Computer-Aided Digital Autopilot Design & Analysis: Methodology, Implementation and Verification

W. V. Albanes, Computer Sciences Corporation
J. B. Meadows, U.S. Army MICOM
Redstone Arsenal, Alabama

ABSTRACT

This paper details the design methodology for a missile digital autopilot using a digitization approach, and a discrete domain design approach. These two designs rely heavily on computerized system analysis tools in the frequency and time domains. Further, three complex frequency planes are available to the designer, therefore, relative merits of each will be discussed.

This paper will also detail the implementation of the autopilot on the missile microcomputer, a six degree of freedom (6DOF) missile simulation, and a hardware in the loop (HWIL) hybrid computer simulation. The 6DOF permits fine-tuning of design parameters for optimum system performance, whereas the HWIL simulation exercises the real flight hardware.

Finally, this paper will also show the final autopilot design verification through comprehensive deterministic and Monte Carlo simulations.

INTRODUCTION

The use of simulation has been shown to reduce the overall cost and startup time of military and civilian projects due to the early correction of design problems. Besides this verification and validation aspect, simulation can be used as part of a computer-aided autopilot design, along with other frequency and time domain analysis techniques. Hardware in the loop autopilot testing and systems simulation further verifies the design, and checks out the actual flight hardware. This redundancy in self checking and verification enhances the designer's confidence in his product and essentially guarantees a successful program.

MICOM's Modular Guidance program objective is to develop control technology using microprocessors for anti-armor missiles. A small but capable digital autopilot has been designed and developed in order to prove the technology. The autopilot is being prepared for flight.

SYSTEM DEFINITION

The T6 airframe is a modular test vehicle designed to accept a variety of seekers and autopilots. Its six-inch diameter 70-pound body yields performance comparable with that expected in future weapon systems. The missile can be either ground or helicopter launched. A typical direct fire (lock-on to target prior to launch) scenario is depicted in Figure 1. The missile consists of four major subsystems. These are (1) laser seeker subsystem (optics/detector and gimbal assemblies, signal processing and controlling electronics, power supply), (2) telemetry subsystem, (3) flight control subsystem (gyros, controller element, actuators), and (4) propulsion subsystem (engine, nozzle, igniter).

The digital autopilot (DAP) performs the classical flight control autopilot functions of attitude stabilization and gyro filtering, and, in addition, provides actuator control, seeker signal filtering, and guidance. The DAP controls four fins so as to successfully reach the desired state, or impact the designated target. The laser seeker generates pitch/yaw LOS-rate signals that are used by the DAP in proportional navigation guidance. Two-degree-of-freedom attitude gyros provide signals for pitch, yaw, and roll stabilization. Figure 2 shows the autopilot configuration.

METHODOLOGY

The task of designing a missile digital autopilot could be accomplished by using both frequency and/or time domain tools. Controls systems tools in the classical and modern areas are available to most designers, however, classical techniques have several advantages over its competitors. The formulation of constraints, including nonlinear ones, cannot be easily treated with modern control techniques. The extension of tools in the classical

CH1437-3/79/0037-0044$00.75
1979 Winter Simulation Conference

domain can most of the time be made easier to understand than the adaptation of modern techniques for the solution of a particular problem. Since the application of digital technology to small terminal homing missiles is in itself at the foreground of technology, it was decided to use extensions of classical techniques (in the frequency domain). It was further decided to verify the results of these extensions with six degree of freedom time response simulations, including detailed nonlinear effects, and also including the real flight hardware.

DAP DESIGN

Two algorithms, representing two different DAP design approaches, were developed for the Army T6 missile. This airframe has flown numerous test flights using an analog autopilot. The two approaches are (1) a digitization of analog filters, and (2) a discrete domain design (D^3) using sampled-data control theory. The digitization approach requires a fast sampling rate in order to match analog filters response, and uses of all of the available microcomputer memory. The D^3 approach, on the other hand, permits a relatively slow sampling rate, and results in a more compact program.

DIGITIZED DAP

A digitization of the T6 missile analog autopilot is accomplished by converting each analog filter to a digital filter in parallel state variable form (1). Since it is impossible to find a discrete transfer function that exactly matches the gain and phase of an analog filter for all frequencies, some compromise is required. The selected frequency response matching criterion is to hold degradation of phase and gain margins at loop crossover frequency to five degrees and one db, respectively. Matching is always better at less than the loop crossover frequency.

This state variable digitization approach is composed of the following six actions:

(1) Expand each analog filter transfer function into partial fractions.

(2) Break the resulting terms into a system of simple integrators, gains and summing points.

