White Paper

Predictive Maintenance Solution for Motor Control

November 2020

Increasing Operational Effectiveness and Reducing Costly Downtime

Factories today utilize an increasing amount of industrial automation and process control equipment to increase manufacturing throughput and improve operational efficiencies. Automation systems and production equipment rely on electric motors' widespread use to drive everything from conveyor belts to robotic arms. Even though motor technology has significantly advanced, they are prone to internal component wear, like all mechanical equipment. Also, repetitive movements of heavy items and excess mechanical stresses can distort and misalign the motor mounting bed, resulting in bearing wear, vibration, and sudden failure.

Within any industrial equipment deployment, the concepts of maintain, repair, and operate (MRO) are well established. Within any MRO process, maintenance forms an essential part in keeping a manufacturing process operating effectively. Several different maintenance regimes are reactive (corrective), preventative, and predictive – the most popular. Of these, predictive maintenance yields the lowest maintenance costs and the optimal disruption resulting from downtime.

The Benefits of a Predictive Maintenance Regime

In a predictive maintenance regime, equipment is monitored continuously to detect the early signs of a potential failure. A predictive regime differs from a preventative routine in which equipment is maintained based on a recommended inspection and service frequency. With this approach, maintenance can be conveniently scheduled before an unexpected, disruptive, and expensive incident occurs.

Monitoring the operation of a motor, for example, can be achieved in many different ways. Thanks to the industrial internet of things (IIoT) and initiatives such as Industry 4.0, the ability to connect, monitor, and determine any production asset's operational status is now extremely straightforward. The increased use of artificial intelligence (AI) and machine learning (ML) techniques can also yield more confidence in the monitoring process and even earlier detection of potential failures.

The Challenges of Inference at the Edge

There are two principle steps in the deployment of any machine learning neural network: training and inference. The training of a neural network (NN) model involves collecting massive amounts of data so that the model's algorithm can learn to infer a result. Training is typically performed in a lab environment using high-performance computer systems. Based on the compute resources used, training is typically a power-hungry process.

However, inference – the process of running the model's algorithm – has also been a task requiring low latency cloud connectivity and significant compute resources, which also required a considerable power budget. Until recently, end-point based inference was only possible by using microprocessor-based computers that consumed significant power and required considerable space. Performing reliable inference at the edge has several benefits. For example, there is no reliance on high latency cloud compute resources with local processing, allowing real-time, deterministic operation. From a security perspective, all data stays locally, reducing the potential of security breaches or device take-over by an adversary.

Endpoint-based Predictive Maintenance Becomes a Reality

With strong interest from industry to develop edge-based inference applications, the demand for energy-efficient devices optimized for the task is high.

An example is the Renesas RA6T1 microcontroller. Optimized for motor control applications, the MCU can also run a neural network that continuously monitors the motor's performance for potential faults or impending failure.

The RA6T1 is a 120 MHz 32-bit Arm Cortex M4 device that hosts high-resolution 32-bit PWM timers and advanced analog functions ideal for developing safe and high precision motor control applications. The PWM timers feature seven different operation modes and can be programmed from zero to 100 % duty cycle. The timers can be synchronized to provide a three-phase complementary output. The analog features include an 11-channel and an 8-channel 12-bit analog to digital converters (ADCs), three channels of which include sample and hold circuits.

Motor Control Hardware Design Resources

Two development boards are available to aid prototyping motor control applications, including an embedded artificial intelligence (e-AI) capability. The Renesas RA6T1 CPU card (part number RTK0EMA170C00000BJ) – see Figure 1, and a motor driver inverter board (part number RTK0EMX270S00020BJ) for use with sensorless brushless DC (BLDC) motors – see Figure 2.

