

This is a repository copy of *AdaptMC: A Control-Theoretic Approach for Achieving Resilience in Mixed-Criticality Systems*.

White Rose Research Online URL for this paper:

<https://eprints.whiterose.ac.uk/133393/>

Version: Accepted Version

---

**Proceedings Paper:**

Papadopoulos, Alessandro, Bini, Enrico, Baruah, Sanjoy et al. (1 more author) (2018) *AdaptMC: A Control-Theoretic Approach for Achieving Resilience in Mixed-Criticality Systems*. In: Altmeyer, Sebastian, (ed.) *Proceeding ECRTS Conference. LIPICS*, Dagstuhl, 14:1-14:22.

---

**Reuse**

Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

**Takedown**

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing [eprints@whiterose.ac.uk](mailto:eprints@whiterose.ac.uk) including the URL of the record and the reason for the withdrawal request.

# 1 **AdaptMC: A Control-Theoretic Approach for** 2 **Achieving Resilience in Mixed-Criticality Systems\***

3 **Alessandro Vittorio Papadopoulos**

4 Mälardalen University

5 [Västerås, Sweden]

6 alessandro.papadopoulos@mdh.se

7 **Enrico Bini**

8 University of Turin

9 [Turin, Italy]

10 bini@di.unito.it

11 **Sanjoy Baruah**

12 Washington University

13 [St. Louis (MO), USA]

14 baruah@wustl.edu

15 **Alan Burns**

16 University of York

17 [York, UK]

18 alan.burns@york.ac.uk

## 19 **Abstract**

20 A system is said to be resilient if slight deviations from expected behavior during run-time  
21 does not lead to catastrophic degradation of performance: minor deviations should result in no  
22 more than minor performance degradation. In mixed-criticality systems, such degradation should  
23 additionally be criticality-cognizant. The applicability of control theory is explored for the design  
24 of resilient run-time scheduling algorithms for mixed-criticality systems. Recent results in control  
25 theory have shown how appropriately designed controllers can provide guaranteed service to hard-  
26 real-time servers; this prior work is extended to allow for such guarantees to be made concurrently  
27 to multiple criticality-cognizant servers. The applicability of this approach is explored via several  
28 experimental simulations in a dual-criticality setting. These experiments demonstrate that our  
29 control-based run-time schedulers can be synthesized in such a manner that bounded deviations  
30 from expected behavior result in the high-criticality server suffering no performance degradation  
31 and the lower-criticality one, bounded performance degradation.

32 **2012 ACM Subject Classification** Computing methodologies → Control methods, Computer  
33 systems organization → Real-time systems

34 **Keywords and phrases** mixed criticality, control theory, run-time resilience, bounded overloads

35 **Digital Object Identifier** 10.4230/LIPIcs.ECRTS.2018.14

## 36 **1 Introduction**

37 There is an increasing trend in embedded systems towards implementing multiple function-  
38 alities upon a shared platform. It may be the case that all these functionalities are not

---

\* This work was partially supported by the Swedish Foundation for Strategic Research under the project  
“Future factories in the cloud (FiC)” with grant number GMT14-0032. This work is also supported by  
NSF grants CNS 1409175, CPS 1446631, and CNS 1563845.



39 equally important to the overall correctness of the embedded system; one widely-studied  
 40 model for representing timing requirements in such systems was proposed by Vestal in a  
 41 seminal paper [33]. Vestal observed that “In many applications, the consequences of missing  
 42 a deadline vary in severity from task to task. In RTCA DO 178B, for example, system  
 43 safety analysis assigns to each task a *criticality level* (ranging from A to D), where erroneous  
 44 behavior by a level A task might cause loss of aircraft but erroneous behavior by a level  
 45 D task might at worst cause inconvenient or suboptimal behavior.”<sup>1</sup> Vestal went on to  
 46 conjecture that “the higher the degree of assurance required that actual task execution times  
 47 will never exceed the WCET parameters used for analysis, the larger and more conservative  
 48 the latter values become in practice.” (This conjecture appears reasonable. Very conservative  
 49 WCET-estimation tools have been developed, typically based upon static analysis of code,  
 50 that yield WCET bounds that may be very large, but that we can trust to a very high level of  
 51 assurance. Less conservative WCET-estimation tools, which are typically measurement based,  
 52 tend to obtain smaller estimates, but these estimates may be trust-worthy to lower levels of  
 53 assurance since the worst-case behaviors of the system may not have become revealed during  
 54 the measurements.) The “Vestal model” for representing, and validating the correctness of,  
 55 mixed-criticality systems is based upon this conjecture. In this model,

- 56 ■ **§1.** A fixed number of distinct criticality levels are defined throughout the system. In  
 57 this paper, we will assume that there are two such criticality levels, designated LO and HI,  
 58 with the interpretation that functionalities designated as being of the LO criticality level  
 59 need to have their correctness validated to a lower level of assurance than functionalities  
 60 designated as being of the HI criticality level.
- 61 ■ **§2.** Each piece of code in the system is characterized as being of one of the criticality levels  
 62 LO or HI, and by two WCET parameter estimates. One WCET estimate is determined  
 63 using tools and techniques consistent with the lower criticality level LO, while the other  
 64 estimate is determined using tools and techniques consistent with the higher criticality  
 65 level HI.
- 66 ■ **§3.** Prior to run-time, the correct timing behavior (e.g., meeting deadlines) of all the  
 67 functionalities are validated under the assumption that each piece of code will execute  
 68 for a duration not exceeding its LO-criticality WCET estimate; in addition, the correct  
 69 timing behavior of the HI-criticality functionalities (but not the LO-criticality ones) are  
 70 validated under the assumption that each piece of code may execute for a duration up to  
 71 its HI-criticality WCET estimate.

## 72 1.1 Verification versus resilience

73 The Vestal approach to modeling and analysis of mixed-criticality systems, as originally  
 74 proposed [33], is concerned solely with *verification* — determining, prior to run-time, whether  
 75 a system will behave correctly during run-time if its run-time behavior is compliant with the  
 76 models used to represent it. Clearly, such pre-runtime verification is desirable in safety-critical  
 77 systems. There is an additional aspect of correctness that is also desirable: the system’s  
 78 run-time behavior should be *resilient* or robust in the event that run-time behavior does not

---

<sup>1</sup> RTCA DO 178B is a guideline dealing with the safety of safety-critical software used in certain avionics systems. Although the term “criticality” typically has a precise technical meaning in most safety standards documents, its use in [33], and subsequent use in much of the mixed-criticality scheduling theory literature, appears to be in a rather general sense as a designation of the level of assurance against failure that is desired. In this paper we are using the term in this more general sense, in keeping with prior literature in mixed-criticality scheduling.

79 conform to the models that were assumed during verification; if this happens, a robust system  
80 design ensures that performance degrades gracefully, if at all. *It is this run-time resilience*  
81 *aspect of system behavior that is the primary focus of this paper.* (While the precise semantics  
82 of graceful degradation should be for a particular system may depend upon the characteristics  
83 of the system, some general principles are applicable; for example, less important aspects of  
84 system functionalities should be compromised before more important ones.)

85 The Vestal model of [33] and its derivatives and generalizations have formed the basis of  
86 a large body of research: schedulability tests, scheduling algorithms, etc. — see, e.g., [5, 6]  
87 for a survey. Much of this research is focused upon the pre-runtime verification aspect  
88 of correctness rather than the run-time resilience. For instance, many mixed-criticality  
89 scheduling algorithms allow for LO-criticality pieces of code to be aborted if any piece of code  
90 executes beyond its LO-criticality WCET estimate. Such a scheduling algorithm may still  
91 pass the pre-runtime verification test (since such tests are only concerned with the correctness  
92 of the HI-criticality functionalities under such circumstances), but would not be considered  
93 resilient. Some recent research has attempted to provide some resilience to LO-criticality  
94 pieces of code in the event of some piece of code executing beyond its LO-criticality WCET  
95 estimate; these approaches are reviewed in Section 7.

## 96 1.2 This research

97 In this paper, we explore the use of control-theoretic principles to achieve resilience in mixed-  
98 criticality systems. We consider over-runs of HI-criticality pieces of code (in the sense of them  
99 executing for more than their LO-criticality WCET estimates) to be *rare events* that are best  
100 coped with by run-time adaptability. Some over-runs can be masked by under-runs by other  
101 HI-criticality pieces of code; others will require system-wide adaptation. These adaptations  
102 should be commensurate with the scale of the over-run — dropping all LO-criticality pieces  
103 of code because a single HI-criticality piece of code has executed for slightly more than  
104 its LO-criticality WCET is clearly an over-reaction. A resilient system should cope with  
105 uncertainty in a measured way.

106 Some recent advances in real-time control (see, e.g., [22] and the references therein)  
107 have motivated us to explore whether the desired resilience can be achieved using a control-  
108 theoretic approach. The scheduling strategy we propose has the HI-criticality workload  
109 executing within an execution-time server that is provisioned with a budget sufficient to  
110 satisfy the LO-criticality WCET requirements of this HI-criticality workload; another, similar,  
111 server is used to encapsulate the execution requirements of the LO-criticality workload. At  
112 run-time if the HI-criticality server's budget proves inadequate for meeting the execution  
113 requirements of the HI-criticality workload (due to some HI-criticality pieces of code executing  
114 for more than their LO-criticality WCET estimates) then the system is deemed to have  
115 suffered a *disturbance* or *perturbation*. We employ a *control feedback* mechanism to govern  
116 budget allocations going forward from the disturbance. This control-theoretic feedback  
117 approach allows a number of questions to be answered concerning the run-time behavior of  
118 the scheduling strategy, such as

- 119 ■ How long following a disturbance will it take the system to return to a non-perturbed  
120 state?
- 121 ■ What guaranteed level of service can be obtained for the LO-criticality workload?
- 122 ■ What is the maximum *magnitude* of disturbance that can be accommodated allowing for  
123 stable control and for the HI-criticality workload to remain schedulable?

