is a strong temperature gradient within the PCM device. Depending on the temperature in the cell, the phase change material undergoes structural changes, such as phase transitions and structural relaxation[24,25].

In our demonstration, we focus on a specific aspect of the PCM dynamics: the crystallization dynamics capturing the progressive reduction in the size of the amorphous region due to the phase transition from amorphous to crystalline (Fig. 2b). When a current pulse (typically referred to as the SET pulse) is applied to a PCM device in the RESET state such that the temperature reached in the cell via Joule heating is high enough, but below the melting temperature, a part of the amorphous region crystallizes. At the nanometer scale, the crystallization mechanism is dominated by crystal growth due to the large amorphous–crystalline interface area and the small volume of the amorphous region[24]. The crystallization dynamics in such a PCM device can be approximately described by

$$\frac{\mathrm{d}u_a}{\mathrm{d}t} = -v_g(T_{int}), \tag{1}$$

where v_g denotes the temperature-dependent growth velocity of the phase change material; $T_{int} = R_{th}(u_a)P_{inp} + T_{amb}$ is the temperature at the amorphous–crystalline interface, and $u_a(0) = u_{a_0}$ is the initial effective amorphous thickness[24]. T_{amb} is the ambient temperature, and R_{th} is the effective thermal resistance that captures the thermal resistance of all possible heat pathways. Experimental estimates of R_{th} and v_g are shown in Figs. 2c, d, respectively[24]. From the estimate of R_{th} as a function of u_a, one can infer that the hottest region of the device is slightly above the bottom electrode and that the temperature within the device decreases monotonically with increasing distance from the bottom electrode. The estimate of v_g shows the strong temperature dependence of the crystal growth rate. Up to approx. 550 K, the crystal growth rate is negligible whereas it is maximum at ~750 K. As a consequence of Eq. 1, u_a progressively decreases upon the application of repetitive SET pulses, and hence the low-field conductance progressively increases. In subsequent discussions, the RESET and SET pulses will be collectively referred to as write pulses. It is also worth noting that in a circuit-theoretic representation, the PCM device can be viewed as a generic memristor, with u_a serving as an internal state variable[26–28].

Statistical correlation detection using computational memory. In this section, we show how the crystallization dynamics of PCM devices can be exploited to detect statistical correlations between event-based data streams. This can be applied in various fields such as the Internet of Things (IoT), life sciences, networking, social networks, and large scientific experiments. For example, one could generate an event-based data stream based on the presence or absence of a specific word in a collection of tweets. Real-time processing of event-based data streams from dynamic vision sensors is another promising application area[29]. One can also view correlation detection as a key constituent of unsupervised learning where one of the objectives is to find correlated clusters in data streams.

In a generic formulation of the problem, let us assume that there are N discrete-time binary stochastic processes arriving at a correlation detector (see Fig. 3a). Let $X_i = \{X_i(k)\}$ be one of the processes. Then $X_i(k)$ is a random variable with probabilities

$$P[X_i(k) = 1] = p \tag{2}$$

$$P[X_i(k) = 0] = 1 - p, \tag{3}$$

for $0 \le p \le 0.5$. Let $\mathbf{X}_j$ be another discrete-time binary stochastic process with the same value of parameter p. Then the correlation coefficient of the random variables $X_i(k)$ and $X_j(k)$ at time instant

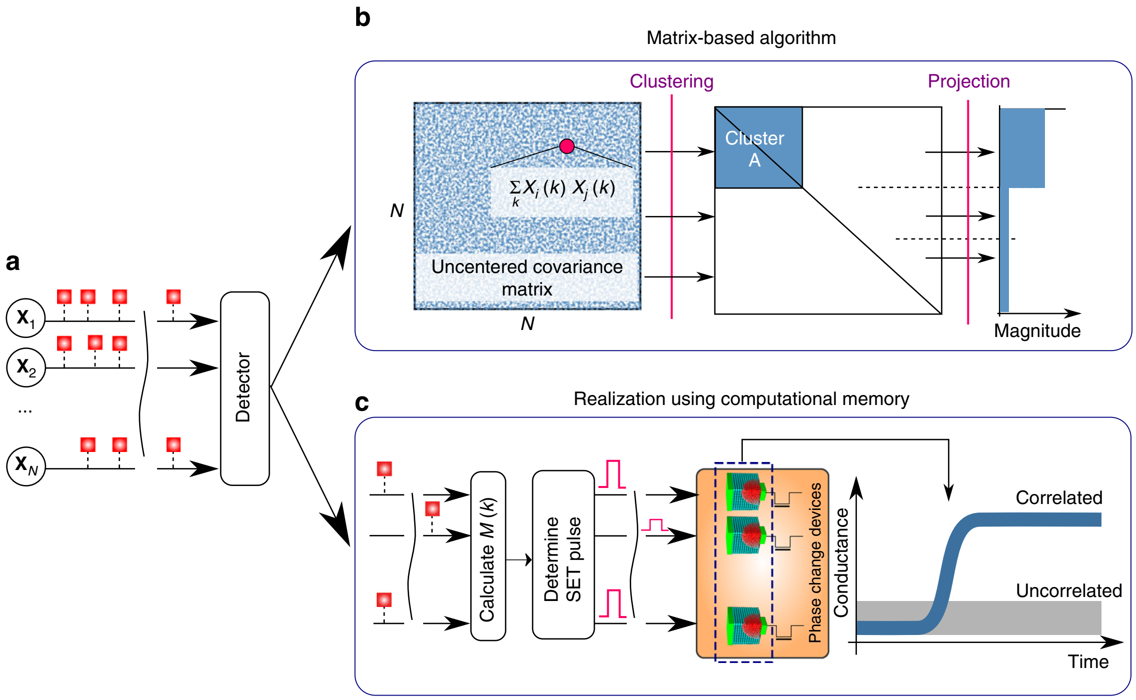

Fig. 3 Temporal correlation detection. **a** Schematic of N stochastic binary processes, some correlated and the remainder uncorrelated, arriving at a correlation detector. **b** One approach to detect the correlated group is to obtain an uncentered covariance matrix. By summing the elements of this matrix along a row or column, we can obtain some kind of numerical weights corresponding to the N processes and can differentiate the correlated from the uncorrelated group based on their magnitudes. **c** Alternatively, the correlation detection problem can be realized using computational memory. Here each process is assigned to a single phase change memory device. Whenever the process takes the value 1, a SET pulse is applied to the PCM device. The amplitude or the width of the SET pulse is chosen to be proportional to the instantaneous sum of all processes. By monitoring the conductance of the memory devices, we can determine the correlated group

