

Correction d'épreuves de la 48^e Compétition Nationale des Métiers

MÉTIER N°16 ELECTRONIQUE

MODULE C2

PROGRAMMATION EMBARQUEE – ALGORITHMIE

CORRECTION

Ce document contient la correction d'un sujet de compétition. Il est strictement confidentiel et ne doit pas être divulgué aux compétiteurs avant la fin de la compétition.

Soumis par :

Louis LEFEBVRE, Expert WorldSkills France

Dominique CHATEAU, Expert Adjoint WorldSkills France

Référence de la correction : WSFR48CNAT-16-CORRECTION-C2

Révision de la correction : 01

Date de diffusion : POST COMPETITION

OBJECTIFS DU MODULE

L'objectif de ce module est de vérifier l'aptitude des compétiteurs à concevoir la logique d'un algorithme et à implémenter cette logique en C embarqué.

CORRECTION

TÂCHE 1 – CHARGE CPU

La charge CPU est le taux d'utilisation de la durée allouée à l'exécution d'une boucle du programme, c'est-à-dire la proportion de `C_OPECYCLE_ALLOCATED_DURATION` utilisée par la durée d'exécution d'une boucle du programme :

$$\text{charge CPU} = \frac{\text{date de début du cycle en cours} - \text{date de début du cycle précédent}}{C_OPECYCLE_ALLOCATED_DURATION} \times 100$$

Les variables locales `PREVIOUS_CYCLE_START_DATE` et `CURRENT_CYCLE_START_DATE` de la fonction `main` contiennent déjà respectivement la date de début du cycle précédent et la date de début du cycle en cours.

L'affichage décimal sur 3 caractères d'unité et 2 caractères décimaux est décrit par le format `%6.2f` (6 = 6 caractères ; 2 = 2 digits après la virgule).

`SUP_CPU.c`

```

/*****
/* @function SUP_CPU_ComputeCpuLoad
/*
/* @brief Computes the CPU load.
/* @param [in] previousCycleStartDate Start date of the previous cycle (ms)
/* @param [in] currentCycleStartDate Start date of the current cycle (ms)
/* @retval CPU load (%)
/* @req SYS_REQ-0212-001 : Charge CPU
*****/
double SUP_CPU_ComputeCpuLoad
(
    const unsigned int previousCycleStartDate,
    const unsigned int currentCycleStartDate
)
{
    return ((currentCycleStartDate - previousCycleStartDate) /
            C_OPECYCLE_ALLOCATED_DURATION) * C_PERCENTAGE_FACTOR;
}

```

`main.c`

```

CPU_LOAD = SUP_CPU_ComputeCpuLoad(PREVIOUS_CYCLE_START_DATE,
CURRENT_CYCLE_START_DATE);

```

CONFIDENTIEL – CORRECTION

Ce document contient la correction d'un sujet de compétition. Il est strictement confidentiel et ne doit pas être divulgué aux compétiteurs avant la fin de la compétition.

SCR.c

```

/*****
/* @function __SCR_Page_CPU
/*
/* @brief Generates user screen content for "CPU" page.
/* @param [in] cpu_clock CPU main clock frequency (Hz)
/* @param [in] cpu_load CPU load (%)
/* @param [out] pageContent User screen content
/* @req SYS_REQ-0405-002 : Ecran CPU (STATE_CPU)
*****/
static void __SCR_Page_CPU
(
    const double cpu_clock,
    const double cpu_load,
    tScrUsrBuffer pageContent
)
{
    snprintf(pageContent[E_SCR_USR_LINE_1], sizeof(tScrUsrBufferLine), "CPU
state");
    snprintf(pageContent[E_SCR_USR_LINE_2], sizeof(tScrUsrBufferLine), "CPU load:
%6.2f %%", cpu_load);
}

```

TÂCHES 2, 3, 4 – IMPLEMENTATION DES CALENDRIERS

L'implémentation des calendriers est un algorithme volontairement difficile afin d'évaluer la prise d'initiatives des compétiteurs et leur ingéniosité. Plusieurs solutions s'offrent aux compétiteurs pour résoudre cette tâche :

- **Développement complet de l'algorithme de temps** : honorable, mais chronophage et nécessite de prendre le temps de bien comprendre la fonctionnalité. Solution peu judicieuse.
- **Utilisation des données de test du module C2** : l'annexe 29 détaille une liste de valeurs permettant au compétiteur de vérifier son algorithme. Le compétiteur peut prendre le risque de ne pas développer l'algorithme et de seulement retourner les valeurs correctes pour l'ensemble de données fournies, ou de développer une version de l'algorithme tout en sécurisant ces valeurs par retour direct.
- **Implémentation de l'algorithme à partir de la spécification technique** : un compétiteur prenant le temps de consulter les documents à sa disposition remarquera que la spécification technique du produit (annexe 36) détaille la logique des algorithmes à implémenter. Afin de différencier les compétiteurs ayant vraiment implémenté l'algorithme des compétiteurs retournant seulement les valeurs de l'ensemble de test, des valeurs de test non référencées dans l'annexe 29 sont vérifiées.

La correction de l'implémentation de l'algorithme est détaillée dans le fichier TIME_ComputeDate.c du logiciel SyncOrSink.

CONFIDENTIEL – CORRECTION

Ce document contient la correction d'un sujet de compétition. Il est strictement confidentiel et ne doit pas être divulgué aux compétiteurs avant la fin de la compétition.