Capítulo 6

Macrofunções e Aplicações

Os primeiros projetos digitais consistiam em especificar as interconexões entre as portas individuais, como fizemos até agora. Entretanto, percebeu-se logo que muitos problemas complexos podem ser divididos em subproblemas comuns. Passou-se, então, como uma prática de projeto o paradigma de **projeto hierárquico**, onde um circuito é dividido em vários blocos funcionais ou **macrofunções** – contadores, somadores, multiplicadores, comparadores, multiplexadores, registros de deslocamento, etc – e, então, realizado através da interconexão entre as implementações já existentes destas (macro)funções.

Para facilitar o projeto e a implementação dos sistemas digitais, muitos fabricantes oferecem pastilhas que integram todas as portas necessárias para realizar tais funções. É comum utilizar os seguintes termos para referenciar a escala da integração de uma pastilha:

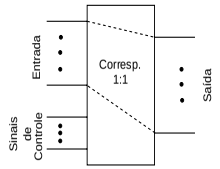
SSI (*Small Scale Integration*)	< 10 portas
MSI (*Medium Scale Integration*)	$10 - 100$ portas
LSI (*Large Scale Integration*)	$100 - 10.000$ portas
VLSI (*Very Large Scale Integration*)	> 10.000 portas

Neste capítulo apresentaremos os projetos de algumas macrofunções mais usuais.

6.1 Decodificadores

Neste capítulo chamamos **decodificador** (*decoder*) um circuito combinacional de m entradas e n saídas, $m < n$, no qual usualmente somente uma ou um grupo das n saídas é ativado para uma específica combinação das m

entradas.



Esta propriedade é bastante útil nas aplicações como:

1. selecionar/ativar exatamente uma linha na ocorrência de um determinado padrão na entrada, e
2. acionar *displays* de LED.

6.1.1 Decodificadores Binários

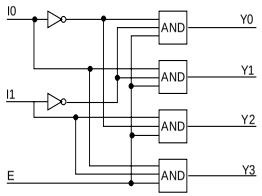
São também conhecidos como decodificador n–to–2^n. Estes decodificadores possuem n entradas cujas 2^n possíveis combinações ativam distintamente as $m = 2^n$ saídas, como ilustra o seguinte decodificador 2–to–4 (o sinal E é o sinal de controle para habilitar a função de decodificação):

E	I1	I0	Y3	Y2	Y1	Y0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Por inspeção, é fácil derivar as seguintes funções para cada saída

$$\begin{aligned} Y3 &= I1 I0 E \\ Y2 &= I1 I0 \prime E \\ Y1 &= I1 \prime I0 E \\ Y0 &= I1 \prime I0 \prime E \end{aligned}$$

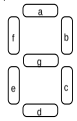
que podem ser implementadas com portas lógicas AND e NOT



Quando uma ou mais linhas na entrada não afetam a saída, como o conversor do BCD para BCD 3 em excesso no Exemplo 3.14, podemos introduzir o estado "don't-care" na tabela-verdade para otimizar o circuito.

6.1.2 Decodificadores de 7 segmentos

Muitos equipamentos digitais provêm diodos emissores de luz (LEDs) ou display de cristal líquido para apresentar informação aos usuários. Esta informação pode ser dados numéricos (dígitos) ou caracteres alfanuméricos. Um dos dispositivos mais simples e populares para apresentar dígitos de 0 a 9 (base decimal) ou de 0 a F (base hexadecimal) é o *display* de 7 segmentos



Através do controle da corrente em cada um dos 7 segmentos do display, a, b, c, d, e, f e g, podemos obter um dos padrões dos 16 dígitos.

Um decodificador de 7 segmentos é um circuito combinacional que aciona em função da entrada (em BCD) as linhas dos 7 segmentos de modo a gerar o padrão correspondente no *display*. Por exemplo, se a entrada for 0101 (5), os segmentos a, f, g, c e d acenderão enquanto os outros segmentos ficarão apagados.

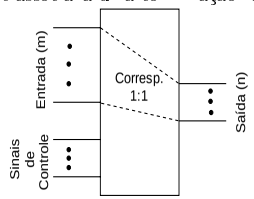
Baseado neste princípio de funcionamento, é fácil sintetizar a lógica deste decodificador na seguinte tabela-verdade:

E	D	C	B	A	a	b	c	d	e	f	g
0	X	X	X	X	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
1	0	0	1	0	1	1	0	1	1	0	1
1	0	0	1	1	1	1	1	1	0	0	1
1	0	1	0	0	0	1	1	0	0	1	1
1	0	1	0	1	1	0	1	1	0	1	1
1	0	1	1	0	0	0	1	1	1	1	1
1	0	1	1	1	1	1	1	0	0	0	0
1	1	0	0	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	0	0	1	1
1	1	0	1	0	0	0	0	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	1
1	1	1	0	0	0	1	0	0	0	1	1
1	1	1	0	1	1	0	0	1	0	1	1
1	1	1	1	0	0	0	0	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0

O estado de cada segmento é uma função das quatro variáveis de entrada, D, C, B e A. Tal função pode ser derivada facilmente com uso de mapas de Karnaugh.

6.2 Codificadores

Exatamente oposto a um decodificador, um **codificador** é um circuito combinacional com m entradas e n saídas, $m > n$, no qual para uma ou um grupo das m entradas é associada uma combinação das n saídas.



Ele é utilizado na

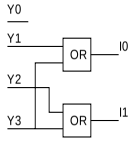
1. conversão de um sinal de entrada num nível de prioridade, e

2. conversão de um sinal de acionamento numa palavra-código, por exemplo, associar a uma tecla do teclado uma palavra-código de ASCII.

O princípio de projeto destes codificadores é análogo ao dos decodificadores, invertendo somente as entradas com as saídas. No caso de um codificador 2^n–to–n, com $n = 2$, teremos a seguinte tabela-verdade:

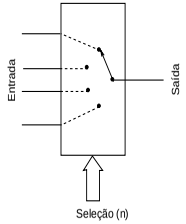
E	Y3	Y2	Y1	Y0	I1	I0
0	X	X	X	X	0	0
1	0	0	0	X	0	0
1	0	0	1	X	0	1
1	0	1	0	X	1	0
1	1	0	0	X	1	1

cujo diagrama lógico pode ser:



6.3 Multiplexadores e Demultiplexadores

Um **multiplexador** (ou **mux**) é um circuito combinacional que seleciona, através dos n sinais de seleção, um (ou um grupo de) sinal dentre os 2^n sinais de entrada e o passa para a saída. Portanto, é também conhecido como **chave digital**.

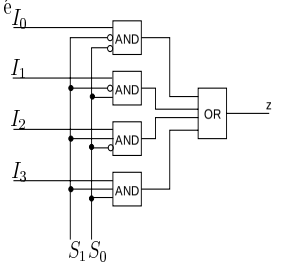


A expressão lógica de um multiplixador pode ser uma soma dos termos de produto onde aparecem n variáveis de seleção e uma das 2^n variáveis de entrada. No caso de um multiplexador com 4 entradas, são necessárias duas

variáveis S_1S_0 para selecioná-las. A função de saída é então

$$z = f(S_1, S_0, I_0, I_1, I_2, I_3) = S_1\prime S_0\prime I_0 + S_1\prime S_0 I_1 + S_1 S_0\prime I_2 + S_1 S_0 I_3,$$

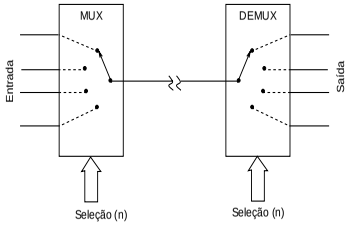
cujo diagrama lógico é



Entre as diversas aplicações dos multiplexadores nos sistemas digitais, podemos destacar

1. roteamento de dados, permitindo o compartilhamento de um mesmo recurso por diferentes dados,
2. sequenciamento de operações,
3. conversão paralelo-serial, e
4. implementação de funções lógicas.

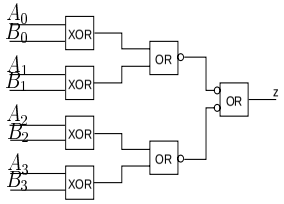
Um **demultiplexador** (ou **demux**), por sua vez, é um circuito combinacional cuja função é inversa a do multiplexador. Ele aceita uma entrada e a repassa para uma das m saídas de acordo com a lógica da seleção. Ele é usualmente utilizado em conjunto com um multiplexador, como na transmissão de dados a longa distância.



