AN OVERVIEW OF HVACSIM+, A DYNAMIC BUILDING/HVAC/CONTROL SYSTEMS SIMULATION PROGRAM

Cheol Park
Daniel R. Clark
George E. Kelly

Building Equipment Division, Center for Building Technology
National Bureau of Standards, Gaithersburg, Maryland

ABSTRACT - In an effort to understand the dynamic interactions between a building shell, an HVAC system, and building controls, a non-proprietary building system simulation program called HVACSIM+ which stands for HVAC SIMulation PLUS other systems, has been developed at the National Bureau of Standards (NBS). HVACSIM+ consists of a main simulation program, a library of HVAC system component models, a building shell model, and an interactive front end program. The main simulation program employs a hierachical, modular approach and advanced equation solving techniques to perform dynamic simulations of building/HVAC/ control systems. In the building shell model, a fixed time step, selected by the user, is employed, while a variable time step approach is used in the HVAC and control systems portion of a simulation. This paper presents the overall architecture of HVACSIM the main simulation program, numerical methods used, HVAC component models, the building shell model, and the results of a sample simulation.

INTRODUCTION

Computer simulations have been a popular means in analyzing building energy use. Compared with experimental investigations, computer simulations do not require the installation of various expensive instruments. Simple changes in input data to a simulation model can measure their impacts on the model.

In an effort to carry out simulation studies involving dynamic interactions between a building shell, an HVAC system, and the building controls, a non-proprietary building system simulation program called HVACSIM+ has been developed at the National Bureau of Standards (NBS). The program HVACSIM+, which stands for HVAC SIMulation PLUS other systems, is capable of modeling the dynamic performance of the HVAC (heating, ventilation, and air-conditioning) system plus HVAC controls, the building shell, the heating/cooling plant, and energy management and control systems (EMCS) algorithms. Although the current version of the HVACSIM+ has not implemented the heating/cooling plant or EMCS algorithms yet, these can be easily added by a user interested in such applications.

The HVACSIM+ program consists of a main simulation routine, a library of HVAC system components models, a building shell model, and an interactive front end program. The main program is called MODSIM and employs a hierachical, modular approach and advanced equation solving techniques to perform dynamic simulations of building/HVAC/control systems. The modular approach is based upon the methodology used in the TRNSYS program[1]. In the building shell model, a fixed (but user selectable) time step is used, while a variable time step approach is employed in the HVAC and control systems portion. This hybrid time step method is believed to be unique in the building systems programs.

The HVACSIM+ program has primarily been developed as a research tool for whole building system studies. Flexibility of the HVACSIM+ allows the simulation of HVAC components, control systems, the building shell, or any combination of these. The ANSI standard Fortran 77 is the language used in the program. With little effort, HVACSIM+ can be tailored to fit the memory requirements of most computers. Fully structural programming enables the user to easily modify portions of codes.

Some of the more important features of HVACSIM+ were previously introduced at a Workshop on HVAC Controls Modeling and Simulation[2,3] along with some case studies[4,5]. A Reference Manual[6] and a Users Guide[7] are also available. The purpose of this paper is to provide an overview of the current version of the HVACSIM+ program and to provide supplementary information which was not presented in previous papers and reports.

ARCHITECTURE OF HVACSIM+

The various portions of HVACSIM+ can be divided into three categories: preprocessing, simulation, and postprocessing. Prior to performing a simulation, the data files for a particular building system simulations must be provided. This can be

accomplished using programs in the preprocessing group. After a simulation, evaluation of outputs from the simulation are made using the postprocessing programs.

Figure 1 shows a flow diagram of programs and data files comprising HVACSIM+. During the processing, a work file for simulation is created by the interactive front end program, HVACGEN[7]. This work file is then converted into the model definition file by the program, SLIMCON. The model definition file has the format which the main program, MODSIM, requires. The work file can be edited by the HVACGEN program interactively. In generating the simulation work file, HVACGEN employs a data file containing component model information.

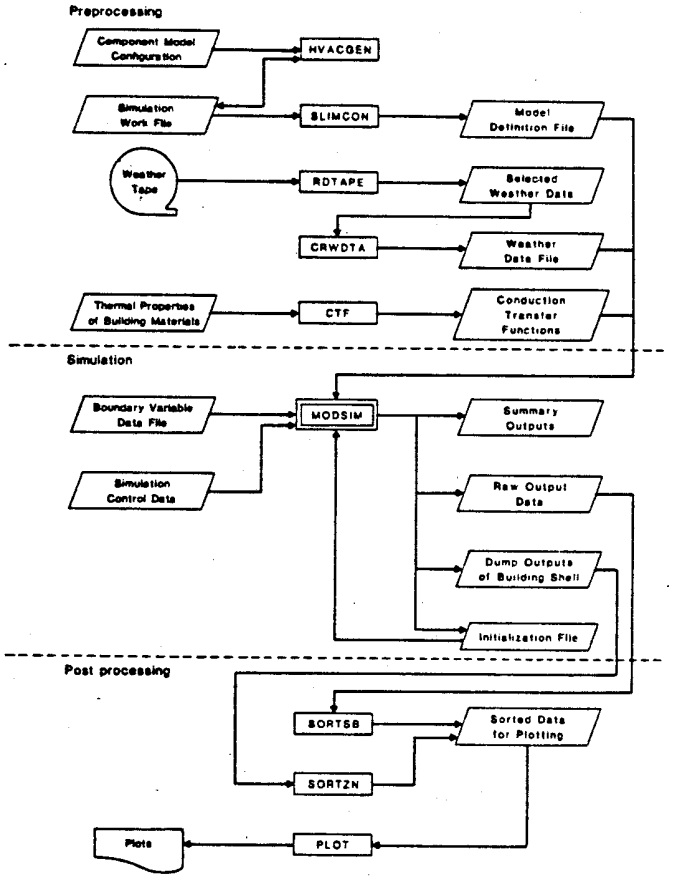

Fig. 1. Flow diagram of programs and data files of HVACSIM+

When a building shell is involved in a simulation, data files of weather conditions and conduction transfer functions for multilayered constructs[8] must also be created. The program, RDTAPE, reads a weather tape (SOLMET, TMY, TRY, or WYEC tapes) or equivalent and selects a portion of the weather data that is of interest. The selected weather data is transformed into the proper input form for MODSIM by the program CRWDTA.

The conduction transfer functions of multilayered building constructs are generated by the CTF program. Except for the front end routine the main routines in the CTF program are taken from the TARP gram[9]. The thermal properties of building materials (thickness, thermal conductivity, density, specific heat, and thermal resistance) can be entered into the data bank by using the CTF program, and multilayered constructs can be formed interactively.

The MODSIM program is the heart of the HVACSIM+. As shown in Figure 2, the MODSIM program consists of a main program and many subprograms for input/output operation, block and state variable status control, integration of stiff ordinary differential equations, solving of a system of simulataneous non-linear algebraic equations, component models of HVAC, controls, building shell, and supporting utility.

