

# **Fault-Tolerant Discrete-Time LTI Filters**

**Christoforos Hadjicostis**

Department of Electrical and Computer Engineering  
and Coordinated Science Laboratory

University of Illinois at Urbana-Champaign

## FAULT-TOLERANT SYSTEMS

### **Fault tolerance describes ability to**

- Withstand internal failures
- Produce desirable overall “behavior” (e.g., output)

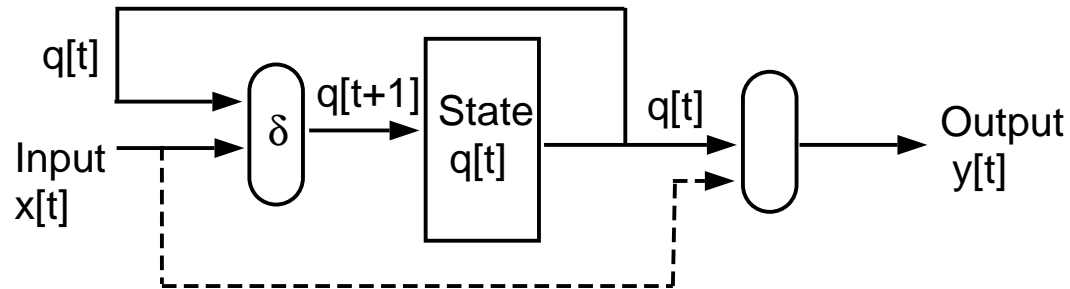
### **Necessary or desirable in**

- Life-threatening circumstances (military, transportation, medical)
- Systems in inaccessible environments (space missions)
- Reliable systems from unreliable components  
(faster, less expensive, less power)

## BACKGROUND: RESEARCH ON FAULT TOLERANCE

- Communication systems (channel noise, error-correcting codes)
- Computational circuits (hardware failures, modular redundancy)
  - Each component fails with *constant* probability
  - Earlier work by von Neumann, Shannon, Winograd, Elias and others; later work by Pippenger, Hajek, Feder, Gács
- Special-purpose systems (Algorithm-Based Fault Tolerance)
  - Protect against a *fixed* number of failures (e.g., a single failure)
  - Recent work by Abraham et al., Redinbo, Musicus, Beckmann, Hadjicostis
- Networked discrete-event systems (link or node failures, failure identification, reconfiguration/rerouting)

