



# FIFO BUFFER

## DESCRIPTION

FIFO buffers are used for optimization of data throughput between communicating devices. Features unique to this core, such as synchronous operation, upper and lower threshold signals and implementation in memory, registers or a combination thereof, make it possible to use it as a Pre-fetch instruction que in a CPU design or in DSP applications such as FFTs Convolution, digital filters etc. May be implemented in FPGA CPLDs or ASICs, with optimal resource usage.

## FEATURES

- Que level count output.
- Full, Empty, Almost Full and Almost Empty threshold signals.
- Implement in **memory**, **registers** or **memory/register** combination.
- **Parallel output** of entire stack available

## APPLICATION

- Communications networks such as ATM, ETHERNET
- Bus controllers such as PCI, I2C
- Communication ports such as USB, UART
- Interface logic such as serial and parallel A to D converters
- Serial memories such as FLASH, EPROM etc
- CPU, Pre-fetch instruction que
- DSP applications such as FFTs and Digital Filters such as FIR, IIR.
- Image Processing applications such as DCT and Convolution



## **VHDL COMPONENT DECLARATION:-**

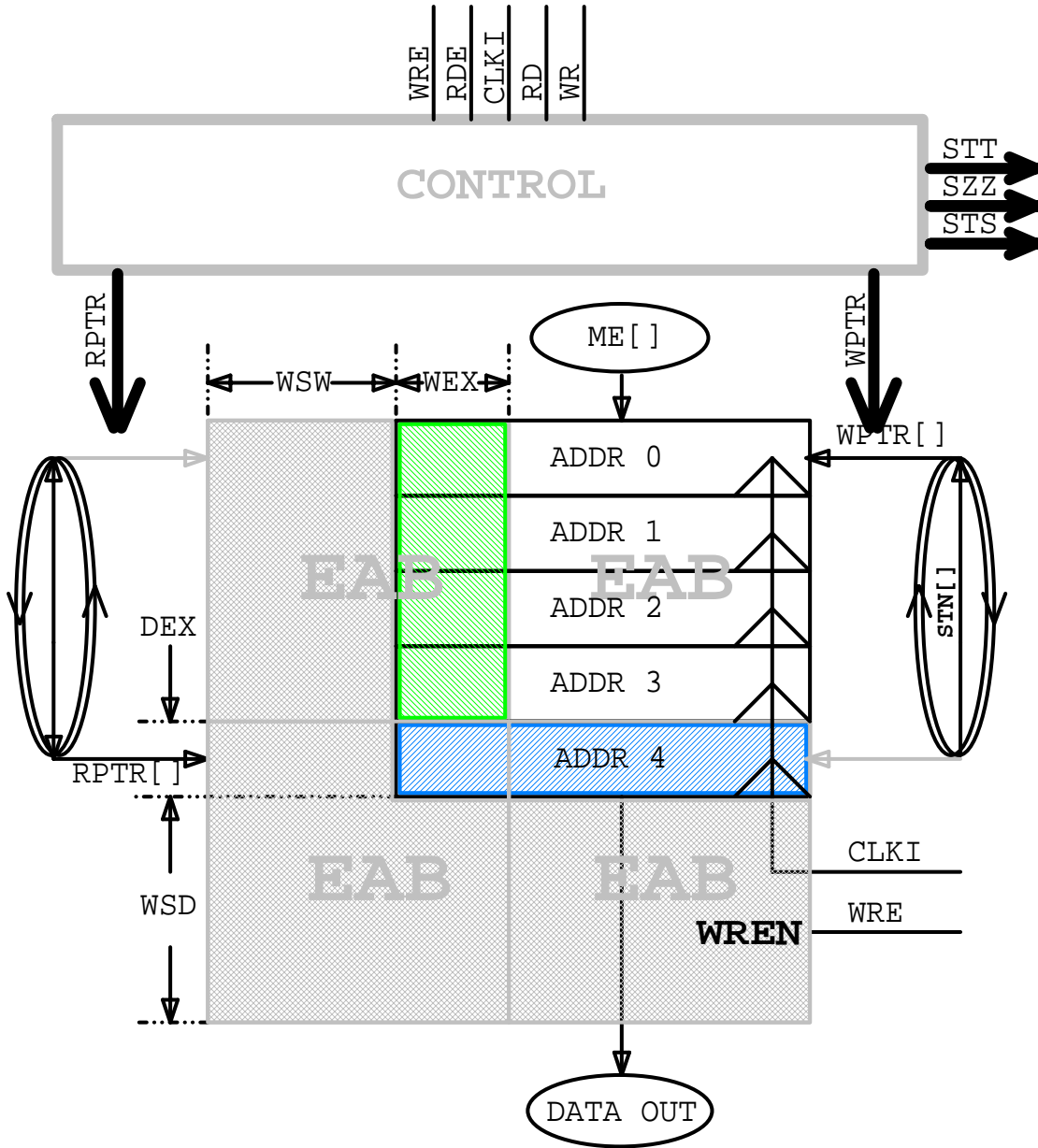
```
COMPONENT F_IFO
  GENERIC (
    OPN      : INTEGER := 0;
    WID      : INTEGER := 16;
    QDI      : INTEGER := 4;
    QDE      : INTEGER := 4;
    IREG     : INTEGER := 0;
    EOP      : INTEGER := 1;
    DVC      : INTEGER := 2;
    SWD      : INTEGER := 16);
  PORT(
    RD       : IN  NODE:= '0';
    WR       : IN  NODE:= '0';
    RDE      : IN  NODE:= '0';
    WRE      : IN  NODE:= '0';
    CLR      : IN  NODE:= '0';
    ME       : IN  BUS1D(WID DOWNT0 0) :=(OTHERS=>'0');
    CLKI     : IN  NODE:= '0';
    SZZ      : BUFFER BUS1D(3 DOWNT0 0);
    STT      : BUFFER BUS1D(3 DOWNT0 0);
    QUE      : BUFFER BUS1D(WID DOWNT0 0);
    STS      : BUFFER BUS1D(15 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)FIFO.DOC - This document
MYLIB.VHD - PACKAGE
F_IFO.VHD - TOP HIERARCHY DESIGN
M_DFF.VHD - FILES USED IN F_IFO
S_DFF.VHD - -do-
B_DFF.VHD - -do-
D_EC0D.VHD - -do-
I_NCDEC.VHD - -do-
A_DSB.VHD - -do-
M_STK.VHD - -do-
```

