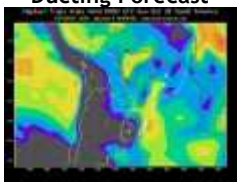
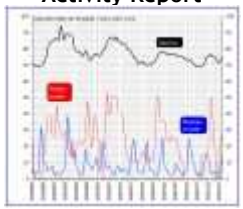
Home
Repetidoras
Arquivos
Projetos
Satélites
QSL's
Links
Amigos
Contato

VHF / UHF Tropospheric Ducting Forecast



L 1.4 2 3 4 5 6 7 8 9+
By: William Hepburn

Solar Terrestrial Activity Report

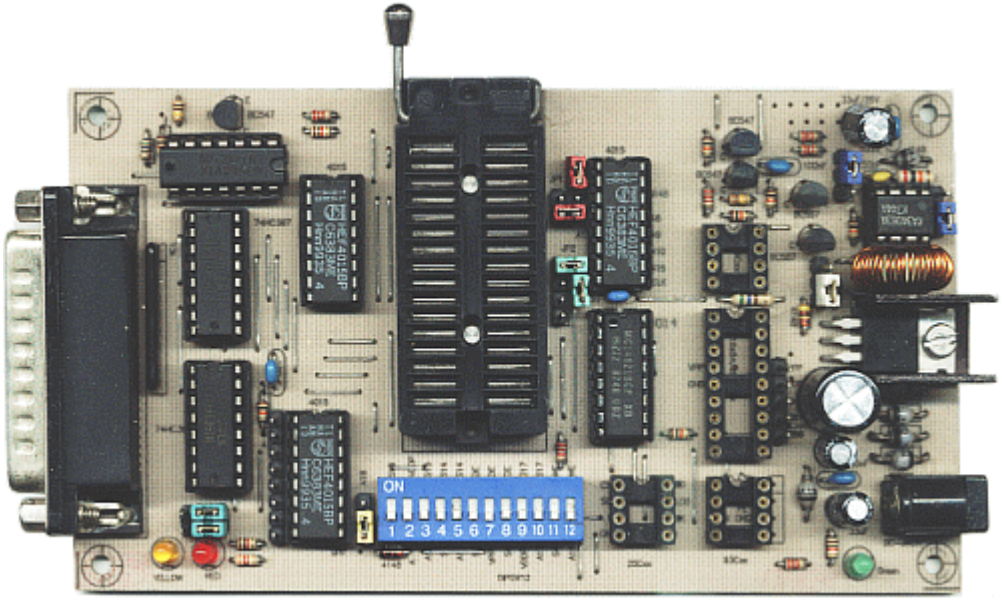


http://www.dxlc.com

MUF Map

English version here.

Convertendo o Programador Willem PCB3b para PCB4.5



Para quem não conhece este programador e meu envolvimento com ele, uma pequena história. O projeto original é e autoria de Willem Kloosterhuis e seu site pode ser visto aqui: http://www.willem.org/

Posteriormente uma versão mais compacta (a famosa PCB3b) e diversos adaptadores para outros dispositivos, foram criados por Gitti Ieo, e pode ser visto aqui: http://se-ed.net/mpu51/eprom/eprom.html

Todas as ultimas versões do software para windows, foram feitas por Gitti Ieo, e a ultima versão é a **0.97ja**

Pois bem, tive contato com Gitti Ieo, por volta de Maio/2002, onde enviei algumas sugestões para o software e alguns bugs encontrados. A partir desta data, Gitti me incluiu como beta-test de novas versões do programa, e varias idéias foram enviadas (assim como bugs) e algumas foram efetivamente aproveitadas. Recebi versões beta de Gitti até Agosto/2004, quanto recebei a ultima versão beta, no caso a 0.97jc3, qual não funcionou corretamente. A partir desta data não tive mais noticias de Gitti e

10 últimos QSL's Recebidos


PY2001SWL-AO-7


PY2001SWL-SO-50

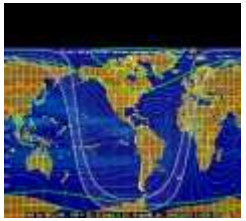

FY0JJ-AO-51


PY5RX - AO-51


ZZ2VJG - AO-51

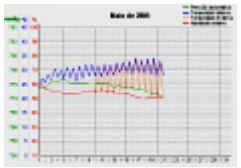

PY1AT - VO-52 SSB


PY1AT - FO-29



http://www.spacew.com

Observações Meteorológicas Locais (GG68oa)



Mês:

1 2 3 4 5 6 7 8 9 10 11 12

By:Luciano - PY2BBS

Previsão do tempo

Contato Recorde:

PT9JA
José Antônio
VHF SSB

GG68rw -> GG49df

563.1 Km

PP5JAK

qualquer noticia de atualizações do software.

Quase um ano depois, vim a descobrir uma versão melhorada do willem montada e vendida neste site: www.sivava.com e qual não foi a minha surpresa ao ver que o software era o mesmo, porem com diversas atualizações, e com dois grandes detalhes: Não estava mais disponível ao publico e não era mais compatível com a versão PCB3b da placa, sendo somente compatível com as novas versões (a atual é a PCB4.5C).

Para mim foi uma decepção, após contribuir com o desenvolvimento do software, mesmo que muito pouco, ver este novo rumo que a coisa tomou. E principalmente a parte do texto que esta no próprio site do sivava:

> "*(NEW!!! The PCB4.5C design by an engineer who developed (Original Version : (Willem EPROM Programmer PCB3B Standard and Universal)) and developed software 0.98D2 / (Now!! Download Free 0.97ja) /0.97j / 0.97i+ / (0.98i more Copy Software version) / 0.97i / 0.97h / 0.97g)*"

Resumindo, o próprio Gitti estava envolvido na atualização. E nada das versões atuais disponíveis para download, somente as antigas e já conhecidas. Porem, pesquisando um pouco no google, acabei encontrando em um fórum russo, uma cópia do software atualizado, no caso a versão 0.98D2.

Ao testa-lo logicamente não funcionou na antiga PCB3b, como já alertado. Então a partir deste ponto minha "missão" resumiu a fazer engenharia reversa através do software para levantar quais foram as modificações e tornar compatível a velha PCB3b, com o novo software.

