







Article

Energy and Performance Trade-Off Optimization in Heterogeneous Computing via Reinforcement Learning

Zheqi Yu ¹, Pedro Machado ², Adnan Zahid ^{1,3}, Amir M. Abdulghani ⁴, Kia Dashtipour ¹, Hadi Heidari ¹, Muhammad A. Imran ¹ and Qammer H. Abbasi ^{1,*}

¹ James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, UK; z.yu.2@research.gla.ac.uk (Z.Y.); a.zahid.1@research.gla.ac.uk (A.Z.); kia.dashtipour@glasgow.ac.uk (K.D.); hadi.heidari@glasgow.ac.uk (H.H.); muhammad.imran@glasgow.ac.uk (M.A.I.)

² Computational Neurosciences and Cognitive Robotics Group, Nottingham Trent University, Nottingham NG1 4FQ, UK; pedro.baptistamachado@ntu.ac.uk

³ School of Engineering and Physical Science, Heriot Watt University, Edinburgh EH14 4AS, UK

⁴ Department of Electrical and Computer Engineering, Sultan Qaboos University, Muscat 123, Oman; amirmohamed.abdulghani@glasgow.ac.uk

* Correspondence: Qammer.Abbasi@glasgow.ac.uk

Received: 7 October 2020; Accepted: 28 October 2020; Published: 2 November 2020



Abstract: This paper suggests an optimisation approach in heterogeneous computing systems to balance energy power consumption and efficiency. The work proposes a power measurement utility for a reinforcement learning (PMU-RL) algorithm to dynamically adjust the resource utilisation of heterogeneous platforms in order to minimise power consumption. A reinforcement learning (RL) technique is applied to analyse and optimise the resource utilisation of field programmable gate array (FPGA) control state capabilities, which is built for a simulation environment with a Xilinx ZYNQ multi-processor systems-on-chip (MPSoC) board. In this study, the balance operation mode for improving power consumption and performance is established to dynamically change the programmable logic (PL) end work state. It is based on an RL algorithm that can quickly discover the optimization effect of PL on different workloads to improve energy efficiency. The results demonstrate a substantial reduction of 18% in energy consumption without affecting the application's performance. Thus, the proposed PMU-RL technique has the potential to be considered for other heterogeneous computing platforms.

Keywords: machine learning; power and performance optimisation; reinforcement learning; heterogeneous computing

1. Introduction

In an embedded system, conventional strategies of low power consumption techniques simply slow down the processor's running speed to reduce power consumption. Power-efficient computing platforms require higher energy efficiency rather than power saving. On a downside, reducing power consumption usually affects the performance efficiency and completion of tasks takes more time due to this performance degradation [1]. At present, embedded systems have been employed to perform multiple computer intensive operations, which demand higher performance and power efficiency. Hence, controlling energy consumption is essential for the effective performance of embedded systems.

The traditional dynamic voltage frequency scaling (DVFS) [2] and adaptive voltage scaling (AVS) [3] techniques are focused on the collaborative architecture of an ARM processing system (PS) [4] and field programmable gate array (FPGA) programmable logic (PL) [5], and usually depend on

the feedback controls. Furthermore, the optimum voltage and frequency can also be obtained by some modelling methods on the system power and performance, such as the static and dynamic synchronization-aware method [6], and nonlinear dynamics control modelling [7].

Both PS and PL have very distinct architectures enabling the possibility of optimising the power consumption without compromising the performance of the overall system. Unlike the PS, PL enables users to describe their architectures. In regular operation of the FPGA (PL end), the total power consumption is composed of the static power consumption, dynamic power consumption and I/O power consumption of the device. Digital logic computing does not consume a significant amount of static power. The energy-saving will mainly focus on dynamic power consumption. Through different PL work states to feed different workloads, the power consumption of internal circuit flipping can be reduced, thus realising the energy saving of dynamic power consumption. In power consumption, the static power is expressed as a logic circuit, which accounts for about 20% [8]. The additional dynamic power, which accounts for 80% of the power consumption, is related to high-frequency circuits that work with serialization and de-serialization of data and is exchanged at input/output ports (IO) [9]. Typically, dynamic power consumption may be reduced by stopping the clocks when no data is being exchanged via the IOs.

In this paper, a technique has been developed to use power measurement utility reinforcement learning (PMU-RL) as a hardware switch control algorithm so that the PL power consumption can be closely regulated. The authors have introduced an RL-algorithm-based power measurement utility (PMU) model for controlling hardware power consumption. The proposed PMU-RL is implemented in the Lynsyn board, which is connected to the Vision, Control and Sensor (VCS)-1 system (Figure A1) [10] (Available online, <https://www.sundance.com/vcs-1/>, last accessed 01/10/2019) (i.e., PL and PS) via the Joint Test Action Group (JTAG) BUS (shown in Figure 1). The results clearly demonstrate that an excellent dynamic energy saving was achieved while meeting the strict performance requirements.

The rest of the paper is structured as follows: Section 2 lists some related works on energy-saving methods based on machine learning. Section 3 describes the system-level model of the proposed algorithm architecture and procedure. Section 4 includes some test and comparison results to verify the designed PMU-RL algorithm. The discussion and future work are presented in Section 5. Finally, Section 6 summarises the paper and draws conclusions.

2. Related Work

Embedded system platforms use dynamic voltage frequency scaling (DVFS) [2] and adaptive voltage scaling (AVS) [3] technologies to save energy, and both use feedback control for power monitoring. DVFS performs the dynamic adjustment of voltage and frequency states based on variation in the workload of equipment to reduce the power consumption [11], while AVS regulates and improves the power conversion efficiency and utilisation rate by adjusting the output voltage of the electronic system power supply to address the requirements of the workload [12].