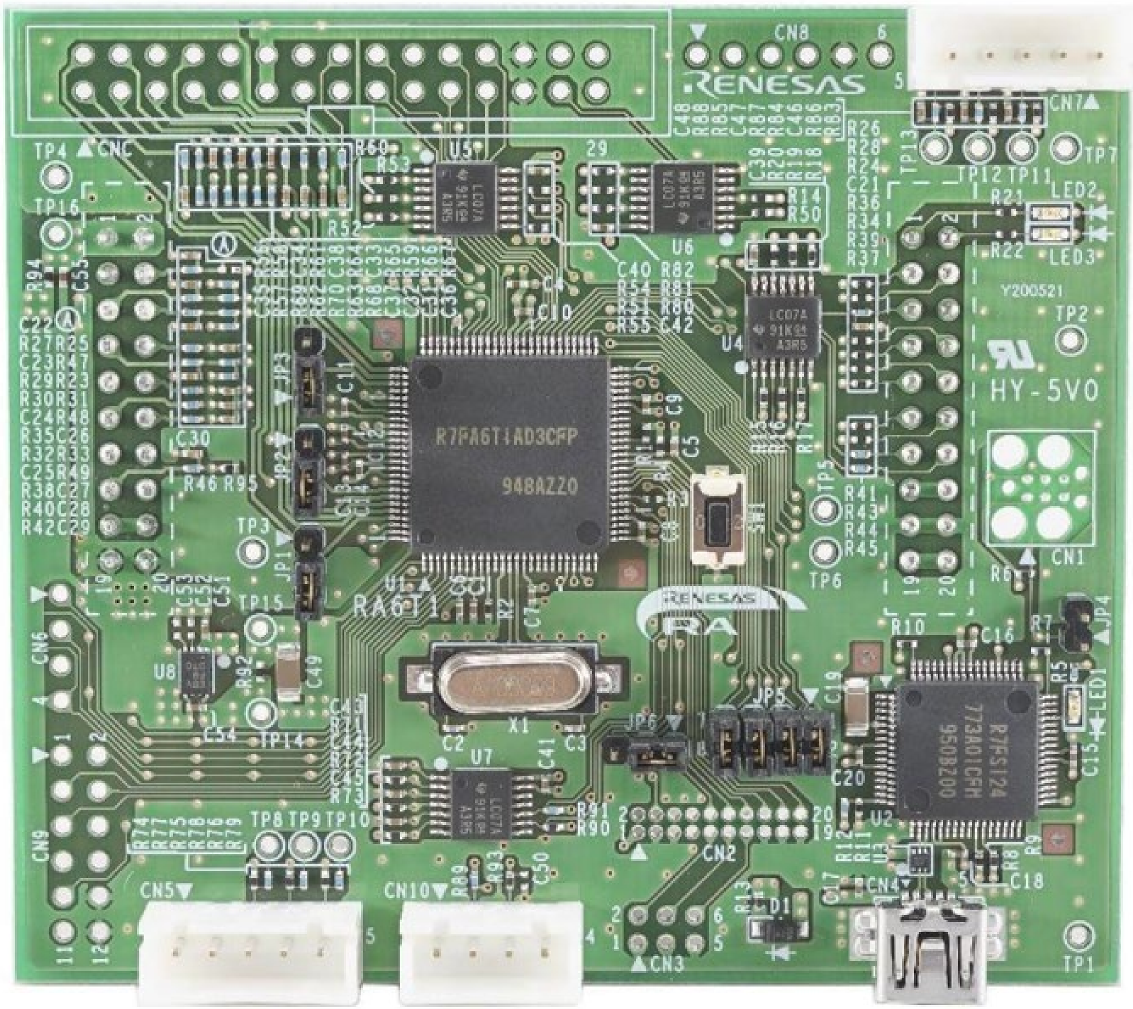

Figure 1: The Renesas RA6T1 CPU board

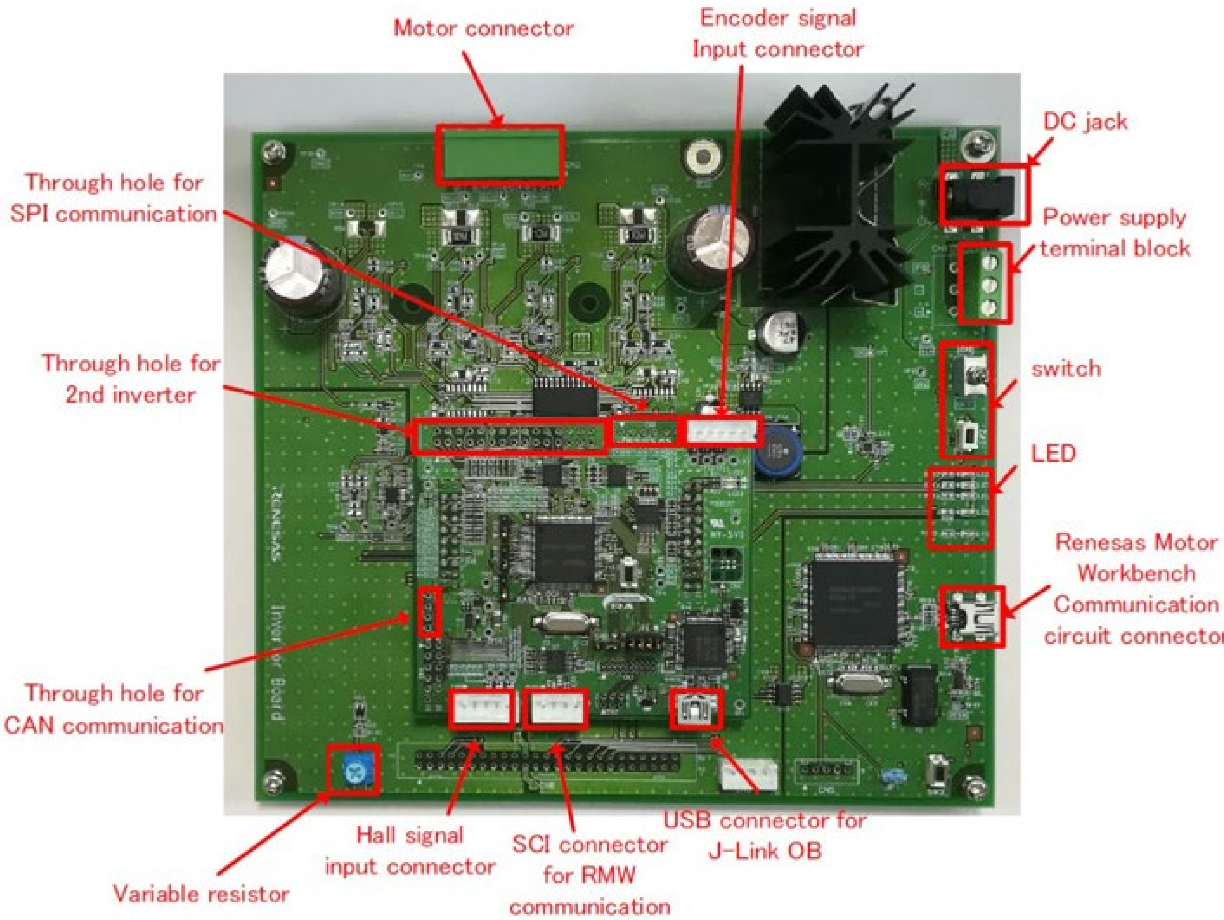

Figure 2: The Renesas RTK0EMX270S00020BJ part number inverter board and RA6T1 CPU card

Software development tools include the Renesas e² studio IDE and the Renesas Motor Workbench 2.0.

Motor Control Hardware Design Resources

This section uses the two boards previously mentioned to implement a motor failure detection algorithm using e-AI. The functional block diagram is indicated in Figure 3. The shunt resistors and the motor drive circuitry are on the inverter board. A brushless DC motor is driven using a sensorless vector control algorithm.

