

```

module plx_arb (
    CLK,
    FRAME_L,
    IRDY_L,
    REQ_L,
    GNT_GATED_L,
    RST_L,
    TRDY_L
);
input  CLK;
input  FRAME_L;
input  IRDY_L;
input  [8:0] REQ_L;
output [8:0] GNT_GATED_L;
input  RST_L;
input  TRDY_L;

reg    [8:0] GNT_L;

//implement a rotating priority gnt scheme!
//    -index of lowest priority is last gnt request.
//    -priority decreases as index decrements
//    -gnt changes upon frame active
reg    [3:0] lastgnt;
wire   [8:0] req;
wire   [3:0] reqindx; //highest req indx
reg    frame_d1, frame_d2;
wire   pciidle;
wire   gntchnng;
wire   dgntall; //1 if idle and req goes away.
reg    dgntall_d1;
reg    [8:0] gnt_d1;
assign gntchnng = (GNT_L == 9'h1fff) | (&REQ_L)
                 | (~frame_d1 & frame_d2 & (~REQ_L & GNT_L))
                 | dgntall_d1
;
assign dgntall = (~REQ_L & GNT_L) & (GNT_L != 9'h1fff) & FRAME_L &
frame_d1;
assign req = ~REQ_L;

reg    trdy_d1, trdy_d2, trdy_d3, trdy_d4;

wire   gate;
assign gate = TRDY_L & (trdy_d1 | trdy_d2 | trdy_d3 | trdy_d4);

always @(posedge CLK or negedge RST_L)
if (~RST_L) begin
    trdy_d1 <= #1 1'b0;
    trdy_d2 <= #1 1'b0;
    trdy_d3 <= #1 1'b0;
    trdy_d4 <= #1 1'b0;
end
else begin
    trdy_d1 <= #1 ~TRDY_L;
    trdy_d2 <= #1 trdy_d1;
    trdy_d3 <= #1 trdy_d2;
    trdy_d4 <= #1 trdy_d3;
end

assign GNT_GATED_L[0] = GNT_L[0] | gate;

```

```

assign GNT_GATED_L[1] = GNT_L[1] | gate;
assign GNT_GATED_L[2] = GNT_L[2] | gate;
assign GNT_GATED_L[3] = GNT_L[3] | gate;
assign GNT_GATED_L[4] = GNT_L[4] | gate;
assign GNT_GATED_L[5] = GNT_L[5] | gate;
assign GNT_GATED_L[6] = GNT_L[6] | gate;
assign GNT_GATED_L[7] = GNT_L[7] | gate;
assign GNT_GATED_L[8] = GNT_L[8] | gate;

```

```

assign reqindx =

```

```

        (lastgnt == 8) ? ((req[0]) ? 0 :
                        (req[1]) ? 1 :
                        (req[2]) ? 2 :
                        (req[3]) ? 3 :
                        (req[4]) ? 4 :
                        (req[5]) ? 5 :
                        (req[6]) ? 6 :
                        (req[7]) ? 7 : 8
                        ) :
        (lastgnt == 7) ? ((req[8]) ? 8 :
                        (req[0]) ? 0 :
                        (req[1]) ? 1 :
                        (req[2]) ? 2 :
                        (req[3]) ? 3 :
                        (req[4]) ? 4 :
                        (req[5]) ? 5 :
                        (req[6]) ? 6 : 7
                        ) :
        (lastgnt == 6) ? ((req[7]) ? 7 :
                        (req[8]) ? 8 :
                        (req[0]) ? 0 :
                        (req[1]) ? 1 :
                        (req[2]) ? 2 :
                        (req[3]) ? 3 :
                        (req[4]) ? 4 :
                        (req[5]) ? 5 : 6
                        ) :
        (lastgnt == 5) ? ((req[6]) ? 6 :
                        (req[7]) ? 7 :
                        (req[8]) ? 8 :
                        (req[0]) ? 0 :
                        (req[1]) ? 1 :
                        (req[2]) ? 2 :
                        (req[3]) ? 3 :
                        (req[4]) ? 4 : 5
                        ) :
        (lastgnt == 4) ? ((req[5]) ? 5 :
                        (req[6]) ? 6 :
                        (req[7]) ? 7 :
                        (req[8]) ? 8 :
                        (req[0]) ? 0 :
                        (req[1]) ? 1 :
                        (req[2]) ? 2 :
                        (req[3]) ? 3 : 4
                        ) :
        (lastgnt == 3) ? ((req[4]) ? 4 :
                        (req[5]) ? 5 :
                        (req[6]) ? 6 :
                        (req[7]) ? 7 :
                        (req[8]) ? 8 :