124     **1.3    Organization**

125     The remainder of this paper is organized as follows. Section 2 presents the background for  
 126     this work, while Section 3 presents AdaptMC, the proposed approach, in detail. Section 4  
 127     discusses how AdaptMC is designed and tuned, while Section 5 presents how hard real-time  
 128     guarantees can be provided, by means of the calculation of the supply bound function.  
 129     Section 6 presents a numerical evaluation of AdaptMC. Section 7 reviews the related work,  
 130     while Section 8 concludes the paper.

131     **2        Background**

132     The use of feedback control to allocate resources has traditionally been applied to time-  
 133     varying workloads [28, 7, 1], and the kinds of offered guarantees have been probabilistic or  
 134     soft real-time. Recently, however, a control scheme called the *Self-Adaptive Server (SAS)* has  
 135     been proposed [22], that provides both good average behavior and hard real-time guarantees.  
 136     Such a guarantee is given by computing the supply bound function [21, 18, 27, 2] of a periodic  
 137     resource supply controlled by feedback [17].

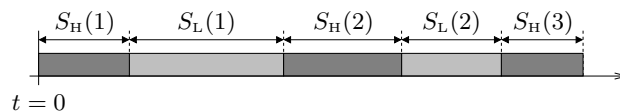
138     The main idea behind SAS is as follows. Each server in the system is assigned a budget  
 139     of time to execute. The server is allowed execute more or less than the budget, but at the  
 140     next round it will be assigned a budget that is corrected with a term that is proportional to  
 141     the over- or under-run of the server. In [22] this simple, yet effective, control structure is  
 142     analyzed under the assumption that the maximum over- or under-run are bounded. The  
 143     designed controller is proven to effectively adapt the budget at run-time, while the supply  
 144     bound function associated with the controller can be computed offline.

145     **3        The Proposed Approach**

146     We are concerned with mixed-criticality systems in which the LO-criticality WCET values  
 147     represent typical or common-case behavior: executions *rarely* exceed these WCET values  
 148     and when they do, it is typically *by small amounts*. We seek to devise resilient scheduling  
 149     strategies for such mixed-criticality systems. As briefly stated in Section 1, our proposed  
 150     scheduling strategy uses two servers, one each for servicing the HI-criticality and LO-criticality  
 151     workloads.<sup>2</sup> In dimensioning these servers’ budgets, our objective will be to modestly over-  
 152     allocate the HI-criticality server in the sense that “most of the time” we would expect the  
 153     entire provisioned budget to not be needed. If an occasional modest over-run occurs in  
 154     the amount of execution required by the HI-criticality server (say, by an amount  $x$  over  
 155     the budgeted amount), our run-time scheduling strategy is to allow the HI-criticality server  
 156     to over-execute by this entire amount  $x$ , and then reduce the budget for the LO-criticality  
 157     server by an amount somewhat smaller than  $x$ . Informally speaking, the hope is that after  
 158     dealing with this one-time over-run, the HI-criticality server will not need to use its entire  
 159     budgeted amount for some duration, and hence can compensate the LO-criticality server over

---

<sup>2</sup> For the kinds of application systems that we are interested in, work (in the form of “jobs”) is typically generated by recurrent – periodic and sporadic – tasks; determining appropriate budget and period parameters for servers capable of accommodating the computational requirements of such recurrent tasks is an important issue that has been widely studied in the real-time scheduling community [18, 27, 2]. However, the issue of dimensioning such servers is orthogonal to the focus of this paper and we will not discuss it further, instead assuming that some appropriate scheme is used to determine appropriate server parameters such that if all jobs execute at their LO-criticality WCET estimates, then each server is able to correctly execute those jobs for which it is responsible.



■ **Figure 1** Server schedule over time.

160 this duration. However, (as we will see) our control-based scheduling strategy is robust to  
 161 scenarios in which the HI-criticality server over-runs for an extended duration as well; if this  
 162 happens, the LO-criticality server ends up getting under-served over an extended duration.

163 In order to develop a control-based strategy capable of achieving these goals, we needed  
 164 to extend and adapt SAS (Self-Adaptive Server) [22] in several directions. The feedback  
 165 mechanism derived in this paper is an extension of SAS to the mixed-criticality context that  
 166 enables:

- 167 1. the adjustment of server budgets based on disturbances at both HI-criticality and LO-  
 168 criticality servers (achieved by *cross gains* of the controller), and
- 169 2. the exploitation of the asymmetric nature of disturbances that are permitted for the  
 170 LO-criticality server (which may occasionally be under-served but never receives more  
 171 than its budgeted amount) to provide less conservative supply bound functions.

172 The presence of these two characteristics, needed in the mixed-criticality context, renders  
 173 the results in [22] inapplicable directly; hence the extensions reported here. Section 3.1  
 174 below describes the adaptive scheduling strategy we have developed; the control algorithm  
 175 underpinning this strategy is described in Section 3.2

### 176 3.1 Run-Time Scheduling Strategy

177 We propose a 2-levels hierarchical scheduler with two schedulers at the top level, one for  
 178 servicing LO-criticality work and the other, for servicing HI-criticality work (see Figure 1).  
 179 Let  $\bar{Q}_H$  and  $\bar{Q}_L$  denote the *target budgets* for the two servers, and  $\bar{P} = \bar{Q}_H + \bar{Q}_L$  the *target*  
 180 *period*. We will describe later the manner in which values are assigned to these target  
 181 budget parameters; intuitively speaking, we would assign them values such that under  
 182 normal circumstances (i.e., all jobs completing within their LO-criticality WCET estimates)  
 183 a periodic schedule with period  $\bar{P}$  in which the HI-criticality server executing for a duration  
 184  $\bar{Q}_H$  is followed by the LO-criticality server executing for a duration  $\bar{Q}_L$ , would meet all timing  
 185 requirements for all the HI-criticality and the LO-criticality workload.

186 During run-time these two servers are repeatedly scheduled alternately. Let us refer to  
 187 the  $k$ 'th time that both servers are scheduled as the  $k$ 'th *round*. Let  $Q_H(k)$  and  $Q_L(k)$  denote  
 188 the *tentative budgets* that the control algorithm computes at the end of the  $k$ 'th round, for  
 189 allocating to the two servers for the  $(k + 1)$ 'th round. Initially, we have  $Q_H(0) = \bar{Q}_H$  and  
 190  $Q_L(0) = \bar{Q}_L$ ; i.e., for the first round the tentative budgets are set to be equal to the target  
 191 budgets.

192 Now suppose that during the  $(k + 1)$ 'th round for some  $k$ , the HI-criticality server needs to  
 193 execute for a duration greater than this tentative budget  $Q_H(k)$  in order to ensure the correct  
 194 execution of all HI-criticality jobs (budget overrun). We allow it to do so, and let  $S_H(k + 1)$   
 195 denote the duration for which it executes —  $S_H(k + 1)$  is called the *actual budget* assigned  
 196 to the HI-criticality server during the  $(k + 1)$ 'th round, and  $\varepsilon_H(k) = (S_H(k + 1) - Q_H(k))$  is  
 197 called the *disturbance* experienced by the HI-criticality server, i.e., the discrepancy between  
 198 the target and actual budget. In response to such a disturbance, our control algorithm

199 modifies the tentative budgets  $Q_H(k+1)$  and  $Q_L(k+1)$  computed for both servers for the  
 200 next round, to compensate for the budget overrun and preserve the bandwidth.

### 201 3.2 The Control Algorithm

202 As stated earlier, our control-based scheduler is designed under the assumption that jobs  
 203 executing beyond their LO-criticality WCET estimates will be rare events. The target budget  
 204  $\bar{Q}_H$  for the HI-criticality server should be chosen to somewhat exceed the minimum needed in  
 205 order to accommodate the LO-criticality WCET requirements for all the HI-criticality jobs;  
 206 hence, if only one or a few jobs over-run their LO-criticality WCETs during a round, such  
 207 over-runs are often masked by the excess budget and by under-runs of other HI-criticality  
 208 jobs. It should only rarely be the case that such over-runs during any round get expressed as  
 209 disturbances (i.e., as an  $\varepsilon_H(k)$  value for some  $k$ ); in the rare events when this does happen,  
 210 our control algorithm requires that it be of magnitude that is bounded by an a priori known  
 211 constant  $\bar{\varepsilon}_H$ :  $|\varepsilon_H(k)| \leq \bar{\varepsilon}_H$ .

212 In order to accommodate these disturbances in the HI-criticality servers, our control  
 213 algorithm will occasionally under-schedule the LO-criticality server, providing it a supply  
 214  $S_L(k+1)$  that is strictly less than the tentative budget  $Q_L(k)$  that had been computed  
 215 for it — when this happens, the LO-criticality server is said to experience a disturbance  
 216  $\varepsilon_L(k) = (S_L(k+1) - Q_L(k))$ . We assume that such a disturbance will also be of magnitude  
 217 that is bounded by another a priori known constant  $\bar{\varepsilon}_L$ , i.e., maximum budget over-run of  
 218 the LO-criticality server.