6.4 Comparadores

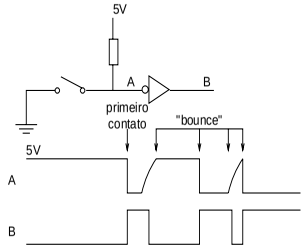
Comparadores são circuitos combinacionais que comparam dois valores binários, bit a bit, e geram uma saída indicando se eles são iguais ou não. As portas XOR são os seus principais constituintes, uma vez que estas podem ser vistas como comparadores de 1 bit (a saída desta porta é igual a 0 quando as duas entradas são iguais; e 1, quando são diferentes).

O seguinte diagrama lógico mostra a estrutura de um comparador de 4 bits.

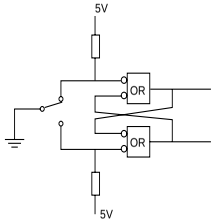


6.5 Debouncers

Uma das formas mais usuais para "entrar os dados" nos sistemas digitais é através de chaves/botões, que ao serem acionados podem oscilar e criar um trem de pulsos "falsos", como mostra o esquema abaixo. Este comportamento é conhecido como **contact bounce**. Tal *bounce* pode ser problemático para alguns circuitos.



E como solução a tal problema utiliza-se o circuito **debouncer** para amenizar o efeito de oscilação. Os circuitos **debouncer** fazem uso dos elementos biestáveis, como ilustra o seguinte esquema.



Na seção 4.1 vimos que os dispositivos biestáveis, exceto no ponto de metaestabilidade, sempre se estabilizam em dois estados bem distintos: 0 e 1. Pequenos pulsos espúrios, usualmente, não tem duração suficiente para levá-los de um estado para outro, devido aos atrasos na propagação destes pulsos através das portas. Estes pulsos somente causam pequenas variações na tensão de saída, mas não chegam a alterar o valor lógico do circuito.

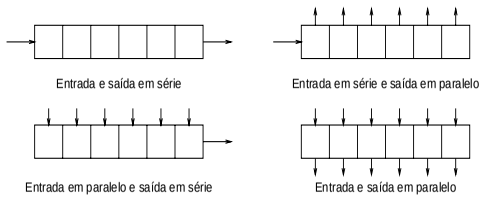
6.6 Registros de Deslocamento

Os **registros de deslocamento** (*shift-registers*) são componentes mais encontrados nos sistemas digitais. Eles são caracterizados por apresentarem a funcionalidade de deslocamento e de armazenamento (memória). Portanto, são amplamente utilizados como

1. memórias temporárias,
2. "deslocadores de dados", e
3. conversores dos formatos de dados paralelo/serial.

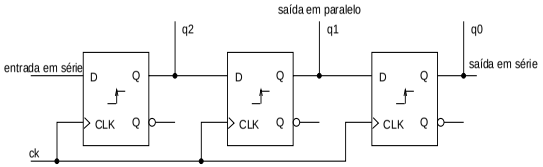
Os registros de deslocamento são máquinas sequenciais síncronas (de *Moore*) constituídas por um conjunto de *flip-flops* ligados em cascata (saída de um ligada na entrada do outro). Eles podem ser classificados em:

- entrada e saída em série,
- entrada em série e saída em paralelo,
- entrada em paralelo e saída em série, e
- entrada e saída em paralelo.

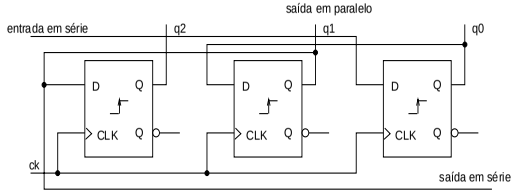


Só com o esquema de deslocamento esboçado, podemos facilmente implementar estes registros com uso de *flip-flops* D.

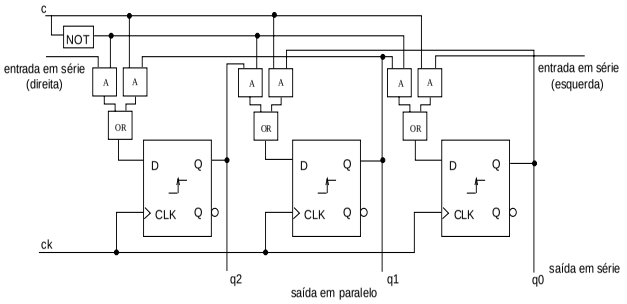
- um registro de deslocamento à direita de 3 bits



- um registro de deslocamento à esquerda de 3 bits

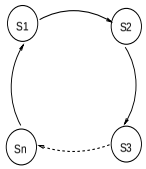


- talvez um pouco mais complexo, um registro de deslocamento à esquerda ou à direita (controle $c = 1$) de 3 bits, com as seguintes equações de transicão: $q_2^* = d_2 \cdot c\prime + q_1 \cdot c$, $q_1^* = q_2 \cdot c\prime + q_0 \cdot c$ e $q_0^* = q_1 \cdot c\prime + d_0 \cdot c$



6.7 Contadores

Um **contador** é um circuito sequencial cujo diagrama de estado corresponde a um ou mais ciclos. O número de estados em cada ciclo define o **mod** de uma relação de equivalência, isto é, um contador de módulo n contém, no mínimo, um ciclo com n estados.



Esta característica permite que o tal circuito seja aplicado nas seguintes situações:

1. conta o número de pulsos (de entrada) recebidos e armazena em código binário o número de contagem,
2. provê um trem de pulsos cuja frequência é menor que a frequência dos pulsos de entrada,
3. provê uma sequência de padrões binários.

Existe uma variedade de implementações de contadores.

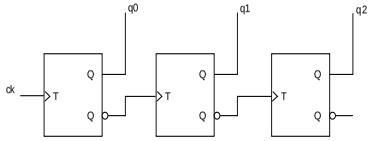
Os contadores podem ser **síncronos** ou **assíncronos**. Nos contadores **síncronos** os elementos de memória mudam de estado simultâneamente,

enquanto nos contadores **assíncronos** os elementos de memória mudam de estado em instantes distintos. Estes podem ser implementados usando-se a saída de um elementos de memória como o sinal de relógio do seguinte, e assim sucessivamente.

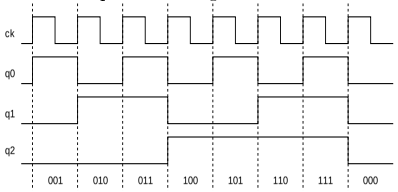
Pode-se ainda classificar os contadores em **progressivos** (*up*) e **regressivos** (*down*). Note-se que um contador *up* pode ser transformado em *down* (e vice-versa) simplesmente invertendo-se os sinais que vão para o próximo estágio.

6.7.1 Contadores *Ripple* Binários

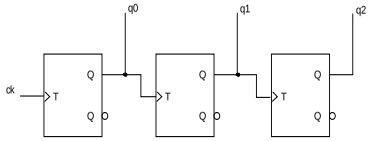
É o mais simples dos contadores. Pode ser implementado com uso de *flip-flops* T. Ele é conhecido como um contador **binário**, porque a sequência de estados é a mesma da sequência de números binários. O termo *ripple* refere-se ao fato de que o sinal (de relógio) não afeta todos os flip-flops direta e simultâneamente. Este sinal é propagado ao longo da cadeia dos flip-flops.



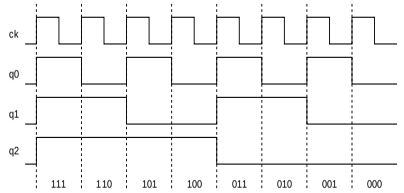
Através do diagrama de tempo é fácil constatar que a contagem do contador é progressiva (*up*), de zero até um valor máximo. Note que o estágio seguinte muda de estado quando o estágio anterior muda do nível 1 para 0.



Caso quisermos fazer uma contagem regressiva (*down*), devemos inverter o estado do seguinte *flip-flop* quando o estágio anterior muda do nível 0 para 1.

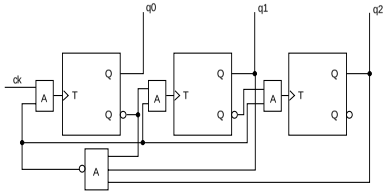


Observe através do diagrama de tempo o comportamento deste contador módulo-8.