Baruah et al. [13] propose a machine-learning algorithm to reduce the energy consumption based on the random forest model, where multiple decision trees are used for frequency selection. The complexity of adaptive power management for voltage and frequency states is not suitable for supervised learning methods. Traditional power and thermal management techniques are based on prior knowledge of the hardware power consumption model and information to adjust the power policy, which is the processing executed workload/application state, such as transient and average power consumption. However, this prior information is not entirely reliable, and it cannot reflect the temporal and spatial uncertainty and changes from the environment, input data or workload/application [14]. The system-level power management policy can consider these uncertainties and variability. It is difficult to achieve the maximum efficiency with optimised resources and power management technologies when such characteristic improvements are ignored [15]. Meanwhile, the complexity and processing overheads of the different machine learning (ML) technologies depend on data types and fundamental principles. For example, compared with

unsupervised learning (such as K-means clustering) and reinforcement learning, neural networks in supervised learning have higher model complexity and require more prior information on computing [14].

Hence, the use of unsupervised learning models is deemed to be an adequate and reliable option for learning patterns from the voltage and frequency states [16]. Hardware scheduling control algorithms using reinforcement learning (RL) are suitable for heterogeneous computational power management applications. For the reinforcement learning algorithm, the main advantage is that it can achieve the best strategy for interacting with the environment without any prior knowledge about the system model. Therefore, developments in hardware can shorten developments in system processing time. This is more friendly to the development of the overall hardware and software codesign, and it achieves the best matching rate even in unknown hardware situations [17]. The hardware parameters are designed to enable a trade-off between quality and computational complexity in the power management solution. By adjusting the parameters of the hierarchical approach, several trade-offs can be achieved, which can be used to optimise energy consumption [18].

In Dhiman and Rosing [19], an RL algorithm is proposed on the multi-task system platform. The approach uses the performance metrics provided by DVFS to control the hardware core to obtain better voltage and frequency. Shen et al. [20] designed an RL solution that uses temperature, performance and power balance samples to calculate the optimum power states of the target embedded system. Fettes et al. [21] suggested an RL method that minimises throughput and delay in controlling dynamic energy. Other works focus on voltage regulator hierarchy using RL for softening voltage switching and achieving higher efficiency [16]. There are a range of works using the RL design for hardware control management that can be applied to CPUs or other on-chip components [21]. Therefore, PMU-RL is an adequate algorithm to decrease the power consumption of heterogeneous hardware architectures.

3. Methodology

Based on a heterogeneous computing platform, this paper proposes a PMU-RL algorithm to select the appropriate working state of the global system and realise energy and performance trade-off optimisation. In this work, the operation of energy-saving is carried out by collecting the hardware operation information for both the PS and PL ends, and the working state of the PL terminal hardware is controlled. An algorithm that is based on the overall topological structure of the heterogeneous model considers the energy-saving action. By collecting the operating information of modules in the heterogeneous system, the algorithm can evaluate and predict the working state of the heterogeneous system, and then select the optimal working state for the overall heterogeneous system.

3.1. System Model

The Tulipp EU (STHEM) (Available online: <https://github.com/tulipp-eu/sthem> (accessed on 1 October 2019)) project (Available online: <http://tulipp.eu/> (accessed on 30 September 2019)) provides an electronic hardware measurement platform with a power measurement utility (PMU) [22]. It has a built-in field programmable gate array (FPGA) power analysis tool. The PMU-RL algorithm in this research used the Tulipp power analysis tool [23] to control the clock state of the PL, which can manage the dynamic power consumption. It was found that the estimated clock state dynamically controlled PL operation, thereby reducing PL power. The closed-loop training for power management is carried out through the real-time power consumption information of PS and PL. The hardware aims to reduce the overall power consumption and balance the performance and power consumption on the operations. The test evaluated PMU-RL algorithm optimization performance by different application scenarios. Meanwhile, in our developed simulation environment, this method could easily be made suitable for different hardware platforms after simply modifying the hardware parameters.

Figure 1 shows the system architecture of the VCS-1 board which is based on the Xilinx UltraScale+ multi-processor systems-on-chip (MPSoC) hardware of the PS and PL ends [24]. The Lynsyn board has

a built-in microprogrammed control unit (MCU) [25] and FPGA [26], which work together with the host computer to operate the Tulipp profiling tools. The Lynsyn board based on the MCU and FPGA samples up to 1 kilo sample per second (kS/s) on the current sensors, and the PMU-RL algorithm analyses the power information to provide feedback for the optimization action in order to control the PL end with profiling tools.

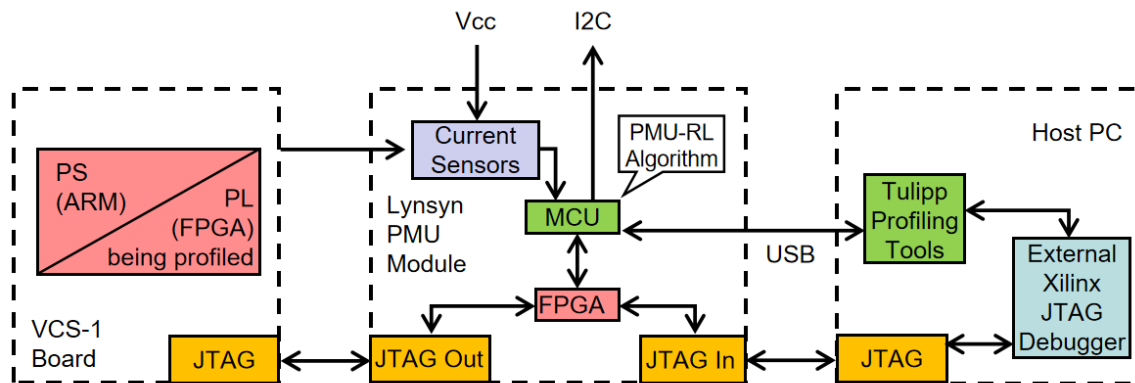


Figure 1. System architecture for field programmable gate array (FPGA) control by power management utility (PMU) module.

