at this point, they may have already been deleted in step **612** if the pressed subset contained more than two. Also, step **624** clears the temporal synchronization marker, indicating that no further synchronization tests need be done for this subset.

[**0256**] Continuing to FIG. **39**B, if the fingers synchronized during touchdown also lift simultaneously, step **618** removes them and any holds from the keypress queue in case they were a pair awaiting a positive release synchronization test. Further tests ensue to determine whether the synchronized fingers meet additional chord tap conditions. As with single finger taps, the synchronized fingers cannot be held on the surface more than about half a second if they are to qualify, as a chord tap. Decision diamond **626** tests this by thresholding the time between the release of the last remaining synchronized finger and the temporal press synchronization marker. A chord tap should also exhibit a limited amount of lateral finger motion, measured either as an average of peak finger speeds or distance traveled since touchdown in decision diamond **628**. If the quick release and limited lateral motion conditions are not met, step **624** clears the synchronization marker with the conclusion that the synchronized fingers were either just resting fingers or part of a chord slide.

[**0257**] If the chord tap conditions are met, step **630** looks up, using the synchronized subset bitfield, any input events such as mouse clicks or keyboard commands assigned to the combination of fingers in the chord tap. Some chords such as those including all four fingertips may be reserved as resting chords **634**, in which case decision diamond **632** will find they have no associated input events. If the chord does have tap input events, step **636** appends these to the main outgoing event queue of the host communication interface **20**. Finally step **624** clears the synchronization marker in readiness for future finger synchronizations on the given hand.

[**0258**] As a further precaution against accidental generation of chord taps while typing, it is also useful for decision diamond **632** to ignore through step **634** the first chord tap which comes soon after a valid keypress without a chord slide in between. Usually after typing the user will need to reposition the mouse cursor before clicking, requiring an intervening chord slide. If the mouse cursor happens to already be in place after typing, the user may have to tap the finger chord a second time for the click to be sent, but this is less risky than having an accidental chord tap cause an unintended mouse button click in the middle of a typing session.

[**0259**] FIG. **40**A shows the detailed steps of the chord motion recognizer module **18**. The chord motion recognition process described below is repeated for each hand independently. Step **650** retrieves the parameters of the hand's identified paths **250** and the hand's extracted motion components from the motion extraction module **16**. If a slide of a finger chord has not already started, decision diamond **652** orders slide initiation tests **654** and **656**. To distinguish slides from glancing finger taps during typing, decision diamond **654** requires at least two fingers from a hand to be touching the surface for slide mode to start. There may be some exceptions to this rule, such as allowing a single finger to resume a previous slide within a second or so after the previous slide chord lifts off the surface.

[**0260**] In a preferred embodiment, the user can start a slide and specify its chord in either of two ways. In the first way, the user starts with the hand floating above the surface, places some fingers on the surface possibly asynchronously, and begins moving all of these fingers laterally. Decision diamond **656** initiates the slide mode only when significant motion is detected in all the touching fingers. Step **658** selects the chord from the combination of fingers touching when significant motion is detected, regardless of touchdown synchronization. In this case coherent initiation of motion in all the touching fingers is sufficient to distinguish the slide from resting fingers, so synchronization of touchdown is not necessary. Also, novice users may erroneously try to start a slide by placing and sliding only one finger on the surface, forgetting that multiple fingers are necessary. Tolerance of asynchronous touchdown allows them to seamlessly correct this by subsequently placing and sliding the rest of the fingers desired for the chord. The slide chord will then initiate without forcing the user to pick up all fingers and start over with synchronized finger touchdowns.

[**0261**] In the second way, the user starts with multiple fingers resting on the surface, lifts a subset of these fingers, touches a subset back down on the surface synchronously to select the chord, and begins moving the subset laterally to initiate the slide. Decision diamond **656** actually initiates the slide mode when it detects significant motion in all the fingers of the synchronized subset. Whether the fingers which remained resting on the surface during this sequence begin to move does not matter since in this case the selected chord is determined in step **658** by the combination of fingers in the synchronized press subset, not from the set of all touching fingers. This second way has the advantage that the user does not have to lift the whole hand from the surface before starting the slide, but can instead leave most of the weight of the hands resting on the surface and only lift and press the two or three fingers necessary to identify the most common finger chords.

[**0262**] To provide greater tolerance for accidental shifts in resting finger positions, decision diamond **656** requires both that all relevant fingers are moving at significant speed and that they are moving about the same speed. This is checked either by thresholding the geometric mean of the finger speeds or by thresholding the fastest finger's speed and verifying that the slowest finger's speed is at least a minimum fraction of the fastest finger's speed. Once a chord slide is initiated, step **660** disables recognition of key or chord taps by the hand at least until either the touching fingers or the synced subset lifts off.

[**0263**] Once the slide initiates, the chord motion recognizer could simply begin sending raw component velocities paired with the selected combination of finger identities to the host. However, in the interest of backward compatibility with the mouse and key event formats of conventional input devices, the motion event generation steps in FIG. **40**B convert motion in any of the extracted degrees of freedom into standard mouse and key command events which depend on the identity of the selected chord. To support such motion conversion, step **658** finds a chord activity structure in a lookup table using a bitfield of the identities of either the touching fingers or the fingers in the synchronized, subset. Different finger identity combinations can refer to the same chord activity structure. In the preferred embodiment, all finger combinations with the same number of non-thumb fingertips refer to the same chord activity structure, so slide chord activities are distinguished by whether the thumb is