Via de Dados com ULA

Versão 2012

RESUMO

Esta experiência tem como objetivo introduzir o conceito de ***via de dados***, que é largamente empregado na implementação de sistemas computacionais, com a familiarização com circuitos *tri-state.* Para isto será implementado um circuito que inclui também uma ULA para realizar operações sobre os dados armazenados em registradores bidirecionais, portas de entrada e portas de saída.

1. INTRODUÇÃO TEÓRICA

O conceito de via (ou "*bus*") foi aplicado para resolver problemas como o mostrado na Figura 1.1, ou seja, a transferência de informações entre vários registradores.

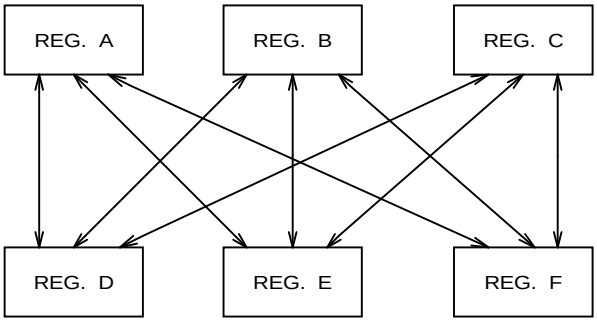

Figura 1.1 - Sistema composto de vários registradores, que transferem informações entre si.

A solução usual para este problema seria colocar uma porta de seleção (multiplexador) na entrada de cada registrador, o que encarece muito o sistema. Contudo, se for possível garantir que neste sistema nunca seja necessário efetuar *simultaneamente* duas transferências distintas, pode-se utilizar o conceito de *via de dados*.

Dentro do conceito de via de dados, é necessário se conhecer alguns dos circuitos básicos que permitem sua implementação e que são mostrados a seguir.

1.1. Circuitos com Saídas "Tri-state"

Qual é a utilidade dos dispositivos "tri-state"[1]? Nos sistemas eletrônicos é comum a utilização intensiva de vias de dados ("*bus*"), que se constitui em um meio físico de comunicação de dados compartilhado entre os vários módulos que compõem o sistema. Para que isto seja possível, utilizam-se os circuitos integrados "tri-state". Seja a figura 1.2 que mostra uma linha física de dados compartilhada por várias portas lógicas.

Os dispositivos "*tri-state*" têm uma entrada de controle, conhecida como "*enable*" ou "*output control*", capaz de colocar as saídas no estado de alta impedância. Assim, a saída de um circuito "tri-state" pode assumir três estados: zero lógico, um lógico e alta impedância.

Uma via de comunicação é criada ligando-se várias saídas juntas, porém os circuitos de controle (que geram os sinais de "*enable*") devem assegurar que, no máximo, uma saída esteja na condição de circuito TTL e todas as outras estejam com impedância alta, de forma a permitir que uma única porta possa transmitir sua informação. Em outras palavras, um controle adequado do sinal "*enable*" de cada circuito permite a multiplexação das várias saídas que estão ligadas à via.

[1] O termo "tri-state" é marca registrada da National Semiconductor Corporation.

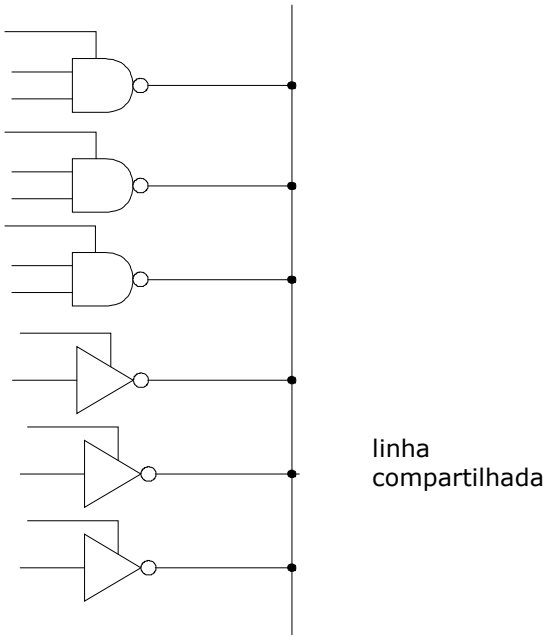

Figura 1.2 - Configuração com várias portas *tri-state* compartilhando uma linha física.

Características Elétricas

Os circuitos "tri-state" são construídos de forma que a passagem para o estado TTL seja mais lenta do que a passagem para o estado de alta impedância. Assim, se um dispositivo aciona simultaneamente uma porta "tri-state" para passar ao estado TTL e outro, da mesma via, para passar ao estado de alta impedância, não deverá ocorrer nenhum problema.

Se duas saídas forçam, erroneamente, níveis lógicos TTL conflitantes ao mesmo tempo, o resultado é igual àquele que ocorre com duas saídas TTL curto-circuitadas. Os dispositivos podem ser danificados e a tensão de saída não estará dentro das faixas dos níveis lógicos. Esta situação deve, portanto, ser evitada.

Uma saída "tri-state" em estado de alta impedância tem uma corrente de fuga de 40 µA, a qual deve ser considerada no dimensionamento do número de portas na via de comunicação. No estado lógico zero, uma saída deve absorver 40 µA de cada saída em alta impedância e 1,6 mA de cada entrada TTL ligada à via de comunicação.

No estado lógico um, uma saída deve fornecer 40 µA de corrente para cada saída em alta impedância, ligada à via de comunicação e 40 µA para cada entrada TTL na via de comunicação. Uma porta "tri-state" típica absorve 16 mA no nível lógico zero, mas fornece 5,2 mA no nível lógico um. Isto permite a ligação de muitas portas numa via de comunicação.

1.2. Circuitos com Saídas "Open Collector"

Os componentes "*open collector*" (coletor aberto) possuem uma implementação semelhante aos componentes com saída do tipo "*totem pole*". A diferença básica é que nas saídas "*open collector*" foi suprimida a ligação entre o coletor e VCC dos transistores de saída.

Naturalmente, percebe-se a necessidade de um resistor externo (resistor de "*pull up*") ligado a VCC para a corrente de polarização dos transistores e conseqüente definição dos níveis de saída.

Com este tipo de componente consegue-se ligar várias saídas conjuntamente, utilizando-se um único resistor, de acordo com a figura 1.3.

