CSCE 312-505
Computer Organization
6 September 2022
Lab I Report

On my honor, as an Aggie, I have neither given nor received unauthorized aid on this academic work.
HUY QUANG LAI

Problem 1

a) Tag 1 tells the compiler to allocate 2-bytes to the stack and treat this value as a 16-bit signed integer.
Tag 2 tells the compiler to allocate a variable amount of bytes to the stack. This value would be treated as an address to FILE data. Additionally, Tag 2 also allocates space on the heap for the File itself
Tag 3 will send the formatted string to the file output stream.

b) Code compilation with output

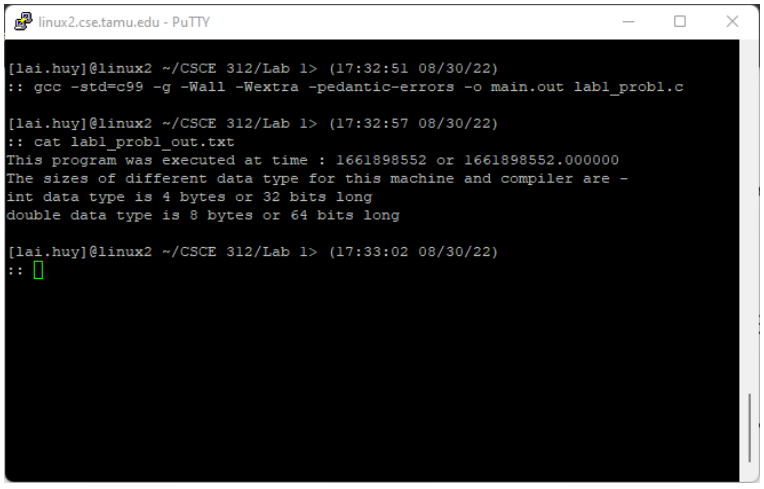

Figure 1: Compilation and Output

c) The data type of "timeval" is a signed 16-byte integer. It indicates the number of second that have elapsed since 00:00:00 on January 1, 1970, Coordinated Universal Time.

Problem 2

a) Output with code

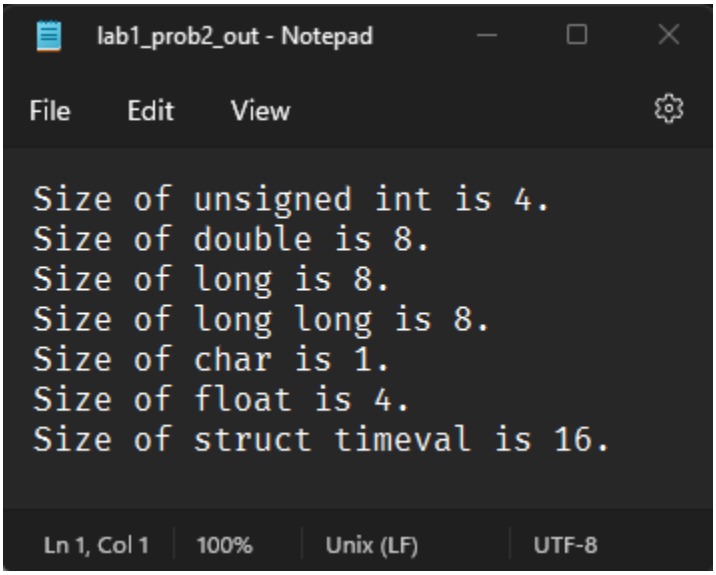

Figure 2: Output

lab1_prob2.c

Lab 1 > lab1_prob2.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>

int main() {
    FILE* file;
    if ((file=fopen("lab1_prob2_out.txt", "w")) == NULL) {
        printf("Error opening the file, so exiting\n");
        exit(1);
    }

    fprintf(file, "Size of unsigned int is %lu.\n", sizeof(unsigned int));
    fprintf(file, "Size of double is %lu.\n", sizeof(double));
    fprintf(file, "Size of long is %lu.\n", sizeof(long));
    fprintf(file, "Size of long long is %lu.\n", sizeof(long long));
    fprintf(file, "Size of char is %lu.\n", sizeof(char));
    fprintf(file, "Size of float is %lu.\n", sizeof(float));
    fprintf(file, "Size of struct timeval is %lu.\n", sizeof(struct timeval));

