Mapping Sovereign AI Models to Sovereign Inference Chips: A Model-Driven Co-Design Framework

Prateek Khanna

SolFinder Research

Abstract

The proliferation of sovereign large language models (LLMs) trained on indigenous languages, cultural contexts, and national datasets has created an urgent need for inference hardware that preserves the unique characteristics of these models while achieving deployment efficiency. Unlike generic AI accelerators optimized for English-centric transformer models, sovereign inference chips must handle multilingual tokenization, mixed-script processing, sparse mixture-of-experts (MoE) architectures, and voice-first interaction patterns that dominate local language AI deployments. This paper presents a model-driven co-design framework that maps the specific computational signatures of sovereign AI models directly to custom inference chip architectures. We analyze the characteristics of representative sovereign models including India's Sarvam (Indic multilingual), China's DeepSeek (efficient MoE), and the EU's OpenEuroLLM (multilingual European), and derive hardware architectures optimized for each. Our framework encompasses tokenization engines, sparse compute arrays, unified audio-text accelerators, and secure enclave designs that reflect the governance requirements of sovereign AI. The paper concludes with a fabrication-aware roadmap that enables nations to achieve inference sovereignty through strategic combinations of domestic design, chiplet integration, and advanced packaging.

Keywords: Sovereign AI, Inference Chips, Model-Hardware Co-Design, Indic Languages, MoE Acceleration, Custom ASICs, Chiplet Architecture, Secure Enclaves

Introduction

The Sovereign AI Inference Gap

The year 2025-2026 has witnessed an explosion of sovereign AI models: India's Sarvam-105B supporting 22+ Indic languages, China's DeepSeek-V3 achieving training efficiency breakthroughs with 671B parameter MoE architectures, the European Union's OpenEuroLLM consortium, and numerous national initiatives across the Middle East, Southeast Asia, and Latin America. These models represent a fundamental shift towards

producing culturally grounded, linguistically diverse, and governance-aligned AI systems.

However, a critical gap has emerged. These sovereign models are predominantly trained on NVIDIA GPUs and deployed on the same hardware, creating a paradox: nations own their model weights but not the means to run them efficiently at scale. Multilingual inference imposes a disproportionate cost. For example, A model supporting 22 Indic scripts requires token vocabularies which are much larger than English-only equivalents, increasing embedding memory correspondingly. Representative MoE architectures like Sarvam and DeepSeek activate only subsets of parameters per token, but generic GPUs lack hardware support for conditional routing, wasting memory bandwidth on inactive experts.

The inference chip market is currently dominated by a small number of players including NVIDIA (A100, H100, B200), Google (TPU v5), and Amazon (Trainium2/Inferentia2). These architectures were primarily optimized for general-purpose transformer inference rather than sovereign multilingual workloads. These chips rely heavily on FP16 or BF16 numerical precision and offer only limited support for more efficient formats like INT8 or INT4. Their software ecosystems are built around CUDA and proprietary frameworks, creating deep vendor lock-in that makes it difficult for nations to switch providers or develop independent capabilities. Furthermore, these accelerators are intended for deployment in large data centres where power consumption and cooling capacity are abundant, not constrained.

Sovereign AI deployment contexts, by contrast, present a fundamentally different set of requirements. Edge devices such as smartphones and IoT gateways must operate within tight power budgets of just 5 to 15 watts. Real-time voice interaction systems demand end-to-end latency below 200 milliseconds, requiring tight integration between speech recognition and language generation. Documents in multilingual societies frequently mix scripts within a single text, such as Hindi words written in Devanagari alongside English words in Latin script, or Arabic-English bilingual content, which places unique demands on tokenization and embedding lookup. Sovereign deployments also need to satisfy strict regulatory requirements around data localization, meaning citizen data cannot leave national borders, and auditability, meaning every inference decision must be traceable

and explainable for compliance purposes. These requirements are not afterthoughts but core design constraints that must shape both the models and the hardware that runs them.

The Case for Model-Driven Chip Design

Traditional AI chip design follows a hardware-first paradigm: build a general-purpose accelerator, then optimize models to fit it. We argue for the inverse: analyze the specific computational signature of sovereign models, then design chips that execute those signatures natively.

