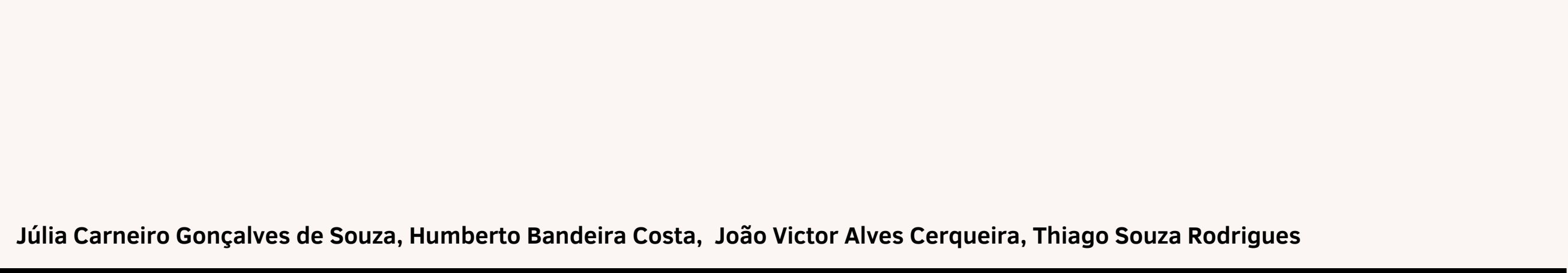
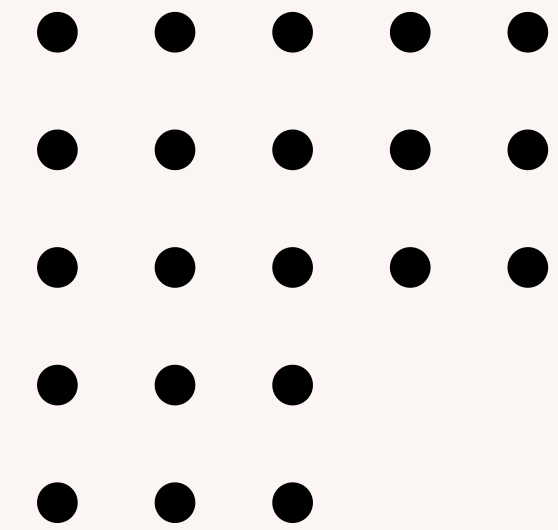


SISTEMA DIGITAL

JOGO DA VELHA

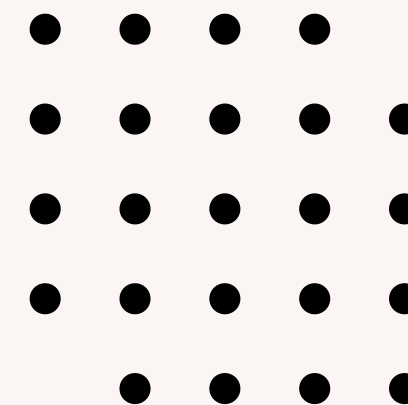


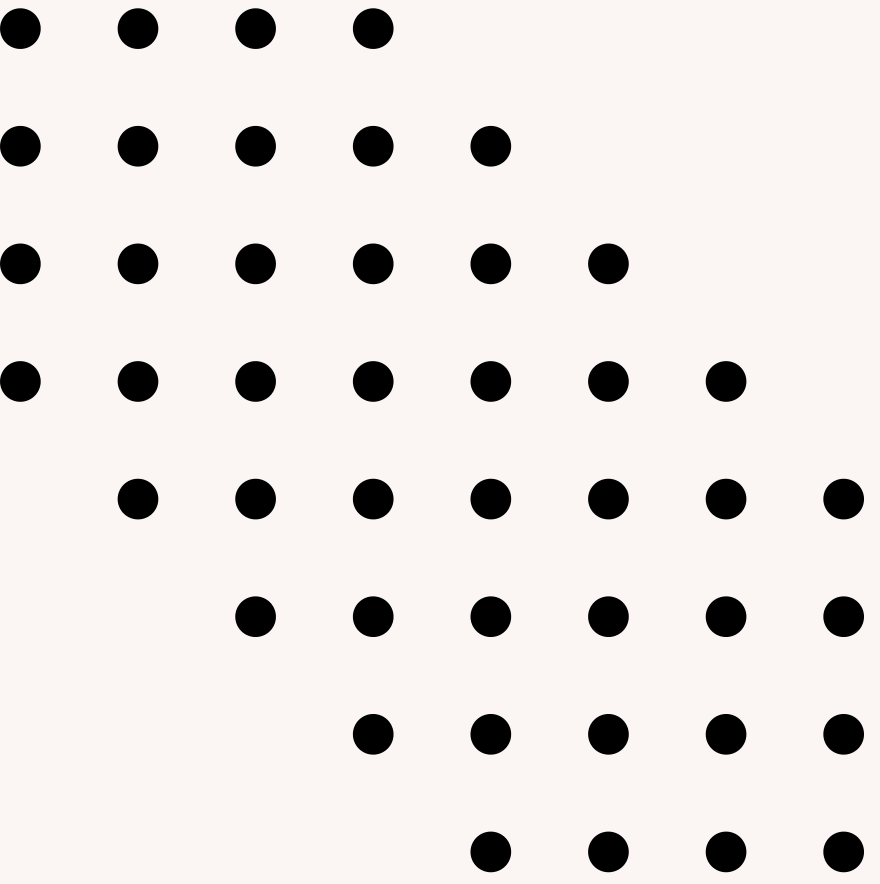
Júlia Carneiro Gonçalves de Souza, Humberto Bandeira Costa, João Victor Alves Cerqueira, Thiago Souza Rodrigues



TÓPICOS

- Introdução ao problema
- Metodologia
- Resultados e Testes
- Possíveis melhorias e Conclusão

