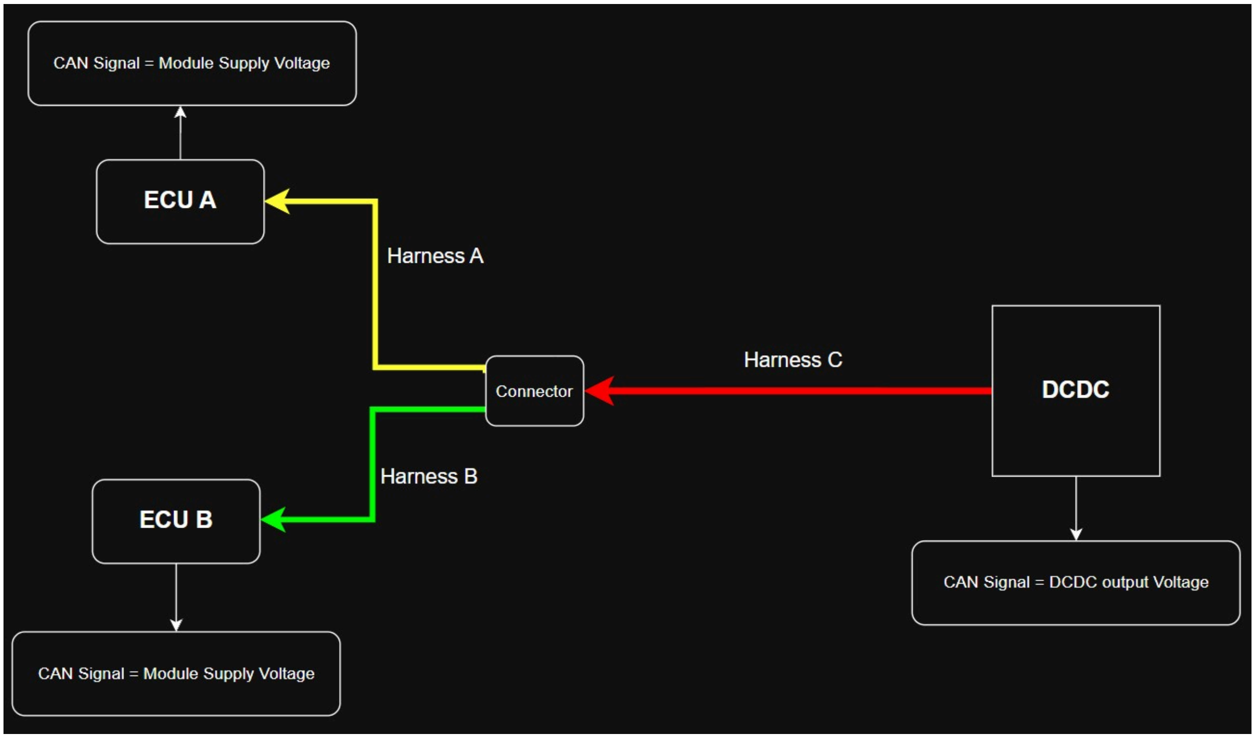
Live Harness Degradation Demo – Interpreting the Log

This section walks through a real test where we manually degraded **ECU A supply voltage** on the UNO Q system and watched our **live predictive harness monitor** respond in real time.



1. Setup recap

- Hardware:
    - Arduino **UNO Q** + MCP2515 CAN transceiver
    - Three "virtual ECUs" on the CAN bus:
        - **ECU A** – `ECUA_Supply_Voltage`
        - **ECU B** – `ECUB_Supply_Voltage`
        - **DCDC** – `DCDC_Output_Voltage`
- Normal operation:
    - All three around **14.1 V** (healthy charging system)
- Fault model:
    - Harness A adds resistance → ECU A sees *lower* voltage than DCDC
    - ECU B stays correct (still ~14.1 V)
    - DCDC remains the reference

On Linux, the Python app decodes CAN using the DBC and computes:

- `ΔA = V_DCDC - V_A`
- `ΔB = V_DCDC - V_B`
- Smoothed deltas: `ΔA_ewma`, `ΔB_ewma`
- Drift trend: `trendA`, `trendB` (approx. V/s)

It then emits:

- **[STATS]** – periodic "health summary"
- **[EARLY][HARNESS_A_DRIFT]** – early warning based on EWMA + trend
- **[ALERT][HARNESS_A]** – hard fault based on absolute thresholds

---

2. Normal operation – all good

At the start of the log, we are in the healthy regime:

```
[STATS] frames_seen=3 decoded=3 ECU_A=14.10 V ECU_B=14.10 V DCDC=14.10 V
...
(repeated)
```

Interpretation:

- We see 3 frames per second (A, B, DCDC).
- All three voltages are identical (14.10 V).
- `ΔA` and `ΔB` are essentially zero.
- No EARLY or ALERT lines – the harness is considered healthy.

