MouseMover for Arduino Pro-Micro

Overview

Having worked in corporate America for the better part of 40 years, I've seen corporate computer security policies change over that time from almost none to nearly insane. Today many IT groups force the computers they maintain to require a password to be entered if the computer goes to sleep, which can occur in as few as 5 minutes, in some cases. A recent employer of mine also went to a two step authentication scheme using your password and a token device that displays a set of numbers that also have to be entered as a second password. While this security is nice and sometimes essential, it can be very annoying if you actually need to get something done (besides entering passwords). I can't count the times I've been interrupted at my desk by a conversation with a co worker only to find that when I get back to my computer, the screen has gone to the lock screen and I have to enter the password *and* a token value to continue. It would be nice to have a mechanism to occasionally automatically move the mouse a little to keep the computer from going to sleep and the lock screen.

Searching the web for a solution to this revealed that others have attempted to solve this problem. Everything from a servo that physically moves a real mouse to software programs that run on the PC as a background task were found to do this function. The servo method seems far too primitive, while the software only PC program versions don't always work because the security rules "see through" this cheap trick. What would be nice is to come up with a low cost device that "looks and smells" like a real mouse, that occasionally moves. This lead to the development of the "MouseMover" device detailed here.

********** DISCLAIMER **********

This firmware and its associated device can be used to override corporate IT department directives and security mechanisms in place by keeping the system from going back to a lock screen. This may not be allowed in a corporate or government environment and may result in mild to severe penalties. If in doubt, check with your IT department before using this device. Use this device at your own risk.

********** DISCLAIMER **********

Project Details

This project implements a simple "MouseMover" that can be used to keep a computer from going to sleep or locking using only a low cost (≈$5-$17) Pro Micro module (Sparkfun DEV-12640 or equivalent clone) with an ATMega32U4 processor chip, a USB cable and software implemented using the Arduino IDE compiler and environment. It can also be used on other Arduino controllers like the Arduino Leonardo or Arduino Micro, but the Pro Micro board is tiny, powered by the USB port and low cost, making it the ideal hardware solution for this task.

This controller moves the mouse cursor a tiny amount every minute or so, simulating the user being at the computer, which prevents host PC from going to sleep or locking the screen. Unlike software only solutions that many IT infrastructure systems know how to ignore, this solution looks as though a real (second) mouse is added to the computer. The normal mouse is still used for computer operation.

Usually moving the mouse cursor a few pixels every minute or so is enough to keep the computer from going to sleep or locking. Because the amount moved is small (even just a few pixels in either the X or Y direction is usually enough) and it's only is done every minute or so, this will not usually interfere with normal computer operation. To maintain some semblance of system security (that IT groups demand) this added mouse controller can be programmed to only run for a limited time, after which it stops moving the mouse, until commanded to run again (or unplugged and re plugged in again).

This controller has a number of user changeable parameters. All of the parameters for the controller can be modified by sending commands to the controller's serial port, which is also added to the host PC when the controller is plugged in. This controller will appear to Windows as a mouse device (HID) and a virtual serial com port. Once the controller is programmed, it will have default parameters selected for all of the changeable parameters, but these can be changed and then saved to EEPROM so that the user selected parameters are used each time the controller is connected to the PC.

This firmware is designed for use with a Pro Micro board (that uses an ATMEGA32U4) and is connected to the PC with its USB port. With this concept, the only hardware required for this solution is an Arduino Pro Micro controller and a USB cable that connects this controller to the PC.

Hardware and I/O

While this project requires no hardware or wiring other than the Pro Micro module and the USB cable to hook it up to the computer, there are a few of the I/O pins that can have jumpers added to alter the devices operation. There is another pin that can have an LED connected to it to indicate that it's running or not, if desired. While not required, it has been found that having the LED connected is a handy addition – It's nice to know when the MouseMover is running and not. The other connections to the jumpers are mostly provided as a convenience, or if control of the MouseMover by jumpers is desired instead of using serial commands.

A drawing of the various optional I/O connections is shown below.

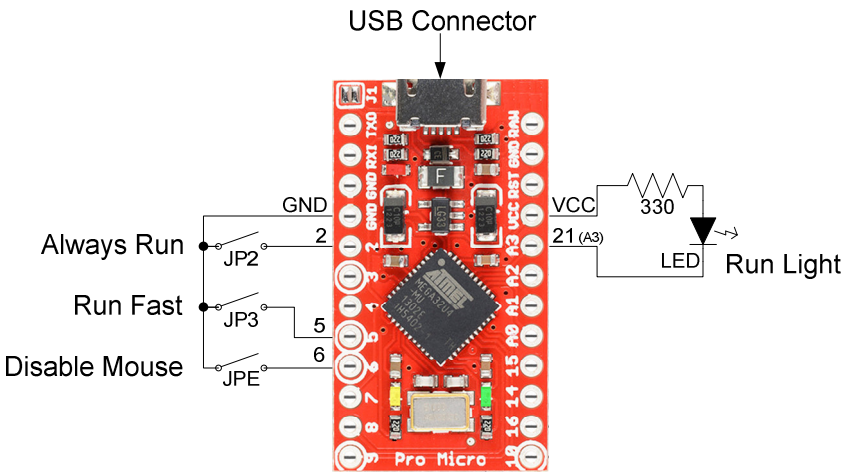


Figure 1 Optional External Connections

One jumper called JPE (defined in the code) can be used to disable the controller. By grounding this pin, the mouse functionality will stop (as though you unplugged the mouse) and will be restored when the jumper is removed. This jumper is on Arduino pin 6 [PD7].

