

# **GUIA BÁSICO DE UTILIZAÇÃO E INSTALAÇÃO DO “LINUX OUT OF THE BOX”**

## **KIT MERCURIOIV**

**Revisão 02 - Outubro/2014**

**Versão do Firmware: 1.1.0**

# 1. Sumário

1.Introdução.....	3
2.Gravando e apagando a imagem do Linux no kit MercurioIV.....	4
3.Conectando ao terminal do kit.....	6
Terminal através da serial.....	6
.....	8
Terminal através de SSH.....	9
4.Instalando os drivers.....	11
Inserindo e removendo um módulo no kernel do Linux.....	11
5.Utilizando os drivers da MercurioIV.....	12
Driver do LCD 16x2.....	12
Exemplos de utilização:.....	13
Sensor de Temperatura MAX7500.....	14
Exemplo de utilização:.....	14
GPIO.....	14
Exemplo de utilização:.....	14
AD Converter.....	16
Exemplo de utilização:.....	17
DA Converter.....	17
Exemplo de utilização:.....	18
VGA.....	18
Exemplo de utilização:.....	19
6. Script de inicialização - RC.....	20
7.Anexo II - Configurações Padrões.....	21
IP.....	21
Credenciais para acesso via SSH.....	21
Parâmetros de comunicação serial.....	21
8.Anexo I - Tabela de GPIO.....	22

## 1. Introdução

O “*Linux Out Of The Box*” é uma pacote de software e lógica programável desenvolvidos para dar suporte Linux no kit Kit MercurioIV. O pacote vem com imagens já compiladas do Linux e device drivers para acesso aos principais recursos do kit.

Desta forma, é possível desenvolver aplicação em Linux sem exigir desenvolvimento de lógica programável e drivers de acesso aos periféricos, diminuindo o “*time-to-market*” do projeto.

Este manual visa explicar o funcionamento de cada periférico e como acessá-lo pelo “*Linux Out Of The Box*”. Embora seja fornecida a imagem completa, é possível instalar os drivers e utilizá-los em um projeto de Linux já existente\*.

\* Embora seja possível utilizar os drivers em um projeto de Linux existente, é necessário que seja utilizado a mesma lógica programável.

## 2. Gravando e apagando a imagem do Linux no kit MercurioIV

O kit MercurioIV é fornecido com uma lógica demonstrativa sem Linux. Para utilizar a imagem do “Linux Out Of The Box”, o primeiro passo é gravá-la na memória flash do kit.

Para esta etapa, é fornecido um script de gravação. Embora o automatize a tarefa, é necessário que o software Quartus esteja instalado (para maiores informações à respeito da instalação do Quartus, verifique o “Guia Rápido” que acompanha o kit).

A tabela 1, apresenta uma lista de pré requisitos para a gravação da flash com a imagem do Linux.

<b>Gravação do firmware</b>	<ul style="list-style-type: none"><li>• <b>Quartus;</b></li><li>• <b>Cabo USB;</b></li><li>• <b>Arquivo da lógica programável: “linux_design.sof” *</b></li><li>• <b>Script de gravação da lógica programável: “write_jic.sh”; *</b></li><li>• <b>Imagem do linux com aplicação embarcada: “zImage.initramfs.gz”. *</b></li></ul> <p>* Estes arquivos estão disponíveis junto com o pacote do Linux.</p>
-----------------------------	--

Tabela 1: Pré requisitos para gravação da imagem do Linux

- Utilizando o “Nios II Command Shell”, navegue até a pasta do pacote do “*Linux Out Of The Box*”. Se você estiver utilizando Linux, poderá usar o próprio terminal do sistema.
- Execute o script “write\_jic.sh”

```
~$ ./write_jic.sh
```

Aguarde o término da gravação que pode levar alguns minutos para ser concluída. Logo após a gravação o Linux já será iniciado e estará pronto para uso. Você poderá visualizar a inicialização do Linux se estiver conectado à um terminal serial, conforme descrito no capítulo 3.

