Eletroboy

GOVERNO DO ESTADO DO RIO DE JANEIRO
SECRETARIA DE ESTADO DE CIÊNCIA, TECNOLOGIA E INOVAÇÃO - SECTI
FUNDAÇÃO DE APOIO À ESCOLA TÉCNICA - FAETEC
ESCOLA TÉCNICA ESTADUAL FERREIRA VIANA - ETEFV
COORDENAÇÃO DE ELETRÔNICA

Antônio Italo Rodrigues de Souza
Jéssica Oliveira Affonso
José de Figueredo Cunha Silva
Matheus Medeiros Carvalho
Pedro Vitor Vieira Belo

ELETROBOY

Rio de Janeiro
2025

Antônio Italo Rodrigues de Souza
Jéssica Oliveira Affonso
José de Figueredo Cunha Silva
Matheus Medeiros Carvalho
Pedro Vitor Vieira Belo

ELETROBOY

Trabalho de Conclusão de Curso apresentado à Coordenação de Eletrônica da ETE Ferreira Viana, da Fundação de apoio à Escola Técnica, como requisito parcial para a obtenção da habilitação profissional de Técnico de Nível Médio em Eletrônica sob a orientação do Professor Luis Henrique Monteiro de Castro e do Professor Marcelo de Almeida Duarte.

Rio de Janeiro
2025

Antônio Italo Rodrigues de Souza
Jéssica Oliveira Affonso
José de Figueredo Cunha Silva
Matheus Medeiros Carvalho
Pedro Vitor Vieira Belo

ELETROBOY

Aprovada em : ____ / _____ / _____ Conceito: _____

_____	_____
Prof. Luis Henrique M. de Castro	Prof. Marcelo de Almeida Duarte
ETE Ferreira Viana	ETE Ferreira Viana
Orientador	Orientador

_____
Prof. XXXXXXXXXXXXXX
ETE Ferreira Viana
ID:

_____
Prof. XXXXXXXXXXXXXX
ETE Ferreira Viana
ID:

_____
Prof. XXXXXXXXXXXXXX
ETE Ferreira Viana
ID:

Rio de Janeiro
2025

RESUMO

RODRIGUES, Antônio Italo; AFFONSO, Jéssica; FIGUEREDO, José; MEDEIROS, Matheus; BELO, Pedro Vitor. *ELETROBOY.* 2025. Trabalho de Conclusão de Curso (Em desenvolvimento) - Curso Técnico em Eletrônica, Escola Técnica Estadual Ferreira Viana, Fundação de Apoio à Escola Técnica, Rio de Janeiro, 2025.

Este projeto tem como objetivo o desenvolvimento de um dispositivo eletrônico inspirado no Game Boy da Nintendo, voltado para o auxílio no aprendizado e na rotina de profissionais da área de eletrônica. O dispositivo funcionará como uma enciclopédia digital, oferecendo conteúdos didáticos sobre eletrônica digital, e contará com quizzes interativos para reforçar o estudo ativo, que busca engajar o estudante na fixação dos conteúdos de forma mais eficiente e incentivar o estudo contínuo. O projeto também apresentará um sistema de recompensa onde, ao completar 100% das questões de um certo conteúdo, o jogo dará estrelas, indicando a conclusão da teoria. A implementação será feita com o microcontrolador ESP32, utilizando uma tela TFT para exibir os conteúdos de forma clara, e a interface contará com botões físicos, inspirados em consoles portáteis, para facilitar a navegação pelos menus. Nos testes realizados com 15 participantes, observou-se engajamento elevado, com taxas de conclusão superiores a 70% nos quizzes e retenção de aprendizado próxima a 76% após duas semanas. Os resultados demonstram que a ferramenta consegue combinar diversão e ensino, proporcionando uma experiência eficaz e motivadora para estudantes e profissionais da eletrônica.

Palavras-chave: Arduino; ESP32; Game Boy; Gamificação no ensino; Educação.

SUMÁRIO

INTRODUÇÃO

A evolução tecnológica tem transformado a maneira que o conhecimento é adquirido e disseminado, impactando profundamente o processo educacional. Na área técnica de eletrônica, o conhecimento sobre aparelhos e componentes é essencial para os futuros profissionais do ramo. Apesar do curso possuir uma forte aplicação prática, muitos dos conceitos iniciais não têm correspondência direta com o dia a dia, sendo inicialmente abstratos, o que pode dificultar o aprendizado. Por lidar com sistemas numéricos e estruturas não intuitivas no cotidiano, exigem abordagens que tornem a aprendizagem mais visual e interativa para facilitar a compreensão do conteúdo. Embora existam simuladores e materiais didáticos tradicionais, um estudo recente de Legaki et al. (2020) indica que a combinação desses métodos com abordagem gamificada em conjunto com a tradicional trazem mais eficiência ao estudo do que a aplicação de apenas um dos métodos.

A Base Nacional Comum Curricular (BNCC) estabelece os direitos de aprendizagem da Educação Básica e ressalta a importância da integração de tecnologias no processo educacional, reconhecendo seu potencial para favorecer a construção do conhecimento e o desenvolvimento de competências dos estudantes. Nesse contexto, a gamificação surge como uma estratégia que utiliza elementos de jogos para tornar o aprendizado mais dinâmico e motivador. Aplicada ao ensino de eletrônica digital, essa abordagem contribui para que conteúdos abstratos sejam compreendidos de forma mais interativa, aproximando o estudante da prática.

Pesquisas apontam que, quando aplicada de forma planejada, a gamificação favorece a retenção do conhecimento e amplia o engajamento dos estudantes. Esse processo está relacionado à transição da motivação extrínseca, baseada em recompensas externas, para a motivação intrínseca, caracterizada pelo prazer em aprender (Silva; Massaro; Paula. 2024).

No ambiente escolar, a gamificação busca engajar os estudantes de uma maneira lúdica, acompanhando as novas gerações num contexto de ascensão tecnológica, podendo tornar o processo de ensino mais interessante e agradável.

OBJETIVO

Desenvolver um console portátil que estimule os alunos a aprenderem os fundamentos

da eletrônica digital, fazendo-os reter mais informação de forma leve e tranquila. Em sua primeira versão, o dispositivo contará com uma parte teórica que contextualiza o assunto específico e uma lista de questões com diferentes níveis de dificuldade para a aplicação prática do conteúdo aprendido. O projeto pretende contribuir no ensino de eletrônica digital e fornecer uma aprendizagem que mantém a atenção dos usuários e, consequentemente, permitindo que absorvam mais o conteúdo da matéria, o que torna o ensino mais interessante, assim como tem o intuito de comprovar a eficácia dessa abordagem na motivação dos alunos e na compreensão do conhecimento, garantindo um aprendizado mais eficiente, testando o protótipo em ambiente escolar para avaliar o engajamento dos usuários e comparar o desempenho em retenção de conceitos com métodos tradicionais de ensino.

MATERIAIS E MÉTODOS:

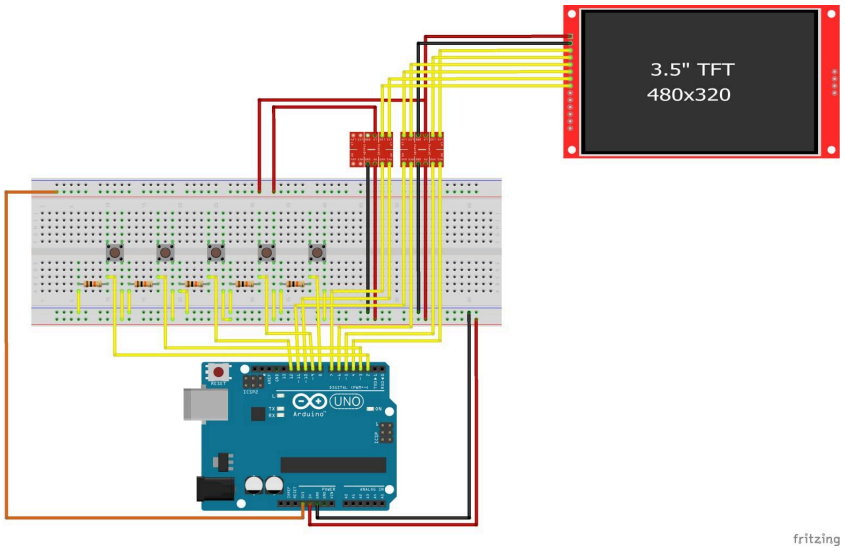
O projeto foi realizado majoritariamente nas dependências da escola, na Tijuca, no estado do Rio de Janeiro. Inicialmente, o projeto foi idealizado utilizando a plataforma Arduino Mega, com a qual foram definidos os objetivos gerais do sistema e suas funcionalidades principais. A escolha pelo Arduino Mega ocorreu por oferecer um número maior de portas digitais e analógicas, o que proporcionou mais liberdade no planejamento inicial do projeto. Para a interação do usuário, foram integrados outros componentes, como botões e resistores para garantir sua funcionalidade, além de uma bateria de Power Bank, acessível e que garante a alimentação do dispositivo.

A etapa seguinte foi a produção da interface do sistema, com foco na implementação da tela e na definição das primeiras configurações visuais e funcionais. Essa fase foi fundamental para estabelecer a forma como o usuário interage com o projeto. Durante esse processo, o Display LCD TFT 3.5″ 480x320 foi identificado, testado e, por fim, definido como o display definitivo do projeto. Sua boa integração com o Arduino Mega permitiu uma visualização clara e eficiente das informações. Apesar de suas qualidades, o modelo apresenta duas características que exigiram atenção: por não ser um shield, foi preciso utilizar jumpers para direcionar conexões e integrar outros componentes ao sistema. Adicionalmente, a conexão do display com o Arduino demandou um conversor de tensão, uma vez que o display opera com apenas 3.3V e o Arduino com 5V, o que precisou de ajustes no layout do circuito.

Durante a fase de testes e desenvolvimento, foi feita uma reavaliação da plataforma utilizada. Embora o Arduino Mega estivesse funcionando corretamente, identificou-se a

possibilidade de migrar o sistema para o Arduino Uno, devido ao seu custo-benefício mais vantajoso. Foram realizados testes com essa nova placa, cuja funcionalidade inicial era incerta, confirmando-se ao final sua boa operatividade.

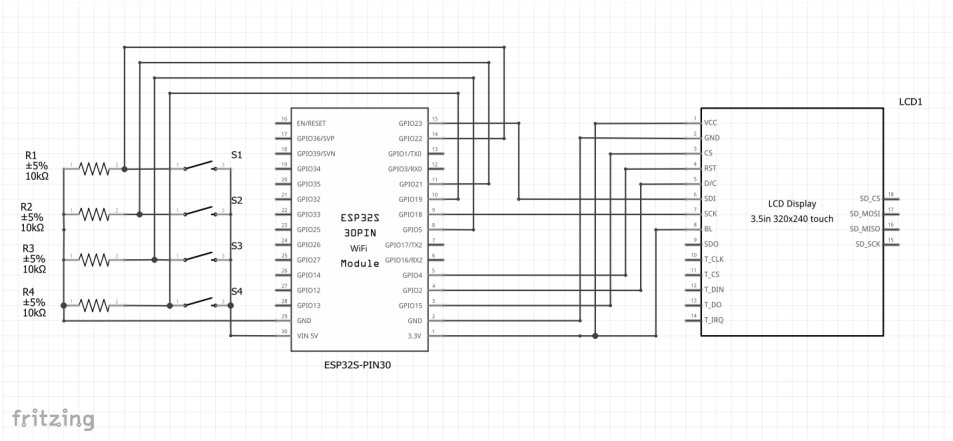
Figura 1— Primeiro diagrama esquemático do projeto, utilizando Arduino Uno e conversores de tensão.



Fonte: Autores, 2025.

Com o sistema montado, observou-se que o projeto estava muito lento e não atendia às expectativas de desempenho. Para resolver os problemas de performance, foi adquirido um ESP32. Essa mudança teve como objetivo acelerar o carregamento do programa e aumentar a capacidade de processamento gráfico do projeto. Com essa modificação, o conversor de tensão deixou de ser necessário, pois tanto o display quanto o ESP32 operam na mesma tensão.

Figura 2 — Diagrama esquemático definitivo, com a placa ESP32.



Após a estabilização da plataforma de hardware com o microcontrolador ESP32, o desenvolvimento foi direcionado ao aprimoramento da interface e à integração do conteúdo didático. No âmbito do software, foi desenvolvido o suporte a caracteres acentuados da língua portuguesa para a correta exibição de todo o material escrito. Por fim, o conteúdo educacional foi embarcado no dispositivo, incluindo a base teórica sobre eletrônica digital e um banco de questões. Tais questões foram ordenadas por níveis de dificuldade, a fim de permitir uma progressão gradual no estudo.