## DISCRETE-TIME DYNAMIC SYSTEMS

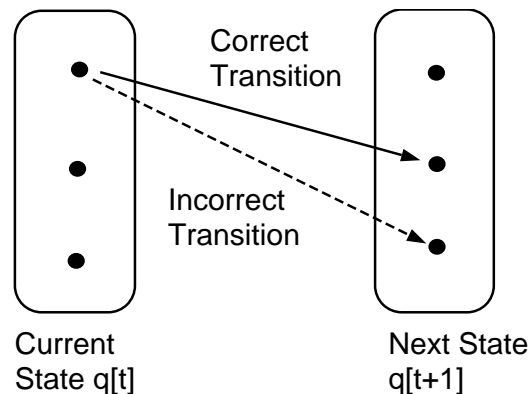


State Evolution:  $q[t + 1] = \delta(q[t], x[t])$

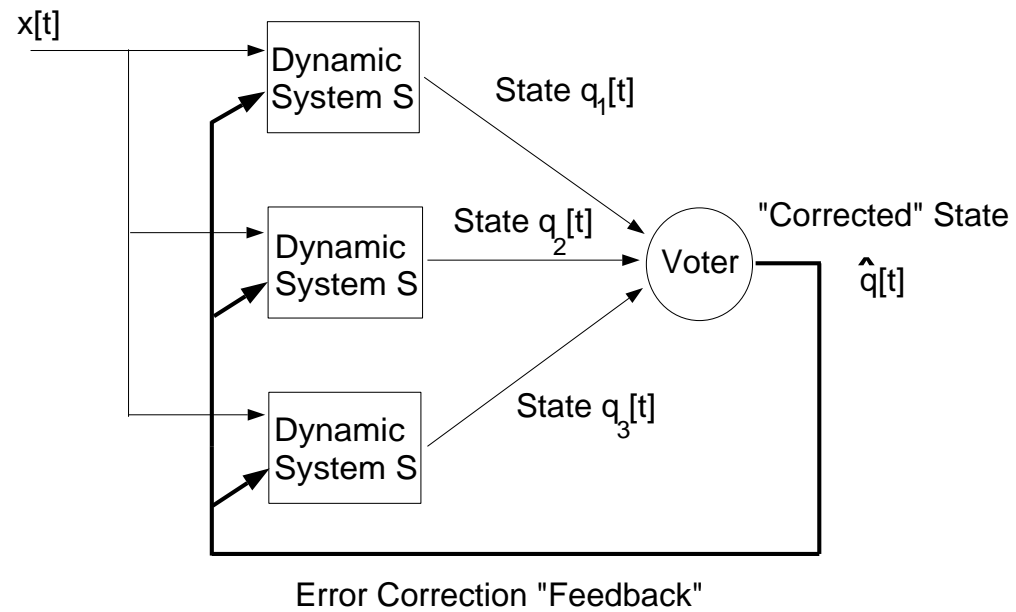
Output Equation:  $y[t] = \lambda(q[t], x[t])$  or  $\lambda(q[t])$

**Examples:** Digital filters, encoders/decoders, computer simulations

**State transition failures:**



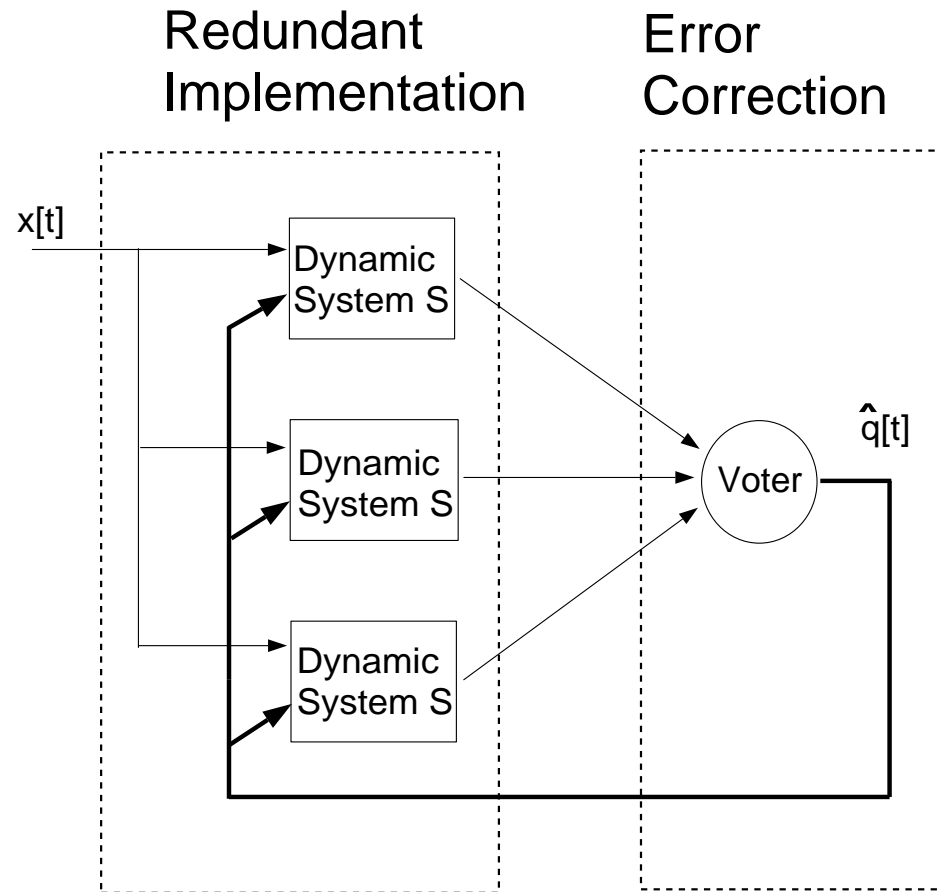
## TRADITIONAL APPROACH: MODULAR REDUNDANCY (VON NEUMANN, 1956)



