Response Time Analysis for Energy-Harvesting Mixed-Criticality Systems

Kankan Wang, Yuhan Lin and Qingxu Deng
Northeastern University, China

***Abstract*—With the increasing demand for real-time computing applications on energy-harvesting embedded devices which are deployed wherever it is not possible or practical to recharge, the worst-case performance analysis becomes crucial. However, it is difficult to bound the worst-case response time of tasks under both timing and energy constraints due to the uncertainty of harvested energy. Based on this motivation, this paper studies response time analysis for Energy-Harvesting Mixed-Criticality (EHMC) systems. We present schedulability analysis algorithm to extend the Adaptive Mixed Criticality (AMC) approach to EHMC systems. Furthermore, we develop two response time bounds for it. To our best knowledge, this is the first work of response time analysis for EHMC systems. Finally, we examine both the effectiveness and the tightness of the bounds by experiments.**

***Index Terms*—Energy-harvesting, mixed-criticality, AMC, response time analysis, real-time scheduling**

I. Introduction

There has been an increasing trends in Cyber-Physical Systems (CPS) design towards integrating multiple applications with different criticalities onto a single shared computing platform. The avionics systems [1] and automotive systems [2] are the typical examples. These systems are generally referred to as Mixed Criticality Systems (MCS) which have components of two or more distinct criticality levels. The design based on MCS allows for efficient resource usage in CPS. The major challenge for MCS is to guarantee the schedulability of the system at all different levels of criticality at the same time.

However, many CPSs are deployed wherever it is not possible or practical to recharge the devices or replace their batteries, e.g., sensor networks where there are a large number of sensor nodes powered by batteries. Therefore, energy harvesting which is the conversion of ambient energy present in the environment into electrical energy is deemed a promising approach. A large number of alternative approaches have been developed to energy harvesting. However, since the harvested energy is inherently unstable and unpredictable, task execution is often delayed due to insufficient energy and this problem brings significant challenges to bound the worst-case response time of tasks in energy-harvesting real-time systems. To address this challenge, [3] uses Real-Time Calculus (RTC) as the energy model, but energy is the only resource constraint. [6] uses probabilistic techniques to analyze the schedulability of Energy-Harvesting Mixed-Criticality (EHMC) systems under both timing and success-ratio constraints, but this scheduling algorithm has exponential complexity and are not applicable to large-scale systems. To simplify the scheduling and analysis, [4] assumes that the replenishment rate is a constant. So far, there is still a lack of general and efficient analysis methods to calculate safe response time bounds for EHMC systems.

In this paper, we address the above problem and mainly focus on the *fixed priority preemptive scheduling* (FPPS) for EHMC systems considering both the energy and timing constaints of real-time tasks as well as the variability of the available harvested energy. Unfortunately, the traditional *work-conserving policy* is not applicable to our problem since it may lead to unbounded priority inversion and badly hurt system schedulability. Therefore, the concept of *energy work-conserving policy* (EWCP) [4] is introduced to avoid this problem. However, since the replenishment rate is time-varying, the length of time interval where sufficient energy is replenished for a job is uncertain. To solve this problem, we develop analysis techniques to safely bound the response time of real-time tasks in such systems. In particular, this paper provides the following contributions:

- We extend the Adaptive Mixed Criticality (AMC) approach to energy-harvesting real-time systems by incorporating EWCP into this model. This resulting algorithm is denoted as AMC-EWCP.
- We develope two response time upper bounds providing sufficient scheduling tests for AMC-EWCP. To our best knowledge, this is the first work of response time analysis for EHMC systems.

The rest of the paper is organized as follows. Sec.II presents related work. Sec.III formally defines system model and relevant notations. Sec.IV provides an overview of AMC-EWCP. Sec.V provides response time analysis for AMC-EWCP. Sec.VI presents the performance evaluation investigating the effectiveness and tightness of the schedulability tests. Sec.VII concludes this paper.

II. Related work

The real-time scheduling problem for energy-harvesting systems was first studied in [10]. Later in [3], [5], RTC is used to analyze the worst-case performance of energy-harvesting real-time systems. For fixed priority scheduling of energy-harvesting systems, [12], [13] presented sufficient schedulability tests for different task models, respectively. An optimal algorithm for FPPS was proposed in [12], which assumed that all tasks comsumes energy faster than it is replenished. To be more general, [13] removed this assumption and present sufficient schedulability tests for such systems. Furthermore, [4] present sustainability analysis for these tests and the priority assignment was also discussed. However, these studies simply assume that the replenishment rate is constant. To relax this assumption, [6] used probabilistic techniques to characterize the availability of energy and proposed scheduling algorithms for EHMC systems. On the other hand, [17], [18] considered real-time task scheduling in intermittently-powered systems, However, [17] implicitly assumes that the energy source is always available. [18] requires generating a schedule for one hyperperiod to test schedulability under its proposed approach. In addition, both of them have the limitation that energy harvesting and task execution are mutually exclusive.

