JOURNAL OF AIRCRAFT
Vol. 32, No. 1, January–February 1995

Multiple-Function Digital Controller System for Active Flexible Wing Wind-Tunnel Model

Sherwood T. Hoadley*
NASA Langley Research Center, Hampton, Virginia 23681
and
Sandra M. McGraw†
Lockheed Engineering and Sciences Company, Inc., Hampton, Virginia 23666

A real-time multiple-function digital controller system was developed for the Active Flexible Wing program, which demonstrated through wind-tunnel tests that digital control can be used with great versatility to perform a multifunction task such as suppressing flutter and reducing loads during rolling maneuvers. The digital controller system (DCS) allowed simultaneous execution of two control laws: 1) flutter suppression and 2) either roll trim or a rolling maneuver load control. The DCS operated within, but independently of, a slower host operating system environment, at regulated speeds up to 200 Hz. It also coordinated the acquisition, storage, and transfer of data for near real-time controller performance evaluation and both open- and closed-loop plant estimation. It synchronized the operation of four different processing units, allowing flexibility in the number, form, functionality, and order of control laws, and variability in selection of sensors and actuators employed. Most importantly, the DCS enabled successful demonstration of active flutter suppression to conditions approximately 26% (in dynamic pressure) above the open-loop boundary in cases when the model was fixed-in-roll, and up to 23% when it was free-to-roll. Aggressive roll maneuvers with load control were achieved above the flutter boundary. The purpose of this article is to present the development, validation, and wind-tunnel testing of this multiple-function digital controller system.

I. Introduction

THE Active Flexible Wing (AFW) program[1,2] was a cooperative effort between the NASA Langley Research Center and the Rockwell International Corporation. One of the specific program objectives was the validation of analysis and synthesis methodologies as applied to the multiple-function control of a sophisticated full-span, free-to-roll, aeroelastically scaled wind-tunnel model.

The control functions being investigated included the suppression of flutter, roll trim control, rolling maneuvers with load control, and load alleviation. To meet the multiple-function control requirements, depicted in Fig. 1, it was necessary to engage the flutter suppression system (FSS) with any of three different roll control systems: roll-trim system (RTS), roll rate tracking system (RRTS), or rolling maneuver load alleviation system (RMLA).

Meeting the primary objectives of the AFW wind-tunnel program required gaining practical experience in designing, fabricating, and implementing a real-time multiple-function multi-input/multioutput (MIMO) digital controller and interfacing hardware. As a result, a versatile digital controller system was developed that operated at 200 Hz within a slower operating system environment allowing for simultaneous multiple function control.

The purpose of this article is to present the development, validation, and the wind-tunnel testing of the AFW digital controller system (DCS). The generic forms of the control laws that can be accommodated by the DCS are described. Techniques employed to verify control laws and procedures used to validate the DCS are also discussed.

II. Digital Controller System Design

Most modern computers operate within a time-share operating system, capable of performing many tasks that share the central processing unit (CPU). These operating systems are not designed to enable one task to operate at regulated

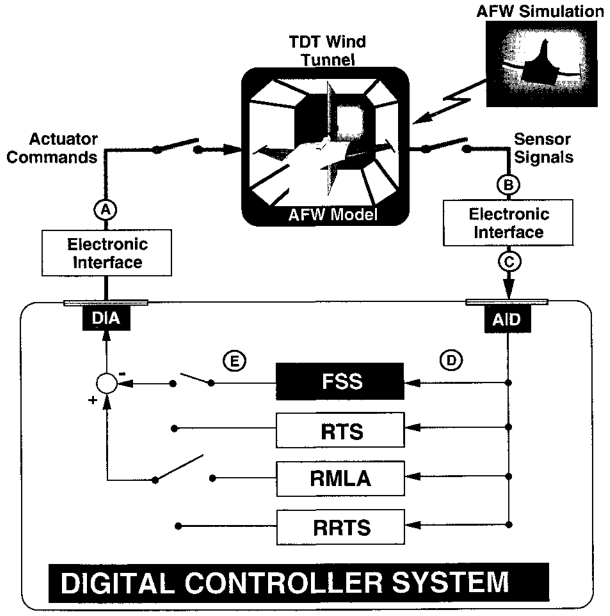


Fig. 1 Basic multiple-function control and interface electronics of the AFW DCS.

Presented as Paper 92-2023 at the Dynamic Specialists Conference, Dallas, TX, April 16–17, 1992; received May 7, 1992; revision received March 7, 1994; accepted for publication March 21, 1994.

*Senior Aerospace Engineer, Aeroelasticity Branch, M/S 340. Associate Fellow AIAA.

†Electrical Engineer; currently at Philips Consumer Electronics, 1 Philips Dr., Knoxville, TN 37914.

frequencies in a real-time fashion, oblivious to other tasks being performed. Consequently, the DCS was designed using a separate dedicated processor as the real-time system executor to perform real-time control functions independent of the host CPU. This real-time system executor had to be capable of controlling data transfer over the data BUS and to start and stop processes. An integer digital signal processor with an internal clock that could be used for regulating speeds was selected to perform this task. However, an integer processor does not lend itself to changing control law parameters quickly because scaling to avoid overflow and underflow for each control law must be included in the code. The design requirement that control laws be easily modified or changed was a driving constraint. It forced the use of separate floating point processors to execute the various control laws. Hence, a dedicated integer processor was used as executor of the real-time system and dedicated floating point processors were used to execute individual control laws.

