Introduction

Complex Safety-Critical Systems

A *complex safety-critical system* is a system, whose safety cannot be shown solely be test, whose logic is difficult to comprehend without the aid of analytical tools, and that might directly or indirectly contribute to put human lives at risk, damage the environment, or cause big economical losses.

- higher performance and lower price gained through increasing complexity ⇒software size is growing exponentially
- reasons for complexity:

  number of functions

  number of states

  discrete behavior discrete systems with discontinuous behavior ⇒small input deviation may cause large output deviations

  invisibility at least 5 of 9 diagram types must be used to properly model systems architecture
- different levels of critical systems

  buisnes-critical high economical loss

  mission-critical failures will lead to mission not possible

  safety-critical risk to human life
- what to do on failures

  fail-operational required to operate also in degraded conditions (some parts of the system not working)

  fail-safe safely shut-down in case of single or multiple failures
- reasons for systems to fail

  inherently incapable design errors during development lead to inadequate system

  overstressed system operation in conditions, for which system was not designed

  variation in the production and design variations in materials, production processes, quality assurance, ...

  wear-out, time-related phenomena components become weaker with use and age ⇒probability of failure increases over time

  errors just errors
- consider safety and failure in design phase ⇒degraded environment requirements

Dealing with Failures

- in the past: "fix-fly-fix"

Role of Formal Methods

- formal verification still in early stages
- difference: verification used for correctness, in safety degraded performance also important

Dependability, Reliability and Safety Assessment

Introduction

fault presence of a defect or anomaly in an item or system

dormant fault fault that has not yet lead to an error

error event: deviation of the desired behavior

latent error error has the potential to cause a failure, but, but has not yet done so

failure inability of a system to perform required function

permanent fault a fault that will not vanish on its own

transient fault a fault that will will vanish with time

Concepts

safety-critical system (safety-related system, safety instrumented system) a system designed to ensure safe operation of equipment or plant it controls

- external vs internal requirements
- functional vs nonfunctional requirements

Safety system not endangering or causing harm to humans or the environment ⇒absence catastrophic failures

primary safety direct effects of system

functional safety safe operation of system under control

indirect safety indirect consequences of failure, e.g. unavailability of service

Reliability characteristics of a system to operate correctly over a given period of time ⇒continuous service

failure rate rate at which a system fails

time to failure time interval between beginning of system operation

failsafe state a state which is completely safe, even if no operation is possible ⇒all traffic lights are red

Availability probability of a system to operate correctly at time t

Integrity possibility, that a system will detect faults during operation ⇒fault detection

fault recovery system is able to restore correct operation

data integrity where consistency of data is important

safety integrity capability of system, to perform all required safety functions satisfactory within the stated period of time

safety integrity level related to classification in risk analysis

high-integrity system reliability and availability ⇒dependable system

maintainability possibility, that a given system can be maintained (prior-to-failure maintenance (e.g. aircraft maintenance) and restore a failed system to operational state)

dependability a system for which reliance can justifiably be placed on the service it delivers

Classification of Faults

nature systemic inherent in given system ⇒wrong design or software bug

random nondeterministically, at any time, e.g. through hardware wear-out, stress conditions

accidental not on purpose

intentional sabotage, often security related

equipment hardware random or systemic

software always systemic

phase of creation design introduced during development

operational wear.out, incorrect operation, unforeseen operation

extent and system boundaries localized e.g. affecting only single module

global i.e. affecting the whole systems

internal inside of the system

external outside of the system

duration permanent do not go away, e.g. all systemic faults

transient/sporadic also transient faults may lead to permanent errors

intermittent appears and disappears

hardware level transistor-level

atomic-level

gate-level

Fault Models

- logical characterization, timing models ⇒fault model
- automatically generate fault vectors
- hardware fault models are easier to identify and formalize

Stuck-At Fault Model

- fault in digital circuit where some module has an input or output bit stuck
- stuck-at-0 (sa0) or stuck-at-1 (sa1)