3.2. Power Measurement Utility (PMU)

The power measurement utility (PMU) Lynsyn board (Available online: <https://store.sundance.com/product/LynSyn/> (accessed on 1 October 2019)) is a device primarily designed for measuring currents of up to 7 different sensors and delivers up to 1 kS/s. For the PMU, it can analyse current measurements and program counter (PC) sampling simultaneously, for example by relying on the PMU to collect power consumption and PC tracking information, which helps to measure the power consumption and source code structures (including processes and loops). The PC sampling counts the power consumption for each function running time, such as operation time and number of operations. Afterwards, the Tulipp analysis utility collects and displays the data from the PMU.

Table 1 lists the power sources being monitored by the current sensors. The built-in sensors connect to different onboard units to closely measure and monitor power consumption. The purpose of each sensor is to collect the current information on different hardware modules, which assists in understanding the current total power consumption. The hardware power consumption under different workload states is monitored by the sensors. These data are then written as parameters into the simulation environment. The simulation environment can provide different power consumption information for the PL end under different workloads, which is similar to the power consumption changes in the real environment.

Table 1. Built-in current sensors to monitor power source.

Sensor	Measurement Voltage	Supplied Hardware
1	3.3 V	VCS-1 bottom board
2	5 V	VCS-1 bottom board
3	5 V	VCS-1 top board
4	3.3 V	VCS-1 top board
5	12 V	VCS-1 top board
6	12 V	VCS-1 bottom board
7	Empty	Empty

3.3. Simulation Environment

The PL end (FPGA) and PS end of power consumption data are sampled during operation. Xilinx ZYNQ MPSoC hardware operates in the power range of 1 to 6 W, which is the minimum operating power condition (220 mA operating current and 100 mW power), and can be achieved in sleep mode. These obtained data are used for setting the initial parameters for the simulation environment. Meanwhile, the environment simulates the FPGA function calling and processing rules. Two measurement applications were applied for performing simulations: ChaiDNN (Xilinx Deep Neural Network library) (Available online: <https://github.com/Xilinx/chaidnn> (accessed on 1 April 2020)) and Xilinx Deep Learning Processor Unit (DPU) (Available online: <https://www.xilinx.com/products/intellectual-property/dpu.html> (accessed on 1 April 2020)).

CHaiDNN [27] is XILINX's hardware acceleration library of deep neural network frameworks. It is designed to maximise the computational efficiency on 6-bit and 8-bit integer data that achieves the highest performance and the best accuracy. Neural networks work in the fixed-point domain for better performance. Users can convert all feature maps and training parameters from float points to fixed points under specified precision parameters [28]. DPU [29] is a convolution neural network IP core, which supports convolution sizes ranging from 1×1 , 2×2 , 3×3 , ..., to 8×8 . It can accelerate convolution computing and achieve efficient object recognition, detection and classification. The DPU computing core is designed on a full pipeline structure, and integrates a large number of convolution operators, adders and non-linear Pulling/ReLU operators, which can support quantization methods with different dynamic precision [30].

These two frameworks cover the complex scenes of neural networks and machine learning computations. They can be applied to object recognition, classification and detection applications in popular image processing areas. Different applications can be realised through the above architecture, and then the real operating power consumption information of the hardware can be acquired by current sensors. Finally, the simulation environment can be built according to the collected information. The PMU-RL algorithm follows the above simulation environment to understand the FPGA clock start and stop rules by different applications that optimise PL call times to save power.

3.4. Reinforcement Learning Algorithm

Reinforcement learning is a special machine learning method that can be trained without prior knowledge [31]. Instead, the RL algorithm learns the control behaviour, the feedback state and repeated patterns while correcting the state of behaviour during the learning phase [32]. The RL work concept is as follows: in the repeated interaction between the control behaviour of the reinforcement learning system and the state of environment feedback and evaluation, the mapping strategy from state to action is continuously modified to learn to optimise the system performance. The learning originates from the behaviour of the environment state mapping and selected strategy of the system. The learning comes from the behaviour of the environment state mapping, and the selected strategy of the system infers the optimal state for optimizing the performance whilst reducing the power consumption.

The PMU-RL algorithm learns from the power consumption patterns and measurements and controls the PL clock states. Each estimated state of the clock is rewarded when the power consumption is decreased without deteriorating the application performance. The closed-loop relationship of the VCS-1 (running the test application), Lynsyn and a host computer (Tulipp profiling tools) enables the constant validation of the estimated power consumption states. The learning processing focuses on the study of environmental status mapping and PL clock state estimation. The workflow of the PMU-RL algorithm is shown in Figure 2. The agent operates on the Lynsyn board of the PMU, and it controls the PL dynamic power consumption via the JTAG port. The test application running on the VCS-1 board provides the agent with current workload to understand the power consumption mode.