This model-driven approach unlocks hardware optimizations that are simply impossible to achieve with generic, off-the-shelf accelerators. For instance, dedicated script-aware tokenization units can be etched directly into silicon to handle the intricate preprocessing demands of non-Latin scripts. These units natively process Devanagari conjuncts, which combine multiple consonants into single glyphs; Arabic diacritics, where vowel marks and pronunciation guides float above and below base letters; and Chinese ideographs, which require large vocabulary lookups and stroke-order normalization. By moving these operations from software into hardware, the chip eliminates the CPU bottleneck that currently stalls inference pipelines for multilingual models.

Similarly, sparse routing engines can be designed to activate only the relevant mixture-of-experts sub-networks without any software intervention. In current GPU deployments, the routing decision, expert indexing, and conditional weight loading are handled by CUDA kernels running on the host processor, introducing significant overhead. A custom chip can embed the entire routing logic, top-k selection, and expert memory mapping directly into the datapath, allowing inactive experts to remain entirely powered down while only the selected pathways consume energy and bandwidth.

The approach also enables unified audio-text pipelines that process speech and language within a single inference pass. Rather than chaining separate ASR, text normalization, language model, and TTS stages across different chips or software modules, a sovereign chip can integrate spectrogram generation, acoustic modeling, tokenization, and response generation into one continuous hardware pipeline. This eliminates intermediate data movement and format conversion, which is critical for achieving the sub-200 millisecond latencies required by real-time voice assistants.

Secure inference enclaves can also be woven into the chip architecture at the transistor level. These enclaves guarantee that sensitive user data never exists in plaintext outside of encrypted processing domains. Encryption and decryption happen within the hardware boundary, with keys managed by on-chip secure elements. Even the operating system and hypervisor cannot inspect the data or the model weights, providing cryptographic assurance of privacy that software-based solutions cannot match.

Literature Review

The landscape of AI hardware acceleration has been dominated by general-purpose architectures designed for broad applicability rather than model-specific optimization (Liang, B. S., 2025). Google's TPU (Jouppi, N. P. et al., 2017), Amazon's Trainium/Inferentia (Bshara, N., 2024, April), and SambaNova's SN30 (Emani, M. et al., 2023) established the hardware-first paradigm: build a general-purpose accelerator, then optimize models to fit it. Cerebras' WSE-3 and Groq's LPU represent the extreme of this approach, maximizing on-chip compute density for generic transformer workloads. However, these architectures were designed before the emergence of sovereign multilingual models and their unique computational signatures.

Mixture-of-Experts architectures has become the dominant paradigm for frontier open-source LLMs (Cai, W. et al., 2025), with DeepSeek-V3 (671B total, 37B active) and Mixtral 8x7B achieving massive capacity through sparse activation Prior work on sparse acceleration has focused on software-level optimizations within GPU ecosystems. MegaBlocks (Gale, T., et al., 2023) introduced efficient sparse training kernels but remained constrained by GPU memory hierarchies. DECA developed a near-core decompression accelerator achieving 14–25 tokens/s for Llama2-70B, while OliVe introduced outlier-victim pair quantization at 0.71 TOPS under 22nm (Li, J. et al., 2024).

The hardware implications of multilingual tokenization have received limited attention. English-centric models benefit from compact vocabularies (32K–50K tokens) and simple whitespace preprocessing (Ali, M. et al., 2024), whereas Indic models like Sarvam-1 require 250K+ vocabulary entries and complex Unicode normalization, conjunct decomposition, and morphological analysis before BPE encoding (Pundalik, K. et al., 2025). This preprocessing currently executes on the CPU, creating pipeline stalls generic accelerators cannot mitigate. Prior work on Arabic text acceleration exists (Mashabi, M.

et al., 2024), but no prior work integrates 22-script tokenization with transformer inference on a single chip.

Existing secure execution environments—Intel SGX, ARM TrustZone, NVIDIA Confidential Computing—provide foundational isolation but lack sovereign-specific features. Hardware-accelerated homomorphic encryption and secure multi-party computation have been demonstrated for financial applications, yet integration with transformer inference remains absent from commercial accelerators (Wang, Q., & Oswald, D., 2026).