O script criará duas partições na flash: uma com a imagem do Linux e outra para os arquivos persistentes. A imagem do Linux é gravada na primeira partição compactada. Em cada inicialização a imagem é descompactada e montada na memória RAM (volátil). Por isso **é importante observar que todos os arquivos que foram escritos no sistema de arquivos serão perdidos quando a placa for desligada ou reiniciada (com exceção de “/mnt/flash”)**. Para permitir a utilização de arquivos persistentes, a segunda partição da flash é montada em “/mnt/flash”. Todos os arquivos salvos nessa partição serão mantidos.

Além de gravar a imagem na memória flash, pode ser necessário apagá-la para que sejam removidos os arquivos de configuração. Para realizar essa tarefa você pode utilizar o script “format\_flash.sh”, na mesma onde está localizado o script “write\_jic.sh”. Em um terminal, digite:

```
~$ ./format_flash.sh
```

Aguarde o término e, se desejar instalar a imagem do Linux, repita os procedimentos mencionados no início deste capítulo.

## 3. Conectando ao terminal do kit

Assim que gravada a imagem do Linux pela primeira vez, você poderá se conectar ao kit MercurioIV de duas formas:

1. Através de serial;
2. Através de SSH.

Independentemente do modo de conexão que você escolha, será necessário um software para a comunicação. Este manual ilustrará a configuração através do PuTTY, mas você poderá utilizar o software de sua preferência.

O PuTTY pode ser encontrado no site: "[www.putty.org](http://www.putty.org)".

### ***Terminal através da serial***

Para utilizar o terminal do sistema através da serial, será necessário uma porta RS-232 disponível em seu computador. Pode ser utilizado um adaptador USB RS-232 se desejar.

Uma vez conectado o cabo, você precisará parametrizar a comunicação serial:

Baud rate	115200
Data bits	8
Stop bits	1
Parity	None
Flow control	None

*Tabela 2: Parâmetros da comunicação serial*

Os passos a seguir exemplificam a configuração da serial no PuTTY.

