Proc. of the Second Intl. Conf. on Advances in Computer and Information Technology -- ACIT 2013

ISBN: 978-981-07-6261-2

An Approach for Model Based Programming using LabVIEW

Usha D
Department of ISE, CORI,
PES Institute of Technology,
Bangalore.

V K Agrawal
CORI, PES Institute of Technology
Bangalore.

Anandita Chakraborty,
CORI, PES Institute of Technology,
Bangalore.

Kochaleema K H
Scientist 'F',
Naval Physical and Oceanographic Laboratory,
Kochi

***Abstract* - The purpose of this paper is to present a detailed description of model based programming (MBP) approach for addressing problems associated with designing complex control systems. Text-based programming tools are pooled with lengthy paths in designing the complex system. With this view in consideration, the concept of graphical programming language is evolved. The graphical languages provide simplicity which can be easily manipulated by the user in an interactive way. These languages use specific spatial grammar for program construction. Lab VIEW being a graphical programming language supports system design platform. Such a platform provides engineers with the tools needed to create and put forth measurements and control system. With its built-in engineering-specific libraries of software functions and hardware interfaces, one can program mathematical model instead of writing text codes. The above said lesson is illustrated by considering a case study i.e., design of Automatic Gain Controller system (AGC) using MBP approach. Automatic Gain Control is used in electronic devices where maintaining a constant signal level at the output in-spite of the variation in the input level. The case study demonstrates that Model based approach helps one in programming complex systems without much knowledge of conventional programming languages.**

Keywords— **Model Based Programming, Automatic Gain Controller, Embedded Systems.**

I. INTRODUCTION

Today, in the electronics system industry, reactive real time systems are persuasive. Their applications range from remote sensing, vehicle control and communication systems to household appliances. The specifications of these applications changes time to time. Due to this need of changing specifications continuously, the use of software programmable components with behaviors that can be easily changed is much required.

These are also known as embedded systems, i.e., systems that use computer programs to perform operations but are not perceived as a computer themselves. They also intend to react continuously to the situation at the speed of the environment. This paper mainly concentrates on the approach of Model based programming for the applications in embedded systems.

Programming complex systems involves reasoning through its every single system interactions which is not only a time consuming process but is also error prone [1]. Model based programming overcomes these limitations by allowing the engineers to program by specifying high-level control strategies and by assembling Logical models of the system hardware and software. Thus, it provides the programmers with embedded languages that avoid common-sense mistakes by automatically reasoning them from hardware models. LabVIEW (short for Laboratory Virtual Instrumentation Engineering Workbench) is one such tool from National Instruments which allows an engineer to enjoy the benefits of model based programming.

In this paper, for getting look and feel of model based Programming (MBP), a simple case study to understand MBP is chosen. The case study chosen is Automatic gain control (AGC)[9]. AGC is an adaptive system found in electronic devices where the average output signal level is fed back to adjust the gain to an appropriate level for a range of input signal levels. This paper demonstrates the design for AGC using the model based programming approach by means of the tool LabVIEW.

The reactive model based programming language (RMPL) presented by Brian. C. Williams et al., [1] which enables the programmer to delegate into and guide the reasoning methods of model based autonomy. The performance of RMPL was fully constructed to cover synchronous programming and hence moving towards the unification of goal-directed AI executive with its underlying real-time language. The potential of model based programming can be made better when compared with all assessed formal language in terms of time complexity. It is observed that combined approach of mathematical models and formal methods can reduce time and space complexity by defining a basic set of equations that are common to many dynamic systems.

The MBP methodology demonstrated in this paper using LabVIEW reduces the cost and time in programming the mathematical models. The intermediate result of the method is a collection of libraries in the form of Virtual Instruments (VI's). Time and space complexity is reduced by avoiding text codes and helping the system engineer to for the faster development of complex system. The project libraries are constructed for all the associated modules of the case study and an association is done for the real time analysis.

Organization: The paper begins with the introduction to model based programming approach in embedded systems. The

Proc. of the Second Intl. Conf. on Advances in Computer and Information Technology -- ACIT 2013

ISBN: 978-981-07-6261-2

remainder of the paper is organized as follows Section 2 presents the overview of literature survey. Section 3 describes the background required for understanding of MBP. Section 4 describes the proposed methodology for designing the case study. Section 5 describes the implementation followed by Section 6 which speaks about our case study on AGC. Section 7 provides the result and discussion on the paper and Section 9 describes the concluding remarks.

