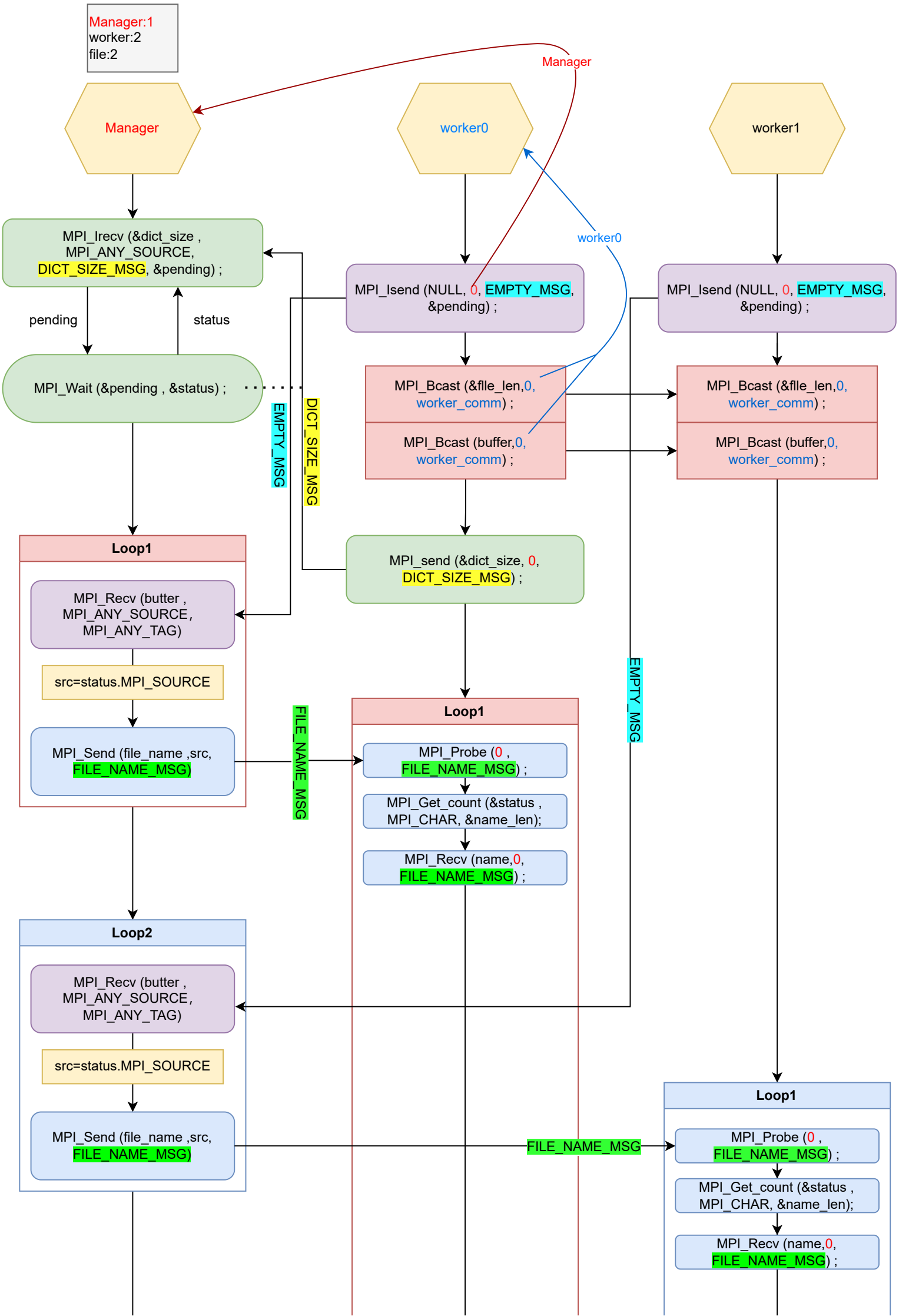
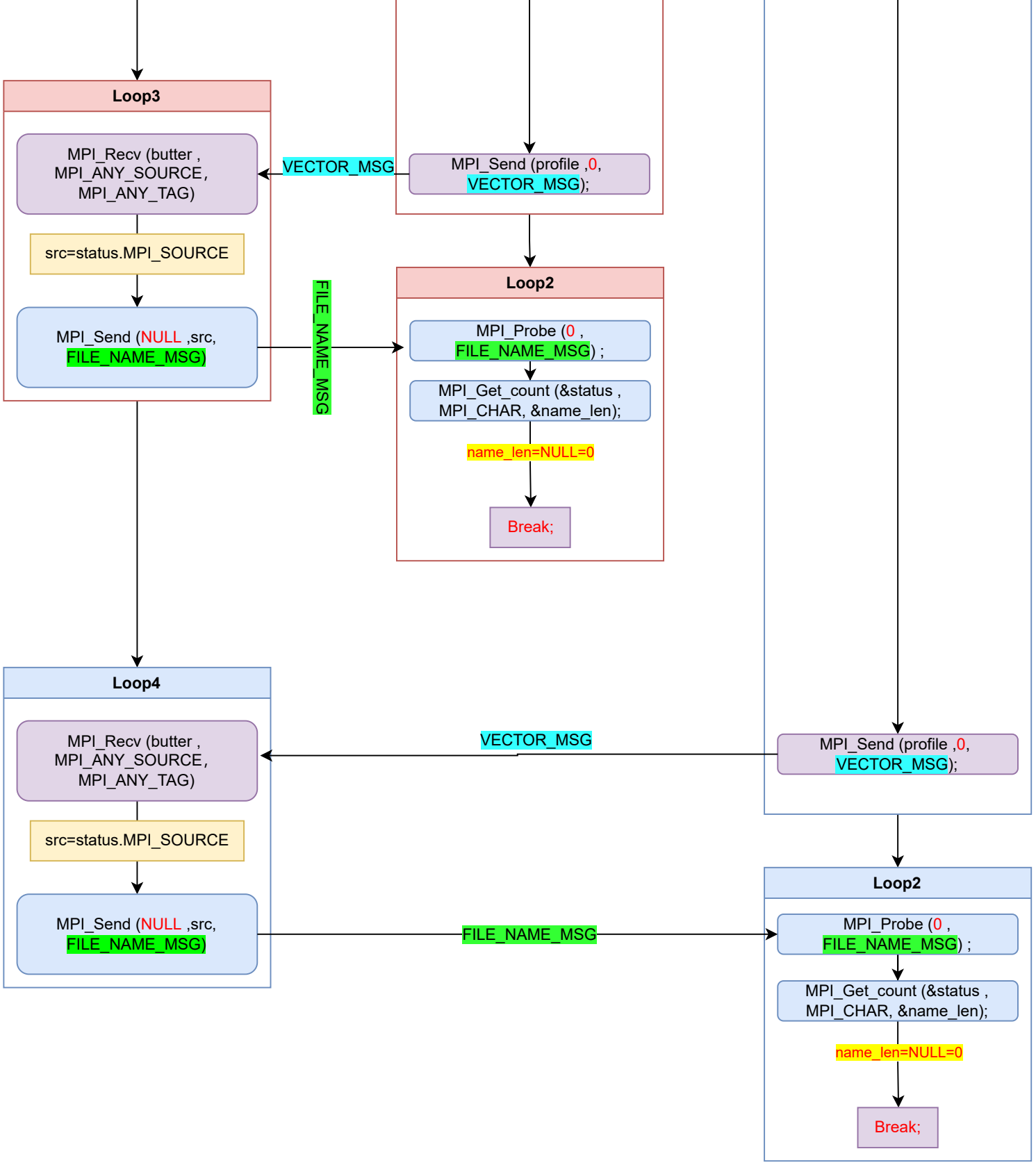
