ARCHITECTURE AND IMPLEMENTATION OF AN ASSOCIATIVE MEMORY USING SPARSE CLUSTERED NETWORKS

McGill University

Department of Electrical and Computer Engineering

ISCAS 2012 – Seoul Korea
May 23

Hooman Jarollahi – Ph.D. Candidate

Naoya Onizawa, Vincent Gripon, Warren Gross

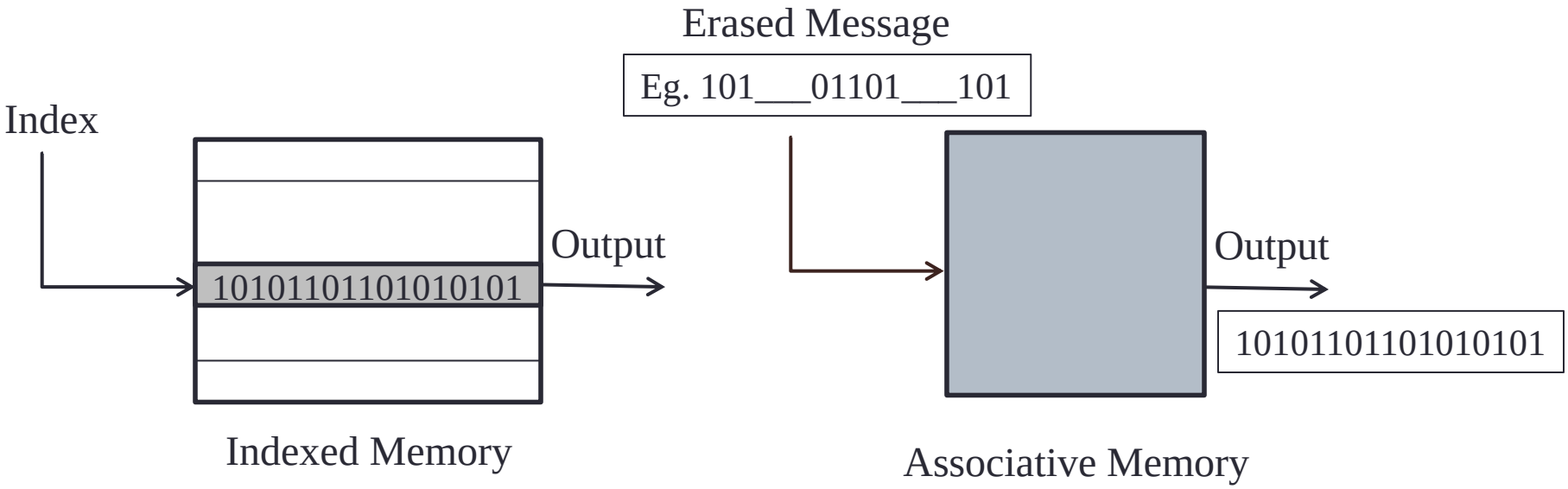
McGill

Outline

- Motivation
- New class of Associative Memories (AMs) using sparse clustered neural networks
- FPGA implementation and results
- Conclusions and future work

Motivation

- Associative Memories: alternatives to indexed memories
- Contents are linked, links are stored
- No need to input an explicit address
- Part of the content of a message is used to retrieve the full message
- Applications: data mining, set implementation



Motivation

- Classical AM: Hopfield Neural Networks (HNN)
  - Fast retrieval of a partially erased messages when implemented in hardware
    - Parallelism

