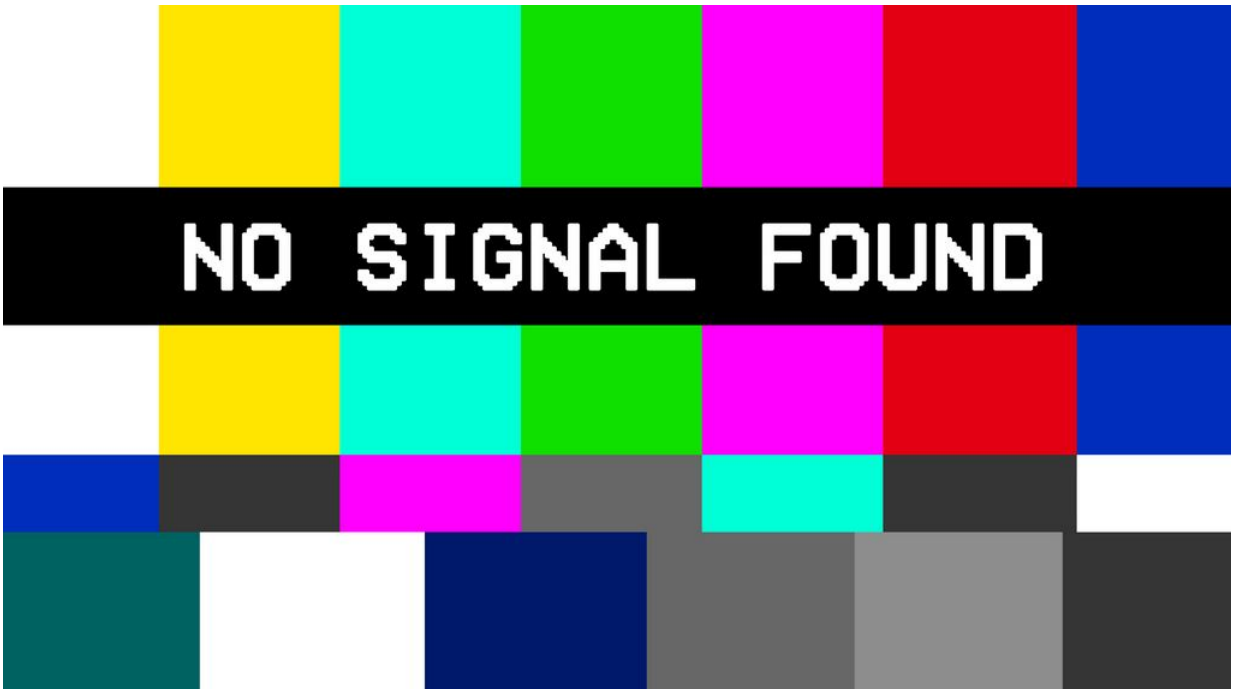
Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Branche

Sujet du cours // labo

Auteur(s) :
Prénom 1 Nom 1
Prénom 2 nom 2

Professeur :
Prénom Nom



10 novembre 2018

Table des matières

1 Conception Header File

1.1 Pragma once

Il permet d'éviter un import multiple de header files en incluant une fois uniquement les fichiers dans la compilation et peut être accompagné des commandes preprocessor ifndef symbol define

1.2 ifndef

Cette commande sert à éviter les boucles d'appel (si la classe A appelle la B et la B appelle la A)

1.3 Exemple

```
#pragma once
#ifndef SERPENTINE_H
#define SERPENTINE_H

extern void serpentine_init();

extern void serpentine_process();

extern void serpentine_reset();

#endif
```

2 Appel fonction C

3 Décrire le passage d'arguments par valeur et par référence lors d'appel de fonctions en C

3.1 passage d'arguments par valeur

Dans ce cas, on passe une valeur à notre fonction, une copie locale que l'on donne à notre fonction, pour effectuer différents traitements.

- Le sous-programme reçoit une copie de la donnée
- Il peut la lire et la modifier sans que la donnée originale ne soit altérée

- Il existe deux positions mémoires distinctes
- La modification de l'une des positions n'a aucune conséquence sur l'autre position.

Exemple : $intfnct1(inta, charc)$

Mais cette methode a quelques inconvénients, on ne peut pas retourner plusieurs éléments. c'est pourquoi en C on peut procéder au passage par référence.

3.2 passage d'arguments par référence lors d'appel de fonctions en C

Pour cette methode on donne l'adresse de notre variable en paramètre de notre fonction. on dit que l'on crée un pointeur qui pointe sur l'adresse de notre variable

- Le sous-programme reçoit l'adresse de la donnée
- Cette donnée est ainsi partagée entre la fonction appelante et la fonction appelée
- La donnée ne se situe que dans une seule position mémoire
- La modification du paramètre par la fonction appelée est visible par la fonction appelante

Exemple : $intfnct2(char * s, int * b);$

4 Différencier les fonctions globales des fonctions locales dans une application codée en C

Les variables globales sont définies à l'extérieur de la fonction. Même si elles sont à l'extérieur de notre fonction elles sont quand même accessibles.

Les variables définies à l'intérieur de notre fonction ne sont pas accessibles à l'extérieur.

Mais cependant, on peut modifier une variables se trouvant à l'extérieur de notre fonction via l'utilisation de pointeurs. Il faut toutefois que le pointeur nous soie donné en paramètre de notre fonction.

4.1 Tips

Il est déconseillé de laisser l'accès à ces variables à nos fonctions. celà a pour effet de crée des effets de bords non voulus.

5 Différencier les variables globales des variables locales et des variables rémanentes dans une application codée en C

wallah c'est pareil que le point précédent.

6 Manipuler correctement les types complexes (énumérations, tableaux, structures, unions, ...) de C

7 Concevoir une interface C permettant d'accéder aux registres d'un périphérique

8 Manipuler correctement les pointeurs en C

9 Allouer et restituer correctement des objets dynamiques en C

10 décrire et utiliser les conversions des types en C

11 Manipuler correctement les pointeurs de fonction en C

12 Expliquer les conditions pour qu'une fonction C soit réentrante

- Ne doit pas utiliser de données statiques (globales) non-constantes
- Ne doit retourner l'adresse de données statiques (globales) non-constantes
- Ne doit traiter que des données fournies par le programme appelant
- Ne doit pas modifier son propre code
- Ne doit pas appeler des sous-programmes non-réentrants