ECE 598HH: Special Topics in Wireless Networks and Mobile Systems

Lecture 9: Backscatter Communications

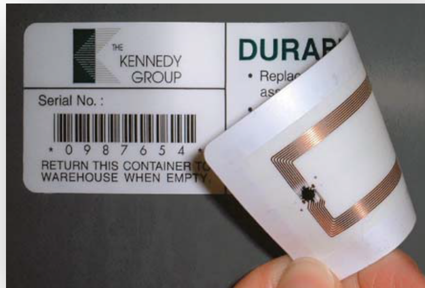
Haitham Hassanieh

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

CSL: COORDINATED SCIENCE LAB

RFIDs


VISA
4000 1234 5678 9010
12/09
B. PARKER


THE KENNEDY GROUP
Serial No. :
RETURN THIS CONTAINER T
WAREHOUSE WHEN EMPTY


ILLINOIS
icard
Card Expires 9/5/2012
HART, BRADLEY ASHTON
Undergraduate
Library 28198009822056
UIN 650098220
Card 5636650098220451


PASSPORT
United States of America

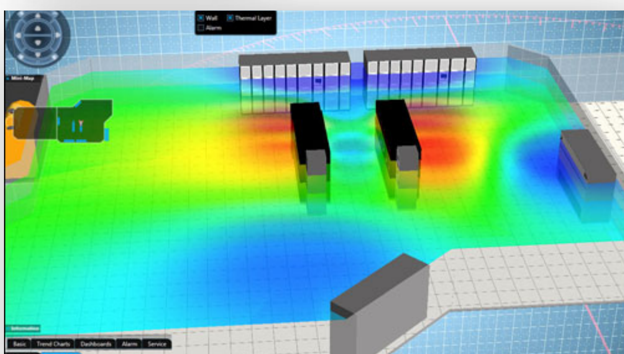

UNITED STATES OF AMERICA
PASSPORT CARD
C00001549
TRAVELER
HAPPY
01 JAN 1981
NEW YORK, USA
15 MAY 18
UNITED STATES DEPARTMENT OF STATE


NEW YORK STATE
ENHANCED
DRIVER LICENSE
ID: 012 345 678
CLASS D
DOCUMENT
SAMPLE, LICENSE
2345 ANYPLACE AVE
ANYTOWN NY 12345
DOB: 06-09-85


PHILADELPHIA


Zipcard
866.4.Zipcar 30080


E-ZPass



Machine-Generated Data

RFID will be a major source of such traffic

- In Oil & Gas – about 30% annual growth rate
- In Healthcare – $1.3B revenue annually
- "number of RFID tags sold globally is projected to rise from 12 million in 2011 to **209 billion** in 2021."

– *McKinsey Big Data Report 2011*

Can we use current wireless protocols for these low power networks?

RFID Requirements

- Small form factor
- Massive scale
- Lifetime

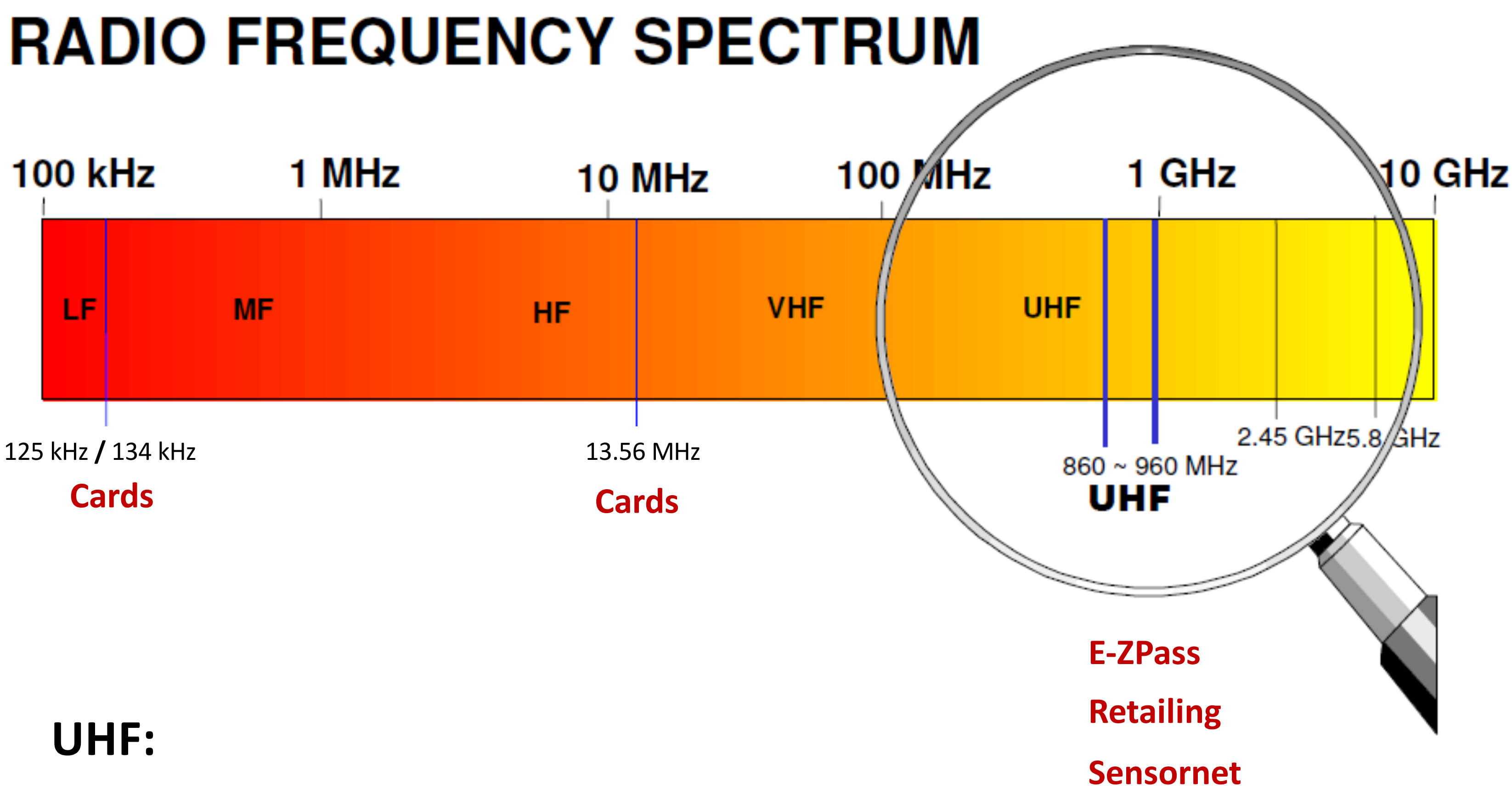
RFID Constraints

- No battery
- Ultra-low cost
- Simple circuitry

- Wireless protocols require power and computation

RFIDs can't perform typical wireless functions like carrier sense or rate adaptation

RFID Background

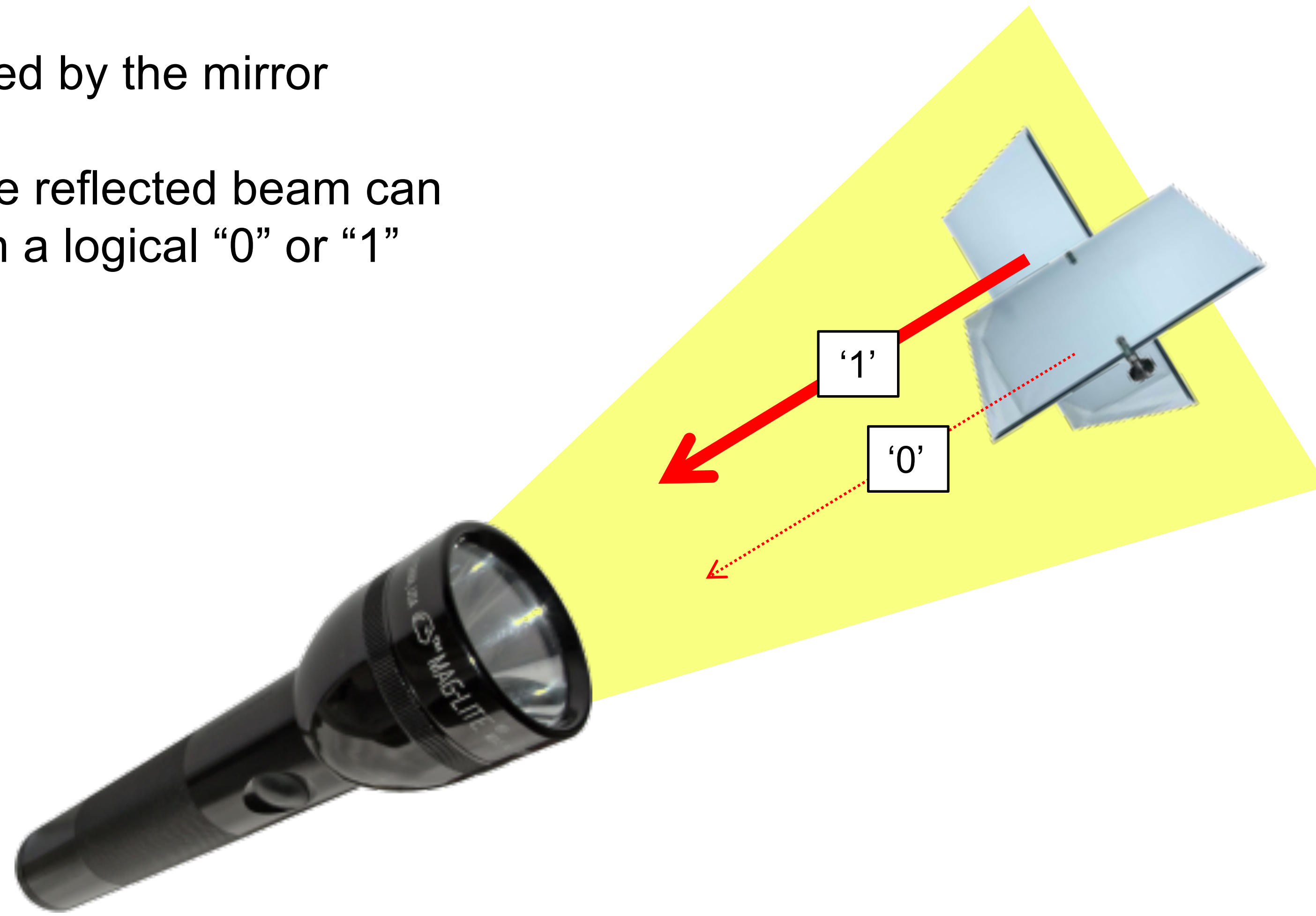


UHF:

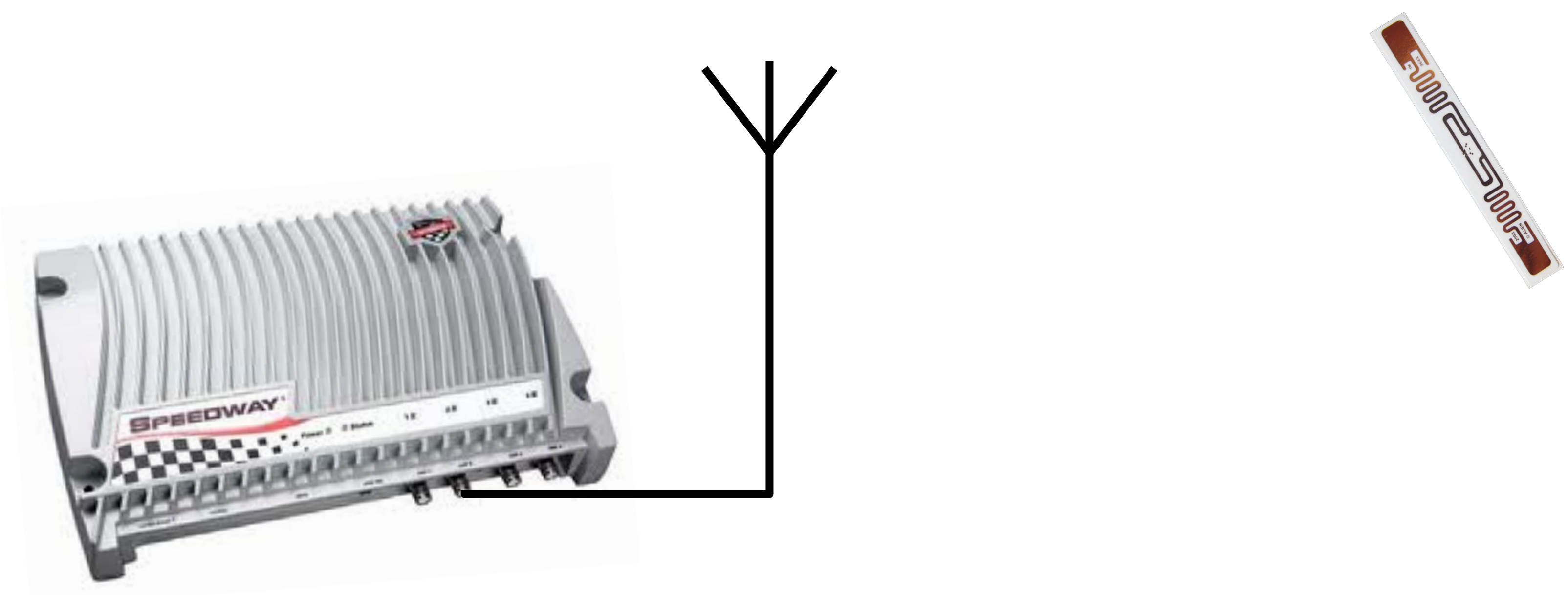
- Achieves higher range (few meters v.s. cm)
- Uses backscatter communication instead of inductive coupling

Backscatter Communication

- A flashlight emits a beam of light
- The light is reflected by the mirror
- The intensity of the reflected beam can be associated with a logical “0” or “1”



