Dependability

Safety-critical systems

- Definitions
  - **Safety** is a property of a system that will not endanger human life or the environment.
  - The **integrity** of a system is its ability to detect faults in its own operation and to inform the human operator.

- **Safety-critical system** is:
  - Safety-related system, or
  - High-integrity system
    - Failure could result large financial loss
    - Examples: banking system, stock market system, server clusters for cloud computing, communication satellites

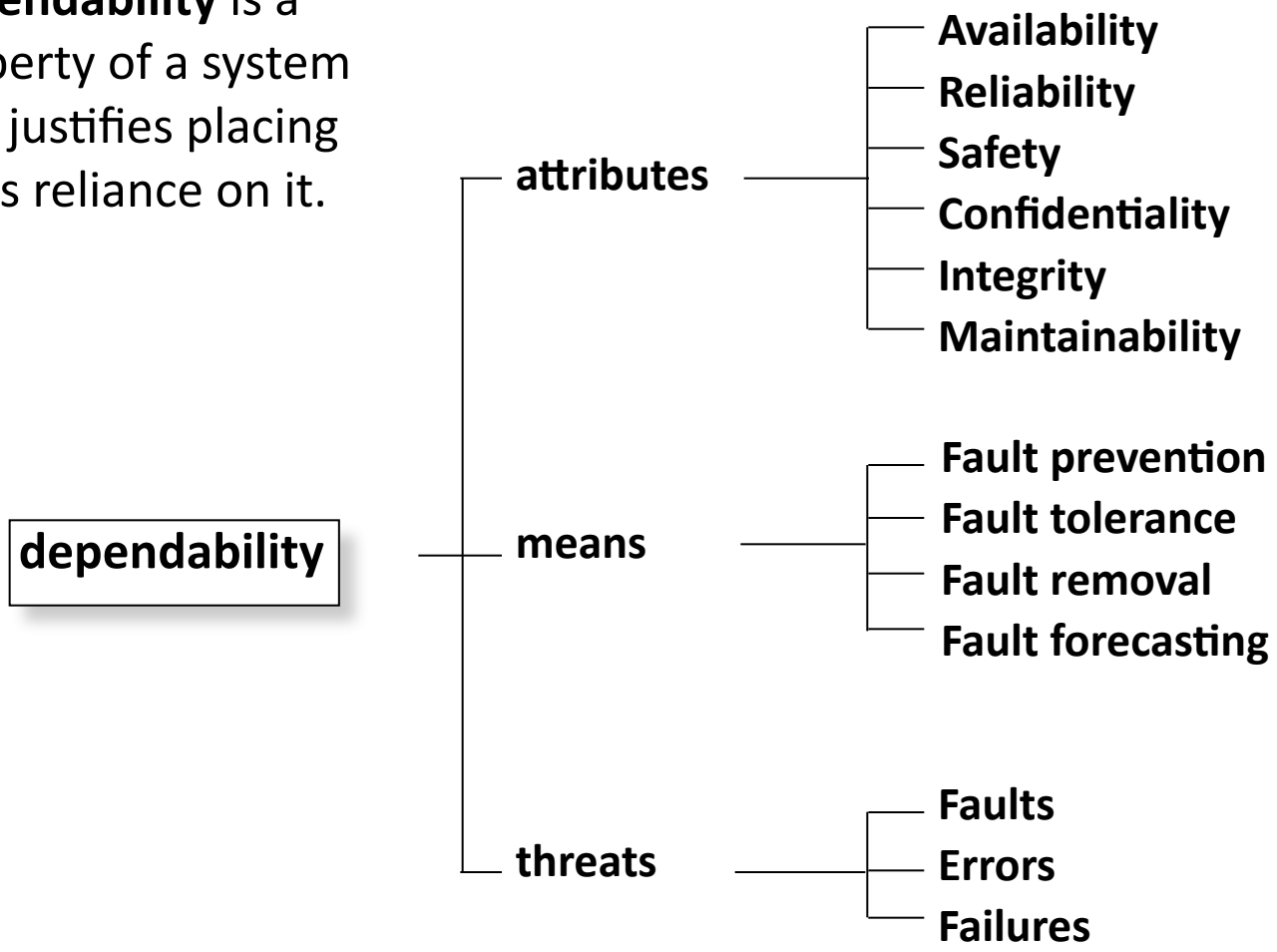
Failsafe operation

- Definition
  - A system is **failsafe** if it adopts “safe” output states in the event of failure and inability to recover.

- Notes
  - Example of failsafe operation
    - Railway signaling system: failsafe corresponds to all the lights on red
  - Many systems are not failsafe
    - Fly-by-wire system in an aircraft: the only safe state is on the ground
    - How to ensure continued operation even in case of failures?

Dependability: an integrating concept

- **Dependability** is a property of a system that justifies placing one's reliance on it.

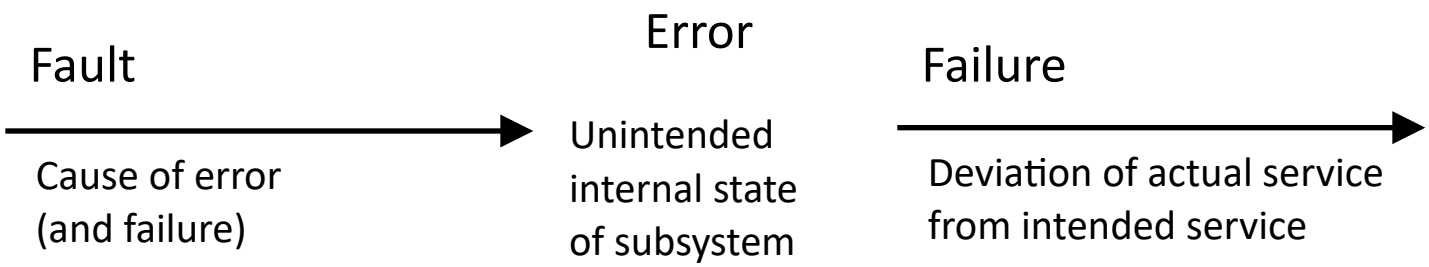


Dependability attributes

- **Reliability**: continuity of correct service
- **Availability**: readiness for correct service
- **Safety**: absence of catastrophic consequences on the user(s) and the environment
- **Confidentiality**: absence of unauthorized disclosure of information
- **Integrity**: absence of improper system alterations
- **Maintainability**: ability to undergo, modifications, and repairs

- **Security**: the concurrent existence of (a) availability for authorized users only, (b) confidentiality, and (c) integrity with 'improper' taken as meaning 'unauthorized'.

Threats: Faults, Errors & Failures



Hazard

- Definitions
  - A **hazard** is a situation in which there is actual or potential danger to people or to the environment.
    - An **incident** (or **near miss**) is an unintended event or sequence of events
      that does not result in loss, but, under different circumstances, has the potential to do so.
    - An **accident** is an unplanned event or sequence of events that causes death, injury, environmental or material damage.

Example hazard: insulin overdose

Risk

- A hazard is a potential for an accident to occur.
  - What are the potential consequences for the accident?
  - How probable (frequent) is the occurrence of such an accident?

- Definition
  - **Risk** is a combination of the *frequency* or *probability* of a specified hazardous event, and its *consequence*.

  - Example
    - Country with population of 50,000,000; approx. 25 people are each year killed by lightning, i.e., 25/50,000,000 = $5x10^{-7}$
    - Risk of being struck by lightning:
      - Probability of $5x10^{-7}$ to be killed by lightning in any given year / individual

Cost vs. benefit

- Safety is expensive: acceptability of risk
  - What is the right cost / safety compromise?
    - 1 mile / hour road speed reduction will reduce accidents by 5%. How much are you prepared to reduce speed to save lives?
    - What is the value of human life?
  - Note: spending more to achieve adequate safety is better than rectifying the problems after an accident.