Bem, então vamos a modificação propriamente dita. Antes irei incluir duas modificações feitas por min, que visam melhorias no programador, a qual recomendo a todos fazerem antes mesmo de atualizar a placa para a versão PCB4.5

Serão descritas quatro melhorias, sendo:

- **Melhorando o conversor DC-DC.**

O calcanhar de Aquiles desta versão do willem é o conversor DC-DC, que gera as diferentes tensões de Vpp (12.5, 15, 21 e 25V). O principal item que causa uma série de problemas é a bobina do conversor, a qual deve ter corpo suficiente para não saturar com a corrente exigida pelas eprom na programação. O maior problema é na programação de eproms antigas, as quais exigem muita corrente no Vpp.

A corrente máxima fornecida pelo conversor na minha placa era de apenas **35mA** aproximadamente, o que é pouco para algumas eproms.

E conseguir uma bobina de 100uH que suporte uma corrente de até uns **100mA** sem saturar é algo complicado, pois a maioria dos choques de RF encontrados no comércio tem núcleo muito pequeno e satura rapidamente com correntes desta ordem. Solução?

Enrolar uma nova bobina usando um núcleo toroidal. O local para encontrar um núcleo ideal é a caixa de sucatas. Procurando encontrei núcleos adequados em velhas placas mãe de PC e em placas de carregadores para celular, para uso veicular. Inclusive

ZZ2NGB - ISS

PY4ZBZ - ISS

PY4ZBZ - AO-51

Mais...

10 últimos QSL's Enviados

ZZ2VJG - AO-51
PY2001SWL - FO-29
PY1SAN - AO-51
PY1SAN - SO-50
PY1SAN - VO-52
PY2CDS - AO-7
PY5RX - AO-7
LW3DRH - AO-51
LW3DRH - FO-29
LW3DRH - VO-52
HK4MKE - FO-29
PY1AT - AO-7
PY5CAM - FO-29

Mais...

Seti@Home

Total Credits: 33218.97

Get Firefox!

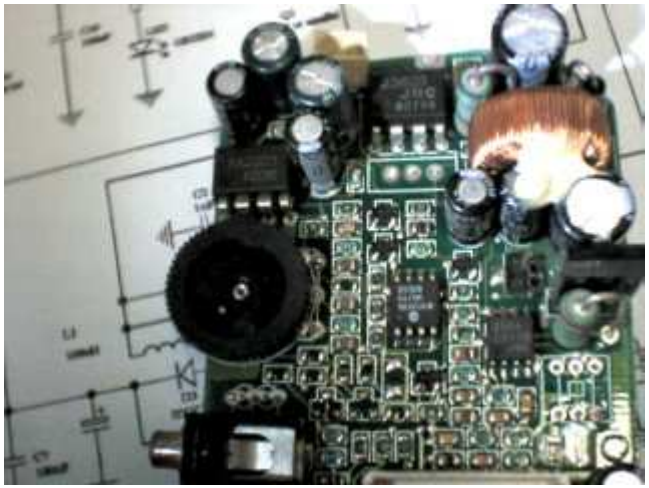
1024 x 768 True Color

José Alcir Kubiak
VHF Rep

GG68rw -> GG42rr

701.0 Km

nestes últimos é encontrado também o **MC34063** ou o **JRC2360D** que é seu equivalente direto, feito pela *Japan Radio Co. LTD.* Na foto baixo veja uma placa destes carregadores.



A bobina pode ser aproveitada diretamente, basta retirar e colocar na placa. No caso de re-aproveitar um ferrite de placa mãe, a bobina devera ser re-enrolada. Veja o mesmo ferrite em uma velha placas PC-Chips SiS530:

Esta são as medidas do ferrite para quem quiser procurar um similar em outra placa:

- Altura: 5mm
- Diâmetro interno: 8mm

- Diâmetro externo: 13mm
- Cor do núcleo: Amarelo e uma borda branca.

Retire todo o fio que tem nele, e enrole 55 espiras de fio esmaltado 24AWG distribuídas uniformemente sobre todo o ferrite.

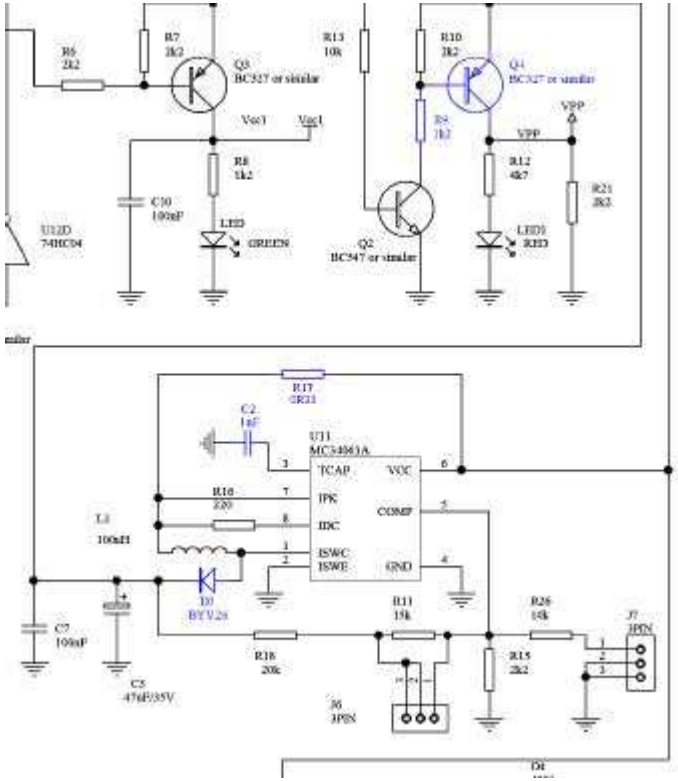
Outras modificações que visam uma melhor performance no conversor DC-DC:

Substitua:

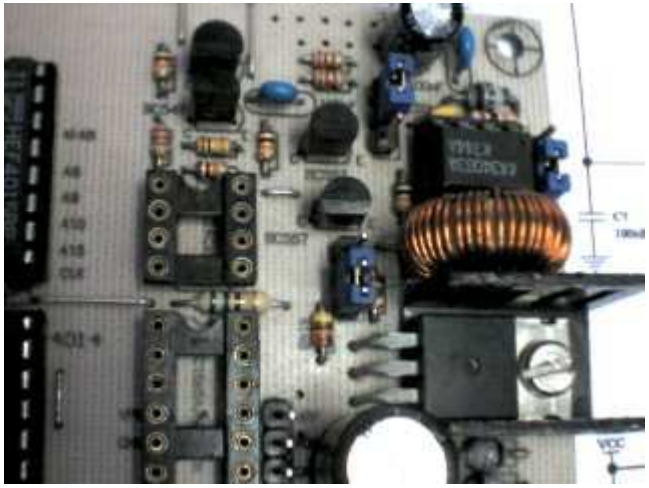
- Transistor **Q4**, original: **BC557**, troque por **BC327** ou **BC328**.
- Resistor **R9**, original **2K2**, troque por **1K2**.
- Resistor **R17**, original: **0.5Ω**, troque por **0.33Ω** Caso não encontre, use minha solução, 3 resistores de 1Ω em paralelo, o que nos dará exatamente 0.33Ω.
- Capacitor **C2**, original: **220pF**, troque por **1nF**, visa abaixar a freqüência de chaveamento.
- Diodo **D3**, original: **1N4148**, troque por **BYV26**, **BYV95C** ou qualquer outro diodo scotky.

O regulador de tensão, **7805** ira esquentar um pouco mais que o normal, é conveniente colocar um pequeno radiador de calor. E finalmente, use uma BOA fonte de alimentação, estou utilizando uma fonte de um scaner velho, de 12V x 1200mA.

Com estas modificações, a corrente fornecida pelo conversor DC-DC, passa da casa dos **100mA**. Trecho do esquema onde foram feitas as modificações, os componentes em azul são os citados acima.



E o setor da placa onde estão todos, exceto o resistor **R9**, que fica entre o **74HC04** e o soquete ZIF. O resistor **R17**, ficou em embaixo da bobina.



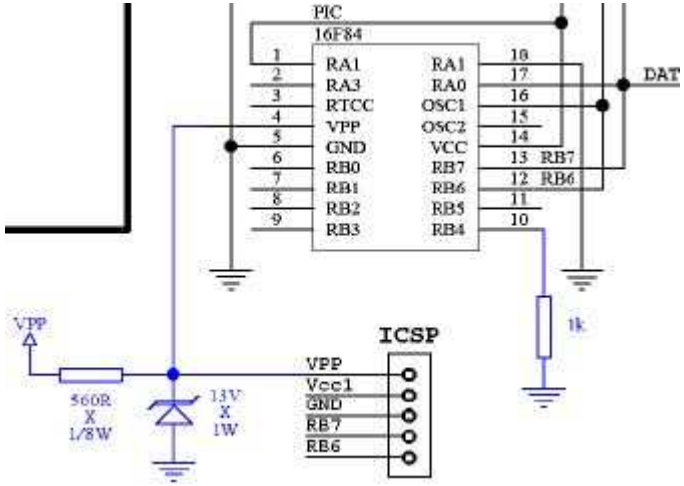
- **Protegendo os PIC contra erros no ajuste da tensão de Vpp.**

Os pobres dos pics são as maiores vitimas caso você mude a tensão de Vpp para 21 ou 25V e esqueça de retornar os jumps para a posição de 12.5V.

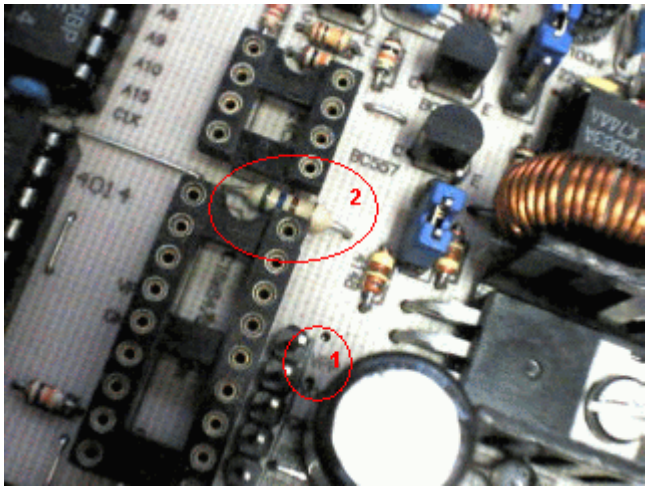
Eu mesmo já cometi este erro duas vezes, fritando dois PIC16F877A.

A solução para isto é simples, basta adicionar um resistor em série com a linha de Vpp, que vai para o soquete ICSP e para o soquete de 18 pinos. E um diodo zener para limitar a tensão em um valor seguro.

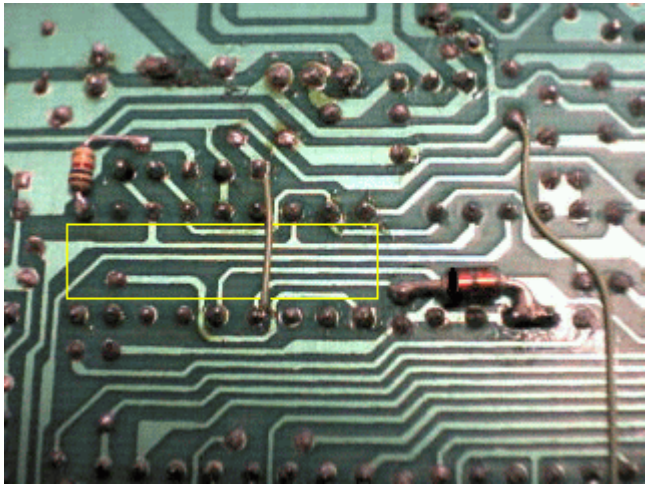
Se você já fez esta modificação aqui, é necessário mover o fio para a outra extremidade onde estava o jump, pois caso contrario neste ponto a modificação estará ligada ao Vpp protegido de 12V dos pics e não mais ao Vpp direto após a modificação. Na figura já coloquei o resistor (em azul) que será adicionado no lugar do jump.



A modificação na placa é bastante simples, não é necessário cortar nenhuma trilha, basta remover dois jumps e proceder a modificação. Na foto abaixo mostro a posição dos jumps, e note que já esta instalado o resistor de **560R** no lugar de um dos jumps, o outro jump fica vazio.



