**-PRIORITY ENCODER BLOCK –**

Written by- Vemuri Kousalya Gayatri

**BLOCK DIAGRAM USING INTERFACE:**

Diagram

Description automatically generated with low confidence

ValidReqID, ReqID

Clock, Reset, Dreq, Dack, Hlda

**BREIF DESCRIPTION ABOUT THE PORTS:**

**Input ports:**

* **Hlda:** Is hold acknowledgement which is given by the processor in response of a HRQ, sent by the DMA.
* **RotatingPriority:** Bit [4] from the Command register [7:0], which indicates if it’s fixed or rotating priority to be performed.

0 - Fixed Priority, 1 - Rotating Priority.

* **SenseDreq:** Bit [6] from the Command register [7:0], indicates when we need to sense ipDreq (Polarity).

0 - Sense active high, 1 - Sense active low.

* **SenseDack:** Bit [7] from the Command register [7:0], indicates the polarity of Dack.

0 - Sense active low, 1 - Sense active high.

* **DMA\_Disable:** Bit [2] from Command register [7:0], which tells if DMA controller is enabled or disabled.

0 - Enable, 1 – Disable.

* **Mask:** It gives information about which all channels have been masked, and those channels requests are ignored.

[4:0] -> bit [0] = 1 that means channel 0 is masked, so on for channel 1,2 and 3.

* **Dreq:** It indicates which all channels are requesting for a DMA transfer.

[4:0] -> bit [0] = 1 that means channel 0 is requesting, so on for channel 1,2 and 3.

**Output ports:**

* **Dack:** Indicates which channel has won the arbitration and received the grant. Only one channel can receive a grant at a time. Therefore, only one bit can be high for Dack.

[4:0] -> bit [1] = 1 that means channel 1 has got the grant/acknowledgement, so on for channel 0,2 and 3.

* **ReqID:** These two bits will tell that which channel has won the Arbitration and will receive the grant.

00 = channel 0, 01 = channel 1, 10 = channel 2, 11 = channel 3.

* **ValidReqID:** This signal indicates if ReqID is valid or not at a given time.

0 - Not valid, 1 – Valid.

* **PendingReq:** These 4 bits tells which all channels’ requests are still pending. It will also consider the one whose transfer is in process. It’s nothing but outputting the Dreq we got.

**SystemVerilog Constructs used:**

Typedef, enum, always\_ff, always\_comb, unique case, functions, immediate assertions, parameters, and concurrent assertions in sanity test bench.

**BLOCK DIAGRAM FOR FSM: -**

ValidReqID

ReqID [1:0]

leastPriority [1:0]

PendingReq [3:0]

tempDack [3:0]

Clock

Reset

DMA\_Disable

Hlda

RotatingPriority

ValidDreq [3:0]

Finite State Machine

**FINITE STATE MACHINE: -**

! ValidDreq / ValidReqID = 0, tempDack = 0, PendingReq = ValidDreq

Hlda / ValidReqID = 1, tempDack [ReqID] = 1, PendingReq = ValidDreq

Hlda / tempDack [ReqID] = 1, ValidReqID = 1, PendingReq = ValidDreq

Hlda & RotatingPriority / leastPriority = ReqID

~Hlda / ValidReqID = 1, tempDack = 0, PendingReq = ValidDreq

(~DMA\_Disable & ~Hlda & (|ValidDreq)) / ValidReqID =1, ReqID, tempDack = 0, PendingReq = ValidDreq

~Hlda / ValidReqID = 0, tempDack = 0

ACKNOWLEDGE

Reset / leastPriority = 3, ReqID = 0, PendingReq = 0

WAITING

ARBITER

**Bullet points describing the functionality of PriorityEncoder module: -**

* Dreq is an asynchronous signal, which can come at any time, but the signal is sampled at the positive edge of the Clock.
* ValidDreq is an internal wire whose value is calculated and used internally.
* When Dreq is received it needs to be sensed according to the polarity programmed by the processor. So Dreq is first XOR with the SenseDreq signal.
* After sensing the signal, it is checked if any channels are masked or not. To check masking the sensed signal gets AND with the compliment of Mask (4 bits).
* Arbitrationdecisionis made when we have a ValidDreq (DMA transfer request from the I/O), and there is no other transfer in process (i.e., Hlda is low), also when DMA\_Disable being low (therefore DMA is enabled).
* Arbitration decision is given to the ReqID register (2 bits), which can be interpreted with the help of ValidReqID signal.
* When Hlda is asserted then tempDack is assigned according to ReqID.
* To send the correct Dack, we need to do an XNOR between tempDack and SenseDack.

TEST PLAN -

**Assertions used in design module (immediate assertions): -**

1. At positive edge of Clock, all the control signals should be known (Reset, RotatingPriority, Hlda, SenseDreq, SenseDack, DMA\_Disable).
2. At positive edge of Clock, ValidDreq, Mask should be known.

**Concurrent assertions in test bench: -**

1. Dack should be correct W.R.T ReqID, throughout the time when ValidReqID and Hlda is asserted.
2. When ValidReqID is asserted and RoatatingPriority is low, ReqID should be correct based on Dreq.
3. We should never see a Dack for masked channel. Consequent should be non-zero, if zero then masked channel has a Dack.