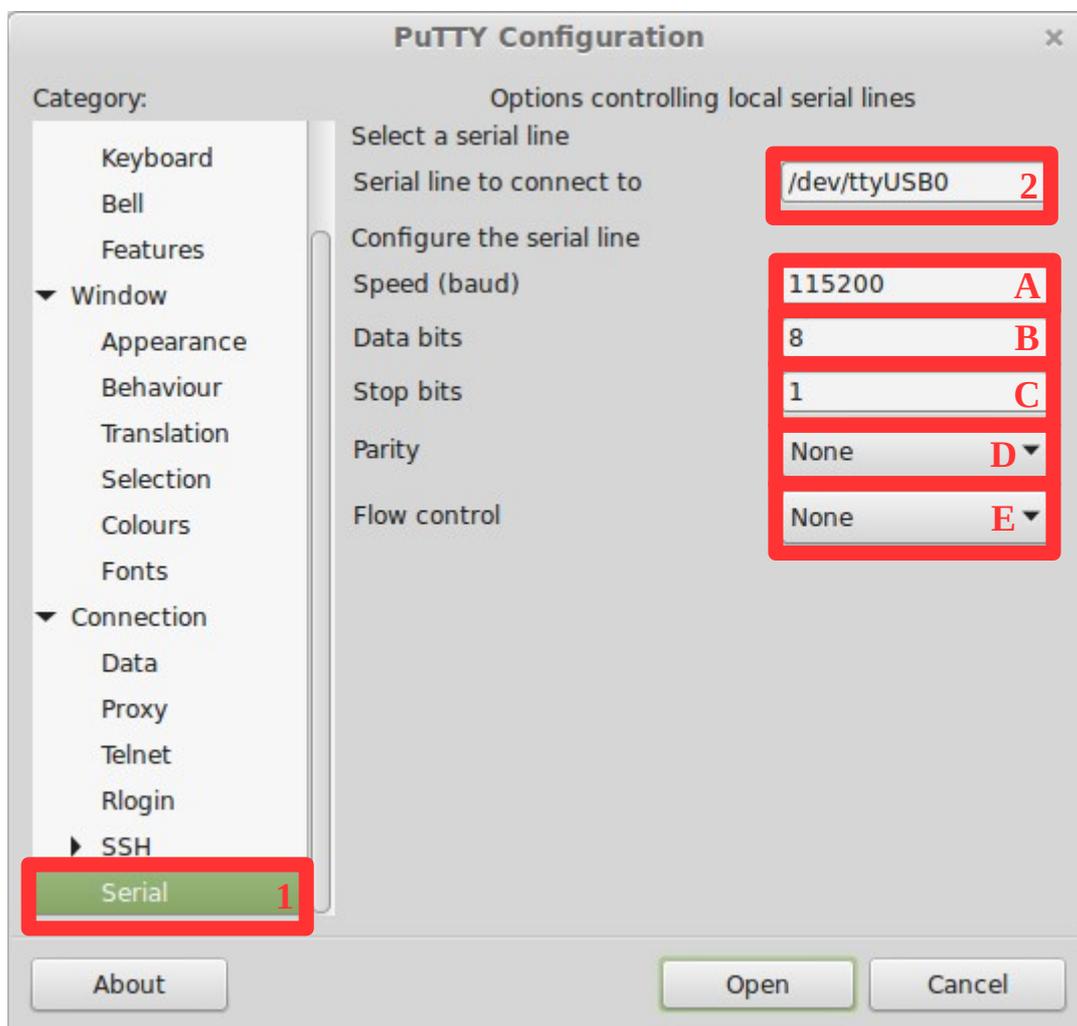


Figura 1: Configuração da serial com o PuTTY

1. Na aba “Serial”, configure a porta de comunicação serial que você irá utilizar. Esta configuração irá depender das configurações do seu computador.  
Ex. “COM1”, “COM3”, etc. (se estiver utilizando Windows);  
“/dev/ttyS0”, “/dev/ttyUSB0” (se estiver utilizando Linux).
2. Altere os parâmetros A, B, C, D e E da comunicação de acordo com a tabela 2.

