Lab09-Nguyen Manh Duc-103792724

timer.asm

It loads the r2 register from the previous code with a value, subtracts one from it, and compares it to zero. If the number reaches zero, the program will terminate; otherwise, the process will be repeated until the number reaches zero.

factorialj.asm

It takes the values 4 in registers r0 and r1 from kernel7.asm and subtracts one from r1 before comparing it to one; if it is true, it will stop; otherwise, it will multiply r0 and r1 and store the result in r0, which was 3(r1)*4(r0) in the first case. As a result, if we repeat it until r1 becomes one, we will end up with 4! in r0.

Kernel7.asm

It sets the GPIO pin to turn on and off and loads the initial value for r0,r1 so that it can be passed to factorial7.asm.

Kernel7.asm file

```
;Calculate
mov r1,#4;input
mov sp,$1000; make room on the stack
mov r0,r1
bl FACTORIAL
mov r7, r0;store answer
BASE = $3F00000; RP2 and RP3; GPIO_SETUP
Movr0, BASE
Bl SETUP_LED
mov r0,BASE
mov r1, r7
bl FLASH
wait:
```

```
b wait
include “timmer2_Param.asm”
include “factorialj.asm”
include “GPIO.asm”
timmer2_Param.asm file
;timer2_Param
Delay: ;this function has 2 parameter
TIMER_OFFSET=$3000
mov r3, r0;BASE – depends
orr r3,TIMER_OFFSET
mov r4, r1;$80000 passed as a parameter
Idrd r6, r7,[r3,#4]
mov r5,r6
loopt1: ;label still has to different from one in_start
Idrd r6,r7,[r3,#4]
sub r8,r6,r5
cmp r8,r4
bls loopt1
bx lr; return
factorialj.asm file
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
```

GPIO.asm file

```
SETUP_LED
GPIO_OFFSET = $200000
orr r0,GPIO_OFFSET
mov r1,#1
lsl r1,#24
str r1,[r0,#4] ; set GPIO1 to output
bx lr
FLASH:
;r0 = base
mov r2,r0
;r1 = number of flashes
orr r0,GPIO_OFFSET
mov r7,r1
loop$
mov r1,#1
lsl r1,#18
str r1,[r0,#28] ;turn LED on
push {r0,r1,r7,lr}; r0,r1,r7 in use push and then set parameters
mov r0,BASE
mov r1,$0F0000
bl Delay
pop {r0,r1,r7,lr}
mov r1,#1
lsl r1,#18
str , [r0,#40]; turn LED off
```

```
push{r0,r1,r7,lr}; r0,r1,r7 in use push and then set parameters
mov r0,BASE
mov r1,$0F0000
bl Delay
pop{r0,r1,r7,lr}
sub r7,#1
cmp r7,#0
bne loop$; end of outer loop. Runs r7 times
bx lr
```