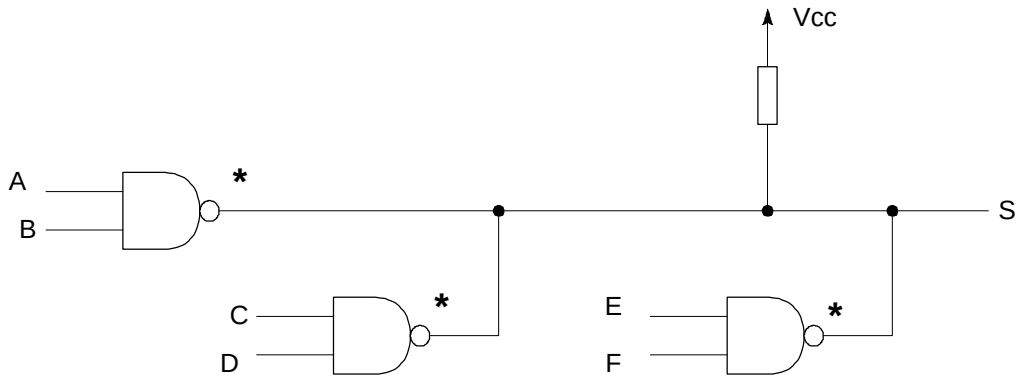

Figura 1.3 - Configuração de Portas NAND tipo coletor aberto, compartilhando linha física

No ponto S tem-se a função lógica $S = \overline{AB} \bullet \overline{CD} \bullet \overline{EF}$, e por esta razão este tipo de ligação é chamado de "*wired-and*". Isto é, saídas coletor aberto ligadas entre si resultam num "AND" lógico das saídas individuais.

Estes componentes também são utilizados para a interligação de várias entradas e saídas em configuração de via de dados (com resistores de "*pull up*" para cada via), apresentando com vantagens o fato de não sofrerem danos se mais de uma saída for ativada (transistor de saída for colocado em condução) ao mesmo tempo, além de serem interfaces passivas, isto é, não precisam alimentar a via (não injetam corrente na via).

1.3. Comparação de Características Elétricas

Comparando-se as saídas do tipo "*tri-state*" e as do tipo "*open collector*", tem-se o seguinte panorama:

- Saídas "*tri-state*": menor consumo; mais rápidas; exigem controle centralizado com ligação em via, pois são sensíveis à ativação de mais de uma saída ao mesmo tempo ("conflito de via");
- Saída "*open collector*": requer polarização externa (resistor de "*pull up*"); é uma interface passiva; quando ligada em configuração de via não precisa de controle centralizado, pois suporta erros de ativação (ativação de mais de uma saída ao mesmo tempo); é lenta (principalmente com cargas capacitivas) por não poder injetar corrente por sua saída (só consegue absorver corrente com seu transistor de saída).

1.4. Via de Dados

Uma *via de dados*, ou simplesmente *via*, é um conjunto de linhas utilizado como *caminho comum*, que interliga entradas e saídas de vários registradores, de modo que as informações possam ser transferidas facilmente de qualquer um destes registradores para qualquer outro, desde que se utilizem os sinais de controle apropriados. A Figura 1.4 mostra um sistema organizado por vias, com três registradores com saídas "*tri-state*".

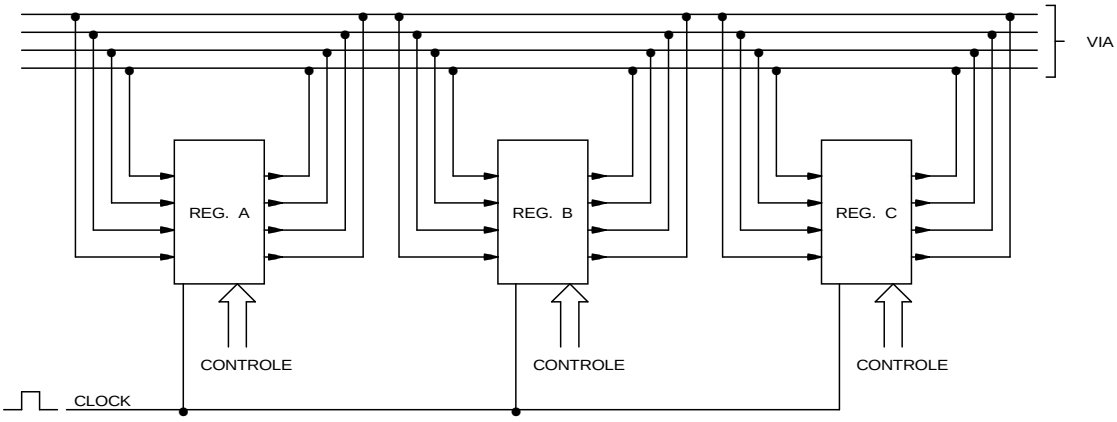

Figura 1.4 - Sistema composto de vários registradores interligados através de uma *Via de Dados*.

Pode parecer incoerente que todas as entradas e todas as saídas correspondentes estejam ligadas entre si. A incoerência, porém, desaparece se for assegurado que apenas um dos três registradores tenha as suas saídas habilitadas, enquanto que as saídas dos outros dois registradores estejam em seu estado de alta impedância. Por exemplo, suponha que os registradores B e C estejam com as saídas desabilitadas, enquanto o registrador A tem as suas saídas habilitadas. Isto, em essência, desliga as saídas de B e C da via, de forma que apenas as saídas do registrador A tenham os seus níveis lógicos presentes nas linhas que compõem a via.

Para realizar, por exemplo, a transferência de dados do registrador A para o registrador C deve-se em primeiro lugar, desabilitar as saídas de B e C e habilitar as saídas do registrador A, colocando-se os seus dados na via. O registrador C, por sua vez, deve ter suas entradas controladas para que *copiem* o que está na via. Todas as outras deverão permanecer desabilitadas. A transferência ocorrerá quando vier o pulso de "*clock*" para o registrador C. A Figura 1.5 mostra um diagrama de tempos dos sinais de controle necessários para que esta transferência seja efetuada.

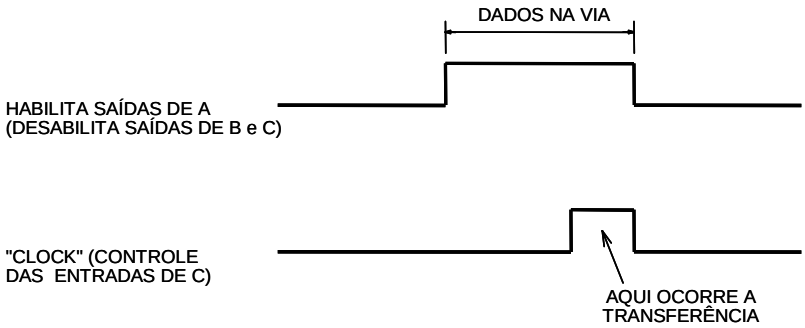

Figura 1.5 - Diagrama de Tempos para a transferência de dados do registrador A para C.

Através do exemplo apresentado, constata-se que na operação de transferência de dados entre dois registradores ligados a uma via, deve-se sempre especificar qual é o registrador de onde vem os dados (origem) e qual é o registrador que deve recebê-los (destino), habilitando tais registradores corretamente. Ou seja, é necessário selecionar (endereçar) os registradores origem e destino.