### Problems:

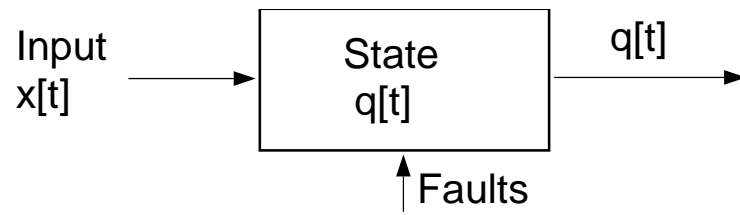
- Replication
- Voter failures

## AVOIDING REPLICATION

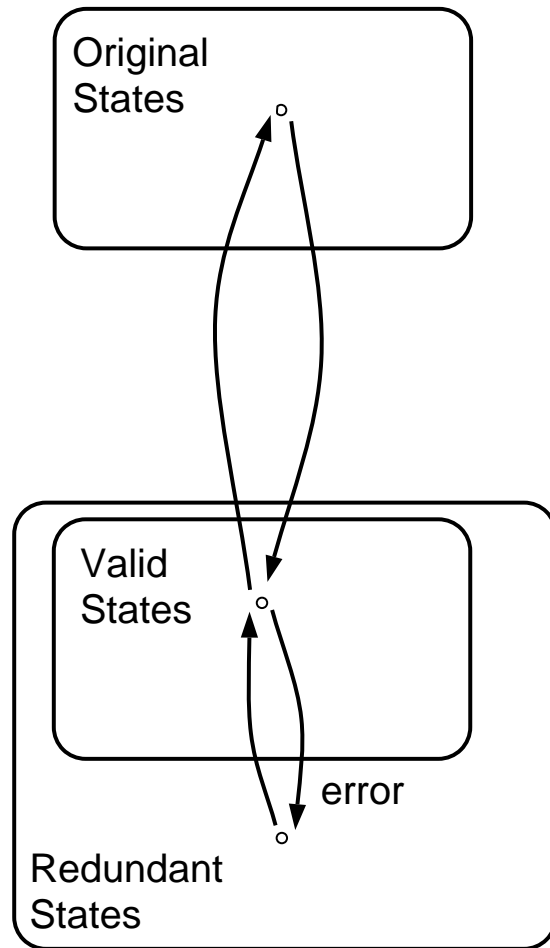
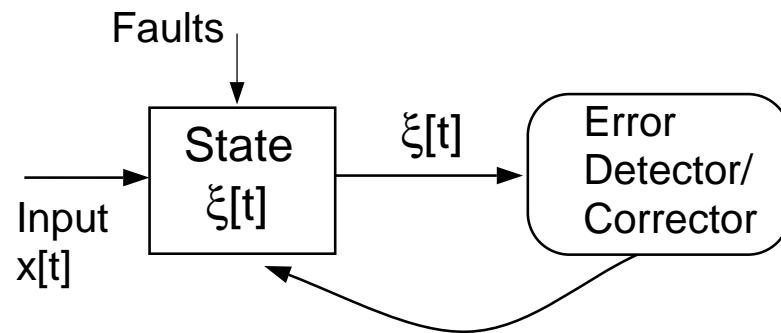


How can we avoid replication and minimize redundant hardware?

# REDUNDANT IMPLEMENTATIONS



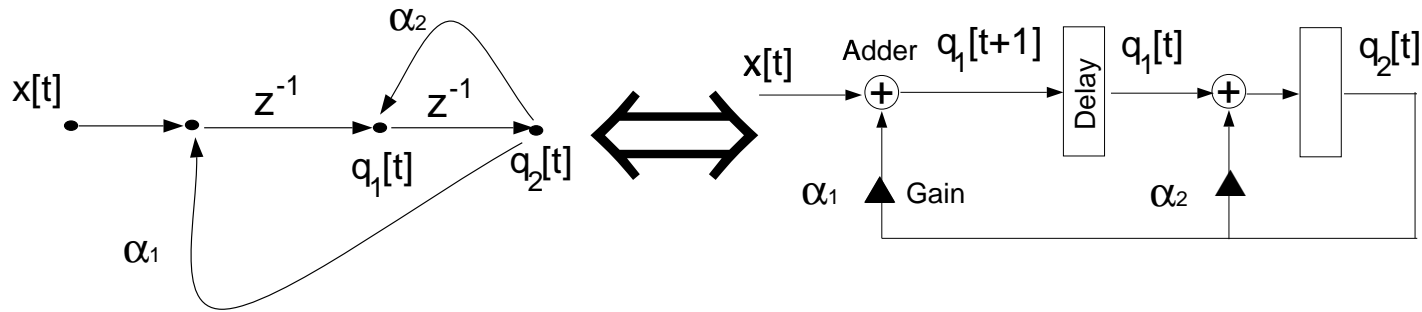
Replace with a larger dynamic system:



