



UNIVERSAL SHIFT REGISTER

DESCRIPTION

The B_SHIFT.VHD ipcore described here is a Universal Shift Register and Barrel shifter. It can be implemented in conventional memory, with sequential access, or registered memory with sequential or parallel access. A single bit or an array of bits, may be shifted, in either direction, at a time. Multiplexers at the input allow for multiple selection of Shift in data, load data, load address and read address.

FEATURES

- Shift any number of bits at a time
- Implement as 1D or 2D
- Implement in memory or registers
- Sequential or Paralel access
- Bi-directional Shift
- Flexible width and depth

FILES YOU GET

- i) FUNC.DOC - Documentation of functions & data types used in the core.
- ii) README.DOC - Compile and licensing information.
- iii) BSHIFT.DOC - This document

MYLIB.VHD	-	PACKAGE
B_SHIFT.VHD	-	TOP HIERARCHY DESIGN FILE
M_DFF.VHD	-	DESIGN FILE BELOW TOP HIERARCHY
S_DFF.VHD	-	-DO-
R_STK.VHD	-	-DO-
M_STK.VHD	-	-DO-
M_YMUX.VHD	-	-DO-
P_AD.VHD	-	-DO-
F_DIV.VHD	-	-DO-
U_DCNT.VHD	-	-DO-
A_DSB.VHD	-	-DO-
D_ECOT.VHD	-	-DO-
REG_MEM.VHD	-	-DO-
R_STK.VHD	-	-DO-



VHDL Component Declaration:

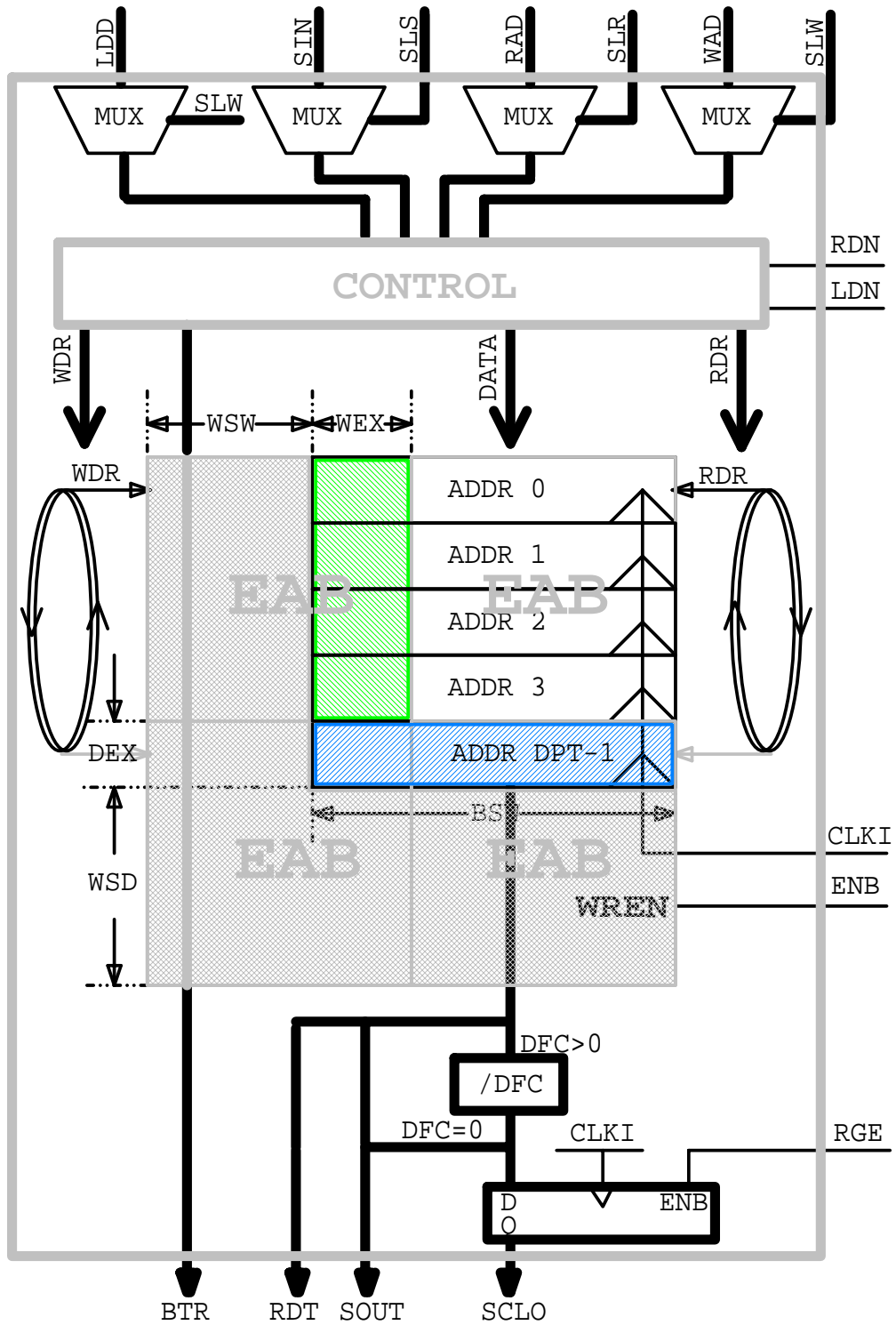
```
COMPONENT S_HIFT
  GENERIC(
    DPT      : INTEGER:=8;
    LWD      : INTEGER:=8;
    SDR      : INTEGER:=0;
    BSW      : INTEGER:=1;
    OPN      : INTEGER:=1;
    EOP      : INTEGER:=1;
    DVC      : INTEGER:=2;
    MXV      : INTEGER:=0;
    MSF      : INTEGER:=0;
    DFC      : INTEGER:=0;
    SEQ      : INTEGER:=0;
    NSS      : INTEGER:=0;
    NSR      : INTEGER:=0;
    NSW      : INTEGER:=0;
    LDA      : INTEGER:=0);
  PORT(
    D        : IN  BUS1D(LWD-1 DOWNT0 0) := (OTHERS => '0');
    * LDD     : IN  BUS2D(NSW*BLD-1 DOWNT0 0,LWD-1 DOWNT0 0)
      :=(OTHERS=>(OTHERS => '0'));
    CLKI    : IN  NODE:= '0';
    RST     : IN  NODE:= '1';
    PRN     : IN  NODE:= '1';
    ENB     : IN  NODE:= '0';
    LDN     : IN  NODE:= '0';
    RDN     : IN  NODE:= '0';
    RGE     : IN  NODE:= '0';
    SLS     : IN  BUS1D(NSS-1 DOWNT0 0):=(OTHERS=>'1');
    SIN     : IN  BUS2D(NSS-1 DOWNT0 0,BSW-1 DOWNT0 0):=(OTHERS=>'0');
    SLR     : IN  BUS1D(NSR-1 DOWNT0 0):=(OTHERS=>'1');
    * RAD    : IN  BUS2D(NSR*BLD-1 DOWNT0 0,LOG2(DPT-1) DOWNT0 0)
      :=(OTHERS=>(OTHERS => '0'));
    SLW     : IN  BUS1D(NSW-1 DOWNT0 0):=(OTHERS=>'1');
    * WAD    : IN  BUS2D(NSW*BLD-1 DOWNT0 0,LOG2(DPT-1) DOWNT0 0)
      :=(OTHERS=>(OTHERS => '0'));
    * OQ     : BUFFER BUS1D(BLD*BSW-1 DOWNT0 0);
    * OD     : BUFFER BUS1D(BLD*BSW-1 DOWNT0 0);
    SOUT    : BUFFER BUS1D(BSW-1 DOWNT0 0);
    * BTR    : BUFFER BUS2D(BLD-1 DOWNT0 0,Log2(DPT-1) DOWNT0 0);
    SCLO    : BUFFER BUS1D(MSF-1 DOWNT0 0);
    * RDT    : BUFFER BUS2D(BLD-1 DOWNT0 0,BSW-1 DOWNT0 0)
  );
END COMPONENT;

*BLD = (1-SEQ)*(DPT-1)+1;
```



STRUCTURE

Fig-1 SEQUENTIAL ACCESS(SEQ=1)





OPERATION

Load and Read operations occur on Blocks of data. A block is a vertical section of the storage stack. It is $BLD=(1-SEQ)*(DPT-1)+1$ words deep, where SEQ and DPT are parameters. A LOAD operation occurs while the LDN i/p is Hi and a SHIFT operation occurs otherwise. Both operations occur synchronous to CLKI at the end of the ENB pulse. The READ operation is asynchronous and occurs in parallel with other operations.

