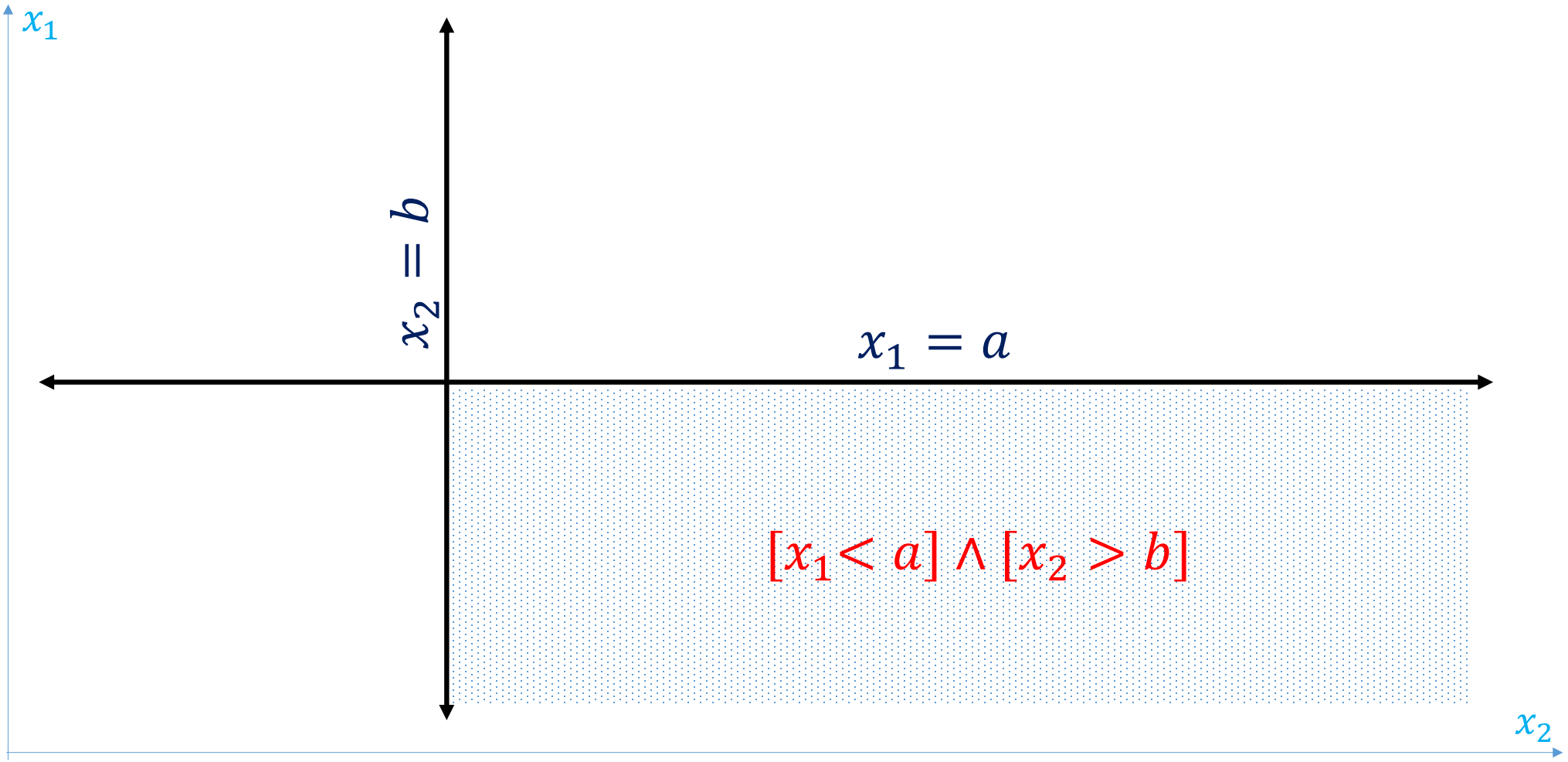
Hashing Tree

Prithwijit Guha
Dept. of EEE, IIT Guwahati

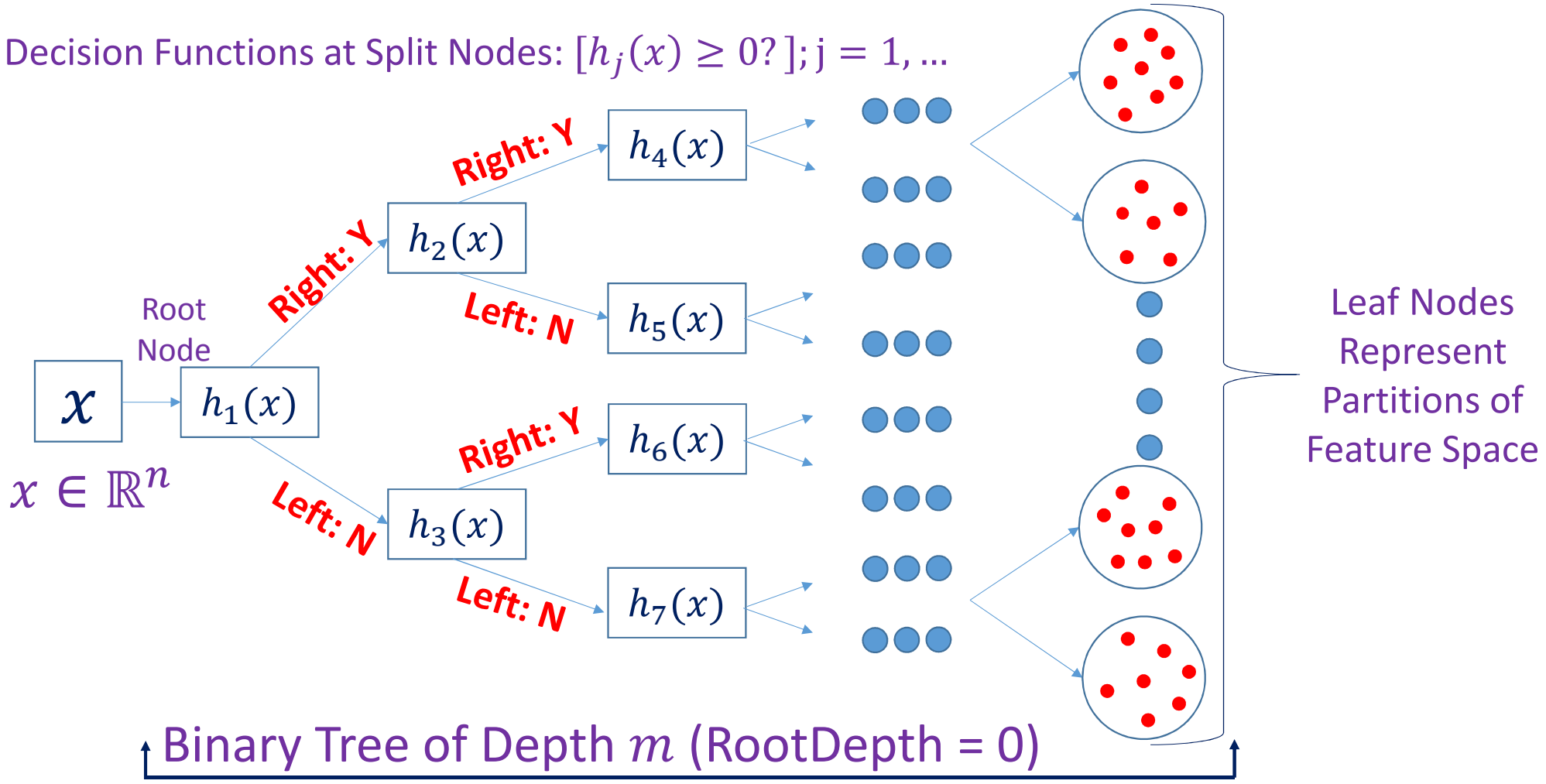
Hashing Techniques

- Dividing Input Space into Partitions
- Assign Integer IDs (Hashes) to each Partition
- Top-Down Hierarchical Partitioning
- Points in Same Partition has the Same Hash
- Proximal Points have High Probability of Similar Hash
- Partitions Generated by Combination of Hyperplanes
- Hyperplanes – Oblique or Axis-Aligned

