EDA122/DIT061 Fault-Tolerant Computer Systems

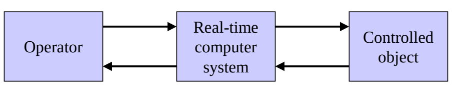
Welcome to Lecture 15

Time-triggered systems

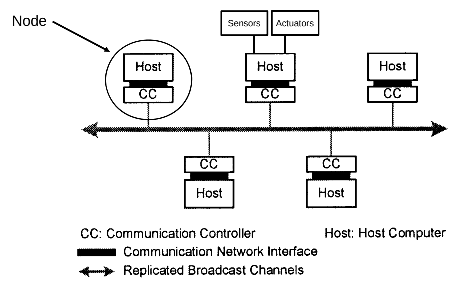
Outline

- Time-triggered systems
  - General principles
  - Time-triggered vs. Event-triggered systems
- The Time-Triggered Architecture (TTA)

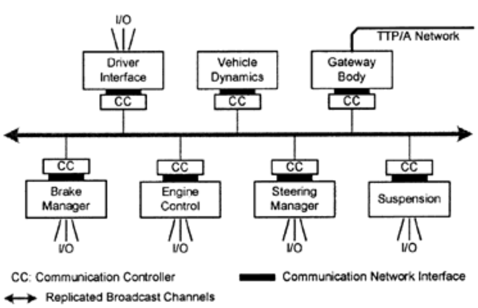
Embedded real-time system



Distributed real-time system



Distributed drive-by-wire system



Characteristics of real-time systems

- Data must be correct with respect to both *value* and *time*
- Correctness in the time domain is determined by *response time* requirements
- The response time requirements must be fulfilled during
  - peak-load situations
  - anticipated fault situations
- Typical response times: 1ms - 10s

Types of real-time systems

Two major categories of real-time systems:

- Event-triggered systems
- Time-triggered systems

Event-triggered systems

- External events initiate program execution (usually through the interrupt mechanism)
- Design for flexibility
- Programs compete for computing resources
- Nodes can attempt to send a message at any time
- Nodes compete for the network and may have to back-off in case of collisions, i.e., when more than one node attempt to send at the same time
- Network arbitration and competition for processing resources can cause jitter in the time at which messages are sent

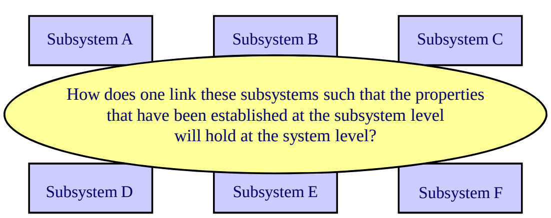
Time-triggered systems

- Program execution is time-triggered and runs according a pre-planned schedule
- Communication is time-triggered and pre-planned runs according a pre-planned schedule
- Design for predictability under peak-load situations
- Requires global time (local clocks must be tightly synchronized)

Important properties of real-time systems

- Timeliness
- Predictability
- Fault tolerance
- Maintainability
- Extensibility
- Composability

Composability



From H. Kopetz, Real-time systems: Design Principles for Distributed Embedded Applications, Kluwer Academic Publishers, ISBN 0-7923-9894-7, 1997, pp. 34

Composability

- In a composable system, it should be possible to add new functions without affecting the temporal properties of any existing functions
- The communication system plays a major role in enabling composability of a distributed system
- Messages sent by a new function must not affect the timing of messages sent by existing functions
- Requires scheduling analysis and control of the network traffic

Event-triggered systems

- Advantages
  - Adaptability
  - High resource utilization under nominal workload conditions
- Drawbacks
  - Peak-load situations (event storms) may lead to system congestion
  - Low resource utilization during peak-load situations
  - Difficult to protect against faults in the environment, e.g., spurious interrupts
  - Composability is difficult to achieve, because temporal control is a global issue

Time-triggered systems

- Advantages
  - High resource utilization under peak-load situations
  - Supports composability because temporal control resides in the communication systems
  - Simplifies implementation of fault-tolerance
  - Forces in-depth analysis of peak-load situations
  - Requires less effort during system integration and testing
- Disadvantages
  - Inflexible
  - Non-optimal resource utilization under low-load situations
  - Requires more design work to pre-plan program execution and message passing

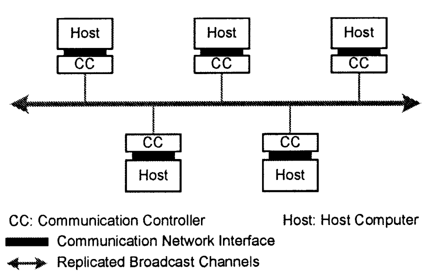
The Time-Triggered Architecture

- Outline
  - System model and structure
  - Fault tolerance
  - Communication model
  - Replica determinism
  - Composability in the temporal domain
- Real product sold by TTTECH