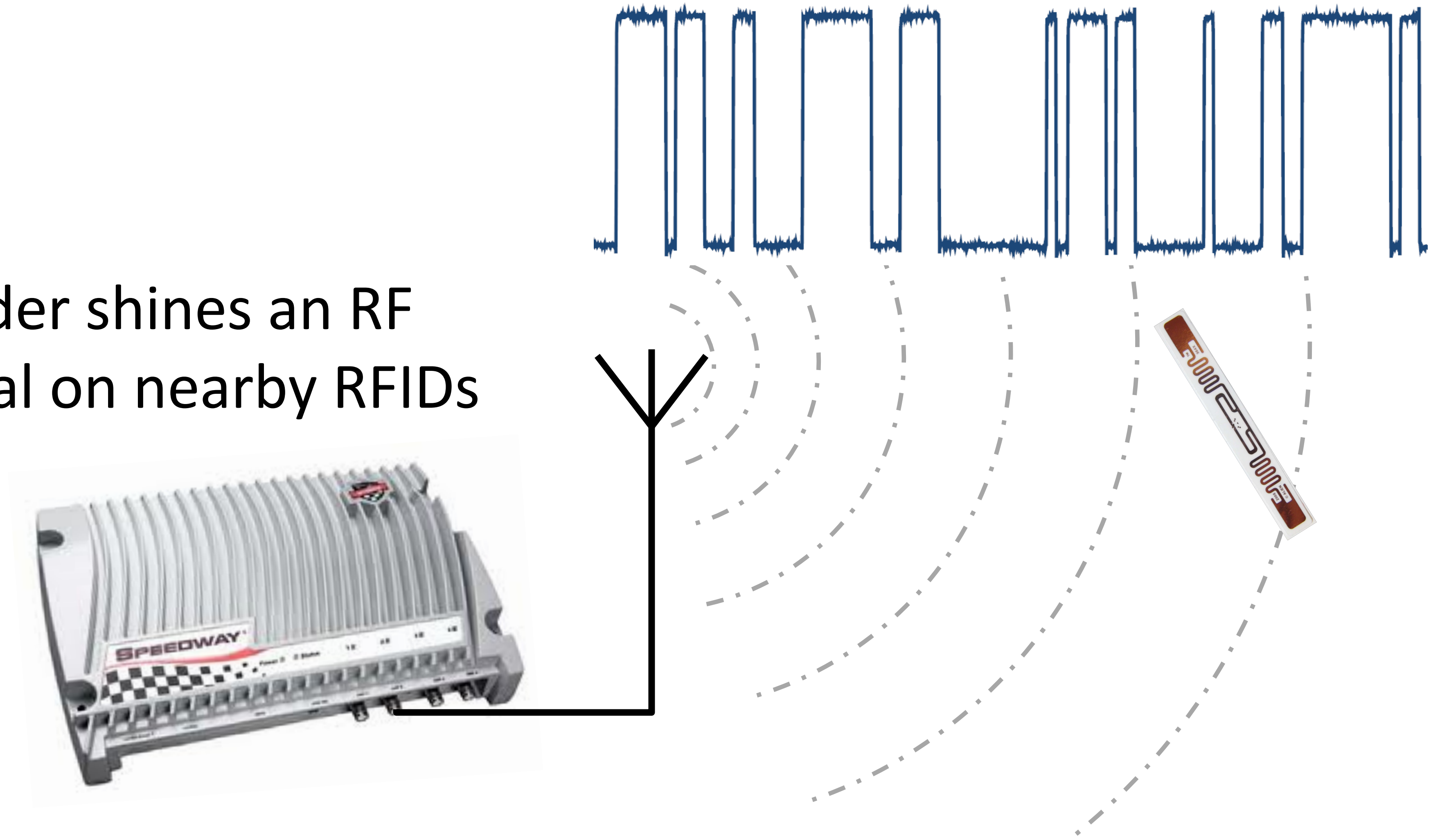
Backscatter Communication



Backscatter Communication

Tag reflects the reader's signal using ON-OFF keying

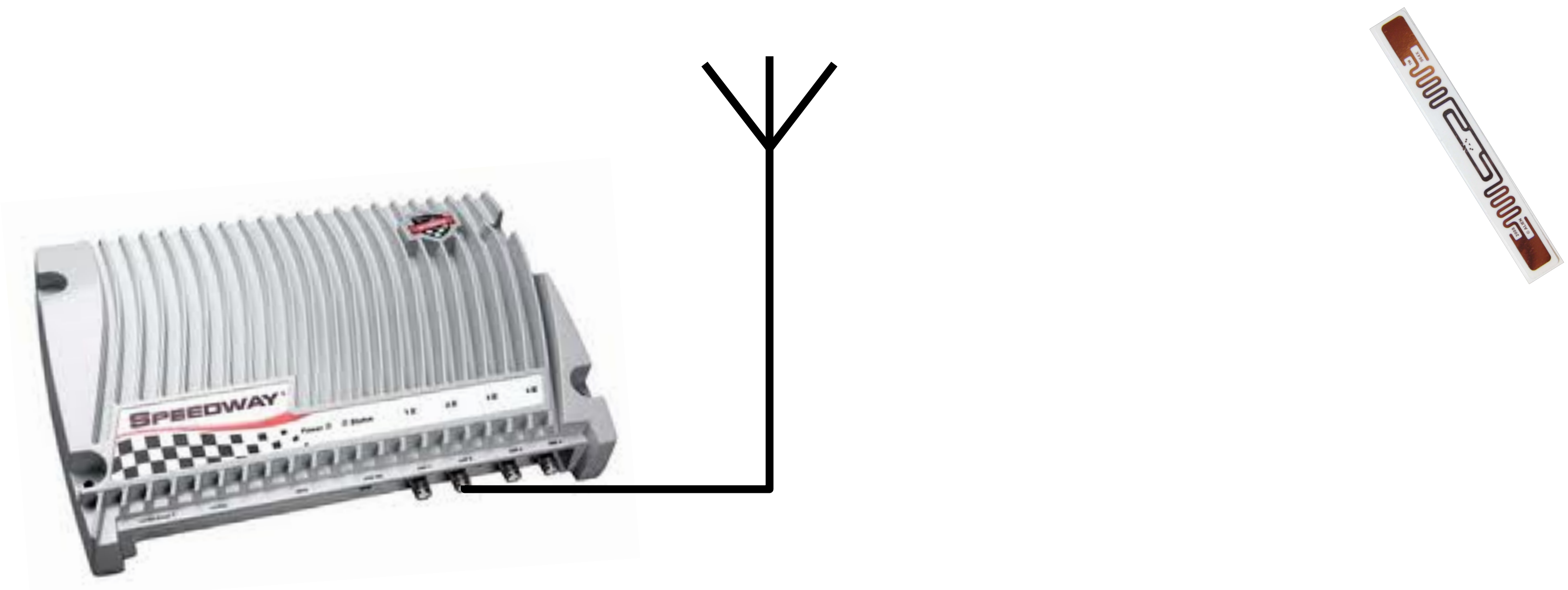
Reader shines an RF signal on nearby RFIDs



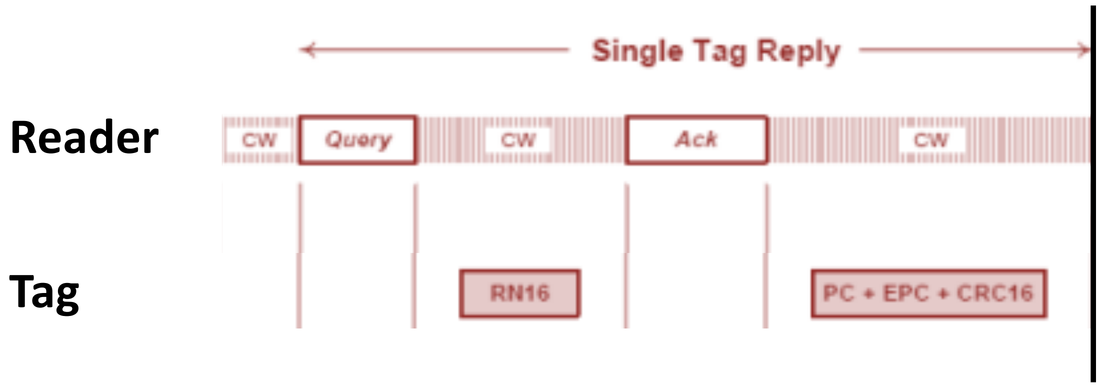
Backscatter Communication

RFIDs are synced by the reader's signal:

- Time synchronization
- Frequency synchronization



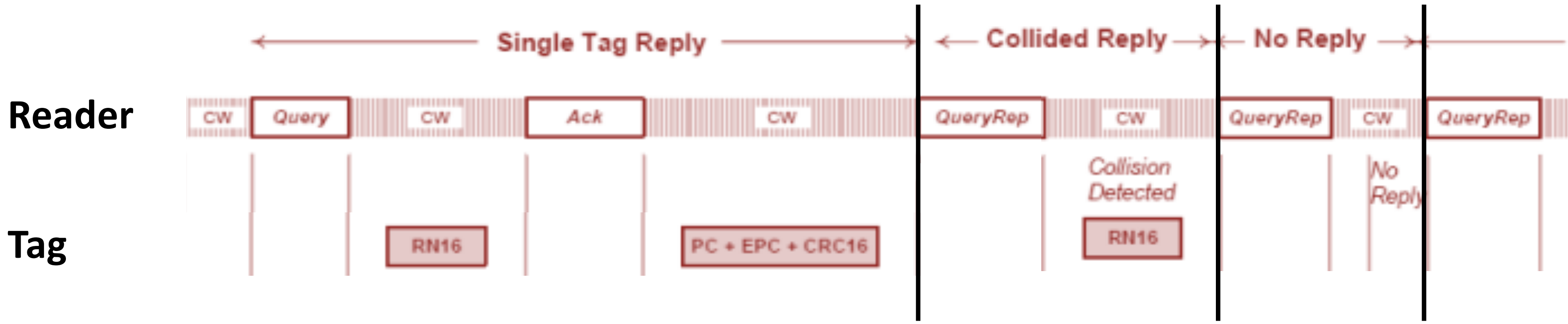
EPC Gen2 Standard – MAC



Slotted Aloha:

- Reader allocates Q time slots and transmits a query at the beginning of each time slot
- Each tag picks a random slot and transmits a 16-bit random number
- In each slot:
  - RN16 decoded → Reader ACKs → Tags transmits 96-bit ID
  - Collision → Reader moves on to next slot
  - No reply → Reader moves on to next slot

EPC Gen2 – MAC



Inefficient:

- If reader allocates large number of slots → Too many empty slots
- If reader allocates small number of slots → Too many collisions
- If reader knows number of tags = K → Allocate K slots → 37% efficiency
- Downlink overhead

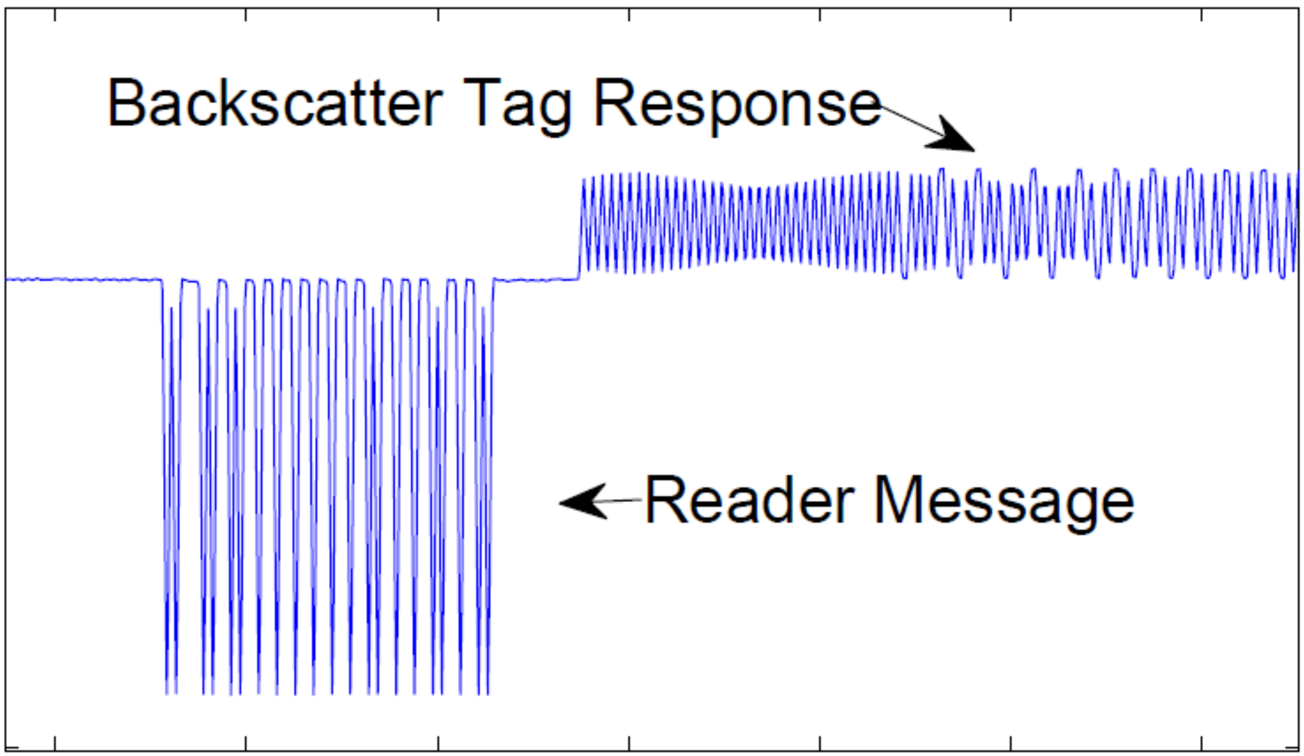
EPC Gen2 – Rate Adaptation

- TDMA
- Fixed modulation: 1 bit/symbol ON-OFF keying (ASK)
- Miller-4 encoding
- No effective adaptation → message loss

Challenges of Backscatter

RFIDs cannot hear each other

→ Collisions



Challenges of Backscatter

RFIDs cannot hear each other

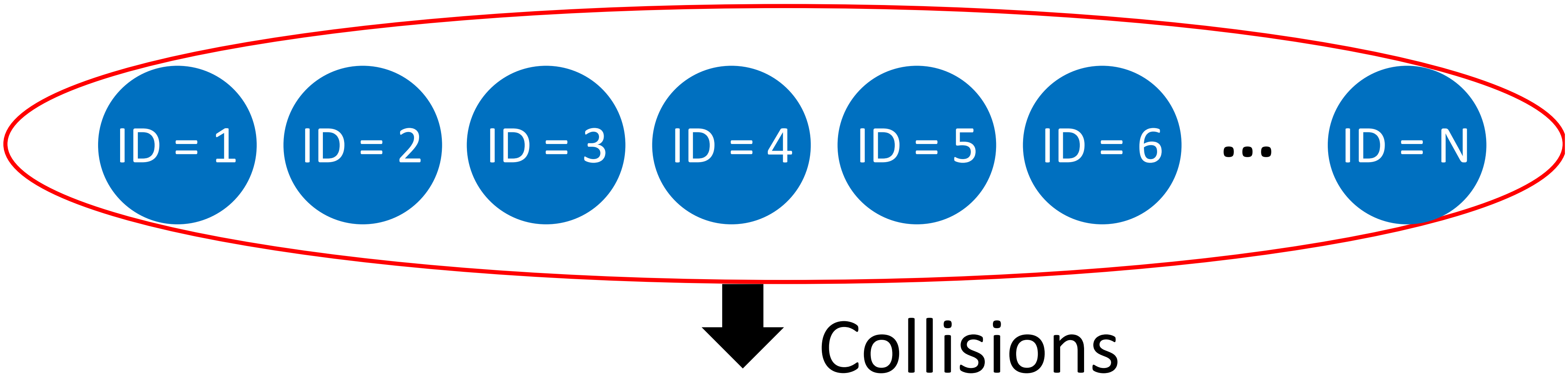
→ Collisions

Cannot adapt modulation to channel quality

→ Don't exploit a good channel to send more bits per symbol

→ Don't react to a bad channel

Network As a Node



Collision becomes a code across the virtual sender's bits

- Deals with collision by decoding collision-code
- Adapts the rate by making collision-code rateless

