Vol 57, No 2—April 2023

# Hardware Conversion of Convolutional Neural Networks: What Is Machine Learning?—Part 3

**Ole Dreessen**, Field Applications Staff Engineer

## Abstract

In this three-part series, we have been exploring the properties and applications of convolutional neural networks (CNNs), which are mainly used for pattern recognition and the classification of objects. Part 3 will explain the hardware conversion of a CNN and specifically the benefits of using an artificial intelligence (AI) microcontroller with a hardware-based CNN accelerator—a technology that is enabling AI applications at the edge of the Internet of Things (IoT). Previous articles in this series are "Introduction to Convolutional Neural Networks: What Is Machine Learning?—Part 1" and "Training Convolutional Neural Networks: What Is Machine Learning?—Part 2."

## Introduction

AI applications require massive energy consumption, often in the form of server farms or expensive field programmable gate arrays (FPGAs). The challenge lies in increasing computational power while keeping energy consumption and costs low. Now, AI applications are seeing a dramatic shift enabled by powerful Intelligent Edge computing. Compared to traditional firmware-based computation, hardware-based convolutional neural network acceleration is now ushering in a new era of computational performance with its impressive speed and power. By enabling sensor nodes to make their own decisions, Intelligent Edge technology dramatically reduces data transmission rates over 5G and Wi-Fi networks. This is powering emerging technologies and unique applications that were not previously possible. For example, smoke/fire detectors in remote locations or environmental data analysis right at the sensor level become reality—all with years of usage on a battery supply. To examine how these capabilities are made possible, this article explores the hardware conversion of a CNN with a dedicated AI microcontroller.

## Artificial Intelligence Microcontroller with Ultra Low Power Convolutional Neural Network Accelerator

The MAX78000 is an AI microcontroller with an ultra low power CNN accelerator, an advanced system on chip. It enables neural networks at ultra low power for resource-constrained edge devices or IoT applications. Such applications include object detection and classification, audio processing, sound classification, noise cancellation, facial recognition, time-series data processing for heart rate/health signal analysis, multisensor analysis, and predictive maintenance.

Figure 1 shows a block diagram of the MAX78000, which is powered up to 100 MHz by an Arm® Cortex®-M4F core with a floating-point unit. To give applications sufficient memory resources, this version of the microcontroller comes with 512 kB of flash and 128 kB of SRAM. Multiple external interfaces are included such as $I^2C$s, SPIs, and UARTs, as well as the $I^2S$—which are important for audio applications. Additionally, there is an integrated 60 MHz RISC-V core. The RISC-V copies data from/to the individual peripheral blocks and the memory (flash and SRAM), making it a smart direct memory access (DMA) engine. The RISC-V core preprocesses the

