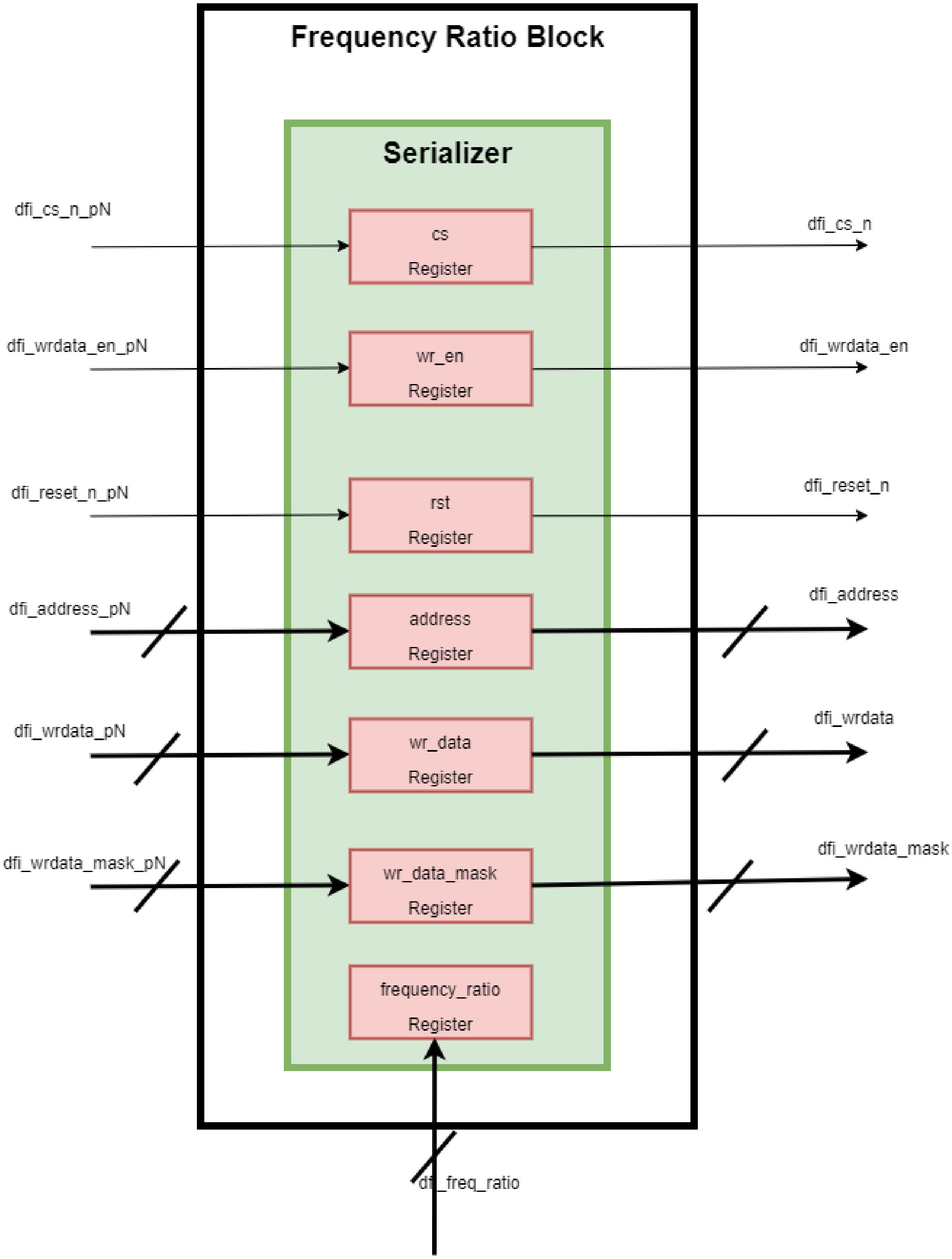
Digital Design Blocks

Sponsored by: Si-Vision

Contents

1.Frequency Ratio Block

1.Block Diagram



2.Block Description

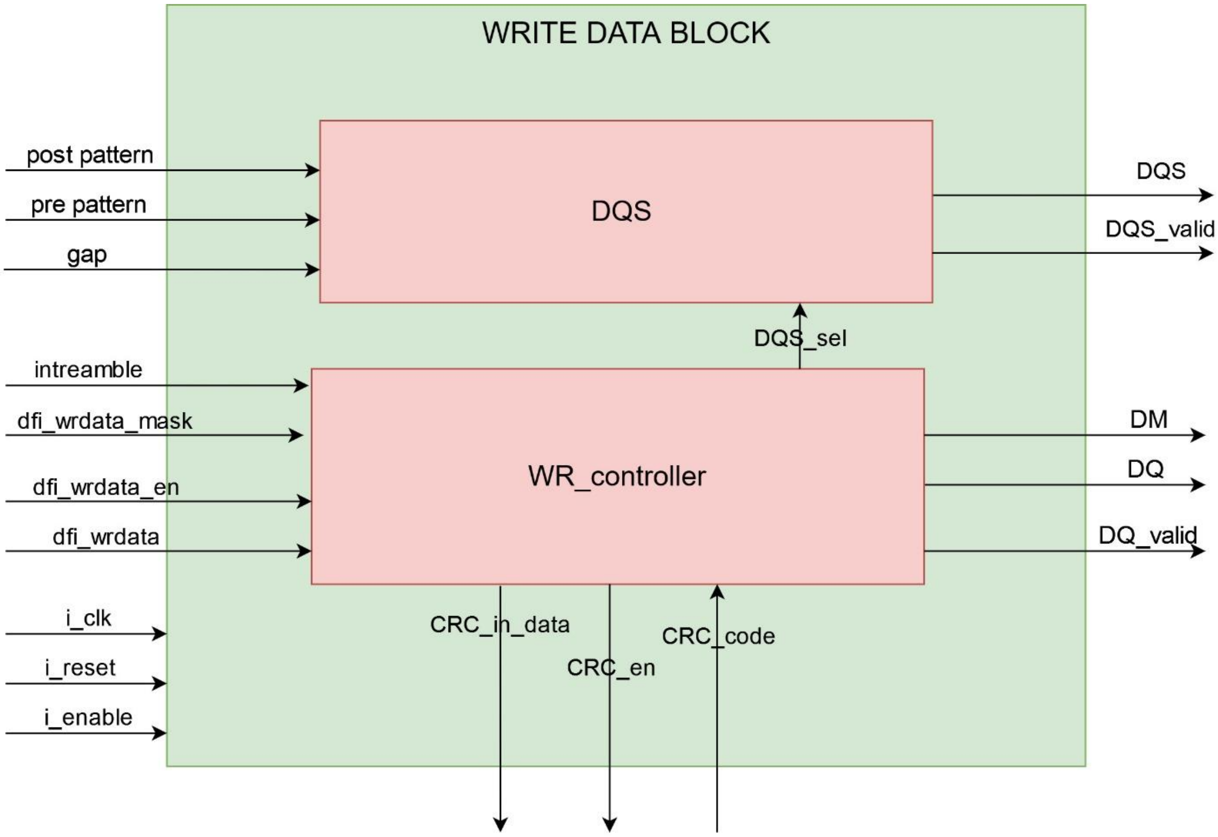
- All signals coming from MC to PHY should pass by this block.
- Frequency Ratio Block converts the signals from MC interface to PHY interface.
- At initialization before any operation, this block should read an input from the register file called dfi_freq_ratio.
- Each signal comes from MC to PHY on maximum 4 different phases, Frequency Ratio Block should out all these signals on one phase only.
- dfi_freq_ratio signal determines the number of phases where each signal come on to this block, for example:
    - dfi_freq_ratio = 000 → ∴ each signal comes on only 1 phase.
    - dfi_freq_ratio = 001 → ∴ each signal comes on only 2 phases.
    - dfi_freq_ratio = 010 → ∴ each signal comes on only 4 phases.
- This block is a serializer block which consists of 7 registers
    - **frequency_ratio Register:**
        - 3 bit register to save the input dfi_freq_ratio in.
        - Dfi_freq_ratio is an input comes to this block at the initialization before operation.
    - **cs Register:**
        - 4 bit register to save the input dfi_cs_n_pN in.
        - Dfi_cs_n is a 1 bit input signal comes on maximum 4 different phases
            - Dfi_cs_0: should be in cs register [0].
            - Dfi_cs_1: should be in cs register [1].
            - Dfi_cs_2: should be in cs register [2].
            - Dfi_cs_3: should be in cs register [3].
        - Dfi_cs is a 1 bit output signal from this block. This output signal should take the value of cs Register[0] and at each cycle we should shift the cs register and the output takes the new value of cs Register [0].
    - **Wr_en Register:**
        - 4 bit register to save the input dfi_wrdata_en_pN in.
        - Dfi_wrdata_en_pN is a 1 bit input signal comes on maximum 4 different phases
            - Dfi_wrdata_en_0: should be in wr_en register [0].
            - Dfi_wrdata_en_1: should be in wr_en register [1].
            - Dfi_wrdata_en_2: should be in wr_en register [2].
            - Dfi_wrdata_en_3: should be in wr_en register [3].
        - Dfi_wrdata_en is a 1 bit output signal from this block. This output signal should take the value of wr_en register [0] and at each cycle we should shift the wr_en register and the output takes the new value of wr_en Register [0].