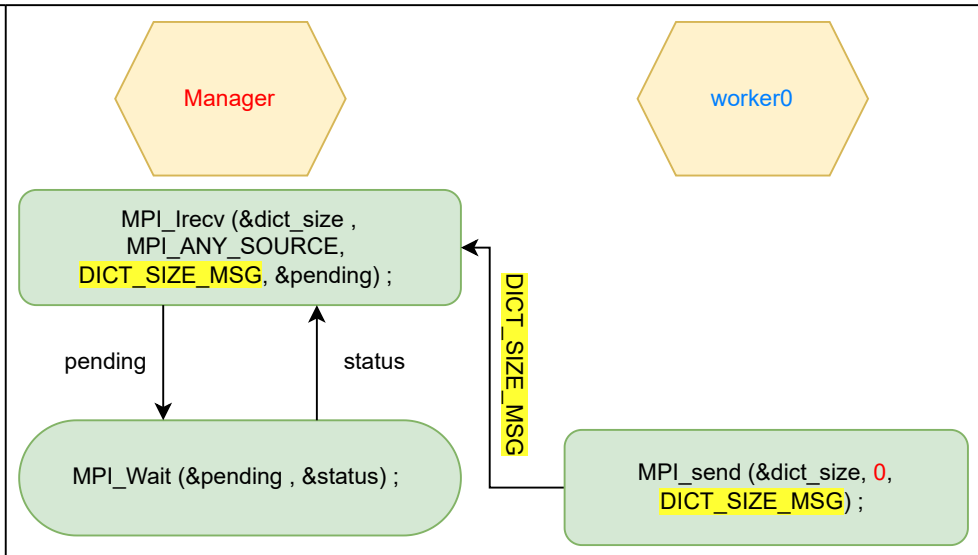

