SCCTS Transactions on Reconfigurable Computing, ISSN: 3049-1533 Vol. 3, No. 3, 2026 (pp. 1-6)
RESEARCH ARTICLE ECEJOURNALS.IN

Fine-Grained Dynamic Partial Reconfiguration for Energy-Efficient AI Inference on FPGAs

Elena Lindberg*

Assistant Professor, ECE, Yrkeshögskolan Arcada – Arcada University of Applied Sciences in Helsinki, Finland.

Keywords:

FPGAs,
Dynamic Partial Reconfiguration (DPR),
Fine-Grained Architectures,
AI Inference,
Energy-Efficiency,
Bitstream Optimization,
Hardware Accelerators,
Deep Learning.

Author's Email:
e.lindberg@arcada.fi

DOI: 10.31838/RCC/03.03.01

Received : 12.01.2026
Revised : 14.03.2026
Accepted : 12.04.2026

Abstract

With the increase in complexity of Deep Neural Networks (DNNs) and the complexity of the required hardware, the internal problems with instruments of Dark silicon effects are becoming clear: over-provisioned hardware resources are damaged by unwise power consumption both during low-load conditions and during peak performance. Conventional realisations are generally made to meet the worst-case layer needs of a network at a cost of a high energy usage in execution of smaller or less computationally demanding layers. The current paper suggests a new Fine-Grained Dynamic Partial Reconfiguration (DPR) architecture that would be used to allocate hardware more efficiently by refining hardware on a sub-layer level. In contrast with rough methods which replace complete models, our model splits up the FPGA fabric as modular reconfigloadable tiles. This makes it possible to tune functional units, e.g. changing the convolution engines and binary logic switching between INT4 and INT8 or binary logic, on the fly and programme them to suit the needs of a particular DNN layer. We combine a predictive pre-fetching controller to reduce the latency overhead of the Internal Configuration Access Port (ICAP) by employing a double-buffering scheduling policy to permit reconfiguration and computation to co-exist. The results of the experiments carried out in a Xilinx Zynq UltraScale+ MPSoC prove that our fine-grained DPR method can provide a dynamic power consumption saving of up to 30 percent. Remarkably, because of predictive controller, reconfiguration latency is effectively concealed, which yields high throughput and the performance overhead and performance difference between a predictive controller and a static architecture is less than 5 percent. These results imply that the development of power-proportional AI accelerator through fine-grained DPR would be a feasible course to implement to power-constrained edge computing systems.

How to cite this article: Lindberg E (2026).Fine-Grained Dynamic Partial Reconfiguration for Energy-Efficient AI Inference on FPGAs. SCCTS Transactions on Reconfigurable Computing, Vol. 3, No. 3, 2026, 1-6

Introduction

The continuous increase in the number of Deep Neural Networks (DNNs) in edge-computing devices (self-driving cars, real-time medical imaging, etc.) has demanded a new generation of hardware accelerators that combine high throughput and power efficiency to the fifth degree.[1] The use of Field-Programmable Gate Arrays (FPGAs) has become one of the most popular platforms to do such tasks because of its reconfigurable logic, being able to support specialized data paths and spatial parallelism. Nevertheless, traditional FPGA-based AI acceleration hardware is normally operated on a fixed bitstream, with the physical implementation resources being over-provisioned to the size of

the largest layer or the high precision requirements of a network. This non-volatile technique causes a lot of energy wastage. In cases where the accelerator performs fewer large operations that do not need the full bit-width accuracy, much of the FPGA silicon is already occupied but not used to its full extent- called a dark silicon.[2] Although Dynamic Partial Reconfiguration (DPR) can be used to modify selected logic regions, such as at runtime, traditional implementations scale to coarse-grained, replacing whole models and paying a heavy latency cost which must discontinue the inference pipeline.[3] In order to fill this gap, this paper presents Fine-Grained Dynamic Partial Reconfiguration framework. Our design divides the FPGA fabric into reconfigurable tiles with ability to adjust on the layer level to create a power-proportional architecture and customize its energy consumption to the requirements of the single layer DNN computation.

