Dear Editor,

We have carefully read all the reviewer comments and made changes to the paper accordingly. We would like to thank the anonymous reviewers for providing detailed comments, which we ourselves think has significantly improved the manuscript in terms of readability and quality.

All the changes made following the reviewer comments are attached to this letter.

Regards,

Authors

# Reviewer 1

|  |  |  |
| --- | --- | --- |
|  | **Comments** | **Response** |
| 1.01 | The presented idea seems attractive, and the paper is well-written. However, one issue that I think deserves further discussion is that, as you mention in Sec. 6.1, the programmer must be aware of the interaction of wait and other pause statements. Your example shows that S is not emitted each ms because of the extra pause. However, the introduction of additional (and in fact at design time unpredictable any) pause statements also changes the semantics of concurrent synchronous programs which comprise threads that communicate with each other.  This becomes also an interesting point in 7 where you discuss the differences between your approach and Bourke, where they specifically do not translate into pause statements and therefore do not change the semantics of such programs. Eventually, the same question arises in the conclusion in 8 where you state "The fundamental idea is to convert real-time wait constructs into logical waits (pause) that INTERACT WELL with the rest of the constructs in these languages [referring to Esterel]). In a similar vein, I'd appreciate a comparison with Multiclock Esterel (see e.g. Berry/Sentovich, CHARME '01). | Similar to our approach, all the approaches introduced in the related works change the semantics of the program. as the reson is that the additional constructs inserted by the programmers change the internal structure (e.g. automata representing program) of the program. Our approach, however, does not face the problems, unlike others, such as missing the external timer signals due to preemptions (Section 6.2.1). This explanation is included at the end of the first paragraph in the related work section (Section 7).  Multiclock-Esterel uses signal broadcasting mechanism for communication between asynchronous modules. SystemJ uses rendezvous style message passing mechanism via channels. Since the message delivery via channels is guaranteed by the semantics of channels, SystemJ programmers do not have to worry about whether introducing the wait statements will break the communication between clock-domains. This explanation is also included in Section 2. |
| 1.02 | - I would add "synchronous languages" to the keyword list | It is added as suggested |
| 1.03 | - In the Motivation 1.1 it is not entirely clear to me what the actual motivation is. You say "You would like to add a similar mechanism for in reactive languages" but not why. This point gets more and more clear later on, however it misses in the initial motivation. | We have extended this paragraph and explained that reactive languages lack such features and why these mechanisms are needed in the real-time applications. |
| 1.04 | - The first time "statically" is mentioned is on page 3 in the contribution sub-section. I would recommend to make it clear earlier (in the introduction or even the abstract) that the introduced concepts transforms the real-time wait statements statically. | We have added this explanation in both abstract and at the end of introduction. |
| 1.05 | - page 6, in 3.3 4. and page 6, in 3.4: "any value of d": Which one is chosen exactly? Is always the lower bound chosen? So, as d is determined statically, d is fixed. Also you're saying it is trivial in 3.4 but it is not clear which d is chosen. And if the lower bound is chosen, where lies the difference between wait\_inbetween and wait\_atleast? | Our approach does not limit whether d should be fixed or can be changed during runtime, as long as the value d satisfies the real-time constraints specified by wait\_inbetween or wait\_atleast constructs. Therefore, d does not necessarily need to be the lower bound. |
| 1.06 | - page 12, in 5: In the description of the experimental results you state, that you allowed the compiler to relax the upper real-time bounds. Is there a particular reason you chose this setting? I ask because you stated earlier that this is the non-default case and that the compiler uses the periodic execution as default. | This option was chosen to show how our algorithm can accurately generate the results compared to the original delay specified by a designer. For example, we have shown in Figure 11 how much the upper bound N should be relaxed (increased) in order to obtain a valid “d”. We have also discussed in Section 5.1 that this relaxation can be removed, resulting in a closer value to the original delay specification, by shortening a critical path. |
| 1.07 | - Throughout the paper the wait commands and their parameters are printed with and without a space in between. Should be checked for consistency. | These are fixed |
| 1.08 | - page 1, keyword list is missing (placeholders are present) | Keywords are inserted |
| 1.09 | - page 2, in 1.1.1, par 3: First mentioning of Esterel: reference? | Reference added |
| 1.10 | - page 3, in 1.1.3: Contrary to the first two definitions of the wait statements, wait\_exact is not defined with an (M) (inconsistency) | Fixed |
| 1.11 | page 4, in 2.2.: "Unfortunately, execution..." Why is this unfortunately? Is DELTA not the unfortunate part? | Fixed |
| 1.12 | - page 5, in 3.2, Figure 5: Is 'while' not a kernel construct? Why can't one just write while(x<d) { pause; x=x+1; } | while(true){…} is **not** a kernel construct. SystemJ programmer can only break this loop using trap or abort statements. |
| 1.13 | - page 7, in 3.4/3.5: repetition: You stated two times, that inserting a pause breaks critical paths. | The second repetition is removed from text. |
| 1.14 | - page 8, in 3.5.1: The first two paragraphs describe the TP. Is this part of this work? If not, maybe you can add a cite here. | Reference added |
| 1.15 | > - page 14, in 5.1: Here, you state again that inserting pause statements help with critical paths. Maybe you should simply reference your earlier remark. "As stated in 3.4 adding addition pause statements..." | The statement is added |
| 1.16 | - page 1, in 1: I would begin a new paragraph before "In this paper..." | Unfortunately, the authors could not find the place what the reviewer is referring to. In page 1, only abstract begins with “In this paper”, but nowhere else. |
| 1.17 | - Many figures do not use the available whitespace. | We tried to fix this as much as was possible |
| 1.18 | - On many pages there are lines that break the column width: pages 2, 3, 5, 6, 7, 8, 10, 11, 12, 16 | Fixed |
| 1.19 | page 4, Figure 4b: "logical time" and "physical time" not aligned | Fixed |
| 1.20 | - page 4, in 2.1, par 2: "children reaction" -> "child reactions"? | Fixed |
| 1.21 | - page 4, in 2.2, par 3, "Figure 4b" mentioned four times in four sentences | Text has been updated |
| 1.22 | - page 6, Figure 6: Don't use @ in the caption. The Figures do not use the full space. | @ is actually a part of the figure (not the caption). It is a part of x axis label for the graphs |
| 1.23 | - page 7ff, in 3.5: "see [11]..." I'd recommend against using citations as nouns. Also on pages 8, 10, 16 | These are fixed |
| 1.24 | - page 10, footnote: TA already introduced as acronym | This footnote is removed |
| 1.25 | - page 10, col 2 middle page: Why is there so much whitespace? | Fixed |
| 1.26 | - page 16, in 6.3.: Misplaced colon "Since, ..." | Fixed |
| 1.27 | - page 17, in 7: You mention Bertin, then Quartz and then continue with the differences between your approach and Bertin", maybe move the Quartz reference to the end of the paragraph. | Quartz ref is moved to the end of the paragraph as suggested |
| 1.28 | - page 18: capital letter in "Worst" and "Best" | Fixed |