    return 0;
}
```

Figure 3: Code

Problem 3

a) The Five Requirements

i. BELL = ER && !DSBF

ii. BELL = ER && !DC

iii. BELL = DSBF && !ER && DC

iv. DLA = !DOS && !KIC

v. BA = BP && CM

b) Truth Table

DOS	DSBF	ER	DC	KIC	DLC	BP	CM	BELL	DLA	BA
X	0	1	X	X	X	X	X	1	X	X
X	X	1	0	X	X	X	X	1	X	X
X	1	0	1	X	X	X	X	0	X	X
0	X	1	X	X	X	X	X	X	1	X
X	X	X	X	X	X	1	1	X	X	1

c) Code

```
void read_inputs_from_ip_if(){

    // 1. Provide your input code here
    // This function should read the current state of the available sensors (8 in total)

    // Hint : You can use scanf to obtain inputs for the sensors
    printf("Is the Driver on the Seat?\t");
    scanf("%u", &driver_on_seat);

    printf("Is the Driver Seat Belt Fastened?\t");
    scanf("%u", &driver_seat_belt_fastened);

    printf("Is the Enginer Running?\t");
    scanf("%u", &driver_seat_belt_fastened);

    printf("Are the Doors Closed?\t");
    scanf("%u", &driver_seat_belt_fastened);

    printf("Is the Key in Car?\t");
    scanf("%u", &driver_seat_belt_fastened);

    printf("Is the Door Lock Leaver activated?\t");
    scanf("%u", &driver_seat_belt_fastened);

    printf("Is the Break Pedal activated?\t");
    scanf("%u", &driver_seat_belt_fastened);

    printf("Is the Driver Seat Belt Fastened?\t");
    scanf("%u", &driver_seat_belt_fastened);
}
```

Figure 4: Read Inputs

```c
void write_output_to_op_if(){

    // 2. Provide your output code here
    // This function should display/print the state of the 3 actuators (DLA/BELL/BA)
    printf("\nBELL:\t%u\n", bell);
    printf("DLA:\t%u\n", door_lock_actu);
    printf("BA:\t%u\n", brake_actu);
}
```

Figure 5: Write Output

```c
// The code segment which implements the decision logic
void control_action(){

    /*
       The code given here sounds the bell when driver is on seat
       AND hasn't closed the doors. (Requirement-2)

       3. Provide your own code to do problems 3, which satisfies 5 requirements
    */
    if (engine_running && !doors_closed)
       bell = 1;

    if (engine_running && !driver_seat_belt_fastened)
        bell = 1;

    if (!driver_on_seat && !key_in_car)
        door_lock_actu = 1;
    else
        door_lock_actu = 0;

    if (brake_pedal && car_moving)
        brake_actu = 1;
    else
        brake_actu = 0;
}
```

Figure 6: Logic

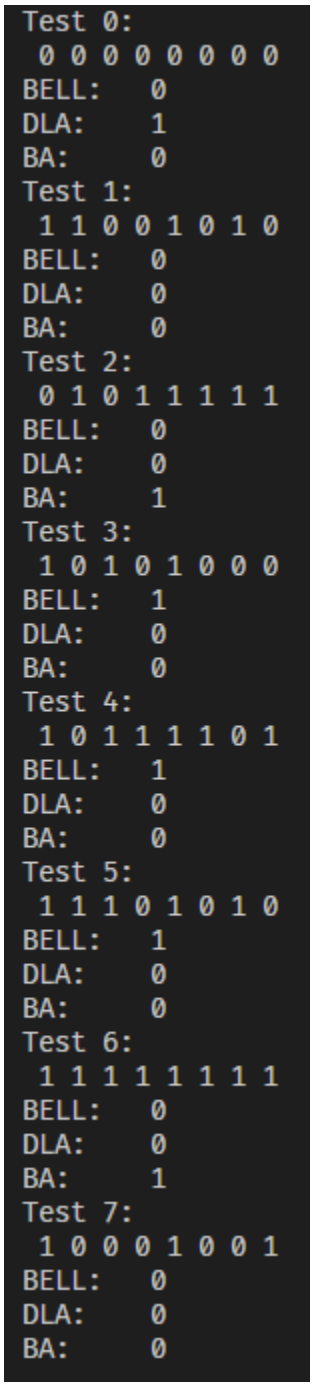

Figure 7: Output

Problem 4