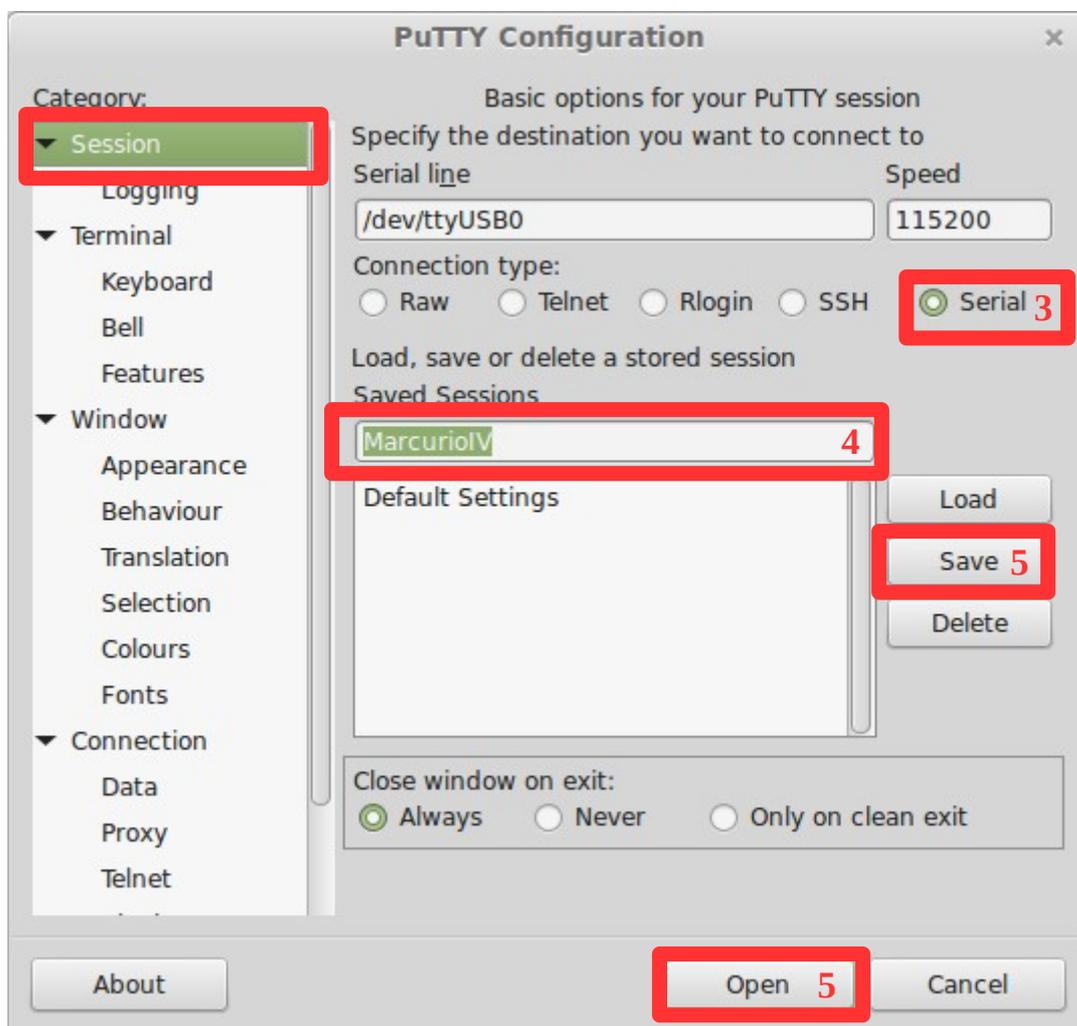


Figura 2: Configuração da serial com o PuTTY

3. Volte à aba “Session”, e selecione o tipo de comunicação: Serial;
4. Dê um nome à sua configuração.
5. Clique em “Save” para manter a configuração.
6. Clique em “Open” para iniciar a comunicação.

## ***Terminal através de SSH***

Para acessar a placa através da serial é necessário estar na mesma rede da placa com um IP de seu computador configurado na mesma sub-rede.

O kit MercurioIV vem configurado de fábrica com o **IP 10.0.0.10**. Você pode configurar a sua placa de rede com um IP na mesma faixa (por exemplo, 10.0.0.5) e posteriormente, alterar o IP do kit Mercurio, se desejar.

Usuário	root
Senha	root

*Tabela 3: Login e senha de acesso ao SSH*

Se você estiver utilizando o PuTTY, você pode seguir os passos a baixo:

1. Configure o “Host Name” com o IP do kit Mercurio (10.0.0.10 se for o padrão de fábrica);
2. Escolha o tipo de comunicação: ssh;
3. Dê um nome à configuração;
4. Clique em “Save” para manter a configuração;
5. Clique em “Open” para iniciar a comunicação.