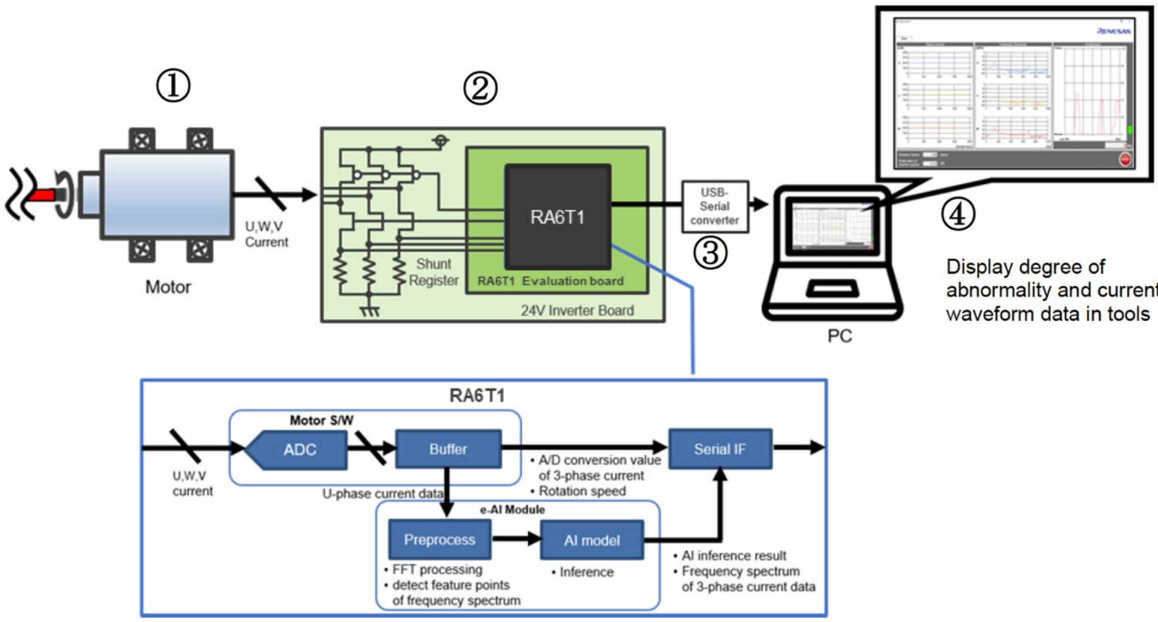

Figure 3: System flow of the RA6T1 e-AI-based motor fault detection prototype

Shunt resistors are employed to monitor the motor's status by measuring each motor phase's feedback voltage. ADCs convert the feedback voltage across the shunt resistors into the digital domain for use within the vector control algorithm to determine the motor's rotor position. The motor drive current is also detected by the shunt resistor and forms the input to the e-AI neural network. Passing the ADC output to the neural network preprocessing from the time domain to the frequency domain is achieved using FFT techniques. Arm's Cortex microcontroller software interface standard (CMSIS) hardware abstraction layer provides a DSP component library of over 60 different functions, including FFTs. Analyzing the frequency data is a more reliable way for the neural network to detect changes in the motor's operating condition. A Windows PC is connected by USB to the MCU board to provide visual access to the data during development and testing.

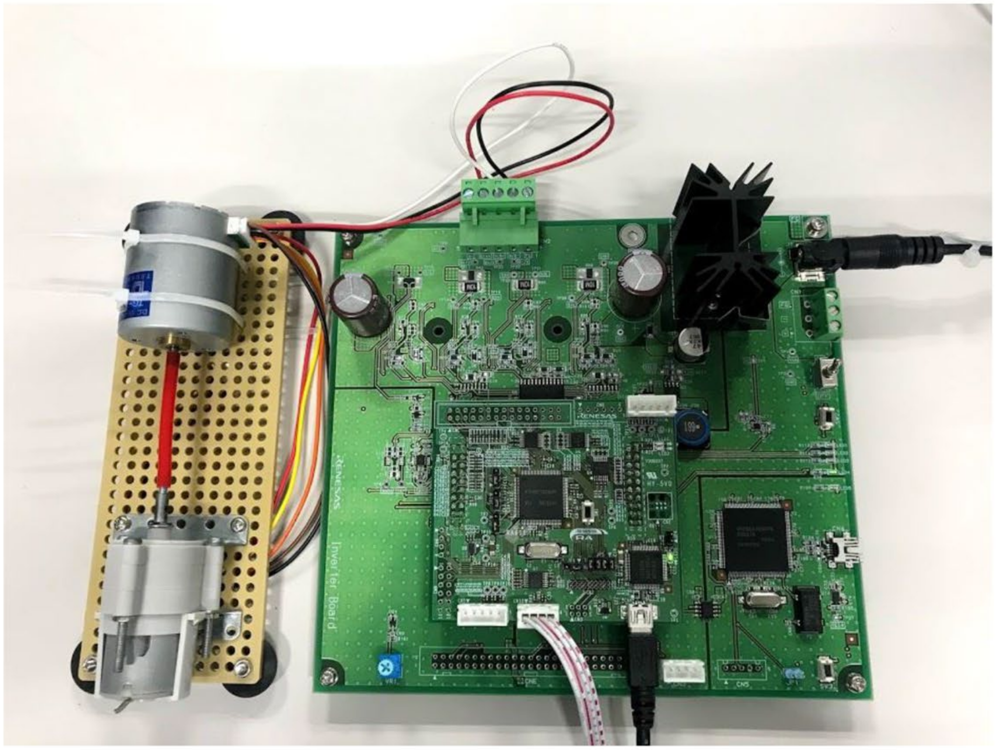

Figure 4: The Renesas RA6T1 development board mounted on the inverter board with a BLDC motor driving a planetary gearbox

A prototype setup is illustrated in Figure 4. A BLDC motor is connected via a flexible driveshaft to a planetary gearbox. The motor and gearbox are mounted on a composite mounting plate that can be deflected slightly with finger pressure to simulate driveshaft misalignment and other anomaly conditions.

