```
      CHECK_STATUS(status);

      command_endpoint = mcapi_endpoint_create(PORT_COMMAND, & status);
      CHECK_STATUS(status);

      data_endpoint = mcapi_endpoint_create(PORT_DATA, & status);
      CHECK_STATUS(status);

      data_recv_endpoint = mcapi_endpoint_create(PORT_DATA_RECV, & status);
      CHECK_STATUS(status);

      remote_command_endpoint = mcapi_endpoint_get(DOMAIN_0, DSP_1,
PORT_COMMAND, &status);
      CHECK_STATUS(status);

      remote_data_endpoint = mcapi_endpoint_get(DOMAIN_0, DSP_1, PORT_DATA,
&status);
      CHECK_STATUS(status);

      remote_data_recv_endpoint = mcapi_endpoint_get(DOMAIN_0, DSP_1,
PORT_DATA_RECV,
                                                     &status);
      CHECK_STATUS(status);

      mcapi_pktchan_connect_i(data_endpoint, remote_data_endpoint, &request,
                        &status);
      CHECK_STATUS(status);

      mcapi_sclchan_connect_i(command_endpoint, remote_command_endpoint,
                         &request, &status)
      CHECK_STATUS(status);

      mcapi_pktchan_connect_i(data_recv_endpoint, remote_data_recv_endpoint,
                        &request, &status);
      CHECK_STATUS(status);

      mcapi_pktchan_send_open_i(&data_chan, data_endpoint, &request,
&status);
      CHECK_STATUS(status);

      mcapi_sclchan_send_open_i(&command_chan, command_endpoint, &request,
                           &status);
      CHECK_STATUS(status);

      mcapi_pktchan_recv_open_i(&data_recv_chan, data_recv_endpoint,
&request,
                           &status);
  CHECK_STATUS(status);

}

void send_data(void *data, int size)
{
      mcapi_status_t status;
      mcapi_pktchan_send(data_chan, data, size, &status);
      CHECK_STATUS(status);
}

void send_dsp_cmd(int cmd)
{
      mcapi_status_t status;
```