ECE3221 Lab 2 Report

Stephen Cole

3553803

Assembly Code:

.global \_start

\_start:

br Start # goto local code

/\*\*\*

ECE3221 LAB#2 –SEQUENTIAL CIRCUIT EMULATION (2)

-----------------------------------------------

DATE: NAME:

-----------------------------------------------

Starting point for Lab2, generates a period 15

pseudo-random sequence shown on the red LEDs

-----------------------------------------------

PORT MAP

0x10000000 = 16-bit output port = RED LEDS

0x10000010 = 16-bit output port = GREEN LEDS

0x10000040 = 16-bit input port = SWITCHES

-----------------------------------------------

\*\*\*/

.org 0x100 # starting address

Start: #

movia r8,0x10000050 # r8 = address of push button

movia r9,0x10000020 # r9 = address of seven seg

movia r10,0x10000040 # r10 = address of switches

movia r11,0x10000000 # r11 = address of red leds

movia r12,0x10000010 # r12 = address of green leds

movia r13,0x00000007

ori r3,r0,1 # r3 = shift register initialized

movi r4,reg

stw r3,(r4)

andi r7,r0,0 # r7 = number of loops

ori r18,r0,8 # r7 = number of loops

movia r16,counts\_per # set top of array

stwio r0,(r9) # set hex to blank

andi r22,r22,0 # set hex register to 0

reset\_array:

stw r0,(r16)

addi r7,r7,1 # increment loop count

addi r16,r16,4 # increment index of array

bne r18,r7,histogram # branch to histogram if r7 is not 8

top:

ori r7,r0,4 # r7 = number of loops

movia r4,reg

ldw r3,(r4)

loop4:

#-------------------------------------------

andi r6,r3,16384 # r6 = bit14 only

srli r6,r6,14 # r6 = bit14 only moved to lsb

andi r5,r3,1 # r5 = bit0 only

xor r6,r6,r5 # new bit0 = bit0 XOR bit14

#-------------------------------------------

slli r3,r3,1 # do one shift

xor r3,r3,r6 # insert new bit0

addi r7,r7,-1 # decrement loop count

bne r7,r0,loop4 # branch to loop4 if r7 is not zero

#-------------------------------------------

movia r4,reg

stw r3,(r4)

stwio r3,(r11) # r3 --> red leds

andi r3,r3,7 # mask 3 lsb

ori r7,r0,4 # reset loop counter to 4

#loop if

beq r7,r0,top # branch to top if r7 is zero

beq r7,r13,top # branch to top if r7 is 7

#-------------------------------------------

stwio r3,(r12)

#-------------------------------------------

andi r7,r0,0 # r7 = number of loops

ori r18,r0,8 # r7 = number of loops

movia r16,counts\_per # set top of array

movia r20,lookup\_table # set top of lookup\_table

ldbio r23,(r8) # load byte from switches

bne r23,r0,histogram # if switch is not pressed display

br top

histogram:

ldw r17,(r16)

cmpeq r19,r3,r7 # if r3 == r7 cmpeq = 1

add r17,r17,r19 # increment r17 if r3 == r7

stw r17,(r16)

bne r19,r0,lookup\_hex # branch unless r19 == 0

addi r7,r7,1 # increment loop count

addi r16,r16,4 # increment index of array

bne r18,r7,histogram # branch to histogram if r7 is not 8

lookup\_hex:

beq r7,r0,display\_hex # if value is 0 go to display

addi r20,r20,1 # increment to find correct 7seg value

addi r7,r7,-1 # decrement loop count

bne r7,r0,lookup\_hex # branch to loop4 if r7 is not zero

display\_hex:

ldb r21,(r20)# load 7seg hex from lookup table

slli r22,r22,8 # move last 7seg value over

or r22,r22,r21 # add new value

andi r22,r22,0xFFFF # force all but last two values to low

stwio r22,(r9) # send to 7seg

br top

reg: .word 8

counts\_per: .word 0,1,2,3,4,5,6,7

lookup\_table: .byte 0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07

Is the game fair? Also comment on how many counts accumulate in each bin over a short period of time; how many dice throws per second are happening?

The game seems to be fair, the counts in the histogram are all roughly the same after running for a very short period of time. The counts accumulate at a very high rate, tracking how many times each histogram bin was written to I was able to estimate that there are roughly 51,000 dice rolls per second.