FPGA Implementation of Linear Observer

Ravindra S. Rana and Bharti Kumari

***Abstract*—This paper proposes implementation of linear observer and full state feedback controller using pole placement method for second order system. System consists of two 1st order differential equations, solved using Euler and Runge Kutta-2(RK-2) method and linear observer implemented on FPGA Vietex-5 platform. Sampling rate of 0.1 millisecond is achieved for hardware co-simulation of linear observer. Direct comparison of resource utilisation to implement linear observer, accuracy and complexity for Euler and RK-2 methods is presented.**

***Index Terms*—FPGA, linear observer, euler, runge kutta-2, system generator.**

I. Introduction

FPGAs have become more resourceful with recent developments in VLSI technology. It has qualities such as parallel processing capability, high sampling rates, flexibility in design and reliability. Not surprisingly, FPGA chips appear more frequently in controller designs [1], [2]. Solving ordinary differential equations is essential in control field [3]. Now in embedded world one of the important ways is use of an FPGA which is suitable for fast implementation and quick hardware verification. The FPGA consists of three major configurable elements: configurable logic blocks (CLBs) arranged in an array that provide the functional elements and implements most of the logic in an FPGA, input-output blocks (IOBs) that provide the interface between the package pins and internal signal lines, and programmable interconnect resources that provide routing path to connect inputs and outputs of CLBs and IOBs onto to the appropriate network. A digital design can be created by using schematic digital design editor that uses graphic symbols of the circuit or by using hardware description languages such as Verilog, Very High Speed Integrated Circuit Hardware Description Language (VHDL).

Controller algorithms could be implemented on FPGAs with a combination of hardwired logic and a floating point arithmetic unit (FPU) without the use of an embedded processor in the design [4]. Chan *et al* [5] have conducted a study on PID controller implementation on an FPGA and they managed to decrease the resources required by a multiplier-based design significantly. What they propose is to replace the multipliers by a distributed arithmetic based design utilizing look up tables and they managed to decrease the resource requirement down to 4 to 13% of the former design. However they increased the computation time from 1 cycle to 13 to 26 cycles. A very similar study by Tao *et al* [6] managed to decrease the logic element requirement from 51.7% to 0.8-1.5%, increasing the computation cycle from 1 to 64-33 cycles.

In this paper an approach to implement linear observer with state feedback controller is proposed for 2nd order plant. Euler and RK-2 methods are used to solve differential equations. Plant states are estimated using both numerical methods and state feedback controller calculated in system generator. Comparison for hardware resource utilization is done to implement observer using both methods. In Section II theory for linear observer, Euler and RK-2 method is given. Implementation and results are discussed in Section III. Hardware co-simulation results are concluded in Section IV.

Manuscript received February 5, 2013; revised April 23, 2013.

The authors are with the College of Engineering Pune, India (e-mail: ranaravindra08@gmail.com,: atmaja.thakur1@gmail.com).

II. Theory

A. Linear Observer

It is often convenient when designing feedback control systems to assume initially that the entire state vector of the system to be controlled is available through measurement. Thus for the linear time-invariant system governed

$$\dot{x} = Ax(t) + Bu(x, t) \tag{1}$$

where $x(t)$ is an $n\times1$ state vector, $u(x, t)$ is an $m\times1$ input vector, A is an $n\times n$ system matrix, and B is an $n\times m$ distribution matrix, one might, design a feedback law of the form

$$u(x, t) = -Kx(t) \tag{2}$$

Which could be implemented if $x(t)$ were available. This is, for example, precisely the form of control law that resu1ts from design techniques that place poles at pre-specified location and from other technique that, insure stability and in some sense improve system performance.

If the entire state vector cannot be measured, as is typical in most complex systems, the control law deduced in the form $u(x, t) = -Kx(t)$ cannot be implemented. Thus either a new approach that, direct accounts for the nonavailabi1ity of the entire state vector must be devised, or a suitable approximation to the state vector must be determined that can be substituted into the control law. In almost every situation the latter approach, that of developing and using an approximate state vector, is vastly simpler than a new direct attack on the design problem.

Adopting this point of view, that an approximate state vector will be substituted for the unavailable state, results in the decomposition of a control design problem into two phases. The first phase is design of the control law assuming that the state vector is available. The second phase is the

DOI: 10.7763/IJFCC.2014.V3.258

design of a system that produces an approximation to the state vector. Such system in a deterministic setting is called an observer. The observer is a dynamic system whose characteristics are somewhat free to be determined by the designer, and it is through its introduction that dynamics enter the overall two-phase design procedure when the entire state is not available.

