Laboratoire n° 5 : Traitement des données

Robin Forestier - Benoît Delay

Table des matières

1. Introduction

L'objectif de ce laboratoire est de développer un système qui, utilisant la plateforme DE1-SoC, est capable de capter et de traiter des données audio et vidéo afin d'en extraire des informations utiles.

Nous examinerons et analyserons également les différents temps d'exécutions des tâches afin d'étudier et d'optimiser les performances temps réel et l'implémentation de celles-ci.

2. Description du système

Le système est composé de deux tâches principales : une tâche audio et une tâche vidéo.

La tâche audio à pour but de récupérer un flux audio via un micro et de le copier sur la sortie audio (casque) de la carte. Elle est composée de trois sous tâche.

- Aquisition du flux audio
- Traitement du flux audio (FFT)
- Affichage du temps d'exécution et de la fréquence dominante de la FFT

La tâche vidéo à pour but de récupérer un flux vidéo d'un fichier et de l'afficher à l'écran via la sortie VGA. Elle est aussi est composée de deux sous-tâches.

- Conversion en niveaux de gris
- Convolution (detection de contour)

2.1. Entrées / Sorties

Pour la partie audio, le micro doit être branché sur l'entrée audio de la carte (ROSE). La sortie audio est disponible sur la prise jack de la carte (VERT).

Pour activer ou désactier une tâche, il suffit d'actionner un switch sur la DE-1.

	Convolution	**Greyscale**	**Video task**	**Audio task**
Switch	9	8	1	0

Et le bouton KEY0 permet d'arreter proprement le programme.

3. Caractérisation des tâches

Premièrement, nous avons mesuré le temps d'exécution de chaque tache séparément. Cela nous permet d'avoir une idée du temps d'exécution de chaque tâche.

Pour cela nous avons utilisé les timer Xenomai, avec la fonction `rt_timer_read()`. Nous avons effectué ~ 50 mesures pour chaque tâche, et avons calculé la moyenne, la variance et l'écart type.

4. Tâche ioctl

La tâche ioctl à pour but de gérer les actions des switches et du bouton KEY0. Elle permet d'activer ou de désactiver les tâches audio et vidéo.

4.1. Mesures

Voici les temps d'exécution de la tâche ioctl. La mesure à été effectuée sur un système non chargé et seul la tâche ioctl était active.

La tache s'exécute périodiquement toutes les 1/10s (10Hz).

```
-------------------------summary1.c---------------------------
  Total of 46 values
    Minimum  = 0.309000 (position = 42)
    Maximum  = 0.589000 (position = 3)
    Sum      = 20.368000
    Mean     = 0.442783
    Variance = 0.008302
    Std Dev  = 0.091117
    CoV      = 0.205784
--------------------------------------------------------------
```

Cette tâche est très rapide, avec un temps d'exécution de ~XXX ms. Elle ne devrait pas poser de problème pour l'ordonnancement des autres tâches.

5. Tâche audio

La tâche audio à pour but de récupérer un flux audio via un micro et de le copier sur la sortie audio (casque) de la carte. Le taux d'échantillonage est de 48kHz.

Elle est composée de trois sous tâche : l'aquisition du flux audio, le traitement du flux audio (FFT) et l'affichage du temps d'exécution et de la fréquence dominante de la FFT.

La tâche audio s'exécute périodiquement toutes les 5,1 ms.

5.1. Mesures

Voici les temps d'exécution de chaque tâche. La mesure à été effectuée sur un système non chargé et seul la tâche audio et ioctl était active.

