Implementation of LPC vocoder and Audio Decoder using ART Designer

Mike Choi(choi@icsl.ucla.edu)
Dongku Kang(dkang@purdue.edu)
Functional Overview of LPC Encoder
Code Development Procedure

1. import <fxp.h>
2. ART Library
3. Matlab
4. C-floating point
5. C-fixed point

Input: FEMALE.PCM (8kHz)
Output: FEMALE_OUT.PCM
Objective: Reduced Computing

- Rewritten code based on Matlab and DSP assembly code for ADSP-2100
- Pre-emphasis and De-emphasis filter suggested by ADI code used.
  \[ H(z) = 1 - 0.9375z^{-1} \]
- Non-overlapping algorithm used.
  Frame size = 180 samples
Fixed Point Simulation using C++

- Fixed Point Simulation and C++ implementation at once!
- Direct Fixed Point Estimation
- ART library enables fixed point simulation in C++
- Environment: Visual-C and GCC
Floating point simulation

- Female.pcm(8kHz mono PCM)

- Algorithm based on ADSP assembly code
- Possible error from assembly code conversion to C-code
VHDL Generation using ART Designer

- **Raw Result Summary**
  - Init 3pot – 3 cycles
  - Run 402 pot – F cycles
  - Registers 48 fields – 1226 bits
  - Mux 48 ports – 1122 muxes
  - Controller 536 words
  - Loop over 10 potential and peak loaded equal to 30000
  - Heavy ALU load by Divider
Optimization

- First optimized source code for loop
  -> No heavy loaded processing unit for loop
- Further optimization of source code
Code Optimization Example

- **ANSI-C**

```c
void a_correlate(fixed* F, fixed* coeff) ormalized autocorr
{
    int i,j; fixed corr=0;
    for(i=0 ; i < FRAME ; i++)
    {
        coeff[i]=0;
        for(j=0 ; j < FRAME ; j++)
        {
            coeff[i] = coeff[i] + F[j]*F[((j+i)%180)];
        }
    }
    for(i=0 ; i < FRAME ; i++)
    {
        coeff[i]/=coeff[0];
    }
}
```

- **Optimized**

```c
void a_correlate(const INT S1[180], INT R[180])
//Generates normalized autocorr
{
    loop1:for(int i=0 ; i < FRAME ; i++)
    {
        R[i]=0;
        INT m;
        loop2:for(int j=0 ; j < FRAME-i ; j++)
        {
            m=S1[j]*S1[j+i];
            R[i]+=m;
        }
    }
}
```

Heavy ACU load loop

Reduced!
Final Result Summary

- Init 3pot – 3 cycles
- Run 344 pot – F cycles
- Registers 46 fields – 821 bits
- Mux 50 ports – 1049 muxes
- Controller 99 bits x 371 words
- Report generation failed due to unpredicted termination
Conclusion

- **C++** fixed point simulation using ART library
- VHDL generation using ART Designer with optimized architecture
- Power and area estimated by Synopsys
Part II MPEG layer-2

Motivation
Objective

- DSP core generation optimized for MPEG Audio
- Fixed Point effect in MPEG audio decoding
MPEG Audio simplified

- Reading organization
- Reading sample
- IDCT
Code modification

- Original code based on MPEG decoder for linux
- File based I/O -> Bit stream based I/O
  (actual decoding was successful on workstation)
- Pointer operation -> array based operation
  (ART Designer does not support pointer!)
1\textsuperscript{st} Challenge

- Generating VHDL using ART Designer

Compile, Architecture Generation \textbf{OK!}

Mapping, Scheduling, RT gen.
2nd Challenge

- Fixed point effect analysis using ART library
3rd Challenge

- Quantized coefficient effect analysis
- Internal operation by floating point
- Fixed point coefficients were generated by ART library (Filter, Scalefactor, IDCT coefficients)
Conclusion

- Different coding style necessary to generate dedicated hardware
- EDA tool is not so smart to easily convert c-code to HDL
- Possible hardware scale seems not so large