Capítulo 8

Conversor digital-analógico no STM32F407

8.1. Introdução

Dizemos que os fenômenos do mundo real são analógicos. Em contradição, a tecnologia de processamento digital está cada vez mais presente no nosso dia a dia. Assim, são necessárias interfaces que convertem informações analógicas – do mundo real – para dados digitais que podem ser processados e, em seguida, reconvertidos para a forma analógica. Um dos exemplos mais comuns dessas interfaces são os aparelhos que reproduzem áudio, como *players* de música, convertendo informações digitais (a música codificada digitalmente em um arquivo) para a forma analógica (o som reproduzido pelos alto-falantes).

Muitos componentes e equipamentos eletrônicos não são compatíveis com sinais digitais, necessitando-os na forma analógica. Em geral, processos de conversão de sinais elétricos para grandezas físicas exigem saídas analógicas. Um bom exemplo é o próprio alto-falante de *um smartphone*: Para poder reproduzir um sinal de áudio, o alto-falante precisa ser excitado por meio de um sinal analógico. É necessária então a transformação das cadeias de bits das saídas dos sistemas digitais em sinais analógicos. Esta é a função do conversor digital-analógico, ou conversor DA, ou ainda simplesmente DAC (*Digital to Analog Converter*).

Um conversor digital para analógico transforma um número digital de entrada, com precisão finita de N bits, em uma tensão analógica de saída, linearmente proporcional ao código digital de entrada. Na Figura 1 é apresentado um diagrama funcional da operação de um conversor DA, onde a entrada digital binária é convertida em um valor analógico de tensão de saída.

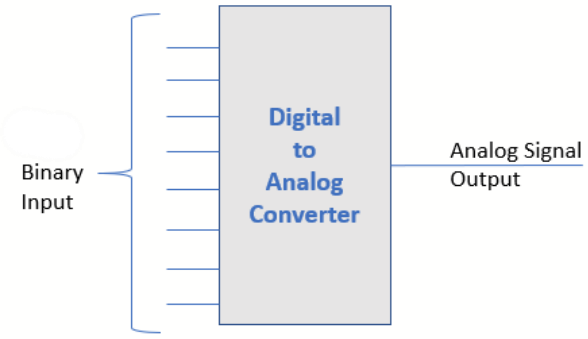


Figura 1 – Diagrama funcional de um conversor DA.

Existem diversas implementações práticas de circuitos para um conversor DA, sendo as redes resistivas ponderadas, redes resistivas R-2R, Sigma-Delta e técnicas de PWM as mais utilizadas. Na Figura 2 é mostrada uma implementação simples de um conversor DA de 4 bits de entrada, construído a partir de um circuito somador inversor baseado em amplificador operacional, conhecida como rede resistiva ponderada.

Suponha que o valor digital a ser convertido em uma tensão analógica tenha 4 bits: $D_3\,D_2\,D_1\,D_0$, com D_3 sendo o bit mais significativo. D_i pode ser 0 ou 1, para i=0, 1, 2 ou 3. Se D_i é 0, a chave correspondente no circuito da Figura 2 é aberta. No caso contrário, a chave correspondente é fechada. As chaves são dispositivos eletrônicos de chaveamento rápido, como transistores.

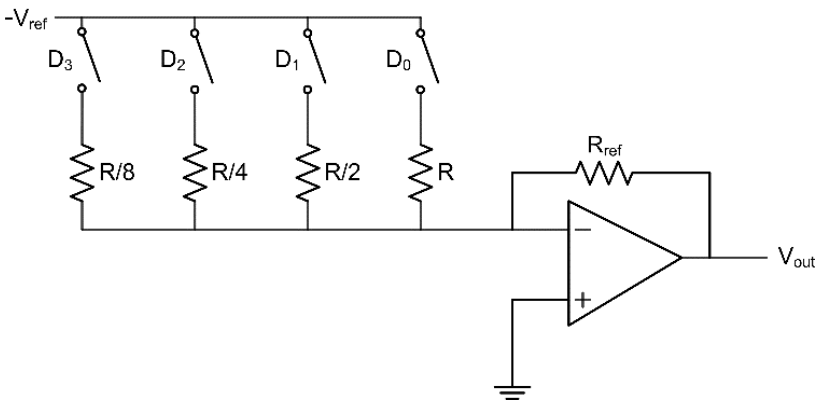


Figura 2 – Implementação básica de um conversor DAC com rede resistiva ponderada.