5.1.1. Aquisition

```
-------------------------summary1.c--------------------------
Total of 56 values
    Minimum  = 0.153250 (position = 0)
    Maximum  = 14.412900 (position = 36)
    Sum      = 165.210700
    Mean     = 2.950191
    Variance = 7.776338
    Std Dev  = 2.788609
    CoV      = 0.945230
-------------------------------------------------------------
```

On observe une grosse difference entre le min et le max mais ceci est facilement explicable par notre façon de gérer le buffer. Nous lisons les données envoyée depuis le microphone et nous les envoyons directement a la tâche de traitement qui elle va s'occuper d'agréger les données ensembles. Une fois que la tache traitement aura suffisemment de données elle pourra commencer le traitement. Pendant le traitement elle ne pourra pas lire tout de suite les messages envoyé par la tâche d'acquisition. Cela va donc créer du délai sur la methode `rt_queue_send`

5.1.2. Traitement

```
-------------------------summary1.c--------------------------
Total of 44 values
    Minimum  = 14.102360 (position = 7)
    Maximum  = 14.947300 (position = 23)
    Sum      = 640.400260
    Mean     = 14.554551
    Variance = 0.083397
    Std Dev  = 0.288785
    CoV      = 0.019842
-------------------------------------------------------------
```

Le traitement des données est assez couteux en temps. Le calcule de la FFT est une opération demandant beaucoup de calculs. Nous sommes en dessous de la limite de 5.1ms pour une lecture à 48kHz.

5.1.3. Affichage

```
-------------------------summary1.c--------------------------
  Total of 45 values
```

```
    Minimum  = 0.205500 (position = 39)
    Maximum  = 0.376600 (position = 25)
    Sum      = 12.995200
    Mean     = 0.288782
    Variance = 0.001424
    Std Dev  = 0.037731
    CoV      = 0.130654
---------------------------------------------------------------
```

Sans surprise, l'affichage des données est très rapide.

6. Tâche vidéo

La tâche vidéo à pour but de récupérer un flux vidéo d'un fichier et de l'afficher à l'écran via la sortie VGA.

Elle est aussi est composée de deux sous-tâches : la conversion en niveaux de gris et la convolution (detection de contour).

6.1. Mesures

Voici les temps d'exécution de chaque tâche. La mesure à été effectuée sur un système non chargé et seul la tâche vidéo et ioctl était active.

6.1.1. Vidéo

```
-------------------------summary1.c--------------------------
  Total of 44 values
    Minimum  = 17.475480 (position = 4)
    Maximum  = 17.596350 (position = 14)
    Sum      = 770.233930
    Mean     = 17.505317
    Variance = 0.000683
    Std Dev  = 0.026126
    CoV      = 0.001492
---------------------------------------------------------------
```

En sachant que la vidéo doit être affichée à 15fps, nous avons un temps d'exécution de ~ 66ms pour chaque image. Nous sommes donc en dessous de la limite de 66.67ms pour afficher une image à 15fps.

On remarque que le temps d'exécution est très stable, avec un écart type de ~ 0.026ms.

6.1.2. Greyscale

```
-------------------------summary1.c--------------------------
  Total of 44 values
```

```
    Minimum  = 27.527860 (position = 25)
    Maximum  = 27.737950 (position = 0)
    Sum      = 1215.155190
    Mean     = 27.617163
    Variance = 0.001920
    Std Dev  = 0.043822
    CoV      = 0.001587
---------------------------------------------------------------
```

Le traitement en niveaux de gris n'est pas très gourmand en temps. Nous sommes aussi en dessous de la limite de 66.67ms pour afficher une image à 15fps.

En comparaison avec la tâche vidéo, la variation du temps d'exécution est l'égérement plus élevé, avec un écart type de ~ 0.043ms (~ 1.5 fois plus élevé).

6.1.3. Convolution

```
-------------------------summary1.c--------------------------

  Total of 44 values
    Minimum  = 110.237930 (position = 15)
    Maximum  = 111.755900 (position = 12)
    Sum      = 4872.175690
    Mean     = 110.731266
    Variance = 0.093900
    Std Dev  = 0.306431
    CoV      = 0.002767
---------------------------------------------------------------
```

La convolution est la tâche la plus gourmande en temps. Nous sommes en dessous de la limite de 66.67ms pour afficher une image à 15fps. Il ne serra pas possible d'ordonnancer cette tâche avec les autres tâches.

