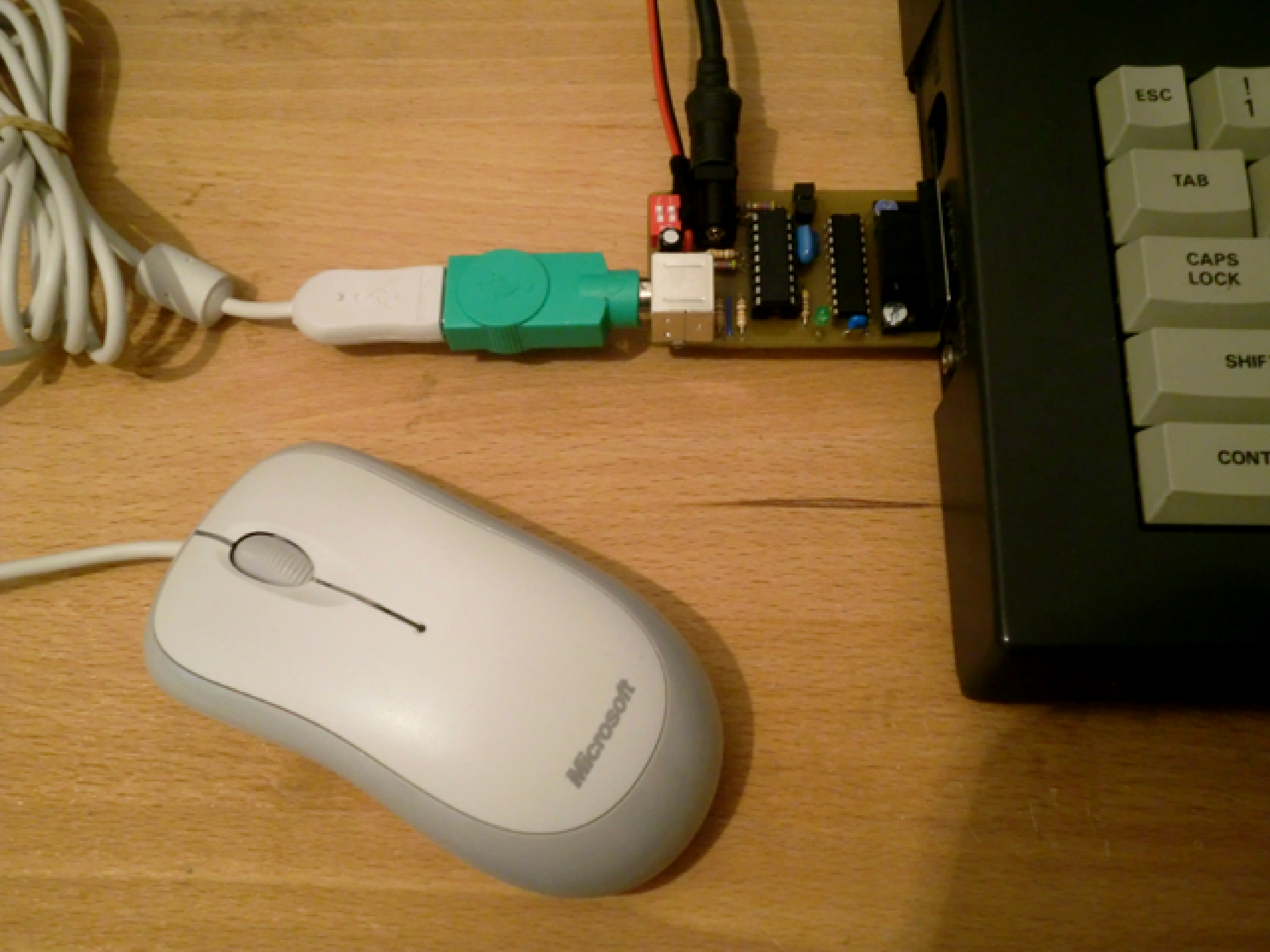
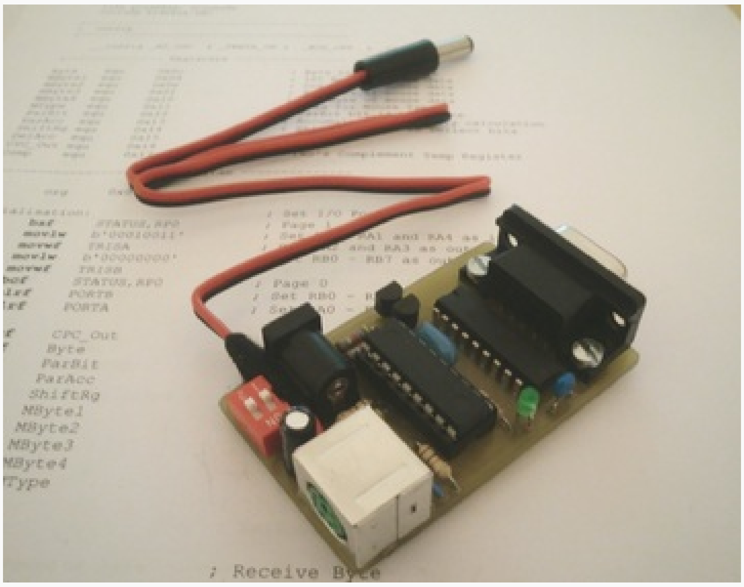
Schneider CPC
ESC
TAB
CAPS LOCK
SHIFT
CONTROL
COPY
Logitech