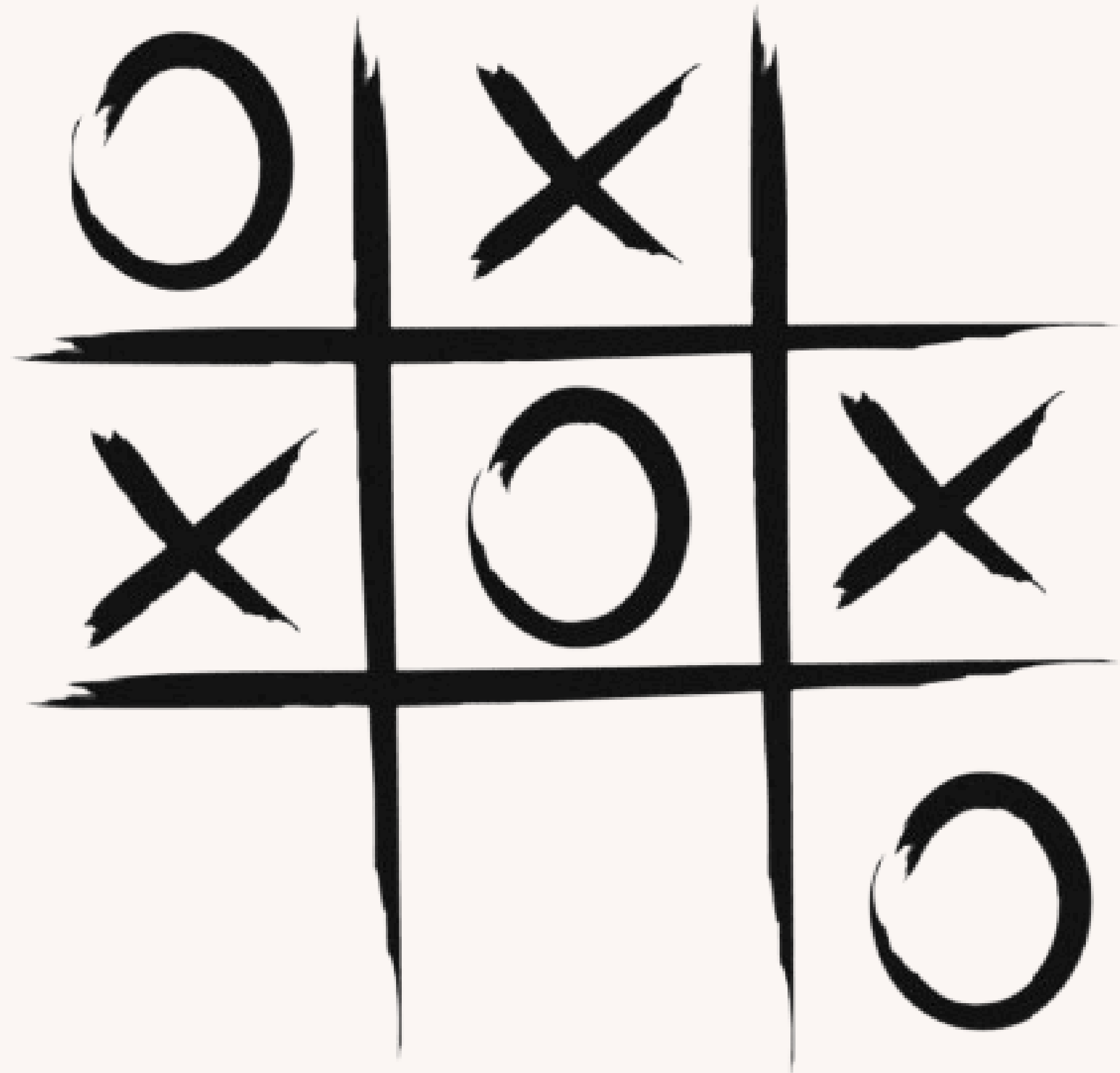


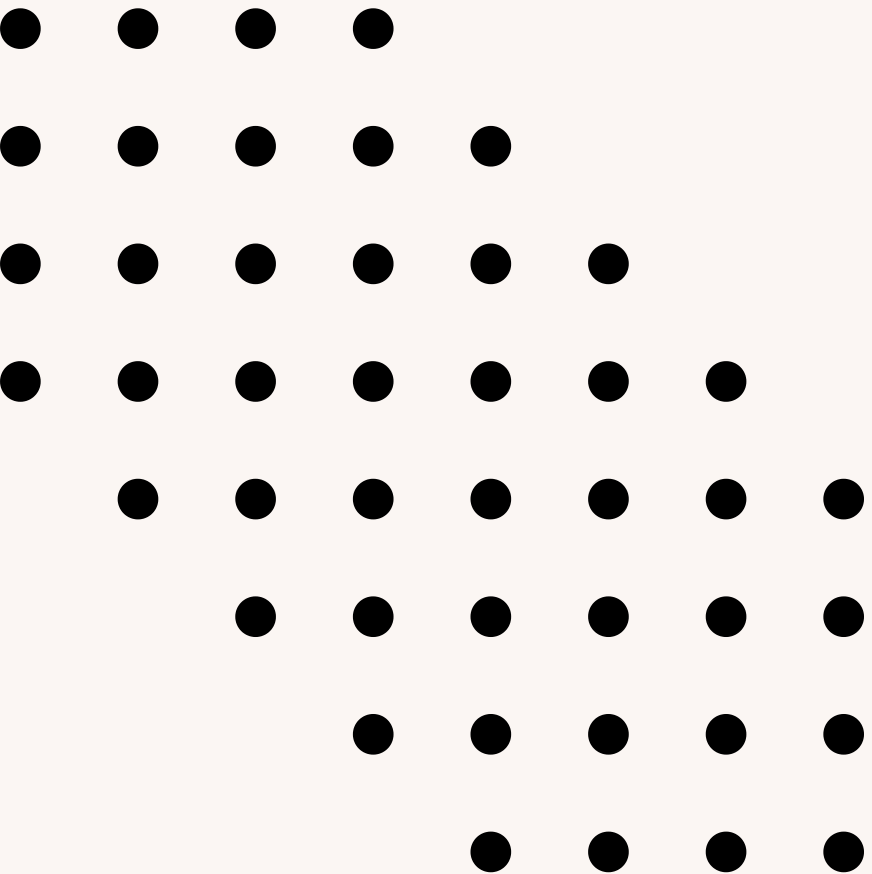


Introdução

A decorative graphic consisting of a black L-shaped line in the bottom-left corner. It starts with a vertical line segment, then turns 90 degrees to the right to become a horizontal line segment.

