INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Nombres: Mora Ayala Jose Antonio, Torres Carrillo Josehf Miguel Angel

Práctica 2.-Checksum

Objetivo: El alumno(s) implementará en lenguaje C una función que reciba una trama (arreglo de caracteres sin signo) y que imprima el checksum correspondiente.

Utilizado para detectar errores en la transmisión de datos a nivel de software. Usado por los protocolos de TCP/IP.

Vamos a trabajar con la estructura de la cabecera IP; por ahora solo basta recordar en que posición del arreglo se encuentra el Checksum. Si consideramos unicamente el paquete IP, entonces el checksum se encuentra en T[10] y T[11]

Prueba 1. En esta trama los valores del Checksum estan en 0000 ya que apenas se va a calcular (ALICIA)

```
Unsigned char T[]={0x45, 0x00, 0x01, 0xe2, 0xd7, 0xdb, 0x40, 0x00, 0x80,
0x06, 0x00, 0x00, 0xc0, 0xa8, 0x01, 0x43, 0x94, 0xcc, 0x3a, 0xdd}
```

Prueba 2: Para esta trama ya se tiene el valor del checksum (es la que recibió BETITO) verificar si el checksum es correcto y en caso de no serlo mostrar el correcto.

```
45 00 01 9c d7 de 40 00 80 06 88 9d c0 a8 01 43 94 cc 3a dd
```

Instrucciones.
-Abrir un block de notas y escribir el código correspondiente en C.
-Las tramas se inicializarán en hexadecimal en un arreglo de caracteres sin signo
-Compilar en consola haciendo uso de gcc (en caso de no tenerlo, instalarlo).
-Una vez terminado deberán probar con los dos ejercicios vistos en clase e incluir aquí las capturas de pantalla

Se entrega este documento, incluir:

Todo tu Código, sugiero utilizar alguna herramienta para dar formato como:

http://www.planetb.ca/syntax-highlight-word

Deberan crear y mostrar **el mapa de memoria** utilizado en su programa (considerar registros de 8 bits)
En el mapa de memoria NO se debe considerar el arreglo de la trama.

M. en C. Nidia A. Cortez Duarte

Práctica 2.-Checksum

Ejemplo de mapa de memoria

Unsigned short int CHECKSUM	0	0	0	0	0	0	0	0
…								

Finalmente incluir tus **Conclusiones**

¿Habías escuchado sobre funciones de comprobación de trama?
¿Qué opinas sobre la integridad de la información que se transmite en las redes?

Criterio	**Valor**	**Tu evaluación**
El programa se escribe en un block de notas y se compila con gcc	**1**	1
Se incluye todo el código	**1**	1
El mapa de memoria refleja todas las variables utilizadas en el programa y han sido seleccionadas de forma consciente.	**1**	1
Las imágenes de la ejecución son claras	**1**	1
Se incluyen las conclusiones solicitadas	**1**	1
TOTAL	**5**	5

INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Nombres: Mora Ayala Jose Antonio, Torres Carrillo Josehf Miguel Angel

Práctica 2.-Checksum

Programa Checksum.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

//Variable encargada de recorrer los ciclos for
unsigned char contador;

//Variable donde se almacenara el resultado del checksum
unsigned char Checksum[4];

//Variable para guardar el valor de acarreo
unsigned char acarreo;

//Tramas - Descomentar la trama que se quiera usar
//unsigned char T[]={0x45, 0x00, 0x01, 0xe2, 0xd7, 0xdb, 0x40, 0x00, 0x80, 0x06, 0x00,
0x00, 0xc0, 0xa8, 0x01, 0x43, 0x94, 0xcc, 0x3a, 0xdd};
unsigned char T[]={0x45, 0x00, 0x01, 0x9c, 0xd7, 0xde, 0x40, 0x00, 0x80, 0x06, 0x88,
0x9d, 0xc0, 0xa8, 0x01, 0x43, 0x94, 0xcc, 0x3a, 0xdd};

