

F541 Aula 1

Sinais analógicos e digitais, sistema decimal de numeração, sistemas de bases diferentes.

1-INTRODUÇÃO:

A interação e a compreensão de qualquer processo físico, implica em medições de grandezas físicas que caracterizam este processo. As grandezas físicas medidas constituem uma curva que evolui no tempo, por exemplo $f(t)$, geralmente contínua, cuja intensidade é um número decimal associado a uma unidade de medida. Os valores associados são chamados analógicos. Exemplos são os sinais fornecidos por voltímetros, termômetros, detectores de intensidade de luz, medidores de acidez (pH-metros), pressão, etc.

A informatização permite que observemos essas curvas em um computador, em tempo real (para fins práticos), e possamos usar as inúmeras ferramentas de análise. Mas, para isso temos que transformar as medidas em tempo real, em medidas digitais.

Para a digitalização da curva $f(t)$ a intensidade do sinal deve ser "amostrada" e "segurada" (sample and hold circuit) por um tempo durante o qual é convertida à base dois e armazenada na memória do processador, repetindo-se o processo a cada amostragem do sinal.

O sistema binário é usado porque fornece um grande número de vantagens associadas à lógica formal, armazenamento, processamento de sinais, transmissão, etc.

2- O SISTEMA BINÁRIO, SISTEMA OCTAL E HEXADECIMAL.

Conversão de um número decimal em binário.

Um número decimal é convertido em binário dividindo-o por 2 consecutivamente e juntando o quociente final aos restos consecutivos das divisões, ex. $13/2=6$ sobra 1. $6/2=3$ sobra 0. $3/2=1$ sobra 1. O número será 1101; ou seja $13_{10}=1101_2$. Para encontrar o valor decimal do binário basta fazer as potências de 2, multiplicar pelo valor 1 ou 0 e somar os resultados: $1101_2=1x2^3+1x2^2+0x2^1+1x2^0=8+4+1=13$

Existem algumas codificações de binários úteis para escrever menos "uns" e "zeros".

- 1) Representação **hexadecimal do binário**. O número em hexadecimal tem 16 algarismos e não 10 ou 2. Usa a representação do número binário separando-o de 4 em 4 casas binárias. Associa um dos 16 números: 0, 1, 2... 8, 9, A, B, C, D, E, F, (equivalente aos números 10, 11, 12, 13, 14, 15 respectivamente).

Ex: $59_{10}=0011\ 1011_2=3B_{16}$

- 2) Representação **octal do binário**.

Usa a representação binária de 3 em 3 casas. Ex: $59_{10}=111\ 011_2=73_8$

- 3) **Codificação BCD** ou Binary-coded decimal.

Cada grupo de 4 bits representa um número **decimal**, de 0 a 9. O número $59_{10}=3B_{16}$

Ex: $0101\ 1001_{BCD}=59_{10}$.

Ilustração de operações com binários:

Números negativos.

Para fazer a representação binária de um número negativo, o código mais usado é o complemento de 2:

$7=0111$ então $-7=1000+1=1001$. Observe que $7+(-7)=0$ basta somar $0111+1001=0000$, vai um.

$32=0010\ 0000$ portanto $-32=1101\ 1111$ $+1=1110\ 0000$

Portanto $0010\ 0000+$

$$\begin{array}{r} 1110\ 0000 \\ 0000\ 0000 \\ \hline \end{array}$$

Somas e multiplicações são feitas como com decimais

$0001*0010=0010$;

$0011*0010=0110$;

$0011*0011=0011+0110=1001$.

Estados lógicos

Os números 1 e 0 em cada casa binária são associados a Estados lógicos:



High = Verdadeiro=1;
Low = Falso=0.

Por exemplo, Um conjunto de variáveis lógicas (digitais), A, B, C, D, pode refletir uma *função lógica* F(A,B, C, D). Os valores das variáveis e das funções *só podem ser 1 ou 0*.

2- Famílias Lógicas .

Para executar as *funções lógicas*, as tecnologias mais conhecidas são: TTL (transistor-transistor-logic) e CMOS. (Complementar Metal oxide semiconductor, (complem.Field effect transistors)). A TTL foi uma das primeiras tecnologias desenvolvidas. Os chips TTL são mais rápidos, mas gastam mais energia, relativamente á técnica CMOS. A CMOS é a tecnologia atualmente usada. Os chips CMOS ficaram mais rápidos e econômicos.