(3) Write down the system state vector differential equation using the integrator outputs as state variables.

(4) Obtain the corresponding difference equation from (2)

$$\underline{x}_{n+1} = \Phi(T)\,\underline{x}_n + \underline{\gamma}\,u_n \qquad (1)$$

(5) Find the z-transform of equation 1 by computing X* using the equation

$$\underline{x}^*(z) = z(I) - \Phi(T)^{-1}\underline{\gamma}\,u^*(z)$$

and substituting x^* into the output response equation

$$y^*(z) = \underline{c}^T x^*(z) + d\,u^*(z) \qquad (2)$$

(6) Now equation 2 is evaluated with

$$z = e^{j\omega t} = \cos\,\omega T + j\,\sin\omega T$$

for various values of T. The largest value of T satisfying the frequency response criterion would normally be selected.

The result of the state variable conversion technique is equivalent to taking the z-transform of the analog filter combined with a zeroth-order hold. This zeroth-order hold introduces phase lag which should be compensated for. Other lag is created by the PWM (which has an 8 msec period), and by the computational delay (time required to evaluate difference equations). Normally compensation for these lags is accomplished by adding a predictor to the digital filter output.

After all the filters were converted, it was estimated that all the available processor memory would be used by the digitized DAP algorithm. Also the time required to multiplex the analog data through the A/D and to evaluate the difference equations was estimated as 3.107 msec. Due to sampling period, 4 msec was chosen. The digital filter coefficients were computed and the frequency responses of these digital filters compared favorably with those of the corresponding analog filters.

DISCRETE DOMAIN DAP

A discrete domain design (D^3) is accomplished by means of sampled-data control theory techniques. These techniques involve the proper transformation of the sampled s-plane block diagram through the z-plane. The necessary transformations are

$$w = \frac{z - 1}{z + 1} \text{ and } z = e^{st}$$

where T is the sampling period in seconds. The above transformations map the left half of the s-plane primary strip into the left half of the w-plane. Familiar s-plane design technique (computer-aided Bode, Nyquist, Root Locus, Routh-Hurwitz, etc.) can be used while working in the w-plane.

The sampling frequency (1/T) was chosen as 125 hertz, and is also the frequency of the PWM carrier. Synchronization to the PWM loop reduces system complexity and enables a relatively easy single-rate design.

All G_{xx} plant dynamics equations are computer-generated at a set of frozen time points along a typical aerodynamic trajectory. These are computer-handled and transformed into proper w-plane transfer functions when then generate Bode, polar and Nichols plots for the coupled uncompensated channels. After proper compensator design at several operating or frozen points, the compensated system stability marigins are obtained.

The attitude channels are a mixture of servo and regulator systems. Roll angle is desired at zero, where-

as pitch and yaw should execute the guidance commands with acceptable response characteristics while maintaining proper attitude stability. Figure 3 illustrates the simplied sampled pitch stabilization system structure. Using sampled-data control theory techniques presented in (2) the block diagram of Figure 3 is converted to the w-plane diagram of Figure 4. The proper design procedure than is to:

(1) Obtain A(s) actuator dynamics, multiply times (H(s) zero-order hold, and transform the result to w-plane to obtain HA (w).

(2) Using well known classical techniques design F(w) actuator compensator.

(3) Obtain $G_{22}(s)$ missile pitch uncoupled dynamics, multiply times H(s) and A(s), and transform the result to the w-plane to obtain HAG(w).

(4) Design $D_2(w)$ pitch filter.

The actual design procedure was of necessity a more complicated and iterative one, especially for the roll and yaw channels which are coupled. Figure 5 shows the computer-handled uncompensated pitch attitude loop for a set of 24 frozen time points, and Figure 6 shows the compensated loop. The roll and yaw loops were designed similarly.

Z-plane design is not as straightforward due to the inherent nonlinearity of the definition. Exponential compensation is not easy to visualize. W-plane compensators do not relate directly to time, as s and z filters do. Therefore, mapping out of w is necessary.

The compensators are then transformed to the z-plane where the microprocessor difference equations are obtained through choice of rectangular programming form. For example, the second order transfer function in the z-plane

$$\frac{y}{n} = \frac{az^2 + bz + c}{z^2 + dz + e}$$

taken through the rectangular programming form results in the difference equations

$$x_2((n+1)T) = c\ u(nT) + 0 - e\ y(nT)$$

$$x_1((n+1)T) = b\ u(nT) + x_2(nT) - d\ y(nT)$$

$$y(nT) = a\ u(nT) + x_1(nT) - 0$$

Table 1 shows the required flight software for all difference equations. It was estimated that the D^3 DAP would use only one-third of the available processor memory. Also the time required to convert the analog inputs to digital form and to evaluate all the difference equations was estimated as 1.649 msec thus running twice as fast as the digitized version. Thus the D^3 approach was selected over the digitized one.

