VHDL による
ロジック回路製作学習会

実施期間：平成 13 年 12 月 19 日～平成 14 年 9 月 30 日
参加者：菊池成人、斉田尚彦、池田裕史
学習会の目的：

「今はこたつの上で、ソフトウェアで電子回路を作る」おととしの 11 月、技術研修委員会企画の講演会で武藤先生がこう述べられたことを参加した方は覚えていますか？あの時、武藤先生が使っていた、ロジック IC 自作のための記述言語が VHDL です。この VHDL を使えば、パソコン上で配線し、動作確認し、CPLD などのデバイスに書き込むことができます。

デジタル回路の製作において、電子部品を揃え、基板を作り、部品をはんだ付けするという作業は、思ったより手間が掛かります。うまく動作しない場合は、部品、回路のチェックや再配線の必要があります。VHDL を使えば、手許にないロジック IC をその都度買って来る必要がなく、回路変更が容易であり、配線によるミスがなくなり、結果として回路製作時間の短縮が図れます。携帯電話など商品サイクルの速い製品のロジック IC 製作は、VHDL による開発が現在主流になっています。

この学習会では、開発環境を構築できるようになること、テキストに沿って学習し VHDL の書き方を理解すること、そして簡単な回路を製作することを目的とします。

教材：

「VHDL によるハードウェア設計入門」（長谷川裕泰著／CQ 出版社）
「見てわかる VHDL」（坂巻佳壽美著／工業調査会）

基礎知識

VHDL とは記述言語 ＝ ソフトウェア

FPGA とCPLD はデバイス

＝ ハードウェア

FPGA と CPLD を製造しているメーカーは、

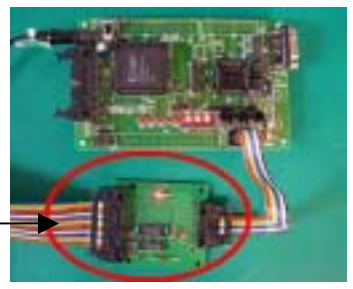
Xilinx（ザイリンクス）社と Altera（アルテラ）社が有名。

CPLD の例 ： Xilinx／XC9500 シリーズ
Altera／MAX シリーズ

FPGA の例 ： Xilinx／Virtex、Spartan シリーズ
Altera／FLEX、APEX シリーズ

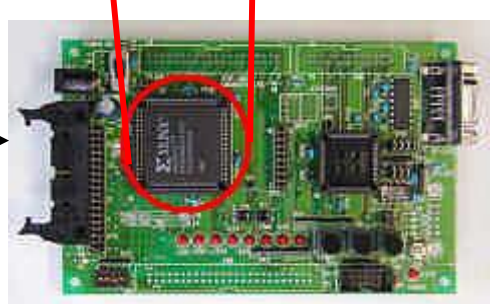
開発環境構築

■FPGA／CPLD 開発に必要なもの



１．パソコン
２．ダウンロードケーブル
　　今回購入したキットは¥6,500。自作も可能。

３．デバイス　：　FPGA／CPLD
　　今回は CPLD を使用。
　　CPLD　5V　44pin　¥600～¥1,350
　　84pin　¥1,350～¥3,600
　　3.3V　44pin　¥300～¥500
４．開発ツール（ソフトウェア）
　　無料で各メーカーなどの HP からダウンロード可能。



５．開発ボード（必ずしも必要ではない）
　　今回購入した開発ボードキットは¥14,300。
　　FPGA 開発ボードは面実装タイプが多い。

■FPGA／CPLD 購入時の注意点

１．FPGA／CPLD はメーカーにより仕様が異なる。そのデバイスに合った開発ツ―ルとダウンロードケーブルが必要。
２．FPGA は、内部構造が RAM のものが多く、外部に ROM が必要。面実装タイプが多く、大規模回路向き。CPLD は入門用に最適。
３．市販の CPLD には、5V 用に 44pin と 84pin、3.3V 用に 44pin がある。また、同じピン数でもマクロセル数が大きい程、複雑な回路を組める。　　（価格は全て平成 14 年 8 月 30 日現在の価格）

