# Disclaimer

- The material provided in this document is not my original work and is a summary of some one else's work(s).
- A simple Google search of the title of the document will direct you to the original source of the material.
- I do not guarantee the accuracy, completeness, timeliness, validity, non-omission, merchantability or fitness of the contents of this document for any particular purpose.
- 

# PERFORMANCE ANALYSIS OF FLEXRAY-BASED ECU NETWORKS

Presented by
Najeeb
2013-6-13

# OUTLINE

- Introduction
- Basic Framework
- Difficulties in Modeling FlexRay
- Modeling FlexRay
- Case Study

# INTRODUCTION

- We will discuss the timing analysis of the Dynamic segment of the FlexRay protocol
- The Dynamic segment is event triggered
- Event-triggered protocols are
  - Efficient in communication bandwidth usage
  - Allow incremental system design
  - Verifying timing properties and detecting faults are very difficult
  - Hence not used for safety-critical and hard real-time systems

# INTRODUCTION

- Given
  - A specification of the tasks running on the different ECUs
  - The scheduling policy used at each ECU
  - A specification of the FlexRay bus (e.g. slot sizes and message priorities)
- This paper answers questions related to
  - Maximum end-to-end delay experienced by any message
  - The amount of buffer required at each communication controller
  - The utilization of the different ECUs and the bus

# RELATED WORK

- Very recently, the first attempt to formally model the behavior of the DYN segment was reported in [1]
- Given the arrival rates of the different message streams mapped onto the DYN segment, [1] computes the worst-case delay experienced by any message due to blocking by the ST segment and contention from higher priority messages
- [1] analyzes the FlexRay bus in isolation, i.e. requires the input rates or periods of the arriving messages and computes the worst-case delay due to transmission over the bus

# RELATED WORK

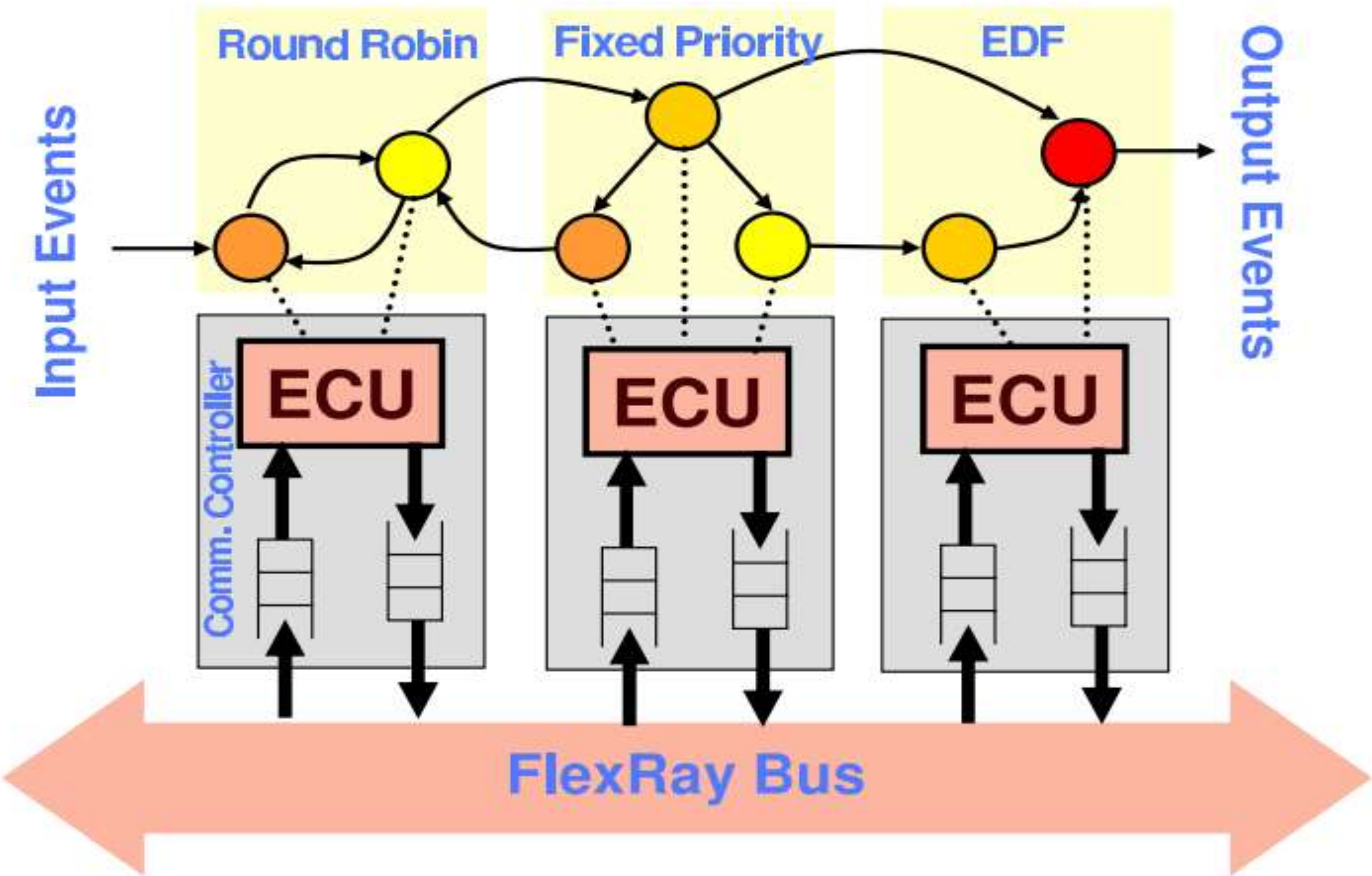
- The framework we present in this paper is fully compositional and models both the ECUs and the FlexRay bus in a seamless manner
- It does not make any a priori assumption on the timing properties of the message streams arriving at the bus
- The frame-work discussed can be used for synthesizing a FlexRay schedule (i.e. determine the slot sizes and message priorities) when maximum end-to-end delays are provided as design constraints

