Synthesis of fixed point code

Vittorio Zaccaria

<2015-07-26 Sun>

Why we need fixed point

- Express fractional numbers, using a fixed number of bits `http://www.digitalsignallabs.com/fp.pdf`
- Useful for systems without floating-point hardware (DSPs, FPGAs, and expensive custom ASIC, microcontrollers).
- Also useful when you need speed, as integer operations are faster than floating point ones.

Integer representation

- The formula for calculating the integer representation x in a $Q_{m.n}$ format of a float number f is (ah):
  $x = round(f * 2^n)$
- To convert it back the following formula is used:
  $f = x * 2^{-n}$

Range and resolution

- For a given $Q_{m.n}$ we have:
  - its range is $[-(2^m), 2^m - 2^{-n}]$
  - its resolution is 2^{-n}

The problem

- operations can produce results that have more bits than the operands (**e.g.** multiplications)

> Example: $Q_{1.7} * Q_{1.7} \to Q_{2.14}$

- Solution: temporarily use a bigger register but then truncate or round back to $Q_{1.7}$

Ada approach

- Ada uses a delta type; with this you can specify what is the minimum difference between two float numbers to be considered different.
- The compiler however will choose $2^{-12} = \frac{1}{4096}$, not $\frac{1}{3600}$.

Concepts

Two's complement

- Assume x is N bits. We define the complement x_2 such that:

$$x_2 + x = 2^N$$

- 2^N is represented by zeroes over N bits and a 1 outside the N bits. So, if we focus on the lower N bit, we'll see that x_2 behaves like a real **inverse** of x in `+'.