k is defined as

$$c = \frac{\mathrm{Cov}\left[X_i(k), X_j(k)\right]}{\sqrt{\mathrm{Var}[X_i(k)]\mathrm{Var}\left[X_j(k)\right]}}. \tag{4}$$

Processes $\mathbf{X}_i$ and $\mathbf{X}_j$ are said to be correlated if $c > 0$ and uncorrelated otherwise. The objective of the correlation detection problem is to detect, in an unsupervised manner, an unknown subset of these processes that are mutually correlated.

As shown in Supplementary Note 1 and schematically illustrated in Fig. 3b, one way to solve this problem is by obtaining an estimate of the uncentered covariance matrix corresponding the processes denoted by

$$\hat{R}_{ij} = \frac{1}{K}\sum_{k=1}^{K} X_i(k)X_j(k). \tag{5}$$

Next, by summing the elements of this matrix along a row or column, we can obtain certain numerical weights corresponding to the processes denoted by $\hat{W}_i = \sum_{j=1}^{N} \hat{R}_{ij}$. It can be shown that if $\mathbf{X}_i$ belongs to the correlated group with correlation coefficient $c > 0$, then

$$E\left[\hat{W}_i\right] = (N-1)p^2 + p + (N_c - 1)cp(1-p). \tag{6}$$

N_c denotes the number of processes in the correlated group. In contrast, if $\mathbf{X}_i$ belongs to the uncorrelated group, then

$$E\left[\hat{W}_i\right] = (N-1)p^2 + p. \tag{7}$$

Hence by monitoring $\hat{W}_i$ in the limit of large K, we can determine which processes are correlated with $c > 0$. Moreover, it can be seen that with increasing c and N_c, it becomes easier to determine whether a process belongs to a correlated group.

We can show that this rather sophisticated problem of correlation detection can be solved efficiently using a computational memory module comprising PCM devices by exploiting the crystallization dynamics. By assigning each incoming process to a single PCM device, the statistical correlation can be calculated and stored in the very same device as the data passes through the memory. The way it is achieved is depicted schematically in Fig. 3c: At each time instance k, a collective momentum, $M(k) = \sum_{j=1}^{N} X_j(k)$, that corresponds to the instantaneous sum of all processes is calculated. The calculation of $M(k)$ incurs little computational effort as it just counts the number of non-zero events at each time instance. Next, an identical SET pulse is applied potentially in parallel to all the PCM devices for which the assigned binary process has a value of 1. The duration or amplitude of the SET pulse is chosen to be a linear function of $M(k)$. For example, let the duration of the pulse $\delta t(k) = CM(k) = C\sum_{j=1}^{N} X_j(k)$. For the sake of simplicity, let us assume that the interface temperature, T_{int}, is independent of the amorphous thickness, u_{a}. As the pulse amplitude is kept constant, $v_{\mathrm{g}}(T_{\mathrm{int}}) = \mathcal{G}$, where $\mathcal{G}$ is a constant. Then from Eq. 1, the absolute value of the change in the amorphous thickness of the i^{th} phase change device at the k^{th} discrete-time instance is

$$\delta u_{\mathrm{a}_i}(k) = \delta t(k) v_{\mathrm{g}}(T_{\mathrm{int}}) = C\mathcal{G}\sum_{j=1}^{N} X_j(k). \tag{8}$$