The observer is governed by

$$\dot{\hat{x}} = A\hat{x}(t) + Bu(x,t) + L(y - \hat{y}) \tag{3}$$

where y available measured output, $\hat{y}$ estimated output, L user designed constant gain matrix, $\hat{x}$ estimated states. Observer can be designed to have arbitrary dynamics if the original system is completely observable. System is completely observer if the matrix

$$[C' \vdots A'C' \vdots \ \cdots \ \vdots \ (A')^{n-1}C']$$

has rank n. If an $n \times n$ matrix A and $m \times n$ matrix C satisfy this condition we shall say pair (C, A) is completely observable.

B. Euler Method

Euler's method is the simplest approach to computing a numerical solution of an initial value problem. However, it has about the lowest possible accuracy. If we wish to compute very accurate solutions, or solutions that are accurate over a long interval, then Euler's method requires a large number of small steps. Consider the first order differential equations:

$$\dot{\hat{x}}_1 = \hat{x}_2 + L_1(y - \hat{y}) \tag{4}$$

$$\dot{\hat{x}}_2 = -a_1\hat{x}_1 - a_2\hat{x}_2 + bu(x,t) + L_2(y - \hat{y}) \tag{5}$$

Combining above two equations in matrix form,

$$\dot{\hat{x}} = A\hat{x} + bu + L(y - \hat{y}) \tag{6}$$

where $A = \begin{bmatrix} 0 & 1 \\ -a_1 & -a_1 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ b \end{bmatrix}$, $L = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix}$, $\hat{x} = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix}$

Starting at some time t_o, the value of $\hat{x}(t_0 + h)$ can then be approximated by the value of $\hat{x}(t_0)$ plus the time step multiplied by the slope of the function, which is the derivative of $\hat{x}(t_0)$

$$\hat{x}(t_0 + h) \approx \hat{x}(t_0) + hf(t, \hat{x}(t)) \tag{7}$$

where h = step size

We will call this approximate value $\hat{x}^*$

$$\hat{x}_1^* = \hat{x}_1 + k_1 \tag{8}$$

$$\hat{x}_2^* = \hat{x}_2 + k_2 \tag{9}$$

$$k_1 = h\{\hat{x}_2 + L_1(y - \hat{y})\} \tag{10}$$

$$k_2 = h\{(-a_1 - k_1)\hat{x}_1 + (-a_2 - k_2)\hat{x}_2 + r + L_2(y - \hat{y})\} \tag{11}$$

For numerical solutions of an initial value problem there are two ways to measure the error. The first is the error of each step. This is called the Local Truncation Error or LTE. The other is the total error for the whole interval [a, b]. We call this the Global Truncation Error or GTE. For the Euler method the LTE is of order $O(h^2)$, i.e. the error is comparable to h^2. We can roughly get the GTE from the LTE by considering the number of steps times the LTE. For any method, if [a, b] is the interval and h is the step size, then $n = \frac{(b-a)}{h}$ is the number of steps. Thus for any method, the GTE is one power lower in h than the LTE. Thus the GTE for Euler is $O(h)$.

Suppose that you need to solve an IVP with an error of less than 10^{-4}. If you use the Euler method, which has GTE of order *O(h)*, then you would need $h \approx 10^{-4}$. So you would need about $n \approx (b-a) \times 10^{-4}$ steps to find the solution.

C. Runge Kutta 2 Method

The RK2 method acts to increase the accuracy by getting a more accurate estimate for the slope throughout the interval. In this case the GTE is $O(h^2)$, so we would need to use $h \approx 10^{-2}$. This would require about $n = \frac{(b-a)}{10^2}$ steps.

The equations are then:

$$k_1 = hf(x_n, y_n) \tag{12}$$

$$k_2 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1) \tag{13}$$

$$y_{n+1} = y_n + k_2 + O(h^2) \tag{14}$$

Let, the same system as shown in equation (6), We will call this approximate value $\hat{x}^*$

$$\hat{x}_1^* = \hat{x}_1 + k_{21} \tag{15}$$

$$\hat{x}_2^* = \hat{x}_2 + k_{22} \tag{16}$$

$$k_{11} = h\{\hat{x}_2 + L_1(y - \hat{y})\} \tag{17}$$

$$k_{21} = h\{\hat{x}_2 + \frac{1}{2}k_{11} + L_1(y - \hat{y})\} \tag{18}$$

$$k_{12} = h\{(-a_1 - k_1)\hat{x}_1 + (-a_2 - k_2)\hat{x}_2 + r + L_2(y - \hat{y})\} \tag{19}$$

$$k_{22} = h\{(-a_1 - k_1)(\hat{x}_1 + \frac{1}{2}k_{11}) + (-a_2 - k_2)(\hat{x}_2 + \frac{1}{2}k_{12} + r + L_2(y - \hat{y})\} \tag{20}$$

When the time step was increased the position error exhibited sigmoidal behavior and with further increase, gathered error approximately linearly with time. Thus, time steps in the range of ten times greater than the Euler's can be used while preserving a reasonable degree of accuracy. Thus the real advantage of higher order methods is that they can run a lot faster at the same accuracy. This can be especially important in applications where one is trying to make real-time adjustments based on the calculations. Such is often the case in robots and other applications with dynamic controls.

