Facultad de Ingeniería - Departamento Ciencias Físicas y Tecnología
Arquitectura de Computadores/Interfaces y Arquitectura Hardware
SEGUNDO EXÁMEN PARCIAL

Nombre	Fecha 05 de abril 2017
Código:	Duración: *2 horas*

1 [50% - INGENIERÍA INVERSA SENTENCIA SWITCH]

Suponga una función *switch-case* en lenguaje de alto nivel con la estructura general mostrada en la figura 1:

```
int switcher(int a, int b, int c)
{
  int answer;
  switch(a) {
  case      :                  /* Case A */
    c = _____________;

  case_____________:           /* Case B */
    answer =_____________;
    break;
  case _____________:          /* Case C */
  case _____________:          /* Case D */
    answer =_____________;
    break;
  case _____________:          /* Case E */
    answer =_____________;
    break;
  default:
    answer =_____________;
  }
  return answer;
}
```

Figura 1

```
1
2
3
4
5   mov eax,[ebp+8]
6   cmp eax,7;
7   ja L2
8   jmp [tablaDir + eax*4]
9   L2:
10  mov eax,[ebp +12];
11  jmp L8
12  L5:
13  mov eax,4
14  jmp L8
15  L6:
16  mov eax, [ebp+12]
17  xor eax,15
18  mov [ebp + 16],eax
19  L3:
20  mov eax,[ebp +16]
21  add eax,112
22  jmp L8
23  L4:
24  mov eax,[ebp + 16]
25  add eax,[ebp +12]
26  sal eax,2
27  L8:
28
29
30
```

Figura 2

En la figura 2 se muestra parte del código de ensamblador generado en el proceso de compilación. La tabla de direcciones asociada el programa y que ubicada en la memoria de programa es:
tablaDir ***dword*** L3, L2, L4, L2, L5, L6, L2, L4
Donde L2, L3,…L6 son los valores de las direcciones que hacen referencia a las etiquetas usadas en el programa.

Se pide entonces que:

A. Rellene las partes que faltan del código de alto nivel. Tenga en cuenta que la función *switcher* accede al parámetro ***a*** usando el direccionamiento con ebp+8, se accede al parámetro ***b*** con ebp+12, y el ***c*** y bp+16. Tenga en cuenta además que sólo hay una forma de adaptar los diferentes casos a la plantilla dada.
B. Completa las líneas faltantes (al inicio y al final) del código ensamblador para que el programa se comporte como subrutina por paso de parámetros usando la pila.
C. Escriba un programa en lenguaje ensamblador x86 equivalente al siguiente programa principal escrito en lenguaje de alto nivel

```
int main()  // programa ítem C
{
int a,b,c, ans
ans=switcher(int a, int b, int c)
}
```

D. Para un caso cualquiera de la sentencia switch, calcule el desempeño (medido en GIPS) en la ejecución del programa ensamblador de la figura 2, si se ejecutara en procesador sin pipeline, asumiendo que cada instrucción se completa en 5 fases (F, D, E, M y W) y que cada fase dura 60 ps. Cuánto en la latencia de la instrucción? En qué porcentaje se mejora el desempeño, si se ejecutara el programa en un procesador con pipeline de 5 estados.

2 [50% - IP header checksum calculation]

Escriba un programa en lenguaje ensamblador que permita realizar la verificación del campo del checksum de 16 bits del encabezado de un datagrama IP. El encabezado IP puede tener entre 20 a 60 bytes de longitud y contiene información esencial para el enrutamiento y la entrega de información en redes de datos.

En la figura 3 se muestra la estructura del encabezado (Header) IP

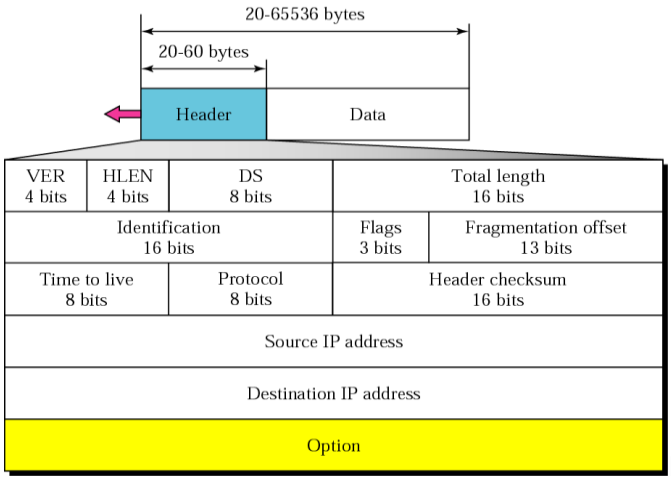


Figura 3

Y en la figura 4 se muestra un ejemplo de cómo se calcula el valor del checksum de 16 bits:

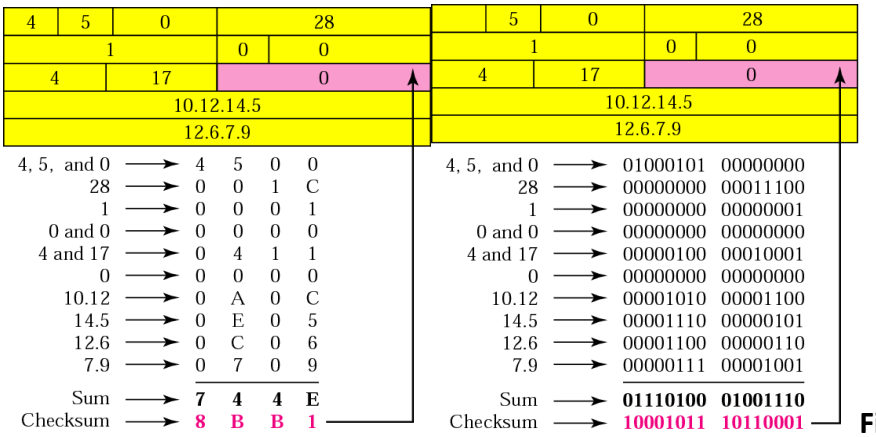


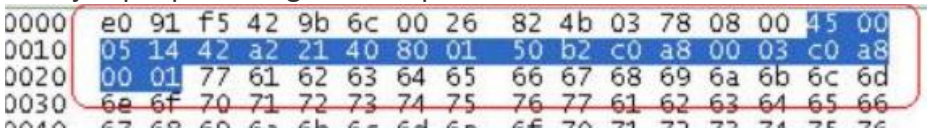
Figura 4

Cuando se recibe un datagrama IP, el receptor realiza la verificación de la integridad de la información calculando a cada encabezado el Checksum, el cual se obtiene, primero realizando sumas de 16 bits de todo los bytes del encabezado (teniendo en cuenta posibles acarreos) y a la suma total (sum) se le saca luego el complemento a 1. Si ése valor es cero significa que el paquete se recibió bien, sin modificaciones, pero si el valor del Checksum es diferente de cero, hubo un error y se descarta el paquete.

De la figura 3, los 4 bits iniciales del encabezado corresponde al campo *VER*, indica la versión del protocolo, es decir 4 porque es Ipv4 (IP versión 4). Los 4 bits siguientes es el campo *HLEN*, en el ejemplo es 5, ya que al multiplicarse dicho valor por VER se obtiene el tamaño total en bytes del encabezado, en el ejemplo es 4 y 5, es decir 4x5=20 bytes, que es la longitud mínima (un header sin el campo Option). Teniendo en cuenta que el tamaño del encabezado IP puede variar entre 20 a 60 bytes, el campo HLEN puede ser un valor entre 5 a 15. (Asumiendo siempre un datagrama IPv4). Si el resultado de dicha multiplicación no da un valor entre 20 y 60, se descarta el paquete, ya que indica que algunos bytes fueron sustraído y agregados.

Se pide entonces:
Escriba un programa en lenguaje ensamblador x86 MASM que permita verificar el Checksum de un encabezado IP. Asuma que los valores del encabezado se definen en el segmento de datos del programa referenciados como una variable llamada *HeaderIP* y que además se define la variable sum y Checksum en el mismo segmento. El programa debe funcionar para encabezados de tamaño variable. Defina una variable tipo char con el nombre DescartaPaquete, en dicha variable su programa debe escribir 'F' o 'T', dependiendo si el paquete se descarta o no ('T' se descarta y 'F' no se descarta, el paquete llegó bien).

Nota:
Por ejemplo para la siguiente captura:



Entonces la declaración de la variable HeaderIp sería:

HeaderIP **BYTE** 45h,00h,00h,1Ch,42h,0a2h,21h,40h,80h,01h,50h,0b2h,0c0h,0a8h,
00h,03h,0c0h,0a8h,00h,01h