- Legal aspects: you could be fined or go to prison
  - Laws for “Safety at work” , “Consumer protection”, “Sale of goods”
  - How can you demonstrate you did the best you could to reduce risk to an acceptable level?
  - The role of **standards** and **certification**

Acceptability of risk

- Acceptability of risk is a complex issue involving
  - social factors, e.g., value of life
  - legal factors, e.g., responsibility of risk
  - economic factors, e.g., cost of risk reduction

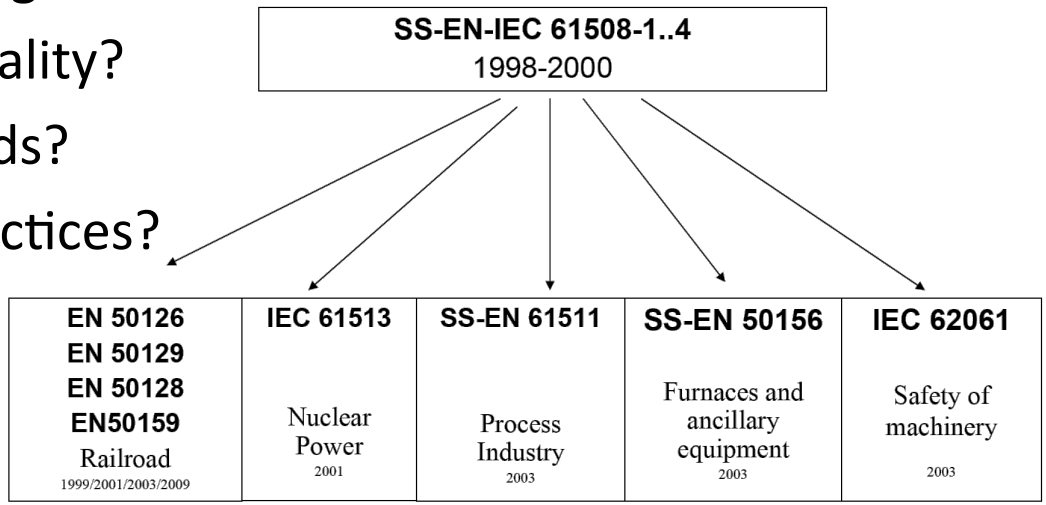
- Ethical considerations
  - Determining risk and its acceptability involves moral judgement
  - Society's view not determined by logical rules
  - Perception that accidents involving large numbers of deaths are perceived as more serious than smaller accidents, though they may occur less frequently

The role of standards

Many industrial sectors have a lot of experience about building safety-critical systems: this knowledge is captured in **standards** and **guidelines**

They provide assistance to engineers

- Appropriate level of quality?
- Use of the right methods?
- Good management practices?
- Legal issues.

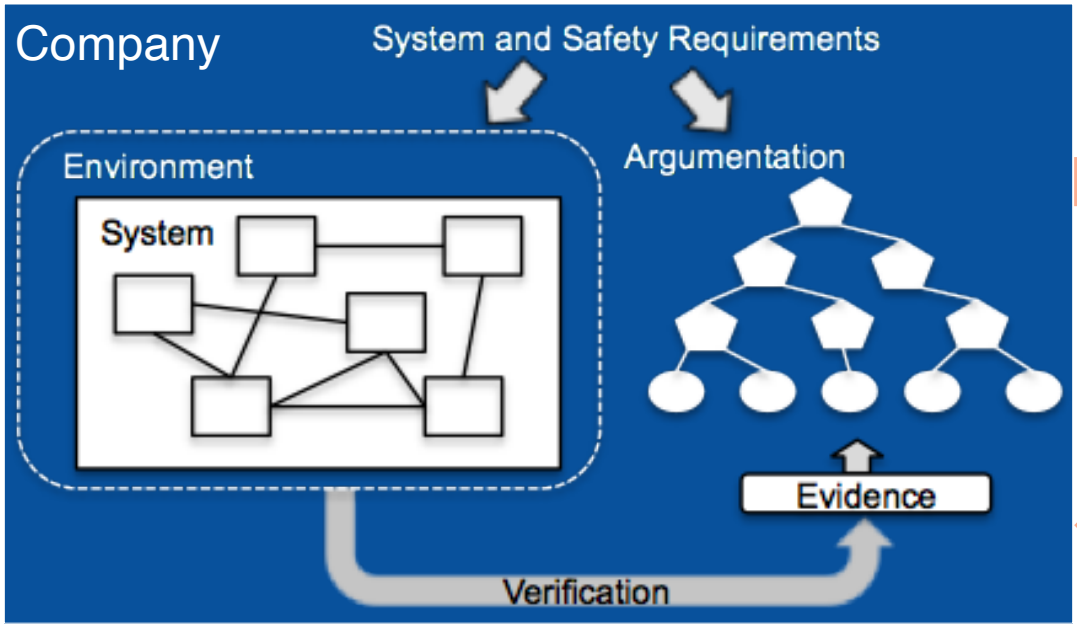


Certification

For safety-critical systems, decision-makers require *pre-release safety assurance evidence* that it manages risk acceptably.

Highly critical systems require **certification** by a **regulatory authority**.

Safety standard



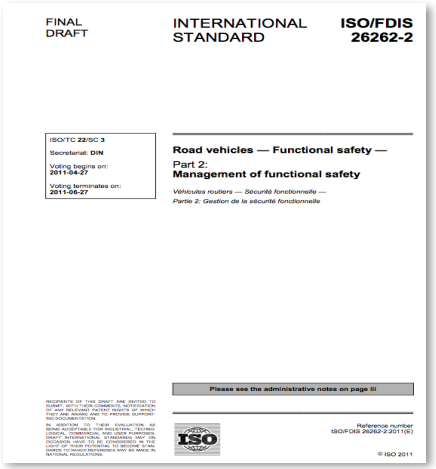
Documents

Independent safety assessor

Certificate

Source: Hans Hansson

Certification – ISO26262 – work products



FINAL DRAFT

INTERNATIONAL STANDARD

ISO/FDIS 26262-2

ISO/TC 22/SC 3
Secretariat: DIN
Voting begins on: 2011-04-27
Voting terminates on: 2011-06-27

Road vehicles — Functional safety —
Part 2: Management of functional safety

Véhicules routiers — Sécurité fonctionnelle —
Partie 2: Gestion de la sécurité fonctionnelle

Please see the administrative notes on page iii

Reference number
ISO/FDIS 26262-2:2011(E)

© ISO 2011

Plans (16)

1. Project management plan
2. Safety plan
3. Safety assessment plan
4. Item integration and test plan
5. Validation plan
6. SW verification plan
7. Design and coding guidelines
8. Production plan
9. Production control plan
10. Maintenance plan
11. Configuration management plan
12. Change management plan
13. Change request plan
14. Documentation management plan
15. HW qualification plan
16. HW component test plan

Artifacts (43)