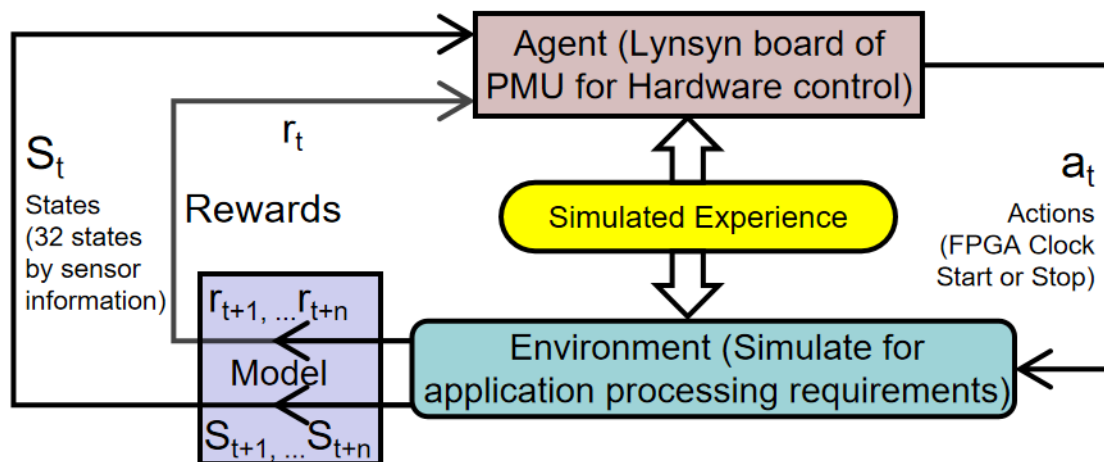


Figure 2. System setup for power measurement utility for reinforcement learning (PMU-RL) algorithm workflow.

3.5. Training Algorithm

The reinforcement learning algorithm is loaded into the built simulation environment, which tests different applications to train the algorithm. All PMU-RL and simulation environment parameters are listed in Table 2. The algorithm is based on a random function to simulate the different applications of the PL making the request for setting PL run time. Meanwhile, the random function settings can be modified to adjust for different hardware. The algorithm deploys a reward growth function that increases the impact on reward feedback to improve the learning process. Algorithm 1 describes the main steps of the proposed PMU-RL algorithm. Setting the Q values to 0 during the initialisation, the algorithm selects actions from the policies available in the Q table. Actions a , observations r and next states s' are updated by the Bellman Equation in Algorithm 1. Finally, the algorithm updates states s and s' . The above steps are repeated to reinforce learning. The algorithm evaluates the effect of learning after each iteration. The learning phase ends when the power optimisation difference is smaller than the theoretical value of power consumption. Therefore, the PMU-RL algorithm learns power consumption data based on the RL algorithm and returns the optimised action to the PMU.

Table 2. PMU-RL algorithm hyper-parameters.

Hyper-Parameter	Value
1. Train Start Size	1000 ms × n min + Random (0–1024)
2. Minibatch Size (Granularity)	64 ms
3. α (Learning Rate)	0.1
4. γ (Discount Rate)	0.9
5. ϵ (Epsilon Greedy)	0.9
6. Reward Growth Rate	0.6
7. Learning Stop	Different < 0.9%
8. FPGA Clock Stop POWER	0.1 w
9. FPGA Low POWER	1 w
10. FPGA Fully Operation POWER	6 w
11. ARM Idle POWER	1 w
12. ARM STEP POWER	2 w

Algorithm 1 Proposed PMU-RL algorithm

```

1: Initialise Q-values of table Q(s, a) to full zero matrices.
2: Observe the current state s.
Require: :
3: States S = (S1, S2..., S32)
4: Action A = A1 and A2 (PL clock start and stop)
5: Reward function: S * A → R
6: Set Greedy = 0.1, LearningRate :  $\alpha \in [0, 1]$  (typically  $\alpha = 0.1$ ), RewardDiscountFunction  $\gamma \in [0, 1]$  (typically  $\gamma = 0.9$ )
7: Update the Q-value with the reward.
8: Find a maximum reward for the next n steps state.
9: Bellman Equation:  $Q(s, a) \leftarrow Q(s, a) + \alpha [R + \gamma \max_{a'} Q(s', a') - Q(s, a)]$ 
10: Model[S][A].R = R;
11: Model[S][A].S_ = S;
12: unsigned S2, A2, S_2, float R2
13: for  $n=0; n < 4; ++n$  do
14:  $Q(s2, a2) \leftarrow Q(s2, a2) + \alpha [R2 + \gamma \max_{a'} Q(s2', a2') - Q(s2, a2)]$ ;
15: S2 = S_2; A2 = GET_ACTION_SATE(S_2);
16: A = GET_ACTION_SATE(S_);
17: S = S_;
18: end
19: return Action;
20: Set state to the new state, until S is termination.

```

At time step t , the agent perceives the corresponding state S_t from the environment, and then selects action A_t based on the condition rules. When the action is taken, the environment responds to the reward $R_t + 1$. Agents constantly interact with the environment, using a trial and error method to get the best strategy [33]. The training follows the greedy algorithm, which is set to 90% exploration and 10% prediction for action selection. The new state is controlled by the selected action and the reward R . The goal is to achieve the theoretical PL total power consumption, which is considered as a threshold in the PMU-RL algorithm. The PMU-RL learning will stop operating when the algorithm power consumption is below the theoretical data of best power consumption. Through the power consumption data captured by the sensor in the real environment, the corresponding power consumption of the whole hardware by running different applications and different workloads is achieved. The power consumption information is loaded into the simulation environment, and the corresponding workload is given by controlling the parameters. When all the power consumption of the workload in the simulated environment is known, the optimal power consumption of the system can be realised by theoretical analysis and calculation under ideal conditions. It can then be set as the training goal of the RL algorithm.

The algorithm continuously interacts with the environment through state feedback and evaluation, and constantly corrects the mapping strategy from state to action during the learning process. The expectation of state s and action a for revenue $q(s, a)$ tries to influence the reward r under the agent's action. The states and actions are stored on a Q-table that stores the Q value, then selects the action that obtains the maximum benefit according to the Q value. The reward of R_t is the performance of a given agent at the time step t . R_t is used for maximising the accumulated rewards, and the selected actions trigger this reward maximisation/punishing.

The PMU-RL algorithm collects the power states from both PS and PL. The state details are as follows.