II. RELATED WORK

Nowadays, embedded systems are playing a major role in our life. This gives rise to a challenge to produce systems that function correctly. Model based design involves forming of the model and ensuring its rightness before it is implemented. This approach is adopted by Li Tan [1] to propose a model-based framework for building self-monitoring embedded system. Model based programming captures the advantages of both constraint programming and embedded system programming. Embedded systems are also used in software components where the program behaviour changes from time to time [2].

Brian C. Williams [3], [4] describes the Reactive Model-based Programming Language (RMPL) with the model-based execution kernel called Titan. The concept of model based programming has been used to describe Titan, into JPL's Mission Data System (MDS), a unified state-based architecture[5], to synthesize several subfields of AI, exploiting research in robot programming, reasoning about action, and model-based reasoning about physical systems [6]. Brian C. Williams and Vineeth Gupta [7] have introduced the probabilistic, hierarchical constraint automata (HCA) which allow Markov processes to be expressed in a compact representation that preserves the modularity of RMPL programs. B.C Williams has implemented model based programming in fault aware systems [8].

Isaac Martinez G. [9] describes the theory and design of AGC circuits ranging from audio to RF applications in his paper. Alegre Pérez et al., in their book [10] have examined AGC loop circuits and have provided solutions in the fields of wireless receivers, such as WLAN and Bluetooth receivers.

III. III. BACKGROUND

A. *Model Based programming*

Embedded systems are achieving unprecedented levels of robustness by dramatically increasing their use of computation. Programming these systems involves mining the particulars of interactions between each single system and there sub-systems. We envisage a future with large networks of highly robust which would be programmed with models, describing themselves and their environments.

To accomplish this, the embedded systems, need to radically reconfigure themselves in response to failures, and then accommodate the failures during their remaining operational lifetime. This requires a rapid development support by use of embedded programming languages that are able to reason about and control underlying hardware from engineering models. The vision spoken above will be brought in to action by using an approach known as **Model Based Programming.**

The notion of model based programming is an approach for writing software for embedded reactive systems as shown in the below Fig 1. It concentrates on providing a programming language centered on hidden state, and which seamlessly incorporates powerful model-based deductive engines into the program.

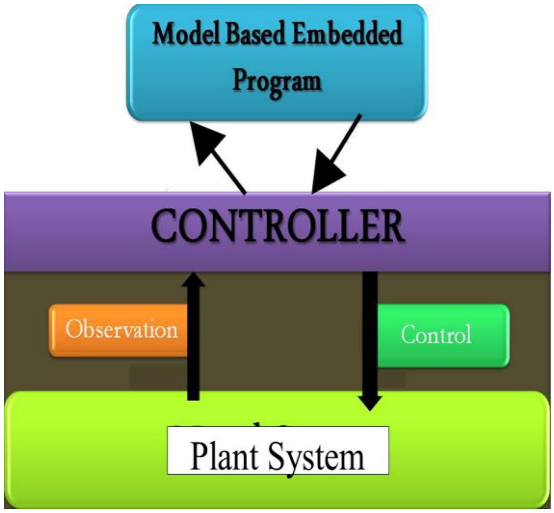

Figure 1 Model Based Programming- An Approach

Model based programming (MBP) addresses problems associated with signal processing, designing complex control systems and communication systems. MBP is an approach to develop embedded systems that can reason about and control their hardware using corresponding software models. Rather than using complex structures and extensive software code, designers can use MBP to define models with advanced functional characteristics using continuous-time and discrete-time building blocks.

These built models used with simulation tools can lead to rapid prototyping, software testing, and verification. Hence, MBP is a language developed to elevate programming to the specification of intended states.

IV. IV. PROPOSED METHODOLOGY

The model design involves working through a series of steps that start from the very basic needs of the design to the final step that delivers the product. Firstly the inputs required describe the specifications and the final step gives the implementation of the design. The intermediate steps logically connect a set of precise relationships that leads to the final model. On obtaining the design, verification processes are carried out to ensure the correctness of the model. The functional specifications of the model explain its operation in detail. On the other hand, properties list of the model gives details of its behavior. The properties are grouped based on whether they are inherent or can be verified syntactically or that which can be verified semantically as shown in the figure 2 below.