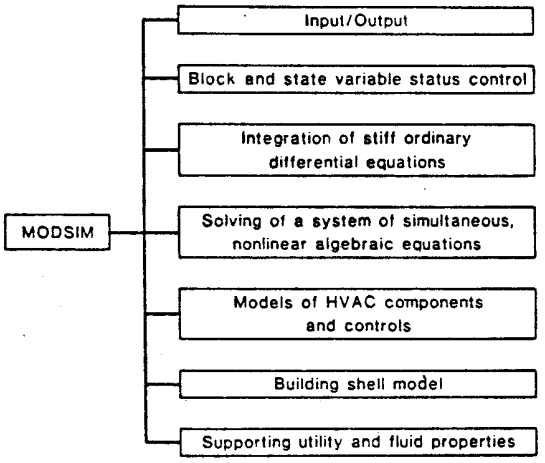

Fig. 2. The structure of MODSIM

The simulation program, MODSIM, calls the model definition, conduction transfer functions, weather, and boundary data files. The boundary data file can be created with a user provided editor program. The state variables associated with this boundary data file are assigned when the HVACGEN generates the work file for a particular simulation.

During the execution of MODSIM, simulation control input data can be entered interactively on a terminal or read from a file. After a successful simulation, four data files are generated. These are the summary, raw output, initialization, and building shell dump data files. After renaming the initialization file as the input file to the MODSIM, a new simulation can be performed starting from the point where the previous simulation ended.

Postprocessing is necessary if graphical presentation of the raw outputs or the dump data from the building shell model is desired. The program, SORTSB, sorts the raw output data, while the program SORTZN does the same to the building shell dump data. The outputs of these programs may then be used for plotting, with a user supplied graphic routine.

THE MODSIM PROGRAM

MODSIM stands for MODular SIMulation. Many ideas for the design of MODSIM came from the TRNSYS program, which was developed at the University of Wisconsin Solar Energy Laboratory[1]. The original MODSIM was first written in Fortran IV by Hill[3]. Since then, MODSIM has been rewritten in structured

Fortran 77 and modified significantly. The important features of the current MODSIM program are described below.

Hierachical, Modular Approach

A hierachical simulation setup data file (model definition file) is employed by MODSIM during a simulation. The hierachical structure comprises superblocks, blocks, and units. As illustrated in Figure 3, a number of units (or a single unit) form a block, and a number of blocks (or a single block) make up a superblock. Superblocks (or a single superblock) comprise a simulation. Figure 3 shows a setup involving 8 units, 4 blocks, and 2 superblocks. Depending upon the status of the state variables in a block or superblock, a system of equations in a block or in a superblock are solved simultaneously. The coupling of superblocks is done weakly through the state variables, since in the interest of economy the whole simulation made up of superblocks is not solved simultaneously.

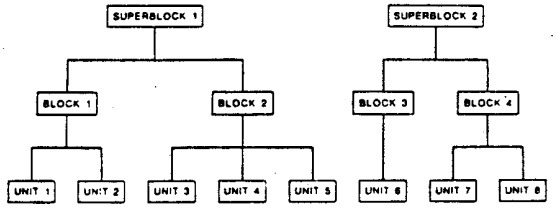

Fig. 3. Hierachical simulation setup

Using a modular approach, a UNIT in MODSIM represents a component model of an HVAC system control, or the building shell. Each physical component is modeled in the subroutine TYPEn, where n is the index number of the type assigned to the specific component. More than one unit can call the same TYPEn subroutine if the same component model is used more than once. For example, if UNIT 2 and UNIT 4 in Figure 3 represent two different fans in the HVAC system, the same TYPE1 subroutine for a fan (n=1) can be used in the simulation. Each subroutine of component model has inputs, outputs, parameters, and a workspace vector for saving results. The component model configuration data file, which is an input file to the HVACGEN program, contains information on the numbers of inputs, outputs, parameters, elements in the saved workspace vector, and a description of the inputs, outputs, and parameters.

Each UNIT has its distinct index number for input and output variables, and values of parameters. This information is transmitted to the corresponding TYPEn subroutine through arguments.

This hierachical, modular approach provides great flexibility in setting up a simulation model. The actual breakdown of a building system into blocks and superblocks is left to the user and depends upon the nature of the system and the type of interactions among its various components. Proper 'blocking' produces good simulation results and reduces computational time. Improper 'blocking' of a simulation model can result in a poor simulation.

Controls of State Variables and Blocks

During a simulation, a large portion of time is spent in solving the system of simultaneous equations. Reduction of the number of equations solved simultaneously in a block or a superblock can result in considerable computational savings. In MODSIM, when some of the state variables reach steady state, these variables are removed from the system of state variables that are solved simultaneously and put aside (or 'frozen') until deviations from the steady-state values are encountered. The criterion for freezing a variable is chosen as

$$\left|x_{n+1} - x_n\right| \leq \frac{1}{2}\left[e_r\left|x_{n+1}\right| + e_a\right], \qquad (1)$$

where x_{n+1} and x_n are the state variables at the current and the previous time, and e_r and e_a are the relative and the absolute error tolerance, respectively. These error tolerances must be specified when the simulation work file is created using the HVACGEN program.

Similarly, a block can be inactivated (or frozen) if all the input variables to the block are frozen. A block is marked active as soon as one of its block inputs becomes unfrozen. When a block is frozen, it is no longer necessary to monitor the frozen state variables in the block.

Hybrid Simulation Time Steps

The MODSIM program incorporates two different types of time steps. One of them is a fixed time step, and the other is a variable time step. The building shell model uses a user selected fixed time interval because the building shell model needs the conduction transfer functions of building constructs which are calculated on the basis of uniformly distributed time sampling. In addition, weather data is usually provided on an hourly basis. Variable time steps are used for all other component models.

This multi-time step approach has its advantage in computational time savings. Many component models for HVAC and controls systems involve ordinary differential equations. When the system is unsteady, a large time step invites numerical instability. To prevent this instability, small time intervals are necessary at an initial startup of a simulation or during periods when sudden change occurs. After the system becomes stabilized, the use of short time step is no longer needed and is wasteful.

Each superblock in a simulation is an independent subsystem in the sense that it proceeds forward in time independently. The variable time step is determined for each superblock, excluding the superblock for the building shell, by the integration routine used to solve the systems of differential equations. The largest time step allowed in a superblock is, however, limited to the fixed time step used in the building shell model.

Time Dependent Boundary Conditions

A state variable which is external to the system

being simulated can be designated as a boundary variable at the time that the simulation work file is generated. The boundary variables may be constant or time dependent. Data on the boundary variables are stored in the boundary data file and read as a simulation progresses. Time intervals in these data files are not required to be equal, since third order Lagrangian interpolation method is ;ed. Sometimes, a change in a boundary variable may be discontinuous (e.g., set point change). In such cases, the integration routine of differential equations is reset at the time of discontinuity to bring the simulation time step to a minimum value. This kind of reset condition is signaled by including in the boundary data file two different data values of a boundary variable at a given time.

