Sumador

Marzo 19, 2019

Maximiliano Sapién Fernández
Esteban Manrique de Lara Sirvent
Fernando Garrote de la Macorra

Objetivos

El objetivo principal planteado en esta práctica consistía en la construcción y programación de software en lenguaje Ensamblador que pudiera realizar la función de un sumador de dos bits , y que el resultado arrojado por el mismo pudiera ser reflejado en un display de 7 segmentos. Lo anterior se lograría utilizando distintos modos de direccionamiento. Asimismo, se estableció que un objetivo secundario de esta práctica sería la puesta en práctica del conocimiento visto en clase relacionado con los distintos tipos de direccionamiento, operaciones como ADD y ANL y conceptos como el de overflow (especie de acarreo cuando ya no hay bits disponibles)

Información referente al concepto de “Decodificador BCD a 7 segmentos”

Un decodificador BCD a 7 segmentos puede ser definido como un elemento digital que trabaja a partir de estados lógicos, con los cuales generar una salida en específico teniendo en cuenta datos de entrada característicos. Su funcionamiento operacional se basa en introducir en las diferentes entradas del decodificador, algún número en código binario que sea igual al número que se quiera representar en los pines de salida del integrado. Un decodificador de este tipo, junto con un conjunto de arreglos de LED -display de 7 segmentos de cátodo común en el caso de la práctica-, permite mostrar números del 0 al 9 (dependiendo del dato recibido en binario). El conjunto decodificador-display puede ser usado en cuestiones que requieran un clase de contador o indicador numérico de cantidades menores a 10 -cuando se trate de números en Base decimal-, o cantidades menores a 16 -cuando se trate de números en Base hexadecimal- (podrían usarse distintos displays en caso de que se quieran obtener números más elevados).

Información referente al programa creado en Ensamblador y los distintos tipos de direccionamientos involucrados

El programa creado para completar los objetivos propuestos en esta práctica contemplaba una serie de pasos: decodificar un dato (proveniente del Puerto 1) que lee los primeros 2 bits menos significativos, decodificar un dato (proveniente del Puerto 2) que lee los primeros 2 bits menos significativos, realizar la operación ADD, que permite unificar ambos datos en uno sólo, hacer una decodificación del dato mediante el uso de un AND Lógico (AND), buscar en una “tabla” de posibles datos decodificados el valor requerido por el usuario y finalmente, mediante direccionamiento indirecto y directo, reflejar el número requerido por el usuario en un display de 7 segmentos. En caso de que el resultado arrojado por el ADD del programa diera un valor con overflow, el programa reflejaría una “E”. En caso contrario, reflejaría el valor del resultado en su representación decimal.

Los 3 tipos de direccionamiento presentes en este programa son:

- Direccionamiento Directo: Se usa este tipo de direccionamiento en los primeros pasos del main; el direccionamiento directo permite mover aquellos ubicado en registro/dirección directa hacia el Acumulador o algún otro registro. Se utilizó este tipo de direccionamiento en cuatro instancias del programa: para mover aquel valor ubicado en el Puerto 1 hacia el Acumulador, para mover el valor del Acumulador (previamente del Puerto 1) a la variable “variable1” creada al inicio del programa, para mover aquel valor ubicado en el Puerto 2 hacia el Acumulador y para mover el valor ubicado en el Acumulador al Puerto 3 (esta instancia de direccionamiento directo era la que permitía que el display de 7 segmentos mostrará el número deseado por el usuario).
- Direccionamiento Inmediato: El direccionamiento inmediato permite mover el valor o “constante” elegida por el usuario hacia un registro o hacia el Acumulador. A diferencia de las instancias del direccionamiento directo, la aparición del de este tipo de direccionamiento es posterior a los primeros pasos del main en el programa. Se utilizó solamente una vez al direccionamiento inmediato en este software: permite mover el valor arrojado por la tabla de valores binarios, que representan a cada uno de los posibles resultados a mostrar en el sumador (0,1,2,3,E), al DPTR (Data Pointer), el cual es considerado un registro accesible para el usuario de 16 bits.
- Direccionamiento Indirecto: Es considerado como el tercer tipo de direccionamiento presente en este programa; este tipo de direccionamiento es

útil cuando se debe trabajar con apuntadores y registros especiales (como es el caso del DPTR en este programa en específico). El direccionamiento indirecto es utilizado en una ocasión en el programa que se desarrolló para esta práctica: mediante el direccionamiento indirecto, se mueve el valor ubicado en el DPTR (después de la aplicación de una instancia de direccionamiento inmediato) al Acumulador (A).

Conclusiones:

Maximiliano Sapién Fernández- En este proyecto el objetivo al principio parecía más fácil que el de las prácticas anteriores era muy similar al cableado de la práctica anterior. Siento que en esta práctica hubo menos confusión al momento de ejecutar las instrucciones porque había conocimiento previo.

En el cableado no tuvimos ningún problema porque cambiamos 2 cables con respecto a la práctica anterior. Todo el cableado que usamos para la práctica anterior no lo quitamos entonces no tuvimos que volver a cablear todres. Esto se debe a que en este proyecto ya teníamos clara la simbología, y que el cableado.

Esteban Manrique de Lara Sirvent- El primer punto a comentar acerca de esta práctica tiene relación con los conocimientos puestos en práctica durante la realización de la misma. Además de continuar con la implementación de diversas formas de direccionamiento, se tuvieron que manejar los conceptos de bits significativos y la operación de ADD; lo anterior debido a que la práctica requería que se trabajara con la suma de los 2 bits más significativos del Puerto 1 y 2. Asimismo, el concepto de “overflow” fue utilizado ya que al trabajar con sólo 2 bits, no era posible representar números mayores al 3d (011 en binario).

Ahora bien, en relación con los conflictos encontrados en la práctica, es imperativo resaltar dos de ellos. El primero consistía en que el programa no era capaz de buscar dentro de la tabla de valores que fueran distintos al overflow; ya fuera que el resultado de la suma fuera 0, 1, 2 o 3, el programa siempre arrojaría al cero como resultado. Lo anterior se resolvió mediante la creación de dos variables y el manejo del acumulador y de las 2 variables intercaladas en el programa. El segundo conflicto que se presentó en esta práctica y que no fue resuelto del todo, es el resultado mostrado en el display al momento de experimentar con el dip-switch; es

interesante que aunque el proceso de debug del software reflejaba un comportamiento adecuado, al momento de mezclar software y hardware, el resultado final no sea el esperado. Podría pensar que el problema recae en el cableado o en que al momento de programar el microprocesador, está habiendo algún problema.

Fernando Garrote de la Macorra - En esta práctica se realizó un sumador de dos bits. El circuito en esta práctica estuvo más fácil de realizar que en las prácticas anteriores. Esto fue debido a que tengo más conocimiento sobre el cableado que en las prácticas anteriores y que el cableado fue muy similar al de la práctica anterior entonces sólo tuvimos que cambiar dos cables.

El proyecto presentó un conflicto con la programación del 8051. El programa no hacía lo que queríamos que hiciera y fue complicado encontrar el error. Al final el programa se “debuggeo” correctamente pero el display no mostró lo que debía de mostrar. No logramos resolver el problema, una de las razones por las que creemos que no sirvió es porque creemos que el microprocesador se quemó, otra razón por la que podría no servir bien es que hay un problema en el cableado y la otra razón por la que puede no servir es que a pesar de que el programa se “debuggeo” correctamente, puede que haya un error en el programa que realizamos.