ESC
!
1
TAB
CAPS
LOCK
SHIF
CONT
Microsoft

PS2Mouse

cpcwiki.eu/index.php/PS2Mouse



Contents

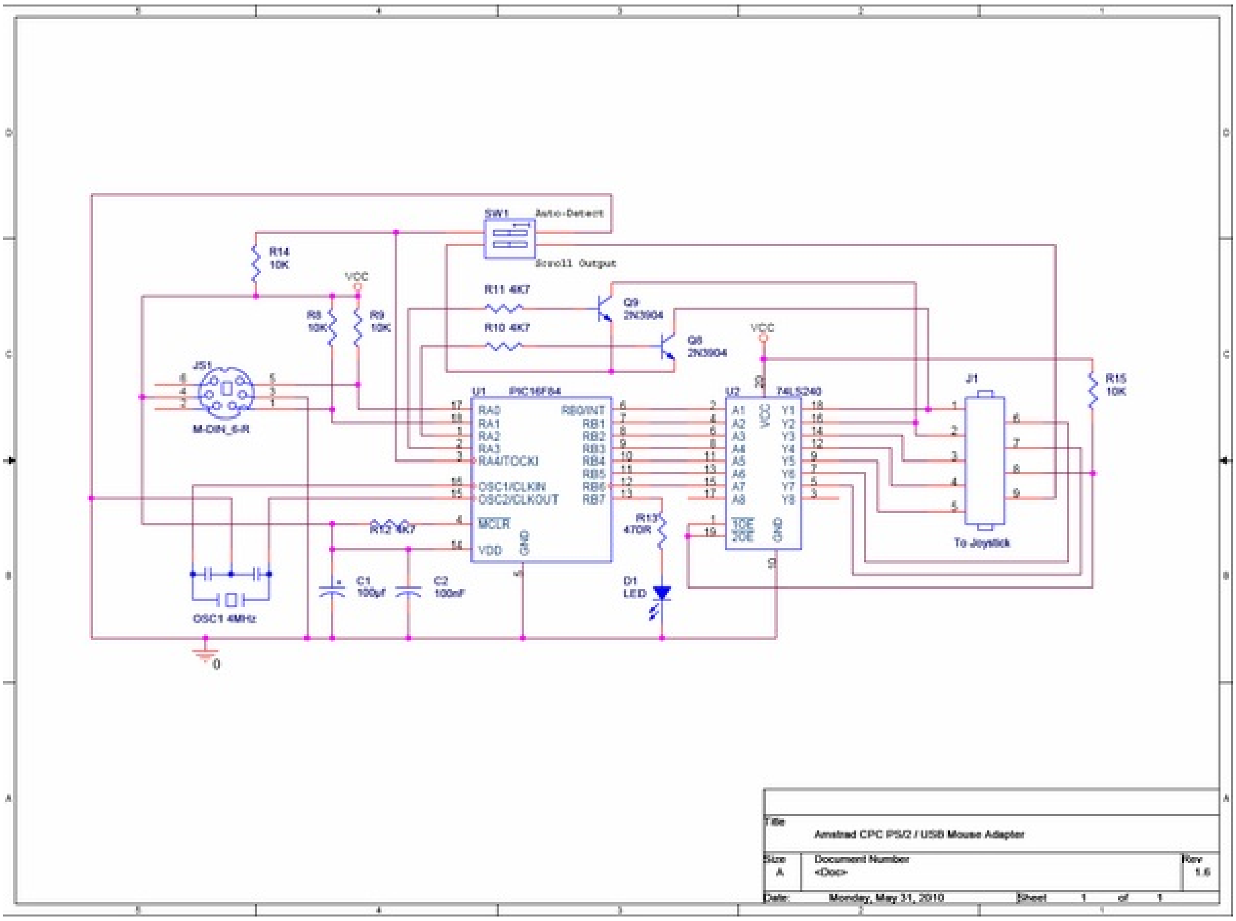
Description

The following project describes how to construct an AMX compatible PS/2 and USB Mouse adapter for the Amstrad CPC. I’ve never actually owned an AMX mouse, in fact I’ve never even seen one other than in pictures. Back in the 80s I had no need for one, but with GUIs like FutureOS and SymbOS now available, I decided it was time to get my own… or at least build an equivalent. Like other projects that I’ve shared in the community, I’ve tried to keep this project as simple as possible to construct and as cheap as possible to build, otherwise it scares people from trying to build it for themselves. That said, you will need to be proficient at soldering and possibly need to produce your own PCB, if you want your end product to look like the one above. I have also supplied a single sided layout, to make it easier for those of us who like to produce our own PCBs. While designing the circuit, I realised that most modern mice have a scroll wheel which wasn’t available back then on the AMX. As the CPC has a few inputs spare on the port (Joy2), I decided to make them accessible by binding them to the Joy2 up and down signals. Although this isn’t strictly AMX compatible, I thought the people at SymbOS or FutureOS, might like to take advantage of them, so consider this adapter is AMX+ compatible. However, due to the joystick ports being split on the CPC Plus series, the scroll wheel feature will only work on standard CPCs unless you create a jumper lead to recombine the two ports again on the Plus.

Warning / Disclaimer

Although I have taken the utmost care preparing this documentation, I do not guarantee that it is error free and I accept no responsibility for damage to anyone’s CPC, Mouse or other personal equipment or injury inflicted on you or others.

The Circuit



The main component on the board is the PIC16F84A. This is a small micro-processor which is pretty much dumb without some firmware to tell it what to do, I will go further into detail about the firmware below. The processor is running at 4 MHz, so the low cost version is all that’s required (there’s a slightly more expensive version which can go up to 20Mhz which would also work). I have used a 4 MHz resonator to save space, but you can also use a 4 MHz Crystal with two 22pf capacitors if you prefer . The other chip seen above is the 74LS240. This is a TTL buffer chip which is triggered by the CPC each time the Amstrad polls the joystick port, this eases the µPs job, because it doesn’t need to constantly watch what the CPCs up to. The two transistors are used to trigger the Scroll-wheel inputs. The rest of the circuit is just the usual smoothing capacitors (to remove glitches in the 5V supply and a few pull-up resistors to ensure a clean switch between Logic 0 and 1 signals. Note that because the layout is single sided, I was forced to use some wire bridges in the design, don't forget to add these, two of them are hidden under IC1 and IC2.

The PIC has two configurable ports RA and RB. In this circuit RB, which has 8 pins, is configured as all outputs, seven of them are used to control the signals to the CPC (four directions and three buttons), while the eight pin controls the LED. RA has only 5 pins, but it is a little more complicated, the first two pins are connected to the Clock and Data line of the mouse port, these change direction permanently from inputs to outputs, depending on which way the communications are going (yes, the mouse needs to get info too). The next two are always outputs and control the scroll wheel signals to the CPC. The last pin is an input connected to one of the small red dip-switches, which can be used to make a setting in the PIC, but more on that later.

Parts List

As I mentioned above, one of my priorities is always to keep the price as low as possible, and according to my local supplier, the cost of the entire board should come in at under 7€. Here’s the Parts list and their approximate prices:

Component	*Quantity*

PIC16F84A-04P	1
4 MHz Ceramic Resonator	1
74LS240	1
2N3904	2
100µf 16V Electrolytic Capacitor	1
100nf Ceramic Capacitor	1
10K ¼W Resistor	4
4K7 ¼W Resistor	3
470Ω ¼W Resistor	1
3mm LED Green	1
18 Pole IC Socket	1
20 Pole IC Socket	1
9 Pole D-Sub Connector 90°	1
6 Pole Mini DIN Socket	1
2 Pole DIP Switch	1
5V Power Socket	1
5V Power Plug	1
160×100×1.5mm 35µ Single Sided photo-resist PCB	1

Shopping cart with all parts for Reichelt Elektronik (Germany)

