UCSC Extension – Digital Design with FPGA

**Homework #8**

Jae-Yang Park ([jaeyangp@gmail.com](mailto:jaeyangp@gmail.com))

PicoBlaze hardware / software implementation

**Problem description**

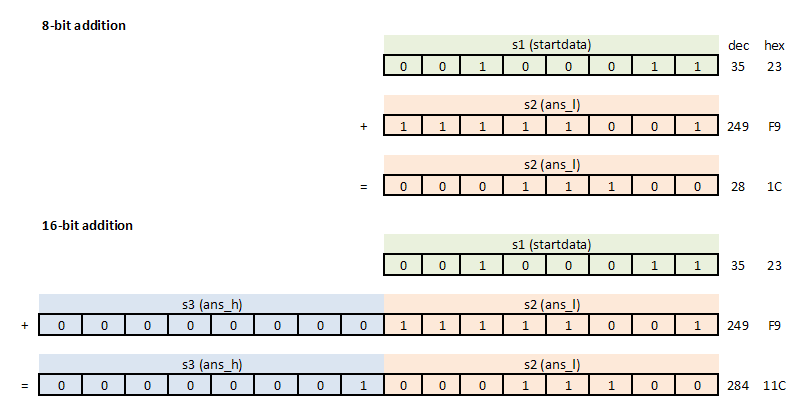
1. Start with a number, 4
2. Increment the number by 35
3. Display the number on the board at a time interval of ‘n’ second

Ex) 4, 39, 74, 109, 144,…, etc

1. Output the above results on the LED’s

8 bit addition (ex. 249 (f9h) + 35 (23h) = 284 (11ch), the result is 9 bits and exceeded 8 bits)

On the 8-bit addition, the result is wrong when the result over 255, but on the 16-bit addition, the result is correct.



**How to do 16-bit addition**

add\_num1:

load startdata, INC\_NUM ; increment at a distance of 35

**add ans\_l, startdata ; add LSB with startdata**

**addcy ans\_h, 00 ; add MSB with carry**

return ; {ans\_h, ans\_l} is complete 16 bits result

**Send results through 8-bit output port**

; send results through output port

display:

**output ans\_l, OUTPUTPORT\_L ; LSB**

**output ans\_h, OUTPUTPORT\_H ; MSB**

return

**Receive 16-bit result at RTL**

**assign out\_port\_reg = {out\_port\_reg\_h, out\_port\_reg\_l};**

always @(posedge CLK1) begin

**if (write\_strobe == 1'b1) begin**

**if (port\_id[1:0] == 2'b01)**

**out\_port\_reg\_l <= out\_port;**

**else if (port\_id[1:0] == 2'b10)**

**out\_port\_reg\_h <= out\_port;**

**end**

**end**

**Display numbers**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 |  | 1544 |  | 3084 |  | 4624 |  | 6164 |  | 7704 |  | 9244 |
| 39 |  | 1579 |  | 3119 |  | 4659 |  | 6199 |  | 7739 |  | 9279 |
| 74 |  | 1614 |  | 3154 |  | 4694 |  | 6234 |  | 7774 |  | 9314 |
| 109 |  | 1649 |  | 3189 |  | 4729 |  | 6269 |  | 7809 |  | 9349 |
| 144 |  | 1684 |  | 3224 |  | 4764 |  | 6304 |  | 7844 |  | 9384 |
| 179 |  | 1719 |  | 3259 |  | 4799 |  | 6339 |  | 7879 |  | 9419 |
| 214 |  | 1754 |  | 3294 |  | 4834 |  | 6374 |  | 7914 |  | 9454 |
| 249 |  | 1789 |  | 3329 |  | 4869 |  | 6409 |  | 7949 |  | 9489 |
| 284 |  | 1824 |  | 3364 |  | 4904 |  | 6444 |  | 7984 |  | 9524 |
| 319 |  | 1859 |  | 3399 |  | 4939 |  | 6479 |  | 8019 |  | 9559 |
| 354 |  | 1894 |  | 3434 |  | 4974 |  | 6514 |  | 8054 |  | 9594 |
| 389 |  | 1929 |  | 3469 |  | 5009 |  | 6549 |  | 8089 |  | 9629 |
| 424 |  | 1964 |  | 3504 |  | 5044 |  | 6584 |  | 8124 |  | 9664 |
| 459 |  | 1999 |  | 3539 |  | 5079 |  | 6619 |  | 8159 |  | 9699 |
| 494 |  | 2034 |  | 3574 |  | 5114 |  | 6654 |  | 8194 |  | 9734 |
| 529 |  | 2069 |  | 3609 |  | 5149 |  | 6689 |  | 8229 |  | 9769 |
| 564 |  | 2104 |  | 3644 |  | 5184 |  | 6724 |  | 8264 |  | 9804 |
| 599 |  | 2139 |  | 3679 |  | 5219 |  | 6759 |  | 8299 |  | 9839 |
| 634 |  | 2174 |  | 3714 |  | 5254 |  | 6794 |  | 8334 |  | 9874 |
| 669 |  | 2209 |  | 3749 |  | 5289 |  | 6829 |  | 8369 |  | 9909 |
| 704 |  | 2244 |  | 3784 |  | 5324 |  | 6864 |  | 8404 |  | 9944 |
| 739 |  | 2279 |  | 3819 |  | 5359 |  | 6899 |  | 8439 |  | 9979 |
| 774 |  | 2314 |  | 3854 |  | 5394 |  | 6934 |  | 8474 |  | 4 |
| 809 |  | 2349 |  | 3889 |  | 5429 |  | 6969 |  | 8509 |  | 39 |
| 844 |  | 2384 |  | 3924 |  | 5464 |  | 7004 |  | 8544 |  | 74 |
| 879 |  | 2419 |  | 3959 |  | 5499 |  | 7039 |  | 8579 |  | 109 |
| 914 |  | 2454 |  | 3994 |  | 5534 |  | 7074 |  | 8614 |  | 144 |
| 949 |  | 2489 |  | 4029 |  | 5569 |  | 7109 |  | 8649 |  | 179 |
| 984 |  | 2524 |  | 4064 |  | 5604 |  | 7144 |  | 8684 |  | 214 |
| 1019 |  | 2559 |  | 4099 |  | 5639 |  | 7179 |  | 8719 |  | 249 |
| 1054 |  | 2594 |  | 4134 |  | 5674 |  | 7214 |  | 8754 |  | 284 |
| 1089 |  | 2629 |  | 4169 |  | 5709 |  | 7249 |  | 8789 |  | 319 |
| 1124 |  | 2664 |  | 4204 |  | 5744 |  | 7284 |  | 8824 |  | 354 |
| 1159 |  | 2699 |  | 4239 |  | 5779 |  | 7319 |  | 8859 |  | 389 |
| 1194 |  | 2734 |  | 4274 |  | 5814 |  | 7354 |  | 8894 |  | 424 |
| 1229 |  | 2769 |  | 4309 |  | 5849 |  | 7389 |  | 8929 |  | 459 |
| 1264 |  | 2804 |  | 4344 |  | 5884 |  | 7424 |  | 8964 |  | 494 |
| 1299 |  | 2839 |  | 4379 |  | 5919 |  | 7459 |  | 8999 |  | 529 |
| 1334 |  | 2874 |  | 4414 |  | 5954 |  | 7494 |  | 9034 |  | 564 |
| 1369 |  | 2909 |  | 4449 |  | 5989 |  | 7529 |  | 9069 |  | 599 |
| 1404 |  | 2944 |  | 4484 |  | 6024 |  | 7564 |  | 9104 |  | 634 |
| 1439 |  | 2979 |  | 4519 |  | 6059 |  | 7599 |  | 9139 |  | 669 |
| 1474 |  | 3014 |  | 4554 |  | 6094 |  | 7634 |  | 9174 |  | 704 |
| 1509 |  | 3049 |  | 4589 |  | 6129 |  | 7669 |  | 9209 |  | 739 |

**SOFTWARE**

START

Sum = start= 4, inc = 35

Inc = 35

Sum = sum + inc @ an interval of n sec

ans\_h=0, ans\_l=4, startdata=35

Is n sec from hardware

NO

YES

ans\_l =ans\_l+startdata

ans\_h=ans\_h+carry+0

Send ans\_l &ans\_h through outputport\_l and outputport\_h

NO

Is the result reached > 9999

YES

**Software (prog\_rom.psm)**

; prog\_rom.psm

; UCSC Extension - Digital Design with FPGA

; Jae-Yang Park

; jaeyangp@gmail.com

;

; input: 0: not reached one sec. 1: one sec happened

; output: ans\_l --> OUTPUTPORT\_L

; ans\_h --> OUTPUTPORT\_H

; {ans\_h, ans\_l} is 16 bit result

; input and output port

constant INPUTPORT, 00 ; '1' at every one second

constant OUTPUTPORT\_L, 01 ; LSB portion of result

constant OUTPUTPORT\_H, 02 ; MSB portion of result

; constant values

constant MAX\_MSB, 27 ; 9999 (270fh)

constant MAX\_LSB, 0f

constant START\_NUM, 04 ; start number

constant INC\_NUM, 23 ; incremenr value, 35 (23h)

; register alias

namereg s0, indata

namereg s1, startdata ; increment value

namereg s2, ans\_l ; LSB of result

namereg s3, ans\_h ; MSB of result

;

; main program

;

main:

call init

call display ; result through output port

;

calc:

input indata, INPUTPORT

call add\_num ; add ans + increment values

call display ; send result through output port to display

compare ans\_h, MAX\_MSB ; check if result reached 9999 (270fh)

jump c, calc ; ans\_h < 9984

;

next\_chk: ; ans\_h > 9984

compare ans\_l, MAX\_LSB ; check if ans\_l > 15, 9984+15 = 9999

jump nc, main ; yes, result is reached 9999

jump calc ; no, result is not reached 9999

;

init:

load ans\_l, START\_NUM ; start number

load ans\_h, 00 ; clear the MSB of result

return

; send results through output port

display:

output ans\_l, OUTPUTPORT\_L ; LSB

output ans\_h, OUTPUTPORT\_H ; MSB

return

;

