

An FPGA-Based Two-Stage Autonomous Parameter Acquisition and Holdover Compensation Method for OCXOs

Jiale Fan

National Key Laboratory of Science and Technology on Electronic Test and Measurement, North University of China, Taiyuan, China

* Corresponding author: Jiale Fan

Abstract: During GNSS outages, distributed measurement nodes must maintain an accurate local time reference using an oven-controlled crystal oscillator (OCXO). Conventional holdover methods that use pre-characterized drift parameters require additional preparation for individual oscillators. This paper presents an FPGA-based two-stage autonomous parameter acquisition and holdover compensation method. When GNSS pulse-per-second (PPS) timing is available, the FPGA first estimates and compensates for the baseline count deviation. Residual deviations are subsequently measured in 256-s windows and stored in SPI NOR Flash with time indices and CRC16 checksums. After GNSS interruption, the stored parameters are retrieved according to elapsed holdover time and used to adjust the locally generated PPS (DPPS) period using integer quotient-remainder compensation. Five OCXOs of the same type were evaluated with 24-h parameter acquisition followed by independent 24-h holdover tests. The measured timing errors remained below 25 μ s over 24 h under the tested conditions. The method provides device-specific parameter acquisition and reuse without host-computer-based offline calibration.

Keywords: FPGA; OCXO; GNSS holdover; autonomous parameter acquisition; timing compensation

1. Introduction

In distributed data acquisition and communication systems, maintaining a unified time reference among different nodes is essential for accurate measurements. Global Navigation Satellite System (GNSS) pulse-per-second (PPS) signals are widely used for time synchronization in distributed measurement systems [1,2]. GNSS timing is not always available in a measurement system. The PPS input may be lost because of signal obstruction, electromagnetic interference, or a receiver fault. In that case, the node has to keep its time reference running with a local clock. An OCXO is often selected for this purpose because its frequency remains relatively stable over short periods. Temperature, aging, and other operating conditions can still cause frequency changes [3].

Previous work has addressed OCXO timing errors in several ways. One approach is to improve the oscillator

itself. For example, a compact OCXO with low phase noise was designed to improve local clock performance [4]. Other studies have focused on estimating frequency changes. These include drift prediction with a GRU network [5] and parameter estimation using improved Kalman filtering [6]. In addition, digital frequency adjustment methods combining direct digital synthesis (DDS) and proportional-integral-derivative (PID) control have been proposed [7]. Related studies have addressed temperature-dependent timing errors [8], oscillator holdover based on historical timing differences [9], and the timing performance of GNSS-based systems [10]. The studies address different parts of the timing problem. Some focus on oscillator behavior, while others estimate or correct the errors that accumulate during operation.

FPGA holdover systems can adjust their output period directly by changing the number of clock cycles. Such corrections can be calculated using integer arithmetic, which is suitable for hardware implementation. Jia et al. [11] used quotient-remainder decomposition to distribute the correction counts over successive seconds. Their method also included compensation for long-term OCXO drift. An FPGA-based autonomous GPS-disciplined oscillator for wireless sensor network nodes has also been reported [12]. Other studies have addressed oscillator synchronization using GNSS PPS signals [13], autonomous clock systems [14], phase-shift compensation [15], and advanced control methods [16,17]. A change in the oscillator or its operating conditions may therefore require another calibration.

To address this issue, this paper proposes an FPGA-based autonomous parameter acquisition and holdover compensation method. When GNSS timing is available, the FPGA first measures the baseline timing deviation between the reference PPS and the locally generated PPS (DPPS) and applies dynamic compensation. After the baseline compensation takes effect, the residual drift is measured over successive time windows and stored in SPI Flash. When GNSS signals are lost, the FPGA retrieves the acquired parameters according to the elapsed holdover time and adjusts the DPPS period through digital compensation. The proposed method integrates parameter acquisition, nonvolatile storage, and holdover compensation within the FPGA, reducing the dependence

on host-computer-based offline calibration and manually preset parameters.

2. System Design

2.1 Hardware Architecture

The proposed holdover system serves as the local timing module of a distributed data acquisition node. When GNSS timing is available, the system performs DPPS calibration and acquires oscillator compensation parameters. After GNSS signal loss, the stored parameters are used to maintain the locally generated pulse-per-second (DPPS) signal.