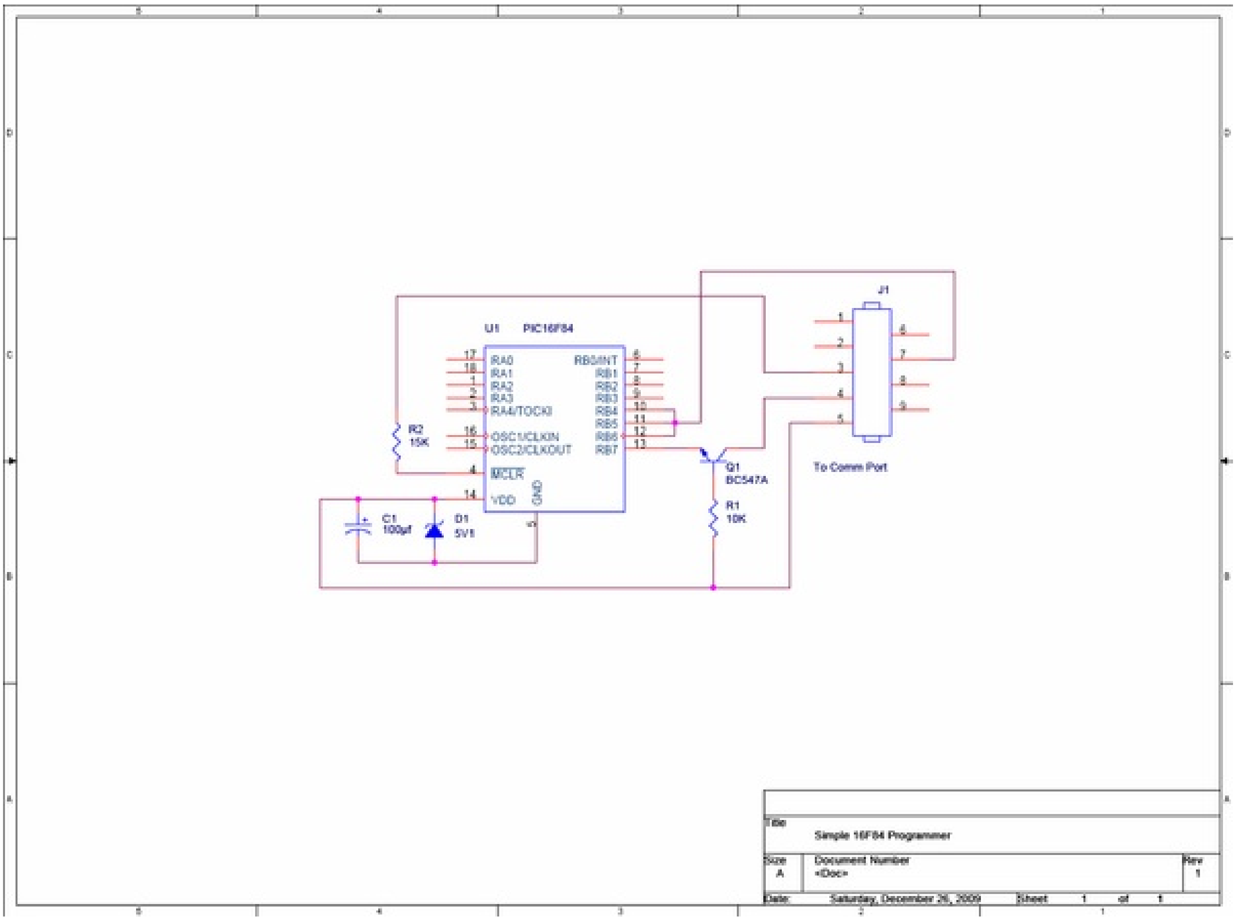
The Firmware

The firmware is the clever bit, it is written in PIC assembler. Its more or less a PS/2 driver with a little bit added to convert every movement into signals the Amstrad can work with, its first job is to get the mouse to life, it sends the required initialisation sequence and sets up some variables available in the mouse. Among other things, I have slightly de-sensitised the mouse, so that a single twitch doesn't send the arrow shooting across the screen. As well as this, it automatically recognises whether the mouse has a scroll wheel and turns it on if it does. After that, it just continuously asks the mouse if it has moved and collects the data the mouse sends back, this is then converted into the corresponding up/down/left/right and fire inputs that the CPC needs. I have also added plug and play detection, so if the mouse is unplugged at any time, the PIC recognises this and re-initialises the mouse when it has been re-connected. This also means that is is possible to swap between different mice (even between PS/2 and USB) during operation, without a re-start. The driver is purely a PS/2 driver, this means that even if a USB mouse is connected, it is being forced into PS/2 mode, so if the mouse does not support the PS/2 protocol it will not work with this device, but luckily all standard USB mice seem to still support PS/2. I haven't published the source code here, but if anyone is interested, I may be willing to share it for educational purposes.

Installing the Firmware

I have supplied a pre-compiled HEX file, so you don't have to worry about installing additional assemblers or having to compile things yourself. You will however need to get the program onto the chip. One of the simplest and cheapest ways of doing this is by building a JDM type programmer. This is a tiny PCB which plugs into the

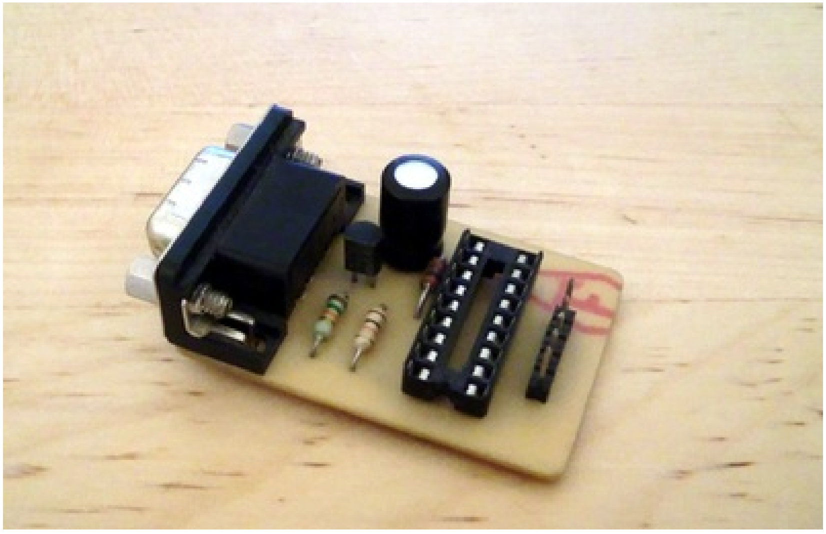
RS232 / Comm Port of a PC, in which you place the PIC and then program it using freeware such as PikLab if you’re a Linux user or ICProg if windows is your preferred PC OS. A quick warning: I'm told that the JDM doesn't work on Laptop Comm-ports because they don't supply the required 12V, so keep this in mind if you build one. The total price to build a JDM programmer should be under 2€. You'll find plenty of information about the JDM programmer and the programs mentioned above, by doing a quick search of the web, but to make things a little easier, I've included the circuit diagram of one popular design. This isn't my design, you'll find it all over the internet, but it's a good and extremely cheap way of getting your PIC programmed. I have also included a PCB layout for this in the resources section below. The layout also includes a 5 pin header connection which allows you to access the 5 pins required to program almost any PIC, so with a simple adapter, this circuit could be used to program a huge range of PICs and not just the 16F84. For future projects perhaps?