NUMERICAL METHODS IN THE MODSIM

The numerical methods employed in the MODSIM program involve techniques for solving systems of simultaneous nonlinear algebraic equations, and integrating stiff ordinary differential equations. A large number of subprograms in the MODSIM are related to these numerical algorithms.

Nonlinear Equation Solver

The subroutine SNSQ with its associate subprograms is used in MODSIM. This routine is a part of the mathematical software package SNLSE in the CMLIB package, NBS[10] and was coded by Hiebert at Sandia National Laboratories by combining the HYBRD and HYBRDJ in the MINPACK code developed by Argonne National Laboratories[11]. The method used in the SNSQ program is based on Powell's hybrid method[12]. Minor modifications were made to the SNSQ routine to achieve better simulations with ™'ACSIM⁺.

A brief mathematical description of the SNSQ routine is presented following closely the approach used in a paper by Hiebert[11].

The system of nonlinear equations can be written in vector form as

$$\underline{f}\ (\underline{x})=\underline{0} \tag{2}$$

where

$$\underline{f} = [f_1,\ f_2,\ldots,f_n]^T,\ \underline{x} = [x_1,\ x_2,\ldots,x_n]^T$$

Expanding $\underline{f}$ in a Taylor series, and neglecting the high order terms, the linearized, approximate system becomes

$$\underline{f}(\underline{x}^*) \approx \underline{f}(\underline{x}^k) + J(\underline{x}^k)\ (\underline{x}^* - \underline{x}^k). \tag{3}$$

where $J(\underline{x}^k)$ is a Jacobian evaluated at $\underline{x}^k$.

If $\underline{x}^*$ is the solution vector of the system, then $\underline{f}(\underline{x}^*)=0$ and the Newton step of the nonlinear system, Δx, can be expressed as

$$\Delta\underline{x} = \underline{x}^{k+1} - \underline{x}^k = -J^{-1}(\underline{x}^k)\underline{f}(\underline{x}^k) \tag{4}$$

In efforts to reduce the number of calculations involved with this approach, a quasi-Newton method ·· used in SNSQ. This method approximates the Jacobian using the Broyden's rank-one update instead of calculating the full Jacobian at each iteration. The Jacobian is calculated at the starting point by either the user-supplied subroutine or a forward-difference approximation, but it is not recalculated until the rank-one method fails to give satisfactory progress. If B_k is the approximation of the Jacobian at the kth iteration, then the updated Jacobian is

$$B_{k+1} = B_k - (B_k\underline{q}_k - \underline{v}_k)\underline{q}^T_k/\underline{q}^T_k\underline{q}_k, \tag{5}$$

where $\underline{q}_k = \underline{x}^{k+1} - \underline{x}^k$, $\underline{v}_k = \underline{f}(\underline{x}^{k+1})-\underline{f}(\underline{x}^k)$, and $\underline{q}^T_k$ is the transpose of $\underline{q}_k$. In the SNSQ routine, the inverse Broyden update is employed. With the inverse Broyden update method, the inverse of the approximate Jacobian, B_k^{-1}, is stored and updated at each iteration.

The local convergence of the quasi-Newton method is superlinear, and required arithmetic operation per iteration is only $O(n^2)$ while the number of function evaluations per iteration is also only n. The shortcoming of the quasi-Newton method is that a good initial guess must be made for successful convergence. To improve this property, Powell[12] suggested a hybrid method.

The hybrid step is a combination of the quasi-Newton and gradient step. The gradient step is chosen to minimize the Euclidean norm of the residuals. The Gauss-Newton and the steepest scaled gradient steps are actually incorporated in the SNSQ routine. The convergence test is successful so that $\underline{x}^k$ is a solution vector if the following condition is satisfied:

$$||d_k\ (\underline{x}^{k+1} - \underline{x}^k)|| \leq e_t||d_k\underline{x}^k|| \tag{6}$$

or if $\underline{f}(\underline{x})=0$. In the above equation, d_k is the diagonal component of the transformed Jacobian matrix using QR-factoriation, e_t is the error tolerance usually specified by the user, and the double bars denote the norms. In HVACSIM⁺, the value of e_t is specified when the model definition file is created by the HVACGEN front-end program. Although the square-root of the machine precision is recommended for the value of e_t in the SNSQ routine, the choice of the value depends upon the particular simulation setup and its initial values. As a rule of thumb, e_t may be greater than or equal to the sum of e_r and e_a.

Beside the chosen value of e_t, the block/superblock structures, defined when a simulation setup is made, strongly influence the convergence characteristics. Even though the use of hybrid step improves the convergence properties, making a good guess for initial conditions is very important to ensure a successful simulation.

As an example, Figure 4 shows the simplified flow diagram for the iterative procedure when x_1 and x_2 are solved simultaneously and x_3 and x_4 remain constant at a time step. In the TYPE subroutines for two units in a block, x_1' and x_2' are determined using the function F_1 and F_2, respectively. Residual functions can be written as:

$$f_1(x_1,x_2,x_3,x_4) = x_1' - x_1 = F_1(x_2,x_3) - x_1 \quad (7)$$

$$f_2(x_1,x_2,x_3,x_4) = x_2' - x_2 = F_2(x_1,x_4) - x_2$$

These function vectors, f_1 and f_2, and state variables, x_i, are entered into the equation solver. When the convergence criterion as given by equation (6) is met, the iteration ceases, and the solutions x_1^* and x_2^* satisfy $f_1 = f_2 = 0$. After the solutions are obtained, the simulation time is increased by h, which is either variable time step or fixed.

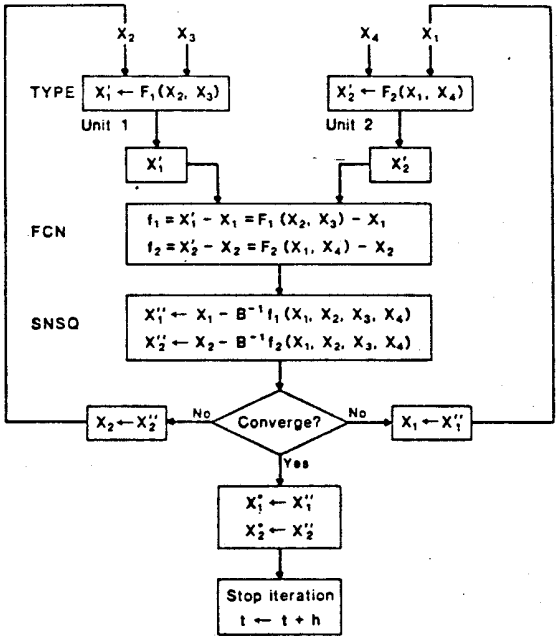

Fig. 4. Simplified flow diagram for the iterative procedure in solving simultaneous, nonlinear equations

Integration of Stiff Ordinary Differential Equations

The use of variable time step and variable order integration techniques to solve sets of differential equations can significantly reduce the amount of computer time required for dynamic simulations. The algorithm employed is the one developed by Brayton, Gustavson and Hachtel[13]. This is an extension of the famous Gear algorithm called DIFSUB[14], which uses the backward differential formulas associated with Nordsieck's method.

