DEO-SynCore：預測式離散運作包絡控制

v2.0 / 2026-07-29 / Neo.K（許筌崴） / 公開概念技術論文／控制子系統架構提案

證據狀態：尚未完成專用矽原型或第三方重現；本文提出的效益均為待驗證假說，不是既成產品性能

https://thisoneisneok.com/html/papers/ccp-13.html

SECTION

從「超頻檔位」到效能、功率、熱與可靠度的可驗證協同

DEO-SynCore: Predictive Discrete Operating-Envelope Orchestration

From Overclocking Levels to Verifiable Performance–Power–Thermal–Reliability Coordination

作者：Neo.K（許筌崴）

機構：一言諾科技有限公司（EveMissLab），台灣

原始版本：2025 年 12 月

公開修訂版：v2.0，2026 年 7 月 29 日

文件性質：公開概念技術論文／控制子系統架構提案

系列定位：O-Chip、SynCore v2.0、DPCPS 與 DryCore 之間的運作狀態協同層

建議縮寫：DEO（Discrete Envelope Orchestration，離散包絡編排）

證據狀態：尚未完成專用矽原型或第三方重現；本文提出的效益均為待驗證假說，不是既成產品性能

SECTION

修訂聲明

本文由《DEO-SynCore：離散能級超頻革命與量子態的工程背叛》重構而來。原稿最有價值的洞察，是認為處理器不應只依賴遲緩的事後反應，而可根據工作負載階段、功率餘裕與熱狀態，主動切換少數具有明確意義的運作狀態。原稿也已在後段意識到：「量子跳躍」只能作為工程類比，實際電壓與溫度仍在連續時間中演化。

然而，原稿同時存在數個根本問題：現代處理器本來就使用離散 P-state、硬體控制效能狀態或抽象效能提示；固定的五組頻率與功耗不是跨製程、跨晶片可通用的物理能級；電壓不能靠切換數組預充超級電容直接送入核心；一般 DVFS 不需要凍結整條流水線；五組 PLL 也不是達成快速轉換的必要條件。更嚴重的是，原稿將假設平台與推演數據寫成 CPU-Z、Cinebench、遊戲、模擬器和 Blender 的「實測結果」，容易讓讀者誤認為硬體已存在。

v2.0 因此進行以下重構：

- 將 DEO 從 Discrete Energy Overclocking 改為 Discrete Envelope Orchestration。其對象不再是單一時脈，而是包含電壓、頻率、核心配置、架構模式、功率上限、溫度、持續時間與可靠度預算的多維運作包絡。
- 承認現有技術基線。Intel Enhanced SpeedStep、Hardware-Controlled Performance States、ACPI CPPC、Linux CPUFreq 與 AMD P-State 已提供離散或細粒度效能控制；DEO 的新意只能來自跨層協同、預測控制、狀態認證與可稽核安全，不是「首次讓頻率離散化」。
- 移除量子力學本體宣稱。DEO 是離散狀態控制器與連續物理系統組成的混合動態系統，不是量子處理器，也不存在宏觀波函數坍縮。
- 刪除固定五檔的通用頻率、電壓、功耗與持續時間。五類包絡只保留功能語義；實際參數必須由每個晶片、封裝、冷卻與使用情境校準。
- 刪除所有虛構實測數據。原稿中的跑分、幀率、功耗、溫度、預測準確率與電費數字，不再作為證據。
- 修正轉換機制。升頻前先建立必要電壓裕度；降壓前先降低頻率；切換必須受電源完整性、PLL／分頻器、時脈閘控、PVT 與硬體護欄約束。
- 將 O-Chip 的「預知」改為多時間尺度負載預測與明確提示。預測可來自效能計數器、排程器、程式階段、API 提示和歷史資料；模型信心不足時必須回退到反應式硬體控制。
- 將 DEO 降為 SynCore 的控制子模組，而非另一顆獨立革命晶片。它可以單獨在現有 CPU 上做軟體原型，但完整價值來自與 SynCore 模式切換、DPCPS 供電和 DryCore 熱管理協同。
- 加入可否證假說、基線比較、消融實驗與證據分級，使後續 MVP 能判斷 DEO 是否真的優於現有 HWP／CPPC／schedutil，而不是只證明它「可以運作」。

因此，新版保留的是「有意義狀態＋預測切換＋爆發預算」的核心，而不是原稿的固定超頻數字與量子化修辭。

SECTION

摘要

現代處理器的運作狀態不是單一頻率參數，而是電壓、時脈、啟用核心、微架構資源、快取與記憶體狀態、封裝功率、局部溫度、冷卻能力及可靠度壓力共同形成的多維決策。既有動態電壓與頻率調整技術已能以 P-state、硬體自主管理或抽象效能提示在多個工作點之間轉換，因此真正未解的問題不是「頻率應不應離散」，而是：系統能否把這些工作點組織成少量可認證、可解釋、可預測切換且具有安全邊界的運作包絡，並與作業系統、應用程式、供電與冷卻協同。

本文提出 DEO-SynCore，即預測式離散運作包絡控制架構。DEO 將一個包絡定義為一組被認證的允許區域，而非固定頻率點。每個包絡包含電壓—頻率範圍、核心與執行模式、功率上限、最高溫度、最大駐留時間、轉換守衛條件與可靠度預算。系統可以在包絡內由硬體執行細粒度調整，但跨包絡切換必須由預測控制器、轉換定序器與獨立安全監督器共同完成。本文提出五類功能包絡：保留、效率、持續、爆發與合格峰值；這五類是語義模板，而非所有處理器都必須採用的固定檔位。

DEO 使用多時間尺度訊號進行決策：微秒級效能計數器描述當前瓶頸；毫秒級排程與執行階段提示描述即將到來的負載；秒級熱模型與冷卻狀態決定爆發預算；較長時間尺度的老化帳本限制高電壓、高電流與熱循環累積。控制問題被表述為受約束的混合系統最佳化，在吞吐量、尾端延遲、能耗、溫度、切換成本與可靠度之間求解。當預測信心不足、感測資料失真或模型超出校準域時，系統必須回退到既有 HWP、CPPC 或硬體保護策略。

本文提出可在現有 Linux 平台開始的軟體 MVP：以 intel_pstate／amd-pstate、CPUFreq、PM QoS、效能計數器、溫度與功率遙測，將包絡映射為最小／最大效能提示、能源偏好與功率上限；再以工作負載提示和短期預測選擇包絡。後續可透過 FPGA、可程式化 PMIC、RISC-V 功率控制器或硬體在迴路平台驗證安全轉換與多域協同。