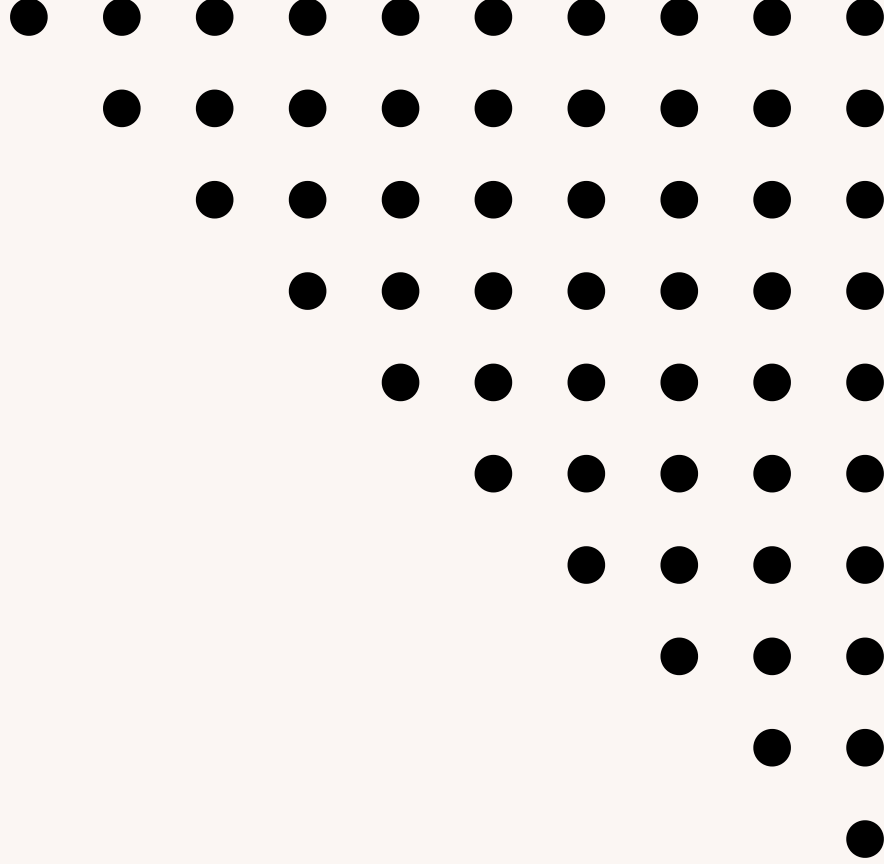
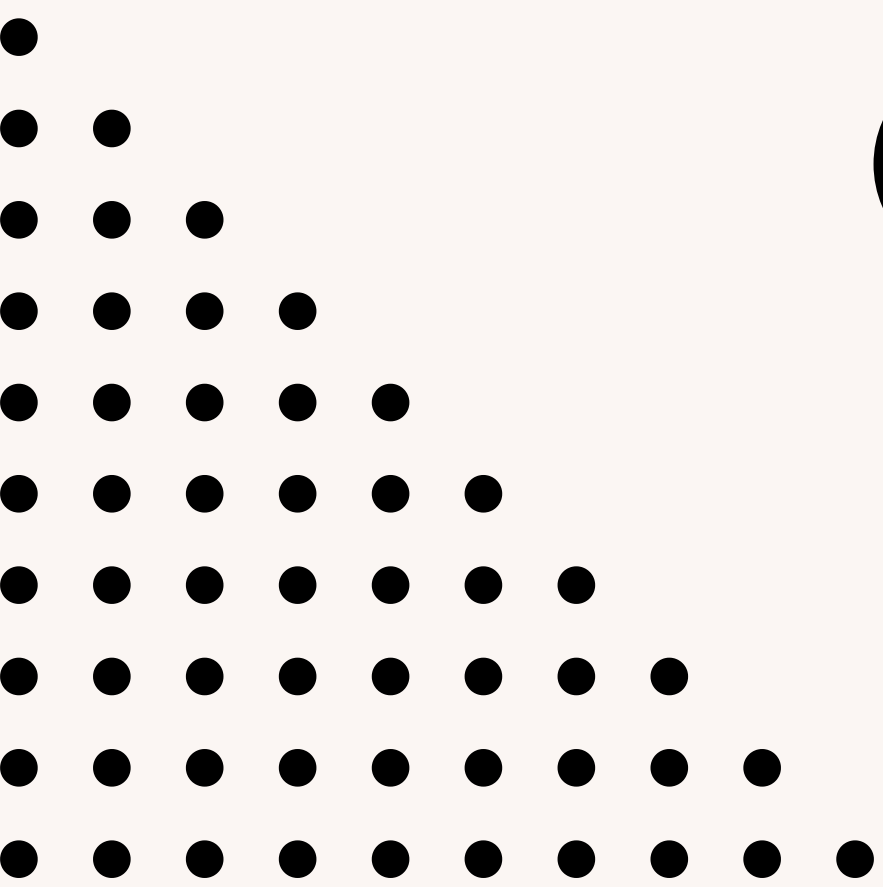
- Tik-Tak-Toe;
- Interação entre hardware e software.
- Desenvolvimento na linguagem C
- Kit de Desenvolvimento DE1-SoC.





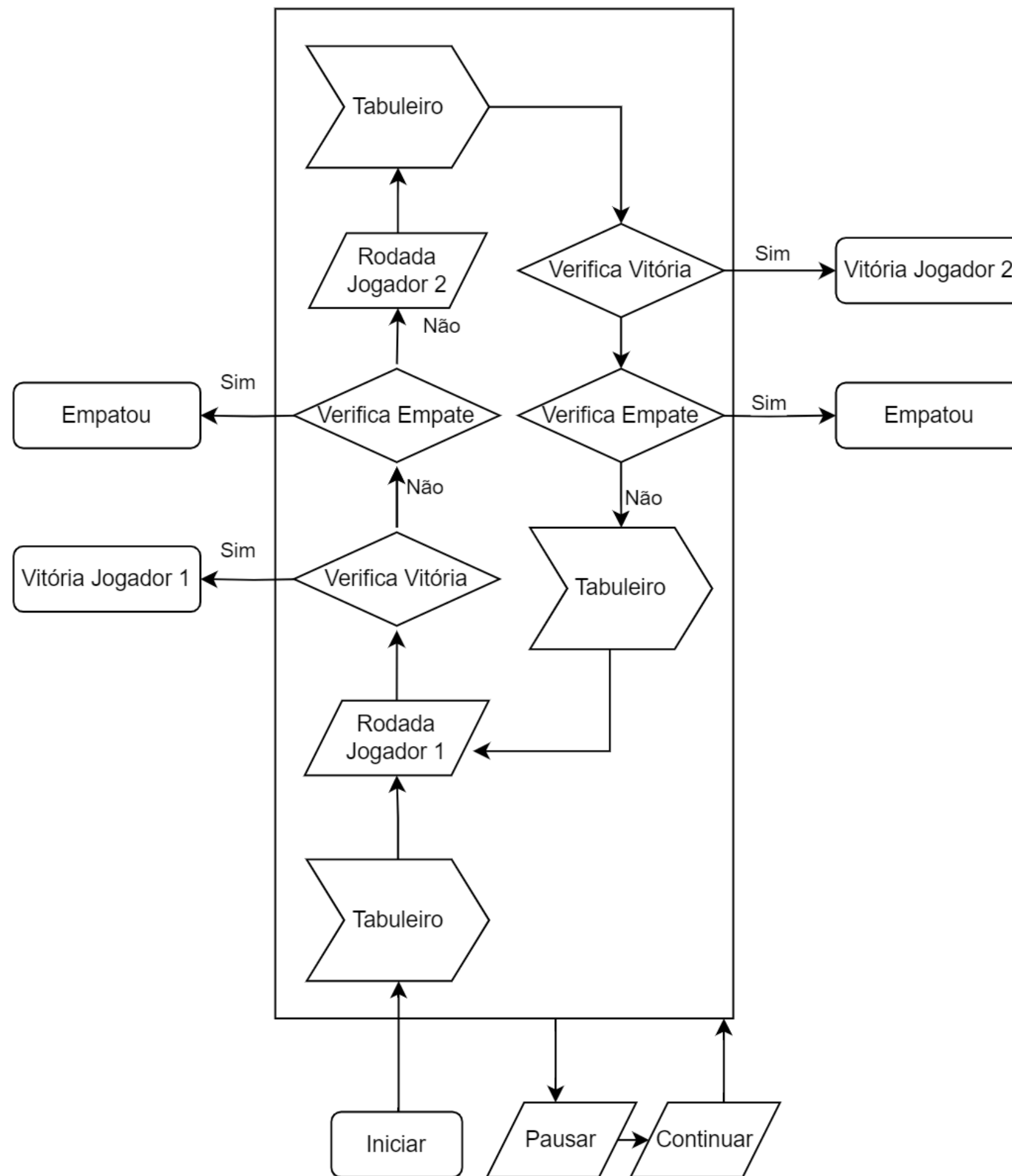
METODOLOGIA

A decorative L-shaped line graphic in the bottom-left corner, consisting of a vertical line segment and a horizontal line segment meeting at a right angle.