A SUN 3/160 Workstation, driven by a UNIX Operating System, was selected as the "shell" of the DCS. The DCS had three special purpose processing units linked via a data BUS, which included an integer digital signal processor (DSP), a floating point DSP board with two microprocessors, and an array processor (AP).

Unlike its analog counterpart comprised primarily of fixed hardware circuitry, the DCS was designed to allow flexibility in the number, functionality, and form of control laws to be implemented; in the selection of sensors and actuators employed; in the number of states in the state-space representations; and in the size and number of tables used for control laws using table-lookup and interpolation. The DCS was also designed to coordinate data acquisition, storage, and transfer.

Components of the DCS and interfacing hardware either to the real-time simulation of the wind-tunnel model or to the wind-tunnel model itself are discussed. Figure 1 indicates the placement of these various components. The DCS chassis housed the host CPU, the disk and tape drives, and the added boards that all communicated across a VME data BUS. The host CPU and the status display panel provided user interface to the real-time system. A SKY Computers, Inc. Challenger-1 (C1) integer DSP board controlled the real-time processing. Most control law computations were performed on a SKY Challenger-C30/V (C30). This was a high-speed, floating point, 32-bit systems oriented digital signal two-processor board. The AP was another SKY board that provided high-speed direct memory access for the DCS and vectorized floating point processing. In case the C30 failed, the AP board performed floating-point calculations for the FSS and RMLA control laws as a backup.[3] Backup for the RTS was performed by the C1 using integer arithmetic. There was no backup for the RRTS system. Two analog-to-digital (A/D) and two digital-to-analog (D/A) converter boards, manufactured by Data Translation, Inc., provided the link between the digital controller and the analog hardware, providing all of the analog/digital data conversions required between the model and the DCS. They converted the incoming analog voltages from the sensors to 12-bit digital values and 12-bit digital signals such as the control surface actuator commands into outgoing analog voltages. The entire real-time operation from sensor input to actuator command output was repeated at regulated speeds up to a maximum requirement of 200 times each second.

The interface electronics hardware rack contained the analog circuitry for processing the analog signals coming from or going to either the wind-tunnel model or the simulator. The filter box housed analog antialiasing filters, analog notch filters, and electrical isolation networks. The analog antialiasing filters were configured to provide either first-order rolloff or fourth-order rolloff with either a 25- or 100-Hz break frequency. The sensor signals coming to the DCS or the commands going to the model could also be filtered through analog notch filters, if desired, to filter out undesired frequency ranges. The patch box allowed direct input/output of analog signals by the DCS without additional filtering.

The status display panel, designed and built in-house by NASA, displayed, through status lights, the real-time status of various control parameters such as the feedback switch. It also displayed the system pulse.

A second SUN Workstation, configured similarly to the DCS, was used not only as a backup for the DCS, but also as a near real-time multisignal digital analyzer to evaluate controller performance and estimate both the open- and closed-loop plants.[4–6] It was linked to the DCS via an Ethernet line, allowing for fast data transfer of blocks of sampled data.

A separate digital signal analyzer, was also used to verify analog signals and to debug problems associated with the DCS plus interface electronics.

III. Functionality of Software Components

As functions of the DCS were identified, separate program components were developed that performed the various functions. These components are described below. Except for the commands required to perform the actual calculations on the AP, all of the software was written in a high level (C) programming language. Operation code command blocks were generated for the AP.

Three user interface programs for the DCS were executed on the host computer. They are the user/controller interface program; the data transfer program; and the information display program. The user/controller interface program provided the DCS operator with communication links to the real-time system. All user options, control law arrays, control parameters, and excitation definitions were specified through this program and downloaded into C1, C30, or AP memory as required. The array-processor command blocks were also determined and downloaded by this program into C1 memory. Through this program, the DCS operator was able to perform such tasks as selecting the mode of operation, changing gains, selecting control laws, opening and closing control law loops, selecting excitations, changing excitation amplitudes, and selecting the control surfaces to be excited. The data transfer program controlled the formatting and sending of the digitized data to external files or tapes for control law verification, performance, and controller performance evaluation (CPE).[4–6] The information display program displayed all pertinent DCS information such as control-surface deflections, errors between commanded and actual deflections, and switch settings verifying operator selections.

The real-time system was controlled by a real-time executor program which resided on the C1 board. As BUS master, this real-time executor provided management of all real-time activities and tasks. It controlled all of the real-time processing, digitizing of analog input signals, converting of output signals to analog voltages, starting control-law execution, summing of digitized excitations with bias commands to statically position, and excite control surfaces. The real-time executor also provided the interface to the status display panel lights, checked for faults, and instructed the host computer when blocks of data were stored and could be transferred.