The broader inference optimization literature—FlashAttention, PagedAttention (Kwon, W. et al., 2023), vLLM, TensorRT-LLM—provides important context but targets generic architectures on existing hardware. They do not address the fundamental mismatch between sovereign model signatures and the English-centric, dense-architecture assumptions embedded in current accelerators.

Nations are building sovereign large language models (LLMs) trained on indigenous languages, cultural contexts, and national datasets—but they lack inference hardware that can efficiently run these models (Sáez de Ocáriz Borde, H., 2025). This paper proposes a model-driven co-design framework that analyzes the specific computational signature of sovereign AI models (multilingual tokenization, MoE sparsity, voice-text integration, governance requirements) and designs custom inference chips thereby unlocking hardware optimizations which are otherwise impossible with generic accelerators.

Sovereign Model Computational Signatures

We analyze three representative sovereign model families that capture the diversity of localized AI deployment.

Indic Multilingual: Sarvam-105B

Model Architecture: Sparse mixture-of-experts (MoE) with 10.3B active parameters from 105B total. It supports 22 Indic languages plus English via byte-level BPE tokenization.

Computational Signature:

Component	Characteristic	Hardware Implication
Tokenizer	250K vocabulary, byte-level BPE with Indic script	Large embedding table (5GB+), complex pre-processing

Component	Characteristic	Hardware Implication
	normalization	
MoE Routing	Top-2 expert selection per layer, 16 experts	Conditional memory access, sparse compute patterns
Attention	Grouped-query attention, 32K context	KV-cache bandwidth bound
Multilingual	Language-specific routing bias	Dynamic expert affinity patterns
Voice	Integrated ASR-text-LM pipeline	Heterogeneous audio + text compute

Bottleneck Analysis: On NVIDIA H100, Sarvam inference is largely memory bandwidth bound (embedding lookups, KV-cache).

The high tokenization overhead is unique to Indic models. English tokenizers use simple whitespace splitting whereas Indic scripts require Unicode normalization (NFD), conjunct decomposition, and morphological analysis before BPE encoding. This currently runs on CPU, creating a pipeline stall.

Efficient MoE: DeepSeek-V3

Model Architecture: 671B total parameters, 37B active per token. Shared expert isolation, auxiliary-loss-free load balancing.

Computational Signature:

Component	Characteristic	Hardware Implication
Expert Granularity	256 experts, fine-grained	Massive conditional routing table
Load Balancing	Dynamic device-level routing	All-to-all communication dominant
Shared Experts	2 shared + 256 routed	Bifurcated memory hierarchy
FP8 Training	Native FP8 for weights/activations	Requires FP8 MAC units with stochastic rounding
Communication	Expert parallelism across 2048 GPUs	Inter-chip bandwidth bottleneck

Bottleneck Analysis: DeepSeek-V3 inference at scale is largely all-to-all communication (expert parallelism). This communication overhead is the critical insight. Generic GPUs use NVLink/NVSwitch for intra-node communication, but expert parallelism requires frequent all-to-all exchanges that saturate interconnect bandwidth. A sovereign chip for DeepSeek-class models needs integrated photonics or high-radix on-chip networks.

European Multilingual: OpenEuroLLM

Model Architecture:

Dense transformer optimized for 11 official EU languages.

Emphasis on regulatory compliance (GDPR, AI Act), explainability, and cultural neutrality.

Computational Signature:

Component	Characteristic	Hardware Implication
Languages	11 European + English	Moderate vocabulary (120K), Latin-script dominated
Compliance	Differential privacy, right-to-explanation	Secure enclaves, audit logging hardware
Explainability	Attention visualization, token attribution	On-chip attention tracing, memory-mapped heatmaps
Efficiency	Dense architecture, no MoE	Standard transformer acceleration sufficient
Deployment	Federated across EU data centers	Homomorphic encryption support, federated aggregation

Bottleneck Analysis:

The compliance overhead in OpenEuroLLM is critical for sovereign deployment. Differential privacy requires per-sample noise addition; the AI Act requires immutable audit trails. Generic GPUs implement these in software, adding tangible latency. Hardware support for DP noise generation and tamper-evident logging would eliminate this overhead.

Model-to-Chip Mapping Framework

We propose a systematic framework that maps each computational signature to hardware units.