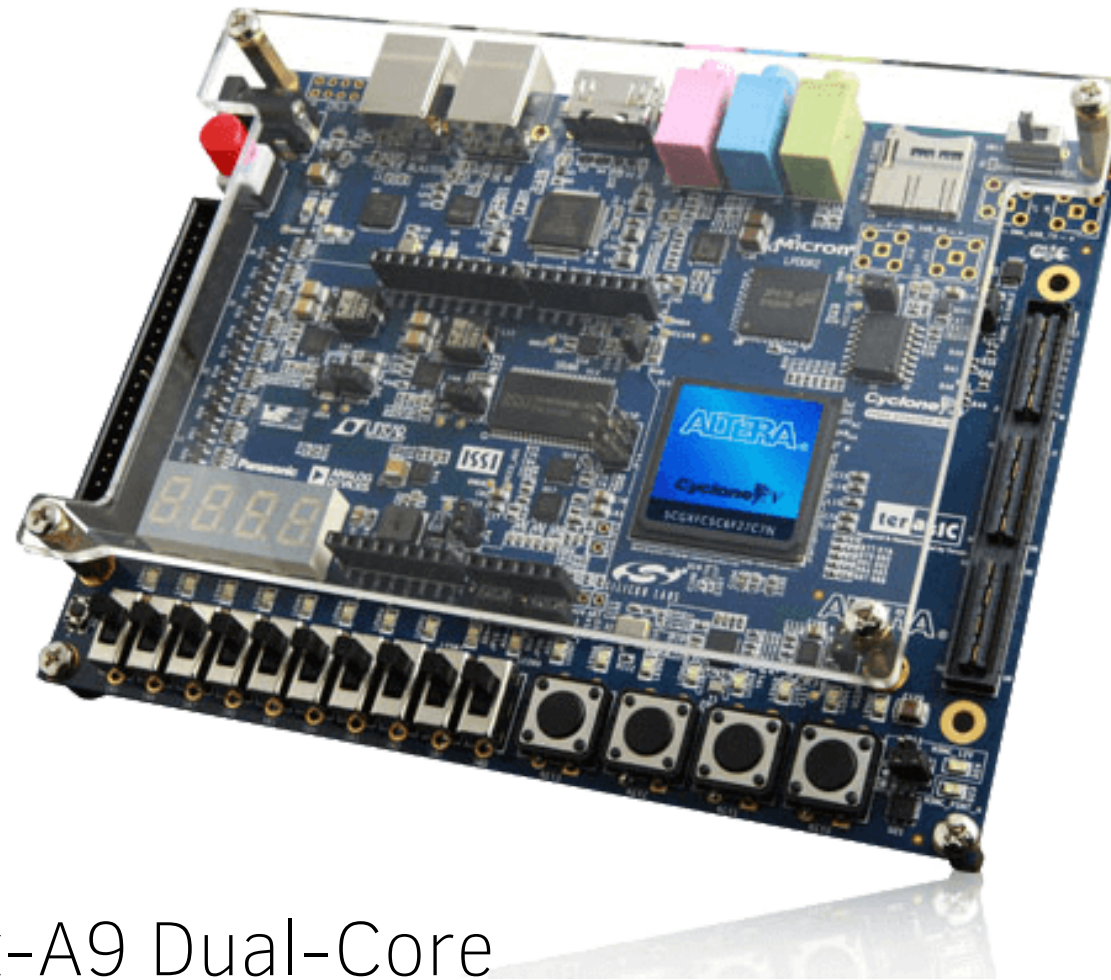
- 
- 
- 1 Fluxograma do Jogo
 - 2 Funcionamento do mouse
 - 3 Periféricos da FPGA (Switchs)

FLUXOGRAMA DO JOGO

O desenvolvimento do projeto teve como ponto de partida a construção de um fluxograma do funcionamento geral do jogo.

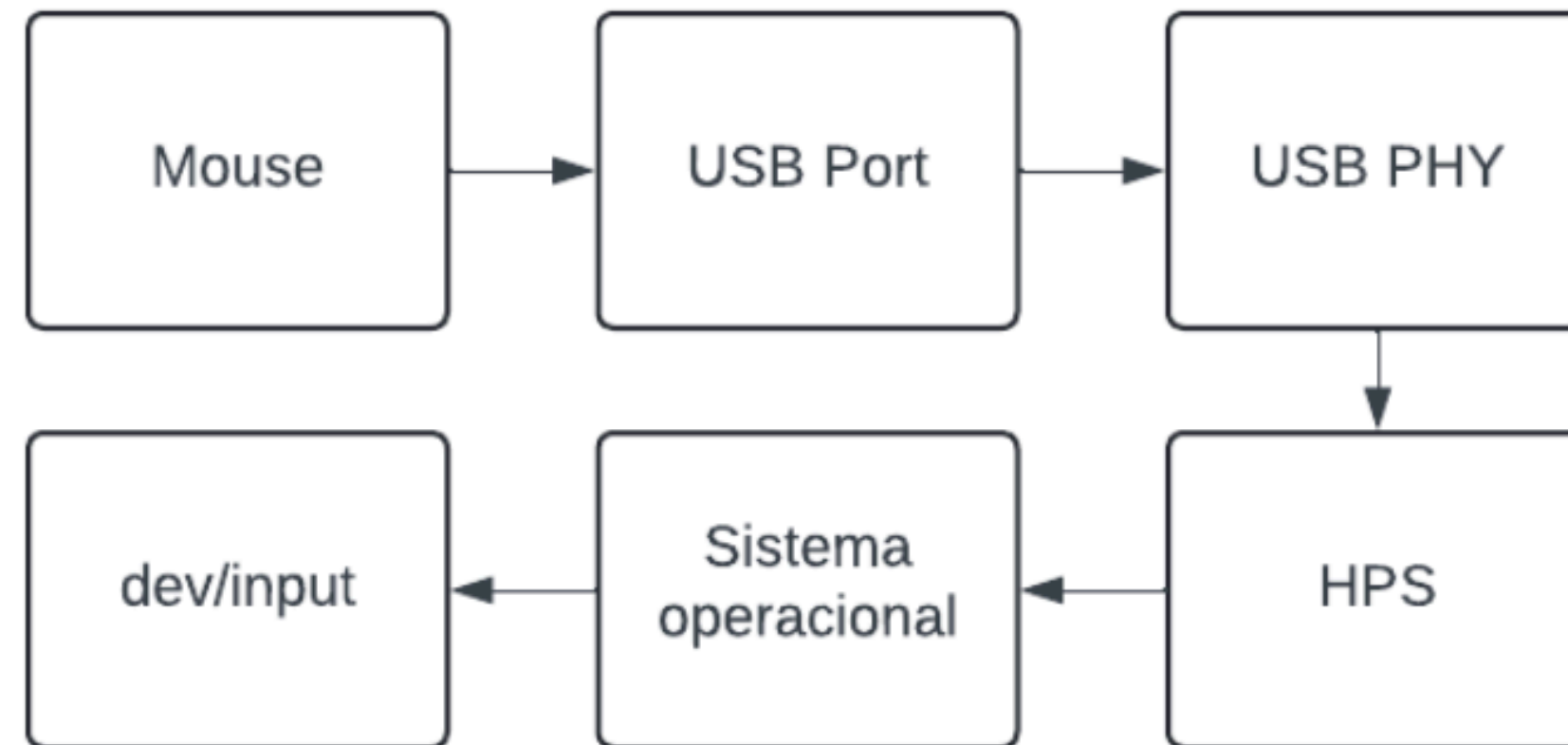


PLACA DE1-SOC



- Processador ARM Cortex-A9 Dual-Core
 - Executa o sistema operacional Linux e de outras aplicações de software.
- FPGA da família Intel Cyclone V
 - Possui diversos periféricos (Switches)

FUNCIONAMENTO DO MOUSE



- Sensor óptico - sinais elétricos, Porta USB, Controlador USB, Sistema Operacional