Network-As-a-Node

Node Identification

Data Communication

The Node Identification Problem

Each object has an ID
Reader learns IDs of nearby objects

Applications

- Inventory management
- Shopping cart

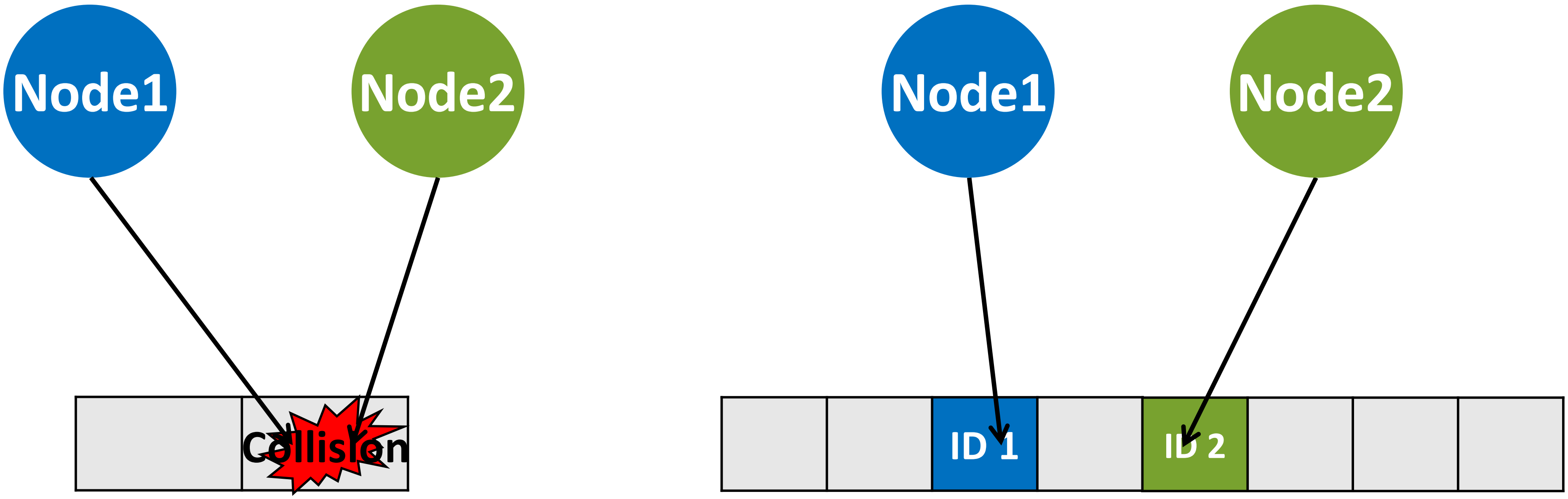
Challenge: RFIDs cannot hear each other

→ Collisions

Current Approach: Slotted Aloha

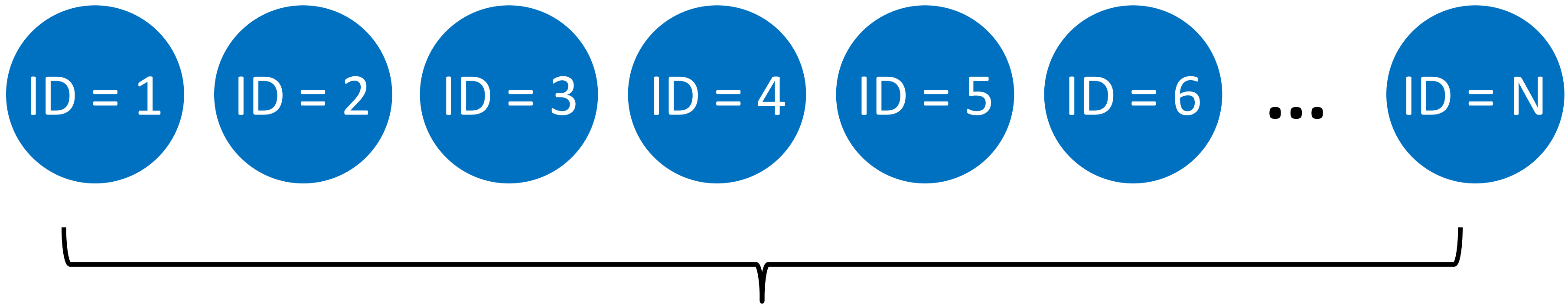
Time is divided into slots;

Each RFID transmits in a random slot

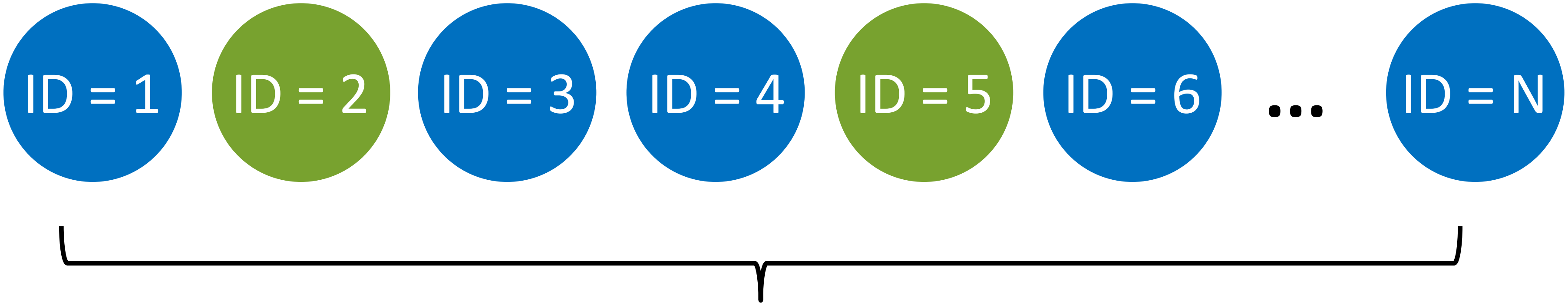


Few Time Slots **OR** Many Time Slots

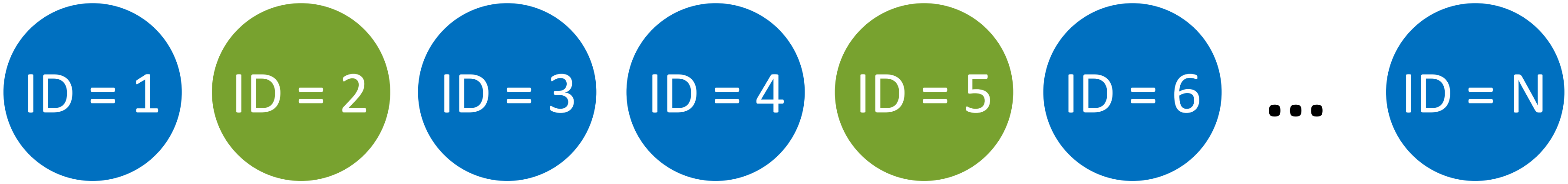
Unreliable **Inefficient**



A million RFIDs in the Wal-Mart store



But only a few (e.g., 20) in the shopping cart



System is represented by a vector $\mathbf{x}$

$x_i = 1$ if node with ID = i is in cart

0	**1**	**0**	**0**	**1**	**0**	**...**	**0**

vector $\mathbf{x}$

0	1	0	0	1	0	…	0

Ideally, want to compress $\mathbf{x}$ and send it to the reader

But $\mathbf{x}$ is distributed across all nodes!

vector $\mathbf{x}$

0	1	0	0	1	0	...	0

$\mathbf{x}$ is Sparse

Use Compressive sensing to compress and send $\mathbf{x}$

Compressive Sensing

Linear Equations:

$$\mathrm{y} = A\mathrm{x}$$

- M equations and N unknowns: $\mathrm{y}_{M\times 1} = A_{M\times N}\mathrm{x}_{\mathrm{N}\times 1}$

- Solve for: x

- If M<N → Cannot solve for x

Compressive Sensing

Compressive Sensing: $\mathrm{y} = A\mathrm{x}$

- If x has at most K << N non-zero entries: i.e. x is sparse
  → Can recover x from M << N measurements
  → $M = O(K \log N/K)$

- A must satisfy Restricted Isometry Property (RIP)
  - E.g. Random 0/1 or +1/-1
  - E.g. Fourier measurements $e^{-2\pi jft/N}$

- x can be sparse in any domain
  - E.g. images are sparse in Wavelet and Fourier domains.
  - $\mathrm{x} = \Phi\, \mathrm{z}$ and z is sparse → can recover x from $\mathrm{y} = A\mathrm{x} = A\Phi\mathrm{z}$

A Virtual Compressive Sensing Sender

Compressive sensing matrix

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \overset{\boldsymbol{A}}{\begin{bmatrix} 0 & 1 & 1 & 1 & & 0 \\ 0 & 0 & 1 & 0 & \cdots & 1 \\ 1 & 1 & 1 & 0 & & 1 \end{bmatrix}} \times \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

- Virtual sender sends $\boldsymbol{y}$
- Reader decodes $\boldsymbol{x}$ using a compressive sensing decoder

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & & 0 \\ 0 & 0 & 1 & 0 & \cdots & 1 \\ 1 & 1 & 1 & 0 & & 1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

Virtual sender mixes information in $\mathbf{x}$

Network can mix information using Collisions

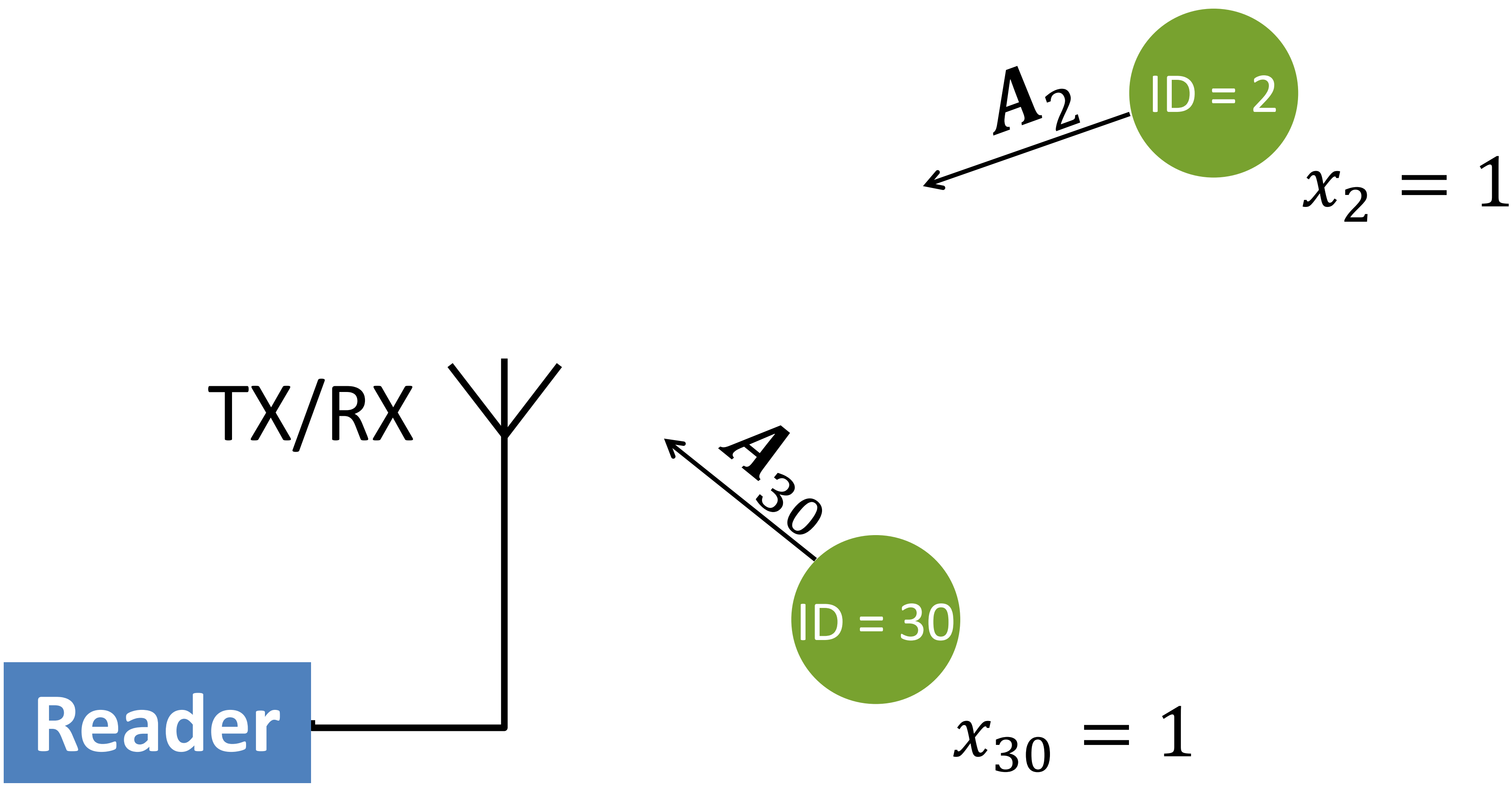
Network Compressive Sensing Using Collisions

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{array}{c} \begin{array}{cccccc} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \boldsymbol{A}_3 & \boldsymbol{A}_4 & & \boldsymbol{A}_N \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & & 0 \\ 0 & 0 & 1 & 0 & \cdots & 1 \\ 1 & 1 & 1 & 0 & & 1 \end{bmatrix} \end{array} \times \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

Node with ID = i transmits $\boldsymbol{A}_i$

Collisions mix on the air

Example: Cart has only ID 2 and ID 30



The reader receives a collision:

$$\boldsymbol{y} = \boldsymbol{A}_2 x_2 + \boldsymbol{A}_{30} x_{30}$$

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_{30} & \cdots & \boldsymbol{A}_N \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{30} \\ \vdots \\ x_N \end{bmatrix}$$

The reader receives a collision:

$$\boldsymbol{y} = \boldsymbol{A}_2 x_2 + \boldsymbol{A}_{30} x_{30}$$

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_{30} & \cdots & \boldsymbol{A}_N \end{bmatrix} \times \begin{bmatrix} 0 \\ x_2 \\ \vdots \\ x_{30} \\ \vdots \\ 0 \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{A}\mathbf{x}$$

Reader uses a compressive sensing decoder to recover $\mathbf{x}$ from $\boldsymbol{y}$

The reader receives a collision:

$$\boldsymbol{y} = \boldsymbol{A}_2 h_2 x_2 + \boldsymbol{A}_{30} h_{30} x_{30}$$

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \cdots & \boldsymbol{A}_{30} & \cdots & \boldsymbol{A}_N \end{bmatrix} \times \begin{bmatrix} 0 \\ h_2 x_2 \\ \vdots \\ h_{30} x_{30} \\ \vdots \\ 0 \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{A}\,\tilde{\mathbf{x}}$$

Allows you to estimate the channel from each tag

Network-As-a-Node

Node Identification

Data Communication

Data communication in RFID networks performs poorly because it lacks rate adaptation

