B. The Control Function

There are at least two further reasons for using the Flux Coupler code to control integrations of the NCAR Climate System Model. The first is that, with communication between components by message–passing, the various model components can be running on different computers. This has already been demonstrated to work by members of the NCAR Scientific Computing Division. A short coupled integration was performed with the ocean component running at the San Diego Computing Center and the remaining model components running on a CRAY at NCAR. The second reason is that model components can be switched in and out of the CSM relatively simply when the Flux Coupler and component models are stand–alone, independent executable programs. However, for a new component to work within the present CSM configuration, it must satisfy a number of requirements. These relate to model initialization, what interface fields and/or fluxes it sends to the coupler (see section C), what fluxes and/or fields it receives from the coupler (see section C), and model shut down.

The Flux Coupler is a separate executable program that will initialize itself and then wait to exchange initial messages from each of the component models. It then sends a brief message to each of the component models about the run set up before waiting for each component model to send its initial state and any output fluxes. The coupler will pass messages with all the component models via four routines for each component model. For example, for the atmospheric component, these are: msg-atmi, msg-atmr, msg-atms, and msg-atmf. These subroutines will in turn invoke a message-passing technique to implement the actual data transfer.

The atmospheric model is a separate executable with the following functionality:

1) it will initialize itself, will receive from the coupler a brief message about the run set up, and then provide its spatial grid and some time information to the coupler (sent and received by the coupler in routine msg-atmi),
2) it will provide its initial model state and output fluxes to the coupler (received by the coupler in routine msg-atmr),
3) it will wait to receive input fluxes; when input fluxes are received (sent by the coupler in routine msg-atms), advance a specified number of steps, and then provide its updated model state and output fluxes to the coupler (again, received by the coupler in routine msg-atmr),
4) repeat (3) until told to stop (a stop message is sent by the coupler in routine msg-atmf).

The land, ocean, and sea–ice models will interact with the coupler in a similar fashion, although they may advance a different number of time steps before communicating with the coupler. Further details of the time control by the coupler are given in section D.

The component models communicate via message passing to the coupler. This readily facilitates coarse parallelization, but has some negative side–effects. All models have output fluxes (*i.e.,* internally computed fluxes), such as solar radiation within the atmosphere model, ice formation within the ocean, ice to ocean solar radiation within the ice model, and ocean runoff within the land model. Because the component models advance in parallel, an output flux of one model cannot be the input flux for another model during the same time interval. The best that can be done is for the coupler to provide these output fluxes to the other models at the relevant synchronization periods. Thus output flux fields will be somewhat uncoordinated in time, *i.e.,* fluxed quantities will not be conserved during any one synchronization interval. This has a further consequence in that these lagged fluxes will be needed to accurately restart and continue a simulation so the coupler must create a restart file which contains these lagged output fluxes.