- **rst Register:**
  - 4 bit register to save the input dfi_reset_n_pN in.
  - dfi_reset_n is a 1 bit input signal comes on maximum 4 different phases
    - dfi_reset_0: should be in rst register [0].
    - dfi_reset_1: should be in rst register [1].
    - dfi_reset_2: should be in rst register [2].
    - dfi_reset_3: should be in rst register [3].
  - Dfi_reset_n is a 1 bit output signal from this block. This output signal should take the value of rst Register [0] and at each cycle we should shift the rst register and the output takes the new value of rst Register [0].
- **address Register:**
  - 56 bit register to save the input dfi_address_pN in.
  - dfi_address is a 14-bit input signal comes on maximum 4 different phases
    - dfi_address_p0: should be in address register [0].
    - dfi_address_p1: should be in address register [1].
    - dfi_address_p2: should be in address register [2].
    - dfi_address_p3: should be in address register [3].
  - Dfi_address is a 14 bit output signal from this block. This output signal should take the value of address Register [13:0] and at each cycle we should shift the address register and the output takes the new value of address Register [13:0].
- **Write data Register:**
  - parameterized register to save the input dfi_address_pN in.
  - dfi_wrdata_pN is a parameterized input signal comes on maximum 4 different phases
    - dfi_wrdata_p0: should be in write data register [parameter-1:0].
    - dfi_wrdata_p1: should be in write data register [parameter-1:0].
    - dfi_wrdata_p2: should be in write data register [parameter-1:0].
    - dfi_wrdata_p3: should be in write data register [parameter-1:0].
  - Dfi_wrdata is a parameterized output signal from this block. This output signal should take the value of write data Register [parameter-1:0] and at each cycle we should shift the address register and the output takes the new value of write data Register [parameter-1:0].

- **Write data mask Register:**
  - parameterized register to save the input dfi_wrdata_mask_pN in.
  - dfi_wrdata_mask_pN is a parameterized input signal comes on maximum 4 different phases
    - dfi_wrdata_mask_p0: should be in write data mask register [parameter-1:0].
    - dfi_wrdata_mask_p1: should be in write data mask register [parameter-1:0].
    - dfi_wrdata_mask_p2: should be in write data mask register [parameter-1:0].
    - dfi_wrdata_mask_p3: should be in write data mask register [parameter-1:0].
  - Dfi_wrdata mask is a parameterized output signal from this block. This output signal should take the value of write data mask Register [parameter-1:0] and at each cycle we should shift the address register and the output takes the new value of write data mask Register [parameter-1:0].

2. WRITE DATA BLOCK

• Block diagram



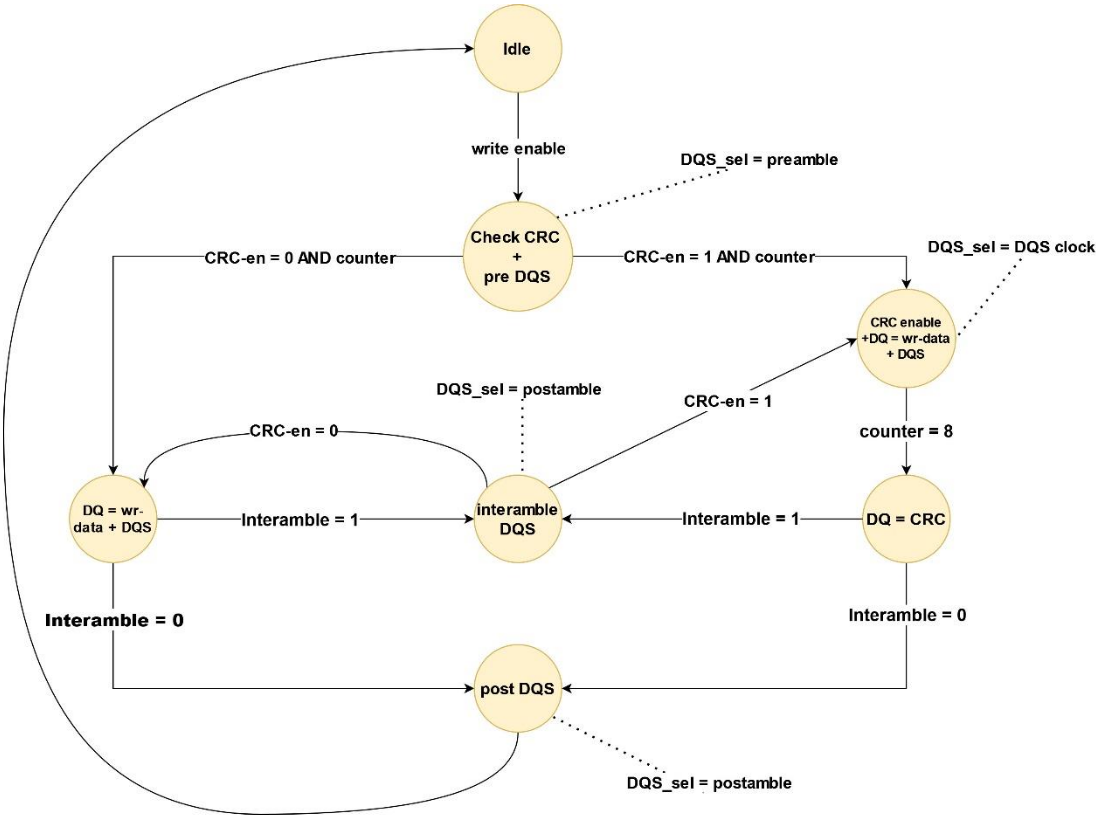
• Block implementation

The block consists of 2 main blocks

1- WR_controller
2- DQS

• WR_controller

- The FSM consists of 7 states.
- The block output will be DQ, DQ_valid and DM.
- It will control DQS block by DQS_sel signal.
- The following figure shows FSM states.