```

```

        (req[0]) ? 0 :
        (req[1]) ? 1 :
        (req[2]) ? 2 : 3
    ) :
    (lastgnt == 2) ? ((req[3]) ? 3 :
        (req[4]) ? 4 :
        (req[5]) ? 5 :
        (req[6]) ? 6 :
        (req[7]) ? 7 :
        (req[8]) ? 8 :
        (req[0]) ? 0 :
        (req[1]) ? 1 : 2
    ) :
    (lastgnt == 1) ? ((req[2]) ? 2 :
        (req[3]) ? 3 :
        (req[4]) ? 4 :
        (req[5]) ? 5 :
        (req[6]) ? 6 :
        (req[7]) ? 7 :
        (req[8]) ? 8 :
        (req[0]) ? 0 : 1
    ) :
    ( (req[1]) ? 1 :
        (req[2]) ? 2 :
        (req[3]) ? 3 :
        (req[4]) ? 4 :
        (req[5]) ? 5 :
        (req[6]) ? 6 :
        (req[7]) ? 7 :
        (req[8]) ? 8 : 0
    );

always @(posedge CLK or negedge RST_L) begin
    if (RST_L == 0) begin
        GNT_L <= 9'h1fe;
        lastgnt <= 'h0;
    end
    else begin
        frame_d1 <= #1 ~FRAME_L;
        frame_d2 <= #1 frame_d1;
        dgntall_d1 <= dgntall;
        gnt_d1 <= ~GNT_L;
        GNT_L[0] <= #2 dgntall
gntchnng)          | (~( req[0] & (reqindx == 0) &
~gntchnng)          | (~GNT_L[0] & req[0] &
                    ) &
                    ~(&REQ_L));
        GNT_L[1] <= #2 dgntall
gntchnng)          | (~( req[1] & (reqindx == 1) &
~gntchnng)          | (~GNT_L[1] & req[1] &
                    );
        GNT_L[2] <= #2 dgntall
gntchnng)          | (~( req[2] & (reqindx == 2) &
~gntchnng)          | (~GNT_L[2] & req[2] &
                    );

```

```

);
GNT_L[3] <= #2 dgntall | ~( (req[3] & (reqindx == 3) &
gntchnng) | (~GNT_L[3] & req[3] &
~gntchnng) );
GNT_L[4] <= #2 dgntall | ~( (req[4] & (reqindx == 4) &
gntchnng) | (~GNT_L[4] & req[4] &
~gntchnng) );
GNT_L[5] <= #2 dgntall | ~( (req[5] & (reqindx == 5) &
gntchnng) | (~GNT_L[5] & req[5] &
~gntchnng) );
GNT_L[6] <= #2 dgntall | ~( (req[6] & (reqindx == 6) &
gntchnng) | (~GNT_L[6] & req[6] &
~gntchnng) );
GNT_L[7] <= #2 dgntall | ~( (req[7] & (reqindx == 7) &
gntchnng) | (~GNT_L[7] & req[7] &
~gntchnng) );
GNT_L[8] <= #2 dgntall | ~( (req[8] & (reqindx == 8) &
gntchnng) | (~GNT_L[8] & req[8] &
~gntchnng) );
lastgnt <= #2 (~FRAME_L & ~frame_d1) ?
({gnt_d1[8],
|gnt_d1[7:4],
gnt_d1[7] | gnt_d1[6] | gnt_d1[3] |
gnt_d1[2],
gnt_d1[7] | gnt_d1[5] | gnt_d1[3] |
})
: lastgnt;
end
end
endmodule

```