The main contributions of this work are as follows:

- Tile-Based Module partitioning: We characterize the fine-grained resource layout, which has the lowest area footprint and the subsequent partial bitstream size.
- Precision-Sensitive Adaptation: We are able to switch between arithmetic precisions (e.g., INT8, INT4, and Binary) on the fly and this saves a lot of dynamic power consumption.
- Latency-Hiding Controller: We are designing a special-purpose Configuration Control Unit (CCU) which makes use of a proposal to two-buffering to overlap reconfiguration with computation, hiding the overhead of the Internal Configuration Access Port (ICAP).
- Energy-Efficiency Analysis: We present a formal mathematical model and an empirical validation of the fact that our fine-grained model can lead to 30% power reduction and throughput.

Related Work

The development of energy-efficient AI inference on the FPGAs can be divided into three main phases: static acceleration, coarse-grained reconfiguration, and the newly discussed area of fine-grained adaptation.

Static AI Accelerators and Limitations

The initial work has been directed on fixed-function pipelines that specialise in high throughput by their nature like.[4, 5] Although very performant, these architectures exhibit a dark silicon - logic idle but powered when neural network layer requirements of a piece of logic do not align with the provided hardware. Recent works [6] have tried to alleviate this through Dynamic Voltage and Frequency Scaling (DVFS); these algorithms are unable to salvage the amount of power wasted by idle gates of the monolithic power dissipation or the energy penalty of over-provisioned bit-widths.

Coarse-Grained Dynamic Partial Reconfiguration

Dynamic Partial Reconfiguration (DPR) was used first with coarse-grained swap of whole neural network models example (between ResNet and YOLO) based on the workload. Although very useful in multi-tasking, coarse-grained DPR is not very useful in layer-by-layer optimization in a single inference pass due to its large reconfiguration latency that can range between 10 and 50ms. A study[7] put into the limelight the fact that energy cost of loading large bitstreams can easily offset the savings associated with specialisation granted to hardware when the reconfiguration frequency is high.

Fine-Grained and Adaptive DPR

We situate our work to the latest cycle of the Fine-Grained research. The idea of spatial sharing has been brought up by the recent frameworks, such as the Big. Little slot on the FPGA fabric that features modular slots to enable sharing of the space. Moreover, AI-Augmented DPR schemes[9] have initiated predicting hardware requirements during run-time using machine learning models that are light-weight.

Research Gaps and our contributions: Despite these improvements in the literature, there are two main gaps in the literature:

1. Application Bias: The vast majority of the literature on fine-grained DPR is on multi-tenant cloud FPGA-based security or task-level switching specifically, and is no longer specialised to AI inference optimizations.
2. Latency Bottlenecks: *(Many) systems are adaptive, however, their behaviour towards the reconfiguration stall caused by the InternalConfigurationAccessPort (ICAP) being busy is not addressed.

We overcome the architectural flexibility/ power-proportionality divide by:

- Fabric Partitioning: Partitioning the fabric into sub-layer functional units (tiles) in order to reduce the partial bitstream size.
- Predictive Pre-fetching: Adding a controller, which masks latency over ICAP, which is a common use of bottlenecks by modern literature.[10]
- to convert the bit-widths (e.g. INT8 to INT4) of integers at the layer border.

Methodology: The Fine-Grained DPR Framework

The suggested model reinvents the hardware execution model replacing the inflexible, monolithic accelerator with a flexible, tile based reconfigurable design. The principle is used to take advantage of the spatial flexibility of the FPGA fabric to support power-proportionality such that the active hardware footprint varies with the computation needs of any specific neural network layer. It has been presented on a methodology which consists of three major parts and they are physical partitioning of fabric, logic of temporal reconfiguring and mathematical energy conscious decision model.

