

Tutorial Raspbian e Arduino

Programação de Periféricos – CC – FACIN – PUCRS

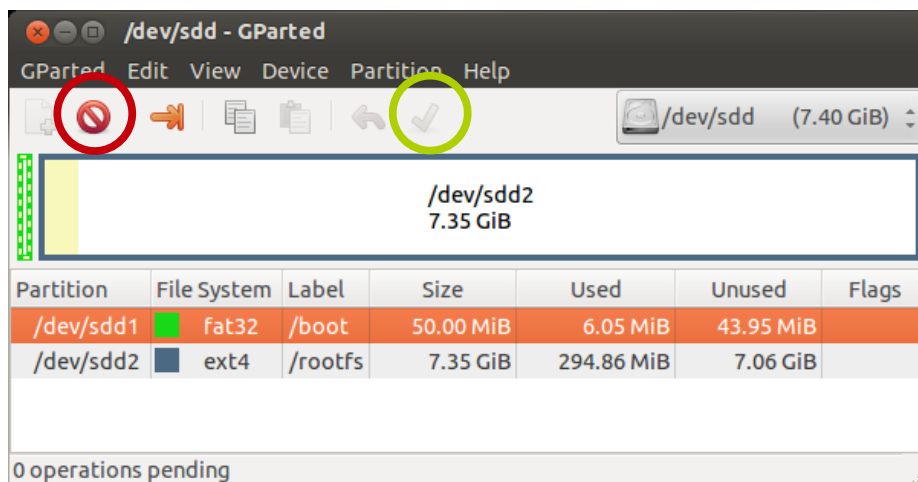
Configuração do Ambiente de Trabalho:

1. No computador host, criar um diretório de trabalho:
Ex: `mkdir ~/raspbian`
2. Ir para o diretório de trabalho:
Ex: `cd ~/raspbian`
3. Baixar o raspbian:
`wget https://corfu.pucrs.br/public/prog_perif/raspbian/2013-09-25-wheezy-raspbian.zip`
4. Extrair:
`unzip 2013-09-25-wheezy-raspbian.zip`

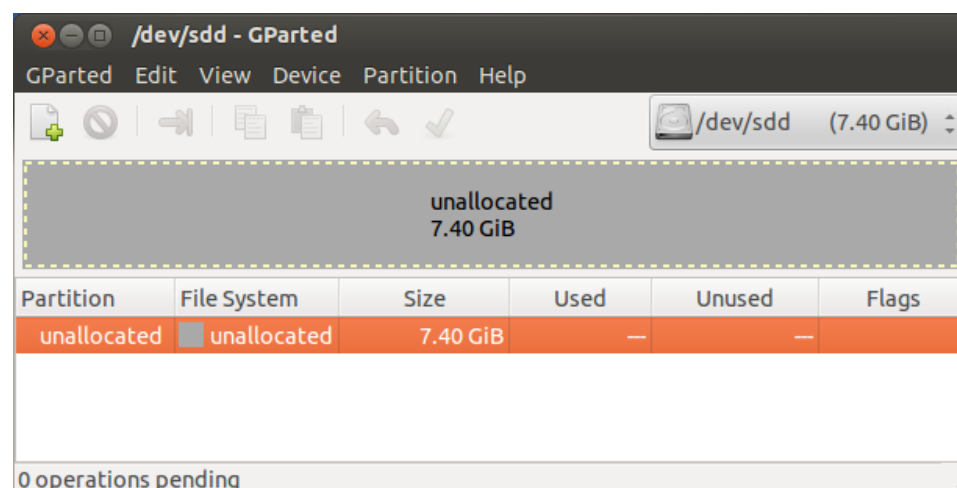
Criando um cartão com o Raspbian:

