



SEQUENTIAL MULTIPLIER

DESCRIPTION

The S_MUL.VHD ipcore described here is a Sequential multiplier for signed or unsigned integer operands.

APPLICATION

- Used in a host of DSP applications
- The ALU of a fixed point CPU.
- Used in a host of DSP applications for communications and power control.

VHDL Component Declaration:

```
COMPONENT S_MUL
  GENERIC (
    WCND    : INTEGER;
    WPLR    : INTEGER;
    WRES    : INTEGER;
    EXTR    : INTEGER:=0;
    NRG     : INTEGER:=1;
    CLTI    : INTEGER;
    SIG     : INTEGER:=0;
    PLSI    : INTEGER:=1;
    WDL     : INTEGER:=1;
    REPC    : INTEGER;
    CLTA    : INTEGER:=0;
    CLTM    : INTEGER:=1;
    CLTS    : INTEGER:=1;
    SSIG    : INTEGER:=1
  );
  PORT (
    MCNX    : IN  BUS1D(WCND-1 DOWNT0 0):=(OTHERS=>'0');
    MPLX    : IN  BUS1D(WPLR-1 DOWNT0 0):=(OTHERS=>'0');
    BEG     : IN  NODE:= '0';
    CLKI    : IN  NODE:= '0';
    RST     : IN  NODE:= '0';
    COM     : IN  BUS1D(0 DOWNT0 0):=(OTHERS=>'0');
    RSLT    : BUFFER BUS1D(WRES-1 DOWNT0 0);
    RSLD    : BUFFER BUS1D(WRES-1 DOWNT0 0);
    DNL     : BUFFER BUS1D(WDL-1 DOWNT0 0);
    IOM     : BUFFER BUS1D(14 DOWNT0 0)
  );
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)SMUL.DOC - This document

- MYLIB.VHD - PACKAGE
- S_MUL.VHD - TOP HIERARCHY DESIGN FILE
- P_LSE.VHD - DESIGN FILE BELOW TOP HIERARCHY
- S_DFF.VHD - -do-
- F_DIV.VHD - -do-
- U_DCNT.VHD - -do-
- M_DFF.VHD - -do-
- I_NCDEC.VHD - -do-
- A_DSB.VHD - -do-
- S_JKF.VHD - -do-
- S_TFF.VHD - -do-
- P_AD.VHD - -do-
- I_P2IP.VHD - -do-
- S_TATE.VHD - -do-

OPERATION

The state machine starts at the end of the 1st CLKI pulse of the BEG input and consists of REPC loops of a multiplier. A loop starts with a START state of CLTS clks(1st loop) or MUX SETTling state of CLTM clks (2nd loop and onwards) allowing the operands for the multiplier to settle. This is followed by one or more iteration states of CLTI clks. A loop counter increments at the end of the last iteration state and indicates the end of a loop and the beginning of a new one. The last loop is followed by an END state of CLTA clks and then the END of LOOP pulse. The START, MUX-SETTLING and END states are inserted only if their duration is more than 0.

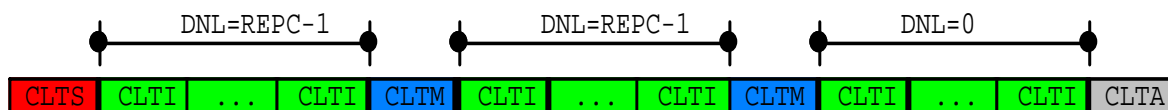
COMPUTATION TIME (CCL)

Note - SIG,WCND,WPLR are design parameters