D1 is a 5.1V Zener Diode, the transistor is a BC547 and the Capacitor is a 100µf electrolytic capacitor (polarity is important with these). The two resistors are standard 1/4W resistors. A standard IC socket should be OK. Note also that there is a bridge wire beside the capacitor on the Sub-D connector side. The programmer takes it's power from the RS232 port and needs no external supply.

When programming the 16F84 using ICProg or PikLab, the Oscillator and Fuse settings need to be correct. The correct settings for this project are: Oscillator = XT, WDT = Unchecked (off), PWRT = Checked (On) and CP = Unchecked (Off). Make sure these are set properly before programming the PIC, otherwise your mouse adapter won't work.

The finished programmer should look something like this:



Construction Tips

As with the original AMX Mouse, the only connections needed to the CPC are through the joystick port and to the 5V supply. I have kept the PCB size to a minimum, to allow the board to be connected directly to the joystick port without getting in the way too much or tearing the connector off the side. You may choose an alternative layout to mine, but remember that if it is too large, it may damage the joystick port and would be better with a fly-lead connection.

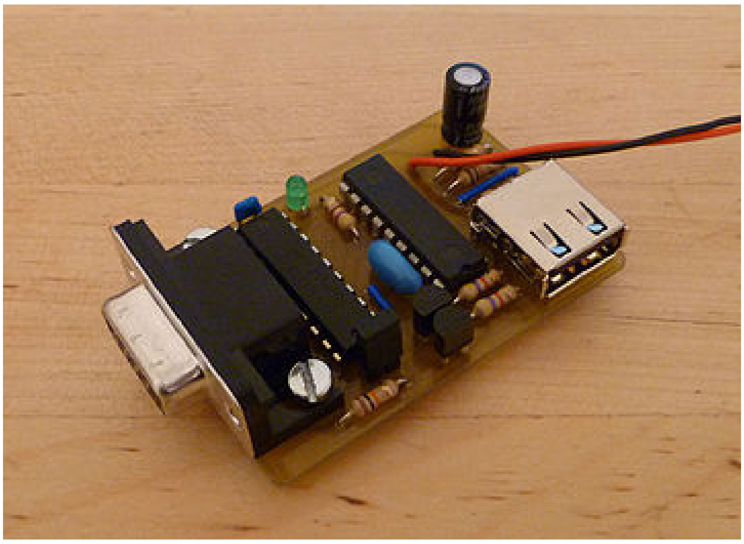
The board presented here measures just 38mm x 60mm and should cause absolutely no stress to the CPCs connector. As well as that, remember that there are connectors left and right of the joystick port that you might want to use simultaneously, the layout here has been designed to keep both the sound socket and tape port free.

A note to beginners: When soldering the parts to the board, always solder the most heat resistant parts first and leave the sensitive ones to last, this avoids destroying a successfully soldered transistor while trying to solder a resistor that’s right beside it. Also, although you may notice that I have soldered the 74LS240 into place, I wouldn’t advise beginners to try this, you will destroy it. Splash out on the extra 4 cents it costs for a suitable socket. If you’re an absolute beginner, there are even sockets available for transistors, but the spacing on this layout probably isn’t correct for them. Also remember that there are wire bridges under each of the ICs! These are really difficult to assemble, if the socket or IC is already soldered into place!

The 9 way Sub-D connector usually has two small bolts protruding left and right of the connector, if you remove these, the metal surround will fall off. I used some superglue to re-attach the metal part. It will work without the metal surround, but this will cause the circuit to bend the CPCs joystick pins over time, so I would recommend you do the same. I also added two M3 screws to hold the Sub-D connector to the PCB, this takes the stress off the pins below the connector, which could otherwise develop cracks in the solder over time as well.

If you intend only using USB mice with the device, it may make more sense to use a USB socket instead of the PS/2 socket plus adapter. I have now added a pure USB version layout below, the finished device looks like this:

The USB version has just one jumper to enable / disable the scroll wheel, the Auto-detect is always active.

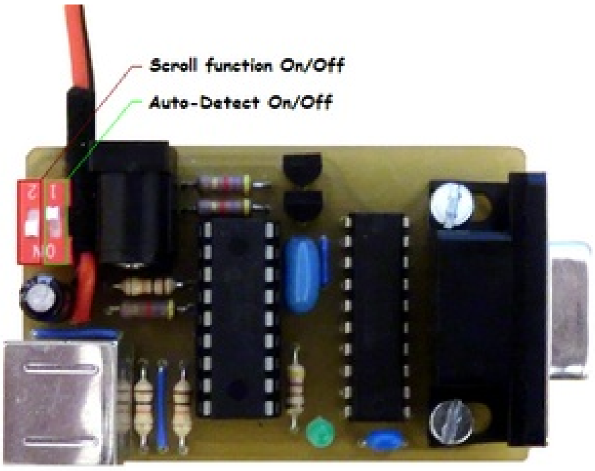


Using The Mouse Adapter

There are two switches on the PCB, the inner one (beside the power socket), should in most cases be left in the OFF position. This is the Auto-detect position. The firmware will auto-detect which type of mouse is connected and initialise it as required, if however you have a mouse which is not being recognised correctly (Error lamp flashes or incorrect output), switching this switch ON will tell the PIC not to auto-detect and to assume a type 00 mouse (standard mouse with no scroll wheel or extra buttons) and may allow you to continue using the mouse. The position of this switch should be chosen before power is applied. In some cases, although you have chosen auto-detect, the PIC may recognise that the mouse is incompatible and dis-enable the scroll wheel although one is present, the mouse should however continue to work as a non-scroll wheel mouse.

The second switch is for turning the scroll wheel signals on and off. Switching this switch ON allows the PIC to pass the scroll wheel movements on to the CPC. If a correctly recognised scroll wheel mouse is connected and this switch in ON, movement to the scroll-wheel will cause a “5” (Joy2 Down) for each roll upwards and a “6” (Joy2 Up) for each scroll downwards, this may have negative effects with software which has assigned these characters to other functions, so for this reason I chose to include the option to block it. This switch can be used at any time without resetting.

The 5V power lead that usually goes to the CPC should be connected directly to the Adapter PCB, the fly-lead is then plugged into the CPC. When the power is applied, the PIC will try to recognise and initialise the mouse. If recognition and initialisation is successful, the LED will blink once for approximately one second. After this any movement of the mouse will make the LED blink. If the mouse fails to initialise (there may be incompatible ones out there or it may just be broken) the LED will flash continuously in one second intervals.



