

Afspraken:

- Vermeld op elk blad naam en voornaam. Dit is zeer belangrijk voor de schriftelijke vraag! Alle pagina's nummeren.
- De voorbereidingstijd voor de eerste twee vragen samen is 0,5 uur, voor de derde vraag 2 uur, waarvan maximum 1 uur voor deel (a). De mondelinge bespreking ervan begint respectievelijk 0,75 en 2,5 uur na het begin van het examen. De vierde vraag wordt schriftelijk opgelost en hiervoor is 1 uur voorzien. De vierde vraag wordt ten laatste 4,25 uur na het begin van het examen afgegeven.
- Toiletbezoek is pas mogelijk na de de mondelinge bespreking van de eerste twee vragen.
- Papier en appendix A zijn voorzien. Papier kan je bijvragen. Enkel eigen schrijfgierief en eten/drank mogen bijkomend gebruikt worden. Dus geen rekenmachine of communicatiemiddelen zoals gsm.
- Vul alvast het aanwezigheidsformulier in. Je laat dit aanvullen bij elke mondelinge bespreking en geeft dit af samen met vraag 4. Het aantal kladbladen moet niet vermeld worden.
- De opgave en appendix A worden samen met de kladbladen apart afgegeven. Ook het niet gebruikte papier wordt teruggenomen. De kladbladen worden na het examen door de docent vernietigd.

Vragen:

1. Wat is de stijgtijd van een poort? Waar hangt deze tijd van af?
Is er een verschil tussen een NMOS- en een CMOS-inverter?
2. Wat is een RISC en wat zijn de verschillende stappen in het ontwerp ervan?
3. Ontwerp een FSMD die $od = 3\sqrt{e-id}$ benadert zoals dit op de achterkant van dit blad in VHDL beschreven is. Bij de realisatie mag je enkel bouwblokken gebruiken waarvan de hardware implementatie in de les bestudeerd is.
Alle real getallen worden in hardware voorgesteld als getallen met een vaste komma. Voor de ingang id en de uitgang od worden 8 bits na de komma gebruikt. Verder worden voor alle getallen zoveel bits gebruikt als nodig om de uitgang met de gewenste nauwkeurigheid te berekenen.

- a) Teken een ASM-schema voor deze schakeling, die de vereisten van de VHDL-beschrijving qua tijdsgedrag respecteert. Probeer zo weinig mogelijk toestanden te gebruiken. Indien één enkel ASM-schema te ingewikkeld lijkt, mag je dit opsplitsen in meerdere ASM-schema's.
- b) Ontwerp het datapad tot op RTL-niveau. Minimalisering is niet expliciet nodig, maar probeer wel zo weinig mogelijk hardware te gebruiken, zonder evenwel nog het ASM-schema te wijzigen. Vergeet ook niet het aantal bits bij iedere verbinding te vermelden.
- c) Beschrijf het controlegedeelte met een toestandsdiagram. Als er meerdere ASM-schema's gebruikt worden volstaat het om enkel het diagram met de meeste toestanden te beschrijven.

4. Maak de goedkoopst mogelijke IC realisatie van de FSM, beschreven in nevenstaande tabel, met ingangen I & J en uitgangen X & Y . Je mag enkel gebruik maken van JK-flipflops en NAND- en/of NOR-poorten met 2 ingangen. Maak gebruik van Karnaugh-kaarten om alle functies te bepalen. Vergeet de nuttige bijkomende informatie (zoals het waarom van een keuze) niet te vermelden!

Huidige toestand	Volgende toestand / XY			
	$IJ = 00$	$IJ = 01$	$IJ = 10$	
S_0	$S_2/10$	$S_1/01$	$S_4/10$	
S_1	$S_2/10$	$S_2/11$	$S_3/01$	
S_2	$S_0/10$	$S_1/01$	$S_0/10$	
S_3	$S_2/10$	$S_2/11$	$S_1/01$	
S_4	$S_3/01$	$S_4/01$	$S_0/10$	

entity fsmd is

port (clk, rst : in bit;

LD : in integer range -1 to 4;

id : in real range -2.0 to 2.0; → 815.1

od : out real;

oa : out bit);

end entity fsmd;

architecture behav of fsmd is

signal n, rsti : natural;

signal odi, x : real;

begin

p1: process is

variable t, y : real;

begin

if n > 0 then

oa <= '0'; y := 1.0;

if n < 3 then y := 3.0; end if;

for i in n downto 1 loop

exit when rsti > 0;

t := x*y;

if (i mod 2) = 0 then t := t/real(i); end if;

if i < 4 then y := 3.0 - t;

else y := 1.0 - t; end if;

wait until clk = '1';

end loop;

n <= 0;

if rsti = 0 then

oa <= '1'; odi <= y;

end if;

end if;

wait until clk = '1';

end process p1;

p2: process (clk, rst, odi) is

variable ni : natural;

begin

ni := LD + 1;

if rst = '0' then

n <= 0; rsti <= 1; oa <= '0';

elsif clk'event and clk = '1' then

rsti <= 0;

if n = 0 and ni > 0 then

x <= id/2.0; n <= ni;

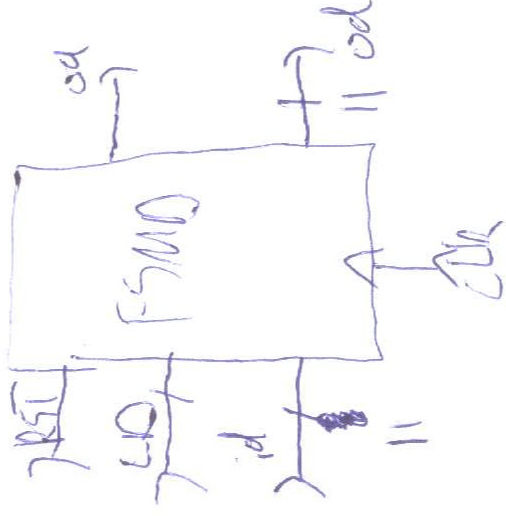
end if;

end if;

odi <= odi;

end process p2;

end architecture behav;



act-ref large asynch. reset

asynch. reset