A second jumper, JP2 (defined in the code), if grounded, will disable the software run timer and keep the mouse "running" all the time, no matter what the programmed run time is. Remove the jumper to restore normal operation. This jumper is on Arduino pin 2 [PD1].

A third jumper, JP3 (defined in the code), if grounded, will cause the mouse to move every 5 seconds, instead of the configured move time. Remove the jumper to restore normal operation. This jumper is on Arduino pin 5 [PC6].

An LED and resistor can be added between pin A3 (Arduino Digital pin 21 [PF4]) and VCC to indicate when the MouseMover is running (Pin is labeled "A3" on the Pro Micro). While not required, it is handy to have this indicator so that you know if the MouseMover is running or not (on or off). The LED can be any LED you might have. The resistor shown is a 330 ohm 1/4W part, but any resistance from about 100 ohms to 1K would be usable, depending on the LED used. There are also LED's available with a built in resistor, such as Lumex SSL-LX3044GD-5V (Digikey 67-1062-ND) that are designed to run directly from the 5V supplied by the module. Using this type of LED simplifies connection to the module – Just solder the two leads of the LED to the two pins indicated. For any LED, be sure to observe the polarity of the LED – The cathode end goes to pin A3 and the anode goes to the resistor and VCC.

Compiling Firmware and Downloading Code

This project was developed using Arduino IDE version 1.8.5. Newer versions, including the latest (Vers. 1.8.15) should also be able to be used without issue. The settings in "Tools" of the IDE should be:

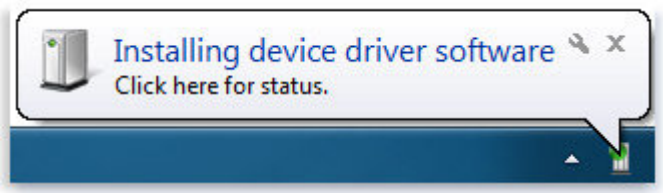
Board: Arduino Leonardo (This can be used for Pro Micro also)
Port: COMxx (as it is assigned by your system)
Programmer: ArduinoISP.

If the Pro Micro device has already been used on the computer system that the Arduino IDE is installed on the next section can be bypassed – Just press the "Upload" button once the file is loaded in the IDE. If this is the first time that the Pro Micro module is being connected to the computer, you will have to install the device driver for it.

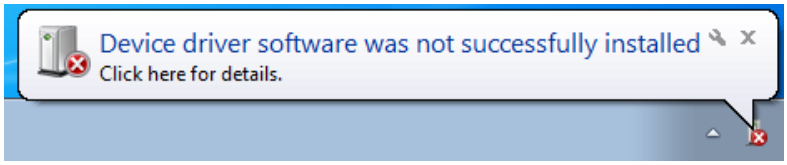
Installing the Pro Micro device on Computer

Depending on the computer and operating system that the Arduino IDE is installed on, the Windows device driver for the Pro Micro controller may need to be installed. For this project, three different drivers are used – One virtual com port for running the final application, One virtual com port for programming the device, and an HID driver. The HID driver is part of Windows, but the other driver (with two ports) will need to be installed. On some systems it will install automatically, while on other systems you may need to install it manually using the proper .INF file for the controller. These are available online, if needed. If the drivers install automatically the following section can be ignored, otherwise follow the instructions detailed next.

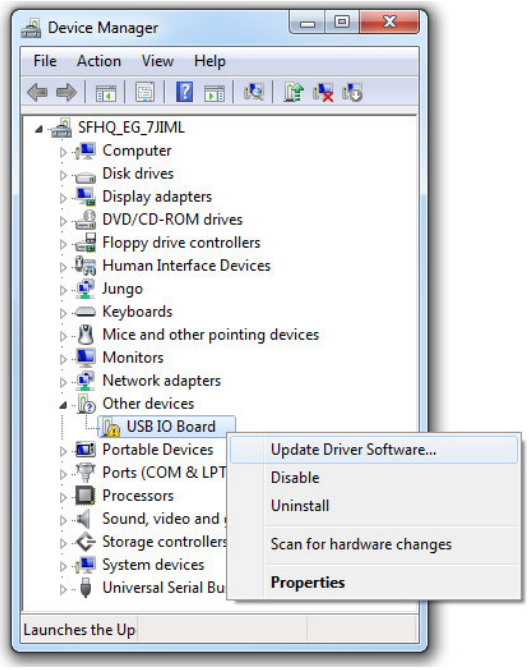
When the Pro Micro device is first plugged in, Windows will display the following message.