The discussion which follows will be a highlighted information of the method by Brayton, et al.[13]. Because a higher order ordinary differential equation can be transformed into a system of first-order differential equations, only integration of first order differential equations will be addressed.

A system of implicit differential algebraic equations can be expressed as

$$\underline{f}(\underline{x},\dot{\underline{x}}, t) = 0, \quad (8)$$

where $\underline{x}$ is a state variable vector which is a function of time, t, and $\dot{\underline{x}}$ is a derivative of $\underline{x}$. If the solution vector $\underline{x}(t)$ of equation (8) had been obtained at previous discrete times, $t=t_n$, $t=t_{n-1}$,..., and $t=t_{n+1-k}$, then the solution $\underline{x}_{n+1}$ at the current time, $t=t_{n+1}$, satisfies

$$\underline{f}(\underline{x}_{n+1}, \dot{\underline{x}}_{n+1}, t_{n+1}) = 0 \quad (9)$$

For stiff equations, the backward differentiation formula (BDF) approximates the present value $\underline{x}_{n+1}$ at $t=t_{n+1}$ in terms of $\underline{x}_{n+1}$, and the k past values: $\underline{x}_n$, $\underline{x}_{n-1}$,..., $\underline{x}_{n-k+1}$. The kth order backward differentiation formula is

$$\dot{\underline{x}}_{n+1} = -\frac{1}{h}\sum_{i=0}^{k} \alpha_i \underline{x}_{n+1-i}, \quad (10)$$

where α_i are constants and h is the present step size $(t_{n+1}-t_n)$. Setting $\underline{g}(\underline{x}_{n+1}) = \dot{\underline{x}}_{n+1}$, and substituting equation (10) into equation (9) yields a set of nonlinear algebraic equations of $\underline{x}_{n+1}$ at time t_{n+1}. This system of nonlinear equations can be solved by a nonlinear equation solver. In the MODSIM program, the previously described SNSQ routine is employed to solve the equations.

At the beginning of simulation, the initial values of $\underline{x}_0$ at t=0 is used with order k=1 for $\underline{x}_1$. Knowing x_0 and x_1, the new value x_2 is computed using $k\leq 2$, and so on. The maximum order of k is limited to 6 since the order k seldom exceeds 6 in most applications.

As discussed already, the Newton method requires a reasonably good guess for the initial iteration. The predicted value of $\underline{x}_{n+1}$ for the initial guess is formulated using the same regressor expression in equation (10):

$$\underline{x}^P_{n+1} = \sum_{i=1}^{k+1} \gamma_i \underline{x}_{n+1-i} \quad (11)$$

where γ_i are constants.

For the kth-order backward differential formula, the local truncation error is given by

$$e_{tr} = E_k + O(h^{k+2}) \quad (12)$$

where

$$E_k = \frac{h}{t_{n+1} - t_{n-k}} (\underline{x}_{n+1} - \underline{x}^P_{n+1}) \quad (13)$$

and the term $O(h^{k+2})$ represents higher-order terms in the step size of degrees greater than or equal to k+2.

Although the algorithm of computation for α_i and γ_i presented by Brayton, et al. involves complexity, it was coded in the MODSIM to improve computational efficiency. Chua and Lin[15] explained the variable step-size, variable-order algorithm in a much easier way to follow.

Figure 5 shows a simplified flow chart of the algorithm for integration of stiff ordinary differential equations which is implemented in the MODSIM. In the TYPE subroutine, the derivative of

calculated. The difference between the derivative $\underline{x}_{n+1}$ and the value of backward differential formula in equation (10) is denoted as $\underline{g}$. The residual function is, in fact, a nonlinear algebraic equation given by:

$$\underline{g} = \underline{G}(\underline{x}_{n+1}) + \frac{1}{h}\sum_{i=0}^{k} \alpha_i \underline{x}_{n+1-i} \qquad (14)$$

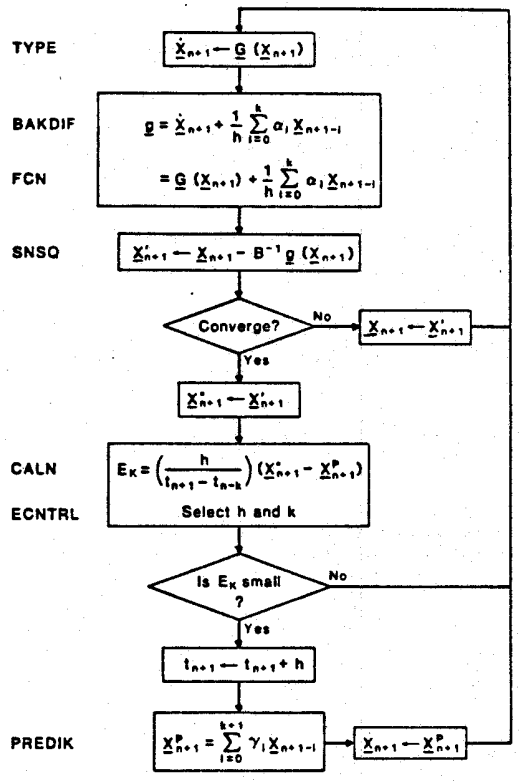

Fig. 5. Simplified flow chart of the algorithm for integration of stiff ordinary differential equations

When $\underline{x}_{n+1}$ is the solution of equation (14), $\underline{g}$ is zero. To find the solution at the present time, numerical iteration using the SNSQ routine is performed and convergence is checked. If the solution is converged close to the real solution, the iteration is terminated and the truncation error of backward differential formula is computed and the order k and the step are determined. The selected step and order are rejected if the truncation error is too large. The strategy of selecting the order and step with the MODSIM is based on the condition:

$$E_k \leq \frac{3h(e_r|\underline{x}_{n+1}|^{+} e_a)}{t_f - t_i} \qquad (15)$$

where t_i and t_f are the initial and final time considered in the integration using the backward differential formula. The time interval, $t_f - t_i$, must be provided as one of input values prior to simulation. This quantity is specified during the e when the simulation work file is generated.

HVAC COMPONENTS AND CONTROLS

A library of component models, suitable for dynamic simulations with short time steps, were developed for use with the HVACSIM+. The library consists of 26 TYPE subroutines for models of HVAC components and controls, and routines for transport delay in ducts and pipes, hysteresis effects of a valve or damper, and heat exchanger fin efficiencies. In addition, properties of air, steam, water, and refrigerant are included. A room model by Hill[4] is also included in the library. This room model accounts for the air mass, the interior mass, and the wall mass but treats heat flows through walls as known inputs. The models of HVAC components and controls in the current version of the HVACSIM+ are listed in Table 1. The component models and the equations upon which they are based are described in great detail elsewhere[4,5,6] and will not be discussed further in this paper.

