République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Tunis El Manar
Ecole Nationale d'Ingénieurs de Tunis
Département Génie Electrique

Bureau d'Études Systèmes Embarqués et Temps Réels

Réalisé par:

Nedin BOUZAIDA

Farah BOUSLEH

Classe:

3AGE1

Année universitaire: 2023/2024

Introduction

Les secteurs industriels dépendent fortement des systèmes temps réels, qui garantissent des niveaux optimaux de performance, de sécurité et de fiabilité. Ces systèmes reposent sur des principes fondamentaux tels que le déterminisme, la priorisation des tâches, l'ordonnancement et une gestion efficace des ressources disponibles.

Principes fondamentaux des systèmes temps réels

DÉTERMINISME

Les systèmes temps réels nécessitent une assurance de délais de réponse prévisibles. Cela implique une maîtrise et une limitation des délais de traitement, d'accès aux ressources et de communication.

PRIORISATION

Les tâches et processus dans un système temps réel sont hiérarchisés en fonction de leur importance et de leurs contraintes temporelles. Ceci assure l'exécution prioritaire des opérations critiques.

ORDONNANCEMENT

La planification des tâches est cruciale pour assurer le respect des contraintes temporelles. Divers algorithmes d'ordonnancement, comme ceux dédiés au temps réel, sont employés pour organiser la séquence d'exécution des tâches.

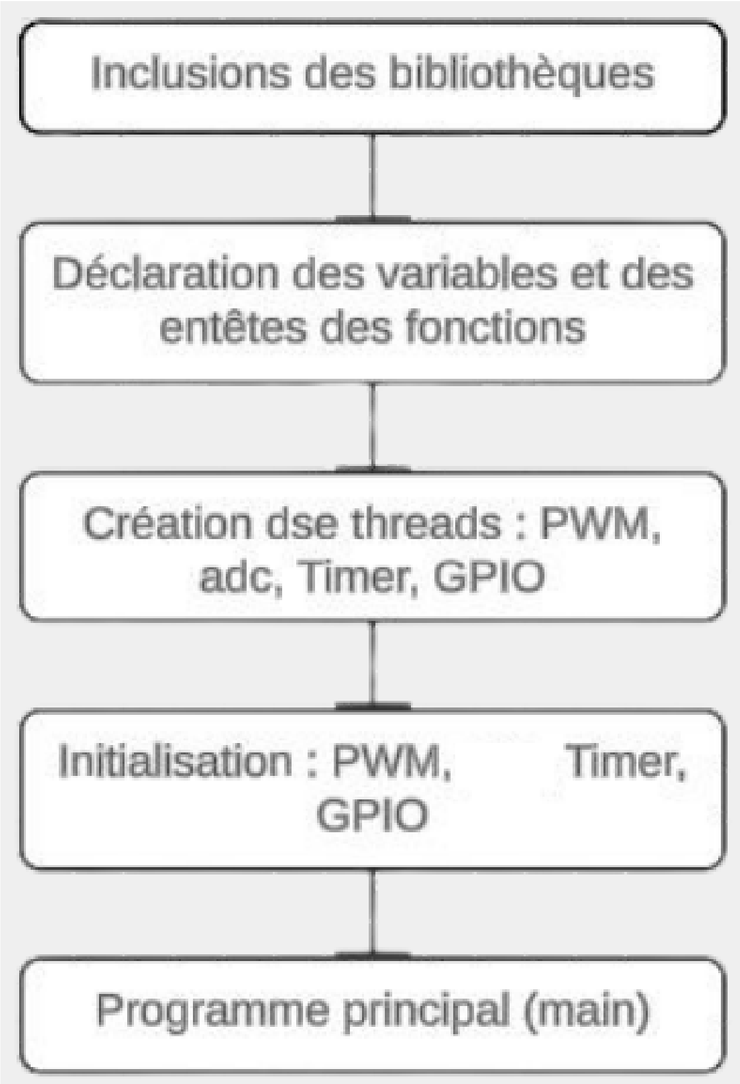
Implémentation d’un système temps réel sur BeagleBone Black

CARACTÉRISTIQUES PRINCIPALES

- **Processeur :** Texas Instruments Sitara AM3358 base sur l’architecture ARM Cortex-A8 cadencé a 1 GHz.
- **Mémoire :** 512 Mo de mémoire RAM DDR3.
- **Stockage** :4 Go de mémoire eMMC intégrée et emplacement pour carte microSD.
- **Connectivité :** Port Ethernet 10/100, ports USB hôte et esclave.
- **Interfaces vidéo :** Sortie HDMI
- **Interfaces d’extension :** Deux en-têtes d’extension pour capes et accessoires.
- **Système d’exploitation:** Prise en charge de divers systèmes d’exploitation Linux (Debian, Ubuntu, etc.)