Un temps d'exécution si élevé est parfaitement compréhensible, car la convolution est une opération demandant plusieur calculs pour chaque pixel de l'image.

6.2. Mesure de la tâche d'aquisition avec une seconde tâche active

Comme demandé à l'étape 2 du laboratoire « *Il est important que la tâche d'acquisition ne soit jamais polluée (...) cette tâche récupère toujours les données à la fréquence souhaitée (15HZ).* »

Pour vérifier cela, nous avons mesuré le temps d'exécution de la tâche d'aquisition avec une seconde tâche active.

Comme comparatif nous prendrons la mesure ci dessu de la tâche d'aquisition seule.

6.2.1. Aquisition et greyscale

```
-------------------------summary1.c--------------------------
  Total of 44 values
    Minimum  = 23.103180 (position = 32)
    Maximum  = 23.967100 (position = 27)
    Sum      = 1026.348170
    Mean     = 23.326095
    Variance = 0.100914
    Std Dev  = 0.317669
    CoV      = 0.013619
-------------------------------------------------------------
```

Tâches	Temp d'exécution
Aquisition	17.5
Greyscale	27.6
Aquisition + Greyscale	23.3

Ces trois mesures nous permettent de constater que la tâche d'aquisition est légérement protégée des autres tâches. En effet, le temps d'exécution de la tâche d'aquisition augmente de ~ 5ms lorsque la tâche greyscale est active. Mais s'exécute plus rapidement que la tâche greyscale seule.

6.2.2. Aquisition et convolution

```
-------------------------summary1.c--------------------------
  Total of 44 values
    Minimum  = 11.847850 (position = 31)
    Maximum  = 23.370770 (position = 36)
    Sum      = 694.882030
    Mean     = 15.792773
    Variance = 28.779574
    Std Dev  = 5.364660
    CoV      = 0.339691
-------------------------------------------------------------
```

On peut observer un patern au sein des mesures.

Mesures	Temp d'exécution
1	23.238460
2	11.848320
3	11.952310
4	23.229040
5	11.955930
6	11.961000

Mesures	Temp d'exécution
7	23.242710
8	11.857240
9	11.979860
10	23.236110

On peut voir que le temps d'exécution fluctue entre ~ 11ms et ~ 23ms. Je ne saurai expliquer pourquoi cette fluctuation est présente.

Tâches	Temp d'exécution
Aquisition	17.5
Convolution	110.7
Aquisition + Convolution	15.8

On observe que le temps d'exécution de la tâche d'aquisition avec la convolution est le plus faible. Cela ne devrait pas être le cas, car la convolution est la tâche la plus gourmande en temps.

Mais si l'on regarde les valeurs maximum, environ 23.2 ms, on observe quand même que la tâche d'aquisition est protégée des autres tâches.

La tâche d'aquisition ne s'exécute plus à la mème vitesse, son temps d'exécution augmente de ~30% lorsqu'elle est en concurrence avec la tâche de convolution. Mais elle reste en dessous de la limite de 66.67ms pour une lecture à 15fps.

7. Ordonnancement des tâches

Pour que notre ensemble de tâches soit ordonnançable avec l'algorithme Rate Monotonic, nous devons respecter la condition d'ordonnançabilité proposée par Liu et Layland.

Une condition suffisante d'ordonnançabilité proposée par Liu et Layland est :

$$\sum_{i=1}^{n} \frac{C_i}{P_i} \leq U_{\text{lub}_{\text{RM}}}(n) = n\left(2^{\frac{1}{2}} - 1\right)$$

Avec `C_i` le temps d'exécution de la tâche i, `P_i` la période de la tâche i et `U_lub_RM (n)` la borne supérieure de l'utilisation du processeur pour `n` tâches.

7.1. calculs

Reprenons les temps d'exécution de chaque tâche.

Tâches	Temp d'exécution
IOctl	0.44
Aquisition audio	2.95
Traitement	14.6
Affichage	0.3
Aquisition vidéo	17.5
Greyscale	27.6
Convolution	110.7

Et voici la période de chaque tâche.