1.5. Seleção de Registradores - Endereçamento

Uma maneira de resolver este problema de seleção dos registradores envolvidos numa transferência é atribuir a cada registrador um endereço. Na verdade, cada registrador pode ter dois endereços, coincidentes ou não, um como destino e outro como origem. Neste caso, devem ser utilizados circuitos decodificadores de endereço para, a partir dos endereços apresentados, gerar os sinais de habilitação para os registradores escolhidos. A Figura 1.6 mostra o exemplo de um sistema que utiliza esta forma de seleção de registradores, com endereços de 3 bits.

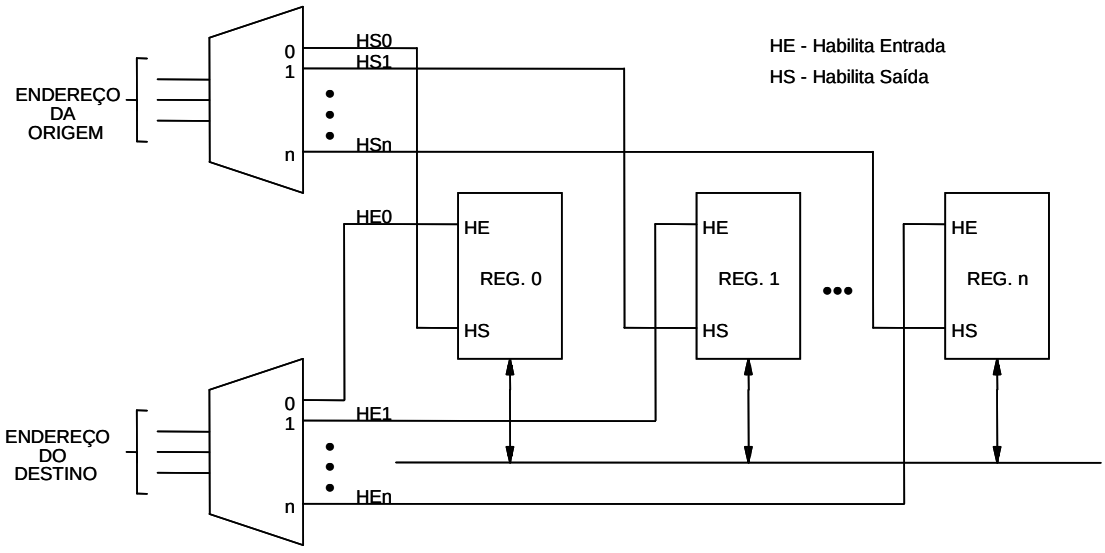

Figura 1.6 - Seleção de Registradores Utilizando Decodificadores.

Deve-se lembrar que neste modelo de registradores devem ser evitados os casos em que se endereça o mesmo registrador, tanto como origem como destino.

Um outro esquema empregado na seleção de registradores utiliza um registrador intermediário nas transferências. Neste esquema, definem-se operações de entrada e de saída deste registrador intermediário. Nas operações de entrada, ele é o destino dos dados vindos de qualquer outro registrador ligado à via; nas operações de saída, ele é a origem dos dados que vão para qualquer um dos outros registradores ligados à via. Deste modo, todas as transferências de dados passam pelo registrador intermediário, sendo necessário apenas determinar o endereço do outro registrador envolvido nesta transferência, e habilitar o sinal de controle correspondente, informando se a operação é de entrada ou de saída. A Figura 1.7 mostra um diagrama deste tipo de endereçamento.

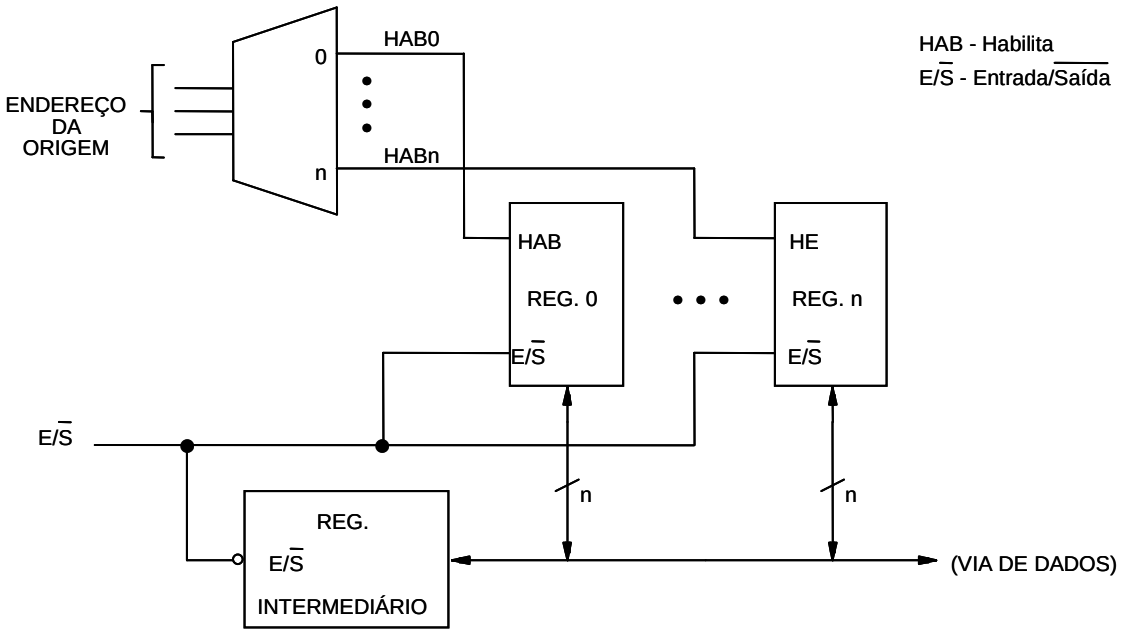

Figura 1.7 - Transferência de Dados utilizando registrador intermediário.

Utilizando os conceitos apresentados, podem ser interligados através de uma via de dados registradores bidirecionais (registradores usados para leitura e gravação de dados) como também portas de entrada e saída.

1.6. Portas de Entrada e Saída

Um sistema (por exemplo, um conjunto de registradores) para se comunicar com o meio externo, deve utilizar as chamadas portas de entrada e de saída.

a) **Porta de Entrada**

Uma porta de entrada pode ser definida como sendo um dispositivo ligado à via de dados, que permite ao sistema ler dados vindos de um periférico de entrada (teclado, conjunto de chaves, etc.).

