LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.std\_logic\_unsigned.all;

USE IEEE.numeric\_std.all;

-- Infos zum Algorithmus

-- http://www.johnloomis.org/ece314/notes/devices/binary\_to\_BCD/bin\_to\_bcd.html

ENTITY int2ascii IS

    Port ( i\_number : IN  integer RANGE 0 TO 59;

           o\_ascii0   : OUT std\_logic\_vector(7 downto 0);

           o\_ascii1   : OUT std\_logic\_vector(7 downto 0));

END ENTITY int2ascii;

ARCHITECTURE behavioral OF int2ascii IS

    SIGNAL s\_bcd0 : unsigned(3 DOWNTO 0);

    SIGNAL s\_bcd1 : unsigned(3 DOWNTO 0);

BEGIN

    PROCESS (i\_number)

        VARIABLE v\_bcd0   : unsigned(3 DOWNTO 0);

        VARIABLE v\_bcd1   : unsigned(3 DOWNTO 0);

        VARIABLE v\_number : unsigned(7 DOWNTO 0);

    BEGIN

        v\_bcd0 := "0000";

        v\_bcd1 := "0000";

        v\_number := to\_unsigned(i\_number, v\_number'length);

        -- Je BCD Zahl (59 = 2x ist für diese Anwendung ausreichend)

        FOR i IN 1 TO v\_number'length LOOP

            -- BCD0 > 0100, addiere 0011

            IF v\_bcd0 > "0100" THEN

                v\_bcd0 := v\_bcd0 + "0011";

            END IF;

            -- BCD1 > 0100, addiere 0011

            IF v\_bcd1 > "0100" THEN

                v\_bcd1 := v\_bcd1 + "0011";

            END IF;

            -- Shift um ein Bit

            v\_bcd1 := shift\_left(v\_bcd1, 1);

            v\_bcd1(0) := v\_bcd0(3);

            v\_bcd0 := shift\_left(v\_bcd0, 1);

            v\_bcd0(0) := v\_number(v\_number'length - i);

        END LOOP;

        -- BCD Zahlen dem entsprechenden Signal zuweisen

        s\_bcd0 <= v\_bcd0;

        s\_bcd1 <= v\_bcd1;

    END PROCESS;

   -- Die Umwandlung von BCD in ASCII erfolgt durch jeweiliges vorranstellen einer 0011,

    -- da die ASCII Ziffern 0-9 auf den Zahlen 30-39 liegen.

    o\_ascii0 <= std\_logic\_vector("0011"&s\_bcd0);

    o\_ascii1 <= std\_logic\_vector("0011"&s\_bcd1);

END ARCHITECTURE behavioral;