PS and PL states: The level of smoothing power consumption depends on the direction of the power change and the updated operating conditions. The definition of high and low smoothing power consumption is as follows. When power consumption increases and then goes into a stabilised state, this is defined as high-smooth power consumption. On the contrary, when power consumption

decreases and then achieves the stabilization state, this will be characterised as low-smooth power consumption. Hence, this suggests that after the power inflection point, the smooth power consumption is obtained, and then enters a stable time interval, and no further adjustments will be needed. However, there is no numerical relationship between high and low smoothing values. In some cases, low smoothing power consumption has a higher value than high smoothing power consumption.

- Up—<Power rises>
- Down—<Power falls>
- High-smooth—<Power consumption rises and then stabilises>
- Low-smooth—<Power consumption drops and then stabilises>

PL states:

- Fully-operational—<Full load for hardware>
- Idle—<Waiting for FPGA hardware acceleration calling>
- OFF—<FPGA clock stop>

There are 32 states and 2 possible actions that are available per Q-table. These 32 states form the different definitions of PS and PL cooperative working states, including 4 changes of each PS end and PL end (power rises/power falls/high-smooth power consumption/low-smooth power consumption) and 2 action states of the PL end (work On and Off). In the algorithm decision-making phase, the information processed by the algorithm is not distributed independently, it is a combination sequence of PS and PL working states.

4. Implementation and Evaluation

The Lynsyn board collects up to 1 kS/s; therefore, the simulation environment was designed using this sampling rate for training and testing of the PMU-RL algorithm. The testing was based on simulation of an application's call function of hardware acceleration.

4.1. Test Setup

In the simulation environment, four processing cases are described which are considered as internal phase operations. There are random permutation and combination generations to simulate various test applications. In this way, the tests align with a realistic function implementation scenario, and can fulfil the different application requirements for hardware acceleration demands. The application is a periodical request known as a PL of hardware acceleration function, as the proposed algorithm is used to start and stop PL clock rules to anticipate and control the PL mode of operation in a reasonable time.

The simulation function set out calculations for PS and PL functions per time-step. The PS and PL have the following four states. In the first state, both PL and PS are in low power mode. In the second state, PS power consumption increases, and the PL remains in low power mode. In the third state, the PS is in high power mode, while the PL is in low power mode. Finally, in the fourth state, the power consumption of the PS is dropped, and PL power consumption shows increments. In the aforementioned states, the main objective is to learn and define the present states and anticipate the next states.

Figure 3 shows the one-step and multistep learning curves for the PMU-RL learning process that demonstrate the PMU-RL agent's gradual approach to additional rewards by taking appropriate action, and then increasing the energy efficiency. Meanwhile, it is also suggested that this multistep PMU-RL algorithm achieves high energy efficiency in learning faster, which means that it is more suitable for power saving. Following the results of multi-step Q learning, it is compared with the Q learning algorithm to obtain the optimal power under different steps, and the differences are shown. The algorithm of 4 steps can achieve the fastest learning effect without requiring more steps.

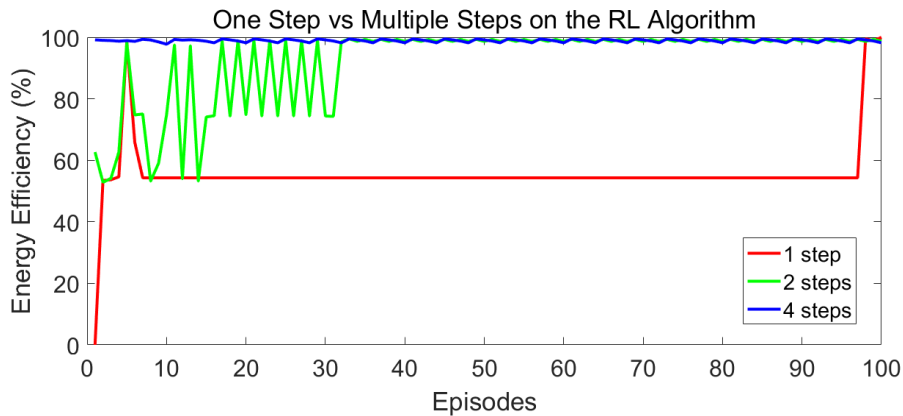


Figure 3. Comparison results for one-step and multistep reinforcement learning (RL) curves.

Reinforcement learning needs to delay the current decision for a period of time only to determine whether the right outcome result is obtained. Therefore, the adverse effect of the decision is not entirely caused by the most recent decision, and could be influenced by an error that might have occurred in the prior phase. Therefore, when selecting the estimate value, the use of n measures for Q learning has more long-term consideration than one step. The basic principle of reinforcement learning is to escape the current state fast when making mistakes, and carefully adjust to achieve convergence when making the right choice.

The n -steps output reflects the duration of time gradient to observe the impact of the current selected action on the entire system. When the current value function is required to be updated, the next n -steps state is employed to ensure that the reward is maximised. The results show that the larger the step size, the better the overall influence of the selected action. Meanwhile, it can learn to choose the best power consumption of the overall environment faster.

Figure 4 is the PMU-RL algorithm learning output for the FPGA power consumption effects and is compared to system power without the PMU-RL algorithm. According to the simulation environment settings, the PMU-RL algorithm can understand the appropriate time for the PL clock start and stop rules for different processing conditions required by hardware acceleration. Meanwhile, the energy efficiency of the system is improved, and energy consumption during idle periods is reduced.