Em sua forma mais simples, uma porta de entrada pode ser formada por um conjunto de "*buffers tri-state*", ou por um registrador com saídas "*tri-state*", cujas entradas são ligadas às saídas de dados do periférico, e cujas saídas são ligadas à via de dados. As saídas de uma porta de entrada, normalmente são habilitadas através de um endereço e um sinal de controle de leitura de periférico (Figura 1.8).

b**) Porta de Saída**

Uma porta de saída pode ser definida como sendo um dispositivo ligado à via de dados, que permite enviar dados para algum periférico de saída (terminal de vídeo, "*displays*", LEDs, etc.).

Neste caso, uma porta de saída pode ser um registrador, cujas entradas são ligadas à via de dados do sistema e cujas saídas são ligadas ao periférico de saída. Esta porta de saída é normalmente selecionada através de um endereço e de um sinal de controle de gravação de periféricos (Figura 1.8).

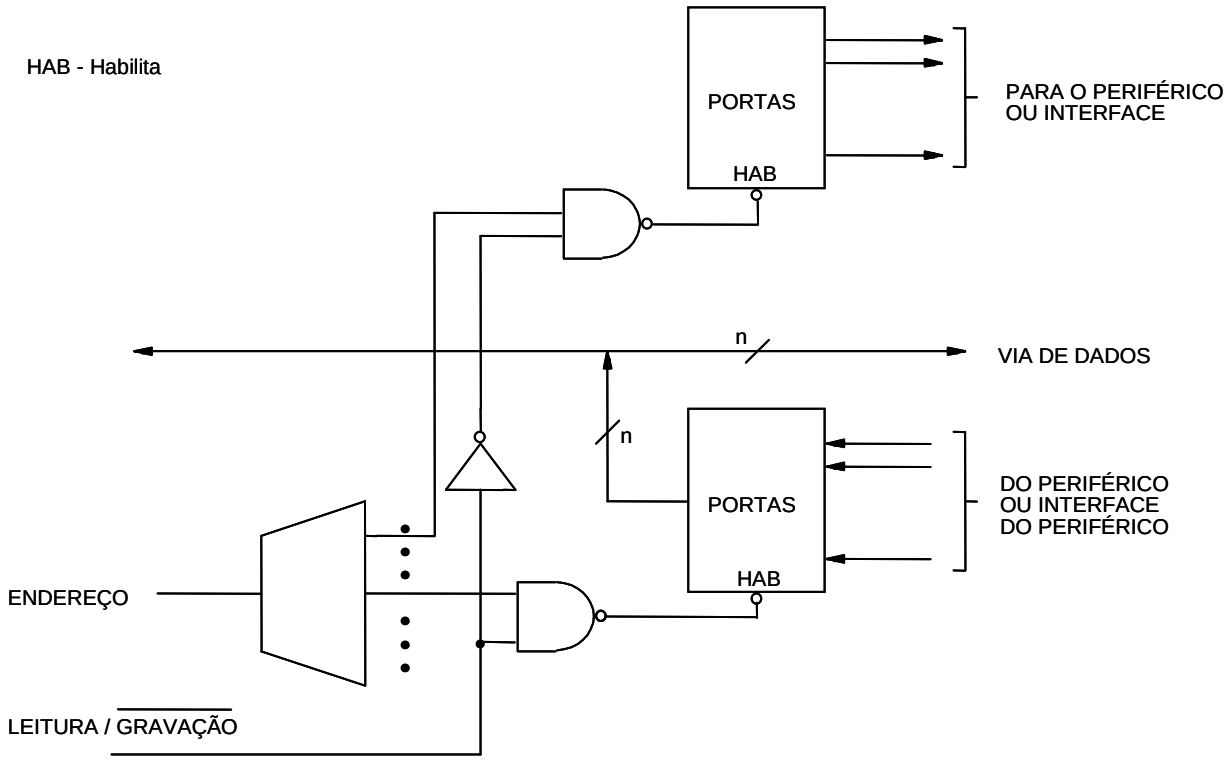

Figura 1.8 - Seleção de Portas de Entrada e de Saída.

Nos exemplos apresentados até aqui se utilizou uma representação simplificada da via, que é descrita a seguir.

1.7. Representação de uma Via de Dados

Em diagramas lógicos, a representação de uma via de dados com todas as suas linhas e respectivas ligações pode comprometer a legibilidade do diagrama correspondente. Por este motivo são utilizadas representações simplificadas nas quais um grupo de linhas é substituído por barras ou por uma só linha, indicando-se apenas o número de linhas que compõem a via. A Figura 1.9 ilustra duas destas representações simplificadas para o sistema da Figura 1.4 anteriormente apresentado.

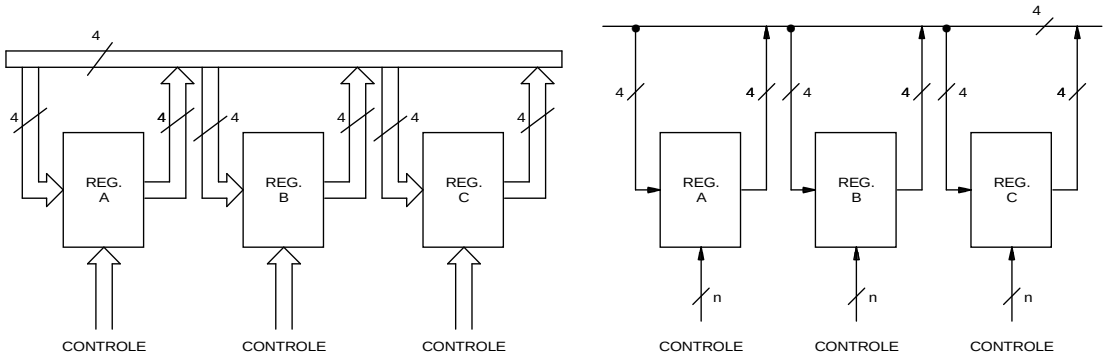

Figura 1.9 - Representações simplificadas de uma Via de Dados à qual estão interligados vários registradores.

Na Figura 1.9, como cada registrador tem as suas entradas e as suas saídas ligadas à via de dados, é óbvio que as entradas e saídas correspondentes estão ligadas entre si. Isto é possível, pois em operação normal um registrador nunca tem suas entradas e saídas habilitadas simultaneamente.

Como entradas e saídas serão ligadas entre si, muitos fabricantes de Circuitos Integrados (CIs) fazem ligação entre elas **internamente**. Isto reduz o número de pinos utilizados para o CI. A Figura 1.10 mostra o diagrama esquemático de um destes registradores ligado a uma via. Dependendo do estado das entradas de controle, as linhas E/S funcionam ou como entradas, ou como saídas, sendo chamadas de linhas de dados bidirecionais.