Code et Réalisation

L’organigramme suivant représente la structure globale de notre programme:



Inclusion des bibliothèques

Ces directives comprennent diverses bibliothèques essentielles au bon fonctionnement du programme, notamment celles dédiées aux opérations système, à la gestion des threads, à la manipulation des GPIO, ainsi qu'à l'utilisation de la base de données

```
#include <sqlite3.h>
#include <pthread.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <signal.h>
#include <gpiod.h>
#include <fcntl.h>
#include <time.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <stdint.h>
#include <limits.h>
#include <pthread.h>
#include <sched.h>
#include <sys/mman.h>
```

Les variables globales du programme et les fonctions auxiliaires

Déclaration des variables et structures nécessaires pour travailler avec une base de données SQLite, un timer, un GPIO (General Purpose Input/Output), et des éléments liés à la gestion des threads.

```
20  sqlite3 *db;
21  char buf[20], str[10];
22  uint32_t T = 900000000, c = 0, r = 0;
23  int fd;
24  int dfile;
25  char kbhit(void);
26  timer_t timerid;
27
28  const char *chipname = "gpiochip1";
29  struct sigevent sev;
30  struct itimerspec trigger;
31  struct gpiod_chip *chip;
32  struct gpiod_line *lineRed;
33  int i;
34  int rt_init(void);
35  int ret;
36  pthread_attr_t attr;
37  pthread_cond_t cv, pwm;
38  pthread_mutex_t lock;
39
```

Développement des fonctions

On a développé des fonctions pour initialiser le timer, GPIO et le PWM :

Ces fonctions vont être appelé au niveau de la fonction main.

- **init_timer** : initialiser le timer
- **init_gpio** : Ouverture des lignes GPIO et les configurer
- **rt_init :** initialiser l'environnement temps réel du programme
- **init_pwm** : initialiser le PWM en configurant le pinmux pour activer la sortie PWM sur un port, définir la période et le rapport cyclique
- **save_to_database**: a pour rôle d'insérer une valeur value dans une table nommée adc_data d'une base de données SQLite nommée "ma_base_de_donnees.db"

```
void init_timer(void)
void init_gpio(void)
int rt_init(void)
void init_pwm(void)
void save_to_database(int value)
```

Création des Threads

Il existe principalement trois threads :