Primary control law processing was performed by programs residing on the C30 board, referred to herein, as the control law processor. This board was comprised of shared global memory and two complete processing nodes, referred to herein as nodes 1 and 2. Each node had its own memory in which arrays defining the control laws and control law execution code were stored. The globally shared memory could be accessed by the real-time executor, the user/controller interface, and by both node processors without interrupting node processing. Two programs, each residing on separate nodes, performed the different control law calculations as follows:

Node 1:
- RTS control law transfer function
- RRTS table look-up and interpolation
- RMLA state-space matrix computations

Node 2:
- FSS state-space matrix computations.

Computations included all unit conversions, scaling, matrix computations, and interpolations. A bivariate table look-up capability with linear interpolation was developed for the RRTS with a fixed number of inputs and outputs, but variably sized tables so that interpolations could be improved by enlarging the number of elements in the tables.

Vectorized floating-point computations used by the backup system were performed on the AP, referred to herein as the backup processor. Because this was a single-processor, however, the simultaneous control law calculations needed for multiple-function control could not be performed by the backup system within the time constraints imposed by a 200-Hz sampling rate.

IV. Generic Forms of Control Laws

A generic form of the control laws for FSS and RMLA was identified such that one set of software would accommodate both types of control laws while imposing minimal implementation constraints on the control law designers. The generic structure provided the designers a choice of sensors with the option to blend them, freedom of control law order with upper limits, scheduling of control law parameters with respect to dynamic pressure, and the selection of various control surfaces with or without distribution of control law outputs to different actuators.

The FSS and RMLA control laws were implemented using the following difference equations representing a linear time-invariant state-space controller:

$$(\boldsymbol{x}_{k+1}) = F(\boldsymbol{x}_k) + G(\boldsymbol{y}'_k) \tag{1}$$

$$(\boldsymbol{u}_{k+1}) = H(\boldsymbol{x}_{k+1}) + E(\boldsymbol{y}'_{k+1}) = H'(\boldsymbol{x}_k) + E'(\boldsymbol{y}'_k) \tag{2}$$

$$(\boldsymbol{\delta}_{k+1}) = D(\boldsymbol{u}_{k+1}) \tag{3}$$

where

$$(\boldsymbol{y}'_k) = B(\boldsymbol{y}_k) \tag{4}$$

$$H' = HF \tag{5}$$

$$E'(\boldsymbol{y}'_k) = HG(\boldsymbol{y}'_k) + E(\boldsymbol{y}'_{k+1}) \tag{6}$$

If $E = 0$ or $(\boldsymbol{y}'_k) \approx (\boldsymbol{y}'_{k+1})$, then $E'(\boldsymbol{y}'_k) \approx (HG + E)(\boldsymbol{y}'_k)$, and these equations could be implemented with a one time-step delay between input and output instead of the standard two time-step delay. The DCS implemented the equations assuming a one time-step delay. The vectors $(\boldsymbol{\delta}_{k+1})$ and $(\boldsymbol{y}'_k)$ in Eqs. (3) and (4), respectively, are subsets, specified by the control law designer, of the analog sensor input signals and actuator command signals.

RMLA and FSS designers provided the control law quadruples

$$(F, G, H', E')$$

sensor blending matrix B, and the control law output distribution matrix D, if desired, and a list of desired sensor inputs and actuator outputs for each control law. There was no specified limitation for the number of states in the controller, nor the number of inputs and outputs. Execution time provided the only real constraint to the overall system, namely, all control laws, data acquisition and storage, and user/operator interface handling had to be performed within a 5 ms time frame for each time step.

FSS designers provided these for both symmetric and antisymmetric control laws, separately. These were then combined into a single system defined by the following equations:

$$F = \begin{bmatrix} F_S & 0 \\ 0 & F_A \end{bmatrix} \quad \text{and} \quad G = \begin{bmatrix} G_S & 0 \\ 0 & G_A \end{bmatrix} \tag{7}$$

$$H' = \begin{bmatrix} H'_S & 0 \\ 0 & H'_A \end{bmatrix} \quad \text{and} \quad E' = \begin{bmatrix} E'_S & 0 \\ 0 & E'_A \end{bmatrix} \tag{8}$$

$$B = \begin{bmatrix} B_S & 0 \\ 0 & B_A \end{bmatrix} \tag{9}$$

$$D = \begin{bmatrix} D_S & 0 \\ 0 & D_A \end{bmatrix} \tag{10}$$

Actual control law design and descriptions of the FSS and RMLA control laws are discussed in Refs. 7–12.

The RRTS system was implemented using look-up tables for each of six control surface commands. The tables were functions of two variables, and the actual commands were determined from bilinear interpolation, performed by the C30 processor, of each of these tables. Reference 13 provides detailed description of how each of these tables were determined.

V. Digital Controller System Modes of Operation

There were seven basic modes of operation defined for the DCS, labeled MAINTENANCE, MANUAL, STATIC, RTS, FSS, RMLA, and RRTS. In all modes of operation, the conversion of the 12-bit signals from the DT boards to 16-bit integers was performed by the real-time executor using shift operations. In the first two modes, no data was stored or saved, and in the first three modes, no control laws were executed. In the last four modes, the averaging of the signals for the FSS control law was performed by the real-time executor using binary shift operations. Signal data to be saved were sent to a block of AP memory for temporary storage.

