



FPGA DESIGN OF A PWM CONTROLLER

DESCRIPTION

Presented here is a design of an integrated circuit for a **PWM controller**. The chip is designed in **VHDL** (Very High speed integrated Hardware Description Language and is implemented on an **FPGA** (Field Programmable Gate Array)

FEATURES

- **Double buffered interface logic** allows configuration to **any bus width**, from **serial** to **parallel** of any width.
- Interface directly on a **dedicated data bus**, without bus interface logic, or on a **shared data bus**
- Instantiate the core as **stand-alone** or **cascaded**, in a daisy chain configuration.
- **Daisy chain multiple cores** on **one port address**.
- The interface logic effects **instantaneous Voltage and Direction** instead of changing one and then the other.
- Can be configured for **any oscillator frequency**
- **Multiple outputs**
- **AC or DC** applications
- Configurable **Dead-band** interval for AC inverter applications

APPLICATION

- DC chopper drive for 4-quadrant DC power/motor control in:-
 - CNC machines -machining,grinding of metal products
 - Rolling mills -manufacture of sheet metal
 - Electriytic cells -extraction of metals
 - DC Welding machines -joining of metals
 - DC Arc furnaces -manufacture of Steel
 - DC Plasma furnaces -manufacture of TAS steels
 - DC Electro slag refining furnaces -refining of steel
 - DC Heat treatment furnaces -hardening,tempering of metals
 - Robotics -motion and manipulator control
 - Servo valves -for flow control of fluids
 - Continuous casting machines -manufacture of wire-rods and flat metal products
 - Slitting machines -manufacture of flat metal products
 - Continuous pickling and annealing lines -manufacture of wire-rods and flat metal products
- Conversion to analog of digital signals such as :-
 - The output of an NCO to analog sine wave or a programmable tone generator.
 - The output of FIR,IIR filters.
- Generating a train of pulses for the gate drive of SCRs in thyristorised converter and inverter bridges.



VHDL Component Declaration:

```
COMPONENT P_WM
  GENERIC( IREG   : INTEGER:=1;
           RENS   : INTEGER:=1;
           GCF    : INTEGER:=20000000;
           CEF    : INTEGER:=20000000;
           CWD    : INTEGER:=16;
           EQL    : INTEGER:=1;
           IBD    : INTEGER:=4;
           OBD    : INTEGER:=4;
           NUM    : INTEGER:=2;
           DBC    : INTEGER:=0;
           CASC   : INTEGER:=0;
           OPN    : INTEGER:=0;
           BSW    : INTEGER:=4;
           UP     : INTEGER:=1);

  PORT( D        : IN          BUS2D( IBD-1 DOWNT0 0, BSW-1 DOWNT0 0)
        :=( OTHERS=>( OTHERS'0' ));
        CLKI     : IN          NODE:='0';
        RST      : IN          NODE:='1';
        DC       : IN          NODE:='1';
        ENB      : IN          NODE:='1';
        CS       : IN          NODE:='0';
        MAST     : IN          NODE:='0';
        CASI     : IN          NODE:='0';
        T        : BUFFER     BUS2D( OBD-1 DOWNT0 1, 1 DOWNT0 0);
        PWX      : BUFFER     NODE;
        CEN      : BUFFER     NODE;
        CASO     : BUFFER     NODE);
END COMPONENT;
```

FILES YOU GET

i) FUNC.DOC	-	Documentation of functions & data types used in the core.
ii) README.DOC	-	Compile and licensing information.
iii) PWM.DOC	-	This document
a) MYLIB.VHD	-	PACKAGE
b) P_WM.VHD	-	TOP HIERARCHY DESIGN FILE
c) M_DFF.VHD	-	DESIGN FILE BELOW TOP HIERARCHY
d) S_DFF.VHD	-	-DO-
e) P_AD.VHD	-	-DO-
f) R_SL.VHD	-	-DO-
g) I_NCDEC.VHD	-	-DO-
h) U_DCNT.VHD	-	-DO-
i) A_DSB.VHD	-	-DO-
j) D_BIL.VHD	-	-DO-
k) S_TFF.VHD	-	-DO-
l) S_JKF.VHD	-	-DO-
m) F_DIV.VHD	-	-DO-
n) B_SHIFT.VHD	-	-DO-
o) P_LSE.VHD	-	-DO-
o) S_TATE.VHD	-	-DO-



INTERFACED CONNECTION

These interface schemes described hereunder require the OPN parameter to be set to 1. The interface signals used here are described in the "INPUT PORTS" section. The data bus may be of any size from 1 bit serial, to parallel of any width. Its width is specified in the BSW parameter.