RFIDs always send 1 bit/symbol

Can’t exploit good channels to send more bits

→ Inefficiency

Can’t reduce rate in bad channels

→ Unreliability

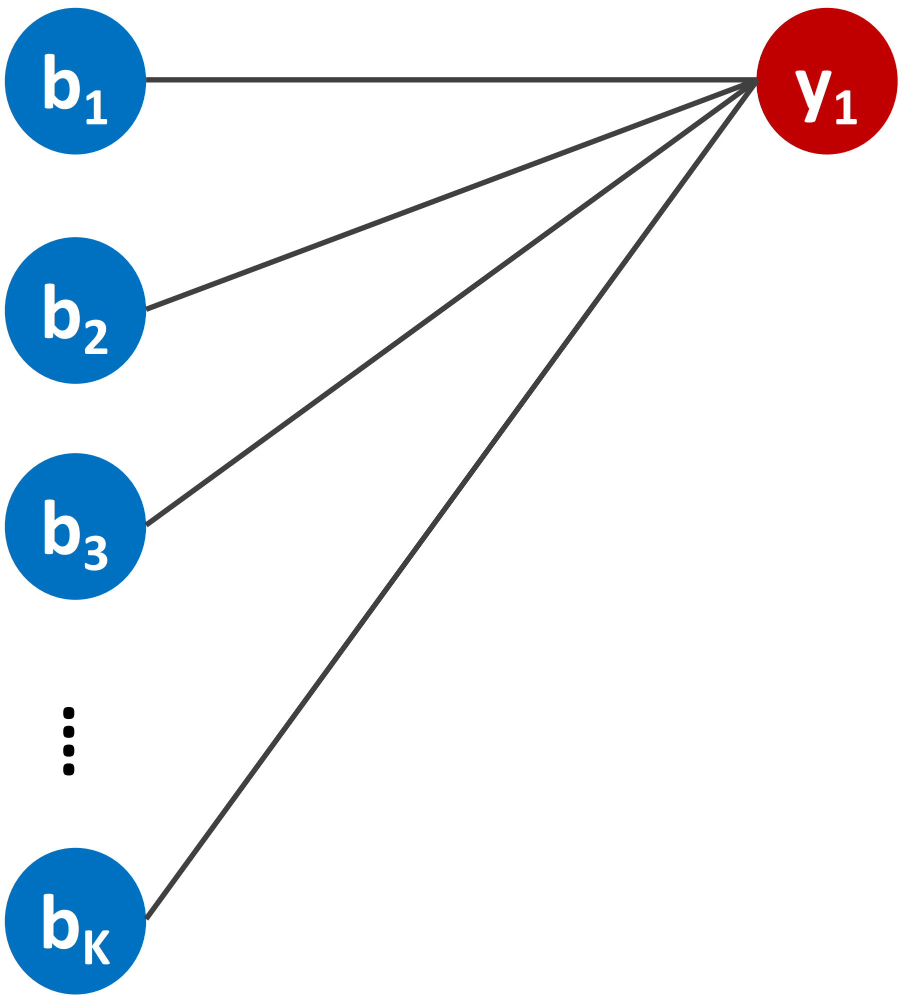
Network-Based Rate Adaptation

- Nodes transmit messages and collide
- Reader collects collisions until it can decode
  - **good channel** → decode from **few collisions**
  - **worse channel** → decode from **more collisions**

Adapts bit rate to channel quality without feedback

Collisions as a Distributed Code

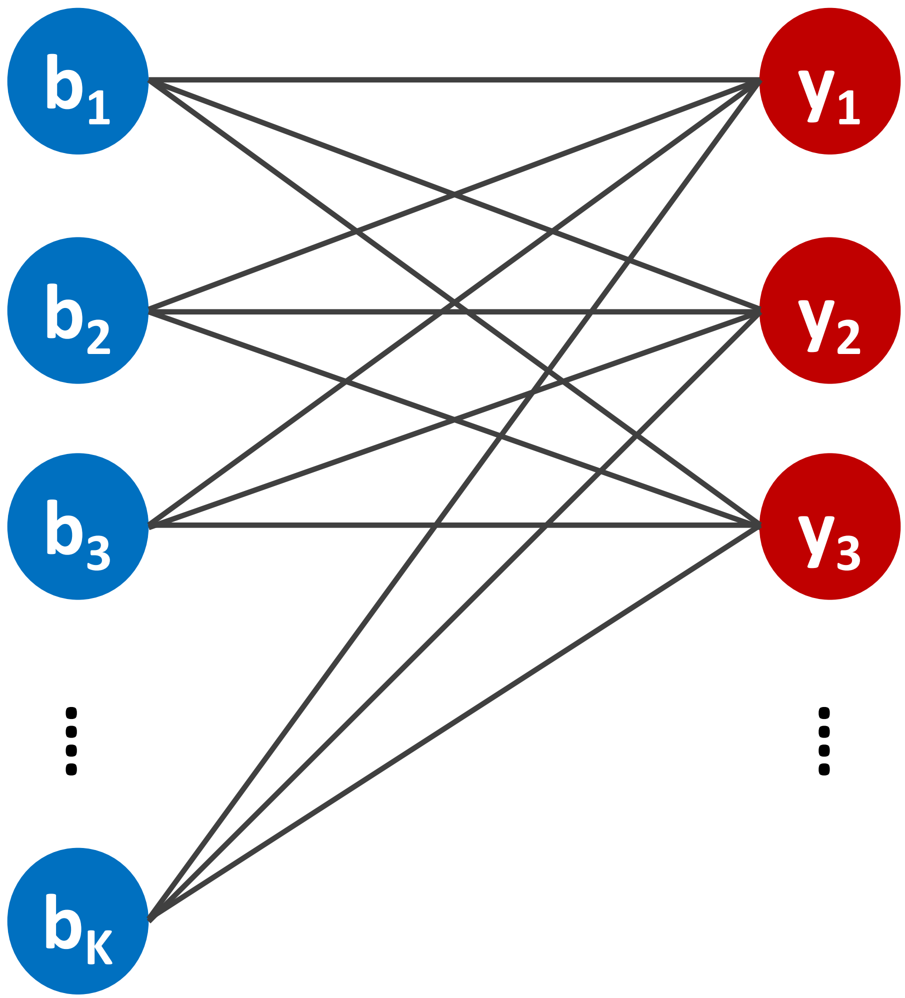
Collisions naturally act like a linear code



$$y_1 = h_1\, b_1 + h_2\, b_2 + \dots + h_K\, b_K$$

But simply colliding is not a good code

Repetition Code → Bad Code!

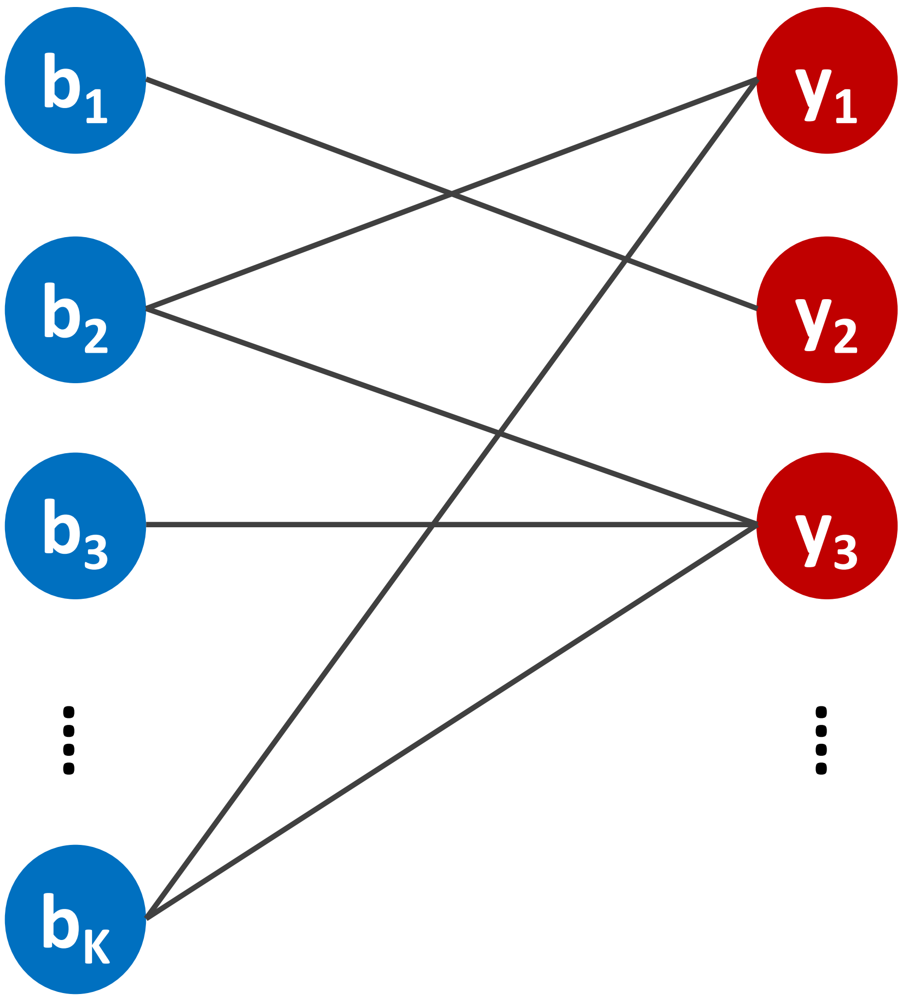


$y_1 = h_1\, b_1 + h_2\, b_2 + \ldots + h_K\, b_K$

$y_2 = h_1\, b_1 + h_2\, b_2 + \ldots + h_K\, b_K$

$y_3 = h_1\, b_1 + h_2\, b_2 + \ldots + h_K\, b_K$

Collisions as a Random Code



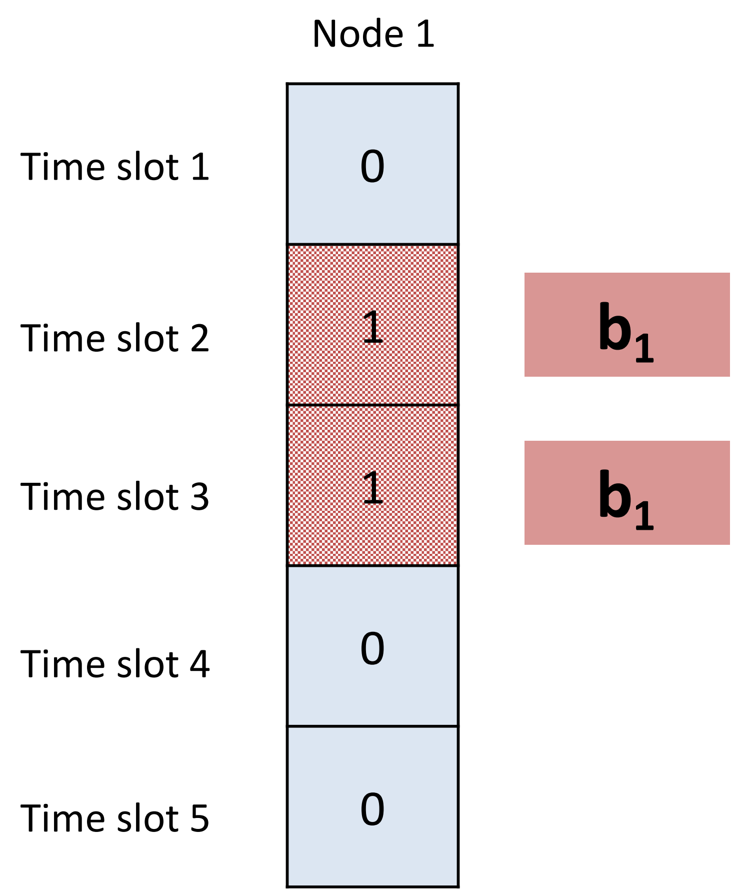
$y_1 = h_2\, b_2 + h_K\, b_K$

$y_2 = h_1\, b_1$

$y_3 = h_2\, b_2 + h_3\, b_3 + h_K\, b_K$

Collision as a Code

- Randomizing at each node
  - 1 => transmits message
  - 0 => remains silent



Collision as a Code

- Randomizing at each node
  - transmits message if 1, remains silent if 0

	Node 1	Node 2	Node 3	Node 4
Time slot 1	0	0	1	1
Time slot 2	1	0	1	0
Time slot 3	1	1	0	1
Time slot 4	0	1	1	0
Time slot 5	0	0	0	1

Collision as a Code

- Creating different linear combinations:

	Node 1	Node 2	Node 3	Node 4
Time slot 1	0	0	1	1
Time slot 2	1	0	1	0
Time slot 3	1	1	0	1
Time slot 4	0	1	1	0
Time slot 5	0	0	0	1

$y_1 = h_3\, b_3 + h_4\, b_4$

Collision as a Code

- Creating different linear combinations:

	Node 1	Node 2	Node 3	Node 4	
Time slot 1	0	0	1	1	$y_1 = h_3\, b_3 + h_4\, b_4$
Time slot 2	1	0	1	0	$y_2 = h_1\, b_1 + h_3\, b_3$
Time slot 3	1	1	0	1	
Time slot 4	0	1	1	0	
Time slot 5	0	0	0	1	

Collision as a Code

- Creating different linear combinations:

	Node 1	Node 2	Node 3	Node 4	
Time slot 1	0	0	1	1	$y_1 = h_3 b_3 + h_4 b_4$
Time slot 2	1	0	1	0	$y_2 = h_1 b_1 + h_3 b_3$
Time slot 3	1	1	0	1	$y_3 = h_1 b_1 + h_2 b_2 + h_4 b_4$
Time slot 4	0	1	1	0	$y_4 = h_2 b_2 + h_3 b_3$
Time slot 5	0	0	0	1	$y_5 = h_4 b_4$

Collision as a Code

- Creating different linear combinations:

Coding Matrix $\mathbf{D}$

$$\mathbf{y} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \mathbf{H} \times \mathbf{b}$$

How to Decode?

- Received noisy symbols $\boldsymbol{y} = \boldsymbol{DHb} + \boldsymbol{n}$
- Possible solution: $\boldsymbol{b} = (\boldsymbol{DH})^{-1}\ \boldsymbol{y}$

$$\mathbf{y} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \times \mathbf{H} \times \mathbf{b}$$

How to Decode?

- Received noisy symbols $\boldsymbol{y} = \boldsymbol{DHb} + \boldsymbol{n}$
- Possible solution: $\boldsymbol{b} = (\boldsymbol{DH})^{-1}\ \boldsymbol{y}$

$$\mathbf{y} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \times \mathbf{H} \times \mathbf{b}$$

How to Decode?

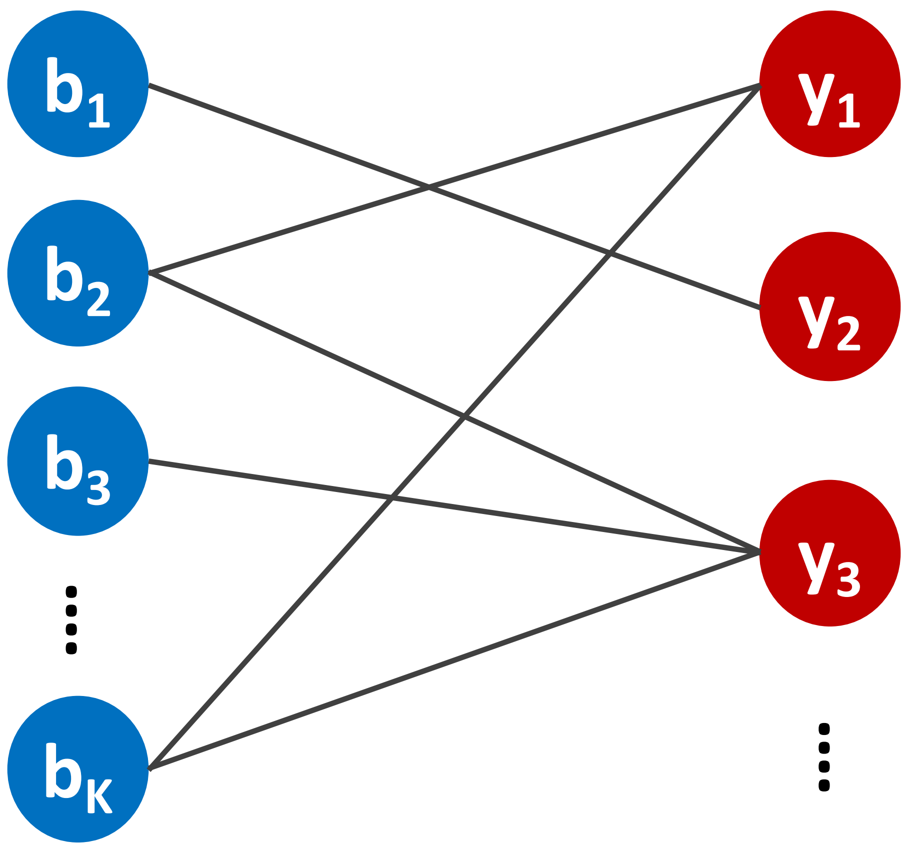
- Received noisy symbols $\boldsymbol{y} = \boldsymbol{DHb} + \boldsymbol{n}$
- Possible solution: $\boldsymbol{b} = (\boldsymbol{DH})^{-1}\, \boldsymbol{y}$

$$\mathbf{y} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \mathbf{H} \times \mathbf{b}$$

Too complex to invert D every time slot!