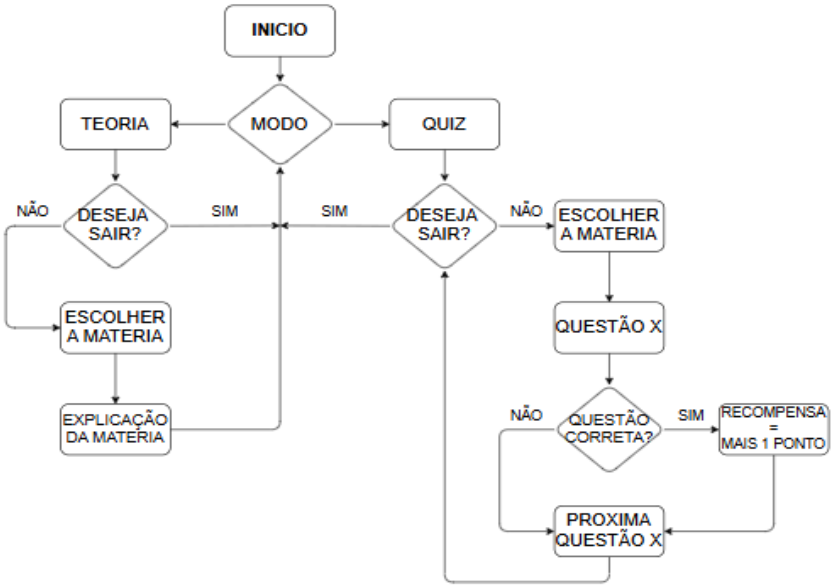
A interconexão dos componentes foi inicialmente realizada em uma protoboard para permitir testes flexíveis e ajustes no circuito. Durante esta fase, enfrentou-se um desafio significativo relacionado à interferência de sinal, manifestada como ruído elétrico vindo do ESP32, afetando o funcionamento dos botões. Para solucionar este problema, foi realizada uma otimização do layout dos fios, encurtando o comprimento dos jumpers e implementado um plano de aterramento comum e robusto para todos os componentes, o que se mostrou eficaz na supressão do ruído. Após o circuito funcionar perfeitamente, as conexões foram transferidas para uma placa de fenolite perfurada e soldadas permanentemente, garantindo a durabilidade das ligações elétricas para o uso contínuo do dispositivo.

Segue abaixo uma lista dos materiais utilizados e a quantia de cada um. Convém destacar que o valor e o frete variam conforme a loja e a localização:

Tabela 1- Lista de material e custo estimado:

Componentes	Preço	Frete	Loja
Placa ESP32 DevKit V1	R$47,90	—	Mercado Livre
Display LCD TFT 3,5 polegadas	R$147,15	R$11,81	MakerHero
Push-Button 6x6x4,2mm - 10 peças	R$1,14	R$10,76	MakerHero
Power Bank 5000mah	R$18,30	—	Mercado Livre
140 Jumpers Rígidos Coloridos Protoboard Caixa Organizadora	R$21,88	—	Mercado Livre

Figura 2 - Fluxograma de funcionamento:



RESULTADOS E DISCUSSÃO:

Foram avaliados 15 participantes (10 alunos do primeiro ano do curso técnico de eletrônica, 2 professores da área técnica e 3 professores do ensino médio) que utilizaram o Eletroboy até que assimilassem cada conteúdo.

Em média, os participantes realizaram 2 sessões para “Conceito de bit, byte e nibble”, 2 sessões para “Sistemas de numeração”e 3 sessões para “Portas lógicas”. Esses resultados

sugerem que conteúdos mais abstratos, como portas lógicas, demandaram maior tempo de estudo e revisão por parte dos usuários.

A adesão ao formato gamificado foi considerada satisfatória: mais de 87% dos usuários completaram 100% dos quizzes de “Conceito de bit, byte e nibble” e 73% completaram todos os quizzes de “Portas lógicas”. Tais números evidenciam que o sistema de recompensas exerceu papel motivador, em consonância com o que estudos apontam sobre o uso de metodologias ativas e da gamificação como ferramentas de engajamento, facilitando a absorção dos assuntos abordados (Madureira; Schneider, 2021).

A fixação do conteúdo foi avaliada por meio de um teste aplicado uma semana e duas semanas após o último uso do dispositivo. Os resultados apontaram que os acertos médios se mantiveram relativamente estáveis, indicando retenção do aprendizado.

Tabela 2 — Comparação de desempenho médio dos participantes nos quizzes e nas avaliações posteriores:

Etapa de avaliação	**Acertos médios (%)**
Quizzes imediatos (no Eletroboy)	82%
Teste após 1 semana	78%
Teste após 2 semanas	76%

O grau de satisfação, medido de 1 (muito insatisfeito) a 5 (muito satisfeito), obteve média geral de 4,3. Aspectos mais bem avaliados foram:

- Usabilidade dos botões físicos: 4,5
- Clareza dos conteúdos teóricos: 3,2
- Percepção de aprendizado: 4,2

Esses resultados demonstram que tanto a interface quanto a apresentação dos conteúdos foram considerados adequados pelos participantes, confirmando a viabilidade do dispositivo como recurso de apoio ao ensino técnico.

A nota mais baixa atribuída à clareza dos conteúdos teóricos (3,2) deve-se, principalmente, às dificuldades relatadas por professores de outras áreas, que não tinham

familiaridade prévia com conceitos básicos de eletrônica. Isso reforça a necessidade de ajustes no detalhamento dos conteúdos para públicos totalmente iniciantes.

De forma geral, tanto a interface quanto a apresentação dos conteúdos foram consideradas adequadas pelos participantes, confirmando a viabilidade do dispositivo como recurso de apoio ao ensino técnico. Comentários qualitativos também reforçam essa percepção: um dos alunos destacou que “as estrelas me motivaram a repetir os quizzes até acertar tudo”, enquanto um professor de eletrônica digital ressaltou que “o dispositivo facilita a revisão rápida de conceitos de lógica digital”.

Ao analisar os diferentes grupos de participantes, observou-se que os professores da área técnica tiveram maior facilidade em concluir os quizzes com menor número de tentativas, enquanto os professores de outras áreas necessitaram de mais sessões, especialmente nos conteúdos de Sistemas de numeração e Portas lógicas. Já os alunos do primeiro ano demonstraram alta motivação pelo formato gamificado, mesmo quando enfrentam dificuldades iniciais.

Apesar dos resultados positivos, algumas limitações devem ser destacadas. O número de participantes era limitado às dependências da escola, restringindo a generalização dos dados. Além disso, os testes ocorreram em um período curto, restringindo a avaliação dos impactos a longo prazo.

CONSIDERAÇÕES FINAIS:

A partir deste trabalho foi concluído que a utilização do Eletroboy como recurso de gamificação no ensino de eletrônica se mostrou uma estratégia eficaz para despertar o interesse dos estudantes e facilitar a compreensão de conceitos técnicos. Ao transformar o processo de aprendizado em uma experiência interativa, observou-se maior engajamento, motivação e retenção do conteúdo, especialmente entre aqueles que não possuíam conhecimento prévio na área.

Dessa forma, fica evidente que a gamificação pode ser aplicada como uma ferramenta complementar ao ensino tradicional, contribuindo para tornar a aprendizagem mais atrativa, dinâmica e significativa. Além disso, o Eletroboy representa um exemplo de como a tecnologia pode ser integrada à educação para aproximar os estudantes de conteúdos complexos de maneira acessível e prazerosa.

REFERÊNCIAS:

BRASIL. **Base nacional comum curricular: educação é a base: ensino médio.** Brasília, DF: MEC, 2018. Disponível em: http://basenacionalcomum.mec.gov.br/. Acesso em: 12 jun 2025.

LEGAKI, N.-Z.; XI, N.; HAMARI, J.; KARPOUZIS, K.; ASSIMAKOPOULOS, V. **The effect of challenge-based gamification on learning: An experiment in the context of statistics education.** International Journal of Human-Computer Studies, v. 144, p. 102496, 2020. Disponível em: https://doi.org/10.1016/j.ijhcs.2020.102496. Acesso em: 10 mar 2025.

MADUREIRA, J.; SILVA, J.; SCHNEIDER, H. **Gamificação no Ensino de Programação de Computadores em turmas do Ensino Médio: uma experiência com o software Kahoot!** v. 19, n. 2 p. 121191, 2021. Disponível em: https://seer.ufrgs.br/index.php/renote/article/view/121191/65830 . Acesso em: 17 mar 2025.

SILVA, C. M.; MASSARO, R.; PAULA, A. V. de. **A gamificação como metodologia ativa no processo de ensino-aprendizagem no ensino superior.** Revista Valore, v. 9, e-9014, 2024. Disponível em: https://revistavalore.emnuvens.com.br/valore/article/view/1341. Acesso em: 24 jun. 2025.

ANEXOS:

Anexo A - Código ESP (Arduino IDE):

```
#include <TFT_eSPI.h>
#include <SPI.h>
#include <EEPROM.h>
#include <string.h> // strcpy, strlen, etc.

#define PBD 13   // Nav 1 (Menu: Descer / Teclado: Direita) - (OK)
#define PBS 14   // Nav 2 (Menu: Subir / Teclado: Baixo)   - (OK)
#define PBA 5    // confirmar / A                          - (OK)
#define PBB 26   // backspace / B                          - (OK)

TFT_eSPI lcd = TFT_eSPI();

#define GAMEBOY_BG   0x9FC7
#define GAMEBOY_TEXT 0x0200
#define GAMEBOY_GRAY 0x738E // Cor cinza para itens desabilitados

#define EEPROM_SIZE 150
#define MAX_JOGADORES 5
#define TAM_NOME 8
```

```
// --- Lógica de Slots ---
#define MAX_SAVE_SLOTS 3
#define BYTES_NOME_POR_SLOT (TAM_NOME + 1)
#define BYTES_SCORE_POR_SLOT sizeof(int)
#define BYTES_ESTRELAS_POR_SLOT MAX_MATERIAS
#define BYTES_QUIZ_FLAGS_PER_SLOT sizeof(uint32_t)
#define BYTES_POR_SLOT_COMPLETO (BYTES_NOME_POR_SLOT +
BYTES_SCORE_POR_SLOT + BYTES_ESTRELAS_POR_SLOT +
BYTES_QUIZ_FLAGS_PER_SLOT) // 27 bytes
#define EEPROM_OFFSET_PLACAR (MAX_SAVE_SLOTS *
BYTES_POR_SLOT_COMPLETO) // 81

#define MAX_MATERIAS 6
const int QUIZZES_POR_MATERIA = 3;
const int OPCOES_POR_QUIZ = 4;

// --- Configuração do Buzzer e Notas ---
const int BUZZZER_PIN = 27;
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_E6  1319

// --- Bitmap da Chave (11x17) ---
const unsigned char large_key_bmp[] PROGMEM = {
  0x1F, 0x00, // 00011111000
  0x20, 0x80, // 00100000100
  0x40, 0x40, // 01000000010
  0x40, 0x40, // 01000000010
  0x40, 0x40, // 01000000010
  0x40, 0x40, // 01000000010
  0x20, 0x80, // 00100000100
```

```
  0x1F, 0x00, // 00011111000
  0xFF, 0xC0, // 11111111110
  0x0E, 0x00, // 00001110000
  0x0E, 0x00, // 00001110000
  0x0E, 0x00, // 00001110000
  0x0E, 0x00, // 00001110000
  0x0E, 0x00, // 00001110000
  0x0E, 0x00, // 00001110000
  0x1F, 0x00, // 00011111000
  0x0E, 0x00  // 00001110000
};

// ---------------- TECLADO ----------------
const int K_ROWS = 4;
const int K_COLS = 8;
const String keyGrid[K_ROWS][K_COLS] = {
{"A","B","C","D","E","F","G","H"},
{"I","J","K","L","M","N","O","P"},
{"Q","R","S","T","U","V","W","X"},
{"Y","Z","SP","BK","-","-","OK","-"}
};

// ---------------- menus e teoria ----------------
String menuPrincipal[] = {
"Novo Jogo",
"Continuar",
"Placar",
"Reset"
};

String menuMaterias[] = {
" Bits e Bytes",
"Sistemas de Numeracao",
"Portas Logicas",
"Algebra de Boole",
"Expressoes Logicas",
"Aritmetica Binaria"
};

struct PaginaTeoria {
String linhas[9];
};
```

```
PaginaTeoria teorias[MAX_MATERIAS][4];

// {Bits, Sist.Num, Portas, Boole, Expr, Aritm}
const int numPaginasPorMateria[MAX_MATERIAS] = {4, 4, 3, 4, 4, 4};

// ---------------- quiz struct ----------------
struct Quiz {
String pergunta1;
String pergunta2;
String opcoes[OPCOES_POR_QUIZ];
int correta;
};
Quiz quizzes[MAX_MATERIAS][QUIZZES_POR_MATERIA];

// ---------------- variáveis gerais ----------------
const int materiasPorPagina = 5;
int paginaMaterias = 0;
const int totalMaterias = MAX_MATERIAS;

// Fases: 0:Menu, 1:Materias, 2:Quiz, 3:Teoria, 4:Continuar
int totalItens[] = {4, materiasPorPagina + 2, 4, 4, 3};

int itemSelecionado = 0;
int paginaTeoria = 0;
bool trava1 = false, trava2 = false;
int materiaAtual = 0;

bool telaMenu = true, telaMaterias = false, telaTeoria = false, telaQuiz = false, telaContinuar
= false;
bool modoNovoJogo = false;

struct Botao {
int pino;
bool ultimoLeitura;
bool clicado;
unsigned long ultimaMudanca;
};
const unsigned long DEBOUNCE_DELAY = 40;
Botao b_nav1      = {PBD, HIGH, false, 0};
Botao b_nav2      = {PBS, HIGH, false, 0};
Botao b_A         = {PBA, HIGH, false, 0};
Botao b_B         = {PBB, HIGH, false, 0};
```

```
struct Jogador {
char nome[TAM_NOME + 1];
int score;
};
Jogador placar[MAX_JOGADORES];
String nomeJogadorAtual = "";
int scoreAtual = 0;

int saveSlotAtual = 0;

// estado do quiz
int paginaQuiz = 0;
int acertosQuiz = 0;
int materiaAtualQuiz = 0;

// ---------------- teclado ----------------
int selRow = 0;
int selCol = 0;
int prevSelRow = 0;
int prevSelCol = 0;

const int GRID_X = 12;
const int GRID_Y = 80;
const int CELL_W = 58;
const int CELL_H = 36;
const int CELL_GAP_X = 6;
const int CELL_GAP_Y = 10;

// ---------------- protótipos ----------------
int getEstrelasTotaisSlot(int slot);
String lerNomeSlot(int slot);
void salvarNomeSlot(int slot, String nome);
int lerScoreSlot(int slot);
void salvarScoreSlot(int slot, int score);
bool verificaSavesExistem();
void mostrarTelaContinuar();
void atualizarMenu();
void menuMateriasTela();
void mostrarTeoria();
void mostrarQuiz();
void atualizarQuiz();
```

```
bool isQuizPointClaimed(int materia, int quizNum);
void claimQuizPoint(int materia, int quizNum);
void tocarVitoria();

// ---------------- leitura de botões ----------------
bool botaoCliqueUnico(Botao &botao) {
bool leitura = digitalRead(botao.pino);
if (leitura != botao.ultimoLeitura) botao.ultimaMudanca = millis();
if ((millis() - botao.ultimaMudanca) > DEBOUNCE_DELAY) {
if (leitura == LOW && !botao.clicado) {
botao.clicado = true;
botao.ultimoLeitura = leitura;
return true;
}
if (leitura == HIGH) botao.clicado = false;
}
botao.ultimoLeitura = leitura;
return false;
}


// ---------------- EEPROM: Funções de Endereço ----------------
int getAddrNome(int slot) { return (slot * BYTES_POR_SLOT_COMPLETO); }
int getAddrScore(int slot) { return (slot * BYTES_POR_SLOT_COMPLETO) +
BYTES_NOME_POR_SLOT; }
int getAddrEstrelas(int slot) { return (slot * BYTES_POR_SLOT_COMPLETO) +
BYTES_NOME_POR_SLOT + BYTES_SCORE_POR_SLOT; }
int getAddrQuizFlags(int slot) { return (slot * BYTES_POR_SLOT_COMPLETO) +
BYTES_NOME_POR_SLOT + BYTES_SCORE_POR_SLOT +
BYTES_ESTRELAS_POR_SLOT; }

// ---------------- EEPROM: progresso ----------------
void salvarNomeSlot(int slot, String nome) {
   if (slot < 0 || slot >= MAX_SAVE_SLOTS) return;
   int addr = getAddrNome(slot);
   char nomeChar[TAM_NOME + 1];
   strncpy(nomeChar, nome.c_str(), TAM_NOME);
   nomeChar[TAM_NOME] = '\0';
   EEPROM.put(addr, nomeChar);
}
String lerNomeSlot(int slot) {
   if (slot < 0 || slot >= MAX_SAVE_SLOTS) return "";
```

```
  int addr = getAddrNome(slot);
  char nomeChar[TAM_NOME + 1];
  EEPROM.get(addr, nomeChar);
  nomeChar[TAM_NOME] = '\0';
  if (nomeChar[0] == '\0' || (byte)nomeChar[0] == 0xFF) return "";
  return String(nomeChar);
}
void salvarScoreSlot(int slot, int score) {
  if (slot < 0 || slot >= MAX_SAVE_SLOTS) return;
  EEPROM.put(getAddrScore(slot), score);
  EEPROM.commit();
}
int lerScoreSlot(int slot) {
  if (slot < 0 || slot >= MAX_SAVE_SLOTS) return 0;
  int score = 0;
  EEPROM.get(getAddrScore(slot), score);
  if (score == -1 || score == 0xFFFFFFFF) score = 0;
  return score;
}
bool isQuizPointClaimed(int materia, int quizNum) {
  uint32_t flags = 0;
  EEPROM.get(getAddrQuizFlags(saveSlotAtual), flags);
  int bit_index = (materia * QUIZZES_POR_MATERIA) + quizNum;
  return (flags & (1UL << bit_index));
}
void claimQuizPoint(int materia, int quizNum) {
  uint32_t flags = 0;
  EEPROM.get(getAddrQuizFlags(saveSlotAtual), flags);
  int bit_index = (materia * QUIZZES_POR_MATERIA) + quizNum;
  flags |= (1UL << bit_index);
  EEPROM.put(getAddrQuizFlags(saveSlotAtual), flags);
}
void salvarProgresso(int materia, int estrelas) {
if (materia < 0 || materia >= MAX_MATERIAS) return;
if (estrelas < 0) estrelas = 0;
if (estrelas > 3) estrelas = 3;
int addr = getAddrEstrelas(saveSlotAtual) + materia;
EEPROM.write(addr, estrelas);
EEPROM.commit();
}
int estrelasMateria(int materia) {
if (materia < 0 || materia >= MAX_MATERIAS) return 0;
```

```
int addr = getAddrEstrelas(saveSlotAtual) + materia;
int v = EEPROM.read(addr);
if (v < 0 || v > 3) return 0;
return v;
}
int getEstrelasTotaisSlot(int slot) {
   int totalEstrelas = 0;
   int offset = getAddrEstrelas(slot);
   for (int i = 0; i < BYTES_ESTRELAS_POR_SLOT; i++) {
      int v = EEPROM.read(offset + i);
      if (v > 0 && v <= 3) totalEstrelas += v;
   }
   return totalEstrelas;
}
bool verificaSavesExistem() {
   for (int i = 0; i < MAX_SAVE_SLOTS; i++) {
      if (lerNomeSlot(i).length() > 0) return true;
   }
   return false;
}

void tocarVitoria() {
  int melody[] = {
    NOTE_CS4, NOTE_B3, NOTE_A3, NOTE_B3, NOTE_CS4,
    NOTE_E4, NOTE_DS4, NOTE_B3, NOTE_GS4, NOTE_FS4,
    NOTE_E4, NOTE_E6
  };
  int durations[] = { 9, 8, 9, 8, 8, 8, 8, 8, 8, 8, 8, 9 };
  int pauses[]    = { 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0 };
  int notas = sizeof(melody) / sizeof(melody[0]);
  int tempoBPM = 120;
  float semibreveMs = 60000.0 / tempoBPM;
  for (int i = 0; i < notas; i++) {
    float duracaoNota = semibreveMs * (4.0 / durations[i]);
    float pausaMs = duracaoNota * (pauses[i] / 4.0);
    tone(BUZZZER_PIN, melody[i], duracaoNota);
    delay(duracaoNota + pausaMs);
    noTone(BUZZZER_PIN);
  }
}

// ---------------- EEPROM: placar ----------------
```

```
void resetarEEPROM() {
for (int i = 0; i < EEPROM_SIZE; i++) EEPROM.write(i, 0);
EEPROM.commit();
lcd.fillScreen(GAMEBOY_BG);
lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
String msg = "Progresso resetado!";
lcd.setTextSize(3);
int x_reset = (lcd.width() - (msg.length() * 6 * 3)) / 2;
lcd.setCursor(x_reset, 120);
lcd.print(msg);
delay(1200);
}
void salvarPlacar() {
for (int i = 0; i < MAX_JOGADORES - 1; i++)
for (int j = 0; j < MAX_JOGADORES - i - 1; j++)
if (placar[j].score < placar[j + 1].score) {
Jogador tmp = placar[j];
placar[j] = placar[j+1];
placar[j+1] = tmp;
}
int addr = EEPROM_OFFSET_PLACAR;
for (int i = 0; i < MAX_JOGADORES; i++) {
EEPROM.put(addr, placar[i].nome); addr += TAM_NOME + 1;
EEPROM.put(addr, placar[i].score); addr += sizeof(int);
}
EEPROM.commit();
}
void carregarPlacar() {
int addr = EEPROM_OFFSET_PLACAR;
for (int i = 0; i < MAX_JOGADORES; i++) {
EEPROM.get(addr, placar[i].nome); addr += TAM_NOME + 1;
EEPROM.get(addr, placar[i].score); addr += sizeof(int);
placar[i].nome[TAM_NOME] = '\0';
if (placar[i].score == -1 || placar[i].score == 0xFFFFFFFF || strlen(placar[i].nome) == 0 ||
(byte)placar[i].nome[0] == 0xFF) {
   strcpy(placar[i].nome, "Vazio"); placar[i].score = 0;
}
}
}
void atualizarPlacar(String nome, int novoScore) {