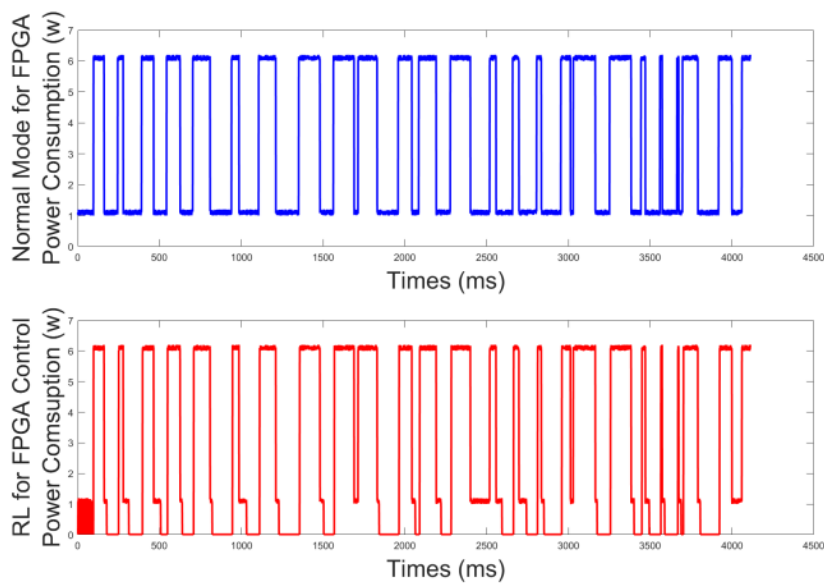


Figure 4. Comparison of power consumption effects with and without RL algorithm.

4.2. Reinforcement Learning Agent Evaluation

The reinforcement learning agent is based on off-policy learning and continued testing after every iteration of instructions. When the agent is tested, the learning algorithm ends, and its strategy remains static and deterministic. At this point, the experience is no longer applicable to the agent. The learning process should avoid the decision based on the disparity between the effects of optimisation results and theoretical values. These tests are based on different simulations of program running times to achieve various application effects. The obtained data are shown in Table 3, which shows power consumption comparison and an improvement in energy efficiency. Through the random operation of different applications in the test, the impact on energy consumption at different times is achieved. The PMU-RL improves the energy efficiency of hardware by up to 18.47%. This means that the algorithm can save power while meeting the performance requirements, and it depends on choosing a suitable time to start and stop the PL clock.

Table 3. Energy efficiency evaluation.

Energy / Times / Work	1 s	10 s	30 s	1 min	2 min
Normal Work Mode	0.98 mWh	95.40 mWh	878.46 mWh	3517.43 mWh	14,000.34 mWh
RL for Hardware Control	0.79 mWh	79.54 mWh	741.84 mWh	2976.09 mWh	11,822.75 mWh
Energy Efficiency Improvement	18.47%	16.62%	15.55%	15.39%	15.55%

5. Discussion

In this study, we aimed to provide a framework for hardware simulation to validate and evaluate the performance and power consumption balance of the Xilinx ZYNQ MPSoC platform. We simulated the running environment of the real program to demonstrate how to use the PMU module of the Lsynsy board to run the RL algorithm to achieve energy savings. Meanwhile, our method was compared to DVFS and AVS, which do not require preprocessing of frequency performance data. The RL algorithm directly provides appropriate energy-saving actions through self-learning feedback. The test results provide theoretical and algorithmic feasibility for extending this method to different heterogeneous hardware, and provide an adjustable decision-making environment to match different hardware parameters.

Table 4 shows a comparison of results achieved in this work with a couple of other similar works. Our designed method was not based on host system processing, and does not include the existing DVFS and AVS tools. In contrast, the conventional approach is based on the voltage and frequency control functions of the DVFA and AVS instruments, which depend on the host device to operate. It requires data preprocessing methods to obtain mass data resources for analysis, and then to perform the voltage and frequency scaling function of the system to save energy. Previous experimental findings were focused on the adaptation of particular applications, which makes it difficult to apply to other types of applications, such as variable proportions of the hardware acceleration requirements set.

Table 4. Comparison of energy efficiency with similar work.

Project	Hardware	Method	Power Efficiency Improved
Our Work	VSC-1 board based on Xilinx ZYNQ UltraScale+ MPSoC Chip and Lsynsyn board	RL on the MCU to control and stop the clocks when no data are being exchanged via the I/Os.	Up to 18% power reduction compared with the original model

Table 4. Cont.

Project	Hardware	Method	Power Efficiency Improved
Wei-xiong et al. [34]	Xilinx ZYNQ UltraScale+ MPSoC ZCU104	Designed fine-grained DVFS by mixed-mode clock manager (MMCM) as the clock generator	About 15% power reduction at 680 mV/285 MHz
Lilia et al. [35]	Xilinx ZYNQ ZC702	Based on the adaptive frequency scaling (AFS), which is a software configuration of the MMCM for frequency scaling and power management bus (PMBus) for voltage scaling	Up to 12% for power savings on the AFS

Our suggested method works on the peripheral hardware of the PMU module, which collects power information directly from the built-in sensors that are considered to be more precise. The RL algorithm attempts to run on the MCU of the PMU and does not consume additional computing resources and data requests. Finally, the hardware connected to the JTAG debug port with the host platform ensures the reliability of hardware control. In addition, the RL algorithm is robust for acquiring data and is easily adapted to optimise power consumption for various types of applications. The algorithm performs the actions for each context switch between tasks, for each FPGA invocation. It depends on monitoring the PL dynamic power consumption by controlling clock states.

In the future, the goal is to develop a more realistic and practical research algorithm that enables a comparison of energy-saving results in longer and more complex scenario outcomes. Finally, the hardware simulation environment will be expanded and not limited to ARM and FPGA hardware. This could control more work modes for coordinated hardware modules, such as sensors, ASICs and other integrated models.