219 Analogously, our run-time scheduler also bounds the “negative” disturbance to the HI-  
 220 criticality server: the amount by which the actual amount of execution supplied during a  
 221 round is less than the tentative budget, to have a magnitude no greater than  $\bar{\varepsilon}_H$ . Summarizing  
 222 the above discussion on disturbances, we obtain the following bounds on the magnitudes of  
 223 the disturbances that could be experienced by both the servers:

$$224 \quad -\bar{\varepsilon}_H \leq \varepsilon_H(k) \leq \bar{\varepsilon}_H, \quad -\bar{\varepsilon}_L \leq \varepsilon_L(k) \leq 0. \quad (1)$$

225 As we had stated earlier, the actual budgets  $S_H(k+1)$  and  $S_L(k+1)$  assigned to the servers  
 226 may be expressed as being equal to the computed tentative budgets  $Q_H(k)$  and  $Q_L(k)$ , plus  
 227 the disturbances  $\varepsilon_H(k)$  and  $\varepsilon_L(k)$ .

$$228 \quad S_H(k+1) = Q_H(k) + \varepsilon_H(k)$$

$$229 \quad S_L(k+1) = Q_L(k) + \varepsilon_L(k)$$

230 The tentative budgets  $Q_H(k+1)$  and  $Q_L(k+1)$  that are computed by the control algorithm  
 231 may similarly be expressed as the sum of tentative budgets computed for the previous round  
 232 and a corrective term (called the “control signal”) denoted  $u_H(k)$  and  $u_L(k)$ , that is computed  
 233 by the control algorithm at the end of each round:

$$234 \quad Q_H(k+1) = Q_H(k) + u_H(k)$$

$$235 \quad Q_L(k+1) = Q_L(k) + u_L(k)$$

236 Letting

$$237 \quad \mathbf{x}(k) = \begin{bmatrix} S_H(k) \\ S_L(k) \\ Q_H(k) \\ Q_L(k) \end{bmatrix}, \quad \mathbf{u}(k) = \begin{bmatrix} u_H(k) \\ u_L(k) \end{bmatrix}, \quad \boldsymbol{\varepsilon}(k) = \begin{bmatrix} \varepsilon_H(k) \\ \varepsilon_L(k) \end{bmatrix},$$

one can express the *control system dynamics* — the change in values of the actual and tentative budgets across rounds that we have discussed above — in a more compact form, as follows:

$$\mathbf{x}(k+1) = \overbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}^{\mathbf{A}} \mathbf{x}(k) + \overbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}^{\mathbf{B}_u} \mathbf{u}(k) + \overbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}^{\mathbf{B}_\varepsilon} \boldsymbol{\varepsilon}(k). \quad (2)$$

We now discuss how the control signals are computed by the control algorithm (this computation is commonly referred to as the *control strategy*). In designing the controller, we assign values to four real-valued *gain* parameters  $K_{\text{HH}}$ ,  $K_{\text{HL}}$ ,  $K_{\text{LH}}$ , and  $K_{\text{LL}}$  — the parameter design is discussed in Section 4 — and compute the control signals as follows:

$$\begin{aligned} u_{\text{H}}(k) &= K_{\text{HH}}(\bar{Q}_{\text{H}} - S_{\text{H}}(k)) &+ K_{\text{HL}}/\gamma(\bar{Q}_{\text{L}} - S_{\text{L}}(k)), \\ u_{\text{L}}(k) &= \gamma K_{\text{LH}}(\bar{Q}_{\text{H}} - S_{\text{H}}(k+1)) &+ K_{\text{LL}}(\bar{Q}_{\text{L}} - S_{\text{L}}(k)). \end{aligned} \quad (3)$$

The parameters  $K_{\text{HH}}$ ,  $K_{\text{HL}}$ ,  $K_{\text{LH}}$ , and  $K_{\text{LL}}$  weigh the discrepancy between the target and actual budgets; the values assigned to these parameters reflect the effect each discrepancy has on the control signal. (Observe that in computing the control signal  $u_{\text{L}}(k)$  that will be applied to the LO-criticality server, we are able to exploit the fact that the value of  $S_{\text{H}}(k+1)$  is already known when the LO-criticality server is scheduled during the  $(k+1)$ 'th round; we therefore choose to exploit this fact to compute a “better” values for  $u_{\text{L}}(k)$ .)

By substituting the control strategy as represented by Eqn (3) into Eqn (2), rearranging terms, and letting  $\gamma$  denote the ratio of the target budgets, i.e.,  $\gamma = \bar{Q}_{\text{L}}/\bar{Q}_{\text{H}}$ , the closed-loop system dynamics may be represented as follows:

$$S_{\text{H}}(k+1) = Q_{\text{H}}(k) + \varepsilon_{\text{H}}(k) \quad (4)$$

$$S_{\text{L}}(k+1) = Q_{\text{L}}(k) + \varepsilon_{\text{L}}(k) \quad (5)$$

$$Q_{\text{H}}(k+1) = Q_{\text{H}}(k) + K_{\text{HH}}(\bar{Q}_{\text{H}} - S_{\text{H}}(k)) + K_{\text{HL}}/\gamma(\bar{Q}_{\text{L}} - S_{\text{L}}(k)) \quad (6)$$

$$Q_{\text{L}}(k+1) = Q_{\text{L}}(k) + K_{\text{LL}}(\bar{Q}_{\text{L}} - S_{\text{L}}(k)) + K_{\text{LH}}\gamma(\bar{Q}_{\text{H}} - S_{\text{H}}(k+1)) \quad (7)$$

or, in a more compact way:

$$\mathbf{x}(k+1) = \mathbf{A}_{\text{CL}} \mathbf{x}(k) + \mathbf{B}_Q \bar{\mathbf{Q}} + \mathbf{B}_{\varepsilon, \text{CL}} \boldsymbol{\varepsilon}(k) \quad (8)$$

with

$$\mathbf{A}_{\text{CL}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -K_{\text{HH}} & -\frac{K_{\text{HL}}}{\gamma} & 1 & 0 \\ 0 & -K_{\text{LL}} & -\gamma K_{\text{LH}} & 1 \end{bmatrix}, \quad \mathbf{B}_Q = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ K_{\text{HH}} & \frac{K_{\text{HL}}}{\gamma} \\ \gamma K_{\text{LH}} & K_{\text{LL}} \end{bmatrix},$$

$$\bar{\mathbf{Q}} = \begin{bmatrix} \bar{Q}_{\text{H}} \\ \bar{Q}_{\text{L}} \end{bmatrix}, \quad \mathbf{B}_{\varepsilon, \text{CL}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ -\gamma K_{\text{LH}} & 0 \end{bmatrix}.$$

The eigenvalues of  $\mathbf{A}_{\text{CL}}$  determine the convergence time towards the value  $\bar{\mathbf{x}}$  for the system state. These can be obtained from the characteristic polynomial of  $\mathbf{A}_{\text{CL}}$ :

$$p(z) = z^4 - 2z^3 + (K_{\text{HH}} + K_{\text{LL}} + 1)z^2 - (K_{\text{HH}} + K_{\text{LL}} + K_{\text{HL}}K_{\text{LH}})z + K_{\text{HH}}K_{\text{LL}}. \quad (9)$$



272 Since the considered system is linear, we can use the superposition principle<sup>3</sup>, and consider  
 273 separately the effect of  $\bar{Q}$  and  $\varepsilon$  on the evolution of  $\mathbf{x}$ . The  $z$ -transform of (8) is:

$$274 \quad \mathbf{X}(z) = (z\mathbf{I} - \mathbf{A}_{CL})^{-1} (\mathbf{x}(0) + \mathbf{B}_Q \bar{Q} + \mathbf{B}_{\varepsilon,CL} \mathbf{E}(z)) \quad (10)$$

276 Evaluating the transfer function from the error  $\varepsilon$  to the state  $x$  for  $z = 1$  computes, in control  
 277 theoretical terms, the asymptotic effect of the unitary constant disturbance  $\varepsilon$  on the state  $x$ ;  
 278 in the considered case, evaluating  $(z\mathbf{I} - \mathbf{A}_{CL})^{-1} \mathbf{B}_{\varepsilon,CL}$  for  $z = 1$  yields:

$$279 \quad (\mathbf{I} - \mathbf{A}_{CL})^{-1} \mathbf{B}_{\varepsilon,CL} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}$$

281 that proves that the effect of  $\varepsilon$  on  $\mathbf{S}$  (the first two rows) vanishes asymptotically to zero  
 282 independently of the values assigned to the gain parameters. The effect of a unitary constant  
 283 disturbance on the budgets  $\mathbf{Q}$ , on the other hand, is to compensate  $\varepsilon$  by reducing the budget  
 284 of exactly a unity so that value of  $\mathbf{S}$  will compensate perfectly the disturbance  $\varepsilon$ .

## 285 4 Designing the Control Algorithm

286 In Section 3 we described how the control logic can be used to adjust the resource budgets  
 287 allocated to HI and LO-criticality servers. In this section, we are going to explore the  
 288 assignment of values to the control gain parameters  $K_{HH}$ ,  $K_{HL}$ ,  $K_{LH}$ , and  $K_{LL}$  such that the  
 289 resulting budget dynamics are guaranteed to possess the desirable control-theoretic properties  
 290 of *compensation* and *stability*.

291 ► **Definition 1** (Compensation property). A single disturbance  $\varepsilon(k)$  on the HI/LO-criticality  
 292 server results in an opposite or null effect on the value of  $S(k+1)$  (i.e., the actual budget) of  
 293 the LO/HI-criticality server, i.e.,

$$294 \quad \exists n > 0 : \varepsilon_i(k) = -\alpha(k+n)u_j(k+n), \quad \alpha(k+n) \geq 0, i, j \in \{\text{H}, \text{L}\}, \text{ and } i \neq j.$$

296 The intuition of the compensation property is that whenever the HI-criticality server exceeds  
 297 its budget ( $S_H(k+1) > Q_H(k)$ ), the LO-criticality server compensates for this disturbance by  
 298 temporarily reducing its budget. On the other hand, when the LO-criticality server requires  
 299 less time for its execution ( $S_L(k+1) < Q_L(k)$ ), then the HI-criticality server will be allowed  
 300 to temporarily increase its budget. Finally, when the HI-criticality server executes for less  
 301 time ( $S_H(k+1) < Q_H(k)$ ), then the LO-criticality server can temporarily increase its budget.

302 The overall objective is to both preserve the bandwidth of the two servers, and to reach  
 303 the target period  $\bar{P} = \bar{Q}_H + \bar{Q}_L$ .

304 ► **Theorem 2.** *If*

$$305 \quad K_{HH} > 0, K_{HL} \geq 0, K_{LH} \geq 0, K_{LL} > 0, \quad (11)$$

306 *then the system (8) exhibits the compensation property.*

<sup>3</sup> The *superposition principle* for linear systems states that the net response caused by multiple stimuli upon such a system is equal to the sum of the responses that would have been caused by each individual stimulus.