MODE	PARAMETER		INPUT			ACTION
	SEQ	LDA	LDN	RDN	ENB	
LOAD	0	0	Hi	X	Pulse	Load locations specified in the WAD[][] block selected
LOAD	0	1	Hi	X	Pulse	Ignore WAD[][] and load entire stack
LOAD	1	X	Hi	X	Pulse	Load location specified in the WAD[][] block selected
SHIFT	X	X	Lo	X	Pulse	Data in the SIN[][] record selected is
READ	0	X	X	0	X	RDT contains the contents of the whole stack
READ	1	X	X	0	X	$RDT[0][]=SOUT[]$.
READ	X	X	X	1	X	RDT[][] contains the contents of the stack whose locations are specified in the RAD[][] block selected

LOAD

A hi bit in the SLW[] i/p selects a block of data and addresses for the operation. If a value in the selected address block exceeds DPT-1, no operation occurs.

READ

A hi bit in the SLR[] i/p selects a block of addresses for the operation. If a value in the selected address block exceeds DPT-1, zero is read for that location.



INPUT PORTS

Name	Width X Depth	Description	Comments
D	LWD X 1	Load Data-1D	1D load data to stack.If LWD is not as specified below, the width is padded or truncated, as necessary. If port is unused, leave unconnected or grounded. ORed w/ the 'LDD' i/p.Loads stack at the + edge of CLKI while ENB=Hi & LDN=Hi When OPN=0 LWD>=DPT*BSW. Converted to 2D. Stack locations loaded are specified in the address block selected in WAD i/p When OPN=1 LWD>=BSW. Stack locations loaded are specified in the address block selected in WAD i/p.
LDD	LWD x (NSW*BLD)	Load Data-2D	2D version of the 'D' input described above:- If LWD is not as specified below, the width is padded or truncated, as necessary. If LDD is unused, leave unconnected or grounded. ORed w/ the 'D' i/p. When OPN=0 When OPN=1 LWD>= BSW. LWD>=BSW.
CLKI	1	Clock	Synchronizes all internal operations
RST	1	Async clear	Active lo, Register Clear. When OPN=1, device memory used can't be cleared
PRN	1	Async preset	Active lo, Shift register Preset. When OPN=1 Device memory used can't be set.
ENB	1	Clk Enb	One CLKI wide,active hi,enable for Shift or Load operation.See " Operation "
LDN	1	Load	When Hi, loads data from D[] or LDD[[]].See " Operation "
RDN	1	Read	When Hi, reads the stack into RDT[[]].See " Operation "
SIN	BSW x NSS	Shift In Data	Data for Shifting into stack in SHIFT mode. See "Operation"
RAD	$\log_2(DPT-1)+2$ x (NSR*BLD)	Read Address	Address of memory location, whose contents are put on the RDT[[]] o/p in READ mode. See " Operation ". Width of RAD[[]] is 1 bit larger than that required to store the largest stack address(DPT-1).Thus a when RAD[[]]>DPT-1,the data read is 0.
WAD	$\log_2(DPT-1)+2$ x (NSW*BLD)	Write Address	Address of memory location, written to the stack in LOAD mode. See " Operation ". Width of WAD[[]] is 1 bit larger than that required to store the largest stack address(DPT-1).Thus a when WAD[[]]>DPT-1, a write doesn't occur.



INPUT PORTS (Contd)

Name	Width X Depth	Description	Comments
RGE	1	Reg Enb	Enables a register for the SCLO[] output.
SLS	NSS x 1	Select	Decoded mux select. A high bit in SLS[] selects the corresponding vector in SIN[1][]. Thus SLS[1] selects SIN[1][] etc.
SLR	NSR x 1	Select	Decoded mux select. A high bit in SLR[] selects the corresponding block of BLD vectors in RAD[1][]. See MYMUX.doc w/ DP=BLD
SLW	NSW x 1	Select	Decoded mux select. A high bit in SLW[] selects the corresponding block of BLD vectors in WAD[1][] and LDD[1][]. See MYMUX.doc w/ DP=BLD

OUTPUT PORTS

Name	Width X Depth	Description	Comments
OQ	BLD*BSW	Register Data-1D	Parallel output of 'Q' node of Shift Register, in 1D format. Unused when OPN=1
OD	BLD*BSW	Register Data-1D	Parallel output of 'D' node of Shift Register, in 1D format. Unused when OPN=1
SOUT	BSW	Shift Out	Data Shifted out SHIFT mode. See "Operation"
BTR	Log2(DPT-1)+1 x BLD	Shift Address	Memory Rd/Wr Address for Shift operation. When SEQ=1 this is the address of the SOUT o/p port. When SEQ=0, BTR[1][]=0 to DPT-1 thus BTR[0][]=0, BTR[1][]=1, ..., BTR[DPT-1][]=DPT-1.
SCLO	MSF x 1	Shift Out	When DFC>0: Scaled(divided by DFC), registered version of SOUT[] When DFC=0: Registered version of SOUT[]
RDT	BSW x BLD	Read Data	Data read in READ mode. See "Operation"