# BASIC FRAMEWORK

- System Architecture
  - Consist of multiple ECUs communicating via a FlexRay bus
  - One or more applications are partitioned into tasks, which are then mapped onto different ECUs
  - ECUs running multiple tasks use a scheduler to share the available processing resources
  - Each task is activated at a certain rate or is triggered by an output from another task
  - Each task consumes a fixed number of processor cycles from the ECU on which it is running

# BASIC FRAMEWORK

- System Architecture



# BASIC FRAMEWORK

- $\alpha^u(\Delta)$ and $\alpha^l(\Delta)$ : maximum and minimum number of times the task can be activated within any interval of length $\Delta$
- $\beta^u(\Delta)$ and $\beta^l(\Delta)$ be upper and lower bounds on the service available to this task
- Let each serviced activation of a task generate a message and $\alpha^{u'}(\Delta)$ and $\alpha^{l'}(\Delta)$ denote upper and lower bounds on the number of such messages generated within any time interval of length $\Delta$

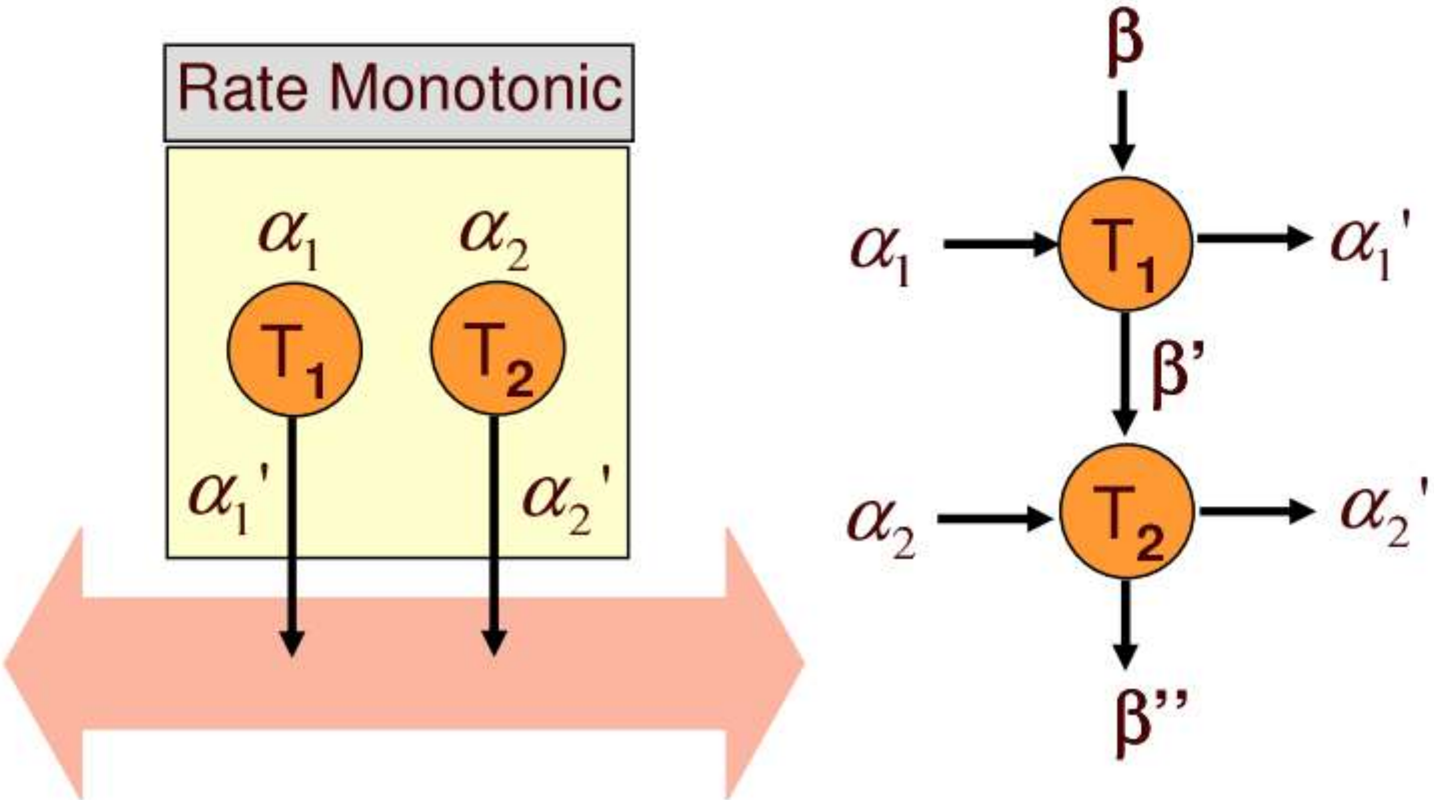
# BASIC FRAMEWORK

- The bounds on the remaining service after processing the activations of this task is given by