DEO 的可檢驗主張不是「五檔一定比連續控制更快」，而是：在特定工作負載與硬體上，少量經認證的多維包絡、預測式切換、遲滯控制與可靠度配額，可能降低狀態震盪、改善相位切換時的尾端延遲，並在相同功率、溫度與壽命約束下提高有效工作量。若它無法在嚴格基線與消融實驗中勝過現有控制器，則此架構應被否證、縮減或只保留為可解釋的策略層。

關鍵詞：DVFS、效能狀態、運作包絡、預測控制、混合動態系統、硬體控制 P-state、CPPC、功率管理、動態熱管理、可靠度管理、SynCore、O-Chip

SECTION

1.1 現代處理器早已不是「連續旋鈕」

早期作業系統常把處理器效能控制理解為一組 P-state：每個狀態對應特定或近似的電壓與頻率工作點。Intel Enhanced SpeedStep 的官方文件已明確把 P-state 轉換描述為前往新目標工作點的離散轉換；後續 Hardware-Controlled Performance States 則把更多決策交給硬體。ACPI CPPC 進一步將效能表示為抽象、無單位的相對尺度，使作業系統提供目標和偏好，由平台自行決定具體電壓、頻率及其他控制量。[1][2]

Linux 的 CPUFreq、intel_pstate 與 amd-pstate 也顯示，現有系統同時包含兩種層次：

- 軟體可見的效能請求、上下限和能源偏好；
- 硬體內部根據 PVT、功率、溫度及核心活動選擇實際工作點。[3][4]

因此，下列敘述不能作為 DEO 的原創性：

> 「處理器不該平滑升頻，而應在離散狀態間跳躍。」

現代處理器本來就會在可用工作點、倍頻、電壓域、時脈域和功率狀態間轉換。DEO 必須解決更高層的問題：如何將多個控制量組織成具有功能語義、安全證書和跨層協同能力的運作包絡。

SECTION

1.2 頻率不是完整的運作狀態

同樣的 4GHz，可能對應完全不同的系統行為：

- 一個核心高負載或全部核心中度負載；
- 向量單元開啟或關閉；
- 快取命中或記憶體停滯；
- SynCore 神核模式、流動模式或混合模式；
- 不同核心電壓、SoC 電壓與記憶體時脈；
- 不同封裝功率上限；

- 不同風扇、泵、冷卻水或熱虹吸狀態；
- 不同晶片溫度與剩餘爆發時間。

處理器的動態功率常以近似式描述：

$$P_{dyn} \approx \alpha C_{eff} V^2 f,$$

其中 α 是活動因子，C_{eff} 是等效切換電容，V 是供應電壓，f 是時脈。此式已說明：頻率只是一個變數；若升頻需要更高電壓，功率可能以更快速度增加。總功率還包括漏電、記憶體、互連、I/O、調節器損耗與冷卻功率：

$$P_{sys} = P_{dyn} + P_{leak} + P_{mem} + P_{io} + P_{vr} + P_{cool}.$$

因此，DEO 的基本單位不能只是 (V,f)，而必須是多維包絡。

SECTION

1.3 為何少量語義狀態仍可能有價值

既有硬體已能細粒度調整，為何還需要少量包絡？其理由不是「中間態沒有物理意義」，而是控制、驗證和軟硬體協同需要穩定介面。

若作業系統、應用程式、電源與散熱各自操控數十或數百個參數，組合狀態數量可能迅速膨脹。即使每個參數只有十個可選值，八個參數的笛卡兒積也可達：

> 10□

種組合。大多數組合沒有被完整驗證，也不適合公開給應用程式直接控制。

少量包絡的價值在於：

- 將大量底層狀態壓縮成可理解的系統意圖；
- 為每類意圖建立安全邊界與測試證書；
- 降低頻繁來回切換造成的震盪；
- 允許應用程式表達「低延遲」「持續吞吐」「短時爆發」而不指定危險電壓；
- 讓功率、熱與可靠度控制器共享同一決策語言。

這是一種控制介面的量化，不是對物理連續性的否認。

SECTION

1.4 DEO 的核心命題

本文提出：

> 處理器應保留底層細粒度調節能力，但向系統暴露一組少量、經認證、具有語義且可預測切換的多維運作包絡。

可概括為：

> 底層連續／細粒度控制
> +
> 上層離散語義包絡
> +
> 安全守衛轉換
> =
> DEO.

SECTION

2.1 包絡向量

對平台 p，定義第 k 個運作包絡的名義描述為：

> e⊠
> =
> (
> cal-V⊠,
> cal-F⊠,
> cal-C⊠,
> cal-M⊠,
> cal-P⊠,
> cal-T⊠,
> Tau⊠,
> cal-R⊠
>).

其中：

- cal-V⊠：允許的電壓區域；
- cal-F⊠：允許的時脈或抽象效能範圍；
- cal-C⊠：可啟用的核心、執行單元與功率域集合；
- cal-M⊠：允許的微架構或 SynCore 運作模式；
- cal-P⊠：封裝、核心、SoC、記憶體與系統功率限制；
- cal-T⊠：溫度、溫差與熱通量限制；
- Tau⊠：最短駐留、最長駐留、冷卻與重入時間；
- cal-R⊠：可靠度、電流密度、電壓壓力與老化預算。

  包絡不是單一點，而是可行集合：

$$
\begin{aligned}
&\text{cal-E}\boxtimes \\
&= \\
&\{ \\
&z \\
&\mid \\
&g_{k,j}(z,x) \leq 0, \\
&\ j=1,\ldots,m\boxtimes \\
&\},
\end{aligned}
$$

其中 z 是底層控制量，x 是當前感測與估計狀態，$g_{k,j}$ 是安全、熱、功率、時序或可靠度約束。

SECTION

2.2 混合動態系統

DEO 的物理狀態在時間中連續演化，但控制器選擇的包絡索引是離散的。令：

- $x(t)$：溫度、電壓、電流、工作佇列、時脈、利用率與老化狀態；
- $q(t) \in \{0,1,\ldots,K-1\}$：當前包絡；
- $u(t)$：硬體控制量；
- $w(t)$：工作負載與環境擾動。

在包絡 q 內，系統滿足：

$$
\begin{aligned}
&\dot{x}(t) \\
&= \\
&F_q\bigl(x(t),u(t),w(t)\bigr).
\end{aligned}
$$

