Archer2 and Allegro2 Keyboard Power User Guide

Please refer to STUD.reg for the latest default registry settings.

The keyboard driver was written such that any key (not including the power button) could be remapped with a registry change. This document describes the structure of the registry for the keyboard driver. The details of this are somewhat complicated and will take a few minutes to digest. If you are just after a quick change, try checking out the examples at the end of this document.

Each type of keyboard has a unique identifier. This ID is read at boot and saved in the following registry key during boot up.

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID]
    "InstalledKeyboardID"=dword:0
```

<u>Here are some existing IDs:</u>

Archer2 Numeric: 2
Allegro2 (Alpha only): 0x80000007
Allegro2 (Beta and later): 0x80000027

The ID is used in the next subkey. The ID needs to be formatted with %X.

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID\2]
```

-OR-

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID\80000027]
```

Then we have a whole lot of registry values within this registry key that define the basic key map tables. Here we are using Archer2 as an example.

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID\2]
    "NumKeys"=dword:4E
    "PressHoldKeyboard"=dword:0
    "NumModifierKeys"=dword:1
    "ShiftModifierKey"=dword:FF             ; Shift key
    "AltModifierKey"=dword:FF               ; Alt modifier key
    "CtrlModifierKey"=dword:FF              ; Ctrl modifier key

    "ModifierTable0"=hex:00,\
        C2,00,C3,00,C1, 00,C4,00,5D,00,\
        00,00,31,00,32, 00,33,00,00,00,\
        00,09,00,00,00, 00,00,08,00,00,\
        00,00,34,00,35, 00,36,00,00,00,\
        00,C0,00,00,00, 00,00,0D,00,00,\
        00,00,37,00,38, 00,39,00,00,00,\
        00,00,00,00,00, 00,00,00,00,00,\
        00,00,BD,00,30, 00,D3

    "ModifierKey1"=dword:3E
    "ModifierTable1"=hex:00,\
        C7,00,C5,00,CF, 00,C6,00,C9,00,\
        00,00,CA,00,26, 00,CB,00,00,00,\
        00,D1,00,00,00, 00,00,D2,00,00,\
        00,00,25,00,C8, 00,27,00,00,00,\
        00,5B,00,00,00, 00,00,0D,00,00,\
```

```
00,00,6B,00,28, 00,6F,00,00,00,\
00,00,00,00,00, 00,00,00,00,00,\
00,00,CC,00,CD, 00,CE
```

All of the keys besides the power button are in a row column matrix. Here is the layout of the Archer2 numeric keyboard. Locations in the matrix that actually have a key will have an abbreviation of the key name (i.e 'Cam' for the camera key), and be highlighted.

01 Cam	02	03 P1	04	05 Trgt	06	07 P2	08	09 ctx	10
11	12	13 1	14	15 2	16	17 3	18	19	20
21	22 tab	23	24	25	26	27	28 bak	29	30
31	32	33 4	34	35 5	36	37 6	38	39	40
41	42 hom	43	44	45	46	47	48 ret	49	50
51	52	53 7	54	55 8	56	57 9	58	59	60
61	62 mod	63	64	65	66	67	68	69	70
71	72	73 dash	74	75 0	76	77 dot			

There are lots of holes in this keyboard. Each key has its own numeric identifier.

"`NumKeys`" defines how many bytes are needed to define the table of the keyboard layout. NumKeys must be one more than the ID of the last key (the key IDs are zero indexed), so NumKeys = 78 decimal (or 4E hex).

There are two main types of keyboards:

1) Press and Hold
2) Modifier

Press and Hold is like the Mesa. Each key can have two functions. Press/Release for the first function, and press/hold(for ~1 second) to trigger the alternate function of that key. A modifier keyboard has a key dedicated to triggering the alternate function of the other keys. More than one modifier key may be present (i.e, FN1, FN2, Blue, Orange, etc.). However, only one modifier may be active at any time (sorry, you can't have Blue+Orange+<some key>).