Input Space Partitioning: Axis Aligned



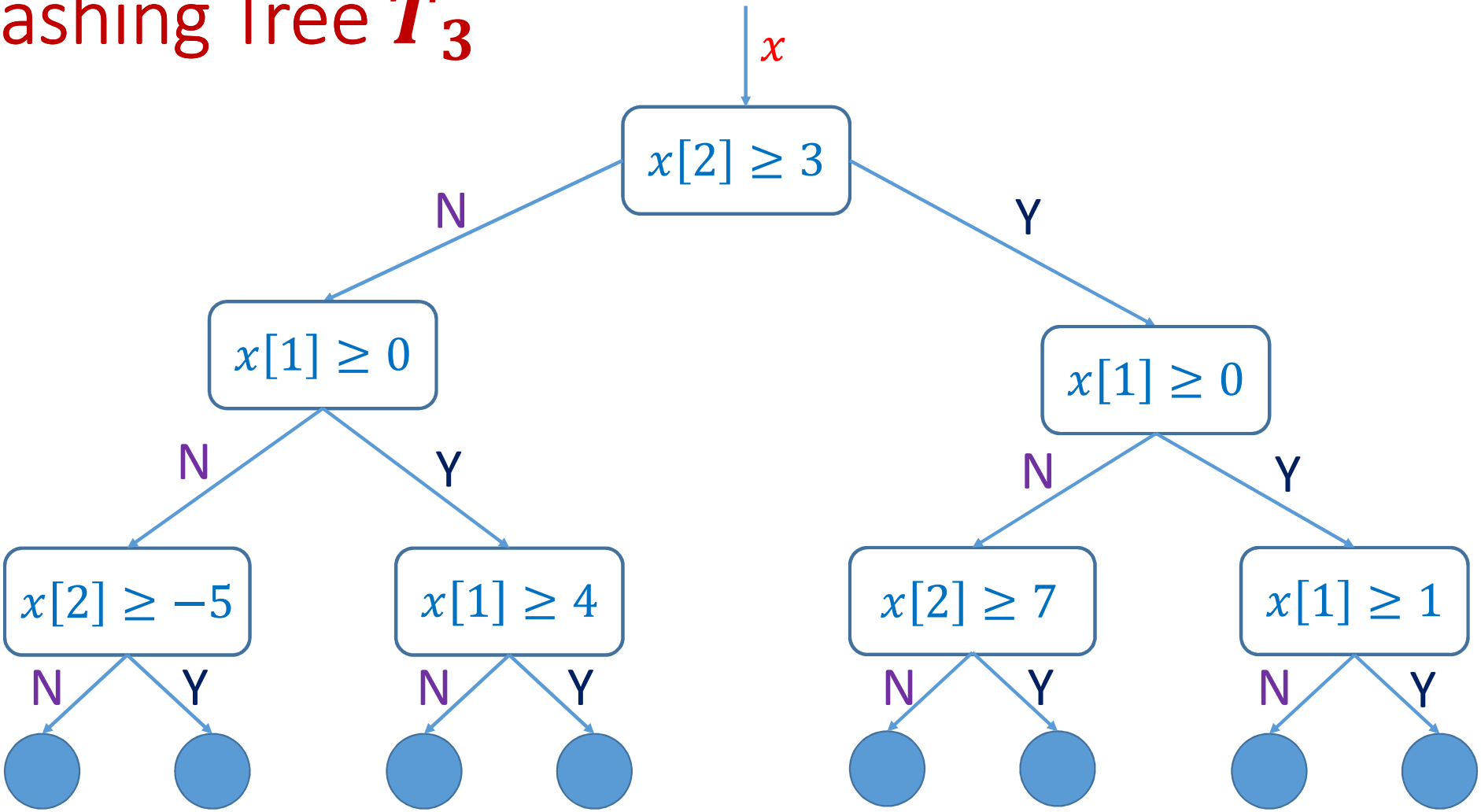
Top-Down Partitioning of Input Space



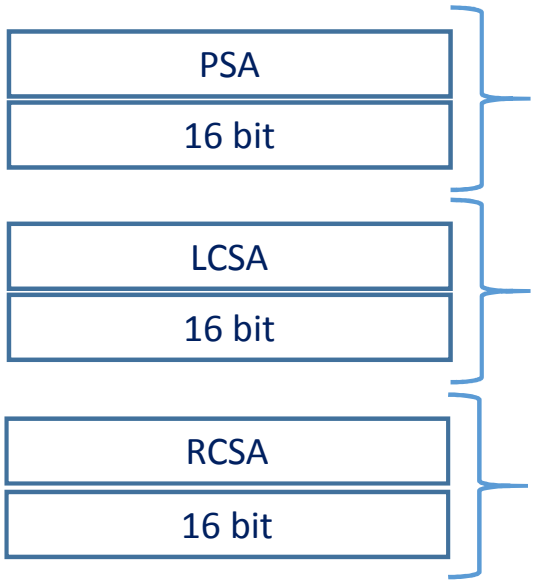
Tree Traversal for Hash Generation

1. Input to Tree of Depth m is $\boldsymbol{x}$ (Root Depth = 0; Leaf Depth = m)
2. Initialize: Hash Bit-string $\boldsymbol{H}$ of Length m is Set to ZEROs
3. IF $\boldsymbol{x}$ is Presented to SPLIT Node $\boldsymbol{q}$
   A. Node $\boldsymbol{q}$ is at Depth $\boldsymbol{j}$ with Decision Function $\boldsymbol{h_q(x)}$: $\boldsymbol{x[k_q] - \theta_q}$
   B. Node $\boldsymbol{q}$ has Children $\boldsymbol{q_L}$ (Left) and $\boldsymbol{q_R}$ (Right)
   C. IF $\boldsymbol{x[k_q] \geq \theta_q}$ SEND $\boldsymbol{x} \rightarrow q_R$; SET $\boldsymbol{H[j] = 1}$
   D. ELSE SEND $\boldsymbol{x} \rightarrow q_L$
   E. Proceed To Next Child Node $\boldsymbol{q_L}$ or $\boldsymbol{q_R}$
4. ELSE IF Data $\boldsymbol{x}$ is Presented to LEAF Node $\boldsymbol{q}$
   A. RETURN the Hash Bit String $\boldsymbol{H}$
   B. EXIT

