Spring 2019

California State University, Northridge

Department of Electrical and Computer Engineering

Lab Experiment 8: Writing and Calling Subroutines

April 07, 2019

Professor: Neda Khavari

Authors:
Ridge Tejuco
Dan Paul Rojas

Introduction

The purpose of this experiment is to become familiar with the use of subroutines and the steps needed to call a subroutine and return from it within other code. The experiment covers the instructions LDM and STM as a way to move blocks of data to and from the stack. The lab uses a full descending stack, which is used to save necessary registers.

The experiment also revisits the concept of debouncing by calling the debouncing subroutines from another file.

Procedure

For all tasks, additional files for the delay subroutine were used. In the following code, the delay label was declared global and the registers R0 and LR were saved to the stack

```
        GLOBAL delay
        AREA subroutine,CODE,READONLY
delay
        STMFD SP!, {R0,LR}                    ; stores R0
```

Then a for loop was used to instantiate the delay.

```
        delays
        SUBS R0,R0,#1
        BNE delays
```

Lastly, the registers were restored.

```
        LDMFD SP!, {R0, PC}
END
```

This code was then ported to C programming language.

```
void delay(int r0)
{
   int d;
   for (d = 1; d <= r0;d++) {}
}
```

The goal in task 1 and 2 was to output a running counter with a delay in between each LED change.

In order to call the delay subroutine, the file was imported, and PRESERVE8 was declared for the subroutines declared in c programming language.

```
        IMPORT delay
        PRESERVE8
```

The following code was used to set up the pins in GPIO mode and their direction as output.

```
;set pinmode to GPIO
        mov R0,#0
        LDR R1,=PINSEL0
```

```
        STR R0,[R1]
;set pin direction to outputs
        MOV R0,#0xFF00
        LDR R1,=IO0DIR
        STR R0,[R1]
```

Then the LEDs were all turned off initially.

```
OFF
        ;turn off all LEDs
        MOV R0,#0xFF00
        LDR R1,=IO0SET
        STR R0,[R1]
```

Before entering the loop, the counter and some shifting masks were set up. R3 was used to turn on the current LED, while R4 was used to turn off the last LED.

```
        MOV R2,#0           ;R2 is counter
        MOV R3,#0x100     ;R3 is shift mask to turn on current
        MOV R4,#0x80
```

Upon entering the loop, a delay was called. Depending on which file was included in the project, the function in delay.s or delay.c would be called.

```
Loop
        LDR R0,=PERIOD
        BL delay
```

If the counter was equal to 8, the next instruction would jump back to OFF.

```
        CMP R2,#8
        beq OFF
```

Else, the program would continue to turn on the next LED

```
        LDR R1, =IO0CLR
        STR R3,[R1]
```

If the counter was greater than 1 then the last LED would be turned off.

```
        CMP R2,#1
        LDR R1,=IO0SET
        STRGT R4,[R1]
```

At the end of the loop, both mask were shifted left once and the counter was incremented.

```
        LSL R3,R3,#1
        LSL R4,R4,#1
        add R2,R2,#1
        b LOOP
stop b stop
        end
```

For task 3, the goal was to toggle the first 4 LEDs on and off.
Just like task 1 and 2, the pins were set to GPIO, though only the first 4 leds were set as input ,and then the LEDs were turned off.

```
        MOV R0,#0x0F00      ;Only the first 4 leds
        LDR R1,=IO0DIR
        STR R0,[R1]
```

After turning the lights off, the program would enter a loop to constantly check if the button p0.14 was pressed. If it was pressed it would enter a delay before the next loop.

```
TOGGLE      ;check if button is pressed
        LDR R1,=IO0PIN
        LDR R2,[R1]
        TST R2,#0x4000
        BNE TOGGLE
        BL delay
```

After the debouncing delay, the next loop would check if the button was let go.

```
TOGGLE_2    ;check if button is let go
        LDR R1,=IO0PIN
        LDR R2,[R1]
        TST R2,#0x4000
        BEQ TOGGLE_2
        bl delay          ;debounce again
```

If the button was let go, the program would debounce again, turn the LEDs on, and then loop back to TOGGLE to check for additional button presses.

```
ON      ;turn on lights
        MOV R3,#1; next state OFF
        LDR R1,=IO0CLR
        MOV R0,#LIGHTS
        STR R0,[R1]
        b TOGGLE ; GO Back to checking button
stop b stop
        end
```

Results

Both task 1 and task 2, both a running counter starting at 0th to the 8th LED. Each LED would light up once at a time with a second delay in between each. This is the expected result
The delay was successful for both the C and ARM language.
For task 3, after flashing the program, the LED would not toggle and stay off. Through debugging, the issue was determined to be a extremely long delay.
After fixing this issue, the first 4 LEDs were initially off, and pressing p0.14 would toggle the LEDs on and off. This is the expected result.