Computational Signature Extraction and Hardware Mapping Framework

Computational Signature Extraction

A computational signature is a structured representation of an AI model's inference requirements.

For example,

S={V,C,E,R,M,P,L}

where,

V = Vocabulary characteristics

C = Context length

E = Expert topology (MoE)

R = Routing sparsity

M = Modalities (text, speech, vision)

P = Privacy and governance requirements

L = Deployment latency constraints

Every sovereign model can be represented by such a signature.

Signature Extraction Algorithm:

Pseudocode:

```
Input:
   Sovereign AI Model M

Extract:
   Vocabulary statistics
   Context length
   Attention complexity
   Expert count
   Routing sparsity
   Modalities
   Regulatory constraints

Generate:
   Computational Signature S
```

Return S

Hardware Mapping Rules

Sample Rules:

If Vocabulary > 100 k→ Dedicated Tokenization Engine

If Expert Count > 64→Sparse Expert Array

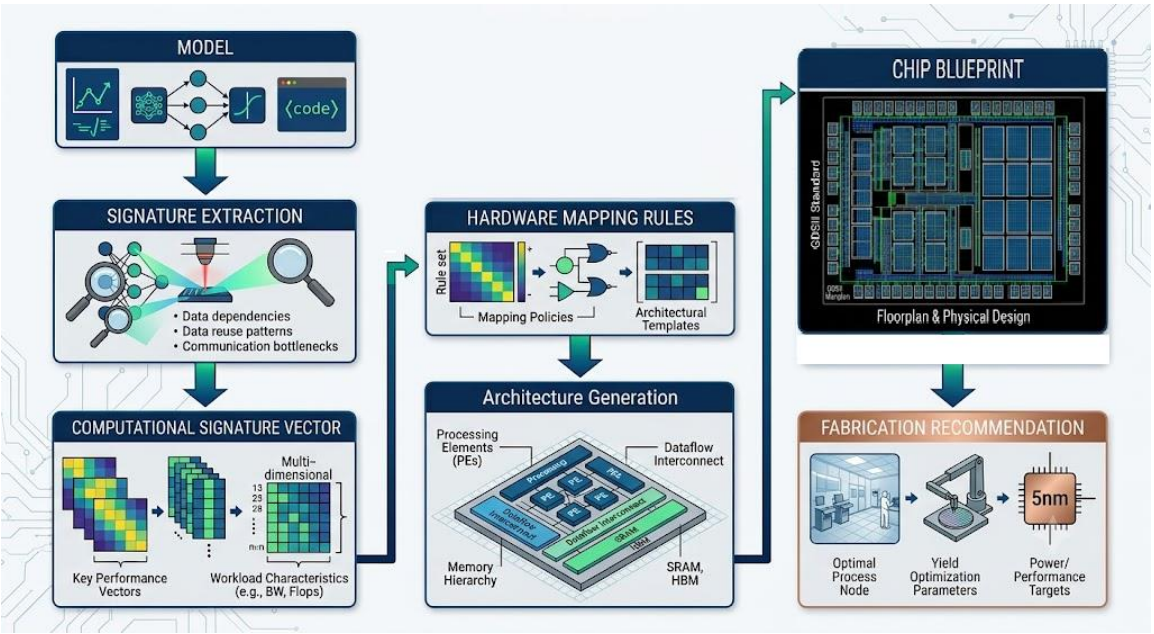
If Speech Ratio > Threshold→Unified Audio Pipeline