- audio : 5.1 ms
- video : 66,6 ms
- ioctl : 100 ms

Testons si notre tâche audio est ordonnançable avec l'algorithme Rate Monotonic.

$$U_{\text{lub}_{\text{RM}}}(2) = 2\left(2^{\frac{1}{2}} - 1\right) = 0.82$$

$$U = \frac{0.44}{100} + \frac{2.95 + 14.6 + 0.3}{5.1} = 3.5$$

La tâche audio n'est pas ordonnançable avec l'algorithme Rate Monotonic.

Testons si notre tâche vidéo est ordonnançable avec l'algorithme Rate Monotonic.

$$U_{\text{lub}_{\text{RM}}}(2) = 2\left(2^{\frac{1}{2}} - 1\right) = 0.82$$

Traitement uniquement :

$$U = \frac{0.44}{100} + \frac{17.5}{66.6} = 0.26$$

Traitement et greyscale :

$$U = \frac{0.44}{100} + \frac{17.5 + 27.6}{66.6} = 0.68$$

Traitement et convolution :

$$U = \frac{0.44}{100} + \frac{17.5 + 110.7}{66.6} = 1.92$$

La tâche vidéo est ordonnançable avec l'algorithme Rate Monotonic si elle est composée uniquement de la tâche de traitement ou de la tache greyscale.

Pour que la tâche vidéo soit ordonnançable avec l'algorithme Rate Monotonic, il faudrait diminuer le framerate vidéo pour obtenir un système fonctionnel. Avec la tâche de convolution, le framerate maximal auquel notre système continue de fonctionner correctement est de ~ 6.3 fps.

Testons si notre ensemble de tâches est ordonnançable avec l'algorithme Rate Monotonic.

Calculons la borne supérieure de l'utilisation du processeur pour 3 tâches.

$$U_{\text{lub}_{\text{RM}}}(3) = 3\left(2^{\frac{1}{2}} - 1\right) = 0.78$$

Calculons l'utilisation du processeur pour 6 tâches.

$$U = \frac{0.44}{100} + \frac{2.95 + 14.6 + 0.3}{5.1} + \frac{17.5}{66.6} = 3.77$$

Nous avons donc `U = 3.77` et `U_lub_RM (6) = 0.78` . Notre ensemble de tâches n'est pas ordonnançable avec l'algorithme Rate Monotonic. Et cela sans aucune modification de la vidéo.

Notre système n'est pas capable de gérer les différents traitements vidéo de manière ordonnée.

Il faudrait diminuer le framerate vidéo pour obtenir un système fonctionnel. Mais si l'on calcule le framerate maximal auquel notre système continue de fonctionner correctement, on trouve :

$$U = \frac{0.44}{100} + \frac{2.95 + 14.6 + 0.3}{5.1} + \frac{17.5}{x} = 2.49$$

Mais ce système n'est pas solvable avec un x positif. Nous ne pouvons pas calculer le framerate maximal auquel notre système continue de fonctionner correctement.

8. Conclusion

Nous pouvons voir qu'il est impossible de respecter 15 fps pour la video. Les algorithmes de traitement que nous utilisons sont lourd et peu optimisé. L'utilisation d'un tel processeur n'est pas le plus optimal. Pour respecter la contrainte ainsi que d'avoir plus d'image par seconde il pourrait être plus efficace d'utiliser une FPGA ou un hardware spécialisé dans ce genre de traitement. Pareil pour l'audio, un dsp serait plus

pertinent car il a un processeur spécialisé pour ce genre de tâche et nous permettrait de meilleur performance.

Nous pourrions aussi implémenter tout les traitement en assembleur afin d'optimiser le calcul mais cela au dela des limites du cours.

Si nous avions mieux gérer notre temps il aurait été pertinent de mesurer avec un oscilloscope le réveil des taches non periodique tel que le traitement ainsi que le logging ! Malheureusement nous n'avons pas eu le temps de mettre en place de tel procédé !