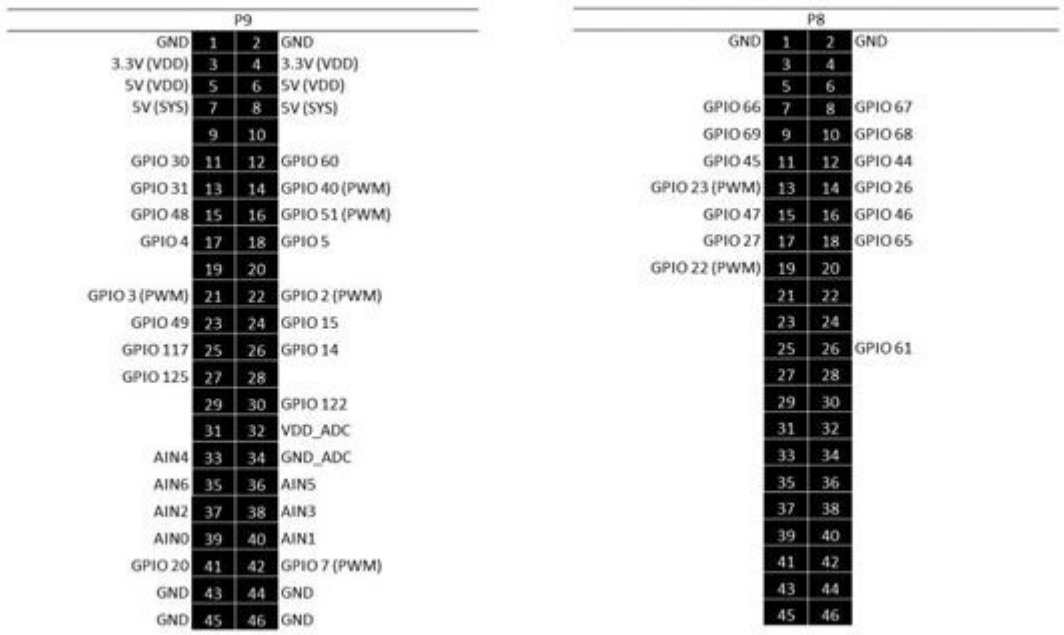
# Laboratório

## Prática 2 – GPIOs OUTPUT

Muitas vezes, a melhor maneira de se familiarizar com uma nova plataforma de trabalho, tal como a BeagleBoneBlack, é iniciar utilizando as funcionalidades de LED, pois é mais fácil para começar a trabalhar com a plataforma e também todo projeto sempre tem LEDS, realizando ação de liga e desliga. Para fazer isso, você deve definir um pino como GPIO (entrada/saída de uso geral) e saída (output), então realiza o controle do componente a partir do seu estado. Nesse projeto iremos utiliza o modulo 1 do GPIO e o pino 28.

### 1 Ligação do circuito para um LED

Primeiro, define as localizações dos pinos utilizados nas etapas que seguirão, referente a figura abaixo:




**Etapas:**

1. Desligue a BeagleBone.
   Antes de ligar qualquer coisa na BeagleBone, uma boa ideia é desligar e remover a fonte de energia.

2. Conecte a tensão na BBB.
   Usando um fio, ligue a tensão de 3.3V da BeagleBone (pinos de 3 ou 4 no expansor P9) para a trilha positiva da protoboard.

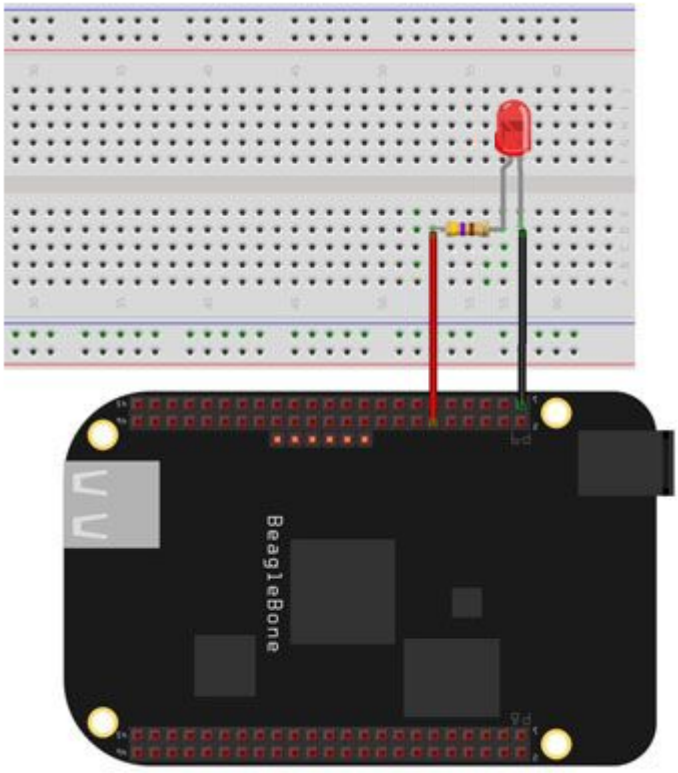
3. Conecte ao terra.
   Conectar o pino GND do BeagleBone (pinos 1 e 2 em ambos os expansores) a trilha negativa da protoboard.