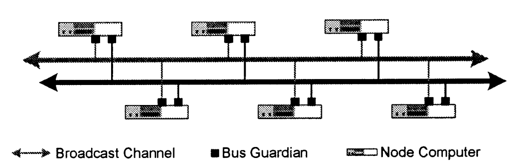
TTA system model

- A computer system is modelled as a set of nodes that are interconnected by a real-time communication system.
- The nodes consist of a communication controller (CC) and a host computer.
- The communication controller constitute an autonomous communication system.
- Network arbitration by Time-Division Multiple Access (TDMA).
  - Messages are transferred between nodes according to a pre-planned and time-triggered schedule
  - All nodes have access to the global time through local clocks that are periodically synchronized with each other

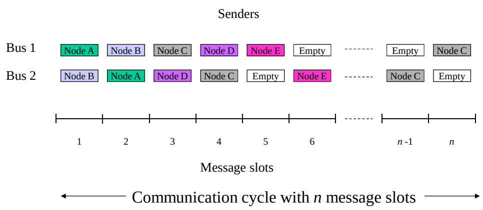
Structure of a TTA cluster



Duplicated Bus

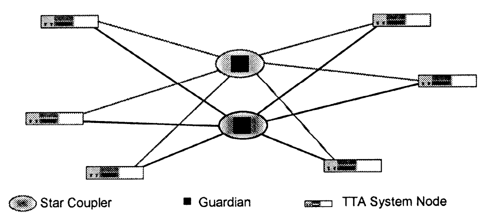


Example of time-triggered bus schedule for a cluster with five nodes

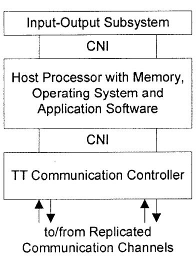


The cluster consists of five nodes denoted A, B, C, D, E

TTA system based on a Star Topology

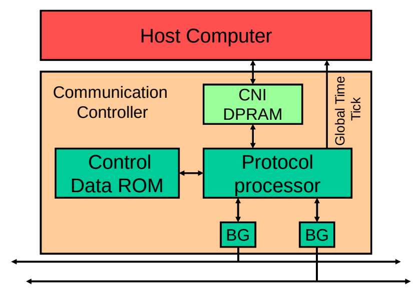


Structure of a TTA node



CNI: communication network interface

Hardware Structure of the Communication Controller in a bus-based TTA node



CNI = Communication Network Interface

DPRAM = Dual port random access memory
BG = Bus guardian

Communication Controller

- Protocol processor
  - Reads messages from the dual-port random access memory (DPRAM) and delivers them to other communication controllers at pre-planned time instances (send operation)
  - Receives messages sent by other communications controllers and writes them to the DPRAM at pre-planned time instances (receive operation)
  - Keeps track of the communication schedule by means of a message descriptor list (MEDL) stored in the Control Data Read-only memory (Control Data ROM)
  - Contains the local clock which is periodically synchronised with the local clocks in the other communication controllers
  - Provides the global time ticks to the host computer

Communication Controller (cont'd)

- Control Data ROM
  - Contains MEDLs (message description lists), i.e., time tables for the pre-planned communication schedules
  - Contain alternate communications schedules. Changing from one communication schedule to another is called a "mode switch"
- CNI DPRAM (Dual ported random access memory)
  - Stores incoming and outgoing messages

Communication Controller (cont'd)

- Bus guardian
  - Provides fault and error containment
  - Protects the system against "babbling idiot" failures of the protocol processor
  - Independent units that monitor the temporal behaviour of the node
  - Prevents the node from sending outside its pre-allocated message slots
  - Should ideally be a totally independent subsystem with its own clock, power supply, and distributed clock synchronisation
  - In practice, bus guardians are implemented on the same die as the other parts to reduce cost (this may not be acceptable in highly critical systems requiring exceptionally high *fault containment coverage*)

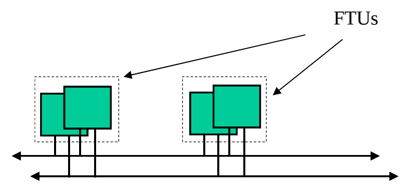
The Time-Triggered Architecture

- Outline
  - System model and structure
  - Fault tolerance
  - Communication model
  - Replica determinism
  - Composability in the temporal domain

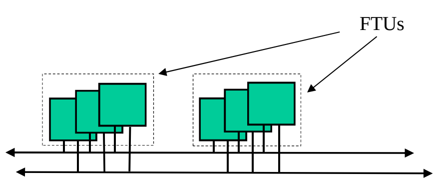
Tolerating node failures

- Tolerating node failures - Two alternatives:
  - Fail silent nodes
    - Node failures are "signalled" by silence
    - Requires *f+1* nodes to tolerate f failures
  - Nodes exhibiting symmetrical value (consistent content) failures
    - Value failures are masked by voting (TMR or NMR)
    - Requires *2f+1* nodes to tolerate *f* failures
- Tolerating transmission failures
  - Message replication – two or four physical messages for each logical message
  - No retransmissions (simplifies timing analysis)