After implementation, it was discovered that the complete cycle took almost 2 msec rather than 1.649 msec, however, the 1 msec output from input remained the same. Therefore, the D^3 DAP is about 4 times faster than the digtized DAP.

COMPUTATIONAL REQUIREMENTS

A computational summary of the arithmetic operations for a complete set of equations is shown in Table 2. The macrocode for addition includes a data transfer from memory to the accumulator register, thus, the complete addition instruction takes 2.25 microseconds. The macrocode for multiplication includes a data transfer from memory to the accumulator, a limit check for overflow, and roundoff of result. In addition, if the multiplier is not a fraction, a scale factor is also accounted for by a series of shift operations. Thus, multiplication times can range from 20 to 40 microseconds. An average value of 30 microseconds is used in Table 2.

The time required to convert the analog data to digital form is approximately 50 microseconds for each input. Since there are 10 analog signals input to DAP, the total multiplexed A/D conversion time is approximately 0.5 milliseconds. Since the total estimated computational time required is 1.649 msec, there should be no problem in completing all the required DAP operations within the 8 msec sample period.

TIMING

The estimated delay between DAP input and DAP output is 0.953 msec, about one fourth that taken by the digitized version. This is the time required to convert analog data to digital form, to execute the first equation of the programming form for each filter, to execute the various limits, rate limits and summing junctions, and to load the PWM counters. Table 3 shows a breakdown of the computational delay from microprocessor input to output. The effects of a one msec delay on stability were shown to present no stability problems.

MEMORY REQUIREMENTS

The memory requirements for the arithmethic sections of the DAP are shown in Table 4. The memory requirements are much less than the available memory (1024 words), about one third of that used by the digitized version.

MICROPROCESSOR HARDWARE ORGANIZATION

The digital autopilot is a MICOM design based on the first commercially available 4-bit slice bipolar microprocessor chip (Monolithic Memories 6701). The computer is configured for 16-bit arithmetic operations with 1024 words of main memory (256 RAM, 768 ROM) and 1024 x 50 bit words of microprogram memory. The main design objectives were to minimize hardware and maximize processing power. Minimum hardware was achieved by utilizing a design which did not have micro-sequencers, pipelining registers, look-ahead carry generators, or program control and branch units. Instead, a six phase clock was logically gated with certain control bits in the microprogram memory which determined buss priority (only one 16-bit buss for both address and data), sequencing and branching. Even with minimum hardware, the biopolar bit-slice design provided ample amounts of processing power with a micro-instruction execution time of 750 nanoseconds.

The DAP accomodates analog input signals with a 16 channel analog multiplexer thru a 12 bit A/D converter. Four of these inputs are currently spares for future expansion. Several discrete inputs are also monitored by the DAP. The digital autopilot drives the fin actuators with four digital pulse width modulators. Internal data (such as digital filter outputs) is monitored during flight with a bit-serial pulse code modulation telemetry output circuit.

Several versions of the DAP hardware are in use in the project. A DIP-wirewrap version was built to verify the initial hardware design and for initial HWIL testing of the software. This design utilizes RAM memory chips for the main and microprogram memory to facilitate software changes. The flight version of the DAP is a hybrid form design to conserve space in the six-inch diameter airframe. A special hybrid breadboard version was also constructed to test individual hybrid microcircuits, both as they were completed and as a total system.

External control of the autopilot for testing purposes is achieved with a Controller/Debug System (3) based on a Hewlett-Packard HP2100 minicomputer. This system allows one to load programs, run programs, inspect and test memory and internal registers, insert break points, plot internal data, and perform various other functions.

MICROPROCESSOR SOFTWARE SUPPORT

Initial software support for the DAP system consisted of an Emulator (4) developed concurrently with the hardware design and the controller/debug system mentioned in the previous section. The coding of a microprogrammable computer by hand proved to be tedious and error prone...to say nothing of the time involved. Therefore, a user-definable (meta) cross-assemblier was designed to process both micro and macro code (5). To further enhance programming, since the largest portion of DAP processing was equation solving, a high order language (HOL) compiler was developed. The HOL is a subset of FORTRAN with an editor/optimizer which optimizes the partitioning of micro/macro code in a post-translation effort. The use of the HOL greatly reduces the turn-around time for application program changes. The use of the HOL did not hinder execution time or memory resources due to the care taken in its design. Another feature which was added to the HOL compiler was a FORTRAN structured program preprocessor. This extended the flexibility and efficiency of the programming process by enabling the use of structured sequence control forms (e.g.,: IF( )-THEN-ELSE-ENDIF,DO WHILE( ) - ENDDO).