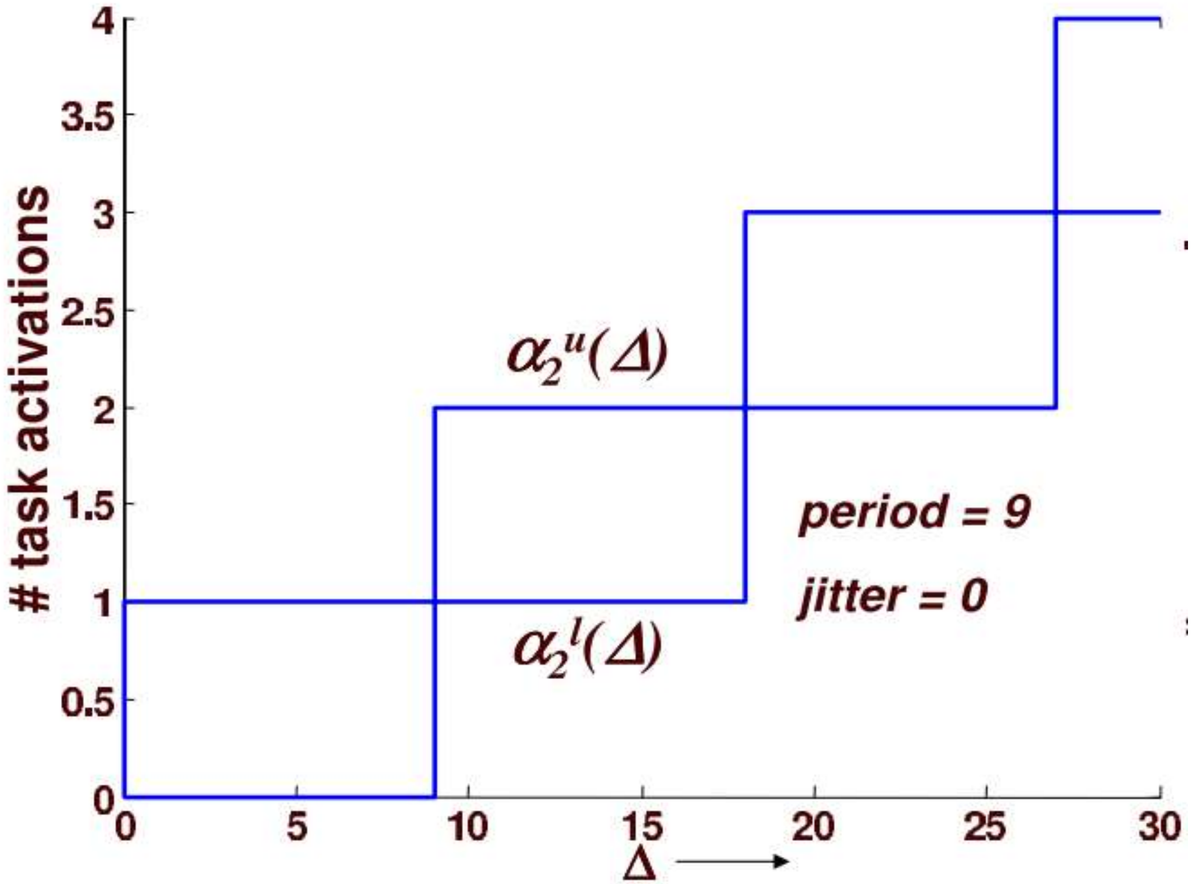
$$\beta^{l'}(\Delta) = \sup_{0 \le \lambda \le \Delta} \{\beta^{l}(\lambda) - \alpha^{u}(\lambda)\}$$

$$\beta^{u'}(\Delta) = \max\{\inf_{\lambda > \Delta} \{\beta^{u}(\lambda) - \alpha^{l}(\lambda)\}, 0\}$$