How Does the Reader Decode?

- Make code sparse → Leverage ideas from LDPC
- Each symbol is a collision of a small random subset of the nodes' bits

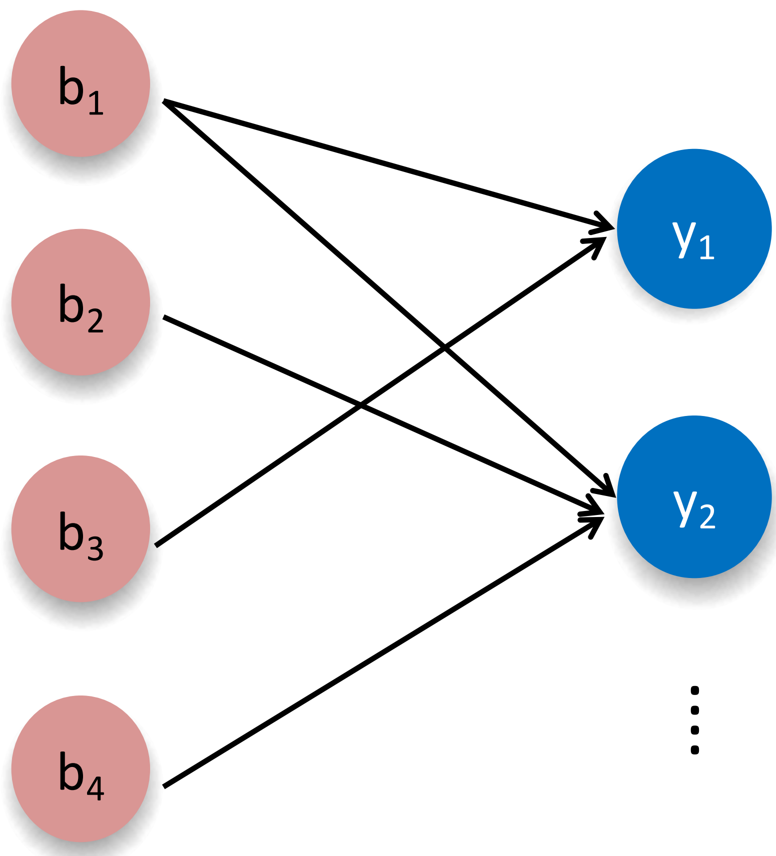


Belief Propagation enables the reader to decode quickly

- Received noisy symbols $\boldsymbol{y} = \boldsymbol{DHb} + \boldsymbol{n}$
- Find binary vector $\boldsymbol{b}$ that minimizes the deviation metric $E(\boldsymbol{b}) = \|\boldsymbol{DHb} - \boldsymbol{y}\|^2$
- Iterative Bit Flipping Decoder

Iterative Bit Flipping Example

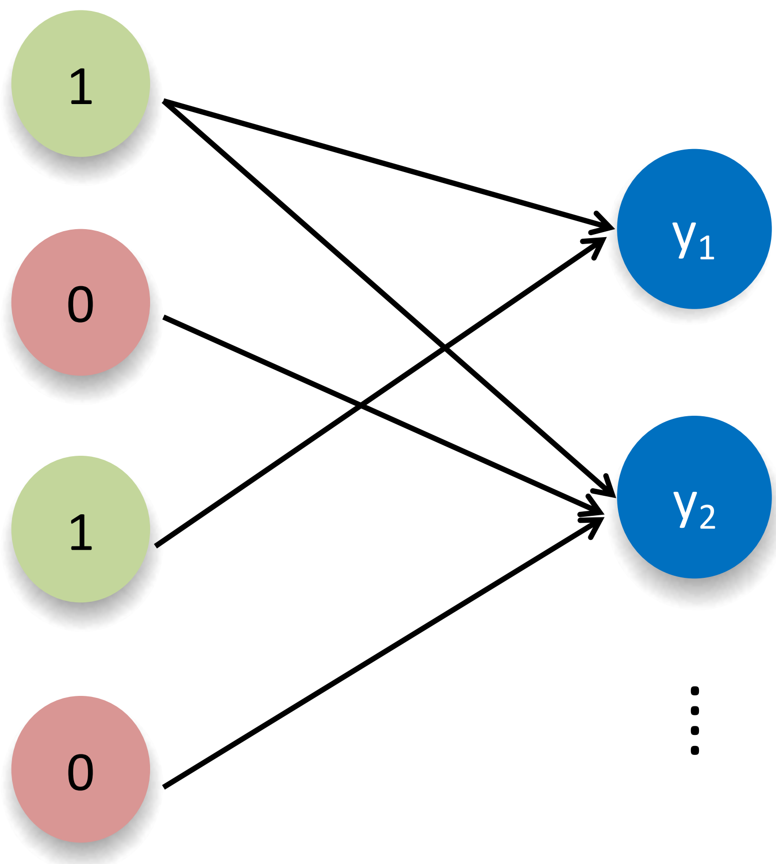
Example:



Iterative Bit Flipping Example

Example: Actual bits $\boldsymbol{b}$ = [1 0 1 0]

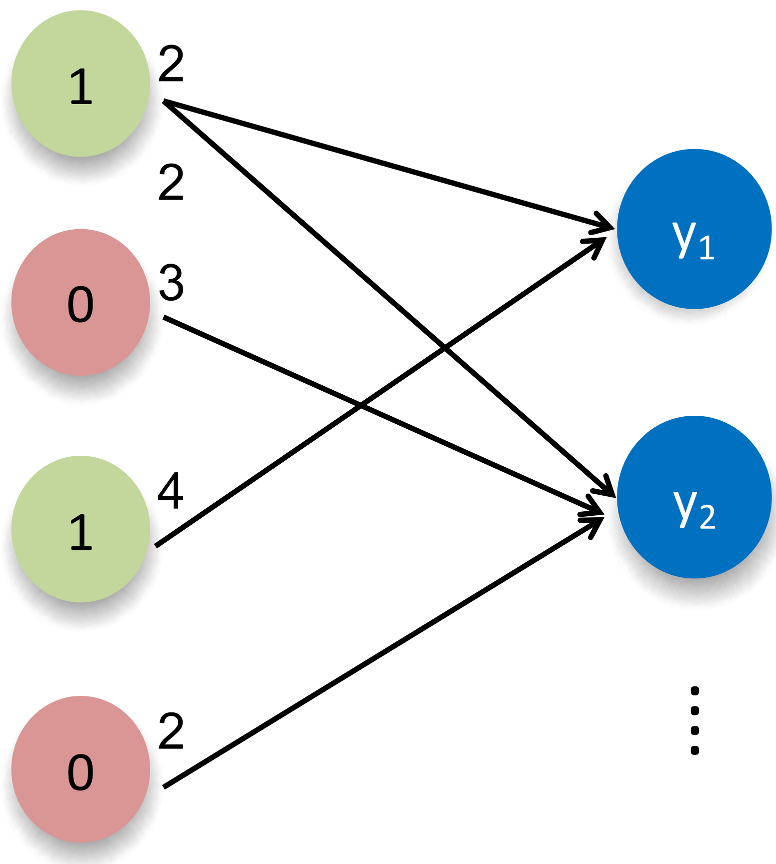
Channels $\boldsymbol{H}$ = [2 3 4 2]



Iterative Bit Flipping Example

Example: Actual bits $\boldsymbol{b}$ = [1 0 1 0]

Channels $\boldsymbol{H}$ = [2 3 4 2]

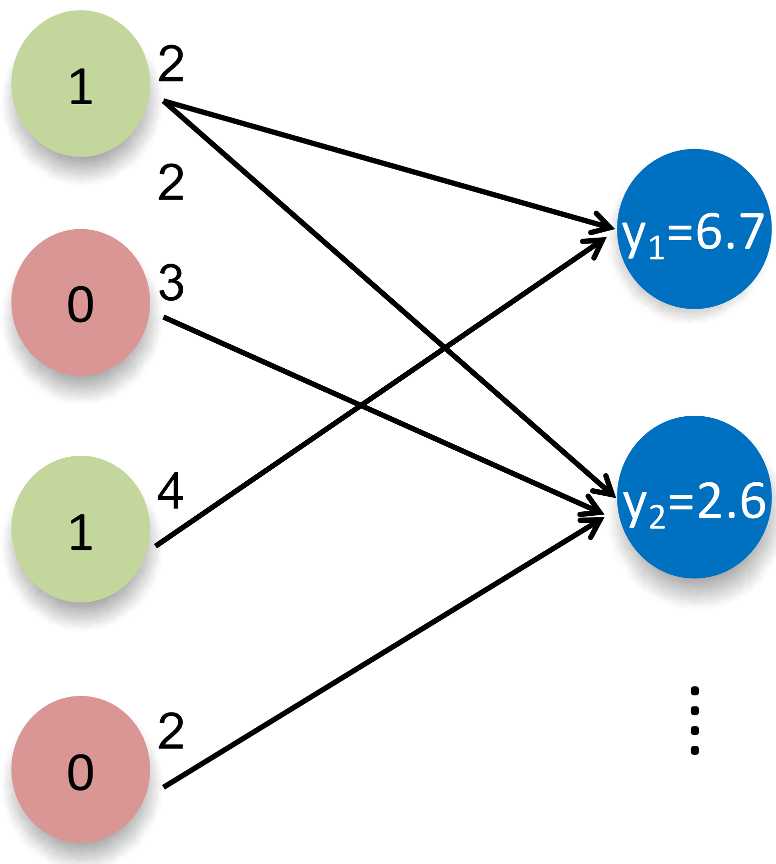


Iterative Bit Flipping Example

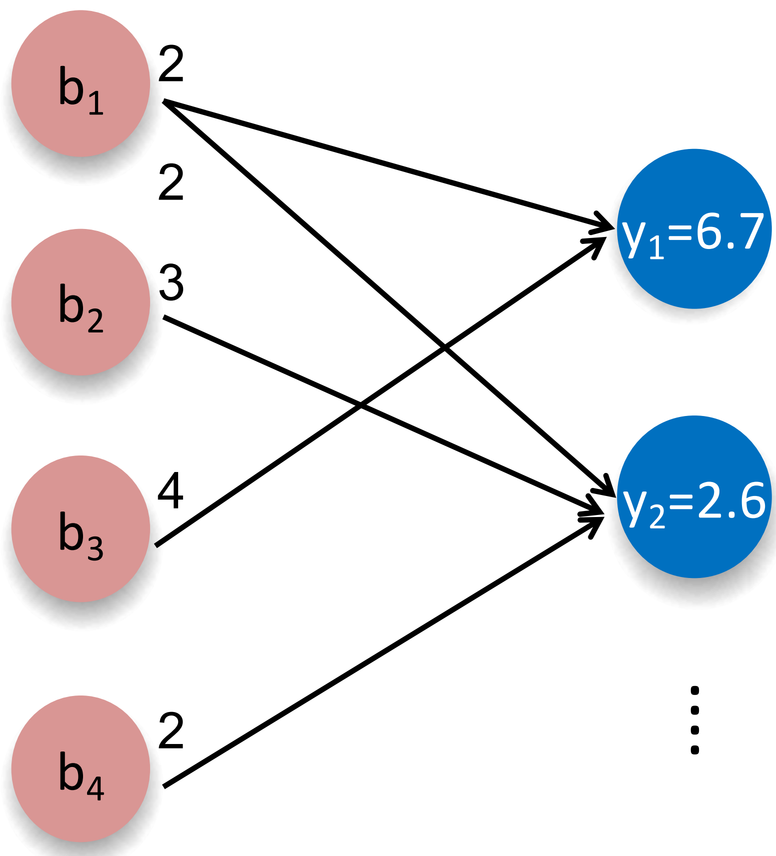
Example: Actual bits $\boldsymbol{b}$ = [1 0 1 0]

Channels $\boldsymbol{H}$ = [2 3 4 2]

Received noisy symbols $\boldsymbol{y}$ = [6.7 2.6]

