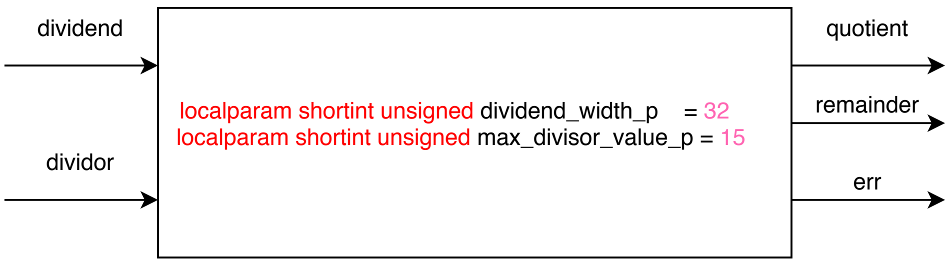
Multiplication-based Arithmetic Divider

RHGL

May 4, 2020

1 Applicable Usage



This is document describes RTL implementation of a very fast unsigned fixed-point number divider with low cost. The division operation is realized by multiplication operator that is easily synthesizable in most of tools. Considering scenarios and characters below for adoptions:

1. limited number of divisor:
   In the RTL implementation, each divisor is handled separately in a case statement, associated with an unique hard-coded *magic number*. So, if there are hundreds of possible divisors in the application, this is not a suitable solution. In the provided RTL example, multiplication_based_divider.sv and multiplication_based_divider.vhdl, the dividend is 32 bits width and possible divisor are [1,2,3, ..., 15].
2. synthesizable
3. high speed
4. low cost

Section 2 explains how to convert a division operation to a multiplication operation.
Section 3 shows the usage of the SV module to calculate *magic number*.

2 Mathematical Proof

We denote **dividend** as N (numerator), **divisor** as D (denominator), **quotient** as Q and **reminder** as R. The fundamental equation to be solved is

$$\frac{N-R}{D} = Q$$

, where N/D can be re-written as $\frac{N}{2^k}\frac{2^k}{D}$, N is a N_W bits unsigned number and $Q = \lfloor \frac{N}{D} \rfloor = \left\lfloor \frac{N}{2^k}\frac{2^k}{D} \right\rfloor$. Assuming there are n divisors, $D_0, D_1, ..., D_{n-1}$, for each D_i, because $2^{k_i}/D_i$ can be an non-integer number, we pre-compute the closest integer number of $\frac{2^{k_i}}{D_i}$ and denote it as $M_i = \left\lceil \frac{2^{k_i}}{D_i} \right\rceil$. To compensate the difference caused by *ceiling* operation, $\lceil\rceil$, we introduce e into equation:

$$M_i = \left\lceil \frac{2^{k_i}}{D_i} \right\rceil = \frac{2^{k_i}+e}{D_i}$$

, where $0 \le e \le D_i - 1$. Because M_i is an integer number, by replacing $\frac{2^k}{D}$ with M_i, we can calculate Q':

$$Q' = \left\lfloor \frac{NM_i}{2^{k_i}} \right\rfloor = \left\lfloor \frac{N}{2^{k_i}} \frac{2^{k_i}+e}{D_i} \right\rfloor = \left\lfloor \frac{N}{D_i} + \frac{Ne}{D_i 2^{k_i}} \right\rfloor$$

We need to find the smallest k_i so that Q is equal to Q'. The constraints are

a) $\frac{Ne}{D_i 2^{k_i}} < 1$

b) The sum of the fractional part of $\frac{N}{D_i}$ and the fractional part of $\frac{Ne}{D_i 2^{k_i}}$ is less than 1

For a), the maximum value of N is $2^{N\_W} - 1$ and $e \le D_i$, so to satisfy a), we have

$$k_i \ge N\_W \tag{constraint\_a}$$

For b), the maximum value of the fractional part of $\frac{N}{D_i}$ is $\frac{D_i - 1}{D_i}$. So b) is equivalent to

$$\begin{aligned} &\frac{D_i - 1}{D_i} + \frac{Ne}{D_i 2^{k_i}} < 1 \\ \Rightarrow &\frac{Ne}{D_i 2^{k_i}} < \frac{1}{D_i}, \text{for all } N \\ \Rightarrow &2^{k_i} > eN_{max} \end{aligned} \tag{constraint\_b}$$

, where $0 \le e \le D_i - 1$ and is used to round up $\frac{2^{k_i}}{D_i}$ to the closest integer number. When **constraint_a** and **constraint_b** are both met, the integer part of Q' is the same as the integer part of Q, and the fractional part will be wiped out by the $\lfloor \rfloor$ operation, so we have $Q == Q'$.

