THE DEVELOPMENT OF A FIRST PASS VERIFICATION MODULE FOR A SAW FILTER DESIGN AUTOMATION SYSTEM

M. J. McCOLLISTER and S. M. RICHIE, University of Central Florida,
Department of Electrical Engineering, Orlando, Florida 32816

ABSTRACT

This paper will present the continuing developments of a non-interactive SAW filter design automation system. An automation system has been developed for the design and analysis of SAW bidirectional transducers and filters as previously presented [1]. All dominant SAW acoustic and electrical effects have been modeled for non-reflecting transducers in a modular architecture. Synthesis and analysis tools form the core of the design automation system around which an automation shell is used to form the basis of a "SAW Compiler" which is capable of non-interactive SAW filter design. The SAW filter design rules are implemented in a declarative language which provides logical decision control of the CAD system. The design automation system controls the execution and iteratively evaluates options, corrects errors, and decides on an optimal design choice within the set of design rules.

This work was funded in part by the Florida High Technology and Industry Council.

1.0 Introduction

A design automation system for SAW filters, called SAWCOM, has been under development. It is designed to run on an IBM-PC or compatible computer with 640K RAM installed and Intel 80x87 numeric coprocessor. The system took as its input frequency and group delay specification, material availability, and operating conditions. From this information it would determine, via a prolog inference engine, what design techniques should be used. After that, it would call a FORTRAN program which would contain the finite impulse response (FIR) synthesis tools, bidirectional transducer analysis tools, and device response calculation tools. This FORTRAN program, called SAWCAD-PC, is a stand alone program previously developed [2]. The code architecture diagram for this system is presented in *Figure 1*.

As it turned out, the design automation system was lacking some common sense. It would go ahead and try to design non-realistic filters or try alternatives that could have been ruled out earlier in the process. Because of this, a realism check was included into the system as well as an enhanced Prolog solution search. The majority of the code was also rewritten to take advantage of the object oriented programming available in C++.

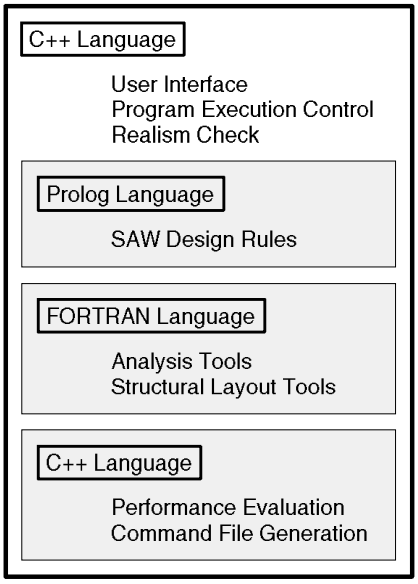

Figure 1, System Code Architecture Diagram

Figure 2 illustrates SAWCOM's system implementation. The shaded blocks indicate the areas being concentrated on in this paper. Therefore, at this point, SAWCOM will only create a file containing the possible solutions which would be presented to SAWCAD-PC.

From the flowchart, the "specs interface" block, the "realism check" and "end processor" contain more detail. The "spec interface" will:

- [Perform I/O screen to the user.]
- Specify material capabilities.
- Obtain available packages.
- Obtain Frequency, phase and delay specifications.
- Obtain Insertion loss specifications.
- [Obtain I/O impedances.]
- Determine if there are external components.
- [Specify if optimum or first solutions desired.]

- [Obtain parameter priority list (only if optimum solution is selected).]