The data word required by the core is assembled internally from the sliced data it receives from the data lines and remains isolated from the core until it is fully assembled.

The width of the data word is specified in the CWD parameter. The MSB slice will load first, LSB will load last. A complete write cycle requires all the slices (NSL) required for the data word must to be sent contiguously. Each data word must be sent contiguously in the "SENDING ORDER" shown below. The cycle is repeated everytime data is to be written to the core.

The number of loads driven are specified in the NUM parameter. When NUM>1, send data for the first one first and then the others in sequence.

Calculation, of the number of slices of data the CPU needs to send (NSL), is shown below:-

NSL CALCULATION

Let $WSZ = \text{LOG}_2(\text{CWD} - 1) + 1$ -- Bits required to store CWD-1
 $ABC = WSZ / BSW$ -- ABC is truncated to an integer. If a remainder exists,
-- 1 should be added to it.

Then NSL is given by:-

```
DO CASE
  CASE BSW>2 AND WSZ<=BSW
    NSL = 2
  CASE BSW>2 AND WSZ>BSW
    NSL = ABC+1
  CASE BSW<=2 AND WSZ>BSW
    NSL = [2/BSW] - 1 + ABC
  CASE BSW<=2 AND WSZ<=BSW
    NSL = [2/BSW]
ENDCASE
```



DATA WORD SENDING ORDER

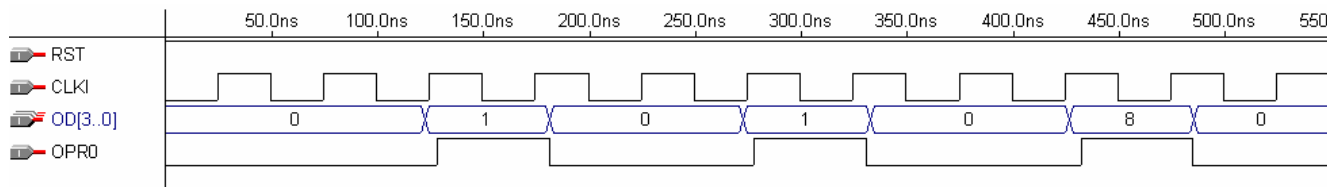
MSB slice will load first, LSB will load last. CONTROL WORD will load first, PWM COUNT will load last. The first driven load, will load first and the last one, last.

CS TIMING-SLICED DATA

Figure shows 9 bits of data, 118H, being loaded using a 4 bit data bus

OPN=0,BSW=4,CWD=512,GCF=20MHz,CEF=20MHz,EQL=0,IBD=1,OBD=1,NUM=1,DBC=0

CASC=0, NSL(setting)=1,NSL(internal)=3



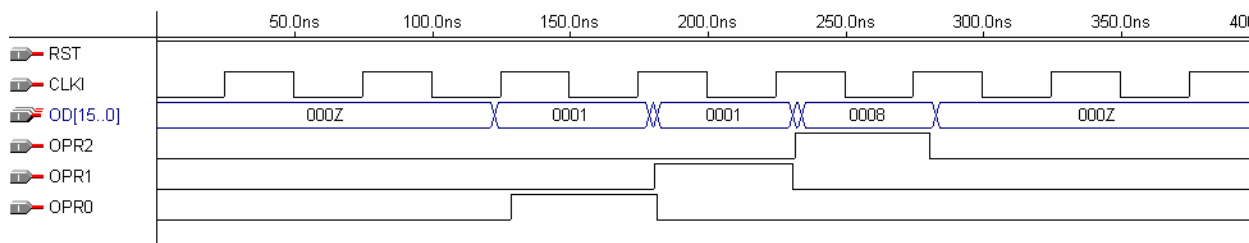
TIMING REQUIREMENTS

- a) Frequency of CS $\leq 1/3$ of CLKI
- b) Hi time of CS ≥ 1 CLKI
- c) Lo time of CS ≥ 1 CLKI