if (nome.length() == 0) return;
bool placarModificado = false;
```

```
for (int i = 0; i < MAX_JOGADORES; i++) {
if (strcmp(placar[i].nome, nome.c_str()) == 0) {
if (novoScore > placar[i].score) { placar[i].score = novoScore; placarModificado = true; }
if(placarModificado) salvarPlacar(); return;
}
}
for (int i = 0; i < MAX_JOGADORES; i++) {
if (placar[i].score == 0 || strcmp(placar[i].nome, "Vazio") == 0) {
strncpy(placar[i].nome, nome.c_str(), TAM_NOME); placar[i].nome[TAM_NOME] = '\0';
placar[i].score = novoScore; placarModificado = true; break;
}
}
if(placarModificado) { salvarPlacar(); return; }
int minIdx = 0;
for (int i = 1; i < MAX_JOGADORES; i++) if (placar[i].score < placar[minIdx].score)
minIdx = i;
if (novoScore > placar[minIdx].score) {
strncpy(placar[minIdx].nome, nome.c_str(), TAM_NOME);
placar[minIdx].nome[TAM_NOME] = '\0';
placar[minIdx].score = novoScore; placarModificado = true;
}
if(placarModificado) salvarPlacar();
}

// ---------------- UI ----------------
void desenharCelula(int r, int c, bool selected) {
int x = GRID_X + c * (CELL_W + CELL_GAP_X);
int y = GRID_Y + r * (CELL_H + CELL_GAP_Y);
if (selected) { lcd.fillRect(x - 3, y - 3, CELL_W + 6, CELL_H + 6, GAMEBOY_TEXT);
lcd.setTextColor(GAMEBOY_BG, GAMEBOY_TEXT); }
else { lcd.fillRect(x - 3, y - 3, CELL_W + 6, CELL_H + 6, GAMEBOY_BG);
lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG); }
lcd.setTextSize(2);
int tx = x + 6; int ty = y + 8;
String k = keyGrid[r][c];
if (k == "SP") k = "_"; else if (k == "BK") k = "<-"; else if (k == "OK") k = "OK"; else if (k
== "-") k = " ";
lcd.setCursor(tx, ty); lcd.print(k);
}
void desenharGradeInicial() { for (int r = 0; r < K_ROWS; r++) for (int c = 0; c < K_COLS;
c++)  desenharCelula(r, c, (r == selRow && c == selCol)); }
void atualizarNomeArea(const String &nome) {
```

```
int nx = 12, ny = 36, nw = lcd.width() - 24, nh = 36;
lcd.fillRect(nx, ny, nw, nh, GAMEBOY_BG); lcd.setTextColor(GAMEBOY_TEXT,
GAMEBOY_BG); lcd.setTextSize(3);
String mostr = nome; if (mostr.length() < TAM_NOME) mostr += "_";
lcd.setCursor(nx, ny); lcd.print(F("Nome: ")); lcd.print(mostr);
}
void mostrarTecladoSimplificado(const String &nome) {
lcd.fillScreen(GAMEBOY_BG); lcd.setTextSize(2); lcd.setTextColor(GAMEBOY_TEXT,
GAMEBOY_BG);
lcd.fillRect(0, 0, lcd.width(), GRID_Y - 8, GAMEBOY_BG);
lcd.setCursor(20, 8); lcd.print(F("Digite seu nome (OK p/ confirmar)"));
atualizarNomeArea(nome); desenharGradeInicial();
}
void entrarNomeJogador() {
String nome = ""; selRow = 0; selCol = 0; prevSelRow = selRow; prevSelCol = selCol;
mostrarTecladoSimplificado(nome);
while (true) {
  bool btnX = botaoCliqueUnico(b_nav1); bool btnY = botaoCliqueUnico(b_nav2); bool
btnA = botaoCliqueUnico(b_A); bool btnB = botaoCliqueUnico(b_B);
  bool moved = false;
  if (btnX) { selCol = (selCol + 1) % K_COLS; moved = true; }
  if (btnY) { selRow = (selRow + 1) % K_ROWS; moved = true; }
  if (moved) { desenharCelula(prevSelRow, prevSelCol, false); desenharCelula(selRow,
selCol, true); prevSelRow = selRow; prevSelCol = selCol; }
  if (btnA) {
    String k = keyGrid[selRow][selCol];
    if (k == "BK") { if (nome.length() > 0) nome.remove(nome.length() - 1); }
    else if (k == "OK") { if (nome.length() == 0) nome = "PLAYER"; nomeJogadorAtual =
nome; salvarNomeSlot(saveSlotAtual, nomeJogadorAtual); scoreAtual = 0; break; }
    else if (k == "SP") { if (nome.length() < TAM_NOME) nome += ' '; }
    else if (k != "-") { if (nome.length() < TAM_NOME) nome += k; }
    atualizarNomeArea(nome);
  }
  if (btnB) { if (nome.length() > 0) { nome.remove(nome.length() - 1);
atualizarNomeArea(nome); } }
  delay(30);
}
}
void mostrarPlacar() {
 lcd.fillScreen(GAMEBOY_BG);
 lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
```

```
  String msg = "PLACAR";
  lcd.setTextSize(3);
  int x_placar = (lcd.width() - (msg.length() * 6 * 3)) / 2;
  lcd.setCursor(x_placar, 20);
  lcd.print(msg);

  carregarPlacar();
  int y = 80;

  lcd.setTextSize(3);
  int font_width = 6 * 3;

  for (int i = 0; i < MAX_JOGADORES; i++) {
    String placarLinha = String(i + 1) + ". " + String(placar[i].nome) + " - " +
String(placar[i].score) + " pts";

    int x_jogador = (lcd.width() - (placarLinha.length() * font_width)) / 2;

    lcd.setCursor(x_jogador, y);
    lcd.print(placarLinha);
    y += 40;
  }

  // --- Easter Egg Chave Grande (11x17) ---
  int bmp_x = lcd.width() - 11 - 5;   // 5 pixels de padding (usando 11 de largura)
  int bmp_y = lcd.height() - 17 - 5; // 5 pixels de padding (usando 17 de altura)
  lcd.drawBitmap(bmp_x, bmp_y, large_key_bmp, 11, 17, GAMEBOY_TEXT);
  // --- Fim da mudança ---

  unsigned long t0 = millis();
  while (millis() - t0 < 3000) {
    if (botaoCliqueUnico(b_A) || botaoCliqueUnico(b_B)) break;
    delay(50);
  }
  mudarTela(atualizarMenu, true, false, false, false, false);
}
void mostrarTelaInsertCoin() {
lcd.fillScreen(GAMEBOY_BG); lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
String msg1 = "START GAME"; lcd.setTextSize(5);
int x_start = (lcd.width() - (msg1.length() * 6 * 5)) / 2;
lcd.setCursor(x_start, 120); lcd.print(msg1);
String msg2 = "Aperte A para jogar"; lcd.setTextSize(2);
```

```
int x_aperte = (lcd.width() - (msg2.length() * 6 * 2)) / 2;
lcd.setCursor(x_aperte, 240); lcd.print(msg2);
while (!botaoCliqueUnico(b_A)) delay(50);
}

// ---------------- SETUP ----------------
void setup() {
Serial.begin(115200);
pinMode(PBD, INPUT_PULLUP); pinMode(PBS, INPUT_PULLUP); pinMode(PBA,
INPUT_PULLUP); pinMode(PBB, INPUT_PULLUP);

EEPROM.begin(EEPROM_SIZE); carregarPlacar();

// --- NOVAS Matérias, Teorias e Quizzes ---

// Matéria 0: Bits e Bytes (3 Quizzes)
quizzes[0][0].pergunta1 = "1- O que e um 'bit'?"; quizzes[0][0].pergunta2 = "";
quizzes[0][0].opcoes[0] = " a) Uma unidade de 4 bits"; quizzes[0][0].opcoes[1] = " b) Uma
unidade de 8 bits"; quizzes[0][0].opcoes[2] = " c) A menor unidade lida";
quizzes[0][0].opcoes[3] = " d) Um conjunto de 16 bits"; quizzes[0][0].correta = 2; // c
quizzes[0][1].pergunta1 = "2- Quantos bits ha"; quizzes[0][1].pergunta2 = "em um byte?";
quizzes[0][1].opcoes[0] = " a) 2"; quizzes[0][1].opcoes[1] = " b) 4"; quizzes[0][1].opcoes[2]
= " c) 16"; quizzes[0][1].opcoes[3] = " d) 8"; quizzes[0][1].correta = 3; // d
quizzes[0][2].pergunta1 = "3- O que 'K' em '8KB'"; quizzes[0][2].pergunta2 = "representa?";
quizzes[0][2].opcoes[0] = " a) Kilo = 1000 bytes"; quizzes[0][2].opcoes[1] = " b) Kilo = 1024
bytes"; quizzes[0][2].opcoes[2] = " c) Kilo = 1000 bits"; quizzes[0][2].opcoes[3] = " d) Kilo
= 1024 bits"; quizzes[0][2].correta = 0; // a

// Matéria 0: Teoria (4 Pág)
teorias[0][0].linhas[0] = "Na eletronica digital,"; teorias[0][0].linhas[1] = "trabalha-se com o
sistema"; teorias[0][0].linhas[2] = "binario (0 e 1).";
teorias[0][0].linhas[3] = "1 = VERDADEIRO (Alto)"; teorias[0][0].linhas[4] = "0 = FALSO
(Baixo)"; teorias[0][0].linhas[5] = "";
teorias[0][0].linhas[6] = "A menor unidade que um"; teorias[0][0].linhas[7] = "processador le
e o 'bit'."; teorias[0][0].linhas[8] = "";
teorias[0][1].linhas[0] = "Bit: Pode assumir"; teorias[0][1].linhas[1] = "apenas 0 ou 1.";
teorias[0][1].linhas[2] = "";
teorias[0][1].linhas[3] = "Nibble: E um conjunto"; teorias[0][1].linhas[4] = "de 4 bits.";
teorias[0][1].linhas[5] = "";
teorias[0][1].linhas[6] = "Byte: E um conjunto"; teorias[0][1].linhas[7] = "de 8 bits (ou 2
nibbles)."; teorias[0][1].linhas[8] = "";
```

```
teorias[0][2].linhas[0] = "O byte e a unidade"; teorias[0][2].linhas[1] = "basica para
representar"; teorias[0][2].linhas[2] = "dados mais complexos,";
teorias[0][2].linhas[3] = "como letras ou numeros."; teorias[0][2].linhas[4] = "";
teorias[0][2].linhas[5] = "Cada bit dentro de um";
teorias[0][2].linhas[6] = "byte e uma parte que"; teorias[0][2].linhas[7] = "forma o todo.";
teorias[0][2].linhas[8] = "";
teorias[0][3].linhas[0] = "Voce ja viu '8KB' ou"; teorias[0][3].linhas[1] = "'16GB' em
arquivos."; teorias[0][3].linhas[2] = "B = Byte";
teorias[0][3].linhas[3] = ""; teorias[0][3].linhas[4] = "As letras antes (K, G)";
teorias[0][3].linhas[5] = "sao prefixos";
teorias[0][3].linhas[6] = "multiplicadores, como 'K'"; teorias[0][3].linhas[7] = "de 'Km' (1000
metros)."; teorias[0][3].linhas[8] = "Kilo (K) = 1000.";

// Matéria 1: Sistemas de Numeração (3 Quizzes)
quizzes[1][0].pergunta1 = "1- Qual e o valor do"; quizzes[1][0].pergunta2 = "decimal 25 em
binario?";
quizzes[1][0].opcoes[0] = " a) 10101"; quizzes[1][0].opcoes[1] = " b) 10011";
quizzes[1][0].opcoes[2] = " c) 11010"; quizzes[1][0].opcoes[3] = " d) 11001";
quizzes[1][0].correta = 3; // d
quizzes[1][1].pergunta1 = "2- Qual e o valor do"; quizzes[1][1].pergunta2 = "binario 101010
em decimal?";
quizzes[1][1].opcoes[0] = " a) 42"; quizzes[1][1].opcoes[1] = " b) 21";
quizzes[1][1].opcoes[2] = " c) 50"; quizzes[1][1].opcoes[3] = " d) 26"; quizzes[1][1].correta
= 0; // a
quizzes[1][2].pergunta1 = "3- Representacao binaria"; quizzes[1][2].pergunta2 = "do
hexadecimal B5?";
quizzes[1][2].opcoes[0] = " a) 11010101"; quizzes[1][2].opcoes[1] = " b) 181";
quizzes[1][2].opcoes[2] = " c) 10110101"; quizzes[1][2].opcoes[3] = " d) 1011101";
quizzes[1][2].correta = 2; // c

// Matéria 1: Teoria (4 Pág)
teorias[1][0].linhas[0] = "Sistema de numeracao e"; teorias[1][0].linhas[1] = "a forma de
organizar e"; teorias[1][0].linhas[2] = "representar quantidades.";
teorias[1][0].linhas[3] = ""; teorias[1][0].linhas[4] = "Decimal (Base 10):";
teorias[1][0].linhas[5] = "Usa 0-9. Pesos em pot. 10";
teorias[1][0].linhas[6] = "Binario (Base 2):"; teorias[1][0].linhas[7] = "Usa 0-1 (Bit). Pesos
em"; teorias[1][0].linhas[8] = "potencias de 2.";
teorias[1][1].linhas[0] = "Hexadecimal (Base 16):"; teorias[1][1].linhas[1] = "Usa 0-9 e
A(10) ate F(15)"; teorias[1][1].linhas[2] = "";
teorias[1][1].linhas[3] = "E usado para 'abreviar'"; teorias[1][1].linhas[4] = "o binario, pois 1
digito"; teorias[1][1].linhas[5] = "HEX = 4 bits (1 nibble).";
teorias[1][1].linhas[6] = ""; teorias[1][1].linhas[7] = ""; teorias[1][1].linhas[8] = "";
```

```
teorias[1][2].linhas[0] = "Conversao: DEC -> BIN"; teorias[1][2].linhas[1] = "Use divisoes sucessivas"; teorias[1][2].linhas[2] = "por 2, pegando o resto.";