## DISCRETE-TIME LINEAR TIME-INVARIANT (LTI) DYNAMIC SYSTEMS

- Hardware implementation, error model
  - Reflection of hardware implementation and failures into design framework
- Characterization of redundant implementations
  - Standard form of redundant implementations
  - “Equivalent” redundant implementations
  - Non-zero redundant dynamics and coupling
- Generalizes/combines modular redundancy and checksum schemes
- New possibilities:
  - Exploit system dynamics/coupling to minimize redundant hardware
  - Development of linear coding schemes with “memory”
  - Reconfigurable decoding schemes

## HARDWARE IMPLEMENTATIONS OF LTI DYNAMIC SYSTEMS

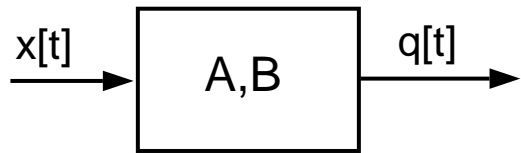


**State Evolution of LTI System:**  $\mathbf{q}[t + 1] = \mathbf{A}\mathbf{q}[t] + \mathbf{B}\mathbf{x}[t]$

$$\mathbf{q}[t] \equiv \begin{bmatrix} q_1[t] \\ q_2[t] \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & \alpha_1 \\ 1 & \alpha_2 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

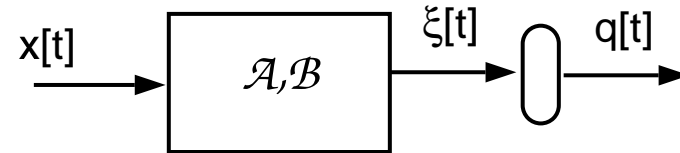
- Given  $\mathbf{A}$ ,  $\mathbf{B}$  there exist many realizations (*signal flow graphs*)
- Focus on signal flow graphs with *delay-free paths of length one*:
  - Entries in  $\mathbf{A}$ ,  $\mathbf{B}$  are values of gain elements
  - State variables in  $\mathbf{q}[t + 1]$  are generated using *separate* gains and adders
  - A fault in an adder or in a gain element corrupts a *single* state variable

## REDUNDANT LTI DYNAMIC SYSTEMS



Original System:

$$\mathbf{q}[t + 1] = \mathbf{A}\mathbf{q}[t] + \mathbf{B}\mathbf{x}[t]$$



Redundant System:

$$\xi[t + 1] = \mathcal{A}\xi[t] + \mathcal{B}\mathbf{x}[t]$$

- **Concurrent Simulation:**  $\mathbf{q}[t] = \mathbf{L}\xi[t]$  (1)
- **State Constraints:**  $\xi[t] = \mathbf{G}\mathbf{q}[t]$  (2)
- **Fault Detection:** When  $\xi$  is *not* in column space of  $\mathbf{G}$ , or  $\mathbf{P}\xi[\cdot] \neq \mathbf{0}$   
(where  $\mathbf{P}\mathbf{G} = \mathbf{0}$ )

◇ **Requirement:**  $\mathcal{A}$  and  $\mathcal{B}$  to be chosen so as to satisfy (1) and (2)

## CHARACTERIZATION OF REDUNDANT LTI DYNAMIC SYSTEMS

$$\left. \begin{array}{l} \text{Original System:} \\ \mathbf{q}[t + 1] = \mathbf{A}\mathbf{q}[t] + \mathbf{B}\mathbf{x}[t] \end{array} \right\} \begin{array}{l} \xi[\cdot] \xrightarrow{=\mathbf{G}\mathbf{q}[\cdot]} \\ \mathbf{q}[\cdot] \xleftarrow{=\mathbf{L}\xi[\cdot]} \end{array} \left\{ \begin{array}{l} \text{Redundant Implementation:} \\ \xi[t + 1] = \mathcal{A}\xi[t] + \mathcal{B}\mathbf{x}[t] \end{array} \right.$$