CS TIMING-UNSLICED DATA

Figure shows 9 bits of data, 118H, being loaded using a 4 bit data bus

OPN=1,BSW=4,CWD=512,NSL=3

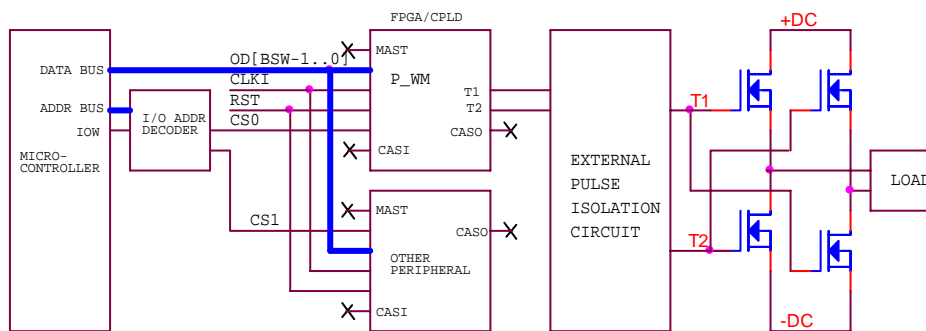


TIMING REQUIREMENTS

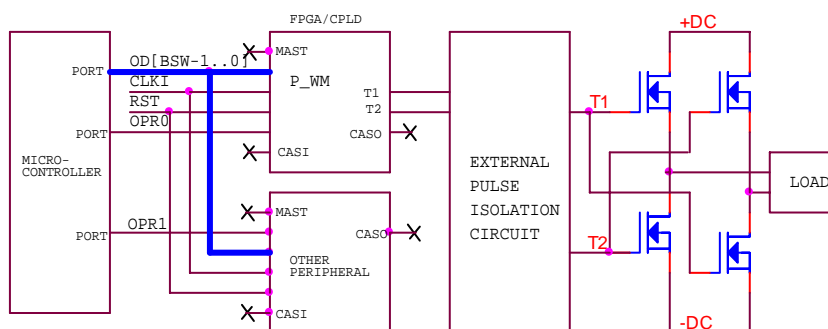
- a) Each CS[] input may come back to back one after the other with no gap in between.
- b) Hi time of each CS[] input ≥ 1 CLKI.



STAND-ALONE INTERFACE – SHARED BUS



STAND-ALONE INTERFACE – SHARED PORT



This option, enabled by connecting the MAST input of each core to '1' and the CASO output of each core to its CASI input, will interface the core to an internal or external CPU with a separate chip select for each core instantiated on the common data bus. Chip-select signals can be generated either by an i/o address decoder, within the PLD or from an output port on a micro-controller. Each chip select line brings along not only a logic overhead and additional pinouts but also the advantage of random access.

In this scheme the core is ready to accept data soon after RST=1. The core with an active CS signal will accept data and only one core at a time must be activated. The calculation of number of slices(NSL) and structure of the data word to be sent by the controller are shown above. The timing constraints for sending each data slice to the core are shown in fig 5. After a core receives all the entire data word, it initializes the data load logic to the first slice.