Stuck-Open/Stuck-On & Stuck-Closed/Stuck-Off Fault Model

- transistor-level fault model ⇒CMOS gate fault
- dependent on exact circuit

Bridging Fault Model

- two circuit components are shorted together
- Wired-AND (positive logic bridging) or WIRED-OR (negative logic bridging)

Delay Fault Model

- element is delayed in assuming new state/ value (rarely: more quickly)
- gate-delay/ transitional fault model: delay of single circuit or gate
- path-delay: total delay along a path

Managing Faults

fault avoidance keep faults out of the design (formal methods, quality control)

fault removal remove fault from the final product (system testing, formal verification)

fault detection detect faults during service, conduct countermeasures

fault tolerance system operates correctly, even if failures occur, can be combined with fault detection

Fault Detection

model-free approach based on data-analysis ⇒detect faults by comparing actual data with historical data, requires prior acquisition of data

knowledge-based approach exploit expert knowledge to detect faults, requires the prior acquisition of knowledge

mode-based approach description of nominal system behavior in terms of nominal model, compare actual behavior

for all approaches:

online during system operation

offline post-mortem

Functionality Checking

- detect wrong operation of hardware components using routines to check their functionality, typically check memory, processor, network

Consistency Checking

- detect faults by comparing the actual output of software/ hardware with expected ranges

Signal Comparison

- check signals withing redundant systems

Instruction and Bus Monitoring

- check operation of processor, check instructions before being fetched to memory, monitor bus for illegal access

Information Redundancy

- include: parity checks, checksums, error correcting codes

Loopback Testing

- check, that signal from sender to receiver is unchanged

Watchdog and Healthc Monitoring

- set watchdog, during operation increase or reset counter, when watchdog times out, something is wrong

Fault Prediction

- evaluate the likelihood that a given system or component will fail over a period of time
- dependent of nature of fault

Fault Tolerance

- fault tolerance most often achieved through some kind of replication of critical components

Triple Modular Redundancy (TMR)

- replicate critical component
- three versions
- have voter for output
  - majority vote

  - average/ median vote

- TMR provides fault tolerance, if:

  at most, one component fails obvious for majority voting, but also for the other

  the voting mechanism does not fail voting is not replicated, single point of failure

  no systemic failures the different instances fail independently

  Isolation of failed component is not an issue i.e. masking a faulty output is sufficient to guarantee that the system will operate as expected

Dealing with Multiple Failures

- increase redundancy: N replications, N should be an odd number $\Rightarrow N$-modular redundancy
- often not feasible due to cost

Dealing with Failure of Voting Component

- simple architecture ⇒more secure than complex component
- replicate voting component, e.g. triple voting component

Deadling with Systemic Failures

- TMR not robust against systemic failures (design) ⇒**common mode faults**
- N-version programming

Fault Tolerance and Fault Detection

hot stand-by redundant component runs in parallel, control can immediately be switched

cold stand-by redundant component is switched off, probability of failing back-up is lower

warm stand-by redundant component is powered but idle

Dealing with Transient Failures

- must be able to detect and replace failure

Emergency Shutdown Systems

- failsafe ⇒system can be brought into safe state ⇒emergency shutdown architecture

Fault Coverage

- fault coverage is a measure for the degree of success of above techniques
- very hard for fault avoidance (how many faults have been avoided)
- fault removal, detection, tolerance ⇒could be experimentally tested, for real systems not really possible
- fault coverage is evaluated over fault model

Reliability Modeling

- quantitative measuring for system reliability
- failure probability ⇒probability that a system will fail
- failure rate ⇒frequency of failure of the component
- Mean Time To Failure $MTTF$: how long does it take on average, until the first failure occurs
- Mean Time To Repair $MTTR$: average time needed to repair a system or component

- Mean Time Between Failures $MTBF = MTTF + MTTR$
- $availability = \frac{MTTF}{MTTF+MTTR} = \frac{MTTF}{MTTB}$

System Reliability

- reliability of system: prototypical system structures ⇒series structure, parallel structure
- more complex systems ⇒put together from serial and parallel structures