



Akida Pico IP: Always-On Tire Monitoring at the Wheel

Version: v1
August 2026

Contents

Technology White Paper

Authors

Ritik Shrivastava
Solutions Architect

Diana Vins
Machine Learning Intern

Nikunj Kotecha
Senior Solutions Architect

BrainChip Inc.

23041 Avenida de Carlotta, #250
Laguna Hills, CA 92653
www.brainchip.com

August 2026

3	Chapter 1	Tire Monitoring on Akida Pico
5	Chapter 2	Key Features and Differentiators
6	Chapter 3	Architecture Overview
9	Chapter 4	The Tire Signal
10	Chapter 5	TENNs Compute Model
12	Chapter 6	Reference Model: Pressure and Load
13	Chapter 7	Validation and Results
14	Chapter 8	Power, Throughput, and Clock
15	Chapter 9	Conclusion
16	References	

Chapter 1

Tire Monitoring on Akida Pico

1.1 The Opportunity

Every passenger vehicle on the road carries a tire pressure monitoring system (TPMS) sensor at each wheel and has done so since regulation made it mandatory in the major automotive markets [1]. These sensors ship in hundreds of millions of units a year, on a cost-driven bill of materials that suppliers work continuously to reduce.

The sensor sits on the rotating assembly, in mechanical contact with the road through the tire. Modern integrated direct-TPMS sensor ICs already carry more than a pressure transducer. In fact, parts from the major suppliers combine pressure, temperature, an MCU, an RF transmitter, and a MEMS accelerometer, the last fitted for wake detection and wheel auto-location [2].

That accelerometer sees the tire deforming on every revolution. The same signal that wakes the sensor carries the characteristics needed to estimate several tire and road conditions: inflation pressure, load, tread wear, and road surface. A low-power, small-area neural processing block reading that one sensor turns a component already in the module into an always-on tire-monitoring node.

Three constraints have kept that from happening:

- ⌘ **Power.** The module is sealed around a small battery that is not replaced for the service life of the vehicle. Any added function must run within a microwatt budget.
- ⌘ **Silicon area.** The sensor ASIC is a high-volume, cost-sensitive part with little spare silicon. A conventional machine-learning front end, such as an FFT or spectrogram stage, is exactly the area a designer cannot afford.
- ⌘ **Compute.** The microcontroller already in the sensor IC is sized for reading pressure and transmitting packets, not for continuous neural-network inference on a three-axis, 10 kHz stream. Running that workload on it would keep the core awake most of the time and drain the battery.

1.2 The Solution

Akida Pico IP is an ultra-low-power, event-based neural processing IP for always-on sensing at minimal die area. It processes 1D sequential data directly across multiple channels and runs continuous inference in the microwatt range, raising the host only when an actionable condition is detected. In this case, the input is a three-axis accelerometer stream from a wheel-mounted sensor, and the host wake condition can be a value of tire pressure or load weight.

At ~0.145 mm² in 22nm and μ W-class power, Pico drops into a sensor ASIC as a pre-validated inference block. It processes raw sensor data on-chip and requires no additional unit such as a DSP for FFT or spectrogram preprocessing. A single Pico block can infer multiple tire quantities at once: the reference model in this document classifies inflation pressure and vehicle load from one accelerometer, in one streaming pass, and further functions attach to the same block as additional outputs rather than additional models.

Reference model result	Value
Die area, 22nm reference configuration	~0.145 mm ² (~0.11 mm ² SRAM)
Preprocessing	Not required
Core power, continuous on a 10 kHz stream	44.6 μ W (Always on Duty cycled)
Inflation pressure, unseen accelerometer	0.933 macro-F1 (3 classes)
Vehicle load, unseen accelerometer	0.886 macro-F1 (3 classes)

Table 1. Reference model summary. Full figures in Sections 7-8.