Proc. of the Second Intl. Conf. on Advances in Computer and Information Technology -- ACIT 2013

ISBN: 978-981-07-6261-2

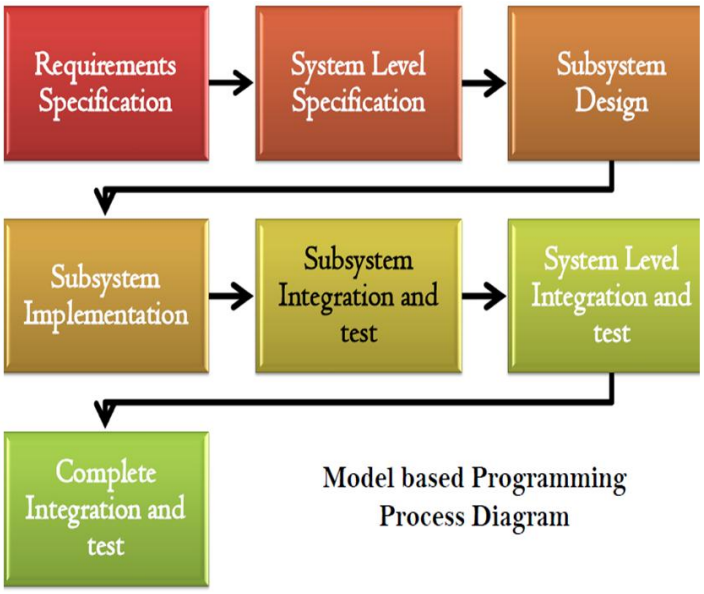

Figure 2. Process Diagram of Model Based Programming Approach.

B. Algorithm:

Step 1: Obtain a high level design of the model.
Step 2: Collect the necessary mathematical equations and theory involved for the development of the modules.
Step 3: Choose the required data and work on it further to make it accurate. Care must be taken as any small mistake here can affect the final performance of the model.
Step 4: Analyze the data chosen to identify its working in real time.
Step 5: From the equations chosen, form the sub modules. Some of the basic sub-modules can be deployed directly from the library. The rest can be designed using basic operations available in the library. Association of all the parts forms one module of the model desired to design.
Step 6: The errors must be verified in each of the module designed (here, variable gain amplifier, amplifier, detector, low pass filter and difference amplifier)
Step 7: The library of modules designed can then be collaborated to form the final model desired to design in the real time situations (here, automatic gain amplifier)

While figure 3 suggests a purely top-down methodology, any real system design needs more interaction between specification and implementation. Nonetheless, when a design is complete, the best way to present and document it is top down. The methodology used here gives a clear insight of the process with in the model based programming approach.

V. IMPLEMENTATION OF MBP APPROACH FOR OUR CASE STUDY.

Model based approach is used to eliminate the gap between the properties which are proven by specifications and the programs that are supposed to implement these specifications along with reasoning on these specifications. These specifications are synthesized and model is generated in the hardware. This programming framework emphasizes modularity in software design. The designer must verify the properties and constraints at each level of abstraction. The concept of time is also very essential for mission-critical sequences. One must

Consider time in the control and the behaviour models in addition to managing the complexity.

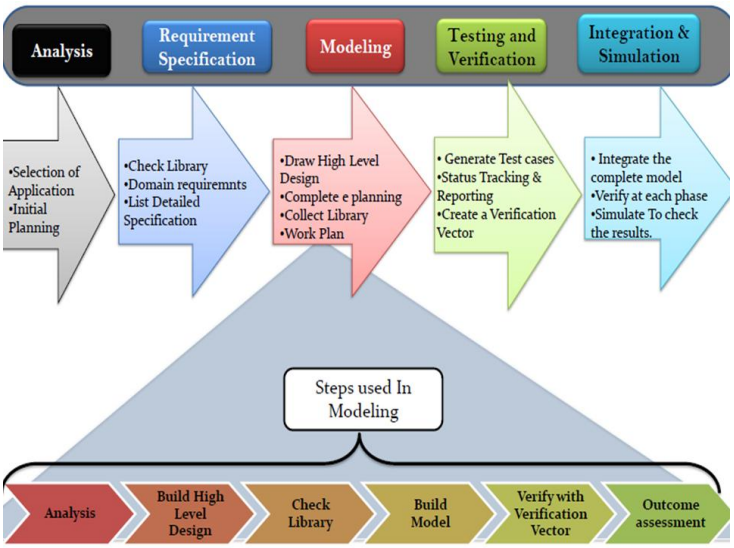

Figure 3. Methodology of Model Based Programming

In model based programming languages programs are built by a set of basic combinators. These combinators are used to define numerous derived control constructs. Finally these constructs can be compiled into automata representations, which are used for the analysis of real-time properties.

