o con n / n o c on

2.1 A word on Cyclic Redundancy Codes

As it is easily imaginable, we all would like to have no errors when transmiting data over
[illegible]3335 0 273(the)Tj 193907 0 T63(General)pr5422 0 T299wlic

n nç n

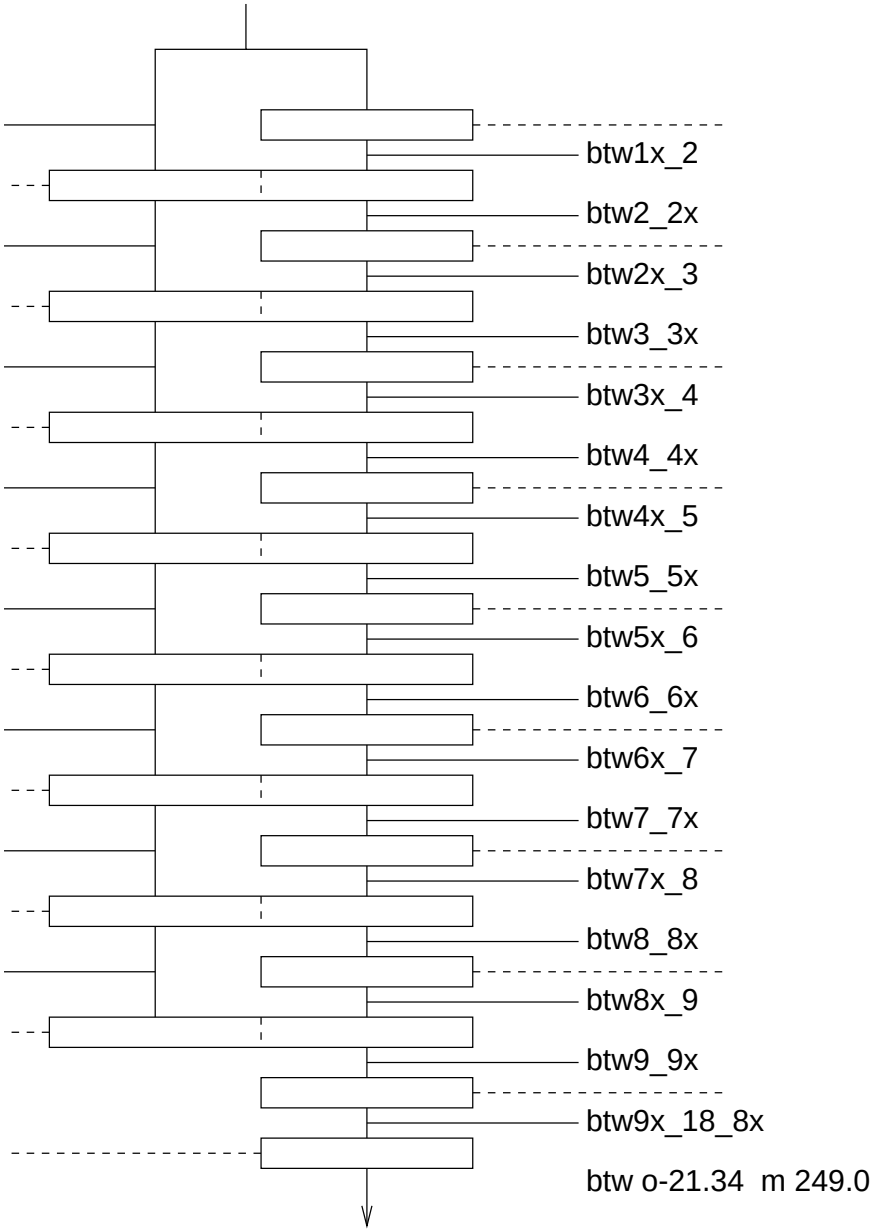
3.1 The algorithm

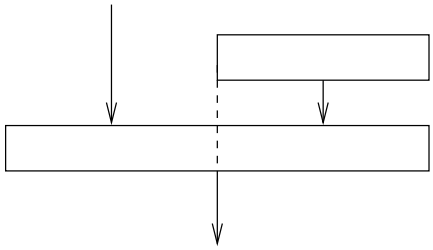
There are several

but I didn’t realize of this “bug” in these early stages of the design). Anyway this is a good register lenght, since it provides us with a good clock speed reduction

φ_1

φ_2


btw1x_2
btw2_2x
btw2x_3
btw3_3x
btw3x_4
btw4_4x
btw4x_5
btw5_5x
btw5x_6
btw6_6x
btw6x_7
btw7_7x
btw7x_8
btw8_8x
btw8x_9
btw9_9x
btw9x_18_8x
btw o-21.34 m 249.0

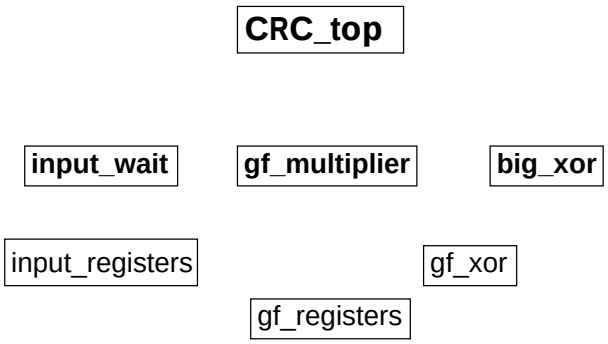


A VHDL code of the CRC generator

A.1 Files dependency tree

Different files and different levels in the hierarchy have been used in the description of the CRC generator for the sake of clearness.