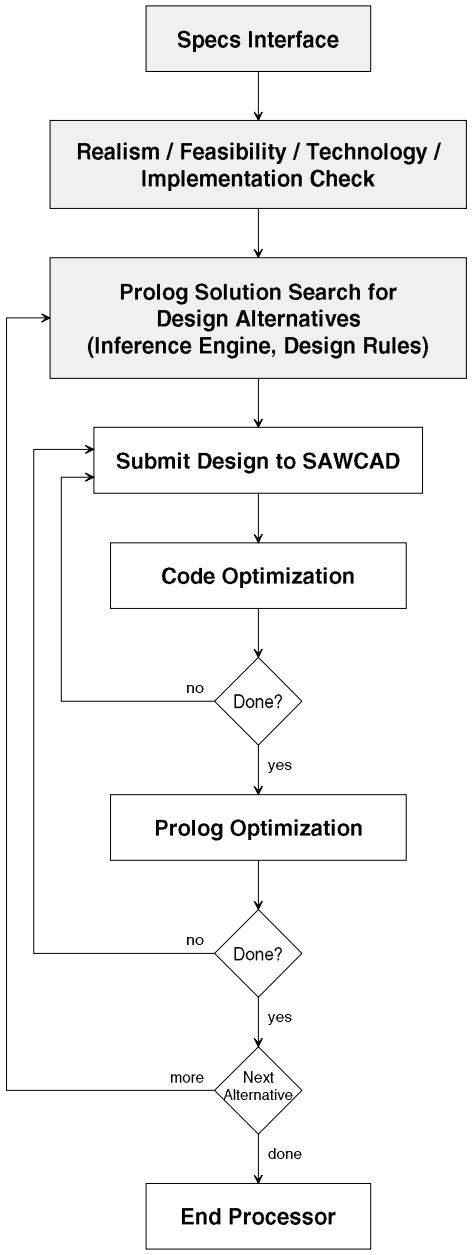

Figure 2, System Implementation of SAWCOM

At this point, there is a simple user interface in which the device specifications are kept in a file which the user edits using a simple text editor. The "realism check" will:

- Perform a check on data to make sure all inputs make sense.
- Perform a technology check of device specifications versus SAWCOM's capabilities.
- Check for vagueness or under-specified or under-constrained requests.
- [Compare with previous simulations (learning mode).]

And the "end processor" will:

- [Sort solutions.]
- [Rate solutions.]
- [Perform any necessary file organization.]
- [Update history.]
- [Report on solutions obtained.]

Items delimited with brackets (i.e. "[", "]") have not yet been implemented.

The Prolog code will determine a number of initial alternatives (or solutions) that may result in a solution(s). The flow of the program is such that one alternative might be submitted to SAWCAD-PC many times until the specifications have been met or it has been determined that the given alternative is not possible. This process will continue until all possible alternatives have been checked.

2.0 Realism Check

The realism check is a simple module that checks for non-sensical inputs. It will verify that the user has at least one substrate selected, the frequencies are realistic, and other values fall within predetermined ranges. When the user specifies a value outside the predetermined constraints, then SAWCOM will indicate that an error has occurred and what the user needs to do about it. This feature had been lacking in previous versions of SAWCOM and its absence presented some difficulty during SAW filter design. SAWCOM would waste time trying to design filters that did not make sense.

3.0 Design Rules

The prolog solution search for design alternatives, as shown in *Figure 2*, contains first order equations to check for possible design alternatives. The equations were restricted to be first order so that alternatives could be quickly eliminated without eliminating any that might be valid. At this point SAWCOM, via. the Prolog code, will generate different alternatives for the same user input. Each alternative will contain four parameters; sampling rate, substrate, technique, and structure. Each of the parameters can take on a number of possible values depending on the parameter. *Table 1* gives a breakdown of each parameter and the values that can be obtained. For example, one alternative might have the sampling rate as 4, ST-Quartz as the substrate, single Eigen as the technique, and a uniform-uniform structure, while other alternatives might be totally different.

After the Prolog inference engine determines all of the alternatives, SAWCOM will then submit each alternative to SAWCAD-PC. Since some of the functions

in SAWCAD-PC involve much computing time, it is very desirable to decrease the number of alternatives as much as possible. If all possible alternative combinations were to be submitted to SAWCAD-PC, then 192 solutions would have to be checked thus wasting valuable computer time. The goal is to have the Prolog code bring the 192 possible alternatives to under ten solutions.