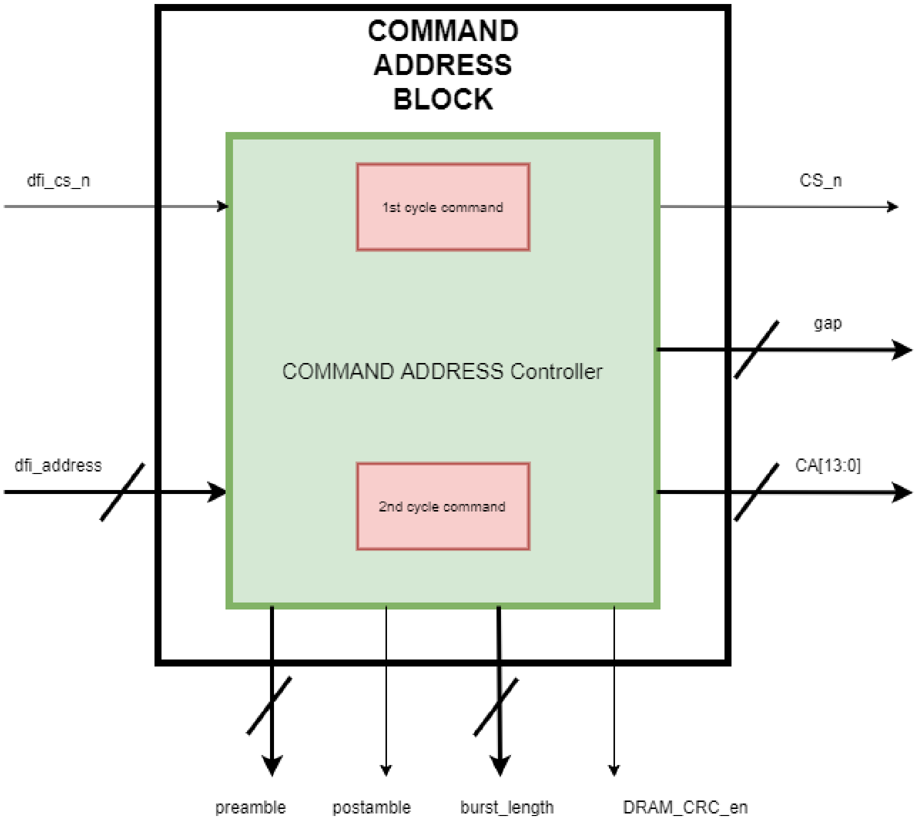


- ## DQS

  - This block stores the different pattern of DQS (pre-amble, post-amble, and inter-amble) and the output will be DQS and DQS_valid according to the DQS_sel signal from WR_controller.