Vale ainda observar que o período do sinal de saída de um *flip-flop* é sempre 2 vezes maior em relação ao período do sinal de saída do estágio anterior. Portanto, estes circuitos são também muito utilizados como **divisores de frequência**.

Podemos ainda **inibir** uma contagem até um estado pré-definido e manter a máquina neste estado, através da "desabilitação do sinal" T. Tais contadores são conhecidos **contadores auto-inibidores**. O circuito seguinte é o de um contador progressivo auto-inibidor no estado 011.

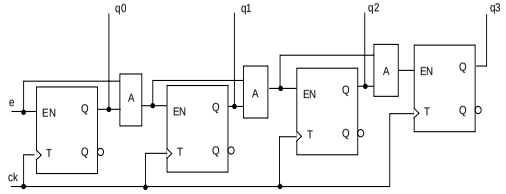


6.7.2 Contadores Síncronos Binários

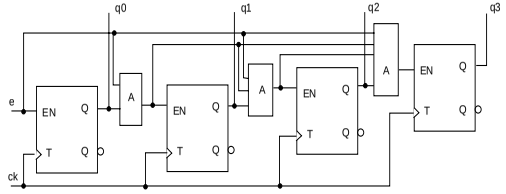
Embora o contador *ripple* seja o contador binário mais simples (requer o menor número de portas lógicas), ele é o mais lento (o último *flip-flop* de uma cascata de n *flip-flops* só recebe o sinal de relógio depois de nt_0, se considerarmos que o tempo de propagação em cada *flip-flop* seja t_0) e é muito sensível a pulsos espúrios.

Os contadores binários síncronos, por sua vez, tem todos os seus flip-flops conectados a um mesmo sinal de relógio, portanto a saída é válida logo depois de um atraso t_Q que corresponde ao atraso de propagação de um flip-flop, no lugar de n *flip-flops*. O controle da mudança de estado é feito através do sinal de habilitação.

Quando o sinal de habilitação se propaga serialmente ao longo dos flip-flops, dizemos que ele é **serial**. Nesta configuração, cada *flip-flop* só inverte o seu estado, se e somente se, o sinal de habilitação é ativado e todos os *flip-flops* que o antecedem estiverem com o valor 1.



Outro esquema de realização dos contadores síncronos binários é o **paralelo**, no qual o sinal de habilitação é ligado diretamente aos pinos de habilitação de cada *flip-flop*, eliminando o atraso que pode ocorrer na propagação deste sinal do bit menos significativo para o mais significativo.



É fácil ver que também pode **inibir** a contagem destes contadores em qualquer estado utilizando o mesmo esquema apresentado para os contadores *ripple*.

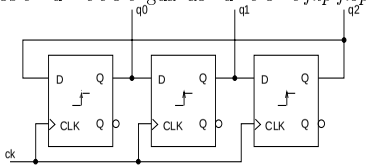
6.7.3 Contadores Cíclicos

Os contadores binários são conhecidos também como contadores de **estágio mínimo**, por requerem o menor número de *flip-flops* para representar um dado conjunto de estados num mesmo ciclo. Muitas vezes, porém, é mais econômico implementar um contador com uso de registros de deslocamento,

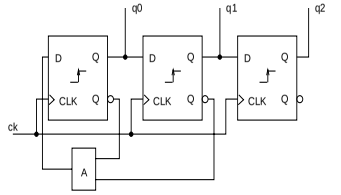
conectando a saída do último *flip-flop* com a entrada do primeiro *flip-flop*. Daí o nome de **contadores cíclicos**. Exemplos destes contadores são os **contadores em anel** e **contadores de Johnson**. Estes contadore tem a grande vantagem de possuirem, caso for necessário, um circuito de decodificação das saídas extremamente simples.

Contadores em Anel

Nos contadores em anel, somete um *flip-flop* contém o valor 1 em cada instante. Este 1 é deslocado ciclicamente entre os *flip-flops*. Portanto, o número de estados válidos em um ciclo é igual ao número de *flip-flops*.

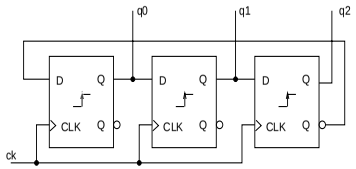


Um dos problemas neste circuito é que caso o sistema entrar num estado inválido devido a ruídos, o circuito ficará gerando indefinidamente saídas inválidas. Para evitar isso, pode-se conectar a entrada do primeiro *flip-flop* com a saída da porta AND das saídas negadas de todos os *flip-flops* menos o último, constituindo assim um contador **auto-corretor**.



6.7.4 Contadores Johnson

Um contador Johnson pode ser obtido a partir de um contador em anel simplesmente trocando a conexão de realimentação, conectando a saída negada do último *flip-flop* com a entrada do primeiro *flip-flop*.



Note que um contador Johnson com m *flip-flops* contém sempre um ciclo com $2m$ estados. Para evitar que o sistema projetado para operar num ciclo caía num outro ciclo, pode-se aplicar a mesma estratégia apresentada para contadores em anel.

6.8 Máquinas (Binárias Sequenciais) Lineares

Estas máquinas são caracterizadas por poderem ser descritas através de uma **função de transferência**. As máquinas sequenciais lineares são amplamente utilizados nas aplicações fundamentadas em Teoria de Corpos Finitos ou Corpo de Galois (GF), desenvolvida pelo matemático francês Ëvariste Galois (ver o site `http://www-history.mcs.st-and.ac.uk/~history/Mathematicians/Galois.html`). como:

1. implementação de circuitos de embaralhador e desembaralhador, para facilitar o sincronismo,
2. geradores pseudo-aleatório, e
3. implementação de codificador e decodificador de códigos corretores de erro.

Veremos que o esquema básico dos circuitos destas máquinas contém três elementos:

1. somador mod 2,
2. conexões, e
3. atrasador unitário (*flip-flop* D).

Note que a soma módulo 2 pode ser implementada com uso de portas lógicas XOR, pois

$$\begin{aligned} 0 \oplus 0 &= 0 \\ 0 \oplus 1 &= 1 \\ 1 \oplus 1 &= 0 \end{aligned}$$

Para estudar máquinas lineares é conveniente representar uma sequência de entrada ou saída

$$c_0, c_1, c_2, \cdots$$

como uma série de potência

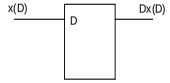
$$c(D) = c_0 + c_1 D + c_2 D^2 + \cdots,$$

também conhecida como **transformada da sequência**.

Ao multiplicarmo uma transformada $x(D) = x_0 + x_1 D + x_2 D^2 + \cdots$ por D, obteremos

$$Dx(D) = x_0 D + x_1 D^2 + x_2 D^3 + \cdots$$

que corresponde à sequência $x(D)$ deslocada de um atraso no tempo. Em termos de circuito, equivale a passar a sequência por um *flip-flop* D, desde que o estado inicial do *flip-flop* seja 0.

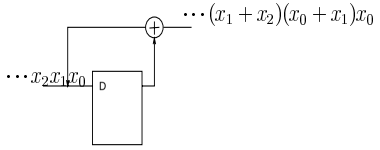


Comparando com a multiplicação de um polinômio $p(X)$ por X, no qual todos os coeficientes p_i são avançados de uma posição, temos neste caso todos os coeficientes atrasados de uma posição. Podemos dizer, portanto, que $D = X^{-1}$ em termos de "operações entre polinômios".

Se adicionarmos Da(D) com x(D) em módulo 2, teremos

$$y(D) = (1+D)x(D) = x_0 + (x_0 + x_1)D + (x_1 + x_2)D^2 + (x_2 + x_3)D^3 + \cdots$$

Dizemos que a(D) é multiplicado por (1+D). Esta expressão pode ser realizado por seguinte circuito sob a condição de que o estado inicial do *flip-flop* seja 0.