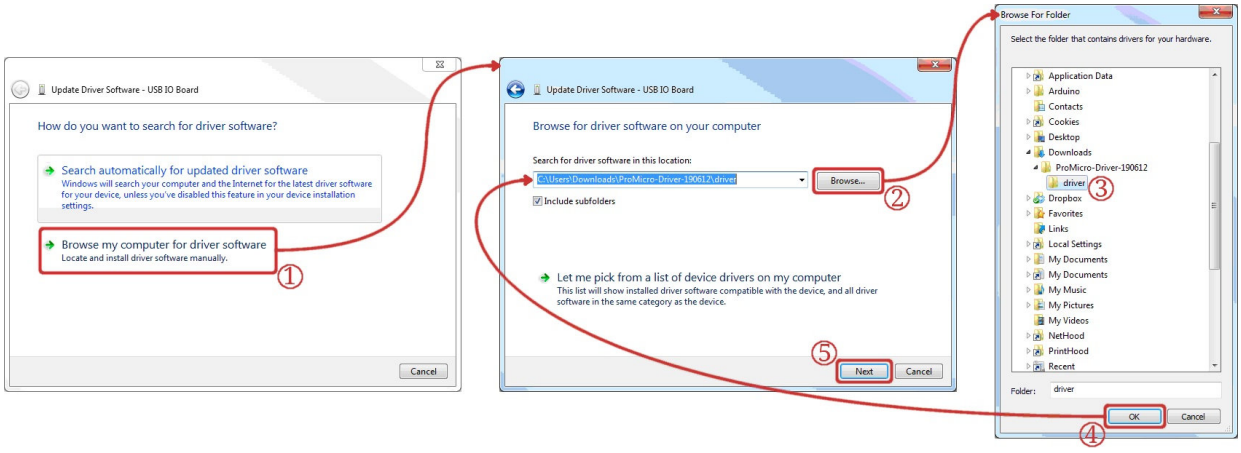
Then, after a while, it may display a message similar to the following



because, unless this driver has been installed before, Windows will probably not know how to find it. This is normal. To install the device driver, Windows will need to know where to find the (.INF) file for this device. This is done by starting Device Manager and then locating the uninstalled device in the tree of devices. To bring up Device Manager right click on 'Computer' and then "Management" or in the start menu enter "Devmgmt.msc" in the "run" text box. Once running a screen similar to the following will be displayed. In the Device Manager, expand the 'Other devices' tree, where a 'USB IO Board' or an 'Unknown Device' with a yellow warning sign over its icon should be found. Right-click on the 'USB IO Board' or the 'Unknown Device' icon and select 'Update Driver Software'.



In the first window that pops up, click 'Browse my computer for driver software'. On the next window, click 'Browse…' and navigate to the location of the driver that has been downloaded. After the driver folder has been selected, click 'OK', then select 'Next'



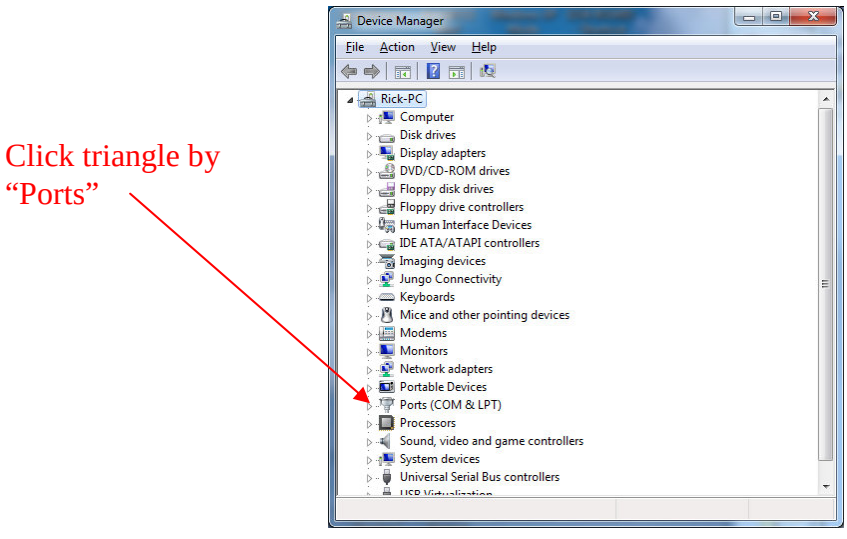
Windows will then attempt to install the driver but may warn about the driver being unsigned. If so, it is safe to select 'Install this driver software anyway' on the warning dialog. If possible use a signed driver instead. Signed drivers are available for either the Sparkfun device, the Arduino Leonardo or the generic Pro Micro modules.

