HI4ハードウェア実験  
第3回実験

実験者：HI4 45号 山口惺司

共同実験者：HI4 46号 吉田旺雅

提出日2024/08/05

# 目的

第1回, 第2回実験では組合せ回路を作成したが, 今回はprocess文を用いて順序回路を作成, 動作確認をすることで学習する.

1. 実験内容

## 課題1

* + 1. 課題内容

以下のプログラムを実行, 動作確認し, どのような回路か考察しなさい.

* + 1. プログラム

library IEEE;

use IEEE.std\_logic\_1164.all;

entity kadai3 is

    port(

        D, CLK : in std\_logic;

        Q, NQ : out std\_logic

    );

end kadai3;

architecture rtl of kadai3 is

begin

    process(CLK)

    begin

        if(CLK'event and CLK = '1') then

            Q <= D;

            NQ <= not D;

        end if;

    end process;

end rtl;

* + 1. ピン設定情報

ピン設定情報を図1に示す.

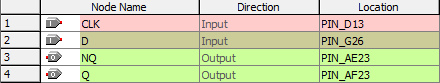


図 1 課題1ピン設定情報

* + 1. 結果

CLKのボタンが押されたときにDが0だった時, Qが0, Dが1だった時, Qが1となった.

NQはQと反対の出力がされた.

* + 1. 考察

processを使って順序回路を実装している.

実行結果からこの回路はD-フリップフロップと推測される.

if文の中にはCLKに変化があり, かつ, CLKが1の時という条件が書かれている.

条件に当てはまった時のDの値をQに入れている.

* 1. 課題2
     1. 課題内容

課題1を参考にJKフリップフロップを構成し, 動作確認しなさい.

* + 1. プログラム

library IEEE;

use IEEE.std\_logic\_1164.all;

entity kadai42 is

    port (

        J, K, CLK : in std\_logic;

        Q : out std\_logic

    );

end kadai42;

architecture rtl of kadai42 is

    signal A : std\_logic := '0';

begin

    process(CLK)

    begin

        if(CLK'event and CLK = '1') then

            if (j = '1' and k = '0') then

                A <= '1';

            elsif (j = '0' and k = '1') then

                A <= '0';

            elsif (j = '1' and k = '1') then

                A <= not A;

            end if;

        end if;

        Q <= A;

    end process;

end rtl;

* + 1. ピン設定情報

ピン設定情報を図2に示す.

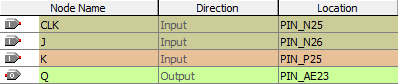


図 2課題2ピン設定情報

* + 1. 結果

実行結果を表1に示す.

表1 課題2-JK-ff実行結果

|  |  |  |
| --- | --- | --- |
| J | K | Q |
| 0 | 0 | Q |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 |  |

* + 1. 考察

実行結果から

J=0, K=0の時, Qを保持(保持状態)

J=0, K=1の時, Qを0にする(リセット状態)

J=1, K=0の時, Qを1にする(セット状態)

J=1, K=1の時, Qを反転させる(トグル状態)

という風になっていることがわかった.

このことから, ただしくJK-ffの回路が組めていることが分かる.

また, プログラムを書く際に, 最初は変数Aを使用しておらず, Qに直接代入するという書き方をしていたが, Q <= not Qという文でエラーが発生した.

これは変数Qが出力用の変数型をしているためだと考えたため, 別の変数Aを経由して代入するように変更したらうまく動作した.

理由は不明だがentityの名前をjkffにすると正しく動作しないことがあり, 名前を変更することで解決した.

* 1. 課題3
     1. 課題内容

以下のプログラムを実行, 動作確認し, どのような回路か考察しなさい.

* + 1. プログラム

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_unsigned.all;

entity COUNT10 is

    port(RESET,CLK:in std\_logic;

        Q :out std\_logic\_vector(3 downto 0)

    );

end COUNT10;

architecture RTL of COUNT10 is

signal TQ:std\_logic\_vector(3 downto 0);

begin

        process (RESET,CLK)begin

            if(RESET='0')then

                TQ <= "0000";

            elsif(CLK'event and CLK='1')then

                if(TQ="1001")then

                    TQ <= "0000";

                else

                    TQ <= TQ+'1';

                end if;

            end if;

        end process;

        Q <= TQ;

end RTL;

* + 1. ピン設定情報

ピン設定情報を図3に示す.