PERFORMANCE VERIFICATION

Prior to flying a prototype missile containing the DAP software performance analysis through extensive simulation is necessary. In so doing, expensive development of hardware prototypes are not necessary, thus, reducing the project's cost. Further, the use of Monte Carlo techniques permit (1) checkout of unknown random parameters, such as in the seeker, (2) evaluation of the effects of random but real-world perturbances on the system, and (3) investigation of the sensitivities of the missile to various initial condition scenarios. The use of simulation also permits checkout of all system nonlinearities which cannot be easily analyzed otherwise.

Future performance verification includes a hardware in the loop simulation, where the microprocessor and certain missile subsystems are exercised by a dynamical simulation program.

6DOF SIMULATION

The six degree-of-freedom (6DOF) time domain Monte Carlo Terminal Homing Simulation Program provides the capability of simulating flights of the T6 missile under varying conditions. Both statistical run sets and deterministic flights are possible.

The program is modularly designed, where the modules describe either hardware components (seeker, actuator, etc.), environmental conditions (winds and atmospheric data), or simulation-unique functions (kinematics, terminal geometry, etc.). See Figure 7.

This six degree-of-freedom all-digital simulation incorporates validated representations of all system elements sufficient for determination of overall prototype weapon system performance. The simulation techniques employed have been validated in conjunction with the firing of over 60 laser semi-active guided missiles. There is, therefore, a sound basis for confidence in the basic approach to the missile system simulation.

The simulation outputs time histories and cross plots of selected key variables, as well as miss distances. In addition, for Monte Carlo sets, a statistical summary and CEP numbers are included. Proper evaluation of output data permits the designer to objectively evaluate his system performance, and verify his simulation.

It must be noted that system fine tuning is accomplished by interaction with the deterministic 6DOF simulation, in order to accommodate any significant nonlinearities. Once a properly tuned design has been obtained, statistical evaluation is the only way to properly measure guided performance Seeker uncertainties in the simulation model, e.g. spot jitter, etc., rule out deterministic evaluation.

PROGRAMMED FLIGHT PERFORMANCE

In a direct guided flight, the seeker acquires the target prior to launch and remains locked-on to the target through launch, powered flight, coast and finally impact. In a programmed (or pre-programmed) flight, no target is used, thus the laser seeker is inoperative and guidance commands are generated from a predetermined time sequence. A programmed flight is ideal for testing the missile stabilization (control) loops and fine tuning the filter parameters. The program consists of a series of eight 5^0 step inputs.

The actual roll, pitch and yaw responses to the programmed inputs are shown in Figure 8.

HWIL SIMULATION

The hardware in the loop (HWIL) simulation consists of a hybrid digital analog real-time computer simulation employing the guidance and control DAP hardware-in-the-loop. This closed loop operation in a realistic simulation offers the opportunity to study the dynamic stability of the hardware system at first hand, with the hope of optimizing performance while reducing the length and cost of the flight testing cycle (6).

The hybrid simulation facility consists of the following major components:

(a) One Xerox sigma 5 digital computer with 64K words (32 bits) of core memory.

(b) Two Comcor Ci 5000, and one EAI 231R analog computers

(c) Interfaces to the digital computer for each analog computer.

The digital computer configuration contains an extensive array of peripheral devices including disk files, a tape drive, and a graphics display terminal. A set of 17 priority external interrupts is included in the system; these may be triggered by events which occur on the analog computers or internally by the digital program. The analog computers are 100-volt machines and both contain a wide range of linear, nonlinear, and logic elements.

The simulation is capable of being run in a repetitive Monte Carlo mode, where programmed random error sources introduce significant real environment inputs and disturbances, thus yielding realistic performance envelopes. Prior to going to that mode of operation, deterministic runs showed several problems.

PROBLEMS ENCOUNTERED

During the HWIL simulation problems encountered were grouped into three categories: (1) noise problems, impacting stabilization filter design, (2) microcode errors, pointing out faulty logic, and (3) hardware errors, including layout problems. These problems were cleared up through software changes (filter modification, logic patches, overflow correction) and minor wiring changes.