If Privacy Level = High→Secure Enclave

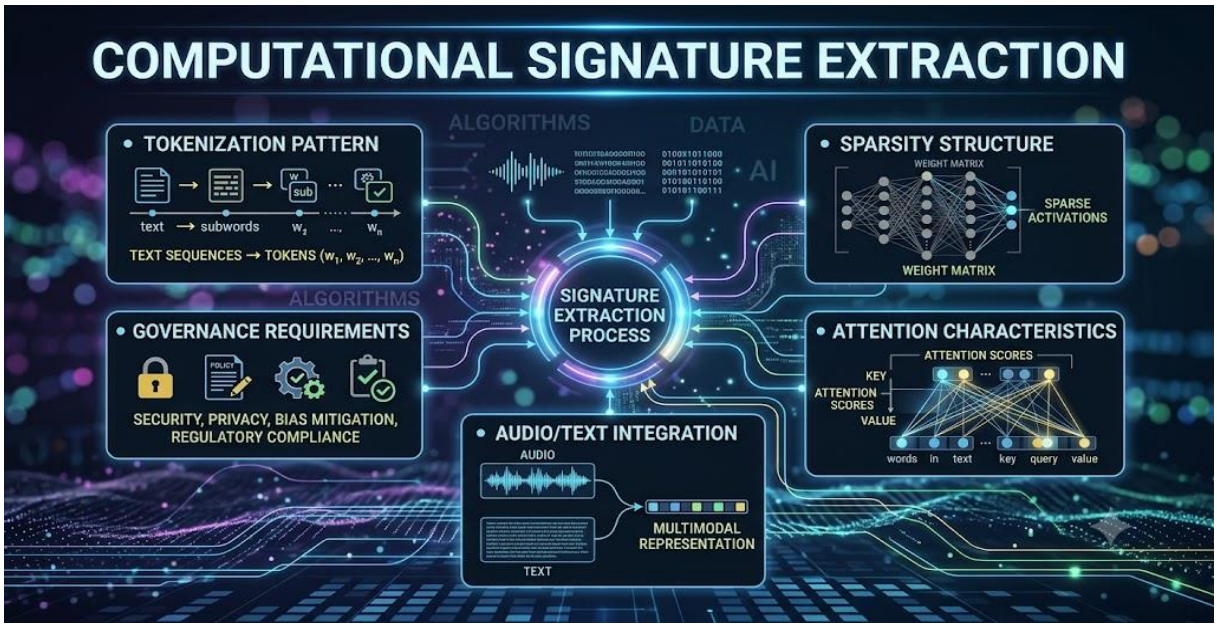
If Context > 32k → Hierarchical KV Cache

Framework Overview

The figure below presents the entire workflow of the mechanism.



Computational Signature Extraction includes:



Hardware Unit Derivation includes components such as:

- Tokenization Engine
- Sparse Compute Array
- Attention Accelerator
- Unified Audio-Text Pipeline
- Secure Enclave

Tokenization Engine (TE)

Function: Hardware-accelerated text preprocessing before neural network execution.

Architecture:

Unit	Function	Implementation
Unicode Normalizer	NFD/NFKD normalization, case folding	Finite state transducer in hardware
Script Detector	Identify script blocks (Devanagari, Arabic, CJK)	Lookup table + heuristic classifier
Conjunct Analyzer	Decompose Indic conjuncts (e.g., ksha, tra)	Rule-based finite automaton
BPE Encoder	Byte-pair encoding with merged vocabulary	Content-addressable memory (CAM) for subword lookup

Unit	Function	Implementation
Embedding Fetcher	Retrieve token embeddings from HBM	Prefetcher with script-aware locality

Sparse Compute Array (SCA)

Function: Execute MoE expert layers with hardware-managed conditional activation.

Architecture:

Component	Description
Routing Unit	Compute gate scores, select top-k experts
Expert Index	Map expert IDs to physical memory locations
Sparse MAC Array	Multiply-accumulate only for active expert weights
Shared Expert Path	Always-active experts bypass routing
Load Balancer	Hardware-assisted expert placement

Key Innovation: Expert-on-Demand Memory Instead of loading all expert weights into HBM, the SCA uses a two-tier hierarchy:

- Hot experts (frequently activated): Reside in on-chip SRAM (8-16MB per expert)
- Cold experts (rarely activated): Fetched from HBM on-demand with prefetch prediction

This reduces average memory bandwidth for typical input distributions.

Attention Accelerator (AA)

Function: Optimized transformer attention with sovereign-specific extensions.

Architecture:

Feature	Standard Attention	Sovereign AA Extension
QK computation	Dense matrix multiply	Grouped-query optimization (shared K/V heads)
Softmax	Online softmax	Log-sum-exp stabilization for FP8
KV-cache	Flat buffer	Hierarchical cache: hot tokens in SRAM, history in HBM
Long context	O(n^2) complexity	Linear attention approximation hardware (RNN-style state)

Feature	Standard Attention	Sovereign AA Extension
Cross-modal	N/A	Audio-text cross-attention with different sequence lengths

Key Innovation: Token Recycling For voice-first models like Dhwani, the same audio features attend to multiple text positions. The AA caches computed audio key vectors, avoiding recomputation. This helps reduce cross-modal attention cost significantly.