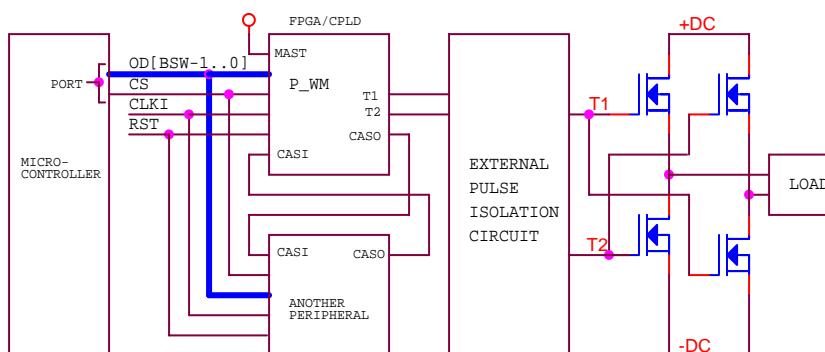


CASCADED INTERFACE

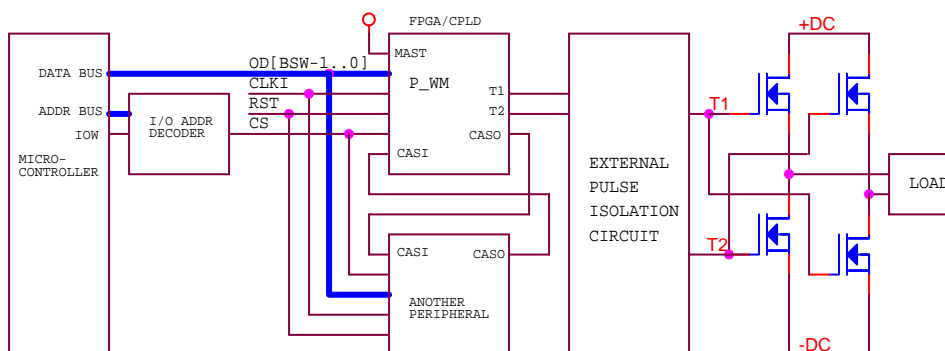
This option is enabled by connecting the MAST input of the first core in the chain to '1' and the CASO output of each core to the CASI input of the next core in the chain. The CASO output of the last core in the chain is connected to the CASI input of the master, or the first core in the chain.

This interface option will interface an internal or external CPU to all cores, instantiated on the common data bus, with a single chip select. Although the chip select logic overhead is minimum in this case, the cores cannot be accessed randomly and must be accessed sequentially one after the other and the entire sequence must be completed everytime, a process known as daisy chaining. The core, configured as the master will load first followed by the next in the chain. After a core receives its data, it pulses the CASO output, which being connected to the next core in the chain, enables it for data reception. After all the cores in the chain are loaded, the master is once again enabled.

CASCADED INTERFACE - SHARED PORT



CASCADED INTERFACE – SHARED BUS





INPUT REGISTER

When OPN=0, i/p data (D) may be registered internally(IREG>0), externally(IREG=0) or not registered(IREG=0).

When OPN=1, bus data may be registered internally(IREG>0) or not registered(IREG=0). Data from the bus remains isolated from the PWM logic until the last slice(as described above) is received. This isolation prevents synchronous transitions at the i/p of the subsequent data registers. Thus irrespective of the options chosen the following apply.

When data is registered with the CEN or PWX signals, timing violations will not occur. If data is unregistered, ensure that it settles after the falling edge of the CEN o/p, well before the next falling edge to avoid timing violations.

When i/p data is registered w/ CEN and EQU=0, any change will reflect immediately on the o/p, However when EQU=1, the first sawtooth intersection with i/p data will stay till the end of the carrier and subsequent changes in the i/p will be ignored.

Data should be registered w/ PWX only when manipulations on it prevent it from being registered with CEN.

The DC input is registered w/ PWX if IREG>0 . It is not registered w/ CEN since AC/DC mode cannot change between carriers.

INPUT PORTS

NAME	DESCRIPTION	WIDTH	DEPTH	COMMENTS
MAST	Master select	1	-	When the core is a master in a cascaded configuration(CASC=1), set MAST to Hi. In a cascaded configuration if it is not the master, set it Lo. In a stand-alone configuration(CASC=0), it is unused and may be left open.
CASI	Cascade in	1	-	If the core is in a cascaded configuration(CASC=1) and is not the master, connect CASI to CASO from the previous core in the chain, if it is the master, set it to CASO of the last core in the chain. In a stand-alone configuration(CASC=0) it is unused and may be left open.
CLKI	Clock	1	-	Positive edge triggered. Synchronizes all internal operations
RST	Reset	1	-	Asynchronous, active lo, resets all internal logic
DC	DC/AC	1	-	T[[]] outputs are DC(1) or AC(0). See table below.
ENB	Clk Enable	1	-	External Clock enable. Used instead of the internal signal if CEF>GCF
CS	Chip select	1	-	Active hi, enables the internal data load logic. Must be synchronous to the rising or falling edge of the CLKI input. Data latches internally at the first rising edge of CLKI after CS goes Hi. Only used when OPN<>0. See “ INTERFACE INFORMATION ” for timing constraints
D	Data bus	BSW	IBD	Data bus, for loading PWM COUNT and CONTROL WORD data.