Original Message: 001101101110
Erased Message: 001____ 11__

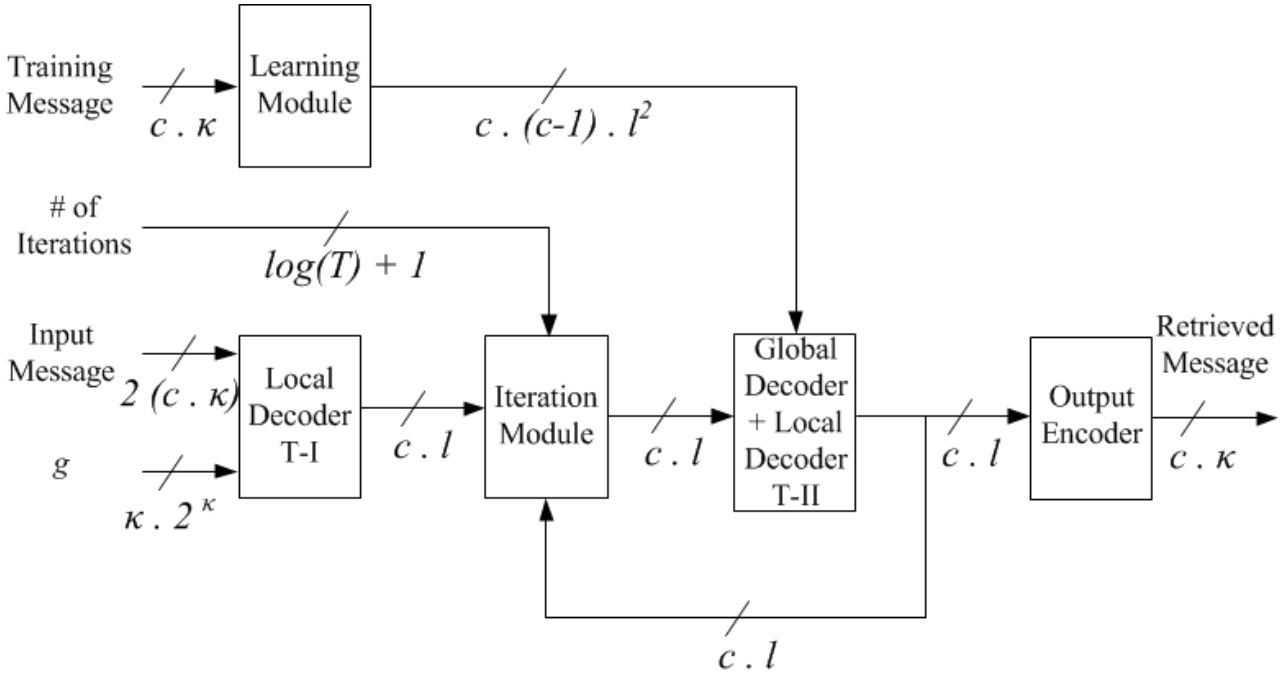
- Storing links between contents (learning)
  - Efficiency: Information bits stored / Memory bits used
- Problems of HNN:
  - Efficiency $\rightarrow$ 0 as learning information is increased
  - A message needs to be as long as the network
    - Long messages + limited capacity $\rightarrow$ low diversity: Number of different messages the network can learn

Gripon-Berrou Neural Networks (GBNN)*

- Case study: 1.6×10^6 bits of physical memory
  - Nearly-optimal efficiency (~1 and ~ ×20 HNN)
  - Large diversity (~ ×250 HNN)
- Binary connections and nodes (existence or non-existence)

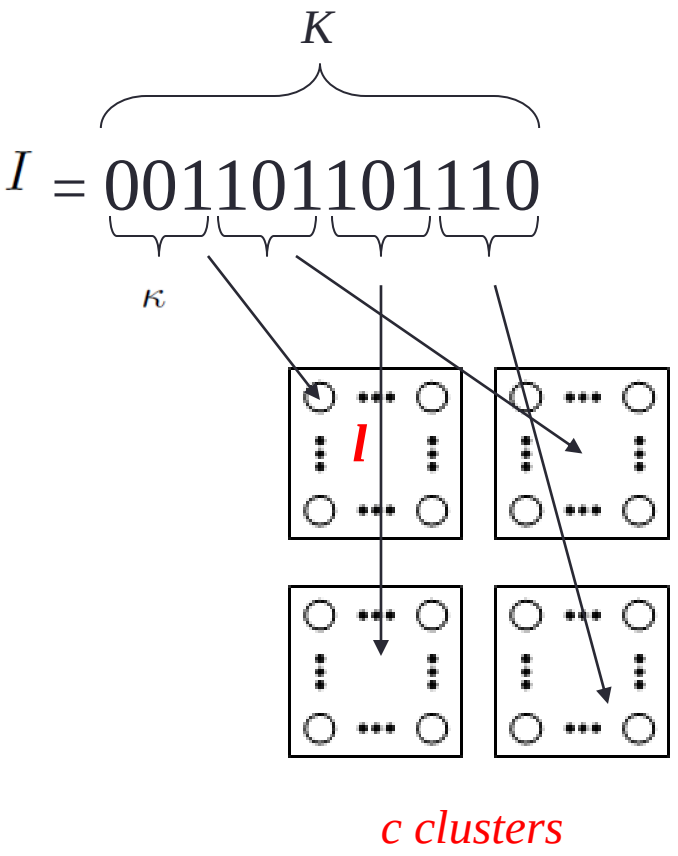
* V. Gripon and C. Berrou, "Sparse neural networks with large learning diversity," *Neural Networks, IEEE Transactions on*, vol. 22, no. 7, pp. 1087 –1096, july 2011.