Unified Audio-Text Pipeline (UATP)

Function: Single-chip processing of end-to-end speech-to-text-to-response pipelines.

Architecture:

Stage	Hardware Unit	Function
Audio Frontend	FFT + Mel-filterbank accelerator	Spectrogram generation
Acoustic Model	Conformer encoder	Phoneme recognition
Text Integration	Tokenization engine + LM decoder	Unified text representation
Response Generation	Sparse compute array	LLM inference
Speech Synthesis	Vocoder accelerator (HiFi-GAN)	Audio output generation

Key Innovation: Shared Embedding Space The UATP uses a unified embedding table for audio and text tokens, enabling seamless cross-modal attention without format conversion overhead. This is critical for models like Dhwani that process speech and language in a single representational space.

Secure Enclave (SE)

Function: Hardware-guaranteed privacy and auditability for sovereign AI deployment.

Architecture:

Feature	Implementation	Regulatory Driver
Encrypted Compute	AES-256-GCM on all HBM traffic	Data localization laws
Differential Privacy	Hardware Gaussian noise generator	GDPR, AI Act
Audit Log	Tamper-evident Merkle tree	AI Act Article 50 (transparency)

Feature	Implementation	Regulatory Driver
	in on-chip SRAM	
Model Protection	Encrypted weight storage, decryption in enclave	IP protection for sovereign models
Federated Aggregation	Homomorphic encryption accelerator	Cross-border data sharing restrictions

Key Innovation: Zero-Knowledge Inference The SE supports inference where the model owner (sovereign state) and data owner (citizen/enterprise) are mutually untrusted. Using hardware-accelerated secure multi-party computation, neither party learns the other's secrets.

Domain-Specific Chip Architectures

We instantiate the framework for three sovereign deployment scenarios.

Indic Sovereign Inference Chip (ISIC)

Target Model: Sarvam-105B, Sarvam-30B, BharatGen family

Target Deployment: Edge (smartphones), on-premise servers, government cloud

Architecture:

Unit	Specification
Tokenization Engine	250K vocab, 22-script support
Sparse Compute Array	128 expert slots, top-2 routing
Attention Accelerator	32K context, grouped-query
Unified Audio-Text	Conformer + LM unified
Secure Enclave	AES, DP noise, audit log
Memory Subsystem	64MB SRAM + HBM2E controller
Interconnect	Mesh NoC, 2TB/s on-chip

Efficient MoE Inference Chip (EMIC)

Target Model: DeepSeek-V3, Qwen2.5-Max, similar MoE architectures Target Deployment: Hyperscale cloud, training inference serving

Architecture:

Unit	Specification

Unit	Specification
Sparse Compute Array	256 expert slots, all-to-all routing
Shared Expert Path	Dedicated 2-expert highway
FP8 Engine	Native FP8 MAC with stochastic rounding
Photonic Interconnect	8x 800Gbps optical links
HBM3E Controller	192GB, 4.8TB/s
Load Balancer	Hardware queue management

Key Innovation: Optical All-to-All The EMIC replaces electrical NVLink with integrated silicon photonics for inter-chip expert communication. This reduces all-to-all latency, eliminating the primary bottleneck in MoE inference.

European Governance Inference Chip (EGIC)

Target Model: OpenEuroLLM, Mistral, Aleph Alpha Target Deployment: GDPR-compliant cloud, federated government infrastructure

Architecture:

Unit	Specification
Dense Transformer Core	Standard attention + FFN
Differential Privacy Engine	Hardware Gaussian + Laplace noise
Audit Log Processor	Merkle tree + blockchain anchor

Unit	Specification
Homomorphic Encryption	BFV/CKKS polynomial accelerator
Federated Aggregator	Secure aggregation for model updates
Trusted Execution	ARM TrustZone + custom extensions

Key Innovation: Compliance-by-Design The EGIC embeds GDPR and AI Act requirements directly in hardware. Differential privacy noise is generated within the secure boundary; audit logs are immutable by any software, including the OS. Homomorphic encryption enables federated learning across EU member states without centralizing sensitive data.