A 10 MHz clock from the OCXO is supplied to the FPGA. The internal PLL raises the frequency to 100 MHz, which is used for timing measurements and DPPS generation. The compensation parameters are stored in Flash. Through UART, the host computer receives system-status information and measurement data. Figure 1 shows the connections among these components.

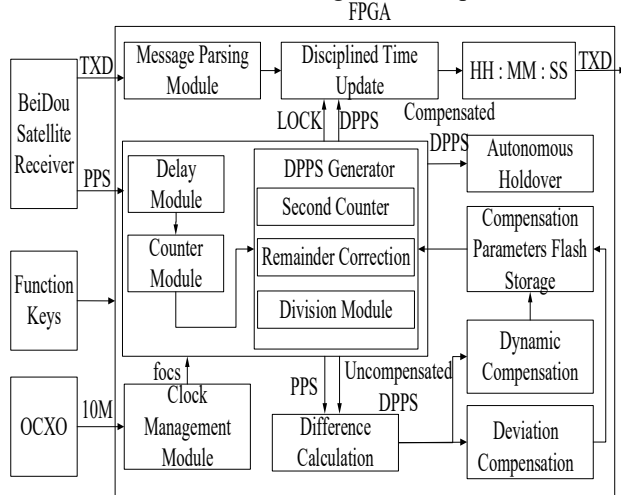


Figure 1. Block diagram of the FPGA-based holdover system.

Two correction terms are used by the controller. They are baseline deviation and residual drift. The required DPPS period adjustment is calculated by the FPGA from these values. Acquired values are written to Flash by parameter-management logic. These values are read when required. Measurements, parameter transfers and DPPS correction are coordinated by a control state machine. The coordination works for both acquisition and holdover modes.

2.2 Operating Procedure

At startup, the FPGA initializes the Flash interface and checks the stored parameter records. If a record passes the integrity check, the baseline parameter can be loaded from Flash for later use. The system starts a new acquisition run. Before using GNSS as the timing reference, the FPGA checks the PPS input and waits for three consecutive valid pulses.

During a new acquisition run, the FPGA compares the GNSS PPS with the uncorrected DPPS to determine the baseline count deviation. The FPGA converts this deviation into a correction and applies it to the DPPS generator. It then measures the remaining difference

between the two signals in successive 256-s windows. Each residual value is stored in Flash with its time index and CRC16 checksum. Only complete 256-s windows are retained; incomplete windows are discarded. The nominal acquisition period is 24 h.

When valid parameters are already available in Flash, the FPGA can use them once holdover is enabled. It loads the baseline correction first, then selects the residual correction according to the elapsed holdover time. The two values determine the DPPS period adjustment. During GNSS loss, the FPGA continues to apply the stored corrections and does not acquire new parameters from the external PPS.

The compensation records remain in Flash after power-off. On restart, the FPGA can read them and restore the required parameters. The UART interface allows the host computer to monitor the operating mode, timing deviations, and Flash status.

3. Autonomous Parameter Acquisition and Holdover Compensation Method

3.1 Two-Stage Autonomous Parameter Acquisition

The method uses two sets of correction data obtained at different stages. FPGA measures the baseline count deviation after the OCXO stabilizes and applies the initial correction. It then measures how the remaining deviation changes during operation. Both parameter sets are retained for later holdover.

The timing circuit uses a 10 MHz OCXO. The FPGA PLL raises this frequency to 100 MHz, giving a counter interval of 10 ns. A nominal one-second period therefore corresponds to 100,000,000 clock cycles. During parameter acquisition, the FPGA compares GNSS PPS and DPPS over a 256-s window.

Both signals are measured with the same 35-bit free-running counter. The FPGA captures their rising-edge timestamps separately. The count accumulated in the i -th window is then obtained by subtracting the corresponding timestamps:

$$\begin{aligned} C_{P,i} &= [T_P(b_i) - T_P(b_{i-1})]_{2^W} \\ C_{D,i} &= [T_D(b_i) - T_D(b_{i-1})]_{2^W} \end{aligned} \quad (1)$$

In Eq. (1), T_P and T_D represent the counter timestamps captured at the GNSS PPS and DPPS edges, respectively. The window boundary is $b_i = iN$, and the counter width W is 35 bits. Since the counter runs continuously, the subtraction is performed modulo 2^W .