In each mode of operation, there was a block of "slow-cycle" code performing 10 different secondary communication tasks, each executed once in every 10 iterations. Included in this was code to read switch settings downloaded from the host user/controller interface program and code to send DCS parameters to the information display program. Types of parameters sent included 1) selected mode of operation; 2) desired sampling frequency; 3) selected control law; 4) selected control law scheduling option; 5) selected option of feedback loop, opened or closed; 6) selected excitation and symmetry; 7) selected point for adding excitation to the system (actuator commands, control law outputs, or sensor inputs); and 8) tunnel parameters. The primary functions of each mode of operation are described below.

The primary function of the MAINTENANCE mode was the checkout of all analog-digital links and hardware. This mode allowed for the individual checking of each input and output signal line and allowed the most basic hardware debugging, without interference from code designed for control law execution or data saving.

The primary function of the MANUAL mode was static positioning of control surfaces and checking of scale factors for each signal with a minimal amount of code involved. This, too, was primarily a debugging mode.

The primary function of the STATIC mode was the static positioning and/or excitation of control surfaces while saving data from the different sensors. Excitation signals could be sent individually or to pairs of control surfaces, either symmetrically or antisymmetrically. This mode was designed primarily for obtaining data about the model to help develop improved plant models.

The next four modes of operation involved execution of various control laws separately or simultaneously. One roll control law could be operated simultaneously with flutter suppression (both switch selectable) in all four modes. The basic differences between the four modes were defined by which control law was dominant. The specific signals that were sampled and saved, the types of commands, such as a roll-trim, that could be executed, the locations in the control loop at which excitations could be added varied with each mode. For instance, in the FSS mode, the FSS control law was dominant, and the signals that were saved in this mode were related directly to FSS control law execution, verification, and performance such as the FSS control law inputs and outputs. Figure 4 of Ref. 14 provides a detailed schematic of the blocks of code and signal flow involved in the execution of these last four modes. If the backup system was employed to calculate control law outputs, commands blocks to execute on the AP were sent by the real-time executor to the AP during each execution time cycle.

The actuator commands for performing roll trim δ_{TRIM}, or a rolling maneuver δ_{RRTS} or δ_{RMLA}, were combined with static positioning or bias commands δ_{BIAS}, and then limited to a maximum deflection of ± 10 deg. Actuator excitations (if any) and FSS actuator commands δ_{FSS} were then combined with the deflection-limited commands to form the final actuator commands. These were then converted to analog signals to be sent to the model.

Each control system feedback was individually switch selectable; i.e., each control law loop could be individually opened or closed, but no two roll control laws could be operated simultaneously. Bias commands, roll-trim commands, and the roll-rate commands were implemented using a ramping procedure that ramped in (or ramped out) the command rather than introducing a command as an instantaneous step.

In both RMLA and RRTS modes of operation, the RTS was used to hold the model at an initial roll angle until the roll-rate command was invoked. Both the RMLA and RRTS modes of operation were coded in such a way that an RMLA or RRTS control law was invoked simultaneously with a specified roll-rate command when the RMLA or RRTS control law was enabled by the operator. Once the desired termination angle was passed, the roll-rate command was ramped out. When the roll-rate was below a specified "capture rate," the RTS was then reinvoked automatically to trim the model to the current roll angle of the model at the time the capture rate was achieved. The roll-rate command implemented by the DCS is shown in Fig. 2. The initial roll-trim angle and termination angle, after which the roll-rate command was ramped back to 0, and the on- and off-amp rates for each command were specified at the time of each maneuver.

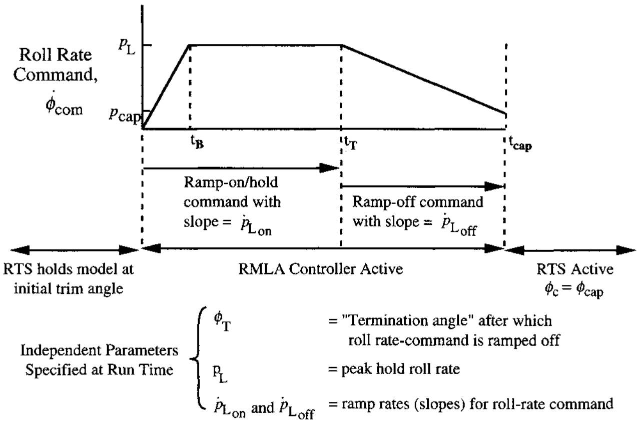
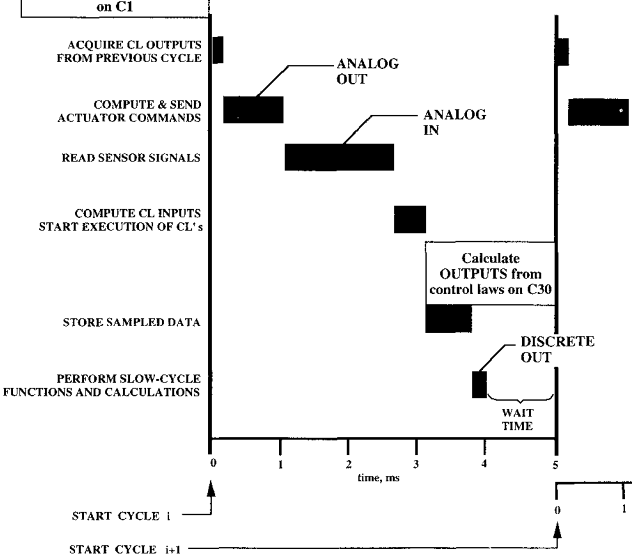


Fig. 2 Ramp-on/hold/ramp-off roll-rate command for RRTS and RMLA control systems.