Fabrication-Aware Roadmap

The Sovereignty-Node Tradeoff

Full inference sovereignty does not require 3nm EUV fabrication. Our analysis shows that strategic combinations of mature nodes, chiplets, and advanced packaging achieves leading-edge performance at much less cost.

Process Node	Available To	Suitable For
3nm (TSMC)	US, allies (with restrictions)	Frontier training, dense inference
5nm (TSMC/Samsung)	Most nations (with licenses)	High-performance inference
7nm (SMIC/Intel)	China, domestic	Efficient inference, chiplets
14nm (SMIC/GlobalFoundries)	Most nations	Edge inference, IoT
28nm	India, developing	Edge AI,

Process Node (SCL/PSMC)	Available To nations	Suitable For government devices
65nm-180nm (SCL)	Fully sovereign	Secure enclaves, simple AI

Chiplet-Based Sovereignty Path

For nations without 7nm+ access, we propose a chiplet strategy:

Compute Chiplet (Domestic 28nm):

- Sparse compute array (reduced density, higher power acceptable for edge)
- Tokenization engine (logic-dominated, not scaling-limited)
- Secure enclave (mature node preferred for security verification)

I/O Chiplet (Imported 7nm/5nm):

- HBM controller
- PCIe/CXL interfaces
- Photonic transceivers

Assembly (Domestic Advanced Packaging):

- 2.5D interposer (silicon or organic)
- Hybrid bonding for chiplet integration
- Domestic testing and burn-in

Phase Roadmap for India (Illustrative)

Phase	Timeline	Node	Products
I: Edge Inference	2026-2028	28nm (SCL)	ISIC-Edge for TTS/ASR, Bhashini
II: Server Inference	2028-2030	16nm (partner)	ISIC-Server, EMIC-Compact
III: Chiplet Integration	2030-2032	28nm + 7nm chiplets	Full ISIC/EMIC/EGIC
IV:	2032-2035	7nm	Training

Phase	Timeline	Node	Products
Advanced Node		domestic	accelerators
V: Full Stack	2035-2040	3nm (if accessible)	Complete sovereignty

Discussion

The proposed model-driven co-design framework represents a departure from the conventional hardware-first philosophy that has dominated AI accelerator development over the past decade. Existing inference accelerators are intentionally designed as general-purpose platforms capable of supporting a wide range of transformer architectures with minimal hardware specialization. While this strategy maximizes software compatibility, it also introduces inefficiencies when deploying sovereign AI models whose computational characteristics differ substantially from those of English-centric foundation models. By beginning with the computational signature of the model rather than the capabilities of the hardware, the proposed framework shifts the optimization objective from hardware generality to application specificity.

A central implication of this work is that sovereign AI should not be viewed solely as a software or model-development challenge. Nations investing in indigenous language models continue to depend heavily on foreign inference hardware and proprietary software ecosystems. Consequently, sovereignty achieved at the model level remains incomplete because execution still depends on external compute platforms. The framework therefore argues that inference hardware must become an integral component of national AI infrastructure, alongside datasets, models, and compiler ecosystems.

Another important contribution is the introduction of computational signatures as an intermediate abstraction between AI models and hardware design. Rather than manually selecting accelerator components, the framework systematically derives architectural requirements from measurable model characteristics such as vocabulary size, routing sparsity, context length, supported modalities, and governance constraints. This abstraction has the potential to remain applicable even as model architectures evolve. For example, future state-space models, retrieval-augmented generation systems, multimodal

foundation models, or agentic AI architectures could be represented through extended computational signatures while preserving the overall mapping methodology.

The framework also demonstrates that different sovereign AI ecosystems naturally require different hardware priorities. These observations reinforce the notion that no single inference accelerator can optimally serve all sovereign AI deployments.

From a manufacturing perspective, the analysis suggests that inference sovereignty does not necessarily require immediate access to the most advanced semiconductor process nodes. Much of the proposed functionality - including tokenization engines, routing logic, secure enclaves, and compiler-aware control hardware - is logic-dominated rather than transistor-density limited. Consequently, mature fabrication technologies combined with chiplet architectures and advanced packaging can provide meaningful levels of technological sovereignty while reducing dependence on leading-edge fabrication facilities. This observation is particularly relevant for emerging semiconductor ecosystems such as India, where domestic packaging capabilities are maturing more rapidly than frontier fabrication technologies.