Table 1. Models of HVAC Components and Controls

TYPE 1	FAN OR PUMP
TYPE 2	CONDUIT (DUCT OR PIPE)
TYPE 3	INLET CONDUIT (DUCT OR PIPE)
TYPE 4	FLOW MERGE
TYPE 5	DAMPER OR VALVE
TYPE 6	FLOW SPLIT
TYPE 7	TEMPERATURE SENSOR
TYPE 8	PROPORTIONAL-INTEGRAL CONTROLLER
TYPE 9	LINERAR VALVE WITH PNEUMATIC ACTUATOR
TYPE 10	HOT WATER COIL MODEL
TYPE 11	HOT WATER TO AIR HEATING COIL
TYPE 12	COOLING OR DEHUMIDIFYING COIL
TYPE 13	THREE-WAY VALVE MODEL
TYPE 14	EVAPORATIVE HUMIDIFIER
TYPE 15	ROOM MODEL
TYPE 16	STICKY PROPORTIONAL CONTROLLER
TYPE 17	MIXING DAMPERS AND MERGE
TYPE 18	PLENUM
TYPE 19	FLOW BALANCE CONTROL
TYPE 20	HIGH/LOW LIMIT CONTROLLER
TYPE 21	CLAMPED SPLIT
TYPE 22	STEAM SPRAY HUMIDIFIER
TYPE 23	STEAM NOZZLE
TYPE 24	IDEAL GAS NOZZLE
TYPE 25	STEAM TO AIR HEATING COIL
TYPE 26	CONTROL SIGNAL INVERTER

BUILDING SHELL MODEL

A building shell model was developed based mainly on the works by Kusuda[16] and Walton[9]. It is used to determine building thermal loads and includes the effects of different kinds of building shell materials, air temperatures, the moisture content of the air, lighting, equipment, occupancy schedule, solar radiation, wind velocity, orientations of the exterior building surfaces, and the effect of shadowing. Since there are so many factors involved, some simplifying assumptions had to be made. The major assumptions include:

(1) Uniform temperature distributions on a building surface (one dimensional heat transfer across a wall)

(2) Simple, daily averaged shadowing effect on building exterior surfaces
(3) Uniform ground temperature distribution
(4) No effects of wind direction, rain, and snow

The approach taken uses the standard response factor method to calculate the conductive heat transfer rates through the building shell. The conduction transfer functions are computed once and stored prior to a simulation. The same time interval used in the calculation of conduction transfer functions of building constructs is used as the period during which the conductive heat fluxes through the building surfaces are assumed to be invariant.

Primary routines for the zone load determination are those dealing with the calculation of building surface temperatures and zone loads. These are treated as component models, and coded as TYPEn subroutines. Subroutines for reading weather data and conduction transfer functions, and calculating solar components (if the weather data file does not provide them) are incorporated. All these subroutines are executed each time a heat balance computation is made. Because of use of the fixed time step, the units representing building surfaces and zone loads must be in a superblock which is separate from those containing units which use a variable time step.

A zone model is also employed in HVACSIM$^+$. It calculates indoor air dry-bulb temperature and humidity ratio on a variable time step basis and takes into account the dynamic operation of the HVAC system and its controls. The zone loads determined by the building shell model are transmitted to the zone model.

Zone Loads

A building can be divided into a number of zones. For each zone, thermal loads are computed as heat flow rates.

Convective heat flow rate across the air film between the zone air and interior surface of the building shell is given by:

$$\dot{Q}_{wall} = \sum_{j=1}^{N_s} h_{is,c,j} A_{s,j} (T_{is,j} - T_i) \qquad (16)$$

where $h_{is,c,j}$, $A_{s,j}$, $T_{is,j}$, T_i, and N_s are the convective heat transfer coefficient of the interior surface of the j-th wall, the surface area, the interior surface temperature, the zone air dry-bulb temperature, and the number of walls, respectively. The term 'walls' includes ceiling, floor, windows, doors, and walls.

The convective heat transfer coefficient of an inner surface is obtained from one of the following expressions[9]:

$$h_{is,c} = \frac{9.482 \mid T_i - T_{is} \mid^{0.333}}{7.238 - \mid \cos\emptyset \mid} \quad \text{if } T_{is} \geq T_i \qquad (17)$$

$$h_{is,c} = \frac{1.810 \mid T_i - T_{is} \mid^{0.333}}{1.382 + \mid \cos\emptyset \mid} \quad \text{if } T_{is} < T_i \qquad (18)$$

where $\emptyset$ denotes the tilt angle of the surface from horizontal plane. The unit of heat transfer coefficients is watts/m^2K and that of heat flow rates is watts.

Heat gain or loss due to infiltration is given by:

$$\dot{Q}_{infl} = C_{p,infl}\, \rho_{infl}\, V_i\, I_{air}\, (T_o - T_i) \qquad (19)$$

where $C_{p,infl}$, ρ_{infl}, V_i, I_{air}, and T_o denote the specific heat and density of infiltrated air, the volume of air in a zone, the air exchange rate, and the outside air dry-bulb temperature, respectively.

The air exchange rate is calculated using wind speed, and the dry-bulb temperature difference between indoor and outdoor air[17].

$$I_{air} = I_{s,air} [0.15 + (0.013)(2.237)V_w + (0.005)(1.8)|T_o - T_i|]/0.695 \qquad (20)$$

where $I_{s,air}$ and V_w are standard air exchange rate (1/h), and wind speed (m/s), respectively. Usually the value of standard air exchange rate is chosen 1.5 for leaky building, 1.0 for standard, and 0.5 for moderately tight buildings.

Convective heat gains from people occupying the zone, equipment and lights are given by:

$$\dot{Q}_{people,c} = (1 - r_p) N_p W_{p,s} \qquad (21)$$

$$\dot{Q}_{equip,c} = (1 - r_e) U_e W_{e,s} \qquad (22)$$

$$\dot{Q}_{light,c} = (1 - r_\ell) U_\ell W_{\ell,s} \qquad (23)$$

where N_p is number of people, $W_{p,s}$ and $W_{e,s}$ are total sensible heats from a person and equipment, and W_ℓ is sensible heat gain from lighting. r_p, r_e, and r_ℓ represent ratios of radiative heat to total sensible heat from people, equipment, and lights. Typical values of these factors are: $r_p = 0.4$, $r_e = 0.2 \sim 0.8$, $r_\ell = 0.8$ for incondescent lights, and 0.5 for fluorescent lights[18]. The quantitites U_e and U_ℓ are defined as utilization coefficients for equipment and lights. Latent heat gains from people and equipment are also considered while moisture absorptance and desorption by the building structure and interior furnishings are not explicitly included in the building shell model.

Long wave radiant heat gains from people and equipment along with the radiative heat from building surfaces are used to obtain mean radiant temperature of the zone[19]. The use of mean radiant temperature is much simpler than using the radiant heat-exchange between walls. Short wave radiation due to lights and the sun are not directly involved in computation of the mean radiation temperature.