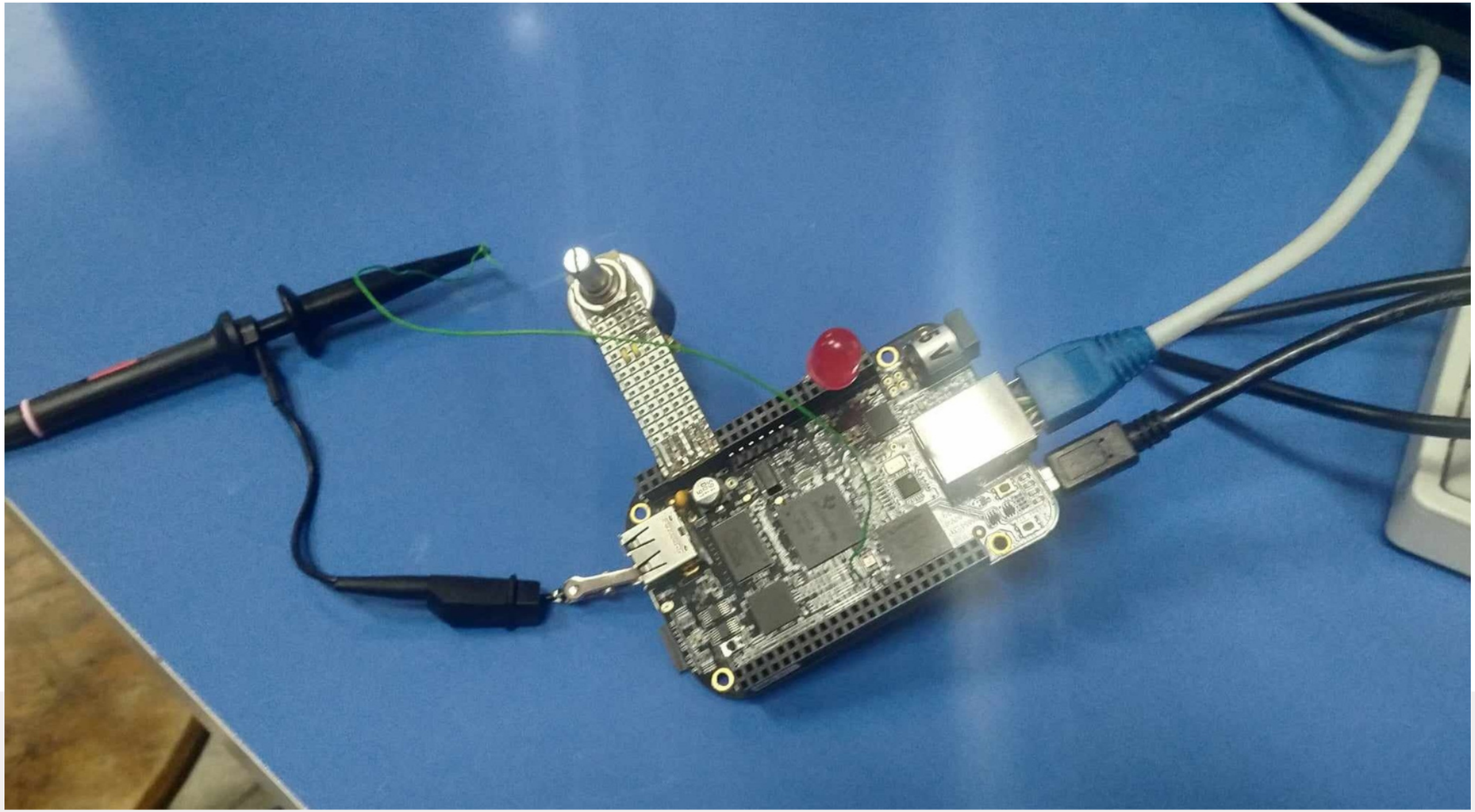
- **Tstimer_thread :** est le callback du thread du timer, alternant l'état d'une ligne GPIO, signalant un thread en attente, et réinitialisant le timer.
- **thread_adc** : Lit la valeur du signal analogique (ADC) à partir d'un fichier système et stocke cett valeur dans la base de données