The proposed AGC model designed in LabVIEW has five sub-models namely variable gain amplifier, amplifier, detector, low pass filter and difference amplifier. LabVIEW programs called virtual instruments (VIs) have both an interactive user interface and a source code equivalent, and accept parameters from higher-level VI's. The front panel contains push buttons, knobs etc. and back panel contains the design.

C. Functional Description of AGC Mathematical Model

One might wonder as to why the designing of an AGC is of any importance? The reason being, the need for selectivity and good control of the output signal's level is a fundamental feature in the design of any systems these days.

Proc. of the Second Intl. Conf. on Advances in Computer and Information Technology -- ACIT 2013

ISBN: 978-981-07-6261-2

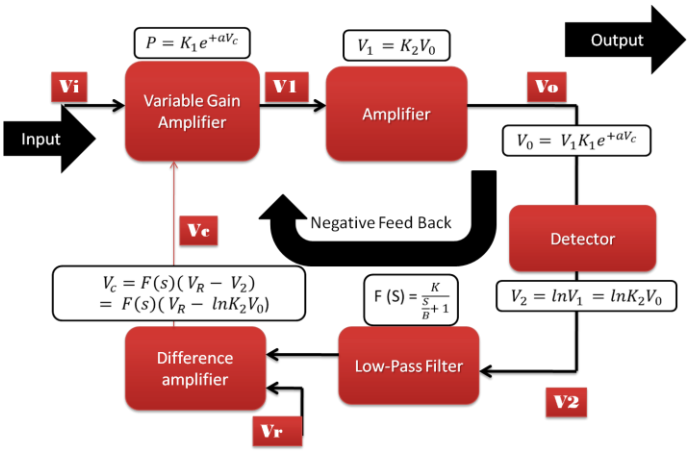

Figure 4. Mathematical equations for various blocks in AGC modeling.[9]

The Mathematical equation to build the models of AGC [9] is clearly shown in the figure 4. The modules of AGC are represented in LabVIEW by designing a set of library which could be called individually on to the block panel for further implementation.

VI. VI. RESULTS AND DISCUSSION

This Section provides us with the results of the model based programming approach designed with the aid of the mathematical models in LabVIEW software. Coming to our case study, the individual modules of the Variable Gain Amplifier (VGA), amplifier, detector, low pass filter and difference amplifier are designed with the help of mathematical expression. A list of libraries are designed as shown in the below diagram.

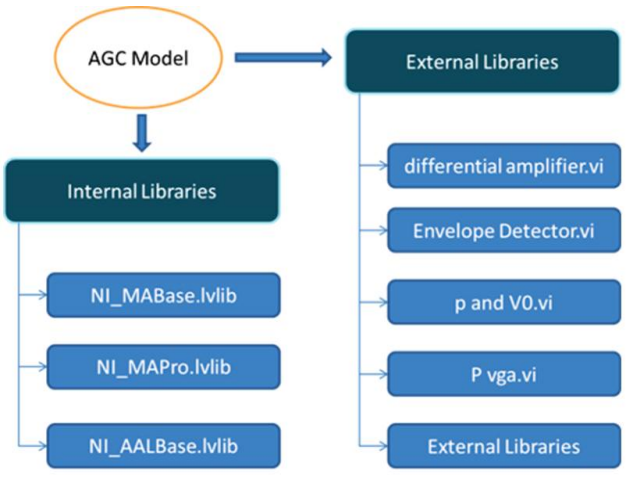

Figure 5 list of internal and external libraries built for AGC model.

The variable gain amplifier is a signal conditioning amplifiers with electronically settable voltage gain. With aid of internal libraries we build the model for variable gain amplifier. Here, frequency and amplitude value of the wave is given as input stating the signal type. $V_o$is the output signal here. A waveform graph is connected at the output end to show the output waveform. Further amplifiers are used to increase the power of a signal by use of an external energy source. It is assumed here that the output of the envelope detector is always positive.

In the circuit of the detector, one signal generator is considered to generate the sine wave with carrier frequency of 100 and which simulates sine waves 50% amplitude modulated by triangular wave. Another sine wave of modulation frequency 0.5 is considered. Both signals are taken as input to the amplifier component, the output of which will be altering current. Then the

RF signal will be passed through wires and that AC will be given as input to the rectifier. The output of the rectifier will be in DC form and finally, the low-pass filter is used to get the ultimate waveform which can be displayed using waveform graph.