III. System Model

We consider an EHMC uniprocessor system consisting of four parts: energy source, harvester, energy storage unit, MCS (as shown in Fig.1). The harvester draws energy from energy source, e.g., wind, ocean waves or solar energy, and transforms the environmental energy into electrical power. The energy storage unit stores electrical power from harvester. Finally, MCS receives energy from storage unit for task execution. We assume that the discrete time concept is used in this paper, i.e., any time parameter involved during scheduling is a non-negative integer.

Energy Source Model. Let $P_r(t)$ denote the replenishment function that is actually fed into the energy storage. In our model we assume that $P_r(t)$ incorporates all energy loss at the time instant t, i.e., all harvested energy at any time instant is fed into energy storage. The total amount of replenished energy E_r in the time interval $[t_1, t_2]$ is given as:

$$E_r(t_1, t_2) = \int_{t_1}^{t_2} P_r(t)dt$$

The property of available energy is characterized by *service curve* $\beta^l(\Delta)$, which lower-bounds the energy replenished in any time interval of length Δ, i.e., $\beta^l(t_2 - t_1) \leq E_r(t_1, t_2) \quad \forall t_2 > t_1$. We use

$$\beta^{-1}(x) = \inf\{\Delta | x \leq \beta^l(\Delta)\} \tag{1}$$

to denote the pseudo-inverse function of $\beta^l(\Delta)$ [14], i.e., the length of any time interval in which the amount of energy denoted as x is harvested, is lower-bounded by $\beta^{-1}(x)$. Our motivation behind selecting this energy model comes from the fact that the service curve can be obtained from traces and can provide harvested energy bounds with high confidence [11].

Energy Storage Model. The EHMC system is equiped with a storage unit that temporally stores the surplus energy. This energy storage is replenished even during the tasks execution. The energy level at time t is denoted by $E(t)$ and has two boundaries E_{min} and E_{max}, i.e., $E_{min} \leq E(t) \leq E_{max}$. For simplicity, we assume $E_{min} = 0$. The initial fill level of the storage unit is $E(0)$. We safely assume that only minimal energy can be derived when the system starts running, i.e., $E(0) = E_{min}$. In order to avoid impacting task scheduling, we further assume E_{max} is sufficiently large.

Task Model. The analysis techniques presented in this paper may be generalized to multiple criticality levels, but we restrict the model to two levels here for the sake of brevity: HI (high) and LO (low), with HI>LO. Each task is either a LO-criticality task (LO-task) or a HI-criticality task (HI-task). Each task τ_i in EHMC system can be characterized by $(T_i, D_i, P_i, C_i^{LO}, C_i^{HI}, E_i^{LO}, E_i^{HI}, \chi_i)$, where T_i is period or minimum inter-arrival time, D_i is relative implicit deadline $(D_i = T_i)$, P_i is the worst-case power consumption, which is the same at both criticality levels, C_i^{LO} is LO-criticality WCET (LO-WCET), C_i^{HI} is HI-criticality WCET (HI-WCET), E_i^{LO} is LO-criticality Worst-Case Execution Consumption (LO-WCEC), E_i^{HI} is HI-criticality WCEC (HI-WCEC), and $\chi_i \in \{LO, HI\}$ is a task criticality level. The LO-WCEC (HI-WCEC) equates to executing for the LO-WCET (HI-WCET), at the maximum rate of power consumption, i.e., $E_i^{LO} = P_i \cdot C_i^{LO}$ and $E_i^{HI} = P_i \cdot C_i^{HI}$. Since HI-WCETs and HI-WCECs are based on conservative assumptions, we assume that $0 < C_i^{LO} \leq C_i^{HI}$ and $0 < E_i^{LO} \leq E_i^{HI}$.

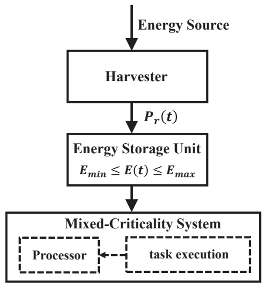


Fig. 1. EHMC system model.