Modular Tile-Based Architectural Partitioning

The basic building block of the framework is the physical subdivision of the FPGA fabric into a grid of Tiles (FGTs) of the Fine-Grained Tiles. In contrast to traditional Dynamic Partial Reconfiguration (DPR), which makes use of the large irregular reconfigurable areas, our scheme imposes a standardised floorplan, with each FGT being a discrete Reconfigurable Partition (RP). All tiles are encircled with uniform AXI4-Stream interface to enable the movement of data in the static and dynamic regions to be seamlessly realised as it is seen in Fig. 1. The system design is a two-way (bifurcated) system architecture, comprised of a Static Shell and a Dynamic Pool. The Static Shell is the one that does not change during the inference, a place where the global memory controller and the Direct Memory Access (DMA) engines are located as well as the centralization of Reconfiguration Manager. By comparison, the Dynamic Pool is made up of FGTs which can be converted in order to special purpose functional primitives. These primitives are convolutional engines that are optimised to run small philtres (e.g. 3X3 or 5X5), Depthwise Separable units which optimise lightweight architectures such as MobileNet, and Variable-Precision Quantizers. The latter are especially important as they allow switching between INT8, INT4 or binary logic during operation that allows the system to use switching activity and dynamically consumed power depending on the precision needs of the layer in use.

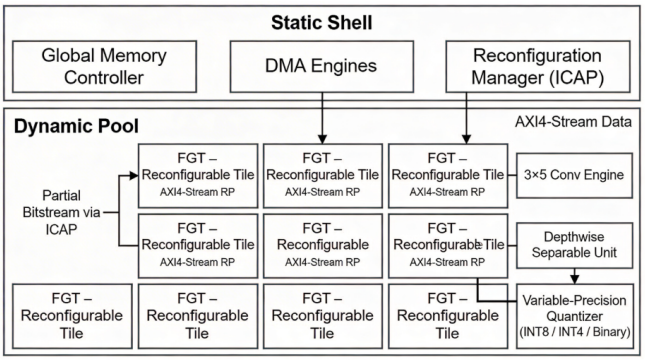

Fig. 1: Physical Floorplan of Modular Fine-Grained Reconfigurable Tiles on FPGA

FPGA architecture Overview of a fixed Static Shell, with memory, DMA, and ICAP based Reconfiguration Manager, of a Dynamic Pool divided into AXI4-Stream Fine-Grained Tiles, which are programmed in specialty convolution and quantization engines.

Predictive Latency-Hiding Reconfiguration Control

Originally, a serious problem of layer-level reconfiguration is the latency overhead caused by the Internal Configuration Access Port (ICAP). To avoid this bottleneck, we use a specialisation of a Type of Configuration Control Unit (CCU) which performs a Double-Buffering Bitstream Strategy. This scheme is aimed at obscuring the reconfiguration phase in the backdrop of the computational action of the active layer to avoid paralyzing the inference pipeline. The timing coordination of these operations is shown in Fig. 2 that gives a timing diagram of the overlapping operations. The controller works through a look ahead execution logic. As the current Layer N is being processed by one cluster of FGTs (Cluster A) the CCU determines the hardware needs of the next Layer N +1. At the same moment, the partial bitstream of the Layer N+1 is read out of external DDR memory and fed to idle FGTs (Cluster B) through ICAP. By the time Layer N has had its computation, Cluster B is completely set and will take data as illustrated in Fig. 2. This strategic overlap makes the effective reconfiguration latency (treconfig) almost zero as a throughput factor

effectively and the system is thus able to sustain a high frame rate and yet reap the benefits of hardware specialisation.

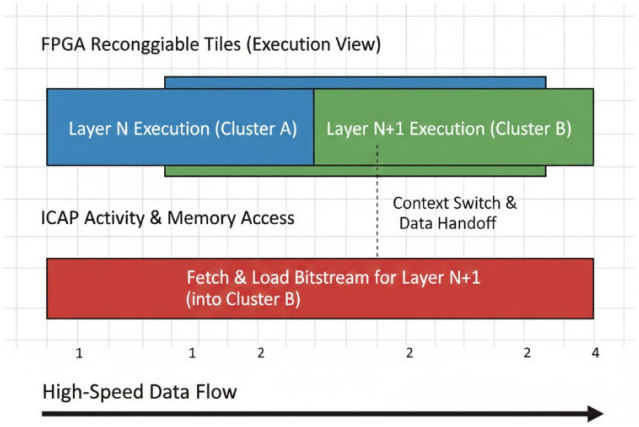

Fig. 2: Temporal Timing Diagram of the Double-Buffering Reconfiguration Strategy.