Chapter 2

Key Features and Differentiators

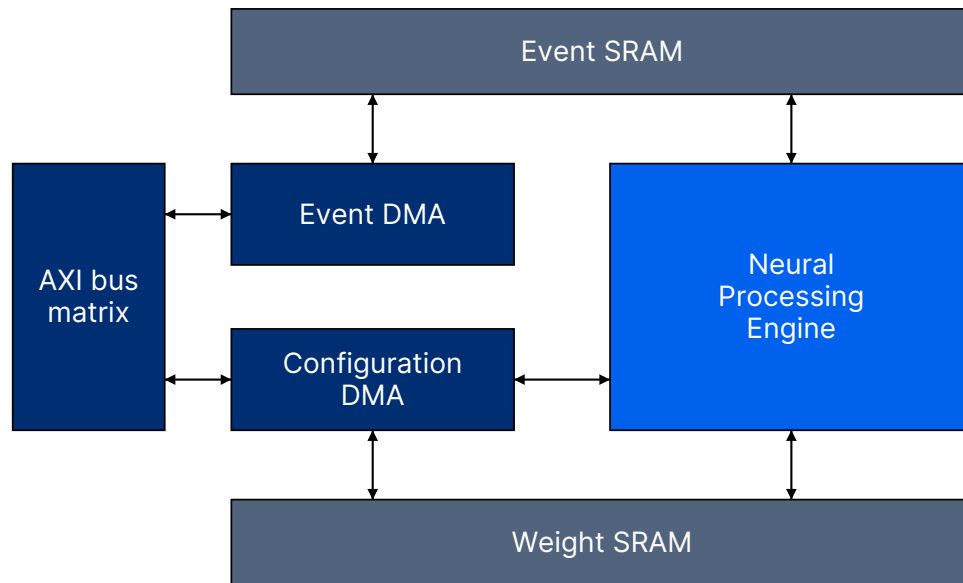
- * **µW consumption always-on inference:** Continuous inference in the microwatt range, within the power budget of a sealed in-wheel sensor. Reference figures are given in Section 8.
- * **No Preprocessing:** Raw accelerometer data is fed directly into the neural processing engine, eliminating an entire class of preprocessing IP and the power, area, and BOM cost that come with it.
- * **Multiple functions on one block:** Additional tire-monitoring functions are additional outputs on a shared network, decoded from the same forward pass. One Pico instance serves multiple classification and potential anomaly detection tasks rather than one instance per task.
- * **Tiny silicon footprint:** ~0.145 mm² total in 22nm, of which ~0.11 mm² is on-chip SRAM.
- * **Streaming-native compute model:** Built around the State Space Model (SSM) inference, which processes one sample at a time with a constant SRAM footprint regardless of sequence length.
- * **Autonomous Pico IP configuration:** Combined with an ultra-low-power MCU, Pico runs the sensing loop end to end, raising a single interrupt only when a meaningful event is detected and allowing the host SoC to remain in deep sleep.
- * **Standard host interfaces:** AXI-4 or AHB, for drop-in integration into a sensor SoC bus architecture.
- * **BrainChip MetaTF SDK:** Models are developed, trained, and quantized using standard TensorFlow workflows; cycle-accurate simulation enables validation before silicon.

Chapter 3

Architecture Overview

Akida Pico IP integrates a single Neural Processing Unit (NPU) with tightly coupled SRAM and dedicated DMA subsystems into a self-contained compute block, purpose-built for inference on 1D time-sequential time series streaming data.

During inference, all data movement stays on chip between the NPU and its Event SRAM and Weight SRAM. There is no external memory access, and once the controller starts an inference, the NPE executes the entire model in hardware with no per-layer or per-operation instruction fetch from a CPU. The bus interface is used only at initialization to load model weights and configuration; once running, the IP operates entirely from local memory. This locality of compute and memory is the primary architectural reason the IP achieves μW -class power.