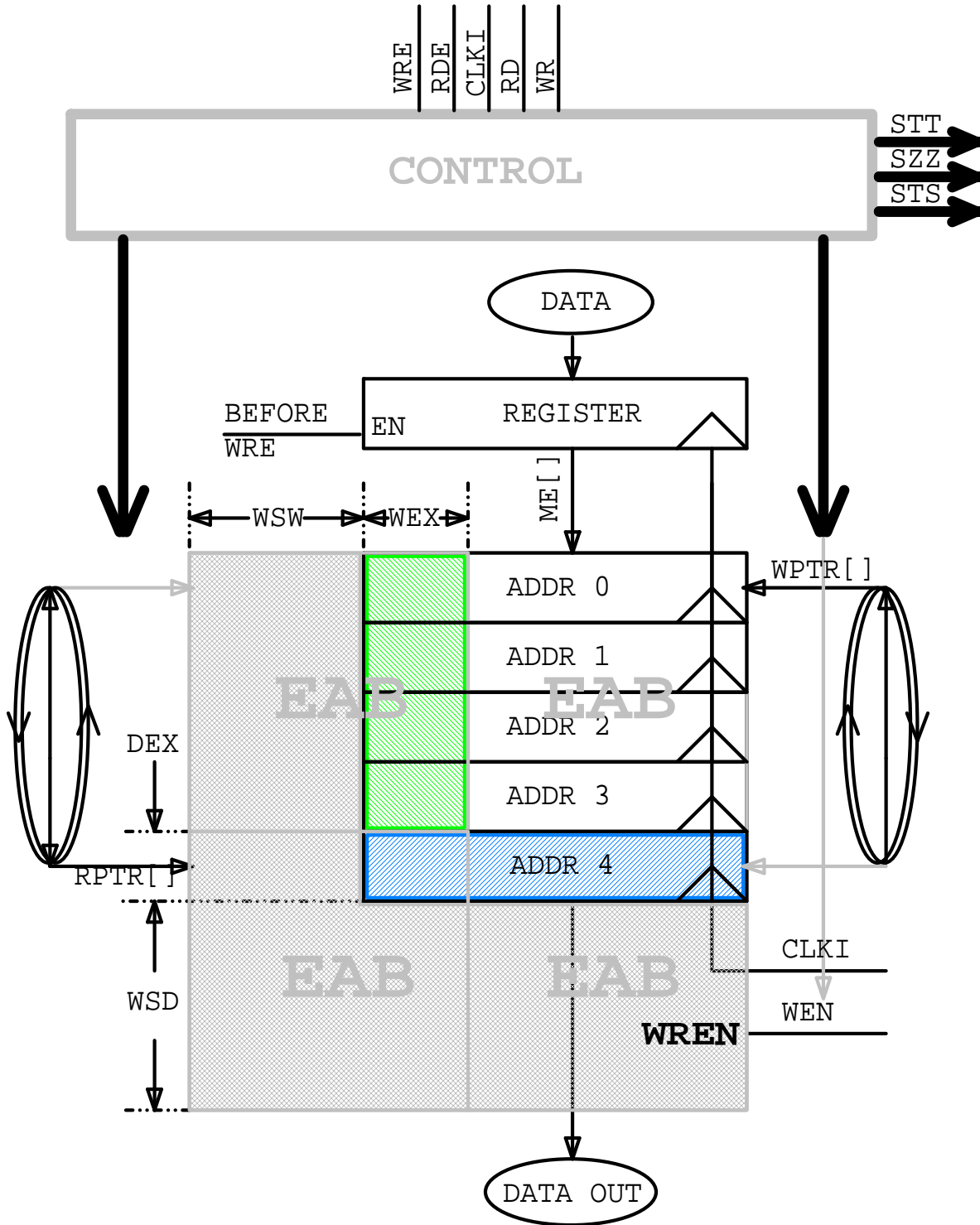


**Fig 5 - STRUCTURE FIFO (QDE=0)**





**Fig 6 - STRUCTURE of FIFO (ODE=1)**





### OPERATION of FIFO

The FIFO operates like a compartmentalized bucket of water w/ flow controlled by unidirectional valves. Water enters through the feed valve (WRE). The feed valve valve branches out to the compartments, through valves which open when the corresponding compartment is empty and the one below is full (valve# WPTR[]), water then enters when the feed valve is opened.

Water exit procedure is somewhat different from the Bucket type FIFO, in that, the central exit route, connecting all the compartments is missing. The valve connected to the lowermost full compartment(valve# RPTR[]) is opened, water then exits when the RDE valve is opened. The bucket therefore empties counter intuitively, **bottom up**.

The external reservoir, when present(QDE=1), may be filled, by opening the entry valve, whenever it is empty(SZZ[1]/ STT[1]-Lo), under user control. Any time after it is filled, the feed valve may be opened, to fill the bucket.