307 **Proof.** First, let us consider the case when  $K_{ii} > 0$ ,  $K_{HL} = K_{LH} = 0$ ,  $i \in \{H, L\}$  makes the HI-  
 308 and LO-criticality systems completely decoupled. It is trivial to show that the compensation  
 309 property holds, since  $\varepsilon_H$  has no effect on the LO-criticality server, and  $\varepsilon_L$  has no effect on the  
 310 HI-criticality server.

311 Therefore, we focus on the case  $K_{ij} > 0$ ,  $i, j \in \{H, L\}$ . Since we are dealing with a  
 312 linear system, we can consider the effect of the disturbances separately, and then use the  
 313 superposition principle. Without loss of generality, let us consider a positive disturbance  
 314  $\varepsilon_H > 0$ , and an initial condition  $S_i(0) = Q_i(0) = \bar{Q}_i$ ,  $i \in \{H, L\}$ . First, consider the case  
 315 when  $K_{ij} > 0$ ,  $i, j \in \{H, L\}$ .  $\varepsilon_H$  has the effect of increasing the value of  $S_H$ , according to (4),  
 316 without affecting immediately the value of  $S_L$ , according to (5). If  $K_{ij} > 0$ ,  $i, j \in \{H, L\}$ , an  
 317 increasing value of  $S_H$  will make decrease both the tentative budgets, as per (6), and (7).  
 318 Therefore, in the next step, the tentative budget allocated to the two servers is decreased,  
 319 with the effect that  $S_H$  is above the desired budget  $\bar{Q}_H$ , while  $S_L$  is below the desired budget  
 320  $\bar{Q}_L$ .

321 Analogous considerations can be done for the respective negative case. This concludes  
 322 the proof. ◀

323 Notice that the compensation property of the control scheme of (8) relates to the transient  
 324 behavior caused by the occurrence of a disturbance — it does not guarantee that the effect  
 325 of a disturbance will eventually vanish. Hence a second essential property of the control  
 326 scheme of (8) is *stability*. If stability is not guaranteed, then it is not possible to preserve the  
 327 bandwidth, and not even to preserve the target period  $\bar{P}$ . We want the effect of transient  
 328 perturbations to be transient, and desire that the actual server budgets tend towards the  
 329 specified target budget values. Theorem 2 guarantees some properties on the initial transient,  
 330 but it does not guarantee the convergence of the system behavior towards the desired budget;  
 331 guaranteeing such convergence is equivalent, in control theory terminology, to requiring  
 332 stability of the controlled system.

333 Stability of discrete-time systems, such as the one specified by Expression (8), is guaranteed  
 334 if and only if the roots of the characteristic polynomial  $p(z)$  of (9) are within the unit circle  
 335 over the complex plane  $\mathbb{C}$ . That is

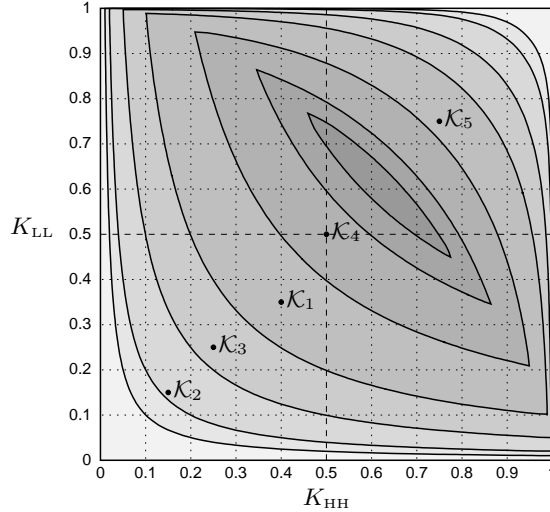
$$336 \quad p(z) = 0 \quad \Rightarrow \quad |z| < 1.$$

337 Such a condition on the polynomial  $p(z)$  can be translated into a condition over the coefficients  
 338 of the polynomial and, in turn, into a condition over the control gains  $K_{HH}$ ,  $K_{HL}$ ,  $K_{LH}$ , and  
 339  $K_{LL}$ . Jury's stability criterion (see, for example, [23, Sec 3.15.2]) offers a necessary and  
 340 sufficient condition for the stability of a discrete-time system in the form of a set of inequalities  
 341 which are functions of the coefficients of the characteristic polynomial. By applying Jury's  
 342 criterion to the polynomial  $p(z)$  of (9), one can obtain four analytic conditions on the values  
 343 of the parameters  $K_{ij}$ ,  $i, j \in \{H, L\}$  that guarantee stability. We do not present these  
 344 conditions here since they are quite lengthy and complex, but point out that they can be  
 345 computed through a symbolic manipulation tool<sup>4</sup> from the expression of  $p(z)$ .

346 The intersection of the inequalities (11) with the stability conditions that are obtained  
 347 with the Jury criterion describes the region of the feasible controller gains that guarantee both  
 348 the compensation property and the stability of the controller. Figure 2 shows the contour  
 349 plot of the stability regions for the parameters  $K_{HH}$ ,  $K_{LL}$ , for different values of  $K_i = K_{HL}K_{LH}$

<sup>4</sup> We used the Matlab function available at <https://se.mathworks.com/matlabcentral/fileexchange/13904-jury> in combination with the Matlab symbolic toolbox.

350 (identified in the figure with different colors). Notice that the region is symmetric with  
 351 respect to the plane  $K_{HH} = K_{LL}$ , and that for increasing  $K_i$  the stability region shrinks.  
 Moreover, for  $K_i = 0$ , the stability region is  $0 < K_{HH} < 1$ , and  $0 < K_{LL} < 1$ .



■ **Figure 2** Region of feasible control gains. The illustrated regions correspond to the values of  $K_i \in \{0, 0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.35\}$ , respectively from the larger region to the smaller one. Black dots represent the gains of the controllers selected for the examples illustrated in Section 6.

352

## 5 Bounding the Resource Supply

353

354 Feedback control for real-time resource allocation was initially used for tracking time-varying  
 355 workloads [28, 7, 1]. Because of the unpredictable nature of variations, the type of offered  
 356 guarantees are probabilistic or soft real-time. Recently, however, it was shown that a control  
 357 scheme can provide both a good average behavior **and** hard-real-time guarantees [22]. Such  
 358 a guarantee was given by computing the “supply bound function” of a periodic resource  
 359 supply controlled by a feedback loop such as the one described by Expression (8).

360 Bounds to supply functions are a commonly used abstraction for modeling the minimum  
 361 amount of a computing resource that is available over time [21, 18, 27, 2]. They have  
 362 demonstrated their applicability to realistic use cases (e.g., avionics [12]) and there exist  
 363 measurement-based tools to determine them from actual system execution traces [20]. Let  
 364 us briefly recall the main concepts. Let  $s(t)$  be the indicator function of the availability of a  
 365 resource:

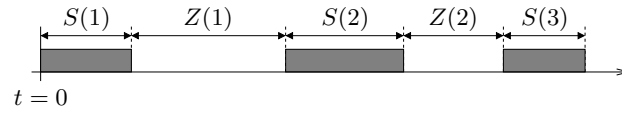
$$366 \quad s(t) = \begin{cases} 1 & \text{the resource is available at time } t \\ 0 & \text{the resource is not available at time } t, \end{cases} \quad (12)$$

367 Then the *supply bound function*  $\text{sbf}(t)$  is such that it is

$$368 \quad \forall t_0, t, \quad \text{sbf}(t) \leq \int_{t_0}^{t_0+t} s(\tau) d\tau. \quad (13)$$

369 Clearly, from (13), the bound  $\text{sbf}(t)$  may not be unique. The aim of much of the research in  
 370 this area is to find valid bounds  $\text{sbf}(t)$  fulfilling (13), which are as high as possible.

371 In [22], the resource availability schedule is modeled as a sequence of *active* intervals  
 372 of duration  $S(k)$  in which the resource is provided, alternating with intervals of *idle* time  
 373 of duration  $Z(k)$ . An example of such a schedule and the corresponding representation by  
 374 means of the sequences  $S(k)$  and  $Z(k)$  is illustrated in Figure 3. Such a model offers some  
 375 advantages over the traditional model by the indicator function of a schedule (as in Eq. 12).  
 In fact, it was proved (Lemma 1 in [22]) that the supply function lower bound  $\text{sbf}(t)$  can



■ **Figure 3** Active intervals interleaved with idle intervals.

376 be written as a function of the sequences of active and idle intervals. Specifically, it was  
 377 shown that if the resource offered by a schedule is modeled by a sequence of supply intervals  
 378 of length  $\{S(k)\}_{k=1,2,\dots}$  interleaved by a sequence of idle intervals of length  $\{Z(k)\}_{k=1,2,\dots}$ ,  
 379 then the following constitutes a valid supply bound function for this resource availability:  
 380

$$381 \quad \text{sbf}(t) = \min \{t - \sigma_Z(n), \sigma_S(n)\}, \quad t \in \mathbb{I}_n, n \in \mathbb{N} \quad (14)$$

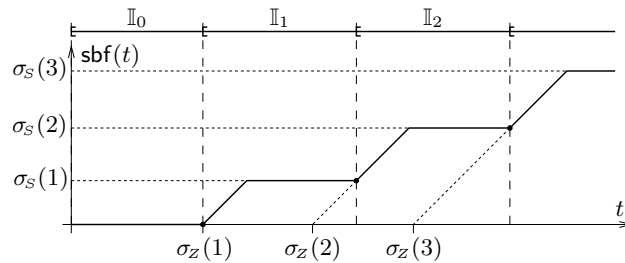
382 with the sequence of intervals  $\{\mathbb{I}_n\}_{n \in \mathbb{N}}$  defined as

$$383 \quad \mathbb{I}_n = \begin{cases} [0, \sigma_S(1)) & n = 0 \\ [\sigma_Z(n) + \sigma_S(n-1), \sigma_Z(n+1) + \sigma_S(n)) & n \geq 1 \end{cases} \quad (15)$$

384 and with

$$385 \quad \sigma_S(n) = \inf_{n_0} \sum_{k=n_0}^{n_0+n-1} S(k), \quad \sigma_Z(n) = \sup_{n_0} \sum_{k=n_0}^{n_0+n-1} Z(k), \quad (16)$$

386 properly extended at  $n = 0$  with  $\sigma_S(0) = \sigma_Z(0) = 0$ . The worst-case nature of the bound is  
 387 condensed in  $\sigma_S(n)$  that is the smallest sum of the lengths of  $n$  consecutive active intervals  
 388 (respectively,  $\sigma_Z(n)$  is the largest sum of the length of  $n$  consecutive idle intervals). Figure 4  
 389 illustrates an example of supply function  $\text{sbf}(t)$ . In the figure, we also draw on top the extent  
 of the intervals  $\mathbb{I}_n$ .



