CI-3815
Organización del Computador

Ejercicios

Manejo de Subrutinas en MIPS

1. Una **suma de verificación**, también llamada **suma de chequeo** o **checksum**, es una función hash que tiene como propósito principal detectar cambios accidentales en una secuencia de datos para proteger la integridad de los datos, verificando que no haya discrepancias. La idea es que se transmita el dato junto con su valor hash, de esta forma el receptor puede calcular el valor hash de la secuencia recibida y la puede comparar con el valor hash recibido. Si hay una discrepancia se pueden rechazar los datos o pedir una retransmisión.

   La idea en nuestro caso, es sumar todas las palabras que son enviadas y transmitir el resultado de dicha suma. Cuando los datos son recibidos se chequea la data y la suma. La función ***checksum*** que se especifica a continuación devuelve el complemento a uno de la suma total de las palabras transmitidas. Se hace de esta forma para que si se envía un mensaje con todos los valores en 0 no sea considerado como un mensaje válido. A continuación se especifica la función en C que lleva a cabo la suma de verificación.

   El objetivo es traducir esta función a una función en MIPS. Se desea que sea lo más corto posible y rápido. Deberán minimizar el uso de la pila y el número de instrucciones. Deben cumplir con las convenciones de MIPS

```
int checksum (int start, int frame[]) {
/* frame[] is the array that contains pointers to data we want to send.
start is guaranteed to be less than 512.*/
  int sum, result, x;
  int max_length = 512;
  result = 0;
  sum = 0;

  for (x = start; x < max_length; x++) {
    result = getdata(frame[x]);          /* don't write getdata, just call it */
    sum = sum + result;
  }
  return ((-sum) - 1);
}
```

2. Escriba las siguientes funciones en MIPS tomando en cuenta la convención de responsabilidad compartida vista en clase

```
int max2(int i, int j)
   {
      if (i<j)  return j;
      else      return i;
   }

int max3(int i, int j, int k)
   {
      return max2(max2(i,j), k);
   }
```

3. Traduzca la siguiente función a MIPS tomando en cuenta la convención de responsabilidad compartida vista en clase

```
int FUN1(int x, int y)
{
      return (x+y) + FUN2(y,x);
}
```

Considere que existe una función FUN2. Su programa debe respetar el orden de ejecución, es decir primero calcula x+y antes de llamar a FUN2.