Akida Pico Block Diagram

- * **Neural Processing Engine (NPE).** Executes TENNs inference directly in hardware. No general-purpose CPU is required for inference.
- * **Event SRAM.** Tightly coupled buffer for activations, intermediate event tensors, and per-step state.
- * **Weight SRAM.** Stores model weights loaded at initialization.
- * **Configuration DMA.** Handles weight and configuration loading across the bus interface.
- * **Event DMA.** Handles input data streaming into the NPE and event-based outputs out of the NPE.
- * **AXI / AHB Bus Interface.** Single host bus port, AXI-4 or AHB, selected at integration time.

3.1 Two configurations of the IP

Akida Pico is offered in two configurations built on the identical core. The only difference is which controller drives the inference loop: one the customer supplies, or one BrainChip supplies with the IP.

Pico IP is the compute core on its own: the Neural Processing Engine, its tightly coupled Event and Weight SRAM, and the DMA subsystems, presented as a bus-attached block. It performs inference and nothing else. Running it requires a controller or host elsewhere in the system to acquire sensor samples, stream frames into the core, read results back out, and carry out the pre- and post-processing around the inference - accumulating per-frame outputs over a decision window, applying thresholds, and deciding when to act. Pico IP assumes that controller already exists in the surrounding design.

Autonomous Pico IP is the same core packaged with an ultra-low-power microcontroller supplied by BrainChip, which acts as that controller. The MCU manages acquisition, feeds the core, and handles the pre- and post-processing, so the pair runs the full sensing loop end to end and raises a single interrupt only when a condition warrants attention. A larger host processor can then remain powered down almost the entire time, woken only on that interrupt.

3.2 Choosing a configuration in a tire sensor module

In a tire sensor module, the choice depends on whether the customer already has a controller or host they can use for the sensing loop.

Where a controller or host is already present in the module or the surrounding chip, the customer integrates the Pico core alone, as a co-processor on the system bus. This is the minimal-addition path: a neural compute block dropped into silicon the customer is already building, with no second controller introduced, and the existing controller is not asked to spend its cycles - or a dedicated DSP or FFT block - on the classification itself.

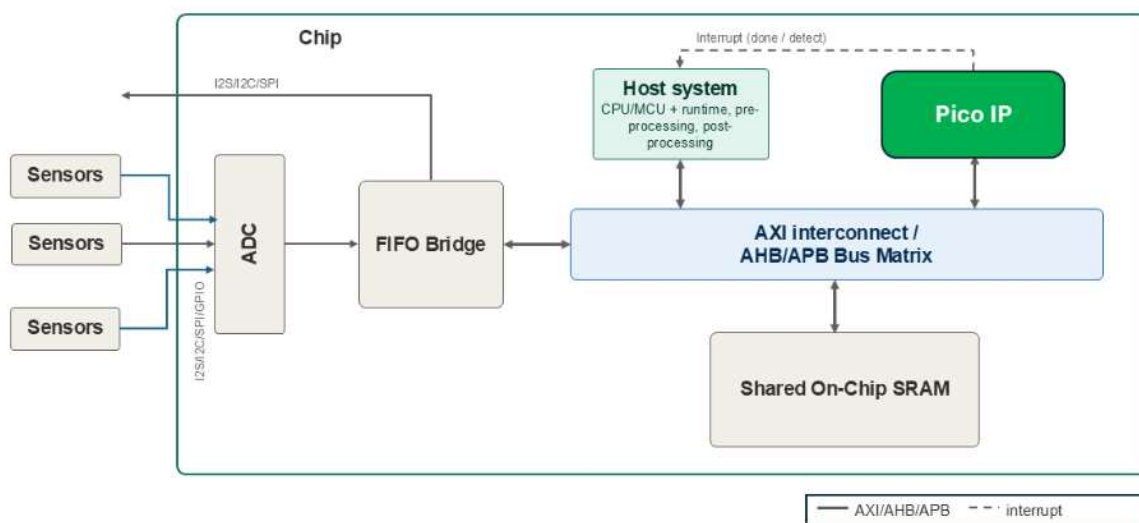


Figure 1. Pico configured as a host-connected co-processor. The host system provides the runtime, pre-processing, and post-processing; Pico IP performs the inference. In a tire sensor module the sensor stream is the wheel accelerometer.

