SLOW DIVISION ALGORITHM CODE

```
module slow_division (
input wire signed [15:0]
dividend,
input wire signed [15:0]
divisor,
output wire signed
[15:0] quotient,
output wire signed
[15:0] remainder
);

wire signed [15:0] M;
wire signed [15:0] A;
wire Q_0;
wire signed [15:0] N;

assign N = dividend;
assign M = divisor;
assign A = 0;
assign Q_0 = 0;

always @(*) begin
if (A[15] == 0 || A[15:0]
>= M) begin
A = A - M;
Q_0 = 1;
end else begin
A = A + M;
Q_0 = 0;
end
A = A << 1;
Q_0 = Q_0 << 1;
end

assign quotient =
{Q_0, A[15:1]};
assign remainder =
A[15:0];

endmodule
```

FAST DIVISION ALGORITHM CODE

```
module fast_division
(
input wire signed
[15:0] dividend,
input wire signed
[15:0] divisor,
output wire signed
[15:0] quotient,
output wire signed
[15:0] remainder
);

wire signed [15:0] N;
wire signed [15:0] D;
wire signed [15:0] Q;
wire signed [15:0] A;

assign N = dividend;
assign D = divisor;
assign Q = 0;
assign A = N;

always @(posedge
clk) begin
if (A[15] == 0 || A[15:0]
>= D) begin
A = A - D;
Q = Q + 1;
end else begin
A = A + D;
Q = Q - 1;
end
end

assign quotient = Q;
assign remainder =
A;

endmodule
```