1. Safety Case
2. Evidence of field monitoring
3. Item definition
4. Impact analysis (modified item)
5. Hazard analysis
6. Safety Goals
7. Functional safety concept
8. Technical safety req. spec.
9. Technical safety concept
10. System design spec.
11. HW/SW interface spec.
12. Production, operation, service, decommissioning. Spec.
13. Integration test spec.
14. HW safety req. spec.
15. HW design spec
16. Analysis of Architecture wrt random failures
17. Analysis of safety goal violation due to random failures
18. Specification of dedicated measures for HW,
19. SW safety req. spec.
20. SW architectural design spec.
21. SW unit design spec
22. SW unit implementation
23. SW unit test spec
24. SW integration test spec
25. Embedded Software (integrated)
26. SW safety req. test spec.
27. SW configuration data spec.
28. SW calibration data spec.
29. Configuration data
30. Calibration data
31. SW configuration & configuration test spec
32. Production requirements (system, HW & SW)
33. Service/maintenance requirements (system, HW & SW)
34. Repair instructions
35. Manual (safety related content)
36. Problem report instructions/process
37. Decommissioning instructions
38. Interfaces within distributed developments (includes 6 artifacts)
39. Change request
40. Change Impact analysis
41. Documentation guideline requirements
42. Software component documentation
43. Proven in use candidate documentation/evidence

Verification/reports (57)

1. Safety plan (confirmation)
2. Safety Case (confirmation)
3. Hazard analysis (confirmation)
4. Item integration and test plan (confirmation)
5. Validation Plan (confirmation)
6. Safety Goals
7. Functional safety concept
8. Technical Safety req. spec
9. Technical safety concept
10. System design spec.
11. HW/SW interface spec.
12. Production, operation, service, decommission. Spec.
13. System level safety analysis (confirmation)
14. Integration test spec
15. Integration test records
16. Validation
17. Safety assessment/audti (confirmation)
18. Release for production
19. HW safety req. spec.
20. HW design spec
21. HW safety analysis report
22. Analysis of Architecture for random failures
23. Analysis of safety goal violation due to random failures
24. HW integration test report
25. SW safety req. spec.
26. SW architectural design spec
27. SW Safety analysis report
28. SW Dependent failure analysis report
29. SW unit design spec
30. SW unit implementation
31. SW unit test spec
32. SW unit test records
33. SW unit test report
34. SW integration test spec
35. SW integration test records
36. SW integration test report
37. SW safety req. test spec.
38. SW safety req. test records
39. SW safety req. test report
40. SW configuration data spec.
41. SW calibration data spec.
42. SW configuration & configuration test spec
43. SW configuration & configuration test records
44. SW configuration & configuration test report
45. Production control measures report
46. Production requirements (system, HW & SW)
47. Assessment report for capability of the production process
48. Maintenance plan
49. Interfaces within distributed developments
50. Change management plan
51. Change request plan
52. Change report
53. Software tool evaluation report

"Safety does not happen by accident"

Source: Hans Hansson

The concept of redundancy

- Definition
  - **Redundancy** is the addition of information, resources, or time beyond what is needed for normal system operation.

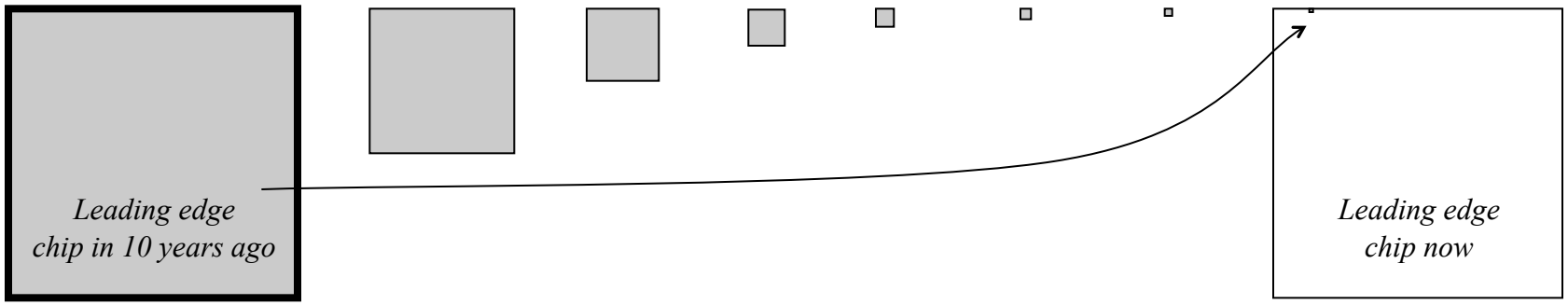
- Digital filter example
  - *Software* redundancy: lines of software to perform a validity checks
  - *Hardware* redundancy : if more memory needed for the software checks
  - *Time* redundancy: each filter calculation performed twice to detect faults
  - *Information* redundancy: output using with a simple parity bit

Hardware redundancy topics

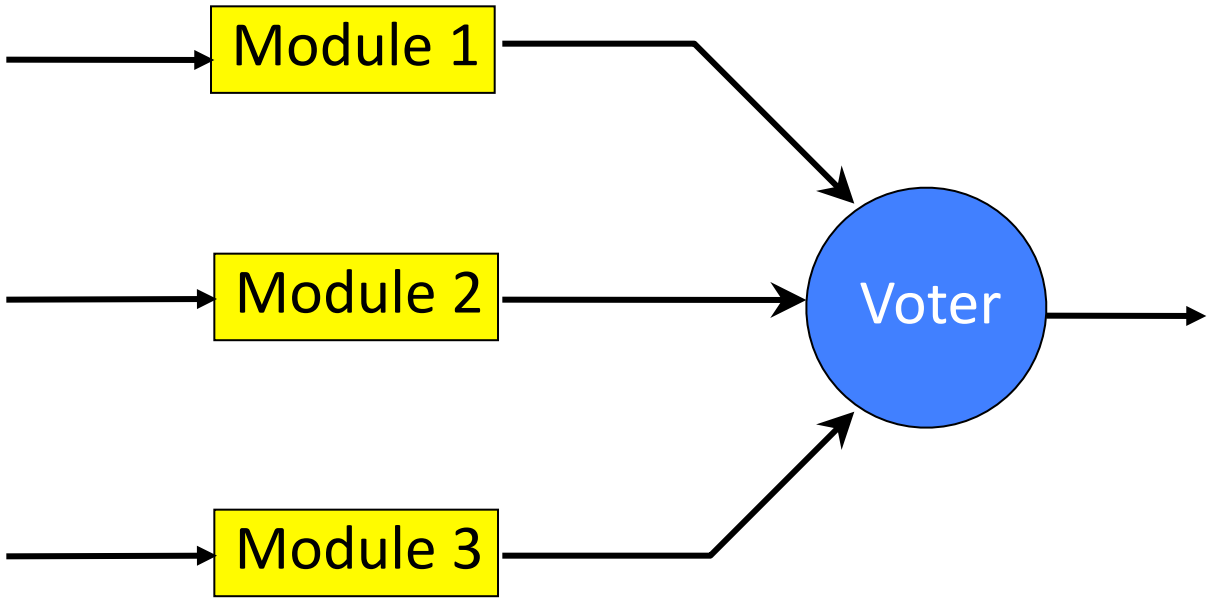
- Hardware redundancy
  - Passive redundancy
  - Active redundancy
  - Hybrid redundancy
- Reliability analysis of redundant hardware
- Choice of hardware redundancy approach

Hardware redundancy

- **Hardware redundancy**
  - Physical replication of hardware is the most common form of redundancy
  - The costs of replicating hardware within a system are decreasing simply because the costs of hardware are decreasing.



Example: Triple Modular Redundancy (TMR)



Masking: the use of sufficient redundancy may allow recovery without explicit error detection.

Information redundancy

- Definitions
  - **Information redundancy** is the addition of redundant information to data to allow fault detection, fault masking or possibly fault tolerance.
  - A **code** is a means of representing information, or data, using a well-defined set of rules.
  - A **code word** is a collection of symbols used to represent a particular piece of data based on specified code.
  - A **binary code** is one in which the symbols forming each word consist of only the digits 0 and 1.