; add numbers

; ans\_l = ans\_l + startdata

add\_num:

compare indata, 00 ; 1: one sec is happened, 0 is not

jump nz, add\_num1 ; no one sec, then return, or go to add number

return

;

add\_num1:

load startdata, INC\_NUM ; increment at a distance of 35

add ans\_l, startdata ; add LSB with startdata

addcy ans\_h, 00 ; add MSB with carry

return ; {ans\_h, ans\_l} is complete 16 bits result

**HARDWARE**

Read data from out\_port selected with port\_id when write\_strobe is active

Combine two 8-bit data into one 16-bit data to display

Convert 16-bit binary text into BCD data

Display BCD data every n second

Send “done” signal to do next calculattion

**RTL**

**Top module**

`timescale 1ns / 1ps

module in\_to\_out(in, out);

parameter IN = 8;

parameter OUT = 16;

input [IN-1:0] in;

output [OUT-1:0] out;

assign out = in;

endmodule

module add16(CLK1, arst, seg, an, Led);

parameter N = 7;

parameter W = 4;

parameter S8 = 8;

parameter S16 = 16;

input CLK1, arst;

output [0:N-1] seg;

output [W-1:0] an;

output [N:0] Led;

wire [9:0] address;

wire [17:0] instruction;

wire [7:0] port\_id, in\_port, out\_port;

wire write\_strobe, read\_strobe, interrupt\_ack;

wire [N-1:0] seg\_out;

reg [7:0] out\_port\_reg\_l, out\_port\_reg\_h;

wire [S16-1:0] out\_port\_reg;

wire [S16-1:0] num;

wire [S16-1:0] text;

wire done;

reg done\_reg;

**assign out\_port\_reg = {out\_port\_reg\_h, out\_port\_reg\_l};**

embedded\_kcpsm3 U(port\_id, write\_strobe, read\_strobe, out\_port, in\_port, interrupt, interrupt\_ack, arst, CLK1);

in\_to\_out #(S16, S16) I(out\_port\_reg, num);

bin2bcd #(S16, S16) M(num, text);

display D(text, CLK1, arst, seg, an);

one\_second ONE (CLK1, arst, done);

always @(posedge CLK1) begin

if (arst == 1'b1) begin

out\_port\_reg\_l <= 7'b0;

out\_port\_reg\_h <= 7'b0;

end

**else if (write\_strobe == 1'b1) begin**

**if (port\_id[1:0] == 2'b01)**

**out\_port\_reg\_l <= out\_port;**

**else if (port\_id[1:0] == 2'b10)**

**out\_port\_reg\_h <= out\_port;**

**end**

**else if (read\_strobe == 1'b1)**

**done\_reg <= 1'b0;**

**else**

**done\_reg <= done;**

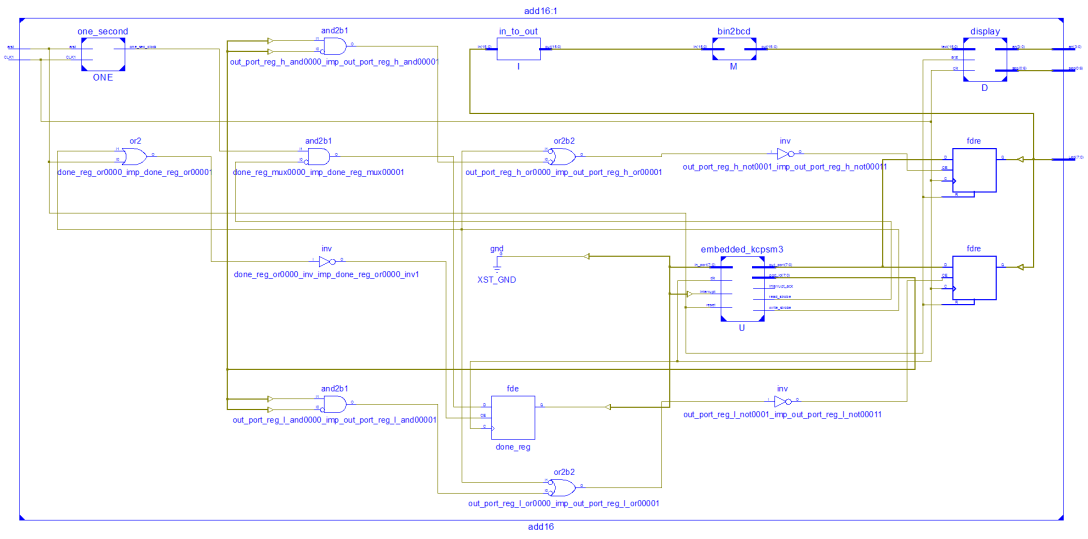
**end**

assign in\_port = done\_reg;

assign Led = out\_port\_reg\_l;

endmodule

**Schematics**

****

**Display Pictures**