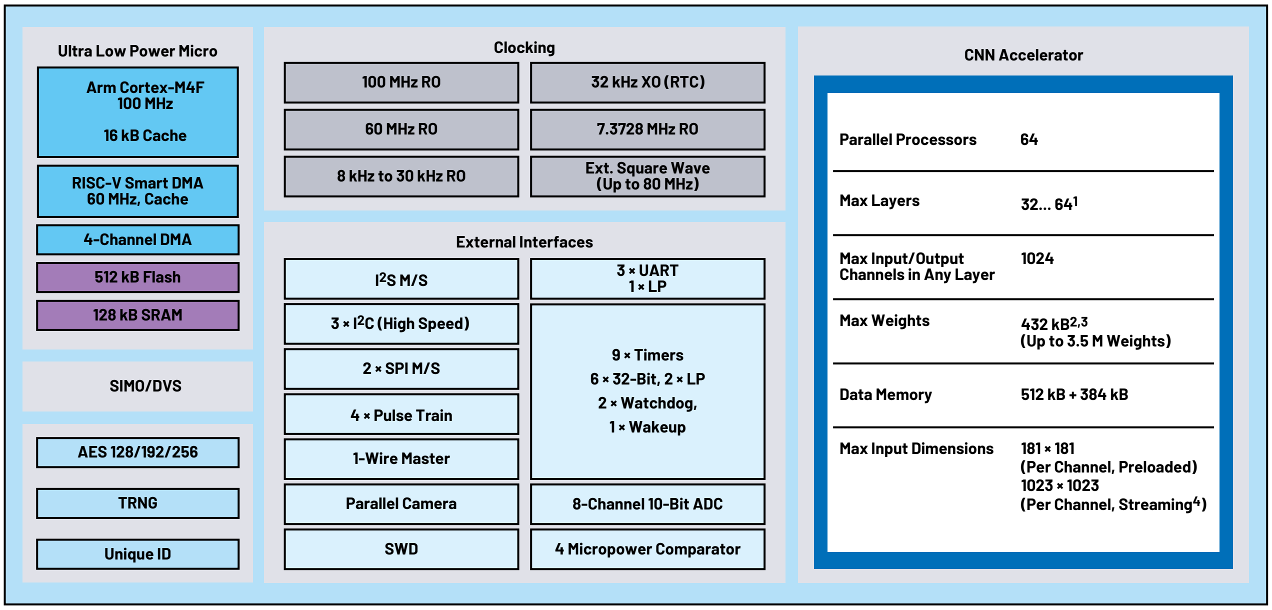

*Figure 1. A MAX78000 block schematics.*

sensor data for the AI accelerator, so the Arm core can be in a deep sleep mode during this time. If necessary, the inference result can trigger the Arm core via an interrupt, and the Arm CPU then performs actions in the main application, passes on sensor data wirelessly, or informs the user.

A hardware accelerator unit for performing inference of convolutional neural networks is a distinct feature of the MAX7800x series of microcontrollers, which sets it apart from the standard microcontroller architecture and peripherals. This hardware accelerator can support complete CNN model architectures along with all the required parameters (weights and biases). The CNN accelerator is equipped with 64 parallel processors and an integrated memory with 442 kB for storing the parameters and 896 kB for the input data. Because the model and parameters are stored in SRAM memory, they can be adjusted via firmware and the network can be adapted in real time. Depending on whether 1-, 2-, 4-, or 8-bit weights are used in the model, this memory can be sufficient for up to 3.5 million parameters. Because the memory capabilities are an integral part of the accelerator, the parameters do not have to be fetched via the microcontroller bus structure with each consecutive mathematical operation. This activity is costly due to high latencies and high power consumption. The neural network accelerator can support 32 or 64 layers, depending on the pooling function. The programmable image input/output size is up to 1024 × 1024 pixels for each layer.

## CNN Hardware Conversion: Energy Consumption and Inference Speed Comparison

CNN inference is a complex calculations task comprising large linear equations in matrix form. Using the power of Arm Cortex-M4F microcontrollers, CNN inference on an embedded system's firmware is possible; however, there are certain drawbacks to consider. With firmware-based inference running on microcontrollers, energy and time are heavily consumed as the commands needed for calculation, along with associated parameter data, need to be retrieved from memory before intermediate results can then be written back.

Table 1 presents a comparison of CNN inference speed and energy consumption utilizing three different solutions. This example model was developed using MNIST, a handwritten digit recognition training set, which classifies digits and letters from visual input data to arrive at an accurate output result. The inference time required by each processor type was measured to determine differences between energy consumption and speed.

**Table 1. CNN Inference Time and Energy per Inference for Three Different Scenarios Utilizing the MNIST Dataset for Handwritten Digit Recognition**

| Scenario | Inference Speed (ms) | Energy per Inference (µWs) |
|---|---|---|
| (1) MAX32630, MNIST network in firmware | 574 | 22887 |
| (2) MAX78000, MNIST network in hardware | 1.42 | 20.7 |
| (3) MAX78000, MNIST network in hardware, optimized for low energy consumption | 0.36 | 1.1 |

In the first scenario, an Arm Cortex-M4F processor integrated into the MAX32630, which runs at 96 MHz, was used to compute inference. In the second scenario, to process the computations, the MAX78000's hardware-based CNN accelerator was used. The inference speed—that is, the time between the presentation of the visual data at the network input and the output of the result—is lower by a factor of 400 when using a microcontroller with a hardware-based accelerator (MAX78000). Moreover, the required energy per inference is a factor of 1100 lower. In a third comparison, the MNIST network was optimized for minimal energy consumption per inference. The accuracy of the result drops in this case from 99.6% to 95.6%. However, the network is much faster, requiring just 0.36 ms per inference. The energy consumption is reduced to a mere 1.1 µWs per inference. In applications that use two AA alkaline batteries (a total of 6 Wh of energy), five million inferences are possible (power consumed by the rest of the circuit is omitted).

These data illustrate the power of hardware-accelerated computation. Hardware-accelerated computing is an invaluable tool for applications unable to utilize connectivity or a continuous power supply. The MAX78000 enables edge processing without the demand for large amounts of energy, broadband internet access, or prolonged inference times.

## Example Use Case for the MAX78000 AI Microcontroller

The MAX78000 enables a multitude of potential applications, but let's examine the following use case as an example. The requirement is to design a battery-powered camera that detects when a cat is in the field of view of its image sensor and consequently enables access to the house, via a digital output through the cat door.