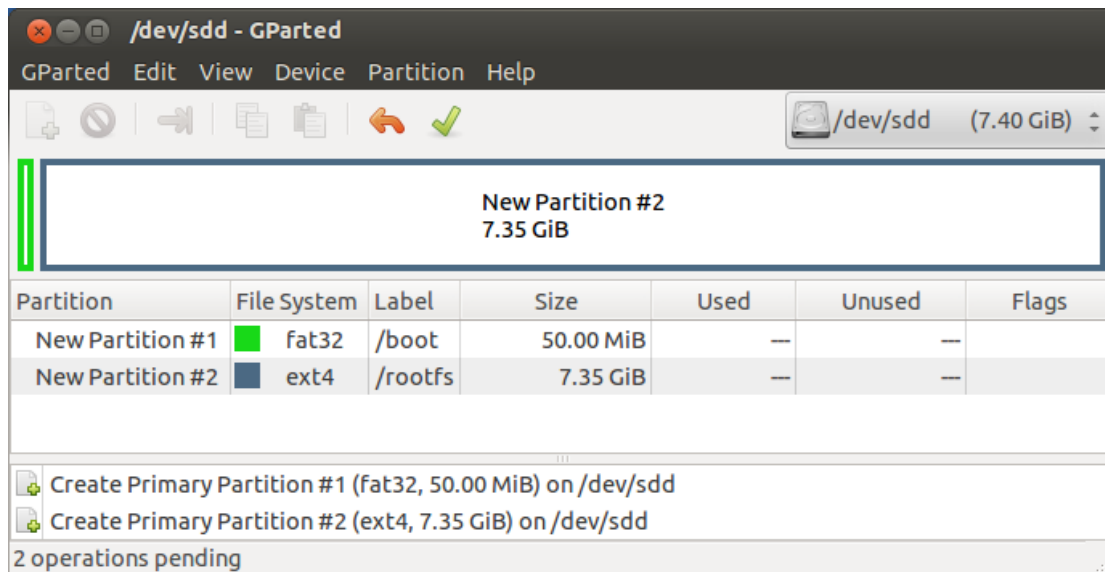
Utilizar a ferramenta `gparted`.

Selecionar o cartão SD, que deve ter 7,4GB, e deletar qualquer partição que exista no cartão, isso pode ser feito com o botão destacado na figura abaixo e aplicar as modificações:

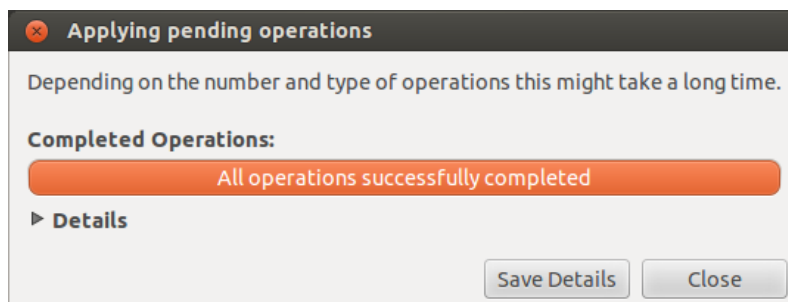


O resultado deve ser similar a figura abaixo:





Aplicar as modificações e fechar a janela que segue:



Copiar a imagem do Raspbian para o cartão:

```
sudo dd bs=4M if=2013-09-25-wheezy-raspbian.img of=/dev/DEV_CARTAO
```

DEV_CARTAO deve ser substituído pelo ponto base onde está mapeado o cartão SD. No caso do exemplo acima: sdd.

Essa etapa deve levar alguns minutos. Uma vez concluído o processo, garantir que todos os arquivos foram devidamente copiados:

```
sync
```

Desmontar qualquer partição do cartão que tenha sido montada, e inseri-lo na Raspberry Pi. Ligar a placa e buscar o IP conforme realizado nas etapas anteriores. Utilizar o comando nmap e filtrar pelo MAC.

Para contactar-se, usar o serviço de ssh. Usar o usuário: “pi” senha: “raspberry”.