■ **Figure 4** An example of supply bound function  $\text{sbf}(t)$  for a resource supply described by sequences  $S(k)$  and  $Z(k)$  of active and idle intervals.

391 **5.1 Characterizing the Server Supply Functions**

392 One criticism of many mixed-criticality scheduling algorithms that have been proposed is  
 393 that the LO-criticality workload is severely penalized (e.g., dropped entirely) in the event of  
 394 the mixed-criticality system behavior exceeding its LO-criticality specifications. As stated  
 395 earlier, this violates the principle of resilience or robustness, which requires that slight  
 396 deviations from LO-criticality specifications should result in slight degradation of performance  
 397 (in mixed-criticality settings, to only the LO-criticality workload). In this section, we discuss  
 398 how an appropriate assignment of values to the gains of the controller  $K_{HH}$ ,  $K_{HL}$ ,  $K_{LH}$ , and  
 399  $K_{LL}$  enables such resilience by guaranteeing some resource supply to the LO-criticality server.

400 Our overall approach is inspired by, and based upon, the analysis proposed by Papadopoulos  
 401 et al. [22]. However, there are several differences in the server requirements/assumptions  
 402 between our model and the model in [22], that renders the main result (Theorem 1 of [22,  
 403 page 231]) inapplicable for our purposes.

404 ■ First, while disturbances were assumed in [22] to have symmetric bounds, in this paper the  
 405 LO-criticality server may only experience a *negative* disturbance, as in (1); equivalently,  
 406 the LO-criticality server is never allowed to execute beyond the tentative budget that is  
 407 computed for it by the control strategy.

408 ■ Second, in our mixed-criticality run-time algorithm, the servers assigning the computing  
 409 resource are *coupled* by cross gains  $K_{HL}$  and  $K_{LH}$ : letting  $i, j \in \{H, L\}$ , it is possible to  
 410 correct the server budget  $S_i(k+1)$  based on any disturbance  $\varepsilon_j(k)$ . This enables a more  
 411 prompt compensation.

412 The following theorem characterizes the relationship between the run-time behavior of the  
 413 two servers, and enables us to determine the supply function of both the HI-criticality and  
 414 LO-criticality servers. In the theorem we use the notation  $h_{ij}(k)$ ,  $g_{ij}(k)$ , and  $r_{ij}(k)$  to denote  
 415 the *impulse*, *step*, and *ramp* responses, respectively, of the system with input  $\varepsilon_j(k)$  and  
 416 output  $S_i(k)$ , with  $i, j \in \{H, L\}$  (see Appendix A for the definitions of the considered input  
 417 signals).

418 ► **Theorem 3.** *Consider a pair of HI-criticality and LO-criticality servers, whose budgets  $S_H(k)$   
 419 and  $S_L(k)$  are subject to disturbances  $\varepsilon_H(k)$  and  $\varepsilon_L(k)$  respectively, with closed-loop system  
 420 dynamics as specified by Equation (8). If the disturbances are bounded as specified by (1),  
 421 then the supply function  $\text{sbf}_H(t)$  of the HI-criticality server is as specified in Equation (14)  
 422 with*

$$423 \begin{aligned} \sigma_S(n) &= n\bar{Q}_H - \bar{\varepsilon}_H \mathcal{N}_{HH}(n) - \frac{\bar{\varepsilon}_L}{2} (\mathcal{I}_{HL}(n) + \mathcal{N}_{HL}(n)), \\ \sigma_Z(n) &= n\bar{Q}_L + \bar{\varepsilon}_H \mathcal{N}_{LH}(n) + \frac{\bar{\varepsilon}_L}{2} (\mathcal{J}_{LL}(n) + \mathcal{N}_{LL}(n)), \end{aligned} \quad (17)$$

424 and the supply function  $\text{sbf}_L(t)$  of the LO-criticality server is as specified in Equation (14)  
 425 with

$$426 \begin{aligned} \sigma_S(n) &= n\bar{Q}_L - \bar{\varepsilon}_H \mathcal{N}_{LH}(n) - \frac{\bar{\varepsilon}_L}{2} (\mathcal{I}_{LL}(n) + \mathcal{N}_{LL}(n)), \\ \sigma_Z(n) &= n\bar{Q}_H + \bar{\varepsilon}_H \mathcal{N}_{HH}(n) + \frac{\bar{\varepsilon}_L}{2} (\mathcal{J}_{HL}(n) + \mathcal{N}_{HL}(n)). \end{aligned} \quad (18)$$

427 The coefficients  $\mathcal{N}_{ij}(n)$ ,  $\mathcal{I}_{iL}(n)$ , and  $\mathcal{J}_{iL}(n)$  used in the equations above are set as

$$\begin{aligned}
 \mathcal{N}_{ij}(n) &= \sum_{k=0}^{\infty} |g_{ij}(k) - g_{ij}(k-n)| \\
 \mathcal{I}_{iL}(n) &= \sup_k \{r_{iL}(k) - r_{iL}(k-n)\} \\
 \mathcal{J}_{iL}(n) &= \sup_k \{r_{iL}(k-n) - r_{iL}(k)\}
 \end{aligned}
 \tag{19}$$

429 with  $i, j \in \{H, L\}$  corresponding to the LO-criticality and HI-criticality servers, respectively.

430 **Proof.** In the appendix (Appendix A). ◀

431 Theorem 3 enables us to determine the supply function of both the HI-criticality and LO-  
 432 criticality servers. In the next section, several design choices for the control gain parameters  
 433 are illustrated and discussed; it is shown how different desired behaviors can be achieved by  
 434 an appropriate choice of gain parameters.

## 435 **6 Evaluation via Simulation**

436 By characterizing the run-time dynamics of both the HI-criticality and the LO-criticality  
 437 server, Equation (8) and Theorem 3 allow us to estimate the system response to different  
 438 kinds of transient deviations from the expected “common-case” behavior, as characterized  
 439 by the LO-criticality WCET estimates. We now explore, via some simulation experiments,  
 440 (i) the manner in which the choice of gain parameter values influences the precise nature of  
 441 resilience exhibited by the run-time scheduler, and (ii) how our proposed scheme compares  
 442 with a simpler alternative strategy that is not based on the application of control-theoretic  
 443 principles.

### 444 **6.1 The Influence of Parameter Values**

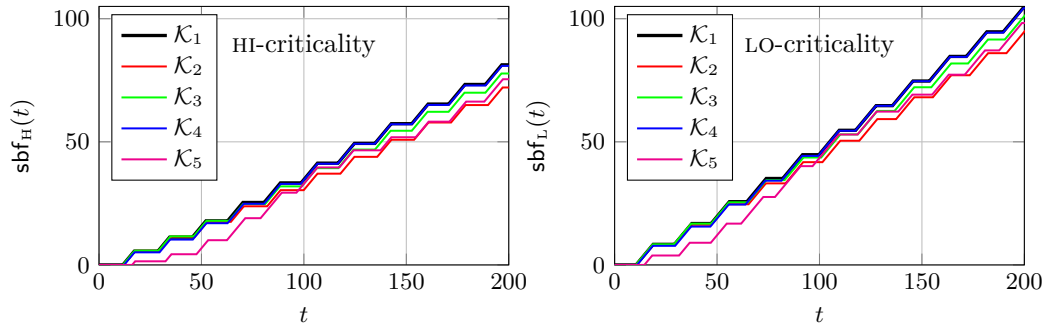
445 A closed-form solution of the dynamics of the system (8) may be obtained with the Lagrange  
 446 formula for the solution of a set of linear difference equations (see, e.g., [23, Section 12.3.5,  
 447 Eq. (12.3-34a)] for a text-book discussion). We consider the following set of parameters that  
 448 are expressed as  $\mathcal{K}_i = \{K_{HH}, K_{HL}, K_{LH}, K_{LL}\}$ :

$$\begin{aligned}
 \mathcal{K}_1 &= \{0.4, 0.1, 0.1, 0.35\}, & \mathcal{K}_2 &= \{0.15, 0.1, 0.1, 0.15\}, & \mathcal{K}_3 &= \{0.25, 0.1, 0.1, 0.25\}, \\
 \mathcal{K}_4 &= \{0.5, 0.1, 0.1, 0.5\}, & \mathcal{K}_5 &= \{0.75, 0.1, 0.1, 0.75\}
 \end{aligned}$$

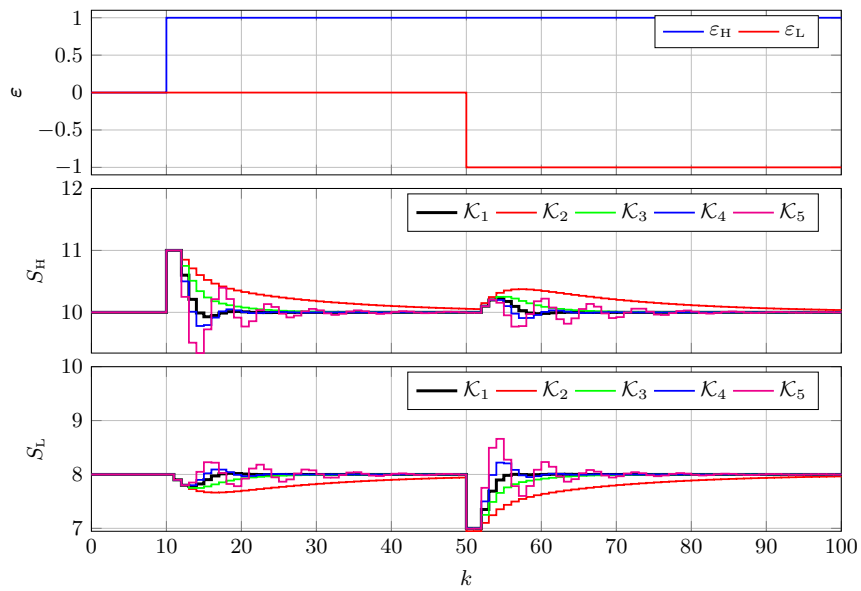
450 Notice that all the selected sets of parameters satisfy the stability conditions, and the  
 451 compensation property conditions, and therefore lie in the region as depicted in Figure 2.