Simplified Block Diagram

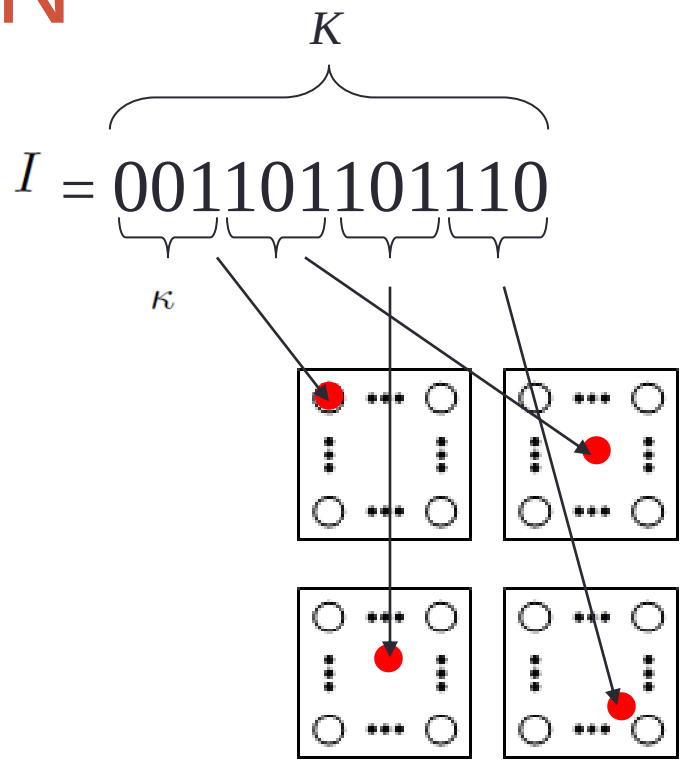


GBNN: Learning

- A network consists of
  - *c* clusters
  - *l* fanals (neurons) per cluster
  - *n* neurons
- A message consists of
  - *K* bits
  - K/c sub-messages
  - $\kappa = K/c = log_2(l)$

