## 实验2 数据表示和运算实验

1. 实验目的

1了解并学习计算机的数据表示方式，了解并学习计算机的算术运算方式，理解不同数据类型的运算属性。

2 了解并学习gdb的使用方法，并运用其进行内存、寄存器检查。

1. 实验内容

1、在32位计算机中运行一个C语言程序，在该程序中出现了以下变量的初值，请在表格中填写它们对应的机器数（用十六进制表示）。在gdb里面可使用x/1xw 查看int/unsigned/float的机器数，使用x/1xh 查看short/unsigned short的机器数，使用x/1xb查看char的机器数，使用x/1xg查看double的机器数：

（1）int x=-32768 （2）short y=522 （3）unsigned z=65530 （4）char c=’@’

（5）float a=-1. 1 （6）double b=10.5 （7）float u = 123456.789e4 （8）double v= 123456.789e4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 变量 | x | y | z | c |
| 机器数 |  |  |  |  |
| 变量 | a | b | u | v |
| 机器数 |  |  |  |  |

运行下面的代码验证输出是否与gdb查看的结果一致：

#include <stdio.h>

int main() {

int x = -32768;

short y = 522;

unsigned z = 65530;

char c = '@';

float a = -1.1;

double b = 10.5;

float u = 123456.789e4;

double v = 123456.789e4;

printf("++++++++++Machine value++++++++++++++++\n");

printf("x = 0x%x\n", x);

printf("y = 0x%hx\n", y);

printf("z = 0x%x\n", z);

printf("c = 0x%hhx\n", c);

printf("a = 0x%x\n", \*(unsigned \*)&a);

printf("b = 0x%llx\n",\*(unsigned long long \*)&b);

printf("u = 0x%x\n", \*(unsigned \*)&u);

printf("v = 0x%llx\n",\*(unsigned long long \*)&v);

printf("++++++++++Real value+++++++++++++++++++\n");

printf("x = %d\n", x);

printf("y = %hd\n", y);

printf("z = %u\n", z);

printf("c = %c\n", c);

printf("a = %f\n", a);

printf("b = %f\n",b);

printf("u = %f\n", u);

printf("v = %f\n",v);

}

2、使用命令gcc –ggdb swap.c –o swap编译下面的swap.c代码，完成后面的实验

void xor\_swap(int \*x, int \*y){

\*y=\*x ^ \*y; /\* 第一步 \*/

\*x=\*x ^ \*y; /\* 第二步 \*/

\*y=\*x ^ \*y; /\* 第三步 \*/

}

int main() {

int a = 1;

int b = 2;

xor\_swap(&a, &b);

}

1. 使用gdb命令查看程序变量的取值，填写下面两个表格：

|  |  |  |  |
| --- | --- | --- | --- |
| a的存放地址(&a) | b的存放地址(&b) | x的存放地址(&x) | y的存放地(&y) |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 执行步数 | x的值(机器值，用十六进制) | y的值(机器值，用十六进制) | \*x的值(程序中的真值，用十进制) | \*y的值(程序中的真值，用十进制) |
| 第一步前 |  |  |  |  |
| 第一步后 |  |  |  |  |
| 第二步后 |  |  |  |  |
| 第三步后 |  |  |  |  |

1. 运行下面的reverse.c，并说明输出这种结果的原因，修改代码以得到正确的逆序数组

#include <stdio.h>

void xor\_swap(int \*x, int \*y) {

\*y=\*x ^ \*y; /\* 第一步 \*/

\*x=\*x ^ \*y; /\* 第二步 \*/

\*y=\*x ^ \*y; /\* 第三步 \*/

}

void reverse\_array(int a[], int len) {

int left, right=len-1;

for (left=0; left<=right; left++, right--)

xor\_swap(&a[left], &a[right]);

}

int main() {

int a[] = {1,2,3,4,5,6,7};

reverse\_array(a, 7);

int i;

for(i = 0; i < 7; ++i)

printf("%d ", a[i]);

printf("\n");

}

3、编译并运行下面的程序，使用gdb指令查看变量的取值，解释语句输出为False的原因并填写在表格中。

#include <stdio.h>

#include <limits.h>

int main(){

int x = INT\_MAX;

float xf = x;

double xd = x;

printf("++++++++++True or False++++++++++++++++\n");

printf("x==(int)xd %s\n",x==(int)xd?"True":"False"); //语句一

printf("x==(int)xf %s\n",x==(int)xf?"True":"False"); //语句二

float p1 = 3.141592653;

float p2 = 3.141592654;

printf("p1!=p2 %s\n",p1!=p2?"True":"False");//语句三

float f = 1.0e20;

double d = 1.0;

double result1 = d+(f-f);

double result2 = (d+f)-f;

printf("result1==d %s\n",result1==d?"True":"False");//语句四

printf("result2==d %s\n",result2==d?"True":"False");//语句五

}

|  |  |  |
| --- | --- | --- |
|  | 输出True/False | 原因 |
| 语句一 |  |  |
| 语句二 |  |  |
| 语句三 |  |  |
| 语句四 |  |  |
| 语句五 |  |  |

4、观察下面data\_rep.c程序的运行：

int main() {

char x = 0x66;

char y = 0x39;

char x\_bit\_not = ~x;

char x\_not = !x;

char x\_bit\_and\_y = x & y;

char x\_and\_y = x && y;

char x\_bit\_or\_y = x | y;

char x\_or\_y = x || y;

int x1 = (1<<31)-1;

int y1 = 1;

int sum\_x1\_y1 = x1 + y1;

int diff\_x1\_y1 = x1 - y1;

int diff\_y1\_x1 = y1 - x1;

unsigned int x2 = (1<<31)-1;

unsigned int y2 = 1;

unsigned int sum\_x2\_y2 = x2 + y2;

unsigned int diff\_x2\_y2 = x2 - y2;

unsigned int diff\_y2\_x2 = y2 - x2;

}

1. 使用命令gdbtui data\_rep进入gdb的TUI调试模式，之后分别输入命令：layout asm和layout regs，再输入命令start启动程序，然后使用si命令进行单步运行。请在单步运行过程中完成下面的表格（如果TUI模式有问题就直接用gdb，忽略掉前面的提示即可）：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 机器数  (十六进制) | 真值  (十进制) |  | 机器数  (十六进制) | 真值  (十进制) |
| x |  |  | y |  |  |
| ~x |  |  | !x |  |  |
| x & y |  |  | x && y |  |  |
| x | y |  |  | x || y |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 机器数  (十六进制) | 真值  (十进制) | OF | SF | CF | AF |
| x1 |  |  |  |  |  |  |
| y1 |  |  |  |  |  |  |
| sum\_x1\_y1 |  |  |  |  |  |  |
| diff\_x1\_y1 |  |  |  |  |  |  |
| diff\_y1\_x1 |  |  |  |  |  |  |
| x2 |  |  |  |  |  |  |
| y2 |  |  |  |  |  |  |
| sum\_x2\_y2 |  |  |  |  |  |  |
| diff\_x2\_y2 |  |  |  |  |  |  |
| diff\_y2\_x2 |  |  |  |  |  |  |

1. 写出上面表格中每个标识位变化的原因，可直接在上表中注明。

**提交要求：**

请提交一个压缩包151220000.zip到ftp的lab02文件夹（注意修改学号和压缩格式，如果需要覆盖第一次提交，请提交151220000\_1.zip，依次类推）。压缩包内部应该是一个目录。

151220000.zip解压后获得目录内容如下：

151220000

|----reverse.c

|----report.pdf // report中应包含实验中的所有表格