OUTPUT PORTS

NAME	DESCRIPTION	DEPTH	WIDTH	COMMENTS
T	PWM output	OBD	2	PWM output. See table below
PWX	Carrier Enable	1	1	Combinational, 1 CLKI pulse at last CLKI of carrier period
CEN	Clock Enable	1	1	Combinational, 1 CLKI pulse at last CLKI of clock enable period
CASO	Cascade out	1	1	Used in a cascaded configuration (CASC=1), to enable the next core in the chain for data loading from the data bus. In a stand-alone configuration, it is unused and drives out Lo.



PARAMETERS-All INTEGER type, all required.

INTERFACE OPTION (OPN)

OPN	DESCRIPTION	COMMENTS
0	Direct interface	The data bus is assumed to always contain valid count data, chip select input (CS). is unused and the PWM output waveforms change at the instant this data changes.
1	Shared interface	When dedicated data bus not available, core can be configured for cascaded / stand-alone use

AC/DC OUTPUT (DC)

DC	CONTROL WORD		OUTPUTS		DESCRIPTION
	BIT1	BIT0	T[][1]	T[][0]	
1	0	0	0	0	DC WAVEFORM-O/P DISABLED, INTERNAL LOGIC ENABLED
1	0	1	PWM	0	DC WAVEFORM-CCW ROTATION
1	1	0	0	PWM	DC WAVEFORM-CW ROTATION
1	1	1	PWM	PWM	DC WAVEFORM-NORMALLY UNUSABLE
0	0	0	0	0	AC WAVEFORM- O/P DISABLED, INTERNAL LOGIC ENABLED
0	0	1	!PWM	0	AC WAVEFORM-PHASE RETURN (NORMALLY UNUSABLE)
0	1	0	0	PWM	AC WAVEFORM-PHASE (NORMALLY UNUSABLE)
0	1	1	!PWM	PWM	AC WAVEFORM-PHASE & PHASE RETURN, 180 DEG PHASE SHIFTED

DEAD BAND COUNT (DBC)

Let :-

$CEC = GCF / CEF$ -- Rounded to integer value

Then the number of CEN pulse delay in rising edge of o/p is given by $DCE :-$

$DCT = (DBC / CEC) * (1 - DC)$ -- Rounded to integer value

Where DBC (parameter) is the delay in CLKI pulses

NAME	DESCRIPTION	COMMENTS
BSW	Width of the D[]	Data bus width. Maximum limited only by compiler. When OPN=1, BSW>=0. When OPN=0, BSW>=[Log2(CWD-1)]+3, if less, it is padded, if more, truncated.
CASC	Cascaded/Stand-alone	Cascaded(1)/Stand-alone(0). Determines the behaviour of MAST and CASI inputs and the CASO output.
DBC	Dead-band count	The rising edge of each waveform a pair of T[][] eg T[0][1..0], is delayed by DBC clock pulses, to avoid switching on two power devices on the same leg, in a 3 phase AC inverter. See above for more info
NUM	# outputs	When DC=1, it is the number of DC loads connected. When DC=0 it is the number of phases. eg Set NUM=1 for 1 DC load or single phase AC o/p, Set NUM=3 for 3 DC loads or three phase AC o/p
OBD	Depth of T[][] o/p	Ideally OBD=NUM. When OBD>NUM, excess depth of T[][] are filled w/ 0. When OBD<NUM, the o/p is truncated
IBD	Depth of D[][] i/p	Ideally, IBD>=(1-OPN)*NUM. Excess depth is ignored. Less will cause error.
EQL	Comparator type	Type of comparator used to detect intersection of PWM COUNT w/ internal saw-tooth. When EQL=0 a "Less than" comparator is used, otherwise "Equal to" .
CWD	Carrier Width	Wavelength of internal saw-tooth carrier in terms of Clock enable pulses
CEF	Clock enable Freq	Frequency of the internal clock enable $CEF \leq GCF$. When $CEF > GCF$, the internal clk enable logic is disabled and the external clk enable (ENB) is used instead. When $CEF=0$, $CEF=GCF$. Determines time available for detection of intersection of PWM COUNT w/ sawtooth.
GCF	Clock Freq	Frequency in hertz of the CLKI input.
OPN	Interface option	See Table above
I REG	I/Ps to Register	Number of I/Ps to register, from 0 to IBD,. When 0, none are registered
RENS	I/P Register option See "I/P Register" above	When 1 - I/P data is registered w/ the CEN o/p When 0 - I/P data is registered w/ the PWX o/p
UP	I/P Polarity	Input is Unipolar(1) or Bi-polar(0). PWM O/Ps are disabled for UP=0