6. Conclusions

In this work, a reinforcement-based learning approach was proposed on the Xilinx ZNYQ board, to automatically explore power usage and performance using current sensor signal processing. We used Tulipp's PMU measurement module in this heterogeneous computing platform to collect the hardware control information for the voltage and current sensors. An intelligent algorithm based on PMU-RL was employed to analyse the collected data for the hardware power information. The suggested algorithm reduces unproductive energy consumption by controlling the FPGA clock start and stop states to prevent sleeping and waking modes in order to comply with the hardware acceleration algorithm requirements. The proposed framework also studied hardware acceleration specifications in the heterogeneous computing model in the simulation environment of the hardware through the suggested energy optimisation approach and can achieve the required hardware control state to enhance energy efficiency. The new PMU-RL methodology can be applied to boards in the future to obtain real hardware impact tests. In future, the proposed PMU-RL algorithm can be deployed to boards to get the required results to investigate real hardware effects.

Author Contributions: Conceptualization, Z.Y.; methodology, Z.Y.; software, Z.Y.; validation, Z.Y.; formal analysis, Z.Y.; investigation, Z.Y.; resources, Z.Y.; data curation, Z.Y.; writing—original draft preparation, Z.Y.; writing—review and editing, P.M., A.Z., A.M.A. and K.D.; visualization, Z.Y.; supervision, H.H., M.A.I. and Q.H.A.; project administration, Q.H.A.; funding acquisition, Q.H.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research project has been partially supported by the the HiPEAC Internship programme in collaboration with Sundance Multiprocessor Technology Ltd through the European Union's Horizon 2020 research and innovation programme project: Optimisation for Energy Consumption and Performance Trade-off, grant agreement No. 779656. The VCS-1 platform has been developed under the H2020 EU project VineScout, grant agreement No. 737669 and the Tulipp profiling tools were developed in the EU H2020 project, grant agreement No. 688403. Adnan Zahid was funded by EPSRC DTG EP/N509668/1 Eng. The authors would also like to thank Sultan Qaboos University (Government of the Sultanate of Oman) for supporting Amir M. Abdulghani.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Hardware Platform

Figure A1 Hardware support vision, control and sensor (VCS)-1 system by Sundance Multiprocessor Technology Ltd. (Chesham, UK).



Figure A1. VCS-1 Hardware platform.

References

1. Ou, Z.; Pang, B.; Deng, Y.; Nurminen, J.K.; Yla-Jaaski, A.; Hui, P. Energy-and cost-efficiency analysis of arm-based clusters. In Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2012), Ottawa, ON, Canada, 13–16 May 2012; pp. 115–123.
2. Höppner, S.; Yan, Y.; Vogginger, B.; Dixius, A.; Partzsch, J.; Neumärker, F.; Hartmann, S.; Schiefer, S.; Scholze, S.; Ellguth, G.; et al. Dynamic voltage and frequency scaling for neuromorphic many-core systems. In Proceedings of the 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, USA, 28–31 May 2017; pp. 1–4.
3. Beldachi, A.F.; Nunez-Yanez, J.L. Accurate Power control and monitoring in ZYNQ boards. In Proceedings of the 2014 24th International Conference on Field Programmable Logic and Applications (FPL), Munich, Germany, 2–4 September 2014; pp. 1–4.
4. Bauer, W.; Holzinger, P.; Reichenbach, M.; Vaas, S.; Hartke, P.; Fey, D. Programmable HSA Accelerators for ZYNQ UltraScale+ MPSoC Systems. In Proceedings of the European Conference on Parallel Processing, Turin, Italy, 27–31 August 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 733–744.