Mapping to Joystick Signals

```
 Row9.Bit0 Joy1up    LOW when mouse moved up
 Row9.Bit1 Joy1down  LOW when mouse moved down
 Row9.Bit2 Joy1left  LOW when mouse moved left
 Row9.Bit3 Joy1right LOW when mouse moved right
 Row9.Bit5 Joy1fire1 LOW when Right mouse button pressed
 Row9.Bit4 Joy1fire2 LOW when Left mouse button pressed
 Row9.Bit6 Joy1fire3 LOW when Middle mouse button pressed
 Row6.Bit0 Joy2up    LOW when mouse wheel turned towards user (standard CPC only)
 Row6.Bit1 Joy2down  LOW when mouse wheel turned towards screen (standard CPC
only)
```

Testing The Mouse

In BASIC, moving the mouse should produce the arrows usually seen when a joystick is connected. Pressing the left button will produce an X and the right button produces a Z. The centre button will not show any visual feedback in Basic, but can be sensed using the Inkey(78) command.

Compatibility

Like I mentioned above, I've never seen an AMX mouse, so I don't know exactly how they react to movement, how fast they were or how they "felt". Technically speaking, the mouse adapter described here is fully AMX compatible and works perfectly with AMX compatible packages. I have tuned the feel of the mouse to closely resemble a PC mouse, but if you build this and find that it's not quite the "AMX feel", let me know and I might be able to tune it back to the 80s. And remember, if you are used to only using a PC mouse, don't expect an 8-Bit mouse to have the same lightening response time, moving the mouse really fast will not have the same result as on a PC, the Amstrad can only read the Joystick port at a certain rate, so moving the mouse slower will give a much smoother and accurate result.

The adapter should work with any PS/2 compatible mouse and almost any USB mouse using a USB to PS/2 adapter plug. If you have a fancy 28 Button Star-Fighter special edition all singing and dancing cyber mouse, don't expect anything more than the basic movements to work.

All wireless PS/2 mice should work perfectly normal, but Bluetooth USB mice will definitely NOT work with this adapter.

I have tested the hardware with the following mice:

(Many thanks to my good friend Knut, who was kind enough to lend me a variety of mice from his enormous "All Things x86" collection)

Yes that’s a wireless PS/2 mouse on the right. (Is that the first wireless CPC mouse ever? Who knows.)

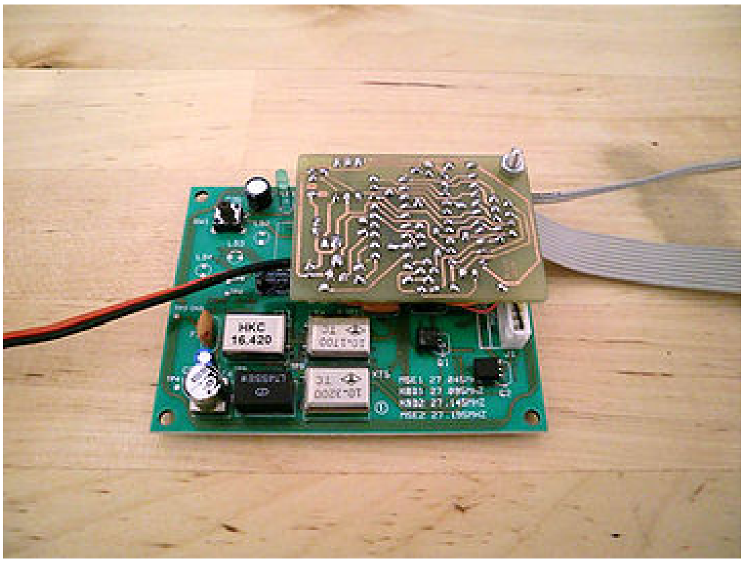
Resources

Wireless Internal Version

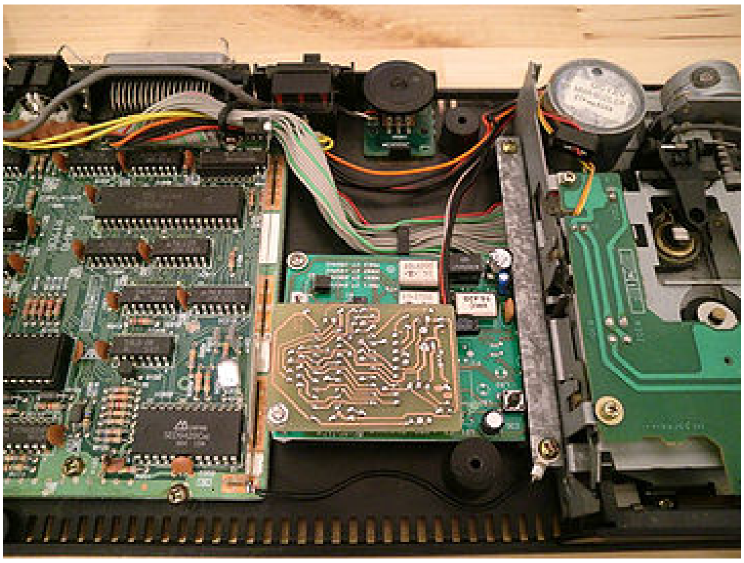
Using the exact same circuit as above, I decided to create an internal wireless mouse for my 6128. The wireless PS/2 mouse receiver I've used is a Logitech C-RA1 seen here in it's original state, which can often be found for a reasonable price at your local flea-market or on your favourite online auction site, but any wireless PS/2 mouse and receiver should work.

The circuit remains the same, except for the fact that I didn't solder any sockets on to the PCB, but added fly-leads for the power, LED, PS/2 and Joystick port connections. I used flat cable for the joystick port connects, so that it can easily be routed below the CPCs mainboard. After removing the receiver circuit board from it's housing and removing the PS/2 lead, I connected the PS/2 wires directly from the receiver board to the adapter PCB and mounted the adapter PCB inverted on top of the receiver.

Unfortunately this particular model of wireless mouse has a relatively large receiver board, due to the receiver also having been used for the Logitech wireless mouse and keyboard combo. Despite this, there's room inside the 6128 to house everything, but you may notice I needed to relocate the earth strap from the mainboard to the drive bracket in order to make space. Alternative receivers might fit better however. The flat cable is routed below the mainboard and soldered directly to the contacts below the joystick port socket.



At the rear of the CPC I added a power switch for when I don't intend using the mouse. I took the required 5V from the lead going to the disc drive. Any of the two black wires is ground, the orange wire is 5V. To avoid drilling even more holes in the original housing, I used a 3mm LED filed down a little and wedged it between the cooling grill at the top edge. Then all that was left to do was put everything back together and my 6128 was fitted out with a wireless AMX compatible mouse.