PARAMETERS

Name	Description
DPT	Depth of Storage Stack
BSW	Width of Storage Stack and number of bits shifted in parallel.
LWD	Width of 'D', 'LDD' input ports. Normally LWD>=BSW. If not port is truncated or padded as required. Unused if ports are unused
SDR	Shift direction-Shift out Top of Stack 1st(1), Bottom of Stack 1st(0)
MXV	Max positive value of SCLO[]. Used when DFC>0
MSF	Width of SCLO[] o/p
DFC	Scale factor to divide SOUT[] by to generate SCLO[]. Factor=1, when DFC=0. DFC>0 assumes that I/Ps D[] & LDD[1][] are signed and O/P SCLO[] is unsigned
SEQ	Stack Load and Read operations are Sequential(1)/Parallel(0)
DVC	Unused when OPN=0. Device Family - CYCLONE(0), FLEX10K(1), ACEX(2)
OPN	Implement in Register(0)/Memory(1)
EOP	An array of 2 integers, each 2 digits wide. unused when OPN=0. Structure is :- (DR2,WR2) eg if DR2=2 & WR2=1, EOP=21. See table below.
LDA	LDA=1: Ignore WAD[1][] and assume WAD[i][]=i, where i=stack index. In other words load all locations of stack w/ contents of LDD[1][], D[1]. This option applies only when SEQ=0, LDA is unused otherwise.
NSS	Width of SLS & depth of SIN i/p. Unused if ports are unused.
NSR	Width of SLR & number of BLD depth Blocks in RAD i/p. See "Operation". Unused if ports are unused.
NSW	Width of SLW & number of BLD depth Blocks in WAD, LDD & D inputs. See "Operation". Unused if ports are unused.



ELEMENTS of EOP ARRAY

WR2	<p>Optimize memory block width to - LOWER(0),NEAREST(1),UPPER(2). If the slice width BSW overflows into an adjacent memory block (Fig 5,6) the WEX and WSW bits of the block are used as follows(if part of depth is implemented in registers WR0,WR2 are ignored and WR2 is forced):-</p> <p><u>WR2=0</u> Adjacent horizontal EAB not used, WEX bits of width placed in registers</p> <p><u>WR2=1</u> When WEX>=WSW, WR2=2 is applied, otherwise WR2=0.</p> <p><u>WR2=2</u> Adjacent horizontal EAB used, WSW bits of EAB width are wasted and zero bits placed in registers.</p>
DR2	<p>Optimize memory block depth to - LOWER(0),NEAREST(1),UPPER(2). If the slice width BSW overflows into an adjacent memory block (Fig 5,6) the DEX and WSD bits of the block are used as follows:-</p> <p><u>DR2=0</u> Adjacent vertical EAB not used, DEX bits of depth placed in registers</p> <p><u>DR2=1</u> When DEX>=WSD, DR2=2 is applied, otherwise DR2=0.</p> <p><u>DR2=2</u> Adjacent vertical EAB used, WSD bits of EAB depth are wasted and zero bits placed in registers..</p>