# BASIC FRAMEWORK

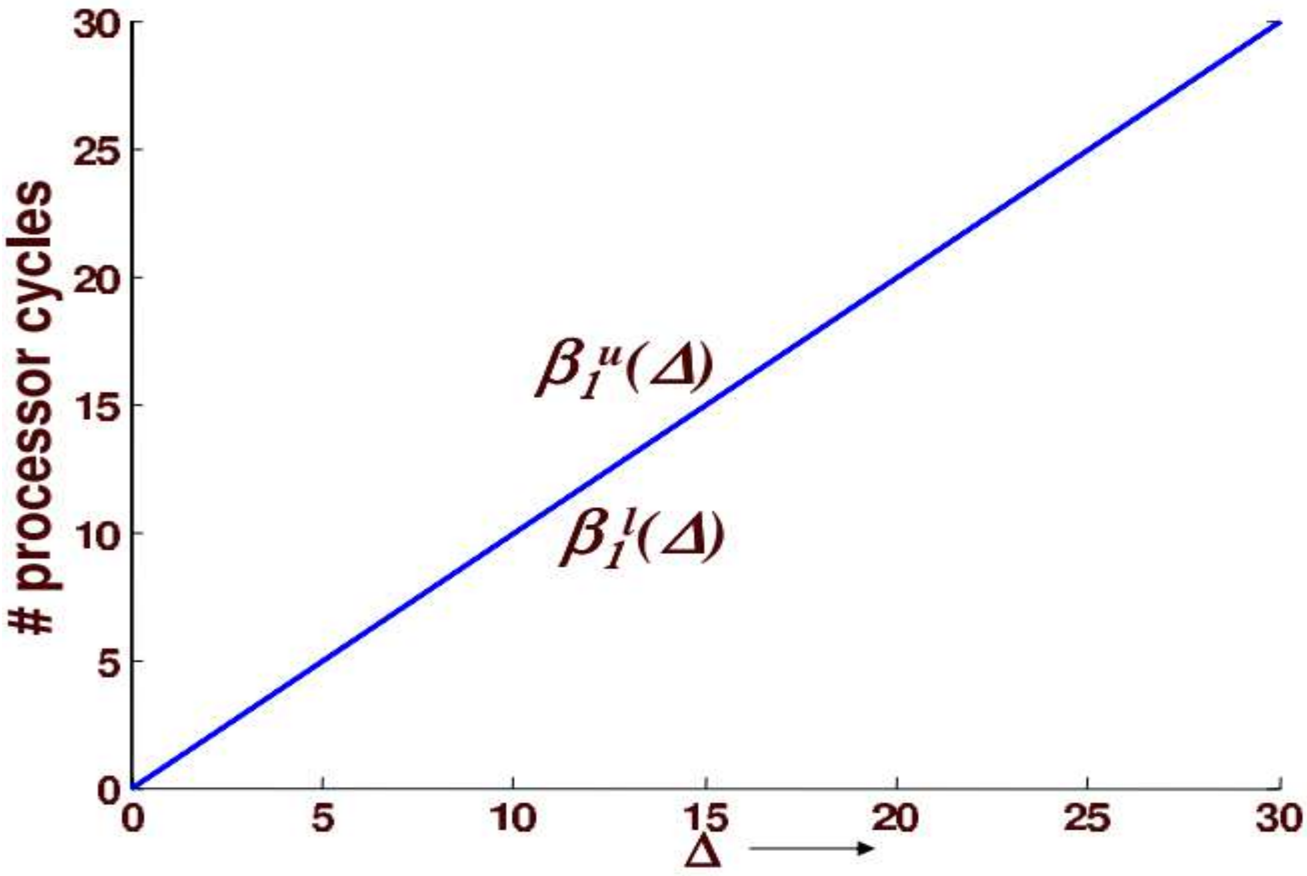


# BASIC FRAMEWORK

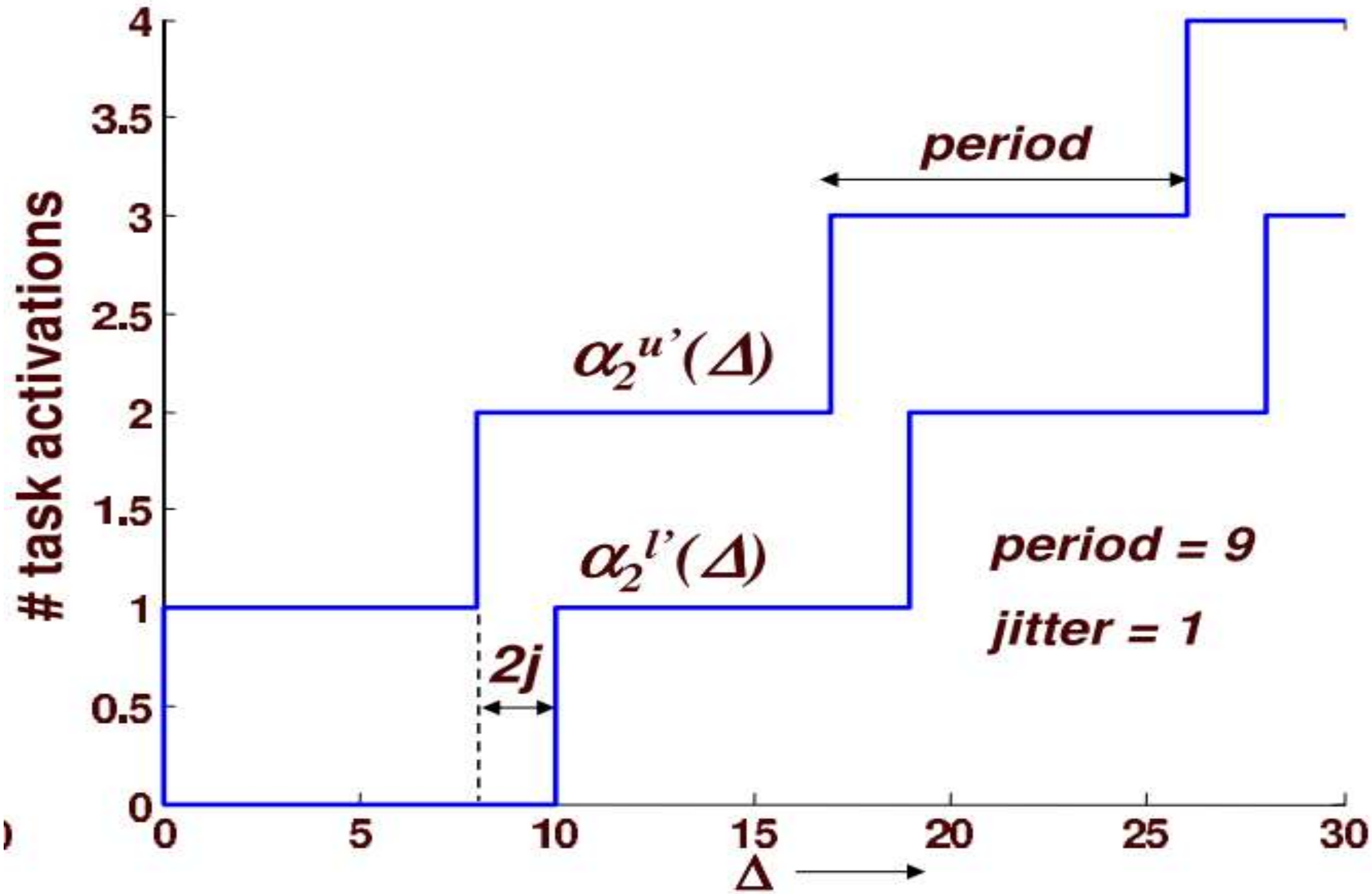


# BASIC FRAMEWORK

- The unloaded service



# BASIC FRAMEWORK



# BASIC FRAMEWORK

- Given $\alpha^u$, $\alpha^l$ and $\beta^u$, $\beta^l$, it is possible to compute the maximum delay experienced by a task before its activation is serviced and the maximum number of backlogged activations

$$\text{delay} \leq \sup_{t\geq 0}\{\inf_{\tau\geq 0}\{\alpha^u(t) \leq \beta^l(t+\tau)\}\}$$

