Faculté des Sciences & Techniques — Université de Limoges

Master 2nde année

TMC

Assembleur ARM

Utilisation d'un simulateur de SoC : renode

Objectifs

- ☐ se familiariser avec gdb et l'exécution de code sur processeur ;
- ☐ disposer d'un environnement simulé pour l'étude de firmware ;

Débogage niveau processeur

Soit le programme suivant :

```
#include <stdio.h>

int main()
{
  printf("Hello world\n");
}
```

1 – Question

a. Trouvez les adresses de l'appel système lié à « write » et dumpez le contenu des registres pour retrouver la chaîne `"Hello world"`.

Installation de renode

2 – Étudiez la proposition de Renode `https://renode.io/`

a. Comment peut-on suivre l'exécution d'un programme sous Linux ?
b. Quelles différences avec l'embarqué ?
c. Qu'est-ce que la « cross-compilation » ?
d. Comment procéder quand il n'y a pas d'OS ?
e. Quel est l'intérêt de simuler un processeur ?
f. Quelle est l'originalité de Renode ?
g. Comparez Qemu et Renode ?

Vous trouverez à `https://git.p-fb.net/PeFClic/docker_rp2040` un dockerfile pour installer :

- ▷ le simulateur Renode ;
- ▷ un émulateur de la carte de développement Raspberry Pico avec ses principaux périphériques.

Simulation de firmware bare-metal

3 – Installation du Pico-sdk :

xterm
```
$ git clone https://github.com/raspberrypi/pico-sdk.git
$ cd pico-sdk
$ git submodule update --init
$ export PICO_SDK_PATH=~/pico-sdk
```

Installation de PicoTool :

xterm
```
$ git clone https://github.com/raspberrypi/picotool.git
$ cmake -B build .
$ cd build
$ make
```

Installation de firmware bare-metal :

xterm
```
$ git clone https://github.com/carlosftm/RPi-Pico-Baremetal.git
$ cd RPi-Pico-Baremetal
$ ls
$ cd 00_asm_blink
$ make
```

Exécutez et étudiez les firmwares proposés sous renode combiné avec gdb.

Pour lancer renode à l'aide du container docker :

1. dans le répertoire où vous avez compilé le firmware :

```
xterm
$ docker run -it --rm --name renode -p 3333:3333 -v $.:/BUILD renode-rp2040
```

   On partage dans le container l'accès au répertoire courant, qui sera monté dans le container sur `/BUILD`.

Les commandes à utiliser dans renode pour l'exécution d'un firmware :

```
xterm
include @boards/initialize_raspberry_pico.resc
sysbus LoadELF @/BUILD/asm-blink.elf
cpu0 PC 0x20040009
cpu1 PC 0x20040009
machine StartGdbServer 3333
logLevel -1 sysbus.gpio
```

Ensuite dans gdb :

```
xterm
target extended-remote :3333
```

Si on lance le firmware pour s'exécuter en continue, vous devriez observer le « clignotement » de la LED (GPIO25) dans renode :

```
xterm
11:11:21.1685 [NOISY] gpio: Setting GPIO25 to: True, time: 00:05:34.399600000,
from: SIO
11:11:21.1685 [NOISY] gpio: Setting GPIO25 to: False, time: 00:05:34.399600000,
from: SIO
11:11:21.1709 [NOISY] gpio: Setting GPIO25 to: True, time: 00:05:34.402000000,
from: SIO
11:11:21.1709 [NOISY] gpio: Setting GPIO25 to: False, time: 00:05:34.402000000,
from: SIO
11:11:21.1733 [NOISY] gpio: Setting GPIO25 to: True, time: 00:05:34.404400000,
from: SIO
11:11:21.1733 [NOISY] gpio: Setting GPIO25 to: False, time: 00:05:34.404400000,
from: SIO
11:11:21.1757 [NOISY] gpio: Setting GPIO25 to: True, time: 00:05:34.406800000,
from: SIO
11:11:21.1757 [NOISY] gpio: Setting GPIO25 to: False, time: 00:05:34.406800000,
from: SIO
11:11:21.1781 [NOISY] gpio: Setting GPIO25 to: True, time: 00:05:34.409200000,
from: SIO
```