We consider a sporadic task set $\tau = \{\tau_i\}$, $1 \leq i \leq n$. Each task τ_i gives rise to a potentially unbounded sequence of jobs. The tasks are in priority order, and each task is characterized by its unique priority i. We use $hep(i)$ to denote the set of all tasks with priorities higher than or equal to that of task τ_i. Let τ^{LO} denote the LO-task set defined as $\tau^{LO} \overset{\text{def}}{=} \{\tau_i \in \tau | \chi_i = LO\}$, and τ^{HI} denote the HI-task set defined as $\tau^{HI} \overset{\text{def}}{=} \{\tau_i \in \tau | \chi_i = HI\}$, and $hpH(i)$ denote the set of HI-tasks with priorities higher than or equal to i, and $hpL(i)$ denote the set of LO-tasks with priorities higher than or equal to i.

The response time of a job from task τ_i is the time distance between its release time and the completion time. The *worst-case response time* of task τ_i is the maximal response time among all its jobs. The target of this paper is to safely bound the worst-case response time of each task $\tau_i \in \tau$.

IV. Overview of AMC-EWCP

In this section, we mainly describe AMC-EWCP. First we need to introduce the concept of EWCP by which unbounded priority inversion can be avoided during task execution.

Definition 1 (from [4]). *A fixed priority scheduling policy is referred to as energy work-conserving policy if while there is an active task requiring execution, the scheduling policy only ever leaves the processor idle if there is insufficient energy available to schedule at least one execution unit of the highest priority active task.*

[7] introduced two scheduling schemes for MCS: Static Mixed-Criticality (SMC) and Adaptive Mixed Criticality

(AMC), and it proved that AMC dominates SMC in terms of schedulability. AMC-EWCP is essentially the same as AMC, but with EWCP:

- The system starts in LO-criticality mode (LO-mode). In LO-mode, jobs of all LO-tasks and HI-tasks are released.
- While in LO-mode, at each instant jobs of all active tasks are scheduled by EWCP. All LO-tasks are constrained by run-time monitoring in order to guarantee no jobs execute for more than their LO-WCETs.
- If the currently-executing job executes for its LO-WCET without signalling completion, the system enters the HI-mode (called mode-switch).
- At mode-switch, all the jobs of LO-tasks will not be executing. Hence, at each instant jobs of all active HI-tasks are scheduled by EWCP.
- The system remains in HI-mode until no HI-tasks are active, then the system returns to LO-mode. (We will not discuss this step any further in this paper)

In next section we will develop upper bounds on the response time of tasks scheduled by AMC-EWCP.

V. Response Time Analysis

[7] presented two methods for computing response time bound: AMC-rtb and AMC-max. We use AMC-rtb as the basis for our analysis and leave AMC-max as future work. In this section, we develop upper bounds on the worst-case response time of task τ_i in LO-mode, HI-mode and mode-switch phase, respectively.

A. Response Time Bound of LO-Mode

In this subsection, we calculate response time bound of task τ_i in LO-mode by two methods, by which two response time upper bounds are obtained.

1) First Response Time Bound: We first introduce three useful notions: *processor demand*, *energy demand* and *time demand.*

Definition 2 (Processor Demand)**.** *The processor demand of the i^{th} priority level is the amount of time necessary to execute jobs with priorities higher than or equal to the priority of the considered task τ_i, until τ_i finishes its active job.*

Definition 3 (Energy Demand)**.** *The energy demand of the i^{th} priority level is the amount of energy required to execute jobs with priorities higher than or equal to the priority of the considered task τ_i, until τ_i finishes its active job.*

Definition 4 (Time Demand)**.** *The time demand of the i^{th} priority level is the minimal amount of time to satisfy both of the processor and energy demand.*

Based on the above definitions, we derive the first response time upper bound by the following theorem.

Theorem 1. *In LO-mode, the worst-case response time of task τ_i is upper bounded by:*

$$R_i^{L1} = \sum_{h \in hep(i)} \left\lceil \frac{R_i^{L1}}{T_h} \right\rceil \cdot \max\left\{ \left\lceil \beta^{-1}(E_h^{LO}) \right\rceil, C_h^{LO} \right\} \tag{2}$$