teorias[1][2].linhas[3] = "Ex: Converter 13"; teorias[1][2].linhas[4] = "13 / 2 = 6, resto 1";
teorias[1][2].linhas[5] = " 6 / 2 = 3, resto 0";
teorias[1][2].linhas[6] = " 3 / 2 = 1, resto 1"; teorias[1][2].linhas[7] = " 1 / 2 = 0, resto 1";
teorias[1][2].linhas[8] = "Resultado (de baixo): 1101";
teorias[1][3].linhas[0] = "Conversao: BIN -> DEC"; teorias[1][3].linhas[1] = "Use a soma das potencias"; teorias[1][3].linhas[2] = "de 2.";
teorias[1][3].linhas[3] = "Ex: 1011 = (1*8)+(0*4)+"; teorias[1][3].linhas[4] = "(1*2)+(1*1) = 11"; teorias[1][3].linhas[5] = "";
teorias[1][3].linhas[6] = "Conversao: HEX <-> BIN"; teorias[1][3].linhas[7] = "Ex: B5 -> 1011 (B) 0101 (5)"; teorias[1][3].linhas[8] = "Resultado: 10110101";

// Matéria 2: Portas Lógicas (3 Quizzes)
quizzes[2][0].pergunta1 = "1- Porta AND."; quizzes[2][0].pergunta2 = "A=1, B=0. Saida?";
quizzes[2][0].opcoes[0] = " a) 0"; quizzes[2][0].opcoes[1] = " b) 1"; quizzes[2][0].opcoes[2] = " c) A"; quizzes[2][0].opcoes[3] = " d) Nao e possivel"; quizzes[2][0].correta = 0; // a
quizzes[2][1].pergunta1 = "2- Porta XOR."; quizzes[2][1].pergunta2 = "A=1, B=1. Saida?";
quizzes[2][1].opcoes[0] = " a) 0"; quizzes[2][1].opcoes[1] = " b) 1"; quizzes[2][1].opcoes[2] = " c) 2"; quizzes[2][1].opcoes[3] = " d) Nao e possivel"; quizzes[2][1].correta = 0; // a
quizzes[2][2].pergunta1 = "3- Qnts portas (AND,OR,"; quizzes[2][2].pergunta2 = "NOT) para fazer 1 XOR?";
quizzes[2][2].opcoes[0] = " a) 2"; quizzes[2][2].opcoes[1] = " b) 4"; quizzes[2][2].opcoes[2] = " c) 3"; quizzes[2][2].opcoes[3] = " d) 5"; quizzes[2][2].correta = 3; // d

// Matéria 2: Teoria (3 Pág)
teorias[2][0].linhas[0] = "Funcoes logicas sao o"; teorias[2][0].linhas[1] = "nucleo da elet. digital."; teorias[2][0].linhas[2] = "Circuitos = Portas Logicas";
teorias[2][0].linhas[3] = ""; teorias[2][0].linhas[4] = "Porta AND (E): (A.B)";
teorias[2][0].linhas[5] = "Saida 1 se TODAS = 1.";
teorias[2][0].linhas[6] = ""; teorias[2][0].linhas[7] = "Porta OR (OU): (A+B)";
teorias[2][0].linhas[8] = "Saida 1 se UMA = 1.";
teorias[2][1].linhas[0] = "Porta NOT (Inversora):"; teorias[2][1].linhas[1] = "Inverte o sinal (A')."; teorias[2][1].linhas[2] = "1 vira 0, 0 vira 1.";
teorias[2][1].linhas[3] = ""; teorias[2][1].linhas[4] = "Porta NAND: (AND + NOT)";
teorias[2][1].linhas[5] = "Saida 0 se TODAS = 1.";
teorias[2][1].linhas[6] = ""; teorias[2][1].linhas[7] = "Porta NOR: (OR + NOT)";
teorias[2][1].linhas[8] = "Saida 1 se TODAS = 0.";
teorias[2][2].linhas[0] = "Portas de Exclusao:"; teorias[2][2].linhas[1] = "";
teorias[2][2].linhas[2] = "Porta XOR:";
teorias[2][2].linhas[3] = "Saida 1 se entradas"; teorias[2][2].linhas[4] = "forem DIFERENTES."; teorias[2][2].linhas[5] = "";
```

```
teorias[2][2].linhas[6] = "Porta XNOR: (XOR + NOT)"; teorias[2][2].linhas[7] = "Saida 1 se entradas"; teorias[2][2].linhas[8] = "forem IGUAIS.";

// Matéria 3: Álgebra de Boole (3 Quizzes)
quizzes[3][0].pergunta1 = "1- Lei do Complemento:"; quizzes[3][0].pergunta2 = "Qual o resultado de A+A'?";
quizzes[3][0].opcoes[0] = " a) 0"; quizzes[3][0].opcoes[1] = " b) 1"; quizzes[3][0].opcoes[2] = " c) A"; quizzes[3][0].opcoes[3] = " d) A'"; quizzes[3][0].correta = 1; // b
quizzes[3][1].pergunta1 = "2- Qual e a Lei de"; quizzes[3][1].pergunta2 = "De Morgan correta?";
quizzes[3][1].opcoes[0] = " a) (A+B)' = A' + B'"; quizzes[3][1].opcoes[1] = " b) (A.B)' = A' + B'"; quizzes[3][1].opcoes[2] = " c) (A.B)' = A' . B'"; quizzes[3][1].opcoes[3] = " d) A + A' = 0"; quizzes[3][1].correta = 1; // b
quizzes[3][2].pergunta1 = "3- Simplifique a"; quizzes[3][2].pergunta2 = "expressao: A + (A . B)";
quizzes[3][2].opcoes[0] = " a) A"; quizzes[3][2].opcoes[1] = " b) B"; quizzes[3][2].opcoes[2] = " c) A . B"; quizzes[3][2].opcoes[3] = " d) A + B"; quizzes[3][2].correta = 0; // a

// Matéria 3: Teoria (4 Pág)
teorias[3][0].linhas[0] = "Algebra Booleana e a"; teorias[3][0].linhas[1] = "base da logica digital."; teorias[3][0].linhas[2] = "Usa 0 (Falso) e 1 (Verd.)";
teorias[3][0].linhas[3] = "LEIS BASICAS:"; teorias[3][0].linhas[4] = "Dominancia (Nulidade):"; teorias[3][0].linhas[5] = "A+1 = 1 | A.0 = 0";
teorias[3][0].linhas[6] = "Identidade:"; teorias[3][0].linhas[7] = "A+0 = A | A.1 = A";
teorias[3][0].linhas[8] = "Idempotencia: A+A=A | A.A=A";
teorias[3][1].linhas[0] = "LEIS 'NORMAIS':"; teorias[3][1].linhas[1] = "";
teorias[3][1].linhas[2] = "Comutativa (ordem):";
teorias[3][1].linhas[3] = "A + B = B + A"; teorias[3][1].linhas[4] = "A . B = B . A";
teorias[3][1].linhas[5] = "";
teorias[3][1].linhas[6] = "Associativa (grupo):"; teorias[3][1].linhas[7] = "A+(B+C) = (A+B)+C"; teorias[3][1].linhas[8] = "";
teorias[3][2].linhas[0] = "Distributiva (chuveiro):"; teorias[3][2].linhas[1] = "A.(B+C) = (A.B)+(A.C)"; teorias[3][2].linhas[2] = "";
teorias[3][2].linhas[3] = "LEIS 'MAGICAS':"; teorias[3][2].linhas[4] = "Complemento (Oposto):"; teorias[3][2].linhas[5] = "A + A' = 1 (Sempre V)";
teorias[3][2].linhas[6] = "A . A' = 0 (Sempre F)"; teorias[3][2].linhas[7] = "";
teorias[3][2].linhas[8] = "";
teorias[3][3].linhas[0] = "Absorcao (resumo):"; teorias[3][3].linhas[1] = "A + (A.B) = A";
teorias[3][3].linhas[2] = "A . (A+B) = A";
teorias[3][3].linhas[3] = ""; teorias[3][3].linhas[4] = "Leis de De Morgan:";
teorias[3][3].linhas[5] = "(A + B)' = A' . B'";
```

```
teorias[3][3].linhas[6] = "(Nega OU vira AND negado)"; teorias[3][3].linhas[7] = "(A . B)' =
A' + B'"; teorias[3][3].linhas[8] = "(Nega AND vira OR negado)";

// Matéria 4: Expressões Lógicas (3 Quizzes)
quizzes[4][0].pergunta1 = "1- Qual a expressao"; quizzes[4][0].pergunta2 = "correta para as
portas?";
quizzes[4][0].opcoes[0] = " a) OR=A.B, AND=A+B"; quizzes[4][0].opcoes[1] = " b)
OR=A+B, AND=A.B, NOT=A'"; quizzes[4][0].opcoes[2] = " c) OR=A', AND=A+B";
quizzes[4][0].opcoes[3] = " d) OR=A.B, NOT=A+B"; quizzes[4][0].correta = 1; // b
quizzes[4][1].pergunta1 = "2- x = A.B + C e"; quizzes[4][1].pergunta2 = "x = (A+B).C sao:";
quizzes[4][1].opcoes[0] = " a) Diferentes (ordem)"; quizzes[4][1].opcoes[1] = " b) Iguais";
quizzes[4][1].opcoes[2] = " c) Sem C"; quizzes[4][1].opcoes[3] = " d) Sem A";
quizzes[4][1].correta = 0; // a
quizzes[4][2].pergunta1 = "3- Circuito para"; quizzes[4][2].pergunta2 = "x = (A+B)' . C ?";
quizzes[4][2].opcoes[0] = " a) AND->NOT->OR com C"; quizzes[4][2].opcoes[1] = " b)
OR->NOT->AND com C"; quizzes[4][2].opcoes[2] = " c) NOT A,NOT B->AND->OR";
quizzes[4][2].opcoes[3] = " d) OR->AND com C"; quizzes[4][2].correta = 1; // b

// Matéria 4: Teoria (4 Pág)
teorias[4][0].linhas[0] = "Circuito logico e um"; teorias[4][0].linhas[1] = "desenho que
representa"; teorias[4][0].linhas[2] = "uma expressao Booleana.";
teorias[4][0].linhas[3] = "Porta AND (E): (A.B)"; teorias[4][0].linhas[4] = "Simbolo: Letra
'D'."; teorias[4][0].linhas[5] = "Saida 1 se TODAS = 1.";
teorias[4][0].linhas[6] = "Porta OR (OU): (A+B)"; teorias[4][0].linhas[7] = "Simbolo:
'Meia-lua'."; teorias[4][0].linhas[8] = "Saida 1 se UMA = 1.";
teorias[4][1].linhas[0] = "Porta NOT (NAO): (A')"; teorias[4][1].linhas[1] = "Simbolo:
Triangulo"; teorias[4][1].linhas[2] = "com bolinha na ponta.";
teorias[4][1].linhas[3] = "Funcao: Inverte o sinal."; teorias[4][1].linhas[4] = "1 vira 0, 0 vira
1."; teorias[4][1].linhas[5] = "";
teorias[4][1].linhas[6] = "REGRA DE OURO:"; teorias[4][1].linhas[7] = "Comece da SAIDA
(final)"; teorias[4][1].linhas[8] = "e volte ate as ENTRADAS.";
teorias[4][2].linhas[0] = "Exemplo de Circuito:"; teorias[4][2].linhas[1] = "Entradas: A, B,
C."; teorias[4][2].linhas[2] = "1. A e B entram em AND.";
teorias[4][2].linhas[3] = "2. C entra em NOT."; teorias[4][2].linhas[4] = "3. Saidas de (1) e
(2)"; teorias[4][2].linhas[5] = "entram em OR.";
teorias[4][2].linhas[6] = "4. Saida de OR e S."; teorias[4][2].linhas[7] = "";
teorias[4][2].linhas[8] = "[Circuito: (A.B) + C']";
teorias[4][3].linhas[0] = "Montando de tras p/ frente:"; teorias[4][3].linhas[1] = "Passo 1:
Ultima porta"; teorias[4][3].linhas[2] = "e OR. S = (Algo) + (Algo).";
teorias[4][3].linhas[3] = "Passo 2: O primeiro 'Algo'"; teorias[4][3].linhas[4] = "vem de uma
AND com A e B."; teorias[4][3].linhas[5] = "S = (A.B) + (Algo).";
```

```
teorias[4][3].linhas[6] = "Passo 3: O segundo 'Algo'"; teorias[4][3].linhas[7] = "vem de uma NOT com C."; teorias[4][3].linhas[8] = "S = (A.B) + C'";

// Matéria 5: Aritmética Binária (3 Quizzes)
quizzes[5][0].pergunta1 = "1- (Soma) Qual o"; quizzes[5][0].pergunta2 = "resultado de 0101 + 0011?";
quizzes[5][0].opcoes[0] = " a) 0111"; quizzes[5][0].opcoes[1] = " b) 1000";
quizzes[5][0].opcoes[2] = " c) 0100"; quizzes[5][0].opcoes[3] = " d) 1001";
quizzes[5][0].correta = 1; // b
quizzes[5][1].pergunta1 = "2- (Sub) Resultado de"; quizzes[5][1].pergunta2 = "100 - 001?";
quizzes[5][1].opcoes[0] = " a) 010"; quizzes[5][1].opcoes[1] = " b) 011";
quizzes[5][1].opcoes[2] = " c) 101"; quizzes[5][1].opcoes[3] = " d) 001";
quizzes[5][1].correta = 1; // b
quizzes[5][2].pergunta1 = "3- (Mult) Resultado de"; quizzes[5][2].pergunta2 = "11 x 10?";
quizzes[5][2].opcoes[0] = " a) 101"; quizzes[5][2].opcoes[1] = " b) 111";
quizzes[5][2].opcoes[2] = " c) 110"; quizzes[5][2].opcoes[3] = " d) 011";
quizzes[5][2].correta = 2; // c

// Matéria 5: Teoria (4 Pág)
teorias[5][0].linhas[0] = "Aritmetica binaria usa"; teorias[5][0].linhas[1] = "o sistema binario (0, 1)"; teorias[5][0].linhas[2] = "para resolver operacoes.";
teorias[5][0].linhas[3] = "Regras semelhantes ao"; teorias[5][0].linhas[4] = "decimal, mas com base 2."; teorias[5][0].linhas[5] = "";
teorias[5][0].linhas[6] = "Permite ao computador"; teorias[5][0].linhas[7] = "manipular calculos."; teorias[5][0].linhas[8] = "";
teorias[5][1].linhas[0] = "1. Soma Binaria"; teorias[5][1].linhas[1] = "0 + 0 = 0";
teorias[5][1].linhas[2] = "0 + 1 = 1";
teorias[5][1].linhas[3] = "1 + 0 = 1"; teorias[5][1].linhas[4] = "1 + 1 = 10 ('vai um' 1)";
teorias[5][1].linhas[5] = "";
teorias[5][1].linhas[6] = "Ex: 1011 (11) + 1101 (13)"; teorias[5][1].linhas[7] = "Resultado: 11000 (24)"; teorias[5][1].linhas[8] = "";
teorias[5][2].linhas[0] = "2. Subtracao Binaria"; teorias[5][2].linhas[1] = "0 - 0 = 0";
teorias[5][2].linhas[2] = "1 - 0 = 1";