當守衛條件成立時，離散狀態可轉換：

$$
\begin{aligned}
&q(t\boxtimes) \\
&= \\
&G\bigl(q(t\boxtimes),x(t),\hat{w}_{t:t+H},c(t)\bigr),
\end{aligned}
$$

其中 hatw_(t:t+H) 是預測視窗，c(t) 是模型信心。

這個模型比「CPU 真的發生量子跳躍」更精確：

> 離散的是控制模式，連續的是物理演化。

SECTION

2.3 包絡內控制與包絡間控制

DEO 將控制分成兩層：

2.3.1 包絡內控制

由現有硬體、韌體或作業系統進行細粒度調整，例如：

- HWP／CPPC 依負載選擇實際效能；
- 調節器控制輸出電壓；
- 時脈產生器調整倍頻；
- 風扇與泵依溫度調速；
- 核心進入不同 idle state。

2.3.2 包絡間控制

由 DEO 選擇高階運作意圖，例如：

- 是否允許進入短時爆發；
- 是否切換 SynCore 的神核、流動或混合模式；
- 是否預留功率給 GPU、記憶體或 I/O；
- 是否啟動冷卻預調節；
- 是否因老化配額或電源完整性限制禁止峰值狀態。

這種分層避免 DEO 與現有硬體控制器互相爭奪每一個微小頻率決策。

SECTION

2.4 包絡不必固定為五個

原稿固定五個能級。v2.0 將其改為預設模板：

> $K_p \in \{3,4,5,\ldots\}$

由平台需求決定。行動裝置可能需要更多低功耗層次；桌面工作站可能只需要效率、持續與爆發三類；資料中心則可能按服務水準、機櫃功率和冷卻區域建立不同包絡。

五類包絡的價值只是提供共同語言，而不是物理定律。

SECTION

3.1 E0：保留包絡（Retention Envelope）

E0 對應系統長時間空閒、等待事件或只維持必要背景功能的狀態。其目標是降低靜態功耗，同時滿足喚醒延遲與資料保留需求。

典型約束包括：

- 深度 idle state 或核心斷電；
- 最少常駐核心；
- 記憶體與必要 I/O 保留；
- 喚醒延遲上限；
- 網路、計時器或安全控制器的最小待命功能。

  Linux CPUIdle 已明確把 idle state 的目標駐留時間、退出延遲和 PM QoS 約束納入選擇，因此 DEO 不應重新發明 idle state，而應把應用程式的延遲需求與整機包絡結合。[5]

SECTION

3.2 E1：效率包絡（Efficiency Envelope）

E1 服務互動性低或吞吐要求不高的任務。控制目標是最小化每單位有效工作量的能耗，而不是最低瞬時功率。

可能包括：

- 偏向能源效率的 HWP／EPP 或 CPPC 提示；
- 較少啟用核心；
- 降低非必要記憶體與互連頻率；
- 延遲不敏感背景任務批次化；
- 避免過度頻繁地進入短暫高效能狀態。

  對某些任務，較高頻率快速完成後進入深度 idle 可能比低頻長時間執行更節能；對記憶體受限任務，升頻可能幾乎不增加吞吐。因此 E1 必須以工作量、停滯與延遲共同判斷，不能簡化為「永遠最低頻率」。

SECTION

3.3 E2：持續包絡（Sustained Envelope）

E2 是能在指定環境與冷卻能力下長時間運作的主要性能狀態。其關鍵不是最高頻率，而是穩態可持續性：

> bar P_(load)
> ≤
> P_(cool,sustained)

且：

> T⊠(t)
> ≤
> T_(safe)
>
> for the required mission duration.

E2 應作為大多數長時間編譯、渲染、科學計算、模型訓練或伺服器負載的基準，而不是把所有持續任務送進峰值包絡。

SECTION

3.4 E3：爆發包絡（Burst Envelope）

E3 使用系統暫時可用的電、熱與可靠度餘裕，服務短時且對延遲敏感的工作階段，例如：

- 互動程式的突發工作；
- 幀時間中的關鍵 CPU 階段；
- 編譯鏈中的序列瓶頸；
- 模型推論的尾端延遲高峰；
- SynCore 模式切換後的關鍵序列區段。

  其核心資源是有限的「爆發預算」。若將熱系統近似為一階 RC 模型：

> C_(th)(dT)/(dt)
> =
> P(t)
> -
> (T(t)-T_a)/(R_(th)),

則可用保守形式限制超出持續冷卻能力的累積能量：

> B_(th)(t)
> =
> ∫ _(t⌧)^(t)
> [P(τ)-P_(cool,sustained)]⌧
> dτ.

當 B_(th) 接近平台校準上限時，控制器必須退出 E3。這比固定「爆發十秒、冷卻五秒」更合理，因為實際可用時間取決於起始溫度、散熱器熱容、環境、風量、液溫與負載分布。

SECTION

3.5 E4：合格峰值包絡（Qualified Peak Envelope）

E4 不是消費者可以任意解除限制的「賽亞人模式」，而是只有在硬體、供電、冷卻、可靠度與安全條件均通過時才允許的峰值包絡。它可用於：

- 製造測試與晶片分級；
- 受控實驗室超頻；
- 時間極短且商業價值很高的工作；
- 有專用冷卻與硬體監督的工作站或 HPC 節點。

必要條件至少包括：

> $V \le V_{qualified},;$
> $I \le I_{qualified},;$
> $T\boxtimes \le T_{qualified},;$
> $Z_{PDN} \le Z_{target},;$
> $D_{age} \le D_{budget},;$
> $\tau_{E4} \le \tau_{qualified}.$

任何一項感測失效或越界，都應由硬體安全層退出 E4，不得等待 AI 控制器決定。

SECTION

3.6 正式名稱與介面暱稱

原稿的 Zen、Kaioken、Super Saiyan 具有傳播力，但不適合作為工程規格。若產品介面希望保留，可限定為使用者介面的暱稱：

正式包絡	可選介面暱稱	規格文件中的地位
E0 保留	Zen	非技術別名
E1 效率	Cruise	非技術別名
E2 持續	Ready	非技術別名
E3 爆發	Kaioken	非技術別名
E4 合格峰值	Super Saiyan	實驗室／展示別名

安全、測試、API 與論文應只使用 E0–E4 或正式名稱。

SECTION

4.1 L0：遙測與狀態估計層

DEO 首先需要可信資料，而不是先需要大型 AI。輸入可包括：

- 每核心利用率、IPC、指令與週期；
- 分支失誤、前端／後端停滯；
- 快取失誤、記憶體頻寬與延遲；
- runnable queue、任務優先度與 deadline；
- 封裝、核心、SoC、GPU、記憶體與 VRM 功率；
- 核心、封裝、冷卻液與環境溫度；
- 電壓下陷、電流、功率限制與熱限制事件；
- 風扇、泵、熱虹吸與冷卻能力；
- 目前包絡、駐留時間與切換歷史；
- 老化與高壓力使用帳本。