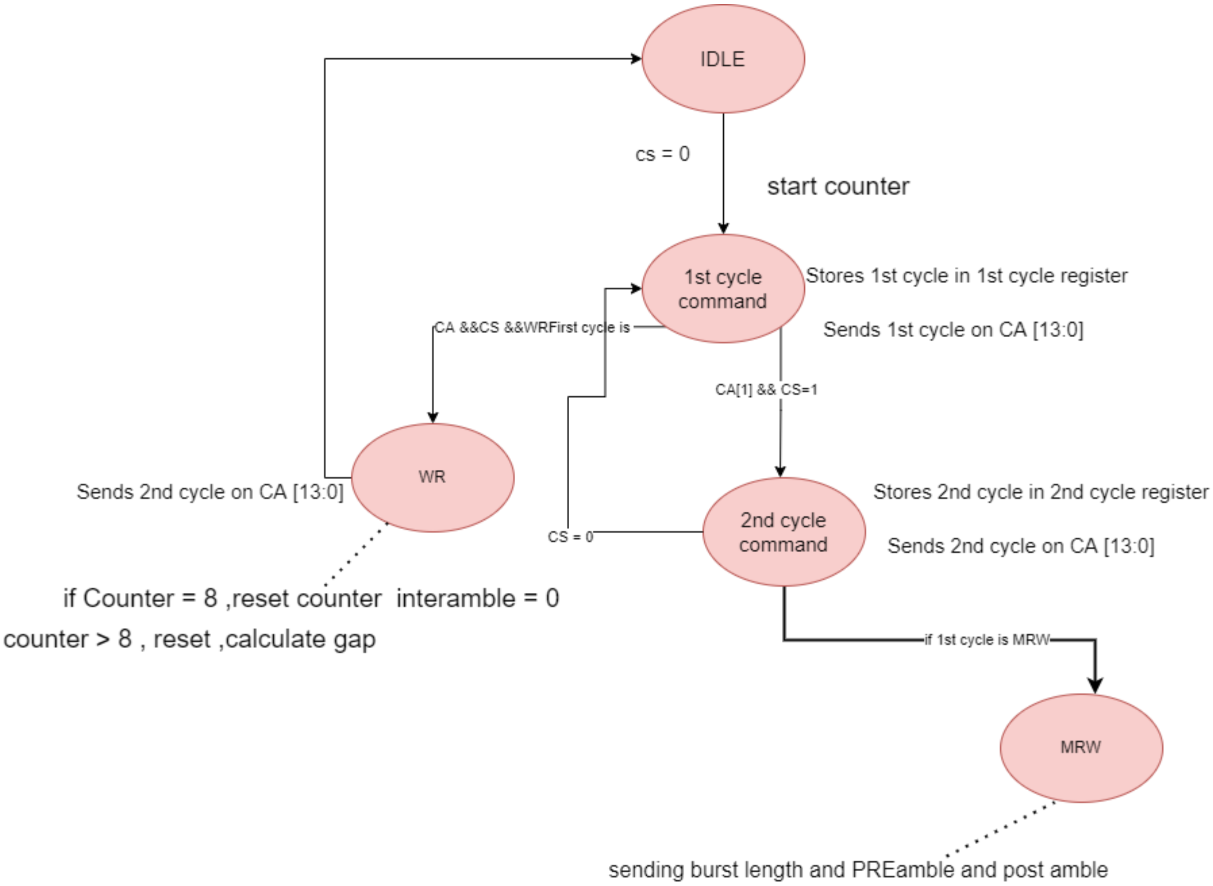
3.COMMAND ADDRESS BLOCK

Block diagram

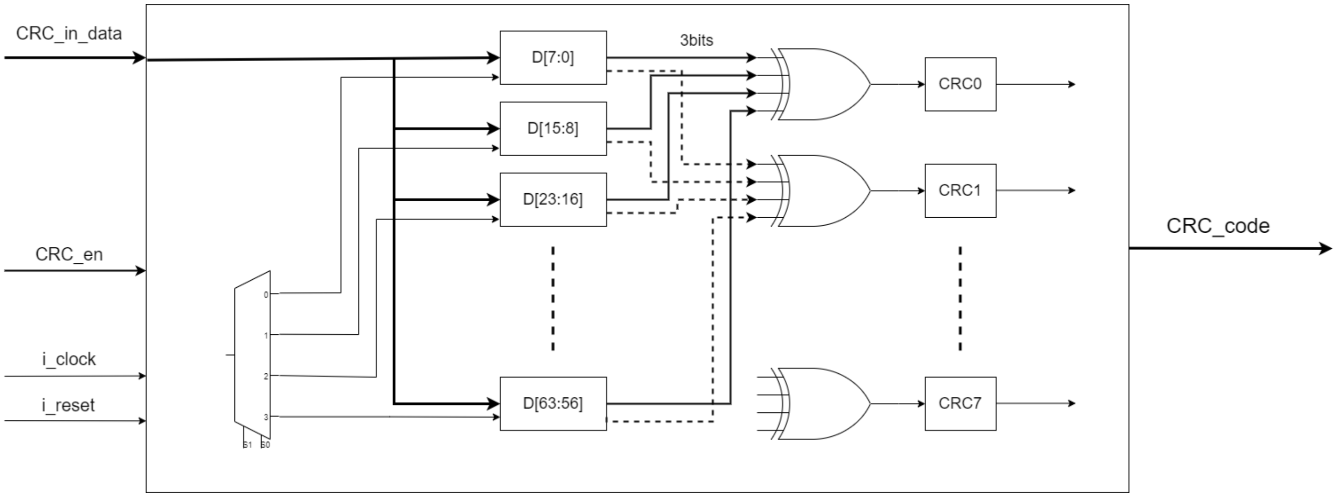


Block implementation

- This block is consist of command address controller.
- The FSM of the controller has 5 different states.
  - Idel
  - 1st cycle: The 1st cycle of the command will be stored and then sent on CA bus.
  - 2nd cycle: The 2nd cycle of the command will be stored and then sent on CA bus.
  - MRW: the burst length, the preamble, the postamble, and the CRC enable will be fetched in this state.
  - WR: the gap will be calculated form the last write command and the current command (if the gap between [BL/2 + 5: BL/2], interamble DQS will be exist).
- The following figure shows FSM states.