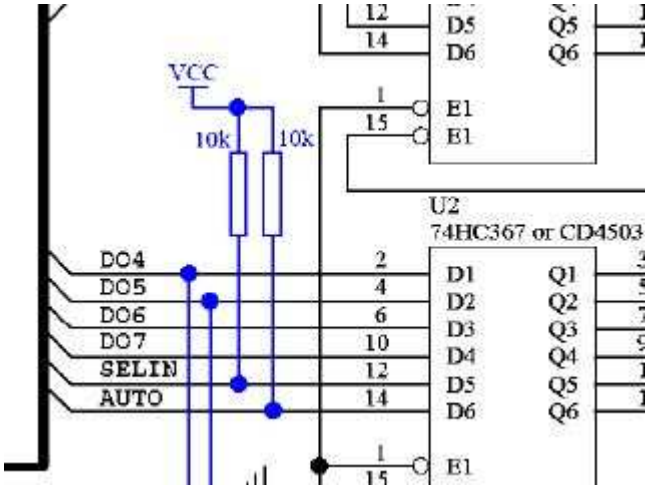
Dentro do circulo 1, foi removido um jump e no circulo 2, foi removido o jump e instalado o resistor de **560R x 1/8W** em seu lugar. Agora falta somente uma pequena ponte do lado debaixo da placa e instalar o diodo zener de **13V x 1W**, conforme a foto abaixo.



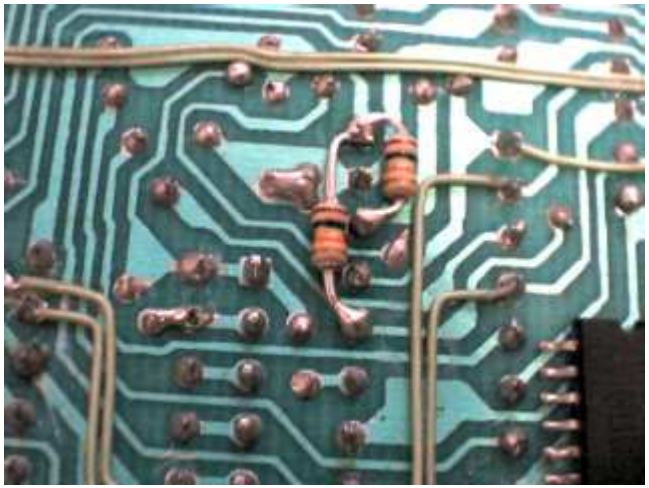
O retângulo amarelo indica o soquete de 18 pinos. O fiozinho que passa sobre o soquete de 18 pinos é a ponte que deve ser adicionada. Já aproveite e coloque o resistor de 1K que esta no pino 10 do soquete, ligado ao GND. Este resistor aterra o pino **Low Voltage Program**, dos PIC16F628. O diodo zener esta com seu anodo soldado aos pinos 1 e 2 do soquete da 24Cxx. Com isto esta modificação esta completa.

♦ Corrigindo o pull-up dos pinos SELIN e AUTO.

No projeto original ficou faltando o pull-up nos pinos **Select In** e **Auto**. Embora este erro já tenha sido alertado pelo próprio Gitti e também no fórum do Willem, resolvi coloca-lo aqui. Veja no esquema esta correção. Não é uma correção critica, mas é aconselhável de fazer.



E veja aqui o local para colocar os resistores na placa. Eles estão ligados ao pino 16 do próprio CI U2.



Não ligue para a confusão de wire-up e o CI no canto inferior direito. Eles fazem parte da ultima modificação.

Melhoras para funcionamento em porta LPT de notebooks

Pode parecer estranho, mas vários notebooks tem diferenças no padrão da porta paralela, usam pull-up de valores mais baixo, para prover uma porta com maior imunidade a ruídos e mais rápida.

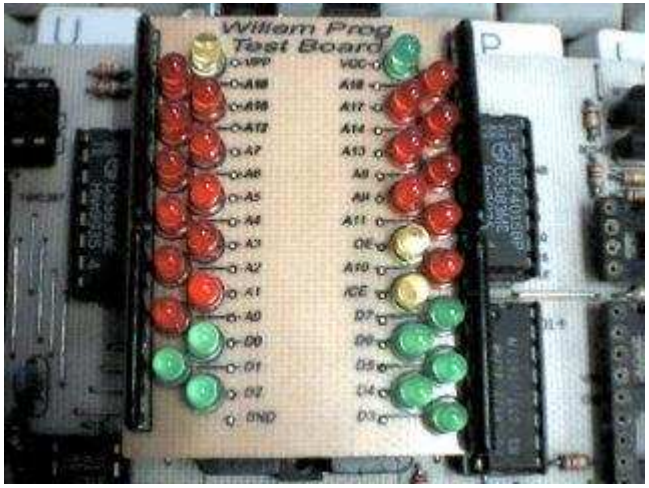
Só que isto causa problemas de instabilidade e até mal funcionamento.

Isto deve ao fato dos CI's da séries CD40xx suportarem um dreno de corrente menor. A dica é substituir os buffers CD4503 e 4069 por 74HC367 e 74HC14, que suportam um dreno muito maior.

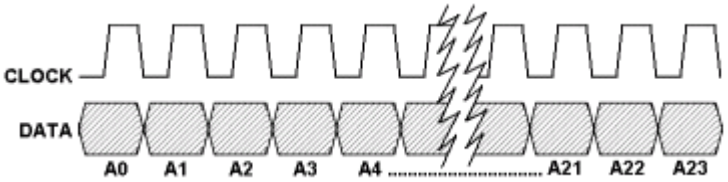
Meu programador não funcionava corretamente em meu notebook, um IBM ThinkPad. Apos a troca dos buffers passou a funcionar perfeitamente.

Conversão para PCB4.5 ou onde a porca torceu o rabo! :)

Quando me propus a descobrir como converter a antiga PCB3b para funcionar com o software novo, minha primeira providencia foi fazer uma pequena giga de teste, para facilitar o debug, desta forma fica muito mais rápido para verificar as alterações dos sinais no soquete ZIF.



O Willeprom usa uma cascata de 3 shift registers duplos (CD4015) e carrega os 24 bits com compõe o barramento de endereços, tal como na figura abaixo.