- **thread_pwm** : Génère un signal PWM en fonction de la valeur lue par le thread ADC.

```
void *thread_pwm(void *v)
void *thread_adc(void *v)
void Tstimer_thread(union sigval sv)
```

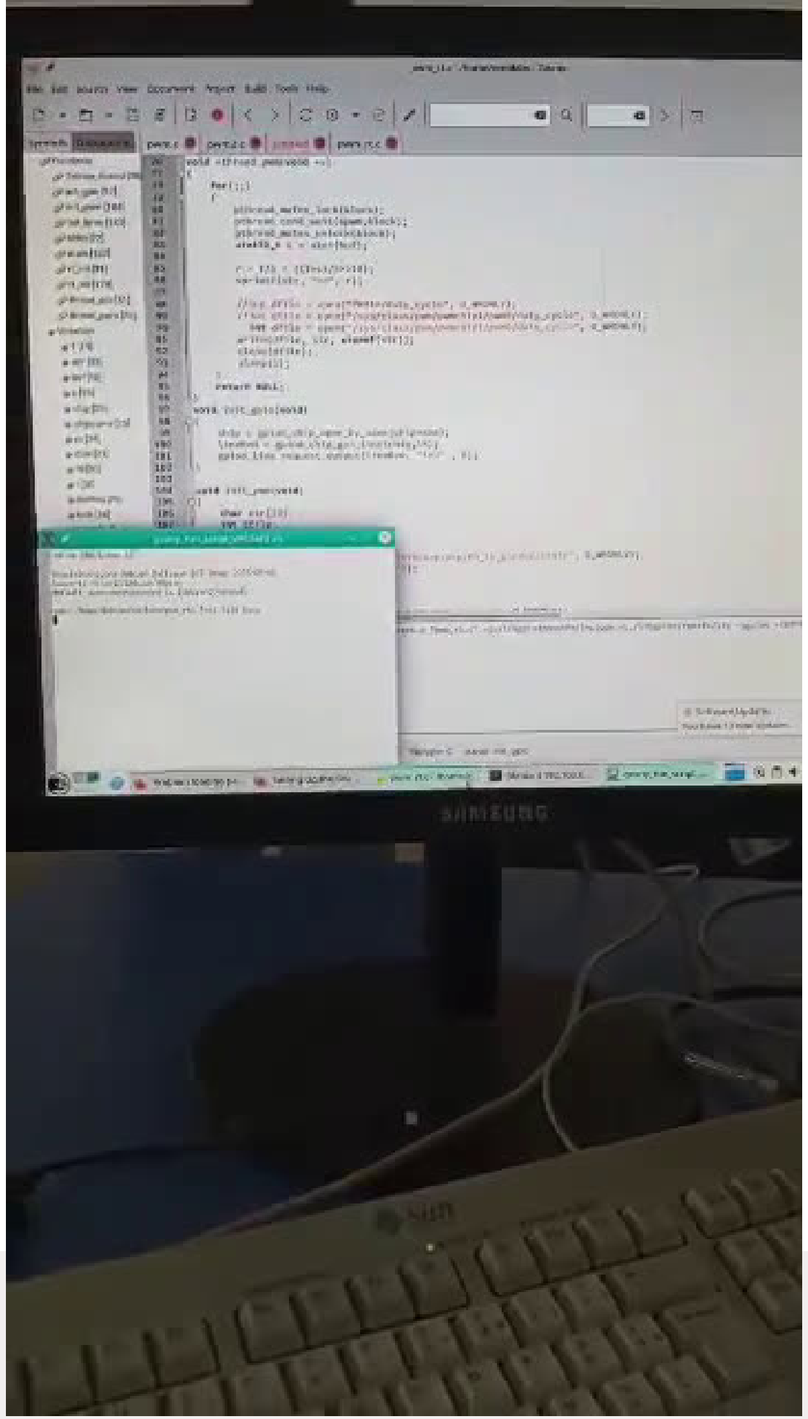
Programme principal (main)