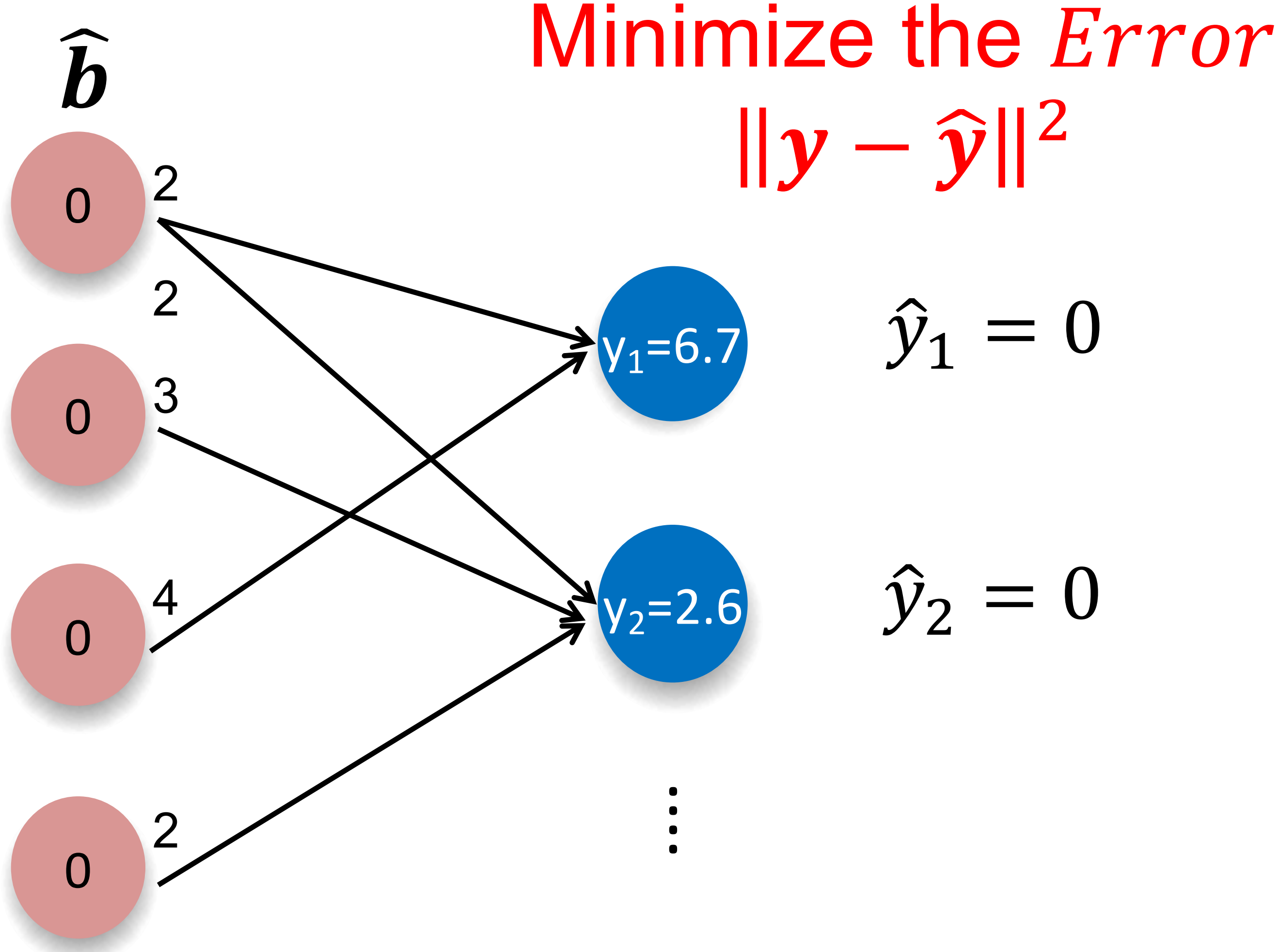


Iterative Bit Flipping Decoder



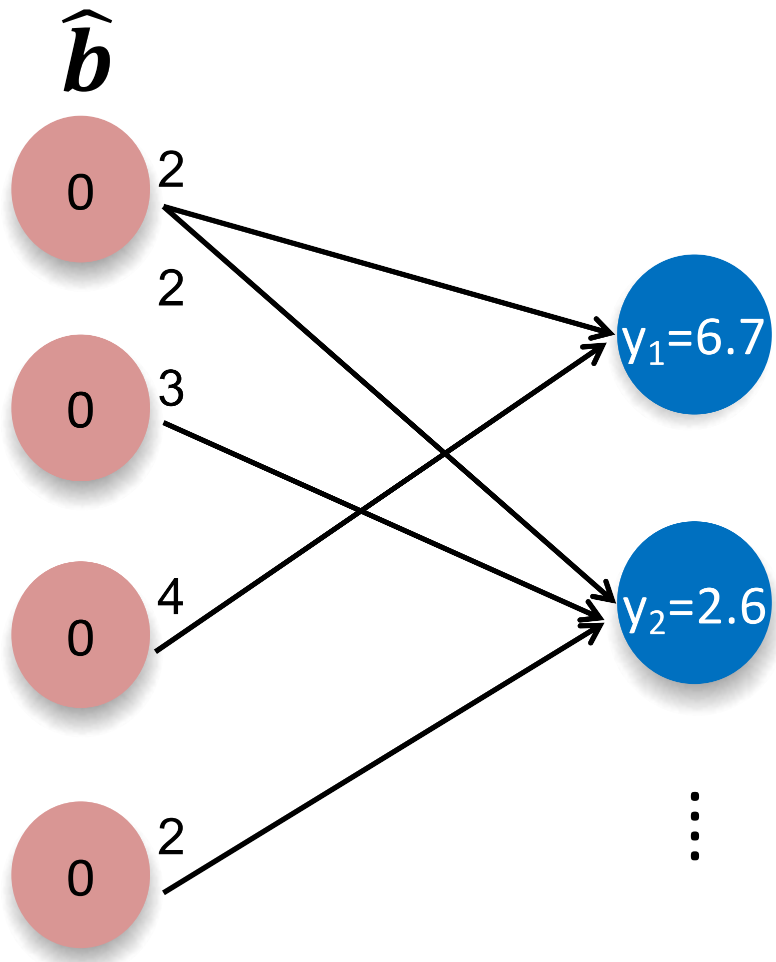
Iterative Bit Flipping Example

- Randomly initializing $\hat{\boldsymbol{b}}$



Iterative Bit Flipping Example

In what order should we flip the bits?



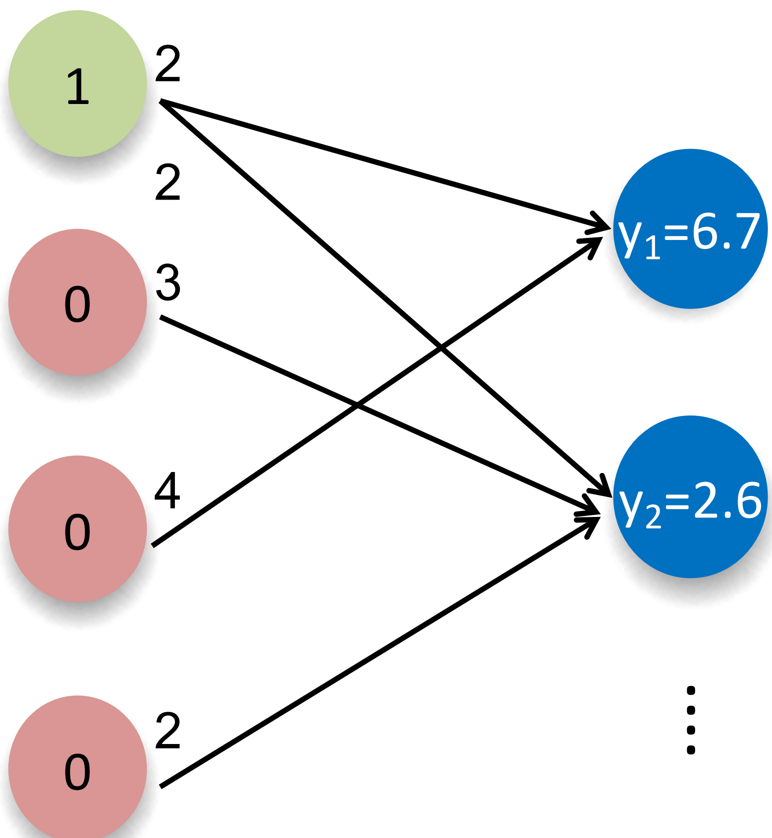
$\hat{y}_1 = 0$

$\hat{y}_2 = 0$

Iterative Bit Flipping Example

- If flipping bit 1

$\Delta Error$ = **-30**



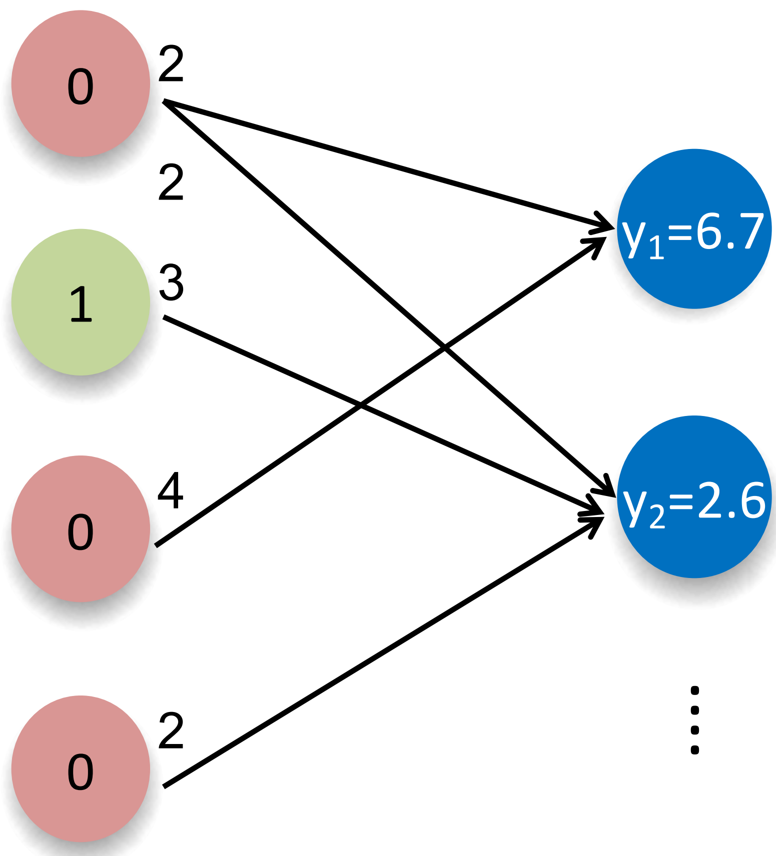
$\hat{y}_1 = 2$

$\hat{y}_2 = 2$

Iterative Bit Flipping Example

- If flipping bit 2

$\Delta Error$ **= -7**



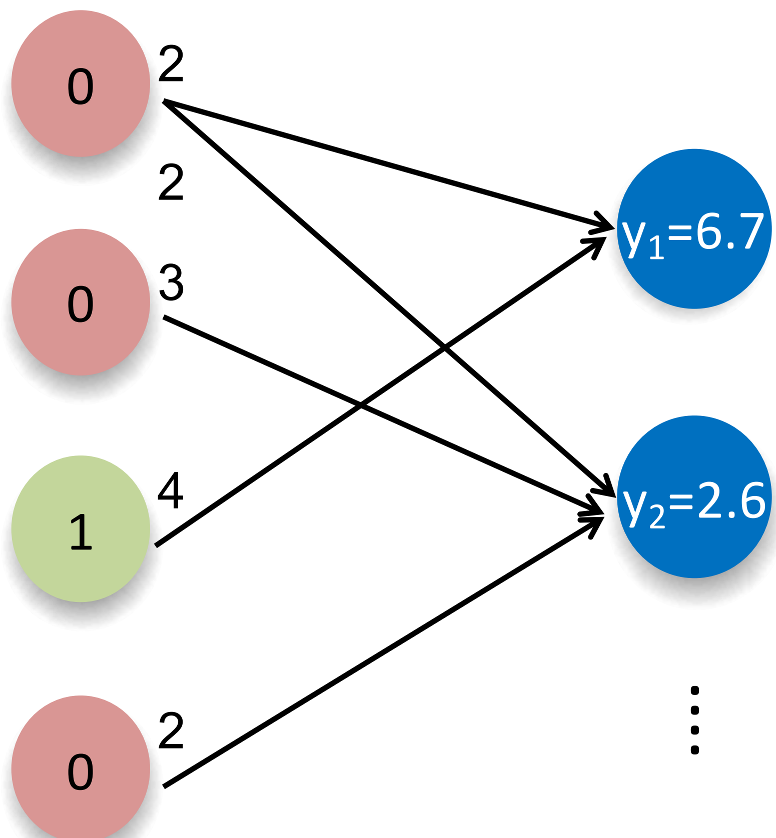
$\hat{y}_1 = 0$

$\hat{y}_2 = 3$

Iterative Bit Flipping Example

- If flipping bit 3

$\Delta Error$ **= -38**

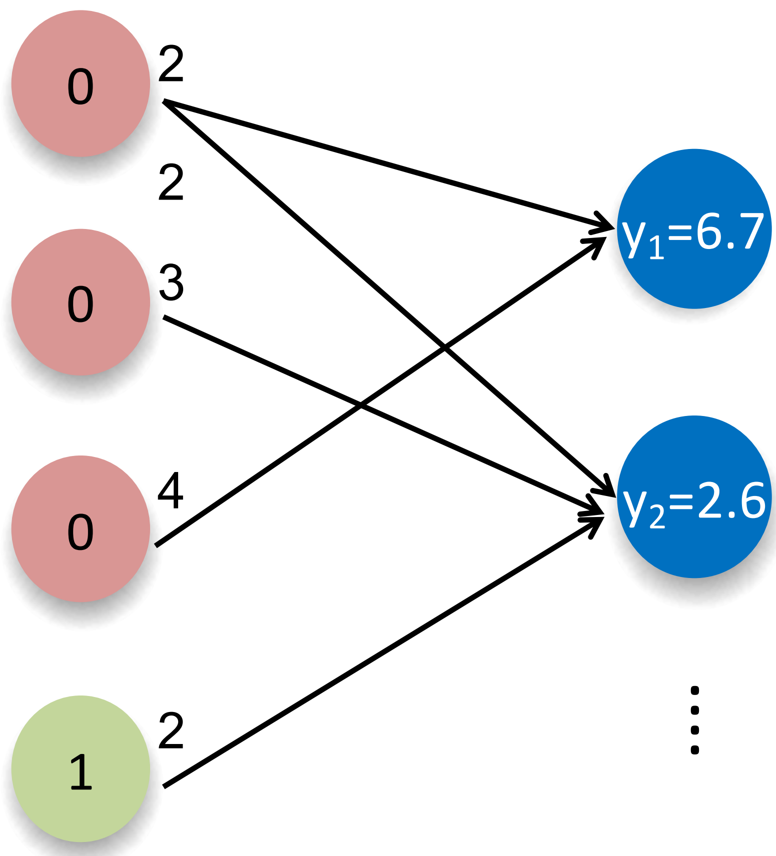


$\hat{y}_1 = 4$

$\hat{y}_2 = 0$

Iterative Bit Flipping Example

- If flipping bit 4



$\hat{y}_1 = 0$

$\hat{y}_2 = 2$

$\Delta Error$ **= -7**

Iterative Bit Flipping Example

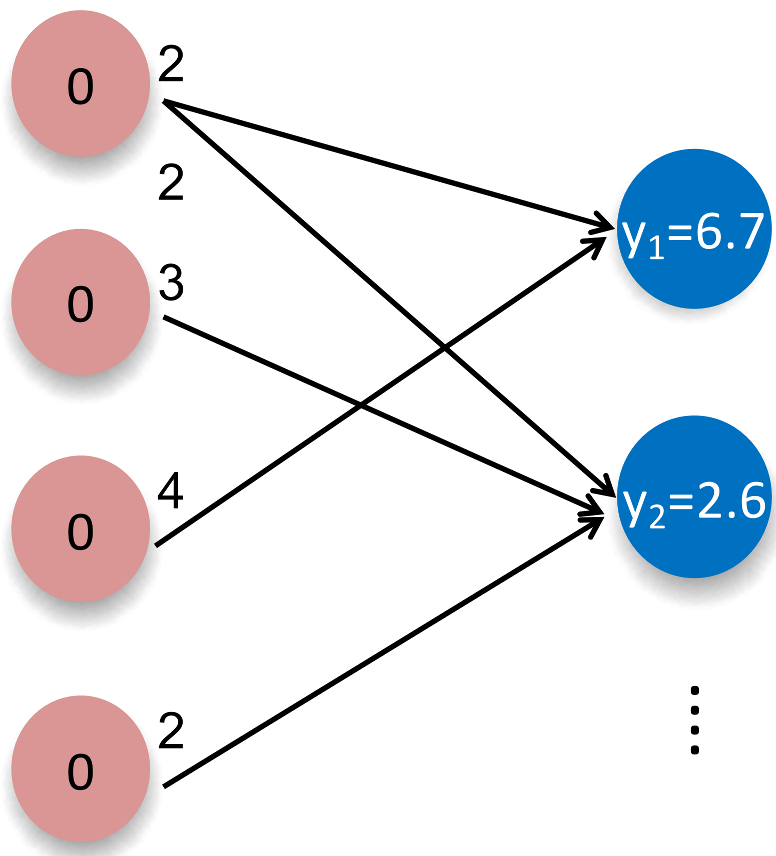
Flip bit that gives biggest reduction in Error