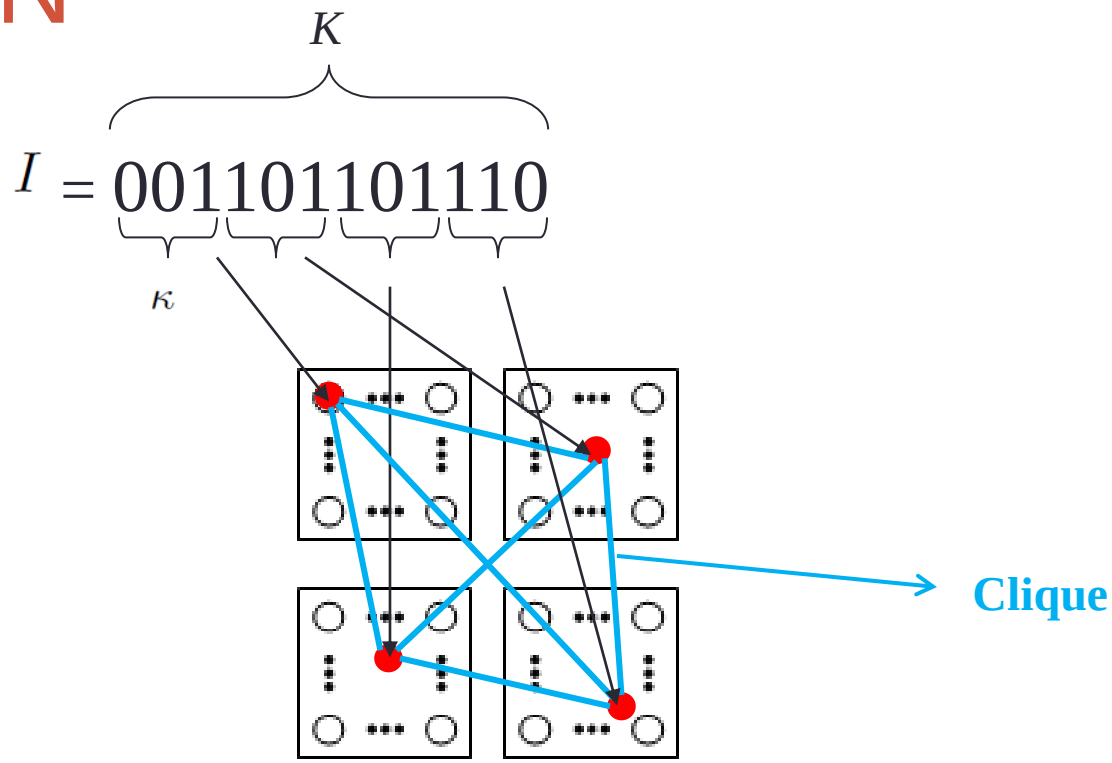


GBNN



- Need to present erasures somehow.
- Replace 0 with -1, 1 stays 1, 0 will mean erased bit.

GBNN



- Need to present erasures somehow.
- Replace 0 with -1, 1 stays 1, 0 will mean erased bit.
- **Clique:** activated fanals with fully-interconnected links

GBNN: Message Retrieval

- Local Decoding Type I (LDT-I):
  - Scalar product of a pre-defined matrix g and input bits
  - Followed by maximum detection
    - Compare-and-Select

$$g = \underbrace{\begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 \\ -1 & -1 & 1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & -1 \\ -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & 1 \\ 1 & -1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{\kappa = 4} \Big\} 2^{\kappa}$$

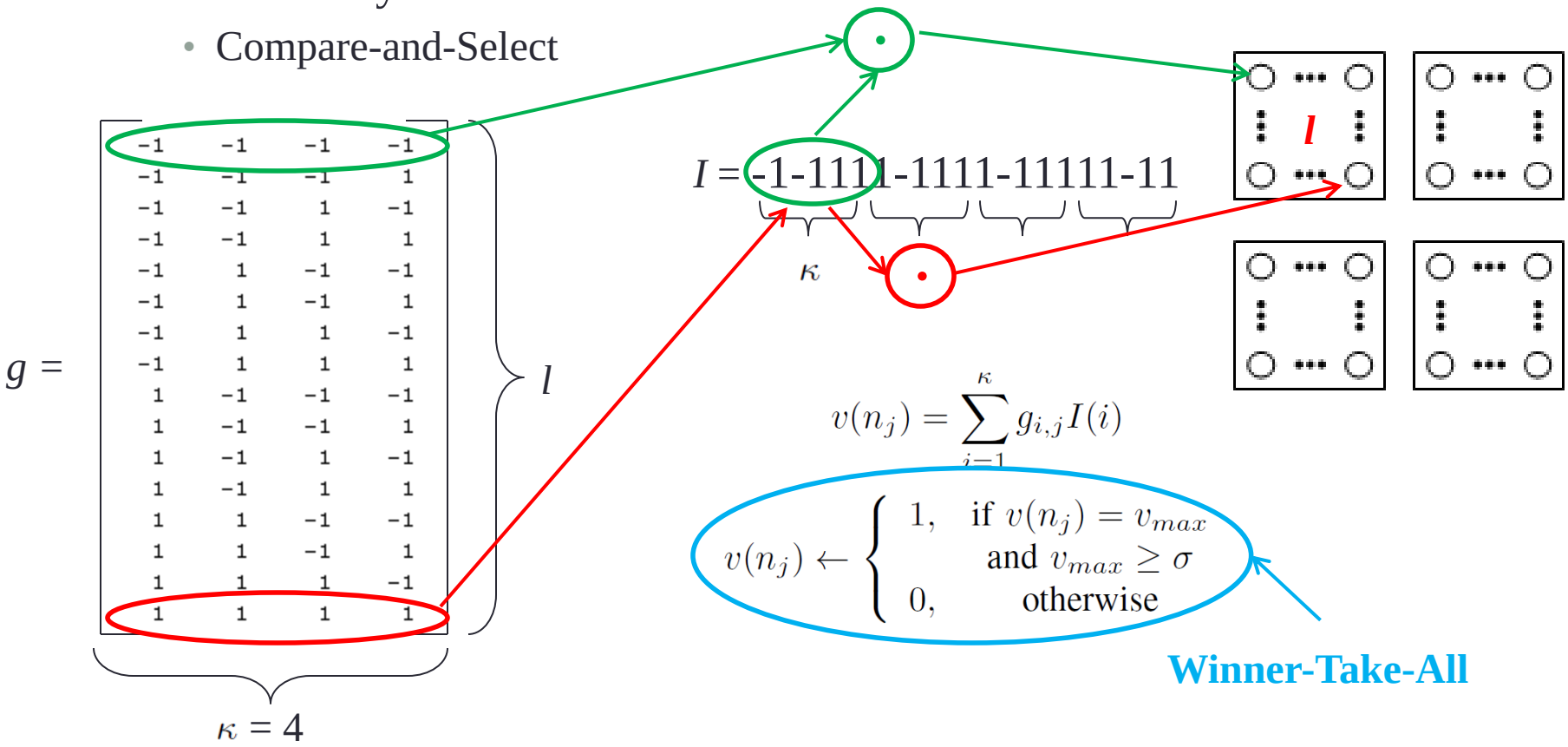
$$I = \underbrace{\text{-1-1111-1111-11111-11}}_{\kappa}$$