Figure 2 depicts an example block diagram for such a design. In this case, the RISC-V core switches the image sensor on at regular intervals and the image data is loaded into the CNN powered by the MAX78000. If the probability of a cat recognition is above a previously defined threshold, the cat door is enabled. The system then returns to standby mode.

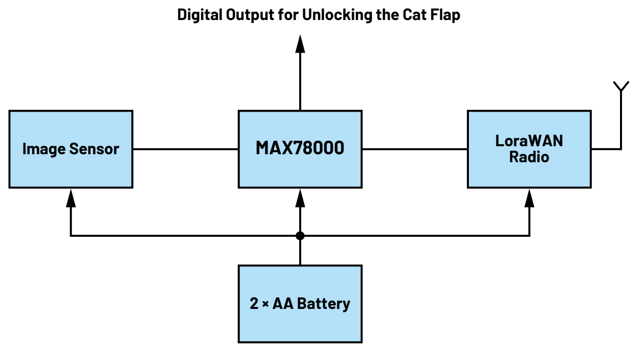

*Figure 2. A block diagram of a smart pet door.*

## Development Environments and Evaluation Kits

The process of developing an AI-on-the-edge application can be divided up into the following phases:

Phase 1: AI–Definition, training, and quantization of the network

Phase 2: Arm firmware–Inclusion of the networks and parameters generated in Phase 1 in the C/C++ application and creation and testing of the application firmware

The first part of the development process involves modeling, training, and evaluating the AI models. For this stage, the developer can leverage open-source tools such as PyTorch and TensorFlow. The GitHub repository provides comprehensive resources to help users map out their journey in building and training AI networks using the PyTorch development environment while taking into consideration the hardware specifications of the MAX78000. Included in the repository are a few simple AI networks and applications like facial recognition (Face ID).

Figure 3 shows the typical AI development process in PyTorch. First, the network is modeled. It must be noted that not all MAX7800x microcontrollers have hardware that supports all data manipulations available in the PyTorch environment. For this reason, the file ai8x.py supplied by Analog Devices must first be included in the project. This file contains the PyTorch modules and operators required for using the MAX78000. Based on this setup, the network can be built and then trained, evaluated, and quantized using the training data. The result of this step is a checkpoint file that contains the input data for the final synthesis process. In this final process step, the network and its parameters are converted to a form that fits into the hardware CNN accelerator. It should be mentioned here that network training can be done with any PC (notebook, server, etc.). However, without CUDA graphics card support, this can take a lot of time—even for small networks, days or even weeks are completely realistic.

In Phase 2 of the development process, the application firmware is created with the mechanism of writing data to the CNN accelerator and reading the results. The