Ou seja coloca um bit por vez no pino data do primeiro 4015 e vai pulsando a linha de clock, efetua esta operação 24 vezes até carregar todo o barramento (obs. no caso de um barramento menor, como de uma 27C512, por exemplo ele carrega somente os bits necessários, a saber, 16 bits), mas o que importa é que ele carrega um bit por vez.

Ao iniciar o debug, usando o teste de hardware com minha giga de leds, notei que o software carregava corretamente os bits de **A0** a **A7**, porem de **A7** acima, não carregava nada e eventualmente aparecia "lixo" no nos endereços acima de **A7**.

Uma observação cuidadosa, permitiu notar que ao carregar unicamente o bit **A0**, ao selecionar pela primeira vez o bit **A0**, era carregado corretamente, porem ao des-selecionar, o bit A0 apagava, porem o bit A8 acendia, e ao selecionar novamente o bit **A0**, o bit **A8** apagava e acendia o bit **A0** e **A16**...

Pensando um pouco, basta notar que somente estão sendo carregados 8 bits, acompanhe abaixo:

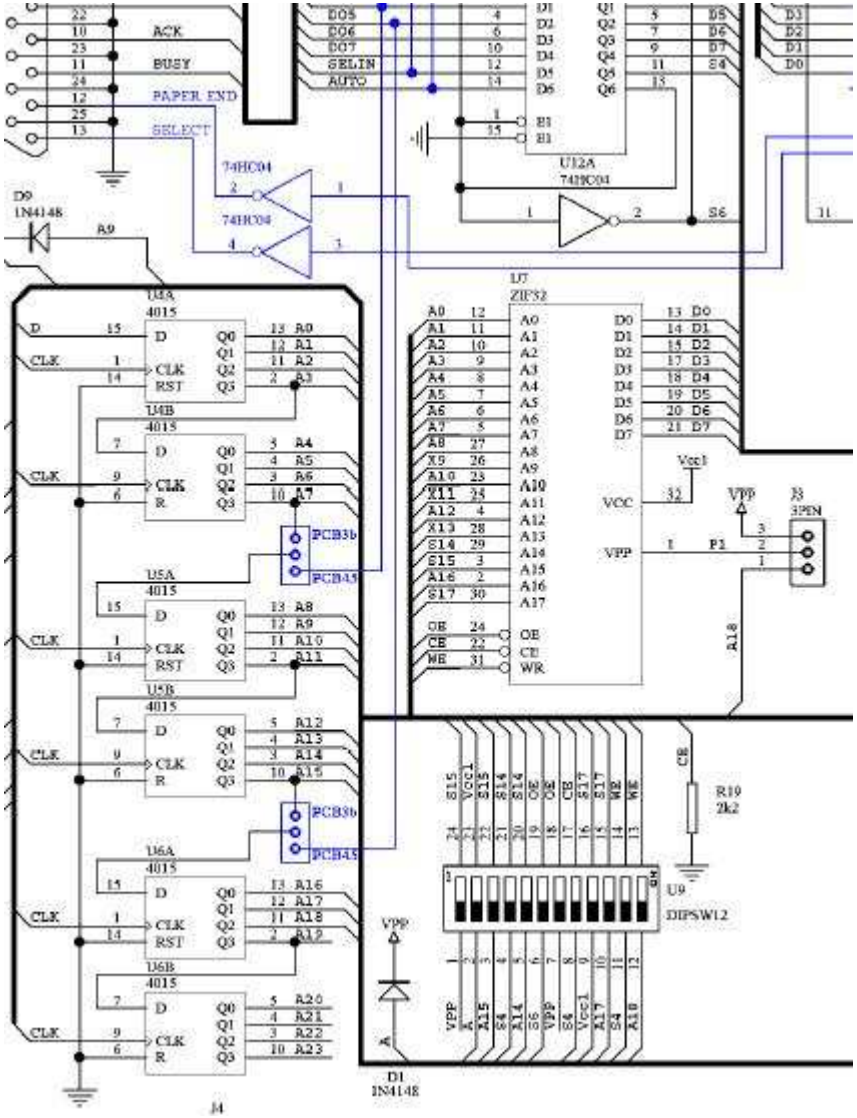
- Ao carregar o bit **A0** em **1**, corresponde a enviar *00000001* para o shift register.
- Ao carregar o bit **A0** em **0**, corresponde a enviar *00000000* para o shitf register.

Agora lembre-se que os dados são carregados serialmente, logo os primeiros 8 bits carregados, deverão ser deslocados para cima, logo a pilha no shift-register fica sendo: *0000000010000000*, o que corresponde justamente a acender o bit **A8**!

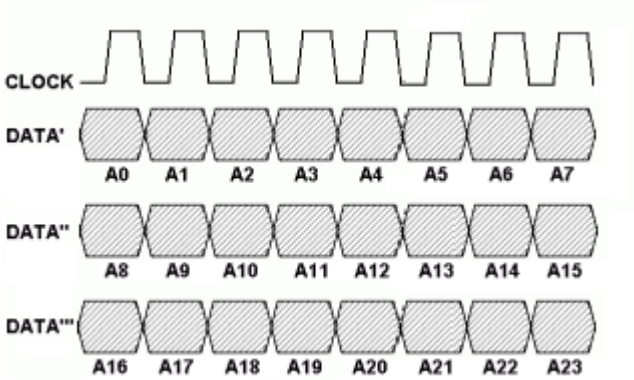
Agora, carregando novamente o bit **A0** em **1** uma nova seqüência *00000001* será enviada, logo os 16 bits que já estão no shift-register deverão ser deslocados acima, logo a seqüência no shift-register fica sendo: *0000000010000000000000001*, o que vai corresponder justamente a acender os bits **A0** e **A16**!!!

Logo conclui-se que o barramento esta partido em blocos menores. Analisando o esquema do gravador, e as seqüências acima, fica fácil deduzir que o barramento foi divido em 3 segmentos de 8 bits! Faltava apenas descobrir por onde deveriam vir os outros dois grupos de 8 bits. Pesquisando com o osciloscópio, encontrei estes dois segmentos, presentes no pinos **DO4** e **DO5** da porta paralela.