4. Ligue o pino GPIO para a protoboard.
   Este exemplo usa GPIO1_28 (pino 12 no expansor P9). Use um jumper para conectá-lo a uma linha vertical na sua protoboard.

5. Conecte o resistor.
   Sem um resistor, um LED queima facilmente. Um resistor de 220 ou 470 deve cair a voltagem suficiente sem reduzir o brilho do LED demais. Ligue a resistência ao jumper que você puxou do pino 12, que liga eficazmente o resistor para GPIO1_28.

6. Ligue o LED.
   Ligue a perna negativa do LED - o cátodo, que normalmente é a perna mais curta - a faixa negativa da placa de ensaio onde estão ligados terra no Passo 3. Conecte a perna positiva - o ânodo - ao resistor.

No circuito que você acabou de construir, a tensão vem do pino GPIO1_28, ao invez de uma bateria, que pode ligar e desligar escrevendo comandos no prompt.




↗️ Na Etapa 2, você conecta o pino de 3.3V do BeagleBone à placa protoboad. Na realidade, para este projeto específico, fazer essa conexão não serve para nada. Em geral, é uma boa prática, no entanto, ter sempre as faixas horizontais na sua placa de ensaio alimentado com uma tensão constante e com o terra do circuito.

Se você conectar o resistor a trilha positiva na sua placa de ensaio, o LED se acende, mas você não tem nenhum controle sobre ele. Sinta-se a vontade para experimentar!

# 2 Configurando e Controlando Dispositivo GPIO

A interface de uso geral combina quatro módulos de input/output de uso geral (GPIO). Cada módulo GPIO fornece 32 pinos dedicados de uso geral com capacidades de entrada e saída; Assim, a interface de propósito geral suporta até 128 (4 × 32) pinos. Esses pinos podem ser configurados para as seguintes aplicações:

- Input(captura)/output(controle) de dados
- Interface de teclado

## Módulo de clock

O registrador PRCM (0x0x44E0_0000) acompanhado do módulo de controle gerencia a passagem (isto é, a desativação) e a ativação dos clocks aos módulos do dispositivo. Os clocks são gerenciados com base nas restrições de exigência dos módulos associados. Primeira coisa que deve realizar é a inicialização do módulo CM_PER clock para o GPIO módulo 1, como mostrado no exemplo abaixo.

```
HWREG(SOC_CM_PER_REGS + module) |= setting

Onde:
setting =  (1<<18) | (0x2<<0)
module = CKM_PER_GPIO1_CLKCTRL
CKM_PER_GPIO1_CLKCTRL  (0x0AC)

SOC_CM_PER_REGS  = SOC_PRCM_REGS + 0
SOC_PRCM_REGS  (0x44E00000)
```

## Módulo de Controle

O módulo de controle inclui status e lógica de controle não endereçados dentro dos periféricos ou o resto da infraestrutura do dispositivo. Este módulo fornece interface para controlar as seguintes áreas do dispositivo:

- Functional I/O multiplexing
- Device control and status
- DDR PHY control and IO control registers
- EDMA event multiplexing control registers

O módulo de controle responde apenas ao tipo de dispositivo e POR (Power On Reset) interno. Ao ligar, os valores de reset para os registradores definem o estado seguro para o dispositivo. No modo de inicialização, somente os módulos a serem usados no momento da inicialização estão associados às PADs. Outros módulos de entrada são internamente ligadas e as placas de saída são desligadas. Após POR, o software define os registros de multiplexação funcional e de configuração do PAD para os valores desejados de acordo com a configuração do dispositivo solicitada.

Os registos de controlo PAD são registadores de 32 bits para controlar o sinal muxing e outros aspectos de cada PAD I/O. Após o POR, o software deve configurar a funcionalidade de multiplexação do PAD e configurações dos registros para os valores desejados de acordo com a configuração de dispositivo solicitada. A configuração é controlada pelos PADs ou por um grupo de PAD. Cada pino configurável tem seu próprio registrador de configuração para controle pullup/down e para a atribuição a um dado módulo.