IDLE
cs = 0
start counter
1st cycle command
Stores 1st cycle in 1st cycle register
Sends 1st cycle on CA [13:0]
CA &&CS &&WRFirst cycle is
CA[1] && CS=1
WR
Sends 2nd cycle on CA [13:0]
CS = 0
2nd cycle command
Stores 2nd cycle in 2nd cycle register
Sends 2nd cycle on CA [13:0]
if Counter = 8 ,reset counter interamble = 0
counter > 8 , reset ,calculate gap
if 1st cycle is MRW
MRW
sending burst length and PREamble and post amble

4.CRC BLOCK

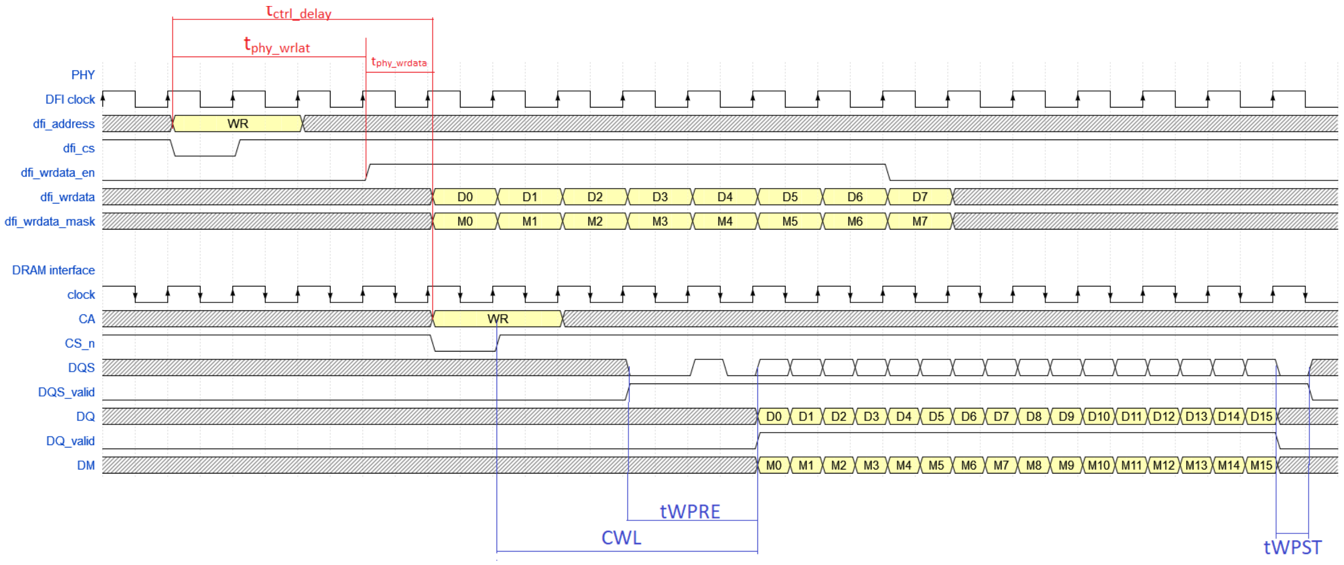


Block implementation

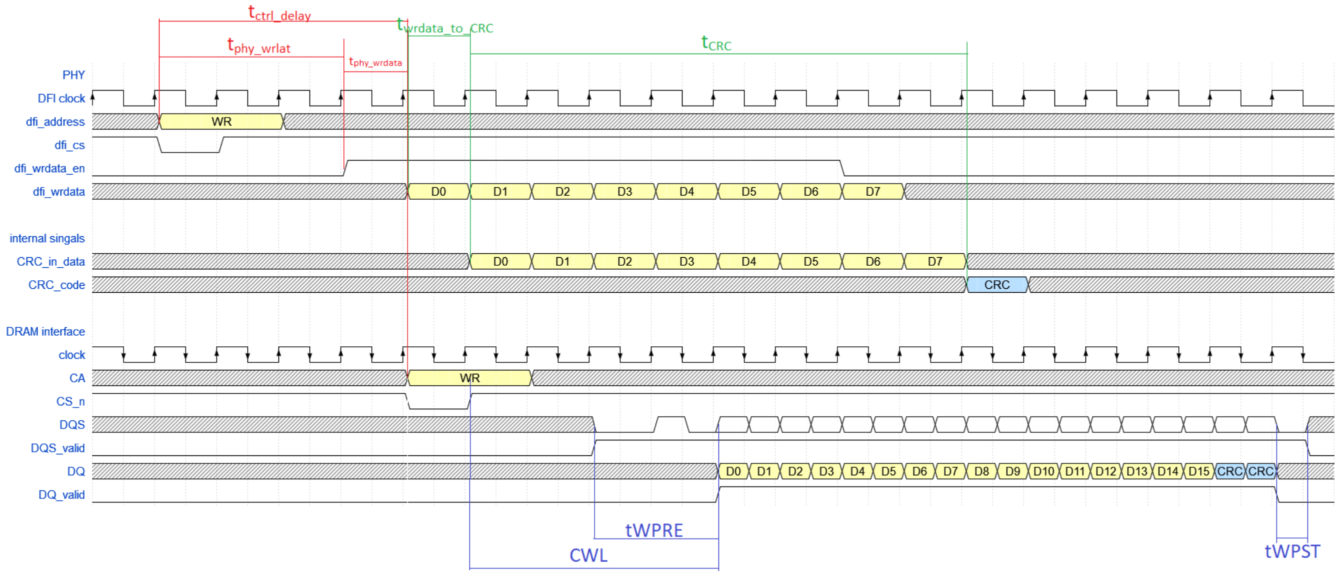
- This block will fetch the data from write data block.
- It will have 8 registers of 8 bits and will be initially hold 1’s.
- For each clock cycle the data will be stored in one register according the selector of the demux.
- After 8 clock cycles the data will be ready to calculate the CRC from it.
- At the 9th clock cycle the CRC code will be ready to be fetched (8bits CRC for 64bits of data).
- This block will be duplicated according to the device type.