A 256-s measurement at 100 MHz corresponds to 25.6×10^9 clock cycles. This is smaller than the maximum value of a 35-bit unsigned counter. The counter wraps approximately every 343.6 s, so a complete measurement window is shorter than one counter cycle. If a rollover occurs between two captured edges, modular subtraction still gives the correct interval count.

The FPGA calculates the PPS and DPPS intervals separately before taking their signed difference. It does not treat the captured timestamps as unlimited, continuously increasing values. Both signal edges must be available for a valid measurement. If a reference PPS

edge is missing, the corresponding window cannot provide a reliable residual deviation.

The signed difference between the two accumulated intervals is calculated as follows:

$$D_i = C_{P,i} - C_{D,i} \quad (2)$$

A positive D_i indicates that the DPPS period is shorter than the reference period, so the DPPS count should be increased; a negative D_i indicates a longer DPPS period. For experimental evaluation, the phase error is defined as

$$\begin{aligned} E(k) &= T_D(k) - T_P(k) \\ e(k) &= E(k)T_c \end{aligned} \quad (3)$$

Where $T_c = 10$ ns is the FPGA counter period. The phase error $e(k)$ is used for performance evaluation and is not used to derive the stored compensation parameters.

Stage 1: Baseline dynamic deviation acquisition.

After OCXO stabilization, the FPGA measures three consecutive valid 256-s windows before applying baseline compensation. The mean signed window deviation is

$$D_{\text{dyn}} = \frac{1}{M} \sum_{j=1}^M D_j, \quad M=3 \quad (4)$$

Where $M = 3$ and D_j is the signed deviation of the j -th measurement window. The baseline parameter is converted into a distributed per-second compensation sequence and applied to the DPPS period.

Using multiple complete windows to obtain a baseline estimate reduces the influence of an individual window on the initial parameter. The estimated baseline is applied to the DPPS generator before residual parameters are collected. The digital operating point is determined in the first stage. The drift of this operating point over time is characterized in the second stage. The mean in Eq (4) is a signed window count value. It is not a direct phase offset in microseconds. This signed count value is converted by the FPGA into a set of integer correction values. The signs of these correction values must match the signs used by the period adjustment logic.

Stage 2: Residual drift acquisition.

Baseline correction is enabled. The GNSS PPS is continuously compared with the corrected DPPS by the FPGA. One residual measurement is finished every 256 seconds. The count difference of the i -th window is calculated:

$$D_{\text{slow},i} = C_{P,i} - \tilde{C}_{D,i} \quad (5)$$

The tilde marks the DPPS count value measured after baseline correction is activated. A signed residual bias for the i -th window is obtained. This value and its operating time index are stored by the FPGA.

Only complete 256-second window data are used. Data collection is stopped before a window ends. The incomplete measurement is discarded. The kept records form the residual parameter sequence for time-keeping mode.

The two parameters are used for different working conditions. Baseline bias is measured before the first correction is applied. Residual drift is measured with DPPS after correction takes effect. Original baseline bias is removed from residual values. Both corrections can be

turned on during time keeping. The initial bias will not be counted twice.

Residual data are stored by the FPGA in original time order. One record is assigned to one specific 256-second interval. A single fixed frequency offset cannot replace this full sequence.

3.2 Digital Compensation of DPPS

Bias is measured within a 256-second window. The period of the DPPS generator is adjusted once per second. The total correction is split into integer clock counts by the FPGA with quotient-remainder decomposition, as shown:

$$D_{\text{slow},i} = Q_i N + R_i, \quad |R_i| < N \quad (6)$$

N is 256. The quotient is the integer correction applied each second. The remainder is the leftover clock count. Division is truncated toward zero. The sign of the measured bias is kept in the remainder. The absolute value of the remainder is smaller than N .

The remainder is not added to a single DPPS period. Instead, the FPGA distributes the extra clock counts across the next 256 s. The correction for the k -th second is written as:

$$u_i(k) = Q_i + \text{sgn}(R_i) s_i(k) \quad (7)$$

$$\sum_{k=1}^N s_i(k) = |R_i| \quad (8)$$

The binary distribution sequence $s_i(k)$ is generated using a Bresenham-style error accumulator. The accumulator adds $|R_i|$ once per second. When its value reaches N , the FPGA produces one extra correction count and subtracts N from the accumulator. Otherwise, no additional count is produced in that second.

