IUT "A" Paul SABATIER
Dpt Informatique

Langage d'assemblage

TP : utilitaire de mise au point gdb

---

1. Appel de **gdb** (à ne pas faire si on utilise **arm-elf-gdb**)

gdb prog ↵ (**prog** est le nom du fichier éxécutable)
ou
gdb↵
(gdb) **file prog**↵

2. Arrêt de **gdb**

quit ↵ ou **q** ↵ (retour au shell)

3. Pose d'un point d'arrêt

a) sur une ligne dans le fichier source:

break nulig ↵ ou **b nulig** ↵ (**nulig** est un numéro de ligne)

b) sur un sous-programme dans le fichier source:

b fonction ↵ (**fonction** est le nom du sous-programme)

c) sur une intruction machine
b *addr ↵ (**addr** est l'adresse de l'instruction)

4. Affichage des points d'arrêts existants

info break ↵

5. Suppression d'un point d'arrêt existant

clear nulig ↵
ou **clear fonction** ↵
ou **delete n** ↵ (**n** est le numéro du point d'arrêt)

6. Exécution du programme **prog** sous le contrôle de **gdb**

run ↵ ou **r** ↵

L'exécution du programme démarre et s'arrête sur un point d'arrêt (si un ou plusieurs points d'arrêts ont été positionnés au préalable et aux bons endroits !!!)
Il est alors possible de visualiser le contenu de variables, de registres du processeur, de portions de la mémoire ou de désassembler une séquence d'instructions machine, puis de continuer l'exécution du programma à partir des commandes décrites ci-dessous.

TSVP

7. Affichages

a) contenu d'une variable :

p /f variable ↵

f définit le format d'impression et peut prendre les valeurs suivantes :
- f → flottant décimal
- u → décimal non signé
- d → décimal signé
- x → hexadécimal
- o → octal
- t → binaire
- c → caractère

b) contenu de registres :

info reg ↵ (affiche le contenu de tous les registres du processeur)
info reg liste ↵ (affiche le contenu des registres énumérés dans **liste**)

c) désassemblage d'instructions machine

disassem etiq↵

Cette commande désassemble le code machine à partir de l'étiquette **etiq**.

8. Reprise de l'exécution du programme

a) jusqu'à un autre point d'arrêt :

continue ↵ ou **c ↵**

b) jusqu'à la ligne suivante :

step ↵ ou **s ↵**
next ↵ ou **n ↵**

Si la ligne courante contient un appel de sous-programme, la commande **step** va exécuter le sous-programme entièrement et arrêter l'exécution sur la ligne suivante, alors que la commande **next** va arrêter l'exécution sur la première ligne du sous-programme.

c) jusqu'à l'instruction machine suivante :

stepi ↵ ou **si ↵**
nexti ↵ ou **ni ↵**

Même différence de fonctionnement que pour b).

9 Affichage de la pile des appels de sous-programmes

backtrace ↵ ou **bt ↵**

Cette commande affiche la séquence des appels de sous-programmes, à partir du programme principal, jusqu'au sous-programme dans lequel l'exécution à été interrompue.