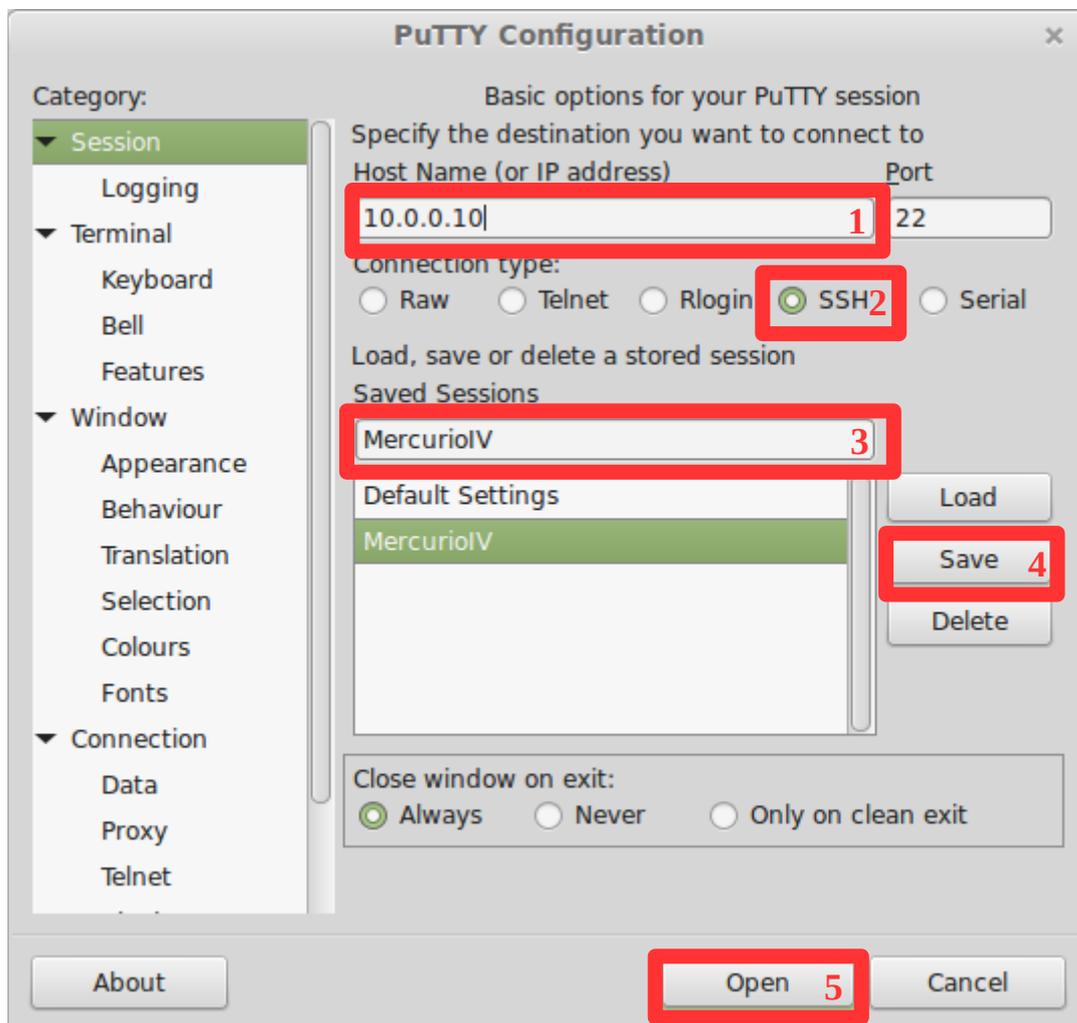


Figura 3: Configuração do SSH com o PuTTY

## 4. Instalando os drivers

Os device drivers da MercurioIV são disponibilizados na forma de módulos do *kernel* que podem ser inseridos ou removidos do sistema dependendo da necessidade.

Se você estiver utilizando a imagem de Linux já compilada, os drivers serão carregados automaticamente e você poderá pular esta etapa.

### ***Inserindo e removendo um módulo no kernel do Linux***

Para instalar um device driver no kit MercurioIV é necessário copiar o módulo para seu projeto. Esta copia pode ser feita através do cartão de memória ou utilizando o comando “*scp*” do Linux (para transferir arquivos desta forma, será necessário conexão de rede).

Através de um terminal (serial ou ssh), vá até a pasta onde está o módulo a ser inserido e digite:

```
root: /> /root/insmod.sh ./NOME_DO_DRIVER.ko
```

Você poderá verificar o aparecimento do “arquivo” do device driver em “/dev/NOME\_DO\_DRIVER” e, dependendo do driver que estiver utilizando, em “/sys/class/”.

Após inserido um módulo, você pode removê-lo a qualquer momento através do comando:

```
root: /> /root/rmmod.sh NOME_DO_DRIVER
```

## 5. Utilizando os drivers da MercurioIV

Os drivers do kit Mercurio são implementados através de “arquivos” do dispositivos. Estes “arquivos” (nós) podem ser acessados pelo sistema de arquivos do Linux e ter seus parâmetros alterados pelo terminal do sistema.