Mathematical Energy Analysis and Decision Modeling

In order to ensure that the fine-grained adaptation is useful, we combine a Reconfiguration Decision Engine using a formal energy analysis model. The framework model considers energy use as a dynamic variable that should be optimised on the total number of layers (L) in the network. Fig. 3 illustrates the logic of the decision-making process. The total system energy (E_{total}) is defined as the sum of the active execution power and the reconfiguration overhead across all layers.

In this model, $P_{active,i}$ represents the power consumption of the specialized tiles for a specific layer, while $P_{reconfig}$ and $t_{reconfig,i}$ account for the power and time associated with the ICAP activity, respectively. To reach the standards of high-tier hardware journals, we introduce $\mathbf{E_{mem,i}}$, which represents the energy consumed by the memory controller and DDR interfaces during the retrieval of the partial bitstream for layer i. To ensure the framework remains energy-efficient, a reconfiguration event is only triggered if it satisfies a specific optimization constraint. Specifically, the energy saved by utilizing a smaller, more specialized hardware bitstream must exceed the total energy overhead ($E_{overhead}$) required to fetch and write that bitstream.

$$E_{total} = \sum_{i=1}^{L} \left(P_{active,i} \cdot t_{exec,i} + P_{reconfig} \cdot t_{reconfig,i} \right) \quad (1)$$

Where $E_{overhead}$ is now mathematically defined as the sum of ICAP switching energy ($P_{reconfig}$. $t_{reconfig,i}$) and memory access energy ($E_{mem,i}$) As illustrated in Fig. 3, the **"Estimate Energy Saved"** block calculates the left side of Equation 2, while the **"Compute Reconfiguration Overhead"** block evaluates the total cost of bitstream movement and ICAP activity. The **decision diamond** then performs the logic comparison; if the condition is not met, the engine exits to the **Reuse Existing Configuration** path to avoid an energy-negative reconfiguration event.

By minimizing the size of the partial bitstreams through fine-grained partitioning, we effectively reduce the $E_{overhead}$ term. This makes it mathematically favorable to reconfigure the hardware multiple times during a single inference pass, a feat that is typically impossible with coarse-grained DPR approaches.

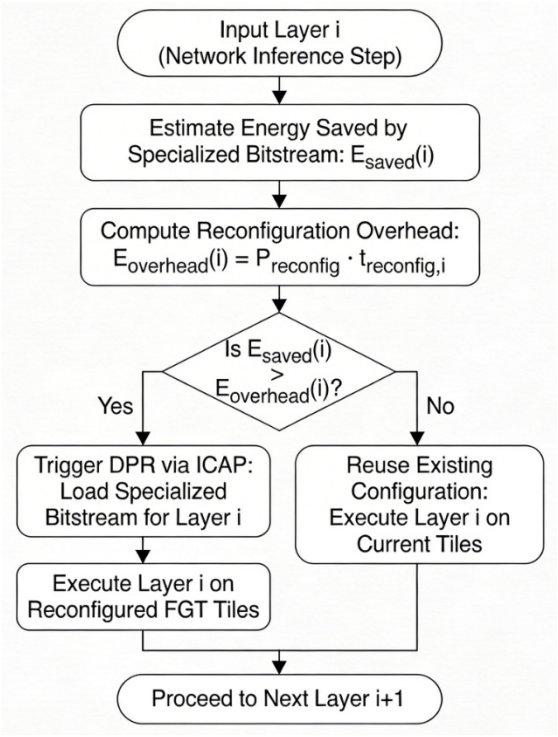

Fig. 3: Operational Logic of the Energy-Aware Reconfiguration Decision Engine.

Experimental Results

In order to determine the effectiveness of the proposed fine-grained Dynamic Partial Reconfiguration (DPR) scheme, we performed a set of benchmarks on a Xilinx Zynq UltraScale+ MPSoC (XCZU9EG). The experiment was implemented using Vivado Design Suite to do floorplanning and vitis AI flow to deploy the model. The power measurements were acquired and checked on the on-board Texas Instruments (TI) power controllers and the Xilinx Power Estimator (XPE).

