Uppgift 3.2.4

R24 används som ”in-parameter”. Men vilka övriga register kan användas för samma syfte? Varför kan man i detta fall inte använda något annat register istället för R24?

Svar: Eftersom delayen använder sig av R24 vilket används hela tiden och därför inte kan användas till något annat. Andra register som kan användas är R18-R27 , R0 och R31 ifall dem kallas från C kod annars så måste dem sparas och kallas senare med andra ord call-saved. De övriga såsom r2-r17, r28, r29 kan användas fritt ifall det är assembler kod som kallar på c kod annars så måste dem sparas och återställas senare med andra ord call-used.

Uppgift 5.2.2

Det finns även en annan funktion som påverkar markören på displayen. Med funktionen lcd_set_cursor_pos() anger man var markören ska placeras. Med en kombination av bitvisa operationer skapar man instruktionen som skickas till displayen. Förklara hur de bitvisa operationerna i funktionen lcd_set_cursor_pos() fungerar. Ge gärna exempel på resultatet av bitoperationerna för en valfri rad och kolumn, exempelvis rad 1 (andra raden) och kolumn 2 (tredje kolumnen).

Svar: Man aktiverar den angivna postionen genom att ange rad och kolumn exempelvis lcd_set_cursor_pos(1,0) vilket betyder att vi aktiverar den biten där där cursorn ska sättas, just i detta fallet så är det C0 vilket är andra raden allra första kolumnen. Den sätter biten på den postionen C0.

Uppgift 5.2.4

Beskriv hur funktionen lcd_write_str() fungerar, genom att referera till hur man använder pekaren för att stega igenom teckensträngen.

Svar: lcd_write_str fungerar på det sättet att man skapar en char pekare som går genom strängen. Pekaren går igenom strängen och skriver ut tecknet som pekaren pekar på. Vid varje gång ökar pekaren och skriver ut andra postitionen på strängen osv. Detta körs medan det är sant (pekaren är inom strängen) annars så slutar den att öka pekaren och returnerar teknet.

Uppgift 7.2.3

I början av funktionen deklareras en tecken-array (numbers), som används för att skapa en sträng av siffertecken. Funktionen tillåter endast att maximalt tre siffror kan matas in. Som ni ser så dimensioneras arrayen till att rymma fyra tecken. Vad används den sista positionen till?

Svar: den extra tecknet är för cursorn som sätts på 4 position eftersom när vi skriver så flyttas cursorn tecknet +1 åt sidan och därför befinner sig där. Utan den så hade vi alltid haft cursor på 3:e talet och därför funnits över det talet som skulle representera siffran position 3.

Uppgift 8.2.5 (redovisas i rapport)

Redogör för era erfarenheter och kunskaper från denna laboration (minst en halv A4-sida):

- Vad har ni lärt er?
- Om ni får välja en sak, upplevde ni något som var intressant/givande?
- Fanns det något som upplevdes som svårt?
- Gick allting bra eller stötte ni på problem? Om allting gick bra, vad var i så fall anledningen detta? Om ni stötte på problem, hur löste ni i så fall dem?

Med denna laboration så har jag lärt mig grunderna för c programmering samt hur man kan kombinera assembler med c programmering för att använda gamla skrivna funktioner i c.

Denna labben har lärt mig hur olika loopar fungerar samt felsöka koden för att finna felet i det samt all förståelse för dem olika metoderna och funktionerna.

En sak som jag upplevde givande och intressant var programmeringen av tangentbordet i input_int metoden samt skriva ut en sträng med hjälp av write metoden.

Delvisa moment som i dem nästsista uppgifterna, då man skulle komplettera metoderna i en

loop och använda sig av input_int och använda pekaren för att det ska fungera. Andra svåra saker vad i början då man inte visste hur man inkluderar assembler filer genom .global. Dessa fick jag lösa genom sökning på nätet och föreläsningar men även filer som finns ute på itslearning. Annars gick allting bra eftersom jag läst på lite på föreläsningarna och repeterat saker för att förstå dem bättre.