Informatique embarquée

J.-M Friedt

FEMTO-ST/département temps-fréquence

jmfriedt@femto-st.fr

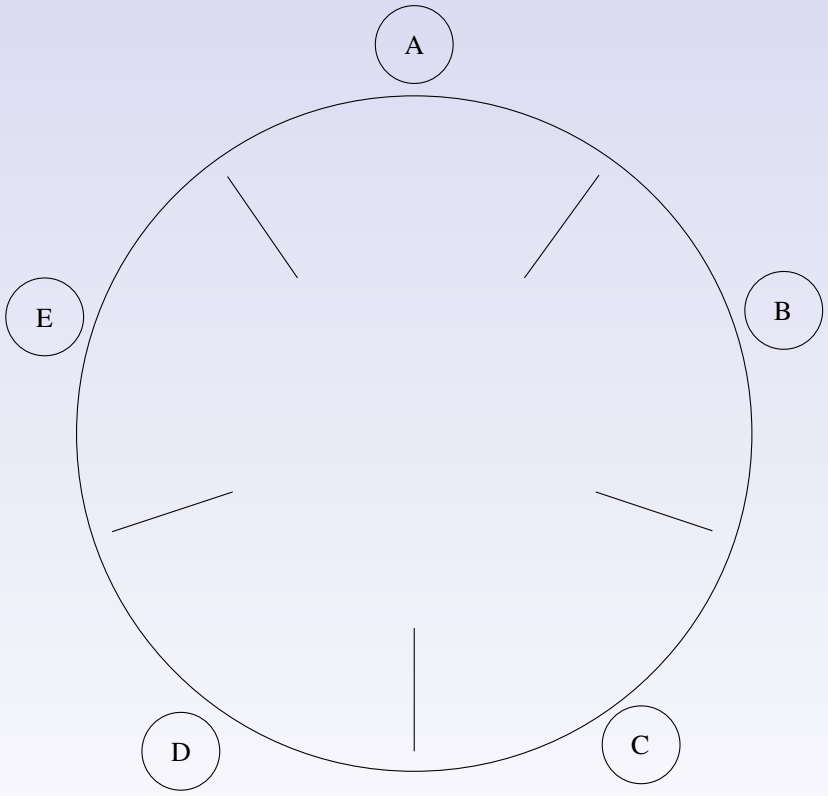
transparents à `jmfriedt.free.fr`

9 mars 2017

Problème des philosophes

Programmation par **contrainte** : donner les transitions autorisées et l'ordonnanceur se charge de trouver la solution

- N philosophes sont assis autour d'une table,
- chaque philosophe a une baguette à sa droite et une baguette à sa gauche,
- chaque philosophe doit se saisir de deux baguettes pour pouvoir manger,
- une fois qu'un philosophe a mangé, il repose les baguettes qu'ils a prises.
- Deux philosophes ne peuvent pas utiliser une même baguette en même temps.



Exprimer ce problème dans FreeRTOS afin qu'il trouve la solution.

Problème des philosophes

- comment représenter un philosophe ?
- comment représenter une baguette ?
- comment se sortir du cas où tous les philosophes ont pris la baguette à leur droite ?

Exemple de solution :

```
eabcdzEwBy4v1xDAoc30xC2
```

1. tout le monde veut se jeter sur les baguettes : eabcd,
2. z prend une baguette, puis E prend 2 baguettes
3. w diamètralement opposé prend une baguette, puis B deux baguettes
4. y prend une baguette
5. 4 a fini de manger : e pose ses baguettes
6. v prend une baguette
7. 1 a fini de manger et pose ses baguettes
8. x prend une baguette, D & A ont fini de manger et posent leurs baguettes
9. o se fait refuser sa 2ème baguette, il repose la 1ère
10. c recommence à prendre 1 baguette, ...

Problème des philosophes

```
$ qemu-system-arm -M stm32-p103 -serial stdio -kernel output/main.bin

eabcdEBung
wng
4D1A03C2
```

qemu supportant le STM32 :
https://beckus.github.io/qemu_stm32/

Voir `hw/arm` : plateforme `stm32_p103.c`

Ajouter un port série :

```
DeviceState *uart3 = DEVICE(object_resolve_path("/machine/stm32/uart[3]", NULL));
assert(uart3);
stm32_uart_connect((Stm32Uart *)uart3, serial_hds[0], STM32_USART3_NO_REMAP);
```

Problème des philosophes

Messages de l'émulateur en cas d'utilisation d'un périphérique non-initialisé

```
USART_InitTypeDef USART_InitStructure;
...
USART_Init(USART2, &USART_InitStructure);
USART_Cmd(USART2, ENABLE);
put_char(USART1,'0');
put_char(USART2,'0');
```

On utilise le port série 1 qui n'a jamais été initialisé ⇒ pas de source d'horloge ni de configuration

```
$ qemu_stm32/arm-softmmu/qemu-system-arm -M stm32-p103 -serial stdio -serial stdio -serial stdio -kernel main.bin

qemu stm32: hardware warning: Warning: You are attempting to use the UART1 peripheral while its clock is disabled.

R00=40013800 R01=00000030 R02=008e0001 R03=00000030
R04=20004fe8 R05=08000b1c R06=00000000 R07=20004fc0
R08=00000000 R09=00000000 R10=00000000 R11=00000000
R12=0000000f R13=20004fc0 R14=08000519 R15=08000808
PSR=20000173 --C- T svc32
qemu: hardware error: Attempted to write to USART_DR while UART was disabled.
CPU #0:
R00=40013800 R01=00000030 R02=008e0001 R03=00000030
R04=20004fe8 R05=08000b1c R06=00000000 R07=20004fc0
R08=00000000 R09=00000000 R10=00000000 R11=00000000
R12=0000000f R13=20004fc0 R14=08000519 R15=08000808
PSR=20000173 --C- T svc32
FPSCR: 00000000
Aborted
```

Conclusion

1. un émulateur permet de travailler en l'absence de plateforme matérielle
2. un émulateur permet de connaître l'état interne du processeur et de prévenir l'utilisateur d'erreurs ...
3. ... sous réserve que les périphériques soient émulés.

ADC, DAC, timer, GPIO, USART fonctionnels pour STM32

Eclipse intègre une émulation du STM32F4 :
`http://gnuarmeclipse.github.io/qemu/` (STM32F4-Discovery)

Analyse de l'exécution de l'émulateur

- Fonctions de callback : un évènement se traduit par l'appel des fonctions implémentant le périphérique
- $\Rightarrow$ séquence de fonctions difficile à suivre en l'absence d'exécution séquentielle
- `valgrind --tool=callgrind -v \ --dump-every-bb=10000000 \ ../qemu-system-arm \ -M stm32-p103 \ -serial stdio \ -kernel temperature/main.bin`
- `kcachegrind callgrind.out.8964` affiche le graphique des appels de fonctions et les ressources requises.