Parameter	Possible Values
Sampling Rate	2 4
Substrate	$LiNbO_3$-128 $LiNbO_3$-YZ ST-Quartz $LiTaO_3$
Technique	single Eigen dual Eigen Remez window
Structure	uniform—uniform uniform—withdrawn uniform—apodized withdrawn—apodized withdrawn—withdrawn apodized—msc—apodized

Table 1, Parameters and Their Possible Values

The Prolog code eliminates possible solutions by checking the relationships between center frequency (F_0), fractional bandwidth, minimum line width, shape factor, band width, transition band widths, temperature range and available substrates. Six rules are being used to determine the possible solutions as stated below:

Rule #1: ST-Quartz cannot be used with apodised-msc-apodised.

Rule #2: Any structure that uses withdrawn weighting must have more than 150 electrodes.

Rule #3: The sampling rate must accommodate the minimum line width.

Rule #4: The Q (quality factor) of the substrate must be less than the Q of the filter.

Rule #5: The window technique cannot be used if the shape factor is greater than 2.

Rule #6: The frequency transition bands must be less than the frequency deviation due to the temperature range (ΔT).

As in any Prolog code, the rules do not have to be listed in any particular order. This order independence helps makes the Prolog program easy to write and understand. *Figure 3* shows a code segment from the Prolog code to illustrate how an inference engine of this caliber can be developed.

```
design_filter :-
   read_fsp(F0, Frac_bw, Min_line_width,
            Shape_factor, Band_width,
            Trans_band_1, Trans_band_2,
            Delta_T, Subs_available),
   substrate_availability(Substrate,
                          Subs_available),
   /* begin rules here */
   sampling_rate(Fs, F0, Min_line_width,
                 Substrate),
   substrate(Substrate, Fs, Frac_bw),
   layout(Substrate, Structure),
   num_electrode_check(Structure, Fs, F0,
                       Band_width),
   synthesis_technique(Technique, Structure,
                       Shape_factor),
   temp_stability_check(Substrate, Delta_T,
                        F0, Trans_band_1,
                        Trans_band_2),
   /* end rules here */
   write_sdr(Structure, Substrate, Technique,
             Fs),
   fail.
```

Figure 3, Code Segment from Prolog Code

This code segment shown in *Figure 3* contains the main predicate used in determining the possible solutions. As marked by the comments, there are six rules being implemented, each of which depend on many variables which ultimately determine the structure, substrate, technique, and sampling frequency to be used. The "fail" at the end of the code allows Prolog to obtain all alternatives and not just the first.

4.0 Example Execution

An example of execution of SAWCOM is shown to illustrate the efficiency of the system. *Figure 4* shows a portion of the frequency specification file that SAWCOM reads.

```
[Frequency Template]
Band1=0,     141,   0,  60.0, -1, -1
Band2=142.5, 157.5, 0.5, 0.5, -1, -1
Band3=159,   300,   0,  60.0, -1, -1

[Substrates Available]
128_LinbO3=y
YZ_LiNbO3=y
ST_Quartz=y
LitaO3=y

[Physical Constraints]
Min_Temperature=-55     ;C
Max_Temperature=125     ;C
```

Figure 4, Portion of Frequency Specification File

The frequency template contains three bands, each containing minimum and maximum frequencies (MHz),

magnitude (linear), magnitude tolerance (dB), phase (degrees) and phase tolerance (degrees). A "-1" is used to indicate a "don't care" situation. In other words, phase is not important in this example. The maximum and minimum frequency differences between adjacent bands are the frequency transition bands. *Figure 5* illustrates where the transition bands occur.