$\Delta Error$ = **-30**

$\Delta Error$ = **-7**

$\Delta Error$ = **-38**

$\Delta Error$ = **-7**



Iterative Bit Flipping Example

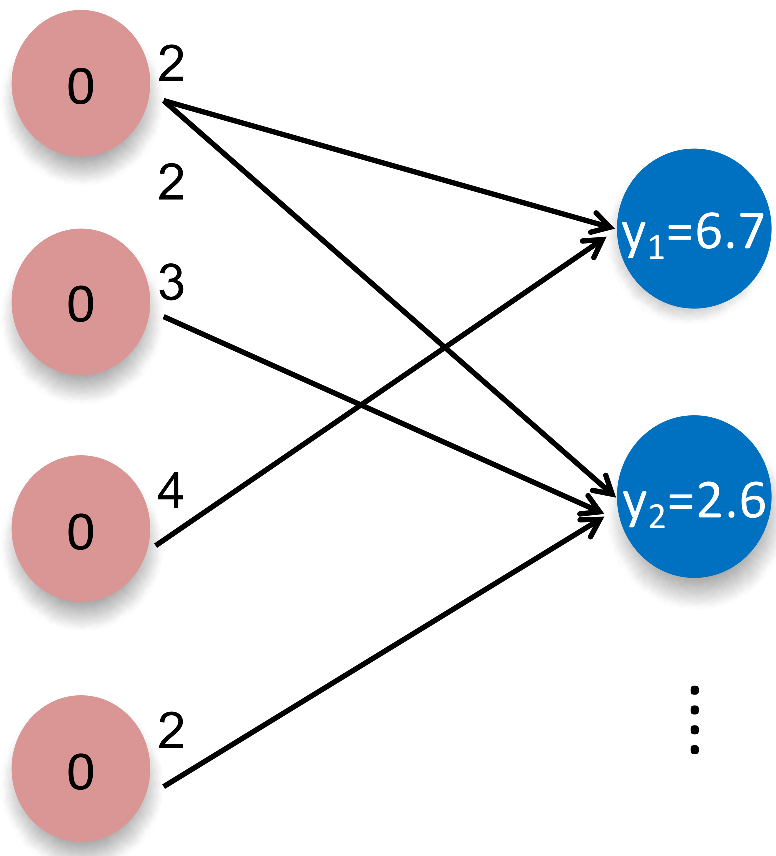
Flip bit that gives biggest reduction in Error

$\Delta Error$ = **-30**

$\Delta Error$ = **-7**

$\boxed{\Delta Error = \mathbf{-38}}$

$\Delta Error$ = **-7**



Iterative Bit Flipping Example

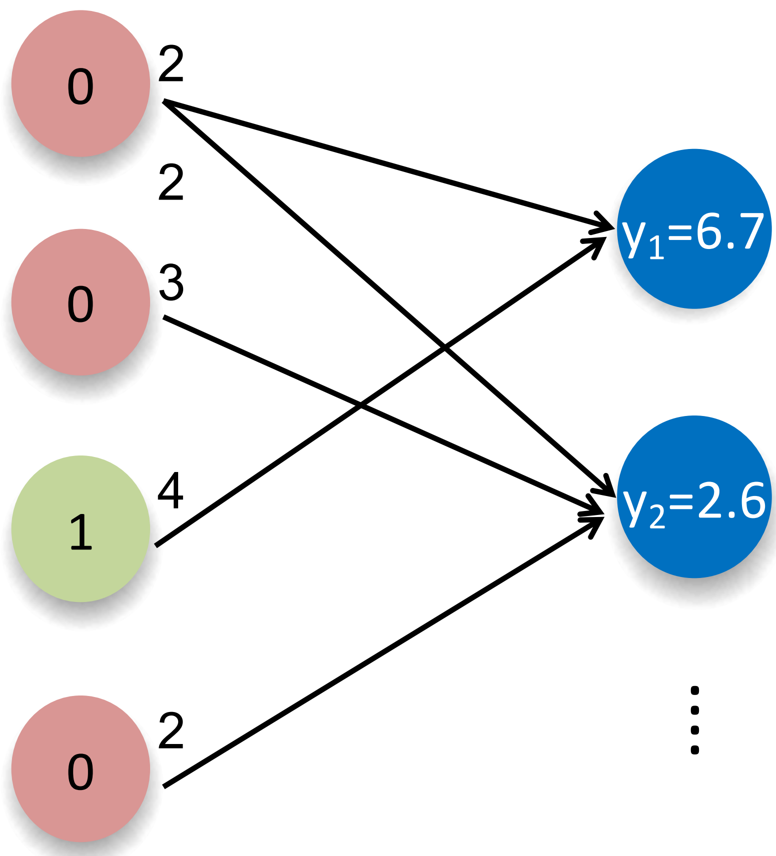
Flip bit that gives biggest reduction in Error

$\Delta Error$ = **-30**

$\Delta Error$ = **-7**

$\boxed{\Delta Error = \mathbf{-38}}$

$\Delta Error$ = **-7**



Iterative Bit Flipping Example

Flip bit that gives biggest reduction in Error

Minimize the $Error$

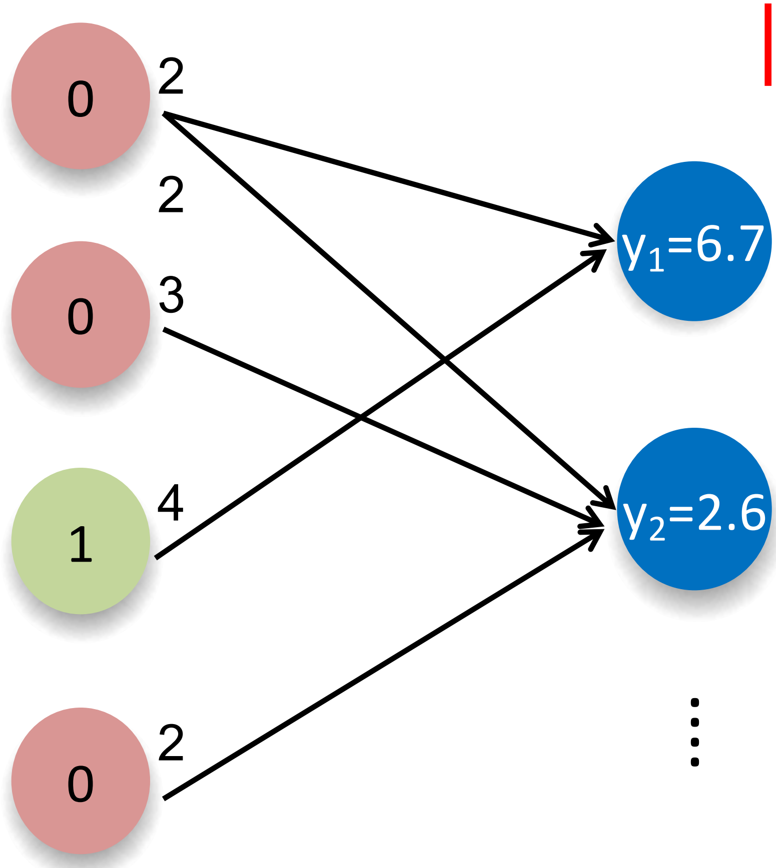
$$\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2$$