Outra possibilidade é utilizar os devices drivers através de aplicativos desenvolvidos para o Linux, que deverão acessar os “nós” dos dispositivos através de funções como “fopen()”.

Os drivers da MercurioIV são implementados através dos nós dos dispositivos em “/dev/DISPOSITIVO” e suas propriedades implementadas em “/sys/class/DISPOSITIVO”.

Nas seções seguintes serão apresentados os drivers de cada dispositivo do kit MercurioIV, explicando como o dispositivo deve ser acessado e parametrizado.

### ***Driver do LCD 16x2***

O driver do LCD implementa um dispositivo em “/dev/” chamado “lcdm4”. A escrita no LCD deve ser feita escrevendo neste caminho do dispositivo.

Observações:

- Como trata-se de um LCD de 16 colunas e 2 linhas, ao escrever uma string com mais do que 16 caracteres no caminho do dispositivo, a string será automaticamente dividida em duas linhas (os caracteres a partir da 17ª posição ficarão na segunda linha);
- Uma quebra de linha na string (“\n”), gera uma “quebra de linha” no LCD. Quando uma quebra existir na segunda linha do LCD, será terminada a escrita no LCD;
- Novas escritas apagam o LCD.

## Exemplos de utilização:

```
root:/> echo "KIT MERCURIO IV DEVELOPMENT BOARD" > /dev/lcdm4
```



```
root:/> echo "MERCURIO IV  
> DEV. BOARD" > /tmp/lcd  
root:/> cp /tmp/lcd /dev/lcdm4
```



Além do dispositivo implementado em “/dev/”, o driver implementa também a propriedade do componente “backlight”, em “/sys/class/lcdm4/”.

As propriedades dos drivers podem ser acessadas e parametrizadas escrevendo 1 ou 0 em seu caminho:

```
root:/> echo 0 > /sys/class/lcdm4/backlight
```

## **Sensor de Temperatura MAX7500**

De forma semelhante ao driver do LCD, o driver do sensor de temperatura implementa o dispositivo em “/dev/” e em “/sys/class”.

Em “/dev/tsensor” é possível obter a leitura do sensor. O valor lido estará em “inteiro”, multiplicado por 10. Para obter o valor correto é necessário dividir o resultado obtido.

Já em “/sys/class/temperature” é possível obter o valor lido pelo sensor em string. O valor pode ser lido pelo terminal do Linux, conforme ilustrado abaixo.

### **Exemplo de utilização:**

```
root: /> cat /sys/class/tsensor/temperature
30.0
```

## **GPIO**

Grande parte dos periféricos do kit são disponibilizados através do driver de GPIO. A documentação completa do driver pode ser encontrada no site:

<https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/Documentation/gpio>

Cada pino do GPIO, LEDs, chaves, botões, proto A e proto B é representado por um GPIO diferente no Linux. Para saber qual GPIO está conectada em cada periférico é necessário consultar a tabela de GPIO, disponível em anexo a este documento.

### **Exemplo de utilização:**

Para configurar o GPIO do LED **vermelho** do LED RGB como saída:

```
root: /> echo 126 > /sys/class/gpio/export
root: /> echo out > /sys/class/gpio/gpio126/direction
```

Para **ligar** o LED **vermelho** do LED RGB:

```
root: /> echo 1 > /sys/class/gpio/gpio126/value
```

Para **desligar** o LED **vermelho** do LED RGB:

```
root: /> echo 0 > /sys/class/gpio/gpio126/value
```

Para configurar a chave 0 (**SW0**) como entrada:

```
root: /> echo 221 > /sys/class/gpio/export
root: /> echo in > /sys/class/gpio/gpio128/direction
```

Para **ler** o valor da chave 0 (**SW0**):

```
root: /> cat /sys/class/gpio/gpio221/value
```

A fim de automatizar a configuração e atribuição de valores, pode-se utilizar os scripts “gpio\_out.sh” e “gpio\_in.sh”, para pinos de saídas e entradas respectivamente. Os scripts estão disponíveis em “/root/”, e devem ser utilizados da seguinte forma:

Para **ligar** o LED **vermelho** do LED RGB (não é necessário configurar antes):

```
root: /> /root/gpio_out.sh 126 1
```

Para **desligar** o LED **vermelho** do LED RBG:

```
root: /> /root/gpio_out.sh 126 0
```

Para **ler** o valor da chave 0 (**SW0**):

```
root: /> cat /root/gpio_in.sh 221
```

Note que são utilizados scripts diferentes para entradas e para saídas.