感測值應附帶時間戳、更新率、可信度與缺失標記。控制器不能把不同延遲與取樣頻率的資料當成同時瞬間值。

SECTION

4.2 L1：多時間尺度負載預測層

「預知」改為四種時間尺度的預測：

預測尺度	主要資料	主要用途
10–100 µs	程式計數器、停滯、硬體事件	細粒度 DVFS 或局部資源調節研究

0.1–10 ms	排程器、佇列、幀／批次階段	預先進入 E2／E3、降低尾延遲
10 ms–2 s	應用 API、工作圖、場景提示	冷卻預調節、模式切換、功率預留
秒至分鐘	熱 RC 模型、液溫、環境與任務長度	爆發預算、持續包絡與機櫃協同

這些區間是研究分層，不是通用硬體規格。既有研究已顯示，在支援細粒度 DVFS 的 GPU 情境中，程式計數器與停滯資訊可用於預測短時間尺度的適當工作點；多域 DVFS 也可透過預測計算、I/O 與記憶體需求重新分配功率預算。[6][7]

預測器可以是：

- 規則與狀態機；
- 指數平滑或 AR 模型；
- 輕量線性模型；
- 決策樹或梯度提升；
- 模型預測控制；
- 受約束強化學習；
- 應用程式明確提示。

  AI 不是必要條件。只要簡單模型能在低延遲與低風險下達成同等效益，就不應為了「AI 化」而增加控制複雜度。

SECTION

4.3 L2：包絡規劃器

規劃器在候選包絡中最小化多目標代價：

> q^*
> =
> argmin_(q∈cal-Q_(safe))
> J(q),

其中：

> J(q)=
> w_Lhat L_(p99)(q)
> +w_Ehat E_(task)(q)
> +w_TΦ_T(q);
> +w_RΦ_R(q)
> +w_S C_(switch)(q)
> -w_What W_(useful)(q).

各項分別代表：

- 預測尾端延遲；
- 任務能耗；
- 熱風險懲罰；
- 可靠度壓力；
- 切換成本；
- 有效工作量或吞吐效益。

  可行集合 cal-Q_(safe) 必須先由硬體與安全約束篩選，AI 不得把非法包絡放回候選集合。

SECTION

4.4 L3：轉換定序器

定序器把高階包絡切換轉換成底層安全操作。一般 DVFS 的基本順序是：

升頻

> 建立電壓與電流裕度
> →
> 確認 PDN／PVT 安全
> →
> 提高時脈或效能請求.

降頻

> 先降低時脈或效能請求
> →
> 確認時序裕度
> →
> 降低電壓.

若切換涉及 SynCore 的架構模式、快取所有權、核心融合、相位耦合或上下文格式，則需額外的 quiescent point、barrier、狀態保存與恢復；這與普通頻率切換不同。

SECTION

4.5 L4：致動器與跨域協同

DEO 可使用的致動器包括：

- HWP／CPPC 最小、最大、期望效能與能源偏好；
- CPUFreq governor 或平台驅動；
- 核心啟停、親和性、任務遷移與優先度；
- 封裝功率與電流上限；
- GPU、記憶體與互連 DVFS；
- SynCore TMC 模式；
- DPCPS 的相數、局部調節器與功率裕度；
- DryCore 的風扇、泵、冷端與預冷設定；
- 機櫃級功率與冷卻分配。

  跨域功率管理很重要，因為固定給 I/O、記憶體或某個加速器的功率預算，可能在其低利用率時被浪費，而計算域同時受限。SysScale 類研究已顯示，多域需求預測與功率重分配可以在特定平台改善性能或能效，但效益高度依賴工作負載與硬體，不能直接外推為 DEO 的既成成果。[7]

SECTION

4.6 L5：獨立安全監督器

安全監督器必須在 DEO 控制器之外，至少提供：

- 過壓、欠壓、過流、過溫硬體保護；
- 感測器一致性與失效偵測；
- 看門狗與控制器逾時；
- 非法轉換攔截；
- 包絡最大駐留時間；
- 緊急退出至安全狀態；
- 事件記錄與不可否認稽核。

若 AI 控制器失效，系統仍必須由硬體保護與既有平台韌體安全運作。

SECTION

5.1 物理過渡態不能被刪除

電壓、時脈、電流和溫度都需要有限時間改變。即使軟體只看到 E2 到 E3 的狀態切換，物理系統仍經歷：

> $V(t),\ f(t),\ I(t),\ T(t).$

因此，「中間態不可觀測」不代表中間態沒有電氣後果。電壓斜率、時脈相位、供電網路諧振、湧入電流與熱循環都可能在轉換期間造成風險。

DEO 的工程目標不是消滅過渡態，而是：

- 縮短不產生有效工作的轉換時間；
- 保證整個過渡路徑都在安全走廊內；
- 避免控制器在相鄰包絡間反覆震盪；

- 讓上層介面把轉換視為原子化操作。

SECTION

5.2 安全走廊

對目標頻率 f，最低安全電壓可寫為：

$$
V
\geq
V_{min}(f,T,\pi)
+
\Delta V_{guard},
$$

其中 π 代表製程、晶片個體與老化狀態。安全走廊還需限制：

$$
\left|\frac{dV}{dt}\right|
\leq S_V,
$$

$$
\left|\frac{df}{dt}\right|
\leq S_f,
$$

$$
\Delta V_{droop}
\leq \Delta V_{allow}.
$$

S_V 和 S_f 不是越大越好；它們必須由調節器、PLL、時脈樹、負載瞬變與驗證結果共同決定。

SECTION

5.3 電源緩衝的正確角色

原稿提出每個能級配備大型預充超級電容，並以開關直接切換核心電壓。這不是一般處理器 DVFS 的合理架構。大型超級電容適合較高電壓匯流排或系統級能量緩衝，不適合當作多組低壓核心電源直接切換。

正確分層應是：

- 晶片內與封裝去耦承擔最快瞬態；

- 近負載調節器恢復電壓；
- 板級電容與多相 VRM 支援較慢電流變化；
- 系統級電容或超級電容緩衝更長的功率突波；
- DPCPS 協調各層目標阻抗與能量裕度。