After watching the progress bar for a while, eventually the message 'Windows has successfully updated your driver software' window will be seen. And the 'Device Manager' should now have a new entry for the 'SparkFun Pro Micro (COM ##)' or 'Arduino Leonardo (COM ##) under the 'Ports' tree. Make note of the COM port number associated with this new device.

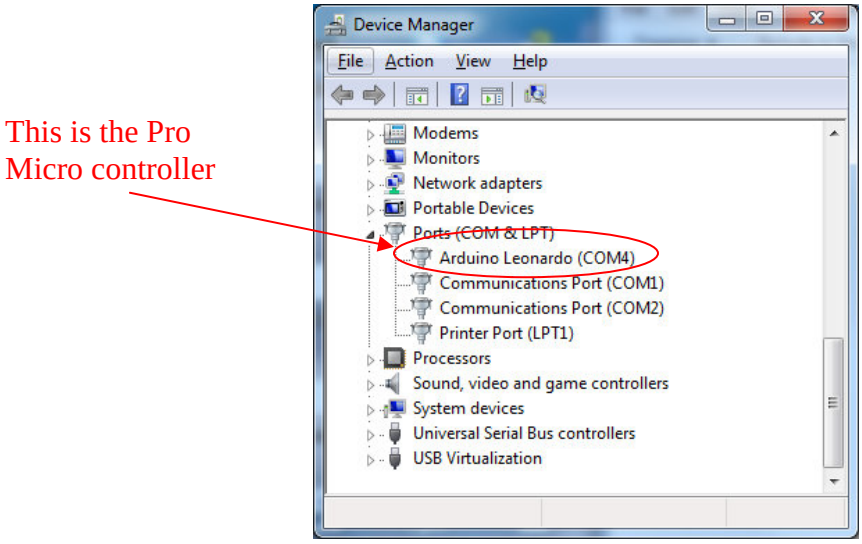
This comport number can be used in future communications with the device, but it may be advantageous to change the com port number to some other number, depending on your needs. On some systems you might want to reassign it to COM99 (or similar) just so it's out of the way of other devices on the system.

Determining and Changing the Com Port Number

To determine which com port number the device is using the Windows device manager program (Devmgmt.msc) can be used. When run it will produce a window similar to the following:

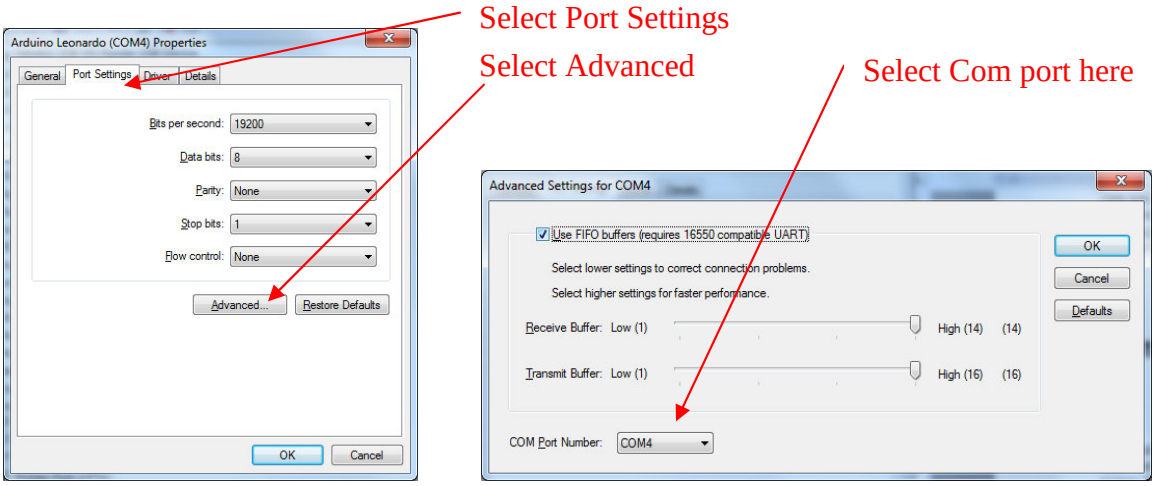


Click on the small triangle to the left of "Ports (COM & LPT)" and the list will expand as shown



As can be seen, on this computer there are 3 com ports installed (and one printer port). Locate the one named “Arduino Leonardo” or “Sparkfun Pro Micro”. This is the com port number of the hardware – COM4 in the sample. Remember this number, as it will be needed when running the program to program and configure the hardware.

If desired, the com port number can be changed using device manager. Right click on the com port for the product and select “Properties” then select the tab “Port Settings”. Then in the properties window that comes up select “Advanced”. The comport numbers can be found in a drop down list in this panel. Select the desired com port number and then click the “OK” button on both windows to change the port. As with changing any com port number, if the port number is changed, either Windows will have to be restarted or, in the case of USB devices, the device will need to be unplugged and then after a few seconds plugged in again to allow the use of the device. The pictures below show the menus.