452 We considered the case of the following target budgets:  $\bar{Q}_H = 10$ ,  $\bar{Q}_L = 8$ , i.e.,  $\gamma = 0.8$ ,  
 453 and  $\varepsilon_H = 1$ ,  $\varepsilon_L = 1$ . The resulting supply functions are presented in Figure 5. One can see  
 454 that the supply function associated with  $\mathcal{K}_1$  is higher than the others.

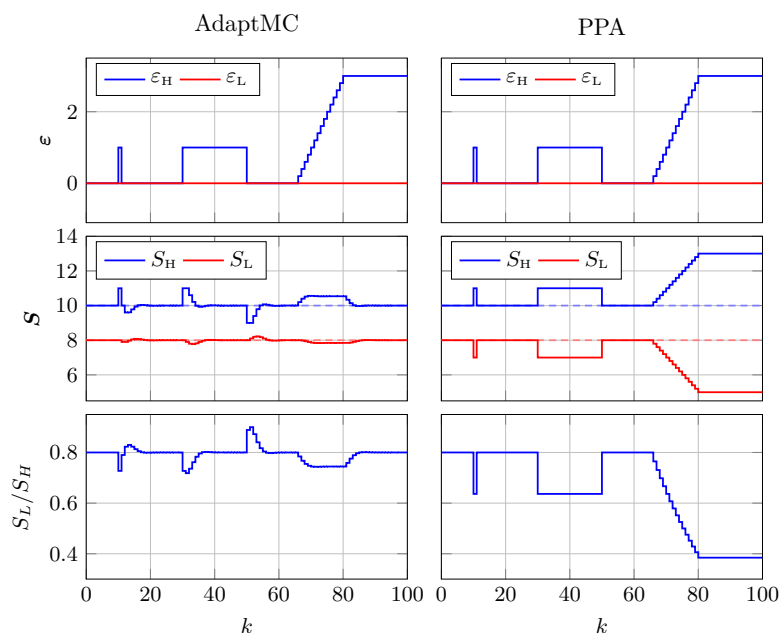
455 If keeping with common practice in control theory, we also analyzed the controller  
 456 response to a *constant* disturbance. Figure 6 shows the effect of the disturbance while  
 457 varying the values of  $K_{ij}$ ,  $i, j \in \{H, L\}$ . From Figure 6 we conclude that the best value for  
 458 the parameters is  $\mathcal{K}_1$ , since it provides a faster convergence to the target budget, and with  
 459 negligible oscillations.



■ **Figure 5** Supply functions for the considered set of control parameters.



■ **Figure 6** Effect of constant disturbances with various selection of  $K_{ij}$ ,  $i, j \in \{H, L\}$ .



■ **Figure 7** Comparison between AdaptMC and PPA.

## 6.2 Comparison with an Alternative Scheme

We now compare the presented approach with a *Period-Preserving Approach (PPA)*, described next. Based upon the findings described in Section 6.1 above, in these experiments we have selected the parameter values  $\mathcal{K}_1$  for AdaptMC.

In the PPA the HI-criticality and LO-criticality servers execute in sequence and periodically, with a fixed period  $P$  (equal to the target period for AdaptMC). Within each period, the HI-criticality server executes as much as it needs, allowing for any overrun, and the remaining budget of the period is allocated to the LO-criticality server. Formally, with the introduced notation:

$$S_H(k+1) = Q_H(k) + \varepsilon_H(k)$$

$$S_L(k) = P - S_H(k+1)$$

where  $P$  now is a fixed value. PPA represents the simplest and most intuitive way to compensate for non-ideal executions of the HI-criticality server.

In order to present the main differences between AdaptMC and PPA, we consider a scenario in which three types of disturbances occur in the system: impulse, constant, and linearly increasing. (In a well-designed mixed-criticality system, the most common form of deviation from expected behavior should be of the kind best modeled as an *impulse* disturbance – an overload that lasts for just one round and occurs rarely enough that the effect of one such overload will have completely dissipated by the time the next one occurs.)

The system is initialized as  $S_H(0) = Q_H(0) = \bar{Q}_H = 10$ , and  $S_L(0) = Q_L(0) = \bar{Q}_L = 8$ ,  $P = 18$  and no disturbance  $\varepsilon$  is present. An impulse overrun occurs at round 10, a constant overrun occurs between rounds 30 and 50, and a linearly increasing disturbance begins at round 65, and increases until it becomes of magnitude  $\bar{\varepsilon}_H$ . Figure 7 summarizes the obtained numerical results. The graphs in the first row show the time evolution of the HI-criticality server overruns: this is the disturbance, and is the same for the AdaptMC and PPA. The



486 graphs in the second row compares the actual time executed by the two servers with the  
 487 two methods. AdaptMC reacts to the disturbances by trying to preserve the target budgets,  
 488 and making minor adjustments to the tentative budgets. PPA, on the other hand, favors  
 489 the overruns of the HI-criticality server, while the execution of the LO-criticality server is  
 490 severely affected. Finally, the last row of Figure 7 shows the ratio between the bandwidth  
 491 allocated for the LO-criticality server, i.e.,  $S_L/P$ , and the actual bandwidth allocated for the  
 492 HI-criticality server, i.e.,  $S_H/P$ . We call this, the *bandwidth ratio*, and it is defined as:  $S_L/S_H$ .  
 493 The target bandwidth is  $\bar{Q}_L/P = 8/18$ , and  $\bar{Q}_H/P = 10/18$ , i.e., the target bandwidth ratio  
 494 is  $\bar{Q}_L/\bar{Q}_H = 8/10$ . The average bandwidth ratio allocated with AdaptMC is much closer  
 495 to the target bandwidth ratio than with PPA, and even the maximum deviation from the  
 496 target bandwidth is minimized by AdaptMC thanks to the feedback scheme.

## 497 **7** Related Work

498 The key property of the control-theoretic approach to budget control described in this paper  
 499 is the dynamic manner in which it modifies budgets to deal with different sizes and types of  
 500 task overruns; this stands in sharp contrast to the approach adopted in most other scheduling  
 501 schemes for mixed-criticality systems. In these schemes during run-time the system is defined  
 502 to be in one of two modes of behaviors. In the LO-criticality or “normal” mode all tasks  
 503 are executing within their LO-criticality WCET estimates and all deadlines (of both HI- and  
 504 LO-criticality tasks) are being met. As soon as any HI-criticality task executes for more  
 505 than its LO-criticality WCET estimate then there is a system-wide mode change to the  
 506 HI-criticality mode. In this new mode the behavior of the system is quite different. The  
 507 change to the HI-criticality mode occurs even if a single HI-criticality task executes for a  
 508 miniscule amount more than its LO-criticality WCET estimate or, at the other extreme, if  
 509 all HI-criticality tasks execute at their HI-criticality WCET estimate. The system responds  
 510 in the same way: there is no attempt to define behaviors that are commensurate with the  
 511 magnitude of the overrun (the disturbance or perturbation as defined in this paper).

512 Following a criticality mode change there are a number of approaches that have been  
 513 developed to define the degraded behavior of the system in the HI-criticality mode. The most  
 514 extreme is to just implement the assumptions made during the verification of the system.  
 515 Here, in the HI-criticality mode, only the HI-criticality tasks are guaranteed; hence all the  
 516 LO-criticality tasks can be abandoned (aborted). This is clearly an unacceptable approach as  
 517 no attempt is made to survive the overrun; there is no resilience in the run-time behavior of  
 518 the system. Forms of resilience that have been developed include:

- 519 1. Reduce priorities of the LO-criticality tasks [3], or similar with EDF scheduling [13].
- 520 2. Increase the periods and deadlines of LO-criticality jobs [32, 31, 15, 30, 29, 25], called  
 521 *task stretching*, the *elastic task model* or *multi-rate*.
- 522 3. Impose only a weakly-hard constraint on the LO-criticality jobs [9].
- 523 4. Decrease the computation times of some or all of the LO-criticality tasks [4], perhaps by  
 524 utilizing an imprecise mixed-criticality (IMC) model [19, 24] or budget control [10].
- 525 5. Abandon LO-criticality work in a disciplined sequence [8, 14, 11, 26, 16].

526 A flexible scheme utilizing hierarchical scheduling is proposed by Gu et al. [10]. They  
 527 differentiate between minor violations of LO-criticality execution time which can be dealt  
 528 with within a component (an internal mode change) and more extensive violations that  
 529 requires a system-wide external mode change.

530 By removing entirely the notion of a mode change (and hence a single perhaps quite  
 531 severe change in system behavior), the approach proposed in this paper results in more

532 gradual and measured responses to rare temporal glitches, such responses being automatically  
 533 delivered by the developed feedback scheme.

## 534 **8** Conclusions and Future Work

535 In this paper we have shown how a control-theoretic approach based upon servers can be  
 536 used to manage the budgets allocated to dual-criticality workloads. The control strategy  
 537 developed automatically responds to minor perturbations in the needs of the HI-criticality  
 538 server with minimum and bounded degradation in the service provided to the LO-criticality  
 539 server. The controller is defined by four “*gain*” parameters whose values must be constrained  
 540 in order to ensure stable and appropriate (compensated) control; nevertheless there remains  
 541 considerable freedom for the designer to tune the behavior of the controller. This has been  
 542 demonstrated by some simple examples.