Encoding vs. decoding

- Definitions
  - The **encoding process** is the process of determining the corresponding code word for a particular data item.
    - Example: given the decimal 9, encoding determines the BCD representation of 1001.
  - The **decoding process** is the process of recovering the original data from the code word.
    - Example: decoding transforms the BCD code 0011 into the decimal 3

Example: Parity codes

- Addition of an extra bit to a binary word such that the resulted code word has either an even number or an odd number of 1s.

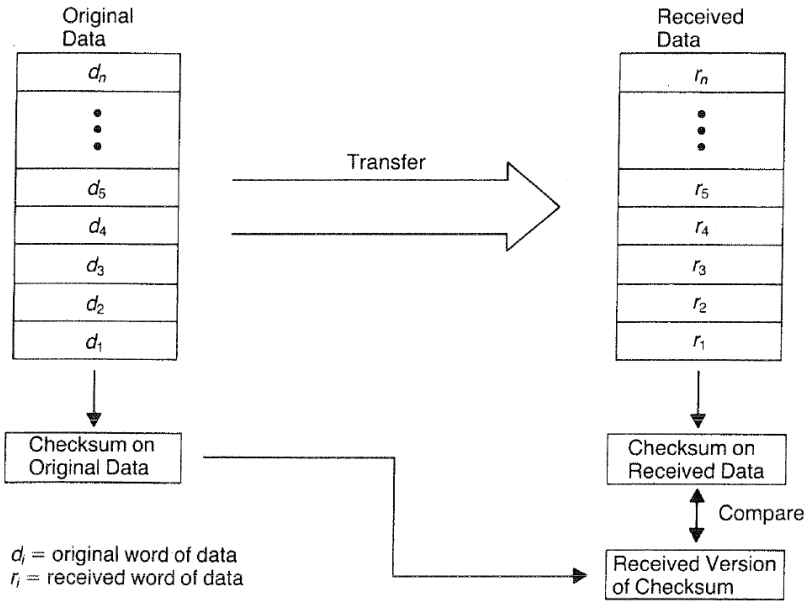
- Can detect any single bit error
  - Even parity
    - One bit flip: odd number of bits
  - Odd parity
    - One bit flip: even number of bits

- Application: memory

TABLE 3.3 Odd and even parity codes for BCD data

Decimal digit	BCD	BCD odd parity	BCD even parity
0	0000	0000 1	0000 0
1	0001	0001 0	0001 1
2	0010	0010 0	0010 1
3	0011	0011 1	0011 0
4	0100	0100 0	0100 1
5	0101	0101 1	0101 0
6	0110	0110 1	0110 0
7	0111	0111 0	0111 1
8	1000	1000 0	1000 1
9	1001	1001 1	1001 0
		↑ Parity bit	↑ Parity bit

Checksum codes



- **Checksum**: sum of the original data, appended to the block of data
- Separable code applicable when blocks of data are transferred from one point to another
- Checksums cannot correct errors
- Four types of checksums
  1. Single-precision
  2. Double-precision
  3. Honeywell
  4. Residue checksums

Codes

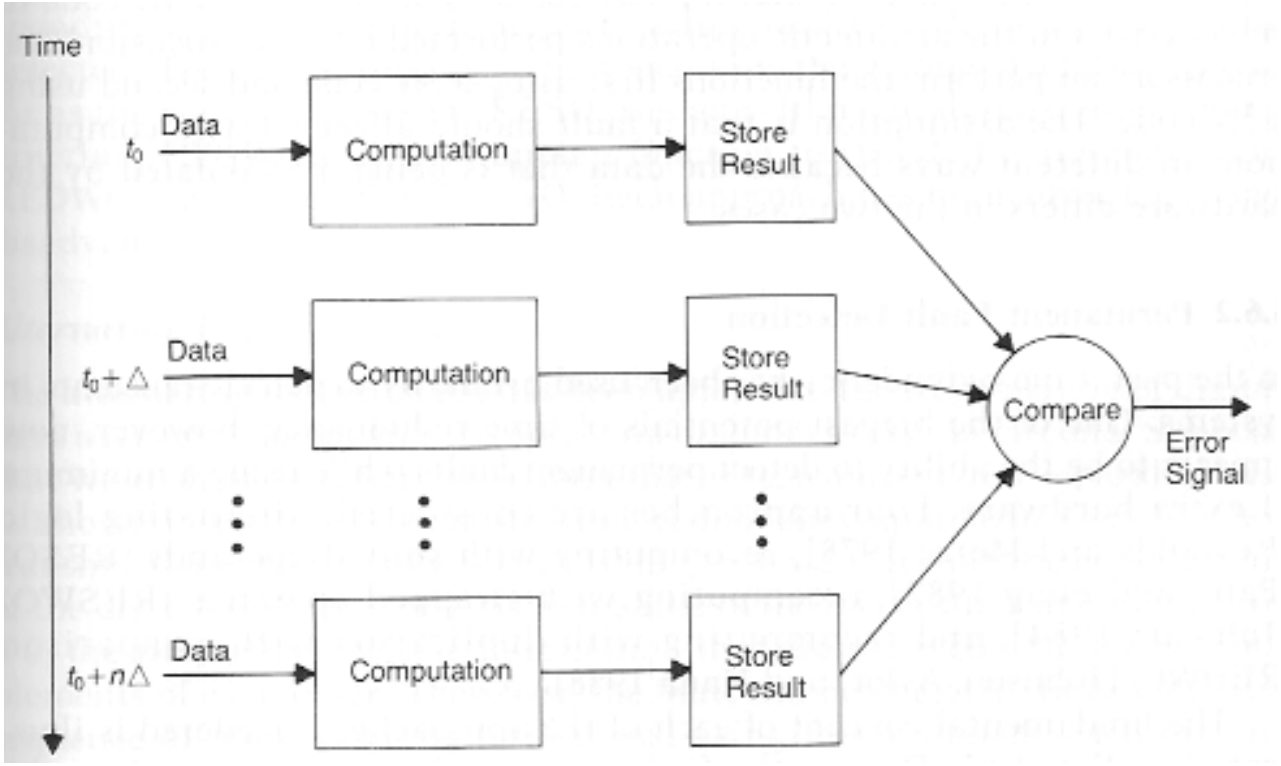
- Error detection vs. error correction codes
  - Detection: code word is invalid
  - Correction: correct word can be identified from the corrupted word
- Separable code vs. nonseparable code
  - Separable: original information is appended with new information
- Selection issues: fulfills the desired error detection/correction while maintaining costs at an acceptable level
  - Key design decisions
    - Separable or not
    - What is required? Detection, correction, both?
    - Number of bit errors to be detected/corrected

Time redundancy

- Disadvantages of hardware and information redundancy
  - Require large amounts of extra hardware for their implementation.