The structure is graphically represented in figure 15



```
library IEEE;
use IEEE.std_l gic_1164.all;
F
```

```
begin  -- behavi r

  -- purp se: Pipelining register activated by phi1
  -- type   : sequential
  -- inputs : phi1, reset, input (16 bits)
  --  utputs:  utput (16 bits)
  p_input_phi1_register: pr cess (phi1, reset)
  begin  -- pr cess p_input_phi1_register
    if reset = '0' then                 -- asynchr n us reset (active l w)
       utput <= ( thers => '0');
    elsif    w y     fy     F w  (     ev    u   bey    y       a       nd  sif  et ='
```

A.2 VHDL

```
end rtf_phi
```

```
-- purp se: GF1x
-- type   : c mbinati nal-- inputs : input_fcs
                       --  utputs p_gf_x r_input: process (input_fcs) gf_x r_input
                                                                utput_wip(01) <= (input_fc
```

```
end ■f_x r_■x;

architecture behavi r  f ■f_x r_■x is

be■in  -- behavi r

  p_■f_x r_■x:
```

```
       utput_wip :  ut  std_l gic_vect r(b1 d wnt  0));

end gf_x r_bx;

a r_
```

A.2.5 gf_multiplier

```
c mp nent ■f_phi1_re■ister
  p rt (
    reset      : in  std_l ■ic;                    -- #RESET
```

```
input_fcs  : in
```

```
input_wip  => btwl_x_9, input_fcs  => btwl_9,
```

A.2.6 big_xor.vhd

```
--============================================================================
-- File        : big_xor.vhd
-- Author      : José María Nadal
-- Date        : Mar, 2001
-- Description : Defines the output stage

-- Copyright (C) 2001  José María Nadal <schmaltzsc auts-es.org>

-- This program is free software; you can redistribute it and/or
--
```

```
    elsif phi2'event and phi2 = '1' then  -- rising clock edge
       output(15 downto  0)  <=  output_xor(15 downto  0);
       output(31 downto  16) <= fcs_input(15 downto  0);
    end if;
  end process p_big_xor_register;

  p_big_xor_combin

        ti      a
```

```
                                        phi1 => phi1, input  => input,
                                         utput => wait_intermediate);

  f_multiplier_1 : f_multiplier p rt map (reset => reset_intermediate,
```

```
library IEEE;
use I
```

```
begin  -- pr cess p_input
 if phi1'event and phi1='1' then
  if
```

```
    wait;
  end pr cess p_reset;

  p_input: pr cess (phi2)
    variable input_file : bit_vect r(61 d wnt  0);
    variable line : line;
  begin  -- pr cess p_input
   if phi2'event
```

```
-- Copyright (C) 2001
```

```
-- Imp rtant: These parameters have been calculated f r a 500MHz
-- relative cl ck -2 ns per stage-
p_phi1: pr cess
begin  -- pr cess p_phi1
  phi1 <= '1', '0' after 2.1 ns;
  wait f r 4.2 ns;
end pr cess p_phi1;

p_phi2: pr cess
begin  -- pr cess p_phi2
  phi2 <= '1', '0' after 0.5 ns, '1' after
                    4.2 ns;
                end pr ce y  y          ;  b  c w  y       c      n; wb      , w
        begin  -
```

```
  big_x r_1 : big_x r p rt map (phi1 => phi1, reset => reset, gf_input => gf_input,
                                input_input => input_input, fcs_input => fcs_input,
                                 utput =>  utput);

end structural;

c nfigurati n cfg_       ;
```

```
    reset   : in    std_l gic;
    input   : in    std_l gic_vect r(15 d wnt  0);
    fcs_ ut :  ut std_l gic_vect r(01 d wnt  0));
end c mp nent;

signal phi1, phi2, reset : std_l gic;
signal input             : std_l gic_vect r(15 d wnt  0);
signal fcs_ ut           : std_l gic_vect r(01 d wnt  0);

-- We a
```

```
  end if;
end pr cess p_ utput;

■R■_1 : ■R■  _t          y          , w y  f          e t w  y      _t
```

B

B.1 Interfaces of the different

C.1 GNU GENERAL PUBLIC LICENSE

Version

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any

designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy

prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for

may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

- In any section entitled “Acknowledgements” or “Dedications”, preserve the section’s title, and preserve in the section all the

C.2.8 Aggreg

LTH, Elektrovetenskap, Lund 2002