543 This initial study has been limited to just two criticality levels and two servers (one  
 544 per level). Future work will first look to increase the number of levels supported, and to  
 545 investigate if there is any benefit to be gained from having more than one HI-criticality server  
 546 (and more than one LO-criticality server).

## 547 ——— References ———

- 548 **1** Luca Abeni, Luigi Palopoli, Giuseppe Lipari, and Jonathan Walpole. Analysis of a  
 549 reservation-based feedback scheduler. In *Proceedings of the 23rd IEEE Real-Time Systems  
 550 Symposium*, pages 71–80, Austix (TX), USA, December 2002.
- 551 **2** Luis Almeida and Paulo Pedreiras. Scheduling within temporal partitions: response-time  
 552 analysis and server design. In *Proceedings of the 4<sup>th</sup> ACM International Conference on  
 553 Embedded Software*, pages 95–103, Pisa, Italy, September 2004.
- 554 **3** Sanjoy K. Baruah and Alan Burns. Implementing mixed criticality systems in Ada. In  
 555 A. Romanovsky, editor, *Proc. of Reliable Software Technologies - Ada-Europe 2011*, pages  
 556 174–188. Springer, 2011.
- 557 **4** Alan Burns and Sanjoy K. Baruah. Towards a more practical model for mixed criticality  
 558 systems. In *Proc. 1st Workshop on Mixed Criticality Systems (WMC), RTSS*, pages 1–6,  
 559 2013.
- 560 **5** Alan Burns and Robert I. Davis. A survey of research into mixed criticality systems. *ACM  
 561 Computer Surveys*, 50(6):1–37, 2017.
- 562 **6** Alan Burns and Robert I. Davis. Mixed-criticality systems: A review (10th edition). [http:  
 563 //www-users.cs.york.ac.uk/~burns/review.pdf](http://www-users.cs.york.ac.uk/~burns/review.pdf) (Accessed on Apr 8th, 2018), 2018.
- 564 **7** Anton Cervin and Johan Eker. Feedback scheduling of control tasks. In *Proceedings of  
 565 the 39th IEEE Conference on Decision and Control*, pages 4871–4876, 2000. doi:10.1109/  
 566 CDC.2001.914702.
- 567 **8** Tom Fleming and Alan Burns. Incorporating the notion of importance into mixed criticality  
 568 systems. In Liliana Cucu-Grosjean and Robert I. Davis, editors, *Proc. 2nd Workshop on  
 569 Mixed Criticality Systems (WMC), RTSS*, pages 33–38, 2014.
- 570 **9** Oliver Gettings, Sophie Quinton, and Robert I. Davis. Mixed criticality systems with  
 571 weakly-hard constraints. In *Proc. 23rd International Conference on Real-Time Networks  
 572 and Systems (RTNS 2015)*, pages 237–246, 2015.
- 573 **10** Xiaozhe Gu and Arvind Easwaran. Dynamic budget management with service guarantees  
 574 for mixed-criticality systems. In *Proc. Real-Time Systems Symposium (RTSS)*, pages 47–56.  
 575 IEEE, 2016.

- 576 **11** Xiaozhe Gu, Arvind Easwaran, Kieu-My Phan, and Insik Shin. Resource efficient isolation  
577 mechanisms in mixed-criticality scheduling. In *Proc. 27th ECRTS*, pages 13–24. IEEE,  
578 2015.
- 579 **12** Ana Guasque, Patricia Balbastre, and Alfons Crespo. Real-time hierarchical systems with  
580 arbitrary scheduling at global level. *Journal of Systems and Software*, 119:70–86, 2016.
- 581 **13** Pengcheng Huang, Georgia Giannopoulou, Nikolay Stoimenov, and Lothar Thiele. Ser-  
582 vice adaptations for mixed-criticality systems. In *Proc. 19th Asia and South Pacific Design  
583 Automation Conference (ASP-DAC)*, Singapore, 2014.
- 584 **14** Pengcheng Huang, Pratyush Kumar, Nikolay Stoimenov, and Lothar Thiele. Interference  
585 constraint graph – a new specification for mixed-criticality systems. In *Proc. 18th Emerging  
586 Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2013.
- 587 **15** Mathieu Jan, Lilia Zaourar, and Maurice Pitel. Maximizing the execution rate of low  
588 criticality tasks in mixed criticality system. In *Proc. 1st WMC, RTSS*, pages 43–48, 2013.
- 589 **16** Jaewoo Lee, Hoon Sung Chwa, Linh T. X. Phan, Insik Shin, and Insup Lee. MC-ADAPT:  
590 Adaptive task dropping in mixed-criticality scheduling. *ACM Trans. Embed. Comput. Syst.*,  
591 16:163:1–163:21, 2017.
- 592 **17** Alberto Leva and Martina Maggio. Feedback process scheduling with simple discrete-time  
593 control structures. *IET control theory & applications*, 4(11):2331–2342, 2010.
- 594 **18** Giuseppe Lipari and Enrico Bini. Resource partitioning among real-time applications.  
595 In *Proceedings of the 15-th Euromicro Conference on Real-Time Systems*, pages 151–158,  
596 Porto, Portugal, July 2003.
- 597 **19** D. Liu, J. Spasic, N. Guan, G. Chen, S. Liu, T. Stefanov, and W. Yi. EDF-VD scheduling  
598 of mixed-criticality systems with degraded quality guarantees. In *Proc. IEEE RTSS*, pages  
599 35–46, 2016.
- 600 **20** Martina Maggio, Juri Lelli, and Enrico Bini. A tool for measuring supply functions of  
601 execution platforms. In *Embedded and Real-Time Computing Systems and Applications  
602 (RTCSA), 2016 IEEE 22nd International Conference on*, pages 39–48. IEEE, 2016.
- 603 **21** Aloysius K. Mok, Xiang Feng, and Deji Chen. Resource partition for real-time systems.  
604 In *Proceedings of the 7<sup>th</sup> IEEE Real-Time Technology and Applications Symposium*, pages  
605 75–84, Taipei, Taiwan, May 2001.
- 606 **22** Alessandro Vittorio Papadopoulos, Martina Maggio, Alberto Leva, and Enrico Bini.  
607 Hard real-time guarantees in feedback-based resource reservations. *Real-Time Systems*,  
608 51(3):221–246, June 2015.
- 609 **23** Paraskevas N. Paraskevopoulos. *Modern Control Engineering*. Automation and Control  
610 Engineering. Taylor & Francis, 2001.
- 611 **24** Risat Mahmud Pathan. Improving the quality-of-service for scheduling mixed-criticality  
612 systems on multiprocessors. In Marko Bertogna, editor, *Proc. Euromicro Conference on  
613 Real-Time Systems (ECRTS)*, volume 76 of *Leibniz International Proceedings in Informatics  
614 (LIPIcs)*, pages 19:1–19:22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017.
- 615 **25** Saravanan Ramanathan, Arvind Easwaran, and Hyeonjoong Cho. Multi-rate fluid schedul-  
616 ing of mixed-criticality systems on multiprocessors. *Real-Time Systems*, Online First, 2017.
- 617 **26** Jiankang Ren and Linh Thi Xuan Phan. Mixed-criticality scheduling on multiprocessors  
618 using task grouping. In *Proc. 27th ECRTS*, pages 25–36. IEEE, 2015.
- 619 **27** Insik Shin and Insup Lee. Periodic resource model for compositional real-time guarantees.  
620 In *Proceedings of the 24<sup>th</sup> Real-Time Systems Symposium*, pages 2–13, Cancun, Mexico,  
621 December 2003.
- 622 **28** John A. Stankovic, Chenyang Lu, Sang H. Son, and Gang Tao. The case for feedback  
623 control in real-time scheduling. In *Proceedings of the Euromicro Conference on Real-Time*,  
624 York, U.K., June 1999.

- 625 **29** Hang Su, Peng Deng, Dakai Zhu, and Qi Zhu. Fixed-priority dual-rate mixed-criticality  
626 systems: Schedulability analysis and performance optimization. In *Proc. Embedded and*  
627 *Real-Time Computing Systems and Applications (RTCSA)*, pages 59–68. IEEE, 2016.
- 628 **30** Hang Su, Nan Guan, and Dakai Zhu. Service guarantee exploration for mixed-criticality  
629 systems. In *Proc. Embedded and Real-Time Computing Systems and Applications (RTCSA)*,  
630 pages 1–10. IEEE, 2014.
- 631 **31** Hang Su and Dakai Zhu. An elastic mixed-criticality task model and its scheduling algo-  
632 rithm. In *Proceedings of the Conference on Design, Automation and Test in Europe, DATE*,  
633 pages 147–152, 2013.
- 634 **32** Hang Su, Dakai Zhu, and Daniel Mosse. Scheduling algorithms for elastic mixed-criticality  
635 tasks in multicore systems. In *Proc. RTCSA*, 2013.
- 636 **33** Steve Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of  
637 execution time assurance. In *Proceedings of the Real-Time Systems Symposium*, pages  
638 239–243, Tucson, AZ, December 2007. IEEE Computer Society Press.

### 639 **A** Proof of Theorem 3

640 Before entering the details of the proofs, we remind that a linear time-invariant (LTI) system  
641 can be uniquely characterized by its *impulse response*  $h(k)$  that is the output  $y(k)$  when the  
642 system is stimulated with an impulsive input  $u(k)$

$$643 \quad u(k) = \begin{cases} 1 & k = 0 \\ 0 & \text{otherwise.} \end{cases}$$

644 In next lemmas, we are also using the *step response*

$$645 \quad g(k) = \sum_{i=0}^k h(k), \quad (20)$$

646 and the *ramp response*

$$647 \quad r(k) = \sum_{i=0}^k g(i) \quad (21)$$

648 of a LTI system.