$\Delta Error$ = **-30**

$\Delta Error$ = **-7**

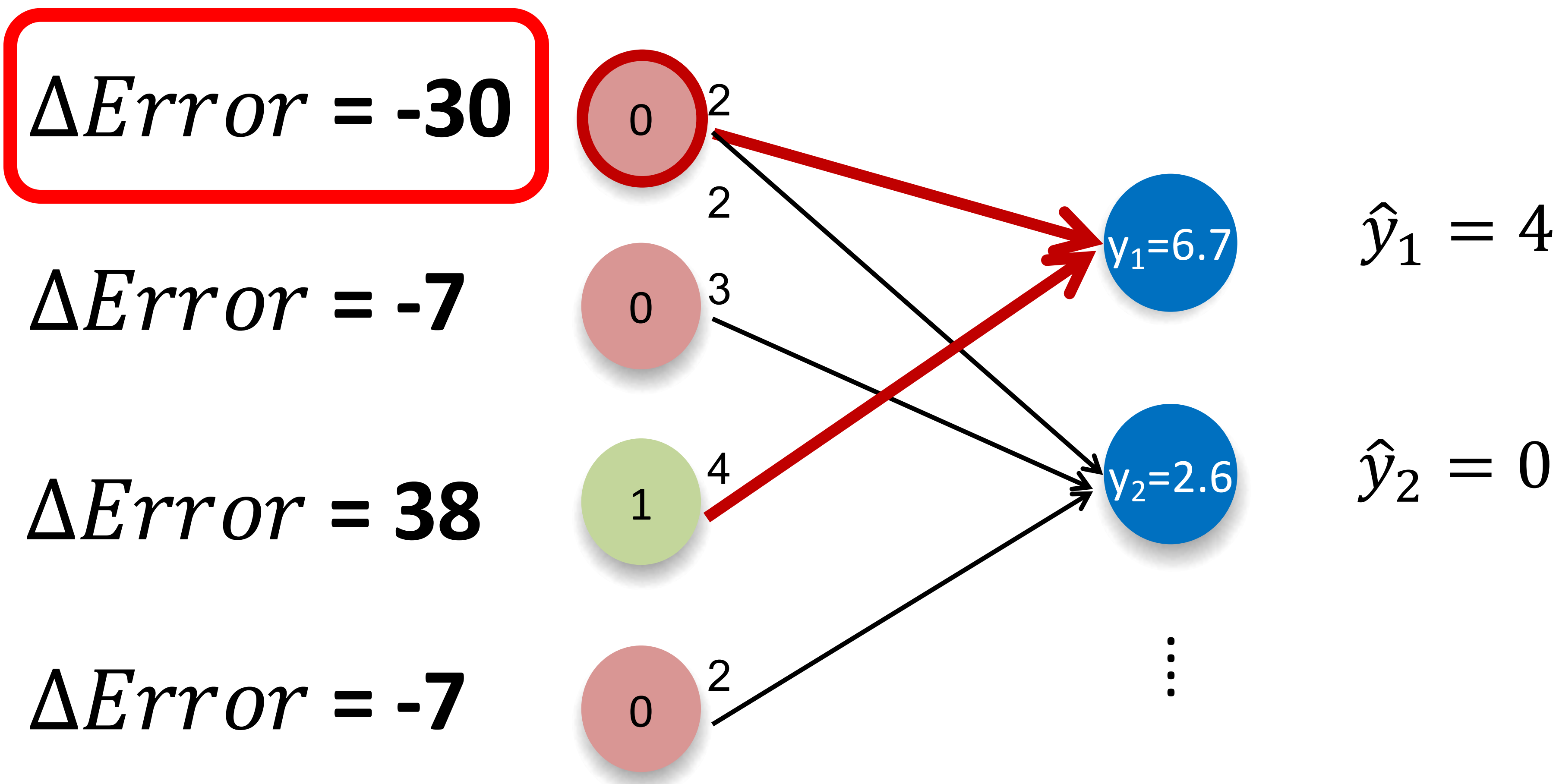
$\boxed{\Delta Error = \mathbf{38}}$

$\Delta Error$ = **-7**



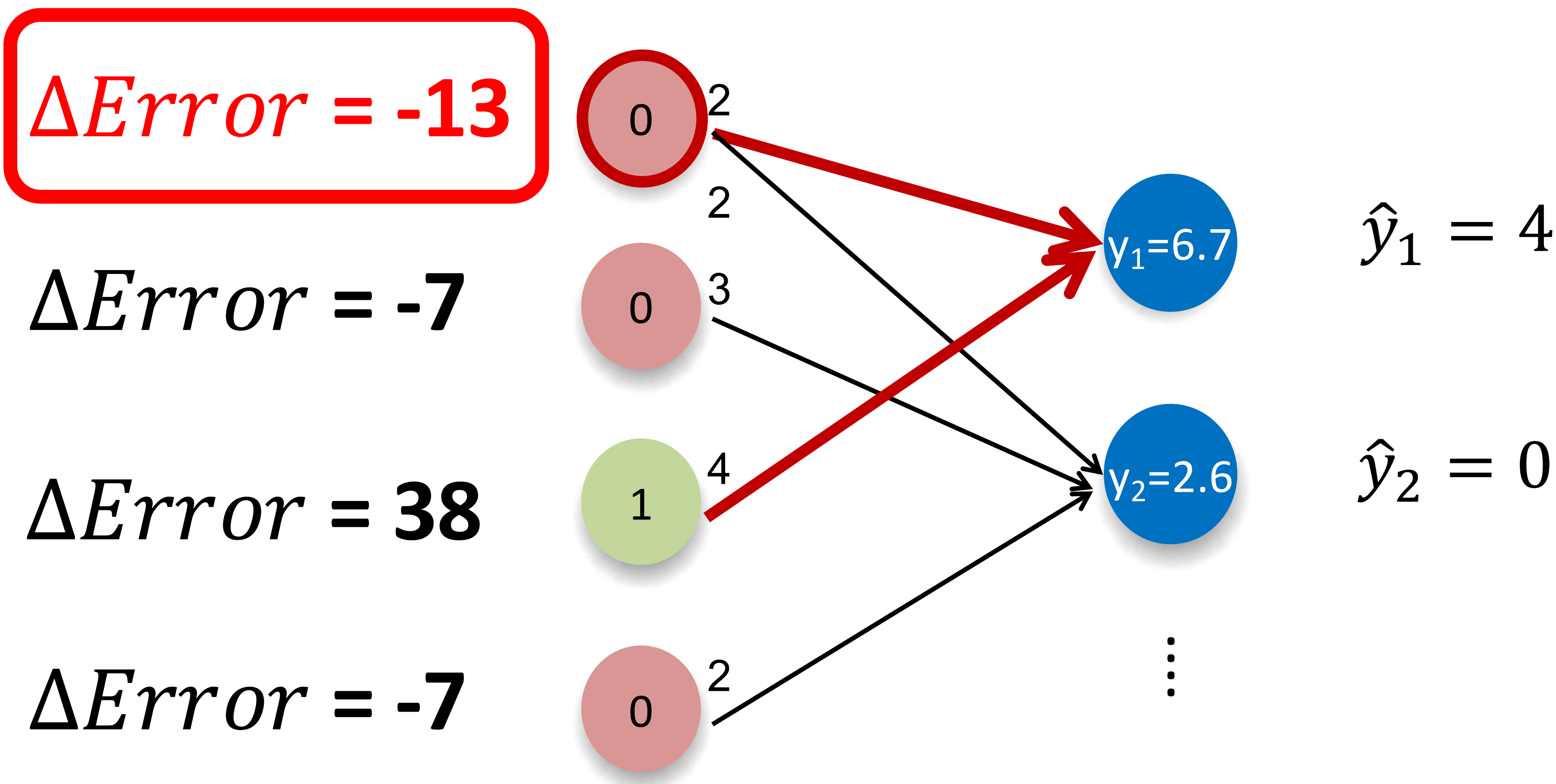
Iterative Bit Flipping Example

Update ΔE only for colliding nodes

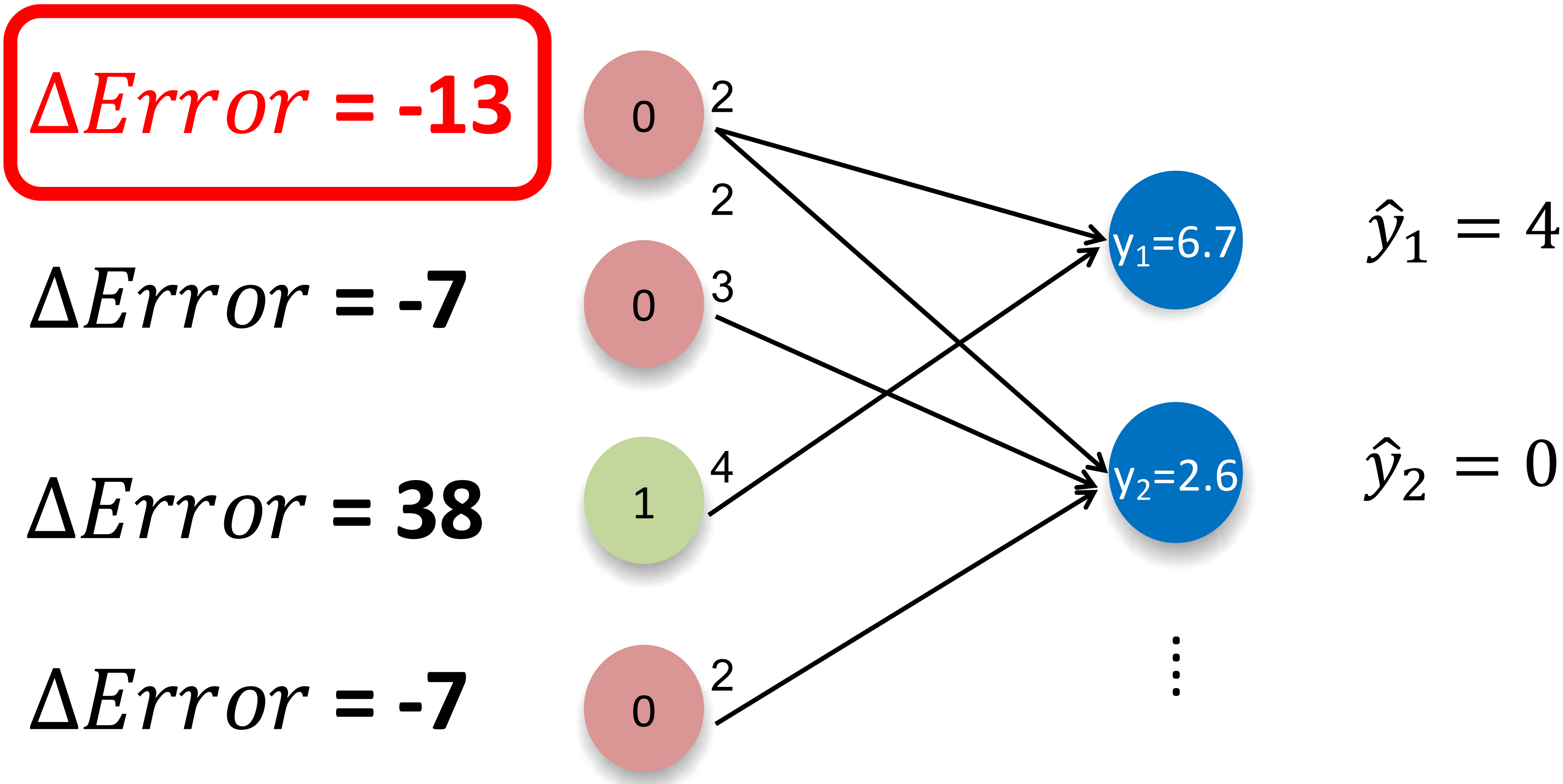


Iterative Bit Flipping Example

Update ΔE only for colliding nodes



Iterative Bit Flipping Example



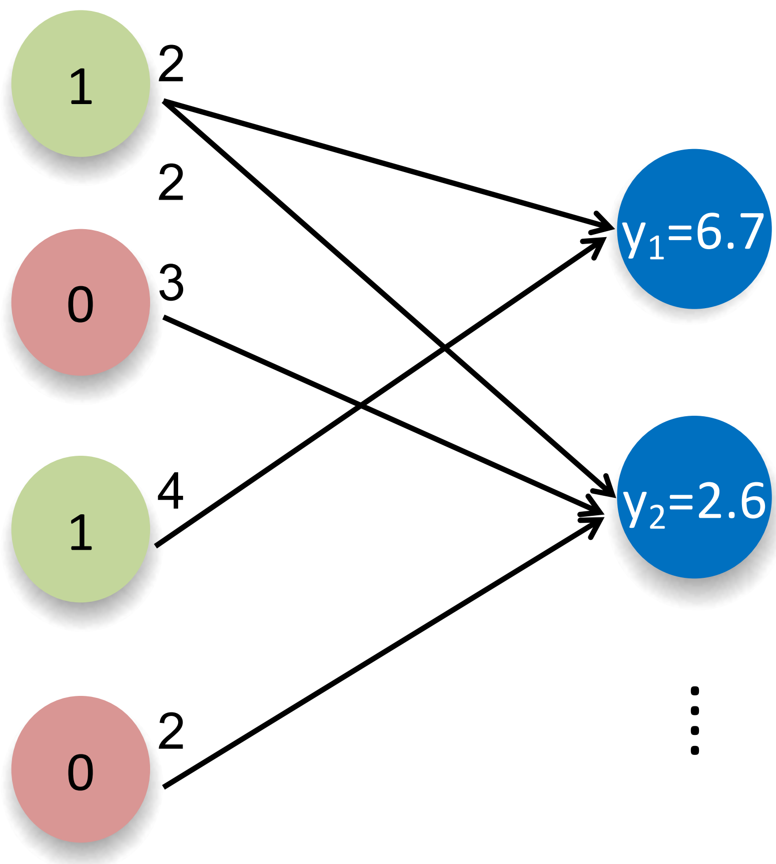
Iterative Bit Flipping Example

$\boxed{\Delta Error = \mathbf{13}}$

$\Delta Error = \mathbf{-7}$

$\Delta Error = \mathbf{38}$

$\Delta Error = \mathbf{-7}$



$$\hat{y}_1 = 6$$

$$\hat{y}_2 = 2$$

Iterative Bit Flipping Example

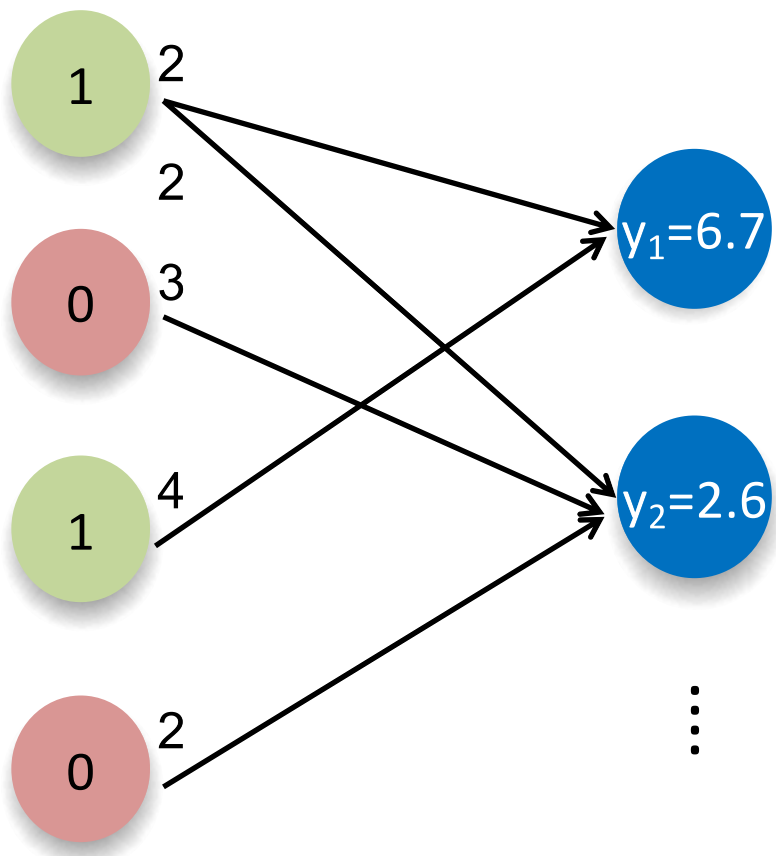
- No further reduction in $Error$ => Terminates
- $\hat{\boldsymbol{b}} = [\,1\ 0\ 1\ 0\,] = \boldsymbol{b}$ (actual bits)

$\Delta Error > \mathbf{0}$

$\Delta Error > \mathbf{0}$

$\Delta Error > \mathbf{0}$

$\Delta Error > \mathbf{0}$



$\hat{y}_1 = 6$

$\hat{y}_2 = 2$

Research Platforms

- RFID tag: passive stickers

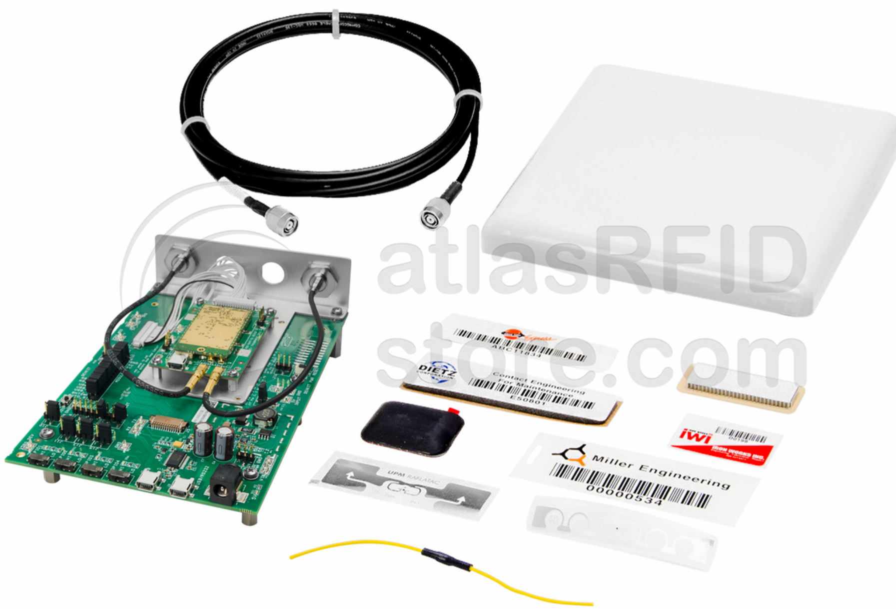
Research Platforms

- RFID tag: computational RFIDs

- MSP430 Microcontroller
  - 8KB RAM + 116KB Flash + 12 bit ADC/DAC
- Sensors:
  - Accelerometer + temperature + voltage + external sensors

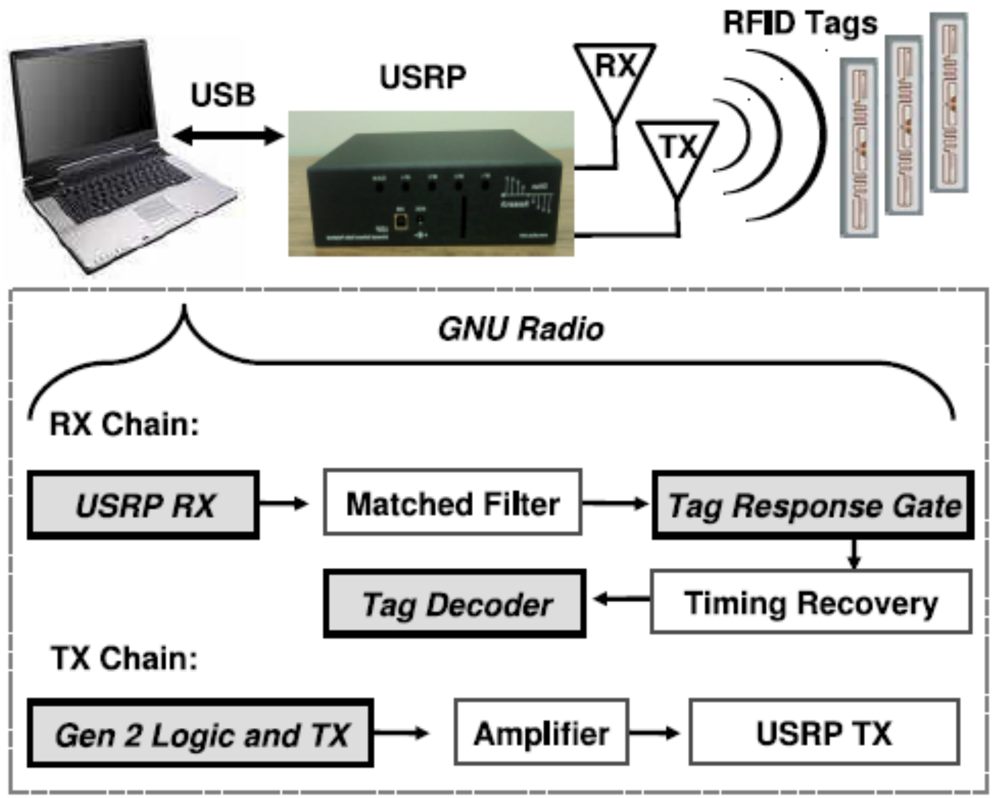
Research Platforms

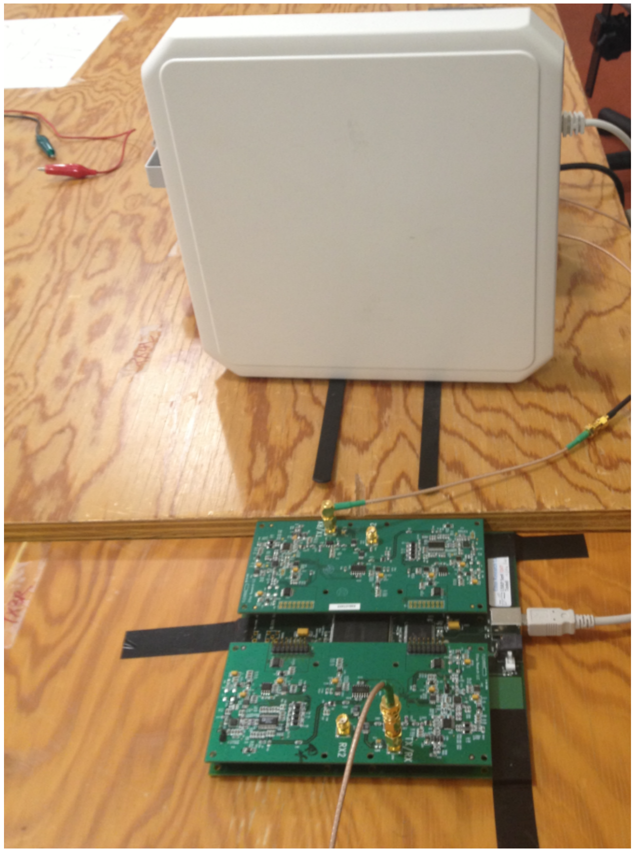
- Reader: Think magic/Impinj



Research Platforms

- Reader: software radio based Gen-2 reader





Conclusion

- Many applications for low power networks.
- Nodes need to be very simple (low cost, low power) → cannot have advanced functionalities
- Need new research ideas that can enable advanced protocol.
- What would you do if I give you so many RFIDs?