Manager:1
worker:2
file:2
Manager
Manager
worker0
worker1
worker0
MPI_Irecv (&dict_size , MPI_ANY_SOURCE, DICT_SIZE_MSG, &pending) ;
MPI_Isend (NULL, 0, EMPTY_MSG, &pending) ;
MPI_Isend (NULL, 0, EMPTY_MSG, &pending) ;
pending
status
MPI_Wait (&pending , &status) ;
MPI_Bcast (&flle_len,0, worker_comm) ;
MPI_Bcast (buffer,0, worker_comm) ;
MPI_Bcast (&flle_len,0, worker_comm) ;
MPI_Bcast (buffer,0, worker_comm) ;
EMPTY_MSG
DICT_SIZE_MSG
Loop1
MPI_Recv (butter , MPI_ANY_SOURCE, MPI_ANY_TAG)
src=status.MPI_SOURCE
MPI_Send (file_name ,src, FILE_NAME_MSG)
MPI_send (&dict_size, 0, DICT_SIZE_MSG) ;
FILE_NAME_MSG
Loop1
MPI_Probe (0 , FILE_NAME_MSG) ;
MPI_Get_count (&status , MPI_CHAR, &name_len);
MPI_Recv (name,0, FILE_NAME_MSG) ;
EMPTY_MSG
Loop2
MPI_Recv (butter , MPI_ANY_SOURCE, MPI_ANY_TAG)
src=status.MPI_SOURCE
MPI_Send (file_name ,src, FILE_NAME_MSG)
FILE_NAME_MSG
Loop1
MPI_Probe (0 , FILE_NAME_MSG) ;
MPI_Get_count (&status , MPI_CHAR, &name_len);
MPI_Recv (name,0, FILE_NAME_MSG) ;