Modificar a senha do usuario pi:

```
sudo passwd pi
```

Modificar a senha do usuario root:

sudo passwd root

Raspbian + Arduino:

Para utilizar o microcontrolador ATmega da placa GertBoard, precisaremos instalar no sistema uma ferramenta que permita programar esse microcontrolador. A ferramenta utilizada será o avrdude. Os passos a seguir são baseados no ambiente disponível em: <https://projects.drogon.net/>

Instalar a IDE do arduino:

sudo apt-get install arduino

Baixar o avrdude modificado para usar a Gertboard:

wget https://corfu.pucrs.br/public/prog_perif/raspbian/avrdude_5.10-4_armhf.deb

Instalar:

sudo dpkg -i avrdude_5.10-4_armhf.deb

sudo chmod 4755 /usr/bin/avrdude

Baixar um script de configurações necessário para integrar a Gertboard e o arduino:

wget https://corfu.pucrs.br/public/prog_perif/raspbian/setup.sh

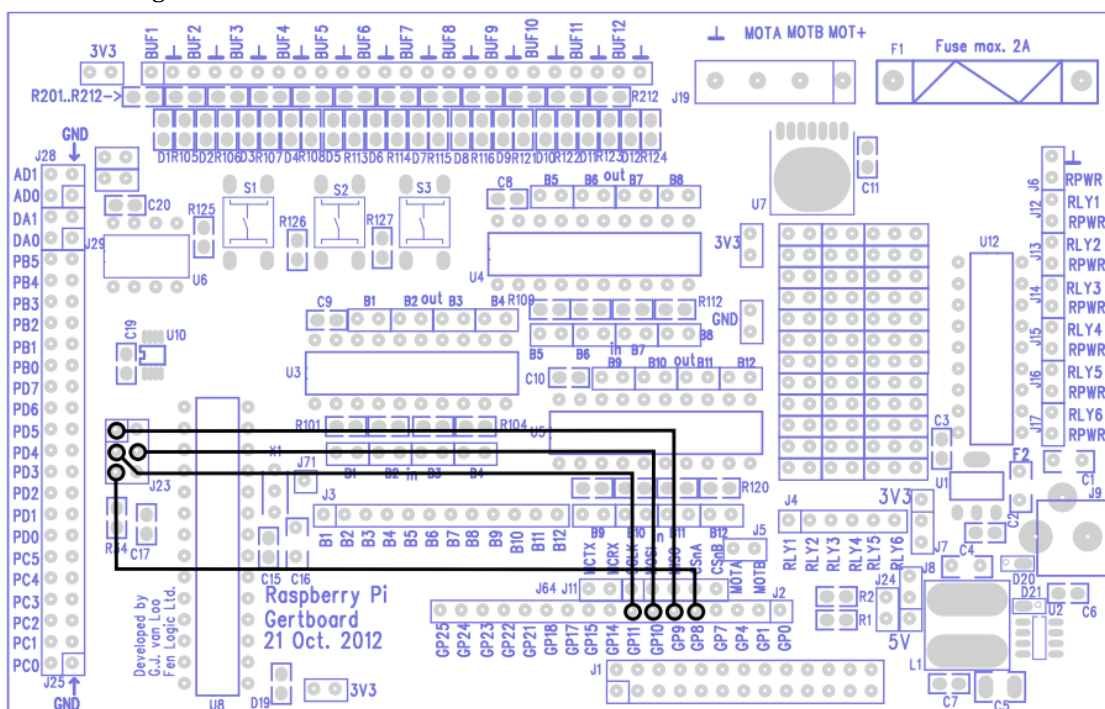
chmod +x setup.sh

sudo ./setup.sh

Reiniciar a Raspberry Pi.

sudo reboot

Conectar os pinos da Gertboard com os pinos de acesso para programar o microcontrolador conforme a figura abaixo:



Conectar-se a Raspberry Pi novamente e executar o comando de configuração abaixo:
avrsetup

Selecionar a opção 1, pois estamos utilizando um ATmega 328p.

Se tudo deu certo, uma mensagem deve ser recebida:

Initialising a new ATmega microcontroller for use with the Gertboard.

```
Make sure there is a new ATmega chip plugged in, and press  
.. 1 for an ATmega328p or 2 for an ATmega168: 1  
Initialising an ATmega328p ...  
Looks all OK - Happy ATmega programming!
```

Programando o ATmega:

A abordagem para utilização do ATmega será feita da seguinte forma:

