Mavlink Interface Modes

1. Synchronous:

In this mode, each movement command will return only once it is completed. An example is below:

> move(angle=0, time=3)

> move(angle=90, time=3)

> move(angle=180, time=3)

> move(angle=270, time=3)

> print( getTemperature() ) # getTemperature is not a movement command

In this example, the first command executes and blocks until it returns upon completion after 3 seconds, allowing the second command to be run, which will block for 3 seconds, as will the third and the fourth. Once all four movements are complete, the current depth will be printed

2. Queuing:

In this asynchronous mode, a command returns directly after initiating. If, upon entering a movement command, there is not currently a movement command executing, this movement command will be immediately executed. If a movement command is currently executing, this movement command will be added to the command queue. The command queue will automatically execute the contained commands in the order added. An example is below.

> move(angle=0, time=3)

> move(angle=90, time=3)

> move(angle=180, time=3)

> move(angle=270, time=3)

> print( getTemperature() ) # getTemperature is not a movement command

In this example, upon running this series of commands, the first command will be called, immediately start executing, and return. The second command will then be added to the queue, and then cease blocking, as will the third and fourth. Then, while the first move command is still executing, the depth will be printed out. After the first command ends, the next three will execute in sequence.

3. ignore:

In this asynchronous mode, a command returns directly after initiating. If, upon entering a movement command, there is not currently a movement command executing, this movement command will be immediately executed. If a movement command is currently executing, this movement command will be ignored. An example is below.

> move(angle=0, time=3)

> move(angle=90, time=3)

> move(angle=180, time=3)

> move(angle=270, time=3)

> print( getTemperature() ) # getTemperature is not a movement command

In this example, upon running this series of commands, the first command will be called, immediately start executing, and return. The Second, third, and fourth commands will be ignored, as a movement command is already executing. Then, while the first move command is still executing, the depth will be printed out. The only movement to be taken will be the first.

4. Override:

In this asynchronous mode, a command returns directly after initiating. If, upon entering a movement command, there is not currently a movement command executing, this movement command will be immediately executed. If a movement command is currently executing, the currently executing movement command will be immediately terminated, and this command will be executed. An example is below.

> move(angle=0, time=3)

> move(angle=90, time=3)

> move(angle=180, time=3)

> move(angle=270, time=3)

> print( getTemperature() ) # getTemperature is not a movement command

In this example, upon running this series of commands, the first command will be called, immediately start executing, and return. The second command will then be called, terminating the first and executing itself. The second and third commands will do likewise. Then, while the last movement command is still executing, the depth will be printed out. The only movement command to be completed will be the last one, while the first three would be only partially executed (move only until the next command was able to terminate it).

Interactions between modes

An example is the easiest way to describe this:

> move(angle=0, time=10, execMode=queue)

> move(angle=180, time=10, execMode=queue)

> move(angle=0, time=10, execMode=queue)

> for i in range(10):

> log(getTemperature())

> sleep(1)

> move(angle=180, time=10, execMode=synchronous) # Using default execMode of Synchronous

In this example, the first move command will executed, and the next 2 put in the queue. The depth logging loop will then be run while the first command is executing. When the 4th move command is called (which will happen at about the start of the second command), it will wait until the CPU is free (until the entire queue has finished execution) to execute, blocking the entire time.

Example 2:

> move(0, 10, execMode=queue)

> move(180, 10, execMode=queue)

> move(270, 10, execMode=ignore)

> move(90, 10, execMode=override)

In this example, the first command immediately executes and the second is added to the queue. The third will be ignored, due to a currently executing command. When the 4th move command is called, it will (1): Clear the Queue, (2): stop the currently executing command, and (3) execute.

Example 3:

> move(0, 10, execMode=ignore) # or override

> move(180, 10, execMode=queue)

In this example, the first command will be executed, and the second command will be added to the queue.