Para o circuito da Figura 2, a tensão analógica de saída V_{out} pode ser calculada como:

$$V_{out} = V_{ref} * \frac{R_{ref}}{R/_8} D_3 + V_{ref} * \frac{R_{ref}}{R/_4} D_2 + V_{ref} * \frac{R_{ref}}{R/_2} D_1 + V_{ref} * \frac{R_{ref}}{R} D_0$$

$$V_{out} = V_{ref} * R_{ref} \left(\frac{D_3}{R/_8} + \frac{D_2}{R/_4} + \frac{D_1}{R/_2} + \frac{D_0}{R} \right),$$

que pode ser reescrita da seguinte forma:

$$V_{out} = V_{ref} * \frac{R_{ref}}{R} \left(2^3 D_3 + 2^2 D_2 + 2 D_1 + 2^0 D_0\right)$$

Fazendo $R_{ref} = R/(2^N - 1)$ na equação acima, com N igual à quantidade de bits, temos que:

$$V_{out} = \frac{V_{ref}}{15} * (8D_3 + 4D_2 + 2D_1 + 1D_0),$$

o que mostra que a tensão de saída é linearmente proporcional ao valor digital a ser convertido e pode variar de 0 à tensão máxima, conhecida como tensão de referência ou V_{ref}.

Um registrador D_{out} é usado para armazenar a entrada digital do DAC e assegurar que sua tensão de saída se mantenha fixa até que o conversor receba outra entrada digital. O registrador pode ser externo, mas normalmente faz parte do circuito do DAC.

Para um conversor de N bits, a tensão de saída DAC_{output} pode ser calculada diretamente a partir do código fornecido na entrada D_{in} como:

$$DAC_{output} = V_{ref} * \frac{D_{out}}{2^N - 1}$$

Um conversor DA pode ser avaliado por três parâmetros: sua resolução, tempo de acomodação (*settling time*) e transitórios (*glitches*).

A resolução é a menor mudança que pode ocorrer na tensão analógica de saída quando a entrada digital varia apenas o seu bit menos significativo. Por simplicidade, também podemos usar o número de bits na entrada do DAC para representar sua resolução. Para um DAC de N bits, o número total de possíveis níveis de saída é 2^N e a sua resolução é definida como:

$$Resolução = \frac{range\ de\ saída}{2^N - 1}$$

Por exemplo, se a tensão de saída varia de 0 a 3,3V (range de saída), então a resolução de um conversor de 12 bits é:

$$Resolução = \frac{range\ de\ saída}{2^N - 1} = \frac{3{,}3}{2^{12} - 1} \cong 805{,}86\ \mu V$$

O tempo de acomodação é o intervalo de tempo entre uma atualização nas entradas digitais do DAC até o instante em que a saída do conversor se torna estável.

Os transitórios (*glitches*) é o primeiro pico transitório de tensão que aparece na saída do conversor. Idealmente, quando a entrada muda, a saída do DAC deveria mudar monotonicamente para o novo valor. Na prática, a saída pode reduzir ou ultrapassar o novo valor devido às capacitâncias parasitas do circuito, ou devido ao tempo de fechamento das chaves da Figura 2. Por exemplo, algumas chaves podem operar mais rápidas do que outras, resultando em transitórios na tensão de saída.

A obtenção de formas de onda na saída de um DAC é possível quando diferentes valores de entrada são fornecidos ao DAC ao longo do tempo. Por exemplo, na figura 3 é mostrado o comportamento da saída de um conversor DA de 4 bits quando as entradas digitais (de 0 a 15) são fornecidas periodicamente e seus valores (amostras) são obtidos a partir de uma função senoidal com valor médio igual a 7. Em vermelho, temos uma função analógica senoidal ideal de referência. Nessa figura, a saída do conversor DA possui uma forma de onda em degraus na qual os valores são mantidos constantes durante o período de amostragem, e, por isso, podemos dizer que um conversor DA é um sistema de retenção de ordem zero. Para suavizar a forma de onda em degraus e aproximá-la da função senoidal de referência, podemos conectar filtros analógicos na saída do conversor DA.

