CSE 379
Lab 5
Spring 2020

Presented by:
Dazhou Liu
Xiaoang Zhang

March 9 2020

Summary

In lab5, we get to learn how to use interrupt in our program and how to clear interrupt. This is not a hard understanding lab if you finish GPIO, UART in lab4 and lab3. We need to initialize interrupt first then find the Nested Vector Interrupt Controller. After initializing all the stuffs, we need to build a handler mode to deal with the interrupt. Mode is returned to what it was when interrupt occurred and control is transferred back to interrupted program.

Description

Division of Work

There are three subroutines along with lab5 that handles the program. Necessary subroutines developed in previous labs are included in *lab_5_library.s*. *Switches_Handler* and lab5.s are done by Dazhou Liu. *UART0_Handler* and *interrupt_init* are done by Xiaoang Zhang. Organizing memories in lab5 library and putting *Switches_Handler* and *UART0_Handler* into lab5 library are done by Dazhou Liu. The rest of lab5 library is done by Xiaoang Zhang.

Purpose of the Program

Objective

The objective of lab5 is to learn the difference between normal program and interrupt and how to service interrupt when it occurs. When an interrupt occurs, flow of program is transferred to interrupt. After finishing handling it, the program must be transferred back to normal program.

Debugging Steps

Three steps are implemented for debugging the program. The elementary step for debugging is building project. After building project, the syntax errors as well as missing items are fixed. The secondary step is to deal with run time errors. In lab5, the main point is the flow of program. Specifically, the program will be diverted to interrupt handler outside the normal program when a

interrupt from keyboard or switches is received. After finishing servicing the interrupt, the flow must be back to the normal program. To verify this flow, break points are set up at the beginning of the two handlers to verify the location of the program after receiving the interrupt. After this step is verified comes the third step. The number of times an interrupt occurs is displayed on Putty so that the result can be analyzed and verified. If the result is not as expected, break point will be implemented to go through the program. The values in registers and memories will also be analyzed and verified with assistance of break points.

Logic

Summary of Flowchart

Four subroutines will be provided. Three of them are subroutines that handle interrupts and one of them is the main program that executes the program. The flowcharts provided are for *UART0_Handler*, *Switches_Handler*, *interrupt_init* and lab5.

Summary of each subroutine

interrupt_init sets up UART0 and switches to interrupt.

UART0_Handler clears and handles interrupt for UART0 and transfer back flow to normal program.

Switches_Handler clears and handles interrupt for switches and transfer back flow to normal program.

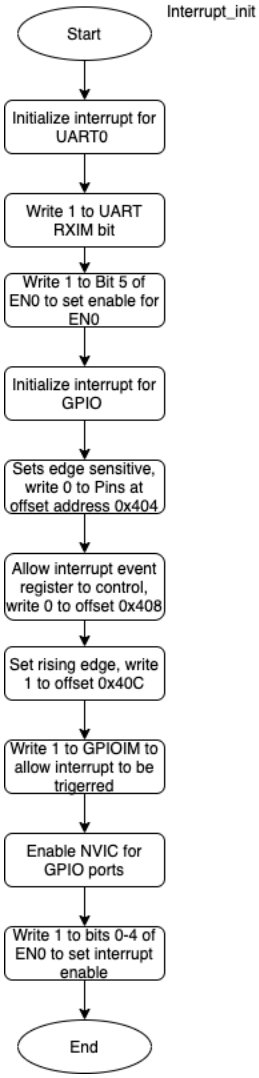
lab5 executes the program as main function.