Once the port number has been changed and either Windows has been restarted or the USB port cable has been disconnected and reconnected, Device Manager can be run again to insure that the com port number is now the number that is desired.

If this is the first time that a product has been connected to the computer in a particular physical USB port, a failure during the programming step may be experienced because the bootloader port Windows

device driver has not yet fully installed by the time the program times out trying to program the device. This is usually noted by the balloon at the bottom of the screen stating “Installing Device Driver”. This installation process usually takes far longer than the Arduino IDE program gives the device to start programming. If this happens, just wait until the Windows device driver finishes installation and then push the “Upload” button again. After this, the program upload function should work normally.

Software Commands

This controller accepts serial commands to perform various functions and report current settings. Open com port with any baud rate (other than 1200 baud) (N,8,1) using a program like Hyperterm, Teraterm, or the terminal window in the Arduino IDE (or your favorite terminal program). You can also create batch files to send commands to the com port. The com port number for the MouseMover can be determined from the previous section.

Once you have verified that communications to the MouseMover are working (You can send “?” and <Enter> and see a help screen), you can then send commands to the MouseMover and set up its parameters, if desired.

The commands for the MouseMover are detailed next. Commands are a single letter (to read the value) or a single letter followed by a number (to set the value) and <CR>.or <CR><LF>, as in X<CR> or X20<CR> (Can also be X=20<CR>) (Note: <CR> is the “Enter” key).

Read	Set	Description
V		Return a version / copyright string.
*IDN?		Return a version / copyright string. (SCPI compatible format)
X	X4	Amount to move mouse in the x direction (0..100) (default 4)
Y	Y0	Amount to move mouse in the y direction (0..100) (default 0) (Note: For X and Y values, the mouse will move plus this many pixels on the first move and then move minus this many pixels on the next move and then alternate between positive and negative moves.)
T	T60	How often to move mouse (seconds) (2..3600) (default 60=1 min)
R	R540	How long to run the MouseMover (minutes) (-1..64800)(45 days max) (0=Turn off, -1=run forever) (default 540=9 hrs)
Z		Return all variables in a comma separated list. Values returned are: X,Y,<set T>,<set R>, <current T>,<current R>, days HH:MM:SS (“days HH:MM:SS” is total run time since controller was plugged in)
E	E0	Reinitialize current variables to default values (shown above)
	E1	Get current variables from EEPROM
	E2	Save current variables to EEPROM
?		Return a help string of commands.
H		Return a help string of commands.

If you want to alter the variables “X”, “Y”, “T”, “R” and then save them as the new default power up values, just set them to new values and then enter “E2<CR>. To restore them to original default values enter “E0<CR>” and then “E2<CR>”.

System Integration

Once the Pro Micro module is programmed with the MouseMover firmware it can be used in your system with various levels of integration.

The simplest use is to merely plug it in to a USB port when you want it to function and unplug it when you don’t want it to function. Each time it is plugged in it will run using the parameters that are saved in EEPROM, which include the amount of time to run. After the MouseMover hardware has been plugged in for the amount of time to run that is programmed, the MouseMover will stop, allowing the computer to go to sleep/lock. By simply unplugging it and then re plugging it in again, a new run time will be used again. Using this method, the user can program the MouseMover with the amount of time to run and then plug it in each day for that amount of “run” time.

Windows Batch file

Instead of plugging and unplugging the MouseMover, a batch file and various shortcuts can be created to enable/disable the device. You can then also set up a task in Windows to call this batch file at whatever time you specify or when you log in. These have been created for Windows 7 and Windows 10, but can be converted for Apple or Linux/Unix systems.

You can create a batch file to send a “Run” command to this device as follows:

```
@echo off
rem MoverPort is comport address in format \\.\COM# (set to your comport#)
set MoverPort=\\.\COM4
rem R=<# minutes to run> (9 hours=540) (0=off)(%1=get from cmd line)
echo R=%1 >%MoverPort%
```

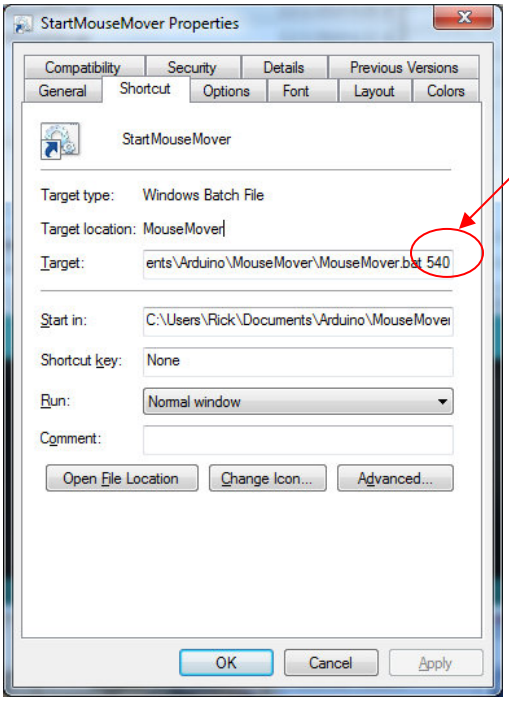
Note: Replace “COM4” in the above batch file with the COM port number used on your machine.
A suggested name for this batch file is “MouseMover.bat”