SAMPLE DESIGN-1

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

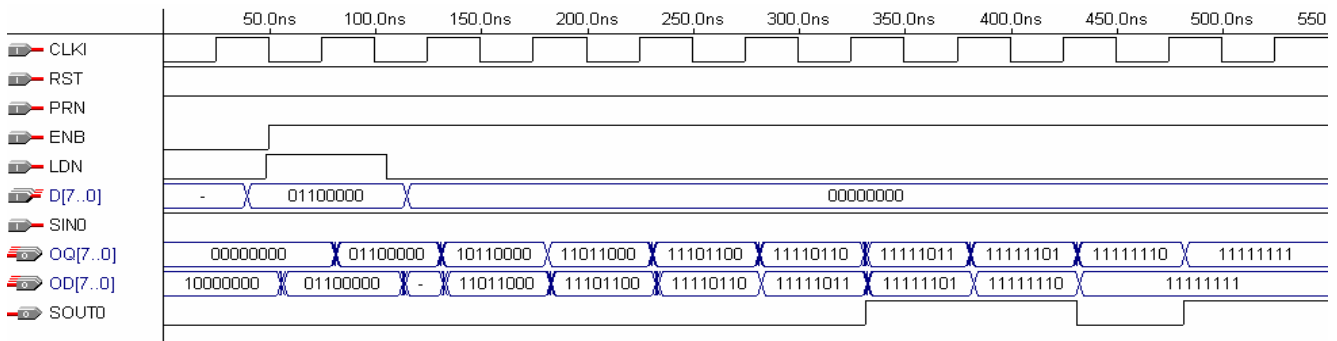
```
LIBRARY MYLIB;
USE MYLIB.MYLIB.ALL;
```

```
ENTITY MYTOP IS
  PORT(CLKI :IN NODE;
        RST :IN NODE;
        PRN :IN NODE;
        LDN :IN NODE;
        ENB :IN NODE;
        SIN :IN BUS1D(1 DOWNT0 0);
        D :IN BUS1D(15 DOWNT0 0);
        OQ :BUFFER BUS1D(15 DOWNT0 0);
        OD :BUFFER BUS1D(15 DOWNT0 0);
        SOUT :BUFFER BUS1D(1 DOWNT0 0)
  );
END MYTOP;
```

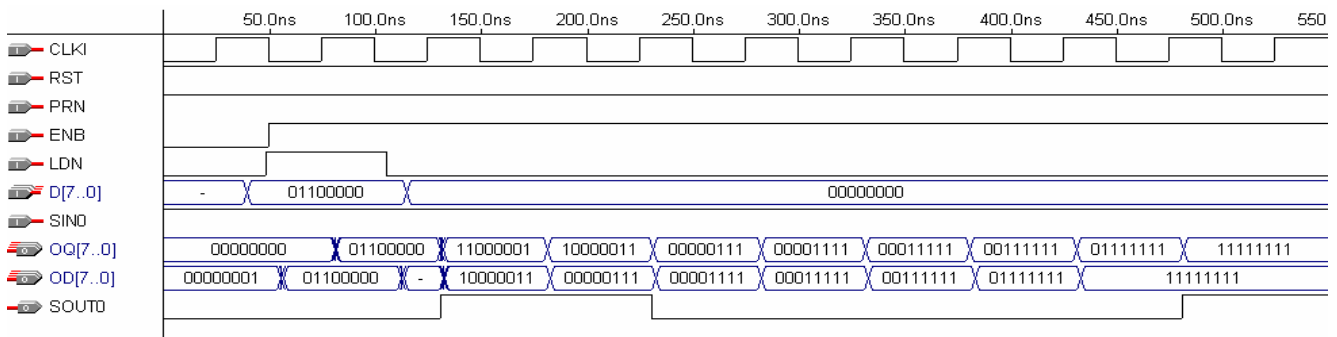
```
ARCHITECTURE MYTOP OF MYTOP IS
BEGIN
  SOUT(1)<='0';
A1: B_SHIFT GENERIC MAP (DPT=>16,LWD=>16,SDR=>0,BSW=>1,OPN=>0,DFC=>0,MXV=>0,
                          MSF=>0,SEQ=>0,DVC=>0,EOP=>0,LDA=>0,NSS=>1,NSR=>0,NSW=>1)
  PORT MAP (D=>D,CLKI=>CLKI,RST=>RST,PRN=>PRN,ENB=>ENB,LDN=>LDN,
            SIN=>SIN(0),OQ=>OQ,OD=>OD,SOUT=>SOUT(0));
END MYTOP;
```



TIMING DIAGRAM – SAMPLE DESIGN 1 with SDR=0, BSW=1



TIMING DIAGRAM – SAMPLE DESIGN 1 with SDR=1, BSW=1



SAMPLE DESIGN-2

```

ARCHITECTURE MYTOP OF MYTOP IS
BEGIN
A1: B_SHIFT GENERIC MAP (DPT=>16,LWD=>16,SDR=>1,BSW=>2,OPN=>0,DFC=>0,MXV=>0,
MSF=>0,SEQ=>0,DVC=>0,EOP=>0,LDA=>0,NSS=>1,NSR=>0,NSW=>1)
PORT MAP (D=>D,CLKI=>CLKI,RST=>RST,PRN=>PRN,ENB=>ENB,LDN=>LDN,
SIN=>SIN,OQ=>OQ,OD=>OD,SOUT=>SOUT);
END MYTOP;

```

TIMING DIAGRAM – SAMPLE DESIGN 2 with SDR=1, BSW=2

