FPGA Project
Booth Multiplier

G Anusha - EE16BTECH11011
K Samhitha - EE16BTECH11019

IIT Hyderabad

April 26, 2019

Abstract

Booth's algorithm is a multiplication algorithm that multiplies two signed binary numbers in 2's compliment notation.As in all multiplication schemes, booth algorithm requires examination of the multiplier bits and shifting of the partial product.

1 Procedure

1. Let M is the multiplicand.
2. Q be the Multiplier.
3. Consider a 1-bit register Q_{-1} and initialize it to 0.
4. Consider a register A and initialize it to 0.

1.1 Conditions

1. If Q_0 and Q_{-1} are same i.e., 00 or 11 then,perform arithmetic right shift by one bit.
2. If $Q_0Q_{-1} = 10$ then, perform A ← A-M and then perform arithmetic right shift.
3. If $Q_0Q_{-1} = 01$ then, perform A ← A+M and then perform arithmetic right shift.

1.2 Example

Consider 2 numbers 6 and 2.
Take 6 as Multiplicand(M) and 2 as Multiplier(Q).
Write 6 and 2 in binary form as 6 = 0110 and 2 = 0010 (Q_3 Q_2 Q_1 Q_0)
Booth's algorithm calculates the product in n steps where n is the number of bits used to represent the number.

INITIALISE	A	B	Q_{-1}	OPERATIONS
	0 0 0 0 ↓↘↘↘↘	0 0 1 0 ↘↘↘↘	0	
Step 1.	0 0 0 0	0 0 0 1 ↓	0 ↓	Arithmetic right shift
Step 2.	1 0 1 0 ↓↘↘↘↘ 1 1 0 1	0 0 0 1 ↘↘↘↘ 0 0 0 0	0 1	A←A-M Then shift
Step 3.	0 0 1 1 ↓↘↘↘↘ 0 0 0 1 ↓↘↘↘↘	0 0 0 0 ↘↘↘↘ 1 0 0 0 ↘↘↘↘	1 0	A←A+M Then shift
Step 4.	0 0 0 0	1 1 0 0 In binary, 12 = 1100 Hence 3*2 = 12	0	Arithmetic right shift

2 Software Required

1. Verilog Language
2. iverilog software

3 Hardware Required

1. Icoboard
2. 2 Arduino Boards
3. JHD 162A LCD screen
4. 4x3 Keypad
5. wires

4 Verilog Codes

4.1 Main code

```
module multiplier(prod, busy, mc, mp, clk, start);
output [15:0] prod;
output busy;
input [7:0] mc, mp;
input clk, start;
reg [7:0] A, Q, M;
reg Q_1;
reg [3:0] count;
wire [7:0] sum, difference;
always @(posedge clk)
begin
```

```
if (start) begin
A ⇐ 8'b0;
M ⇐ mc;
Q ⇐ mp;
Q_1 ⇐ 1'b0;
count ⇐ 4'b0;
end
else begin
case ({Q[0], Q_1})
2'b0_1 : {A, Q, Q_1} ⇐ {sum[7], sum, Q};
2'b1_0 : {A, Q, Q_1} ⇐ {difference[7], difference, Q};
default: {A, Q, Q_1} ⇐ {A[7], A, Q};
endcase
count ⇐ count + 1'b1;
end
end
alu adder (sum, A, M, 1'b0);
alu subtracter (difference, A, ∼ M, 1'b1);
assign prod = {A, Q};
assign busy = (count ¡ 8);
endmodule
module alu(out, a, b, cin);
output [7:0] out;
input [7:0] a;
input [7:0] b;
input cin;
assign out = a + b + cin;
endmodule
```

4.2 Testbench code

```
module testbench;
reg clk, start;
reg [7:0] a, b;
wire [15:0] ab;
wire busy;
multiplier multiplier1(ab, busy, a, b, clk, start);
initial begin
clk = 0;
$display("first example: a = 3 b = 17");
a = 3; b = 17; start = 1; #50 start = 0;
#80 $display("first example done");
$display("second example: a = 7 b = 7");
a = 7; b = 7; start = 1; #50 start = 0;
#80 $display("second example done");
$finish;
end
always #5 clk = !clk;
always @(posedge clk) $strobe("ab: $stime);
endmodule
```

4.3 pcf file

```
set_io clk R9
set_io X[0] N9
set_io X[1] P9
set_io X[2] M8
set_io X[3] N7
set_io X[4] L9
set_io X[5] G5
set_io X[6] L7
set_io X[7] N6
set_io prod[0] R10
set_io prod[1] T11
set_io prod[2] T14
set_io prod[3] T15
set_io prod[4] T9
set_io prod[5] T10
set_io prod[6] T13
set_io prod[7] T14
```

4.4 Running the code in iverilog

To install iverilog, run the following command in terminal.
sudo apt install iverilog
To run the code in iverilog software, we need main code and testbench code.
Run the following commands in terminal.
iverilog -o sim maincode.v testbench.v
./sim

5 References

- https://www.geeksforgeeks.org/computer-organization-booths-algorithm/
- https://en.wikipedia.org/wiki/Booth%27s_multiplication
- https://www.ijrter.com/published_special_issues/31-03-2017/implementation-of-booth-multiplier-and-modified-booth-multiplier.pdf