5- Timing diagrams.

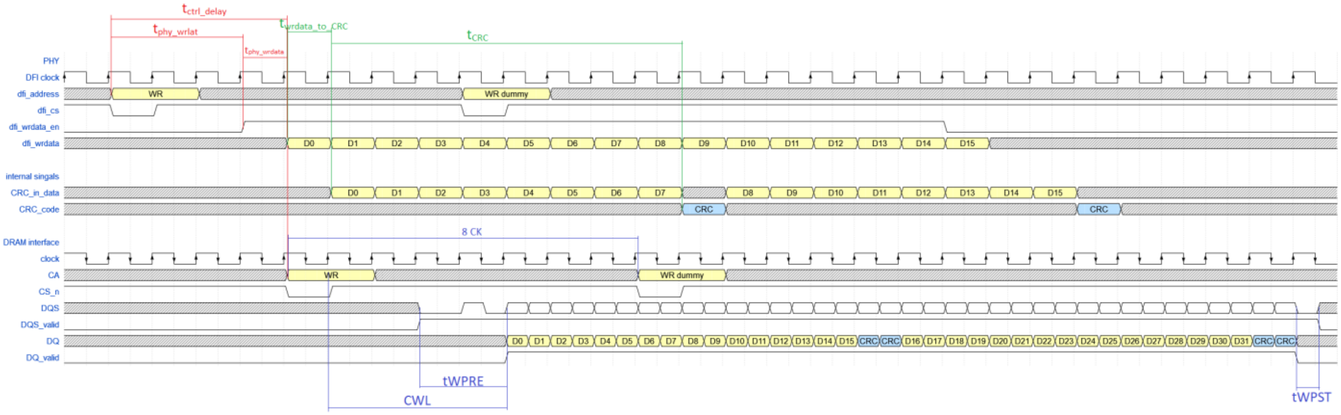
1- Single write command with BL16 with mask.



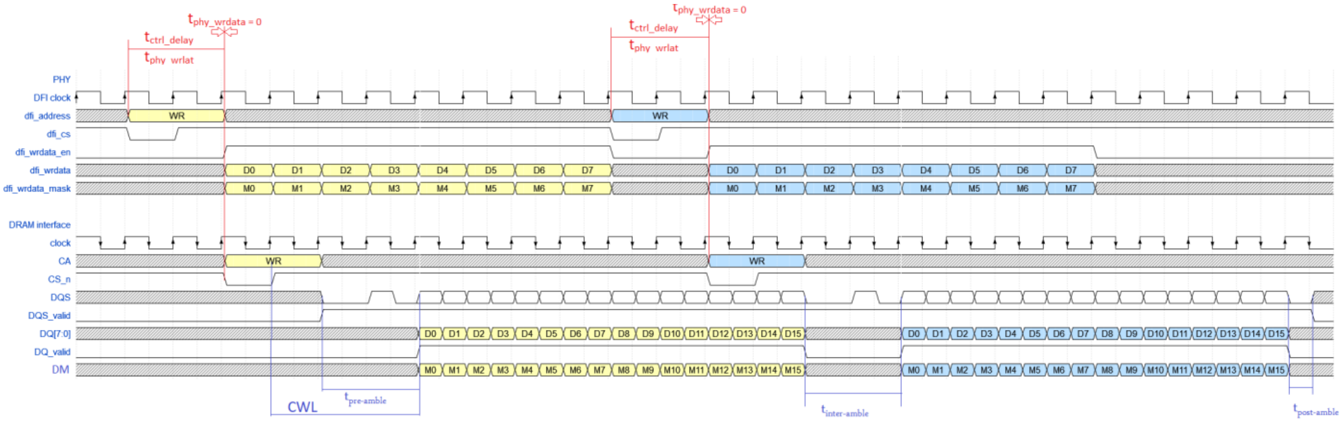
2- Single write command with BL16 with CRC.