Several standard simulation problems were encountered in the initial HWIL simulation. These typically included modeling problems, such as with the actuator analog model gains, aerodynamic hinge moments, and the like. They occurred due to the lack of adequate modeling and test procedures. Testing should include not only the closed loop sinusoidal response, but also open loop time and frequency responses. The correction of some hardware problems by means of software patches points out the power and versatility of using microprocessors as digital controller elements

COMPLETE VERIFICATION

Complete agreement between frequency domain tools, 6DOF, and HWIL simulations was accomplished after several iterations. Compete agreement was defined loosely as comparing within the ink pen width of the brush recorder by an equal-scale overlay of the 6DOF, and engineering judgement in the case of the frequency domain tools. The partial use of real test trajectory data partially validated the simulation and the original equations of motion (7) as well.

CONCLUSION

Digital autopilots provide a flexibility not found before in flight control systems. Packaging DAPs in small tactical missiles is now possible due to the microprocessor revolution.

It has been shown that a discrete domain design digital autopilot is more efficient and performs as well or better than a digitized version. Therefore D^3 DAPs seem attractive to use in small missiles.

The redundant verification of autopilots adds a high degree of confidence to the final design. It also highly increases the probability of a successful flight.

Final validation of the D^3 techniques as well as of the hardware and the simulation will soon be accomplished by the test firing of a DAP-equipped T6 missile.

REFERENCES

1. Plunkett, K. W., Meadows, J. B., Albanes, W. V., and Bosley, J. T., Design and Analysis of a Microprocessor-Based Digital Autopilot for Terminal Homing Missiles, U. S. Army MIRADCOM TR-T-78-57, March 1978.

2. Kuo, B. C., Analysis and Synthesis of Sampled-Data Control Systems, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1963.

3. Baxter, W. F., Microprocessor Controller/Debug System, Report No. RG-76-61, U.S. Army MIRADCOM, Redstone Arsenal, Alabama, May 1976.

4. Brookshire, J. R., Multipurpose Digital Microprocessor Emulator, Report No. RG-76-62, U. S. Army MIRADCOM, Redstone Arsenal, Alabama, May 1976.

5. Scott, J. E., Jr., A Case Study of the Software/Firmware Development for a Microprocessor-Based Computer, U.S. Army Software Symposium Proceedings, October 1978.

6. Pastrick, H. L., Will, C. M., Isom, L. S., Jolly, A. C., Hazel, L. H., and Vinson, R. J., Hardware-In-The-Loop Simulation: A Guidance System Optimization Tool, 1974 AIAA Mechanics and Control of Flight Conference, AIAA Paper 74-929.

7. Yates, R. E., Templeton, J. A., Bosley, J. T., Dannenberg, K. D., and O'Hanian, S. L., Analysis and Design of a Terminal Homing System Digital Autopilot, U. S. Army MICOM TR RG-76-48, Nov. 1975.

