岸 哲夫

GDBコマンドのまとめ（その1）

今回と次回は，今までの連載のまとめとして，GNUデバッガGDBのコマンドの一覧を2回に分けて説明します．（筆者）

1. メモリの調査について

● メモリ調査xコマンド

examineの省略形であるxコマンドを使用すると，メモリの調査が可能です．パラメータには，調査するメモリの大きさ，その領域をどのように表示するのかということを記述します．

次に示す文字で，調査するメモリ領域のサイズを指定します．

- b：バイトで調査
- h：半ワードで調査
- w：ワードで調査．GDBが動くマシンで「ワード」の概念は4バイトです．したがって，このサイズは32ビットです．
- g：巨大ワード（8バイト）で調査．

次に示す文字で，メモリの内容を表示する方法を指定します．

- x：符号なし16進整数値で表示
- d：符号付き10進整数値で表示
- u：符号なし10進整数値で表示
- o：符号なし8進整数値で表示
- a：絶対アドレス，またはシンボルに定義された相対アドレスで表示
- c：文字定数で表示
- f：浮動小数点で表示．これはwとgのサイズでのみ有効です．
- s：nullターミネートされた文字列として表示．指定されたユニット・サイズは無視され，代わりにnullキャラクタまで達する多くのバイト列を持つことになります．
- i：命令語を表示．指定されたユニット・サイズは無視され，命令語を構成するバイト数は，そのマシンで利用されているオペコードとアドレッシング・モードの種類に依存します．

リスト1 検証用プログラムのソース・リスト（test1.c）

```
#include <stdio.h>
#include <string.h>
#include <limits.h>
#include <float.h>
int func1(void)
{
        int a;
        a++;
        return(10);
}
int main(void)
{
        int a=0;
        int b=-1;
        char c='d';
        long d=LONG_MAX;
        long long int e=LLONG_MAX;;
        char *f="test";
        float g=DECIMAL_DIG;
        int (*p) (void);
        p=func1;
        printf("a banchi__%p¥n",&a);
        printf("b banchi__%p¥n",&b);
        printf("c banchi__%p¥n",&c);
        printf("d banchi__%p¥n",&d);
        printf("e banchi__%p¥n",&e);
        printf("g banchi__%p¥n",&g);
        printf("f banchi__%p¥n",&f);
        printf("P banchi__%p¥n",&p);
        return  0;
}
```

● xコマンドの動作確認

次に，メモリ検証用のプログラムのソースを**リスト1**に示します．

```
gcc test1.c -g -o test1  -std=gnu99
```

上記のコマンドでC99規格のオプションを付けてコンパイルします．もちろん，GDBを使うための-gも忘れてはいけません．

それぞれの変数のアドレスを表示してGDBで使用します．関数func1()のアドレスも*pにセットします．

各変数のアドレスとxコマンドでの表示結果は，**図1**に示すとおりです．

図1の①に示すようなコマンドを打鍵すると，変数a，bをバイト表現で符号付き10進整数値，符号なし10進整数値で表示します．

また，**図1**の②に示すようなコマンドを打鍵すると，変数bを半ワード，ワード表現で符号なし10進整数値で表示します．この場合，意味のない値が表示されています．

図1の③に示すようなコマンドで文字定数を表示することが可能です．

ここで，GDBを再度起動します．メモリの状況により，**図2**のように各変数のアドレスが変わっています．このときのxコマンドの実行結果を**図2**の①に示します．

KEYWORD ── GDB，メモリ調査，スタック，コア・ファイル