Preprocessing Data Flow

The same preprocessing technique is used for both the neural network training process and inference. Fault detection can only be achieved by training the neural network model to learn the difference between normal and anomaly conditions. For example, in the prototyping scenario highlighted, an anomaly condition would indicate a driveshaft misalignment.

Figure 5 illustrates the output waveform from one of the three shunt resistors. The analog to digital conversion is framed every 512 points, and the last 64 points are overlapped to avoid missing data.

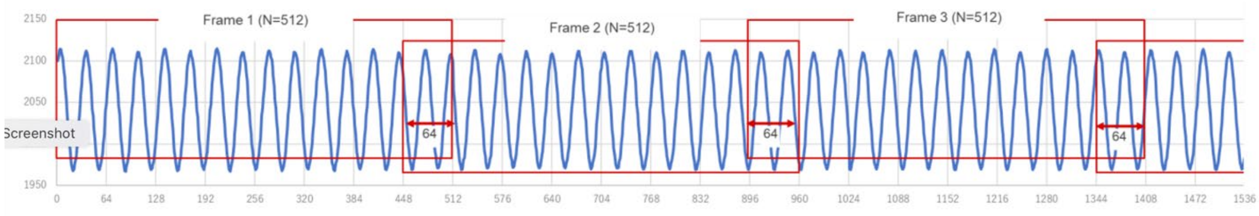

Figure 5: Shunt output voltage highlighting the overlapping of the framing window

After FFT conversion to the frequency domain, the output is indicated in the frequency spectrum chart of Figure 6. Anomaly conditions are indicated by peak values, which are then isolated into a 32-point window. Note that only the data from one motor phase is used to train the AI model.

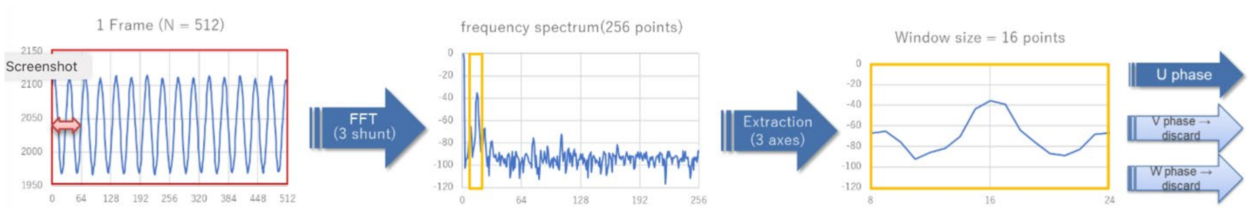

Figure 6: Shunt data preprocessing flow from analog data to isolating a frequency peak artifact

AI Model Development Flow

In addition to the Renesas e² studio IDE, GCC toolchain, and a Windows 10 PC operating environment, the Google TensorFlow Lite for Micro framework is used to transfer the trained model to the embedded environment. Renesas also provides a complete demo package of tools, including those for data collection and training. An initial AI model is supplied with the demonstration project, which is ready for training. Renesas has created the initial AI model supplied using TensorFlow and trained it using different datasets. The AI model was then converted into a TensorFlow Lite for Micro format.