The expression of mean radiant temperature is given by:

$$T_{mr} = \frac{\sum_{j=1}^{N_s} h_{is,r,j} A_{s,j} T_{is,j} + \dot{Q}_{people,r} + \dot{Q}_{equip,r}}{\sum_{j=1}^{N_s} h_{is,r,j} A_{s,j}} \quad (24)$$

where $\dot{Q}_{people,r}$, $\dot{Q}_{equip,r}$, $h_{is,r,j}$ are the radiative heat gains from people and equipment, and the radiant heat transfer coefficient, respectively.

Surface temperatures

Each surface of a zone is treated as a unit. For example, if a zone is enclosed by seven different surfaces, seven units are needed. Interior surface temperature is important quantity which governs heat flow rates associated with the building envelope. A heat balance at the j-th interior surface is used to determine the interior surface temperature by:

$$T_{is,j} = (h_{is,c,j} T_i + h_{is,r,j} T_{mr} + \dot{q}_{sol,i,r} + \dot{q}_{light,r} + \dot{q}'_{i,j} + Y_{o,j} T_{os,j}) / (h_{is,c,j} + h_{is,r,j} + Z_{o,j}) \quad (25)$$

where $\dot{q}_{sol,i,r}$ and $\dot{q}_{light,r}$ are shortwave radiative heat fluxes originated from the sun and lights. It is assumed that these shortwave radiant fluxes are uniformly absorbed by surfaces in a zone. The external surface temperature is $T_{os,j}$ and conduction transfer function terms at the present time are $X_{o,j}$, $Y_{o,j}$, and $Z_{o,j}$. The inward conductive flux at the inside of surface j at the present time due to ...t temperature history is $q'_{i,j}$.

The current conductive heat flux at the inner surface is

$$\dot{q}_{i,j,n} = Y_{o,j} T_{os,j} - Z_{o,j} T_{is,j} + \dot{q}'_{i,j} \quad (26)$$

where

$$\dot{q}'_{i,j} = \sum_{m=1}^{N_t} Y_{m,j} T_{os,j,n-m} - \sum_{m=1}^{N_t} Z_{m,j} T_{is,j,n-m} + \sum_{k=1}^{N_f} R_{k,j} \dot{q}_{i,j,n-k} \quad (27)$$

In the equations above, the subscript n is the current time, while m denotes past time. Note that the time interval is fixed. The number of terms of conduction transfer functions and the number of the order are N_t and N_f, respectively. The flux, $R_{k,j}$, is related to the overall conductance, U_j, as

$$U_j(1 - \sum_{k=1}^{N_f} R_{k,j}) = \sum_{m=0}^{N_t} X_{m,j} = \sum_{m=0}^{N_t} Y_{m,j} = \sum_{m=o}^{N_t} Z_{m,j} \quad (28)$$

The values of $R_{k,j}$ and U_j as well as $X_{m,j}$, $Y_{m,j}$ and $Z_{m,j}$ are computed by the CTF program.

Or the exterior surface, simple heat balance equation is

$$h_{os,j}(T_o - T_{os,j}) + \dot{q}_{sol,o,j} + (\dot{q}'_{o,j} + Y_{o,j} T_{is,j} - X_{o,j} T_{os,j}) = 0 \quad (29)$$

where $h_{os,j}$, $\dot{q}_{sol,o,j}$, and $\dot{q}'_{o,j}$ are the convective plus radiative heat transfer coefficient of the outer surface (W/m^2K), the solar radiative flux, and the heat flux. The coefficient, $h_{os,j}$, is given in an expression as a function of wind speed.

$$h_{os,j} = a_o + a_1 V_w + a_2 V^2_w \quad (30)$$

in which a_o, a_1 and a_2 are coefficients which can be determined by the surface roughness index. Walton provided the values of these coefficients with respect to roughness index in his TARP reference manual[9]. The wind speed is the reported value without modification for surface height or orientation. From equation (29), the outer surface temperature can be computed.

The solar fluxes used in equations (25) and (29) are evaluated by using the following expressions.

$$\dot{q}_{sol,i,r} = \frac{\sum_{k=1}^{N_g} A_{g,k} \tau_{g,k} S_{c,k} I_{sol,k}}{\sum_{j=1}^{N_s} A_{s,j}} \quad (31)$$

$$\dot{q}_{sol,o,j} = \alpha_w I_{sol,j} \quad (32)$$

in which $A_{g,k}$, $\tau_{g,k}$, $S_{c,k}$, and $I_{sol,k}$ are the k-th glass window area in a zone, transmittance of glass window, shading coefficient, and total solar radiation coming through the k-th window. For the exposed walls, the effect of solar radiation depends upon the absorptivity, α_w. In both equations (31) and (32), approximations are made in terms of optical properties for the sake of simplicity. Rigorous calculation methods can be found elsewhere[9,16].

Zone Model

The building shell model discussed above computes the heat flow rates and fluxes for a zone at a fixed time interval. The zone model discussed below uses the outputs from the shell model but employs a variable time step in its simulation.

The zone model in the current HVACSIM+ is modeled as a TYPEn subroutine and is used to determine zone air dry-bulb temperature and humidity ratio. As a routine, it should reside in a superblock where a variable time step is being employed.

The zone air temperature is obtained using

$$(C_{fur} + C_{air}) \frac{dT_i}{dt} = \dot{Q}_s + \dot{Q}_{infl} + \dot{Q}_{wall} + \dot{Q}_{light,c} + \dot{Q}_{people,c} + \dot{Q}_{equip,c} \quad (33)$$

All heat flow rates, except the heat flow rate

delivered by supply air, Q_s, are computed by the shell model and transmitted through COMMON blocks to the zone model. In equation (33), C_{fur} and C_{air} are the capacitance of furnishing and zone air.

In terms of humidity ratio, W, the moisture content of zone air is expressed as:

$$(m_{fur} + \rho_i V_i) \frac{dW_i}{dt} = (\dot{Q}_{people,\ell} + \dot{Q}_{equip,\ell})/h_{fg} + \dot{m}_{infl} (W_o - W_i) + \dot{m}_s (W_s - W_i) \qquad (34)$$

where m_{fur} is the effective mass of furnishing, and h_{fg} is latent heat of vaporization of water, which can be obtained from the fluid property library. The outdoor humidity ratio, W_o, comes from weather data. $\dot{m}_{infl}$ denotes mass flow rate due to infiltration.

Figure 6 illustrates simplified data flows for a simulation of a simple, two story building with a flat top. The subroutines TYPE50, TYPE51, and TYPE52 are the calculation routines for zone loads, surface temperatures, and zone air temperature and humidity, respectively. In this case, the construct between the first and second floor is an interior interface. Thus the exterior temperature of the ceiling of the first floor zone is the interior temperature of the floor of the second floor zone, and vice versa.

The simulation setup in Figure 6 involves 17 units in superblock 1 (8 for the first floor zone and 9 for the second), and 2 units in superblock 2, (one for each zone). Any units describing HVAC components that are connected to a zone must be included in superblock 2.