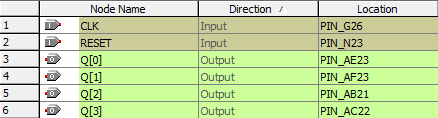


図3 課題3ピン設定情報

* + 1. 結果

RESETに設定されているボタンを押すと, Q=0000

CLKを押すたびにQの値が1ずつ増えていく.

Qが1001になると次のQは0000となる.

カウントされている様子を表2に示す.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLKを押した回数の合計 | Q[3] | Q[2] | Q[1] | Q[0] |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 |

* + 1. 考察

実行結果から, この回路は10進カウンタであることがわかる.

RESETボタンを押すとQの値が何であろうと0000にすることができる.

* 1. 課題4
     1. 課題内容

以下のプログラムは「ステートマシン」と言われる記述で構成されています.このプログラムを実行, 動作確認し, どのような回路か考察しなさい.

* + 1. プログラム

library IEEE;

use IEEE.std\_logic\_1164.all;

entity st\_man is

    port(RESET,CLK,X:in std\_logic;

        Y   :out std\_logic\_vector(1 downto 0));

end st\_man;

architecture RTL of st\_man is

type STATE is(STATE0,STATE1,STATE2);

signal IS\_STATE, NE\_STATE:STATE;

begin

    process (RESET,CLK) begin

        if(RESET='0')then

            IS\_STATE <= STATE0;

        elsif(CLK'event and CLK='1')then

            IS\_STATE <= NE\_STATE;

        end if;

    end process;

    process (IS\_STATE,X) begin

        case IS\_STATE is

            when STATE0 =>

                Y <= "00";

                if(X='1')then

                    NE\_STATE <= STATE1;

                else

                    NE\_STATE <= STATE0;

                end if;

            when STATE1 =>

                Y <= "01";

                if(X='1')then

                    NE\_STATE <= STATE2;

                else

                    NE\_STATE <= STATE1;

                end if;

            when STATE2 =>

                Y <= "10";

                if(X='1')then

                    NE\_STATE <= STATE0;

                else

                    NE\_STATE <= STATE2;

                end if;

        end case;

    end process;

end RTL;

* + 1. ピン設定情報

ピン設定情報を図4に示す.

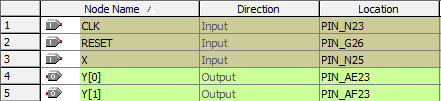


図4 課題4ピン設定情報

* + 1. 結果

Xに割り当てられているボタンが押されるたびにYが”00”→”01”→”10”→”00”という風に遷移していった.

* + 1. 考察

状態遷移の詳細を以下に示す.

STATE0:

出力Yは"00".

Xが'1'の場合, 次の状態はSTATE1.

Xが'0'の場合, 次の状態はSTATE0.

STATE1:

出力Yは"01".

Xが'1'の場合, 次の状態はSTATE2.

Xが'0'の場合, 次の状態はSTATE1.

STATE2:

出力Yは"10".

Xが'1'の場合, 次の状態はSTATE0.

Xが'0'の場合, 次の状態はSTATE2.

このようにステートマシンはある値が変化すると状態が遷移し, その状態の時に任意の動作を行うことができるシステムである.

* 1. 課題5
     1. 課題内容

4進カウンタをステートマシン記述で作成し, 動作確認しなさい.

* + 1. プログラム

library IEEE;

use IEEE.std\_logic\_1164.all;

entity QuaternaryCounter is

    port(

        RESET, CLK, X: in std\_logic;

        Y : out std\_logic\_vector(1 downto 0));

end QuaternaryCounter;

architecture rtl of QuaternaryCounter is

type STATE is (STATE0, STATE1, STATE2, STATE3);

signal IS\_STATE, NE\_STATE : STATE;

begin

    process(RESET, CLK)

    begin

        if(RESET='0')then

            IS\_STATE <= STATE0;

        elsif(CLK'event and CLK='1')then

            IS\_STATE <= NE\_STATE;

        end if;

    end process;

    process(IS\_STATE, X)

    begin

        case IS\_STATE is

            when STATE0 =>

                Y <= "00";

                if(X='1')then

                    NE\_STATE <= STATE1;

                else

                    NE\_STATE <= STATE0;

                end if;

            when STATE1 =>

                Y <= "01";

                if(X='1')then

                    NE\_STATE <= STATE2;

                else

                    NE\_STATE <= STATE1;

                end if;

            when STATE2 =>

                Y <= "10";

                if(X='1')then

                    NE\_STATE <= STATE3;

                else

                    NE\_STATE <= STATE2;

                end if;

            when STATE3 =>

                Y <= "11";

                if(X='1')then

                    NE\_STATE <= STATE0;

                else

                    NE\_STATE <= STATE3;

                end if;

        end case;

    end process;

end rtl;

* + 1. ピン設定情報

ピン設定情報を図5に示す.