//Funcion donde haremos las operaciones principales para obtener el checksum
void comprobacionChecksum ();

int main(void){

	if (T[10] == 0x00 && T[11] == 0x00){

		//Programa Primera Parte - usar primera trama
		comprobacionChecksum(T);
	}
	else{

		//Programa Segunda Parte - Usar segunda trama
		comprobacionChecksum(T);
			T[10] = 0x00;
			T[11] = 0x00;
		comprobacionChecksum(T);
	}
}

void comprobacionChecksum (){

	//Inicializacion de variables
	for ( contador = 0; contador < 4; contador++ ){

		Checksum[contador] = 0x00;
	}
```

M. en C. Nidia A. Cortez Duarte

Práctica 2.-Checksum

```
	acarreo = 0x00;

	//Sumatoria de la primera columna de los valores hexadecimales
	for ( contador = 0 ;  contador < 10; contador++){

		Checksum[0] = Checksum[0] + T[(contador*2)+1] -
((T[(contador*2)+1]/0x10)*0x10);
	}

	acarreo = (Checksum[0]/0x10)*0x10;

	Checksum[0] = Checksum[0] - acarreo;

	acarreo = acarreo/0x10;

	//Sumatoria de la segunda columna de los valores hexadecimales

	for ( contador = 0 ;  contador < 10; contador++){

		Checksum[1] = Checksum[1] + (T[(contador*2)+1]/0x10);
	}

	Checksum[1] = Checksum[1] + acarreo;

	acarreo = (Checksum[1] / 0x10)*0x10;

	Checksum[1] = Checksum[1] - acarreo;

	acarreo = acarreo/0x10;

	//Sumatoria de la tercera columna de los valores hexadecimales

	for ( contador = 0 ;  contador < 10; contador++){

		Checksum[2] = Checksum[2] + T[(contador*2)] - ((T[(contador*2)]/0x10)*0x10);
	}

	Checksum[2] = Checksum[2] + acarreo;

	acarreo = (Checksum[2] / 0x10)*0x10;

	Checksum[2] = Checksum[2] - acarreo;

	acarreo = acarreo/0x10;
```

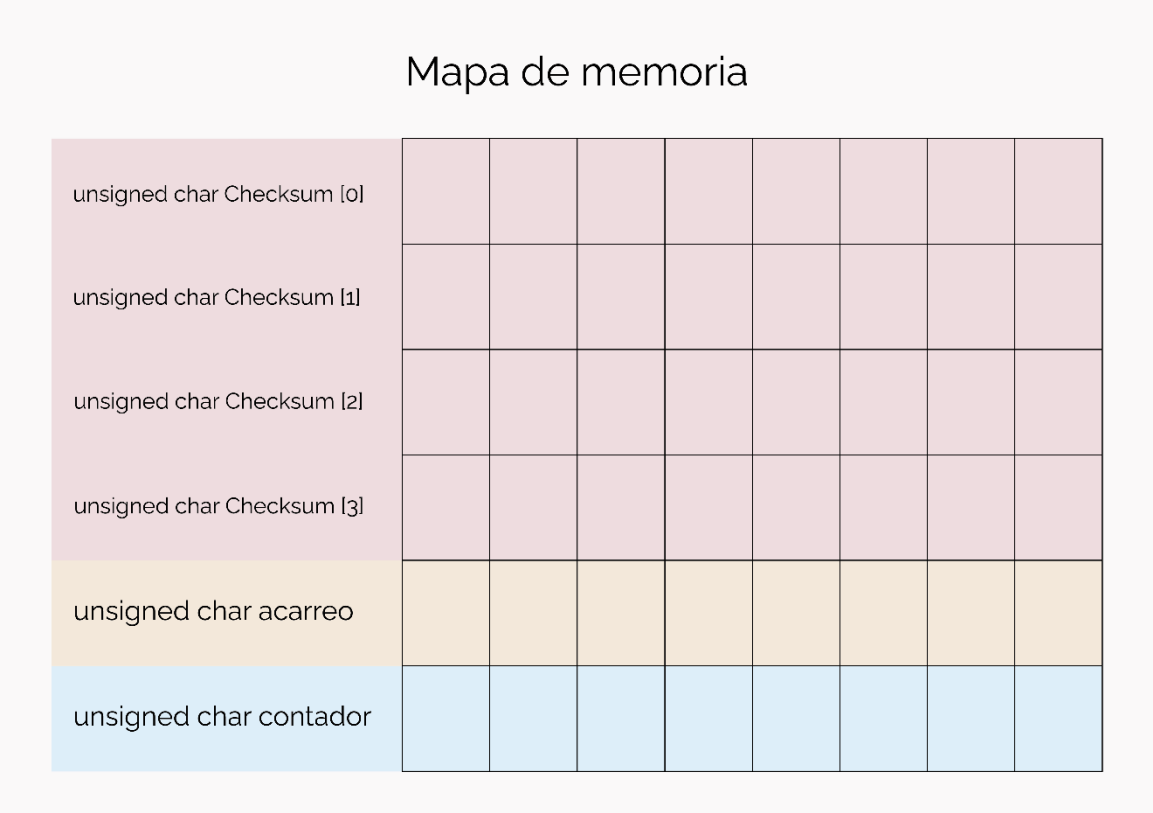
Práctica 2.-Checksum