$$v(n_j) = \sum_{i=1}^{\kappa} g_{i,j} I(i)$$

$$v(n_j) \leftarrow \begin{cases} 1, & \text{if } v(n_j) = v_{max} \\ & \text{and } v_{max} \geq \sigma \\ 0, & \text{otherwise} \end{cases}$$

GBNN: Message Retrieval

- Local Decoding Type I (LDT-I):
  - Scalar product of a pre-defined matrix g and input bits
  - Followed by maximum detection
    - Compare-and-Select

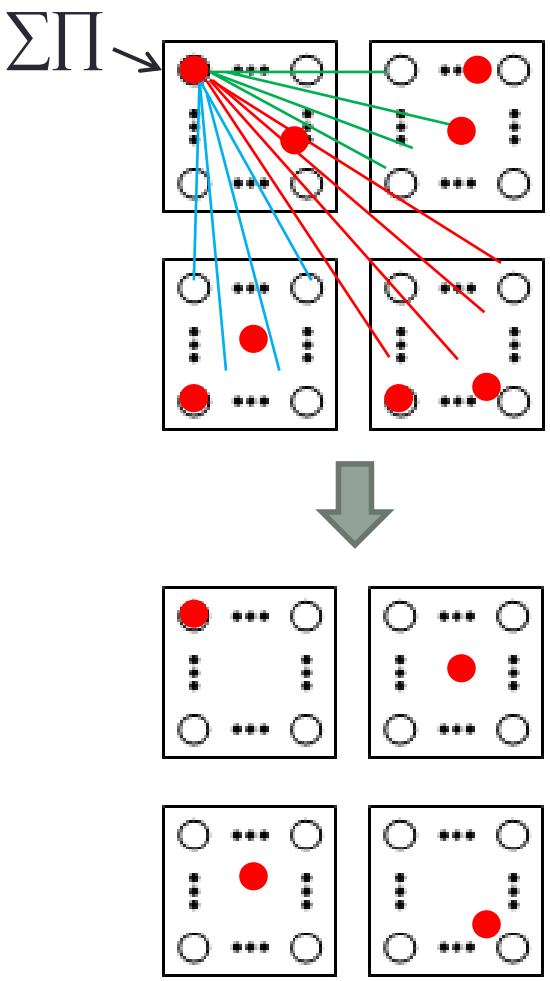


GBNN: Message Retrieval

- Global Decoding followed by LDT-II
- Iterative process

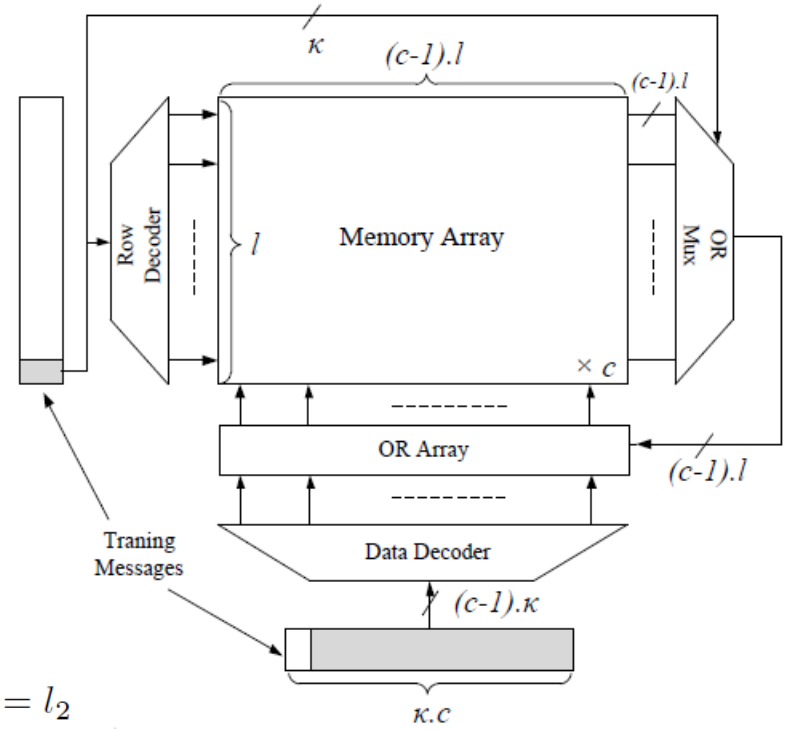
$$\forall i,j, v(n_{i,j}) \leftarrow \sum_{i'=1}^{c}\sum_{j'=1}^{l} w_{(i',j')(i,j)} v(n_{(i',j')}) + \gamma v(n_{(i,j)})$$