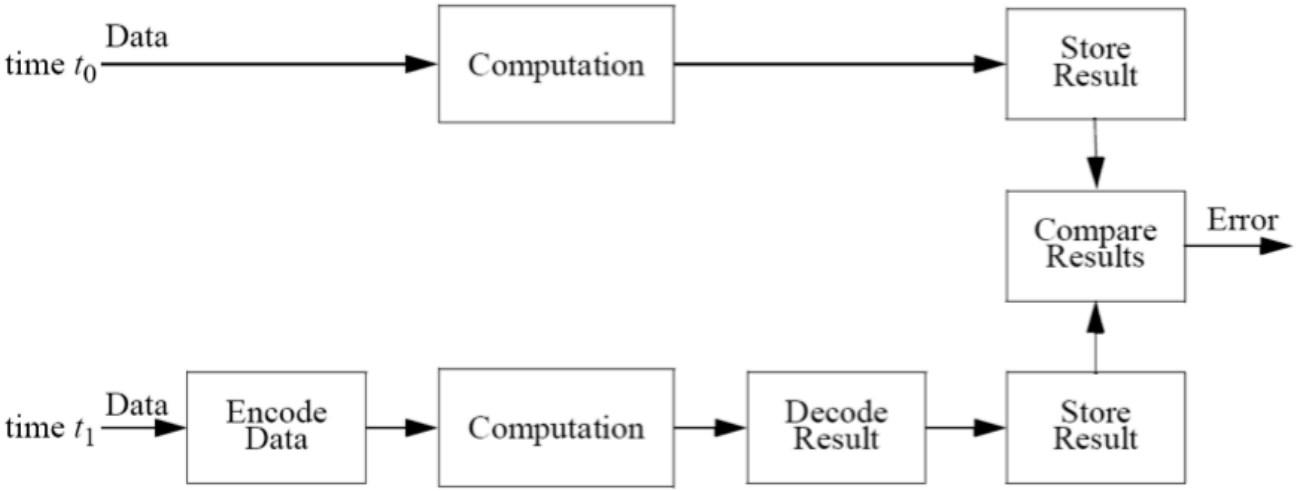
- **Time redundancy**: reduce the extra hardware at the expense of using additional time.
  - Hardware is a physical entity that impacts weight, size, power consumption and cost.
  - Time may be readily available in *some applications*.

Time redundancy: transient fault detection



Time redundancy: permanent fault detection

- Used to detect permanent errors inside the module performing the computation.
- Input data is changed (encoded differently) for the second run.
  - Example encoding functions might include the complementation operator and an arithmetic shift.



Software fault-tolerance

- Software almost inevitably contains defects/bugs
  - Do everything possible to reduce the fault rate
  - Use fault-tolerance techniques to deal with software faults
- Formal proof that the software is correct—
  not practical for large pieces of software
- Instead, use "Software fault-tolerance"
  - Acceptance tests
  - Single-version fault-tolerance
  - N-Version programming
  - Recovery blocks
  - Exception handling
  - Checkpointing

N-Version Programming

- N independent teams of programmers develop software to same specifications - N versions are run in parallel - output voted on
- If programs are developed independently - unlikely that they will fail on same inputs

- Assumption - failures are statistically independent; probability of failure of an individual version = q

- Probability of no more than m failures out of N versions:

$$p_{ind}(N, m, q) = \sum_{i=0}^{m} \binom{N}{i} q^i (1-q)^{N-i}$$

Independent vs. Correlated Versions

- Correlated failures between versions can increase overall failure probability by orders of magnitude
- Example:
  - N=3, can tolerate up to one failed version for any input; q = 0.0001 - an incorrect output once every ten thousand runs
  - If versions independent - failure probability of 3-version system

$$q^3 + 3q^2(1-q) \approx 3 \times 10^{-8}$$

- Suppose versions are statistically dependent and there is one fault, causing system failure, common to two versions, exercised once every million runs
  - Failure probability of 3-version system increases to over 10^{-6}, more than 30 times the failure probability of uncorrelated system

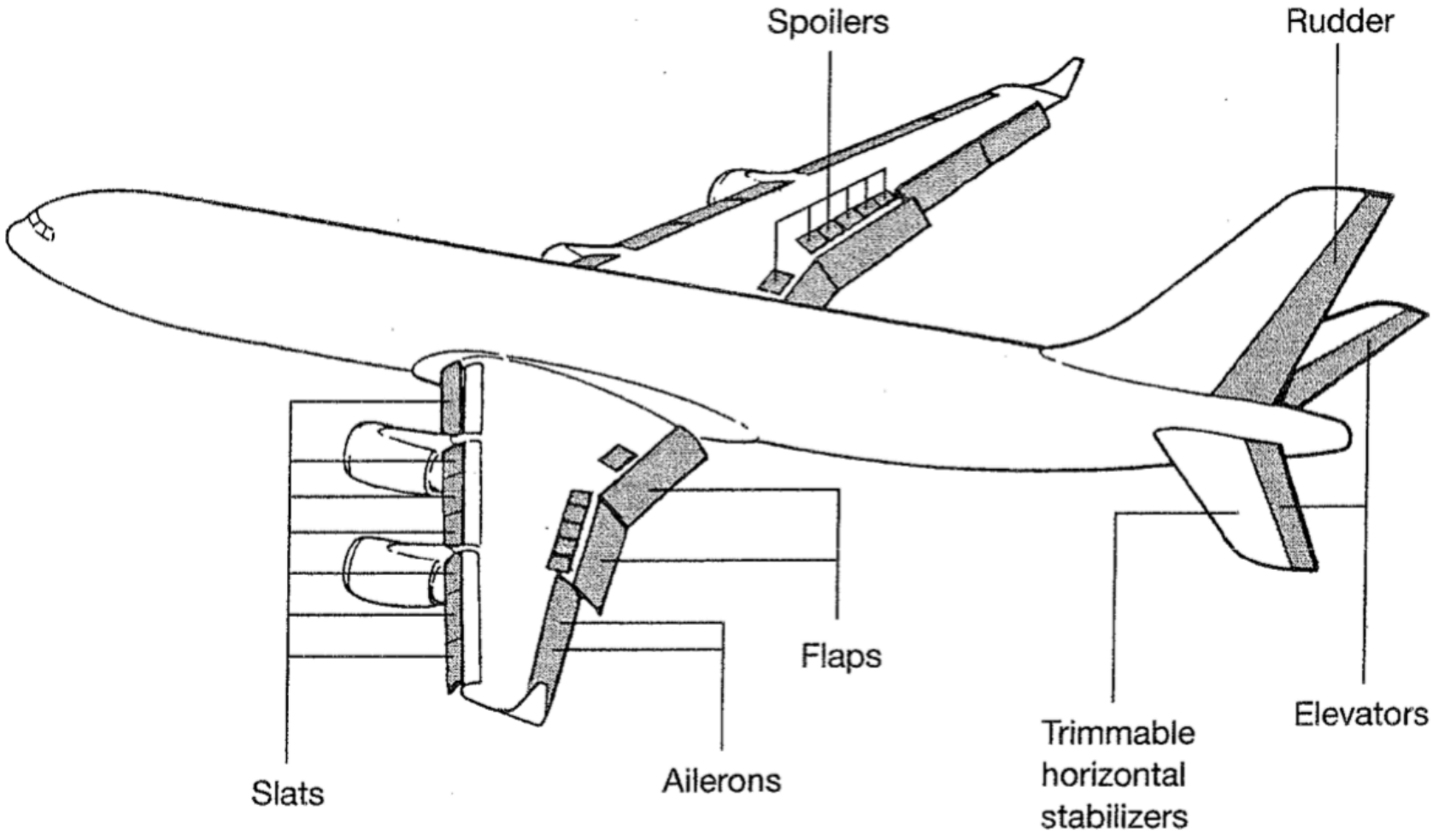
Software fault-tolerance: Checkpointing

- Computers today are much faster,
  but applications are more complicated
- Applications which still take a long time -
  (1) Database Updates
  (2) Fluid-flow Simulation - weather and climate modeling
  (3) Optimization - optimal deployment of resources by industry
  (4) Astronomy - N-body simulations and modeling of universe
  (5) Biochemistry - study of protein folding

