CS 3733: Operating Systems

Assignment 1: Binary Reading of 64-bit Integers

1 Objectives:

Practice with system calls to: a) perform file operations; and b) utilize binary I/O in order to read and parse a stream of memory addresses as binary data.

2 Description:

Overview: You will write a program (filename: `binaddrs.c`) to read an arbitrary number of input files whose contents consist of a sequence of physically contiguous 64-bit integers stored in binary form (i.e., no ASCII, UTF-8, nor any other character set, just raw bytes representing the values as they would appear in memory). You may utilize either high-level or low-level I/O for this assignment; the only requirement is that the input file be binary data.

For this first assignment, we will not be utilizing an output file, but rather sending the human-readable converted addresses to standard output. Furthermore, for each address, we will split it into its least significant byte and most significant bytes (see example output below).

You will utilize the `uint64_t` data type provided by the `stdint.h` header as a buffer. The program will sequentially open each of the specified input files in turn (with proper error trapping to gracefully handle file non-existence, non-readable files, etc.; skip to next input file in sequence in case of error on file open). It will then loop through the contents using binary I/O to read one or more 64-bit values at a time (your buffer can be whatever size you would like). Finally, perform the two bitwise operations necessary on each address to isolate the address's least significant byte, as well its corresponding value of this same address with the low byte removed (not simply zeroed, but trimmed off entirely).

Repeat until current input file is exhausted; when successfully opening a new input file, reset the line/address counter back to "1".

Example invocation:

```
./binaddrs smaddrs.bin testaddrs.bin
```

Example output:

```
ssilvestro@fox03:~/a1$ ./binaddrs smaddrs.bin testaddrs.bin
Address    1: 0x00000000000000bc -> 0x00000000000000 : 0xbc
Address    2: 0x00000000000002ec -> 0x00000000000002 : 0xec
Address    3: 0x0000000000000160 -> 0x00000000000001 : 0x60
Address    4: 0x00000000000002a0 -> 0x00000000000002 : 0xa0
Address    5: 0x000000000000036c -> 0x00000000000003 : 0x6c
Address    6: 0x000000000000034c -> 0x00000000000003 : 0x4c
Address    7: 0x0000000000000068 -> 0x00000000000000 : 0x68
Address    8: 0x00000000000002dc -> 0x00000000000002 : 0xdc
Address    9: 0x0000000000000150 -> 0x00000000000001 : 0x50
Address   10: 0x0000000000000284 -> 0x00000000000002 : 0x84
Address   11: 0x0000000000000118 -> 0x00000000000001 : 0x18
Address   12: 0x0000000000000378 -> 0x00000000000003 : 0x78

...

Address 4092: 0x0000000000000098 -> 0x00000000000000 : 0x98
Address 4093: 0x0000000000000104 -> 0x00000000000001 : 0x04
Address 4094: 0x0000000000000b70 -> 0x0000000000000b : 0x70
Address 4095: 0x0000000000000010 -> 0x00000000000000 : 0x10
Address 4096: 0x0000000000000370 -> 0x00000000000003 : 0x70
```

You output should be as close as humanly possible to that shown above, including spacing. There are four spaces for the address/line number, which is right aligned. There are 16 zero-padded spaces for the 64-bit address. Finally, there are 14 zero-padded spaces for the most significant 7 bytes and 2 zero-padded spaces for the least significant byte.

Lastly, the *example* input files utilized above are located as follows:
`/usr/local/courses/ssilvestro/cs3733/*.bin`.

3 Submission Requirements

Assignment submission will be conducted on Blackboard.

1. **Source code:** This assignment should have only one source file (specifically named `binaddrs.c`) and at most one header file, named `binaddrs.h`, if you desire. This is a diminutive program and should consume very little of your time to construct. As such, your program **must** have full error trapping for all library calls (sans obvious exceptions, such as the `printf` family).

2. **Submission:** Please have *only the one or two files above* zipped into a single file named **a1-*abc123*.zip**, where *abc123* represents your MyUTSA ID.