teorias[5][2].linhas[3] = "1 - 1 = 0"; teorias[5][2].linhas[4] = "0 - 1 = 1 (empresta 1)";
teorias[5][2].linhas[5] = "";
teorias[5][2].linhas[6] = "Ex: 10010 (18) - 1101 (13)"; teorias[5][2].linhas[7] = "Resultado: 0101 (5)"; teorias[5][2].linhas[8] = "";
teorias[5][3].linhas[0] = "3. Multiplicacao Binaria"; teorias[5][3].linhas[1] = "0x0=0, 0x1=0, 1x0=0, 1x1=1"; teorias[5][3].linhas[2] = "Ex: 101 (5) x 11 (3) = 1111 (15)";
teorias[5][3].linhas[3] = ""; teorias[5][3].linhas[4] = "4. Divisao Binaria";
teorias[5][3].linhas[5] = "Semelhante a divisao longa";
```

```
teorias[5][3].linhas[6] = "decimal."; teorias[5][3].linhas[7] = "Ex: 1100 (12) / 10 (2) = 110
(6)"; teorias[5][3].linhas[8] = "";
// --- FIM DAS MATÉRIAS ---


lcd.init();
// --- MUDANÇA: Rotação da tela invertida ---
lcd.setRotation(3);
mostrarTelaInsertCoin();
atualizarMenu();
}

// ---------------- LOOP principal ----------------
void loop() {
bool desce = botaoCliqueUnico(b_nav1); bool sobe = botaoCliqueUnico(b_nav2); bool
confirmar = botaoCliqueUnico(b_A); bool voltar = botaoCliqueUnico(b_B);
int fase = getFaseAtual(); int total = totalItens[fase];
if ((desce || sobe) && !trava1 && !trava2 && !telaTeoria) {
   int dir = desce ? 1 : -1; itemSelecionado = (itemSelecionado + dir + total) % total;
   if (telaMenu) atualizarMenu(); else if (telaMaterias) menuMateriasTela(); else if (telaQuiz)
atualizarQuiz(); else if (telaContinuar) mostrarTelaContinuar();
   if (desce) trava1 = true; if (sobe) trava2 = true;
}
if (confirmar) processarConfirmar(); if (voltar) processarVoltar();
if (trava1 && digitalRead(b_nav1.pino) == HIGH) trava1 = false; if (trava2 &&
digitalRead(b_nav2.pino) == HIGH) trava2 = false;
delay(8);
}
int getFaseAtual() { if (telaMenu) return 0; if (telaMaterias) return 1; if (telaQuiz) return 2; if
(telaTeoria) return 3; if (telaContinuar) return 4; return 0; }

void processarConfirmar() {
if (telaMenu) {
   if (itemSelecionado == 0) { modoNovoJogo = true; mudarTela(mostrarTelaContinuar,
false, false, false, false, true); }
   else if (itemSelecionado == 1) { if (verificaSavesExistem()) { modoNovoJogo = false;
mudarTela(mostrarTelaContinuar, false, false, false, false, true); } }
   else if (itemSelecionado == 2) mostrarPlacar();
   else if (itemSelecionado == 3) { resetarEEPROM(); atualizarMenu(); }
}
else if (telaContinuar) {
   saveSlotAtual = itemSelecionado;
```

```
    if (modoNovoJogo) {
        entrarNomeJogador(); salvarScoreSlot(saveSlotAtual, 0);
        EEPROM.put(getAddrQuizFlags(saveSlotAtual), (uint32_t)0);
        for(int m = 0; m < MAX_MATERIAS; m++) salvarProgresso(m, 0);
        scoreAtual = 0; mudarTela(menuMateriasTela, false, true, false, false, false);
    } else {
        String nome = lerNomeSlot(saveSlotAtual);
        if (nome.length() > 0) { nomeJogadorAtual = nome; scoreAtual =
lerScoreSlot(saveSlotAtual); mudarTela(menuMateriasTela, false, true, false, false, false); }
    }
}
else if (telaMaterias) {
    int inicio = paginaMaterias * materiasPorPagina; int fim = min(inicio + materiasPorPagina,
totalMaterias);
    if (itemSelecionado < fim - inicio) { materiaAtual = inicio + itemSelecionado;
paginaTeoria = 0; mudarTela(mostrarTeoria, false, false, false, true, false); }
    else { int opcaoExtra = fim - inicio; if (paginaMaterias > 0 && itemSelecionado ==
opcaoExtra) { paginaMaterias--; itemSelecionado = 0; menuMateriasTela(); } else if (fim <
totalMaterias && itemSelecionado == opcaoExtra) { paginaMaterias++; itemSelecionado =
0; menuMateriasTela(); } }
}
else if (telaTeoria) {
    int numPaginas = numPaginasPorMateria[materiaAtual];
    if (paginaTeoria < numPaginas - 1) { paginaTeoria++; mostrarTeoria(); }
    else { paginaQuiz = 0; acertosQuiz = 0; materiaAtualQuiz = materiaAtual;
mudarTela(mostrarQuiz, false, false, true, false, false); }
}
else if (telaQuiz) {
    int m = constrain(materiaAtualQuiz, 0, MAX_MATERIAS - 1);
    int q = constrain(paginaQuiz, 0, QUIZZES_POR_MATERIA - 1);

    if (itemSelecionado == quizzes[m][q].correta) {
        acertosQuiz++;
        bool points_claimed = isQuizPointClaimed(m, q);
        if (!points_claimed) { scoreAtual += 10; claimQuizPoint(m, q);
mostrarResultado("Acertou!!", "+10 pts"); }
        else { mostrarResultado("Acertou!!", "Correto!"); }
    } else { mostrarResultado("Errou!!", "Incorreto!"); }

    paginaQuiz++;
    if (paginaQuiz < QUIZZES_POR_MATERIA) { itemSelecionado = 0; mostrarQuiz(); }
    else {
```

```
      if (acertosQuiz > estrelasMateria(materiaAtualQuiz)) {
salvarProgresso(materiaAtualQuiz, acertosQuiz); }
      salvarScoreSlot(saveSlotAtual, scoreAtual); atualizarPlacar(nomeJogadorAtual,
scoreAtual);

      if (acertosQuiz == QUIZZES_POR_MATERIA) { tocarVitoria(); }

      paginaQuiz = 0; acertosQuiz = 0; mudarTela(menuMateriasTela, false, true, false, false,
false);
   }
}
}

void processarVoltar() {
if (telaMaterias) { salvarScoreSlot(saveSlotAtual, scoreAtual);
atualizarPlacar(nomeJogadorAtual, scoreAtual); mudarTela(atualizarMenu, true, false, false,
false, false); }
else if (telaTeoria && paginaTeoria > 0) { paginaTeoria--; mostrarTeoria(); }
else if (telaTeoria || telaQuiz) mudarTela(menuMateriasTela, false, true, false, false, false);
else if (telaContinuar) mudarTela(atualizarMenu, true, false, false, false, false);
}

void mudarTela(void (*func)(), bool menu, bool materias, bool quiz, bool teoria, bool
continuar) {
telaMenu = menu; telaMaterias = materias; telaQuiz = quiz; telaTeoria = teoria; telaContinuar
= continuar; itemSelecionado = 0;
if (!telaMenu) lcd.fillScreen(GAMEBOY_BG);
func();
}
void mostrarResultado(String msg, String msg2) {
lcd.fillScreen(GAMEBOY_BG); lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
lcd.setTextSize(4); int x_msg = (lcd.width() - (msg.length() * 6 * 4)) / 2;
lcd.setCursor(x_msg, 140); lcd.print(msg);
lcd.setTextSize(2); int x_msg2 = (lcd.width() - (msg2.length() * 6 * 2)) / 2;
lcd.setCursor(x_msg2, 200); lcd.print(msg2);
delay(900);
}
void atualizarMenu() {
   lcd.fillScreen(GAMEBOY_BG); lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
   lcd.setTextSize(4); lcd.setCursor(40, 20); lcd.print(F("Eletroboy"));
   lcd.setTextSize(2); lcd.setCursor(280, 20); lcd.print(F("o video-game"));
```

```
  lcd.setCursor(320, 40); lcd.print(F("da")); lcd.setCursor(285, 60);
lcd.print(F("eletronica"));
  int y[] = {130, 170, 210, 250}; lcd.setTextSize(3); bool savesExistem =
verificaSavesExistem();
  desenharItem(menuPrincipal[0], 0, 40, y[0]);
  if (itemSelecionado == 1) lcd.setTextColor(GAMEBOY_BG, GAMEBOY_TEXT); else if
(!savesExistem) lcd.setTextColor(GAMEBOY_GRAY, GAMEBOY_BG); else
lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
  lcd.setCursor(40, y[1]); lcd.print(menuPrincipal[1]);
  desenharItem(menuPrincipal[2], 2, 40, y[2]); desenharItem(menuPrincipal[3], 3, 40, y[3]);
}
void mostrarTelaContinuar() {
  lcd.fillScreen(GAMEBOY_BG); lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
lcd.setTextSize(3);
  if (modoNovoJogo) { lcd.setCursor(40, 30); lcd.print(F("Escolha um slot:")); }
  else { lcd.setCursor(40, 30); lcd.print(F("Escolha um jogo:")); }
  int y = 100; lcd.setTextSize(3);
  for (int i = 0; i < MAX_SAVE_SLOTS; i++) {
    String nome = lerNomeSlot(i); int estrelas = getEstrelasTotaisSlot(i); int score =
lerScoreSlot(i); String linha;
    if (nome == "" && !modoNovoJogo) {
      if (itemSelecionado == i) lcd.setTextColor(GAMEBOY_BG, GAMEBOY_GRAY);
else lcd.setTextColor(GAMEBOY_GRAY, GAMEBOY_BG);
      linha = "[Slot " + String(i+1) + " Vazio]";
    } else if (nome == "") {
      if (itemSelecionado == i) lcd.setTextColor(GAMEBOY_BG, GAMEBOY_TEXT);
else lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
      linha = "[Slot " + String(i+1) + " Vazio]";
    } else {
      if (itemSelecionado == i) lcd.setTextColor(GAMEBOY_BG, GAMEBOY_TEXT);
else lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
      linha = nome + " " + String(score) + "p " + String(estrelas) + "*";
    }
    lcd.setCursor(40, y); lcd.print(linha); y += 60;
  }
}
void menuMateriasTela() {
lcd.fillScreen(GAMEBOY_BG); lcd.setTextSize(3);
int inicio = paginaMaterias * materiasPorPagina; int fim = min(inicio + materiasPorPagina,
totalMaterias);
int y = 40;
int x[] = {380, 440, 380, 400, 440};
```

```
for (int i = inicio; i < fim; i++) {
desenharItem(menuMaterias[i], i - inicio, 40, y);
int estrelas = estrelasMateria(i);
if (estrelas > 0) {
  lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
  lcd.setCursor(x[i % materiasPorPagina], y);
  for (int s = 0; s < estrelas; s++) lcd.print("*");
}
y += 40;
}
int opcaoExtra = fim - inicio;
if (paginaMaterias > 0) { desenharItem("<< Voltar", opcaoExtra, 40, y); y += 40;
opcaoExtra++; }
if (fim < totalMaterias) desenharItem("Proximo >>", opcaoExtra, 40, y);
}

void mostrarTeoria() {
lcd.fillScreen(GAMEBOY_BG);
lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
lcd.setTextSize(2);
PaginaTeoria pagina = teorias[materiaAtual][paginaTeoria];
int y = 40;
for (int i = 0; i < 9; i++) {
if (pagina.linhas[i].length() > 0) {
lcd.setCursor(30, y);
lcd.print(pagina.linhas[i]);
y += 30;
}
}
lcd.setTextSize(1);
lcd.setCursor(320, 300);
lcd.print("Pagina " + String(paginaTeoria + 1) + "/" +
String(numPaginasPorMateria[materiaAtual]));
}

void atualizarQuiz() {
int y[] = {110,160,210,260};
lcd.setTextSize(3);
int m = constrain(materiaAtualQuiz, 0, MAX_MATERIAS - 1);
int q = constrain(paginaQuiz, 0, QUIZZES_POR_MATERIA - 1);
for (int i = 0; i < OPCOES_POR_QUIZ; i++) desenharItem(quizzes[m][q].opcoes[i], i, 40,
y[i]);
```

```
}

void mostrarQuiz() {
lcd.fillScreen(GAMEBOY_BG);
lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
lcd.setTextSize(3);
int m = constrain(materiaAtualQuiz, 0, MAX_MATERIAS - 1);
int q = constrain(paginaQuiz, 0, QUIZZES_POR_MATERIA - 1);
lcd.setCursor(40, 20);
lcd.print(quizzes[m][q].pergunta1);
lcd.setCursor(40, 50);
lcd.print(quizzes[m][q].pergunta2);
atualizarQuiz();
}

void desenharItem(String texto, int indice, int x, int y) {
if (indice == itemSelecionado) lcd.setTextColor(GAMEBOY_BG, GAMEBOY_TEXT); else
lcd.setTextColor(GAMEBOY_TEXT, GAMEBOY_BG);
lcd.setCursor(x, y); lcd.print(texto);
}
```