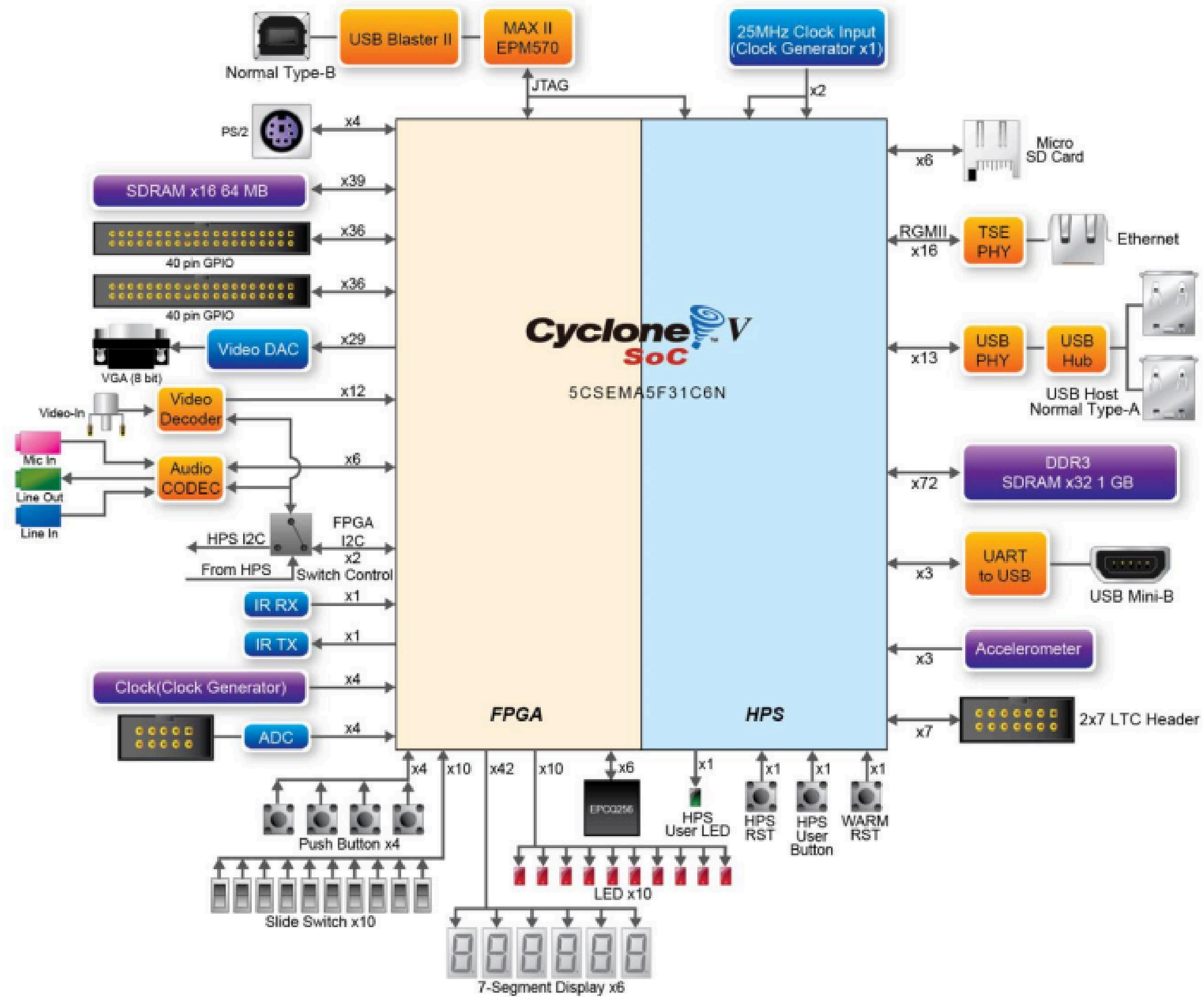
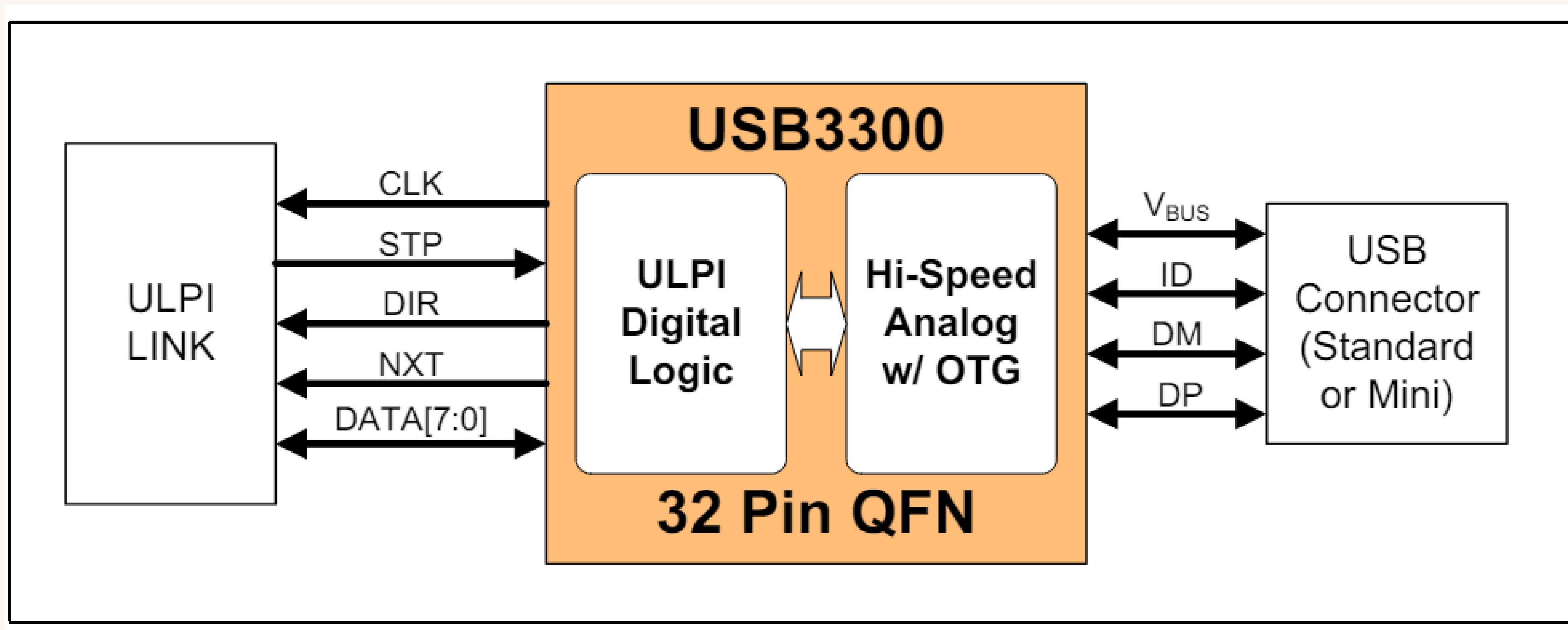
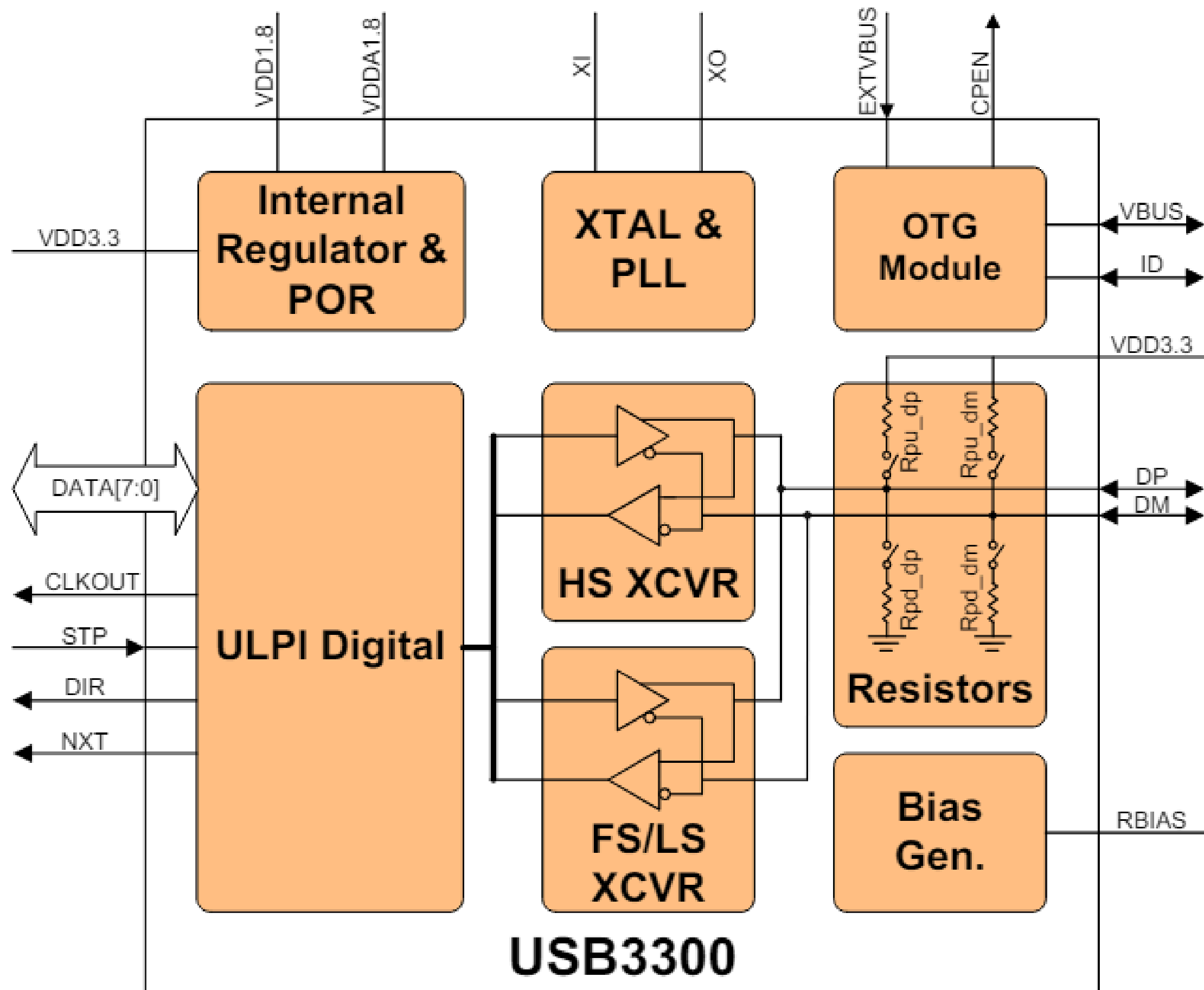
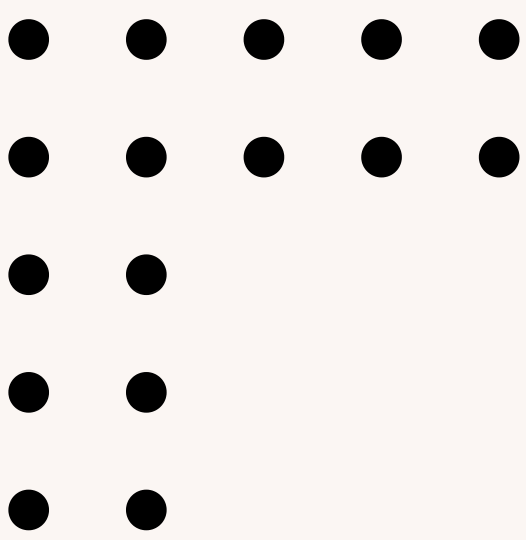