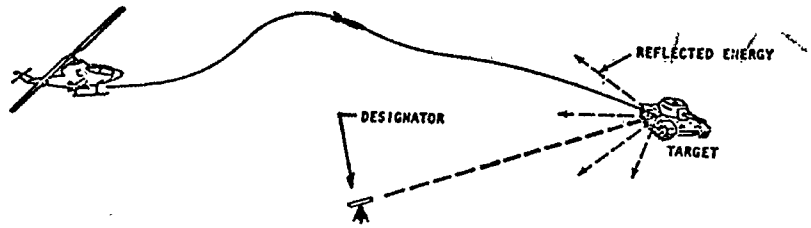


FIGURE 1. TYPICAL SCENARIO

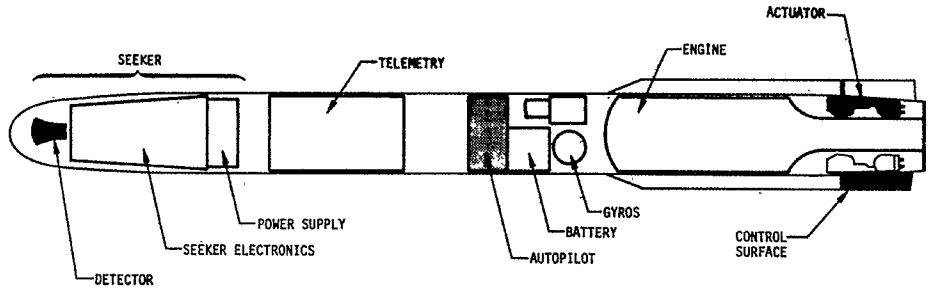


FIGURE 2. MISSILE SUBSYSTEMS COMPONENTS

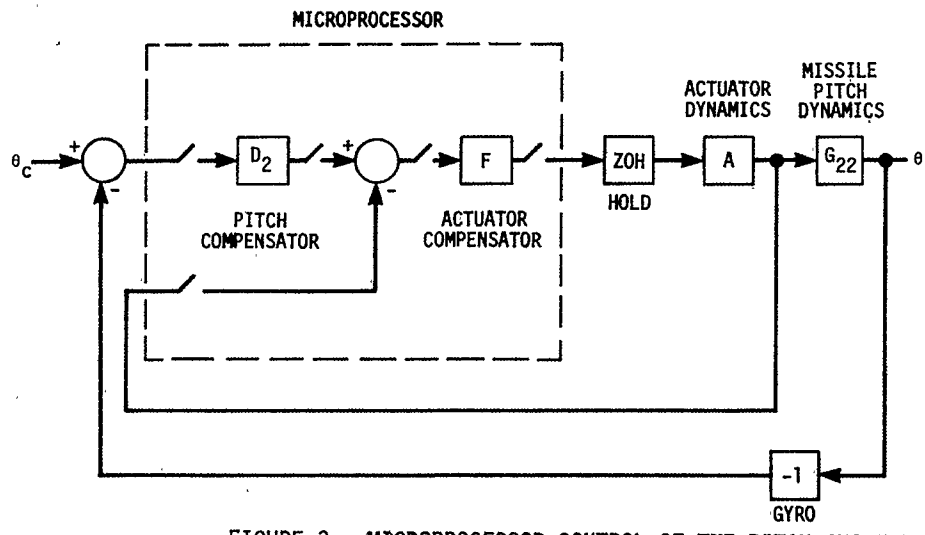


FIGURE 3. MICROPROCESSOR CONTROL OF THE PITCH CHANNEL

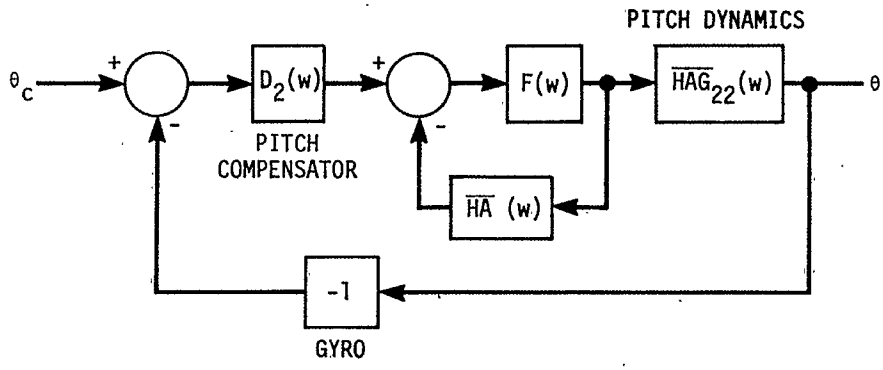
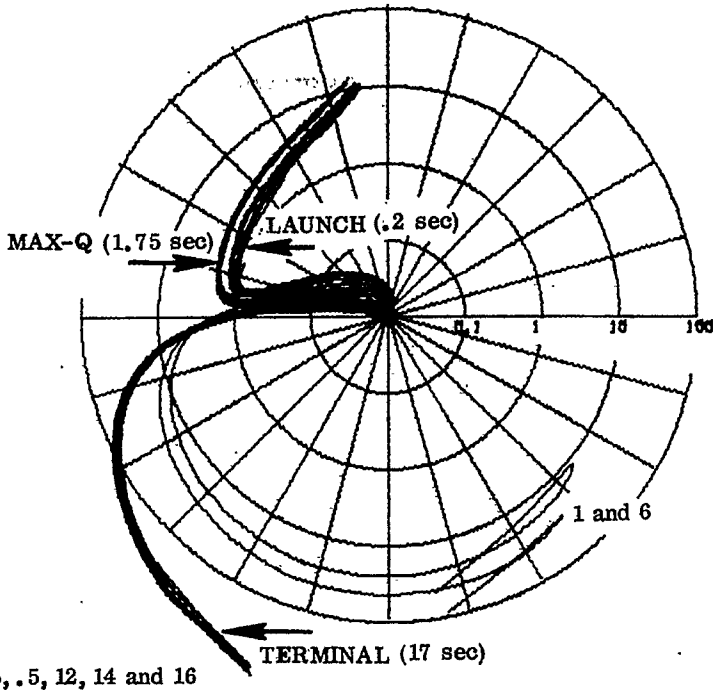