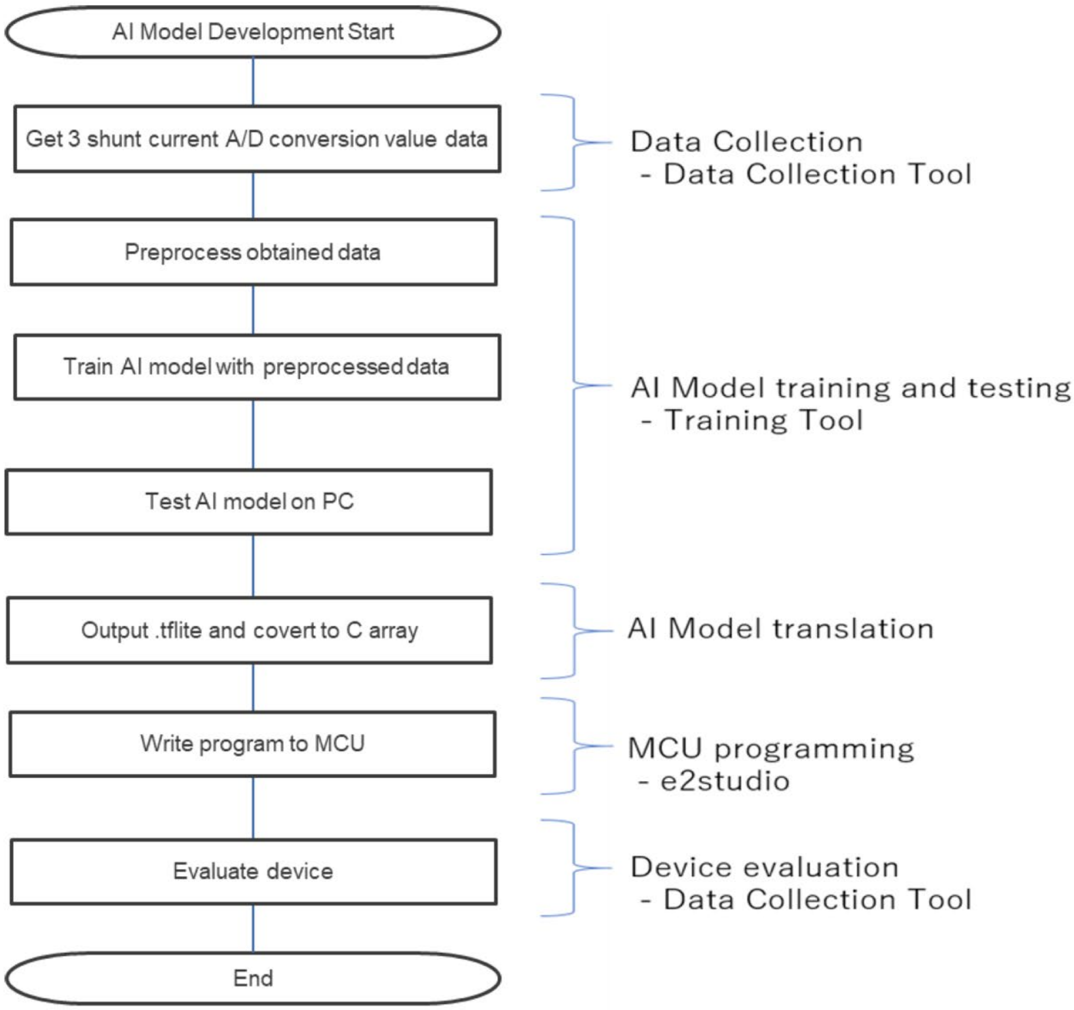

Figure 7: AI model development flowchart

Figure 7 highlights the essential steps from data collection through to inference testing. Data collection involves using the supplied Windows data collection tool to start the motor and collect data. Data are collected with a visual display of the shunt current and frequency spectrum, as illustrated in Figure 8.