After all N seconds, the total number of extra correction counts is exactly $|R_i|$. Their sign is handled separately, according to the original deviation. The baseline correction is distributed in the same way.

After the GNSS reference is lost, the FPGA selects the stored residual correction according to the time elapsed since holdover began:

$$i(t) = \text{floor}(t/N) + 1 \quad (9)$$

Here, t is the number of seconds elapsed in holdover. The first stored window has index 1. A residual parameter can be selected only when its index falls within the available records.

The final DPPS period count is

The index in Eq. (9) changes only when the elapsed holdover time enters the next 256-s interval. For a nominal 24-h acquisition, 337 complete windows are available, accounting for 86,272 s; the remaining 128 s do not form another full training record. Accordingly, the recorded residual sequence is limited to indices corresponding to completed acquisition windows. The treatment of any later interval without a matching stored residual record must be consistent with the implemented control logic and is not established by the acquisition rule alone.

$$C(k) = C_0 + u_{\text{dyn}}(k) + u_i(k) \quad (10)$$

In Eq. (10), the nominal DPPS period C_0 is 100,000,000 clock cycles. Two correction terms are

added to this count: $u_{\text{dyn}}(k)$, obtained from the stored baseline parameter, and $u_i(k)$, selected from the residual parameter sequence.

The FPGA uses their sum to set the DPPS period at the next second boundary. Once holdover starts, the stored drift parameters are no longer updated from GNSS PPS measurements.

3.3 Storage and Retrieval of Drift Parameters

SPI NOR Flash holds the baseline correction and the residual values in separate memory areas. Each residual record includes four items: a parameter identifier, a time index, a signed correction value, and a CRC16 checksum. The index is used to locate the correction for a given holdover interval, while CRC16 is used to check the stored record.

In the FPGA implementation, timestamps are captured with a 35-bit free-running counter. The correction values are handled using signed 16-bit registers.

The FPGA checks the CRC16 value of each stored record during initialization. Parameters are loaded only when the corresponding records pass this check. After the OCXO has stabilized, the FPGA restores the valid baseline correction from Flash.

During holdover, the FPGA reads the residual value associated with the current 256-s interval. This allows it to apply the previously acquired correction without collecting new timing measurements from the external PPS.

If the parameter storage is erased, the system returns to acquisition mode on the next reset. Figure 2 summarizes the overall procedure.

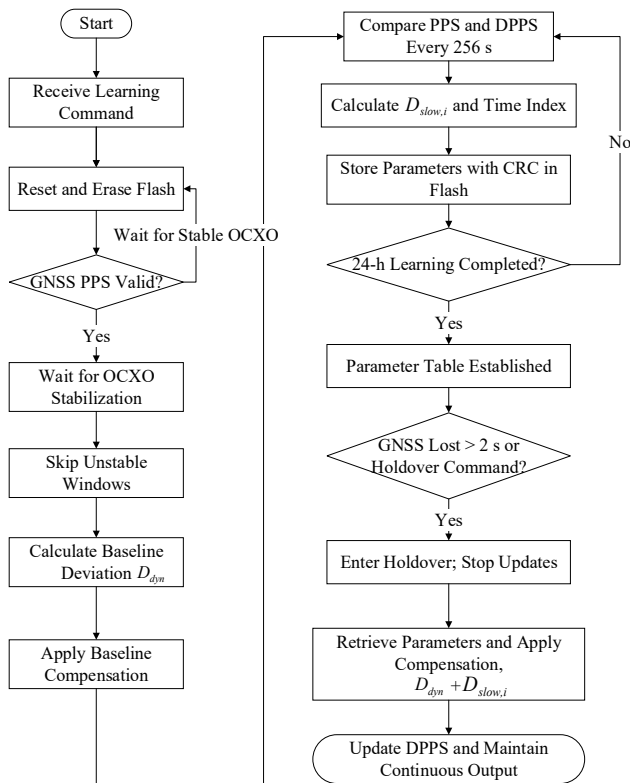


Figure 2. Autonomous parameter acquisition, storage, and holdover compensation procedure.