Figure 2-3 Block diagram of DE1-SoC

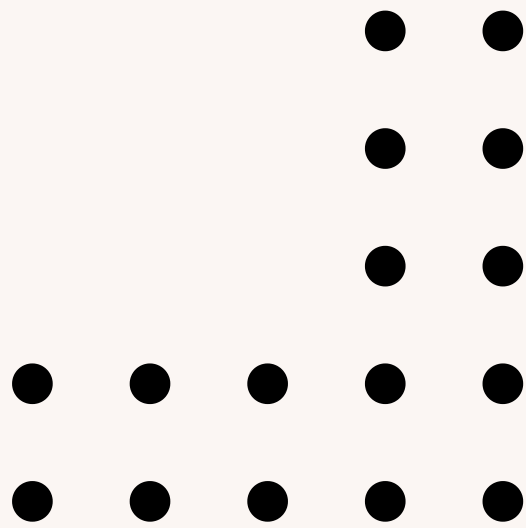


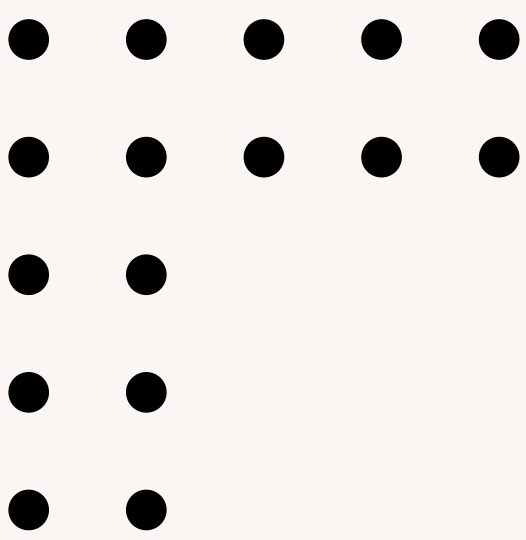




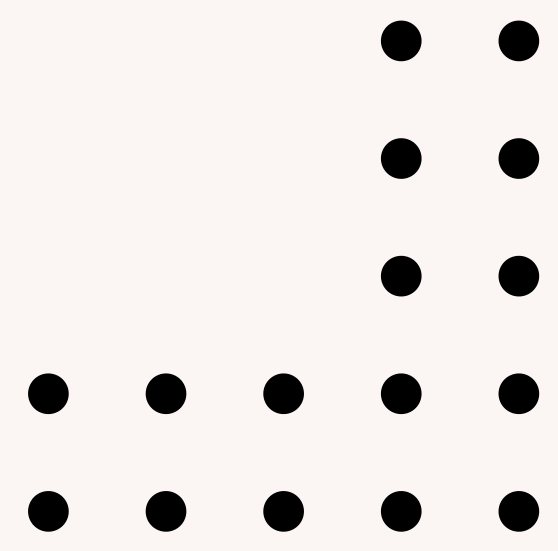
```
joavac@joavac-VirtualBox:/dev/input$ sudo cat mice
```