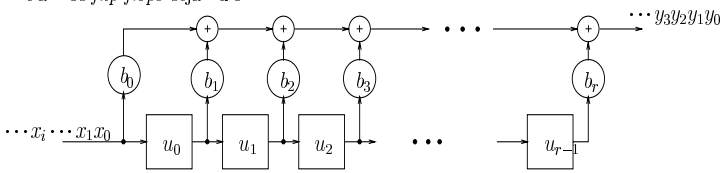
À expressão (1+D) chamamos de **função de transferência** do circuito.
Seja agora

$$b(D) = b_0 + b_1 D + b_2 D^2 + \cdots + b_{r-1} D^{r-1} + b_r D^r$$

a função de transferência, que aplicada sobre a transformada da sequência de entrada resulta em

$$\begin{aligned} y(D) = b(D)x(D) \;=\; & b_0x_0 + (b_1x_0 + b_0x_1)D + \\ & +(b_2x_0 + b_1x_1 + b_0x_2)D^2 + \cdots + \\ & +(b_rx_{i-r} + \cdots + b_1x_{i-1} + b_0x_i)D^i + \cdots, \end{aligned}$$

que é realizável através do seguinte circuito, se assumirmos que o estado inicial dos *flip-flops* seja nulo



Observe que inicialmente temos na saída exatamente b_0x_0. No primeiro pulso do relógio, o coeficiente x_0 é deslocado para u_0 e temos na saída $b_1x_0 + b_0x_1$. No segundo pulso, x_1 é deslocado para u_0 e x_0 para u_1. Portanto, temos na saída $b_2x_0 + b_1x_1 + b_0x_2$. E assim sucessivamente até x_0 chegar no u_{r-1}. A partir daí a saída será $b_rx_{i-r} + \cdots + b_1x_{i-1} + b_0x_i$ para todos x_i na entrada.

Exemplo 6.1 *Dada uma sequência de entrada* $11010111001\cdots$ *e uma função de transferência* $b(D) = 1 + D + D^2 + D^3 + D^6$*, qual é a sequência de saída?*

A transformada da sequência de entrada é

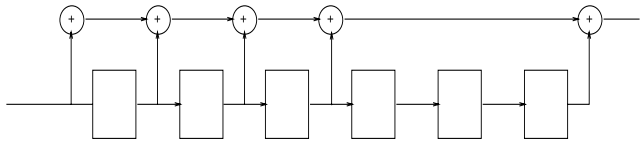
$$x(D) = 1 + D + D^3 + D^5 + D^6 + D^7 + D^{10} + \cdots$$

que multiplicada pela função de transferência b(D) resulta-se em

$$y(D) = b(D)x(D) = 1 + D^3 + D^8 + D^9 + \cdots$$

Esta é a tansformada da sequência $1001000011\cdots$.

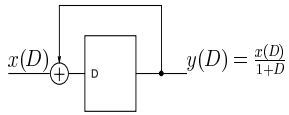
Um circuito que realiza a função de transferência deste exemplo é



Se dividirmos x(D) por (1+D) obteremos

$$y(D) = \frac{x(D)}{D+1} = x_0 + (x_0 + x_1)D + (x_0 + x_1 + x_2)D^2 + \cdots$$

que pode ser realizado pelo seguinte circuito, considerando que o estado inicial do *flip-flop* seja 0.

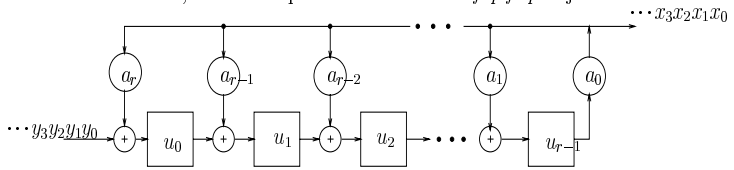


$\frac{1}{1+D}$ é uma **função de transferência** do circuito, cuja entrada é $x(D)$ e a saída $y(D)$.

Consideremos uma expressão mais genérica

$$\frac{1}{a(D)} = \frac{1}{a_0 + a_1 D + a_2 D^2 + \cdots + a_{r-1} D^{r-1} + a_r D^r}.$$

Esta função de transferência pode ser implementada com uso do seguinte circuito com r atrasos, assumindo que o estado inicial dos *flip-flops* seja nulo.



Note que, enquanto x_0 não chegar na unidade u_{r-1}, a saída é 0. Podemos considerar os r zeros como r atrasos na "resposta do circuito". A partir daí, temos primeiro na saída a_0 e o seguinte estado para as duas entradas do XOR que antecede cada *flip-flop*:

	u_0	u_1	u_2	$\cdots$	u_{r-1}
Entrada 1	x_r	x_{r-1}	x_{r-2}	$\cdots$	x_1
Entrada 2	a_ra_0	$a_{r-1}a_0$	$a_{r-2}a_0$	$\cdots$	a_1a_0

Note que na entrada 2 das portas XOR temos os coeficientes do polinômio-divisor multiplicado pelo a_0. Assim quando ocorrer um pulso de relógio os *flip-flops* passarão para o seguinte estado

u_0	u_1	u_2	$\cdots$	u_{r-1}
$x_r \oplus a_r a_0$	$x_{r-1} \oplus a_{r-1} a_0$	$x_{r-2} \oplus a_{r-2} a_0$	$\cdots$	$x_1 \oplus a_1 a_0$

Teremos a seguinte configuração nas entradas das portas XOR antes do segundo pulso de relógio:

	u_0	u_1	u_2	$\cdots$	u_{r-1}
Entrada 1	x_{r+1}	$x_r \oplus a_r a_0$	$x_{r-1} \oplus a_{r-1} a_0$	$\cdots$	$x_1 \oplus a_1 a_0$
Entrada 2	$a_r a_1$	$a_{r-1} a_1$	$a_{r-2} a_1$	$\cdots$	$a_1 a_1$

Observe que embora pareçam complexas as expressões, elas só podem assumir um dos dois valores lógicos: 0 e 1. O circuito pode ser interpretado como um divisor. A saída é o quociente da divisão e o conteúdo dos *flip-flops* é, de fato, o resultado parcial da subtração entre o dividendo e o divisor multiplicado por um termo do polinômio-quociente. Em cada pulso de relógio, este resultado parcial é "deslocado" e um novo termo do dividendo é agregado, como ocorre na execução de uma divisão, passo a passo.

Exemplo 6.2 *Dada uma sequência de entrada* $11010111001\cdots$ *e uma função* $a(D) = 1 + D + D^3$*, qual é a sequência de saída considerando que a função de transferência seja* $\frac{1}{a(D)}$*?*

A transformada da sequência de entrada é

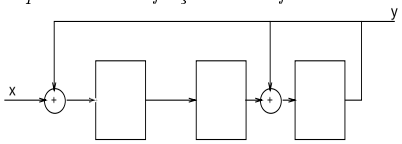
$$x(D) = 1 + D + D^3 + D^5 + D^6 + D^7 + D^{10} + \cdots$$

que dividida pela função de transferência a(D) resulta-se em

$$y(D) = \frac{x(D)}{a(D)} = 1 + D^5 + D^7 + \cdots$$

Esta é a tansformada da sequência $10000101\cdots$.

Um circuito que realiza esta função de transferência é



Finalmente, consideremos uma máquina (binária) linear descrita através de uma expressão na forma

$$\frac{Y}{X} = \frac{1 + b_{k-1}D + b_{k-2}D^2 + \cdots + b_0 D^k}{1 + a_{k-1}D + a_{k-2}D^2 + \cdots + a_0 D^k}, \tag{6.1}$$

onde X e Y são, respectivamente, as sequências de entrada e de saída. É fácil ver que as funções de transferência $b(D)$ e $\frac{1}{a(D)}$ são casos particulares desta expsressão.

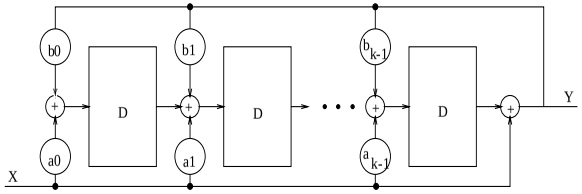
Podemos utilizar uma das três seguintes técnicas para implementar a eq.(6.1). Estas técnicas diferem basicamente na forma como manipulamos a expressão:

- por realização direta,
- por realização em cadeia, e
- por realização com registros de deslocamento.