因此 DEO 只向 DPCPS 請求「可用爆發功率與轉換準備」，不直接切換電容到核心。

SECTION

5.4 時脈切換

多 PLL 預鎖定可以是某些設計選項，但不是固定五個包絡就必須放置五個 PLL。可行方法包括：

- 一個或多個 PLL 配合整數／分數分頻器；
- 數位頻率合成器；
- 每時脈域獨立 PLL；
- glitch-free clock multiplexer；
- 硬體自主管理的倍頻與時序校準。

選擇取決於面積、功耗、抖動、鎖定時間與時脈域數。DEO 規格應描述可接受的轉換延遲和抖動，不應預先指定所有實作都要五 PLL。

SECTION

5.5 遲滯、最短駐留與反震盪

若負載在門檻附近波動，控制器可能在 E1、E2 或 E3 間頻繁切換。可加入：

- 進入與退出門檻不同的遲滯；
- 最短駐留時間；
- 切換成本懲罰；

- 預測信心門檻；
- 冷卻或重入時間。

  例如：

> $$q_{t+1}=E3 \text{ only if } \hat{d}_t>\theta_{up} \wedge c_t>c_{min},$$

而退出條件為：

> $$q_{t+1}=E2 \text{ if } \hat{d}_t<\theta_{down} \vee B_{th}>B_{max},$$

其中：

> $$\theta_{down}<\theta_{up}.$$

SECTION

6.1 三類預測來源

6.1.1 硬體事件預測

使用程式計數器、IPC、cache miss、memory stall、branch miss 與執行單元利用率，辨識短期工作負載階段。此方法不需要理解使用者正在「玩哪一款遊戲」，但可能受程式相位快速變化影響。

6.1.2 排程與應用提示

作業系統知道 runnable queue、deadline、優先度和即將喚醒的任務；應用程式則可能知道下一幀、下一批推論、下一個關鍵幀或下一段編譯工作。明確提示通常比黑箱猜測更可靠。

可設計抽象 API：

```
struct deo_hint {
  uint32_t latency_class;
  uint32_t expected_duration_us;
  uint32_t parallelism_class;
  uint32_t memory_intensity;
  uint32_t confidence;
};
```

應用程式不應指定電壓或解鎖 E4，只能描述工作意圖。

6.1.3 歷史與模型預測

對重複工作、伺服器請求、影片幀、模型層或編譯階段，可用歷史資料預測下一階段。模型必須定期校準，並對未知應用輸出不確定度。

SECTION

6.2 信心感知控制

令預測器輸出需求分布：

> $p(d_{t:t+H} \mid o_{0:t})$

及信心 c⊠。控制策略應遵循：

- 高信心且高價值事件：允許提前切換；
- 中信心：可預備冷卻或功率，但延後真正升級；
- 低信心：回退到反應式控制；
- 域外輸入：禁止使用 E4，自動降低控制權。

SECTION

6.3 預測錯誤的成本

預測錯誤不是只有「多花一點電」。它可能造成：

- 不必要的切換損耗；
- 其他核心或 GPU 失去功率預算；
- 產生熱點，讓真正負載到來時反而沒有爆發餘裕；
- 增加熱循環與可靠度壓力；
- 引入尾端延遲抖動。

因此應定義錯誤成本：

> C_(mis)
> =
> C_(energy)
> +C_(thermal)
> +C_(contention)
> +C_(aging)
> +C_(latency).

SECTION

6.4 與 oracle 的 regret

評估預測控制時，可建立離線 oracle：它知道完整未來負載，能選擇最佳包絡。實際控制器的 regret 為：

> cal-R_T
> =
> Σ_(t=1)^(T)
> [
> J(q⊠)-J(q⊠^(oracle))
>].

DEO 不必達到 oracle，但應證明在不同工作負載下，其 regret 低於純反應式或固定包絡基線。

SECTION

7.1 熱預算不是固定秒數

爆發能力取決於：

- 初始晶片與冷卻介質溫度；
- 封裝與散熱器熱容；
- 總熱阻；
- 空氣或液體入口溫度；
- 負載在晶片上的空間分布；
- 風扇、泵、冷水機和熱虹吸狀態；
- 是否有其他元件共享冷卻能力。

因此 E3／E4 的最大時間應由模型與感測動態決定：

> τ_(max)
> =
> sup{
> τ:
> T⊠(t)≤ T_(safe),
> ∀ t∈[t⊠,t⊠+τ]
> }.

HotSpot 類等效熱阻—熱容模型可以在架構研究中估計溫度動態，但最終控制器仍需以實際封裝與冷卻平台校準。[8]

SECTION

7.2 功率與電流預算

DEO 應同時檢查：

$$
P_{pkg} \le P_{limit},;
$$

$$
I_{rail} \le I_{limit},;
$$

$$
\Delta V_{droop} \le \Delta V_{allow},;
$$

$$
T_{vrm} \le T_{vrm,max}.
$$

即使冷卻能力足夠，電源網路也可能無法安全支援峰值電流；反之，PSU 額定瓦數充足也不代表晶片與封裝局部 PDN 安全。DEO 必須使用 DPCPS 提供的可用功率證書，而不是只讀取 PSU 標稱功率。

SECTION

7.3 可靠度帳本

高電壓、高溫、高電流密度與頻繁熱循環都可能影響壽命。單一 Arrhenius 公式不能完整描述 NBTI、HCI、TDDB、電遷移、焊點疲勞與封裝材料老化，因此本文不給出「某溫度等於某年壽命」的簡化結論。

可建立抽象累積壓力：

$$
D_{age}(t+\Delta t) = D_{age}(t) + \Delta t \, \Phi(V,T,J,\nabla T,\Delta T_{cycle},q),
$$

其中 Φ 必須由製程與封裝可靠度模型校準。空間熱分布也不能只以平均溫度取代；近期溫度感知電遷移分析顯示，即使平均溫度相同，不同熱分布也可能造成顯著不同的壽命估計。[9]

可靠度帳本可用於：

- 限制每日／每任務的 E4 配額；
- 對老化晶片收緊包絡；
- 在長時間任務中優先選擇 E2；
- 對高熱梯度核心進行任務遷移；

- 記錄超規操作供保固與研究分析。

SECTION

7.4 爆發不是免費能量

E3／E4 只是在時間上重新分配功率、熱與可靠度預算。它不能創造能量，也不能保證總能耗下降：

> E_(task)
> =
> ∫_(t⌧)^(t⌧)P_(sys)(t) dt.