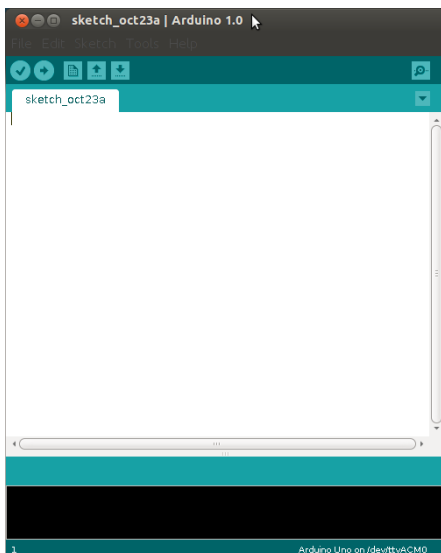
- Programas serão implementados no computador host, utilizando a IDE do Arduino.
- A IDE permite “crosscompilar” os programas para o ATmega328p.
- Os códigos crosscompilados deverão ser enviados para a Raspberry Pi.
- A Raspberry Pi utiliza o avrdude para programar o ATmega.

Caso a IDE do arduino não esteja instalada no computador host:

sudo apt-get install arduino

Utilizar a IDE

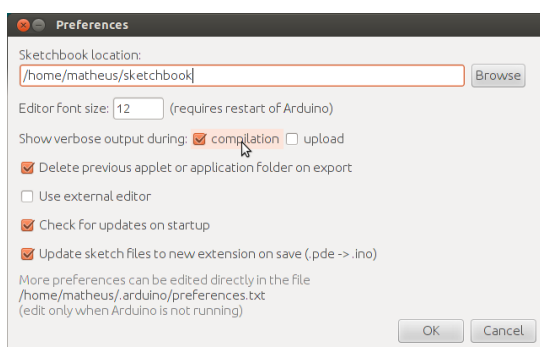
arduino



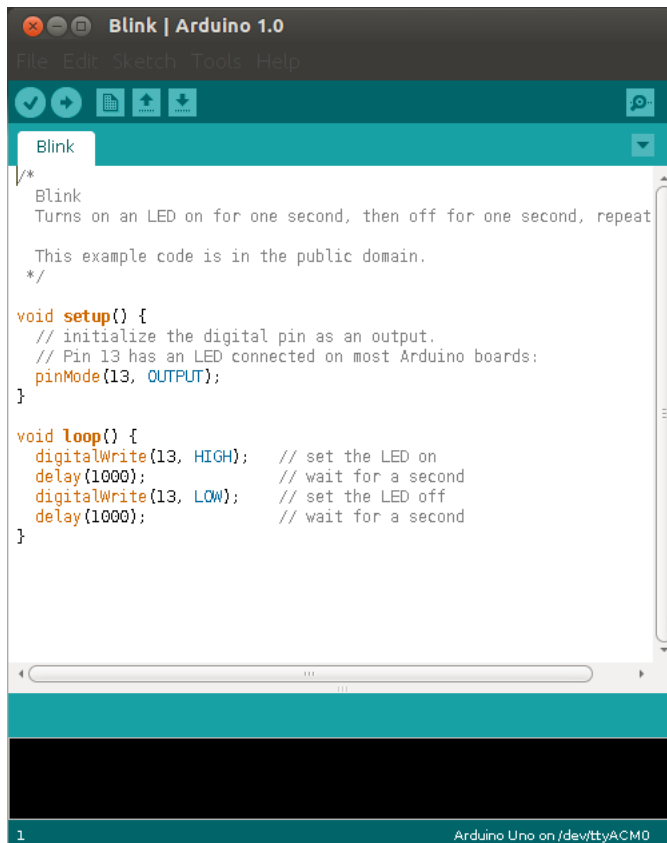
Configurar para que a IDE imprima todos passos durante a crosscompilação de um programa:

File → Preferences

Selecionar a caixa: Show verbose output during: compilation



Compilar o exemplo disponível em:
File → Examples → 1.Basic → Blink



Para compilar:
Sketch → Verify/Compile (Ou ctrl+R)

A ultima linha de execução impressa no terminal indica o local em que o código crosscompilado está. Por exemplo:

```
avr-objcopy -O ihex -R .eeprom /tmp/build1619961600674932294.tmp/Blink.cpp.elf
/tmp/build1619961600674932294.tmp/Blink.cpp.hex
```

Esse código sempre tem a extensão .hex. Envia-lo para a Raspberry Pi:
`scp /tmp/build1619961600674932294.tmp/Blink.cpp.hex pi@_IP_DA_RPI_:/`

Agora temos o código disponível na Raspberry Pi. Conectar-se usando ssh.