Note

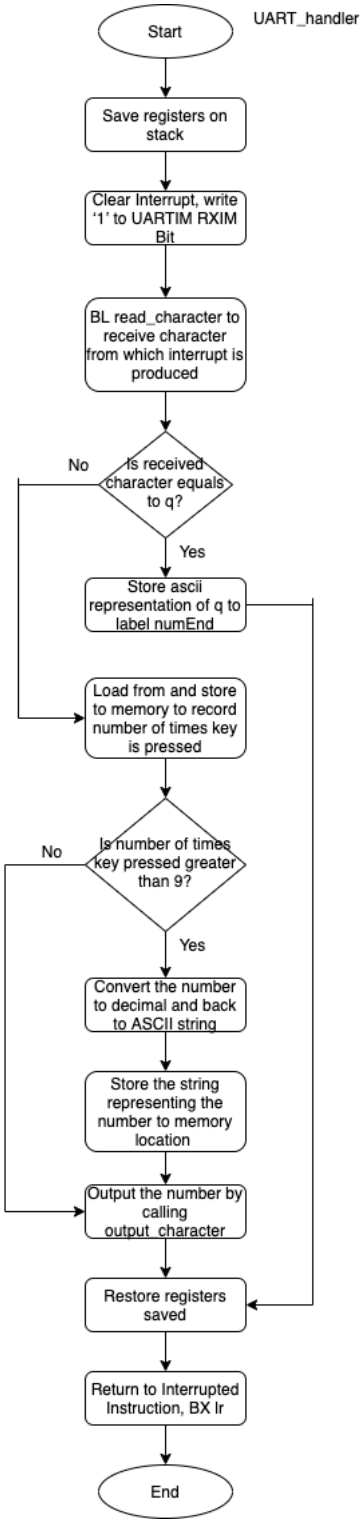
For the switches on ARM board, multiple contacts can be made by single human press due to unstable physical connections between the switches and underlying circuits. This makes displays on Putty unstable. To optimize this situation, button debouching is implemented. Two loops that each iterates 9000 times are deployed in *Switches_handler*. By doing this, this flow is slowed down so that the multiple contact under single press can be blocked temporarily.

