Message Format

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Byte | N | | N+1 | |
| 1-4 | Message Number | | | |
| 5-8 | Message Length | | | |
| 9-12 | TTS | | | |
| 13-6 | TTF | | | |
| 17-20 | Reserved | | | |
| 21-24 | OM | | | |
| 25-28 | ATR | | | |
| 29-32 | Reserved | ATM | | ATS |
| 33-36 | SM | | | Reserved |

Legacy Code

1. MessageDefine

Struct GfeMsgHeader

{

UINT32 messageNum;

UINT32 messageLength

}

Struct GfeMsgStatus

{

HmsMsgHeader hmsMsgHeader;

UINT32 TTS;

UINT32 TTF;

UINT32 Reserved1;

UINT32 OM;

UINT32 ATR

UINT8 Reserved2;

UINT8 ATM;

UINT16 ATS;

UINT16 SM;

UINT16 Reserved3;

}

void CGfeManager::ProcessGfeMsg()

{

If( m\_flagHeaderExist == false)

{

If(m\_RingBuffer.occupiedRingBuffer() >= MSG\_HEADER\_LENGTH)

{  
 m\_RingBuffer.getRingBuffer((char\*)recvBuffer, HMS\_MSG\_HEADER\_LENGTH);

m\_flagHeaderExist = true

memcpy((char\*)m\_stMsgHeader, recvBuffer, sizeof(MsgHeader));

if(!m\_stMsgHeader.isValid())

{

Int iRemainedBytes = m\_RingBuffer.occupiedRingBuffer();

If(iRemainedBytes > 0)

{

m\_RingBuffer.GetRingBuffer((char\*)recvBuffer+sizeof(GfeMsgHeader), iRemaindBytes)

m\_RingBuffer.flushBuffer();}

m\_flagHeaderExist = false;

break;

}

}

Int iRecvMsgBodyLength = m\_stGfeMsgHeader.messageLength;

If( (iRecvMsgBodyLength >0) && (iRecvMsgBodyLength <= m\_RingBuffer.occupiedRingBuffer())

{

m\_RingBuffer.getRingBuffer((char\*)recvBuffer+sizeof(GfeMsgHeader), iRecvMsgBodyLength);

m\_flagHeaderExist = false;

}

GfeMsgHeader header;

Memset(&header, 0x00, sizeof(GfeMsgHeader));

Memcpy(&header, recvbuffer, sizeof(GfeMsgHeader));

Switch(header.messageNumber)

{

Case MSG\_NUM\_GFE\_STATUS:

GfeMsgStatus status;

Memset(&status, 0x00, sizeof(GfeMsgStatus));

Memcpy(&status, recvbuffer, sizeof(GfeMsgStatus));

//status.변수이름으로 접근 상태 데이터 처리

}

DDS Code

DataReaderListenerIml::on\_data\_available(DDS::DataReader\_ptr reader)

{

char \* topicName = reader->get\_topicdescription()-get\_name();

If(strcmp( topicName. “GfeStatus) == 0)

{  
 AS::DDSDataTypes::GfeStatusDataReader\_var reader\_i = AS::DDSDataTypes::GfeStatusDataReader::\_narrow(reader);

DDS::ReturnCode\_t error = reader\_i->take\_next\_sample(m\_message, m\_info);

If( error = DDS::RETCODE\_OK)

{  
 if( info.valid\_data)

{

Message.변수이름으로 접근 상태데이터 처리

}

}

}