Fault-Tolerant Units (FTUs) with duplicated fail-silent nodes



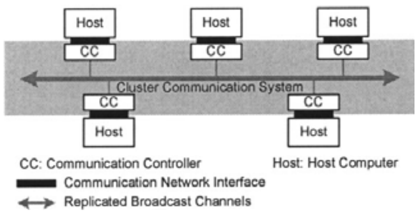
Fault-Tolerant Units (FTUs) with nodes working in TMR



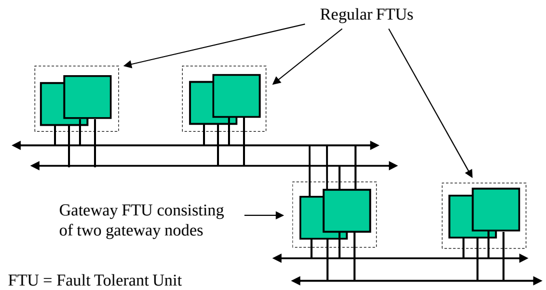
TTA Clusters

- A cluster is a set of nodes that shares a communication network
- A system can consist of one or several clusters, and hence several communication networks
- Clusters are connected via gateway nodes

Cluster Communication Systems



TTA System with two clusters and duplicated nodes



The Time-Triggered Architecture

- Outline
  - System model and structure
  - Fault tolerance
  - Communication model
  - Replica determinism
  - Composability in the temporal domain

Communication model

- TTA supports *state messages* and *event messages*
- State messages
  - Broadcast *state information* from one node to all other nodes
  - Updates the local views of the global state
  - Information remains in the sender after a send operation
  - Information is not consumed when read in the receiver
- Event messages
  - Information consumed (disappears) by the sender after a send operation
  - Uses dedicated message slots that carries different event messages
  - Information must be queued at the receiver and is consumed when read
- We will focus on state messages and state information

State information

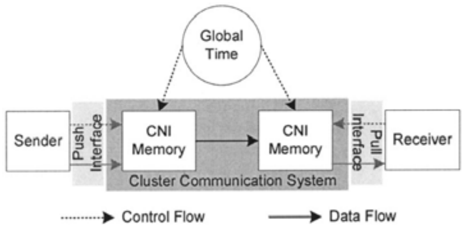
- Stored in a distributed database – each node has their own copy of the state information
- Semantics similar to a global variable
  - A state message carrying a new version of state information overwrites the previous state information.
  - Idempotent
    - Can be read many times (information is not consumed)
    - Can be written many times by replicated messages
- Validity of state information is limited in time

Message semantics and control flow

- Unacknowledged datagram
  - No handshaking (implicit flow control through time-triggered communication)
  - No retransmission of corrupted or lost messages
  - Transmission errors are handled by sending redundant messages (typically two or four messages)
  - *Elementary* unidirectional dataflow (sender's operations is not influence by the receiver)

Explicit flow control and message retransmission increases delay jitter and should therefore be avoided in hard real-time system

The TTA Interface



Elementary unidirectional dataflow: no control signals exchanged between sender and receiver

Communication Network Interface (CNI)

- The CNI separates the local processing within a node from the global interactions among the nodes
- The CNI uses two unidirectional elementary data-flow interfaces, one from the host computer to the communication system, and one in the other direction.
- Information push: a message is sent by writing it to the CNI memory before a pre-defined time
- Information pull: a message is received by reading it from the CNI memory before a pre-defined time

The Time-Triggered Architecture

- Outline
  - System model and structure
  - Fault tolerance
  - Communication model
  - Replica determinism
  - Composability in the temporal domain

Replica Determinism

- In systems that use replication and voting, it is imperative that non-faulty replicated units produce identical results
- This implies that non-faulty replicated units must have a consistent view of the application state when they calculate a result
- This property is called “replica determinism”

The Time-Triggered Architecture

- Outline
  - System model and structure
  - Fault tolerance
  - Communication model
  - Replica determinism
  - Composability in the temporal domain

Principles to achieve Composability in the Temporal Domain

- Independent development of nodes
  - Separation of node design from architectural design
  - Supported by a precise specification of the CNI
- Stability of prior services
  - Ensures that a validated service of a node is not refuted by integration
  - Supported by the information pull interface at the receiver (the host reads the DPRAM independently of the communication controller), which makes the communication overhead very low and highly predictable.
- Constructive integration
  - Integration by extension of existing communication schedule
    - Use empty message slots if possible
    - Otherwise a reconstruction (and revalidation) of the communication schedule is necessary
- Replica determinism
  - Implementation of replica determinism is simplified if all nodes have access to a globally synchronized sparse time base. (Note: This statement is not well explained in the paper.)

Overview of Lecture 16

- Guest lecture by Lars Holmlund, Saab AB, Linköping
- Read *before the lecture:*
  - Section 15.3 in course book.
  - Lecture slides

Overview of Lecture 17

- Time-triggered systems (cont'd)
- Error detection
- Wrap-up and course summary
- Read *before the lecture*:
  - Lecture slides
  - The Time-Triggered Architecture (see reading instructions)