|  |
| --- |
| SWInburne university |
| LAB 09 |
| COS10004 – Computer System |
|  |
| **NGO CONG THANH** |
| **11/18/2021** |

|  |
| --- |
| SWINBURNE UNIVERSITY |

**Kernel7.asm:**

Initializing base, set function to the GPIO 18, set value to GPIO 18 to turn the light on or off, call timer function and factorial from another file.

**Factorialj.asm:**

It take r1 and r0 form the kernel7.asm file then subtract #1 from r1, and compare r1 with #1. Whether it is true, it will stop. After that, r0 will be multiplied with r1 and store the value I r0. This will continue when r1 reach 1, we have the value factorial of 4.

**TIMER.asm:**

Set up timer function, get r2 from kernel, subtract 1 from r2, and then compare r2 with 0. If r2 is not equal to #0, it will loop back until r2 becomes 0.

FASARM Code:

Kernel7.asm:

;Calculate

mov r1,#4 ;input

mov sp,$1000 ;make room on the stack

mov r0,r1

bl FACTORIAL

mov r7,r0 ;store answer

BASE = $3F000000 ;RP2 and RP3 ;GPIO\_SETUP

mov r0,BASE

bl SETUP\_LED

mov r0,BASE

mov r1,r7

bl FLASH

wait:

b wait

include "TIMER.asm"

include "factorialj.asm"

include "GPIO.asm"

TIMER.asm:

;TIMER - dumb timer

;r2=number of loops

TIMER:

wait1$:

sub r2,#1

cmp r2,#0

bne wait1$

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

GPIO.asm:

SETUP\_LED:

GPIO\_OFFSET = $200000

orr r0,GPIO\_OFFSET

mov r1,#1

lsl r1,#24

str r1,[r0,#4]

bx lr

FLASH:

mov r2,r0

orr r0,GPIO\_OFFSET

mov r7,r1

loop$:

mov r1,#1

lsl r1,#18

str r1,[r0,#28]

mov r1,#1

lsl r1,#18

str r1,[r0,#40]

push {r0,r1,r7,lr}

mov r0,BASE

mov r1,$0F0000

bl TIMER

pop {r0,r1,r7,lr}

sub r7,#1

cmp r7,#0

bne loop$

bx lr

TIMER2.asm:

Delay: ;this function has 2 parameters

TIMER\_OFFSET=$3000

mov r3,r0 ;BASE - depends on Pi model

orr r3,TIMER\_OFFSET

mov r4,r1 ;$80000 passed as a parameter

ldrd r6,r7,[r3,#4]

mov r5,r6

loopt1: ;label still has to be different from one

in \_start

ldrd r6,r7,[r3,#4]

sub r8,r6,r5

cmp r8,r4

bls loopt1

bx lr ;return