SAMPLE DESIGN (1)

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

LIBRARY MYLIB;
USE MYLIB.MYLIB.ALL;

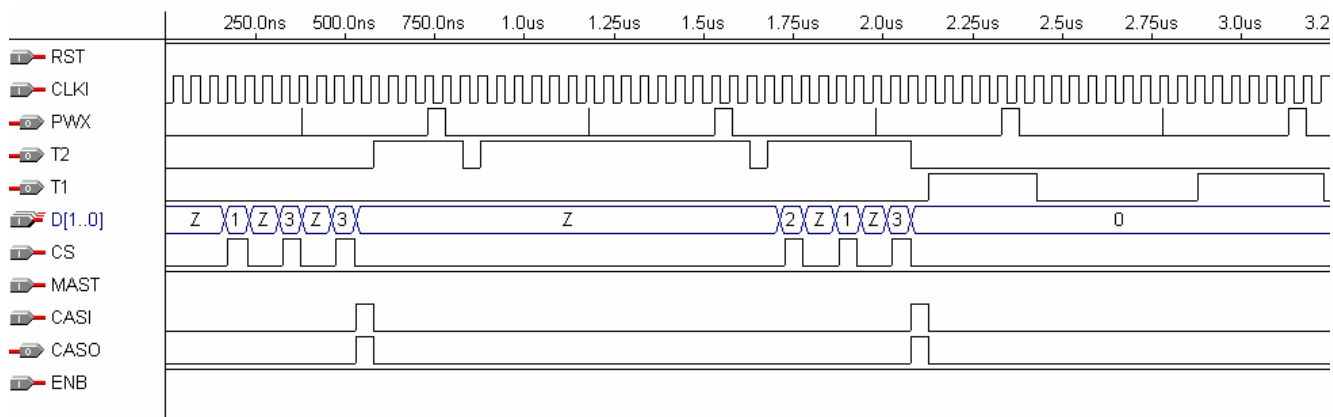
ENTITY MYTOP IS
  PORT(CLKI :IN NODE;
        RST :IN NODE;
        ENB :IN NODE;
        CS :IN NODE;
        D :IN BUS1D(1 DOWNT0 0);
        T :BUFFER BUS1D(2 DOWNT0 1);
        PWX :BUFFER NODE;
        CASO :BUFFER NODE
  );
END MYTOP;

ARCHITECTURE MYTOP OF MYTOP IS

BEGIN

A1: P_WM GENERIC MAP(IREG=0,GCF=>20000000,CEF=>20000000,CWD=>16,EQL=>0,IBD=>1,
                    OBD=>1,NUM=>1,DC=>1,DBC=>0,CASC=>0,OPN=>1,BSW=>2,UP=>1)
  PORT MAP(D,CLKI,RST,'1',CS,'1',OPEN,T,PWX,OPEN,CASO);
END MYTOP;
```

TIMING DIAGRAM – SAMPLE DESIGN (1)



SAMPLE DESIGN (2)

```
LIBRARY IEEE;
```



```
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

LIBRARY MYLIB;
USE MYLIB.MYLIB.ALL;

ENTITY MYTOP IS
  PORT(CLKI :IN  NODE;
        RST  :IN  NODE;
        CS   :IN  BUS1D(0 DOWNT0 0);
        D    :IN  BUS1D(3 DOWNT0 0);
        T    :BUFFER BUS1D(2 DOWNT0 1);
        PWX  :BUFFER NODE
  );
END MYTOP;

ARCHITECTURE MYTOP OF MYTOP IS

BEGIN
A1: P_WM GENERIC MAP(IREG=>0,GCF=>20000000,CEF=>20000000,CWD=>16,EQL=>0,IBD=>1,
                    OBD=>1,NUM=>1,DC=>1,DBC=>0,CASC=>0,OPN=>0,BSW=>4,UP=>1)
  PORT MAP(D,CLKI,RST,'1',CS,'0','0',T,PWX,OPEN,OPEN);
END MYTOP;
```

TIMING DIAGRAM – SAMPLE DESIGN (2)