Para enviar o código para o Atmega, usar o avrdude (Verificar se os fios de programação estão corretamente conectados):

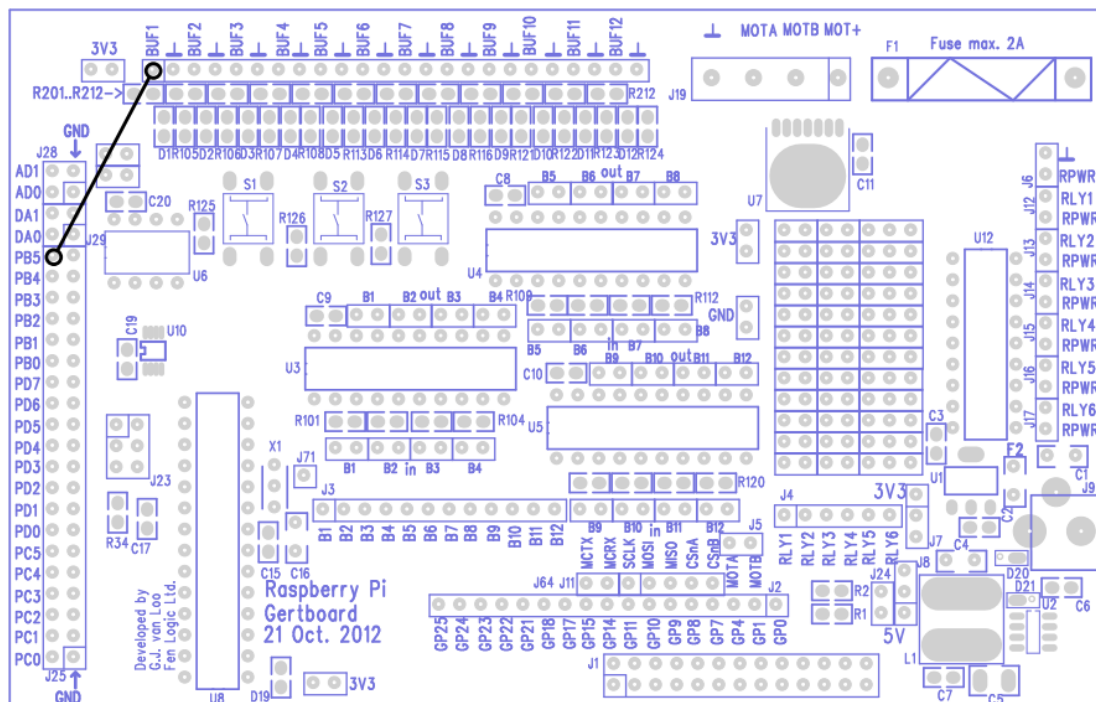
```
avrdude -v -p atmega328p -c gpio -U Blink.cpp.hex
```

Se tudo deu certo, a seguinte mensagem será recebida:
avrdude done. Thank you.

O código de exemplo executa a seguinte operação: alterna entre lógico 1 e 0 no pino 13 do ATmega a cada 1 segundo. Para visualizar a operação, conectar o pino do ATmega na Gertboard. Seguir o seguinte esquema:

Arduino Pin	GB pin	Arduino Pin	GB pin	Arduino Pin	GB pin
0	PD0	7	PD7	A0	PC0
1	PD1	8	PB0	A1	PC1
2	PD2	9	PB1	A2	PC2
3	PD3	10	PB2	A3	PC3
4	PD4	11	PB3	A4	PC4
5	PD5	12	PB4	A5	PC5
6	PD6	13	PB5		

Para conectarmos o pino 13 do ATmega no led1 da GertBoard, por exemplo:



Pronto! Agora temos um ambiente que utiliza o ATmega da GertBoard, programando-o pela RaspberryPi.

Exercícios:

Implementar todos os testes com o ATmega disponíveis nas páginas 44-49 do manual da GertBoard.