高頻縮短執行時間可能降低或提高 E_(task)，取決於電壓提升、漏電、記憶體等待、冷卻功率與任務是否能隨頻率加速。每個包絡都必須用實際工作量與完整系統能耗評估。

SECTION

8.1 O-Chip：提供意圖與預測，不擁有安全權限

O-Chip 可提供：

- 程式相位分類；
- 依賴圖與關鍵路徑估計；
- 應用／排程提示；
- 短期需求與信心；
- 任務價值與 deadline。

但 O-Chip 不應直接解除電壓、溫度或 E4 限制。DEO 接收它的建議，再經安全可行集合篩選。

SECTION

8.2 SynCore：把包絡擴展到架構模式

對 SynCore v2.0，包絡的 cal-M⌧ 可包含：

- 模式 A：神核融合；
- 模式 B：流動計算；
- 模式 C：混合模式；
- 傳統分散多核模式；
- Q-Storage 或上下文保留狀態。

  此時切換成本不只是頻率延遲，還包含：

> C_(switch)
> =
> C_(state)
> +C_(cache)
> +C_(fabric)
> +C_(thermal)
> +C_(power).

DEO 的主要角色是判斷：某段工作值得承擔這些切換成本嗎？

SECTION

8.3 DPCPS：提供可用功率證書

DPCPS 可回報：

- 當前可用包絡功率；
- PDN 電壓偏差與目標阻抗裕度；
- 可啟用調節器與相數；
- VRM 溫度與電流限制；
- 局部能量緩衝狀態；
- 預估可維持的負載突波。

DEO 不應在 DPCPS 回報無足夠裕度時進入 E3／E4。

SECTION

8.4 DryCore：提供可用熱證書

DryCore 可回報：

- 晶片與冷端溫度；
- 有效熱阻估計；
- 冷卻介質入口條件；
- 風扇／泵／熱虹吸狀態；
- 露點與冷凝安全；
- 可用持續熱移除能力；
- 暫態熱容量與估計爆發時間。

  DEO 可以提前提高風量或冷卻能力，但必須計入額外冷卻能耗與噪音。

SECTION

8.5 統一閉環

完整控制鏈為：

> 工作負載／提示
> →
> O-Chip 預測
> →
> DEO 包絡決策
> →
>
> SynCore 模式
> DPCPS 功率配置
> DryCore 熱配置

DEO 是協調層，不取代各子系統的本地快速控制。

SECTION

9.1 MVP 目標

第一階段不需要製造 SynCore，也不能直接控制商用 CPU 的每個電壓轉換細節。MVP 的問題應縮小為：

> 在現有 Linux CPU 上，語義包絡＋預測提示是否能在相同安全與 QoS 約束下，優於既有 governor 或硬體自主管理的預設策略？

SECTION

9.2 軟體架構

可建立：

- DEO daemon：收集效能計數器、排程、溫度、功率與應用提示；
- 包絡設定檔：將 E0–E3 映射到平台允許的效能上下限、EPP、功率上限、核心集合與 PM QoS；
- 預測器：先使用規則、EWMA 或輕量模型；
- 安全層：禁止超出 OEM 與韌體允許範圍，不嘗試改寫未公開電壓；
- 追蹤器：記錄每次決策、切換、溫度、功率與性能結果。

E4 不應在一般商用平台 MVP 中啟用；它需要專用硬體、外部量測與風險承擔。

SECTION

9.3 包絡映射示例

以下只是介面示意，不是通用參數：

包絡　CPU 效能提示　核心策略　功率／熱策略

E0　最低可接受效能＋深 idle　保留少量核心　嚴格延遲 QoS 與喚醒策略

E1	偏效率 EPP	減少活躍核心或批次背景任務	低功率上限
E2	平衡／持續效能	依工作負載配置	穩態熱限制
E3	高效能提示	關鍵任務綁定高性能核心	短時較高功率＋熱預算

實際可用介面依 CPU、韌體、核心版本與權限不同。AMD P-State 透過 CPPC 提供相對效能提示；Intel P-State 可透過 HWP 模式與 EPP 影響硬體選擇；Linux CPUFreq 也提供 schedutil 等 governor。[3][4]

SECTION

9.4 工作負載集合

為避免只挑有利案例，至少包括：

- 互動式短工作：網頁、程式啟動、壓縮與小型編譯；
- 序列 CPU 工作：單執行緒編譯、解譯器或模擬；
- 多執行緒工作：LLVM／Linux kernel 編譯、Blender、FFmpeg；
- 記憶體受限工作；
- 混合 CPU／GPU 工作；
- 延遲敏感服務；
- 長時間穩態負載；
- 隨機與不可預測負載；
- 多任務競爭場景。

  應優先使用可公開重現的 benchmark、版本、輸入資料與腳本。

SECTION

9.5 基線

至少比較：

- 平台預設硬體自主管理；

- Linux schedutil；
- performance governor；
- powersave／EPP 偏效率；
- 固定功率上限；
- DEO 無預測版本；
- DEO 有預測版本；
- 離線 oracle。

若只與最差固定頻率比較，不能證明 DEO 優於現代控制器。

SECTION

9.6 量測指標

性能：

- 任務完成時間；
- 吞吐量；
- 平均、p95、p99 延遲；
- 幀時間分布與 jitter；
- deadline miss ratio。

能耗與功率：

- CPU／封裝／整機能量；
- 峰值與平均功率；
- energy per task；
- EDP、ED^2P；

- 冷卻功率。

控制品質：

- 狀態切換數量；
- 包絡駐留分布；
- 預測 precision、recall、calibration；
- 相對 oracle 的 regret；
- 控制器 CPU 與記憶體開銷。

熱與安全：

- 峰值溫度；
- 溫度梯度與循環幅度；
- 熱／功率 throttling 時間；
- 電壓與電流限制事件；
- 感測失效時的回退行為。

SECTION

9.7 消融實驗

需要逐項移除：

- 工作負載預測；
- 應用提示；
- 遲滯；
- 熱模型；
- 可靠度配額；

- 多域功率重分配；
- 包絡語義，只保留原始細粒度控制。

  如此才能判斷效益到底來自哪一層。

SECTION

10.1 V0：追蹤回放與數位雙生

收集真實工作負載的：

- 效能計數器；
- 任務排程；
- 頻率與效能狀態；
- 功率與溫度；
- 應用階段。

在離線環境回放，測試不同包絡與預測策略。熱模型可先使用校準 RC 模型或 HotSpot 類工具。

SECTION

10.2 V1：現有 CPU 軟體原型

利用官方允許介面調整效能提示、功率上限與排程。目標不是改寫電壓，而是驗證控制策略。

SECTION

10.3 V2：FPGA／RISC-V 硬體在迴路

建立可程式化時脈域、外部 PMIC、電子負載與熱模型，驗證：

- 轉換守衛；
- 電壓—頻率定序；

- 感測延遲；
- 控制器逾時；
- 錯誤預測；
- 電源下陷；
- 熱模型偏差。

ControlPULP 類開源片上功率控制平台顯示，平行可程式化控制器可在很小面積開銷下執行多輸入多輸出的即時功率管理策略，並適合作為 DEO 進階原型的參考。[10]