### Standard Redundant Implementations:

$(\mathcal{A}, \mathcal{B})$  is a redundant implementation for  $(\mathbf{A}, \mathbf{B})$  iff  $(\mathcal{A}, \mathcal{B})$  is similar to the following standard form:

$$\xi_\sigma[t + 1] = \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{A}_{12} \\ \mathbf{0} & \mathbf{A}_{22} \end{bmatrix}}_{\mathcal{A}_\sigma} \xi_\sigma[t] + \underbrace{\begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix}}_{\mathcal{B}_\sigma} \mathbf{x}[t]$$

for some matrices  $\mathbf{A}_{12}, \mathbf{A}_{22}$

**Specifically:** There exists invertible  $\mathcal{T}$  such that

$$\mathcal{A}_\sigma = \mathcal{T}^{-1} \mathcal{A} \mathcal{T}, \quad \mathcal{B}_\sigma = \mathcal{T}^{-1} \mathcal{B}, \quad \xi[\cdot] = \mathcal{T} \xi_\sigma[\cdot]$$

◇ **Related work:** Šiljak's "inclusion principle"

## NOTE ON SIMILARITY TRANSFORMATIONS

### Original System:

$$\begin{aligned}\xi[t + 1] &= \mathcal{A}\xi[t] + \mathcal{B}\mathbf{x}[t] \\ \mathbf{y}[t] &= \mathcal{C}\xi[t] + \mathcal{D}\mathbf{x}[t]\end{aligned}$$

### Similar System:

$$\begin{aligned}\xi'[t + 1] &= (\mathcal{T}^{-1}\mathcal{A}\mathcal{T})\xi'[t] + (\mathcal{T}^{-1}\mathcal{B})\mathbf{x}[t] \\ \mathbf{y}[t] &= (\mathcal{C}\mathcal{T})\xi'[t] + \mathcal{D}\mathbf{x}[t]\end{aligned}$$

Both systems have:

- Same input-output behavior
- Relationship between state vectors:

$$\xi'[t] = \mathcal{T}^{-1}\xi[t]$$

## EXAMPLE 1: TRIPLE MODULAR REDUNDANCY

$$\xi[t+1] \equiv \begin{bmatrix} \mathbf{q}^1[t+1] \\ \mathbf{q}^2[t+1] \\ \mathbf{q}^3[t+1] \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A} \end{bmatrix} \xi[t] + \begin{bmatrix} \mathbf{B} \\ \mathbf{B} \\ \mathbf{B} \end{bmatrix} \mathbf{x}[t]$$

Possible decoding, encoding and parity check matrices are:

$$\mathbf{L} = \begin{bmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \mathbf{I}_n \\ \mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} -\mathbf{I}_n & \mathbf{I}_n & \mathbf{0} \\ -\mathbf{I}_n & \mathbf{0} & \mathbf{I}_n \end{bmatrix}$$

Similar *standard* redundant implementation:

$$\xi_\sigma[t+1] = \left[ \begin{array}{c|cc} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A} \end{array} \right] \xi_\sigma[t] + \left[ \begin{array}{c} \mathbf{B} \\ \mathbf{0} \\ \mathbf{0} \end{array} \right] \mathbf{x}[t]$$

Corresponding *standard* decoding, encoding and parity check matrices:

$$\underbrace{\mathbf{L}_\sigma}_{\mathbf{L}\mathcal{T}} = \begin{bmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \underbrace{\mathbf{G}_\sigma}_{\mathcal{T}^{-1}\mathbf{G}} = \begin{bmatrix} \mathbf{I}_n \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad \underbrace{\mathbf{P}_\sigma}_{\mathbf{P}\mathcal{T}} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_n \end{bmatrix}$$

## EXAMPLE 2: FAULT-TOLERANT STATE FILTERS (CHATTERJEE ET AL., 1993)

Encode using a matrix  $\mathbf{C}$  (of dimension  $d \times n$ ):

$$\xi[t+1] = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{CA} & \mathbf{0} \end{bmatrix} \xi[t] + \begin{bmatrix} \mathbf{B} \\ \mathbf{CB} \end{bmatrix} \mathbf{x}[t]$$

