Estructura de Computadors II
Primera Pràctica

Maquinari Mapat en Memòria

Martí Paredes Salom
DNI: 43460172Q
Xavier Vives Marcus
DNI: 43462573W

Introducció:

L'objectiu d'aquesta pràctica és repassar els coneixements que tenim de programació en 68K i utilizar el maquinari mapat en memòria.

El funcionament de la pràctica és que a través del hardware simulat per el 68K poguem mostrar en els 7-seg, en text, el número del polsador pitjat.

Subrutines:

MAPINIT:

Comprovam les condicions i depenent del seu valor les duim a terme. La seva execució no és més que varies cridades al TRAP #15 seguit de guardar en memòria les direccions de mapatge.

STR2SEG:

Iniciam un bucle emprant el DBRA de 8 iteracions. Amb cada iteració ens mourem un byte a la dreta, analitzarem de quina lletra es tracta (usant codificació ASCII) i una vegada ho sabem, posar al display de la iteració corresponent (és a dir, si és la primera iteració escriurem al darrer display, si és la segona, al penúltim…) el byte que farà que és representi la lletra al display.

Cal notar que si no trobam quina lletra és, considerarem que no és cap lletra i que no hem d'activar cap segment del display.

BITPOS:

El funcionament d'aquesta rutina és que donat per paràmetre un byte detectar a quina posició es troba el bit menys significatiu. Això ho hem aconseguit amb la instrucció BTST on anam comprovant de dreta a esquerra si en aquella posició hi ha un 1, si n'hi ha un, bota a una etiqueta on posam el número corresponent. Si quan recor els 8 bits no troba ningún 1 posa en la sortida un 8.

Per aquesta subrutina no hem emprat ningún registre extra, el paràmetre el reben a D0 i per a donar la sortida el sobreescrivim.

MAPPRBIT:

Per que aquesta subrutina funcionàs correctament hem hagut de primerament, extreure la informació apuntada per A0, guardar-la dins D0 i seguidament fer un NOT.B D0. Aquest pas és clau perquè la lògica dels polsadors està invertida. És a dir, si no hi ha cap polsador pitjat, a la memòria sortiran tos 1s, en canvi si en pitjam un, per exemple el primer, a la memòria hi haurà un FD (tot 1s i un 0 al final). Feim això perquè la BITPOS ens indica quina és la posició del primer bit a 1. Si no ho féssim la resposta **sempre** seria 1 **excepte** en el cas en el que el primer polsador està pitjat, que en aquest cas seria 2.

Una vegada hem tornat de BITPOS, comprovam quin dels botons està pitjat (com hem explicat abans, el polsador que estigui pitjat serà l'únic que apareixi com un 1 a la memòria) i escrivim el que pertoqui en cada cas, que no suposa més que fer que A0 apunti a la regió de memòria on guardam el text, escriure el text i una cridada a STR2SEG.

Conclusions:

Cal notar que les subrutines més **importants** son **STR2SEG** i **MAPRBIT.** És clar que les altres també ho son però la seva implementació no resulta tan crucial per el funcionament del programa ja que podrien ser implementades directament dins de MAPRBIT.

La pràctica no és complicada en si però la dificultat resideix en el fet de que estam programant en ensamblador. Per tant deixar-se un "#" a l'hora de fer un MOVE o un RTS pot resultar en 40 minuts de maldecaps intentant trobar on és el problema. També saber què hi ha d'haver a la memòria que correspon als displays de 7 segments per cada una de les lletres. No és complicat com a tal però acabes tardan moltíssim.