Then call the batch file with a parameter as in:

`MouseMover 540`	(This will set the runtime to 540 minutes)
`MouseMover 0`	(This will turn off the MouseMover)

Windows Shortcuts (.lnk file)

With the batch file created and placed in a directory of your choice you can then create a Windows shortcut to call this batch file by merely double clicking on the shortcut icon. To create the shortcut, right click on the batch file and click “Create Shortcut”. You can then rename the short cut to a more recognizable name such as “StartMouseMover” or “StopMouseMover”. Once renamed (if desired) you can then edit it to send the amount of time to run, as in



Note parameter added to "Target" string

Then, if desired another shortcut can be created to turn MouseMover off by copying the already created shortcut and then changing the parameter to zero.

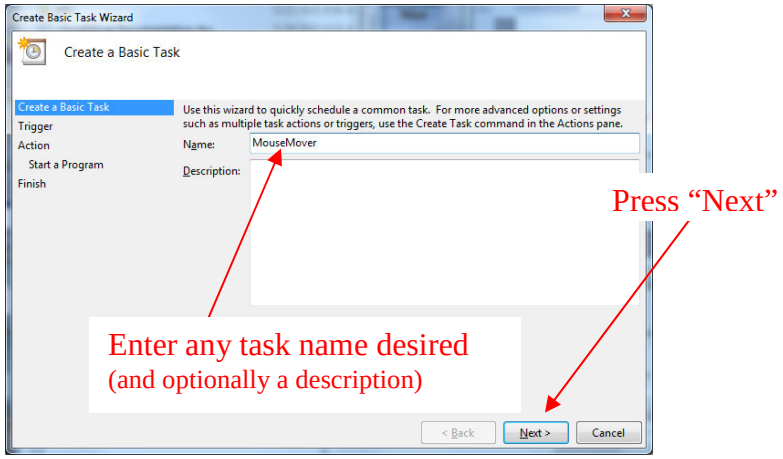
Once created, you can just double click the shortcut(s) to turn on the MouseMover for the amount of time you specified in the shortcut file or turn it off if the time in the shortcut is zero.

Windows Task

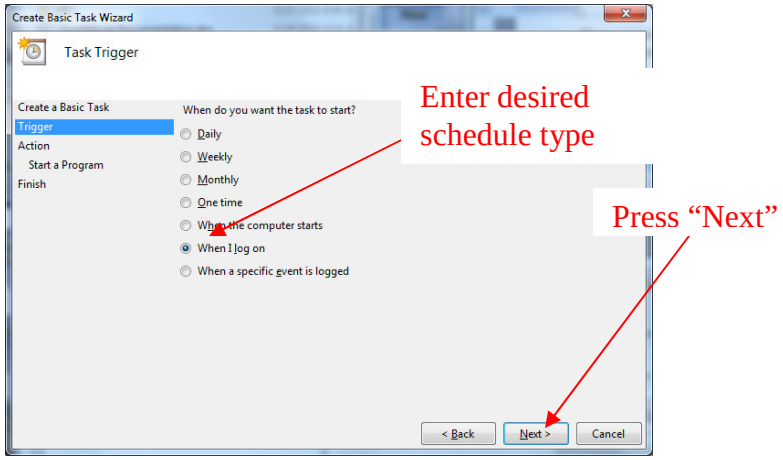
The user can also create a scheduled task to call this batch file every weekday (or whatever interval is desired) or every time the user logs on. This can either be done by running the Windows task scheduler and creating a new task or by typing a command on the command line in a CMD box to create the scheduled task.

To run the task scheduler in Windows, do the following:

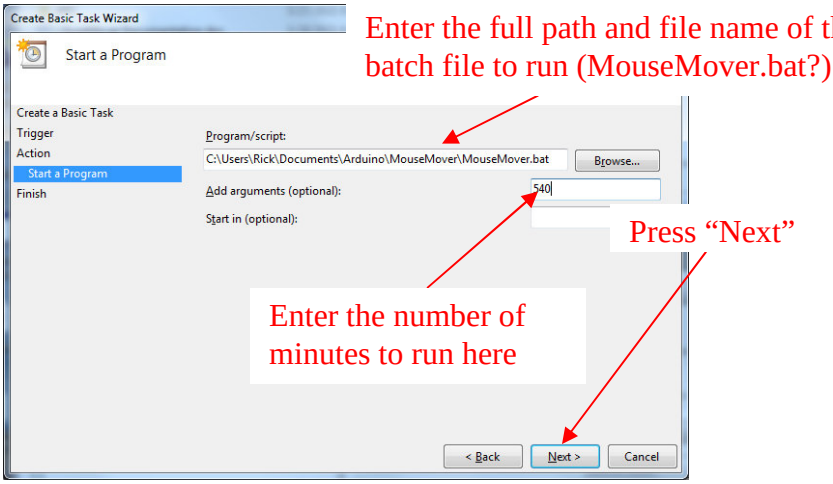
1. In the window run line type "Schedule Tasks" or "taskschd.msc /s" and run it (hit return). This is also available as one of the items in the Computer Management program (Compmgmt.msc).
2. In the "Action" part of the window, click "Create Basic Task…"
3. Provide a name for the task, such as "MouseMover". Click "Next".