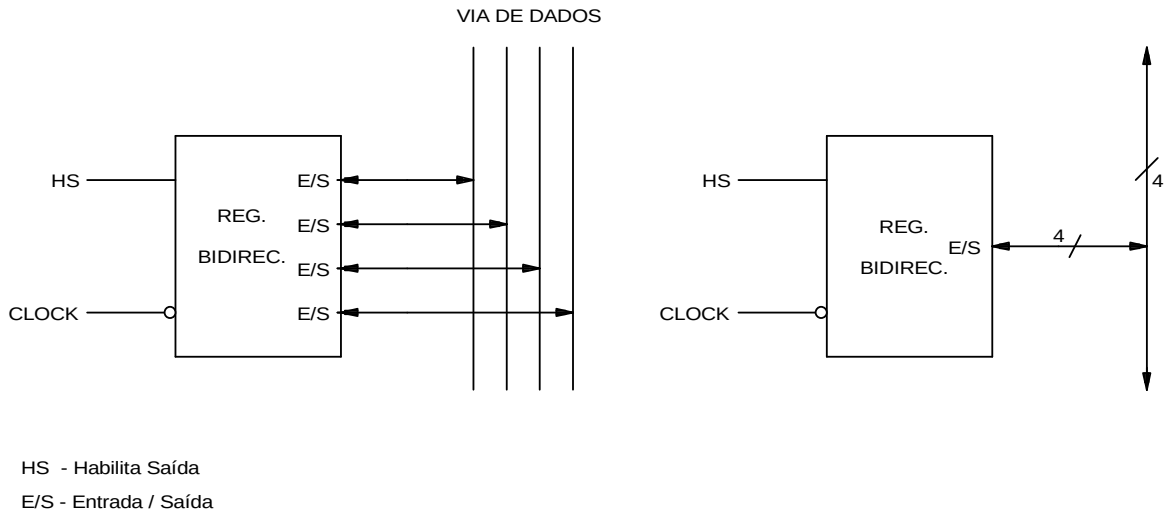

Figura 1.10 - Representação de Linhas de Dados Bidirecionais.

1.8. Unidade Lógica e Aritmética

A Unidade Lógica e Aritmética (ULA) é um circuito combinatório responsável pela execução de somas, subtrações e funções lógicas, em um sistema digital. Na figura 1.11 é mostrado um esquema simplificado de uma ULA.

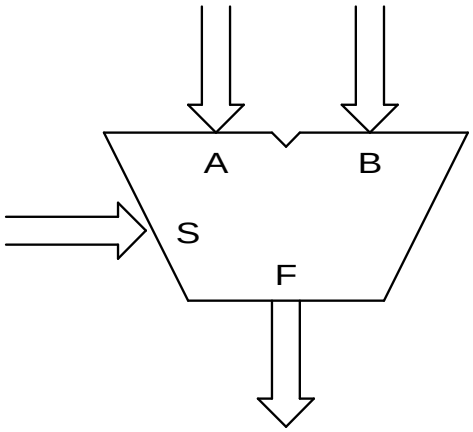

Figura 1.11 - Esquema Simplificado de uma ULA.

A operação que deve ser executada com os dados de entrada (A e B) é determinada pelos sinais de controle (S) e o resultado é obtido na saída (F). A complexidade da ULA é proporcional à complexidade do sistema em que será utilizada; assim, sistemas simples permitem o uso de ULAs simples e sistemas sofisticados exigem ULAs sofisticadas.

Uma vez estabelecido o porte do sistema, existe também o compromisso entre velocidade e preço. Por exemplo, as calculadoras eletrônicas exigem ULAs que permitem operações complexas, porém com velocidade de operação baixa, reduzindo-se o custo; já os computadores de grande porte exigem velocidade de operação elevada, aumentando o custo da ULA.

Atualmente têm-se várias alternativas de circuitos integrados que incluem uma ULA. Entre elas estão o 74181, 74381 e 74LS881.

A seguir será apresentada uma descrição de uma ULA integrada, o circuito integrado 74181.

1.9. Circuito Integrado 74181 – uma ULA de 4 bits

O circuito integrado MSI 74181 é uma ULA de 4 bits que tem possibilidade de executar 16 operações aritméticas binárias e 16 operações lógicas. A figura 1.12 apresenta um diagrama simplificado deste circuito integrado.

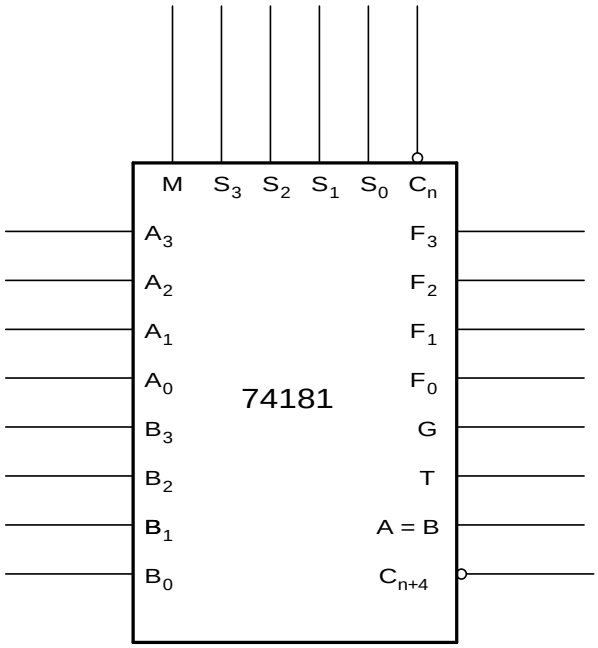

Figura 1.12 - Diagrama Simplificado da ULA 74181.

As tabelas I e II a seguir apresentam a descrição dos pinos e as operações da ULA, respectivamente.

Tabela I - Descrição dos pinos da ULA 74181.

Pinos	Tipo	Descrição
A_0 a A_3 e B_0 a B_3	Entrada	Dados de entrada
C_n	Entrada	Bit de "vem um"
S_0 a S_3	Entrada	Seleção da operação
M	Entrada	Modo de operação: M=0 - para operações aritméticas M=1 - para operações lógicas
F_0 a F_3	Saída	Dados de saída (resultado)
C_{n+4}	Saída	Bit de "vai-um"
G e T	Saídas	Utilizadas para expansão utilizando *carry look-ahead*
A=B	Saída	Indica igualdade das duas entradas

Tabela II - Sinais e Operações da ULA 74181.