```

WDX=SIG*(WPLR-1)+(1-SIG)*WPLR           --Width of Multiplier w/o sign
WDY=SSIG*SIG*(WCND-1)+(1-SIG*SSIG)*WCND --Width of Multiplicand w/o sign
IF WDY>WDX
  NITR= WDY                               --Number of iterations
ELSE
  NITR= WDX                               --Number of iterations
ENDIF
CCL=CLTI*NITR                             --Computation clocks

```



START	STATE
ITERATION	STATE
MUX SETTling	STATE
END	STATE



INPUT PORTS

NAME	DESC	WIDTH	COMMENTS
MCNX	Multiplicand	WCND	Signed/Unsigned integer
MPLX	Multiplier	WPLR	Signed/Unsigned integer. When SIG=1, the operand w/ less complex sign bit logic should be put here, for best synthesis results.
BEG	Start Process	1	Active hi, 1CLKI wide (or more), loads operands and starts loop.
CLKI	clock	1	Synchronizes all internal operations
RST	reset	1	Resets all internal registers
COM	-	16	For external control. IOM o/p from an external IP2IP component. See IP2IP.DOC

OUTPUT PORTS- Note: IOM(5),(6),(9),(10),(12),(13) are unused

NAME	DESC	WIDTH	COMMENTS
RSLT	Result ('Q' node)	WRES	'Q' node of Result register. Registered internally. Remains steady after IOM(8). When SIG=0 WRES is normally = WPLR + WCND. When WRES < WPLR + WCND, result is truncated When WRES > WPLR + WCND, result is padded with 0's. When SIG=1 WRES is normally = WPLR + WCND -SSIG -1. When WRES < WPLR + WCND -SSIG -1, result is truncated When WRES > WPLR + WCND -SSIG -1, result is sign extended to WRES bits.
RSLD	Result ('D' node)	WRES	Same as 'RSLT' o/p node, except it is the 'D' input of the Result register. May be registered externally with the IOM(8) output. Holds valid data only in the last iteration upto the end of IOM(8).
DNL	Loop Count	WDL	Replace phrase "Sequential Process" with "Multiplier" and read o/p ports with same name in "IP2IP.DOC"
IOM()	Control	16	

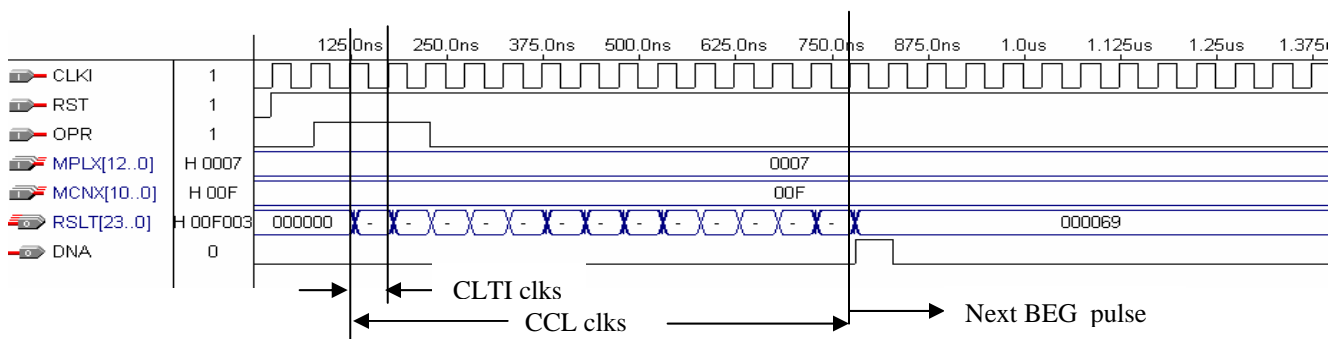
PARAMETERS (All integer type)

NAME	REQD	MIN	DESCRIPTION
WCND	YES	1	Width of multiplicand, input port
WPLR	YES	1	Width of multiplier, input port
WRES	YES	1	Width of result, output port
EXTR	NO	0	Make 1 for external control by an IP2IP component.
NRG	NO	0	Multiplicand is to be registered internally-Yes/No-1/0. If 'No', Multiplicand must be steady for CCL number of CLKI pulses after start of computation, as shown below. If 'YES' the multiplicand must be stable in the 1 st CLKI of the BEG input. Multiplier loading is not optional. It is loaded in the 1 st CLKI pulse of the BEG i/p, when it must be valid.
SIG	NO	0	Inputs are in signed 2's compliment format (1) or unsigned (0)

PLSI,WDL,REPC,CLTI,CLTA,CLTM,CLTS
Replace phrase "Sequential Process" with "Multiplier" and read parameters with same name in "IP2IP.DOC"

SSIG	NO	0	Both operands have same sign(1), different signs(0). Unused when SIG=0
------	----	---	--

TIMING DIAGRAM – WCND=11, WPLR=13, WRES=24





SAMPLE DESIGN

```
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;
          MCNX      :IN  BUS1D(10 DOWNT0 0);
          MPLX      :IN  BUS1D(12 DOWNT0 0);
          BEG       :IN  NODE;
          RSLT      :BUFFER BUS1D(23 DOWNT0 0);
          DNA       :BUFFER NODE);
END MYTOP;

ARCHITECTURE MYTOP OF MYTOP IS
BEGIN
A1: S_MUL GENERIC MAP(WCND=>11,WPLR=>13,WRES=>24,NRG=>1,SIG=>0,PLSI=>1
                     WDL=>0,REPC=>0,CLTA=>0,CLTM=>0,CLTS=>0,SSIG=>0)
    PORT      MAP(MCNX,MPLX,BEG,CLKI,RST,OPEN,RSLT,OPEN,OPEN,IOM(7)=>DNA);
END MYTOP;
```