4. Experimental Verification

The experimental platform consisted of an FPGA-based timing recorder, a GNSS timing receiver, a 10 MHz OCXO, an oscilloscope, and a computer used for serial monitoring, as shown in Figure 3. The OCXO output was multiplied to 100 MHz by the FPGA PLL, corresponding to a nominal 10-ns counting interval. The oscilloscope was used to observe timing signals, while measurement and operating information could be monitored through the UART interface.



Figure 3. Experimental platform used for timekeeping evaluation.

Five OCXOs of the same type were selected to evaluate the proposed method. The experiments consisted of two independent stages: parameter acquisition under GNSS availability and holdover verification after GNSS signal interruption.

During the parameter acquisition stage, the system first measured the baseline timing deviation and applied dynamic compensation. It then operated continuously for 24 h, measuring the residual timing deviations in successive 256-s windows and storing the corresponding compensation parameters in SPI Flash.

After parameter acquisition, the system was restarted while retaining both the baseline dynamic parameter and residual drift parameters in SPI NOR Flash. After OCXO stabilization, the stored baseline parameter was read and applied; the residual parameters were then retrieved according to the elapsed holdover time. The GNSS reference was disconnected from the compensation control path for an independent 24-h holdover test. A reference PPS was used only by an independent measurement channel to evaluate DPPS timing error, without participating in compensation control or parameter updates.

4.1 Experimental Results and Analysis

During the parameter acquisition stage, the timing deviations of the five OCXOs were measured over 24 h. The results are shown in Figure 4.

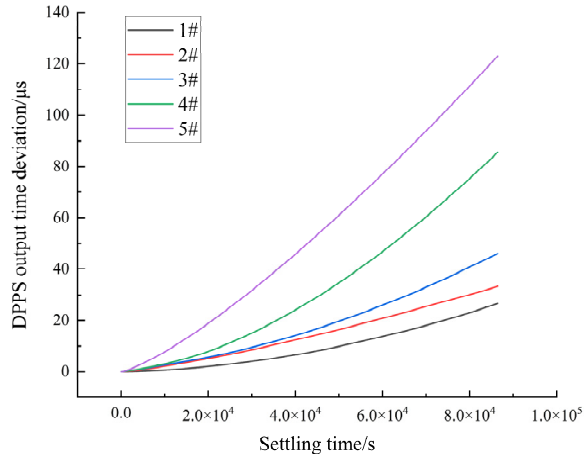


Figure 4. Measured timing deviations of five OCXOs during the parameter acquisition stage.

The measured 24-h timing deviations differed among the five oscillators, reaching approximately 20–145 μs . This variation indicates that the individual OCXOs did not share identical residual drift characteristics under the test conditions. Their measured deviations were therefore recorded as device-specific parameters for subsequent holdover use.

After the acquisition stage, the stored parameters were used for independent holdover compensation. Figure 5 shows the timing error curves of the five OCXOs during the 24-h holdover experiment.

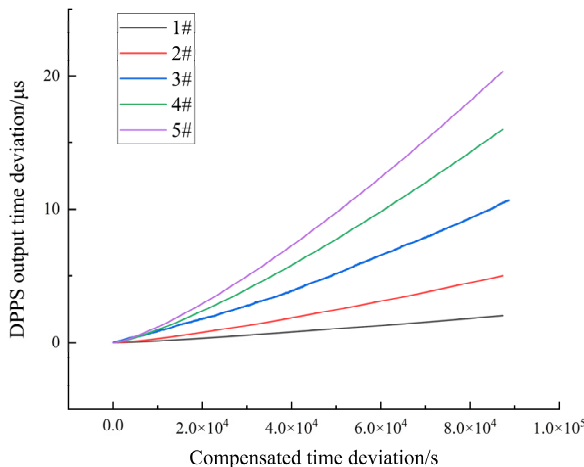


Figure 5. Independent 24-h holdover timing errors using stored autonomous parameters.

As shown in Figure 5, the timing errors of all five OCXOs remained below 25 μs over the independent 24-h holdover period. The experiment demonstrates that the parameters acquired with GNSS available can be retrieved and applied after restarting the system without a PPS input to the compensation control. The acquisition-stage curves in Figure 4 and the independent holdover curves in Figure 5 represent different runs; they are not

presented as a paired, same-condition compensation-on/off comparison.