Hashing Tree T_3



Node Data Block



Parent Node Starting Address
Set to ZEROs for ROOT NODE

Left Child Node Starting Address
Set to ZEROs for LEAF NODES

Right Child Node Starting Address
Set to ZEROs for LEAF NODES

DEPTH 8 bit → Storing NODE Depth

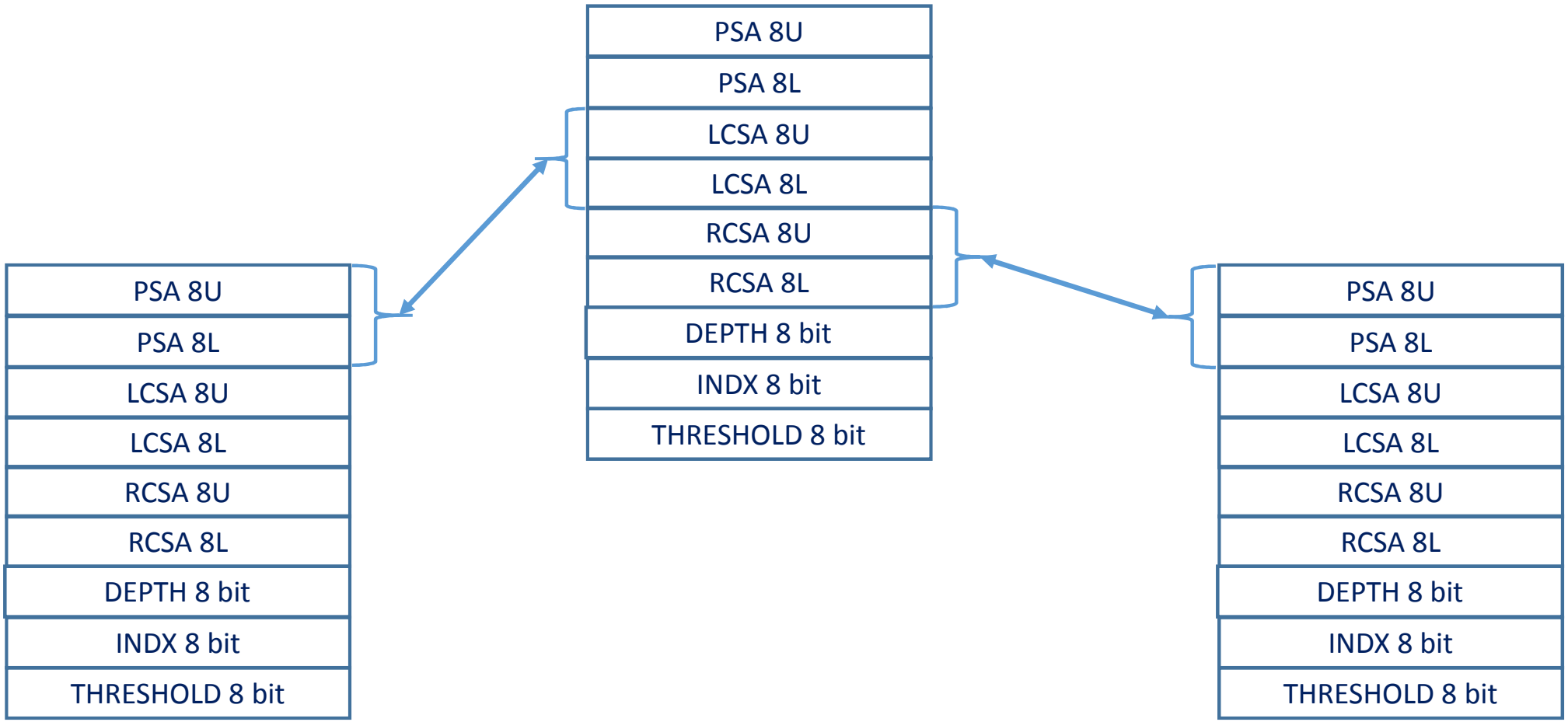
INDX 8 bit → Storing Index k_q of $x[k_q] \geq \theta_q$ for SPLIT NODES; Set to -1 for LEAF NODES

THRESHOLD 8 bit → Storing Index θ_q of $x[k_q] \geq \theta_q$ for SPLIT NODES; Ignored for LEAF NODES