A large amount of raw mouse input data is displayed as a continuous stream of hexadecimal values.





```
joaovac@joaovac-VirtualBox:/dev/input$ sudo cat mice | od -t x1 -w24
0000000 38 e4 db 08 01 00 08 01 00 08 01 00 08 01 00 08 01 00 28 01 ff 28 01 ff
0000030 08 01 00 08 01 00 28 00 ff 08 01 00 28 01 ff 28 00 ff 08 01 00 28 01 ff
0000060 28 00 ff 28 01 ff 28 00 fe 08 01 00 28 00 ff 28 00 ff 08 01 00 28 00 ff
0000110 28 00 ff 28 00 ff 28 00 fe 08 01 00 28 00 fe 28 00 ff 28 00 fe 28 00 fe
0000140 28 00 ff 28 00 ff 38 ff fe 28 00 fe 28 00 ff 28 00 ff 18 ff 00 38 ff ff
0000170 28 00 ff 38 ff ff 38 ff ff 28 00 ff 18 ff 00 18 ff 00 28 00 ff 18 ff 00
0000220 38 ff ff 28 00 ff 28 00 ff 28 00 ff 28 00 ff 28 00 ff 28 00 ff 08 01 00
0000250 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01
0000300 08 01 00 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01
0000330 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 18 ff 03 08 00 02 18 ff 01
0000360 08 00 01 08 00 01 08 00 02 08 00 01 18 ff 01 08 00 01 08 00 01 18 ff 01
0000410 08 00 01 08 00 01 08 00 01 18 ff 00 28 00 ff 28 00 ff 38 ff ff 28 00 ff
0000440 28 00 ff 18 ff 00 28 00 ff 28 00 ff 28 00 ff 18 ff 00 28 00 ff 28 00 ff
0000470 38 ff ff 28 00 ff 28 00 ff 28 00 ff 28 00 ff 18 ff 00 28 00 fe 28 00 ff
0000520 28 00 ff 28 00 ff 28 00 ff 28 00 ff 08 00 01 08 00 01 08 00 01 08 00 01
0000550 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01 08 00 01
*
```



PERIFÉRICOS DA FPGA (SWITCHS)

- Esses interruptores são utilizados para uso como entradas de dados.
- Cada interruptor é conectado diretamente a um pino no Cyclone V SoC FPGA.
- Quando o interruptor está na posição PARA BAIXO, fornece um nível lógico baixo ao FPGA, e quando o switch é na posição UP fornece um alto nível lógico.



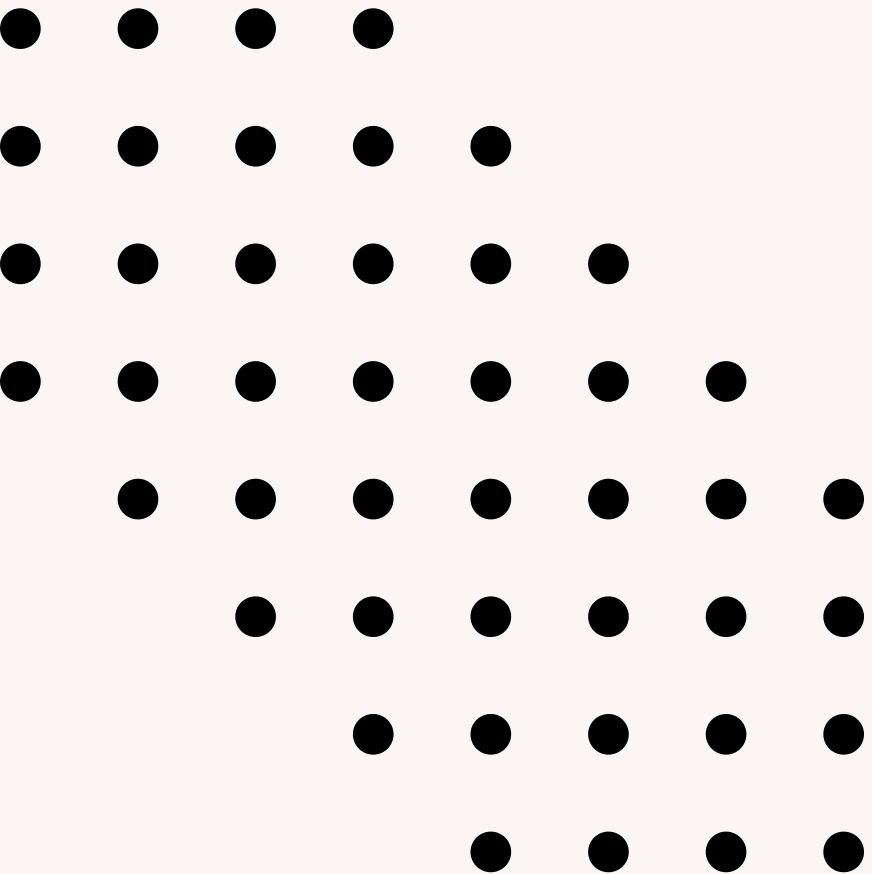
```
#ifndef SW_H
#define SW_H

/**
 * Function SW_open: opens the slide switch SW device
 * Return: 1 on success, else 0
 */
int SW_open (void);

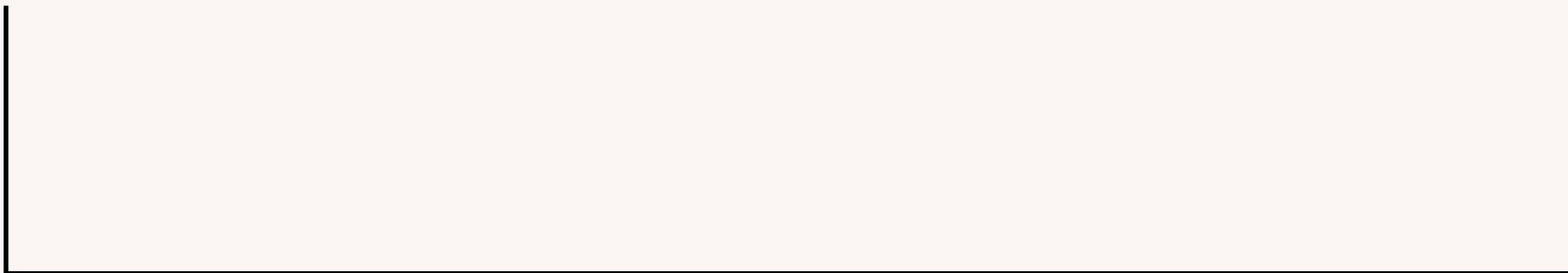
/**
 * Function SW_read: reads the SW device
 * Parameter data: pointer for returning data. If no switches are set *data = 0.
 * If all switches are set *data = 0b1111111111
 * Return: 1 on success, else 0
 */
int SW_read (int * /*data*/);

/**
 * Function SW_close: closes the SW device
 * Return: void
 */
void SW_close (void);

#endif
```