Proof: We prove the theorem by contradiction, assuming that there exists a job of task τ_k which is unfinished in the time interval $[0, R_k^{L1}]$. Since the lower service curve is super-additive, we have:

$$\begin{aligned}
\beta^l\left(R_k^{L1}\right) \geq\ & \beta^l\left(\sum_{j \in hep(k)\cap\Theta_c} \left\lceil \frac{R_k^{L1}}{T_j} \right\rceil \cdot \beta^{-1}(E_j^{LO})\right) + \\
& \beta^l\left(\sum_{\ell \in hep(k)\cap\Theta_g} \left\lceil \frac{R_k^{L1}}{T_\ell} \right\rceil \cdot C_\ell^{LO}\right) \\
\geq\ & \sum_{j \in hep(k)\cap\Theta_c} \left\lceil \frac{R_k^{L1}}{T_j} \right\rceil \cdot E_j^{LO} + \\
& \sum_{\ell \in hep(k)\cap\Theta_g} \left\lceil \frac{R_k^{L1}}{T_\ell} \right\rceil \cdot \beta^l\left(C_\ell^{LO}\right) \\
\geq\ & \sum_{h \in hep(k)} \left\lceil \frac{R_k^{L1}}{T_h} \right\rceil E_h^{LO}
\end{aligned} \tag{3}$$

where $\Theta_c \overset{\text{def}}{=} \{\tau_h \in \tau | \beta^{-1}(E_h^{LO}) > C_h^{LO}\}$ and $\Theta_g \overset{\text{def}}{=} \{\tau_h \in \tau | \beta^{-1}(E_h^{LO}) \leq C_h^{LO}\}$. Then according to Def.3, we know that the energy demand of the k^{th} priority level is satisfied. On the other hand, from Eq. (2), we have:

$$R_k^{L1} > \sum_{h \in hep(k)} \left\lceil \frac{R_k^{L1}}{T_h} \right\rceil C_h^{LO} \tag{4}$$

Inequality (4) shows that the processor demand of the k^{th} priority level is satisfied according to Def. 2. Finally, combining (3) and (4), we show that the time demand of the k^{th} priority level is satisfied according to Def. 4, which contradicts to our assumption. Proved. ■

2) Second Response Time Bound: The response time bound in Theorem 1 is relatively simple, but may lead to gross overestimation. In the following we develop a more precise response time bound. Before going into details, we first introduce some useful concepts below.

Definition 5. *Let Δ_i satisfy $\Delta_i = \beta^{-1}(P_i)$. Then, task τ_i is a consuming task if $\Delta_i > 1$. τ_i is a gaining task if $\Delta_i \leq 1$. Let Γ_c denote the set of consuming tasks, and Γ_g denote the set of gaining tasks.*

Definition 6 (from [4])**.** *Execution unit is defined as a non-divisible time unit of execution of a job. Replenishment unit is defined as the minimum indivisible unit of idling time used to replenish energy. An execution sequence is an ordered collection X of execution units from 1 to L_X, where L_X is the number of execution units in the sequence. Each element $X[m]$ ($m \in \{1...L_X\}$) of the sequence indicates the task that the execution unit belongs to. $E_X[m]$ denotes the energy required by execution unit $X[m]$. $E_X^*[m]$ denotes the total energy required by execution units from the start of the sequence up to and including execution unit $X[m]$. Thus:*

$$E_X^*[m] = \sum_{q=1...m} E_X[q] \tag{5}$$

$I_X[m]$ denotes the minimum number of replenishment units required to provide sufficient energy to execute $X[m]$ at the end of the subsequence $X[1]$ to $X[m]$. $I^*_X[m]$ denotes the minimum number of replenishment units required to execute all of the subsequence $X[1]$ to $X[m]$ in order. Thus:

$$I^*_X[m] = \max_{q=1\ldots m} I_X[q] \tag{6}$$

Based on energy source model introduced in Section III, $I_X[m]$ can be given by:

$$I_X[m] = \max\left(0, \left\lceil \beta^{-1}\left(E^*_X[m]\right)\right\rceil - m\right) \tag{7}$$

Then, we introduce the following auxiliary lemma.

Lemma 1. *If any execution sequence X contains only execution units of consuming tasks, then we have*

$$I^*_X[L_X] = I_X[L_X] = \left\lceil \beta^{-1}\left(E^*_X[L_X]\right)\right\rceil - L_X \tag{8}$$

Proof: We use ℓ_m to denote the value satisfying the following equation

$$\ell_m = \beta^{-1}\left(E^*_X[m]\right) \tag{9}$$

First, we show by induction that $\ell_m - m > 0$ if any execution sequence X contains only execution units of consuming tasks.

Base Case: For k=1, $\ell_1 > 1$. Since sequence X contains only execution units of consuming tasks, the claim is true according to Def.6.

Inductive Step: Suppose that for k the claim $\ell_k > k$ is true. Now we prove that $\ell_{k+1} > k+1$.

