```
    tmp_endpt = mcapi_create_anonymous_endpoint(&err);
    CHECK_STATUS(err);

    // send the shmem handle
    mcapi_msg_send(tmp_endpt, tpu_rmem_endpt, sMem,
                   sizeof(sMem), &err);
    CHECK_STATUS(err);

    // connect the channels
    mcapi_connect_pktchan_i(sig_endpt, sig_rmem_endpt,
                            &r1, &err);
    CHECK_STATUS(err);

    // wait on the connection
    while (!mcapi_test(&r1,NULL,&err)) {
        // KEEP WAITING
    }

    // now open the channels
    mcapi_open_pktchan_recv_i(&sig_chan, sig_endpt,
                              &r1, &err);
    CHECK_STATUS(err);

    // wait on the channels
    while (!(mcapi_test(&r1,NULL,&err)) {
        // KEEP WAITING
    }

    // now ALL of the bootstrapping is finished
    // we move to the processing phase below

    while (1) {
       // NOTE – get an MRAPI lock
       mrapi_mutex_lock (sMem_mutex, &lock_key, 0,
                         &mrapi_status);
       CHECK_STATUS(mrapi_status);

       // read the shared memory
       if (sPtr[0] != 0) {
          // process the shared memory data
       } else {
          // PANIC -- error with the shared mem
      }

      // NOTE – release the MRAPI lock
      mrapi_mutex_unlock(sMem_mutex, &lock_key,
                         &mrapi_status);
      CHECK_STATUS(mrapi_status);

      // now get data from the signal processing task
      // would be a loop if there were multiple sig tasks
      mcapi_pktchan_recv(sig_chan, (void **) &sDat,
                         &tSize, &err);
      CHECK_STATUS(err);

     // Compute new carb params & update carb
    }
}
```