SECTION

10.4 V3：多域原型

加入 CPU、記憶體、加速器、DPCPS 與 DryCore 模型，驗證功率與冷卻跨域競爭。

SECTION

10.5 V4：SynCore 模式模擬

在架構模擬器或 FPGA 上建立：

- 神核融合切換成本；
- 流動模式切換成本；
- 快取與上下文轉移；
- TMC 與 DEO 的權限分工；
- 不同模式的包絡認證。

SECTION

10.6 V5：專用矽與獨立重現

只有在 V0–V4 顯示可重複效益後，才值得進行專用矽或封裝原型。公開主張至少要附：

- 晶片與平台版本；
- 韌體與 kernel；
- 工作負載與輸入；
- 感測器校準；
- 完整 baseline；
- 原始資料與統計；
- 失敗案例；
- 第三方重現。

SECTION

11.1 H1：語義包絡降低控制震盪

在相同 QoS 與安全限制下，包絡＋遲滯控制相較於不帶切換成本的細粒度策略，應降低無效狀態切換與由此產生的能耗、抖動或熱循環。

否證條件：切換數量下降但性能或能耗顯著惡化，或現有硬體控制器本身已做到更好。

SECTION

11.2 H2：預測切換改善階段邊界的尾端延遲

對具有可預測階段的工作負載，DEO 應在不增加超過預設能量與熱預算的情況下，改善 p95／p99 延遲或 deadline miss ratio。

否證條件：相對 HWP／CPPC／schedutil 無顯著改善，或預測成本與錯誤抵消收益。

SECTION

11.3 H3：動態爆發預算優於固定週期

基於即時熱狀態、冷卻能力與任務價值的 E3 預算，應優於固定「爆發十秒、冷卻五秒」。

否證條件：簡單固定策略在跨環境、跨工作負載下同樣或更穩定。

SECTION

11.4 H4：跨域包絡優於只控制 CPU 頻率

當工作負載瓶頸在 CPU、GPU、記憶體與 I/O 之間轉換時，跨域功率重分配應比只升 CPU 頻率有更好的有效工作量／瓦特。

否證條件：跨域協同的遙測、延遲或控制複雜度使效益消失。

SECTION

11.5 H5：可靠度配額能避免峰值狀態被濫用

在長期壓力模擬與硬體在迴路測試中，可靠度帳本應限制高壓力包絡的累積暴露，而不造成不可接受的性能損失。

否證條件：可靠度模型不具可校準性，或配額與真實老化沒有相關性。

SECTION

11.6 H6：五類包絡不是最佳答案

本文主動提出反命題：固定五類可能過粗或過多。實驗應比較：

> $K \in \{2,3,4,5,8,\text{adaptive}\}$.

若自適應或較細粒度策略全面優於五類，DEO 應保留「包絡認證」而放棄固定五類介面。

SECTION

12.1 與現有硬體控制器衝突

HWP、CPPC、SMU、EC、BIOS、作業系統 governor 和廠商韌體可能同時控制功率與頻率。DEO 若沒有明確權限模型，可能出現控制迴路互相追逐。

緩解：上層只設定允許範圍與意圖，讓本地硬體閉環維持快速控制；不得繞過 OEM 安全保護。

SECTION

12.2 平台不可觀測性

商用 CPU 不公開完整電壓、PVT、老化與內部功率決策。軟體 MVP 只能驗證策略層，不能證明轉換電路本身。

SECTION

12.3 預測器域外失效

新程式、加密工作、隨機負載或對抗性輸入可能讓模型失準。

緩解：不確定度、域外偵測、反應式回退、E4 禁止與保守安全集合。

SECTION

12.4 安全與資安

若應用程式可以發送高效能提示，惡意程式可能長期占用爆發資源、製造熱循環或造成其他租戶性能下降。

緩解：權限、配額、租戶隔離、簽章提示、核心仲裁與不可變事件記錄。

SECTION

12.5 公平與服務品質

單一高價值工作進入 E3 可能讓其他任務失去功率或熱預算。控制器必須把公平、deadline 和服務水準納入目標函數。

SECTION

12.6 超頻責任

E4 不代表廠商保證的安全超頻。任何超出額定規格的操作都必須清楚標示風險，並與一般 DEO 包絡分離。公開版研究不應鼓勵繞過電壓、電流或溫度保護。

SECTION

12.7 控制器本身的開銷

若預測模型耗用大量 CPU、記憶體或功率，其效益可能自我抵消。輕量 DVFS 感測模型研究已顯示，少量效能計數器與低複雜度模型可達到很低的執行開銷；DEO 應以此作為設計方向，而非預設大型神經網路。[11]

SECTION

13.1 證據等級

等級	定義	可使用的措辭
E0	概念與形式模型	提出、假設、預期、可檢驗
E1	離線模擬／追蹤回放	在指定模型與資料下觀察到
E2	現有平台軟體原型	在指定硬體與官方介面上量測到
E3	硬體在迴路／FPGA／PMIC 原型	在原型條件下驗證
E4	專用硬體整合	在工程樣機上重現
E5	第三方重現與公開資料	獨立重現、可外部審查

目前本文整體為 E0。任何未來性能數字都必須標示對應等級。

SECTION

13.2 禁止混用的措辭

不得再使用：

- 「實測數據（推演）」；
- 「已證明」但只有估算；
- 「一定提升」但沒有基線與置信區間；
- 「量子跳躍」暗示宏觀量子機制；

- 「零延遲」「零風險」「無限維持」；
- 以虛構 CPU、頻率、跑分或遊戲幀率呈現為產品結果。

SECTION

13.3 最低公開資料

每一項性能主張至少包含：

- 處理器、主機板、記憶體、冷卻與電源；
- BIOS、微碼、作業系統、kernel 與 governor；
- 環境溫度與量測方法；
- 工作負載版本、輸入與執行腳本；
- 重複次數、統計方法與誤差；
- baseline 與消融；
- 功率、溫度、頻率與包絡時間序列；
- 失敗、失準與回退案例。