$$v(n_j) \leftarrow \begin{cases} 1, & \text{if } v(n_j) = v_{max} \\ & \text{and } v_{max} \geq \sigma \\ 0, & \text{otherwise} \end{cases}$$



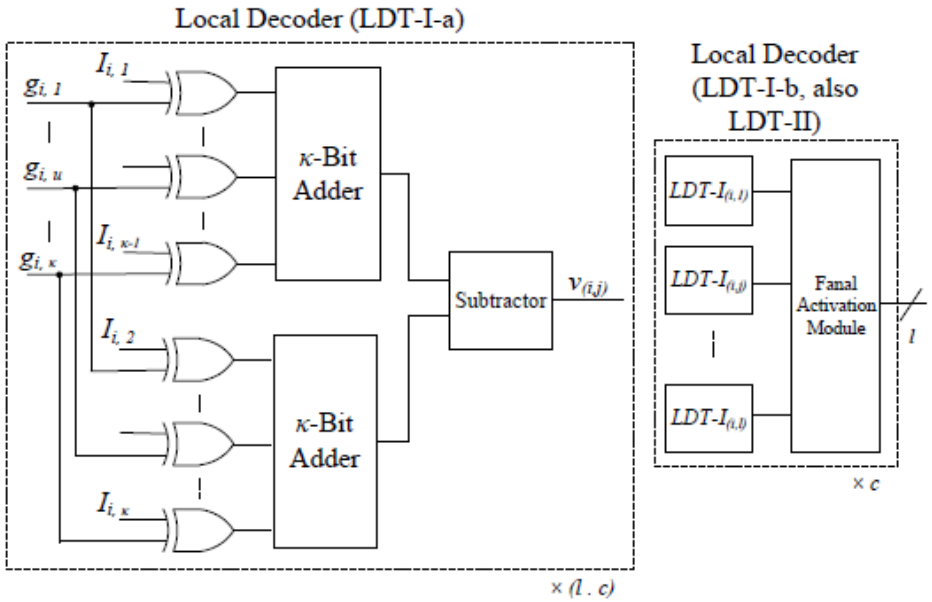
Hardware Architecture: Learning Module

- Learning Module
- Memory used: $c(c\text{-}1)l^2$ bits



$$w_{(c_1,l_1)(c_2,l_2)} = \begin{cases} 1, & \text{if } \begin{cases} c_1 \neq c_2 \\ \text{and } \exists m \in \{m_1 \ldots m_M\} \\ C(m)_{c_1} = l_1 \text{ and } C(m)_{c_2} = l_2 \end{cases} \\ 0, & \text{otherwise} \end{cases}$$

Local Decoding Type I (LDT-I)