2.1 An Example

Considering an example, where $N\_W == 32$ and $D \in [1, 2, 3, ..., 15]$, for the D_i that is not a power of 2, the corresponding M_i, K_i are

D_i	3	5	6	7	9	10
M_i inHex	AAAA_AAAB	CCCC_CCCD	AAAA_AAAB	1_2492_4925	38E3_8E39	CCCC_CCCD
M_i width	32	32	32	33	30	32
K_i	33	34	34	35	33	35

D_i	11	12	13	14	15
M_i inHex	ba2e_8ba3	AAAA_AAAB	4ec4_ec4f	1_2492_4925	8888_8889
M_i width	32	32	31	33	32
K_i	35	35	34	36	35

Table 1: An example: $N\_W == 32$ and $D \in [1, 2, 3, ..., 15]$

2.2 Calculation Shortcuts

In the circumstances below, it is not necessary to calculate M_i for D_i.

1. When $D_i == 2^k$, division can be simplified by **logic shift** followed by $\lfloor \rfloor$ operation.
2. When $D_i == D_a D_b$ $(i > a; i > b)$,

$$Q_i = \left\lfloor \frac{N}{D_i} \right\rfloor = \left\lfloor \frac{NM_i}{2^{k_i}} \right\rfloor \tag{1}$$

$$Q'_i = \left\lfloor \frac{\left\lfloor \frac{NM_a}{2^{k_a}} \right\rfloor M_b}{2^{k_b}} \right\rfloor \tag{2}$$

If Q_i (in (1)) is equal to Q'_i (in (2)), then Q_i can be calculated step-by-step without pre-computing M_i. (2) can be rewritten as

$$\begin{aligned}
&N = Q_a * D_a + R_a,\ Q_a = \left\lfloor \frac{NM_a}{2^{k_a}} \right\rfloor,\ 0 \le R_a < D_a \\
&Q_a = Q_b * D_b + R_b,\ Q' = Q_b = \left\lfloor \frac{Q_a M_b}{2^{k_b}} \right\rfloor,\ 0 \le R_b < D_b \\
&\Rightarrow N = Q_b * D_b * D_a + R_b * D_a + R_a \\
&\Rightarrow \frac{N}{D_a D_b} = Q_b + \frac{R_b}{D_b} + \frac{R_a}{D_a D_b}
\end{aligned} \tag{3}$$

Obviously, $\frac{R_a}{D_a D_b} < \frac{1}{D_b}$, so $\frac{R_b}{D_b} + \frac{R_a}{D_a D_b} < \frac{R_b}{D_b} + \frac{1}{D_b} \le 1$. Therefore, in (3), $\lfloor \frac{N}{D_a D_b} \rfloor == \lfloor Q_b + \frac{R_b}{D_b} + \frac{R_a}{D_a D_b} \rfloor$ and Q_b is the final quotient which can be calculated in two steps.

3. Furthermore, when $D_i == D_a D_b * ... * D_z$, we can calculate Q_z step-by-step and it is easy to prove the finial Q_z (Q') is equal to Q, so that we don't bother to calculate M_i for each D_i.

In Table 1, it is not necessary to calculate M_i for $6, 9, 10, 12, 14, 15$.

3 SV module to calculate M_i

Use find_magic_number.sv to calculate magic numbers for the given dividend width and divisor. There are 3 *localparam* in this file:

1. **dividend_width**
2. **k_upbou**: The SVmodule iteratively tries k is equal to an integer number from **dividend_width** to **k_upbound** until the magic number is found. *k_upbound* should be greater than dividend at least. If there no *magic number* is found, increase *k_upbound* to a larger number.
3. **divisor**

Note: need to set **UVMHOME** for your simulator, because **'uvm_info** is used.

4 Verification

To be continued...