Para realizar a inicialização do pino 28 para o GPIO módulo 1, siga as instruções como mostrado no exemplo abaixo.

```
HWREG(SOC_CONTROL_REGS + module) |= mode

Onde:
mode = 7
module = CM_conf_gpmc_ben1
CM_conf_gpmc_ben1          (0x0878)
SOC_CONTROL_REGS    (0x44E10000)
```

## Direção do GPIO

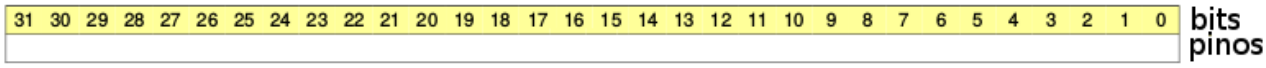
O registrador GPIO_OE é usado para habilitar os recursos de saída dos pinos. Na reinicialização, todos os pinos GPIO relacionados são configurados como entrada e capacidades de saída estão desativados. Este registrador não é usado com o módulo, sua única função é carregar a configuração dos PADs. Quando o aplicativo está usando um pino como uma saída e não quer a geração de interrupção a partir deste pino, o aplicativo pode/tem que configurar corretamente os registradores que habilita interrupção.

```
addr_temp = SOC_GPIO_1_REGS + GPIO_OE
val_temp =  HWREG(addr_temp)

val_temp &= ~(1<<pin);
val_temp |= (dir<<pin);

HWREG(addr_temp) = val_temp;

Onde:
dir = 0
SOC_GPIO_1_REGS      (0x4804C000)
GPIO_OE                    (0x134)
```

## Registrador Set de Saída de Dados (GPIO_SETDATAOUT)

Uma operação de escrita no registrador set de saída de dados, significa setar 1 no bit correspondente no registrador set de saída de dados, onde temos 32 bits para 32 pinos como mostrado na figura abaixo; Um bit escrito 0 não tem efeito. Uma leitura do registrador set de saída de dados retorna o valor do registrador.

**Figura 1:** GPIO_SETDATAOUT Register

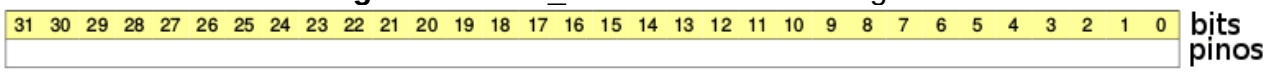



```
addr_temp = SOC_GPIO_0_REGS + GPIO_SETDATAOUT;
val_temp = 1<<pin;

HWREG(addr_temp) |= val_temp;

Onde:
SOC_GPIO_0_REGS         (0x44E07000)
GPIO_SETDATAOUT         (0x194)
```

## Registrador Clear de Saída de Dados (GPIO_CLEARDATAOUT)

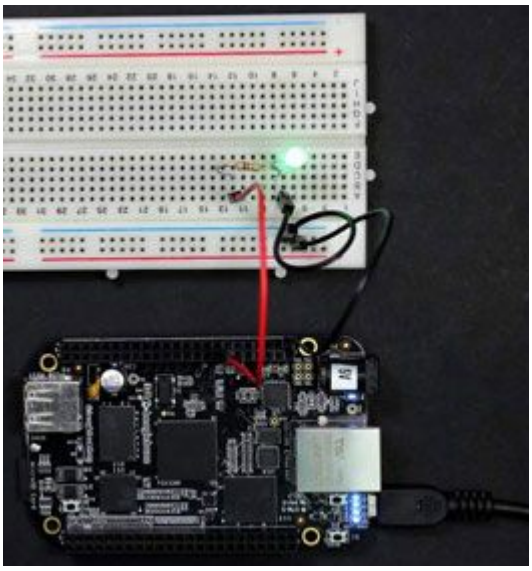
Uma operação de escrita no registrador clear de saída de dados, significa setar 1 no bit correspondente no registrador clear de saída de dados, onde temos 32 bits para 32 pinos como mostrado na figura abaixo; Um bit escrito 0 não tem efeito. Uma leitura do registrador set de saída de dados retorna o valor do registrador.

**Figura 2:** GPIO_CLEARDATAOUT Register




```
addr_temp = SOC_GPIO_0_REGS + GPIO_CLEARDATAOUT;
val_temp = 1<<pin;

HWREG(addr_temp) |= val_temp;

Onde:
SOC_GPIO_0_REGS         (0x44E07000)
GPIO_SETDATAOUT         (0x190)
```

Agora o LED deve estar em:



↗️ Se a intensidade luminosa do LED parece fraca, tente um valor menor de resistência. Não tente inferior a 220Ohms, apesar de tudo.

**PRÀTICA:** Agora escreva um programa em C que controle o LED das mais diversas formas.