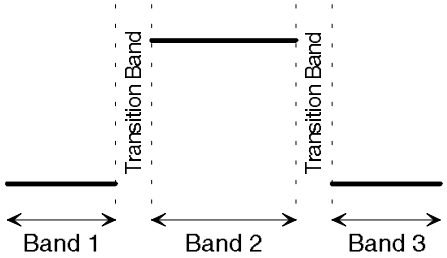

Figure 5, Illustration of Frequency Template

The particular example shown in *Figure 4* resulted in 8 alternatives to be submitted to SAWCAD-PC. *Figure 6* shows these alternatives in which ST-Quartz and $LiTaO_3$ are the only substrates even though the user stated that all four substrates were available.

```
structure____= uniform_apodized
substrate____= st_quartz
technique____= single_eigen
sampling_rate= 4

structure____= uniform_apodized
substrate____= st_quartz
technique____= remez
sampling_rate= 4

structure____= uniform_apodized
substrate____= litao3
technique____= single_eigen
sampling_rate= 2

structure____= uniform_apodized
substrate____= litao3
technique____= remez
sampling_rate= 2

structure____= apodized_msc_apodized
substrate____= litao3
technique____= dual_eigen
sampling_rate= 2

structure____= uniform_apodized
substrate____= litao3
technique____= single_eigen
sampling_rate= 4

structure____= uniform_apodized
substrate____= litao3
technique____= remez
sampling_rate= 4

structure____= apodized_msc_apodized
substrate____= litao3
technique____= dual_eigen
sampling_rate= 4
```

Figure 6, Result of Example Execution

5.0 Future Progress and Conclusions

SAWCOM will eventually go though the whole design process, beginning with the user indicating the desired SAW filter specifications all the way to the fully automated design. As indicated in the "end processor", a history file will be updated with each design so that future SAW filters can take advantage of similar filters previously designed. Many more rules will be incorporated in the Prolog solutions search as well as a second Prolog engine that helps the interaction with SAWCAD-PC.

References

[1] Richie, S. M. and Malocha, D. C. "A First Generation CAD Compiler for Surface Acoustic Wave Filters Using Bidirectional Transducers." *IEEE Ultrasonics Symp. Proc.*, 1989.

[2] Richie, S. M., Abbott, B. P., and Malocha, D. C. "Description and Development of a SAW Filter CAD System" *IEEE Trans. Microwave Theory and Techniques*., vol. 36, no. 2, pp. 456-466, 1988.

[3] Richie, S. M., McCollister, M. J., and Malocha, D. C. "Development of the Rule Base for a Design Automation System for SAW Filters Using Bidirectional Transducers," *IEEE Ultrasonics Symp. Proc.*, 1989, pp. 155-158.

[4] Morgan, D. P. *Surface-Wave Devices for Signal Processing*. New York: Elsevier, 1991.

[5] McCellan, J. H., Parks, T. W., and Rabiner, L. R. "A Computer Program for Designing Optimum FIR Linear-phase Digital Filters," *IEEE Trans. Audio Electroacoust.*., vol. AU-21, pp. 506-526, 1973.

[6] Malocha, D. C. and Bishop, C. D. "The Classical Truncated Cosine Series Functions with Applications to SAW Filters," *IEEE Trans. Ultrasonics, Ferroelectrics and Frequency Control*, vol. UFFC-34, pp. 75-85, 1987.

[7] Vasile, C. F. "A Numerical Fourier Transform Technique and its Applications to Acoustic Surface Wave Bandpass Filter Synthesis and Design," *IEEE Trans. Sonics and Ultrasonics*, vol. SU-21, pp. 7-11, 1974.

[8] Hartmann, C. S., Bell, D. T., and Rosenfeld, R. C. "Impulse Model Design of Acoustic Surface Wave Filters," *IEEE Trans. Microwave Theory Tech*., vol. MTT-21, pp. 162-175, 1973.

[9] Datta, S., Hunsinger, B. J., and Malocha, D. C. "A Generalized Model for Periodic Transducers with Arbitrary Voltages," *IEEE Trans. Sonics and Ultrasonics*, vol. SU-27, pp. 235-242, 1979.