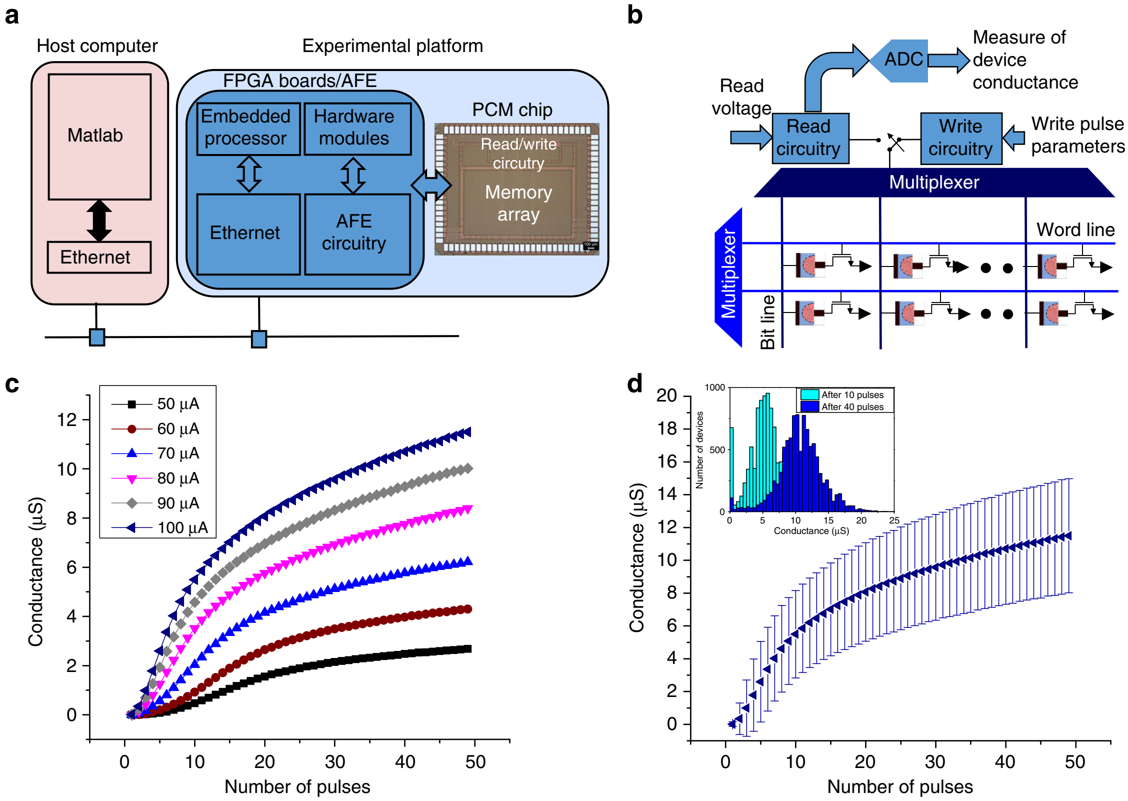

Fig. 4 Experimental platform and characterization results. **a** Schematic illustration of the experimental platform showing the main components. **b** The phase change memory array is organized as a matrix of word lines (WL) and bit lines (BL), and the chip also integrates the associated read/write circuitries. **c** The mean accumulation curve of 10,000 devices showing the map between the device conductance and the number of pulses. The devices achieve a higher conductance value with increasing SET current and also with increasing number of pulses. **d** The mean and standard deviation associated with the accumulation curve corresponding to the SET current of 100 μA. Also shown are the distributions of conductance values obtained after application of the 10th and 40th SET pulses

The total change in the amorphous thickness after K time steps can be shown to be

$$\begin{aligned} \Delta u_{\mathrm{a}_i}(K) &= \sum_{k=1}^{K} \delta u_{\mathrm{a}_i}(k) X_i(k) \\ &= C\mathcal{G} \sum_{k=1}^{K} \sum_{j=1}^{N} X_i(k) X_j(k) \\ &= C\mathcal{G} \sum_{j=1}^{N} \sum_{k=1}^{K} X_i(k) X_j(k) \\ &= KC\mathcal{G} \sum_{j=1}^{N} \hat{R}_{ij} \\ &= KC\mathcal{G} \hat{W}_i. \end{aligned} \tag{9}$$

Hence, from Equations 6 and 7, if $\mathbf{X}_i$ is one of the correlated processes, then Δu_{a_i} will be larger than if $\mathbf{X}_i$ is one of the uncorrelated processes. Therefore by monitoring Δu_{a_i} or the corresponding resistance/conductance for all phase change devices we can determine which processes are correlated.

Experimental platform. Next, we present experimental demonstrations of the concept. The experimental platform (schematically shown in Fig. 4a) is built around a prototype PCM chip that comprises 3 million PCM devices. More details on the chip are presented in the methods section. As shown in Fig. 4b), the PCM array is organized as a matrix of word lines (WL) and bit lines (BL). In addition to the PCM devices, the prototype chip integrates the circuitry for device addressing and for write and read operations. The PCM chip is interfaced to a hardware platform comprising two field programmable gate array (FPGA) boards and an analog-front-end (AFE) board. The AFE board provides the power supplies as well as the voltage and current reference sources for the PCM chip. The FPGA boards are used to implement the overall system control and data management as well as the interface with the data processing unit. The experimental platform is operated from a host computer, and a Matlab environment is used to coordinate the experiments.

An extensive array-level characterization of the PCM devices was conducted prior to the experimental demonstrations. In one experiment, 10,000 devices were arbitrarily chosen and were first RESET by applying a rectangular current pulse of 1 μs duration and 440 μA amplitude. After RESET, a sequence of SET pulses of 50 ns duration were applied to all devices, and the resulting device conductance values were monitored after the application of each pulse. The map between the device conductance and the number of pulses is referred to as accumulation curve. The accumulation curves corresponding to different SET currents are shown in Fig. 4c. These results clearly show that the mean conductance increases monotonically with increasing SET current (in the range from 50 and 100 μA) and with increasing number of SET pulses. From Fig. 4d, it can also be seen that a significant variability is associated with the evolution of the device conductance values. This variability arises from inter-device as well as intra-device variability. The intra-device variability is