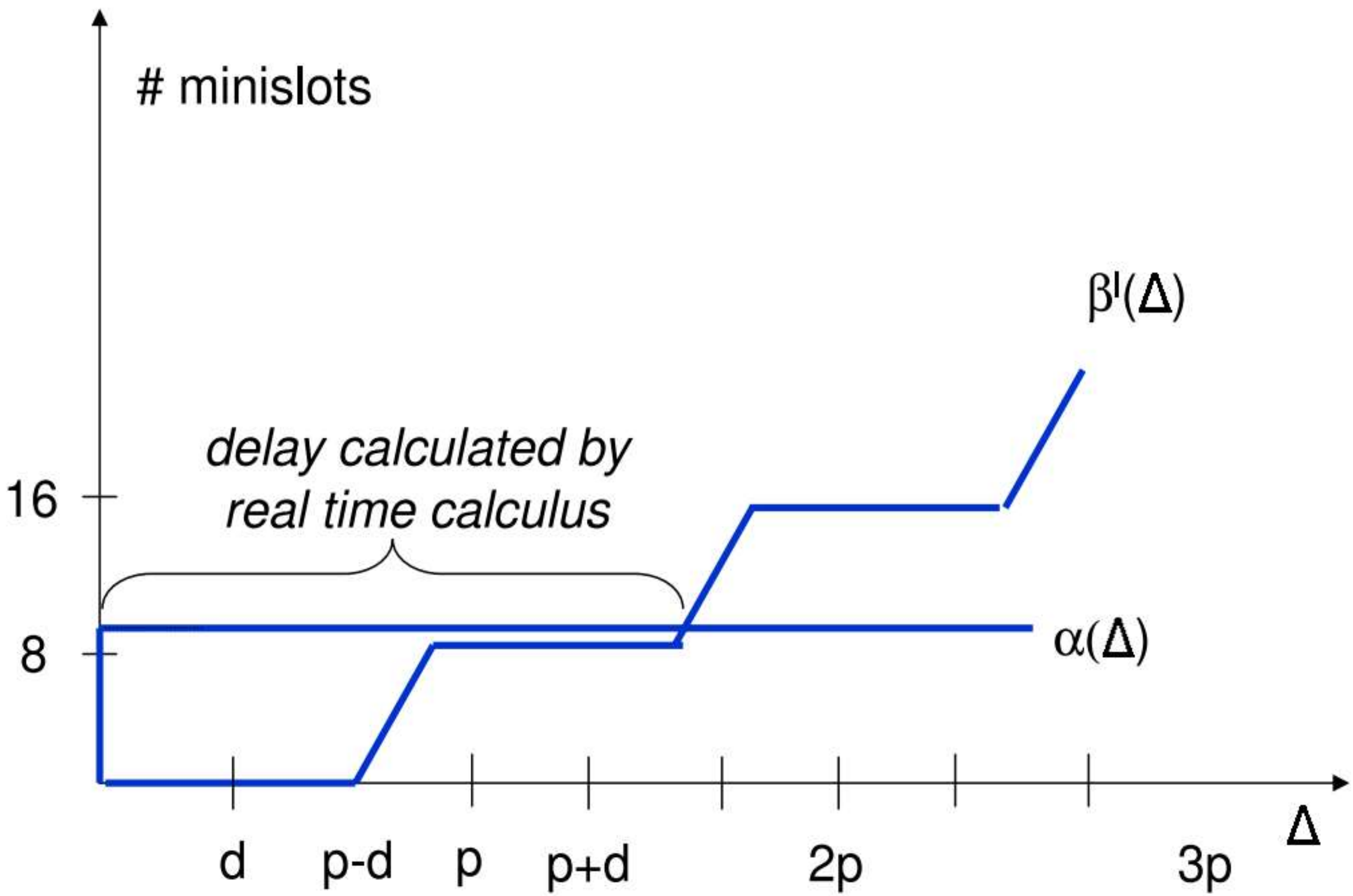
$$\text{backlog} \leq \sup_{t\geq 0}\{\alpha^u(t) - \beta^l(t)\}$$

- Although we have used this framework to analyze a processing element, the same technique is also applicable to communication resources (e.g. buses)

# DIFFICULTIES IN MODELING FLEXRAY

- B would be used to model the total service offered by the DYN segment and successive Bs would be computed from the message sizes and message generation rates of the different tasks
- This approach does not work because of the following properties of FlexRay
  - A task is only allowed to send a message if it fits into the remaining portion of the DYN segment
  - Once a task misses its turn in the DYN segment it has to wait till the next communication cycle before it can access the bus
  - A task can send at most one message in each DYN segment
  - One minislot is consumed from the available service each time a task is not ready to transfer a message before the next task is allowed to send its message on the bus