```
	//Sumatoria de la cuarta columna de los valores hexadecimales

	for ( contador = 0 ;  contador < 10; contador++){

		Checksum[3] = Checksum[3] + (T[(contador*2)]/0x10);
	}

	Checksum[3] = Checksum[3] + acarreo;

	acarreo = (Checksum[3] / 0x10)*0x10;

	Checksum[3] = Checksum[3] - acarreo;

	acarreo = acarreo/0x10;

	Checksum[0] = Checksum[0] + acarreo;

	//Obteniendo el inverso

	for ( contador = 0; contador < 4; contador++ ){

		Checksum[contador] = ~Checksum[contador];
		Checksum[contador] = Checksum[contador] - ((Checksum[contador]/0x10)*0x10);
	}

	if (T[3] == 0xe2){

		printf("El CHECKSUM ES: %0.1x%0.1x%0.1x%0.1x\n", Checksum[3], Checksum[2],
Checksum[1], Checksum[0]);
	}

	if (Checksum[0] == 0x08 && Checksum[1] == 0x0e && Checksum[2] == 0x0f && Checksum[3]
== 0x08 && T[3] == 0x9c){

		printf("El CHECKSUM ES: %0.1x%0.1x%0.1x%0.1x (ALICIA) \n", Checksum[3],
Checksum[2], Checksum[1], Checksum[0]);
	}
	else if ( T[3] == 0x9c ){