FIGURE 4. PITCH LOOP IN THE W-PLANE



.75, .5, 12, 14 and 16

FIGURE 5. UNCOMPENSATED PITCH LOOP POLAR PLOTS FOR 24 OPERATING POINTS

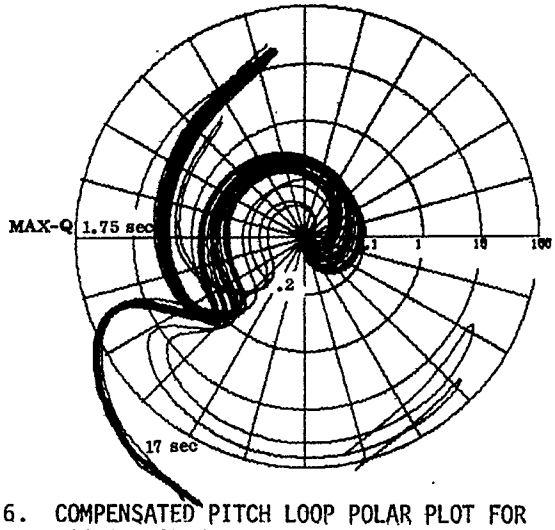


FIGURE 6. COMPENSATED PITCH LOOP POLAR PLOT FOR 24 OPERATING POINTS, WITHOUT G_t

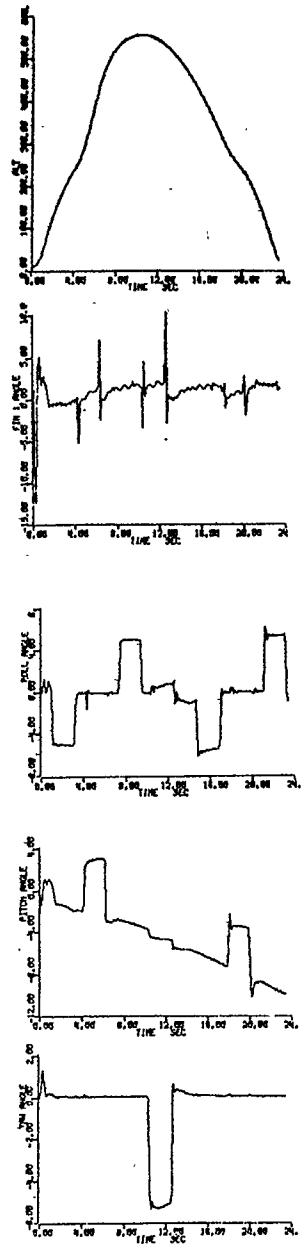


FIGURE 8. DAP PROGRAMMED FLIGHT

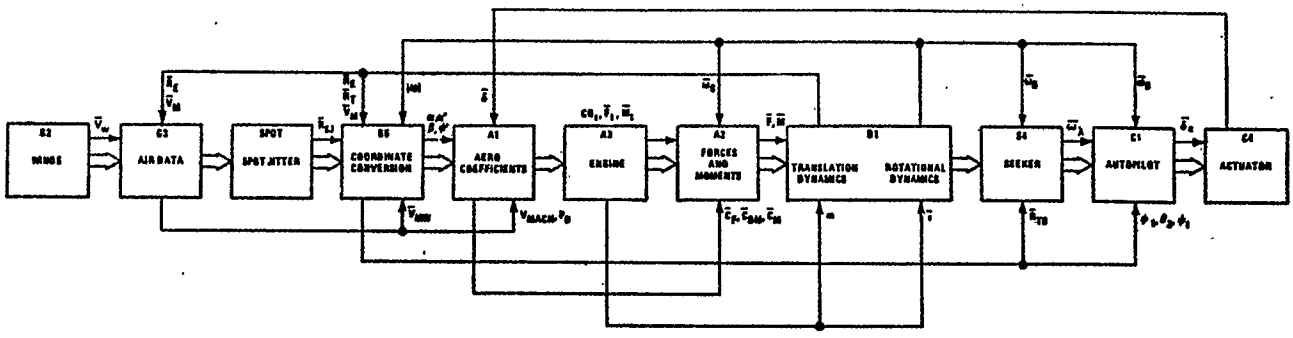


FIGURE 7. 6DOF MODULAR BLOCK DIAGRAM

	WQC=AO*(WLAMQ+X2)
	WRC=AO*(WLAMR+X4)
	BLQSS=X6
	BLRSS=X8
First Summing Junction	B13=BLQSS-BTHTS
	B24=BLRSS+BPSIS
	B13XX=A5*(B13-X9)*GT
	B24XX=A5*(B24-X10)*GT
	BRLSS=A7*(BPHI+X12)*GT
Second Summing Junction	B13SS=B13XX+B24XX
	B24SS=B13XX-B24XX
	RATE LIMIT B13SS and B24SS to K1
	LIMIT B13SS and B24SS to ± K2
Third Summing Junction	ERRF1=B13SS-BRLSS-DELT1+DELX
	ERRF2=B24SS-BRLSS-DELT2+DELX
	ERRF3=B13SS+BRLSS-DELT3+DELX
	ERRF4=B24SS+BRLSS-DELT4+DELX
	CERRF1=A10*(ERRF1+X14)
	CERRF2=A10*(ERRF2+X16)
	CERRF3=A10*(ERRF3+X18)
	CERRF4=A10*(ERRF4+X20)
	N1=512+CERRF1*K4
	N2=512+CERRF2*K5
	N3=512+CERRF3*K6
	N4=512+CERRF4*K7
	SHIFT OFF TWO BITS AND THEN MASK OFF BITS ONE THRU EIGHT FOR PWM COUNTER
	X2=B1*WQC-A1*WLAMQ-X1
	X1=B2*WQC
	X4=B1*WRC-A1*WLAMR-X3
	X3=B2*WRC
	X6=A3*(WQC+GBIAS)+B3*BLQSS+X5+BLQSS
	X5=A4*(WQC+GBIAS)-B4*BLQSS
	X8=A3*WRC+B3*BLRSS+X7+BLRSS
	X7=A4*WRC-B4*BLRSS
	X9=A6*B13
	X10=A6*B24
	X12=A8*BPHI-B8*BRLSS-X11
	X11=A9*BPHI+B9*BRLSS
	X14=ERRF1-B11*CERRF1-X13
	X13=B12*CERRF1
	X16=ERRF2-B11*CERRF2-X15
	X15=B12*CERRF2