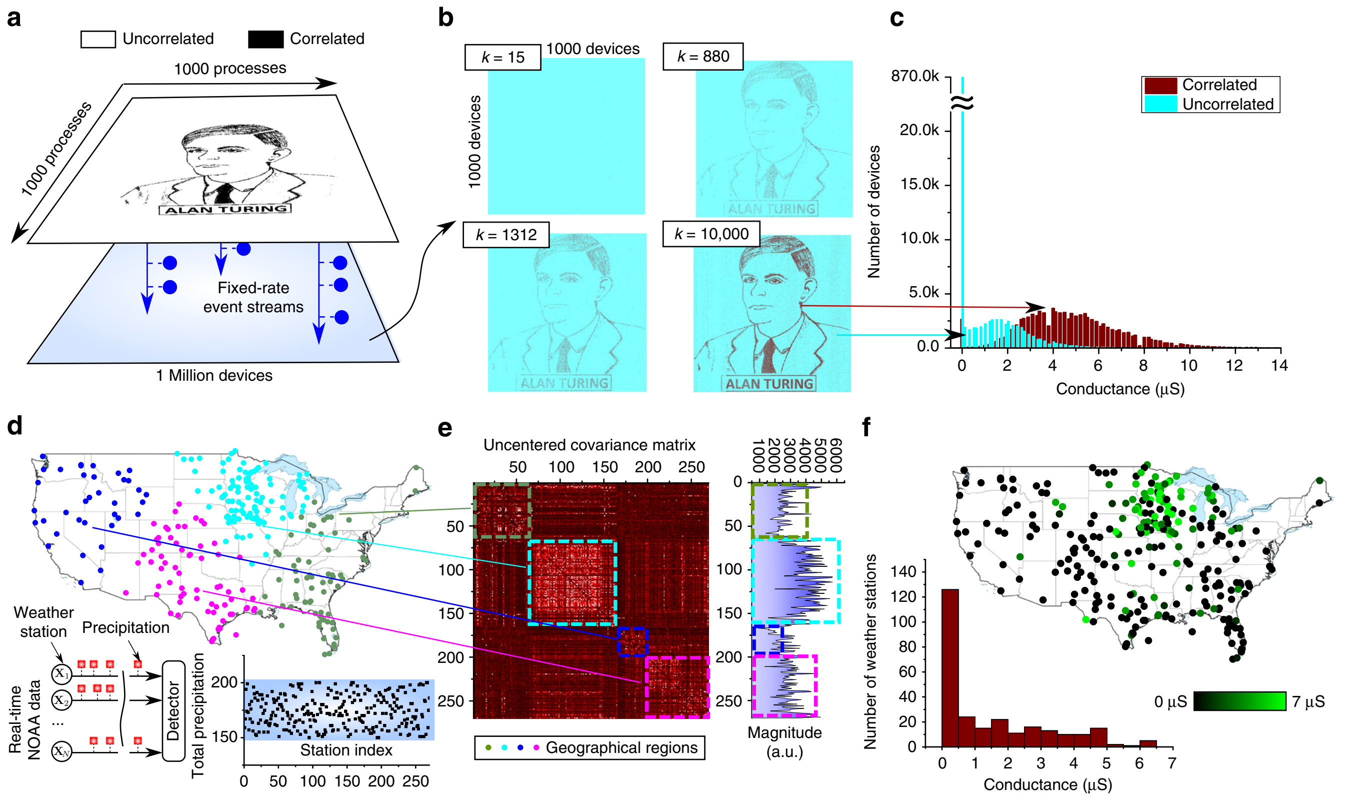

Fig. 5 Experimental results. **a** A million processes are mapped to the pixels of a 1000 × 1000 pixel black-and-white sketch of Alan Turing. The pixels turn on and off in accordance with the instantaneous binary values of the processes. **b** Evolution of device conductance over time, showing that the devices corresponding to the correlated processes go to a high conductance state. **c** The distribution of the device conductance shows that the algorithm is able to pick out most of the correlated processes. **d** Generation of a binary stochastic process based on the rainfall data from 270 weather stations across the USA. **e** The uncentered covariance matrix reveals several small correlated groups, along with a predominant correlated group. **f** The map of the device conductance levels after the experiment shows that the devices corresponding to the predominant correlated group have achieved a higher conductance value

traced to the differences in the atomic configurations of the amorphous phase created via the melt-quench process after each RESET operation[30,31]. Besides the variability arising from the crystallization process, additional fluctuations in conductance also arise from $1/f$ noise[32] and drift variability[33].

Experimental demonstration with a million processes. In a first demonstration of correlation detection, we created the input data artificially, and generated one million binary stochastic processes organized in a two-dimensional grid (Fig. 5a). We arbitrarily chose a subset of 95,525 processes, which we mutually correlated with a relatively weak instantaneous correlation coefficient of 0.1, whereas the other 904,475 were uncorrelated. The objective was to see if we can detect these correlated processes using the computational memory approach. Each stochastic process was assigned to a single PCM device. First, all devices were RESET by applying a current pulse of 1 µs duration and 440 µA amplitude. In this experiment, we chose to modulate the SET current while maintaining a constant pulse duration of 50 ns. At each time instance, the SET current is chosen to be equal to $0.002^{*}M(k)$ µA, where $M(k) = \sum_{j=1}^{N} X_j(k)$ is equal to the collective momentum. This rather simple calculation was performed in the host computer. Alternatively, it could be done in one of the FPGA boards. Next, the on-chip write circuitry was instructed to apply a SET pulse with the calculated SET current to all PCM devices for which $\mathbf{X}_i(k) = 1$. To minimize the execution time, we chose not to program the devices if the SET current was less than 25 µA. The SET pulses were applied sequentially. However, if the chip has multiple write circuitries that can operate in parallel, then it is also possible to apply the SET pulses in parallel. This process of applying SET pulses was repeated at every time instance. The maximum SET current applied to the devices during the experiment was 80 µA.

As described earlier, owing to the temporal correlation between the processes, the devices assigned to those processes are expected to go to a high conductance state. We periodically read the conductance values of all PCM devices using the on-chip read circuitry and the on-chip analog-to-digital convertor (ADC). The resulting map of the conductance values is shown in Fig. 5b. Also shown is the corresponding distribution of the conductance values (Fig. 5c). This distribution shows that we can distinguish between the correlated and the uncorrelated processes. We constructed a binary classifier by slicing the histogram of Fig. 5c according to some threshold, above which processes are labeled correlated and below which processes are labeled uncorrelated. The threshold parameter can be swept across the domain, resulting in an ensemble of different classifiers, each with its own statistical characteristics (e.g., precision and recall). The area under the precision-recall curve (AUC) is an excellent metric for quantifying the performance of the classifier. The AUC is 0.93 for the computational memory approach compared to 0.095 for a random classifier that simply labels processes as correlated with some arbitrary probability. However, the performance is still short of that of an ideal classifier with AUC equal to one and this is attributed to the variability and conductance fluctuations discussed earlier. However, it is remarkable that in spite of these issues, we are able to perform the correlation detection with significantly high accuracy. Note that there are several applications, such as sensory data processing, where these levels of accuracy would be sufficient. Moreover, we could improve the accuracy by using multiple devices to interface with a single random process and by averaging their conductance values. This concept is also illustrated in the experimental demonstration on weather data that is described next. The conductance fluctuations can also be minimized using concepts such as projected phase change memory[34].