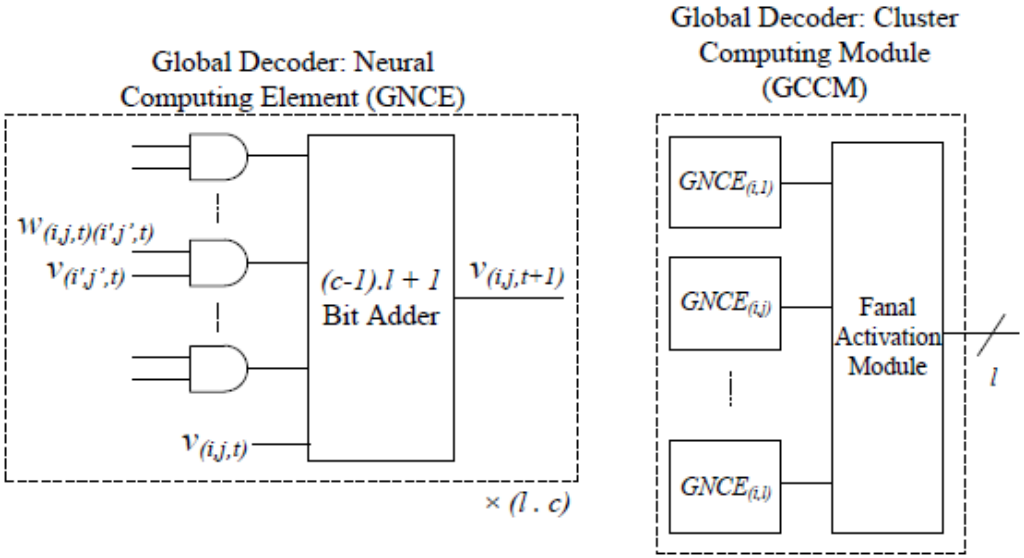


$$v(n_j) = \sum_{i=1}^{\kappa} g_{i,j} I(i)$$

$$v(n_j) \leftarrow \begin{cases} 1, & \text{if } v(n_j) = v_{max} \\ & \text{and } v_{max} \geq \sigma \\ 0, & \text{otherwise} \end{cases}$$

- Input messages are antipodal → 2 bits per input bit required (+1 → 01) , -1 → 10, erasure → 00)
- Fanal Activation Module: Max-Function
- Threshold value (σ) is set to 0 for LDT-I

Global Decoding + LDT-II



$$\forall i, j, v(n_{i,j}) \leftarrow \sum_{i'=1}^{c} \sum_{j'=1}^{l} w_{(i',j')(i,j)} v(n_{(i',j')}) + \gamma v(n_{(i,j)})$$

$$v(n_j) \leftarrow \begin{cases} 1, & \text{if } v(n_j) = v_{max} \\ & \text{and } v_{max} \geq \sigma \\ 0, & \text{otherwise} \end{cases}$$

- Threshold value is set to *c* for LDT-II
- Fanal Activation Module : Max-function (Compare-and-Select)

Proof-of-Concept Design Parameters

No. of Neurons (n)	128
No. of Clusters (c)	8
No. of Fanals per Cluster (l)	16
Message Length (bits)	32
κ	4
Maximum Diversity (Upper bound) (M_{max})	224

Maximum Diversity: $$M_{max} = \frac{(c-1)n^2}{2c^2 log_2(n/c)}$$

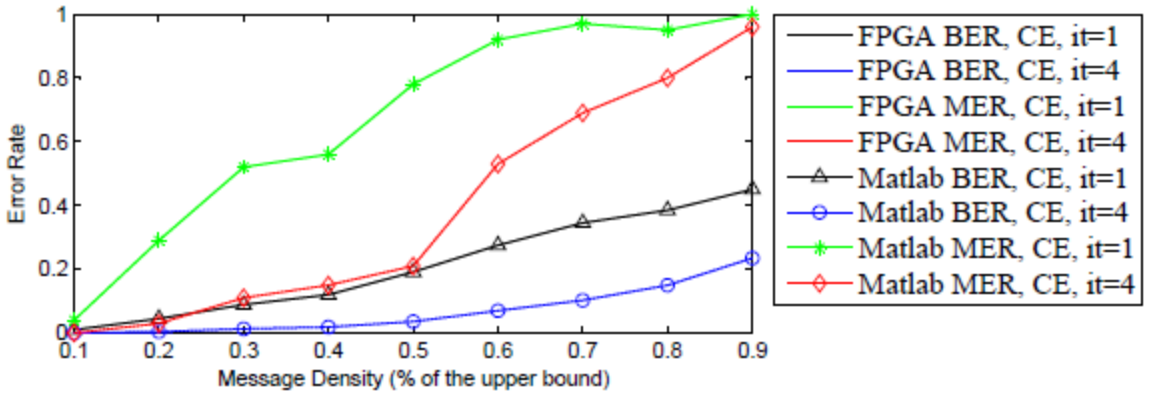
FPGA results

- Software delay measured on AMD Opteron 8387 (2.8 GHz)
- Hardware delay calculated using Altera Stratix IV