Fig 7 - Bucket Analogy (QDE=1)

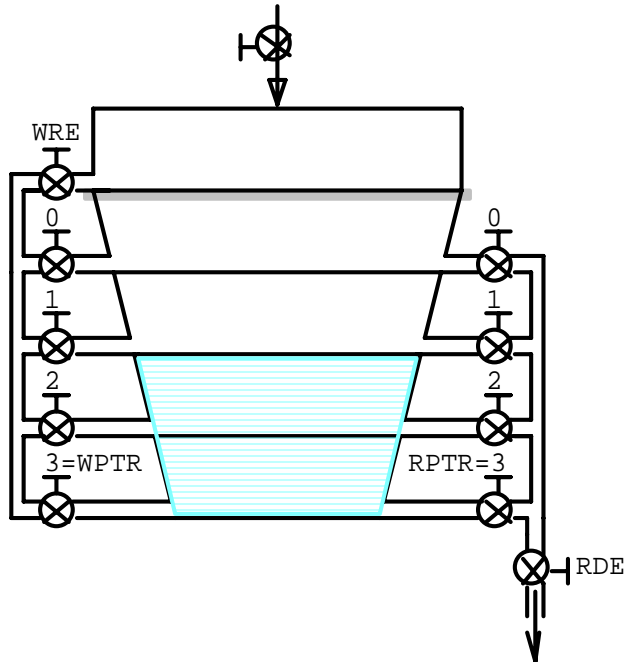
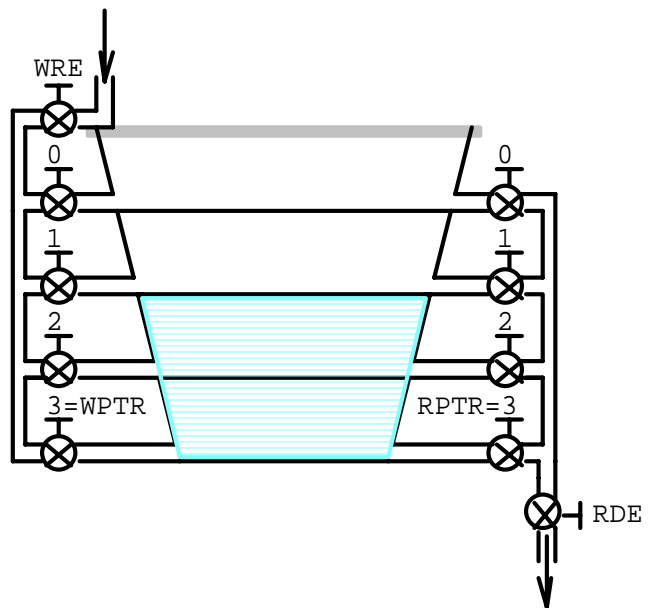


Fig 8 - Bucket Analogy (QDE=0)





### INPUT PORTS(All are Active hi)

NAME	DESC	WIDTH	COMMENTS
ME	Input data	WID	Data input to FIFO
WR	Write	1	ME[] written to FIFO. When IREG=0, + edge may occur at or before + edge of WRE, -edge must be coincident w/ - edge of WRE. Provides settling time to internal logic before WRE. When IREG=1, WR is registered internally. The registered o/p must adhere to the above constraints.
RD	Read	1	Oldest data in FIFO put on QUE. When IREG=0, + edge may occur at or before + edge of RDE, - edge must be coincident w/ - edge of RDE. Provides settling time to internal logic before RDE. When IREG=1, RD is registered internally. The registered o/p must adhere to the above constraints.
WRE	Write Enb	1	Write enable. 1 CLKI wide, See WR
RDE	Read Enb	1	Read enable. 1 CLKI wide, See RD
CLR	Reset	1	Asynchronous. Resets the STS, SZZ and STT outputs to empty and resets the FIFO stack. When OPN=1 the part of the FIFO stack implemented in EABs can't be reset and is undefined
CLKI	Clock	1	Positive edge triggered. Synchronizes all internal operations.