4. In the "Task Trigger" window, select when you'd like the task to trigger. This can be a time schedule or when you log in. When selected click "Next".

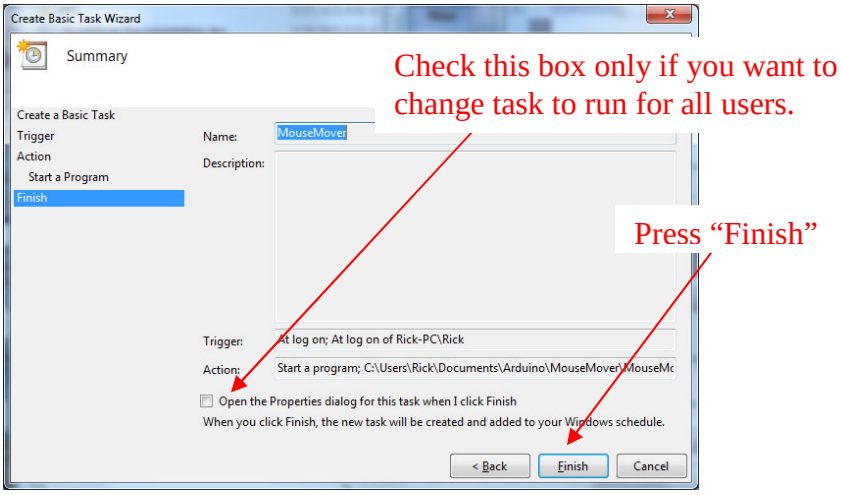


5. In the "Action" window, select "Start a program" and click "Next". This will bring you to a window where you enter the name of the batch file created earlier, and the time to run as in

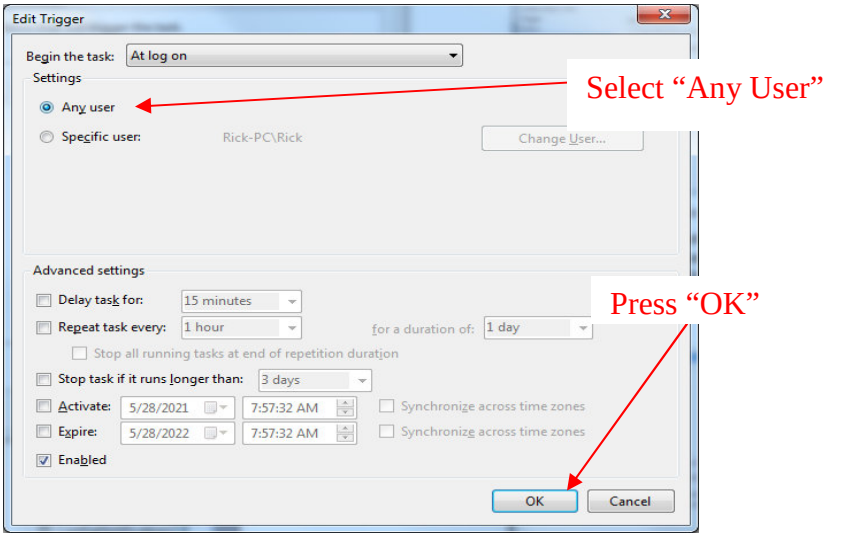


Once the batch file name and the run time is specified, press “Next”

6. Next, you will need to decide if the task will run just for yourself or for all users. If you just want it to run for yourself, press the “Finish” button and you’re done and can close the task scheduler. If you want the task to run for all users, check the box “Open the properties dialog for this task when I click finish”, then press “Finish” to continue to the next step.