VHDL の基本構造

```
library ieee ;
use ieee.std_logic_1164.all ;
```

パッケージ呼び出し部

```
entity (entity 名) is
    <ポート文>
end (entity 名) ;
```

entity 宣言部
入出力ポートの定義

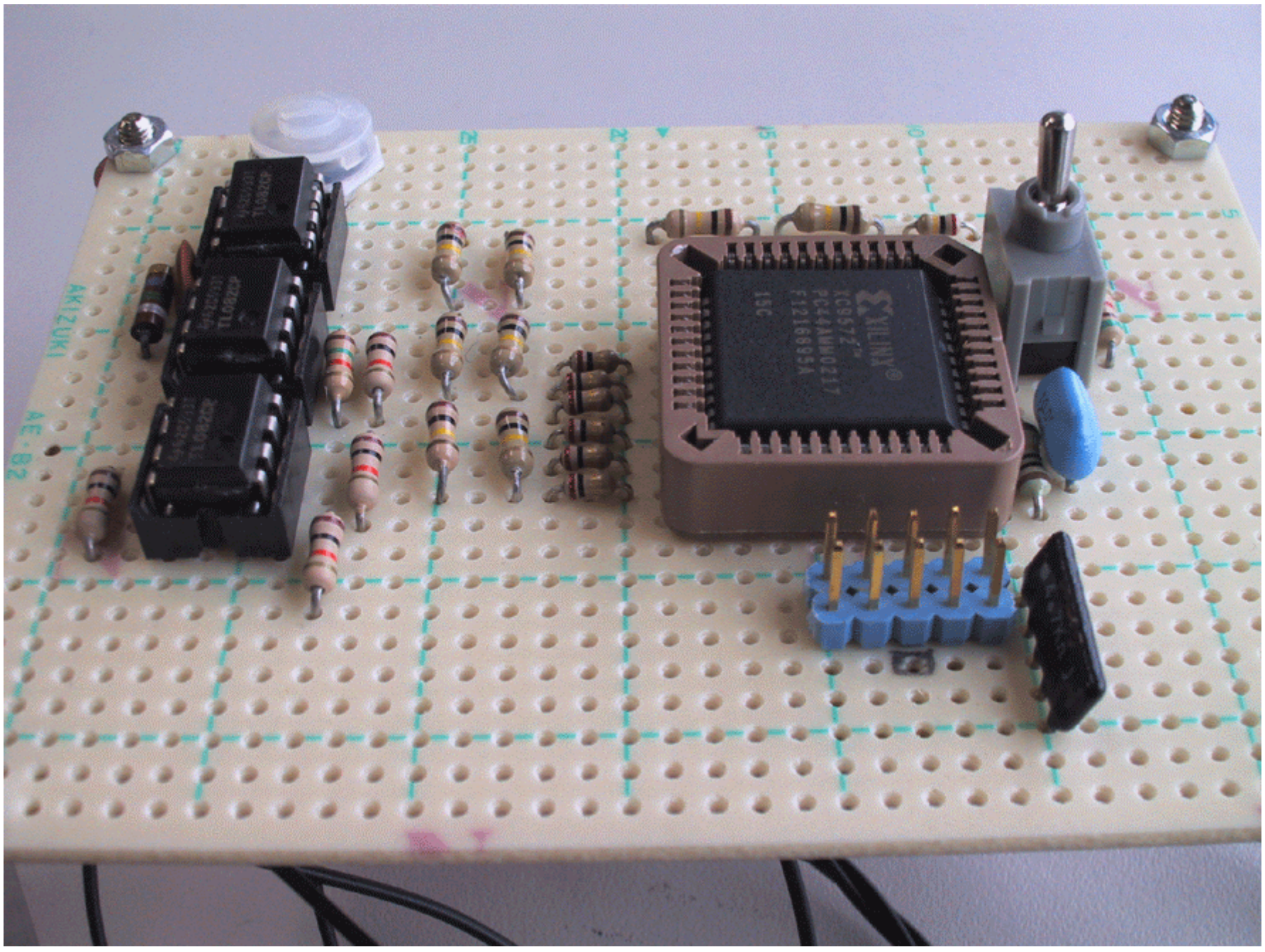
```
architecture (architecture 名) of (entity 名) is
    <宣言文>
begin
    <同時処理文>
    または<プロセス文>
end (architecture 名) ;
```