III. Implementation and Result

The selection of the word size is up to the designer and is to be chosen by considering both the system properties (such as feedback resolution) as well as the features of the implementation method.

Depending on the accuracy needed either for a given system simulation or for ODE solving, we can define a specific data type for modules and basic elements. For each basic element such as subtraction, addition and multiplier we define a fix-point data type with proper "integer" and "precision" ranges.

The MSB (most significant bit) bit is for sign, and for subtraction we use the 2's complement method. We use 32 bits for sign registers and storing the values. All components are synchronal and the system is operated by a common clock. At its most basic level, a DSP48 is a multiplier with a combination of adders and many optional operations. It has a 32-bit sign input signal and a 64-bit sign output result. This result is then converted to 32-bits. For implementing a system simulation or solving an ODE by a flow diagram, we need some basic and fundamental arithmetic components, such as summation, subtraction, and multiplier.

Still now, there is no general purpose architecture for solving any type of ODE equations [7], but we propose a method for designing a system and solving ODEs in a straight forward process (flow diagram). The flow diagram is consisting of many basic elements that are coupled together. Fig.1 shows a flow diagram for solving the so-called linear observer equation (see Equation (18) and (19)).

$$k_{21} = h\{\hat{x}_2 + \frac{1}{2}k_{11} + L_1(y - \hat{y})\}$$

$$\Downarrow$$

$$k_{21} = \frac{3}{2}hk_{11}$$

$$\Downarrow z$$

$$k_{22} = t_1 \times k_{11} + t_2 \times k_{12}$$

$$t_1 = \frac{h}{2}(-a_1 - k_1), \quad t_2 = 1 + \frac{h}{2}(-a_1 - k_1)$$

Fig. 1. Flow Diagram.

TABLE I: Resources Used For Implementing A 32-Bit Ode Solver For Linear Observer Equation

S. No.	Devices	EULER	RK-2
1	DSP48Es (128)	26	32
2	Slice LUT- Flip Flop pairs (44800)	1296	1118
3	Slice LUTs (44800)	977	1059
4	route-thrus (89,600)	118	200

Pipeline registers are a unique advantage of the DSP48 block compared to other FPGA DSP architectures [8]-[10]. Xilinx has dedicated many DSP48 slice in Virtex-5 family for speeding-up of calculations. The number of resources used after synthesis for solving the Euler and RK-2 equations in terms of DSP48 slices, Flip-Flop slices and LUT Slices is shown in the Table I.

As shown in Table II RK-2 uses more multipliers, adders etc. compared to Euler method. In Euler method, stable estimated states are available after 7 clock cycles where as using RK-2, it takes 10 clock cycles. State feedback controller uses that estimated states to generate control action, hence in RK-2 method control signal also delayed by 3 clock cycle compared to Euler method.

TABLE II: Synthesis Report

S. No.	Devices	EULER	RK-2
1	Multipliers	8	11
2	Adder	8	10
3	Subtracter	1	1
4	Register(64 bit)	24	33
5	1- bit Xor	27	33

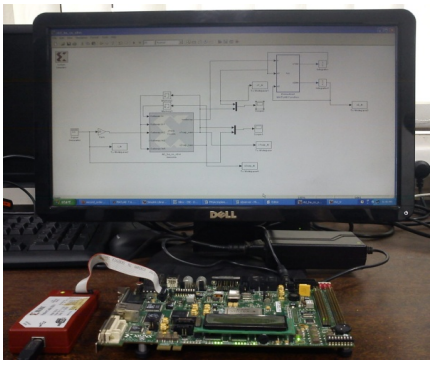

Fig. 2. Experimental setup.

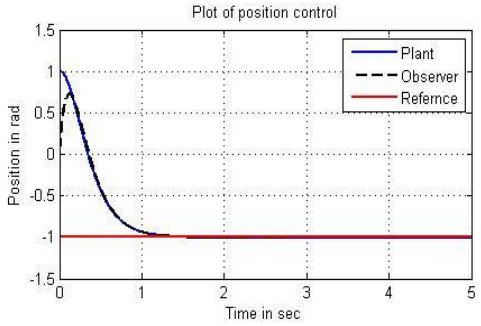

Fig. 3. Shows the observer hardware co-simulation response using euler method.