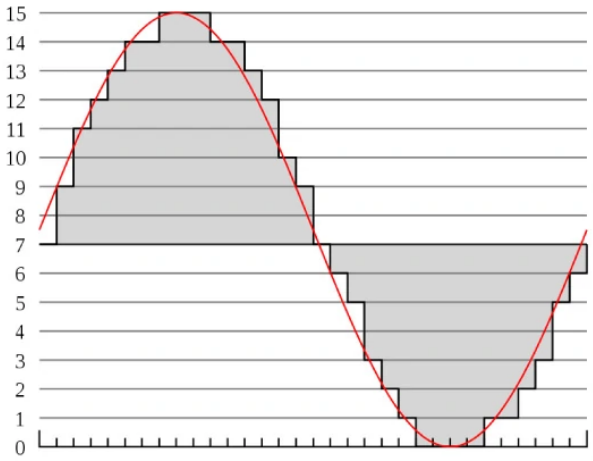


Figura 3 – Sinal de saída de um DAC quando as entradas são obtidas a partir de uma função senoidal de média 7.

8.2. Conversor DA no STM32F407

O módulo DAC no STM32F407 consiste em dois conversores digital-analógico independentes conectados ao barramento APB1, com resolução máxima de 12 bits e range de saída de 0 a 3,3V.

No modo de operação duplo, as conversões podem ser feitas independentemente ou simultaneamente quando os dois canais são agrupados para operações de atualização síncrona.

O conversor também pode ser usado em conjunto com o periférico de acesso direto à memória (DMA – *Direct Memory Access*). Nesse modo, os dados a serem convertidos são enviados da memória ao conversor diretamente pelo DMA, sem o intermédio do processador, que pode ficar livre para execução de outras tarefas.

As principais características do módulo DAC do STM32F407 são:

- Resolução configurável de 8 ou 12 bits
- Alinhamento de dados, no registrador de saída de dados, à esquerda ou à direita no modo de 12 bits
- Geração automática de formas de ondas triangulares e de ruído
- Disparos internos e externos para conversão

É mostrado na Tabela 1 o roteamento entre as saídas do DAC e os pinos de I/O do STM32F407.

Tabela 1. Definição dos pinos de saída do DAC

Canal de saída analógico	Pino
DAC1	PA4
DAC2	PA5

Para usar um canal do módulo DAC, o pino de saída correspondente (PA4 ou PA5) deve inicialmente ser configurado no modo analógico e o sinal de clock da interface digital do periférico deve ser habilitado no módulo RCC . Quando um canal DAC é ativado, o pino GPIO correspondente é diretamente conectado à saída do conversor.

Cada canal pode ser independentemente ligado setando o bit EN correspondente no registrador CR (*Control Register*) do módulo DAC. O canal DAC é ativado depois de um tempo de inicialização, chamado de t_{WAKEUP}. O bit EN ativa apenas a macro célula analógica do DAC correspondente. A interface digital do DAC, onde estão os registradores de controle e de dados, é ativada ao ligar o clock do módulo DAC por meio do bit DACEN do registrador APB1ENR do módulo RCC, independentemente do estado do bit EN.

O módulo DAC possui dois *buffers* analógicos, um para cada saída, que são usados para reduzir a impedância de saída e podem alimentar cargas externas diretamente. Cada *buffer* de saída pode ser desativado usando o bit BOFF correspondente no registrador CR do módulo DAC.