Autonomous Pico IP

Where no controller is available for this work, or the customer prefers not to load an existing one, Autonomous Pico IP supplies the ultra-low-power MCU alongside the core as a self-contained sensing subsystem that runs at microwatt power while the host stays powered down, woken only on a detection event. Because the always-on block is tiny and the host is off almost the entire time, this arrangement keeps the complete product within its power budget. The radio transmits a classification result rather than a raw waveform, and only on an event, so the RF link stays within its budget as well.

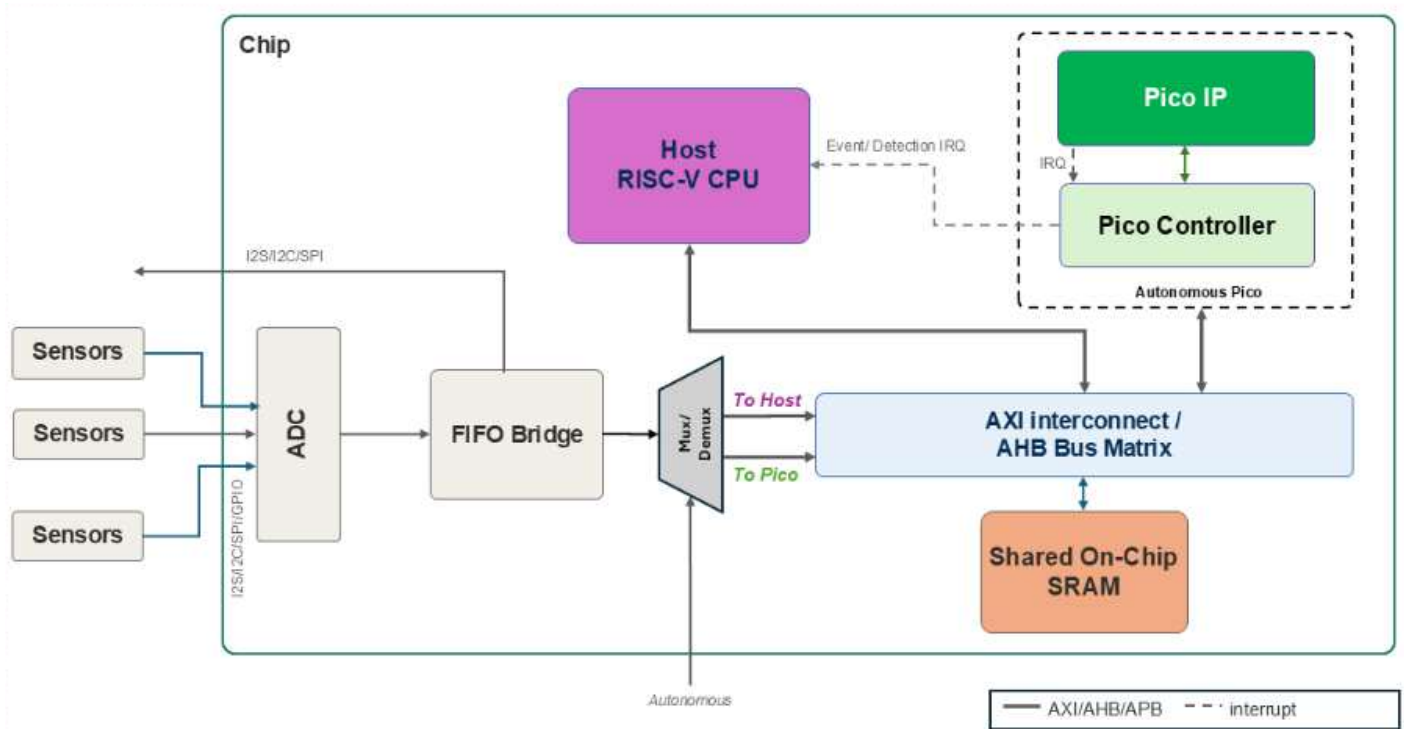


Figure 2. The Autonomous Pico system. The Pico Controller runs the sensing loop and raises an Event/Detection interrupt to the host, which otherwise stays powered down. In a tire sensor module the sensor stream is the wheel accelerometer.

