Version du 24/03/2021

Questionnaire d’auto-évaluation pour l’accès à la filière Systèmes Embarqués (SE) de Télécom Paris

Introduction

Ce questionnaire, à destination des étudiants souhaitant accéder directement à la filière Systèmes Embarqués (SE) de Télécom Paris sans passer par la 1re année, permet d’évaluer si vous avez les pré-requis nécessaires pour suivre cette filière dans de bonnes conditions. Au moment du démarrage de la filière, vous devriez être capable de répondre aux différentes questions sans avoir à consulter des ressources complémentaires. Si suite à ce questionnaire vous avez des questions ou des doutes sur les réponses, n’hésitez pas à contacter le responsable de la filière : Guillaume Duc (guillaume.duc@telecom-paris.fr).

Note : ce questionnaire ne couvre pas nécessairement tous les pré-requis de façon exhaustive.

Représentation des données

Soit le mot binaire (sur 8 bits) : 1111 1100. Quelle est la valeur, exprimée en base 10, de ce mot s’il représente un entier non signé ? Quelle est la valeur de ce mot, exprimé en base 10, s’il représente un entier codé en complément à deux (CA2) ?

Vous devez également être à l’aise sur le passage entre une notation binaire et une notation hexadécimale.

Langage C

La connaissance de base du langage C est impérative. La syntaxe de base du langage doit être maîtrisée et ne sera pas testée dans la suite de ce questionnaire.

Types des variables

Quels sont les types élémentaires du langage C ?

Visibilité et durée de vie des variables

```
// Exemple 1 : ficher a.c
int a;
void f() {
   // …
```

```
// Exemple 2 : fichier b.c
void f() {
   int a;
   // ...
```

} 	}

Pour chacun de ces deux exemples, la variable a est visible (utilisable) :

1. Uniquement dans la fonction f
2. Dans l'ensemble du fichier .c
3. Dans l'ensemble du programme

Pour chacun de ces deux exemples, la durée de vie de la variable a (durée pendant laquelle est est utilisable) est :

1. La durée d'un appel à la fonction f
2. La durée de vie du programme

Enfin, pour chacun de ces deux exemples, indiquez dans quelle zone (tas, pile, data) de la mémoire la variable a est stockée.

Pointeurs

Qu'est-ce qu'un pointeur ?

```
#include <stdio.h>
int main() {
   int a = 4;
   int *p = &a;
   printf("val = %d\n", *p);
   return 0;
}
```

Qu'affiche le code précédent ? Pourquoi ?

```
#include <stdio.h>
int main() {
   int *p = 0;
   printf("val = %d\n", *p);
   return 0;
}
```

Qu'affiche le code précédent ? Pourquoi ?

```
#include <stdio.h>
int main() {
   int t[3]; t[0] = 10; t[1] = 11; t[2] = 12;
   int *p = t;
   printf("val = %d\n", *(p+1));
   return 0;
}
```

Qu'affiche le code précédent ? Pourquoi ?

Version du 24/03/2021

Chaînes de caractères

Écrivez le code d'une fonction string_len qui prend en argument une chaîne de caractères quelconque et qui affiche le nombre de caractères de cette chaîne.

```
#include <stdio.h>
int main() {
   char c = '1';
   printf("c = %c (%d)\n", c, c);
   return 0;
}
```

Qu'affiche le programme précédent ?

Structures

Soit la structure suivante :

```
struct s {
   int a;
   int b;
};
```

Comment définir une variable v de ce type ? Comment accède-t-on aux différents membres (a et b) de cette variable v ?

```
void f(struct s *v)
```

Soit la fonction ayant la signature ci-dessus. Comment accéder, à l'intérieur de celle-ci, aux différents membres de son argument ?

Allocation mémoire

```
int * alloc() {
   int a[16];
   return a;
};
```

Cette fonction a été écrite pour allouer dynamiquement de la mémoire. Fonctionne-t-elle correctement ? Pourquoi ?

Quelle fonction permet d'allouer dynamiquement de la mémoire ?

Compilation séparée

```
#include <stdio.h>
int main() {
   int i = 3;
   printf("carre(i) = %d\n", carre(i));
   return 0;
}
int carre(int a) {
   return a * a;
```

```
}
```

Lors de la compilation, le compilateur émet un avertissement au niveau de la quatrième ligne. Indiquez pourquoi et deux méthodes pour corriger.

```
// a.c
int a = 42;
int main() {
   int b = inc();
   return 0;
}
```

```
// b.c
int inc() {
   return a + 1;
}
```

Corrigez ces deux fichiers pour que leur compilation ne produise pas de messages d'erreur ou d'avertissement, de manière la plus idiomatique possible.

