ID.Fix
Optimisation de la précision des calculs dans les systèmes embarqués : méthodes et outils

Avril 2009

Olivier Sentieys, Daniel Ménard
IRISA/INRIA
Université de Rennes 1
ENSSAT Lannion
sentieys@irisa.fr

Number Representation

- Floating-Point

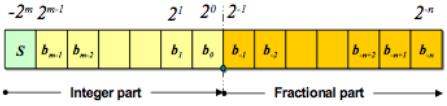
  $x = (-1)^s \times m \times 2^{e-127}$

  s: sign, *m*: mantissa, *e*: exponent

- Pros and cons
  - Simple to use
  - High dynamic range

  but
  - Accuracy problems
  - Complex operators

- Fixed-Point Numbers



$$x = (-1)^s(-2)^m + \sum_{i=-n}^{m-1} b_i.2^i$$

s: sign, *m*: magnitude, *n*: fractional

  - Efficient operators
  - Stable, predictable

but complex to use, overflow, precision loss

Fixed-Point Arithmetic

- Dynamic range $[-2^m, 2^m[$
- Precision: 2^{-n}
- Arithmetic rules
  - Multiplication, addition `(int)(((INT64)a * (INT64)b) >> N)`
  - Scaling operations are explicit in software code
- Precision (noise)

```
  1.640625
* 2.3125
= 3.7939453125
```

Overflow

```
  1.640625
+ 2.5125
= 4.153125
```


Fixed-Point Arithmetic

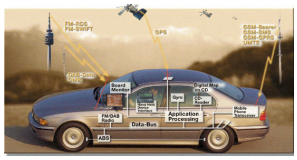
- Speed, power, cost
- Efficient computation
  - Faster execution time
    - Fix/float: ratio of 9
  - Energy efficient (us/mW)
    - Fix/float: ratio of 5
  - Cost efficient (us/$)
    - Fix/float: ratio of 5

  (DSP C6x from TI)
- Stable calculations across platforms
  - Predictable by simulations
- Accuracy is limited but often sufficient for many applications

Fixed-Point Arithmetic is Mainstream in Embedded Systems

5

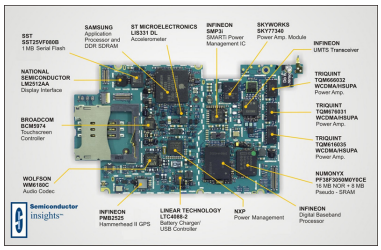
- Many application domains
  - Telecommunications, wireless
    - e.g. wireless receiver is sampled with low precision
    - e.g. error-correcting codes still work for very-low precision
  - Signal and image processing, audio, video
  - Control (automotive)
  - Graphics (OpenGL ES 1.x)



Fixed-Point Arithmetic is Mainstream in Embedded Systems

6

- Most of embedded platforms use only fixed-point arithmetic
  - Embedded processors, microcontroller, digital-signal processors, application platforms
  - FPGA, system-on-chip
  - FPU is big and hot!



Fixed-Point Conversion

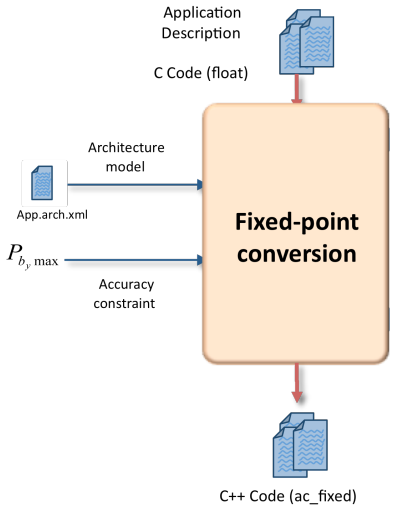
- Determine the number of bits for each data
  - Range estimation: integer part word-length
  - Precision estimation: fractional part word-length
- Insert scaling instructions after every operations
  - Arithmetic rules
- Manual conversion
  - Long, tedious and error-prone
    - 25% to 50% of the total design time (according to Mathworks)
- Strong need of tools to reduce the time-to-market

Fixed-Point Conversion

- Existing tools for helping the designer in the fixed-point conversion process
  - Fixed-point Matlab Toolbox (Mathworks)
  - AccelDSP (Xilinx)
  - Catalytic (Mentor Graphics)
- Performance evaluation using bit-true simulation
  - Fixed-point simulation is very long
  - Data word-length optimisation time is prohibitive
  - Limited design space exploration

ID.Fix
A Framework for Fixed-Point Conversion

9



- Source-to-source code transformation
  - C with float to C/VHDL using fxed-point types
  - Compiler framework: GECOS (Eclipse)
- Contracts
  - ANR ROMA
  - Nano 2012 (STMicro.)
  - INRIA: engineer

ID.Fix
A Framework for Fixed-Point Conversion

10

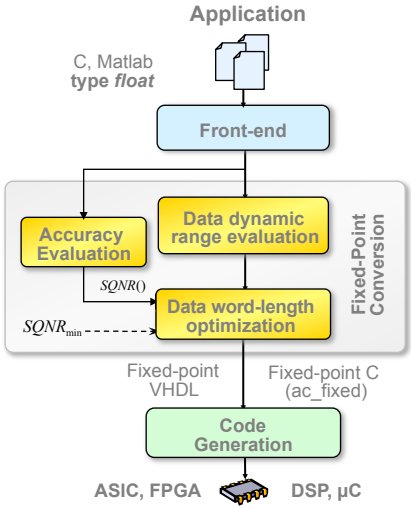
Highlights from ID.Fix tool

- Analytical estimation of accuracy
  - Output noise power P_b is a mathematical expression which is formulated for every data wordlengths
  - P_b can be linked to application performance
    - with system-level floating-point simulations
    - e.g. BER (bit error rate), PSNR (peak signal-to-noise ratio)
- System-level estimation and design space exploration are possible
- Optimisation time of the process is significantly reduced

ID.Fix
A Framework for Fixed-Point Conversion

- Automatic conversion from floating-point to fixed-point
- Wordlength optimisation
  - Cost minimisation under accuracy constraint

$$\min_j \left[T_{exec}\left(WL_j\right), Cost\left(WL_j\right)\right]$$
$$SQNR(WL_j) \geq SQNR_{min}$$



Conclusion

- Fixed-point arithmetic in embedded systems
- State-of-the-art tools are limited by long simulation time
- ID.Fix is a framework for automatic fixed-point conversion and design-space exploration
- Current work
  - Hierarchical approach for system-level estimation
  - Extension of source code support in the compiler