La fonction main coordonne le projet en initialisant les threads ADC et PWM pour des opérations asynchrones en temps réel. Elle gère la base de données SQLite, ouvrant la connexion vers ma_base_de_donnees.db et créant la table adc_data si nécessaire. Les valeurs ADC sont insérées dans la table via save_to_database. En cas d'erreur, un message s'affiche, assurant une gestion robuste des données persistantes. Ainsi, la fonction main garantit une synchronisation entre threads et une intégration réussie de la base de données, adaptée à un environnement temps réel.

Test et Réalisation

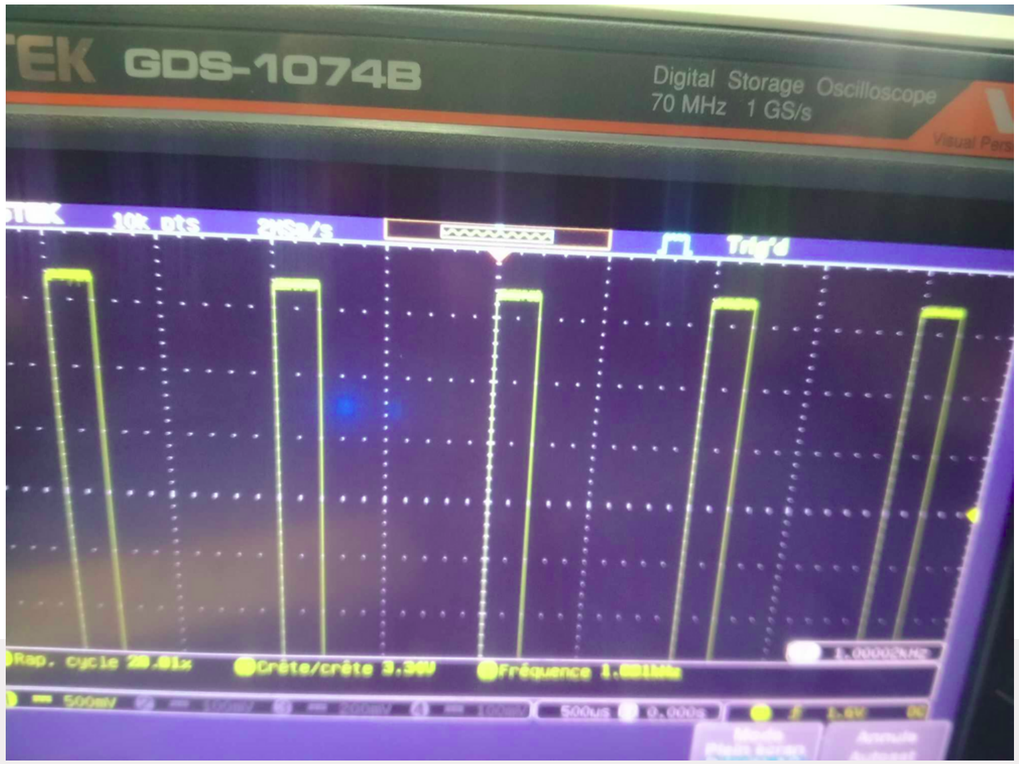


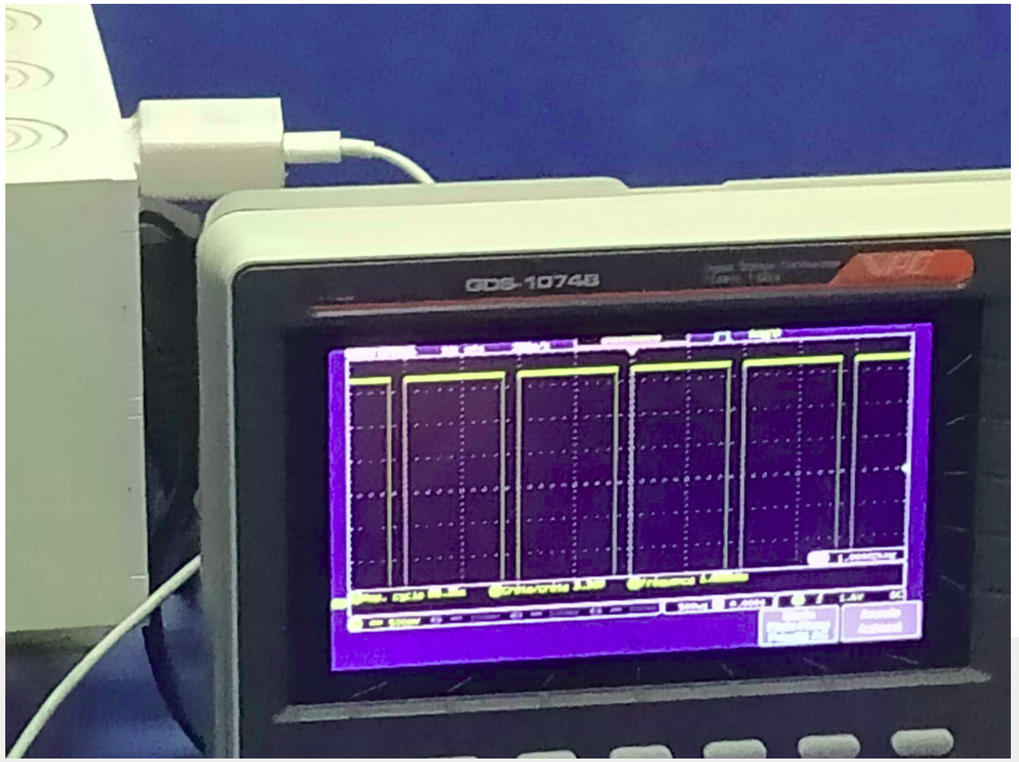
Test et Réalisation



```
File  Edit  View  Bookmarks  Settings  Help
debian@BeagleBone:/sys/bus/iio/devi
debian@BeagleBone:/sys/class/pwm/pw
total 0
-r--r--r-- 1 root gpio 4096 Jan  1
-rw-rw-r-- 1 root gpio 4096 Jan  1
-rw-rw-r-- 1 root gpio 4096 Jan  1
-rw-rw-r-- 1 root gpio 4096 Jan  1
-rw-rw-r-- 1 root gpio 4096 Jan  1
drwxrwxr-x 2 root gpio    0 Jan  1
-rw-rw-r-- 1 root gpio 4096 Jan  1
debian@BeagleBone:/sys/class/pwm/pw
debian@BeagleBone:~$ cd emblabs/
debian@BeagleBone:~/emblabs$ ls -l
total 88
-rwxr-xr-x 1 debian debian 89976 De
debian@BeagleBone:~/emblabs$ ./pwm
value : 1003
value : 1003
value : 1003
value : 1002
value : 1003
value : 1003
value : 1003
value : 1003
alue : 1003
alue : 1003
alue : 1003
alue : 1003
```

Observation du Signal PWM sur l'Oscilloscope





Conclusion

Les systèmes temps réels jouent un rôle essentiel dans divers secteurs industriels, garantissant des niveaux optimaux de performance, de sécurité et de fiabilité. Leur fonctionnement repose sur des principes fondamentaux tels que le déterminisme, la priorisation, l'ordonnancement et la gestion des ressources. Malgré les défis auxquels ils font face, les récents progrès technologiques renforcent la capacité des systèmes temps réels à répondre de manière efficace aux exigences de plus en plus complexes de notre société interconnectée.

Merci de votre attention !