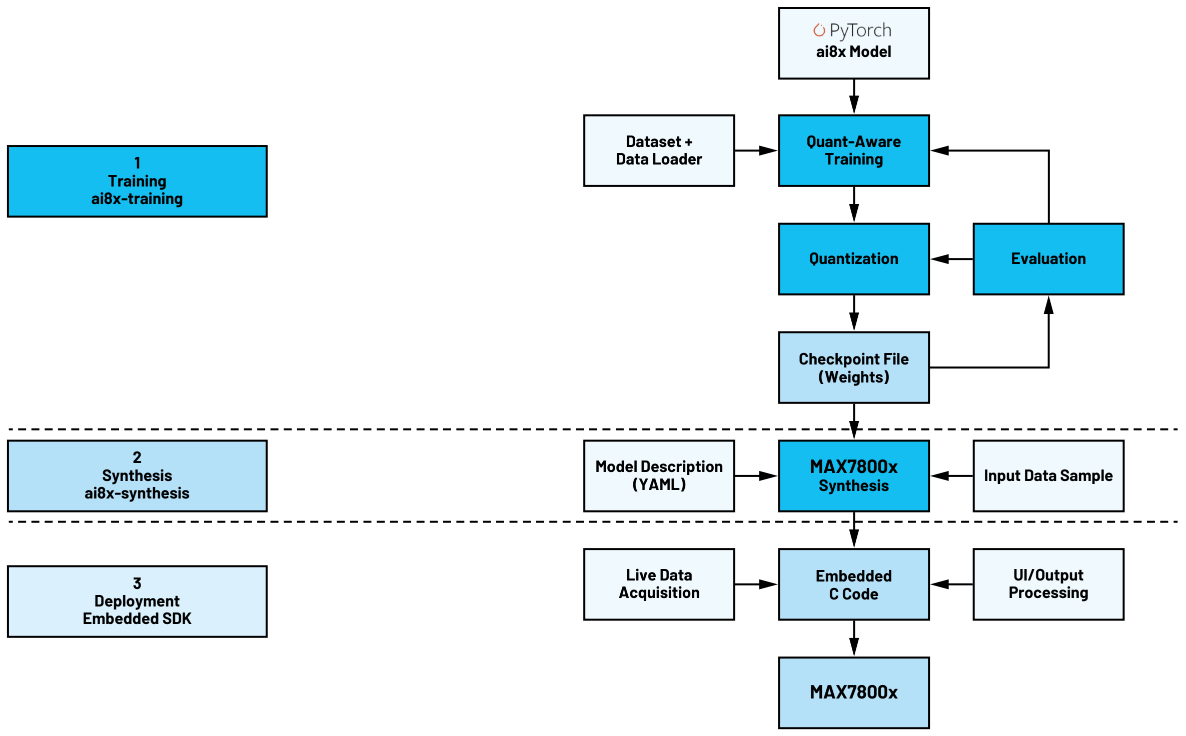

*Figure 3. An AI development process.*

files created in the first phase are integrated into the C/C++ project via #include directives. Open-source tools such as Eclipse IDE and the GNU Toolchain are also used for the development environment for the microcontroller. ADI provides a software development kit (Maxim Micros SDK (Windows)) as an installer that already contains all the necessary components and configurations. The software development kit also contains peripheral drivers as well as examples and instructions to ease the process of developing applications.

Once the project has been compiled and linked without any errors, it can be evaluated on the target hardware. ADI has developed two different hardware platforms for this purpose. Figure 4 shows the MAX78000EVKIT, and Figure 5 shows the MAX78000FTHR, which is a somewhat smaller, feather form factor board. Each board comes with a VGA camera and a microphone.

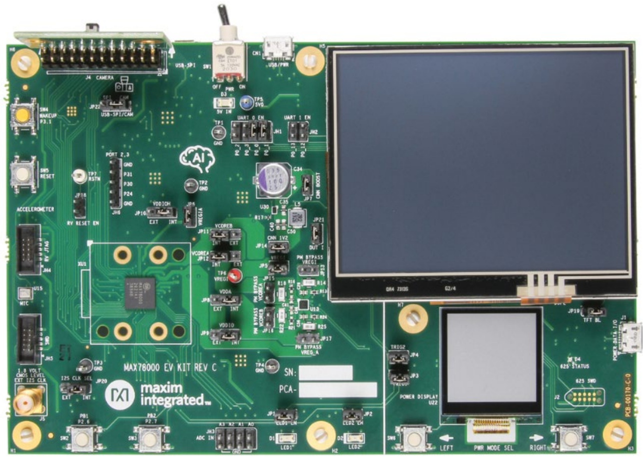

*Figure 4. A MAX78000 evaluation kit.*

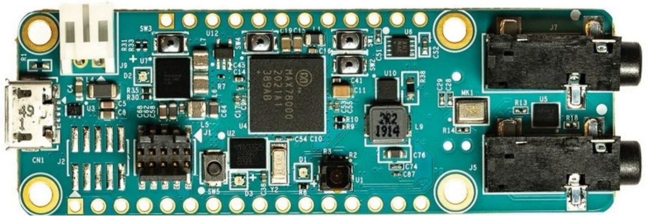

*Figure 5. A MAX78000FTHR evaluation kit.*

## Conclusion

Previously, AI applications required massive energy consumption in the form of server farms or expensive FPGAs. Now, with the MAX78000 family of microcontrollers with a dedicated CNN accelerator, it's possible to power AI applications from a single battery for extended periods. This breakthrough in energy efficiency and power is making edge-AI more accessible than ever before and unlocking the potential for new and exciting edge-AI applications that were previously impossible. For more information, visit Ultra Low Power Artificial Intelligence (AI) MCUs.

## References

"Session 2 - AI at the Edge: A Practical Introduction to Maxim Integrated's MAX78000 AI Accelerator." Analog Devices, Inc.

Video Series: Understanding Artificial Intelligence. Analog Devices, Inc.

The PyTorch logo is used with permission under the Creative Commons Attribution-Share Alike 4.0 International license.

## About the Author

Ole Dreessen is a field applications staff engineer at Analog Devices. Prior to joining ADI in 2014, he held positions at Avnet Memec and Macnica, supporting communication technologies and high performance microprocessors. Ole has broad expertise in microcontroller and security topics and is an experienced presenter at conferences and distribution events. In his free time, he is an active member of the Chaos Computer Club, where he has worked on concepts such as reverse engineering and embedded security.