Na realização direta, a entrada X é levada diretamente às portas XOR com base no fato de que, após algumas manipulações algébricas, podemos reescrever a eq.(6.1) em

$$Y = X + D(b_{k-1}X + a_{k-1}Y + D(b_{k-2}X + a_{k-2}Y + \cdots + D(b_0 X + a_0 Y)\cdots))$$

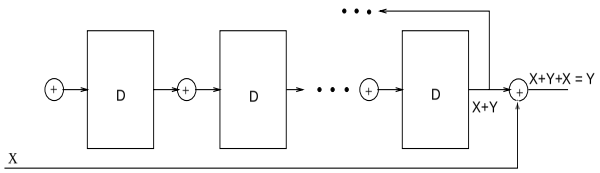
Supondo que as portas lógicas XOR (somadores binários) tenham 3 entradas, o circuito segue o seguinte esquema



Um outro rearranjo da eq.(6.1) nos leva à expressão

$$Y + X = D((b_{k-1}X + a_{k-1}Y) + D((b_{k-2}X + a_{k-2}Y) + \cdots + D((b_0 X + a_0 Y))\cdots)),$$

que, dependendo dos valores a_j e b_j, podemos conectar uma entrada da porta XOR com a linha X ($a_j = 0$ e $b_j = 1$), o ponto Y ($a_j = 1$ e $b_j = 0$) ou o ponto X+Y ($a_j = 1$ e $b_j = 1$) do esquema abaixo

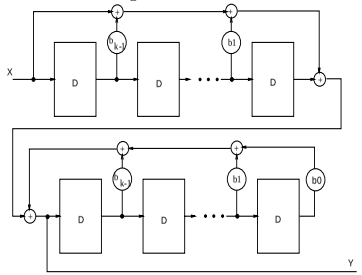


A este esquema de implementação denominamos **realização em cadeia**. Note que neste esquema só portas XOR com duas entradas são utilizadas na implementação.

Uma terceira forma de rearranjar a eq.(6.1) é

$$Y = (1+b_{k-1}D+b_{k-2}D^2+\cdots+b_0D^k)X+(1+a_{k-1}D+a_{k-2}D^2+\cdots+a_0D^k)Y.$$

Para esta expressão podemos utilizar dois registros de deslocamento na implementação: um para o polinômio do numerador e o outro para o polinômio do denominador da função de transferência, uma vez que a porta XOR não aparece entre os *flip-flops* de um mesmo registro, mas apenas na realimentação e entre os dois registros.



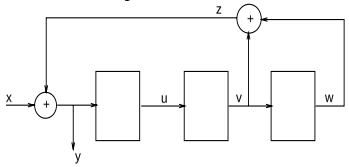
6.8.1 Embaralhadores e Desembaralhadores

Embaralhamento (*scrambling*) refere à técnica de eliminar longas sequências de zeros ou uns, tornando-as em sequências com ocorrências mas aleatórias de zeros e uns. Esta técnica é muito utilizada na Comunicação Digital na qual sequências longas de zeros e uns podem causar, entre outros problemas, a perda de sincronismo do sistema. Um circuito capaz de embaralhamento é conhecido como **circuito embaralhador**.

Com uso do Cálculo Probabilístico (propriedade de autocorrelação) pode-se mostrar que uma função de transferência da forma $\frac{Y}{X} = \frac{1}{P(D)}$ é capaz

de embaralhar qualquer sequência de entrada X, desde que ela não seja uma sequência inteiramente de 1s ou de 0s. Portanto, pode-se realizar um embaralhador como uma máquina linear.

Exemplo 6.3 *Considere o seguinte circuito que a sequência de entrada seja* $1000001101\cdots$. *Determine a sequência de saída* Y.



A função de transferência do circuito é

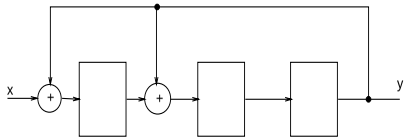
$$\frac{Y}{X} = \frac{1}{1+0.D+1.D^2+1.D^3} = \frac{1}{1+D^2+D^3}$$

Considerando que o estado inicial dos flip-flops seja 0, então o conteúdo dos flip-flops para diferentes instantes n é:

n	y	u	v	w
0	1	0	0	0
1	0	1	0	0
2	1	0	1	0
3	1	1	0	1
4	1	1	1	0
5	0	1	1	1
6	1	0	1	1
7	0	1	0	1
8	1	0	1	0
9	0	1	0	1

Ou seja, gera-se a sequência de saída $Y = 01011101010\cdots$

Observe ainda que o circuito dado é, em termos da sequência de saída, equivalente ao circuito



A cada circuito embaralhador, de entrada X e saída Y, existe um **circuito desembaralhador**, de entrada Y e saída Z, para recuperar o sinal original. Este circuito deve ter uma função de transferência tal que

$$\frac{Y}{X}\frac{Z}{Y} = D^j,$$

ou seja,

$$\frac{Z}{Y} = D^j \frac{X}{Y}.$$

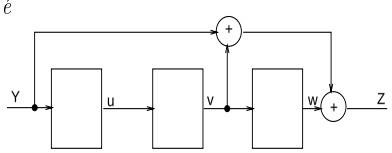
Particularmente, se o sinal recuperado Z não for deslocado em relação ao sinal original X, então

$$\frac{Z}{Y} = \frac{X}{Y}.$$

Exemplo 6.4 *Um circuito cuja função de transferência é*

$$\frac{Z}{Y} = 1 + D^2 + D^3.$$

pode ser o desembaralhador do circuito do exemplo 6.3. O diagrama lógico deste circuito é



Se inserirmos a sequência $Y = 10111010100\cdots$ *com os flip-flops inicialmente em 0, teremos a seguinte sequência de saída* z_n*:*

n	y	u	v	w	z
1	1	0	0	0	1
2	0	1	0	0	0
3	1	0	1	0	0
4	1	1	0	1	0
5	1	1	1	0	0
6	0	1	1	1	0
7	1	0	1	1	1
8	0	1	0	1	1
9	1	0	1	0	0

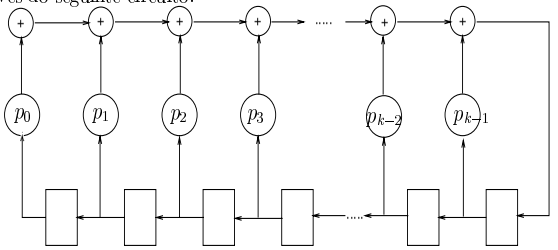
que é exatamente igual à sequência original $X = 100000110\cdots$.

6.8.2 Geradores Pseudo-aleatórios

Há um teorema que garante que as soluções não-nulas de y_{i+j} de uma equação de recorrência que relaciona o estado corrente y_{i+k} com os k estados anteriores y_{i+j},

$$\sum_{j=0}^{k} p_j y_{i+j} = 0 \text{ ou } y_{i+k} = -\sum_{j=0}^{k-1} p_j y_{i+j} \tag{6.2}$$

são cíclicas. O ciclo contém k estados, onde k é o menor inteiro positivo que $p(x) = \sum_{j=0}^{k} p_j x^j$ divide $x^k - 1$. A equação de recorrência pode ser realizada através do seguinte circuito:

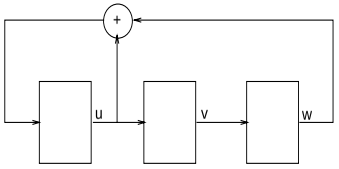


Quando $p(x)$ é um **polinômio primitivo** de grau m, o menor inteiro n para o qual ele divide $x^n - 1$ é $n = 2^m - 1$ (seção A.3). Daí conclui-se que se a **função de recorrência** é um polinômio primitivo de grau m, então o circuito contém um ciclo com $2^n - 1$ estados. O estado faltante é o (*all-0 state*). Tal circuito é também conhecido como **gerador de pseudo-aleatórios** ou **máquina de sequência máxima**.

Para facilitar a síntese destes circuitos, podemos utilizar a notação de transformada

$$Y.P(D) = 0.$$

Exemplo 6.5 *Considere um gerador pseudo-aleatório construído com a função de recorrência* $P(D) = 1 + D + D^3$ *(polinômio primitivo de grau 3 no corpo* $GF(2)$*)*

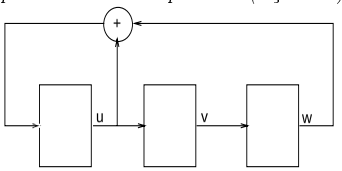


Os flip-flops passam pelos seguintes estados ciclicamente:

n	u	v	w
0	*0*	*0*	*1*
1	*0*	*1*	*0*
2	*1*	*0*	*1*
3	*0*	*1*	*1*
4	*1*	*1*	*1*
5	*1*	*1*	*0*
6	*1*	*0*	*0*

No Exemplo A.16 mostra-se uma técnica algébrica para determinar esta sequência cíclica.