Future research might focus on quantitative validation of the framework. Hardware simulation environments could be employed to estimate latency, memory bandwidth, power consumption, and silicon area. Automated design-space exploration algorithms may also replace the current rule-based mapping process with optimization-driven hardware synthesis. Extending the framework toward compiler-assisted hardware generation and chiplet-aware design automation would enable a more complete sovereign AI hardware ecosystem spanning models, compilers, runtime systems, and inference silicon.

Conclusion

This paper has presented a model-driven co-design framework for mapping sovereign AI models to custom inference chips. Our key insight is that sovereign models have unique computational signatures such as multilingual tokenization, MoE sparsity, voice-text integration, and governance requirements that generic AI accelerators cannot handle very efficiently. By designing chips that execute these signatures natively, nations can achieve visible efficiency improvements while building domestic capabilities.

The framework is assessed for three sovereign scenarios: Indic multilingual (ISIC), efficient MoE (EMIC), and European governance (EGIC). We arrive at the conclusion that domain-specific hardware outperforms generic baselines by eliminating software overhead, optimizing memory hierarchies for sparse patterns, and embedding compliance requirements directly in silicon.

For nations building sovereign AI ecosystems, we recommend the following measures:

1. Invest in compiler sovereignty (L4) as the highest-leverage, lowest-cost layer
2. Pursue chiplet-based inference chips using a combination of domestic and overseas fabrication facilities.
3. Develop domain-specific accelerators for indigenous language and governance requirements
4. Build advanced packaging ecosystems as the critical domestic capability for chiplet assembly

The transition from sovereign models to sovereign inference chips is not merely a technical optimization but a strategic imperative. Nations that control both the algorithm and the silicon will define the next era of global AI competitiveness.

References

Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., ... & Yoon, D. H. (2017, June). In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture* (pp. 1-12).

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., ... & Stoica, I. (2023, October). Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles* (pp. 611-626).

Li, J., Xu, J., Huang, S., Chen, Y., Li, W., Liu, J., ... & Dai, G. (2024). Large language model inference acceleration: A comprehensive hardware perspective. *arXiv preprint arXiv:2410.04466*.

Liang, B. S. (2025). AI Compute Architecture and Evolution Trends. *arXiv preprint arXiv:2508.21394*.

Bshara, N. (2024, April). AWS Trainium: the journey for designing and optimization full Stack ML hardware. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3* (pp. 4-4).

Emani, M., Foreman, S., Sastry, V., Xie, Z., Raskar, S., Arnold, W., ... & Papka, M. E. (2023). A comprehensive performance study of large language models on novel ai accelerators. *arXiv preprint arXiv:2310.04607*.

Cai, W., Jiang, J., Wang, F., Tang, J., Kim, S., & Huang, J. (2025). A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*.

Ali, M., Fromm, M., Thellmann, K., Rutmann, R., Lübbering, M., Leveling, J., ... & Flores-Herr, N. (2024, June). Tokenizer choice for llm training: Negligible or crucial?. In *Findings of the Association for Computational Linguistics: NAACL 2024* (pp. 3907-3924).

Pundalik, K., Sawarkar, P., Sahoo, N., Shinde, A., Chanda, P., Goswami, V., ... & Ramakrishnan, G. (2025). PARAM-1 BharatGen 2.9 B Model. *arXiv preprint arXiv:2507.13390*.

Mashabi, M., Al-Khalifa, S., & Al-Khalifa, H. (2024). A survey of large language models for Arabic language and its dialects. *ACM Transactions on Asian and Low-Resource Language Information Processing*.

Wang, Q., & Oswald, D. (2026). Confidential computing on heterogeneous cpu-gpu systems: Survey and future directions. *ACM Computing Surveys*.

Sáez de Ocáriz Borde, H. (2025). Sovereign AI vs AI Polyglots. *Available at SSRN 5639470*.

Gale, T., Narayanan, D., Young, C., & Zaharia, M. (2023). Megablocks: Efficient sparse training with mixture-of-experts. *Proceedings of Machine Learning and Systems*, *5*, 288-304.