## **AD Converter**

O conversor analógico-digital funciona através de uma FIFO implementada em lógica programável. As aquisições são disponibilizadas na lógica e consumidas pelo driver do AD.

Semelhante aos demais drivers da MercurioIV, em “/dev/adcm4” pode ser feito a leitura da amostra mais antiga inserida na FIFO (16 bits por amostra), ou também pode ser feita uma única leitura do valor atual através da classe do dispositivo em “/sys/class/adcm4/value”. Também é possível configurar a frequência de aquisição, ou fazer aquisições individuais sem definir a frequência (“/sys/class/adcm4/frequency\_hz”). Quando for lido um valor através de “/sys/class/adcm4/value”, a FIFO será esvaziada descartando todos os dados, e o dado que será lido será o próximo dado colocado na FIFO pela lógica programável.

Além da frequência e valor da última amostra, a classe do dispositivo oferece informações adicionais sobre o estado da FIFO (nível de preenchimento e esvaziar a FIFO).

Obs. Nesta versão do “Linux Out Of The Box”, só está disponível o driver do AD 1A.

## Exemplo de utilização:

Para **ler** o valor do AD 1A:

```
root:/> cat /sys/class/adcm4/value
```

Para **alterar para 100 Hz** o valor da frequência de aquisição do AD 1A:

```
root:/> echo 100 > /sys/class/adcm4/frequency_hz
```

## **DA Converter**

De forma análoga ao AD, o conversor digital-analógico funciona com uma FIFO, porém nesta implementação a escrita é feita pelo driver e consumida pela lógica programável. A lógica programável consome os dados da FIFO em uma frequência parametrizável. Quando a FIFO estiver vazia, a lógica programável continuará atualizando o DA na taxa configurada, porém o dado será repetido até chegue novos dados na FIFO.

Assim como no ADC, pode ser parametrizado o valor da frequência através da classe do dispositivo em “/sys/class/dacm4”, além ser possível monitorar o nível de preenchimento da FIFO.

O driver é implementado através do “/dev/dacm4” (cada amostra deve ter 16 bits), e a tem a classe do dispositivo em “/sys/class/dacm4”.

Obs. Nesta versão do “Linux Out Of The Box”, só está disponível o driver do AD 1A.

## Exemplo de utilização:

Para **alterar** o valor do DA para **500** (valor em inteiro) A:

```
root: /> echo 500 > /sys/class/dacm4/value
```

Para **alterar para 100 Hz** o valor da frequência do DA A:

```
root: /> echo 100 > /sys/class/adcm4/frequency_hz
```

## VGA

O VGA implementado através de duas formas: um buffer para imagens e um buffer para caracteres. Cada um deles utiliza de uma memória compartilhada entre a lógica programável e o Linux, que são mapeadas através de seus drivers e disponibilizadas em “/dev/”.

Obs. Os caracteres sempre serão sobrepostos à imagem.

O VGA tem a resolução limitada em 160x120 (monocromático), e disponibiliza a memória compartilhada em “/dev/vgam4”. Cada pixel deve conter um byte (escala de cinza) e ser escrito no nó do dispositivo.

Se o número de bytes escritos no dispositivo for menor do que o número de pixels do VGA inteiro, os pixels seguintes serão mantidos.

Para escrever um caractere, deve ser utilizado o driver disponível em “/dev/charvgam4”. De forma análoga ao VGA, se o número de caracteres escrito no buffer for menor (80x60 caracteres), os caracteres seguintes serão mantidos, enquanto escritas maiores farão com que os caracteres iniciais sejam sobrepostos.

