C. Reading the attribute value with (deprecated) Fortran MPI-1 calls

```
LOGICAL FLAG
INTEGER IERR, GET_VAL, GET_STRUCT

! Upon successful return, GET_VAL == &set_val, possibly truncated
CALL MPI_ATTR_GET(MPI_COMM_WORLD, KEYVAL1, GET_VAL, FLAG, IERR)
! Upon successful return, GET_STRUCT == &set_struct, possibly truncated
CALL MPI_ATTR_GET(MPI_COMM_WORLD, KEYVAL2, GET_STRUCT, FLAG, IERR)
! Upon successful return, GET_VAL == 17
CALL MPI_ATTR_GET(MPI_COMM_WORLD, KEYVAL3, GET_VAL, FLAG, IERR)
```

D. Reading the attribute value with Fortran MPI-2 calls

```
LOGICAL FLAG
INTEGER IERR
INTEGER (KIND=MPI_ADDRESS_KIND) GET_VAL, GET_STRUCT

! Upon successful return, GET_VAL == &set_val
CALL MPI_COMM_GET_ATTR(MPI_COMM_WORLD, KEYVAL1, GET_VAL, FLAG, IERR)
! Upon successful return, GET_STRUCT == &set_struct
CALL MPI_COMM_GET_ATTR(MPI_COMM_WORLD, KEYVAL2, GET_STRUCT, FLAG, IERR)
! Upon successful return, GET_VAL == 17
CALL MPI_COMM_GET_ATTR(MPI_COMM_WORLD, KEYVAL3, GET_VAL, FLAG, IERR)
```

Example 16.18 A. Setting an attribute value with the (deprecated) Fortran MPI-1 call

```
INTEGER IERR, VAL
VAL = 7
CALL MPI_ATTR_PUT(MPI_COMM_WORLD, KEYVAL, VAL, IERR)
```

B. Reading the attribute value in C

```
int flag;
int *value;

/* Upon successful return, value points to internal MPI storage and
   *value == (int) 7 */
MPI_Comm_get_attr(MPI_COMM_WORLD, keyval, &value, &flag);
```

C. Reading the attribute value with (deprecated) Fortran MPI-1 calls

```
LOGICAL FLAG
INTEGER IERR, VALUE

! Upon successful return, VALUE == 7
CALL MPI_ATTR_GET(MPI_COMM_WORLD, KEYVAL, VALUE, FLAG, IERR)
```

D. Reading the attribute value with Fortran MPI-2 calls

```
LOGICAL FLAG
INTEGER IERR
INTEGER (KIND=MPI_ADDRESS_KIND) VALUE

! Upon successful return, VALUE == 7 (sign extended)
CALL MPI_COMM_GET_ATTR(MPI_COMM_WORLD, KEYVAL, VALUE, FLAG, IERR)
```

Example 16.19 A. Setting an attribute value via a Fortran MPI-2 call

```
INTEGER IERR
INTEGER(KIND=MPI_ADDRESS_KIND) VALUE1
INTEGER(KIND=MPI_ADDRESS_KIND) VALUE2
VALUE1 = 42
VALUE2 = INT(2, KIND=MPI_ADDRESS_KIND) ** 40

CALL MPI_COMM_SET_ATTR(MPI_COMM_WORLD, KEYVAL1, VALUE1, IERR)
CALL MPI_COMM_SET_ATTR(MPI_COMM_WORLD, KEYVAL2, VALUE2, IERR)
```

B. Reading the attribute value in C

```
int flag;
MPI_Aint *value1, *value2;

/* Upon successful return, value1 points to internal MPI storage and
   *value1 == 42 */
MPI_Comm_get_attr(MPI_COMM_WORLD, keyval1, &value1, &flag);
/* Upon successful return, value2 points to internal MPI storage and
   *value2 == 2^40 */
MPI_Comm_get_attr(MPI_COMM_WORLD, keyval2, &value2, &flag);
```

C. Reading the attribute value with (deprecated) Fortran MPI-1 calls

```
LOGICAL FLAG
INTEGER IERR, VALUE1, VALUE2

! Upon successful return, VALUE1 == 42
CALL MPI_ATTR_GET(MPI_COMM_WORLD, KEYVAL1, VALUE1, FLAG, IERR)
! Upon successful return, VALUE2 == 2^40, or 0 if truncation
! needed (i.e., the least significant part of the attribute word)
CALL MPI_ATTR_GET(MPI_COMM_WORLD, KEYVAL2, VALUE2, FLAG, IERR)
```

D. Reading the attribute value with Fortran MPI-2 calls

```
LOGICAL FLAG
INTEGER IERR
INTEGER (KIND=MPI_ADDRESS_KIND) VALUE1, VALUE2

! Upon successful return, VALUE1 == 42
CALL MPI_COMM_GET_ATTR(MPI_COMM_WORLD, KEYVAL1, VALUE1, FLAG, IERR)
! Upon successful return, VALUE2 == 2^40
CALL MPI_COMM_GET_ATTR(MPI_COMM_WORLD, KEYVAL2, VALUE2, FLAG, IERR)
```