By $\ell_k > k$ and the definition of β^{-1}, we have

$$\begin{aligned}\beta^l(\ell_{k+1}) &\geq E^*_X[k+1] \\ &= E^*_X[k] + E_X[k+1] \qquad (10)\\ &> \beta^l(k) + \beta^l(1) \qquad (11)\end{aligned}$$

We assume that $\beta^l(\ell_{k+1}) \leq \beta^l(k+1)$. By (11), we have $\beta^l(k+1) > \beta^l(k) + \beta^l(1)$. However, there exist some counter-examples. For example, if β^l is given by $\beta^l(\Delta) = \Delta$, we have $\beta^l(k+1) = \beta^l(k) + \beta^l(1)$. Hence, we can derive $\beta^l(\ell_{k+1}) > \beta^l(k+1)$. Since service curves are monotonically non-decreasing, we then have $\ell_{k+1} > k+1$. Therefore, by induction, if any execution sequence X contains only execution units of consuming tasks, we get $\ell_m - m > 0$. Applying (9) to $\ell_m - m > 0$, we have

$$\left\lceil \beta^{-1}\left(E^*_X[m]\right)\right\rceil - m > 0 \tag{12}$$

Combining (7) and (12), we have:

$$I_X[m] = \left\lceil \beta^{-1}\left(E^*_X[m]\right)\right\rceil - m \tag{13}$$

On the other hand, we show that $I_X[m]$ is non-decreasing. First we show that $\lceil \ell_{m+1} \rceil > \lceil \ell_m \rceil + 1$ if any execution sequence X contains only execution units of consuming tasks.
By (10) and the definition of consuming tasks, we have

$$\beta^l(\ell_{m+1}) > E^*_X[m] + \beta^l(1) \tag{14}$$

We assume that $\beta^l(\ell_{m+1}) \leq \beta^l(\ell_m) + \beta^l(1)$. Applying this to (14), we have $\beta^l(\ell_m) > E^*_X[m]$. However, this is incorrect in the following counter-example: if β^l is given by $\beta^l(\Delta) = \Delta$ and ℓ_m numerically equals to $E^*_X[m]$, we have $\beta^l(\ell_m) = E^*_X[m]$. Therefore, $\beta^l(\ell_{m+1}) > \beta^l(\ell_m) + \beta^l(1)$. Then we can derive $\beta^l(\ell_{m+1}) > \beta^l(\ell_m + 1)$. Since service curves are monotonically non-decreasing, we therefore have

$$\begin{aligned}&\ell_{m+1} > \ell_m + 1 \\ &\Rightarrow \lceil \ell_{m+1} \rceil \geq \lceil \ell_m + 1 \rceil \\ &\Rightarrow \lceil \ell_{m+1} \rceil \geq \lceil \ell_m \rceil + 1\end{aligned} \tag{15}$$

Applying (9) to (15), we have

$$\left\lceil \beta^{-1}(E^*_X[m+1])\right\rceil - \left\lceil \beta^{-1}(E^*_X[m])\right\rceil \geq 1 \tag{16}$$

Applying (13) and (16) to (7), we have

$$I_X[m+1] - I_X[m] \geq 0 \tag{17}$$

Inequality (17) implies that $I_X[m]$ is non-decreasing with respect to m. Hence, we therefore have

$$I^*_X[m] = I_X[m] \tag{18}$$

Combining (13) and (18), we conclude the proof. ■

Based on the above lemma, we can derive the second response time bound from the following theorem.

Theorem 2. *In LO-mode, the worst-case response time of task τ_i is upper bounded by:*

$$R_i^{L2} = \lceil \lambda_i^{LO} \rceil + \sum_{j \in \Gamma_g \cap hep(i)} \left\lceil \frac{R_i^{L2}}{T_j} \right\rceil C_j^{LO} \tag{19}$$

where

$$\lambda_i^{LO} = \beta^{-1}\left(\sum_{k \in \Gamma_c \cap hep(i)} \left\lceil \frac{R_i^{L2}}{T_k} \right\rceil E_k^{LO} \right) \tag{20}$$

Proof: Assume that there is an execution sequence which contains only $\sum_{k \in \Gamma_c \cap hep(i)} \left\lceil \frac{R_i^{L2}}{T_k} \right\rceil C_k^{LO}$ execution units of consuming tasks. By Lemma 1, we can derive

$$\lceil \lambda_i^{LO} \rceil > \sum_{k \in \Gamma_c \cap hep(i)} \left\lceil \frac{R_i^{L2}}{T_k} \right\rceil C_k^{LO} \tag{21}$$

Applying (21) to (19), we have:

$$R_i^{L2} > \sum_{h \in hep(i)} \left\lceil \frac{R_i^{L2}}{T_h} \right\rceil C_h^{LO} \tag{22}$$