# DIFFICULTIES IN MODELING FLEXRAY



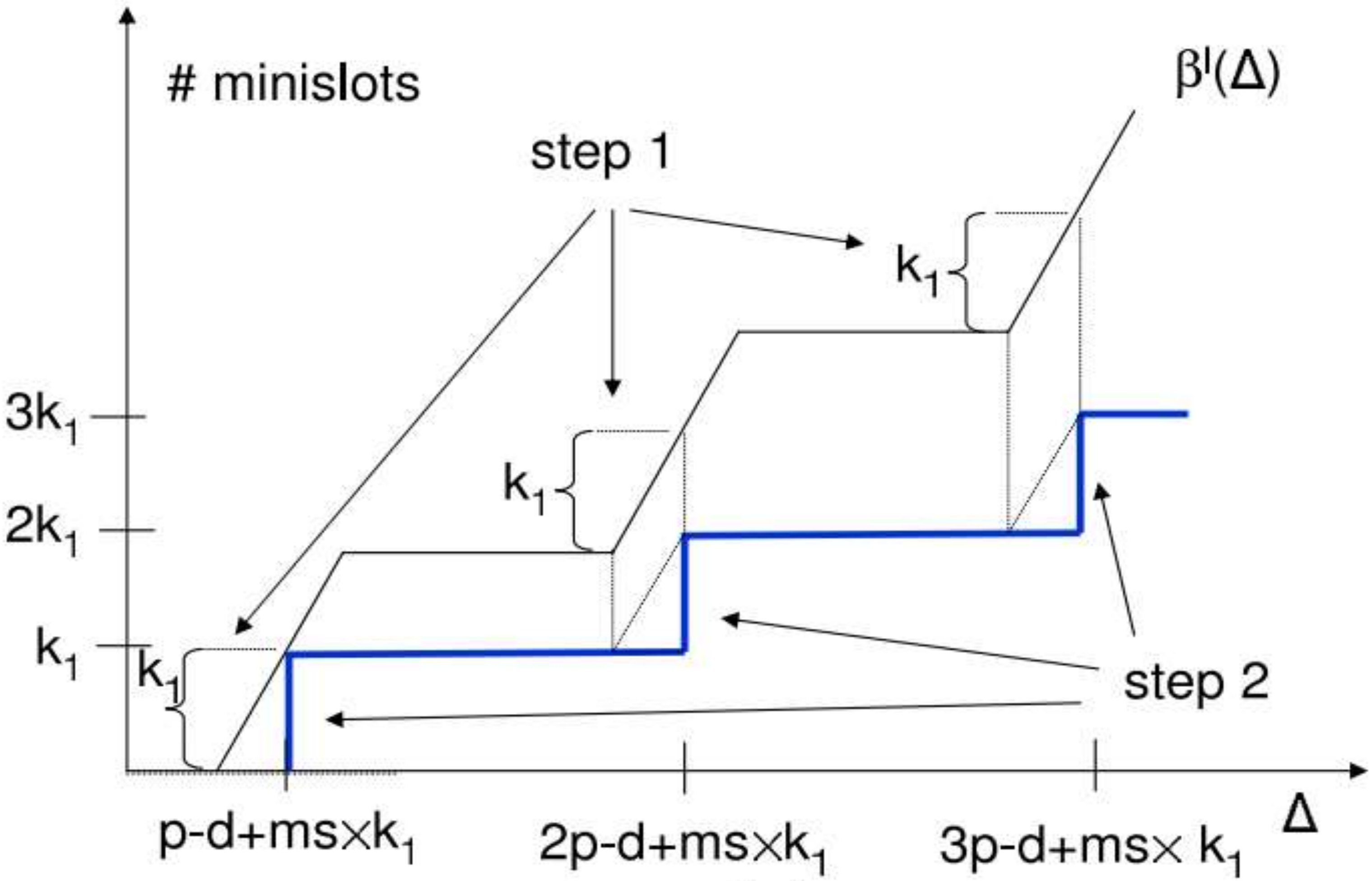
# MODELING FLEXRAY

- FlexRay restricts the amount of available service that can actually be used
- We need to model how much of the service can actually be used

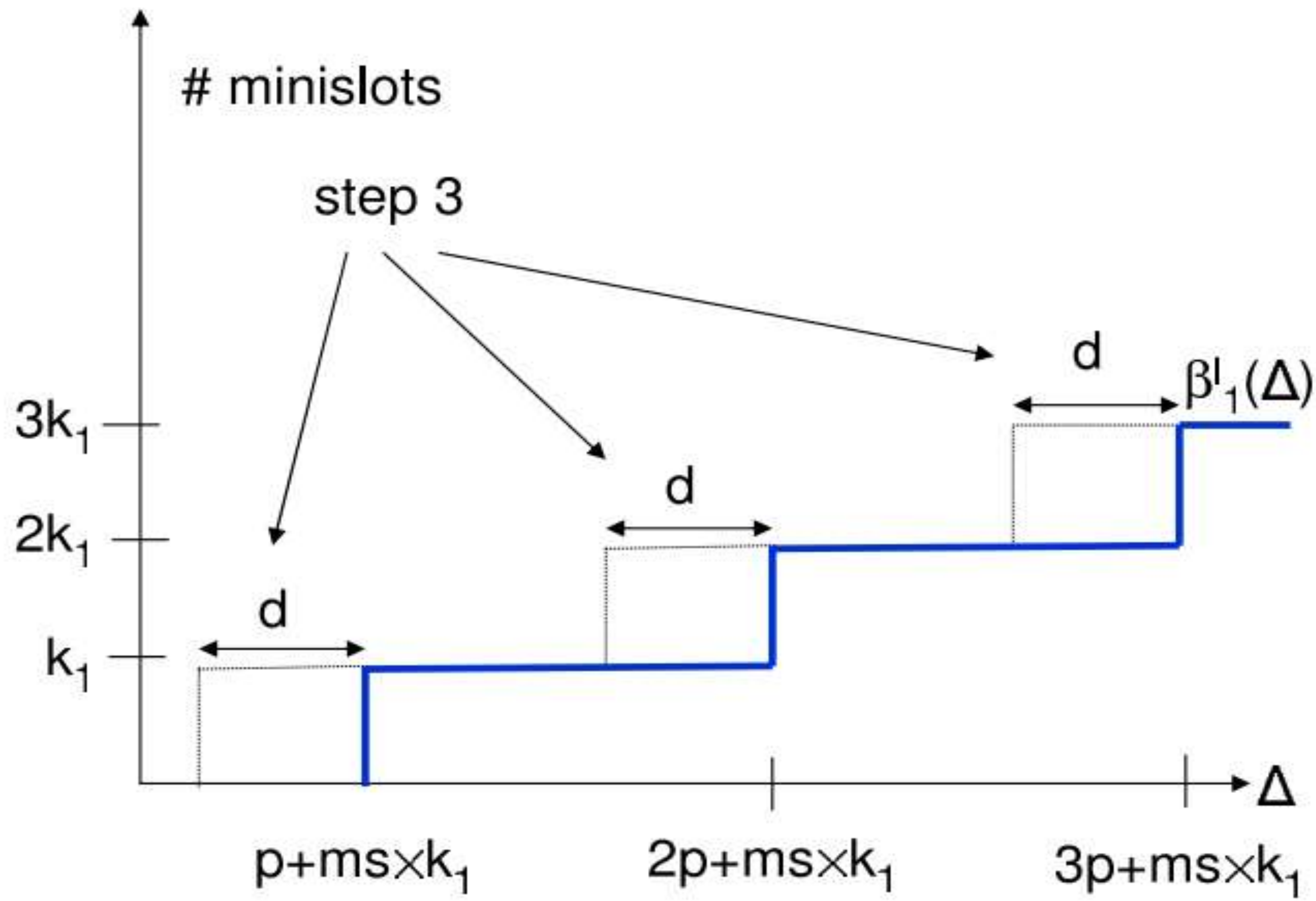
# MODELING FLEXRAY

- To obtain $\beta^{l}_{1}$, the function $\beta^{l}$ needs to be algorithmically transformed
  1. Extract $k_1$ minislots of service during each communication cycle from $\beta_l$
  2. Discretize the service bound obtained from Step 1, i.e. convert it into a step-function
  3. The resulting service bound is shifted by d time units