649 Thanks to the linear and time-invariance of the system, the output  $y(k)$  to any input  
650  $u(k)$  is given by the *convolution* of the impulse response  $h(k)$  and the input  $u(k)$ , that is

$$651 \quad y(k) = h(k) \otimes u(k) = \sum_{i=0}^k u(i)h(k-i).$$

652 With these basic notions recalled, next we state a technical lemma that bounds the output  
653  $y(k)$  of a LTI system when the input  $u(k)$  belongs to a bounded interval  $[\tilde{u} - \bar{\varepsilon}, \tilde{u} + \bar{\varepsilon}]$ .

654 **► Lemma 1.** *Given an asymptotically stable discrete-time LTI system with impulse response*  
655  *$h(k)$ , step response  $g(k)$ , input  $u(k)$ , and output*

$$656 \quad y(k) = h(k) \otimes u(k).$$

657 *If the input  $u(k)$  is bounded as follows*

$$658 \quad u(k) = \tilde{u} + \varepsilon(k), \quad \tilde{u} \in \mathbb{R}, \quad -\bar{\varepsilon} \leq \varepsilon(k) \leq \bar{\varepsilon},$$

659 then, the output  $y(k)$  is bounded by

$$660 \quad |\tilde{u}| \inf_k \{\text{sign}(\tilde{u})g(k)\} - \bar{\varepsilon} \|h\|_1 \leq y(k) \leq |\tilde{u}| \sup_k \{\text{sign}(\tilde{u})g(k)\} + \bar{\varepsilon} \|h\|_1, \quad (22)$$

661  
662 with the  $\ell_1$ -norm of a signal defined as

$$663 \quad \|h\|_1 = \sum_{k=0}^{\infty} |h(k)|.$$

664 **Proof.** By definition of  $y(k)$  as convolution of the impulse response  $h(k)$  with the input  
665 signal  $u(k)$ , it follows

$$\begin{aligned} 666 \quad y(k) &= \sum_{i=0}^k u(i)h(k-i) = \sum_{i=0}^k (\tilde{u} + \varepsilon(i))h(k-i) \\ 667 \quad &= \tilde{u} \sum_{i=0}^k h(k-i) + \sum_{i=0}^k \varepsilon(i)h(k-i) \\ 668 \quad &= \tilde{u} g(k) + \sum_{i=0}^k \varepsilon(i)h(k-i) \\ 669 \quad &\leq |\tilde{u}| \sup_k \{\text{sign}(\tilde{u})g(k)\} + \bar{\varepsilon} \|h(k)\|_1 \end{aligned}$$

670  
671 with

$$672 \quad \|h(k)\|_1 = \sum_{k=0}^{\infty} |h(k)|.$$

673 Analogously

$$674 \quad y(k) \geq |\tilde{u}| \inf_k \{\text{sign}(\tilde{u})g(k)\} - \bar{\varepsilon} \|h(k)\|_1,$$

675  
676 which concludes the proof. ◀

677 The next Corollary determines the upper and lower bounds to the sum of  $n$  consecutive  
678 outputs, by exploiting Lemma 1.

679 **► Corollary 1.** *Given an asymptotically stable discrete-time LTI system, if the input  $u(k)$   
680 bounded as follows*

$$681 \quad u(k) = \tilde{u} + \varepsilon(k), \quad \tilde{u} \in \mathbb{R}, \quad -\bar{\varepsilon} \leq \varepsilon(k) \leq \bar{\varepsilon}.$$

682 Then, the sum of  $n$  consecutive outputs is bounded by

$$683 \quad -(|\tilde{u}| \mathcal{I}(n) + \bar{\varepsilon} \mathcal{N}(n)) \leq \sum_{k=n_0}^{n_0+n-1} y(k) \leq |\tilde{u}| \mathcal{J}(n) + \bar{\varepsilon} \mathcal{N}(n), \quad (23)$$

684  
685 with

$$686 \quad \mathcal{N}(n) = \sum_{k=0}^{\infty} |g(k) - g(k-n)|, \quad (24)$$

$$687 \quad \mathcal{I}(n) = \sup_k \{-\text{sign}(\tilde{u})(r(k) - r(k-n))\} \quad (25)$$

$$688 \quad \mathcal{J}(n) = \sup_k \{\text{sign}(\tilde{u})(r(k) - r(k-n))\} \quad (26)$$

689  
690 and  $g(k)$  and  $r(k)$  being the step and ramp response, respectively.

691 **Proof.** The output  $y(k)$  of a LTI system is the convolution of the impulse response  $h(g)$  and  
692 the input  $u(k)$

$$693 \quad y(k) = h(k) \otimes u(k).$$

694 Because of the linearity of the convolution, the sum of  $n$  consecutive output is

$$695 \quad \sum_{i=k}^{k+n-1} y(i) = \left( \sum_{i=k}^{k+n-1} h(i) \right) \otimes u(k) = (g(k) - g(k-n)) \otimes u(k).$$

697 Finally, by applying Equation (22) of Lemma 1, Equation (23) of the Corollary follows. ◀

698 **Proof of Theorem 3.** Let us first determine the supply function  $\text{sbf}_H(t)$  of the HI-criticality  
699 server. We aim at modeling the resource supplied to the HI-criticality server as a sequence of  
700 active intervals of lengths  $S(k)$ , interleaved by a sequence of idle intervals of lengths  $Z(k)$   
701 that corresponds to the schedule of the LO-criticality server. Formally,

$$702 \quad S(k) = S_H(k), \quad Z(k) = S_L(k). \quad (27)$$

703 In fact, by doing so, Lemma 1 of [22] can give us the supply function of (14) through the  
704 proper value of  $\sigma_S(n)$  and  $\sigma_Z(n)$ , as defined in (16).

705 First of all, the system of (8) that determines the dynamics of  $S_H(k)$  is linear. Hence, by  
706 the superposition principle the output  $S_H(k)$  is equal to the sum of three components:

- 707 1. the output  $\bar{Q}_H$  when  $\varepsilon_H(k) = 0$  and  $\varepsilon_L(k) = 0$ ,
- 708 2. the output  $y_{HH}(k)$  when  $\bar{Q}_H = 0$  and  $\varepsilon_L(k) = 0$ , and
- 709 3. the output  $y_{HL}(k)$  when  $\bar{Q}_H = 0$  and  $\varepsilon_H(k) = 0$ ,

710 that is

$$711 \quad S_H(k) = \bar{Q}_H + \underbrace{h_{HH}(k) \otimes \varepsilon_H(k)}_{y_{HH}(k)} + \underbrace{h_{HL}(k) \otimes \varepsilon_L(k)}_{y_{HL}(k)} \quad (28)$$

712 and  $h_{Hi}(k)$  is the response of  $S_H(k)$  to an impulse on the input  $\varepsilon_i(k)$ , with  $i \in \{L, H\}$ .

713 Let us now compute  $\sigma_S(n)$  that is, from (16), a lower bound to the sum of the length of  
714  $n$  consecutive budgets  $S_H(k)$

$$715 \quad \sigma_S(n) = \inf_{n_0} \sum_{k=n_0}^{n_0+n-1} S(k) = \inf_{n_0} \sum_{k=n_0}^{n_0+n-1} S_H(k) = n\bar{Q}_H + \inf_{n_0} \sum_{k=n_0}^{n_0+n-1} (y_{HH}(k) + y_{HL}(k)). \quad (29)$$

717 To bound the sum of  $n$  consecutive values of  $y_{HH}(k)$  and  $y_{HL}(k)$ , we can invoke Corollary 1.

718 Let us start with

$$719 \quad y_{HH}(k) = h_{HH}(k) \otimes \varepsilon_H(k).$$

720 From the hypothesis of (1),  $\varepsilon_H(k)$  is bounded by

$$721 \quad -\bar{\varepsilon}_H \leq \varepsilon_H(k) \leq \bar{\varepsilon}_H$$

722 and then Eq. (23) of Corollary (1) states that

$$723 \quad -\bar{\varepsilon}_H \mathcal{N}_{HH}(n) \leq \sum_{k=n_0}^{n_0+n-1} y_{HH}(k),$$

## 14:22 AdaptMC: Control Theory for Mixed-Criticality Systems

724 with  $\mathcal{N}_{\text{HH}}(n)$  as in (19). Similarly, from the asymmetric bound to  $\varepsilon_{\text{L}}(k)$  of (1), from (23) it  
725 follows that

$$726 \quad -\frac{\bar{\varepsilon}_{\text{L}}}{2}(\mathcal{I}_{\text{HL}}(n) + \mathcal{N}_{\text{HL}}(n)) \leq \sum_{k=n_0}^{n_0+n-1} y_{\text{HL}}(k),$$

727 from which the expression of  $\sigma_{\text{S}}(n)$  of (17) follows.

728 The expression of  $\sigma_{\text{Z}}(n)$  of (17) can be found by following similar steps:

- 729 **1.** by setting the sequence of idle intervals  $Z(k)$  equal to the sequence of the LO-criticality  
730 budgets  $S_{\text{L}}(k)$ , as in (27);
- 731 **2.** by writing the sequence  $S_{\text{L}}(k)$  as the sum of  $\bar{Q}_{\text{L}}$  and the sequences  $y_{\text{LH}}(k)$  and  $y_{\text{LL}}(k)$  that  
732 corresponds to the responses to the disturbances  $\varepsilon_{\text{L}}(k)$  and  $\varepsilon_{\text{L}}(k)$  on  $S_{\text{L}}(k)$  (similarly as  
733 in (28); and
- 734 **3.** by exploiting Corollary 1 to bound  $y_{\text{LH}}(k)$  and  $y_{\text{LL}}(k)$ .

735 The expressions of  $\sigma_{\text{S}}(n)$  and  $\sigma_{\text{Z}}(n)$  give the expression of the  $\text{sbf}_{\text{H}}(t)$ .

736 Analogously, by setting

$$737 \quad S(k) = S_{\text{L}}(k), \quad Z(k) = S_{\text{H}}(k),$$

738 and following the same steps illustrated above, it is possible to determine the proper values  
739 of  $\sigma_{\text{S}}(n)$  and  $\sigma_{\text{Z}}(n)$  of (18) and then the supply function  $\text{sbf}_{\text{L}}(t)$  of the LO-criticality server.  
740 This concludes the proof.  $\blacktriangleleft$