"`PressHoldKeyboard`". This registry value can either be 1 for Press and Hold or 0 for a Modifer keyboard. The numeric keyboard is initially setup to be a Modifier keyboard.

Caution: The Press Hold keyboard has not been tested for a while, as both Archer2 and Allegro2 do not use this feature by default. There may be some bugs with this style of keyboard.

If "`PressHoldKeyboard`" == 0, then "`NumModifierKeys`" defines how many modifier keys there are. The numeric keyboard has one by default. Although it is referred to as "shift" it is not really a shift key in the sense that it will capitalize letters. It is a modifier or function key.

"`ShiftModifierKey`" defines which key is the true "shift" key. It will precede other key codes with the VK_SHIFT key code. The Shift key is treated special. If press/released once, it acts like the shift key is being held down so that the next key pressed is shifted. If press/released twice in a row, it goes into CAPS_LOCK mode until pressed again. Use 0xFF to indicate there is not a shift key.

"`AltModifierKey`" and "`CtrlModifierKey`" defines which keys are Alt and Ctrl. These are also treated special. These are treated as sticky keys so that if pressed/released, it acts like the key is being

held down so the next key is “Alt’d” or “Ctrl’d”. You cannot press it twice to lock it on. Pressing it twice will de-activate (release) the Alt or Ctrl function.
And now to the meat of how this all works:

“`ModifierTable0`” is a binary registry value that identifies the key codes that will be generated for the keys. 0x00 should be used for where holes are located. **The values in this table are VK key codes** (see also the Special Keys below). Let’s look at a simple example first, such as the number ‘3’ button highlighted below:

```
"ModifierTable0"=hex:00,\
    C2,00,C3,00,C1, 00,C4,00,5D,00,\
    00,00,31,00,32, 00,33,00,00,00,\
```

0x33 is the VK key code for generating a ‘3’.

“`ModifierTable1`” shows a similar table, but this one is for when the first modifier key is active. `"ModifierKey1"` defines the key index to the modifier key for this table.

You could have more modifier keys and tables – up to 5 total tables (0 through 4, so four modifier keys besides shift, Ctrl, and Alt):
“`ModifierTable2`”
“`ModifierKey2`”

If the desired function for a particular key can be done with a single VK key code, then this is all you would need. However, several symbols require multiple VK key codes to generate. Take left parenthesis for example. To generate this symbol, these VK key codes must be generated:

VK_SHIFT down, ‘9’ down, ‘9’ up, VK_SHIFT up. This is 0x10 down 0x39 down, 0x39 up, 0x10 up.

If you look at your desktop QWERTY keyboard, you will see left parenthesis on the ‘9’ key (key code of 0x39).

A few other things we want our keys to do are to launch a program with a certain key, or show up as a reprogrammable key in the windows buttons control panel.

To do all of these other special functions besides a single VK key code is where the “`SpecialKeys`” subkey comes in.

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID\2\SpecialKeys]
    "MaxExpansion"=dword:2
    "SpecialKeyMin"=dword:C0
    "SpecialKeyMax"=dword:D2
    "DecimalSeparator"=dword:D3
    "ThousandsSeparator"=dword:D4
    "C0"="A2_KeyAction"
    "C0_CmdLine"="2"            ; Home (Today Screen)
    "C1"=hex:5B,C1              ; VK_LWIN,      VK_APP1
    "C2"=hex:5B,C2              ; VK_LWIN,      VK_APP2
    "C3"=hex:5B,C3              ; VK_LWIN,      VK_APP3
    "C4"=hex:5B,C4              ; VK_LWIN,      VK_APP4
    "C5"=hex:5B,C5              ; VK_LWIN,      VK_APP5
    "C6"=hex:5B,C6              ; VK_LWIN,      VK_APP6
    "C7"="A2_KeyAction"
    "C7_CmdLine"="9"            ; Launch New Note
```