State constraints:

$$\xi[t] = \mathbf{G}\mathbf{q}[t] = \begin{bmatrix} \mathbf{I}_n \\ \mathbf{C} \end{bmatrix} \mathbf{q}[t] = \begin{bmatrix} \mathbf{q}[t] \\ \mathbf{C}\mathbf{q}[t] \end{bmatrix}$$

Similar *standard* redundant implementation (use  $\mathcal{T} = \begin{bmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{C} & \mathbf{I}_d \end{bmatrix}$ ,  $\xi = \mathcal{T}\xi_\sigma[t]$ ):

$$\xi_\sigma[t+1] = \left[ \begin{array}{c|c} \mathbf{A} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array} \right] \xi_\sigma[t] + \left[ \begin{array}{c} \mathbf{B} \\ \mathbf{0} \end{array} \right] \mathbf{x}[t]$$

## SYSTEMATIC CONSTRUCTION OF REDUNDANT IMPLEMENTATIONS

1. Start from standard redundant implementation

2. Select a *suitable* parity check matrix  $\mathbf{P}$

$$\text{Syndrome} \equiv \mathbf{s}[t] = \mathbf{P}\xi_f[t] = \mathbf{P}(\xi[t] + \mathbf{e}[t]) = \mathbf{P}\mathbf{e}[t]$$

3. Choose  $\mathcal{T}$  such that  $\mathbf{P}\mathcal{T} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_d \end{bmatrix}$  (recall  $\mathbf{P}\mathcal{T} = \mathbf{P}_\sigma$ )

4. Develop a redundant implementation with

$$\mathcal{A} = \mathcal{T} \begin{bmatrix} \mathbf{A} & \mathbf{A}_{12} \\ \mathbf{0} & \mathbf{A}_{22} \end{bmatrix} \mathcal{T}^{-1}, \quad \mathcal{B} = \mathcal{T} \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{L} = \mathcal{T}^{-1} \underbrace{\begin{bmatrix} \mathbf{I}_n & \mathbf{0} \end{bmatrix}}_{\mathcal{L}_\sigma}, \quad \mathbf{G} = \mathcal{T} \underbrace{\begin{bmatrix} \mathbf{I}_n \\ \mathbf{0} \end{bmatrix}}_{\mathbf{G}_\sigma}$$

◇ **Advantage:** Use the flexibility in the choice of  $\mathbf{A}_{12}$  and  $\mathbf{A}_{22}$

## APPLICATION: SINGLE ERROR CORRECTION (1)

**Goal:** Provide *single-error correction* capability to

$$\mathbf{q}[t+1] = \underbrace{\begin{bmatrix} .2 & 0 & 0 & 0 \\ 0 & .5 & 0 & 0 \\ 0 & 0 & .1 & 0 \\ 0 & 0 & 0 & .6 \end{bmatrix}}_{\mathbf{A}} \mathbf{q}[t] + \underbrace{\begin{bmatrix} 2 \\ -1 \\ 1 \\ -2 \end{bmatrix}}_{\mathbf{b}} x[t]$$

**Step 1:** Consider a *standard* redundant implementation ( $\mathbf{A}_{12} = \mathbf{0}$ )

$$\xi_{\sigma}[t+1] = \left[ \begin{array}{cccc|ccc} .2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .6 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & .2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & .5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .1 \end{array} \right] \xi_{\sigma}[t] + \left[ \begin{array}{c} 2 \\ -1 \\ 1 \\ -2 \\ \hline 0 \\ 0 \\ 0 \end{array} \right] x[t]$$

with decoding, encoding and parity check matrices:

$$\mathbf{L}_{\sigma} = \begin{bmatrix} \mathbf{I}_4 & \mathbf{0} \end{bmatrix}, \quad \mathbf{G}_{\sigma} = \begin{bmatrix} \mathbf{I}_4 \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{P}_{\sigma} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_3 \end{bmatrix}$$