# Reviewer 3

|  |  |  |
| --- | --- | --- |
|  | **Comments** | **Response** |
| 3.01 | > (2) For the proposed scheme, the authors analyze how to add the real time constrains in the GALS programming languages and synchronous languages. However, the authors do not discuss how these programs interact with the real hardware. So, it is better for the authors to consider how to interact the proposed scheme with the real hardware. In embedded system, the storage performance has become the most important issue to affect the real time requirements. It is better for the authors to discuss how to combine the proposed real-time scheme with the storage. Below lists several papers that can be referred by the authors: | We included discussions on this matter at the end of Section 3.5.3. We also included the references as suggested by the reviewer. |
| 3.02 | (3) In the experiment, the authors show and detailed analyze the experimental results. However, the authors do not discuss how the experimental results are obtained. It is better for the authors to give the detailed description about this which can make readers more convince about the results. | In Section 5 paragraphs 6-8, we have already explained that the experimental results were obtained by first statically estimating the WCRT and BCRT of the programs for the TP-JOP platform based on the approach presented in Subsections 3.5.1-3.5.3. The Algorithm 1 (and also 2 if we chose to relax the upper bounds) is then applied, which gives the number of logical ticks *d* that satisfy the real-time postponement. Please note that all the delay specifications (M and N) are already given in Table 2 in Section 5, hence we have all the inputs (WCRT, BCRT, M and N) to Algorithm 1, hence the logical ticks *d* can be obtained, which is the result of this experiment. |