Compatible Software

Further Info

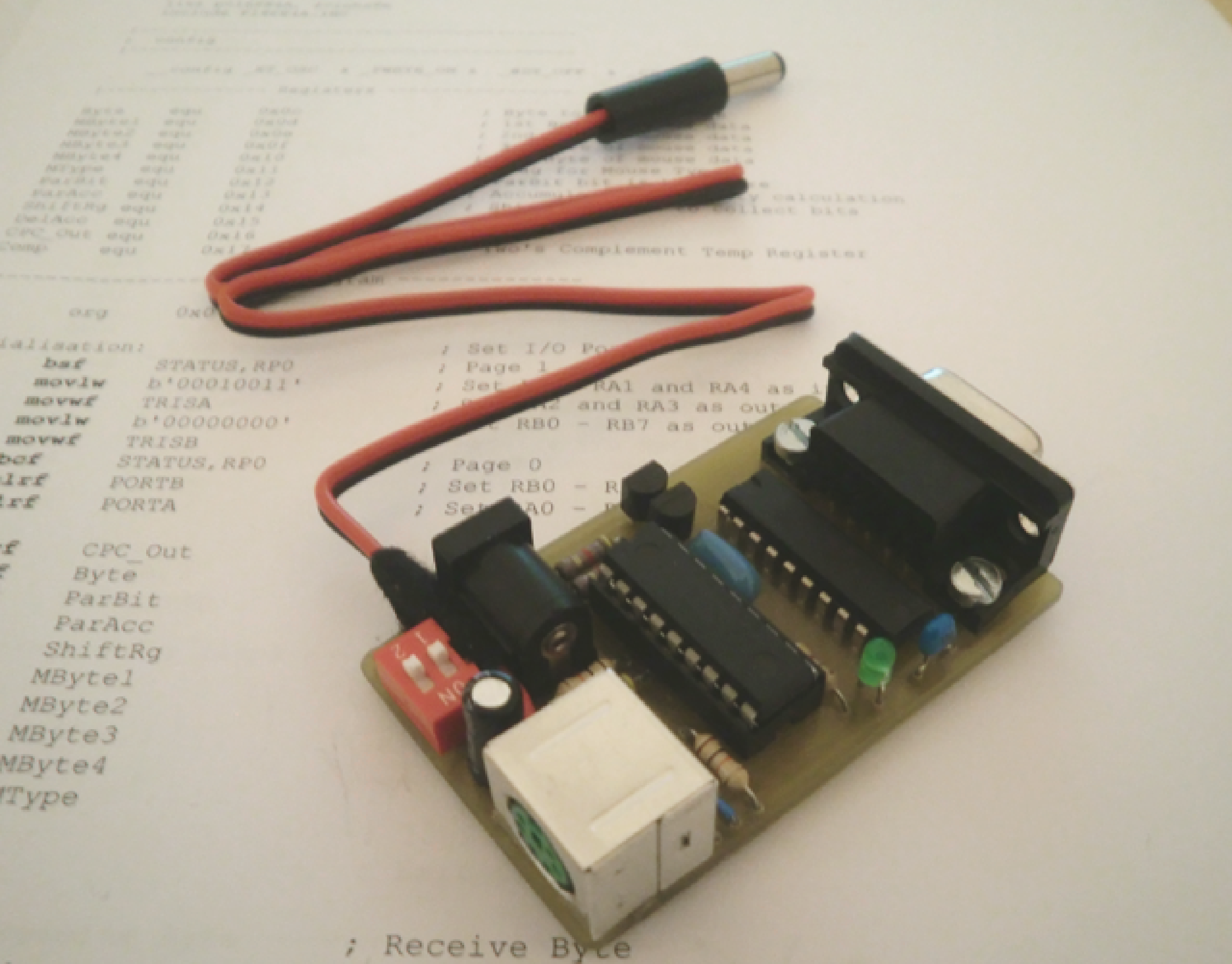
Version 1.0 of the Firmware is here. This version adds Plug & Play functionality, auto-detection for the type of mouse and includes some slight tuning to improve the mouse reaction time. The schematic and layout have not been changed.

Update: A minor change to the Schematic has been made to add a 10K pull-up resistor to the OE inputs of the

74LS240. This addition should make the circuit more robust in the very unlikely event of the adapter being used in conjunction with other hardware attached to the Mouse port either internally or externally. Should you already have built the adapter, there's is no need to add this extra component unless you are experiencing problems with the reliability of the adapter. The layout and component placement files have been updated to reflect this change.

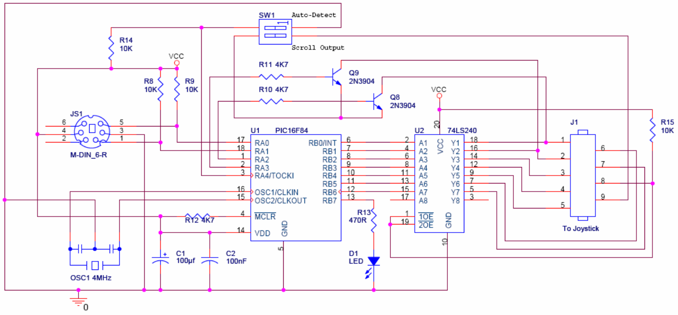
If you have any further questions regarding this project, please contact me through the Wiki Forum.

Bryce.



```
;   config
__config _XT_OSC & _PWRTE_ON & _WDT_OFF &
;--------------- Registers ---------------
  Byte     equ     0x0c          ; Byte
 MByte1    equ     0x0d          ; 1st         data
 MByte2    equ     0x0e          ; 2nd         data
 MByte3    equ     0x0f          ;       yte of Mouse data
 MByte4    equ     0x10          ;       Byte of Mouse data
 MType     equ     0x11          ;     g for Mouse Type
 ParBit    equ     0x12          ;   ParBit bit
 ParAcc    equ     0x13          ;   Accumul            y calculation
 ShiftRg   equ     0x14          ;                  collect bits
 DelAcc    equ     0x15
 CPC_Out   equ     0x16
 Comp      equ     0x17          ;   Two's Complement Temp Register

        org     0x0

ialisation:                      ; Set I/O Po
  bsf       STATUS,RP0           ; Page 1
  movlw     b'00010011'          ; Set     RA1 and RA4 as i
  movwf     TRISA                ;       2 and RA3 as out
  movlw     b'00000000'          ;   t RB0 - RB7 as ou
  movwf     TRISB
  bcf       STATUS,RP0           ; Page 0
  lrf       PORTB                ; Set RB0 - R
  lrf       PORTA                ; Se  A0 -

 f      CPC_Out
 f      Byte
        ParBit
        ParAcc
        ShiftRg
        MByte1
        MByte2
        MByte3
        MByte4
        Type

                         ; Receive Byte
```

