Computer Systems Week 9 Lab

Daniel Coady (102084174)

09/10/2019

How the Code Works

kernel7.asm

In this source file we simply get the factorial of 4 and then flash the LED that many times.

TIMER.asm

In this source file we simply wait the amount of time passed through to the TIMER function.

factorialj.asm

In this source file we recursively calculate the factorial of the number passed through to the FACTORIAL function.

Program Input

The input to the program is in r1. This value is then used to get a factorial which is used as the amount of flashes to be displayed by the LED connected to GPIO18.

My Modified Code

kernel7.asm

```
BASE = $3F000000

;Calculate
mov r1,#4 ; input
mov sp,$1000 ; make room on the stack
mov r0,r1
bl FACTORIAL
mov r7,r0 ; store answer
bl init ; initialise GPIO base address and GPIO18 for output

mov r1, r7 ; store amount of flashes
bl flash ; flash the LED
wait:
b wait

include "TIMER.asm"
include "factorialj.asm"
include "INIT.asm"
include "FLASH.asm"
```

TIMER.asm

```
; TIMER
; no args
TIMER_OFFSET = $3000
DELAY = $20000
TIMER:
  mov r0, BASE
  orr r0, TIMER_OFFSET ; timer address
  mov r1, DELAY ; store delay
  ldrd r2, r3, [r0, #4] ; store the start time
  mov r4, r2 ; store least significant 32-bits of start time in r4

  timerdelay:
    ldrd r2, r3, [r0, #4] ; store the current time
    sub r5, r2, r4 ; r5 = current time - start time
    cmp r5, r1
  bls timerdelay ; goto timerdelay if r5 <= delay
bx lr
```

factorialj.asm

```
FACTORIAL:
sub r1,r1,#1
cmp r1,#1
beq EXIT
mul r0,r0,r1
push {r1,lr}
; push onto the stack without changing the stack pointer
bl FACTORIAL   ; call FACTORIAL
EXIT:
pop {r1,lr}  ; pop off the stack
bx lr ; RETURN
```

INIT.asm

```
; init function -- sets up GPIO18 for writing and retrieves GPIO base address
; no arguments
; returns BASE + GPIO_OFFSET
GPIO_OFFSET = $200000
init:
  mov r0,BASE
  orr r0,GPIO_OFFSET
  mov r1,#1
  lsl r1,#24
  str r1,[r0,#4] ; set GPIO18 to output
bx lr
```

FLASH.asm

```
; flash function -- will flash the LED on GPIO18 a set amount of times
; r0 = GPIO base address
; r1 = num of times to flash
; returns nothing
flash:
  loop$:
    mov r2,#1
    lsl r2,#18
    str r2,[r0,#28]  ; turn LED on
    mov r3,$0F0000

    push {lr, r0, r1, r2, r3, r4, r5}
    bl TIMER
    pop {lr, r0, r1, r2, r3, r4, r5}
    mov r2,#1
    lsl r2,#18
    str r2,[r0,#40]  ; turn LED off
    mov r3,$0F0000

    push {lr, r0, r1, r2, r3, r4, r5}
    bl TIMER
    pop {lr, r0, r1, r2, r3, r4, r5}

  sub r1,#1
  cmp r1,#0
  bne  loop$  ; end of outer loop. Runs r1 times
bx lr
```