Os chips de ambas as tecnologias podem ser alimentados com o mesmo valor de tensão, 5.0V. Porém, apresentam 1s e 0s em diferentes tolerâncias de valores, Veja abaixo:

(-0.25V a 0.8V) = "0" (2.0V a 5.2V) = "1" Input	 U1A TTL	(0.0V a 0.4V) = "0" (2.4V a 5.0V) = "1" Output
(-0.25V a 1.5V) = "0" (3.50V a 5.5V) = "1" Input	 U1A CMOS	(0.0V a 0.4V) = "0" (4.5V a 5.0V) = "1" Output

Note que a imunidade ao ruído de tensão do CMOS é maior do que a do TTL. Porém o ruído em circuitos digitais, também depende de outros fatores como taxa de subida e descida de sinais, estabilidade, etc.

3- Portas lógicas (Gates)

A funções mais comuns são: AND, OR, NOT, EXOR, NAND, NOR, conforme a Tabela abaixo.

Lembre que nas operações de OR (ou, soma) $C = A + B$, AND (interseção, produto) $D = A \cdot B$, NOT (inversão, complemento) $E = \bar{D}$. Os elemento A, B, C, D, E só podem valer 1 ou 0.

A operação EXOR é conhecida como meia soma.

A	B	AND	NAND	OR	NOR	EXOR
0	0	0	1	0	1	0
0	1	0	1	1	0	1
1	0	0	1	1	0	1
1	1	1	0	1	0	0

Para a construção das funções lógicas usamos a *álgebra de Boole* e as leis de *De Morgan*, discutidas em classe.

Experimental:

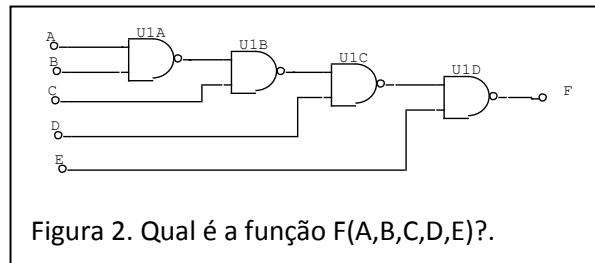
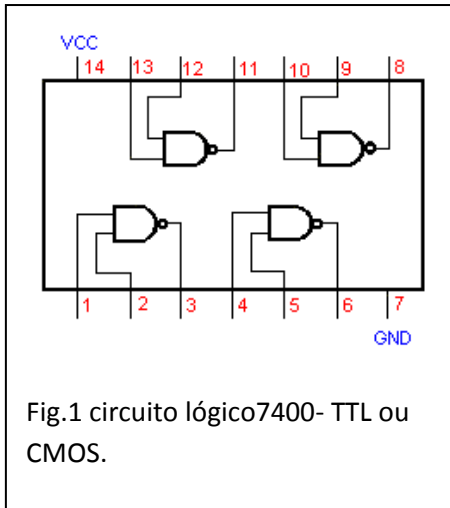
1- Procure na internet pelo manual do IC 74C00, um CMOS, que copia o TTL-7400. Coloque num proto-board e alimente-o com 5V.

Utilize o manual para identificar os pinos de alimentação e das portas NAND.

4- Monte a função lógica da Figura 2 usando o TTL 7400 da Figura 1.

Use um LED com uma resistência de 150Ω em série, para identificar o estado lógico das portas/funções. Simule as entradas, conectando-as a GND (0) ou a VCC (1), ou seja, 0,0 ou 5,0 Volts.

Descubra a “tabela verdade” da função F(A,B,C,D,E), Fig. 2. Explique o funcionamento. Mostre a função de saída.

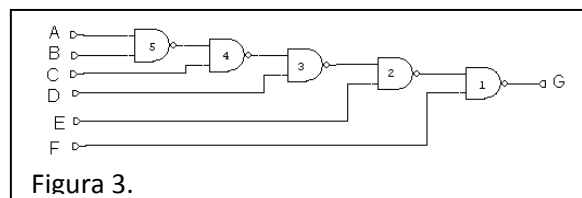


- 5- Implemente um AND com 2 NANDs.
- 6- Implemente um EXOR com 4 Nands.

7- NANDs de nível par e impar. Na lógica digital combinacional o estado presente da variável de saída é função do estado presente das variáveis de entrada. Conforme a fig. 3

$$\overline{\overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} \cdot \overline{E} \cdot \overline{F}} = G, \text{ ou:}$$

$((\overline{A} + \overline{B}) \cdot \overline{C} + \overline{D}) \cdot \overline{E} + \overline{F} = G$ (use os teoremas de De Morgan.). Observe que os **NANDs de nível par funcionam como ANDs. Que os NANDs de nível impar funcionam como OR complementando as entradas.**



8- Usando NANDS do IC 74C00, implemente o circuito da Figura 4. Mostre a função F(ABCDE) usando os conceitos de NANDs de nível par e impar.