architecture 宣言部
内部回路の定義

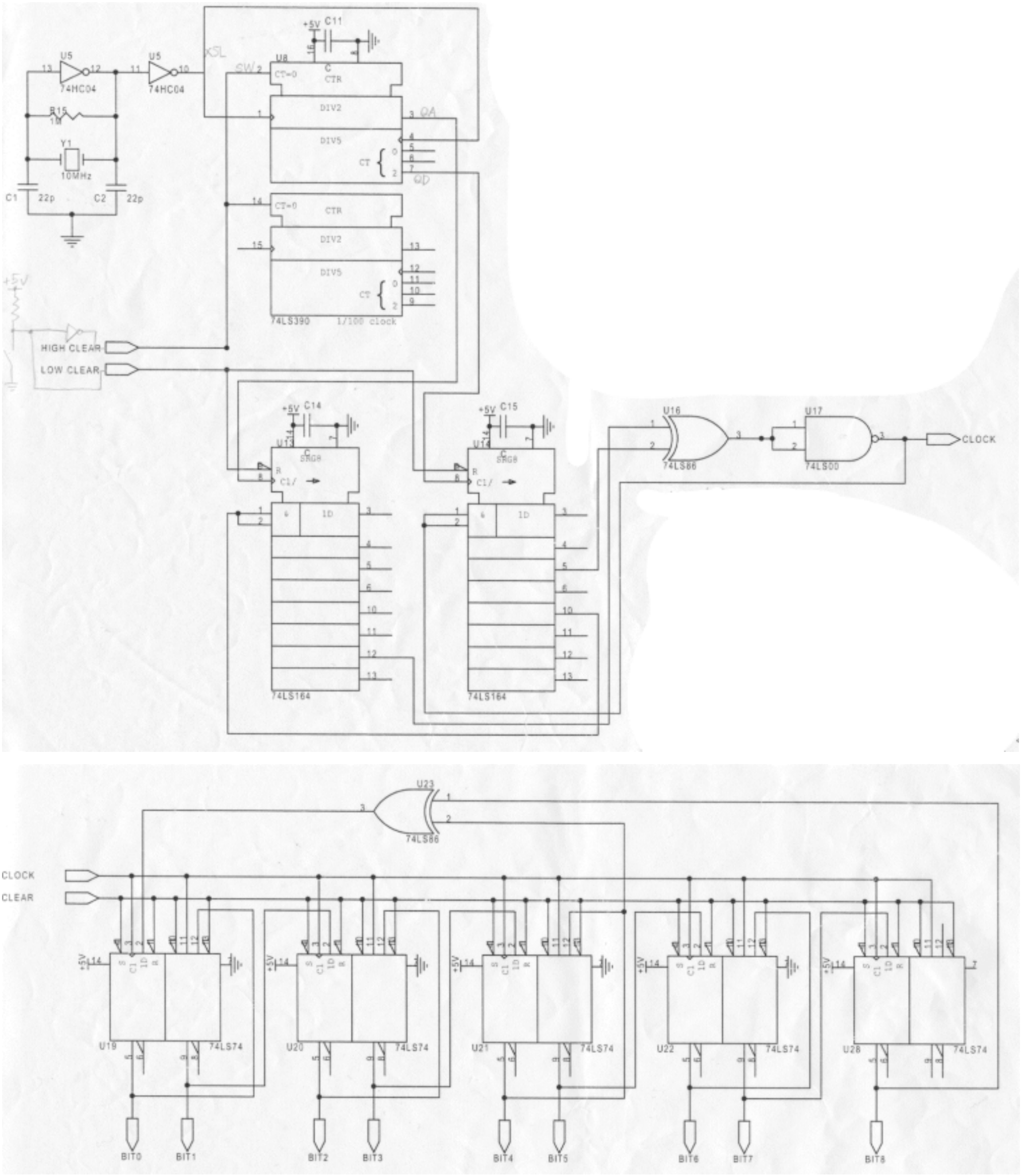
あとは以上に加えて、＜ポート文＞、＜宣言文＞、＜同時処理文＞、＜プロセス文＞の書き方を憶えるだけ。

回路の製作

今回、電気系共通実験室で製作していたノイズ・ジェネレーターを CPLD を使って簡略化することを試みた。下の写真が完成した回路である。以前は複数のロジック IC を使っていた部分を全て、1 つの CPLD に収めている。（図では茶のソケットにはまっているのが CPLD）