---

3. Slow degradation of Harness A – early drift detection

We then start manually dropping **ECU A** while keeping **ECU B** and **DCDC** at 14.1 V:

```
[STATS] ... ECU_A=14.00 V ECU_B=14.10 V DCDC=14.10 V
[STATS] ... ECU_A=13.90 V ECU_B=14.10 V DCDC=14.10 V
[STATS] ... ECU_A=13.80 V ECU_B=14.10 V DCDC=14.10 V
[STATS] ... ECU_A=13.70 V ECU_B=14.10 V DCDC=14.10 V
```

At this point, the deltas are growing:

- `ΔA` ≈ 0.1 → 0.4 V
- `ΔB` ≈ 0.0 V (ECU B still matches DCDC)
- Once the smoothed delta and trend cross our **early drift thresholds**, the system starts issuing predictive warnings:

```
[EARLY][HARNESS_A_DRIFT] ECU_A=13.70 V ECU_B=14.10 V DCDC=14.10 V |
  ΔA_ewma=0.31 V, trendA=0.0160 V/s, ΔB_ewma=0.00 V
[EARLY][HARNESS_A_DRIFT] ECU_A=13.60 V ECU_B=14.10 V DCDC=14.10 V |
  ΔA_ewma=0.40 V, trendA=0.0181 V/s, ΔB_ewma=0.00 V
```

Key points:

- **ΔA_ewma ~ 0.3–0.4 V** – ECU A is consistently low vs DCDC.
- **trendA ~ 0.016–0.02 V/s** – ECU A is *moving downward* over time.
- **ΔB_ewma ≈ 0** – ECU B still tracks DCDC perfectly.

This is our **predictive maintenance layer**:
the system is telling us "Harness A is drifting" **before** a hard fault.

---

4. Severe degradation – hard fault on Harness A

We continue pulling ECU A down into a clear-fault region:

```
[ALERT][HARNESS_A] ECU_A=12.90 V ECU_B=14.10 V DCDC=14.10 V |
  ECU_A low vs DCDC (ΔA=+1.20 V, ΔB=+0.00 V, DCDC≈14.1 V)
...
[ALERT][HARNESS_A] ECU_A=11.90 V ECU_B=14.10 V DCDC=14.10 V |
  ECU_A low vs DCDC (ΔA=+2.20 V, ΔB=+0.00 V, DCDC≈14.1 V)
...
```

```
[ALERT][HARNESS_A] ECU_A=10.40 V ECU_B=14.10 V DCDC=14.10 V |
  ECU_A low vs DCDC (ΔA=+3.70 V, ΔB=+0.00 V, DCDC≈14.1 V)
```

Here the **rule-based** thresholds kick in:

- DCDC is still ≈ 14.1 V (good supply).
- ECU B is still ≈ 14.1 V (harness B OK).
- ECU A has fallen several volts below DCDC:
  - ΔA = 1.2, 1.7, 2.2, 2.7, 3.3, 3.6, 3.7 V…

At the same time, the statistical indicators reflect a severe fault:

```
[EARLY][HARNESS_A_DRIFT] ECU_A=10.80 V ECU_B=14.10 V DCDC=14.10 V |
  ΔA_ewma=2.12 V, trendA=0.0922 V/s, ΔB_ewma=0.00 V
...
[EARLY][HARNESS_A_DRIFT] ECU_A=10.40 V ECU_B=14.10 V DCDC=14.10 V |
  ΔA_ewma=3.37 V, trendA=0.0907 V/s, ΔB_ewma=0.00 V
```

- **ΔA_ewma > 2–3 V** – large, sustained drop.
- **trendA ~ 0.09–0.10 V/s** – ECU A is falling *fast* over time.
- `ΔB_ewma` remains ~0, confirming the issue is localized to harness A.

This is where, in a real system, we would:

- Raise a high-severity predictive event,
- Log the episode for off-board diagnostics,
- Potentially derate / limp the system if this harness powers safety-relevant loads.

---

5. Recovery – and memory of what happened

When you start ramping ECU A **back up** (10.4 → 10.7 → 11.3 → 11.8 → 12.4 → 13.0 → 13.5 → 14.0 V), the log shows:

STATS following ECU_A as it improves:

```
[STATS] ... ECU_A=13.50 V ECU_B=14.10 V DCDC=14.10 V
[STATS] ... ECU_A=14.00 V ECU_B=14.10 V DCDC=14.10 V
```

-

EARLY drift metrics gradually decay:

```
[EARLY][HARNESS_A_DRIFT] ECU_A=13.50 V ... ΔA_ewma=1.78 V,
trendA=0.0128 V/s
...
[EARLY][HARNESS_A_DRIFT] ECU_A=14.00 V ... ΔA_ewma=1.32 V,
trendA=0.0021 V/s
```

-

The EWMA and trend don't instantly snap back to zero – they "remember" that this harness has been misbehaving, which is exactly what you want for **recent-history-aware** diagnostics. Eventually, as ECU A returns to normal and stays there, the metrics decay and the console returns to simple `[STATS]` lines with no alerts.

---

6. Summary

This log demonstrates the full predictive maintenance flow on the UNO Q:

1. **Normal** operation – all ECUs at 14.1 V.
2. **Early drift detection** – small but systematic drop on ECU A vs DCDC/B.
3. **Hard fault alert** – large sustained drop on ECU A, with B + DCDC still healthy.

4. **Recovery** – ECU A returns to normal; the algorithm's EWMA/trend slowly relax as evidence of fault subsides.

From a demo perspective, this is very powerful: you can show the **live App Lab console**, tweak only ECU A's TX values, and watch the system move from **green** → **amber** → **red** → **recovery**, exactly as a real telematics/predictive node would on a vehicle CAN network.