Developing an RTS system with good performance using both the floating point DSP (C30) and the integer DSP (C1) for the backup system was a lengthy process performed using hot bench simulation (HBS).[14–16] A number of iterations on both the control law form and gains were made to achieve this goal. A number of adjustments to codes were also made to improve the execution speed on the DCS in order to accommodate larger control systems while acquiring data for on-line analyses.[4–6] Adjustments were made to the RRTS system to accommodate larger tables of data for improved performance; however, other than using a floating point DSP for control law execution, no iterations on the basic design of the DCS were made to improve controller performance for RMLA and FSS.

VI. Timing Requirements

Each of the different tasks performed by the real-time executor required a varying amount of time depending on the mode and options selected, the signals employed, and the size of each control law being executed. The timing requirements for each real-time execution cycle in the DCS, during AFW wind-tunnel testing, which involved the processing of a control law are shown in Fig. 3. The times shown here are for a total execution time-step of 5 ms, the time required for the sampling frequency of 200 Hz. Referring to the figure, times are approximate for all control law execution modes. It took approximately 0.15 ms to acquire all the control-law (CL) outputs from the control-law processor generated in the preceding cycle, and 0.8 ms to combine them and send them to the DA converters. Approximately 1.7 ms were used to process the incoming signals, about 0.5 ms to send the signals to the control-law processor and start execution, and 0.7 ms to store sampled data in direct access storage. There were various functions, such as sending out discrete signals or reading operator instructions, which did not need to be performed at 200 Hz. These were grouped into 10 groups and performed at a slower rate (one-tenth of the 200-Hz rate). At most, 0.2 ms were used for each of these slow-cycle functions. At the end of 5.0 ms, the next cycle would start. Execution of the control laws had to be performed during the time the control-law processor was started and the end of the cycle. To insure completion, the C30 set a flag to indicate execution was finished. The C1 would wait, if necessary, for this flag before acquiring the CL outputs and starting the next cycle. For all the wind-tunnel tests performed with the AFW using the DCS



EXECUTOR Processing on C1
ACQUIRE CL OUTPUTS FROM PREVIOUS CYCLE
COMPUTE & SEND ACTUATOR COMMANDS
READ SENSOR SIGNALS
COMPUTE CL INPUTS START EXECUTION OF CL's
STORE SAMPLED DATA
PERFORM SLOW-CYCLE FUNCTIONS AND CALCULATIONS
ANALOG OUT
ANALOG IN
Calculate OUTPUTS from control laws on C30
DISCRETE OUT
WAIT TIME
0
1
2
3
4
5
time, ms
0
1
START CYCLE i
START CYCLE i+1

Fig. 3 Timing requirements of the real-time execution of the DCS.

as described herein, all executions, including those executed by the C30, were completed within one time cycle.

In the primary system, even the slowest FSS control law could be executed with any roll control law at the required 200-Hz frequency. Many of the options available in the primary system, however, had to be eliminated or reduced in scope when employing the backup system in order to operate at 200 Hz.[3]

VII. Validation of Digital Controller System

Validation of the DCS was performed in various stages: during system development using hot bench simulation, during end-to-end tests in the calibration lab, and in the wind tunnel. These were performed for both the primary and backup systems of the DCS. Figure 1 identifies the basic locations of signals at which verification was performed. In each case, the first step was to verify the correct values of all input and output signals against known values at locations A, B, and C in Fig. 1. This was done first without, and then with, the interface electronics (analog filters) in the loop. An oscilloscope was used to check each output signal at A. Once output signals were verified, a fixed voltage was hooked to each input, separately, at location C and then at B. As this was done, each input value was checked and verified.

Once signals were verified, initial validation of implemented control laws was performed. To do this, open-loop (no plant) control law time responses (at E) due to some excitation into one of the control law inputs (at D) were generated. For FSS and RMLA control laws, the input was usually a unit step. In the case of accelerometer inputs this equaled a 1-*g* step excitation. For the RRTS control law, one cycle of a sine wave, whose range encompassed the input range of the control law, was used. This was done for each combination of control/sensor pairs.

Once the time responses compared favorably to corresponding plots provided by the control law designers, FSS and RMLA control laws were further validated by generating control law transfer functions for each control/sensor pair. These were generated by inputting an excitation with a specified frequency range content into each control law input, performing FFT calculations, and generating corresponding transfer functions. These had to compare identically to transfer functions generated analytically by the control law designers, or differences, such as those due to the antialiasing filters, had to be accounted for by the control law designer. These transfer functions were generated between various points without a model in the loop: E/D, E/C, E/B, and A/B. These last transfer functions for A/B were obtained without the model in the loop (loop open) but with the software feedback switch closed, using a digital signal analyzer which could be hooked directly to the analog signals to debug problems associated with the DCS plus interface electronics. During HBS validation, transfer functions for E/D with the simulation model in the loop were extracted from the closed-loop system. The next stage was to repeat these verification procedures during end-to-end testing. Final verification was performed in the wind-tunnel with the model in the loop.