Conclusions

The experiment was a success as the delay subroutine was called in from another file. The subroutine was successfully called through c programming language and in ARM assembly. Looking at the stack verified that the registers used in the subroutines were saved on the stack and then restored to the appropriate registers after returning from the subroutine.

```
;task3.s
        GLOBAL user_code
PINSEL0 EQU 0xE002C000
IO0DIR EQU 0xE0028008
IO0PIN EQU 0xE0028000
IO0SET EQU 0xE0028004
IO0CLR EQU 0xE002800C
LIGHTS EQU 0x0F00
PERIOD EQU 0x30000
        PRESERVE8
        AREA mycode, CODE, READONLY
user_code
        IMPORT delay
        mov R0,#0               ;set pinmode to GPIO
        LDR R1,=PINSEL0
        STR R0,[R1]

        MOV R0,#LIGHTS          ;set pin direction to outputs;leave 14 as input
        LDR R1,=IO0DIR
        STR R0,[R1]
OFF
        MOV R3,#0               ;sets next state to ON
        MOV R0,#LIGHTS          ;turn off the 4 LEDs
        LDR R1,=IO0SET
        STR R0,[R1]
TOGGLE                          ;check if button is pressed
        LDR R1,=IO0PIN
        LDR R2,[R1]
        TST R2,#0x4000
        BNE TOGGLE
        LDR R0,=PERIOD
        bl delay
TOGGLE_2                        ;check if button is let go    LDR R1,=IO0PIN
        LDR R2,[R1]
        TST R2,#0x4000
        BEQ TOGGLE_2
        bl delay                ;debounce again

        CMP R3,#1 ;R3 = 1 => B OFF
        BEQ OFF    ; R3 = 0 => B ON
ON                              ;turn on lights
        MOV R3,#1               ;next state OFF
        LDR R1,=IO0CLR
        MOV R0,#LIGHTS
        STR R0,[R1]
        b TOGGLE                ;GO Back to checking button
stop b stop
   End
```

```
;task12.s
        GLOBAL user_code
        IMPORT delay
PINSEL0 EQU 0xE002C000
IO0DIR EQU 0xE0028008
IO0SET EQU 0xE0028004
IO0CLR EQU 0xE002800C
OUTPUT EQU 0xFF00
PERIOD EQU 0x30000
        PRESERVE8
        AREA mycode, CODE, READONLY
user_code
        mov R0,#0           ;set pinmode to GPIO
        LDR R1,=PINSEL0
        STR R0,[R1]
        MOV R0,#OUTPUT      ;set pin direction to outputs
        LDR R1,=IO0DIR
        STR R0,[R1]
OFF
        MOV R0,#OUTPUT      ;turn off all LEDs
        LDR R1,=IO0SET
        STR R0,[R1]

        MOV R2,#0           ;R2 is counter
        MOV R3,#0x100       ;R3 is shift counter
        MOV R4,#0x80
LOOP
        LDR R0,=PERIOD
        BL delay
        CMP R2,#8           ;check counter = 8
        beq OFF

        LDR R1, =IO0CLR     ;turn on current led
        STR R3,[R1]

        CMP R2,#1           ;turn off last led
        LDR R1,=IO0SET
        STRGT R4,[R1]

        LSL R3,R3,#1                ;increment counter and shift R3
        LSL R4,R4,#1
        add R2,R2,#1
        b LOOP
stop b stop
   end
```

```
;delay.s
        GLOBAL delay
        AREA subroutine,CODE,READONLY
delay
        STMFD SP!, {R0,LR}          ; stores R0 which holds duration of delay in main program
delays
        SUBS R0,R0,#1
        BNE delays

        LDMFD SP!, {R0, PC}         ; loads previously stored values from stack
        END
```

```
;delay.c
void delay(int r0)
{
   int d;
   for (d = 1; d <= r0;d++)
   {}
}
```