Note-se ainda que um gerador construído com a **função recíproca** *de $P(D)$, $P^*(D) = D^3P(D^{-1})$, contém também (uma outra) máxima sequência, excluindo o all-0 state. Este resultado é esperado, uma vez que o recíproco de um polinômio primitivo é também primitivo (seção A.3).*



Tabelas de polinômios primitivos podem ser encontradas em várias publicações (ver por exemplo o site `http://ks.fernuni-hagen.de/~rieke/primitiv/welcome.html.en`). Aqui, só por curiosidade, listamos para cada valor de n, de 1 a 32, um polinômio primitivo.

n	p(x)
1,2,3,4,6,7,15,22	$1+x+x^n$
5,11,21,29	$1+x^2+x^n$
10,17,20,25,28,31	$1+x^3+x^n$
9	$1+x^4+x^n$
23	$1+x^5+x^n$
18	$1+x^7+x^n$
8	$1+x^2+x^3+x^4+x^n$
12	$1+x+x^4+x^6+x^n$
13	$1+x+x^3+x^4+x^n$
14,16	$1+x^3+x^4+x^5+x^n$
19,27	$1+x+x^2+x^5+x^n$
24	$1+x+x^2+x^7+x^n$
26	$1+x+x^2+x^6+x^n$
30	$1+x+x^2+x^{23}+x^n$
32	$1+x+x^2+x^{22}+x^n$

Uma aplicação bastante difundida dos geradores pseudo-aleatórios é a geração de padrões de teste para circuitos lógicos.

Apêndice A

Estruturas Algébricas

Este apêndice tem como objetivo introduzir alguns conceitos elementares de estruturas algébricas que dão um subsídio melhor para compreender as máquinas lineares. Oportunamente, vocês verão que muitas estruturas algébricas são fundamentais em Comunicações Digitais e Teoria de Códigos.

A palavra álgebra é de origem árabe ("al-jabr"), que significa "reunificação". Ela é largamente utilizada em Matemática para referenciar assuntos que tratam da manipulação e da análise dos conjuntos discretos de objetos.

Um **conjunto** é uma coleção arbitrária de objetos, ou elementos. Um conjunto pode ser finito (p. ex., o conjunto de disciplinas do curso da Engenharia Elétrica), infinito mas contável (p. ex., o conjunto de números naturais) e infinito e não-contável (p. ex., o cojunto de números reais). A **cardinalidade** de um conjunto é o número de elementos contidos num conjunto.

A.1 Grupos

Uma estrutura algébrica relativamente simples, com um conjunto não-vazio $\mathcal{G}$ munido de apenas uma operação binária $\%$, isto é, a cada par de elementos $x, y \in \mathcal{G}$ está associado um elemento $x\%y \in \mathcal{G}$, é o **grupo**, para o qual valem os seguintes axiomas:

1. Associatividade: $\forall x, y, z \in \mathcal{G}, (x\%y)\%z = x\%(y\%z)$;
2. Existência de um elemento identidade: $\exists e \in \mathcal{G}$, tal que $x\%e = e\%x = x, \forall x \in \mathcal{G}$; e

3. Existência de um elemento inverso: $\exists -x \in \mathcal{G}$ tal que $x\%(-x) = (-x)\%x = e, \forall x \in \mathcal{G}$.

   Diz-se que o grupo $\mathcal{G}$ é **comutativo** (ou **abeliano**, se ele goza adicionalmente a seguinte propriedade:

4. Comutatividade: $x\%y = y\%x, \forall x, y \in \mathcal{G}$.

Exemplo A.1 *O conjunto de inteiros forma um grupo (comutativo) infinito sob a operação de adição (de inteiros), mas não sob a operação de multiplicação (definida no sentido clássico), pois o inverso de cada número não se encontra neste conjunto.*

Exemplo A.2 *O conjunto de matrizes $n \times n$ cujos elementos pertencem ao conjunto de números reais formam um grupo comutativo (infinito) sob adição (de matrizes), mas não sob multiplicação, pois sabemos que a multiplicação de duas matrizes não é comutativa.*

Há um teorema que afirma que os elementos do conjunto $S_m = \{0, 1, 2, 3, \cdots, m-1\}$ formam um grupo comutativo de ordem m sob adição módulo m para qualquer inteiro positivo m (define-se como adição módulo n (ou $mod\ n$) de dois números o resto da divisão módulo n da soma usual destes dois números). O elemento-identidade deste grupo é 0 e o inverso de cada elemento x é $(m-x)$.

Exemplo A.3 *O conjunto $S_2 = \{0, 1\}$ é um grupo comutativo sob adição módulo 2 cujo elemento identidade é 0. O inverso de 0 e 1 são, respectivamente, 0 e 1.*

É possível ainda mostrar que para qualquer inteiro primo p, os elementos do conjunto $Z_p = \{1, 2, 3, \cdots, p-1\}$ constituem um grupo comutativo finito (mais exatamente, de ordem $\{p-1\}$) sob multiplicação módulo n (define-se como multiplicação módulo n de dois números o resto da divisão módulo n do produto usual destes dois números). Note-se que se p não for um inteiro primo, então existe um $m, n \in Z_p$, tal que $1 < m, n < p$ e $mn \equiv 0\ mod\ p$, que não pertence a Z_p. O elemento-identidade do grupo é 1.

Exemplo A.4 *O conjnto $Z_3 = \{1, 2\}$ é um grupo comutativo sob multiplicação módulo 3, cujo elemento-identidade é 1. O inverso de 1 é 1 (pois $1.1 \equiv 1$) e o de 2 é 2 (pois $2.2 \equiv 1$).*

Uma transformação f de um grupo $\mathcal{G}$ num grupo $\mathcal{G}\prime$ é chamado **homomorfismo**, se $f(xy) = f(x)f(y), \forall x, y \in \mathcal{G}$. Neste caso, $\mathcal{G}$ é o núcleo de f e $f(\mathcal{G}) \in \mathcal{G}\prime$ é a imagem da aplicação f.

Exemplo A.5 *Seja $\mathcal{G}$ o grupo de números reais sob adição e seja $\mathcal{G}\prime$ o grupo dos números reais positivos sob multiplicação. A transformação $f : \mathcal{G} \to \mathcal{G}\prime$ definida por $f(x) = 2^x$ é um homomorfismo, porque*

$$f(x+y) = 2^{x+y} = 2^x 2^y = f(x)f(y).$$

Exemplo A.6 *Seja $\mathcal{G}$ o grupo de números complexos não-nulos sob multiplicacão e seja $\mathcal{G}\prime$ o grupo dos números reais não-nulos sob multiplicação. A transformação $f : \mathcal{G} \to \mathcal{G}\prime$ definida por $f(z) =\mid z \mid$ é um homomorfismo, porque*

$$f(z_1 z_2) =\mid z_1 z_2 \mid =\mid z_1 \mid\mid z_2 \mid = f(z_1)f(z_2).$$

A.2 Anéis e Corpos

Seja $\mathcal{A}$ um conjunto não-vazio com duas operações binárias, uma operação de adição (denotada por $+$) e uma operação multiplicação (denotada por $\cdot$). Então, $\mathcal{A}$ é chamado **anel** se $\mathcal{A}$ é um grupo abeliano sob adição, isto é, são verificadas as seguintes propriedades

1. Associatividade da soma: $(x+y)+z = x+(y+z), \forall x, y, z \in \mathcal{A}$;
2. Existência do elemento neutro: $\exists 0 \in \mathcal{A}$ tal que $x+0 = 0+x = x, \forall x \in \mathcal{A}$;
3. Existência de inverso aditivo: $\exists -x \in \mathcal{A}$ tal que $x+(-x) = (-x)+x = 0, \forall x \in \mathcal{A}$;
4. Comutatividade da soma: $x+y = y+x, \forall x, y \in \mathcal{A}$;

e satisfaz ainda os seguintes axiomas:

1. Associatividade do produto: $(x.y).z = x.(y.z), \forall x, y, z \in \mathcal{A}$;
2. Distributividade do produto em relação à soma: $x.(y+z) = x.y + x.z$ e $(x+y).z = x.z + y.z$, $\forall x, y, z \in calA$;

Um anel $\mathcal{A}$ é comutativo, se ele satisfaz ainda a propriedade: $\forall x, y \in \mathcal{A}, x.y = y.x$.

$\mathcal{A}$ é chamado **anel com elemento-identidade**, se existe um elemento não-nulo $1 \in \mathcal{A}$ tal que $\forall x \in \mathcal{A}, x.1 = 1.x = x$.