From (22), we know the processor demand of the i^{th} priority level is satisfied according to Def. 2. On the other hand, by (19), we have:

$$\beta^l\left(R_i^{L2}\right) \geq \beta^l\left(\lambda_i^{LO}\right) + \sum_{j \in \Gamma_g \cap hep(i)} \left\lceil \frac{R_i^{L2}}{T_j} \right\rceil \beta^l\left(C_j^{LO}\right) \tag{23}$$

If $\tau_j \in \Gamma_g$, by Def. 5 we have:

$$\begin{aligned}\beta^l\left(C_j^{LO}\right) &\geq \beta^l\left(C_j^{LO} \cdot \Delta_j\right) \\ &\geq \beta^l\left(\Delta_j\right) \cdot C_j^{LO} \\ &\geq P_j \cdot C_j^{LO} \\ &= E_j^{LO}\end{aligned} \tag{24}$$

Applying (20) and (24) to (23), we therefore have:

$$\beta^l\left(R_i^{L2}\right) \geq \sum_{h \in hep(i)} \left\lceil \frac{R_i^{L2}}{T_h} \right\rceil E_h^{LO} \tag{25}$$

By (25), we know that the energy demand of the i^{th} priority level is satisfied according to Def. 3. Finally, by (22) and (25), we know the time demand of the i^{th} priority level is satisfied according to Def. 4. Proved. ■

3) Comparing Theorem 1 and 2: In the following we prove dominance relationships between Theorem 1 and 2.

Theorem 3. *Schedulability test based on Theorem 2 dominates the test based on Theorem 1.*

Proof: Given a time interval of length x. Then we define:

$$f_i^{L1}(x) = \sum_{h \in hep(i)} \left\lceil \frac{x}{T_h} \right\rceil \cdot \max\left\{\left\lceil \beta^{-1}(E_h^{LO}) \right\rceil, C_h^{LO}\right\} \tag{26}$$

$$f_i^{L2}(x) = \left\lceil \lambda_i^{LO} \right\rceil + \sum_{j \in \Gamma_g \cap hep(i)} \left\lceil \frac{x}{T_j} \right\rceil C_j^{LO} \tag{27}$$

where

$$\lambda_i^{LO} = \beta^{-1}\left(\sum_{k \in \Gamma_c \cap hep(i)} \left\lceil \frac{x}{T_k} \right\rceil E_k^{LO}\right) \tag{28}$$

We prove the theorem by showing that $f_i^{L2}(x) \leq f_i^{L1}(x)$.

According to Lemma 4 in [14], we know β^{-1} is an upper curve, which is sub-additive. Applying this property to (28), we have:

$$\lambda_i^{LO} \leq \sum_{k \in \Gamma_c \cap hep(i)} \left\lceil \frac{x}{T_k} \right\rceil \cdot \beta^{-1}(E_k^{LO}) \tag{29}$$

Applying (29) to (27), we conclude the proof. ■

B. Response Time Bound of HI-Mode

In HI-mode, all LO-tasks are descheduled, and only HI-tasks can execute. Therefore the analysis is almost identical to the case of LO-mode, except that we only need to consider HI-tasks by replacing all occurrences of LO with HI in the above equations.

C. Response Time Bound of Mode-Switch Phase

In this subsection, we calculate two upper bounds on the worst-case response time of task τ_i during mode-switch phase, based on Theorem 1 and 2, respectively.

Theorem 4. *During mode-switch phase, an upper bound on the worst-case response time of each HI-task τ_i based on Theorem 1 is given by:*

$$R_i^{M1} = \sum_{j \in hpH(i)} \left\lceil \frac{R_i^{M1}}{T_j} \right\rceil \cdot \max\left\{\left\lceil \beta^{-1}(E_j^{HI}) \right\rceil, C_j^{HI}\right\} + \sum_{h \in hpL(i)} \left\lceil \frac{R_i^{L1}}{T_h} \right\rceil \cdot \max\left\{\left\lceil \beta^{-1}(E_h^{LO}) \right\rceil, C_h^{LO}\right\} \tag{30}$$

where R_i^{L1} is calculated in Theorem 1.