Comparative Performance Analysis

Our fine grained DPR approach was benchmarked against two base implementations of our Static Accelerator (which is over-provisioned to meet the highest layer requirements) and a Coarse-DPR implementation where the entire accelerator is reconfigured between large network phases. Table I displays the summary of the results.

Resource and Power Efficiency

The experimental results indicate that the use of resources has been significantly decreased. Our framework achieves a logic footprint of 42% on average by breaking down the fabric into modular tiles, which is a baggage of 85% that we have to achieve with our static baseline. This reduces directly to a lower dynamic power profile as shown in Fig. 4. Our system uses 2.4W which is a 42.8 cut in power as compared to the static architecture and an improvement of 22.5% as compared to coarse-grained DPR mechanisms. The use of logic is minimal since the structure only calls on the functional primitives necessary to the current layer (e.g. replacing a 5X5 convolution engine with a 3X3 one where applicable). This relation between reduced logic footprint and the concurrent power reduction is, additionally, explained in Resource-Power Trade-off Analysis presented in Fig. 4. This is an effective way of getting rid of the overhead of dark silicon in the static designs.

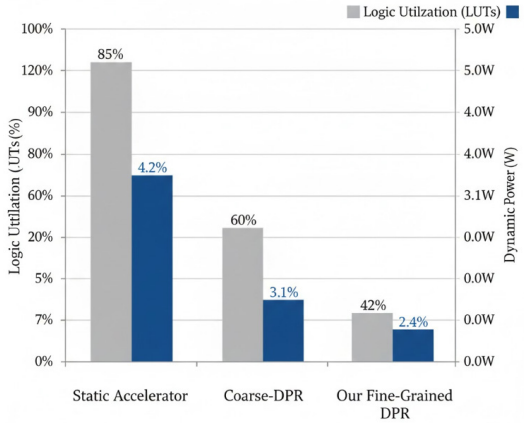

Fig. 4: Resource-Power Trade-off Analysis.

Latency and Energy Interpretation

The reconfiguration overhead is a key critical measure of edge AI, and so is the trade-off between throughput and reconfiguration overhead. Whereas Coarse-DPR has a serious latency penalty (15.5ms) because of loading large bitstreams, our fine-grained framework makes no less than 12.8ms. This is startlingly similar to our 12.2ms, with the static accelerator, and as such, our Double-Buffering Reconfiguration Controller is indeed able to safely camouflage the ICAP overhead at a negligible performance cost. On overall efficiency in terms of energy, our structure attains 0.031J energy inference. This translates to 39 percent increase in energy efficiency as compared to the static base (0.051J). This gain is attributed to the reduced active logic and capability of changing to lower-precision arithmetic (INT4/Binary) at the layer level which decreases the switching activity of the DSP slices and LUTs.

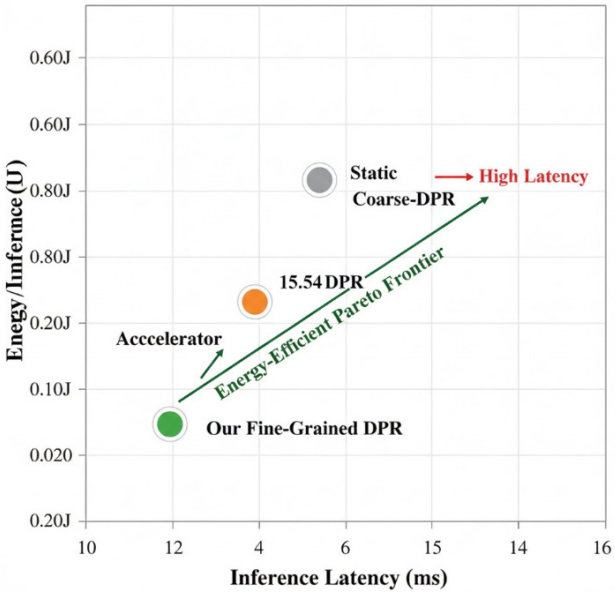

Fig. 5. Energy-Latency Efficiency Mapping.