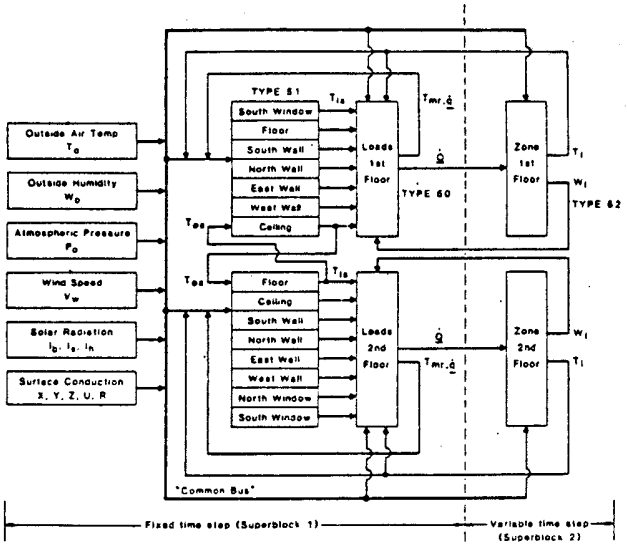

Fig. 6. Data flows of a simulation using the building shell model

CAPACITIES AND LIMITATIONS

The HVACSIM$^+$ program uses PARAMETER statement of Fortran 77 language to make it easy to change data storage requirements during executions. The current version of HVACSIM$^+$ is capable of simulating up to 10 superblocks, 50 blocks, and 200 units. The building shell model can simulate up to 6 zones with 10 surfaces per zone. There are three types of simulations possible: the HVAC and control simulation using a variable time step, the building shell simulation using a fixed time interval, or the building/HVAC/control system simulation using both fixed and variable time steps.

Simulations involving the building shell model require a two step operation. In the first step, an initialization run must be performed at a fixed time interval to create regressor vectors of conductive heat fluxes and surface temperatures on the building surfaces. During this step, only the building shell portion of the MODSIM is active and other superblocks or blocks are frozen. After this initialization, the second step involves simulation of the building/HVAC/control system model under dynamic conditions.

The convergence properties of the equation solver depend on the block or superblock structure and initial values of state variables. Some care must be exercised in defining blocks particularly when control loops are involved. Reasonably good estimation of initial conditions is also needed for successful simulations.

Superblocks are assumed to be only weakly coupled. No simultaneous equations are defined between them, and they are allowed to evolve independently in time through the state variables. Consequently, care is required in dividing a system into superblocks. Difficulties can arise due to the independent time evolution of superblocks, since the inputs to a superblock may change at a time when that superblock is not scheduled to be called. To mitigate this problem, a superblock input scan option (INSOPT) may be selected when the simulation work file is generated. When this option is selected, INSOPT=1, all superblock inputs are scanned after each time step. If the inputs to a superblock which was not called during the time step have changed more than an allowable amount, the superblock is called and state variables are computed based on the new inputs.

SAMPLE SIMULATIONS

A single-duct air-handling unit connected to a zone was simulated under summer operating conditions. The objective of this simulation was to observe dynamic behavior of the PI controller on the cooling coil as it tried to maintain the discharged air temperature leaving the supply fan at its set point. Figure 7 shows the system which was simulated. The outdoor air enters the system through a set of dampers and is mixed with return air. The mixed air then passes through an air filter, a heating coil and a cooling coil. The heating coil is not shown and is considered only as a flow resistance during the cooling season. A fan supplies the conditioned air to the zone. A return air fan withdraws air from the zone, and a part of it is exhausted to the outside.

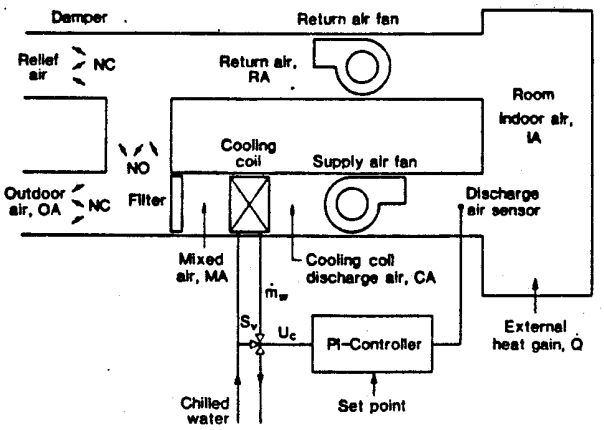

Fig. 7. An air-handler and a room used in the sample simulation

In this system, the mass flow rate of the chilled water is controlled by a three-way valve, which is governed by a proportional-integral controller (PI-controller). The controller operates to minimize the difference between the temperature measured by a sensor in the duct downstream of the supply fan and the set point. Feedback by a room thermostat is not considered and damper openings are constant during this particular simulation. The room model developed by Hill[4] was employed in the simulation. The outdoor air (OA) dry-bulb temperature and external heat gain, Q, are included and entered in the boundary data file. These boundary values are shown in Figure 8 as a function of the simulation time. In terms of time of day, simulation starts at 5:00 a.m. and ends at 5:00 a.m. of the next day.

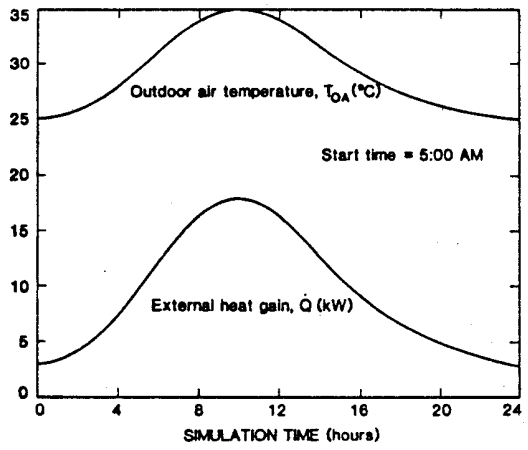

Fig. 8. Outdoor air temperature and external heat gain imposed in the sample simulation

As shown in Figure 9, as the zone load increases, the chilled water flow rate, $\dot{m}_w$, increases. The controller output, U_c, and the relative valve stem position, S_v, are also presented in this figure. It can be seen that the controller output drops suddenly when the cooling demand starts to decrease. This is due to the hysteresis in the valve. In Figure 10, the mixed (MA), return (RA), indoor (IA), and cooling coil discharge air (CA) temperatures are shown along the outside air temperature (OA).