# MODELING FLEXRAY



# MODELING FLEXRAY

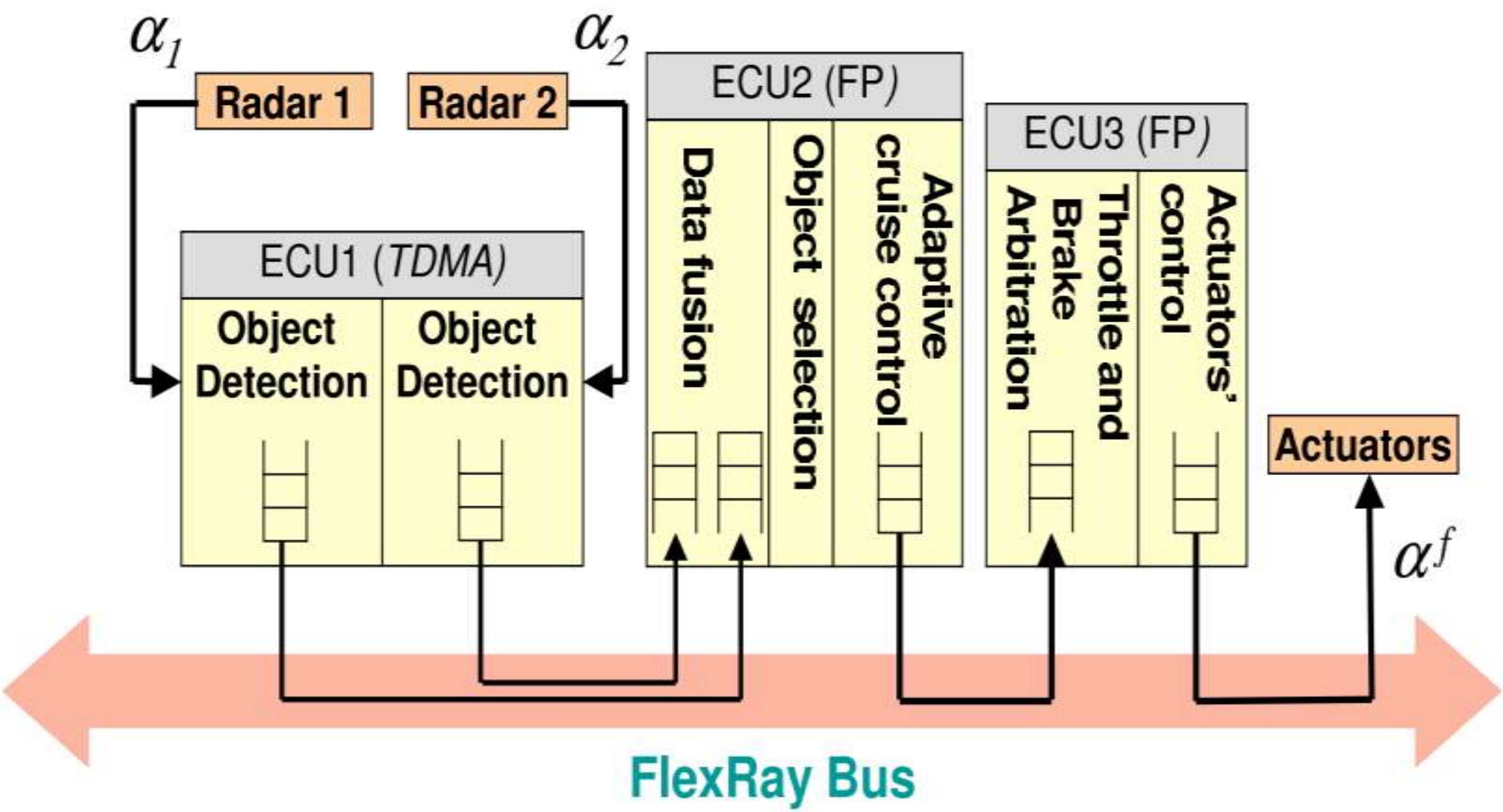


# MODELING FLEXRAY

- The resulting service bound, which we denote as $\beta^{l}_{1}$ correctly represents the minimum or guaranteed service from the DYN segment that is available to messages from T1
- $B^{l}_{1}$ can be used to compute the maximum delay suffered by any $m_1$, the maximum number of backlogged $m_1$s and the timing properties of the transmitted messages