Chapter 4

The Tire Signal

A rolling tire deforms where it meets the road, and a sensor on the tire passes through that deformation once per revolution. An accelerometer in that position records the structural response of a loaded tire: a quasi-periodic signal whose shape depends on the state of the tire. Multiple tire quantities can be decoded from it, which allows one sensor to serve several monitoring functions [3].

Inflation pressure and vehicle load both act on the geometry of the contact region: higher load lengthens the contact patch, and higher inflation pressure shortens it – the pair the reference model reads. Other quantities act on the same signal through other mechanisms. Tread depth changes the stiffness of the tread blocks that strike the road and the frequencies they produce. Road surface and friction change the character of the vibration the road drives into the tire [4], [5]. Wheel position appears in the orientation of the gravitational component across a revolution. Each is a candidate monitoring function on the same sensor.

4.1 Model Input

The reference model takes three axes of accelerometer data input at approximately 10 kHz and processes the samples directly. There is no preprocessing such as FFT, spectrogram, envelope analysis, or hand-designed feature extraction in the pipeline; the only preparation is a per-recording, per-channel normalization in the data path, with a fixed scale folded into the model. Section 5 explains why the model needs no frequency-domain front end.

The input is not restricted to acceleration. Pico processes 1D sequential data of any kind, so a module carrying additional sensors, such as a pressure or temperature time series, for instance, can feed them to the same block.

Chapter 5

TENNs Compute Model

Akida Pico runs TENNs, Temporal Event-based Neural Networks, a State Space Model (SSM) [6] Architecture and co-designed with the silicon that runs it. The reference model uses TENNs-R, the recurrent form [7]. An SSM carries a temporal context in a fixed-size recurrent state, which is the property that lets it run streaming inference on a 1D signal within a microwatt power and memory budget.

5.1 State Space Models

An SSM maintains a compact hidden state $h(t)$ that compresses all history into a fixed-size vector. At each step it updates the state and produces an output from two equations:

$$h(t+1) = A \cdot h(t) + B \cdot x(t)$$
$$y(t) = C \cdot h(t)$$

$x(t)$ is the current input sample, $h(t)$ the hidden state, $y(t)$ the output, and A , B , C are learned during training. Three properties follow, each of which matters at the wheel:

- ⊛ **Compressed history.** All history is folded into $h(t)$. There is no buffer and no sliding window - a fixed-size state vector carries forward everything the model has learned from past input.
- ⊛ **Fixed-size, constant memory.** The state vector is the same size after one second or one hour. SRAM usage is constant and known at design time.
- ⊛ **Continuous sample-by-sample processing.** The model processes the stream frame by frame, consuming one 64-sample frame per step. This is native streaming inference.