Comparisons of some of these transfer functions for a control/sensor pair used for one of the FSS control laws are shown in Figs. 4–7. The analytically generated transfer function in this case, shown by the dashed line, was for a 200-Hz digitized control law with a 1-time step delay built in. The output for all of these transfer functions is at the control law output point E, and does not include the negative sign for negative feedback. As should be expected in Figs. 4 and 5, the two curves are coincident. Figure 4 shows the initial open-loop, control law only transfer function (E/D), in which the excitation is added internally at the control law input point, D; Fig. 5 shows the same transfer function after being extracted from the closed-loop HBS data; and Fig. 6 shows the open-loop DCS + analog filters (E/B) in which the excitation is added at the sensor input point B, and includes the analog antialiasing filters. Phase differences due to the analog antialiasing filters are seen in this figure, but do not show up in Figs. 4 and 5 because the filters in those cases are seen as part of the plant. Any differences in phase and gain between the control law transfer functions generated analytically and experimentally were accounted for by the control law designer and approved by the test engineer before testing the control law in the wind tunnel. Figure 7 shows the control law transfer function (E/D) ex-

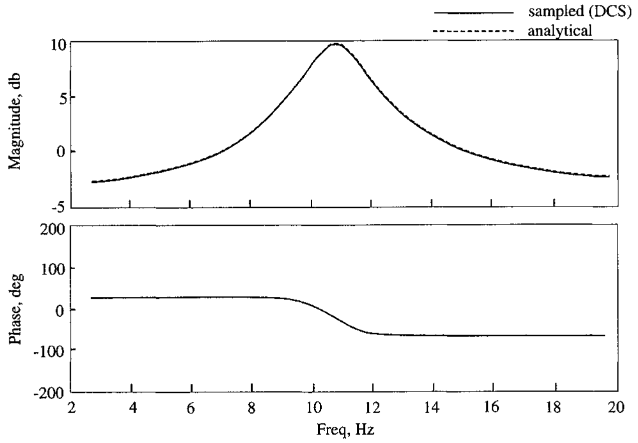


Fig. 4 Comparison of digital controller only FSS control law transfer function (E/D) with analytically generated one for validation of digital controller.

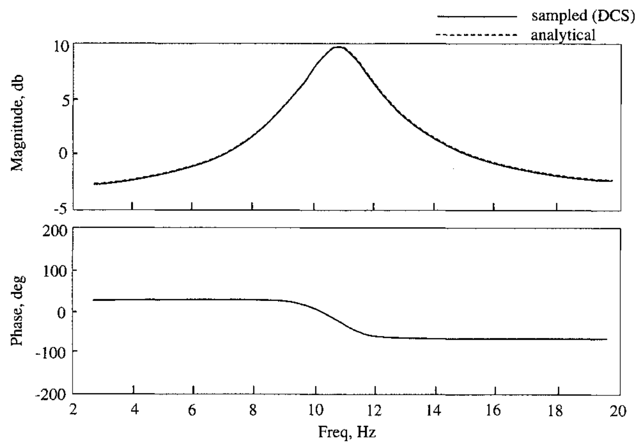


Fig. 5 Comparison of FSS control law transfer function (E/D), extracted from HBS closed-loop system, with analytically generated one for validation of digital controller.

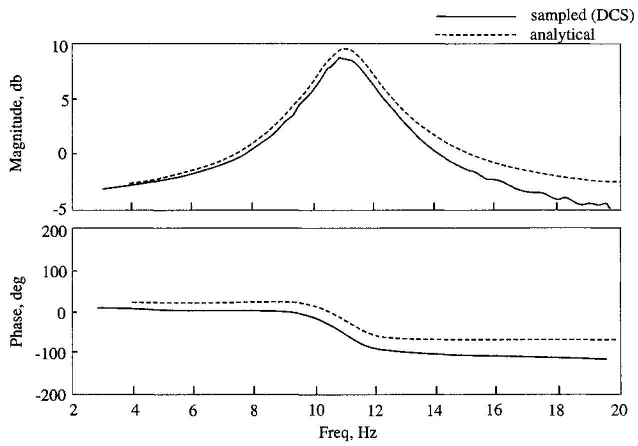
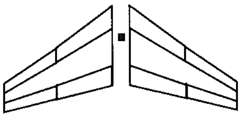
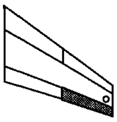
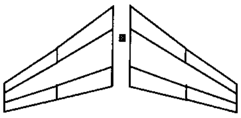
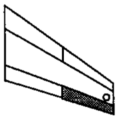
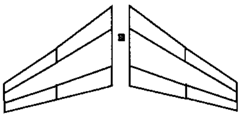
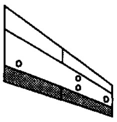
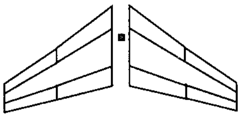
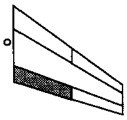


Fig. 6 Comparison of FSS control law transfer function (E/B), including antialiasing filters, with analytically generated one for validation of digital controller.