Une fois corrigés, les deux fichiers précédents sont compilés avec les commandes ci-dessous :

```
$ gcc -Wall -g -c a.c
$ gcc -Wall -g -c b.c
$ gcc -g -o main a.o b.o
```

Pour chacune des trois commandes, indiquez ce qu'elles font, le ou les fichiers d'entrée et de sortie, et la signification des arguments.

À quoi sert la compilation séparée ?

```
// Fichier a.h
int a;
int carre(int a) {
   return a * a;
}
```

Corrigez ce fichier d'entête pour qu'il soit le plus idiomatique possible (indice : 3 éléments à corriger).

Systèmes d'exploitation UNIX/Linux

Utilisation de base de la ligne de commandes

Vous devez être à l'aise avec la ligne de commande (shell). Vous devez notamment être capable : d'afficher le contenu d'un répertoire (y compris les fichiers « cachés »), de vous déplacer dans l'arborescence des répertoires (en utilisant un chemin relatif ou absolu), de créer un répertoire, de copier/déplacer/supprimer des fichiers, de lancer un exécutable présent dans le répertoire courant, de lancer un exécutable en tâche de fond…

Vous devez également connaître la signification de la variable d'environnement PATH.

Gestion des processus

Que font les appels systèmes fork, exec, wait et exit ? Que signifie leurs valeurs de retour ?

```
#include <sys/types.h>
#include <sys/wait.h>
```

Version du 24/03/2021

```
#include <unistd.h>
#include <stdio.h>

int main() {
   pid_t res;
   printf("1\n");
   res = fork();
   printf("2 – fork = %d\n", res);
   if (res == 0) {
      execl("/usr/bin/ls", "ls", (char *) NULL);
      printf("3\n");
   }
   else if (res > 0) {
      int s;
      wait(&s);
      printf("4\n");
   }
   printf("5\n");
   return 0;
}
```

Qu'affiche le programme précédent (en supposant que fork ne retourne pas une erreur, et que le fichier /usr/bin/ls existe et est exécutable) ?

Synchronisation

Qu'est-ce qu'un sémaphore ? Quelles sont les opérations possible sur un sémaphore ? Que font chacune de ces opérations ?

Comment faire de l'exclusion mutuelle à l'aide d'un sémaphore ?

Processeur

Vous devez avoir des notions de base sur le fonctionnement d'un processeur et d'un ordinateur. Vous devez en particulier être capable de définir ce qu'est : un processeur, une instruction, un registre, un compteur ordinal (ou compteur de programme), une pile (au sens structure de données), une interruption.

Électronique numérique

Logique combinatoire

Vous devez être à l'aise avec la logique combinatoire : qu'est-ce que la logique combinatoire, les fonctions logiques de base (ET, OU, OU exclusif…), les constructions de base (multiplexeur, décodeur…), la notion de temps de propagation.

Comment construire un additionneur complet 1 bit (entrées : opérande a, opérande b, retenue entrante, sorties : résultat, retenue sortante, tous les signaux étant sur 1 bit) ?

Version du 24/03/2021

Comment construire un additionneur 8 bits à partir d’additionneurs 1 bit ?

Logique séquentielle

À quoi sert une bascule D (D-flip flop) ?

Proposer le schéma d’un compteur modulo 256 qui s’incrémente à chaque cycle d’une horloge clk.

Vous disposez d’une horloge clk et d’un signal a relié à un bouton (l’appui sur le bouton dure plusieurs cycles d’horloge). Proposez le schéma d’un compteur modulo 256 qui compte le nombre de fois où le bouton est appuyé.

Qu’est-ce qui limite la fréquence maximale de fonctionnement d’un circuit électronique ?

Langage de description matériel

Vous devez connaître la syntaxe de base d’un langage de description matériel (HDL, par exemple : Verilog, SystemVerilog ou VHDL).

Décrivez, dans un de ces langages, les deux exemples mentionnés plus haut dans la partie séquentielle.

Git

Vous devez être capable d’utiliser git, depuis la ligne de commande.