		printf("EL CHECKSUM ES INCORRECTO (BETITO)\n");
	}
}
```

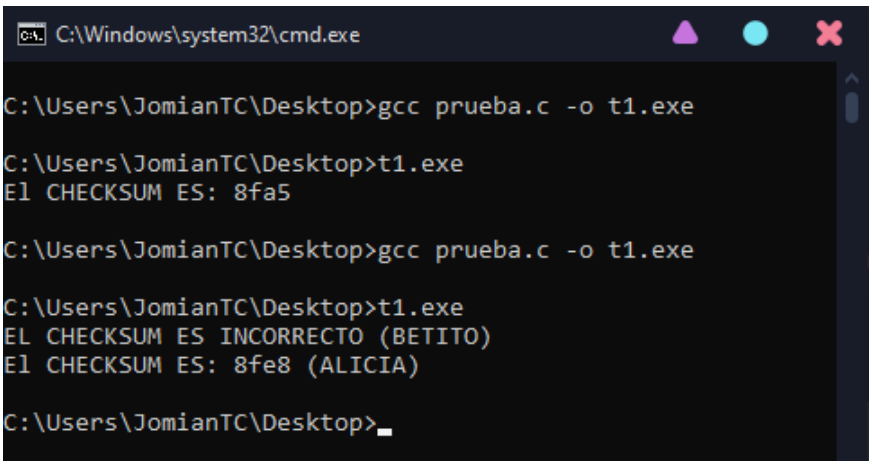
INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Nombres: Mora Ayala Jose Antonio, Torres Carrillo Josehf Miguel Angel

Práctica 2.-Checksum

Mapa de memoria para Checksum.c



Ejecución Checksum.c



M. en C. Nidia A. Cortez Duarte

Práctica 2.-Checksum

Checksumv2.c

A continuación presentamos una alternativa del programa anterior, en este damos la propuesta de un programa que puede identificar y proporcionar el checksum para cualquier trama deseada, aunque se usan variables de tipo int, no encontramos una forma para optimizar esta cuestión de la memoria utilizada, pero nos parece buena solución al ejercicio planteado.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void evaluaT(unsigned char T[]);

int main(int argc, char const *argv[])
{
    unsigned char T[] = {0x45, 0x00, 0x01, 0xe2, 0xd7, 0xdb, 0x40, 0x00, 0x80, 0x06,
0x00, 0x00, 0xc0, 0xa8, 0x01, 0x43, 0x94, 0xcc, 0x3a, 0xdd};
    unsigned char A[] = {0x45, 0x00, 0x01, 0x9c, 0xd7, 0xde, 0x40, 0x00, 0x80, 0x06,
0x8f, 0xe8, 0xc0, 0xa8, 0x01, 0x43, 0x94, 0xcc, 0x3a, 0xdd};
    unsigned char J[] = {0x45, 0x00, 0x00, 0x42, 0x00, 0xe2, 0x00, 0x50, 0x80, 0x06,
0x9b, 0xbc, 0x94, 0xcc, 0x3a, 0x11, 0x94, 0xcc, 0x3a, 0x1e};

    evaluaT(T);
    evaluaT(A);
    evaluaT(J);
}

void evaluaT(unsigned char T[])
{
    unsigned int suma = 0x00, acarreo = 0x00;
    unsigned char i, tam;

    i = 0x00;
    tam = 0x14;
    // Agarrando valores pares
    while (tam > 1)
    {
        acarreo = (((T[i] << 8) & 0xFF00) | ((T[i + 1]) & 0xFF));
        suma += acarreo;
        if ((suma & 0xFFFF0000) > 0)
        {
            suma = suma & 0xFFFF;
            suma += 1;
        }
        i += 2;
        tam -= 2;
    }
    suma = ~suma;
    suma = suma & 0xFFFF;
    if (suma == 0)
    {
```