Note that the correlations need to be detected within a certain period of time. This arises from the finite conductance range of the PCM devices. There is a limit to the u_a and hence the maximum conductance values that the devices can achieve. The accumulation curves in Fig. 4d clearly show that the mean conductance values begin to saturate after the application of a certain number of pulses. If the correlations are not detected within a certain amount of time, the conductance values corresponding to the correlated processes saturate while those corresponding to the uncorrelated processes continue to increase. Once the correlations have been detected, the devices need to be RESET, and the operation has to be resumed to detect subsequent correlations. The application of shorter SET pulses is one way to increase this time period. The use of multiple devices to interface with the random processes can also increase the overall conductance range.

As per Eq. 6, we would expect the level of separation between the distributions of correlated and uncorrelated groups to increase with increasing values of the correlation coefficient. We could confirm experimentally that the correlated groups can be detected down to very low correlation coefficients such as $c = 0.01$ (Supplementary Note 2, Supplementary Movie 1 and Supplementary Movie 2). We also quantified the performance of the binary classifier by obtaining the precision-recall curves and could show that in all cases, the classifiers performed significantly better than a baseline, random classifier (Supplementary Fig. 2). Experiments also show that there is a potential for this technique to be extended to detect multiple correlated groups having different correlation coefficients (Supplementary Note 3).

Experimental demonstration with weather data. A second demonstration is based on real-world data from 270 weather stations across the USA. Over a period of 6 months, the rainfall data from each station constituted a binary stochastic process that was applied to the computational memory at one-hour time steps. The process took the value 1 if rainfall occurred in the preceding one-hour time window, else it was 0 (Fig. 5d). An analysis of the uncentered covariance matrix shows that several correlated groups exist and that one of them is predominant. As expected, also a strong geographical correlation with the rainfall data exists (Fig. 5e). Correlations between the rainfall events are also reflected in the geographical proximity between the corresponding weather stations. To detect the predominant correlated group using computational memory, we used the same approach as above, but with 4 PCM devices interfacing with each weather station data. The four devices were used to improve the accuracy. At each instance in time, the SET current was calculated to be equal to $0.0013 \times M(k)$ µA. Next, the PCM chip was instructed to program the 270×4 devices sequentially with the calculated SET current. The on-chip write circuitry applies a write pulse with the calculated SET current to all PCM devices for which $\mathbf{X}_i(k) = 1$. We chose not to program the devices if the SET current was less than 25 µA. The duration of the pulse was fixed to be 50 ns, and the maximum SET current applied to the devices was 80 µA. The resulting device conductance map (averaged over the four devices per weather station) shows that the conductance values corresponding to the predominant correlated group of weather stations are comparably higher (Fig. 5f).

Based on a threshold conductance value chosen to be 2 µS, we can classify the weather stations into correlated and uncorrelated weather stations. This conductance threshold was chosen to get the best classifier performance (see Supplementary Note 2). We can also make comparisons with established unsupervised classification techniques such as *k*-means clustering. It was seen that, out of the 270 weather stations, there was a match for 245 weather stations. The computational memory approach classified 12 stations as uncorrelated that had been marked correlated by the *k*-means clustering approach. Similarly, the computational memory approach classified 13 stations as correlated that had been marked uncorrelated by the *k*-means clustering approach. Given the simplicity of the computational memory approach, it is remarkable that it can achieve this level of similarity with such a sophisticated and well-established classification algorithm (see Supplementary Note 4 for more details).

Discussion

The scientific relevance of the presented work is that we have convincingly demonstrated the ability of computational memory to perform certain high-level computational tasks in a non-von Neumann manner by exploiting the dynamics of resistive memory devices. We have also demonstrated the concept experimentally at the scale of a million PCM devices. Even though we programmed the devices sequentially in the experimental demonstrations using the prototype chip, we could also program them in parallel provided there is a sufficient number of write modules. A hypothetical computational memory module performing correlation detection need not be substantially different from conventional memory modules (Supplementary Note 5). The main constituents of such a module will also be a memory controller and a memory chip. Tasks such as computing $M(k)$ can easily be performed in the memory controller. The memory controller can then convey the write/read instructions to the memory chip.

In order to gain insight into the potential advantages of a correlation detector based on computational memory, we have compared the hypothetical performance of such a module with that of various implementations using state-of-the-art computing hardware (Supplementary Note 6). For this study, we have designed a multi-threaded implementation of correlation detection, an implementation that can leverage the massive parallelism offered by graphical processing units (GPUs), as well as a scale-out implementation that can run across several GPUs. All implementations were compiled and executed on an IBM Power System S822LC system. This system has two POWER8 CPUs (each comprising 10 cores) and 4 Nvidia Tesla P100 graphical processing units (attached using the NVLink interface). A detailed profiling of the GPU implementation reveals two key insights. Firstly, we find that the fraction of time computing the momentum $M(k)$ is around 2% of the total execution time. Secondly, we observe that the performance is ultimately limited by the memory bandwidth of the GPU device. We then proceed to estimate the time that would be needed to perform the same task using a computational memory module: we determine the time required to compute the momentum on the memory controller, as well as the additional time required to perform the in-memory part of the computation. We conclude that by using such a computational memory module, one could accelerate the task of correlation detection by a factor of 200 relative to an implementation that uses 4 state-of-the-art GPU devices. We have also performed power profiling of the GPU implementation, and conclude that the computational memory module would provide a significant improvement in energy consumption of two orders of magnitude (Supplementary Note 6).

An alternative approach to using PCM devices will be to design an application-specific chip where the accumulative behavior of PCM is emulated using complementary metal-oxide semiconductor (CMOS) technology using adders and registers (Supplementary Note 7). However, even at a relatively large 90 nm technology node, the areal footprint of the computational phase change memory is much smaller than that of CMOS-only approaches, even though the dynamic power consumption is comparable. By scaling the devices to smaller dimensions and by using shorter write pulses, these gains are expected to increase several fold[35,36]. The ultra-fast crystallization dynamics and non-volatility ensure a multi-time scale operating window ranging from a few tens of nanoseconds to years. These attributes are particularly attractive for slow processes, where the leakage of CMOS would dominate the dynamic power because of the low utilization rate.

