6.2.8 MRAPI pseudo-code

6.2.8.1 Initial Mapping

```
////////////////////////////////////////
// The control task
////////////////////////////////////////
void Control_Task(void) {
    mrapi_shmem_hndl_t sMem;   /* handle to the shmem */
    mrapi_mutex_hndl_t sMem_mutex;
    char* sPtr;
    mrapi_key_t lock_key;
    uint8_t tFlag;
    mcapi_endpoint_t tpu_rmem_endpt;
    mcapi_endpoint_t sig_endpt, sig_rmem_endpt;
    mcapi_endpoint_t tmp_endpt;
    mcapi_pktchan_recv_hndl_t sig_chan;
    struct SIG_DATA sDat;
    size_t tSize;
    mcapi_request_t r1, r2;
    mcapi_status_t err;
    mrapi_status_t mrapi_status;
    mrapi_parameters_t parms;
    mrapi_info_t version;

    // init the system
    mcapi_initialize(CNTRL_NODE, &err);
    CHECK_STATUS(err);

    mrapi_initialize(AUTO_USE_CASE_DOMAIN_ID, CNTRL_NODE,
                     parms,&version, &mrapi_status);
    CHECK_STATUS(mrapi_status);

    // first create local endpoints
    sig_endpt = mcapi_create_endpoint(CNTRL_PORT_SIG,
                                      &err);
    CHECK_STATUS(err);

    // now we get two rmem endpoints
    mcapi_get_endpoint_i(TPU_NODE, TPU_PORT_CNTRL,
                         &tpu_rmem_endpt, &r1, &err);
    CHECK_STATUS(err);

    mcapi_get_endpoint(SIG_NODE, SIG_PORT_CNTRL,
                       &sig_rmem_endpt, &r2, &err);
    CHECK_STATUS(err);

    // wait on the endpoints
    while (!((mcapi_test(&r1,NULL,&err)) &&
             (mcapi_test(&r2,NULL,&err))) {
        // KEEP WAITING
    }

    // create our mutex for the shared memory region
    sMem_mutex =
        mrapi_mutex_create(SMEM_MUTEX_ID, MRAPI_NULL,
                           &mrapi_status);
    CHECK_STATUS(mrapi_status);

    // allocate shmem and send the handle to TPU task
    sMem = mrapi_shmem_create(SHMEM_ID, SHMEM_SIZE,
                              MRAPI_NULL, 0, MRAPI_NULL,
                              0, &mrapi_status);
    CHECK_STATUS(mrapi_status);

    sPtr = (void*) mrapi_shmem_attach(sMem,&mrapi_status);
    CHECK_STATUS(mrapi_status);
```