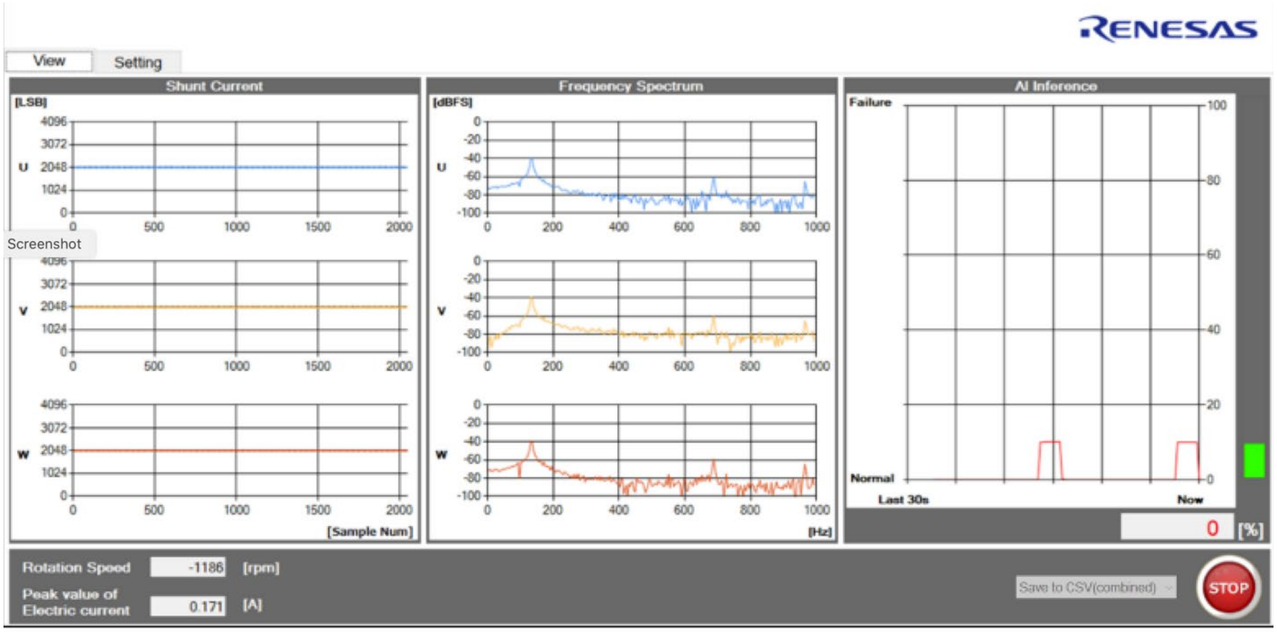

Figure 8: Screenshot of the data collection tool in use

Data Collection

The data collection process should be conducted with the motor operating at different shaft rotation speeds and normal (no driveshaft deflection) and anomaly (gentle pressure on the mounting plate) conditions. The data collection screenshot also shows the inference result conducted by the RA6T1. The data files collected from the collection tool are used for the next stage of the process: model training. Several data collection parameters can be configured, such as the sampling frequency, frame size, and the overlap window. The degree of anomaly detected depends on several factors, such as different shaft speeds and the load conditions imposed on the motor. Separate data files are used for model training and testing.

Model Training

After data collection, the data are processed by the Renesas e-AI training tool on the PC.

The project's demonstration data includes normal and anomaly data sets with motor speeds of 850, 900, 950, and 1000 rpm. During testing, a model accuracy of 96 % was achieved when running the motor between 850 and 950 rpm.

The e-AI training tool outputs a *.tflite model that is supported by the Google TensorFlow Lite for Micro framework. TensorFlow Lite for Micro has been explicitly designed to support running neural network models on resource-constrained 32-bit Arm Cortex microcontrollers. The e-AI training tool includes a converter that takes the machine learning algorithm model and converts it into a C array. With this approach, the converted model can be incorporated into the MCUs embedded firmware using the conventional MCU IDE toolchain.

Testing the Predictive Maintenance Model

The increasing level of automation in homes, buildings, and industry, is leading to much greater use of electric motors. Designers who are already using electric motors face the requirement of optimizing their implementations.

For engineers who are adding them to their designs for the first time, understanding how to do so quickly and effectively becomes a challenge.

Renesas offers a combination of an optimized microcontroller, ready-to-use motor control software, and a tool chain that draws on open source and proprietary offerings and takes advantage of Arm's open development ecosystem. The Renesas Motor Workbench 2.0 adds to this offering by enabling users to take advantage of Renesas' experience of advanced control strategies. The overall combination of features will help users increase their debugging efficiency and shorten their development cycles as they implement products using electric motors based on the latest Renesas RA6T1 group.

Conclusion

Predictive maintenance is the most efficient and effective way of managing a sizeable asset-based production facility or manufacturing site. Incorporating edge-based inference on microcontrollers and the normal core function, motor control, for example, saves considerable deployment costs. Also, conducting inference at the edge removes the need for constant and costly network bandwidth and latency challenges.

The Renesas RA6T1 microcontroller family of devices are optimized for motor control applications. Peripheral functions for motor control applications include high resolution PWM timers, high speed analog comparators, and a 12-bit A/D converter. Available in a variety of LQFP package formats, the RA6T1 comprises a 120 MHz Arm Cortex-M4 core with up to 512 kB flash memory and 64 kB RAM. Typical applications for the RA6T1 include industrial automation drives for AC Drive, compressors, fans and HVAC equipment. Development support resources include the Renesas e² studio IDE, the Renesas Flexible Software Package (FSP) which contains motor control algorithms, and the Renesas Motor Workbench. Hardware design resources include an RA6T1 evaluation CPU card and a motor control development platform.

Learn More

1. RA6T1 Product Page
2. RA6T1 Motor Control Evaluation System
3. RA Partner Ecosystem
4. Flexible Software Package (FSP)
5. RA Family of MCUs