## Exemplo de utilização:

Para colocar a imagem da Mercurio IV no fundo da tela:

```
root: /> cp /root/mercurio.bin /dev/vgam4
```

Para deixar o primeiro pixel da tela branco:

```
root: /> echo -e '\0377' > /dev/vgam4
```

Para deixar o primeiro e o segundo pixel da tela branco:

```
root: /> echo -e '\377\377' > /dev/vgam4
```

Para deixar o primeiro da tela preto:

```
root: /> echo -e '\0' > /dev/vgam4
```

Para escrever na primeira linha e segunda do VGA:

```
root: /> echo "Mercurio IV  
> Board" > /dev/charvgam4
```

Ou:

```
root: /> echo -e 'Mercurio IV \nBoard' > /dev/charvgam4
```

## 6. Script de inicialização - RC

O Linux conta com um script utilizado para iniciar os módulos, serviços e aplicações que precisam ser executadas junto com a inicialização do sistema.

Na inicialização do sistema, será verificado a existência do script rc na partição de arquivos da flash: “/mnt/flash/rc”. Este arquivo pode ser configurado pelo usuário habilitando / desabilitando as aplicações que desejar. Se o arquivo rc não for encontrado em “/mnt/flash”, o sistema o criará com as configurações padrões.

## 7. Anexo II - Configurações Padrões

### *IP*

10.0.0.10
-----------

### *Credenciais para acesso via SSH*

Usuário	root
Senha	root

### *Parâmetros de comunicação serial*

Baud rate	115200
Data bits	8
Stop bits	1
Parity	None
Flow control	None

## 8. Anexo I - Tabela de GPIO

Componente	Pino / Número	Número do GPIO
LED RGB	REG	127
	GREEN	128
	BLUE	129
PROTO A	A0	143
	A7	144
	A6	145
	A5	134
	A4	133
	A3	132
	A2	131
	A1	130
PROTO B	B0	135
	B1	136
	B2	137
	B3	138
	B4	139
	B5	140
	B6	141
	B7	142
GPIO 0	D00	162
	D01	166
	D02	167
	D03	168
	D04	169

GPIO 0	D05	170
	D06	171
	D07	172
	D08	173
	D09	174
	D10	175
	D11	176
	D12	177
	D13	163
	D14	164
	D15	166
	D16	161
	D17	160
	D18	159
	D19	158
	D20	157
	D21	156
	D22	155
	D23	154
	D24	153
	D25	152
	D26	151
	D27	150
	D28	149
	D29	148
	D30	147
	D31	146

GPIO 1	D00	194
	D01	198
	D02	199
	D03	200
	D04	201
	D05	202
	D06	203
	D07	204
	D08	205
	D09	206
	D10	207
	D11	208
	D12	209
	D13	195
	D14	196
	D15	197
	D16	193
	D17	192
	D18	191
	D19	190
	D20	189
	D21	188
	D22	187
	D23	186
	D24	185
	D25	184
	D26	183

GPIO 1	D27	182
	D28	181
	D29	180
	D30	179
	D31	178
KEYPAD	KEY0	210
	KEY1	211
	KEY2	212
	KEY3	213
	KEY4	214
	KEY5	215
	KEY6	216
	KEY7	217
	KEY8	218
	KEY9	219
	KEY10	220
	KEY11	221
SWITCHES	SW0	222
	SW1	223
	SW2	224
	SW3	225
LED MATRIX	C0	226
	C1	227
	C2	228
	C3	229
	C4	230
	R0	231

LED MATRIX	R1	232
	R2	233
	R3	234
	R4	235
	R5	236
	R6	237
	R7	238
SEVEN SEGMENT	A	242
	B	243
	C	244
	D	245
	E	246
	F	247
	G	248
	DP	249
SEVEN SEGMENT	A	250
	B	251
	C	252
	D	253
	E	254
	F	239
	G	240
	DP	241