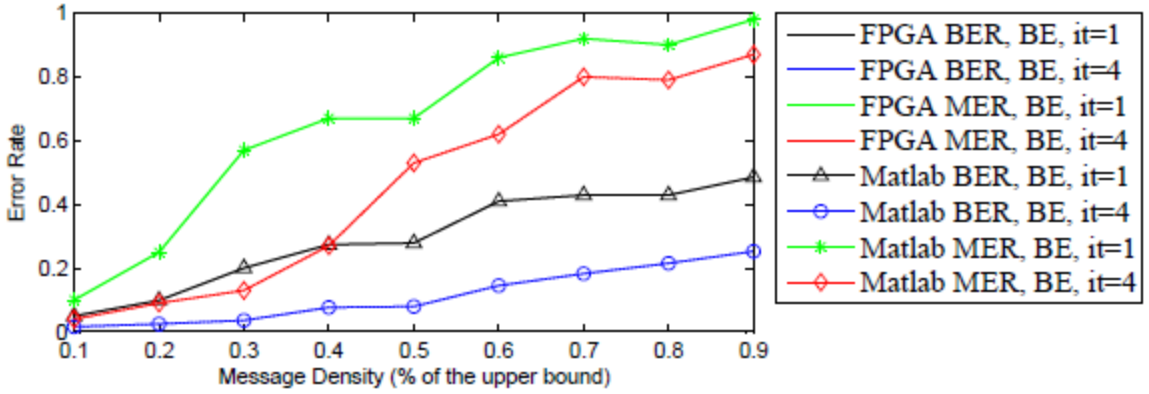
Memory usage dedicated to w (bits)	14,336
Dedicated Logic Registers	$15,783/182,400(9\%)$
Combinational Look-up Tables (LUT)	$35,224/182,400(19\%)$
Total pins	$169/888(19\%)$
Slow 900mv 85C maximum frequency (Mhz)	107.15
Training, Retrieving delay (per message)	10 ns, 50 ns
Software to hardware delay ratio	≈ 2000

Software vs. FPGA Implementation

50% Cluster Erasure



10% Bit Erasure



Conclusions and Future Work

- Proof-of-concept architecture and implementation of a new class of associative memories based on sparse clustered neural networks
- Applications: data mining, set implementation
- Advantage to conventional HNN:
  - Large diversity, nearly-optimal efficiency, lower complexity (binary vs. Integer connections)
- Architecture and Implementation on FPGA (Altera Stratix IV)

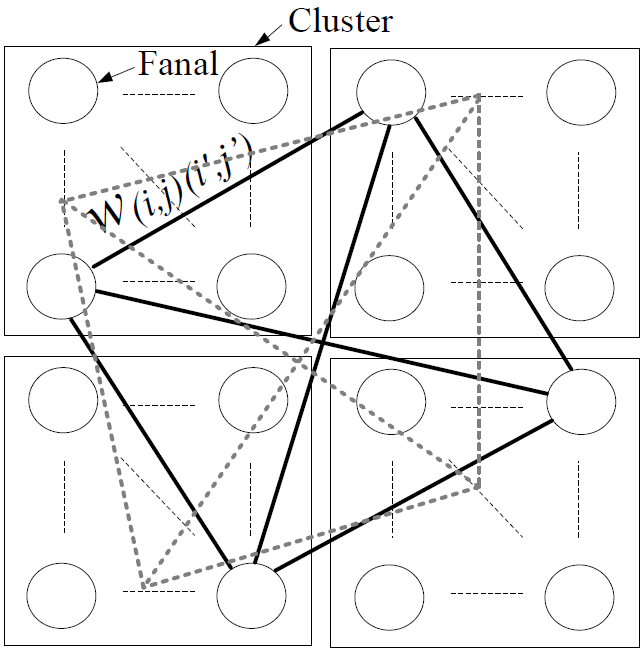
Conclusions and Future Work

- ~2000 times faster than software when implemented on hardware *(n=128, c=8)*
- Verified H/W using S/W, compared BER and MER
- Maximum-Function is expensive and resource hungry: investigation of efficient implementation
- Large scale implementation utilizing external memory resources

Thank you!

Q/A

GBNN: Learning



$$\kappa = K/c = log_2(l)$$

- A message is partitioned into equally-sized sub-messages
- Each sub-message is mapped to a neuron in a cluster using a scalar product of a pre-defined matrix with the sub-message
- Clique is formed