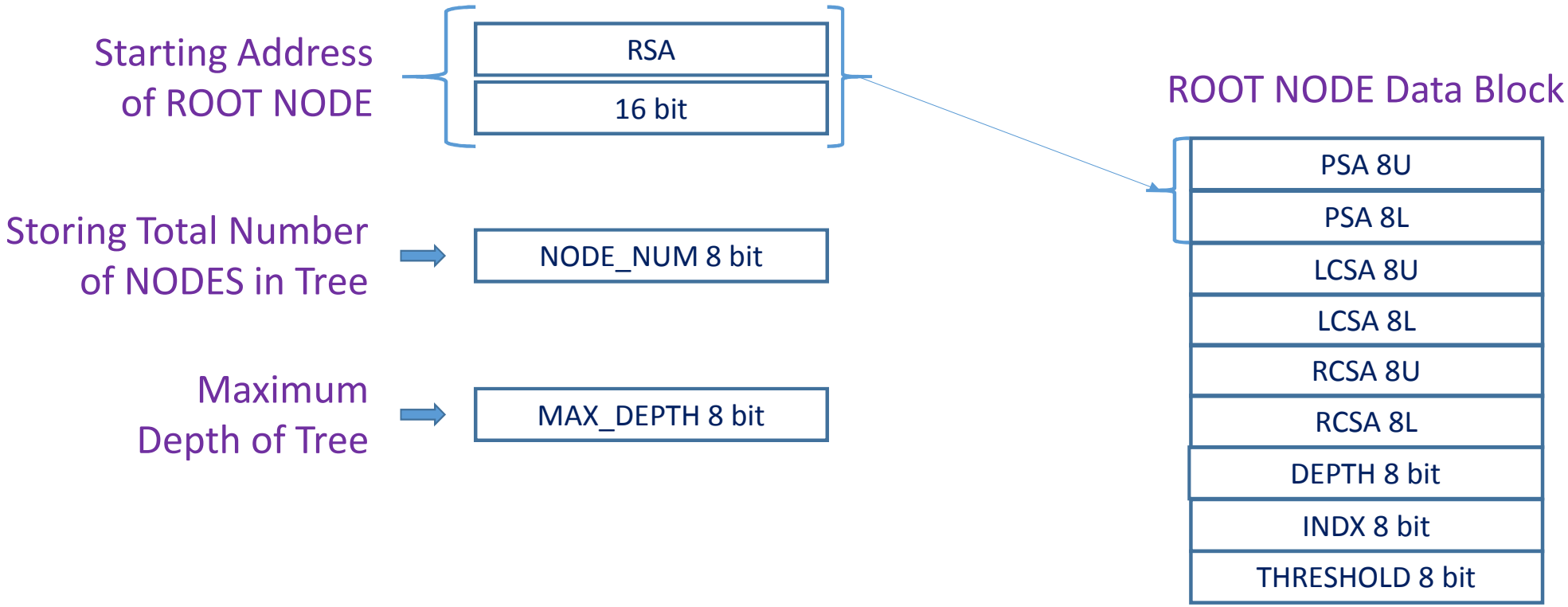
Node Data Block: 9 Bytes

PSA 8U
PSA 8L
LCSA 8U
LCSA 8L
RCSA 8U
RCSA 8L
DEPTH 8 bit
INDX 8 bit
THRESHOLD 8 bit

Tree is a Collection of Connected Nodes



Tree Data Block



Assignment Problem

Subroutine: Adding each Node to the Tree using the Data Block Structure Stated in Slide 7.

Subroutine: Computation of 8-bit Hash for a 2D Input Space

Realize the Tree T_3 (Slide 6) using 8085 Assembly Code using above mentioned Subroutines. Also, evaluate the Hash for the four Inputs $(\pm 5, \pm 5)$.

Applications of Hashing

- Multi-modal Database Indexing
- Input Space Refers to Feature Space of Data
- Fast Near Duplicate Detection in Archives
- Hierarchical Grouping
- Search in Image/Video Databases
- Audio Similarity Identification
- Digital Audio-Video Fingerprinting


भारतीय प्रौद्योगिकी संस्थान गुवाहाटी
Indian Institute of Technology Guwahati

Thank You