Loop3
MPI_Recv (butter , MPI_ANY_SOURCE, MPI_ANY_TAG)
src=status.MPI_SOURCE
MPI_Send (NULL ,src, FILE_NAME_MSG)
VECTOR_MSG
MPI_Send (profile ,0, VECTOR_MSG);
FILE_NAME_MSG
Loop2
MPI_Probe (0 , FILE_NAME_MSG) ;
MPI_Get_count (&status , MPI_CHAR, &name_len);
name_len=NULL=0
Break;
Loop4
MPI_Recv (butter , MPI_ANY_SOURCE, MPI_ANY_TAG)
src=status.MPI_SOURCE
MPI_Send (NULL ,src, FILE_NAME_MSG)
VECTOR_MSG
MPI_Send (profile ,0, VECTOR_MSG);
FILE_NAME_MSG
Loop2
MPI_Probe (0 , FILE_NAME_MSG) ;
MPI_Get_count (&status , MPI_CHAR, &name_len);
name_len=NULL=0
Break;

Green

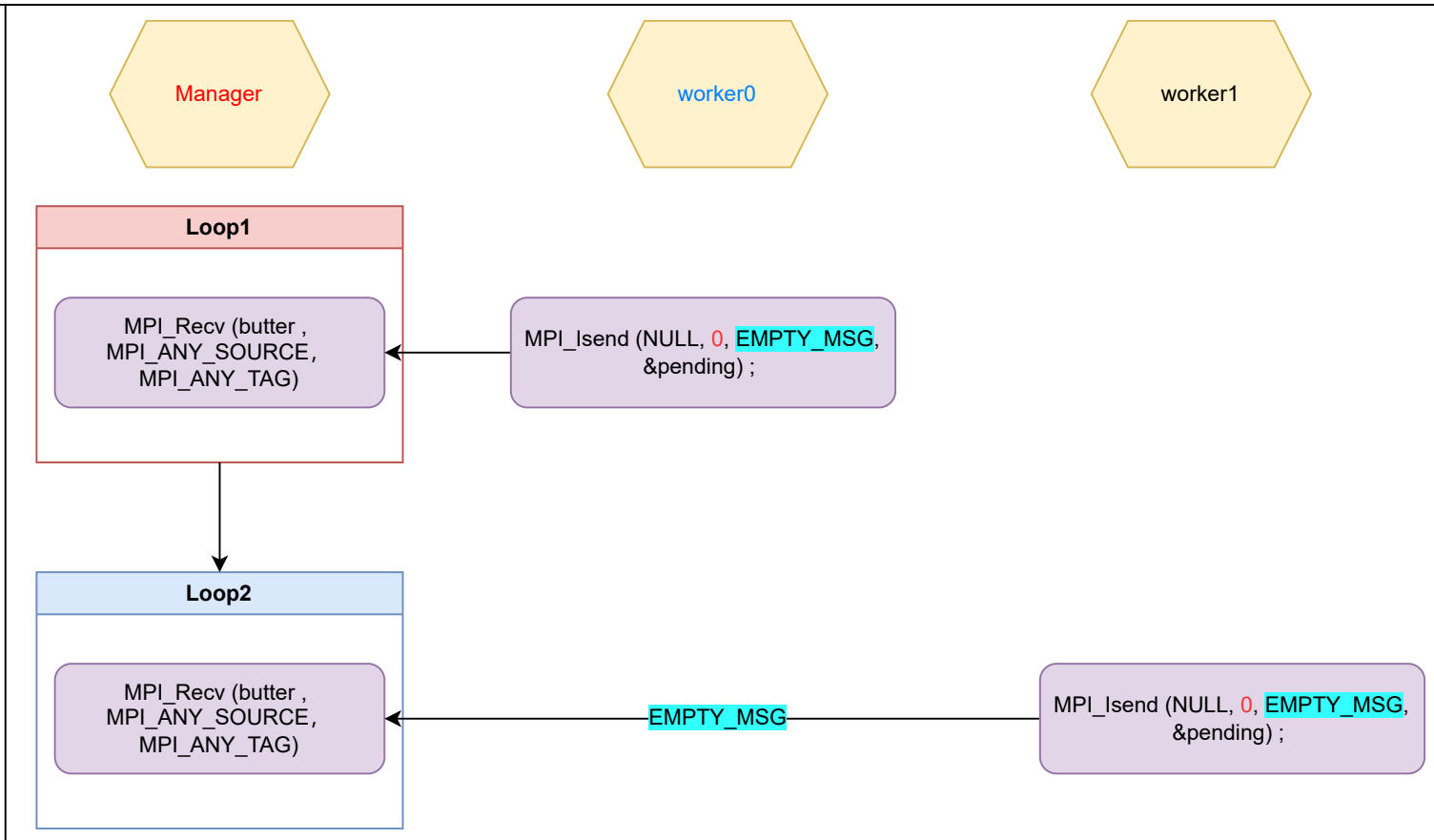



**綠色的部分:**

worker0使用MPI_send透過DICT_SIZE_MSG通道將字典大小傳送給manager，而manager也透過MPI_Irecv從DICT_SIZE_MSG通道接收字典大小的訊息，並且使用MPI_Wait來確保訊息有存入&dict_size變數中。

