



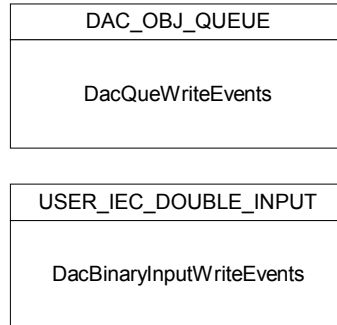
Applied Systems Engineering, Inc.

Technical Note 26

Processing System Queue Writes

Technical Note 26: System Queue Processing

When the IECPT is running in master translation mode with the system queue property true, all events are written to the system queue. The type of event written to the system queue (DAC object DAC_OBJ_QUEUE) is contained in the structure QUEUEINPUTEVENT. QUEUEINPUTEVENT is the union of all possible events supported by the GPT. The note will show how the user can route writes to the system queue to specific protocol object write events handlers. Assume the following configuration:



The device profile contains two objects. The DAC queue object DAC_OBJ_QUEUE and the IEC double input object USER_IEC_DOUBLE_INPUT. Both of these objects support a method for handling GPT event writes. DAC_OBJ_QUEUE uses the method DacQueWriteEvents, while the USER_IEC_DOUBLE_INPUT object uses the method DacBinaryInputWriteEvents. The following code sample illustrates how writes to system queue DacQueWriteEvents can be redirected to the method DacBinaryInputWriteEvents.

```
Routine DacQueWriteEvents(,pEnv,objId,count, LPQUEUEINPUTEVENT p )
{
    DACOBJREF    i;
    LPDACOBJECT  op;

    for(i=0;i<count;i++,p++)
    {
        /* Each QUEUEINPUTEVENT identified the object the event */
        /* Belongs to */
        switch( p->ObjId )
        {
            case USER_IEC_DOUBLE_INPUT :

                /* Get Pointer to definition */
                DacDbPtr( pEnv, p->ObjId, &op )

                /* Call object specific write events handler */
                /* &p->dacpnt is of type LPBINARYINPUTEVENT, */
                /* This will call DacWriteBinaryInputEvents */
                (*op->PutEvent)( pEnv, objId, 1, &p->dacpnt );
                break;
        }
    }
}
```

Technical Note 26: Processing System Queue Event Writes

```
        default :
            /* Do something else */
        }
    }
    return i;
}
```

The above example routed a specific type of event to an object specific handler. We can all route classes of events as in the following example.

```
Routine DacQueWriteEvents(,pEnv,objId,count, LPQUEUEINPUTEVENT p )
{
    DACOBJREF i;
    LPDACOBJECT op;

    for(i=0;i<count;i++,p++)
    {
        /* Each QUEUEINPUTEVENT identified the object the event */
        /* Belongs to */
        /* Get Pointer to definition */
        DacDbPtr( pEnv, p->ObjId, &op );

        switch( op->Class )
        {
            case CLASS_BINARY_INPUT :

                /* Call object specific write events handler */
                /* &p->dacpnt is of type LPBINARYINPUTEVENT, */
                /* This will call DacWriteBinaryInputEvents */
                /* This will be called for single, double, */
                /* bitstring inputs */
                (*op->PutEvent)(pEnv,objId,1,&p->dacpnt);
                break;

            default :

                /* Do something else */

        }
    }
    return i;
}
```