The difference amplifier finds the difference between two waves and then the comparator component is used to get the difference between values of them. One can study the waveform graph by connecting with the output wave values.

Therefore, one can implement each and every module likewise. Respective code can also be generated after defining the library functions. The files with extension .vi must be written to fulfill the requirement and thereby easily generating the code.

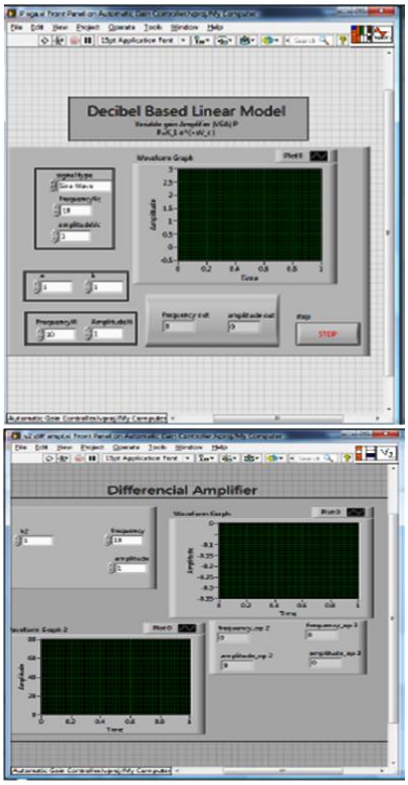

Figure 6. Front views of the proposed model

ISBN: 978-981-07-6261-2

The user can view our model as shown in Fig 5. These modules are implemented as individual VI's for each mathematical equation in LabVIEW. The VI's consists of series of library modules as per the requirement. One can design it using other tools such as Charon, Estrel, Lustre, Signal etc. with respect to their requirements. Considering its positive characteristics over traditional methods, model based programming approach can be thus used in modeling various kinds of structures.

VII. CONCLUSION

Model based programming is by all means a powerful method in the designing of a complex structure. The ease provided by it in terms of reduction in time required to frame the design and to verify the errors lessen the stress on any programmer. The approach is simpler to implement models when compared to the text based programming.

As per the case study designing the mathematical models of AGC using LabVIEW not only makes it easier for the programmer to design the model but also makes the implementation simple for the user. The wide range of modules available in the LabVIEW library provides the programmer engineer with ample amount of choices for development of any model. Thus, it can be concluded that the methodology proposed in this paper is useful for the present day applications and the approach used to model it is efficient for both the programmer and the user.

ACKNOWLEDGMENT

This research has been made possible through the funding of the NRB (Naval Research Board). We are extremely grateful to them for the needful data they have supplied.

REFERENCES

[1] Li Tan: Model Based Self-Monitoring Embedded Programs With Temporal Logic Specifications

[2] Edwards.S.,Lavagno L., Lee E.A., Sangiovanni-Vincentelli A. (1997) Design of Embedded Systems: Formal Models, Validation, and Synthesis. Proceedings of the IEEE.85,366-390.

[3] Williams B.C., Ingham M.D.(2002). Model Based Programming: Controlling Embedded Systems by Reasoning About Hidden State. Conf. on Principles and Practice of Constraint Programming.

[4] Brian C. Williams, Michel D. Ingham, Seung H. Chung and Paul H. Elliott : Model Based Programming of Intelligent Embedded Systems and Robotic Space Explorers

[5] Horvath G., Ingham M., Chung S., Martin O., Williams B.(2006). Practical Application of Model Based Programming and State-Based Architecture to Space Missions.

Jet Propulsion Laboratory, National Aeronautics and Space Administration.

[6] Mcllraith S.A. Model-Based Programming using Golog and the Situation Calculus. *Citeseerx*

[7] Williams B.C., Gupta V.(1999). Unifying Model-based and Reactive Programming within a Model-Based Executive.*Caelum Research Corporation.*

[8] Williams B.C., Ingham M., Chung S., Elliott P., Hofbaur M. (2003). Model based Programming of Fault-Aware Systems. *AI Magazine*.

[9] Isaac Martinez G : Automatic Gain Control (AGC) circuits - Theory and design

[10] Alegre Pérez, J.P; Pueyo, S.C; LÓpez, B.C : Chapter 2, AGC Fundamentals from the book " Automatic Gain Control, Techniques and Architecture for RF Receivers"

[11]http://en.wikipedia.org/wiki/Model-based_design