下の回路図は CPLD **内の回路図で、**Not **ゲート×**4**、**Exclusive OR **ゲート×**2**、**Dual Flip－Flops**×**9**、**Dual Decade Counters**×**1**、**8bit Shift Register**×**2 **で構成されている。**



```
--TOP Layer of M-Sequence Signal Generator(Noise Generator)-----

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity Ngenerator is
    Port ( RST : in std_logic;
           XSL : in std_logic;
           XL1 : in std_logic;
           XL2 : out std_logic;
           BT : out std_logic_vector(8 downto 0));
end Ngenerator;

architecture Behavioral of Ngenerator is

signal CLK, SW, QA, QC, QD, QE, QG, CLCK : std_logic;

component DFF
    Port ( CLK : in std_logic;
           CLR : in std_logic;
           Q : out std_logic_vector(8 downto 0));
end component;

component DDC
    Port ( CLK : in std_logic;
           CLR : in std_logic;
           QA : out std_logic;
           QD : out std_logic);
end component;

component SR8
    Port ( CLK : in std_logic;
           CLR : in std_logic;
                 SI : in std_logic;
           QC : out std_logic;
           QE : out std_logic;
           QG : out std_logic);
end component;

begin
 XL2 <= not XL1;
 SW <= not RST;
 CLK <= not XSL;
 CLCK <= not(QC xor QG);
 U0: DDC port map(CLK, SW, QA, QD);
 U1: SR8 port map(QA, RST, QE, open, open, QG);
 U2: SR8 port map(QD, RST, CLCK, QC, QE, open);
 U3: DFF port map(CLCK, RST, BT);
end Behavioral;
```

メイン
プログラム

7474
Dual Flip－Flops ×9ケ

74390
Dual Decade Counters

74164
8bit Shift Register

```
--74LS74 D-FFs x9-------------------------------------------------
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

7474
Dual Flip－Flops ×9ケ

```
entity DFF is
    Port ( CLR : in std_logic;
           CLK : in std_logic;
           Q : out std_logic_vector(8 downto 0));
end DFF;
```

```
architecture RTL of DFF is
signal CNT : std_logic_vector(8 downto 0);
begin
  Q <= CNT;
  process(CLK, CLR) begin
     if (CLR = '0') then CNT <=(others => '1');
       elsif(CLK'event and CLK = '1') then
         CNT(8 downto 1) <= CNT(7 downto 0);
         CNT(0) <= CNT(4) xor CNT(8);
       end if;
  end process;
end RTL;
```

```
--74LS390 Dual Decade Counters-----------------------------------
```

74390
Dual Decade Counters

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

```
entity DDC is
    Port ( CLK : in std_logic;
           CLR : in std_logic;
           QA : out std_logic;
           QD : out std_logic);
end DDC;
```

```
architecture RTL of DDC is
signal D2 : std_logic;
signal D5 : std_logic_vector(2 downto 0);
begin
  QA <= D2;
  QD <= D5(2);
  process(CLK, CLR) begin
     if(CLR = '1') then
         D2 <= '0';
       D5 <= "000";
       elsif(CLK'event and CLK='1') then
       D2 <= not D2;
           if(D5 = "100") then
         D5 <= "000";
           else
             D5 <= D5 + '1';
           end if;
       end if;
  end process;
end RTL;
```

```
--74LS164 8bit Shift Register-------------------------------------
```

74164
8bit Shift Register

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

```
entity SR8 is
    Port ( CLK : in std_logic;
           CLR : in std_logic;
                 SI : in std_logic;
           QC : out std_logic;
           QE : out std_logic;
           QG : out std_logic);
end SR8;
```

```
architecture RTL of SR8 is
signal CNT : std_logic_vector(7 downto 0);
begin
  QC <= CNT(2);
  QE <= CNT(4);
  QG <= CNT(6);
  process(CLK, CLR) begin
     if (CLR = '0') then CNT <=(others => '0');
       elsif(CLK'event and CLK = '1') then
         CNT(7 downto 1) <= CNT(6 downto 0);
            CNT(0) <= SI;
       end if;
  end process;
end RTL;
```