```c
enum Inputs { DOS = 1, DSBF = 2, ER = 4, DC = 8, KIC = 16, DLC = 32, BP = 64, CM = 128 };

//The code segment which implements the decision logic
void control_action(){

    /*
       The code given here sounds the bell when driver is on seat
       AND hasn't closed the doors. (Requirement-2)
       Replace this code segment with your own code to do problems 3 and 4.
    */

    //if (engine_running && !doors_closed) bell = 1;
    if ((input & 12) == 4)
        output = output | 1;
    if ((input & (ER + DSBF)) == 4)
        output |= 1;
    if ((input & (KIC + DOS)) == 14)
        output |= 2;
    if ((input & (BP + CM)) == BP + CM)
        output |= 4;
}
```

Figure 8: Code

```
gcc -std=c99 -g -Wall -Wextra -pedantic-errors -o main.out lab1_prob4_framework.c
lab1_prob4_framework.c: In function ‘read_inputs_from_ip_if’:
lab1_prob4_framework.c:33:2: warning: format ‘%d’ expects argument of type ‘int *’,
  scanf("%d", &input);
  ^
lab1_prob4_framework.c: In function ‘main’:
lab1_prob4_framework.c:65:14: warning: unused parameter ‘argc’ [-Wunused-parameter]
 int main(int argc, char *argv[])
              ^
lab1_prob4_framework.c:65:26: warning: unused parameter ‘argv’ [-Wunused-parameter]
 int main(int argc, char *argv[])
                          ^
./main.out
Case 0:  0 0 0
Case 1:  0 0 0
Case 2:  0 0 1
Case 3:  1 0 0
Case 4:  1 0 0
Case 5:  1 0 0
Case 6:  0 0 1
Case 7:  0 0 0
```

Figure 9: Output

Problem 5

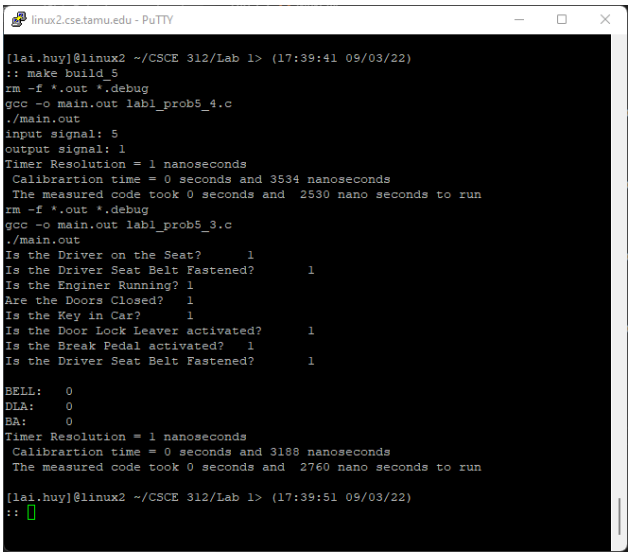

Figure 10: Execution time on linux.cse.tamu.edu

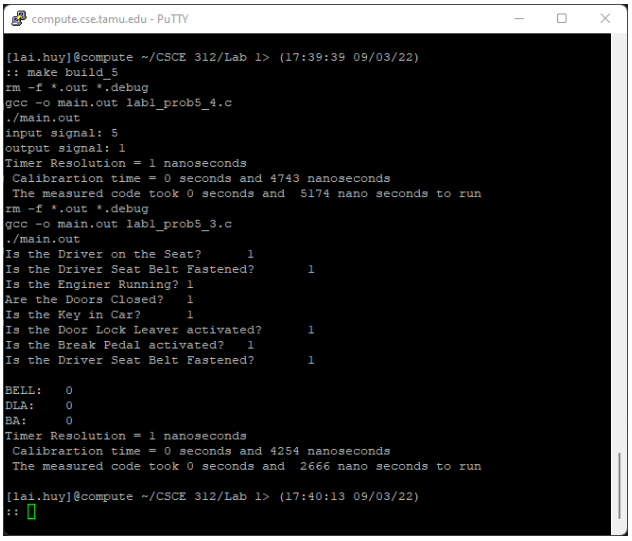

Figure 11: Execution time on compute.cse.tamu.edu