```
    "C8"=hex:5C,75              ; VK_RWIN,      VK_F6
    "C9"="taskmgr"
    "CA"=hex:10,39              ; VK_SHIFT,     '9'         issues '('
    "CB"=hex:10,30              ; VK_SHIFT,     '0'         issues ')'
    "CC"=hex:10,38              ; VK_SHIFT,     '8'         issues '*'
    "CD"=hex:10,32              ; VK_SHIFT,     '2'         issues '@'
    "CE"=hex:10,33              ; VK_SHIFT,     '3'         issues '#'
    "CF"="A2_RecordButton"
    "CF_KeyUpEventName"="A2_Record_Button"
    "D0"="A2_BarcodeButton"
    "D1"=hex:10,09              ; VK_SHIFT,     Tab         backtab
    "D2"="A2_KeyAction"
    "D2_CmdLine"="6"            ; Toggle Touchscreen
```

Basically, we pick a range of VK key codes that are never or rarely used. In this example, we start at 0xC0 and go to 0xD2. So we take this range for our special keys. The "`SpecialKeyMin`" and "`SpecialKeyMax`" define the range of key codes we are using for special keys. If a key's value in the ModiferTable falls within this range, it does not just send that VK value up to windows. Instead it processes based on the values within the SpecialKeys registry subkey.

If Special Keys are NOT desired to be used, you must still define a range of one key and give it a definition, such as:

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID\2\SpecialKeys]
    "SpecialKeyMin"=dword:F0
    "SpecialKeyMax"=dword:F0
    "F0"=hex:F0              ; Just pass the same VK key code through
```

Let's look at a few examples to help understand this.
Left parenthesis example:

```
"ModifierTable1"=hex:00,\
        C7,00,C5,00,CF, 00,C6,00,C9,00,\
        00,00,CA,00,26, 00,CB,00,00,00,\
```

On Archer2, if you press "Shift – 1", you get the left parenthesis. Remember we call it shift, but it is really a function or modifier key (you can thank marketing for calling it Shift). The above highlighted value (CA) is at the location of the '1' key under ModiferTable1. 0xCA falls between SpecialKeyMin (0xC0), and SpecialKeyMax (0xD2), so instead of sending the VK key code of CA up to the system, the keyboard driver looks at the registry values for "CA" under the SpecialKeys registry key.

```
    "CA"=hex:10,39              ; VK_SHIFT,     '9'         issues '('
```

VK_SHIFT is 0x10, and '9' is 0x39 (there isn't a VK_9).
The keyboard driver will then send VK_SHIFT down, '9' down, '9' up, VK_SHIFT up to windows which will generate a '(' character.

"Home" key example:
To get to home, it is easiest to actually programmatically do the following (C code):

```
        hWnd = FindWindow(TEXT("DesktopExplorerWindow"), TEXT("Desktop"));
        if (hWnd != NULL)
        {
            SetForegroundWindow(hWnd);
        }
```

The A2_KeyAction.exe program can be called with the command line set to '2' in order to get to the home screen. Now if we observe the ModifierTable0 to find the home key's value:

```
"ModifierTable0"=hex:\
        00,\                                ; 0 is unused
        C2,00,C3,00,C1, 00,C4,00,5D,00,\    ; 1st row  :  1-10
        00,00,31,00,32, 00,33,00,00,00,\    ; 2nd row  : 11-20
        00,09,00,00,00, 00,00,08,00,00,\    ; 3rd row  : 21-30
        00,00,34,00,35, 00,36,00,00,00,\    ; 4th row  : 31-40
        00,C0,00,00,00, 00,00,0D,00,00,\    ; 5th row  : 41-50
```

And now the "C0" values in the SpecialKeys section are:

```
    "C0"="A2_KeyAction"
    "C0_CmdLine"="2"            ; Home (Today Screen)
```

Ok, now that we have a few examples to refer to, here is the formal definition of the Specialkeys section:

For each value between `SpecialKeyMin` and `SpecialKeyMax` inclusive, there can be registry values with two different formats:

If it is a binary format, it indicates which VK codes (multiple) to send, the length must be <= "`MaxExpansion`". The `MaxExpansion` value defines how many VK keys can be stacked up. I have yet to encounter a scenario where more than two is needed.
If it is a string format, it specifies the .exe to launch (.exe is automatically appended to the string). There can also be a few more registry values when this is in use:
"XX_CmdLine" is a registry string value to indicate what to send on the command line. This is optional.
"XX_KeyUpEventName" is a string value to indicate the name of a named event the keyboard driver will set when that key is released. The only place I have used this is for the audio notes (you press and hold the button which launches an .exe to start recording, and then let go of the button to stop recording).