\*NOTE :- If WR and RD are 1 CLKI wide, they may be shorted to WRE and RDE respectively

### OUTPUT PORTS

NAME	DESC	WIDTH	COMMENTS
SZZ	Status Word	4	FIFO status at the end of a command, in progress, is indicated in this port from the start of the command to after its end. See 'Status Word'
STT	Status Word	4	FIFO status at the end of a command ie Current status. See 'Status Word'
QUE	Data o/p	WID	Data output from FIFO
STS	FIFO Level	SWD	Number of words present in FIFO or its level. Ranges from 0 to DD

1>COMMAND - From the rising edge of WR or RD signals to their falling edge

2>DD - QDI+QDE

### STATUS WORD SZZ[i],STT[i]

i	DESCRIPTION
0	Almost empty->Words left<=1
1	Full->Words left==DD
2	Empty->Words left==0
3	Almost full ->Words left>=DD-1

### FUNCTION

clr	clki	rd	rde	wr	wre	Function
L	↓	X	X	X	X	No change (requires ↑ clki edge)
H	X	X	X	X	X	Resets FIFO to empty. See CLR port
L	X	L	X	L	X	No change
L	X	X	L	X	L	No change
L	↑	H	H	H	H	Write data to FIFO, read FIFO, update QUE,SZZ,STT,STS
L	↑	H	H	L	L	Read FIFO, update QUE,SZZ,STT,STS
L	↑	L	L	H	H	Write data to FIFO, update SZZ,STT,STS
L	↑	H	H	H	L	Incorrect operation
L	↑	H	H	L	H	Incorrect operation
L	↑	H	L	H	H	Incorrect operation
L	↑	L	H	H	H	Incorrect operation
L	↑	L	H	H	L	Incorrect operation
L	↑	H	L	L	H	Incorrect operation



## PARAMETERS-(all INTEGER type)

NAME	DESCRIPTION
OPN	Implement in Register(0)/Memory(1)
WID	Width of ME and QUE ports and of FIFO stack.
QDI	Depth of FIFO stack
QDE	FIFO is fed by an external buffer, loaded on or before +edge of WRE- Yes(1)/No(0)
IREG	WR,RD inputs to be internally registered- Yes(1)/No(0)
DVC	Unused when OPN=0.Device Family - CYCLONE(0),FLEX10K(1),ACEX(2)
SWD	Width of STS o/p. Normally SWD>= MSB(DD)+1, if not, STS is padded or truncated as necessary.
EOP	An integer array, 1 digit wide X 2 digits deep. unused when OPN=0. Structure is :- (DR2,WR2) eg if DR2=2 & WR2=1, EOP=21. See table below.

## ELEMENTS of EOP ARRAY

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



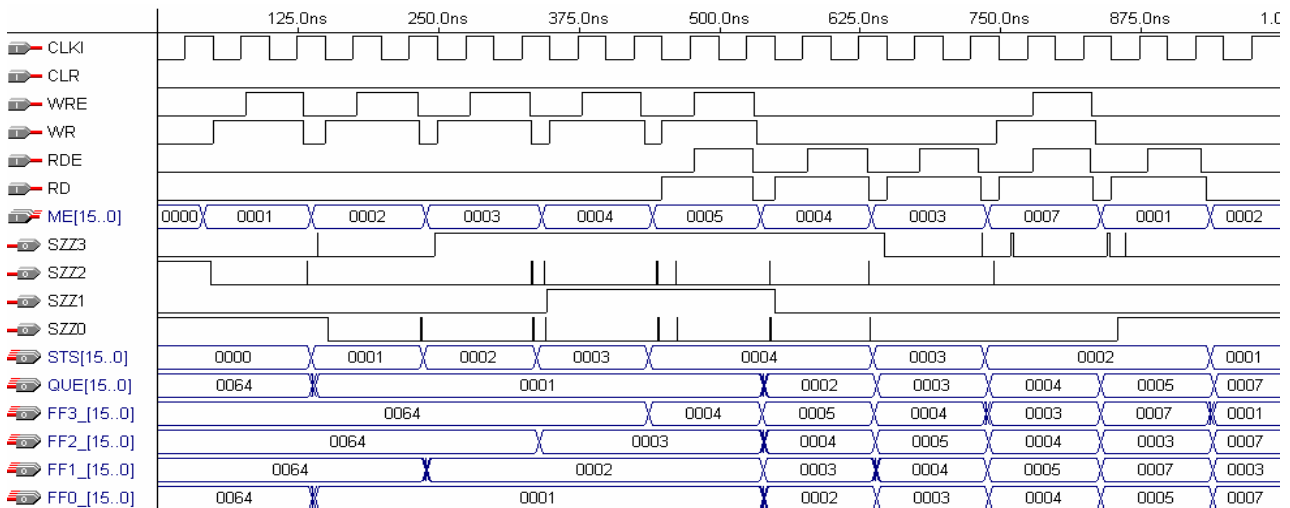
## SAMPLE DESIGN

```
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(
    RD      :IN  NODE;
    WR      :IN  NODE;
    RDE     :IN  NODE;
    WRE     :IN  NODE;
    CLR     :IN  NODE;
    ME      :IN  BUS1D(WID DOWNT0 0);
    CLKI    :IN  NODE;
    SZZ     :BUFFER BUS1D(3 DOWNT0 0);
    QUE     :BUFFER BUS1D(WID DOWNT0 0);
    STS     :BUFFER BUS1D(15 DOWNT0 0)
);
END MYTOP;
ARCHITECTURE MYTOP OF MYTOP IS
BEGIN
    A1: SUBSTITUTE STATEMENT FROM SIMULATION
END MYTOP;
```