5. Crockett, L.; Northcote, D.; Ramsay, C.; Robinson, F.; Stewart, R. *Exploring ZYNQ MPSoC: With PYNQ and Machine Learning Applications*; University of Strathclyde: Glasgow, UK, 2019.
6. Han, J.J.; Wu, X.; Zhu, D.; Jin, H.; Yang, L.T.; Gaudiot, J.L. Synchronization-aware energy management for VFI-based multicore real-time systems. *IEEE Trans. Comput.* **2012**, *61*, 1682–1696. [[CrossRef](#)]
7. Sapuppo, F.; Schembri, F.; Fortuna, L.; Bucolo, M. Microfluidic circuits and systems. *IEEE Circuits Syst. Mag.* **2009**, *9*, 6–19. [[CrossRef](#)]
8. Seifoori, Z.; Ebrahimi, Z.; Khaleghi, B.; Asadi, H. Introduction to emerging sram-based fpga architectures in dark silicon era. In *Advances in Computers*; Elsevier: Amsterdam, The Netherlands, 2018; Volume 110, pp. 259–294.
9. Walls, C. *Embedded Software: The Works*; Elsevier: Amsterdam, The Netherlands, 2012.
10. Oliveira, B.G.; Lobo, J. Interactive Demonstration of an Energy Efficient YOLOv3 Implementation in Reconfigurable Logic. In Proceedings of the 2019 5th Experiment International Conference, Funchal, Portugal, 12–14 June 2019; pp. 235–236.
11. Chen, Y.L.; Chang, M.F.; Yu, C.W.; Chen, X.Z.; Liang, W.Y. Learning-Directed Dynamic Voltage and Frequency Scaling Scheme with Adjustable Performance for Single-Core and Multi-Core Embedded and Mobile Systems. *Sensors* **2018**, *18*, 3068. [[CrossRef](#)] [[PubMed](#)]
12. Keller, B.A.; Nikolic, B.; Asanović, K.; Callaway, D. *Energy-Efficient System Design through Adaptive Voltage Scaling*; eScholarship; University of California: Berkeley, CA, USA, 2017.
13. Baruah, T.; Sun, Y.; Dong, S.; Kaeli, D.; Rubin, N. Airavat: Improving energy efficiency of heterogeneous applications. In Proceedings of the 2018 Design, Automation Test in Europe Conference Exhibition (DATE), Dresden, Germany, 19–23 March 2018; pp. 731–736.
14. Pagani, S.; Sai Manoj, P.D.; Jantsch, A.; Henkel, J. Machine learning for power, energy, and thermal management on multi-core processors: A survey. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2018**, *39*, 101–116. [[CrossRef](#)]
15. Liu, W.; Tan, Y.; Qiu, Q. Enhanced Q-learning algorithm for dynamic power management with performance constraint. In Proceedings of the 2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010), Dresden, Germany, 8–12 March 2010; pp. 602–605.
16. Bai, Y.; Lee, V.W.; Ipek, E. Voltage regulator efficiency aware power management. *ACM Sigops Oper. Syst. Rev.* **2017**, *51*, 825–838. [[CrossRef](#)]
17. Da Silva, L.M.; Torquato, M.F.; Fernandes, M.A. Parallel implementation of reinforcement learning q-learning technique for fpga. *IEEE Access* **2018**, *7*, 2782–2798. [[CrossRef](#)]
18. Singh, A.K.; Leech, C.; Reddy, B.K.; Al-Hashimi, B.M.; Merrett, G.V. Learning-based run-time power and energy management of multi/many-core systems: Current and future trends. *J. Low Power Electron.* **2017**, *13*, 310–325. [[CrossRef](#)]
19. Dhiman, G.; Rosing, T.S. Dynamic voltage frequency scaling for multi-tasking systems using online learning. In Proceedings of the 2007 international symposium on Low power electronics and design, Portland, OR, USA, 27–29 August 2007; pp. 207–212.
20. Shen, H.; Lu, J.; Qiu, Q. Learning based DVFS for simultaneous temperature, performance and energy management. In Proceedings of the Thirteenth International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 19–21 March 2012; pp. 747–754.
21. Fettes, Q.; Clark, M.; Bunescu, R.; Karanth, A.; Louri, A. Dynamic Voltage and Frequency Scaling in NoCs with Supervised and Reinforcement Learning Techniques. *IEEE Trans. Comput.* **2018**, *68*, 375–389. [[CrossRef](#)]
22. Sadek, A.; Muddukrishna, A.; Kalms, L.; Djupdal, A.; Podlubne, A.; Paolillo, A.; Goehringer, D.; Jahre, M. *Supporting Utilities for Heterogeneous Embedded Image Processing Platforms (STHEM): An Overview*; ARC Springer: Cham, Switzerland, 2018; pp. 737–749.
23. Kalb, T.; Kalms, L.; Göhringer, D.; Pons, C.; Marty, F.; Muddukrishna, A.; Jahre, M.; Kjeldsberg, P.G.; Ruf, B.; Schuchert, T.; et al. TULIPP: Towards ubiquitous low-power image processing platforms. In Proceedings of the 2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS), Agios Konstantinos, Greece, 17–21 July 2016; pp. 306–311.
24. Duhem, T.F.; Christensen, S.F.; Paolillo, H.A.; Kalms, R.L.; Peterson, S.M.; Schuchert, I.T.; Jahre, N.M.; Muddukrishna, N.A.; Rodriguez, H.B. Towards Ubiquitous Low-Power Image Processing Platforms. 2020. Available online: <http://tulipp.eu/wp-content/uploads/2020/06/Towards-Ubiquitous-Low-power-Image-Processing.pdf> (accessed on 1 September 2020).

25. Muddukrishna, A.; Djupdal, A.; Jahre, M. *Power Profiling of Embedded Vision Applications in the Tulipp Project*. ACM MCC17. 2017. Available online: http://tulipp.eu/wp-content/uploads/2018/06/MCC17_paper_22.pdf (accessed on 1 September 2020).
26. Kalb, T.; Kalms, L.; Göhringer, D.; Pons, C.; Muddukrishna, A.; Jahre, M.; Ruf, B.; Schuchert, T.; Tchouchenkov, I.; Ehrensträhle, C.; et al. Developing Low-Power Image Processing Applications with the TULIPP Reference Platform Instance. In *Hardware Accelerators in Data Centers*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 181–197.
27. Restuccia, F.; Biondi, A.; Marinoni, M.; Cicero, G.; Buttazzo, G. AXI HyperConnect: A Predictable, Hypervisor-Level Interconnect for Hardware Accelerators in FPGA SoC. In Proceedings of the 2020 57th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 20–24 July 2020.
28. Reiche Myrgård, M. Acceleration of Deep Convolutional Neural Networks on Multiprocessor System-on-Chip. Available online: <https://www.diva-portal.org/smash/get/diva2:1326323/FULLTEXT01.pdf> (accessed on 29 September 2020).
29. Yao, S.; Guo, K. Deep Processing Unit (Dpu) for Implementing an Artificial Neural Network (Ann). US20180046903A1, 15 February 2018.
30. Zhu, J.; Wang, L.; Liu, H.; Tian, S.; Deng, Q.; Li, J. An Efficient Task Assignment Framework to Accelerate DPU-Based Convolutional Neural Network Inference on FPGAs. *IEEE Access* **2020**, *8*, 83224–83237. [[CrossRef](#)]
31. Madden, M.G.; Howley, T. Experiments with reinforcement learning in environments with progressive difficulty. In Proceedings of the 14th Irish Conference on Artificial Intelligence and Cognitive Science, Trinity College Dublin, Dublin, UK, 17–19 September 2003.
32. Woolf, B.P. *Building Intelligent Interactive Tutors: Student-Centered Strategies for Revolutionizing E-Learning*; Morgan Kaufmann: Burlington, MA, USA, 2010.
33. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
34. Jiang, W.; Yu, H.; Zhang, J.; Wu, J.; Luo, S.; Ha, Y. Optimizing energy efficiency of CNN-based object detection with dynamic voltage and frequency scaling. *J. Semicond.* **2020**, *41*, 022406. [[CrossRef](#)]
35. Kechiche, L.; Touil, L.; Ouni, B. Toward the Implementation of an ASIC-Like System on FPGA for Real-Time Video Processing with Power Reduction. *Int. J. Reconfig. Comput.* **2018**, *2018*, 2843582. [[CrossRef](#)]

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).