The "`DecimalSeparator" and "ThousandsSeparator`" are used for '.' and ',' on English, and vica-versa for some other languages. If you go to the regional control panel on Archer2 running an English OS, and switch it to the French locale, then the '.' key will then generate a ',' character. This is because in that language, commas are preferable for decimal separators. Specifying these special keys this way in the keyboard driver makes it easier to follow the regional settings of the device.

Reloading registry changes without rebooting:
The reload key is used if registry changes are made during runtime, and you want the changes to take effect **on the next keypress**. Set the following value:

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID]
    "Reload"=dword:1
```

About the APPx keys:
Microsoft has their own set of "special keys", called hot keys, that are triggered by the following VK key codes:

```
    "C1"=hex:5B,C1              ; VK_LWIN,      VK_APP1
    "C2"=hex:5B,C2              ; VK_LWIN,      VK_APP2
    "C3"=hex:5B,C3              ; VK_LWIN,      VK_APP3
    "C4"=hex:5B,C4              ; VK_LWIN,      VK_APP4
    "C5"=hex:5B,C5              ; VK_LWIN,      VK_APP5
    "C6"=hex:5B,C6              ; VK_LWIN,      VK_APP6
```

These are the keys that will show up in the Microsoft Buttons Control Panel as reprogrammable keys. Then you use this set of the registry (or use the control panel) to change what they do (consult Microsoft Documentation on the details of these registry keys and values).

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Shell\Keys\40c1]
    "Name"="Target"
    "Flags"=dword:0
    @="\"\\Windows\\Buttons.lnk\""
    "ResetCmd"="\"\\Windows\\Buttons.lnk\""
    "Icon"="\\Windows\\A2_Resources, -1501"
```

Etc. for 40c2 through 40c6.

Examples of Changes:

Example 1: Change the "Shift – Back" to do screen-rotate instead of touchscreen disable (toggle touchscreen):

This combo starts here in the modifier table:

```
    "ModifierTable1"=hex:00,\
        C7,00,C5,00,CF, 00,C6,00,C9,00,\
        00,00,CA,00,26, 00,CB,00,00,00,\
        00,D1,00,00,00, 00,00,D2,00,00,\
```

So the good news is that it is already a "Special Key". We just need to change the "D2" special key registry which starts out looking like:

```
    "D2"="A2_KeyAction"
    "D2_CmdLine"="6"            ; Toggle Touchscreen
```

The A2_KeyAction with command line '4' will do rotate screen (toggle). With '5' will rotate around all four orientations. So we just need to change `"D2_CmdLine"="4" ; Rotate screen`

Example 2: Change dot ('.') to be toggle flashlight.

If an application did not already exist to do this, one would have to be written. The A2_KeyAction with Command line "17" will do this. As we are not replacing any special keys, we need a new one, so we will take D3, increasing the range of the special keys by one. Now we need to set "`SpecialKeyMax`" to 0xD3 and then define the D3 special key.

```
[HKEY_CURRENT_USER\ControlPanel\Keybd\KeyboardID\2\SpecialKeys]
    "SpecialKeyMax"=dword:D3
    "DecimalSeparator"=dword:D4     ; Need to bump this back one
    "ThousandsSeparator"=dword:D5   ; Need to bump this back one
    "D3"="A2_KeyAction"
    "D3_CmdLine"="17"
```

Normally, you would also have to check and adjust any keys that were using D3 and D4 (Decimal and Thousands separator) in the Modifier Tables.