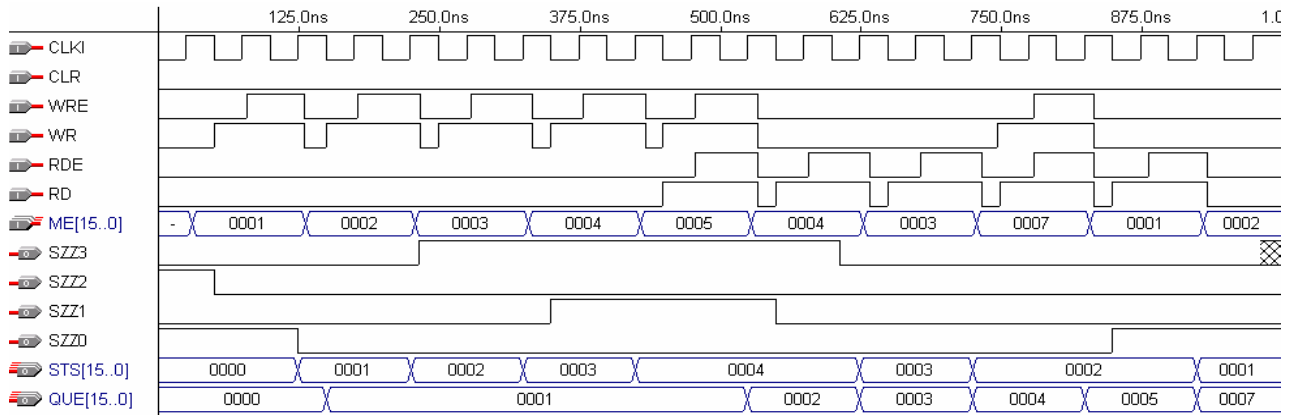
## TIMING DIAGRAMS

A1: F\_IFO GENERIC MAP (OPN=>0,WID=>15,QDI=>6,QDE=>0,IREG=>0,WR2=>0,DR2=>0)  
PORT MAP (RD,WR,RDE,CLR,ME,CLKI,SZZ,OPEN,QUE,STS);





A1: F\_IFO GENERIC MAP (OPN=>1,WID=>15,QDI=>6,QDE=>0,IREG=>0,WR2=>1,DR2=>1)  
PORT MAP (RD,WR,RDE,CLR,ME,CLKI,SZZ,OPEN,QUE,STS);



A1: F\_IFO GENERIC MAP (OPN=>1,WID=>15,QDI=>3,QDE=>1,IREG=>0,WR2=>2,DR2=>2)  
PORT MAP (RD,WR,RDE,CLR,ME,CLKI,SZZ,OPEN,QUE,STS);