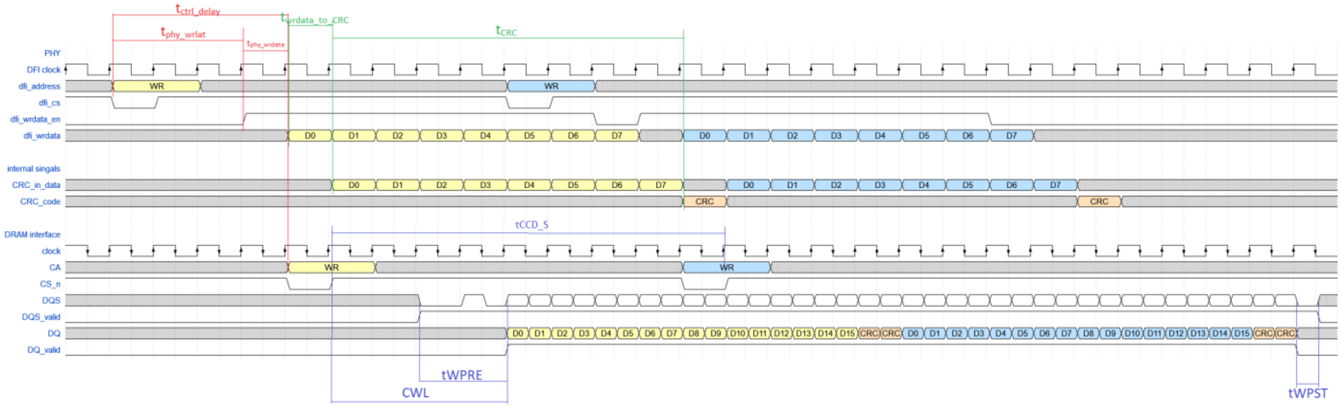
3- Single write command with BL32 with CRC.



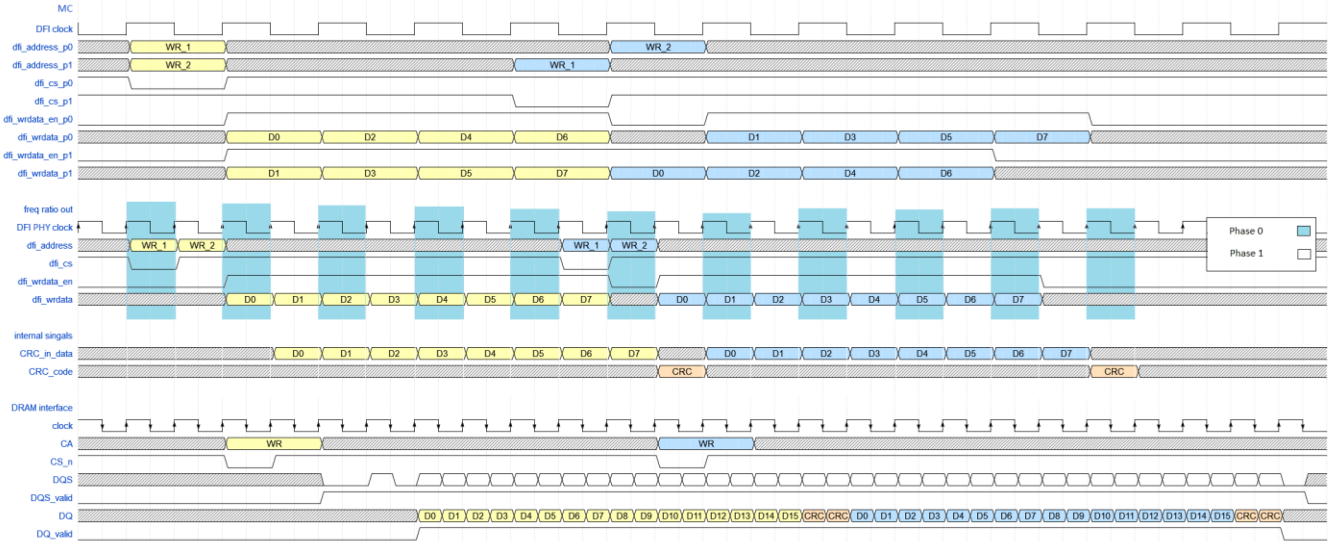
4- Two independent write command with BL16 with mask.



5- Back to back write command with BL16 with CRC.



6- Back to back write with frequency ratio of 2:1 and BL16 with CRC.



MC
DFI clock
dfi_address_p0
WR_1
WR_2
dfi_address_p1
WR_2
WR_1
dfi_cs_p0
dfi_cs_p1
dfi_wrdata_en_p0
dfi_wrdata_p0
D0
D2
D4
D6
D1
D3
D5
D7
dfi_wrdata_en_p1
dfi_wrdata_p1
D1
D3
D5
D7
D0
D2
D4
D6
freq ratio out
DFI PHY clock
dfi_address
WR_1
WR_2
WR_1
WR_2
dfi_cs
dfi_wrdata_en
dfi_wrdata
D0
D1
D2
D3
D4
D5
D6
D7
D0
D1
D2
D3
D4
D5
D6
D7
Phase 0
Phase 1
internal singals
CRC_in_data
D0
D1
D2
D3
D4
D5
D6
D7
D0
D1
D2
D3
D4
D5
D6
D7
CRC_code
CRC
CRC
DRAM interface
clock
CA
WR
WR
CS_n
DQS
DQS_valid
DQ
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D14
D15
CRC
CRC
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D14
D15
CRC
CRC
DQ_valid