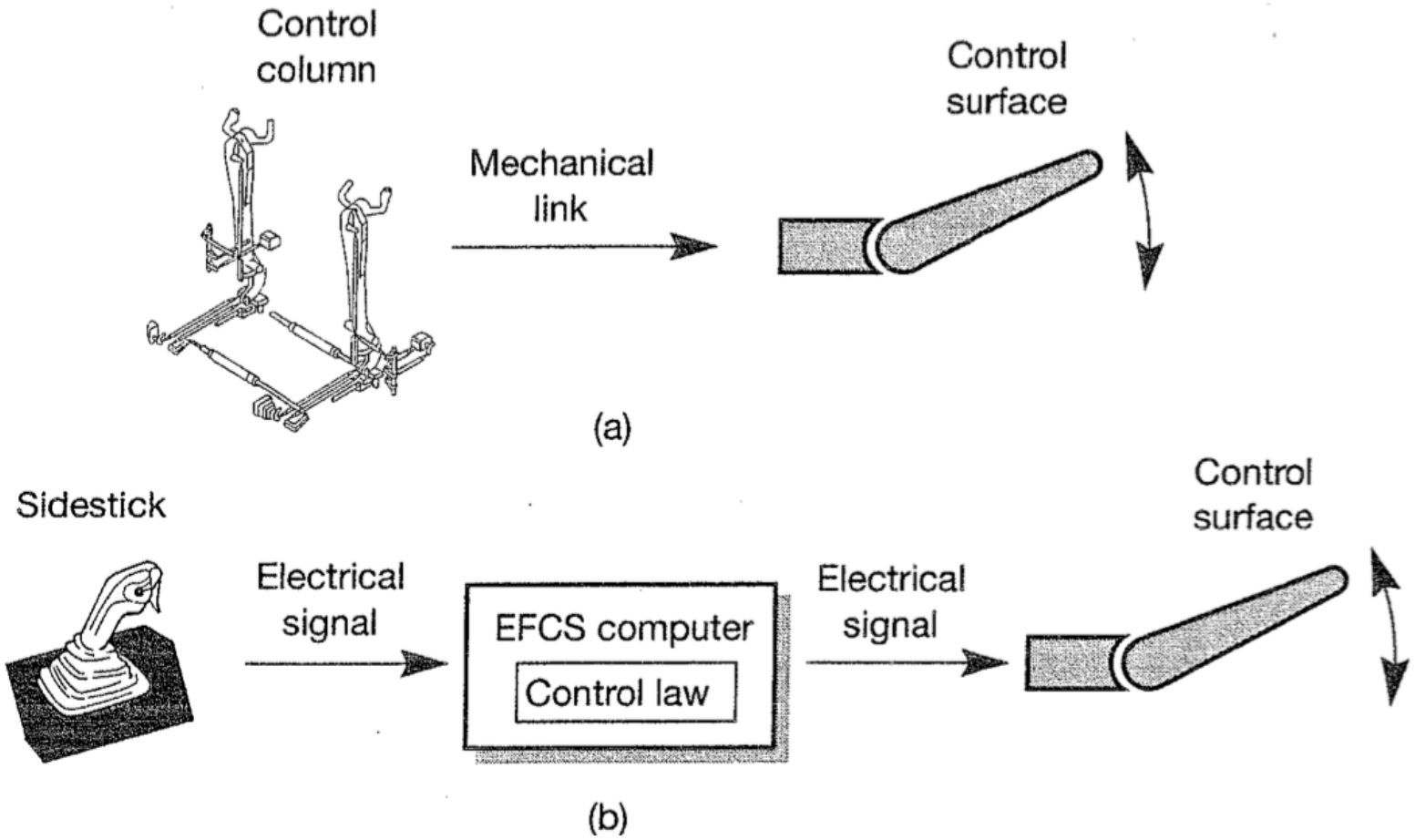
- When execution time is very long - both probability of failure during execution and cost of failure become significant
- Solution: **checkpointing**

Example: Airbus 330/340 Fight Control System

Flight control surfaces



(a) Mechanical vs. (b) Electrical controls



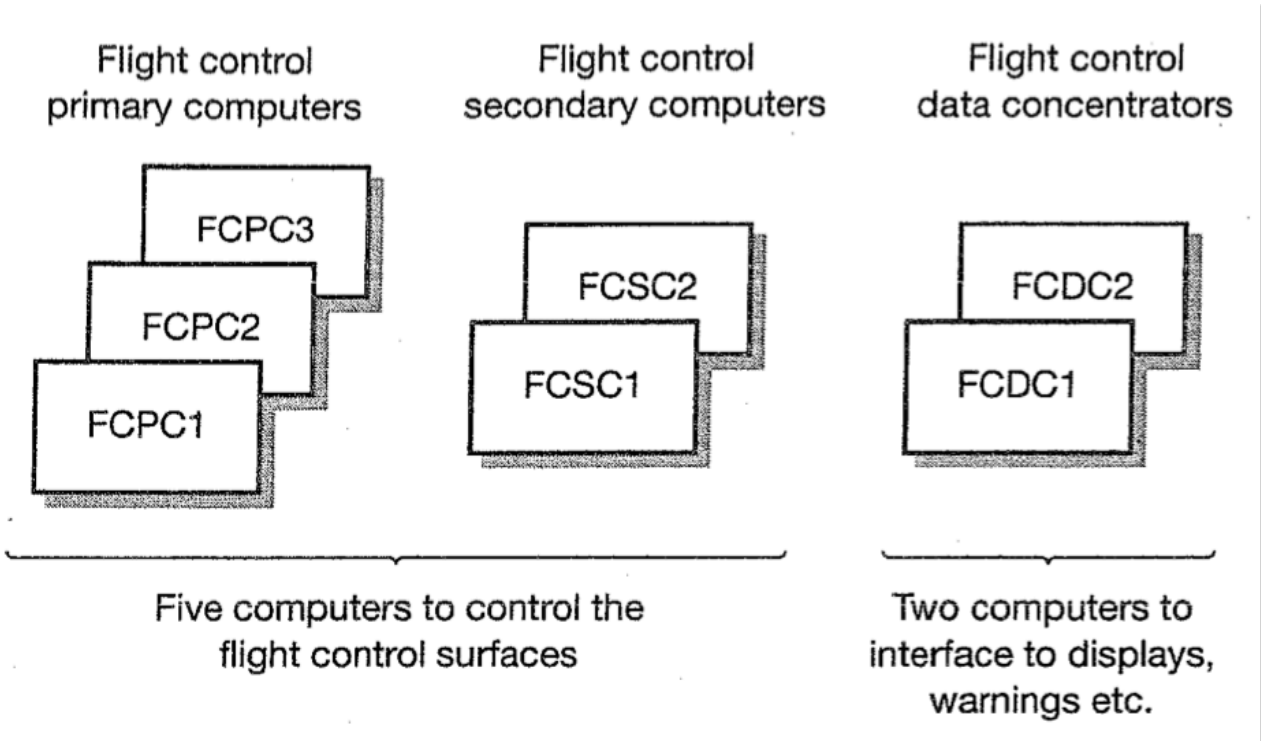
Advantages of “fly-by-wire”

- Pilot workload reduction
  - The fly-by-wire system provides a more usable interface and takes over some computations that previously would have to be carried out by the pilots.
- Airframe safety
  - By mediating the control commands, the system can ensure that the pilot cannot put the aircraft into a state that stresses the airframe or stalls the aircraft.
- Weight reduction
  - By reducing the mechanical linkages, a significant amount of weight (and hence fuel) is saved.

Fault tolerance

- Fly-by-wire systems must be fault tolerant as there is no “fail-safe” state when the aircraft is in operation.
  - In the Airbus, this is achieved by replicating sensors, computers and actuators and providing “graceful degradation” in the event of a system failure.
    In a degraded state, essential facilities remain available allowing the pilot to fly and land the plane.