Uma anel comutativo com elemento-identidade $\mathcal{A}$ é chamado **domínio de integridade**, se $\mathcal{A}$ não tem **divisores de zero**, isto é, se $xy = 0 \Rightarrow x = 0 ou y = 0$.

Exemplo A.7 *O conjunto dos números inteiros é um domínio de integridade (anel de inteiros) com elemento-identidade em relação às operações usuais de adição e multiplicação.*

Exemplo A.8 *Um grupo comutativo (dos inteiros) sob adição módulo p é um anel de inteiros módulo p.*

Um anel comutativo com elemento-identidade $\mathcal{A}$ é chamado **corpo** (ou em inglês, *field*) se $\forall x$(não-nulo) $\in \mathcal{A}$ tem inverso multiplicativo, isto é, existe um elemento $x^{-1} \in \mathcal{A}$, tal que $aa^{-1} = a^{-1}a = 1$.

Note-se que um corpo é, necessariamente, um domínio de integridade, pois, se $xy = 0$ e $x \neq 0$, então $y = 1.y = x^{-1}.x.y = x^{-1}.0 = 0$.

Exemplo A.9 *O conjunto de números racionais, de números reais e números complexos são corpos em relação às operações usuais de adição e multiplicação.*

Corpos de ordem (cardinalidade) finita foram descobertos pelo matemático francês Evariste Galois e são denominados **Corpos de Galois**. Um corpo de Galois de ordem q é usualmente denotado por $GF(q)$.

Exemplo A.10 *Os elementos do conjunto binário $\{0,1\}$ formam um corpo $GF(2)$ as operações lógicas AND ($\cdot$) e XOR ($+$), pois como estas operações correspondem, respectivamente, às operações de adição e de multiplicação módulo 2, elas satisfazem todos os axiomas impostos para um corpo. Portanto, $\{0,1\}$ sob as duas operações lógicas forma um corpo.*

Exemplo A.11 *Os elementos do conjunto $Z_q = \{0, 1, 2, 3, \cdots, p-1\}$, onde p é primo, formam um corpo $GF(q)$ sob adição e multiplicação módulo p. Pelo que já vimos, estes elementos formam um grupo comutativo sob adição módulo p e o seu subconjunto $\{1, 2, 3, \cdots, p-1\}$ forma um grupo comutativo sob multiplicação módulo p. E como adição e multiplicação módulo p são distributivas na aritmética usual, segue-se que Z_q é um corpo.*

Define-se como **ordem de um elemento** β no corpo $GF(q)$ o menor inteiro positivo m ($m \neq 0$) tal que $\beta^m = 1$.

Exemplo A.12 *Seja um corpo* $GF(7)$ *formado pelos elementos* $\{0,1,2,3,4,5,6\}$ *sob operações de adição e multiplicação módulo 7, ou seja*

$+$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

$\cdot$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Por inspeção, a ordem do

1. *elemento 1:* $m = 1$*;*
2. *elemento 2:* $m = 3$ *(1, 2,* $2^2 = 4$, $2^3 = 1$*);*
3. *elemento 3:* $m = 6$ *(1, 3,* $3^2 = 2$, $3^3 = 6$, $3^4 = 4$, $3^5 = 5$, $3^6 = 1$*);*
4. *elemento 4:* $m = 3$ *(1, 4,* $4^2 = 2$, $4^3 = 1$*);*
5. *elemento 5:* $m = 6$ *(1, 5,* $5^2 = 4$, $5^3 = 6$, $5^4 = 2$,$5^5 = 3$,$5^6 = 1$*);*
6. *elemento 6:* $m = 2$ *(1, 6,* $6^2 = 1$*).*

Um elemento do corpo $GF(q)$ é chamado **elemento primitivo** se a sua ordem é $(q-1)$. Uma conclusão importante é que a partir deste elemento, pode-se "gerar" todos os elementos não-nulos do corpo $GF(q)$. E ainda mais, **todos os elementos não-nulos podem ser representados como potências deste elemento primitivo**.

Para **construir corpos (finitos) com** p^m **elementos, sendo** p **primo e** $m > 1$, veremos na seção seguinte que podemos utilizar os polinômios primitivos sobre o corpo $GF(p)$. Veremos ainda que operações diferentes das operações módulo são necessárias para satisfazer todas as propriedades impostas aos corpos.

A.3 Anéis de Polinômios

Seja $\mathcal{K}$ um corpo. Um polinômio f sobre $\mathcal{K}$ é uma sequência infinita de elementos de $\mathcal{K}$ na qual todos, exceto um número finito deles, são zero:

$$f = (\cdots, 0, a_n, \cdots, a_1, a_0) ou f(x) = a_n x^n + \cdots + a_1 x^1 + a_0$$

O elemento a_k é chamado k-ésimo coeficiente de f. Se n é o maior intiero para o qual $a_n \neq 0$, então dizemos que o **grau** de f é n, e o representamos por grau $f = n$.

Se g é outro polinômio sobre $\mathcal{K}$, digamos

$$g = (\cdots, 0, b_m, \cdots, b_1, b_0),$$

então a **soma** $f + g$ é o polinômio obtido adicionando os coeficientes correspondentes. Isto é, se $m \leq n$, então

$$f + g = (\cdots, 0, a_n, \cdots, a_m + b_m, \cdots, a_1 + b_1, a_0 + b_0).$$

Exemplo A.13 *A soma de dois polinômios,* $(x^5 + x^2)$ *e* $(x^2 + x + 1)$*, no corpo* $GF(2)$ *resulta em*

$$(x^5 + x^2) + (x^2 + x + 1) = x^5 + 2x^2 + x + 1 = x^5 + x + 1.$$

Além disso, o produto fg é o polinômio

$$fg = (\cdots, a_n b_m, \cdots, a_1 b_0 + a_0 b_1, a_0 b_0),$$

isto é, o k-ésimo coeficiente c_k de fg é dado pela expressão

$$c_k = \sum_{i=0}^{k} a_i b_{k-i} = a_0 b_k + a_1 b_k - 1 + \cdots + a_k b_0.$$

A coleção de todos os polinômios $a_0 + a_1 x + a_2 x^2 + \cdots + x^n$, $a_i \in GF(q)$ e $n \in N$, é de particular interesse na pesquisa de Códigos Algéricos. Utilizamos a notação $GF(q)[x]$ para denotá-los.

O conjunto dos polinômios sobre um corpo $\mathcal{K}$, sob as operações de adição e multiplicação, forma um anel comutativo com elemento-identidade e sem divisores de zero, isto é, um **domínio de integridade**.

Sejam f e g polinômios sobre um corpo $\mathcal{K}$ com $g \neq 0$. Então, existem polinômios q e r tais que

$$f = qg + r,$$

onde ou $r = 0$ ou grau $r <$ grau g, se $r \neq 0$.

Um polinômio $f(x)$ é **irredutível** no corpo $GF(q)$, se $f(x)$ não pode ser fatorado no produto de polinômios de grau menor no anel de polinômios $GF(q)[x]$.

Exemplo A.14 *Considere os seguintes polinômios*

- x^2+1 *é irredutível no anel* $GF(3)[x]$*, mas não no anel* $GF(2)[x]$ *(*$(x+1)(x+1)$*).*
- x^4+x^2+1 *e* $x^{21}+x^2+1$ *são irredutíveis no anel* $GF(2)[x]$*.*

Um polinômio irredutível $f(x) \in GF(q)[x]$ de grau m é dito **primitivo**, se o menor inteiro positivo n para o qual $f(x)$ divide $x^n - 1$ é $n = q^m - 1$.

Exemplo A.15 *Considere os seguinte polinômios:*

- $xcfp^3 + x + 1$ *é primitivo no anel* $GF(2)[x]$*, pois é divisor de* $x^7 - 1$ *(*$2^3 - 1 = 7$*).*
- $x^4 + x + 1$ *é primitivo no anel* $GF(2)[x]$*, pois é divisor de* $x^{15} - 1$ *(*$2^4 - 1 = 15$*).*
- $x^4+x^3+x^2+x+1$ *é irredutível no anel* $GF(2)[x]$*, mas não é primitivo, pois ele é fator de* $x^5 - 1$*.*

Uma propriedade interessante de um polinômio primitivo de grau m $f(x) \in GF(q)$ é que o seu recíproco dado por $x^m f(\frac{1}{x})$ também é primitivo.