Seleção				Funções Lógicas (M = 1)	Funções Aritméticas (M = 0)	
S_3	S_2	S_1	S_0		C_n = 1 (sem *carry*)	C_n = 0 (com *carry*)
0	0	0	0	$F = \overline{A}$	F = A	F = A + 1
0	0	0	1	$F = (\overline{A \text{ OR } B})$	F = A OR B	F = (A OR B) + 1
0	0	1	0	$F = \overline{A} \cdot B$	$F = A \text{ OR } \overline{B}$	$F = A \text{ OR } \overline{B} + 1$
0	0	1	1	F = 0	F = -1 (*)	F = 0
0	1	0	0	$F = \overline{A} \cdot \overline{B}$	$F = A + A \cdot \overline{B}$	$F = A + A \cdot \overline{B} + 1$
0	1	0	1	$F = \overline{B}$	$F = (A \text{ OR } B) + A \cdot \overline{B}$	$F = (A \text{ OR } B) + A \cdot \overline{B} + 1$
0	1	1	0	$F = A \oplus B$	F = A - B - 1	F = A - B
0	1	1	1	$F = A \cdot \overline{B}$	$F = A \cdot \overline{B} - 1$	$F = A \cdot \overline{B}$
1	0	0	0	$F = \overline{A} \text{ OR } B$	F = A + A.B	F = A + A.B + 1
1	0	0	1	$F = (\overline{A \oplus B})$	F = A + B	F = A + B + 1
1	0	1	0	F = B	$F = A \text{ OR } \overline{B} + A \text{ . } B$	$F = A \text{ OR } \overline{B} + A \text{ . } B + 1$
1	0	1	1	F = A . B	F = A . B - 1	F = A . B
1	1	0	0	F = 1	F = A + A	F = A + A + 1
1	1	0	1	$F = A \text{ OR } \overline{B}$	F = (A OR B) + A	F = (A OR B) + A + 1
1	1	1	0	F = A OR B	$F = A \text{ OR } \overline{B} + A$	$F = A \text{ OR } \overline{B} + A + 1$
1	1	1	1	F = A	F = A - 1	F = A

* $(-1)_{10}$ é representado por $(1111)_2$ em Complemento de 2.

As saídas G e T são os sinais "gerador de vai-um", correspondentes ao bit mais significativo e, utilizando-se o circuito integrado 74182, *look-ahead carry generator*, permitem a expansão da largura da palavra a ser manipulada.

As operações de subtração são executadas em complemento de dois (C2). Por exemplo, (-1) é representado por (1111). As operações lógicas são executadas bit a bit. Por exemplo, se a operação AND é aplicada às entradas A = 1011 e B = 0110, resulta F = 0010.

O resultado de uma operação de comparação é apresentado na saída A = B. Para tanto, deve-se executar a operação A - B - 1 com $C_n = 1$; se as duas entradas são iguais, a saída A = B toma o valor 1. Esta saída tem a configuração *open-colector* para que seja possível implementar uma função *wired-and* dentre duas ou mais saídas deste tipo pertencentes a diferentes ULAs, quando se deseja expandir a largura da palavra a ser manipulada.

A saída C_{n+4} representa o sinal de vai-um do último bit da palavra. Ela pode ser usada para propagar o vai-um para o próximo estágio quando não há preocupação com a velocidade do circuito. O sinal C_{n+4} também pode ser usado em conjunto com a saída A = B para indicar as condições A>B e A<B.

OBSERVAÇÃO: Na realidade, em se tratando de um circuito combinatório, a ULA 74181 pode trabalhar com operandos representados em lógica positiva ou negativa. A Tabela II acima mostra o significado dos bits de seleção de operações quando se considera o uso de lógica positiva. Consulte o manual (*datasheet*) do componente para o caso do uso de lógica negativa.

Nesta experiência adotaremos a convenção de lógica positiva, portanto desconsidere eventuais referências ao uso de lógica negativa nos pinos da ULA no Altera Quartus II.

1.10. Um Fluxo de Dados Simples utilizando a ULA 74181

A figura 1.13 mostra um fluxo de dados simples utilizando o circuito integrado 74181. De uma maneira geral, este circuito é encontrado como a base do fluxo de dados de vários microprocessadores.

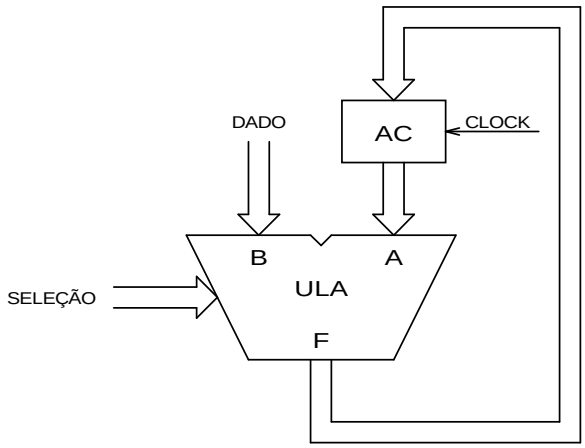

Figura 1.13 - Fluxo de dados de um circuito simples utilizando o 74181.

Neste caso, todas as operações da ULA são executadas entre o conteúdo do acumulador (AC) e o conteúdo da via (DADO). Assim, por exemplo, uma operação de soma de dois números **X + Y** é executada seguindo-se os seguintes passos:

Passo	S_3	S_2	S_1	S_0	M	C_n	B	Comentário
1	0	0	1	1	1	d*	d*	AC ← 0
2	1	0	1	0	1	d*	X	AC ← X
3	1	0	0	1	0	1	Y	AC ← X + Y

* OBS: **d** significa que o valor não importa (do inglês "*don't care*").

Após cada passo deve-se gerar um pulso no sinal CLOCK para forçar a cópia de saída da ULA no acumulador.

Operações mais complexas poderão ser executadas com outras sequencias de operações simples como aquelas relacionadas na Tabela III. De uma maneira geral, estas operações simples podem ser classificadas em operações lógicas e aritméticas e operações de transferência de registradores.

2. PARTE EXPERIMENTAL

2.1. Especificação do Projeto

O circuito a ser desenvolvido na parte experimental inclui uma via de dados *tri-state*, contendo uma ULA de 4 *bits*, registradores bidirecionais, portas de entrada e portas de saída. A figura 2.1 abaixo apresenta um diagrama de blocos de parte do fluxo de dados do circuito a ser implementado.

