Multi-cycle Division

1, Description

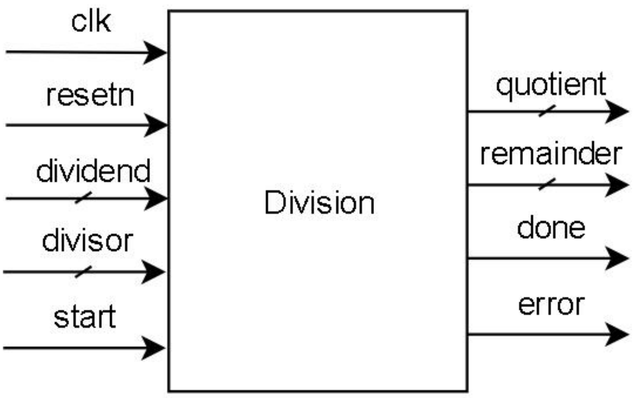
· Division of two N-bit-number in multi-cycle.

· Multi-cycle Division receives inputs including dividend, divisor, start control signals, and returns quotient, remainder along with done, error status signals.

2, Parameter, input and output signals

Parameter	Value	Description
N	Default: 4	Data width in bit of dividend and divisor

Signal	Width	I/O	Description
clk	1	Input	System clock signal
resetn	1	Input	System asynchronous reset, active low
dividend	N	Input	Dividend number
divisor	N	Input	Divisor number
start	1	Input	Start request, set to start divide
quotient	N	Output	Quotient number Init value: 0
remainder	N	Output	Remainder number Init value: 0
done	1	Output	Set to indicate that computation is complete Init value: 0
error	1	Output	Set to indicate that the divisor is not valid (divisor = 0) Init value: 0



Division block diagram

3, Design

Local signal	Width	Reset value	Description
a	N	0	Temporary dividend register
b	N	1	Temporary divisor register
count	$clog(N) + 1	0	Count division period

- Division of two N-bit number takes place over N cycles.

- The method of dividing two numbers is as follows:

Cycle *Remainder--Dividend* *Divisor* *Quotient*

Start 0000--1101 0001 0000

- At start Remainder < Divisor -> shift left Remainder-Dividend, Quotient

1 0001--1010 0001 0000

+ Remainder >= Divisor -> Remainder - Division, Quotient + 1

0000--1010 0001 0001

Then shift left 3 register

2 0001--0100 0001 0010

+ Remainder >= Divisor -> Remainder - Division, Quotient + 1

0000--0100 0001 0011

Then shift left

3 0000--1000 0001 0110

Remainder < Divisor -> Only shift left

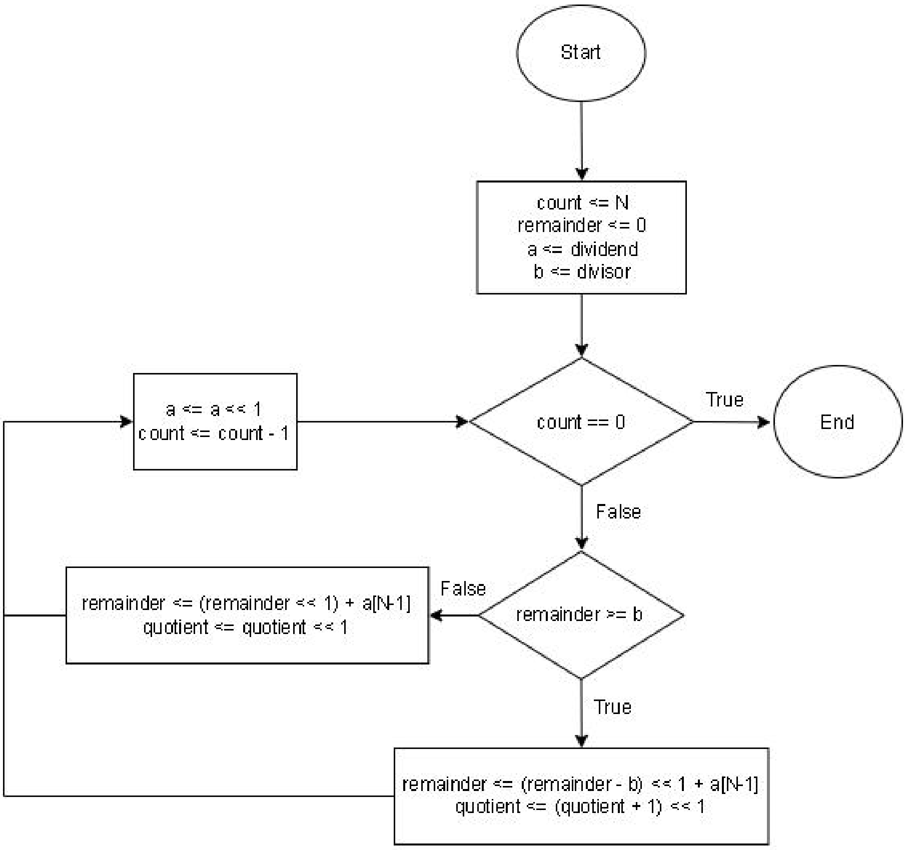
4 0001--0000 0001 1100

+ Remainder >= Divisor -> Remainder - Division, Quotient + 1

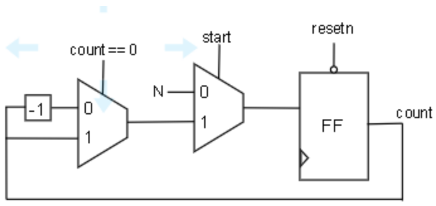
0000--0000 0001 1101

So Quotient = 1101, Remainder = 0000

- Algorithm flowchart of division:

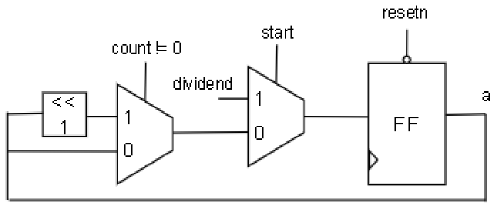

Start
count <= N
remainder <= 0
a <= dividend
b <= divisor
count == 0
True
End
False
a <= a << 1
count <= count - 1
remainder >= b
False
remainder <= (remainder << 1) + a[N-1]
quotient <= quotient << 1
True
remainder <= (remainder - b) << 1 + a[N-1]
quotient <= (quotient + 1) << 1

- Count value is decreased by 1 each cycle from start till the count value is zero.



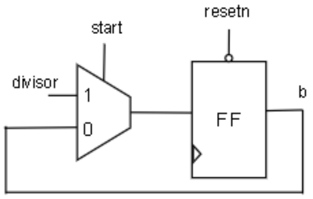
Logic update count

- The dividend is assigned to the temporary dividend register (a) when the start bit is set. Each cycle, the temporary dividend register is shifted left by 1 until the computation is complete.



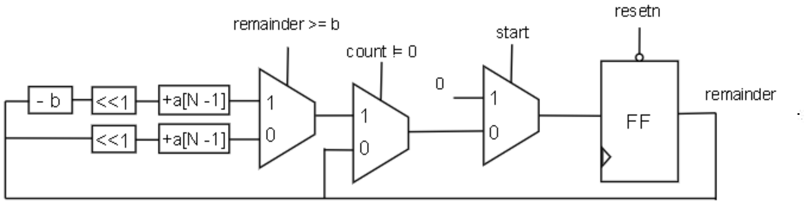
Logic update temporary dividend

- The divisor is assigned to the temporary register (b) when the start bit is set.



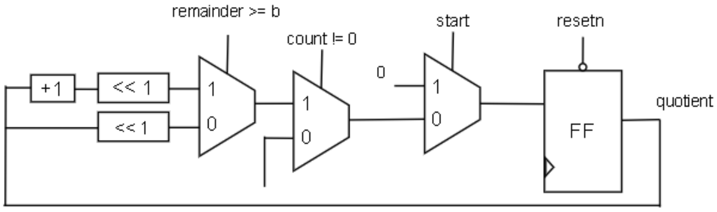
Logic update temporary divisor

- When the computation starts, the remainder value is 0. Each cycle, if the remainder is not less than the divisor, the remainder is decreased by the divisor. Then, the divisor is shifted left by 1 and the MSB of the dividend is added.



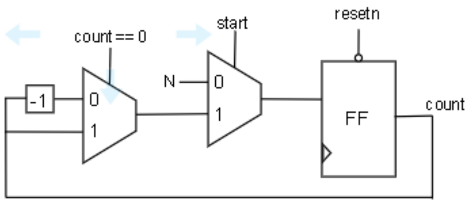
Logic update remainder

- When the computation starts, the quotient value is 0. Each cycle, if the remainder is not less than the divisor, the quotient is incremented by 1 and shifted left by 1; otherwise, it is only shifted left by 1.



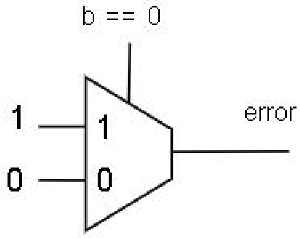
Logic update quotient

- The done flag is set when the countdown reaches 1 or when the error flag is set



Logic update done flag

- The error flag is set when the temporary divisor register is assigned a value of 0.



Logic update error flag