library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity regbank is

port( clk : in std\_logic;

rst : in std\_logic;

write\_en : in std\_logic;--\ enables for

rx\_en : in std\_logic;---> Write,Read rx 1 and ry 2.

ry\_en : in std\_logic;--/ comes from the decoder

rx\_addr : in std\_logic\_vector(3 downto 0);--Register addresses for Rx to be read

ry\_addr : in std\_logic\_vector(3 downto 0);--Register addresses for RY to be read

write\_addr : in std\_logic\_vector(3 downto 0);--Regbank address for where write\_data will be written

write\_data : in std\_logic\_vector(7 downto 0);--data being written into the regbank

rx\_data : out std\_logic\_vector(7 downto 0);--

ry\_data : out std\_logic\_vector(7 downto 0) --

);

end regbank;

architecture behv of regbank is

type rf\_type is array (0 to 15) of std\_logic\_vector(7 downto 0);--\*

signal tmp\_rf: rf\_type := (others => "00000000");

begin

write: process(rst, write\_addr, write\_en, write\_data)

begin

if rst='1' then -- high active

tmp\_rf <= (tmp\_rf'range => "00000000");

else

if write\_en='1' then

tmp\_rf(conv\_integer(write\_addr)) <= write\_data;

end if;

end if;

end process;

read1: process( rst, rx\_en, rx\_addr)

begin

if rst='1' then

rx\_data <= "00000000";

else

if rx\_en='1' then

rx\_data <= tmp\_rf(conv\_integer(rx\_addr));

end if;

end if;

end process;

read2: process( rst, ry\_en, ry\_addr)

begin

if rst='1' then

ry\_data <= "00000000";

else

if ry\_en='1' then

ry\_data <= tmp\_rf(conv\_integer(ry\_addr));

end if;

end if;

end process;

end behv;