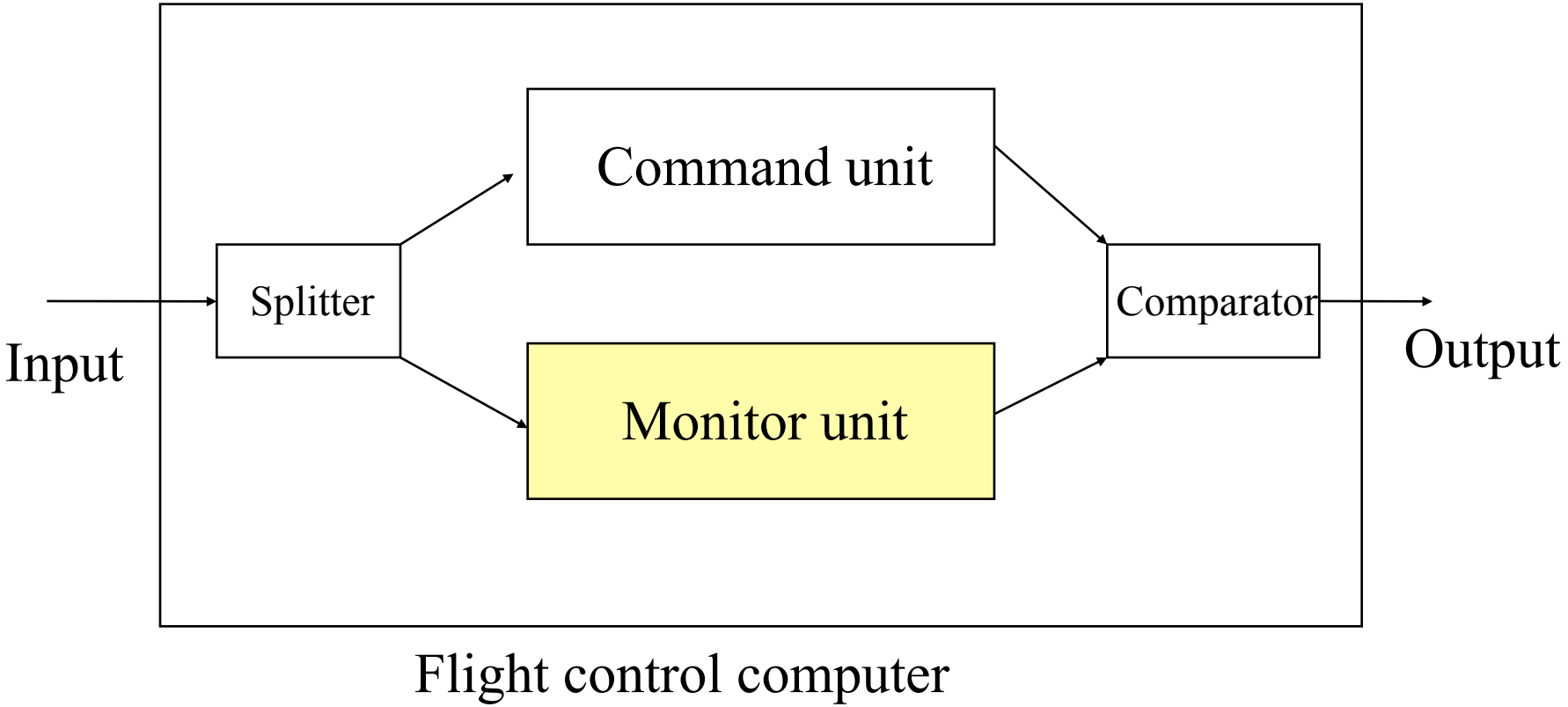
Flight **C**ontrol **S**ystem (FCS) architecture



Hardware organization

- Three primary flight control computers
  - Responsible for calculations concerned with aircraft control and with sending signals to the actuators associated with the control surfaces and engines.
- Two secondary flight control computers
  - Backup systems for the flight control computers.
  - Control switches automatically to these systems if the primary computers are unavailable.
- Only one computer is required for flight control.
  - Therefore, **quintuple redundancy** is supported. All operational computers operate in parallel so there is no switching delay.
- Two data concentrator computers
  - Gather information from the flight control system and pass this to warning and display systems, flight data recorders and maintenance systems.

Computer organization



Hardware diversity

- The primary and secondary flight control computers use different processors, and are designed and supplied by different companies.
  - Primary computers: 80386 at 16 Mhz, supplied by Aérospatiale
  - Secondary computers: 80186 at 12 Mhz, supplied by Sextant Avionique
- The processor chips for the different computers are supplied by different manufacturers.
- All of this reduces the probability of common errors in the hardware causing system failure.

Software diversity

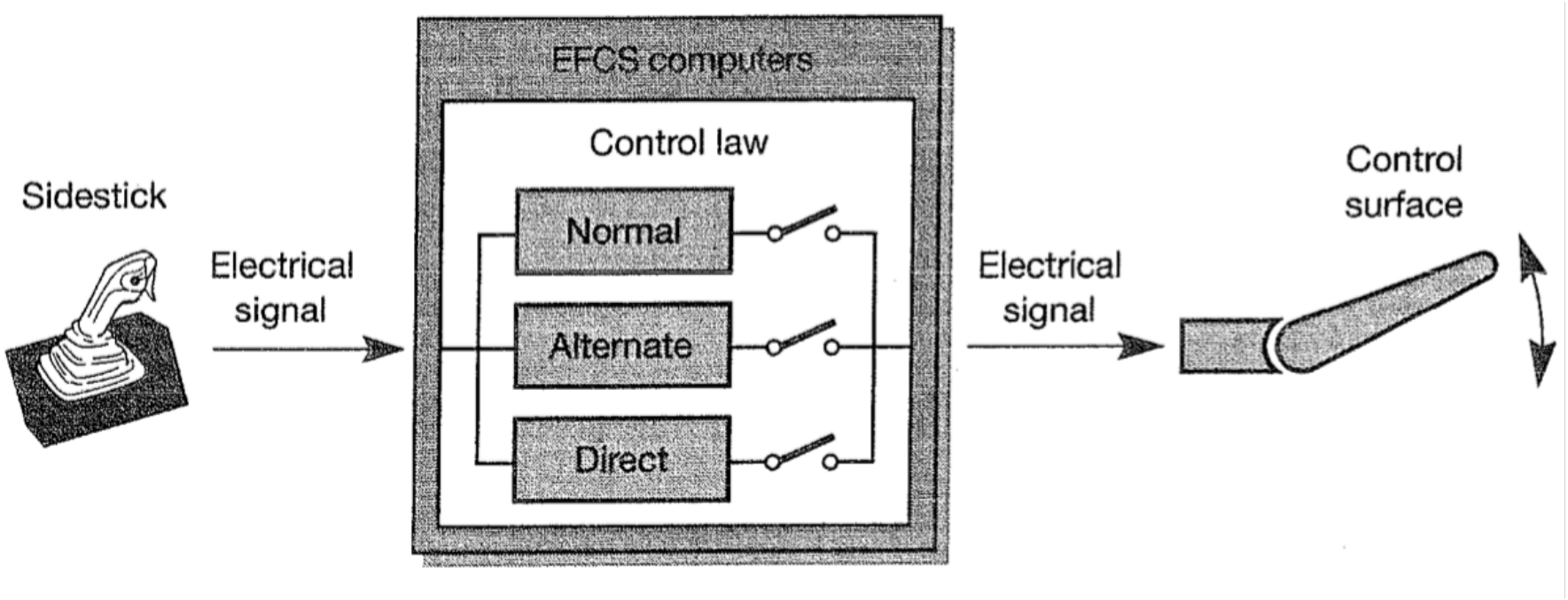
- The software for the primary and secondary flight control computers has been developed by different teams.
- For the secondary computers, different languages are again used for the different channels in each machine.

