Manual de Uso — Central de Alarme Residencial

1. Visão Geral

Este sistema de alarme residencial foi desenvolvido sobre uma plataforma Raspberry Pi 3 Model B+, executando Ubuntu Server 20.04 LTS. A implementação utiliza a linguagem C em conjunto com a biblioteca WiringPi para controle de GPIO. O sistema monitora sensores digitais de segurança — incluindo abertura de portas, movimento por PIR (Infravermelho Passivo) e presença de gás inflamável — e aciona alertas sonoros e visuais via buzzer e LEDs.
O serviço é configurado para iniciar automaticamente com o sistema operacional via Systemd, garantindo alta disponibilidade após reinicializações ou quedas de energia.

2. Arquitetura e Componentes do Sistema

Hardware:

- Placa Controladora: Raspberry Pi 3 Model B+
- Fonte de Alimentação Estável: 5V/2.5A
- Sensores:
  - Sensor magnético de porta — GPIO 14
  - Sensor de movimento PIR — GPIO 15
  - Sensor de gás (MQ-2) — GPIO 18
- Indicadores de Alerta:
  - LED vermelho (porta aberta) — GPIO 23
  - LED verde (movimento) — GPIO 24
  - Buzzer ativo (gás detectado) — GPIO 25

Software:

- Sistema Operacional: Ubuntu Server 20.04.
- Biblioteca GPIO: WiringPi (instalada manualmente).
- Repositório de Código: https://github.com/Guilherme-M-X/SistemasEmbarcados

- Local do Executável: /usr/local/bin/WiringPi/central
- Serviço Systemd: /etc/systemd/system/central.service

3. Instalação do Sistema

3.1 Conexões Físicas:

Conecte os sensores e atuadores aos pinos GPIO conforme a tabela:
- Sensor de Porta: GPIO 14
- Sensor PIR: GPIO 15
- Sensor de Gás (MQ-2): GPIO 18
- LED Vermelho: GPIO 23
- LED Verde: GPIO 24
- Buzzer: GPIO 25

3.2 Configuração Inicial:

1. Clone e compile a biblioteca WiringPi (caso não esteja instalada):

sudo apt update && sudo apt upgrade
sudo apt install make (caso não esteja instalado)
git clone https://github.com/WiringPi/WiringPi.git
cd WiringPi
./build

2. Compile o código-fonte:
   gcc central.c -o central -lwiringPi
   sudo mv central /usr/local/bin/WiringPi/

3. Crie o arquivo de serviço systemd e habilite-o:
   sudo nano /etc/systemd/system/central.service

[Unit]
Description=Central de Alarme Residencial

After=network.target

[Service]
ExecStart=/usr/local/bin/WiringPi/central
Restart=always
User=pi

[Install]
WantedBy=multi-user.target

Habilitar:
sudo systemctl enable central
sudo systemctl start central

4. Funcionamento do Sistema

O sistema funciona de forma assíncrona utilizando interrupções para garantir baixa latência na detecção de eventos. Cada sensor é monitorado em tempo real:

- Porta aberta: Acende o LED vermelho.
- Movimento detectado: Acende o LED verde.
- Gás detectado: Aciona o buzzer sonoro.

Todos os indicadores são desativados automaticamente após o fim da condição.

Mensagens no terminal:
Ao ser iniciado manualmente, o sistema exibe:

'Sistema iniciado. Aguardando eventos...'.

Quando iniciado via systemd, opera em segundo plano.

5. Operação e Manutenção

5.1 Acesso Remoto:

Para acessar o sistema remotamente, obtenha o IP da Raspberry Pi e conecte-se via SSH:

ssh rasp@<endereço_IP>

5.2 Comandos de Gerenciamento:

```
sudo systemctl status central   # Verifica o status
sudo systemctl start central    # Inicia o serviço
sudo systemctl stop central     # Para o serviço
sudo systemctl restart central  # Reinicia o serviço
```

5.3 Boas Práticas:

- Realizar testes regulares dos sensores.
- Proteger fisicamente a Raspberry Pi contra variações térmicas e umidade.
- Substituir periodicamente sensores sujeitos a desgaste mecânico.
- Manter o sistema atualizado com patches de segurança do sistema operacional.

6. Diagnóstico e Solução de Problemas

- Sistema não responde:

  * Reinicie o Serviço e caso necessário a Raspberry.
  * Verifique a fonte de alimentação.
  * Utilize o comando 'systemctl status central' para verificar o serviço.
  * Examine os logs com 'journalctl -u central'.

- Sensores não disparam eventos:

  * Teste manualmente com scripts simples.
  * Verifique as conexões físicas e pinos GPIO.
  * Confirme se a biblioteca WiringPi está corretamente instalada.

7. Considerações Técnicas Avançadas

O código fonte (central.c) utiliza:

- Funções de interrupção com 'wiringPiISR()';
- Controle de GPIO com 'digitalWrite()';

Possíveis melhorias futuras:

- Registro de eventos em banco de dados SQLite;
- Envio de notificações via MQTT ou Telegram;
- Desenvolvimento de interface web com Flask ou Node.js;
- Implementação de modos armado/desarmado com autenticação.

8. Conclusão

O sistema de alarme proposto oferece uma solução eficiente e de baixo custo para segurança residencial, com arquitetura modular e facilidade de manutenção. Permite expansões futuras, sendo uma base sólida para projetos de automação e IoT. Para personalizações, edite o código 'central.c', recompile e substitua o binário conforme instruções.