It can be shown that a single-layer spiking neural network can also be used to detect temporal correlations[30]. The event-based data streams can be translated into pre-synaptic spikes to a synaptic layer. On the basis of the synaptic weights, the post-synaptic potentials are generated and added to the membrane potential of a leaky integrate and fire neuron. The temporal correlations between the pre-synaptic input spikes and the neuronal-firing events result in an evolution of the synaptic weights due to a feedback-driven competition among the synapses. In the steady state, the correlations between the individual input streams can be inferred from the distribution of the synaptic weights or the resulting firing activity of the postsynaptic neuron. Recently, it was shown that in such a neural network, PCM devices can serve as the synaptic elements[37,38]. One could argue that the synaptic elements serve as some form of computational memory. Even though both approaches aim to solve the same problem, there are some notable differences. In the neural network approach, it is the spike-timing-dependent plasticity rule and the network dynamics that enable correlation detection. One could use any passive multi-level storage element to store the synaptic weight. Also note that the neuronal input is derived based on the value of the synaptic weights. It is challenging to implement such a feedback architecture in a computational memory unit. Such feedback architectures are also likely to be much more sensitive to device variabilities and nonlinearities and are not well suited for detecting very low correlations[37,39].

Detection of statistical correlations is just one of the computational primitives that could be realized using the crystallization dynamics. Another application of crystallization dynamics is that of finding factors of numbers, which we referred to in the introduction[20]. Assume that a PCM device is initialized in such a way that after the application of X number of pulses, the conductance exceeds a certain threshold. To check whether X is a factor of Y, Y number of pulses are applied to the device, re-initializing the device each time the conductance exceeds the threshold. It can be seen that if after the application of Y pulses, the conductance of the device is above the threshold, then X is a factor of Y. Another fascinating application of crystallization dynamics is to realize matrix–vector multiplications. To multiple an $N \times N$ matrix, A, with a $N \times 1$vector, x, the elements of the matrix and the vector can be translated into the durations and amplitudes of a sequence of crystallizing pulses applied to an array of N PCM devices. It can be shown that by monitoring the conductance levels of the PCM devices, one obtains a good estimate of the matrix–vector product (Supplementary Note 8). Note that such an approach consumes only N devices compared to the existing approach based on the Kirchhoff's circuit laws that requires $N \times N$ devices.

In addition to the crystallization dynamics, one could also exploit other rich dynamic behavior in PCM devices, such as the dynamics of structural relaxation. Whenever an amorphous state is formed via the melt-quench process, the resulting unstable glass state relaxes to an energetically more favorable ideal glass state[25,40–42] (Supplementary Note 9). This structural relaxation, which codes the temporal information of the application of write pulses, can be exploited to perform tasks such as the detection of rates of processes in addition to their temporal correlations (Supplementary Note 9). It is also foreseeable that by further coupling the dynamics of these devices, we can potentially solve even more intriguing problems. Suggestions of such memcomputing machines that could solve certain non-deterministic polynomial (NP) problems in polynomial (P) time by exploiting attributes, such as the inherent parallelism, functional polymorphism, and information overhead are being actively investigated[43,44]. The concepts presented in this work could also be extended to the optical domain using devices such as photonic PCM[45]. In such an approach, optical signals instead of electrical signals will be used to program the devices. These concepts are also not limited to PCM devices: several other memristive device technologies exist that possess sufficiently rich dynamics to serve as computational memory[46]. However, it is worth noting that PCM technology is arguably the most advanced resistive memory technology at present with a very well-established multi-level storage capability[21]. The read endurance is assumed to be unlimited. There are also recent reports of more than 10^{12} RESET/SET endurance cycles[47]. Note that in our experiments, we mostly apply only the SET pulses, and in this case the endurance is expected to be substantially higher.

To summarize, the objective of our work was to realize a high-level computational primitive or machine-learning algorithm using computational memory. We proposed an algorithm to detect temporal correlations between event-based data streams that exploits the crystallization dynamics of PCM devices. The conductance of the PCM devices receiving correlated inputs evolves to a high value, and by monitoring these conductance values we can detect the temporal correlations. We performed a large-scale experimental demonstration of this concept using a million PCM devices, and could successfully detect weakly correlated processes in artificially generated stochastic input data. This experiment demonstrates the efficacy of this concept even in the presence of device variability and other non-ideal behavior. We also successfully processed real-world data sets from weather stations in the United States and obtained classification results similar to the k-means clustering algorithm. A detailed comparative study with respect to state-of-the-art von Neumann computing systems showed that computational memory could lead to orders of magnitude improvements in time/energy-to-solution compared to conventional computing systems.

Methods

Phase change memory chip. The PCM devices were integrated into the chip in 90 nm CMOS technology[32]. The phase change material is doped $Ge_2Sb_2Te_2$ (d-GST). The bottom electrode has a radius of approx. 20 nm and a length of approx. 65 nm, and was defined using a sub-lithographic key-hole transfer process[48]. The phase change material is approx. 100 nm thick and extends to the top electrode. Two types of devices are available on-chip. They differ by the size of their access transistor. The first sub-array contains 2 million devices. In the second sub-array, which contains 1 million devices, the access transistors are twice as large. All experiments in this work were done on the second sub-array, which is organized as a matrix of 512 word lines (WL) and 2048 bit lines (BL). The selection of one PCM device is done by serially addressing a WL and a BL. A single selected device can be programmed by forcing a current through the BL with a voltage-controlled current source. For reading a PCM cell, the selected BL is biased to a constant voltage of 200 mV. The resulting read current is integrated by a capacitor, and the resulting voltage is then digitized by the on-chip 8-bit cyclic ADC. The total time of one read is 1 μs. The readout characteristic is calibrated by means of on-chip reference polysilicon resistors.

