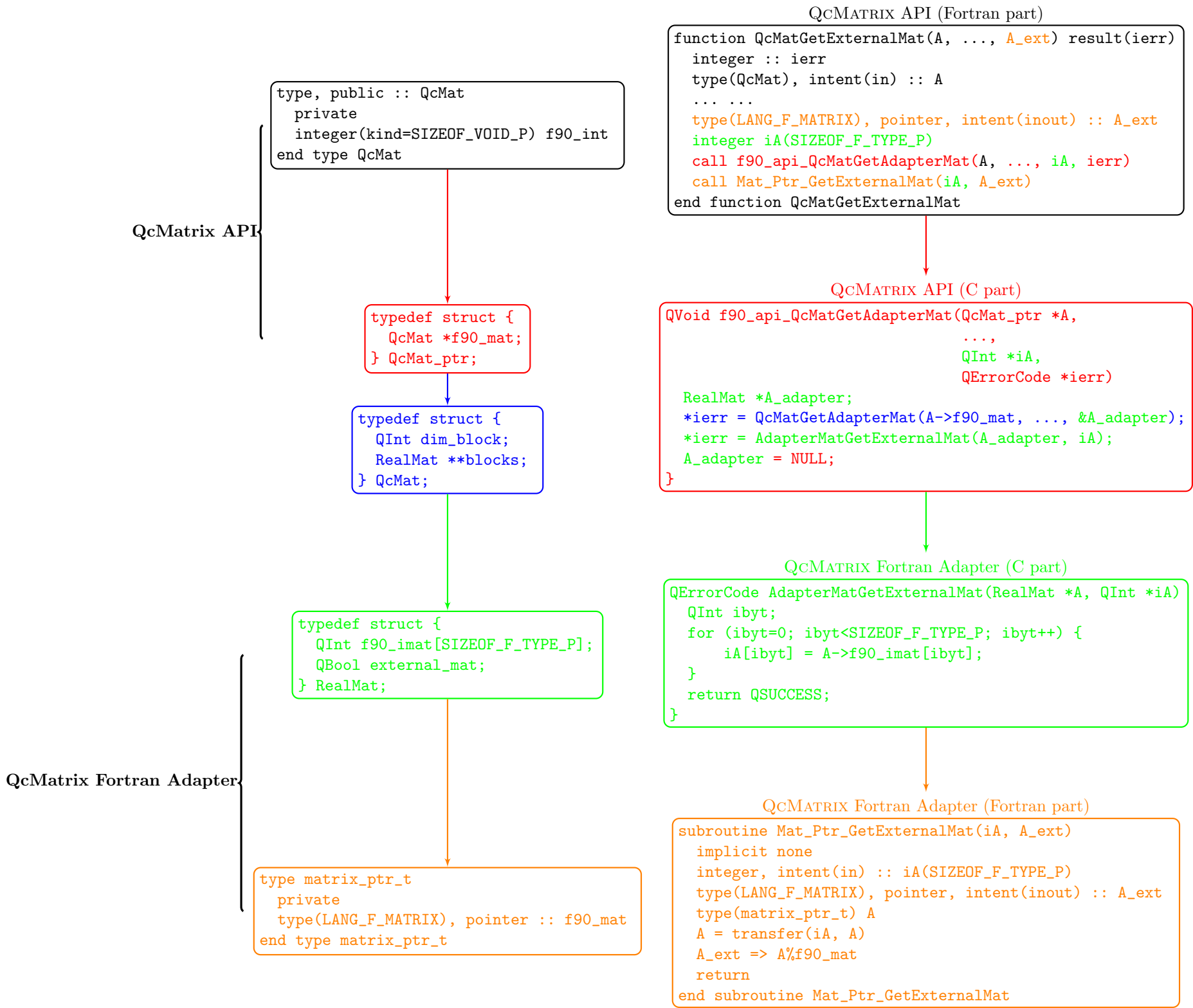
QcMatrix API (Fortran part)
function QcMatGetExternalMat(A, ..., A_ext) result(ierr)
integer :: ierr
type(QcMat), intent(in) :: A
... ...
type(LANG_F_MATRIX), pointer, intent(inout) :: A_ext
integer iA(SIZEOF_F_TYPE_P)
call f90_api_QcMatGetAdapterMat(A, ..., iA, ierr)
call Mat_Ptr_GetExternalMat(iA, A_ext)
end function QcMatGetExternalMat
QcMatrix API
type, public :: QcMat
private
integer(kind=SIZEOF_VOID_P) f90_int
end type QcMat
QcMatrix API (C part)
typedef struct {
QcMat *f90_mat;
} QcMat_ptr;
QVoid f90_api_QcMatGetAdapterMat(QcMat_ptr *A,
...,
QInt *iA,
QErrorCode *ierr)
RealMat *A_adapter;
*ierr = QcMatGetAdapterMat(A->f90_mat, ..., &A_adapter);
*ierr = AdapterMatGetExternalMat(A_adapter, iA);
A_adapter = NULL;
}
typedef struct {
QInt dim_block;
RealMat **blocks;
} QcMat;
QcMatrix Fortran Adapter (C part)
QErrorCode AdapterMatGetExternalMat(RealMat *A, QInt *iA)
QInt ibyt;
for (ibyt=0; ibyt<SIZEOF_F_TYPE_P; ibyt++) {
iA[ibyt] = A->f90_imat[ibyt];
}
return QSUCCESS;
}
typedef struct {
QInt f90_imat[SIZEOF_F_TYPE_P];
QBool external_mat;
} RealMat;
QcMatrix Fortran Adapter
QcMatrix Fortran Adapter (Fortran part)
subroutine Mat_Ptr_GetExternalMat(iA, A_ext)
implicit none
integer, intent(in) :: iA(SIZEOF_F_TYPE_P)
type(LANG_F_MATRIX), pointer, intent(inout) :: A_ext
type(matrix_ptr_t) A
A = transfer(iA, A)
A_ext => A%f90_mat
return
end subroutine Mat_Ptr_GetExternalMat
type matrix_ptr_t
private
type(LANG_F_MATRIX), pointer :: f90_mat
end type matrix_ptr_t