Flowchart

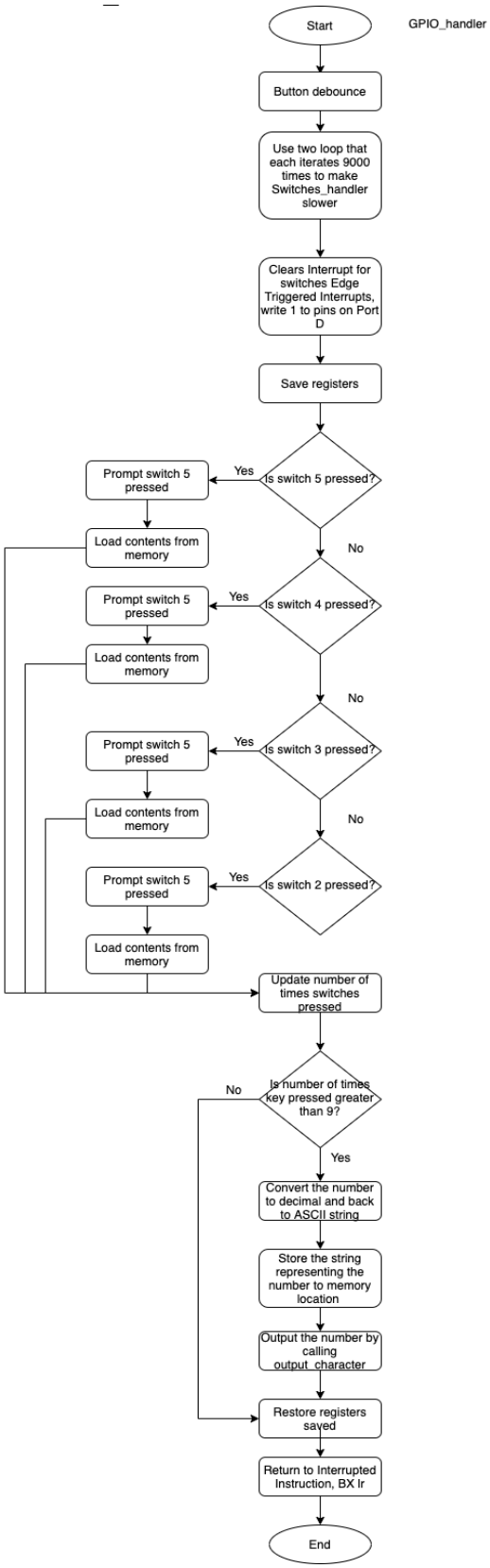
Interrupt_init



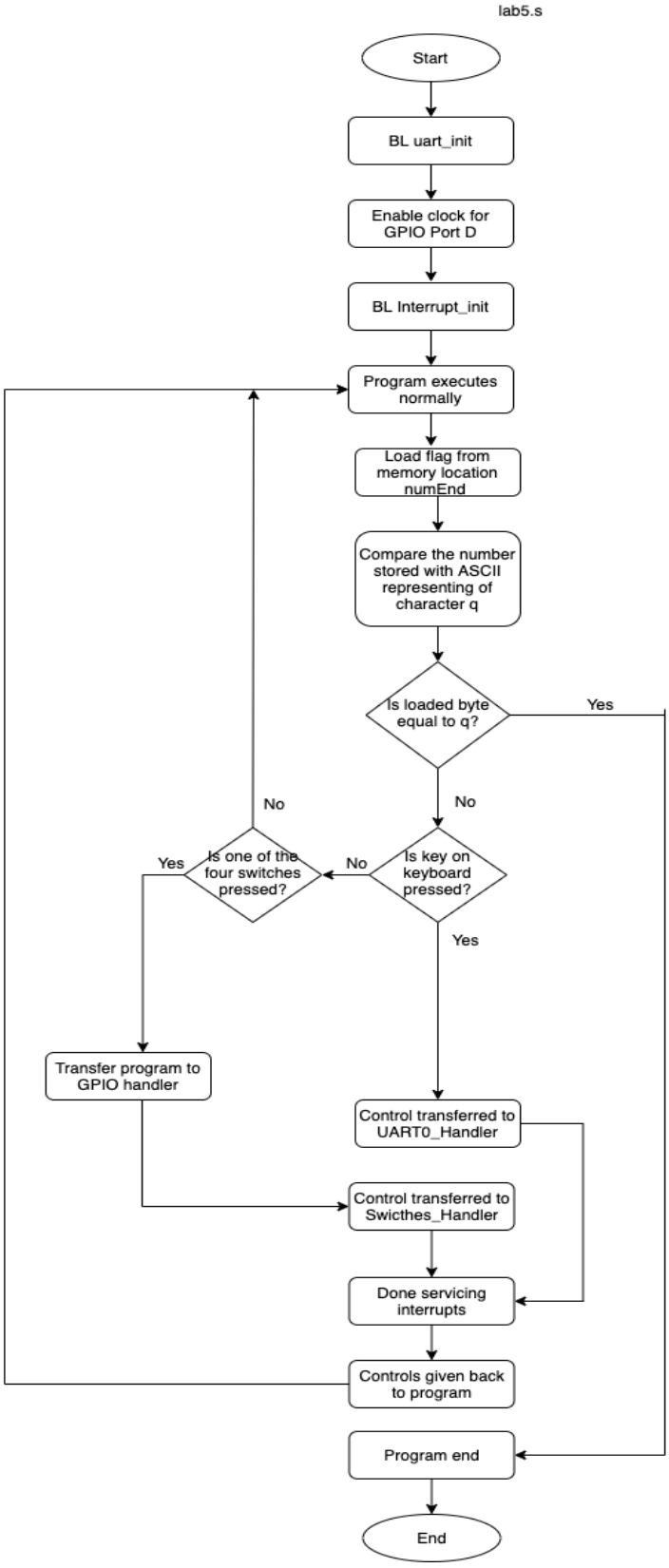
UART_handler



Switches_handler



lab5.s


lab5.s
Start
BL uart_init
Enable clock for GPIO Port D
BL Interrupt_init
Program executes normally
Load flag from memory location numEnd
Compare the number stored with ASCII representing of character q
Is loaded byte equal to q?
Yes
No
Is key on keyboard pressed?
No
Is one of the four switches pressed?
Yes
No
Yes
Transfer program to GPIO handler
Control transferred to UART0_Handler
Control transferred to Swicthes_Handler
Done servicing interrupts
Controls given back to program
Program end
End