Anexo B - Texto utilizado para matérias e questões:

Conceito de Bit, Nibble e Byte:

O campo da eletronica digital se apoia fundamentalmente no sistema de base binario, que utiliza apenas os digitos "0" e "1". Essa base e regida pela Algebra Booleana, onde o estado VERDADEIRO ou nivel logico ALTO e representado por "1", e o estado FALSO ou nivel logico BAIXO e simbolizado por "0".

A menor unidade de informacao que um processador consegue ler e o bit. Cada bit so pode assumir um desses dois valores possiveis (0 ou 1), representando um sinal desligado ou ligado. Ao agrupar quatro desses bits, forma-se um conjunto maior chamado nibble. A unidade de informacao mais comum e o byte, que e o agrupamento de oito bits (o equivalente a dois nibbles). Em essencia, o byte e a unidade composta por oito fracoes de informacao binaria (bits) que, juntas, permitem representar dados mais complexos nos sistemas digitais.

Ao ver tamanhos de arquivos ou velocidades de conexao (como "8KB" ou "16GB"), o "B" e a abreviacao de Byte. As letras que antecedem o "B" sao prefixos multiplicadores. No entanto, na computacao, esses prefixos nao seguem o padrao

decimal exato do dia a dia (onde "K" significa 1.000). Como os computadores operam na base binaria, eles organizam a memoria em potencias de 2, que estao sempre dobrando (2, 4, 8, 16, ate 1.024). Por convencao tecnica, o valor de 1.024, que e o resultado de 2^ (10) e o numero mais proximo de 1.000 que o computador "entende" naturalmente, foi adotado. Por isso, 1 Kilobyte (KB) foi definido como 1.024 bytes.

Questão 1: Qual é a menor unidade de informação que um processador consegue reconhecer, e quais são os dois únicos valores que ela pode assumir?

a) O Nibble, assumindo apenas os valores ALTO e BAIXO.

b) O Byte, assumindo valores de 0 a 7.

c) O Bit, assumindo apenas os valores "0" ou "1" .

d) O Kilobyte, assumindo os valores 1.000 ou 1.024.

Questão 2: Quantos bits há em um byte?

a) 2

b) 4

c) 16

d) 8

Questão 3: O que representa a letra "K" antes do "B" em "8KB"?

a) Kilobyte, que equivale a 1000 bytes

b) Kilobyte, que equivale a 1024 bytes

c) Kilo, que equivale a 1000 bits

d) Kilobit, que equivale a 1000 bits

Sistemas de Numeração:

A base de toda a eletrônica digital é o **sistema binário**, que utiliza apenas os dígitos **0** e **1**, cada um chamado de **bit**. Semelhante ao sistema decimal, o binário é posicional, mas os valores das posições dobram a cada passo: 1, 2, 4, 8, etc.

Existe também o **sistema hexadecimal** (base 16, com 0-9 e A-F), que serve como um **atalho prático** para o binário. Sua grande utilidade é que **cada dígito hexadecimal representa exatamente 4 bits** (um nibble), compactando longas sequências binárias.

Regras de Conversão Essenciais:

- **Decimal para Binário:** Usa-se o método das **divisões sucessivas por 2**. O número binário é formado lendo-se os restos na ordem inversa (de baixo para cima).
- **Binário para Decimal:** Soma-se o valor posicional das casas (1, 2, 4, 8...) **apenas** onde o bit é 1; os bits 0 são ignorados.
- **Binário para Hexadecimal:** É uma **tradução direta**. Para converter binário, separamos o número em grupos de 4 bits da direita para a esquerda e traduzimos cada grupo para o dígito hexadecimal correspondente. Se necessário, completamos com zeros à esquerda.

Questao 1: Quais dígitos o sistema binário utiliza?

A) 0–9

B) A–F

C) 2 e 3

D) 0 e 1

Questao 2: Cada dígito hexadecimal representa quantos bits?

A) 2 bits

B) 8 bits

C) 1 bit

D) 4 bits

Questao 3: No método de conversão de binário para decimal, o que deve ser somado?

A) Apenas os bits 0

B) Todos os bits, 0 e 1

C) O valor das posições onde o bit é 0

D) Os valores posicionais apenas onde o bit é 1

ÁLGEBRA DE BOOLE:

A Álgebra Booleana, desenvolvida por George Boole, é a disciplina matemática que fundamenta a lógica digital. Diferente da álgebra convencional, ela opera exclusivamente com dois valores, ou estados: 0 (representando Falso, ou nível lógico baixo) e 1 (representando Verdadeiro, ou nível lógico alto).

Para operar com estes valores, definem-se três operadores lógicos fundamentais. O operador + (adição lógica) representa a operação OU (OR). O operador · (multiplicação lógica) representa a operação E (AND). Por fim, o apóstrofo (') ou a barra superior (overbar) representa a operação NÃO (NOT), ou inversão. As variáveis A e B são usadas para representar qualquer valor booleano (0 ou 1).

A partir destes operadores, derivam-se os postulados e leis fundamentais. A Lei da Identidade estabelece que A + 0 = A (A OU Falso = A) e A · 1 = A (A E Verdadeiro = A). Em ambos os casos, a identidade da variável A é preservada. Inversamente, a Lei da Dominância (ou Anulação) demonstra como um valor pode dominar a expressão: A + 1 = 1 (qualquer valor OU Verdadeiro é sempre Verdadeiro) e A · 0 = 0 (qualquer valor E Falso é sempre Falso).

Existem leis que diferem da álgebra convencional. A Lei da Idempotência (A + A = A e A · A = A) estabelece que operar uma variável com ela mesma resulta na própria variável. A Lei do Complemento é crucial: A + A' = 1 significa que uma variável OU seu inverso é sempre Verdadeiro (ex: 'chovendo' OU 'NÃO chovendo'). De forma análoga, A · A' = 0 significa que uma variável E seu inverso é sempre Falso (é impossível 'chover' E 'NÃO chover' simultaneamente).

Esta álgebra também obedece a propriedades familiares, como a Lei Comutativa (A + B = B + A) e a Lei Associativa, onde o agrupamento não altera o resultado. Contudo, as Leis de De Morgan são as mais poderosas para a simplificação de expressões e circuitos. Elas definem como inverter uma operação composta: a negação de uma operação OU é a operação E das negações individuais, formalmente (A + B)' = A' · B'. Por exemplo, negar '(praia OU parque)' é o mesmo que afirmar '(NÃO praia) E (NÃO parque)'.

O inverso é igualmente verdadeiro: a negação de uma operação E é a operação OU das negações, formalmente (A · B)' = A' + B'. Em um exemplo prático de circuito: se 'O alarme NÃO é disparado se (a porta está fechada E o sensor está ativo)', a lógica inversa (quando ele TOCA) é 'se (a porta NÃO está fechada OU o sensor NÃO está ativo)'. Estas leis são essenciais para a otimização e o design de circuitos lógicos.

Questões – Álgebra de Boole

Questão 1: De acordo com a Lei da Complementaridade, qual é o resultado da expressão $A + \bar{A}$?

a) 0
b) 1
c) A
d) $\bar{A}$

Questão 2: Qual das expressões abaixo representa corretamente a Lei de De Morgan?

a) $\overline{(A + B)} = \bar{A} + \bar{B}$
b) $\overline{(A \cdot B)} = \bar{A} + \bar{B}$
c) $\overline{(A \cdot B)} = \bar{A} \cdot \bar{B}$
d) $A + \bar{A} = 0$

Questão 3: Simplifique a expressão lógica:
$A + (A \cdot B)$

a) A
b) B
c) $A \cdot B$
d) $A + B$

Questão 4: A expressão $(A \cdot 0) + (A \cdot 1)$ pode ser simplificada para:

a) 0
b) 1
c) A
d) $\bar{A}$

Questão 5: A operação $(A + B) \cdot (A + \bar{A})$ é equivalente a:

a) A
b) B
c) $A + B$
d) $A \cdot B$

Funções lógicas e portas lógicas:

As funções lógicas são o núcleo da eletrônica digital, determinando como os sinais binários (0 e 1) devem ser processados. Os circuitos que executam essas operações são chamados de portas lógicas, e são os blocos construtores de dispositivos complexos, como a Unidade

Lógica Aritmética (ULA) de um processador. Cada porta recebe uma ou mais entradas e produz uma única saída.

A Porta AND (E) está diretamente ligada à multiplicação lógica (A·B) que vimos na Álgebra Booleana. Ela só produz uma saída "1" (Verdadeiro) se todas as suas entradas forem 1. Se qualquer uma das entradas for 0, a saída será 0.

A Porta OR (OU) segue a soma lógica (A+B). Ela produz uma saída "1" se pelo menos uma de suas entradas for 1. A única forma de uma porta OR ter uma saída 0 é se todas as suas entradas forem 0.

A Porta NOT (NÃO), também conhecida como inversora, é a mais simples. Ela tem apenas uma entrada e simplesmente inverte o sinal: se entra 1, sai 0; se entra 0, sai 1. Ela representa a operação de negação (o 'apóstrofo' ou a barra em cima da letra na álgebra).

Com base nessas três, criam-se as portas derivadas. A Porta NAND é a contração de "NOT-AND", ou seja, é o inverso exato da porta AND. Sua saída será 0 somente se todas as entradas forem 1 (em todos os outros casos, a saída é 1). É o mesmo que pegar a saída de uma porta AND e ligá-la em uma porta NOT.

Da mesma forma, a Porta NOR significa "NOT-OR". Ela é o inverso da porta OR. Sua saída será 1 somente se todas as suas entradas forem 0. Em qualquer outro caso, a saída será 0.

Por fim, existem as portas de "exclusão". A Porta XOR (OU Exclusivo) é muito útil para comparações. Ela só retorna 1 se as suas entradas forem diferentes (uma 0 e a outra 1). Se as entradas forem iguais (ambas 0 ou ambas 1), a saída será 0.

A Porta XNOR (NÃO-OU Exclusivo) é o inverso da XOR. Ela faz o oposto: retorna 1 somente se as entradas forem iguais (ambas 0 ou ambas 1). Se as entradas forem diferentes, a saída será 0. Essas portas são essenciais para construir circuitos que realizam cálculos matemáticos.

Questão 1: Considere a porta lógica **AND**. Qual será a saída se as entradas forem:

A = 1 ; B = 0

a) 0

b) 1

c) A

d) Não é possível determinar

Questão 2: Uma porta lógica **XOR** possui as entradas A = 1 e B = 1. Qual será o valor da saída?

a) 0

b) 1

c) 2

d) Não é possível determinar

Questão 3: Você tem à disposição apenas portas AND, OR e NOT,e apenas pode usar duas entradas. Quantas portas seriam necessárias para montar uma porta XOR com essas portas? (Dica: uma mesma entrada pode ser utilizada em duas portas diferentes.)

a) 2

b) 4

c) 3

d) 5

Expressões lógicas a partir de um circuito:

A partir das portas lógicas descritas, representações algébricas podem ser feitas. Tais representações foram mostradas também na mesma matéria de portas lógicas. As expressões booleanas (ou lógicas) são uma maneira compacta de apresentar o funcionamento das portas lógicas, explicando o que o circuito digital faz para gerar uma saída com uma fórmula

algébrica. As expressões A·B, A+B e Ā são as portas AND, OR e inversora, respectivamente, e suas derivações também tem certas equações algébricas que as representam. Isso será o núcleo principal deste assunto: olhar para um circuito e identificar de que forma ele será representado algebricamente. As únicas coisas necessárias para isso ser feito é seguir as entradas do circuito lógico e saber usar a álgebra de Boole, conteúdo já explicado anteriormente.

Com isso em mente, podemos criar expressões booleanas a partir de circuitos reais com portas lógicas para determinar qual será sua saída.

Vamos imaginar dois exemplos simples:

Exemplo 1 – A porta AND vem antes da OR

Primeiro, as entradas A e B passam por uma porta AND. Essa porta só "liga" (dá 1) quando A e B estão ligadas ao mesmo tempo, o que podemos escrever como $A \cdot B$.

Em seguida, o resultado dessa operação ($A \cdot B$) vai junto com a entrada C para uma porta OR. A porta OR combina esses dois sinais, gerando a soma lógica entre eles. Assim, a saída final do circuito, que chamaremos de x, é dada por:

$x = A \cdot B + C$

Em outras palavras, o circuito só dá saída 1 se (A e B forem 1) ou se C for 1.