5. Conclusion

An FPGA-based two-stage autonomous parameter acquisition and holdover compensation method has been implemented for OCXO-based timing nodes. During GNSS availability, the system acquires a baseline dynamic correction and then measures the residual drift in successive 256-s windows. The parameters are saved in SPI NOR Flash and retrieved by elapsed holdover time to adjust the DPPS period without external PPS-based updates. Experiments on five same-type OCXOs, using 24-h parameter acquisition and independent 24-h holdover tests, showed maximum absolute timing errors below 25 μs over 24 h under the tested conditions. This method reduces reliance on host-computer-based offline calibration while retaining a simple FPGA-oriented compensation structure.

References

- [1] Liu H., Zhu Y., Bi R., et al. Time-base synchronous storage and testing technology for shock-wave overpressure. *Modern Electronics Technique*, 2025, 48(6): 1–8 (in Chinese).
- [2] Xie W., Wang K., Fu W., et al. Regional multi-station real-time time transfer using an undifferenced multi-GNSS network solution. *GPS Solutions*, 2024, 28: 198.
- [3] Xu L., Ye P., Liao S., et al. How does ambient temperature fluctuation influence the short-term frequency stability of OCXO? *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2023, 70(8): 893–902.
- [4] Dai C., Zhan K., Jiang S., et al. Design of a miniaturized low-phase-noise oven-controlled crystal oscillator. *Journal of Time and Frequency*, 2024, 47(2): 97–103 (in Chinese).
- [5] Wang B., Yan X., Zhang H., et al. Frequency drift prediction of an OCXO based on a GRU neural network. *Chinese Journal of Electron Devices*, 2024, 47(2): 502–508 (in Chinese).
- [6] Huang Y., Wang H., Tu S., et al. An improved Kalman-filter-based timing calibration and holdover method for satellites. *Journal of Beijing University of Aeronautics and Astronautics*, 2025, 51(6): 2157–2164 (in Chinese).
- [7] Hao Y., Liang K., Yu T., et al. Precision disciplining of crystal oscillators based on DDS frequency micro-steps. *Systems Engineering and Electronics*, 2026, 48(5): 1706–1714 (in Chinese). <https://doi.org/10.12305/j.issn.1001-506X.2026.05.26>.
- [8] Fan D., Liu Y., Li X., et al. Influence of temperature on time delays in GNSS common-view equipment. *Journal of Time and Frequency*, 2024, 47(3): 153–158 (in Chinese).
- [9] Fan D., Liu Y., Li X., et al. A high-performance time holdover method based on crystal oscillators. *Journal of Time and Frequency*, 2021, 44(1): 26–32 (in Chinese).
- [10] Chen S., Liu Y., Guo M., et al. Real-time BDS-3 PPP timing performance considering DCB/OSB. *Journal of Time and Frequency*, 2025, 48(3): 188–198 (in Chinese).
- [11] Jia P., Ma Y., Guo X. A holdover timing method for distributed shock-wave pressure measurement systems. *Journal of Ordnance Equipment Engineering*, 2026, 47(1): 218–224 (in Chinese). <https://doi.org/10.11809/bqzbgcxb2026.01.025>.
- [12] Bui T. Q. T., Elango A., Landry R. J. FPGA-based autonomous GPS-disciplined oscillators for wireless sensor network nodes. *Sensors*, 2022, 22(9): 3135. <https://doi.org/10.3390/s22093135>.

- [13] Kubiczak P., Jessa M., Kasznia M. Fast frequency and phase synchronization of high-stability oscillators with 1 PPS signal from satellite navigation systems. *GPS Solutions*, 2025, 29: 52. <https://doi.org/10.1007/s10291-025-01818-8>.
- [14] Gill E., Palo S. Towards a GNSS-assisted autonomous heterogeneous clock system for very small satellites in the Earth–Moon system. *Acta Astronautica*, 2025, 234: 207–213.
- [15] Li Z., Peng J., Miao M., et al. Compensation method based on phase shift between pins of crystal resonator. *Sensors*, 2025, 25(3): 932.
- [16] Wang D., Liao W., Liu B., et al. A multi-constraint co-optimization LQG frequency steering method for LEO satellite oscillators. *Sensors*, 2025, 25(15): 4733.
- [17] Cao Z., Li K., Liu S., et al. MMC: GNSS-based high-precision clock generation and holdover system with MRAC control and MLP aging prediction. *Measurement*, 2026, 272: 121075. <https://doi.org/10.1016/j.measurement.2026.121075>.