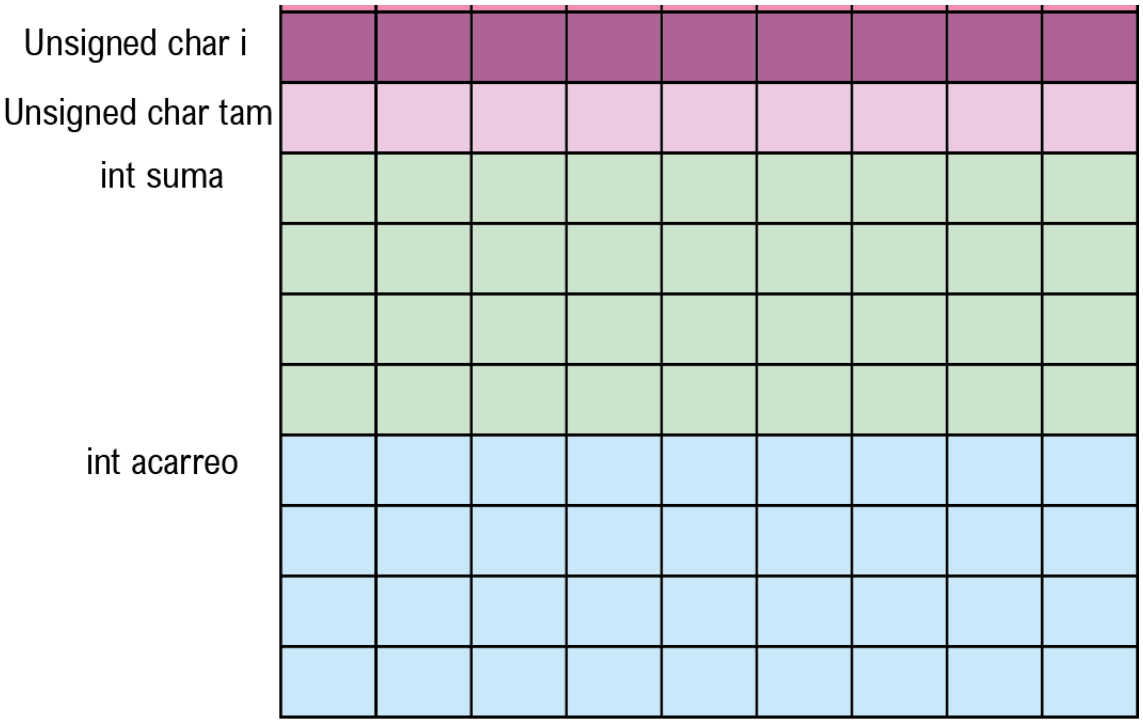
Práctica 2.-Checksum

```
        printf("El checksum es correcto\n");
    }
    else if (T[10] == 0 && T[11] == 0)
    {
        printf("El checksum correcto es: 0x%.4x\n", suma);
    }
    else
    {
        suma = 0x00, acarreo = 0x00;
        i = 0x00;
        tam = 0x14;
        T[10] = 0;
        T[11] = 0;
        while (tam > 1)
        {
            acarreo = (((T[i] << 8) & 0xFF00) | ((T[i + 1]) & 0xFF));
            suma += acarreo;
            if ((suma & 0xFFFF0000) > 0)
            {
                suma = suma & 0xFFFF;
                suma += 1;
            }
            i += 2;
            tam -= 2;
        }
        printf("La suma es: 0x%.4x\n", suma);
        suma = ~suma;
        suma = suma & 0xFFFF;
        printf("El checksum correcto es: 0x%.4x\n", suma);
    }
}
```

INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Nombres: Mora Ayala Jose Antonio, Torres Carrillo Josehf Miguel Angel

Práctica 2.-Checksum

Mapa de Memoria



Ejecución

```
C:\Users\TONY AYALA\OneDrive\College\Quinto_Semestre\Redes_de_Computadoras>.\checksumTunas
Trama lado Alicia:
El checksum correcto es: 0x8fa5
Trama lado Beto:
La suma es: 0x7017
El checksum correcto es: 0x8fe8
Trama ejemplo:
El checksum es correcto
```

M. en C. Nidia A. Cortez Duarte

INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Nombres: Mora Ayala Jose Antonio, Torres Carrillo Josehf Miguel Angel

Práctica 2.-Checksum

Preguntas:

- ¿Habías escuchado sobre funciones de comprobación de trama?
  - No, nunca había escuchado de tales funciones y mucho menos me imaginaba la forma en que estas eran empleadas, lo cual mediante las lecturas e investigaciones realizadas para esta practica realmente me dejan impresionado, me parece que no es muy complejo de implementar, pero sí de comprender al 100% , pues hay distintos métodos de realizar el programa en cuestión para poder comprobar una trama, y si a esto le sumamos que queremos ocupar la menor cantidad de recursos posibles, el reto aumenta, pues estamos acostumbrados a declarar variables sin realmente están conscientes del costo que estas generan o la memoria que ocupara, cuando podemos realizar una gran variedad de operaciones a nivel binario que de igual forma ayudan mucho .
- ¿Qué opinas sobre la integridad de la información que se transmite en las redes?
  - Me parece que es muy importante, puesto que como hemos visto en clase y mediante las explicaciones que se proporcionan en internet la forma en que son mandados y recibidos los datos importa mucho así como todos aquellos procesos que realizan un correcto control y seguimiento de ellos para que sean íntegros, y no sufran de modificaciones durante el proceso de emisión o recepción

M. en C. Nidia A. Cortez Duarte