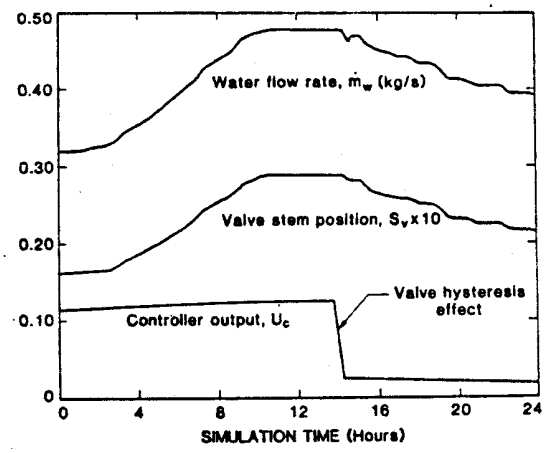

Fig. 9. Chilled water flow rate, valve stem position, and controller output

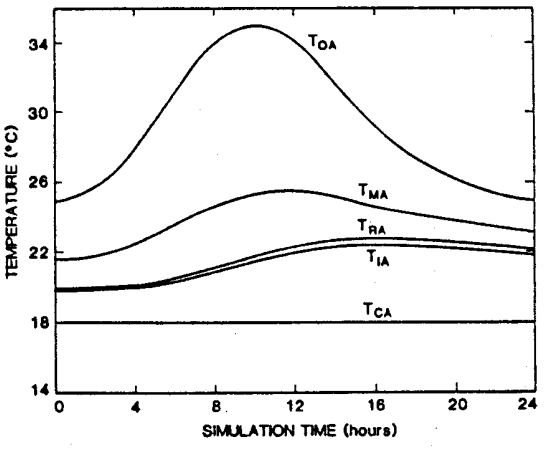

Fig. 10. Temperature histories obtained from the sample simulation

In this example, all units representing components of the system were set up in a single superblock and no routines for the building shell were employed. In the superblock, there are 3 blocks containing 24 units altogether. With 68 state variables, there are 46 simultaneous equations, of which 11 equations are first-order ordinary differential equations. A simulation of the air-handling system for a multi-zone application employing the building shell model will be the subject of a future paper.

SUMMARY

This paper has presented an overview of the HVACSIM$^+$ simulation program, which was recently developed at the National Bureau of Standards for modeling the dynamic performance of whole building systems. The program's architecture, its hierarchical modular structure of superblocks and blocks and units, its control of state variables, its ability to freeze and unfreeze blocks, its advance equation solving techniques, and its building shell and zone models were discussed in some detail. A simple example was also given illustrating HVACSIM$^+$'s

ability to model the proportional and integral control of a cooling coil in an air handler supplying conditioned air to a single zone.

Of particular importance, HVACSIM$^+$ was shown to have the ability to simultaneously solve sets of non-linear, stiff differential equations and non-linear algebraic equations using advanced variable order/variable time step integration methods and to allow different superblocks to be simulated using different time steps. It thus becomes possible using HVACSIM$^+$ to simulate the dynamic performance of whole building/HVAC/control systems with the control dynamics being modeled 'second-by-second,' the HVAC system and zone dynamics calculated 'minute-by-minute,' and the heating/cooling loads through the building shell being determined on a 15 minute to 1 hour basis. This avoids the costly and error prone process of trying to simulate an entire building on a time step dictated by the fastest dynamics present in the system, usually those of the control system.

The program HVACSIM$^+$ has been shown to be a powerful research tool that will, for the first time, allow researchers to simulate the dynamic performance of entire building/HVAC/control systems. It should lead to new techniques for optimizing whole building performance and form the basis for a new generation of design and simulation tools for HVAC system and control engineering, architects, and building code officials.

ACKNOWLEDGMENTS

The authors are indebted to the Building Systems Division, Department of Energy, and the U.S. Navy Civil Engineering Laboratory, Department of Defense, for funding this project, and gratefully acknowledge Mr. George N. Walton for his helpful suggestions, and Mr. William B. May, who directed the development of the front end program HVACGEN and provided hardware and software support for this project.

REFERENCES

[1] Klein, S.A., et al., 'TRNSYS, A Transient System Simulation Program,' Report 38-12, University of Wisconsin, Dec. 1983.

[2] Kelly, G.E., Park, C., Clark, D.R., and May, W.B., 'HVACSIM$^+$, A Dynamic Building/HVAC/Control Systems Simulation Program,' Proc. of Workshop on HVAC Controls Modeling and Simulation, Georgia Inst. of Tech., Atlanta, GA, Feb. 2-3, 1984.

[3] Hill, C.R., 'Simulation Techniques for Building Systems,' Proc. of Workshop on HVAC Controls Modeling and Simulation, Georgia Inst. of Tech., Atlanta, GA, Feb. 2-3, 1984.

[4] Hill, C.R., 'Simulation of a Multizone Air Handler,' ASHRAE Trans., Vol. 91, Part 1, 1985.

[5] Clark, D.R., Hurley, C.W., and Hill, C.R., 'Dynamic Models for HVAC System Components,' ASHRAE Trans. Vol. 91, Part 1, 1985.

[6] Clark, D.R., 'HVACSIM$^+$ Building Systems and Equipment Simulation Program Reference Manual,' NBSIR 84-2996, National Bureau of Standards, Jan. 1985.

[7] Clark, D.R., and May, W.B., Jr., 'HVACSIM$^+$ Building Systems and Equipment Simuation Program Users Guide,' Draft NBSIR (to be published), National Bureau of Standards.

[8] Kusuda, T., 'Thermal Response Factors for Multilayer Structures of Various Heat Conduction Systems,' ASHRAE Trans., Vol. 75, 1969.

[9] Walton, G.N., 'Thermal Analysis Research Program Reference Manual,' NBSIR 83-2655, National Bureau of Standards, March 1983.

[10] NBS 'Guide to Available Mathematical Software (GAMS),' Center for Applied Math., National Bureau of Standards, Oct. 1981.

[11] Hiebert, K.L., 'An Evaluation of Mathematical Software that Solves Systems of Nonlinear Equations,' ACM Trans. Math. Software, Vol. 8, No. 1, March 1982, pp. 5-20.

[12] Powell, M.J.D., 'A Hybrid Method for Nonlinear Equations in Numerical Methods for Nonlinear Algebraic Equations,' P. Rabinowitz, Ed., Gordon and Breach, London, 1970.

[13] Brayton, R.K., Gustavson, F.G., and Hachtel, G.D., 'A New Efficient Algorithm for Solving Differential - Algebraic Systems Using Implicit Backward Differential Formulas,' Proc. IEEE, Vol. 60, No. 1, Jan 1972, pp. 98-108.

[14] Gear, C.W., 'The Automatic Integration of Ordinary Differential Equations,' Comm. ACM, Vol. 14, March 1971, pp. 176-179.

[15] Chua, L.O., and Lin, P., Computer-Aided Analysis of Electronic Circuits, Prentice-Hall, 1975.

[16] Kusuda, T., 'NBSLD, The computer Program for Heating and Cooling Loads in Buildings,' BSS 69, National Bureau of Standards, July 1976.

[17] Kusuda, T. and Saitoh, T., 'Simplified Analysis Calculations for Residential Applications,' NBSIR 80-1961, National Bureau of Standards, July 1980.

[18] Building Environment Division of NBS, 'Technical Guidelines for Energy Conservation,' AFCEC-TR-77-12 and NBSIR 77-1238, National Bureau of Standards, June 1977

[19] Walton, G.N., 'A New Algorithm for Radiant Interchange in Room Loads Calculations,' ASHRAE Trans., Vol. 86, Part 2, 1980.