Table 1 Multiple-function MIMO testing accomplished with AFW digital controller

	Roll control		Flutter suppression		Maximum test condition, %q above open-loop flutter	
Multiple functions	Controls/ sensors	Control law	Controls/ sensors	Control law	Cruise	Roll maneuver
RMLA + FSS 1		Ninth-order state-space		Fifth-order state-space	23%	11%
RRTS + FSS 2		Six degrees-of-freedom bilinear table lookup		Third-order state-space	23%	17%
RRTS + FSS 3		Six degrees-of-freedom bilinear table lookup		Eleventh-order state-space	23%	11%
RRTS + FSS 4		Six degrees-of-freedom bilinear table lookup		Third-order state-space	23%	11%

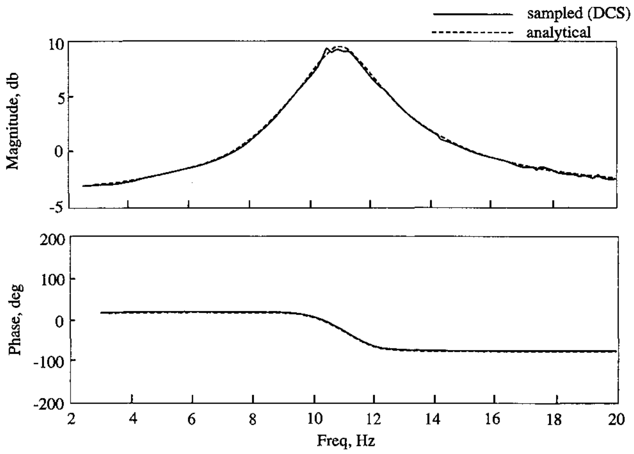


Fig. 7 Comparison of FSS control law transfer function (E/D), extracted from closed-loop wind-tunnel system, with analytically generated one for validation of digital controller.

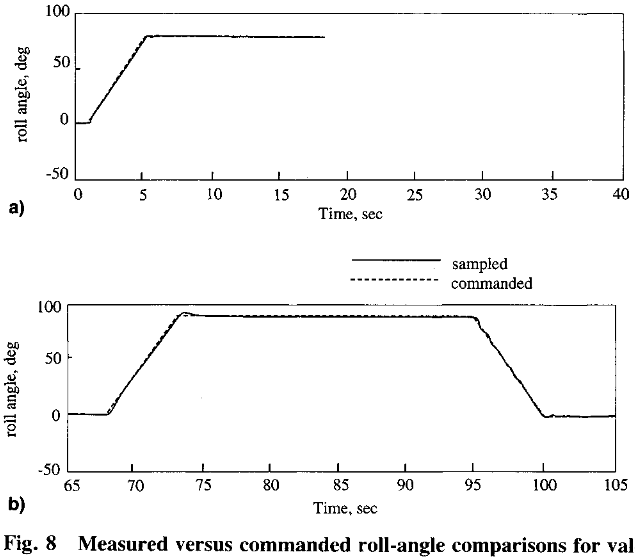


Fig. 8 Measured versus commanded roll-angle comparisons for validation of digital controller roll trim system: a) 0–80-deg roll trim command during HBS and b) 0–90-deg followed by 90–0-deg roll trim command in wind tunnel at 205 psf.

tracted from the closed-loop system at a dynamic pressure of 260 psf, 11% above the open-loop flutter. Although noisy, because signal-to-noise ratios were low, it verifies that the control law was operating as prescribed, thus further validating the FSS controller during wind-tunnel testing.

The RTS was verified by extensive use with HBS. Figure 8a shows a comparison plot of the measured simulation roll angle and the commanded roll angle generated during an HBS run. Plots such as these were generated with the RTS loop closed. As can be seen, the model roll angle followed the commanded roll angle extremely well with little overshoot. Figure 8b shows a roll-trim maneuver in the wind tunnel at a dynamic pressure of 205 psf, verifying RTS performance in the wind tunnel. Plots such as this were generated for various roll commands to verify the RTS prior to any testing of the RRTS, RMLA, or simultaneous FSS and roll control, since the RTS was used in all these modes of operation.

The RRTS and RMLA systems were further verified by generating closed-loop HBS plots of roll-rate commands and actual measured roll-rate. Sensor and control law output data were saved and plotted for various commands. These data were provided to the control law designers for verification.

VIII. Controller Performance

The final "validation" of the DCS was its demonstrated use in the wind tunnel. Table 1 depicts the different combinations and complexity of control laws that were tested in the tunnel. Roll trim was achieved simultaneously with flutter suppression up to 23% above the open-loop boundary. Rolling maneuvers with load control were performed up to 17% above the open-loop boundary. All tests were performed while saving data for on-line analysis within a total combined operating time of less than 5 ms, allowing the DCS to operate at the required 200-Hz sampling frequency. Once in the tunnel, no coding changes were made to the DCS to improve controller performance because it was considered too risky to change code without the opportunity to test it with HBS.