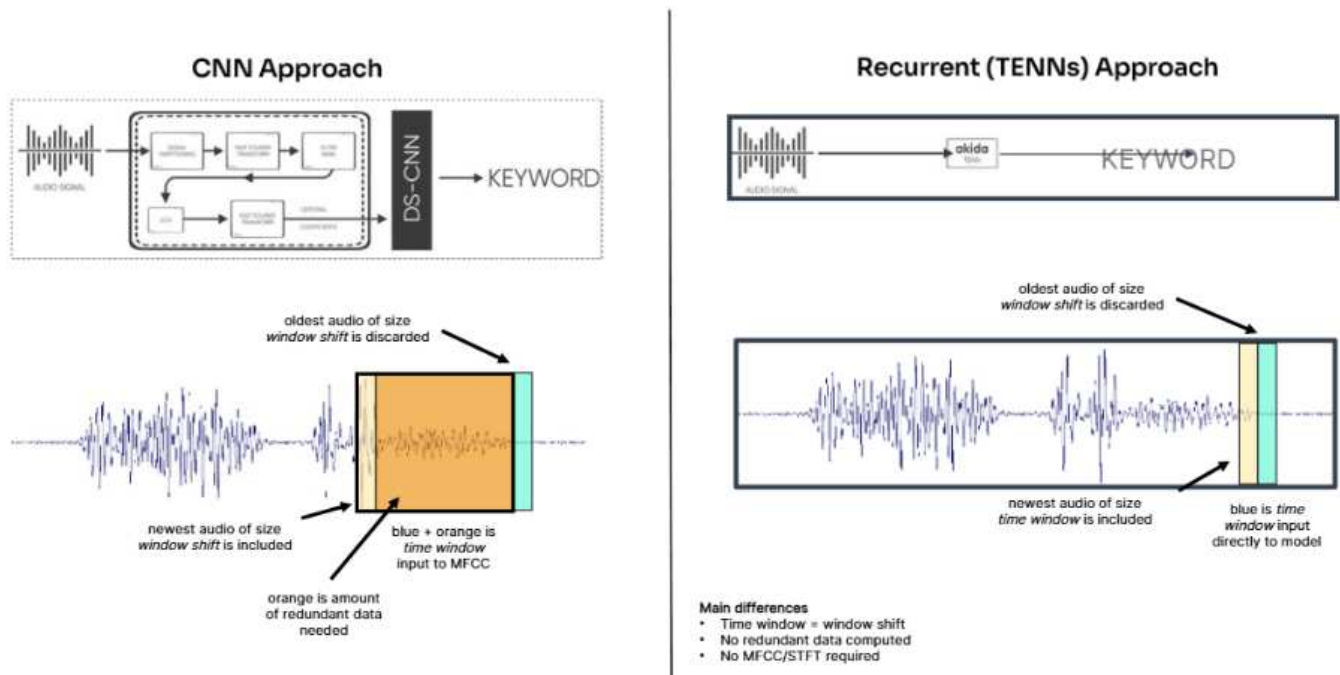
TENNs-R uses two forms of the same model. Training runs in a kernelized, convolutional form, where the computation is parallel and scales linearly with sequence length rather than quadratically as attention does. Inference runs in the recurrent form, which carries a fixed-size state from one step to the next: the state holds the history the model needs, so the signal is consumed as consecutive non-overlapping chunks rather than as overlapping windows, and no sample is reprocessed. At each step the recurrent state and the incoming chunk produce the next state and the output - here, the class logits for pressure and load.

5.2 Eliminating the Preprocessing using SSM

Typically, standard solutions for classifying a vibration signal work in the frequency domain and require a transform stage, an FFT or spectrogram, ahead of the classifier. The transform is needed because a conventional classifier holds no context across time: it sees each window in isolation, so the signal's temporal structure must be converted into frequency features before the classifier can use it. An SSM does not need the FFT stage because it holds that context in its recurrent state, carried from one frame to the next, and classifies the raw sample stream directly.

Removing the transform stage has three benefits:

- * **Reduced compute.** The multiply-accumulate operations an FFT or spectrogram would run on every window are never spent, which also lowers power.
- * **Lower latency.** Raw samples go straight into the network, with no transform or feature-extraction step to wait for.
- * **Smaller die, lower BOM cost.** No DSP or FFT block is needed in the sensor ASIC, so that silicon is never paid for.



Core insight: A Conv1D network gains temporal reach by adding compute, and an LSTM gains capacity by adding parameters. An SSM needs neither: one state update per step gives it long temporal reach in a fixed memory footprint, which is why it fits a μW in-wheel budget.

Chapter 6

Reference Model: Pressure and Load

The reference model classifies inflation pressure and vehicle load from one three-axis accelerometer, both from a single forward pass over a shared network. It is provided as a validated reference implementation and a measurement vehicle, not as a production-calibrated product. It was developed, trained, and quantized in the BrainChip MetaTF SDK using standard TensorFlow workflows, validated in cycle-accurate simulation, and measured on the Pico FPGA.

Output	Classes
Inflation pressure	low / nominal / high
Vehicle load	light / medium / heavy

Table 2. Reference model outputs.