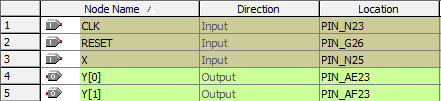


図5 課題5ピン設定情報

* + 1. 結果

Xに割り当てられているボタンが押されるたびにYが”00”→”01”→”10”→”11”→”00”という風に遷移していった.

* + 1. 考察

課題4を参考にして作ったため, コード自体はほとんど変わらずSTATE3の時の動作を追加しただけである.

実行結果もほとんど変わらず課題4と違って”11”までカウントできるようになった,

実行結果から4進カウンタが正しく実装できていることがわかった.

* 1. 課題6
     1. 課題内容

VHDL実験ボードの入出力を考慮し, オリジナルの順序回路を構成しなさい.レポートには, 回路の仕様とプログラムリスト, ピン設定, 実行結果を示すこと.

今回は, JK-ffを用いて同期式16進カウンタを作成し, 7セグメントに表示させる回路を構成した.

* + 1. プログラム

library ieee;

use ieee.std\_logic\_1164.all;

entity kadai46 is

    port (

        CLK1 : in std\_logic;

        Z1 : out std\_logic\_vector(6 downto 0)

    );

end kadai46;

architecture rtl of kadai46 is

signal A0, A1, A2, A3 : std\_logic := '0';

signal tmp0, tmp1 : std\_logic := '0';

component kadai42

    port (

        J, K, CLK : in std\_logic;

        Q : out std\_logic

    );

end component;

component segment

    port (

        A: in std\_logic\_vector(3 downto 0);

        Z: out std\_logic\_vector(6 downto 0)

    );

end component;

begin

    jkff1: kadai42 port map(J => '1', K => '1', CLK => CLK1, Q => A0);

    jkff2: kadai42 port map(J => A0, K => A0, CLK => CLK1, Q => A1);

    tmp0 <= A1 and A0;

    jkff3: kadai42 port map(J => tmp0, K => tmp0, CLK => CLK1, Q => A2);

    tmp1 <= tmp0 and A2;

    jkff4: kadai42 port map(J => tmp1, K=> tmp1, CLK => CLK1, Q => A3);

    seg1: segment port map(A(0) => A0, A(1) => A1, A(2) => A2, A(3) => A3, Z => Z1);

end rtl;

* + 1. ピン設定情報

ピン設定情報を図6に示す.

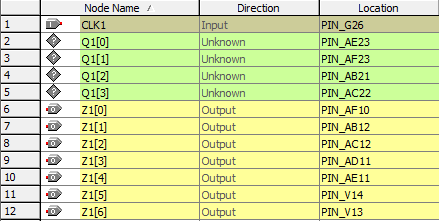


図6 課題6ピン設定情報

* + 1. 結果

CLKが押されるたびにカウントが進んでいた.

Z1を7セグメントとしてみた時の実行結果を表3に示す.

表3 課題6実行結果

|  |  |
| --- | --- |
| CLKを押した回数の合計 | Z1 |
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | A |
| 11 | b |
| 12 | C |
| 13 | d |
| 14 | E |
| 15 | F |
| 16 | 0 |

* + 1. 考察

まず今回作成した同期式16進カウンタの回路図を図7に示す.

ダイアグラム, 概略図

自動的に生成された説明

図7 同期式16進カウンタ

課題2で作成したJKffの回路と前の実験で作成した7セグメントに出力する回路をコンポーネントとして宣言し, 今回の回路を作成した.

実行結果からCLKが押されるたびにカウンタが1ずつ進んでいることが分かるため, 回路は正しく実装できていることが分かる.

JKffの回路をコンポーネントとして宣言するときに, どれをどれに代入すればいいかわかりにくく, 苦労した.

1. 感想

今回の実験ではprocessを用いて順序回路を作成したが, 2年生の時に学習したD-FFやJK-FFなどの順序回路が出てきて, どのような回路かが思い出せず調べながらの実験となったため大変だった.

課題6では今まで学習したことを利用して回路を作成できたためよかった.