KMK FW - Lock Status

This extension exposes host-side locks like caps or num lock.

Enabling the extension

```
from kmk.extensions.lock_status import LockStatus

locks = LockStatus()
keyboard.extensions.append(locks)
```

Read Lock Status

Lock states can be retrieved with getter methods and are truth valued -- `True` when the lock is enabled and `False` otherwise.

Method	Description
`locks.get_num_lock()`	Num Lock
`locks.get_caps_lock()`	Caps Lock
`locks.get_scroll_lock()`	Scroll Lock
`locks.get_compose()`	Compose
`locks.get_kana()`	Kana

React to Lock Status Changes

The best way to react to changes in lock status is to extend the LockStatus class. When a lock status change happens, the 'after_hid_send' function is invoked so you would override LockStatus's to inject your own logic. Be aware though that this function is also critically important to the functionality of LockStatus so be sure to invoke the 'super()' version of your class to trigger the default functionality of LockStatus.

```
# in your main.py
from kb import KMKKeyboard
from kmk.extensions.lock_status import LockStatus
from kmk.extensions.LED import LED

keyboard = KMKKeyboard()
leds = LED(led_pin=[board.GP27, board.GP28])
```

```
class LEDLockStatus(LockStatus):
    def set_lock_leds(self):
        if self.get_caps_lock():
            leds.set_brightness(50, leds=[0])
        else:
            leds.set_brightness(0, leds=[0])

        if self.get_scroll_lock():
            leds.set_brightness(50, leds=[1])
        else:
            leds.set_brightness(0, leds=[1])

    def after_hid_send(self, sandbox):
        super().after_hid_send(sandbox)  # Critically important. Do not
forget
        if self.report_updated:
            self.set_lock_leds()

keyboard.extensions.append(leds)
keyboard.extensions.append(LEDLockStatus())
```