# Lab 3 Report

**3200105515 颜晗**

This lab is about the interrupt. We need to know how to prepare for generating an interrupt request and how can we return to user program after we handled the interrupt (we need to store the data of which registers). Because we need to distinguish the input characters (numbers are handled differently from the rest of characters), we could use many subroutines and BR instructions and the point is that the number of registers is too few so we need to decide how to use registers properly.

**Arithmetic**

We need to print 40 characters each line continuously and infinitely. There will be three parts. In the system booting code, we will give the keyboard privilege of interrupt which means the KBSR should be set to x4000 at first. There is an interrupt vector table like trap vector table. We should put the starting address (x0800) of the interrupt routine we write in the right address (x0180).

In the user program, we need to print the number continuously. We could use the register R5 as a counter. Each time we print a character, the number in R5 add 1 and store the result in R5. When the number is 40, we start a newline and clean R5. The implementation of the comparison depends on subtraction.

In the interrupt service routine, we need change what we should print. That is, change R0. We also need to print character which is not number in this routine. The important thing is that because we may run DELAY in user program when we handle the interrupt and there are also subroutines in interrupt service routine, so we need to store the R7 in one location and restore it before we RTI.

**Code**

1. LD     R1,INT           ;键盘中断地址
2. LD     R2, INT\_START
3. STR    R2, R1,#0         ;中断权限
4. LD     R2, KB\_EN
5. STI    R2, KBSR2
6. INT         .FILL   x0180
7. INT\_START   .FILL   x0800
8. KBSR2       .FILL   xFE00
9. KB\_EN       .FILL   x4000

These are the code for preparing. Put x0800 in x0180. Put x4000 in KBSR (xFE00).

1. GETC                    ;获取字符
2. ADD   R3, R0, #-10      ;enter判断
3. BRz   FUN1
4. LD    R4, INF           ;通过48，57的比较（减法）
5. ADD   R3, R0, R4        ;结果正负跳转至不同函数
6. BRn   FUN3              ;R3存结果，R4存比较数
7. LD    R4, SUB
8. ADD   R3, R0, R4
9. BRp   FUN3
10. BRnz  FUN2

We need to get characters and distinguish them. The ascii value of 0 and 9 are 48 and 57. We just make some subtraction and check the condition code then we could decide we need to BR to which line and run which subroutine. FUN1 is for enter. FUN2 is for a new number. FUN3 is for other characters.

As for user program, just print and don’t need to pay much attention.

**The question of TA.**

1.Introduce the PSR.

Bit [15] specifies the privilege, 1 means USER mode and 0 means supervisor privilege. Bits [10:8] specify the priority level of the program. Bits [2:0] contain the condition code which show us the number we get recently is negative, zero or positive.

2. What the data in Supervisor Stack show when we handle an interrupt?

The data(stored in location x2FFF and x2FFE) specify PSR and PC of user program.