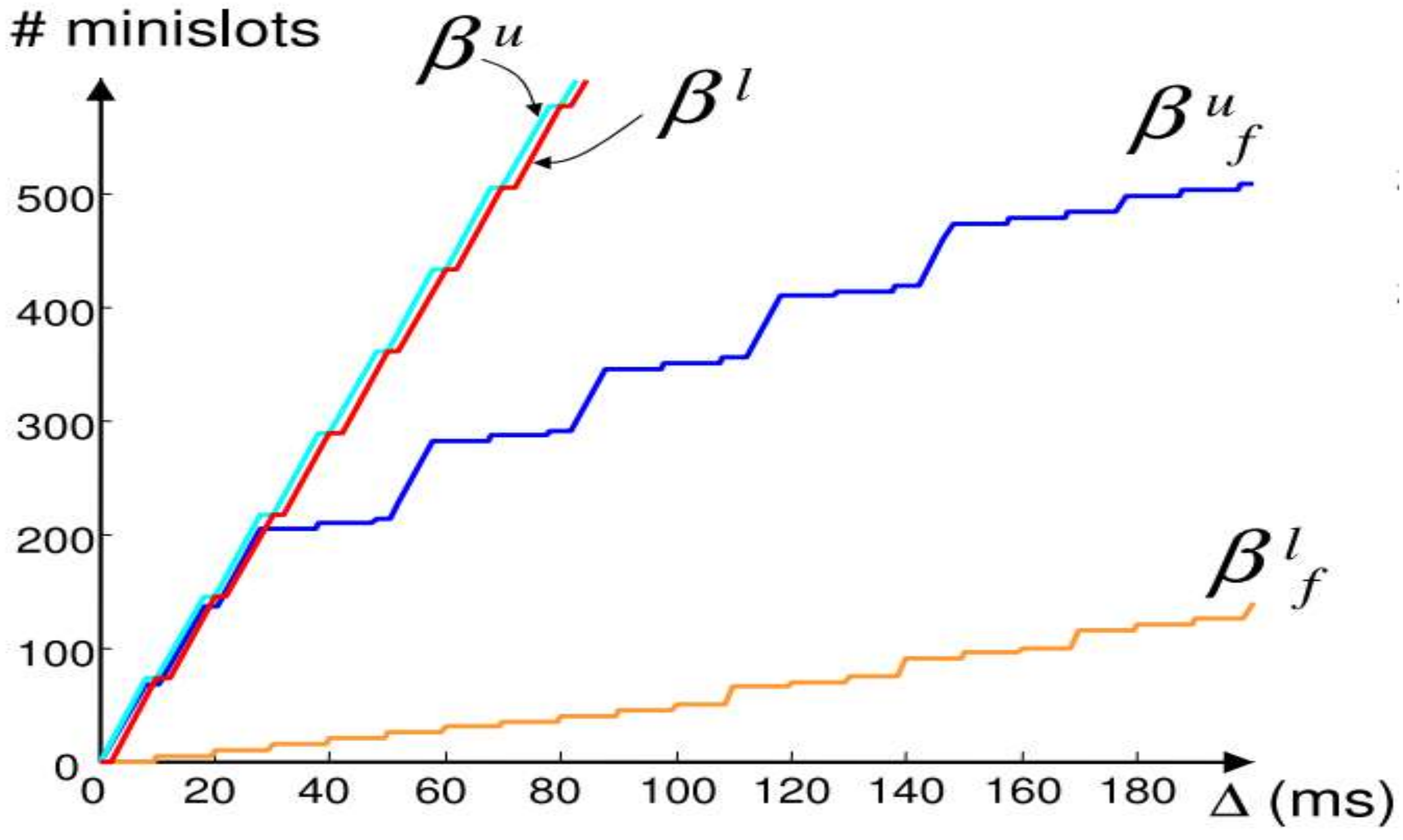
# CASE STUDY

- Adaptive Cruise Control System



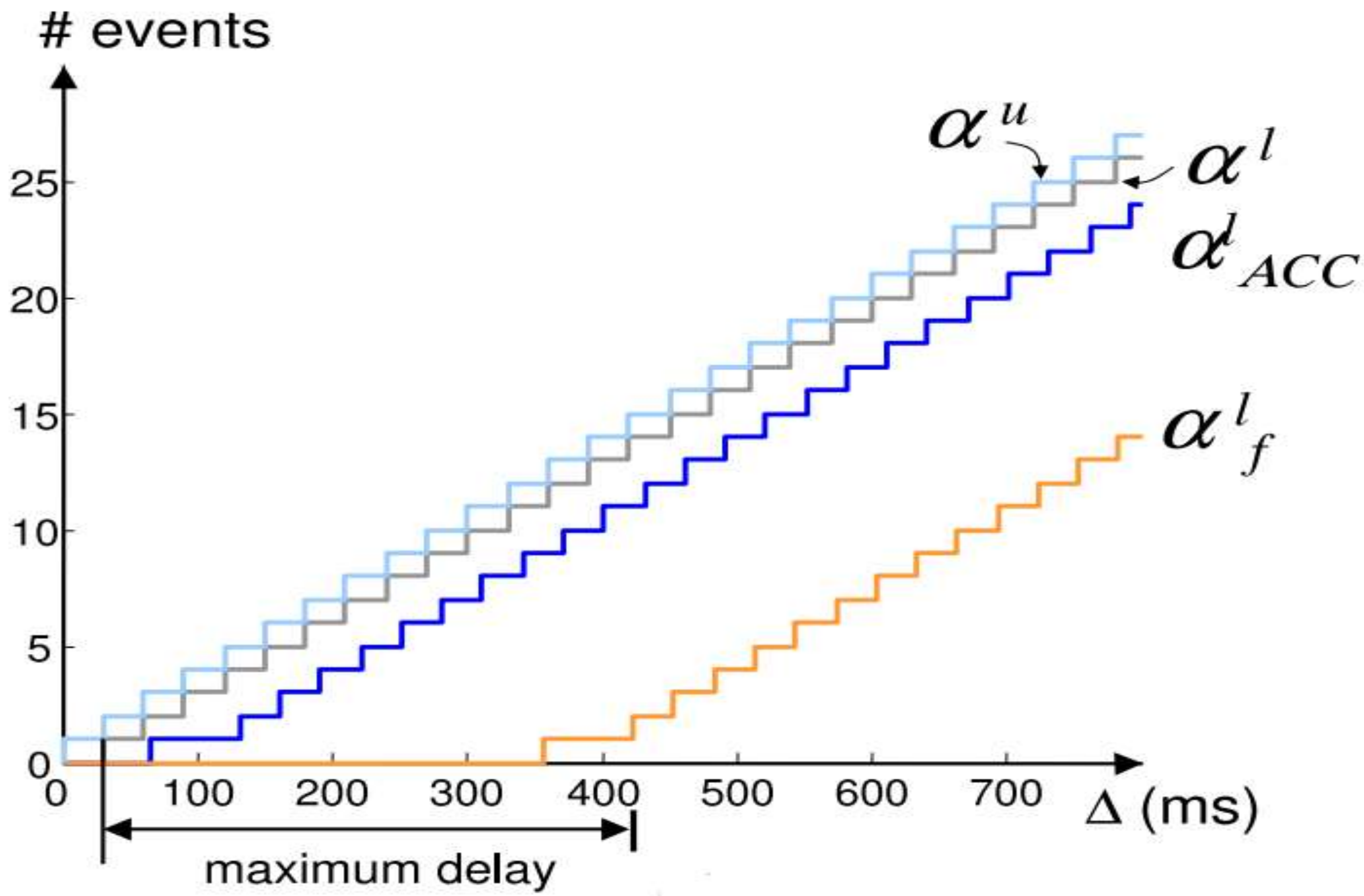
# CASE STUDY

- Service bounds after processing m1-m4



# CASE STUDY

- Bounds on arrival rates of messages



# REFERENCES


1. T. Pop, P. Pop, P. Eles, Z. Peng, and A. Andrei. Timing analysis of the FlexRay communication protocol. In18th Euromicro Conference on Real-Time Systems (ECRTS), 2006.

# Thank You