Generation of 1M random processes and experimental details. Let $\mathbf{X}_r$ be a discrete binary process with probabilities $P(X_r(k) = 1) = p$ and $P(X_r(k) = 0) = 1 - p$. Using $\mathbf{X}_r$ as the reference process, N binary processes can be generated via the stochastic functions[39]

$$\theta = P(X_i(k) = 1 | X_r(k) = 1) = p + \sqrt{c}(1 - p) \tag{10}$$

$$\phi = P(X_i(k) = 1 | X_r(k) = 0) = p\left(1 - \sqrt{c}\right) \tag{11}$$

$$P(X_i(k) = 0) = 1 - P(X_i(k) = 1). \tag{12}$$

It can be shown that $E(X_i(k)) = p$ and $Var(X_i(k)) = p(1 - p)$. If two processes $\mathbf{X}_i$ and $\mathbf{X}_j$ are both generated using Eqs. 10–12, then the expectation of their product is given by:

$$E\left(X_i(k)X_j(k)\right) = P\left(X_i(k) = 1, X_j(k) = 1\right)$$

$$= \sum_{v \in \{0,1\}} P\left(X_i(k) = 1, X_j(k) = 1 | X_r(k) = v\right) P\left(X_r(k) = v\right).$$

Conditional on the value of the process $\mathbf{X}_r$, the two processes $\mathbf{X}_i$ and $\mathbf{X}_j$ are statistically independent by construction, and thus the conditional joint probability $P(X_i(k) = 1,\ X_j(k) = 1 | X_r(k) = v)$ can be factorized as follows:

$$E\left(X_i(k)X_j(k)\right) = \sum_{v \in \{0,1\}} P\left(X_i(k) = 1 | X_r(k) = v\right) P\left(X_j(k) = 1 | X_r(k) = v\right)$$

$$P\left(X_r(k) = v\right)$$

$$= \theta^2 p + \phi^2 (1 - p)$$

$$= p^2 + cp(1 - p),$$

where the final equality is obtained by substituting the preceding expressions for θ and ϕ, followed by some simple algebraic manipulation. It is then straightforward to show that the correlation coefficient between the two processes is equal to c as shown below:

$$Cov\left(X_i(k)X_j(k)\right) = E\left(X_i(k)X_j(k)\right) - E\left(X_i(k)\right)E\left(X_j(k)\right)$$

$$= p^2 + cp(1 - p) - p^2 \tag{13}$$

$$\frac{Cov\left(X_i(k)X_j(k)\right)}{\sqrt{Var(X_i)Var(X_j)}} = c$$

For the experiment presented, we chose an $\mathbf{X}_r$ where $p = 0.01$. A million binary processes were generated. Of these, $N_c = 95{,}525$ are correlated with $c > 0$. The remaining 904,475 processes are mutually uncorrelated. Each process is mapped to one pixel of a 1000 × 1000 pixel black-and-white sketch of Alan Turing: white pixels are mapped to the uncorrelated processes; black pixels are mapped to the correlated processes. The seemingly arbitrary choice of N_c arises from the need to match with the black pixels of the image. The pixels turn on and off in accordance with the binary values of the processes. One phase change memory device is allocated to each of the one million processes.

Weather data-based processes and experimental details. The weather data was obtained from the National Oceanic and Atmospheric Administration (http://www.noaa.gov/) database of quality-controlled local climatological data. It provides hourly summaries of climatological data from approximately 1600 weather stations in the United States of America. The measurements were obtained over a 6-month period from January 2015 to June 2015 (181 days, 4344 h). We generated one binary stochastic process per weather station. If it rained in any given period of 1 h in a particular geographical location corresponding to a weather station, then the process takes the value 1; else it will be 0. For the experiments on correlation detection, we picked 270 weather stations with similar rates of rainfall activity.

Data availability. The data that support the findings of this study are available from the corresponding author upon request.

Received: 15 January 2017 Accepted: 21 September 2017
Published online: 24 October 2017