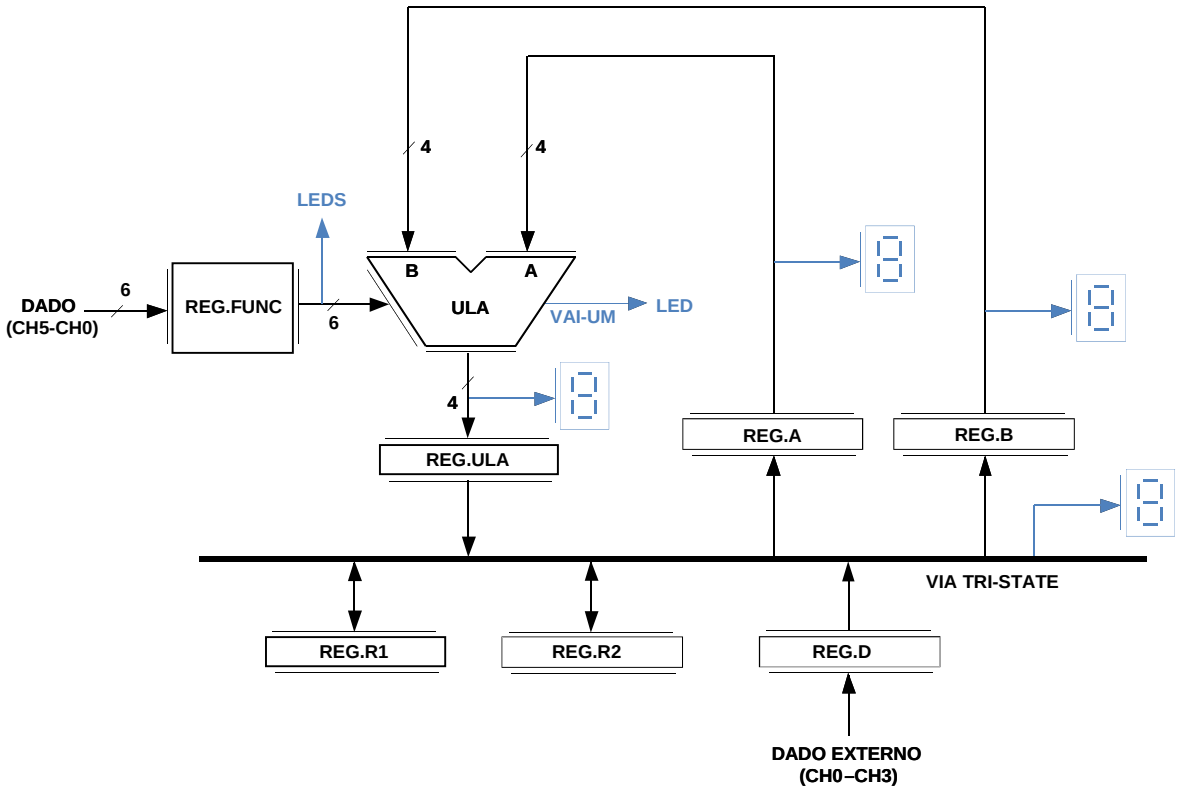

Figura 2.1 – Diagrama de blocos de parte do circuito de via de dados com ULA.

A função da ULA é determinada pela entrada de dado especificada pelas chaves CH0 a CH6 e armazenada no registrador REG.FUNC. Esta função é realizada sobre os dados provenientes dos registradores REG.A e REG.B e o resultado pode ser armazenado no registrador REG.ULA.

A saída do registrador REG.ULA é ligada a uma via de dados *tri-state*, onde vários outros registradores estão conectados. O registrador de dados REG.D é usado para inserir um valor novo ao circuito proveniente das chaves CH0 a CH3. Valores podem ser armazenados e posteriormente acessados via registradores REG.R1 e REG.R2 (estes registradores podem ser usados para guardar valores para uso posterior). E finalmente, os registradores REG.A e REG.B também são ligados a via.

Todos os registradores ligados na via de dados contêm sinais de controle de suas respectivas saídas *tri-state* para permitir a correta transferência de dados entre eles, conforme apresentado na seção 1.4. A tabela 2.1 mostra os sinais de controle dos principais componentes do fluxo de dados.

Para facilitar a implementação da transferência de dados entre registradores, alguns cuidados devem ser levados em conta no projeto:

- Somente uma saída deve estar habilitada por vez na via de dados;
- A saída do registrador que irá colocar seu dado na via deve estar estável antes dela ser copiada em outro registrador;
- Para realizar a cópia, os registradores devem possuir um sinal de habilitação da carga de valor. Use este sinal ao invés de usar a entrada de *clock* nesta operação.

Como suporte para as atividades de teste e depuração, convém monitorar alguns sinais do circuito da via de dados com ULA. Na figura 2.1, são apresentados alguns pontos interessantes, a saber: valor da função da ULA, saída da ULA e do bit de vai-um, entradas da ULA e valor presente na via de dados. Outros sinais podem ser acrescentados nesta lista.

Tabela 2.1 – Sinais de controle.

componente	sinal de controle	descrição
REG.FUNC	LOAD_RFC	Carrega valor no registrador.
ULA	ULA_FCT	Código de operação da ULA de 6 *bits*.
REG.ULA	CLEAR_RULA	Zera valor do registrador.
	LOAD_RULA	Habilita cópia de dado para o registrador.
	EN_RULA	Habilita saída do registrador para colocar dado na via *tri-state*.
REG.RA	CLEAR_RA	Zera valor do registrador.
	LOAD_RA	Habilita cópia de dado para o registrador.
REG.RB	CLEAR_RB	Zera valor do registrador.
	LOAD_RB	Habilita cópia de dado para o registrador.
REG.D	CLEAR_RD	Zera valor do registrador.
	LOAD_RD	Carrega valor no registrador.
	EN_RD	Habilita saída do registrador para colocar dado na via *tri-state*.
REG.R1	CLEAR_R1	Zera valor do registrador.
	LOAD_R1	Habilita cópia de dado para o registrador.
	EN_R1	Habilita saída do registrador para colocar dado na via *tri-state*.
REG.R2	CLEAR_R2	Zera valor do registrador.
	LOAD_R2	Habilita cópia de dado para o registrador.
	EN_R2	Habilita saída do registrador para colocar dado na via *tri-state*.

A operação do circuito deve seguir os seguintes comandos (Tabela 2.2):

1. Entrada de dados
2. Transferência de dados entre registradores
3. Seleção de função da ULA
4. Execução da função

O comando deve ser escolhido nas entradas Cm1 e Cm0, e depois executado ao acionar o botão B1.

Tabela 2.2 – Comandos do circuito.

Comando	Cm_1Cm_0
Entrada de dados	**00**
Transferência de dados entre registradores	**01**
Seleção de função da ULA	**10**
Execução da função	**11**

A entrada de dados no circuito deve ser feita através do registrador de dados REG.RD. O valor deve ser especificado nas chaves CH0 a CH3.

Os dados podem ser transferidos de um registrador a outro. Para isto, as identificações dos registradores de origem e destino da transferência devem ser especificadas nas chaves CH0 a CH3.

CH3	CH2	CH1	CH0
d	**d**	**o**	**o**

Cada registrador é identificado como mostrado na tabela 2.3.

Tabela 2.3 - Codificação dos registradores nas instruções de transferência de dados.

Código de registrador	Registrador origem	Registrador destino
00	RULA	RA
01	RD	RB
10	R1	R2
11	R2	R2

A função a ser executada pela ULA deve ser selecionada com as chaves CH0 a CH5, conforme especificado na documentação do circuito integrado 74181.