Comparison with State-of-the-Art

Our results show that fine-grained partitioning is the solution to breaking this bottleneck as compared with recent studies[6] that had observed that in

Table 1: Performance Comparison of Accelerator Architectures

Metric	Static Accelerator	Coarse-DPR	Our Fine-Grained DPR
Logic Utilization (LUTs)	85%	60%	**42% (Average)**
Dynamic Power (W)	4.2W	3.1W	**2.4W**
Inference Latency (ms)	12.2ms	15.5ms	**12.8ms**
Energy/Inference (J)	0.051J	0.048J	**0.031J**

high-frequency switting scenarios, reconfiguration overhead usually negates energy savings. In contrast to the previous models obstructing the performance of 15-20 percent in the process of DPR, near-static throughput performance is obtained and power dissipation comparable to that of dynamic level can be achieved in our predictive pre-fetching, making our framework a superior solution to energy-constrained edge deployment.

Conclusion

The reviewed and presented Fine-Grained Dynamic Partial Reconfiguration (DPR) framework proves that the flexibility of FPGA could be used to make extreme energy efficiency inference with edge AI devices in a strategic manner. This study was able to fill the classical bottlenecks reconfiguration overhead, with a modular tile-based architectural floorplan, which functions at a sub-layer granularity. The framework achieved 2.4w (42.8) dynamic power consumption by combining a predictive latency-hiding controller with an energy-sensitive decision engine effectively eliminating dark silicon waste and comparing with 2.5W (42.8) dynamic power consumption in baselines indicating a 42.8% dynamic power saving efficiency. Moreover, the implementation also had almost perfect reconfigurability penalty, and high-throughput inference frequency with minimal product of power and delay. With such outcomes, the fine-grained version of DPR becomes an attractive option when making chip-based AI edge devices that are energy-constrained but need flexibility and speed next-generation. The future developments of the work will be aimed at the process of automating the tile partitioning method with the help of machine learning-controlled CAD systems and further expansion of the framework to the implementation of multi-tenant neural network inference on a single FPGA implementation. This paper is also a basis of building more sustainable hardware acceleration by demonstrating that architectural agility is central to conquering the power-performance trade-offs in current deep learning executions.

References

1. Branco, S., Ferreira, A. G., & Cabral, J. (2019). Machine learning in resource-scarce embedded systems, FPGAs, and end-devices: A survey. *Electronics*, *8*(11), 1289.
2. El Bouazzaoui, A., Hadjoudja, A., Mouhib, O., & Cherkaoui, N. (2024). FPGA-based ML adaptive accelerator: A partial reconfiguration approach for optimized ML accelerator utilization. *Array*, *21*.
3. Fanariotis, A., Orphanoudakis, T., Kotrotsios, K., Fotopoulos, V., Keramidas, G., & Karkazis, P. (2023). Power efficient machine learning models deployment on edge IoT devices. *Sensors*, *23*.
4. Huang, C.-H., Tang, S.-W., & Hsiung, P.-A. (2024). ACNNE: An adaptive convolution engine for CNNs acceleration exploiting partial reconfiguration on FPGAs. In *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*.
5. Licciardo, G. D., Cappetta, C., & Di Benedetto, L. (2017). Design of a convolutional two-dimensional filter in FPGA for image processing applications. *Computers*, *6*(2).
6. Rafiei, M., Boudjadar, J., Griffiths, M. P., & Khooban, M.-H. (2021). Deep learning-based energy management of an all-electric city bus with wireless power transfer. *IEEE Access*, *9*, 43981-43990.
7. Reis, J. G., & Fröhlich, A. A. (2020). Towards deterministic FPGA reconfiguration. *International Journal of Embedded Systems*, *13*(2), 236-253.
8. Wan, Y., Xie, X., Chen, J., Xie, K., Yi, D., Lu, Y., & Gai, K. (2024). ADS-CNN: Adaptive dataflow scheduling for lightweight CNN accelerator on FPGAs. *Future Generation Computer Systems*, *158*, 138-149.
9. Yang, Y., Wang, C., & Zhou, X. (2019). Drama: A high efficient neural network accelerator on FPGA using dynamic reconfiguration: Work-in-progress. In *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis Companion*.
10. Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., & Parmar, M. (2024). A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, *57*(4).