  - Primary computers, Aérospatiale, 800 KB of software
    - channel: assembler
    - monitor PL/M
  - Secondary computers, Aérospatiale, 300 KB of software
    - channel: assembler
    - monitor: Pascal

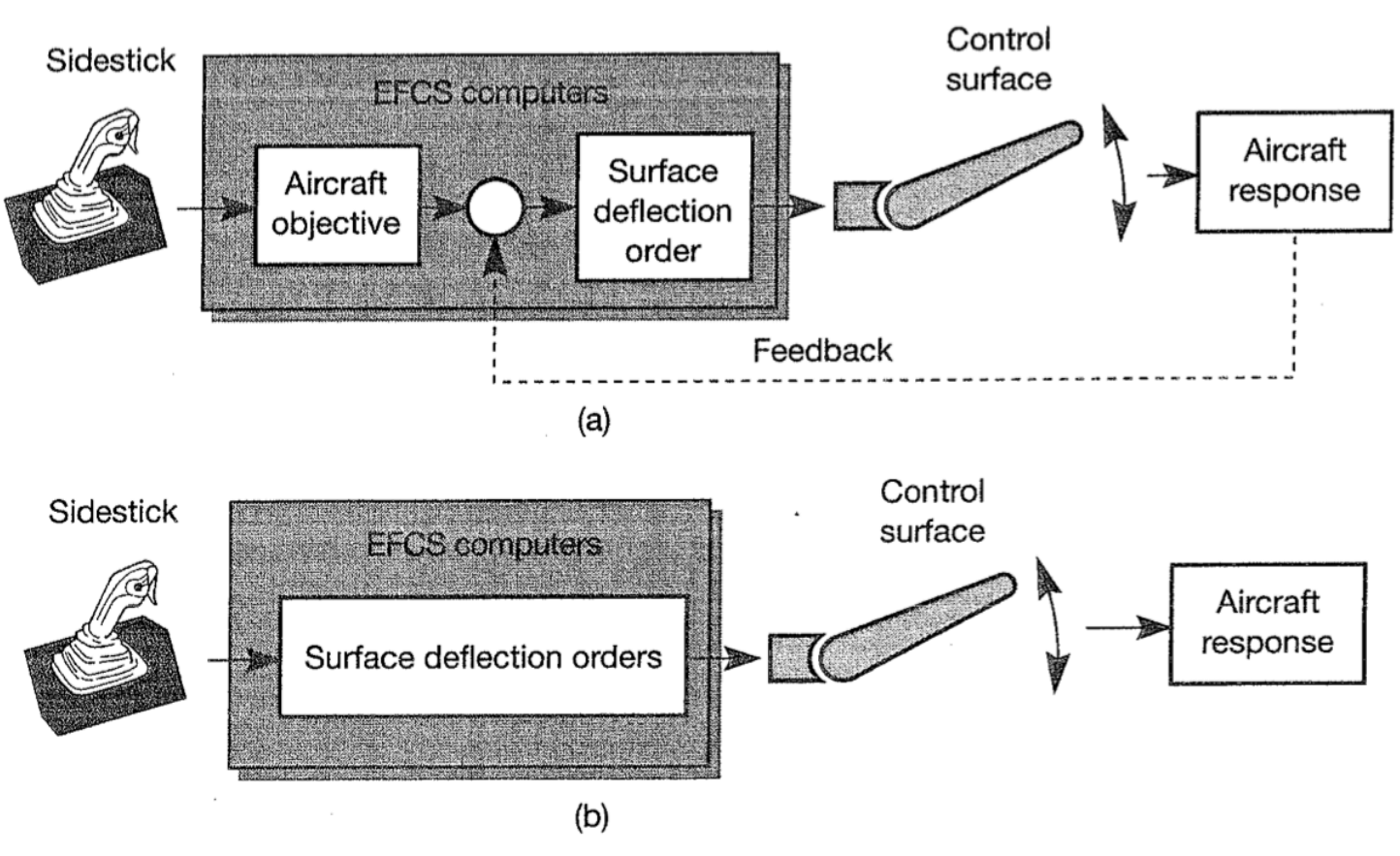
Dynamic reconfiguration

- The FCS may be reconfigured dynamically to cope with a loss of system resources.
- Dynamic reconfiguration involves switching to alternative control software while maintaining system availability.
- Three operational modes are supported
  - Normal - control plus reduction of workload
  - Alternate - minimal computer-mediated control
  - Direct - no computer-mediation of pilot commands
- At least 2 failures must occur before normal operation is lost.

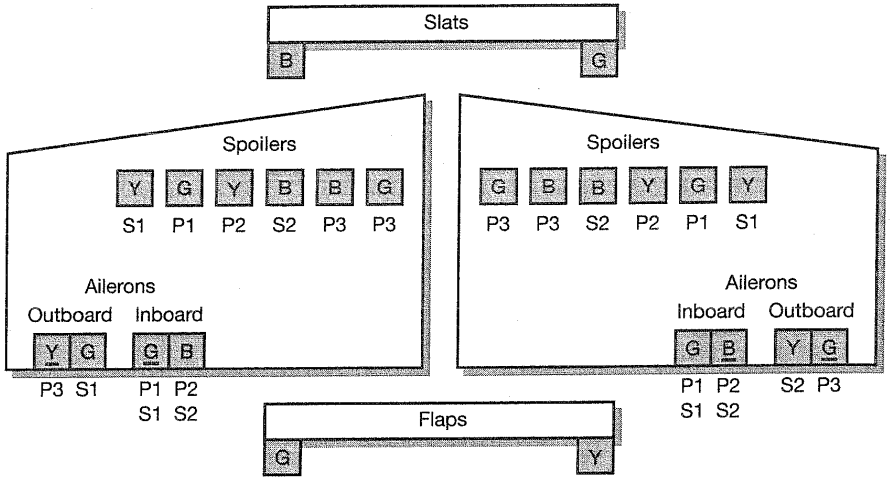
Flight control laws

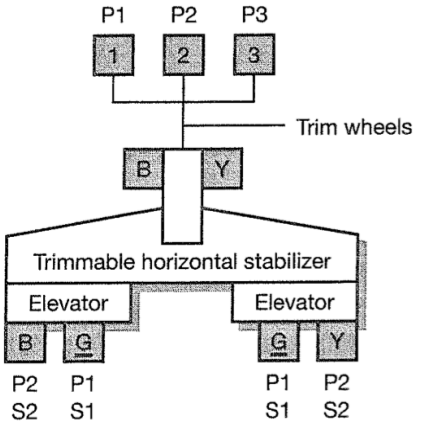


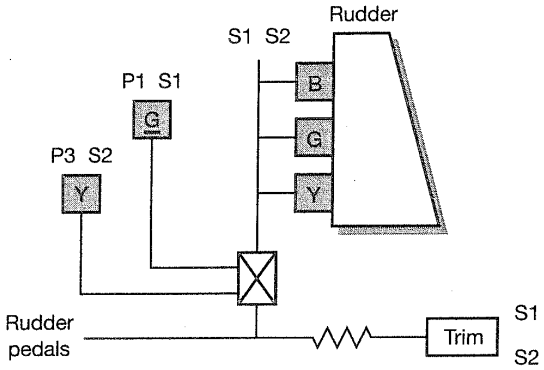
Flight control laws, cont.



Control diversity







P1, P2, P3	Primary computers
S1, S2	Secondary computers
B Blue	Hydraulic supply
G Green	
Y Yellow	

(Underscore indicates priority)

Control diversity, cont.

- The linkages between the flight control computers and the flight surfaces are arranged so that each surface is controlled by multiple independent actuators.

- Each actuator is controlled by different computers so loss of a single actuator or computer will not mean loss of control of that surface.

- The hydraulic system is 3-way replicated and these take different routes through the plane.