VOLUME
POWER
OFF
ON



Title

Amstrad CPC PS/2 / USB Mouse Adapter

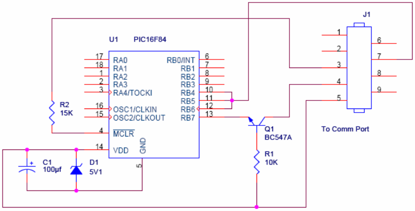
Size A

Document Number <Doc>

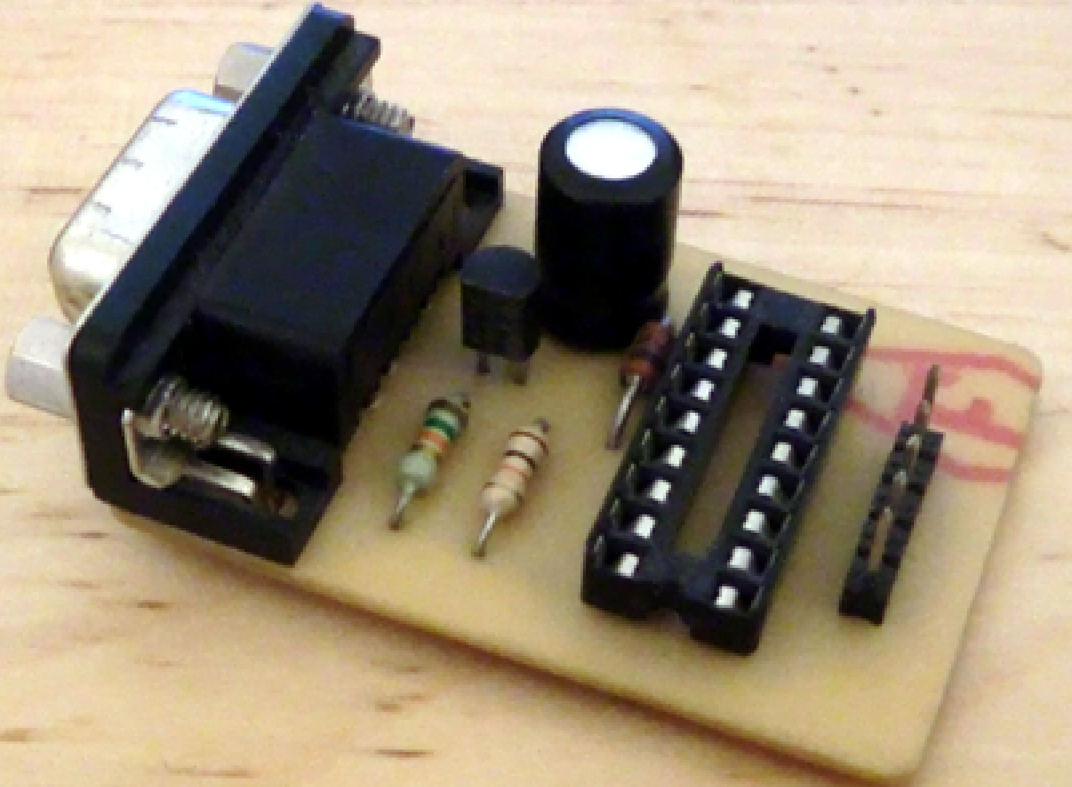
Rev 1.6

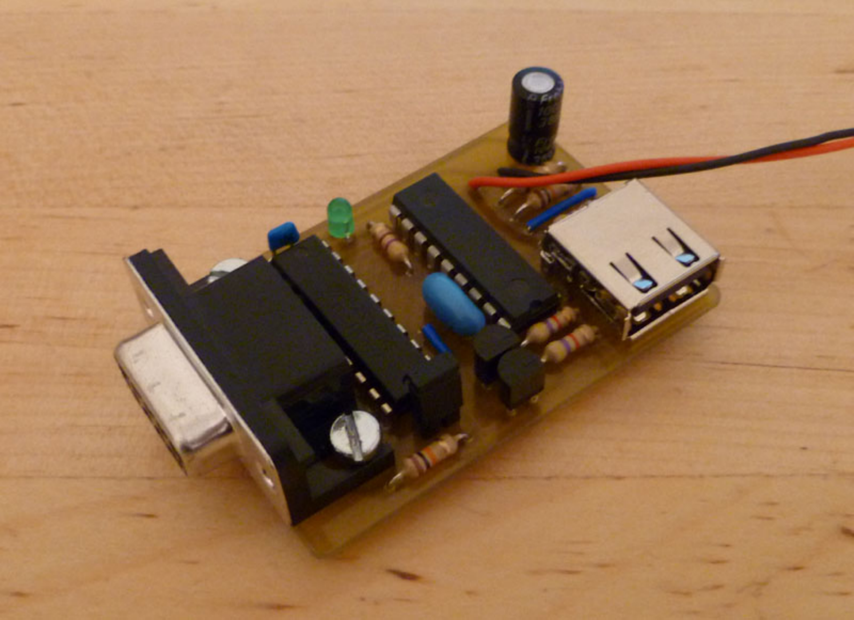
Date: Monday, May 31, 2010

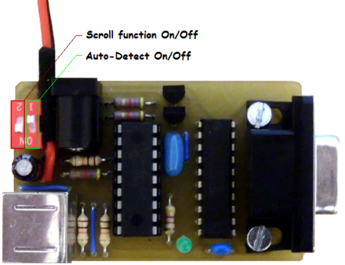
Sheet 1 of 1

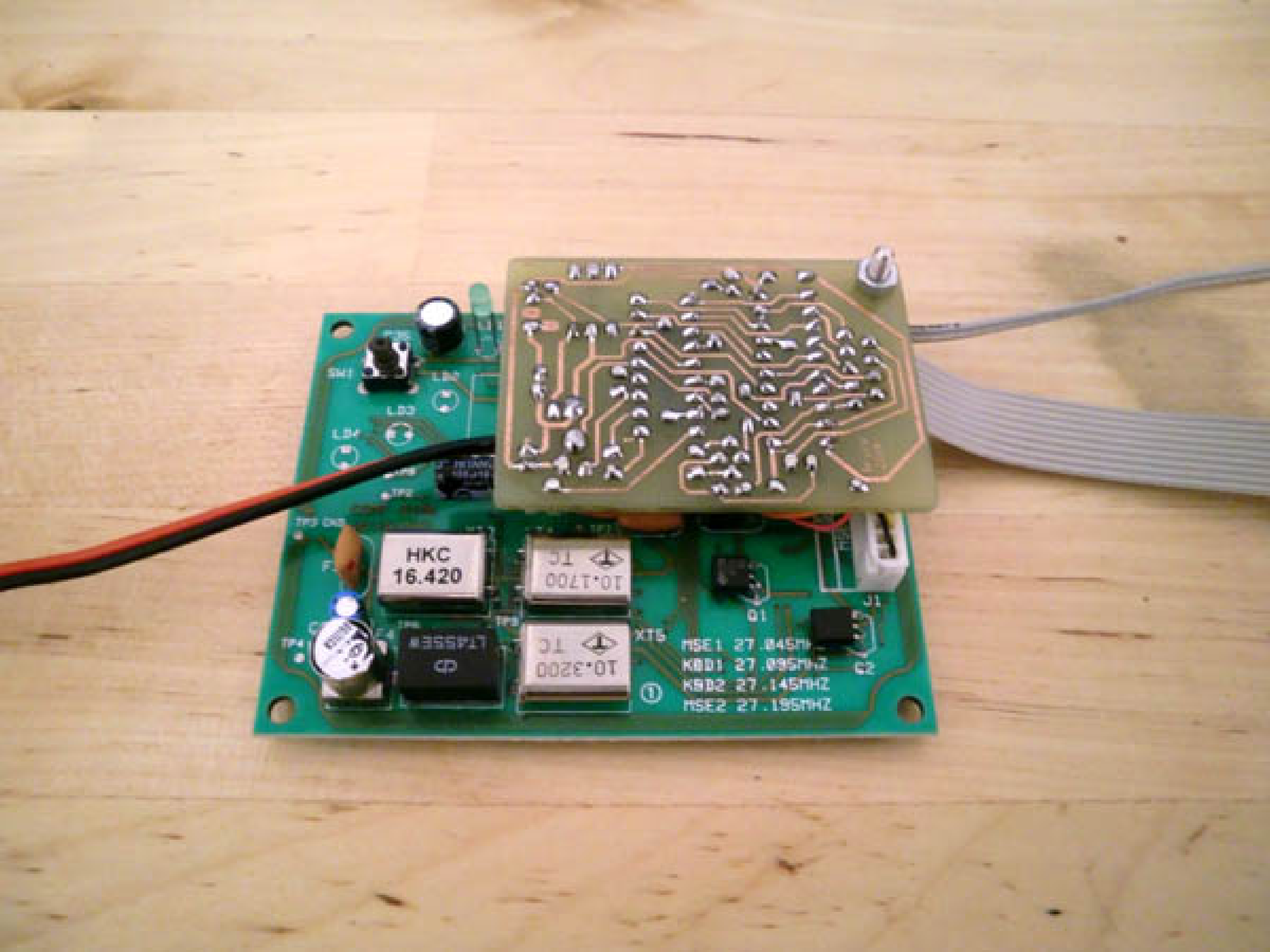


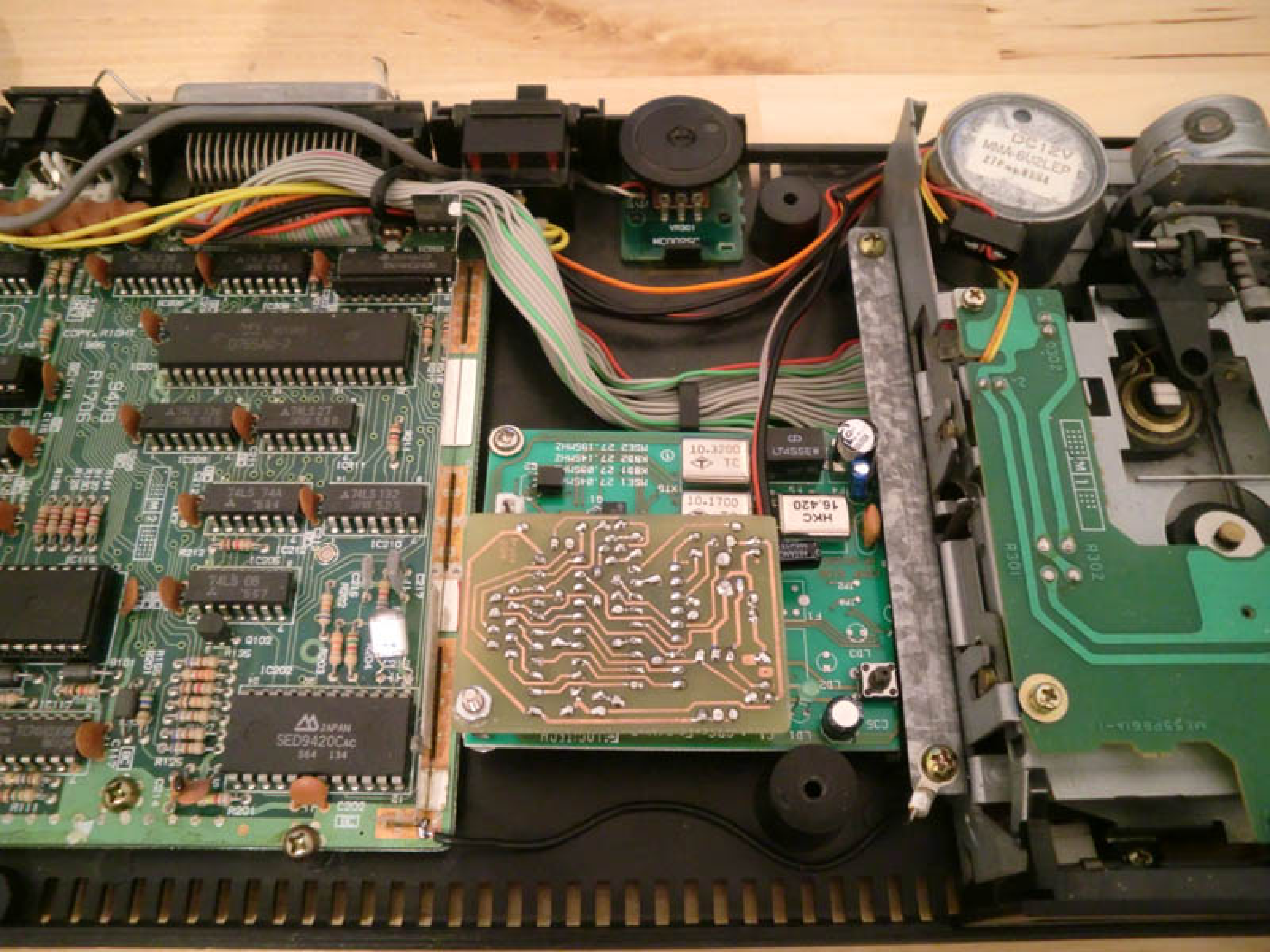
Title	Simple 16F84 Programmer	
Size A	Document Number <Doc>	Rev 1
Date:	Saturday, December 26, 2009	Sheet 1 of 1






Scroll function On/Off
Auto-Detect On/Off
2
1
ON


SW1
LD2
LD3
LD4
TP4
HKC
16.420
10.1700
TC
10.3200
TC
XT6
Q1
J1
C2
MSE1 27.045MHZ
KBD1 27.095MHZ
KBD2 27.145MHZ
MSE2 27.195MHZ


DC12V
COPY. RIGHT
74LS 74A
74LS 132
74LS 27
74LS 08
JAPAN
SED9420CAC
364 134
10.3200
10.1700
HKC
16.420
LT4558
R301
R302