References

1. Seshadri, V. et al. Buddy-ram: Improving the performance and efficiency of bulk bitwise operations using DRAM. *arXiv:1611.09988* (2016).
2. Seshadri, V. et al. in *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, 185–197 (ACM, 2013).
3. Beck, A. et al. Reproducible switching effect in thin oxide films for memory applications. *Appl. Phys. Lett.* **77**, 139–141 (2000).
4. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80–83 (2008).
5. Chua, L. Resistance switching memories are memristors. *Appl. Phys. A* **102**, 765–783 (2011).
6. Wong, H.-S. P. & Salahuddin, S. Memory leads the way to better computing. *Nat. Nanotechnol.* **10**, 191–194 (2015).
7. Borghetti, J. et al. 'Memristive' switches enable 'stateful' logic operations via material implication. *Nature* **464**, 873–876 (2010).
8. Kvatinsky, S. et al. Magic: Memristor-aided logic. *IEEE Trans. Circ. Syst. II Exp. Briefs* **61**, 895–899 (2014).
9. Zha, Y. & Li, J. Reconfigurable in-memory computing with resistive memory crossbar. In: *Proceedings of the 35th International Conference on Computer-Aided Design*, 120 (ACM, 2016).
10. Vourkas, I. & Sirakoulis, G. C. Emerging memristor-based logic circuit design approaches: A review. *IEEE Circ. Syst. Mag.* **16**, 15–30 (2016).
11. Burr, G. W. et al. Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Trans. Electron Devices* **62**, 3498–3507 (2015).
12. Shafiee, A. et al. in *Proceedings of the 43rd International Symposium on Computer Architecture*, 14–26 (IEEE Press, 2016).
13. Chi, P. et al. in *Proceedings of the 43rd International Symposium on Computer Architecture*, 27–39 (IEEE Press, 2016).
14. Bojnordi, M. N. & Ipek, E. in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 1–13 (IEEE, 2016).
15. Burr, G. W. et al. Neuromorphic computing using non-volatile memory. *Adv. Phys. X* **2**, 89–124 (2017).
16. Gallo, M. L. et al. Mixed-precision memcomputing. *arXiv:1701.04279* (2017).
17. Sheridan, P. M. et al. Sparse coding with memristor networks. *Nat. Nanotechnol.* **12**, 784–789 (2017).
18. Wright, C. D. et al. Arithmetic and biologically-inspired computing using phase-change materials. *Adv. Mater.* **23**, 3408–3413 (2011).
19. Wright, C. D., Hosseini, P. & Diosdado, J. A. V. Beyond von-Neumann computing with nanoscale phase-change memory devices. *Adv. Funct. Mater.* **23**, 2248–2254 (2013).
20. Hosseini, P. et al. Accumulation-based computing using phase-change memories with FET access devices. *IEEE Electron Device Lett.* **36**, 975–977 (2015).
21. Burr, G. W. et al. Recent progress in phase-change memory technology. *IEEE J. Emerg. Select. Top. Circ. Syst.* **6**, 146–162 (2016).
22. Sebastian, A. et al. Non-resistance-based cell-state metric for phase-change memory. *J. Appl. Phys.* **110**, 084505 (2011).
23. Le Gallo, M. et al. Subthreshold electrical transport in amorphous phase-change materials. *New. J. Phys.* **17**, 093035 (2015).
24. Sebastian, A., Le Gallo, M. & Krebs, D. Crystal growth within a phase change memory cell. *Nat. Commun.* **5**, 4314 (2014).
25. Raty, J. Y. et al. Aging mechanisms in amorphous phase-change materials. *Nat. Commun.* **6**, 7467 (2015).
26. Corinto, F., Civalleri, P. P. & Chua, L. O. A theoretical approach to memristor devices. *IEEE J. Emer. Select. Top. Circ. Syst.* **5**, 123–132 (2015).
27. Ascoli, A., Corinto, F. & Tetzlaff, R. Generalized boundary condition memristor model. *Int. J. Circ. Theory Appl.* **44**, 60–84 (2016).
28. Secco, J., Corinto, F. & Sebastian, A. Flux–charge memristor model for phase change memory. *IEEE Trans. Circ. Syst. II: Exp. Briefs* 1 (2017).
29. Liu, S.-C. & Delbruck, T. Neuromorphic sensory systems. *Curr. Opin. Neurobiol.* **20**, 288–295 (2010).
30. Tuma, T., Pantazi, A., Le Gallo, M., Sebastian, A. & Eleftheriou, E. Stochastic phase-change neurons. *Nat. Nanotechnol.* **11**, 693–699 (2016).
31. Le Gallo, M., Tuma, T., Zipoli, F., Sebastian, A. & Eleftheriou, E. in *46th European Solid-State Device Research Conference (ESSDERC)*, 373–376 (IEEE, 2016).
32. Close, G. et al. in *IEEE International Electron Devices Meeting (IEDM)*, 29–5 (IEEE, 2010).
33. Pozidis, H. et al. in *4th IEEE International Memory Workshop (IMW)*, 1–4 (IEEE, 2012).

34. Koelmans, W. W. et al. Projected phase-change memory devices. *Nat. Commun.* **6**, 8181 (2015).
35. Xiong, F. et al. Low-power switching of phase-change materials with carbon nanotube electrodes. *Science* **332**, 568–570 (2011).
36. Loke, D. et al. Breaking the speed limits of phase-change memory. *Science* **336**, 1566–1569 (2012).
37. Tuma, T. et al. Detecting correlations using phase-change neurons and synapses. *IEEE Electron Device Lett.* **37**, 1238–1241 (2016).
38. Pantazi, A. et al. All-memristive neuromorphic computing with level-tuned neurons. *Nanotechnology* **27**, 355205 (2016).
39. Gütig, R. et al. Learning input correlations through nonlinear temporally asymmetric hebbian plasticity. *J. Neurosci.* **23**, 3697–3714 (2003).
40. Boniardi, M. & Ielmini, D. Physical origin of the resistance drift exponent in amorphous phase change materials. *Appl. Phys. Lett.* **98**, 243506 (2011).
41. Sebastian, A., Krebs, D., Le Gallo, M., Pozidis, H. & Eleftheriou, E. in *IEEE International Reliability Physics Symposium (IRPS)*, MY–5 (IEEE, 2015).
42. Zipoli, F., Krebs, D. & Curioni, A. Structural origin of resistance drift in amorphous GeTe. *Phys. Rev. B* **93**, 115201 (2016).
43. Traversa, F. L. & Di Ventra, M. Universal memcomputing machines. *IEEE Trans. Neural Netw. Learn. Syst.* **26**, 2702–2715 (2015).
44. Di Ventra, M. & Pershin, Y. V. Just add memory. *Sci. Am.* **312**, 56–61 (2015).
45. Ros, C. et al. Integrated all-photonic non-volatile multi-level memory. *Nat. Photon.* **9**, 725–732 (2015).
46. Waser, R. & Aono, M. Nanoionics-based resistive switching memories. *Nat. Mater.* **6**, 833–840 (2007).
47. Kim, W. et al. in *IEEE International Electron Devices Meeting (IEDM)*, 4–2 (IEEE, 2016).
48. Breitwisch, M. et al. in *IEEE Symposium on VLSI Technology*, 100–101 (IEEE, 2007).

Acknowledgements

We would like to thank several colleagues from IBM Research–Zurich and the IBM T. J. Watson Research Center, USA in particular Haris Pozidis, Milos Stanisavljevic, Urs Egger and Matthew BrightSky. We would also like to thank Charlotte Bolliger for help with the preparation of the manuscript and Sheethal Alice Tharian for help with the artwork. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 682675).

Author contributions

A.S. and T.T. conceived the idea. A.S., N.P., and M.L.G. performed the experiments. A.S. and T.T. analyzed the data. T.P. and L.K. performed the comparative study with conventional computing systems. E.E. provided managerial support and critical comments. A.S. wrote the manuscript with input from all the authors.

Additional information

Supplementary Information accompanies this paper at doi:10.1038/s41467-017-01481-9.

Competing interests: The authors declare no competing financial interests.

Reprints and permission information is available online at http://npg.nature.com/reprintsandpermissions/

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.