Exemplo 2 – A porta OR vem antes da AND

Agora, imagine outro circuito em que as entradas A e B passam primeiro por uma porta OR. Ela gera uma saída 1 sempre que A ou B forem 1, o que se escreve como $A + B$.

Depois disso, o resultado dessa operação ($A + B$) é conectado a uma porta AND junto com a entrada C. A porta AND exige que C também seja 1 para que o resultado final seja 1. Assim, a expressão da saída é:

$x = (A + B) \cdot C$

Aqui, o circuito só gera saída 1 quando C está ligada e pelo menos uma das entradas A ou B também está.

O cuidado principal é respeitar a ordem das operações. Primeiro resolvemos o que estiver dentro de parênteses, em seguida as operações de multiplicação lógica (AND) e por último as somas lógicas (OR). O desenho do circuito é feito seguindo essa ordem, sempre indo das operações internas para as externas até chegar à saída final.

Por exemplo, se temos a expressão $x = A \cdot B + C$, isso significa que A e B passam primeiro por uma porta AND. O resultado dessa operação segue para uma porta OR junto com a entrada C, e a saída dessa porta é o valor final de x.

No caso da expressão $x = (A + B) \cdot C$, o funcionamento muda. Agora, A e B passam primeiro por uma porta OR. O resultado dessa soma lógica é levado até uma porta AND, que também recebe a entrada C, e a saída dessa porta é o valor final de x.

Um caso mais elaborado pode ser observado na expressão $x = \overline{(A + B)} \cdot C$. Nesse exemplo, A e B passam por uma porta OR, em seguida o resultado é invertido por uma porta NOT, e somente depois esse valor é conectado a uma porta AND junto com a entrada C. A saída da AND corresponde ao resultado final.

Assim, cada parte da expressão booleana se transforma diretamente em um bloco do circuito. Basta seguir a ordem correta das operações e representar cada símbolo por sua porta equivalente!

Questão 1:

Qual das alternativas corresponde corretamente às portas lógicas básicas e suas expressões booleanas?

a) A + B representa a porta AND, A · B representa a porta OR e $\bar{A}$ representa a porta NOT.

b) A + B representa a porta OR, A · B representa a porta AND e $\bar{A}$ representa a porta NOT.

c) A + B representa a porta AND, A · B representa a porta NOT e $\bar{A}$ representa a porta OR.

d) A + B representa a porta NOT, A · B representa a porta OR e $\bar{A}$ representa a porta AND.

Questão 2 (nível intermediário):

Por que as expressões **$x = A \cdot B + C$** e **$x = (A + B) \cdot C$** representam circuitos diferentes?

a) Porque no primeiro a porta AND vem antes da OR, e no segundo a OR vem antes da AND.

b) Porque no primeiro circuito não existe a variável C.

c) Porque no segundo circuito não existe a variável A.

d) Ambas são idênticas, apenas escritas de forma diferente.

Qual alternativa descreve corretamente o circuito da expressão $\mathbf{x = (A + B)^{-} \cdot C}$?

a) Uma porta AND entre A e B, seguida de uma porta NOT e depois uma OR com C.

b) Uma porta OR entre A e B, o resultado passa por uma NOT, e em seguida é ligado a uma AND com C.

c) Duas portas NOT, uma em A e outra em B, seguidas por uma AND e uma OR com C.

d) Uma porta OR entre A e B diretamente conectada a uma porta AND com C, sem inversão.

Aritmética binária:

Nos textos anteriores, exploramos como os computadores usam a lógica para tomar decisões. Vimos a Álgebra Booleana, onde o sinal + significa OU (e 1 OU 1 resulta em 1), e como as Portas Lógicas (AND, OR, NOT) usam essa álgebra para controlar circuitos. Agora, vamos ver outra grande tarefa de um computador: fazer cálculos. Além de tomar decisões lógicas, os processadores precisam somar, subtrair, multiplicar e dividir, e para isso eles também usam o sistema binário.

Entramos agora na Aritmética Binária. Aqui, é crucial entender que os símbolos voltam a ter seus significados matemáticos tradicionais. O sinal + não significa mais "OU"; ele significa SOMA, como na escola. É neste novo contexto que 1 + 1 não é 1, mas sim 2. Como estamos em binário, o número 2 é escrito como 10 (lê-se "um-zero").

A soma binária, então, segue regras simples. 0 + 0 = 0 e 0 + 1 = 1. A regra principal é 1 + 1 = 10, que significa: "escreva 0 na coluna atual e leve 1 (o 'vai um') para a próxima coluna à esquerda". Se, por acaso, uma coluna já tiver um "vai um" e você precisar somar 1 + 1 + 1, o resultado é 3, que em binário é 11. Neste caso, você escreve 1 na coluna atual e leva 1 para a próxima".

A subtração binária também é parecida com a decimal. 1 - 1 = 0 e 1 - 0 = 1. O desafio é quando temos 0 - 1. Aqui, você deve "pedir emprestado" 1 da próxima coluna à esquerda. Ao fazer isso, o 0 se transforma em 10 (dois). A operação vira 10 - 1 = 1. A coluna que "emprestou" o 1, por sua vez, tem seu valor diminuído em 1. Se ela era 1, vira 0.

A multiplicação binária é, na verdade, mais simples que a decimal, pois você só precisa multiplicar por 0 ou por 1. As regras são 1 x 1 = 1, e qualquer coisa vezes 0 é 0. O processo é o mesmo da escola: você multiplica o número de cima por cada dígito do de baixo (o que resulta em simplesmente copiar o número ou escrever uma linha de zeros) e vai deslocando cada resultado para a esquerda. No final, você soma todas as linhas usando as regras da soma binária.

QUESTÕES:

Questão 1: Qual é o resultado da adição binária: 0101 + 0011?

a) 0111
b) 1000
c) 0100
d) 1001

Questão 2: Qual é o resultado da SUBTRAÇÃO binária com 'borrow' (emprestar): 100−001?

a) 010
b) 011
c) 101
d) 001

Questão 3: Qual é o resultado da MULTIPLICAÇÃO binária: 11×10?

a) 101
b) 111
c) 110
d) 011

DIARIODEBORDOFEIRADASCHATUBA

Diário de Bordo – Projeto Eletroboy

03/03/2025 – Início oficial do projeto com **Arduino Mega**. Definição dos objetivos gerais, levantamento de requisitos e planejamento de arquitetura inicial.

10/03/2025 –Realização dos Primeiros testes com componentes básicos e integração inicial da placa.
17/03/2025 – Avanço na programação base, ajustes nos protótipos e validação das funções essenciais.
24/03/2025 – Primeiras análises de compatibilidade da tela com o sistema e estudo das limitações físicas dos pinos.
31/03/2025 – Início da configuração do **Display LCD TFT 3.5"** e testes de comunicação com a placa.

07/04/2025 – Integração da tela concluída e execução dos primeiros testes visuais.
14/04/2025 – Ajustes no layout do circuito devido à ocupação dos pinos pelo shield da tela.
21/04/2025 – Refinamento da interface gráfica inicial e organização dos elementos na tela .
28/04/2025 – Testes de estabilidade do sistema com tela e sensores conectados.

05/05/2025 – Migração de testes para **Arduino Uno** visando economia e comparação de desempenho.

12/05/2025 – Adaptação da interface ao Uno e replanejamento da integração com a tela.
19/05/2025 – Implementação inicial dos botões físicos.
26/05/2025 – Testes individuais dos botões para garantir precisão e confiabilidade.

02/06/2025 – Identificação de lentidão no sistema durante a execução de múltiplos elementos.
09/06/2025 – Pesquisa de alternativas para melhorar desempenho.

16/06/2025 – Decisão de transição para o **ESP32** devido à maior capacidade e velocidade.
23/06/2025 – Aquisição do **ESP32** e realização dos primeiros testes com a nova plataforma.
30/06/2025 – Início da adaptação do código para a nova plataforma.

07/07/2025 – Otimização de desempenho e testes mais complexos com o ESP32.
14/07 a 04/08/2025 – **Período de recesso do projeto**.

04/08/2025 – Retomada do projeto e revisão geral do sistema.
11/08/2025 – Implementação final da lógica do **quiz**.

18/08/2025 – Construção da base da carcaça do dispositivo.
25/08/2025 – Conclusão da carcaça física e preparação para os testes finais.

Guia 3

Diário de Bordo – Projeto Eletroboy (versão ampliada)

Março – Fundação do Projeto

03/03/2025 – Início oficial do projeto com Arduino Mega. Definição dos objetivos gerais, levantamento de requisitos e planejamento da arquitetura inicial.
10/03/2025 – Primeiros testes com componentes básicos, verificação das ligações e integração inicial da placa.
17/03/2025 – Avanço na programação base, ajustes nos primeiros protótipos e validação das funções essenciais.
24/03/2025 – Análise de compatibilidade da tela com o sistema, estudo das limitações físicas dos pinos e organização preliminar do circuito.
31/03/2025 – Início da configuração do Display LCD TFT 3.5" e testes de comunicação com a placa.

Abril – Consolidação da Interface

07/04/2025 – Integração da tela concluída e execução dos primeiros testes visuais.
14/04/2025 – Ajustes no layout do circuito devido à ocupação dos pinos pelo shield da tela e reorganização do cabeamento.
21/04/2025 – Refinamento da interface gráfica inicial e reposicionamento dos elementos na tela.
28/04/2025 – Testes de estabilidade do sistema com tela e sensores conectados.

Maio – Ajustes de Hardware

05/05/2025 – Migração temporária dos testes para Arduino Uno para comparação de desempenho.
12/05/2025 – Adaptação da interface ao Uno e replanejamento da comunicação com a tela.
19/05/2025 – Implementação inicial dos botões físicos e primeiros testes de leitura digital.
26/05/2025 – Testes individuais dos botões garantindo precisão e confiabilidade.

Junho – Problemas, Pesquisas e Evolução

02/06/2025 – Identificação de lentidão no sistema durante a execução de múltiplos elementos gráficos.
09/06/2025 – Pesquisa de alternativas e estudo de plataformas mais rápidas.
16/06/2025 – Decisão de transição para o ESP32 devido à maior capacidade de processamento.
23/06/2025 – Aquisição e primeiros testes com o ESP32.
30/06/2025 – Início da adaptação do código para a nova plataforma.

Julho – Otimizações

07/07/2025 – Otimização de desempenho e testes mais complexos com o ESP32.
14/07 a 04/08/2025 – Recesso do projeto.

Agosto – Retomada e Estrutura Física

04/08/2025 – Retomada do projeto e revisão geral do sistema.
11/08/2025 – Implementação final da lógica do quiz, ajustes de fluidez entre telas e correções de navegação.
18/08/2025 – Início da construção da base da carcaça física.
25/08/2025 – Conclusão da parte principal da carcaça e preparação para testes práticos.

Setembro – Materiais, Corte e Ajustes Estruturais

01/09/2025 – Compra dos materiais estruturais: madeira MDF fina, chapas de acrílico para o visor e suportes internos. Planejamento detalhado das medidas.
04/09/2025 – Primeiro corte da madeira e do acrílico. Identificação de erro nas medições da abertura da tela — ajustes não encaixaram.
08/09/2025 – Replanejamento das medidas e criação de moldes simples para evitar novos erros.
12/09/2025 – Segundo corte da madeira e do acrílico, desta vez com encaixes mais precisos. Montagem inicial da estrutura frontal da carcaça.
18/09/2025 – Lixamento, alinhamento das peças, reforço interno e testes de encaixe do visor.
22/09/2025 – Fixação temporária do ESP32 e da tela dentro da estrutura para verificar espaço interno e ventilação.

Outubro – Montagem, Circuito Interno e Interatividade

02/10/2025 – Instalação dos botões na carcaça e testes de resistência mecânica.
06/10/2025 – Primeiro teste do buzzer. Implementação de sons simples para feedback.
07/10/2025 – Instalação do sistema de pontuação por nome.
10/10/2025 – Criação do sistema de melodias para aumentar a interatividade do jogo. Ajuste do volume evitando distorção.
15/10/2025 – Organização dos cabos internos e reposicionamento da placa para evitar interferência da tela.
20/10/2025 – Reforço das laterais da carcaça, fixação definitiva dos botões e testes de resposta rápida.
27/10/2025 – Pintura da carcaça, ajustes estéticos e polimento do acrílico frontal para melhor nitidez da tela.

Novembro – Finalização

03/11/2025 – Teste completo do quiz, correção de pequenos bugs e revisão da interface.

07/11/2025 – Ajustes finais no sistema sonoro, sincronização das melodias com eventos do jogo.
11/11/2025 – Limpeza interna da carcaça, organização definitiva dos fios e fixação final do ESP32.
14/11/2025 – Testes de uso contínuo para garantir estabilidade, checagem de aquecimento e durabilidade.
18/11/2025 – Preparação dos materiais de apresentação, revisão do roteiro e organização das demonstrações.
21/11/2025