	X18=ERRF3-B11*CERRF3-X17
	X17=B12*CERRF3
	X20=ERRF4-B11*CERRF4-X19
	X19=B12*CERRF4
	N=N+1
	IF(N.GE.100)N=100
	GT+5-(.04)*N
	IF(LAUNCH FLAG=NO)N=0

TABLE 1. FINAL DAP SOFTWARE

MEMORY REQUIREMENTS	MEMORY STORAGE
ARITHMETIC INSTRUCTIONS	250
COEFFICIENTS	18
INPUT VARIABLES	10
OUTPUT VARIABLES	4
INTERNAL VARIABLES	39
TOTAL	321

TABLE 4. MEMORY REQUIREMENTS FOR ARITHMETIC SECTIONS

DIGITAL AUTOPILOT SECTION	NUMBER REQUIRED	PROCESSOR COMPUTATIONAL REQUIREMENTS		
		ADD/SUB	MULTIPLY	DATA XFER
GUIDANCE FILTER	2	6	8	16
ACCUMULATOR	2	10	8	21
PITCH/YAW FILTER	2	2	4	8
ROLL FILTER	1	4	5	9
FIN FILTER	4	16	16	32
SUMMING JUNCTION	3	16	0	20
RATE LIMIT	2	4	0	4
LIMIT	6	6		12
TOTALS		64	41	122
ESTIMATED PROCESSOR INSTRUCTION EXECUTION TIME (MICROSECOND)		2.25	30.0	2.25
ESTIMATED COMPUTATIONAL REQUIREMENTS (MSEC)		0.144	1.230	0.275
TOTAL ESTIMATED COMPUTATIONAL TIME REQUIRED (MSEC)			1.649	

TABLE 2. TOTAL COMPUTATIONAL SUMMARY (ESTIMATED)

DIGITAL AUTOPILOT SECTION	NUMBER REQUIRED	PROCESSOR COMPUTATIONAL REQUIREMENTS		
		ADD/SUB	MULTIPLY	DATA XFER
GUIDANCE FILTER	2	2	2	4
ACCUMULATOR	2	0	0	2
PITCH/YAW FILTER	2	2	2	4
ROLL FILTER	1	1	1	2
FIN FILTER	4	4	4	8
SUMMING JUNCTION	3	16	0	20
RATE LIMITS	2	4	0	4
LIMITS	6	6	0	12
TOTALS		35	9	46
ESTIMATED PROCESSOR INSTRUCTION EXECUTION TIME (SEC)		2.25	30.0	2.25
ESTIMATED COMPUTATIONAL REQUIREMENTS (MSEC)		0.079	0.270	0.104
TOTAL COMPUTATIONAL DELAY TIME (MSEC)				0.453

TABLE 3. COMPUTATIONAL DELAY TIME