而manager使用non-block的Recv來接收訊息則可以在發出請求後，繼續執行其他步驟，直到需要這筆資料時再使用MPI_Wait來確保資料已經到達。

Purple

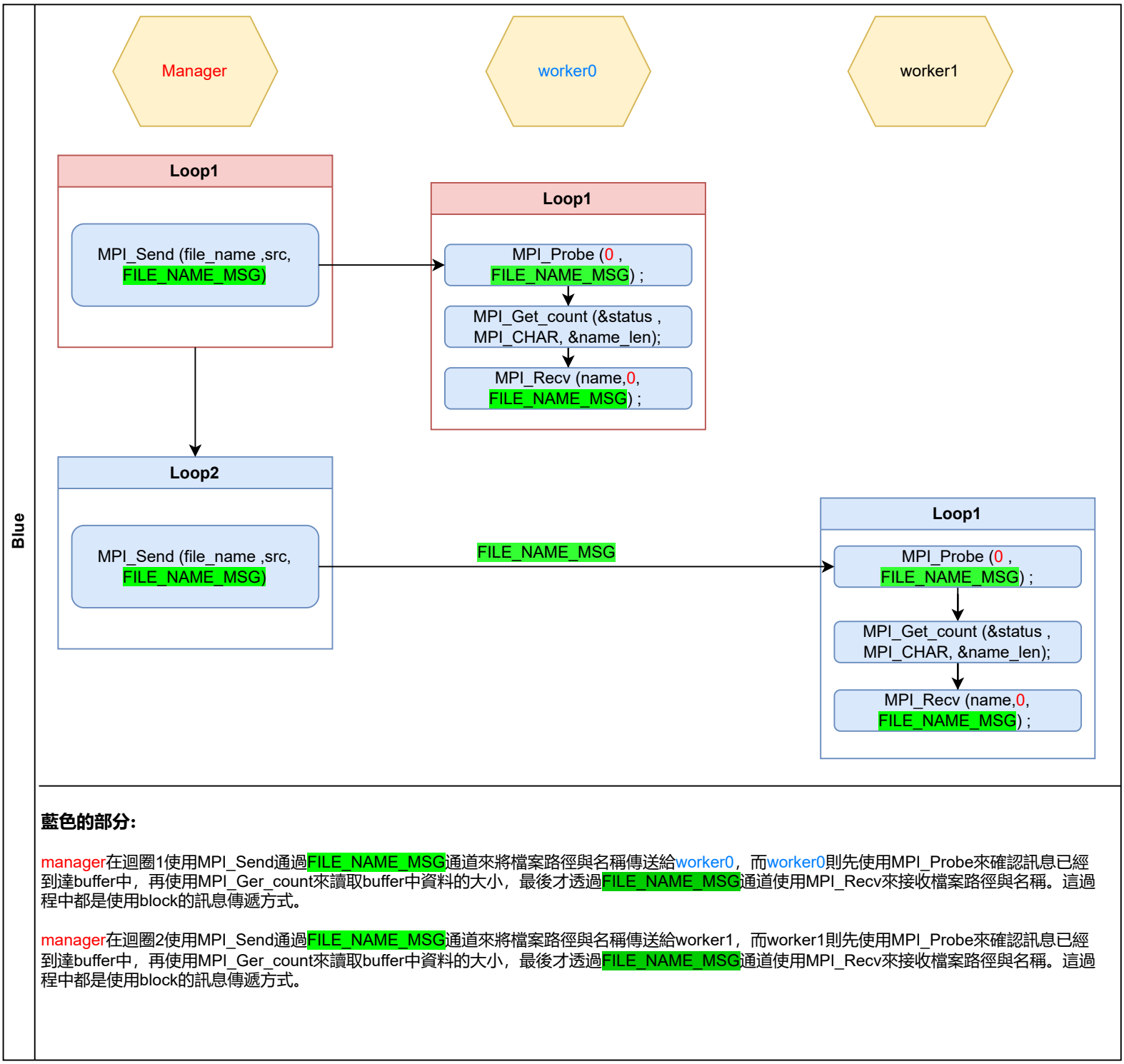



**紫色的部分:**

worker0使用MPI_Isend透過EMPTY_MSG通道告知manager目前手上並沒有工作之後繼續執行自己下一步程序，而manager則在迴圈1中使用MPI_Recv透過MPI_ANY_TAG通道來接收這項訊息。而MPI_Recv為block的所以可以確保訊息有被送達。

worker1也使用MPI_Isend透過EMPTY_MSG通道告知manager目前手上並沒有工作之後繼續執行自己下一步程序，而manager則在迴圈2中使用MPI_Recv透過MPI_ANY_TAG通道來接收這項訊息。而MPI_Recv為block的所以可以確保訊息有被送達。

而worker使用non-block的send來告知manager自己目前的狀態，發出請求後則可以開始處理像是MPI_Bcast等等步驟，而不用等待manager接收到訊息，可以讓溝通與運算同時進行來省下時間。




**藍色的部分:**

manager在迴圈1使用MPI_Send通過FILE_NAME_MSG通道來將檔案路徑與名稱傳送給worker0，而worker0則先使用MPI_Probe來確認訊息已經到達buffer中，再使用MPI_Ger_count來讀取buffer中資料的大小，最後才透過FILE_NAME_MSG通道使用MPI_Recv來接收檔案路徑與名稱。這過程中都是使用block的訊息傳遞方式。

manager在迴圈2使用MPI_Send通過FILE_NAME_MSG通道來將檔案路徑與名稱傳送給worker1，而worker1則先使用MPI_Probe來確認訊息已經到達buffer中，再使用MPI_Ger_count來讀取buffer中資料的大小，最後才透過FILE_NAME_MSG通道使用MPI_Recv來接收檔案路徑與名稱。這過程中都是使用block的訊息傳遞方式。