原稿最深的問題，不是它使用了動漫或量子類比，而是把類比反過來當成物理證明。v2.0 提出一個較穩健的命題：

> 物理系統可以連續演化，控制介面仍可離散組織。

人類也常用少量語義狀態管理連續世界：飛機有起飛、爬升、巡航、進場與降落，但速度、高度、推力和溫度仍連續變化；資料庫交易對上層是提交或回滾，底層卻包含大量中間寫入；作業系統的 process state 是離散的，電晶體與電流卻在連續時間中運作。

因此，DEO 不需要宣稱「連續控制是一種背叛」。更精確的說法是：

> 當底層狀態空間過於龐大時，系統需要一組可理解、可驗證且可安全切換的宏觀包絡；但這些包絡必須尊重連續物理過程，而不是否定它。

原稿所說「真正的範式轉換不是參數優化，而是認知重構」仍可保留，但應改寫為：

> 真正的重構，不是把十七個頻率檔位粗暴刪成五個，而是把頻率、電壓、架構模式、功率、熱與壽命重新視為同一個受約束的動態決策。

DEO-SynCore 的最終價值不在「跳得多快」，而在於三件事：

- 系統是否知道為何切換；
- 切換是否在完整安全走廊內；
- 切換所換得的工作量，是否值得消耗的能量、熱與壽命。

如果這三個問題能被量測、驗證與重現，DEO 才是一個工程架構；否則它仍只是一個有吸引力的比喻。

本文將 DEO-SynCore 從固定五檔、量子跳躍與極限超頻的概念產品，重構為預測式離散運作包絡控制。新版承認現有 P-state、HWP、CPPC 和 CPUFreq 已具備細粒度效能管理，並把研究重點轉向多維包絡、跨層預測、安全轉換、爆發預算、可靠度帳本與可否證驗證。

DEO 不要求物理世界離散，也不要求所有硬體使用五個頻率。它要求平台把底層複雜狀態壓縮成少量經認證的運作意圖，並允許硬體在包絡內自主調節、在包絡間受守衛切換。O-Chip 提供工作負載意圖與預測，SynCore 提供可重組運算模式，DPCPS 提供功率證書，DryCore 提供熱證書，而獨立安全層保留最終否決權。

下一個實際里程碑不是宣稱 6.5GHz 或某個遊戲幀率，而是在現有 Linux 平台完成可重現的 E0–E3 軟體 MVP，證明或否證：語義包絡與預測切換是否真的能在同等安全與服務品質下，優於現有硬體自主管理。只有通過這一步，DEO 才值得進入 FPGA、硬體在迴路與 SynCore 整合。

[1] Intel, Intel 64 and IA-32 Architectures Software Developer's Manual, Power and Thermal Management, Enhanced Intel SpeedStep and Hardware-Controlled Performance States sections.

https://www.intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html

[2] UEFI Forum, Advanced Configuration and Power Interface Specification 6.6, Collaborative Processor Performance Control.

https://uefi.org/specs/ACPI/6.6/

[3] Linux Kernel Documentation, CPU Performance Scaling and intel_pstate CPU Performance Scaling Driver.

https://docs.kernel.org/admin-guide/pm/cpufreq.html

https://docs.kernel.org/admin-guide/pm/intel_pstate.html

[4] Linux Kernel Documentation / AMD, amd-pstate CPU Performance Scaling Driver.

https://docs.kernel.org/admin-guide/pm/amd-pstate.html

[5] Linux Kernel Documentation, CPU Idle Time Management.

https://docs.kernel.org/admin-guide/pm/cpuidle.html

[6] S. Bharadwaj et al., “Predict; Do not React for Enabling Efficient Fine Grain DVFS in GPUs,” 2022.

https://arxiv.org/abs/2205.00121

[7] J. Haj-Yahya et al., “SysScale: Exploiting Multi-domain Dynamic Voltage and Frequency Scaling for Energy Efficient Mobile Processors,” 2020.

https://arxiv.org/abs/2005.07613

[8] K. Skadron et al., “HotSpot: a Dynamic Compact Thermal Model at the Processor-Architecture Level,” Microelectronics Journal, 2003.

https://www.cs.virginia.edu/~skadron/Papers/hotspot_mej.pdf

[9] H. Lu and S. X.-D. Tan, “EMSpice 3: Full-chip Temperature-Aware Multiphysics Electromigration and IR-Drop Analysis,” 2026.

https://arxiv.org/abs/2604.10743

[10] A. Ottaviano et al., “ControlPULP: A RISC-V On-Chip Parallel Power Controller for Many-Core HPC Processors with FPGA-Based Hardware-In-The-Loop Power and Thermal Emulation,” 2023.

https://arxiv.org/abs/2306.09501

[11] S. Mazzola et al., “A Data-Driven Approach to Lightweight DVFS-Aware Counter-Based Power Modeling for Heterogeneous Platforms,” 2023.

https://arxiv.org/abs/2305.06782

SECTION

附錄 A：最小 DEO 決策流程

```
input:
 telemetry x_t
```

workload hint h_t
predictor confidence c_t
certified envelopes E

1. validate sensors and timestamps
2. estimate current power, thermal and reliability state
3. predict workload demand over multiple horizons
4. remove envelopes that violate hard constraints
5. compute objective for remaining envelopes
6. apply hysteresis, minimum dwell and switching cost
7. request transition through platform-authorized interfaces
8. verify transition completion and constraint margins
9. log decision, outcome and prediction error
10. on anomaly or low confidence, return to safe reactive control

SECTION

附錄 B：包絡描述檔概念格式

```
platform: example-platform
revision: 0.1
envelopes:
  E1_efficiency:
    performance_min: 20
    performance_max: 55
    energy_preference: efficiency
    power_cap_w: platform_calibrated
    max_temperature_c: platform_calibrated
    minimum_dwell_ms: 20
  E2_sustained:
    performance_min: 45
    performance_max: 85
    energy_preference: balance
    power_cap_w: platform_calibrated
    max_temperature_c: platform_calibrated
    minimum_dwell_ms: 50
  E3_burst:
    performance_min: 75
    performance_max: 100
    energy_preference: performance
    power_cap_w: platform_calibrated
    thermal_budget_j: model_calibrated
    maximum_dwell_ms: model_calibrated
    reentry_guard: required
```

上述值只是欄位示例，不能直接用於任何真實處理器。