02321 Hardware/Software Programmering (E10)

Home assignment 6

Morten Hillebo (s072923)
Kim Rostgaard Christensen (s084283)

Problem 7.6

A	x300A
B	x3005
C	x300B
D	x3001
E	x3004
F	x3007

Label	Asm	Bin
D	LD R1,+8	0010 001 000001000
E	ADD R1,R1,#-1	0001 001 001 1 11111
F	BRp -3	0000 001 111111101

Problem 8.2

Why is a ready bit not needed if synchronous I/O is used?

In synchronous I/O we assume that data is always available, at the predefined monotonic rate. If the keyboard skipped a cycle, the char from the last cycle would be read - assuming that the register is not cleared.

Problem 8.6

What problem could occur if a program does not check the Ready bit of the KBSR before reading the KBDR?

The result would be that you effectively did synchronous I/O, as you assume data is always available. The problem would be that you would not be able to distinguish a press on 'a' and then 'a' from one press on 'a' for example.

Problem 8.10

What problem could occur if the display hardware does not check the DSR before writing to the DDR?

The DSR indicates that the display is done updating from the DDR. If you write to DDR while it is being read by the display, the data could be corrupted.

Memory mapped IO with polling

```
1  .ORIG x3000
2
3  START   LD      R2,Newline
4  L1      LDI     R3,DSR
5          BRzp    L1              ; Loop until Monitor is ready
6          STI     R2,DDR          ; Move cursor to new clean line
7  ;
8          LEA     R1,Prompt       ; Starting address of prompt string
9  Loop    LDR     R0,R1,#0        ; Write the input prompt
10         BRz     Input           ; End of prompt string
11 L2      LDI     R3,DSR
12         BRzp    L2              ; Loop until Monitor is ready
13         STI     R0,DDR          ; Write next prompt character
14         ADD     R1,R1,#1        ; Increment Prompt pointer
15         BRnzp   Loop            ; Get next prompt character
16 ;
17 Input   LDI     R3,KBSR
18         BRzp    Input           ; Poll until a character is typed
19         LDI     R0,KBDR         ; Load input character into R0
20
21         LD      R2,Newline ; Put newline
22 NL      LDI     R3,DSR
23         BRzp    NL              ; Loop until Monitor is ready
24         STI     R2,DDR          ; Move cursor to new clean line
25 ;
26
27         LEA     R1,Number       ; Starting address of number string
28 LoopN   LDR     R4,R1,#0        ; Write the string
29         BRz     L3              ; End of string
30 L2N      LDI     R3,DSR
31         BRzp    L2N              ; Loop until Monitor is ready
32         STI     R4,DDR          ; Write next character
33         ADD     R1,R1,#1        ; Increment char pointer
34         BRnzp   LoopN           ; Get next character
35
36 L3      LDI     R3,DSR
37         BRzp    L3              ; Loop until Monitor is ready
38         STI     R0,DDR          ; Echo input character
39
40
41 ; Check if number is even
42         AND R0,R0,#1
43         BRz     iseven
44
45 isodd
46         LEA     R1,Odd    ; Starting address of odd string
47 LoopO   LDR     R4,R1,#0        ; Write the string
48         BRz     START           ; End of even string
49 L2O     LDI     R3,DSR
50         BRzp    L2O              ; Loop until Monitor is ready
51         STI     R4,DDR          ; Write next character
52         ADD     R1,R1,#1        ; Increment char pointer
```

```
53          BRnzp   LoopO       ; Get next character
54          BRnzp   START
55  iseven
56          LEA     R1,Even     ; Starting address of odd string
57  LoopE   LDR     R4,R1,#0    ; Write string
58          BRz     START       ; End of odd string
59  L2E     LDI     R3,DSR
60          BRzp    L2E          ; Loop until Monitor is ready
61          STI     R4,DDR      ; Write next character
62          ADD     R1,R1,#1    ; Increment char pointer
63          BRnzp   LoopE       ; Get next character
64          BRnzp   START
65  ;
66
67  DSR     .FILL   xFE04
68  DDR     .FILL   xFE06
69  KBSR    .FILL   xFE00
70  KBDR    .FILL   xFE02
71  Newline .FILL   x000A       ; ASCII code for newline
72  Prompt  .STRINGZ "Input a number:"
73  Number  .STRINGZ "Number "
74  Even    .STRINGZ " is even"
75  Odd     .STRINGZ " is odd"
76  .END
```