O comando de execução de função armazena o resultado da função executada pela ULA no registrador REG.ULA.

Um exemplo de uso deste circuito é apresentado a seguir:

1. Selecionar comando de entrada de dados;
2. Inserir o valor 5 no registrador D;
3. Selecionar comando de transferência de dados;
4. Transferir conteúdo do registrador D para o registrador A;
5. Selecionar comando de entrada de dados;
6. Inserir o valor 2 no registrador D;
7. Selecionar comando de transferência de dados;
8. Transferir conteúdo do registrador D para o registrador B;
9. Selecionar comando de seleção de função;
10. Selecionar função de soma na ULA;
11. Selecionar o comando de execução de função;
12. Armazenar resultado no registrador da ULA;
13. Selecionar comando de transferência de dados;
14. Transferir conteúdo do registrador da ULA para o registrador R1;
15. Selecionar comando de entrada de dados;
16. Inserir o valor 10 no registrador D;
17. Selecionar comando de transferência de dados;
18. Transferir o conteúdo do registrador D para o registrador A;
19. Selecionar comando de transferência de dados;
20. Transferir o conteúdo do registrador R1 para o registrador B;
21. Selecionar comando de seleção de função;
22. Selecionar a função da operação XOR na ULA;
23. Selecionar o comando de execução de função;
24. Executar a operação armazenando o resultado no registrador da ULA.

Estes 24 passos anteriores executam a expressão $(5+2)\oplus 10 = 13$.

O projeto do circuito de via de dados com ULA deve conduzir as seguintes atividades:

1. Projeto do fluxo de dados com o uso de componentes MSI disponíveis no Laboratório Digital (vide figura 2.1) além de decodificadores de endereço para o controle dos registradores;
2. Desenvolvimento da unidade de controle para a execução dos comandos;
3. Síntese do circuito para a placa de desenvolvimento FPGA.

A unidade de controle deve levar em conta as operações a serem implementadas e as restrições de temporização para a transferência de dados entre os registradores usando a via de dados *tri-state*. Por exemplo, se a transferência envolver a cópia de dados do registrador REG.D (origem) para o registrador REG.A (destino), o sinal EN_RD deve ser ativado antes e, somente no período de clock seguinte, o sinal LOAD_RA deve ser ativado. Convém ressaltar que o sinal EM_RD deve permanecer ativado durante a realização da operação de transferência. A máquina de estados usada para modelar a unidade de controle deve levar em conta estas restrições.

A figura 2.2 ilustra a interface externa do circuito de via de dados com ULA.

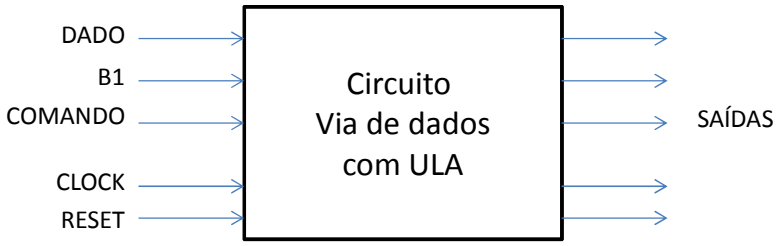

Figura 2.2 – Interface do circuito.

2.2. Montagem Experimental do Projeto

O circuito projetado deve ser implementado na placa DE2 da Altera com a seguinte designação mínima de sinais:

- DADOS : chaves SW0 a SW5;
- COMANDO : chave SW16 e SW17;
- B1 : botão KEY0;
- CLOCK : clock de 27 MHz ou de 50 MHz;
- RESET : botão KEY3.

O conjunto **mínimo** de sinais de depuração é composto por:

- Função da ULA: *leds* vermelhos LEDR0 a LEDR5;
- Vai-um da ULA: *led* verde LEDG0;
- Saída do registrador da ULA: *display* HEX0;
- Saída do registrador A: *display* HEX1;
- Saída do registrador A: *display* HEX2;
- Conteúdo da via de dados: *display* HEX3.

Outros sinais do projeto podem usar os outros recursos disponíveis da placa. O planejamento deve conter uma tabela com a designação destes sinais adicionais.

DICA: Para os testes do circuito, sugere-se que sinais extras de teste e depuração sejam monitorados em *leds* e *displays* na placa de desenvolvimento. Estes sinais devem monitorar partes do fluxo de dados e também o estado da máquina de estados da unidade de controle.

a) Programe o projeto sintetizado na placa DE2 no Laboratório Digital e teste o circuito.
b) Apresente os casos de teste programados para avaliar o correto funcionamento.
c) Relate quaisquer ocorrências experimentais.

2.3. Modificação do Projeto

d) (OPCIONAL) Ao término da implementação do projeto base do circuito de via de dados com ULA, uma pequena modificação será solicitada. O professor da turma irá apresentar a especificação da modificação.
e) Ao terminar o projeto, programe a placa DE2 no Laboratório Digital e teste o circuito modificado.

3. BIBLIOGRAFIA

- FREGNI, E.; SARAIVA, A. M. **Engenharia do Projeto Lógico Digital: Conceitos e Prática**. Editora Edgard Blücher, 1995.
- SIGNETICS. **TTL Logic Data Manual**. Signetics, 1982.
- FREGNI, E.; LANGDON JR., G.G. **Projeto de Computadores Digitais**". 2ª Edição, Editora Edgard Blücher, 1976.
- TOCCI. R. S.; LASKOWSKI, L. P. **Microprocessors and Microcomputers** - Hardware and Software. Prentice-Hall, 2ª Edição.
- TOCCI, R.L.; WIDMER, N.S.; MOSS, G.L. **Digital Systems: Principles and Applications**. 11th ed., Prentice-Hall, 2011.

4. MATERIAL DISPONÍVEL

Circuitos Integrados TTL:

- Portas lógicas: 7400, 7402, 7404, 7408, 7432, 7486.
- Componentes MSI: 7474, 74126, 74138, 74139, 74173, 74181.

5. EQUIPAMENTOS NECESSÁRIOS

- 1 painel de montagens experimentais.
- 1 fonte de alimentação fixa, 5V ± 5%, 4A.
- 1 osciloscópio digital.
- 1 multímetro digital.
- 1 gerador de pulsos.
- 1 computador compatível com IBM-PC com software Altera Quartus II.
- 1 placa de desenvolvimento FPGA DE2 da Altera com o dispositivo Altera Cyclone II EP2C35F672C6

Histórico de Revisões

B.J.S./2001 – revisão
E.T.M./2004 – revisão
E.T.M./2005 – revisão
E.T.M./2006 – revisão
E.T.M./2008 – revisão
E.T.M./2012 – revisão e adaptação da parte experimental.