## APPLICATION: SINGLE ERROR CORRECTION (2)

**Step 2:** Desirable parity check matrix (Hamming code, locates single faults):

$$\mathbf{P} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

**Step 3:** Find  $\mathcal{T}$  such that  $\mathbf{P}\mathcal{T} = \mathbf{P}_\sigma$

**Step 4:** Obtain a similar (non-standard) redundant implementation:

$$\xi[t+1] = \begin{bmatrix} .2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .6 & 0 & 0 & 0 \\ 0 & -.3 & .1 & 0 & .2 & 0 & 0 \\ .3 & 0 & 0 & -.1 & 0 & .5 & 0 \\ -.1 & 0 & 0 & -.5 & 0 & 0 & .1 \end{bmatrix} \xi[t] + \begin{bmatrix} 2 \\ -1 \\ 1 \\ -2 \\ -2 \\ 1 \\ -1 \end{bmatrix} x[t]$$

◇ **Extensions:** Other error-correcting or real number codes for multiple faults



## EXTENSION 2: CHECKSUM SCHEMES WITH “MEMORY”

- *Temporal* failure corrupts the 4th state variable at time 0

$$\xi_f[0] = \xi[0] + \mathbf{e}[0] = \xi[0] + \begin{bmatrix} 0 \\ 0 \\ 0 \\ c \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad c \neq 0$$

- Syndrome at time step 0

$$\mathbf{s}[0] \equiv \mathbf{P}\xi_f[0] = c \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

- Syndrome at time step  $m$

$$\mathbf{s}[m] \equiv \mathbf{P}\xi[m] = c\mathbf{A}_{22}^m \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = c \begin{bmatrix} 0 \\ (.5)^m \\ (.1)^m \end{bmatrix} \quad \left( \text{recall that } A_{22} = \begin{bmatrix} .2 & 0 & 0 \\ 0 & .5 & 0 \\ 0 & 0 & .1 \end{bmatrix} \right)$$

### EXTENSION 3: RECONFIGURABLE DECODING (1)

The state evolution equation in the previous example was given by

$$\xi[t + 1] = \begin{bmatrix} .2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .6 & 0 & 0 & 0 \\ 0 & -.3 & .1 & 0 & .2 & 0 & 0 \\ .3 & 0 & 0 & -.1 & 0 & .5 & 0 \\ .1 & 0 & .2 & -.3 & 0 & 0 & .3 \end{bmatrix} \xi[t] + \begin{bmatrix} 3 \\ -1 \\ 7 \\ 0 \\ -9 \\ -2 \\ -10 \end{bmatrix} x[t]$$

#### Observations:

- Permanent fault in second state variable  $\xi_2$  at time step  $t$  propagates to fifth state variable ( $\xi_5$ ) at time step  $t + 1$
- Permanent fault in  $\xi_1$  at time step  $t$  propagates to variables  $\xi_6$  and  $\xi_7$  at time step  $t + 1$

### EXTENSION 3: RECONFIGURABLE DECODING (2)

Previously:

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & -1 & -1 & 0 \\ -1 & -1 & 0 & -1 \\ -1 & 0 & -1 & -1 \end{bmatrix}$$

Condition:  
 $\mathbf{L}\mathbf{G} = \mathbf{I}_4$

When  $\mathcal{A}(2, 2)$  is *permanently* corrupted, future  $\xi_2[\cdot]$  and  $\xi_5[\cdot]$  are invalid. Appropriate decoding and encoding mappings are as follows:

$$\mathbf{L}_r = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_r = \begin{bmatrix} 1 & 0 & 0 & 0 \\ * & * & * & * \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ * & * & * & * \\ -1 & -1 & 0 & -1 \\ -1 & 0 & -1 & -1 \end{bmatrix}$$

Reconfigurability if:  
 $\mathbf{L}_r\mathbf{G}_r = \mathbf{I}_4$

◇ **End Effect:** Parity checks involving 2nd and 5th state variables are now *invalid*

## CONTRIBUTIONS AND FUTURE WORK

- **Contributions:**

- Reflection of hardware faults through appropriate hardware implementations and error models
- Characterization of standard redundant dynamic systems
- Efficient use of redundancy by exploiting algorithm or system structure

- **Future Work:**

- General signal flow graphs (i.e., for arbitrary implementations using factored state variables or other techniques)
- Matching coding schemes to system dynamics
- Protection schemes for analog systems
- Monitoring discrete event systems and max-plus dynamic systems
- Hierarchical and/or distributed error detection and correction