Davide Carini (Matricola: 890064 - Codice Persona: 10568649)

PROVA FINALE

(Progetto n° 11 di Reti Logiche)

Corso di Reti Logiche

Anno Accademico 2019/2020

Professore: Carlo Brandolese

**Indice**

1. Introduzione
   1. Scopo del Progetto
   2. Esempio a 8 bit
   3. Entity

Macchina a Stati

Testing

Il documento deve descrivere l’architettura del circuito e non la sua implementazione. Questo significa che la sua macchina a stati non la deve sintetizzare lei e neppure descrivere in termini di diagrammi degli stati.

Spiegare la funzione dell’architettura.

La macchina a stati è un contatore

Esempi

Descrivere architettura e le emtity principali che lo compongono.

Dovrei avere 2 registri per salvare Q e R ad ogni iterazione del ciclo.

If si implementa con mux pilotato da R >= D

Le architecture è preferibile chiamarle RTL oppure struct.

**INTRODUZIONE**

Scopo del Progetto (DIVISORE INTERO CON RESTO)

Lo scopo è progettare un divisore intero su 32 bit basato sul metodo di “divisione lunga”. Siano N il dividendo, D il divisore, Q il quoziente , R il resto e n la dimensione delle parole. L’algoritmo è descritto dal seguente pseudocodice:

if( D == 0 ) {

error();

}

Q = 0

R = 0

for(i = n-1; i >= 0; i-- ) {

R = R << 1

R[0] = N[i]

if( R ≥ D ) {

R = R − D

Q[i] = 1

} else {

Q[i] = 0

}

}

Deve essere realizzata una rete sequenziale che implementi il divisore basato su tale algoritmo. Una volta realizzato il componente, è richiesto di realizzare un test-bench per la simulazione e la verifica del corretto funzionamento nei diversi casi.

L’algoritmo di “divisione lunga” è l’algoritmo standard più utilizzato per la divisione di numeri espressi in notazione decimale.

Esempio ( con n=8 bit )

Pongo N=100011112 (14310) e D=000010012(910). Verifico inizialmente che il divisore sia diverso da 0 (condizione rispettata). Dopo inizializzo quoziente e resto a 0.

Ciclo FOR

i=n-1=7 i=3

R= 00 R=010000

R=01 R=010001

Condizione if false Condizione if true

Q=0 R=1000

i=6 Q=00001

R=010 i=2

R=010 R=10000

Condizione if false R=10001

Q=00 Condizione if true

i=5 R=1000

R=0100 Q=000011

R=0100 i=1

Condizione if false R=10000

Q=000 R=10001

i=4 Condizione if true

R=01000 R=1000

R=01000 Q=0000111

Condizione if false i=0

Q=0000 R=10000

R=10001

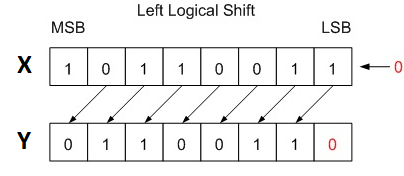
Condizione if true

R=10002 -> 810

Q=000011112 -> 1510

La divisione è stata fatta con parole di 8 bit in modo tale da semplificare il ciclo iterativo. Con parole di 32 bit, il ciclo for avrà 32 iterazioni differenti.

**LEFT SHIFTER (con n=8 bit) ok**





entity shift\_left is

port( x: in std\_logic\_vector ( 7 down to 0);

y: out std\_logic\_vector ( 7 down to 0) );

end shift\_left;

architecture rtl of shift\_left is

begin

y <= x(6 down to 0 ) & ‘0’ ;

end rtl;

Ho utilizzato il costrutto di slicing (slice=porzione di un vettore) e l’operatore di concatenamento.

**FULL SUBTRACTOR ( con n=32 bit) ok**

<http://unina.stidue.net/Architettura%20dei%20Sistemi%20di%20Elaborazione/Materiale/Addizionatori%20v1.01.pdf>

fatto su ISE

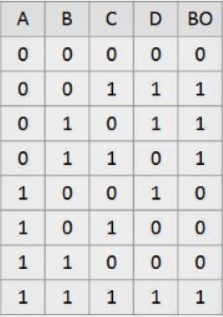
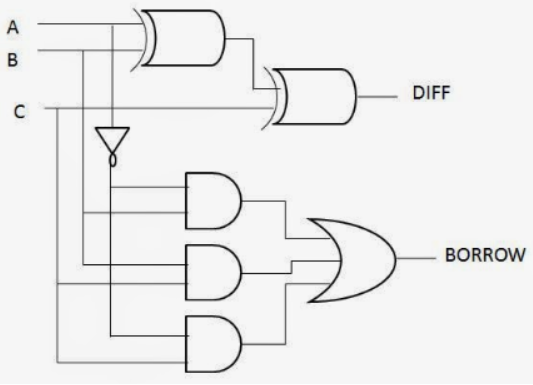


Figure 1 tabella della verità

Difference = A XOR B XOR C

Borrow = A' \* B + B \* C + A' \* C

entity full\_subtractor is

generic (N: INTERGER := 32); -- N-> number of input bits

port(a,b,c: in std\_logic\_vector (N-1 down to 0);

diff,bor: out std\_logic\_vector (N-1 down to 0) );

end full\_subtractor;

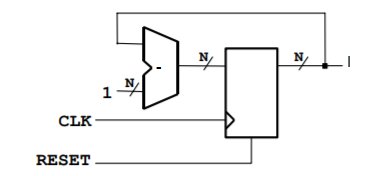
architecture rtl\_sub of full\_subtractor is

diff <= a XOR b XOR c;

bor <= ((NOT a AND b) OR ( b and c ) OR (NOT a AND c )) ;

end rtl\_sub;

**Contatore DOWN con N=5 bit**

****

entity COUNTER\_DOWN is

generic ( N: integer) ;

port( CLK: in std\_logic;

RESET: in std\_logic;

I: out std\_logic\_vector( 0 to N-1 )

);

end COUNTER\_DOWN;

architecture rtl of COUNTER\_DOWN is

signal TY: std\_logic\_vector( 0 to N-1 );

begin

--Counts

count: process(CLK, RESET)

begin

if (RESET = ‘1’) then

TY<= ‘11111’ ;

elseif ( CLK’ event and CLK =’1’ ) then

TY<=TY – ‘1’;

end if;

end process;

--Assigns the output signal

I <= TY;

end rtl;

Pensare ad architettura pipeline (non so se la strada è quella corretta…)

architeture pipe of SAMPLE is

signal T1, T2: std\_logic;

begin

regs: process( CLK )

begin

if( CLK’event and CLK = ’1’ ) then

T1 <= h( X );

T2 <= g( T1 );

Y <= f( T2 );

end if;

end process;

end pipe;

<https://www.allaboutcircuits.com/technical-articles/how-to-vhdl-description-of-a-simple-algorithm-the-control-path/>

cycle without while

if( D== 0){

error();

}

Q=0;

R=0;

i=31;

if( i == 0){

--go to end

}

else{

R=R << 1;

R[0]=N[i];

If( R >= D){

R=R-D ;

Q[i]=1;

i--;

--go to if( i== 0)

}

Else{

Q[i]=0;

i--;

--go to if( i== 0)

}

}

<https://staff.emu.edu.tr/muhammedsalamah/Documents/CMPE224/labs/Exp7.pdf>