Dai bastou apenas segmentar o barramento e ligar o pino **15** de **U5** ao pino **D04** e o pino **15** de **U6** ao pino **DO5** da porta paralela, e *voilá*! Logo o esquema ficou conforme abaixo:

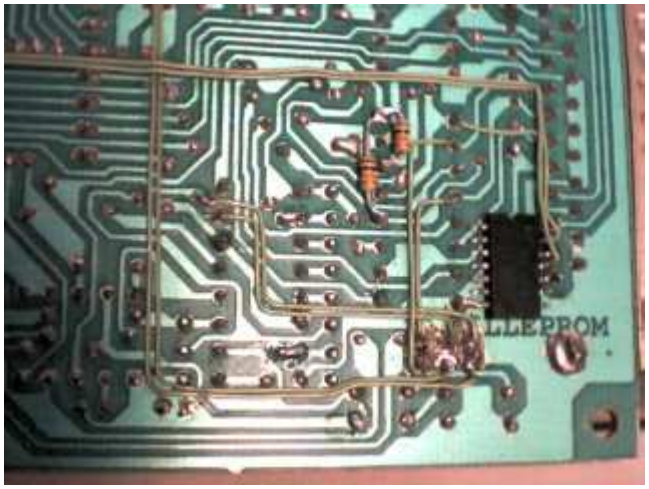


Agora se parar e analisar, vai descobrir o porque o gravador fica muito mais rápido quando usado no novo formato de barramento. Pois agora ao invés de carregar todos os bits necessários ao barramento de um em um, eles são divididos em três grupos de 8 bits, o que significa que em qualquer caso serão necessários apenas 8 pulsos de clock para carregar o barramento todo, enquanto no método antigo era necessário a mesma quantidade de pulsos que a quantidade de bits do barramento da eprom.



Simples, prático e eficiente!

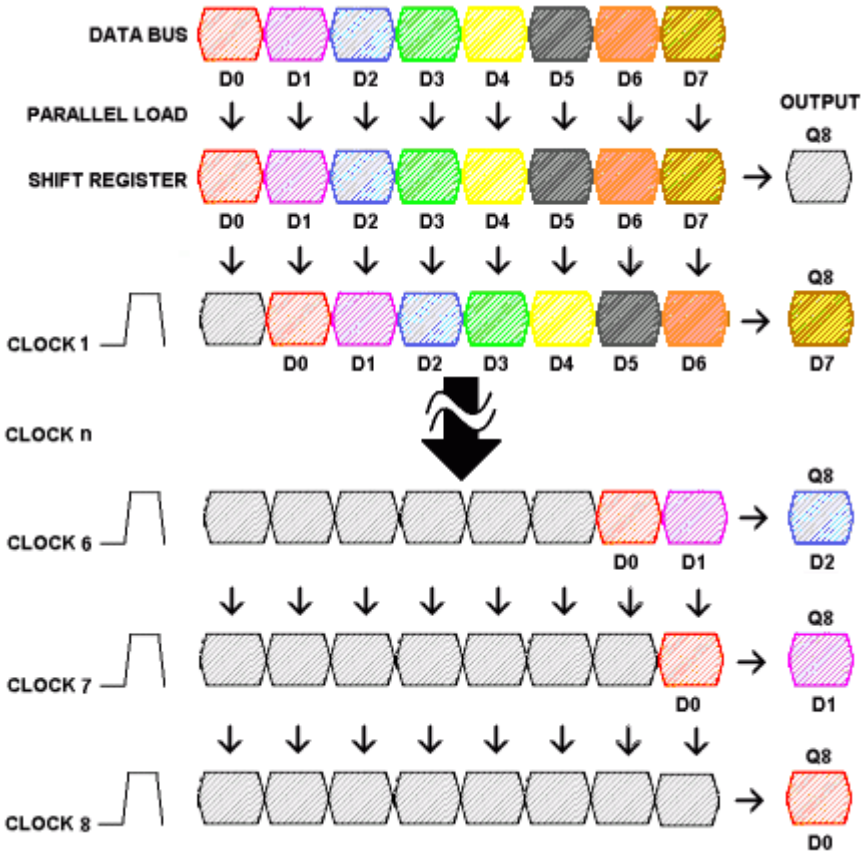
Note que optei por incluir os jumps para a seleção de modos, de forma que a placa poderá funcionar tanto no modo PCB3b quanto no modo PCB4.5. Os jumps foram instalados próximos aos leds de Vcc e Vpp, ali, por baixo da placa há uma área de cobre maciça ligada ao GND, com uma pequena broca fiz os 6 pads para a instalação dos jumps.



Porém ao fazer esta modificação, notei que durante a leitura de uma eprom, o gravador ainda não funcionava corretamente, apensar do endereçamento correto no barramento, não eram lidos dados consistentes da eprom. Ou seja, ainda tem mais alguma coisa a ser alterada.

Vamos a pesquisa novamente.

O barramento de dados é lido por um outro shift register, agora um CD4014, que lê os 8 bits em uma única tacada, e os transmite serialmente para a porta paralela. Para enviar os 8 bits para a porta paralela são necessários 8 pulsos de clock individuais, ou seja, pulsa clock lê bit, pulsa clock, lê o bit. Tal como na figura abaixo:

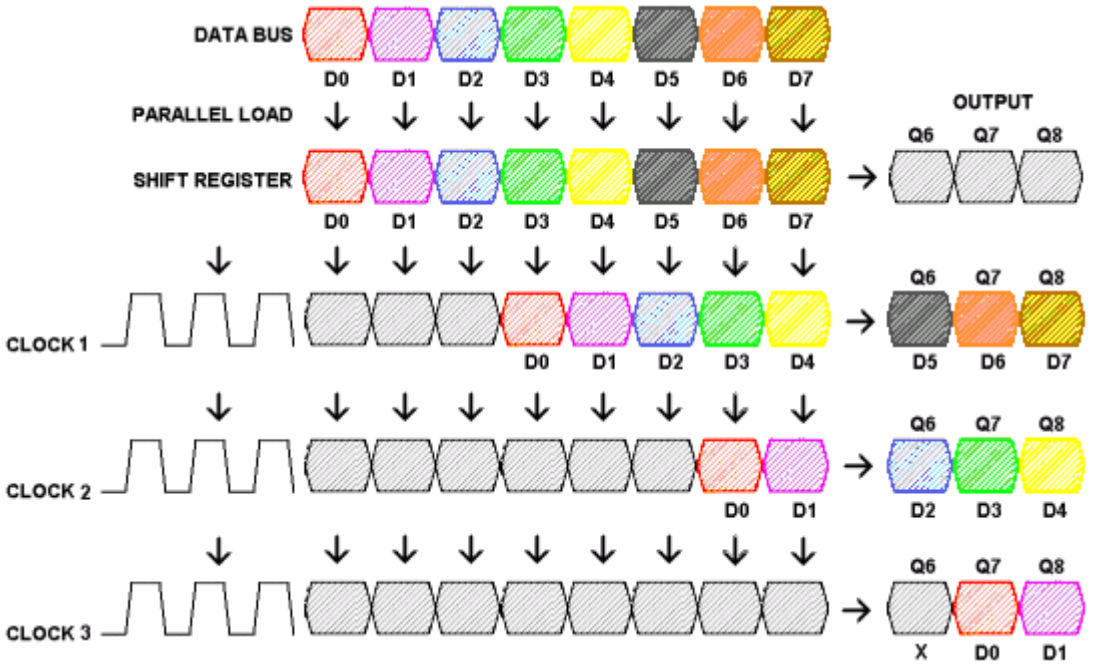


Desta forma note que o programa precisa ficar chaveando para escrita no pino clock e leitura do dado. A grosso modo são necessárias no mínimo 16 operações para a leitura de um byte.

Ao analisar o funcionamento da placa sob o software novo, notei que apenas 2 bits eram lidos do barramento, o que indicava uma nova divisão de barramento, agora no barramento de dados. Ao colocar resistores de pull-up nas linhas de dados, para montar o bit 0FFh, observei que era lido como sendo 090h.

O 4014 tem saídas em seus últimos 3 estágios do shift register, e observado o dado lido, da para perceber que os bits estão sendo lidos de 3 em 3 pelo software. Observe: 10010000

Então o que acontece? O software esta esperando os outros dois bits por algum lugar, porem como não há nada conectado, ele lê como zero. A forma como a leitura e feita a 3 bits e exemplificada abaixo:

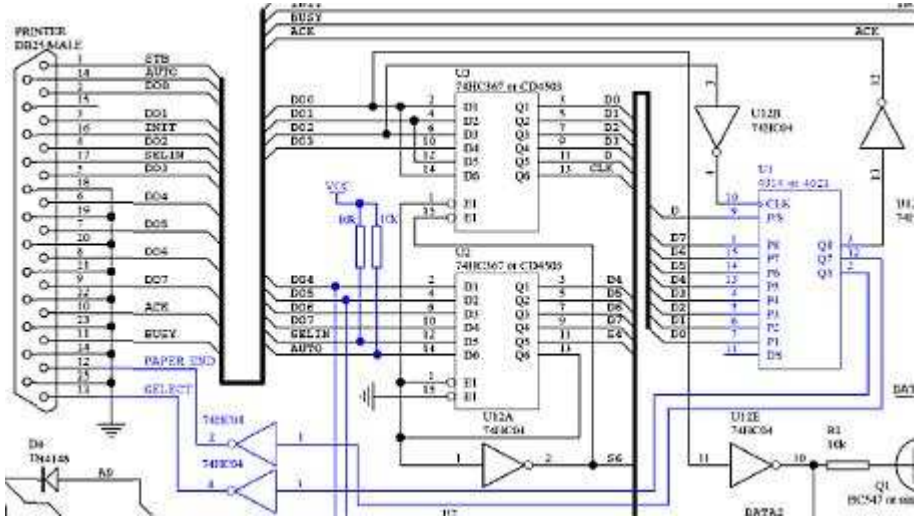


Desta forma, note que são carregados 3 bits por pulso com 3 pulsos de clock simultâneos. Fazendo uma analise como fiz anteriormente, seriam necessários no mínimo 12 operações para a leitura do byte (três pulsos de clock, uma leitura, três pulsos de clock, uma leitura e mais três pulsos de clock e uma leitura)

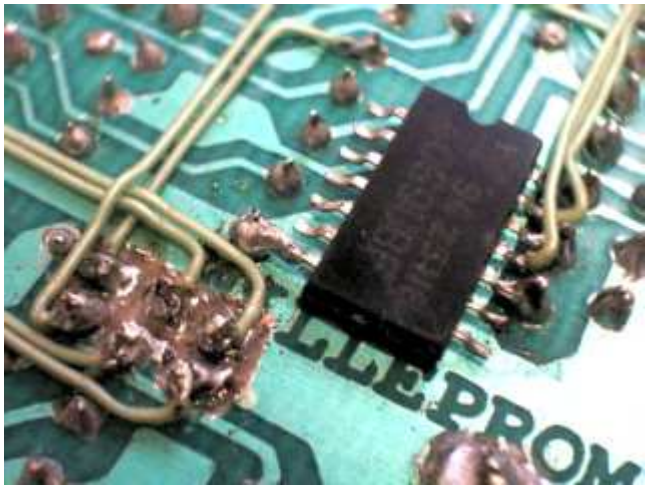
Pesquisando com um resistor conectado a linha de 5V, descobri que a leitura dos bits faltantes é feita pelos pinos **paper end** e **select** da porta paralela.

Logo bastaria conectar os pinos **Q6** e **Q7** do **4014** a estes pinos da porta paralela e pronto. Mas... há um problema, o pino **Q8**, originalmente passa por uma porta inversora (**NOT**) e observando a foto da placa no site do sivava, não há ci extra, e todas as portas do **74HC04** estão ocupados. Pensando um pouco, acredito que o circuito deve ter sido modificado de forma a liberar duas das portas do **74HC04**, creio ter sido alterado o circuito do transistor **Q5**, para liberar a porta **U12D** e também o circuito do transistor **Q3** para liberar a porta **U12C**.

Isto seria um problema pois envolveria mais modificações na placa. Então optei por acrescentar um **74HC04 SMD** por baixo da placa para utilizar as duas portas inversoras extras. No trecho abaixo do esquema, veja como ficou esta parte:



E nesta ultima imagem, como instalei o CI extra por baixo da placa. Os pinos 2 e 4 foram soldados diretamente aos pinos do DB25.