The model emits six logits from one pass; the first three are decoded for pressure, the second three for load. Both derive from the same backbone, and only the width of the output head is task-specific.

Parameter	Value
Architecture	TENNs-R (recurrent SSM), streaming
Parameters (INT8 stateful, deployed)	16,696
Quantization	INT16 input; INT8 weights and activations; INT16 SSM state and outputs
Sensor	3-axis accelerometer, wheel-mounted
Sample rate	~10 kHz
Frame size	64 samples (6.4 ms)
Preprocessing	None - no DSP, FFT, or spectrogram
Decision rule	Per-frame logits accumulated over the recording, argmax per task
Platform	BrainChip Pico FPGA @ 50 MHz

Table 3. Reference model configuration.

Chapter 7

Validation and Results

A deployed sensor does inference on a tire/sensor that it has not encountered. Testing a model on the same tires and accelerometers it trained on says nothing about that case. All results here were obtained under one of two cross-validation protocols constructed to exclude such leakage.

- * **Accelerometer generalization. Per fold,** every recording from one physical accelerometer is withheld from training and used for test. Six folds for six accelerometers.
- * **Tire generalization. Per fold,** every recording from one physical tire is withheld from training and used for test. Ten folds for ten tires.

Quantization calibration windows are drawn only from training recordings. Metrics are reported at the recording level as mean and standard deviation across folds.

Task	Protocol	k	Float acc	Float F1	Int8 acc	Int8 F1
Pressure	Accelerometer	6	0.945 ±.038	0.932 ±.045	0.946 ±.040	0.933 ±.048
Pressure	Tire	10	0.960 ±.058	0.948 ±.075	0.950 ±.050	0.934 ±.066
Load	Accelerometer	6	0.886 ±.028	0.881 ±.037	0.894 ±.048	0.886 ±.053
Load	Tire	10	0.906 ±.058	0.892 ±.071	0.893 ±.055	0.882 ±.065

Table 4. Recording-level accuracy and macro-F1, mean ± 1 s.d. across folds.

- * **Quantization.** The deployed model uses INT16 input, INT8 weights and activations, and INT16 SSM state and outputs. Quantization costs little: the macro-F1 drop from the floating-point model is within a fraction of a standard deviation on every task and protocol. The figures reported are those of the model that would ship.
- * **Generalization.** On an accelerometer withheld from training, the model reaches 0.933 macro-F1 on inflation pressure (94.6% accuracy); on a tire withheld from training, 0.934 (95.0%). Vehicle load, from the same pass, reaches 0.886 and 0.882 macro-F1 (89.4% and 89.3% accuracy). Both are three-class graded outputs, not threshold alarms.

Chapter 8

Power, Throughput, and Clock

The figures are for the Akida Pico core running the deployed quantized reference model. Cycle counts were measured on the Pico FPGA; power and energy are BrainChip characterization values for the 22nm reference configuration.

Metric	Value
Cycles per frame	~17,803
Latency @ 50 MHz	0.356 ms
Throughput @ 50 MHz	2,809 inferences/s
Throughput @ 400 MHz (scaled)	22,469 inferences/s
Core power, always on duty cycled a 10 kHz stream	44.6 μ W
Energy per inference	0.285 μ J

Table 5. Akida Pico core, reference model. Cycle count measured on FPGA; power and energy from 22nm characterization.

A 10 kHz stream framed in 64-sample blocks presents a frame every 6.4 ms, and each frame takes 0.356 ms of compute at 50 MHz. The core is active for 5.6% of each frame period; 44.6 μ W is the average across that cycle, and it represents continuous operation, which is the upper bound of usage, since a deployed module wakes on rotation and samples periodically rather than without interruption.

Throughput at 50 MHz is 2,809 inferences per second against the 156 required to keep pace with the stream. Because the work per frame is a fixed cycle count, clock frequency trades active time against instantaneous power and average power stays approximately constant across the range. Clock can therefore be selected on integration requirements. A lower clock shares an existing slow domain and avoids a PLL; a higher clock shortens the active interval and leaves more of the cycle budget for additional functions.