Proof: The proof is similar to Theorem 1 except that HI-tasks can execute in $[0, R_i^{M1}]$ and all LO-tasks are only allowed to execute in $[0, R_i^{L1}]$ due to mode-switch. ■

Theorem 5. *During mode-switch phase, an upper bound on the worst-case response time of each HI-task τ_i based on Theorem 2 is given by:*

$$R_i^{M2} = \left\lceil \lambda_i^{HI} \right\rceil + \sum_{h \in hpH(i) \cap \Gamma_g} \left\lceil \frac{R_i^{M2}}{T_h} \right\rceil C_h^{HI} + \sum_{\ell \in hpL(i) \cap \Gamma_g} \left\lceil \frac{R_i^{L2}}{T_\ell} \right\rceil C_\ell^{LO} \tag{31}$$

where R_i^{L2} is calculated in Theorem 2 and

$$\lambda_i^{HI} = \beta^{-1}\left(\sum_{k \in hpH(i) \cap \Gamma_c} \left\lceil \frac{R_i^{M2}}{T_k} \right\rceil E_k^{HI} + \sum_{j \in hpL(i) \cap \Gamma_c} \left\lceil \frac{R_i^{L2}}{T_j} \right\rceil E_j^{LO}\right)$$

The proof is similar to Theorem 2 and thus omitted.

Theorem 6. *Schedulability test based on Theorem 5 dominates the test based on Theorem 4.*

The proof is similar to Theorem 3 and thus omitted.

VI. Experimental Evaluation

In this section, we evaluate the effectiveness of our analysis techniques with solar energy source. The service curve is obtained from the energy input traces and the synthetic taskset is used along with service curve for this evaluation.

The power trace generation. To collect solar energy trace, we use AD7606 which is a data acquisition system with 16-bit, bipolar input, simultaneous sampling ADC to measure the voltage across the solar panel connected to a load resistor, then these voltage data are sent to the STM32RBT6 launchpad by SPI. We record the energy trace from the solar panel during a day in August at $40°N$ and $120°E$. We use the segment starting from 11:26am to 11:31am to generate service curve. In order to efficiently compute this curve, we use a piecewise linear approximation of this curve which is given by: $\beta^l(\Delta) = 5.5 \cdot [\Delta - 0.4]^+$ where the operator $[\ ]^+$ is defined as $[x]^+ = \max(0, x)$.

Taskset parameter generation. Task processor utilizations ($U_i = C_i/T_i$) were generated using the UUnifast algorithm [8]. Task energy utilizations ($U_i^e = E_i/\beta^l(T_i)$) were generated using an adapted version of UUnifast to control the type of task consumed. Task periods were generated between 2 and 25,200 time units with a hyper-period limitation technique [9]. Task deadlines were set equal to their periods. The LO-WCET of each task was given based on utilization and period: $C_i^{LO} = U_i \cdot T_i$. The HI-WCET of each task was given by $C_i^{HI} = CF \cdot C_i^{LO}$ where CF is the Criticality Factor and default $CF = 2.0$. The probability that a generated task was a HI-task was set by the Criticality Proportion, default $CP = 0.5$. The LO-WCEC of each task was given by: $E_i^{LO} = U_i^e \cdot \beta^l(T_i)$. The HI-WCEC of each task was given by $E_i^{HI} = CF \cdot E_i^{LO}$.

Experiments. In our experiments, we varid U and U^e in the range $[0.05, 1]$ in steps of 0.05. For each pair (U, U^e), 10000 task sets were generated. The default cardinality of the tasksets was 6. We investigated the performance of the following schedulability tests: (i) Theorem 4, denoted as UB1, (ii) Theorem 5, denoted as UB2, (iii) schedulability test where a taskset must be deemed schedulable according to response time bound in both LO-mode (the minimal response time bound obtained in between Theorem 1 and 2) and HI-mode (the minimal response time bound obtained by the method that is similar to LO-mode), denoted as UB-H&L. If any taskset is unschedulable by test UB-H&L, then it is also unschedulable by test UB1 and UB2. In case (i) and (ii), we use Audsley's algorithm [15] to assign priorities since [7] shows that an optimal priority ordering for AMC can be obtained by Audsley's algorithm. In case (iii), we use deadline monotonic priority ordering (DMPO) to assign priorities since DMPO is optimal in this case (the reasoning is similar to Theorem 11 in [4]). In some figures, we show the *weighted schedulability measure* $W_y(p)$ [16] for each schedulability test y as a function of parameter p. For each value of p, this measure combines results for all of the task sets τ generated for all of a set of equally spaced utilization levels (0.05 to 1 in steps of 0.05). Let $S_y(\tau, p)$ be the binary result (1 or 0) for the schedulability test y for a taskset Γ with parameter value p:

$$W_y(p) = \left(\sum_{\forall \tau} u(\tau) \cdot S_y(\tau, p) \right) \Big/ \sum_{\forall \tau} u(\tau) \qquad (32)$$

where $u(\tau)$ is the utilization of task set τ.