◆ Testes de desempenho

Depois de feitas as modificações, fui comprovar se realmente há uma melhoria na velocidade do programador. Todos os testes de velocidade, fiz utilizando uma Flash AM29F040B, plcc com o adaptador PLCC -> DIP.

Para o teste foram usandos dois computadores diferentes, para verificar até que ponto o clock do processador influencia no tempo de gravação.

- AMD K6-II 350MHz, com 256Mb de RAM e Windows Milleniun Edition
- Athlon XP 2400+ (2GHz), com 1Gb de RAM e Windows 2000 Professional

O arquivo para gravação na eprom foi usado um arquivo mp3, o qual preencheu toda a memória flash. Foi usada uma mp3, pelo motivo de que seu conteúdo é bastante aleatório e não existem grandes seqüências do byte 0FFh, o qual não é efetivamente gravado pelo software.

O tempo computa a gravação e verificação da flash AM29F040B. Considere uma variação de 5% nos tempos medidos como aceitável.

Micro 1: K6-II 350MHz

Modo PCB3b : 1079 segundos (17:59 minutos)
Modo PCB4.5: 392 segundos (6:32 minutos)

Micro 2: Ahtlon XP 2400+

Modo PCB3b : 420 segundos (7:00 minutos)
Modo PCB4.5: 198 segundos (3:18 minutos)

Depois, foram testados os seguintes dispositivos no gravador, para verificar a correta compatibilidade e funcionamento, todos com resultado 100%:

- D27C128
- NM27C256
- W27E512
- TMS27C010
- MX28F1000
- EN29F002NT
- F29C51002T
- AT29C010A
- N82802
- W49F002
- SST29EE010
- PIC12C508A/OTP
- PIC16F628A
- PIC16F877A
- AT24C32 SMD
- AT24C256

Conclusões

Uma coisa que não entendo é o motivo de tanto segredo por parte de Gitti e Sivava por estas modificações, pois em menos de 3 dias trabalho (contando somente o tempo que dediquei a isto) foi possível levantar todas as modificações.

O projeto do programador, de autoria do Willem sempre foi aberto e livre e esta filosofia foi o que proporcionou um bom desenvolvimento do mesmo, tanto que Gitti foi quem fez todas as ultimas atualizações do software para windows, e muitas das alterações do hardware e também muitos adaptadores.

Gitti e Sivava bem que poderiam ter tornado publicas estas informações, e tenho certeza que não prejudicaria suas vendas, e receberiam uma grande e valiosa colaboração de todos os usuários.

Apenas para citar o quanto pode ser de grande ajuda o software e esquemático livre, no tempo que fiz testes de versões beta do software, encontrei dois bugs, que foram prontamente corrigidos por Gitti, e uma sugestão de melhoria que foi adicionada ao software.

A titulo de curiosidade, os bugs foram:

- Bug da programação dos PICs 12C508/9 OTP
- Acréscimo da opção dos PICs 12C508/9 JW
- Acréscimo da opção on/off Vcc (MCLRE) no modo ICSP

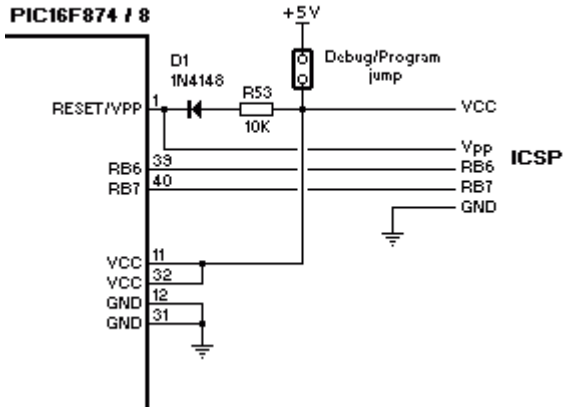
Agora um pequeno esclarecimento a esta ultima função, visto que muita gente não entende para que ela serve.

Imagine-se programando um software para um PIC16F877A, o processo normal seria ligar o cabo ICSP, programar o firmware, desligar o cabo e ligar a alimentação da placa.

Com este recurso não é necessário este processo, basta ligar o dipswitch 2 e marcar o box **MCLRE +5V** que o código carregado no PIC ira iniciar sua execução imediatamente após a programação, usando a alimentação do próprio gravador para isto.

É necessário prover apenas isolação para o pino de reset, para que a tensão de Vpp não entre para a linha de alimentação e isolação para as linhas RB6 e RB7, para que não haja conflitos com o pic e o programador, ou simplesmente não usar as linhas RB6 e R7 durante a fase de desenvolvimento.

Inicio



Softwares alternativos.

Existem vários softwares que podem ser configurados para funcionar com o Willeprom. Todos eles são somente para programação de dispositivos seriais, como PIC's e memórias I2C.

Três deles que valem a pena serem citados:

- **IC-Prog** by Bonny Gijzen
- **WinPIC800** by Sisco Benach
- **WinPIC** By DL4YHF

Um comentário digno de nota, é sobre o **WinPIC** By DL4YHF e sua fantástica idéia de como adicionar suporte a novos PIC's ao software. Todos os dispositivos podem ser descritos em um arquivo de texto (devices.ini). Desta forma fica muito, mas muito simples de adicionar novos dispositivos ao software. Basta pegar todos os dados no datasheet do PIC desejado e criar uma nova entrada no arquivo.

Mais simples e prático impossível, pois desta forma mesmo que o autor descontinue as atualizações, qualquer usuário pode continuar a adicionar novos dispositivos.

Outra grande idéia, é a descrição do arquivo de interface do programador, desta forma também é possível colocar o software para funcionar com praticamente qualquer gravador de dispositivos seriais, tanto pela porta paralela, quanto pela porta serial (sim, ele também suporta programadores RS232!).

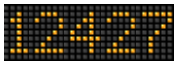
Na versão atual do software, (V2.9d - updated: April 8th, 2006) não esta incluso o arquivo de configuração para o funcionamento com o Willeprom. Porém eu já o criei, e remeti ao autor, e será incluído no próximo update.

Enquanto isso não acontece, você pode fazer o donwload aqui: **Willem_OnLptPort.ini**

Dowloads

- Esquema PCB4.5
- Eprom_0.98D5
- **Este texto em formato PDF.**

Visitante:



Atualizada em: 29/outubro/2006