Outra propriedade importante é que **todas as raízes de um polinômio primitivo $p(x)$ no corpo $GF(q)$ tem a mesma ordem. Esta ordem é** $q^m - 1$, pois sendo $p(x)$ o divisor de $x^{q^m-1} - 1$, então $\alpha^{q^m-1} - 1 = 0$ ou $\alpha^{q^m-1} = 1$. Pode-se mostrar, por contradição, que $q^m - 1$ é o menor inteiro positivo para o qual $\alpha^{q^m-1} = 1$.

Seja $p(x) = x^m + a_{m-1}x^{m-1} + \cdots + a_1 x + a_0$ o polinômio primitivo no $GF(q)[x]$. Se α é a raíz de $p(x)$, ou seja, $p(\alpha) = \alpha^m + a_{m-1}\alpha^{m-1} + \cdots + a_1\alpha + a_0 = 0$. Então,

$$\alpha^m = -a_{m-1}\alpha^{m-1} + - \cdots - a_1\alpha - a_0$$

Note-se que para α^i, $i \geq m$, podemos reescrever a expressão de α^i em função das potências $i < m$ de α. Assim, podemos associar a cada potência $i \geq m$ de α uma representação polinomial na forma $b_{m-1}\alpha^{m-1} + \cdots + b_1\alpha + b_0$ com os coeficientes $b_i \in GF(q)$. Com isso, temos uma correspondência biunívoca entre as distintas potências de α e o conjunto de polinômios em α com os coeficientes no corpo $GF(q)$ de grau menor ou igual a $(m-1)$.

Pode-se mostrar que as $q^m - 1$ potências de α são os elementos não-nulos do corpo $GF(q^m)$. Daí, chegamos um resultado importante: as **raízes dos polinômios primitivos de grau m no anel $GF(q)[x]$ são elementos primitivos no corpo $GF(q^m)$**. E ainda mais, os elementos do corpo $GF(q^m)$ podem ser representados ou como potências de um elemento primitivo (uma raiz) ou como combinação das potências i, $i < m$, deste elemento primitivo.

Note-se que se considerarmos o conjunto $\{1, \alpha, \alpha^2, \cdots, \alpha^{m-1}\}$ como a "base" da representação polinomial, podemos ainda representar os elementos do corpo $GF(q^m)$ através dos vetores, cujas "coordenadas" pertencem ao corpo $GF(q)$.

Exemplo A.16 *Seja o polinômio primitivo* $p(x) = x^3 + x + 1$ *de grau* $m = 3$ *no anel* $GF(2)[x]$*. Seja* α *a raíz de* $p(x)$*. Então,* $p(\alpha) = \alpha^3 + \alpha + 1 = 0$*, ou* $\alpha^3 = \alpha + 1$*. Todos os elementos não-nulos no corpo* $GF(2^3)$ *podem ser representados como potências consecutivas da raíz* α *(representação exponencial), isto é,* $\{\alpha^0, \alpha^1, \alpha^2, \alpha^3, \cdots, \alpha^6, \alpha^7 = \alpha^0\}$*, ou como combinações das potências (representação polinomial), ou como vetores no corpo* $GF(2)$*:*

Exponencial		*Polinomial*	*Vetorial*
α^0	=	1	*(0,0,1)*
α^1	=	α	*(0,1,0)*
α^2	=	α^2	*(1,0,0)*
α^3	=	$\alpha + 1$	*(0,1,1)*
α^4	=	$\alpha^2 + \alpha$	*(1,1,0)*
α^5	=	$\alpha^3 + \alpha^2 = \alpha^2 + \alpha + 1$	*(1,1,1)*
α^6	=	$\alpha^4 + \alpha^3 = (\alpha^2 + \alpha) + (\alpha + 1) = \alpha^2 + 1$	*(1,0,1)*
0	=	0	*(0,0,0)*

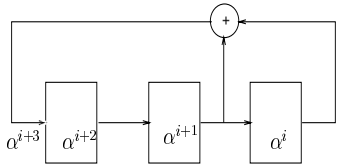
Usando a notação matricial teremos os seguintes elementos no corpo $GF(2^3)$*:*

$$\left[\alpha^0\ \alpha^1\ \alpha^2\ \alpha^3\ \alpha^4\ \alpha^5\ \alpha^6\right] = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Estes elementos são, de fato, as soluções da função de recorrência, pois

$$p(\alpha) = \alpha^3 + \alpha + 1 = \alpha^3 + \alpha^1 + \alpha^0 = 0$$

e $\alpha^i p(\alpha) = 0, \forall i \in N$.



Assim para uma coordenada do vetor (uma linha na matriz), teremos a seguinte sequência (cíclica) de estados ($\alpha^{i+2} + \alpha^{i+1} + \alpha^i$)*:* $001 \rightarrow 010 \rightarrow 101 \rightarrow 011 \rightarrow 111 \rightarrow 110 \rightarrow 100$*, que corresponde exatamente à solução obtida exaustivamente no Exemplo 6.5.*

Usando este tipo de construção de um corpo $GF(q^m)$ podemos construir facilmente as tabelas que definem completamente as operações de adição e multiplicação entre os elementos do corpo $GF(q^m)$. A adição de dois seus elementos é efetuada com uso da representação polinomial, como ilustra o seguinte exemplo.

Exemplo A.17 *A soma de dois elementos* α^3 *e* α^5 *do corpo* $GF(8)$ *é*

$$\alpha^3 + \alpha^5 = (\alpha + 1) + (\alpha^2 + \alpha + 1) = \alpha^2$$

Podemos ainda expressar a multiplicação entre dois dos elementos de um corpo $GF(q^m)$, $\alpha^a = (a_0 + a_1\alpha + \cdots + a_{m-1}\alpha^{m-1})$ e $\alpha^b = (b_0 + b_1\alpha + \cdots + b_{m-1}\alpha^{m-1})$, através da representação polinomial

$$\begin{aligned} \alpha^a.\alpha^b &= \alpha^{(a+b) \bmod (q^m-1)} \\ &= (a_0 + a_1\alpha + \cdots + a_{m-1}\alpha^{m-1})(b_0 + b_1\alpha + \cdots + b_{m-1}\alpha^{m-1}) \bmod p(\alpha) \end{aligned}$$

Exemplo A.18 *Seja* $p(x) = x^2 + x + 1$ *um polinômio primitivo no anel* $GF(2)[x]$*. Seja* α *a raíz de* $p(x)$*. Isso implica que* $\alpha^2 + \alpha + 1 = 0$ *ou* $\alpha^2 = \alpha + 1$*. Os elementos do corpo* $GF(2^2) = GF(4)$ *podem ser representados nas seguintes maneiras:*

Exponencial		*Polinomial*		*Vetorial*
α^0	=	*1*	$\leftrightarrow$	*(0,1)*
α^1	=	α	$\leftrightarrow$	*(1,0)*
α^2	=	$\alpha + 1$	$\leftrightarrow$	*(1,1)*
0	=	*0*	$\leftrightarrow$	*(0,0)*

Ainda mais, com uso das representações exponenciais e polinomiais podemos ver que as seguintes operações foram definidas no corpo $GF(4)$

$+$	*0*	α^0	α^1	α^2
0	*0*	α^0	α^1	α^2
α^0	α^0	*0*	α^2	α
α^1	α^1	α^2	*0*	α^0
α^2	α^2	α^1	α^2	*0*

$\cdot$	*0*	α^0	α^1	α^2
0	*0*	*0*	*0*	*0*
α^0	*0*	α^0	α^1	α^2
α^1	*0*	α^1	α^2	α^0
α^2	*0*	α^2	α^0	α^1

Se rotularmos os elementos α^0*,* α^1 *e* α^2 *por inteiros 1, 2, e 3, respectivamente, podemos criar as seguintes tabelas (verdade) para o corpo* $GF(4)$ *(note-se que é* **diferente** *da adição e multiplicação módulo 4, sob as quais o conjunto* $\{0,1,2,3\}$ *não define um corpo* $GF(4)$*):*

$+$	*0*	*1*	*2*	*3*
0	*0*	*1*	*2*	*3*
1	*1*	*0*	*3*	*2*
2	*2*	*3*	*0*	*1*
3	*3*	*2*	*1*	*0*

$\cdot$	*0*	*1*	*2*	*3*
0	*0*	*0*	*0*	*0*
1	*0*	*1*	*2*	*3*
2	*0*	*2*	*3*	*1*
3	*0*	*3*	*1*	*2*