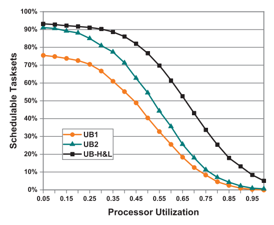
Fig.2(a) shows the percentage of task sets that are deemed schedulable by each of the tests varies with processor utilization. The dominance relationship between UB1 and UB2 is evidenced. Then we show how the weighted schedulability measure varies with different key parameters. Fig.2(b) varies the energy utilization, Fig.2(c) varies the criticality factor and Fig.2(d) varies task deadlines from 25% to 100% of the task's period. From the above results we observe that UB2 provides a tighter bound than UB1 under different parameter settings, giving overall performance that is close to the limit illustrated by the UB-H&L upper bound.

VII. Conclusions

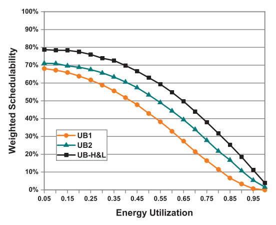
In this paper, we study the modeling and analysis for EHMC systems. We extend the AMC approach to EHMC systems and derive two response time upper bounds. Next step, we will extend the scheduling policy to Earliest Deadline First (EDF) and also extend the task model to arbitrary deadlines. We would also like to consider more general mixed-criticality systems.

Acknowledgment

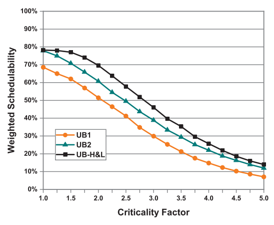
This work is supported by the LinoNing Revitalization Talents Program (XLYC1902017) and National Natural Science Foundation of China (Grant No.62072085).



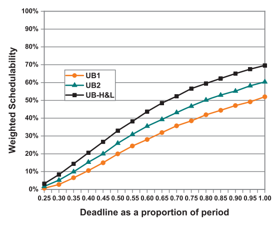
(a) Success Ratio



(b) Varying the energy utilization



(c) Varying the Criticality Factor



(d) Varying relative deadlines

Fig. 2. Experiment results.

References

[1] P.J.Prisaznuk. Integrated modular avionics. In National Aerospace and Electronics Conference, 1992.
[2] AUTOSAR. AUTomotive Open System ARchitecture. www.autosar.org, 2005.
[3] Nan Guan et al. Modular Performance Analysis of Energy-Harvesting Real-Time Networked Sytems. In RTSS, 2015.
[4] Yasmina Abdeddaïm et al. Response time analysis for fixed priority real-time systems with energy-harvesting. Journal of RTS, 2016.
[5] C. Moser et al. Real-Time Scheduling with Regenerative Energy. In ECRTS, 2006.
[6] Sedigheh Asyaban et al. Analysis and Scheduling of a Battery-Less Mixed-Criticality System with Energy Uncertainty. ACM Transactions on Embedded Computing Systems, 2016.
[7] S. K. Baruah et al. Response-Time Analysis for Mixed Criticality Systems. In RTSS, 2011.
[8] E. Bini et al. Measuring the performance of schedulability tests. Journal of RTS, 2005.
[9] Joel Goossens et al. Limitation of the hyper-period in real-time periodic task set generation. In Proceedings of the RTS embedded system, 2001
[10] André Allavena et al. Scheduling of frame-based embedded systems with rechargeable batteries. In Workshop on power management for real-time and embedded systems (in conjunction with RTAS01), 2001.
[11] Lothar Thiele et al. Performance evaluation of network processor architectures: combining simulation with analytical estimation. In Computer Networks, 2003.
[12] Yasmina Abdeddaïm et al. The optimality of PFP_{ASAP} algorithm for fixed-priority energy-harvesting real-time systems. In ECRTS, 2013.
[13] Yasmina Abdeddaïm et al. Schedulability analysis for fixed priority real-time systems with energy-harvesting. In RTNS, 2014.
[14] Victor Pollex et al. Generalizing Response-Time Analysis. In RTCSA, 2010.
[15] N. Audsley. On priority assignment in fixed priority scheduling. Information Processing Letters, 2001.
[16] A. Bastoni et al. Cache-related preemption and migration delays: Empirical approximation and impact on schedulability. In OSPERT, 2010.
[17] D. Lee et al. Real-time schedulability analysis and enhancement of transiently powered processors with NVMs. IEEE Transactions on Computers, 2021.
[18] B. Islam et al. Scheduling computational and energy harvesting tasks in deadline-aware intermittent systems. In RTAS, 2020.