CPSC 240: Computer Organization and Assembly Language
Assignment 05, Fall Semester 2024

CWID: 885024539   Name: Riley Blacklock

Quiz Questions:

From the textbook "X86-64 Assembly Language Programming with Ubuntu," study quiz questions 5 and 6 on page 137. Students do not need to submit answers to the quiz questions as they are found in Appendix D of the textbook.

Programming:

1. Download the “CPSC-240 Assignment05.docx” document.
2. Convert the following C/C++ variable declarations and arithmetic operations to x86-64 assembly language. Find an even number from the "array" array and copy that even number into the "even" array. NOTE: variable sizes and program functions should be equivalent to C/C++ instructions.
3. Use the “yasm/nasm” assembler to assemble the program, the “ld” linker to link the object code, and the “ddd/gdb” debugger to simulate the executable code.

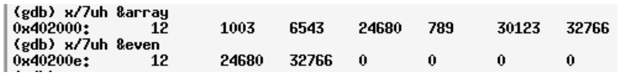
```
unsigned short array[7] = {12, 1003, 6543, 24680, 789, 30123, 32766};    // use dw for 16-bit array
unsigned short even[7];                                   // use dw to declare 16-bit variable
register long rsi = 0, rdi = 0;                           // no need to declare register rsi and rdi
do {
        if(array[rsi] % 2 == 0) {
                even[rdi] = array[rsi];
                rdi++;
        }
        rsi++;
} while(rsi < 7);
```

4. Assemble the "doWhile.asm" file and link the "parity.o" file to get the "parity" executable file.
5. Run the "parity" file with the DDD/GDB debugger to display the simulation results of array and even.
6. Insert source code (parity.asm) and simulation results (GDB window) of the memory array (array and even) in the document. Use hand calculation to verify simulation results.
7. Save the file in pdf or docx format and submit the pdf or docx file to Canvas before the deadline.

[Insert the source code of parity.asm here]

```
; unsigned short array[7] = {12, 1003, 6543, 24680, 789, 30123, 32766};
; unsigned short even[7];
; register long rsi = 0, rdi = 0;
; do {
;       if (array[rsi] % 2 == 0) {
;       even[rdi] = array[rsi];
;       rdi++;
; } rsi++;
; } while (rsi < 7);

section .data
SYS_exit    equ 60
EXIT_SUCCESS      equ 0
array         dw 12, 1003, 6543, 24680, 789, 30123, 32766
even          times 7 dw 0

section .text
  global _start
_start:
  mov rsi, 0
  mov rdi, 0

.loop:
  cmp rsi, 7
  jge   .exit
  mov ax, [array + rsi*2]
  test  ax, 1
  jnz   .not_even
  mov [even + rdi*2], ax
  inc   rdi

.not_even:
  inc   rsi
  jmp  .loop

.exit:
  mov eax, SYS_exit
  xor   edi, edi
  syscall
```

[Insert parity simulation result (GDB window with array and even) here]

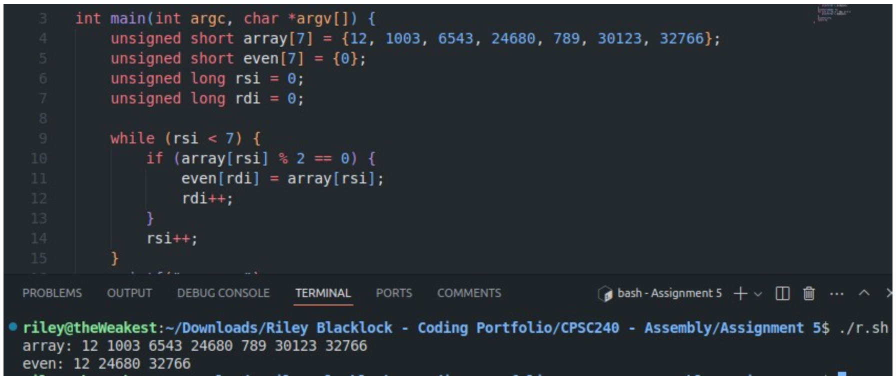


```
(gdb) x/7uh &array
0x402000:       12      1003    6543    24680   789     30123   32766
(gdb) x/7uh &even
0x40200e:       12      24680   32766   0       0       0       0
```

array = 12, 1003, 6543, 24680, 789, 30123, 32766

even = 12, 24680, 32766, 0, 0, 0, 0

[Insert the simulation result verification here]



```
int main(int argc, char *argv[]) {
    unsigned short array[7] = {12, 1003, 6543, 24680, 789, 30123, 32766};
    unsigned short even[7] = {0};
    unsigned long rsi = 0;
    unsigned long rdi = 0;

    while (rsi < 7) {
        if (array[rsi] % 2 == 0) {
            even[rdi] = array[rsi];
            rdi++;
        }
        rsi++;
    }
```

```
riley@theWeakest:~/Downloads/Riley Blacklock - Coding Portfolio/CPSC240 - Assembly/Assignment 5$ ./r.sh
array: 12 1003 6543 24680 789 30123 32766
even: 12 24680 32766
```

12, 24680, 32766 are the only even numbers within the array