The backup FSS system had been validated during 1989 tests of the AFW, and all FSS control laws were verified in the calibration lab using the backup system. However, the RMLA and RTS backup systems were only validated with the HBS. They were never actually validated in wind-tunnel testing.

Although considered beyond the scope of this article, detailed descriptions and controller performance evaluation of each control law tested are presented in the references. References 7–10 describe and evaluate the various FSS control laws that were tested; Refs. 11 and 12 describe and evaluate the RMLA control law; and Ref. 13 describes and evaluates the RRTS system.

IX. Concluding Remarks

A versatile digital controller system that operated at 200 Hz within a 60-Hz host operating system environment was developed to actively control an aeroelastic wind-tunnel model. It allowed for simultaneous execution of two control laws; data acquisition, storage, and transfer; flexibility in the form and order [number of degrees of freedom (DOF)] of control laws implemented; and flexibility in the number and selection of sensors and actuators employed. The system design coordinated and synchronized the operation of four different computing units: 1) a host SUN 3/160 central processing unit, 2) an integer digital signal processor, 3) a floating point digital signal multiprocessor, and 4) an array processor.

Most importantly, the DCS enabled the successful demonstration of active flutter suppression while performing aggressive roll maneuvers up to 17% (in dynamic pressure) above the flutter boundary. It enabled the successful demonstration of active flutter suppression to a point 23% above the open-loop boundary when the model was free-to-roll. By the simultaneous execution of both symmetric and antisymmetric flutter suppression control laws, it also enabled the successful demonstration of active flutter suppression up to 26% above the open-loop boundary when it was fixed-in-roll. While executing control laws, the digital controller system acquired data for near real-time controller performance evaluation as well as open- and closed-loop plant estimation.

References

[1] Perry, B., III, Cole, S., and Miller, G., "A Summary of the Active Flexible Wing Program," AIAA Paper 92-2080, April 1992.

[2] Perry, B., III, Mukhopadhyay, V., Hoadley, S. T., Cole, S. R., Buttrill, C. S., and Houck, J. A., "Digital-Flutter-Suppression-System Investigations for the Active Flexible Wing Wind-Tunnel Model," AIAA Paper 90-1074, April 1990; also NASA TM-102618, March 1990.

[3] Hoadley, S. H., Buttrill, C. S., McGraw, S. M., and Houck, J. A., "Development, Simulation Validation, and Wind-Tunnel Testing of a Digital Controller System for Flutter Suppression," *Proceedings of the 4th Workshop on Computational Control of Flexible Aerospace Systems* (Williamsburg, VA), 1991, pp. 583–613 (NASA 10065).

[4] Pototzky, A. S., Wieseman, C. D., Hoadley, S. T., and Mukhopadhyay, V., "Development and Testing of Methodology for Evaluating the Performance of Multi-Input/Multi-Output Digital Control Systems," AIAA Paper 90-3501, Aug. 1990.

[5] Pototzky, A. S., Wieseman, C. D., Hoadley, S. T., and Mukhopadhyay, V., "On-Line Performance Evaluation of Multiloop Digital Control Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 878–884.

[6] Wieseman, C. D., Hoadley, S. T., and McGraw, S. M., "On-Line Analysis Capabilities Developed to Support the AFW Wind-Tunnel Tests Wing Program," AIAA Paper 92-2084, April 1992.

[7] Mukhopadhyay, V., "Flutter Suppression Digital Control Law Design and Testing for the AFW Wind-Tunnel Model," AIAA Paper 92-2095, April 1992.

[8] Christhilf, D. M., and Adams, W. M., Jr., "Multifunction Tests of a Frequency Domain Based Flutter Suppression System," AIAA Paper 92-2096, April 1992.

[9] Waszak, M. R., and Srinathkumar, S., "Flutter Suppression for the Active Flexible Wing: Control System Design and Experimental Validation," AIAA Paper 92-2097, April 1992.

[10] Klepl, M. J., "A Flutter Suppression System Using Strain Gages Applied to Active Flexible Wing Technology: Design and Test," AIAA Paper 92-2097, April 1992.

[11] Woods-Vedeler, J. A., and Pototzky, A. S., "Rolling Maneuver Load Alleviation Using Active Controls," AIAA Paper 92-2099, April 1992.

[12] Woods-Vedeler, J. A., Pototzky, A. S., and Hoadley, S. T., "Active Load Control During Rolling Maneuvers," NASA TP-3455, Oct. 1994.

[13] Moore, D. B., "Maneuver Load Control Using Optimized Feedforward Commands," AIAA Paper 92-2100, April 1992.

[14] Hoadley, S. H., and McGraw, S. M., "The Multiple-Function Multi-Input/Multi-Output Digital Controller System for the AFW Wind-Tunnel Model," AIAA Paper 92-2083, April 1992.

[15] Buttrill, C. S., Bacon, B., Heeg, J., Houck, J. A., and Wood, D., "Simulation and Model Reduction for the AFW Program," AIAA Paper 92-2081, April 1992.

[16] Buttrill, C., and Houck, J., "Hot Bench Simulation of the Active Flexible Wing Wind-Tunnel Model," AIAA Paper 90-3121, July 1990.