Scope. These figures cover the Pico core. The MEMS accelerometer, the microcontroller, and the RF transmitter draw power in addition and are outside this characterization.

Chapter 9

Conclusion

Akida Pico enables the integration of always-on tire intelligence directly within a wheel-sensor ASIC. The reference implementation estimates both tire inflation pressure and vehicle load from a single accelerometer input while occupying approximately **0.145 mm²** of **22 nm** silicon and consuming only **44.6 μ W** of power and **0.285 μ J per inference**. The solution eliminates the need for FFT-based preprocessing or handcrafted feature-extraction stages, thereby reducing system complexity, latency, silicon area, energy consumption, and bill-of-materials cost.

The model achieves a macro-F1 score of **0.933** for pressure classification and **0.886** for load classification on data collected from an accelerometer not seen during training, demonstrating strong generalization performance with quantization (INT8 weights and activations).

The suitability of Akida Pico for this application stems from the alignment between the workload and the hardware architecture. Tire monitoring requires continuous analysis of low-bandwidth 1D sensor data within a compact, sealed, and energy-constrained environment. Pico is specifically optimized for such workloads, enabling inference to be performed directly within the wheel module using only microwatts of power and minimal silicon area.

Pressure and load estimation serve as a demonstration of a broader capability. The same Pico block and sensor infrastructure can support additional tire-monitoring functions, including tread-depth estimation, road-surface classification, friction estimation, and wheel-position identification where suitable training data exists. Because Pico operates on general 1D sensor streams, the architecture can also extend beyond tire monitoring to other always-on sensing applications within the vehicle. This ability to consolidate multiple sensing functions onto a single ultra-low-power inference engine enables richer functionality without materially increasing silicon area, memory footprint, or power consumption. The incremental cost of each added function depends on the task: some reuse the shared features at minimal cost, while others may need additional model capacity.

Akida Pico IP is available now for licensing.

References

- [1] National Highway Traffic Safety Administration, “Federal Motor Vehicle Safety Standard No. 138: Tire Pressure Monitoring Systems,” 49 CFR Part 571, final rule 2005 (100% phase-in from 1 September 2007); European Union, Regulation (EC) No 661/2009 and UNECE Regulation No. 64, mandatory for new M1 vehicles from 1 November 2014.
- [2] Infineon Technologies AG, “XENSIV™ SP40PLUS (SP400) Tire Pressure Monitoring Sensor - Data Sheet,” integrating a MEMS pressure sensor, Z-axis accelerometer for motion detection and angular measurement, temperature and supply-voltage sensors, 8051 microcontroller, and RF transmitter, with Angular Position Sensing firmware for wheel localization.
- [3] S. Khaleghian, O. Ghasemalizadeh, and S. Taheri, “Estimation of the Tire Contact Patch Length and Normal Load Using Intelligent Tires and Its Application in Small Ground Robot to Estimate the Tire-Road Friction,” *Tire Science and Technology*, vol. 44, no. 4, pp. 248-261, 2016.
- [4] S. Khaleghian and S. Taheri, “Terrain classification using intelligent tire,” *Journal of Terramechanics*, vol. 71, pp. 15-24, 2017.
- [5] A. J. Niskanen and A. J. Tuononen, “Three 3-axis accelerometers fixed inside the tyre for studying contact patch deformations in wet conditions,” *Vehicle System Dynamics*, vol. 52, sup. 1, pp. 287-298, 2014.
- [6] A. Gu, K. Goel, and C. Ré, “Efficiently Modeling Long Sequences with Structured State Spaces,” arXiv:2111.00396, 2022.
- [7] Y. R. Pei, R. Shrivastava, and F. Sidharth, “Real-time Speech Enhancement on Raw Signals with Deep State-space Modeling (aTENNuate),” arXiv:2409.03377, 2024.



brainchip ™

Version: v1
August 2026