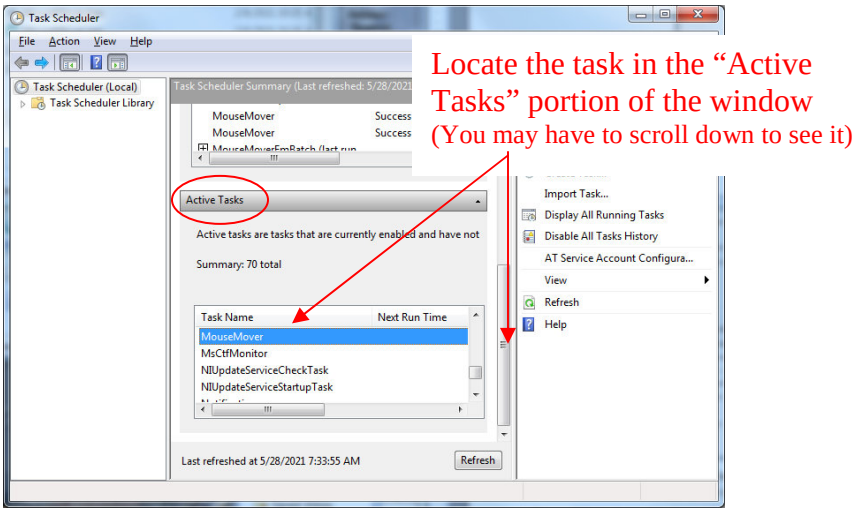


7. If you checked the box “Open the properties dialog for this task when I click finish”, click on the tab “Triggers” (in the center part of the window) and then double click the entry “At log on” (or whatever trigger action you chose). This will bring up an “Edit Trigger” window.

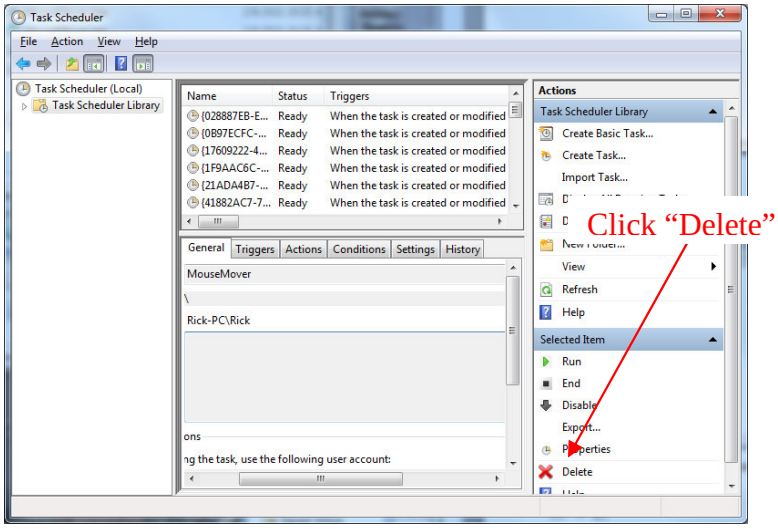


Select the button “Any User” and then press “OK”. Then press OK again to leave the properties window. The task is now set up and will run when it was programmed to run. The Task Scheduler program can then be closed.

8. If it is desired to delete the scheduled task, locate the task in the “Active Tasks” section of the Task Scheduler window.



9. Double click on the task, which will change the window to



10. On the right side of the window click “Delete”. After answering “Yes” to the dialog box “Do you want to delete this task”, the task will be removed.

11. Close the Task scheduler.

The task can be created using the Task scheduler program shown above, but for some users with an IT locked down computer, this may not be an option. At a previous employer (before I retired) I was unable to add a task, but did have the option of running a program with “elevated privileges”. Using this option a CMD.EXE box could be opened with admin privileges. If this is the case, you can either run “taskschd.msc /s” from the CMD box and follow the steps above, or you can just enter the following command in the CMD box to create the task:

```
SCHTASKS /Create /TN MouseMover /SC ONLOGON /TR
"'C:\Users\Name\Documents\Arduino\MouseMover\MouseMover.bat' "540""
```

Notice the use of single and double quotes in the name of the program to run. This allows the program to have the parameter 540 as part of the command. Obviously the program name and path will need to be changed to wherever the batch file is located and the time (540) can be adjusted as desired.

No matter which way this task is added to the task scheduler, it must be added as a process that has admin privileges. This can be done by opening a CMD.exe box with "Run as Administrator" (right click on CMD.EXE icon and select "Run as Administrator") or putting the 'SCHTASKS' command in a batch file and then running the batch file by right clicking on it and selecting "Run as Administrator".

The task can be deleted with the command `SCHTASKS /Delete /TN MouseMover`

In my use of this program I found that setting up a task that runs each time you log in to the computer works best (instead of a fixed time schedule). Using this method, you sign in each morning one time and the computer will stay running for the next number of hours programmed and then will go to sleep/lock.

Another possibility as to the use of this product would be to plug it in while you're at your computer and then unplug it and take it with you when you leave your computer. This should be less of a security risk as it guarantees you're at your computer when the MouseMover is active (which is the whole reason that IT groups impose the lock screen timeout to begin with).

It is hoped that the use of the MouseMover will eliminate one of the "inconveniences" of the IT imposed security measures. Please heed the disclaimer above if using this device in a corporate environment. Obviously you can also use it on your home computer as you see fit.

********** DISCLAIMER **********

This firmware and its associated device can be used to override corporate IT department directives and security mechanisms in place by keeping the system from going back to a lock screen. This may not be allowed in a corporate or government environment and may result in mild to severe penalties. If in doubt, check with your IT department before using this device. Use this device at your own risk.

********** DISCLAIMER **********