RESULTADOS E TESTES





Partes do código

```
int PegarMovimento(char data[3], int vetor[4], int control) {
    signed char x, y;
    int direita = 0, esquerda = 0, cima = 0, baixo = 0;

    x = data[1];
    y = data[2];

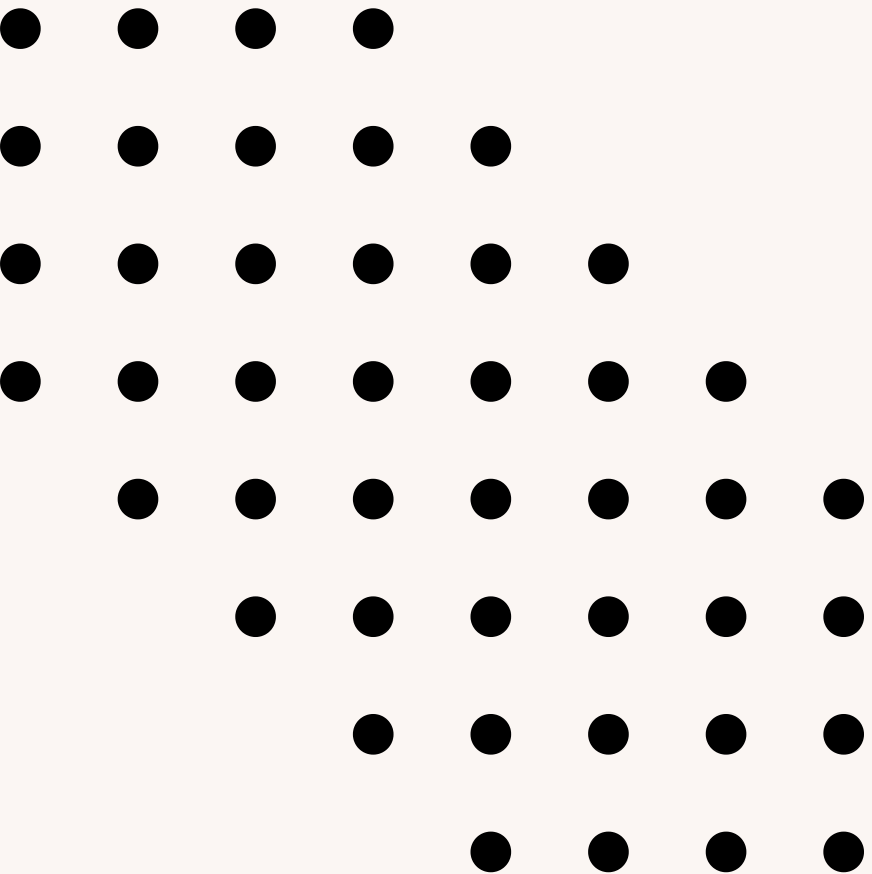
    if (x > SENSI && control == 0 && direita < 2) {
        direita += 1;
        control +=1;
    }
    else if (x < -SENSI && control == 0 && esquerda < 2) {
        esquerda += 1;
        control +=1;
    }
    else if (y > SENSI && control == 0 && cima < 2) {
        cima += 1;
        control +=1;
    }
    else if (y < -SENSI && control == 0 && baixo < 2) {
        baixo += 1;
        control +=1;
    }
    vetor[0] = direita;
    vetor[1] = esquerda;
    vetor[2] = cima;
    vetor[3] = baixo;
    return control;
}
```



```
int fd = open("/dev/input/mice", O_RDONLY);
if(fd == -1){
    printf("Nao foi possivel abrir o arquivo!\n");
}
```

```
byte = read(fd,data,sizeof(data));
```

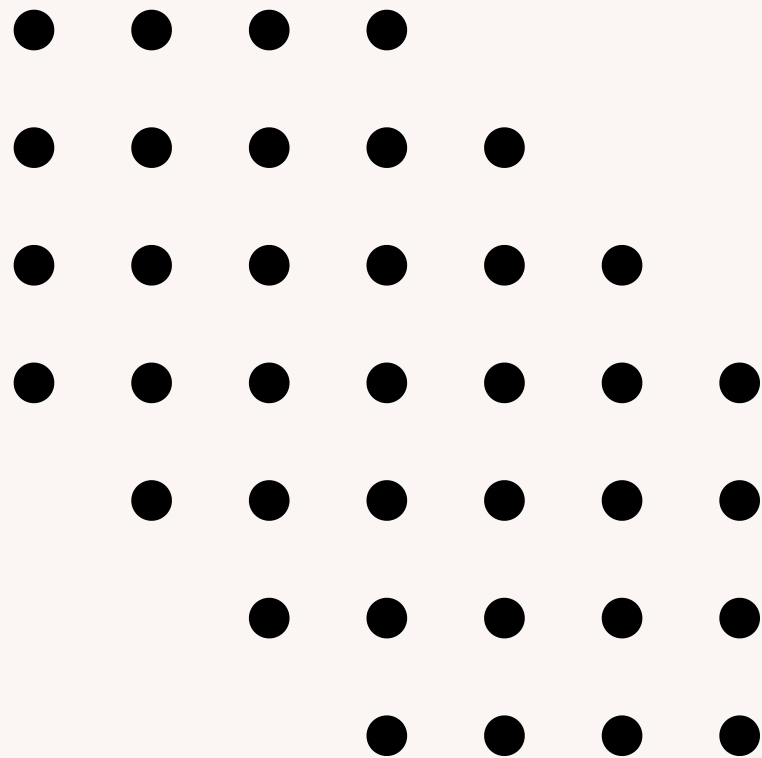




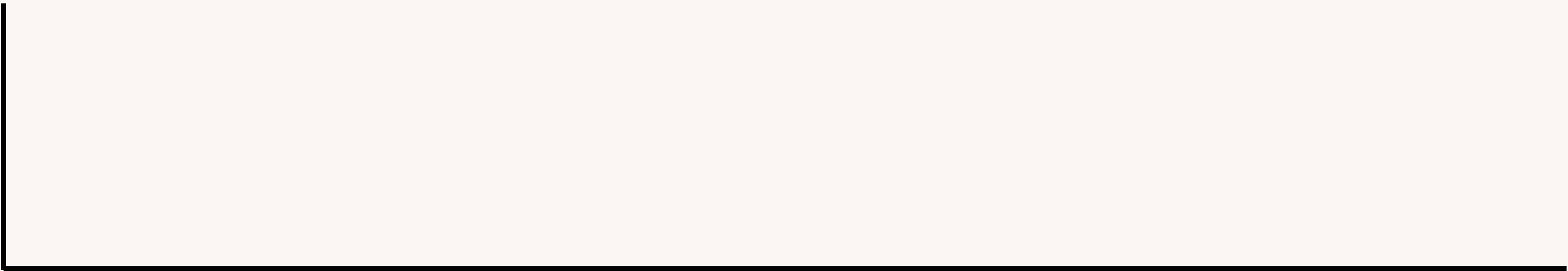
CONCLUSÃO

CONCLUSÃO

- Desenvolvimento em linguagem C.
- Utilização exclusiva de recursos disponíveis no kit FPGA DE1-SoC.
- Interação de dois jogadores.
- Captura das jogadas pelo mouse.
- Opções de iniciar, pausar e continuar o jogo.
- Aplicação de conhecimentos de interação hardware-software.
- Compreensão das políticas de gerenciamento do sistema operacional Linux em arquitetura ARM.
- Entendimento dos princípios básicos da arquitetura da plataforma DE1-SoC.
- Aprendizado de diversos outros conceitos importantes para o processo de aprendizagem.



POSSÍVEIS MELHORIAS





POSSÍVEIS MELHORIAS

- Utilizar o mostrador de 7 segmentos para exibir a pontuação dos jogadores.
 - Indicar qual jogador está na vez de efetuar a jogada no mostrador de 7 segmentos.
 - Aprimorar a movimentação do mouse na tela para torná-la mais suave.
 - Melhorar a parte visual do jogo para proporcionar uma experiência mais agradável e intuitiva.
 - Adicionar a opção de jogar com dois mouses, permitindo que cada jogador tenha seu próprio controle durante o jogo.
- 