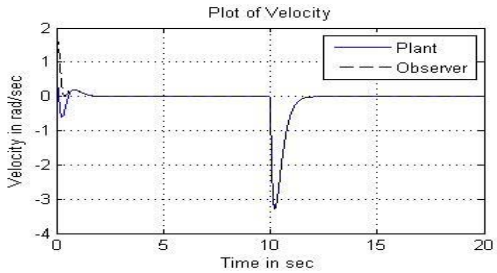

Fig. 4. Shows the observer hardware co-simulation response using RK-2 method

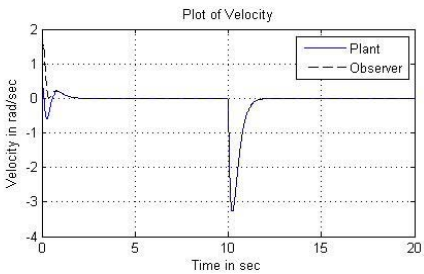

Fig. 5. Shows the observer hardware co-simulation response using Euler method

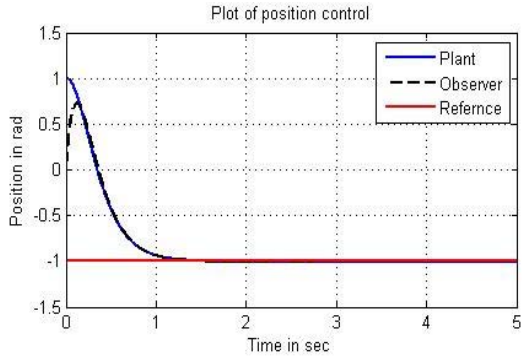

Fig. 6. Shows the observer hardware co-simulation response using RK-2 method

IV. Conclusion

In this paper we implement Euler and RK-2 method for linear observer. We have compared the implementation results with respect to accuracy, complexity and resources utilized to implement linear observer on FPGA. From the results, we conclude that RK-2 method gives better accuracy but utilizes more resources of FPGA compared to Euler method.

Reference

[1] J. U. Cho, Q. N. Le, and J. W. Jeon, "An FPGA-Based Multiple-Axis Motion Control Chip," *IEEE Trans. on Industrial Electronics,* vol. 56, no. 3, pp. 856-870, March 2009.

[2] Y. S. Kung, R. F. Fung, and T. Y. Tai, "Realization of a Motion Control IC for X-Y Table Based on Novel FPGA Technology," *IEEE Trans. on Industrial Electronics*, vol. 56, no. 1, pp. 43-53 Jan. 2009

[3] B. F. Mullins. A Guide to Solving Simple Ordinary Differential Equations (ODE's). [Online]. Available: http://www.physics.utoronto.ca/~poptor/Com00/ODEGuide.pdf

[4] B. R. Mutlu, U. Yaman, M. Dolen, and A. B. Koku, "Performance evaluation of different real-time motion controller topologies implemented on a FPGA," in *Proc. International Conference on Electrical Machines and Systems*, 2009, pp. 1-6, 15-18, Nov 2009.

[5] Y. F. Chan, M. Moallem, and W. Wang, "Efficient implementation of PID control algorithm using FPGA technology," in *Proc. of 43rd IEEE Conference on Decision and Control*, vol. 5, pp. 4885-4890, Dec 2004.

[6] Y. D. Tao, H. Lin, Y. Hu, X. H. Zhang, and Z. C. Wang, "Efficient implementation of CNC Position Controller using FPGA," in *Proc. of 6th IEEE International Conference on Industrial Informatics (INDIN 2008)*, pp.1177-1182, 13-16, July 2008.

[7] A. Stoica, X. Guo, R. S. Zebulum, D. Keymeulen, and M. I. Ferguson. On-chip evolutionary synthesis of reconfigurable analog computing circuits. [Online]. Available: http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/6434/1/030281.pdf

[8] D. Phanthavong, "Designing with DSP48 Blocks Using Precision Synthesis," Mentor Graphics Corporation, 2005.

[9] G. S. Hyalij, A. U. Deshpande, P. D. Shendge, B. M. Patre, "Real Time Implementation of Time Delay Controller for DC Motor Speed Control," *International Journal of Recent Trends in Engineering*, vol. 1, no. 3, May 2009.

[10] V. Subasri, K. Lavanya, and B. Umamaheswari, "Implementation of Digital PID controller in Field Programmable Gate Array (FPGA)," in *Proc. International Conference on Power Electronics*, 2006

Ravindra S. Rana belongs to Mehsana, Gujrat (India). He graduated in Bachelors of Engineering, Instrumentation and Control discipline from Gujrat University in 2010 and then completed his Masters in Process Instrumentation from College of Engineering Pune, India in 2012.

As a student he has worked in digital control systems. His research interest includes design of advanced controllers and Observer-controller combinations for uncertain systems.

Bharti Kumari belongs to Mumbai (India). She graduated in Bachelors of Engineering, Instrumentation and Control discipline from Pune University in 2010 and then completed her Masters in Process Instrumentation from College of Engineering Pune, India in 2012.

As a student she has worked on fuel cell for Master's Thesis from her college, in association with NCL Pune. Her research interests include fuel cell and its control.