Para realizar uma conversão basta escrever o dado em um dos registradores de saída. Cada canal possui três registradores de saída de dados (DHR - *Data Holding Register*). Dependendo do modo de operação selecionado, o software precisa carregar o dado no registrador específico, conforme descrito abaixo:

• Para utilização de um canal único, existem três possibilidades, conforme mostrado na Figura 4:

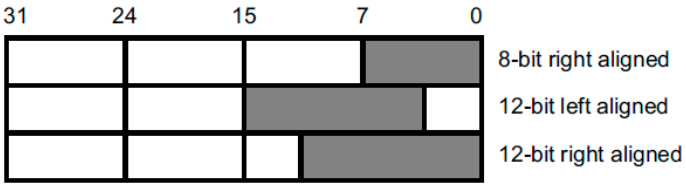


Figura 4 – Alinhamento de dados para um canal único do DAC.

- Alinhamento à direita de 8 bits: carregar dados no registrador DHR8Rx[7:0] do módulo DAC;
- Alinhamento à esquerda de 12 bits: carregar dados no registrador DHR12Lx[15:4] do módulo DAC;
- Alinhamento à direita de 12 bits: carregar dados no registrador DHR12Rx[11:0] do módulo DAC.

Dependendo do registrador DHRyyyx (x=1 ou 2, de acordo com o canal selecionado) usado, os dados gravados pelo usuário são armazenados no registrador DHRx correspondente, que são registradores internos não mapeados na memória. O registrador DHRx é então carregado no registrador DORx (*Data Output Register*) automaticamente (caso nenhum evento de disparo seja selecionado) ou por algum evento de disparo de software/hardware selecionado no registrador CR do módulo DAC.

• Para utilização dos dois canais DAC, existem três possibilidades, conforme mostrado na Figura 5:

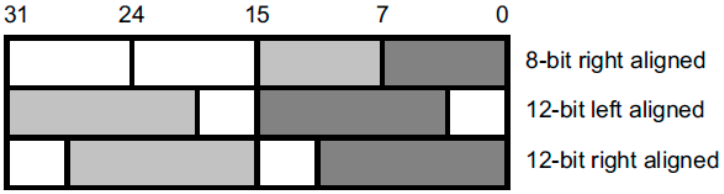


Figura 5 – Alinhamento de dados com canais duplos do DAC.

- Alinhamento à direita de 8 bits: carregar dados para o canal DAC1 no registrador DHR8RD[7:0] e dados para o canal DAC2 no registrador DHR8RD[15:8] do módulo DAC;
- Alinhamento à esquerda de 12 bits: carregar dados do canal DAC1 no registrador DAC_DHR12LD[15:4] e dados para o canal DAC2 no registrador DHR12LD[31:20] do módulo DAC;
- Alinhamento à direita de 12 bits: carregar dados para o canal DAC1 no registrador DHR12RD[11:0] e dados para o canal DAC2 no registrador DHR12LD[27:16] do módulo DAC.

Dependendo do registrador DHRyyyD carregado, os dados gravados pelo usuário são armazenados nos registradores DHR1 e DHR2, que são registradores internos não mapeados na memória. Os registradores DHR1 e DHR2 são então automaticamente carregados nos registradores DOR1 e DOR2, respectivamente, caso nenhum evento de disparo seja selecionado, ou por algum disparo de software/hardware selecionado no registrador CR.

Os registradores DOR não podem ser gravados diretamente e qualquer transferência de dados para o canal DAC deve ser realizada carregando um dos registradores DHR. Os dados armazenados no registrador DHR são transferidos automaticamente para o registrador DOR após um ciclo de clock do barramento APB1, se nenhum evento de disparo for selecionado. No entanto, quando um evento de disparo é selecionado e um gatilho ocorre, a transferência é realizada após três ciclos de clock de APB1. O controle de habilitação de eventos de disparo é feito pelo bit TEN e a fonte de disparo é selecionada pelos bits TSEL[2:0] do registrador CR.

Se o bit de controle TENx do registrador CR estiver setado, a conversão não é mais realizada ao carregador os dados nos registradores DHR, mas é disparada por um evento externo, como saídas de disparo de *timers*/temporizadores (estudaremos sobre temporizadores no capítulo 10), a linha de interrupção externa 9 ou um disparo por software. Os bits de controle TSELx[2:0] do registrador CR determinam qual dos 8 eventos possíveis acionará a conversão, conforme mostrado na Tabela 2.

Tabela 2. Disparos externos do DAC

Fonte do disparo	tipo	TSEL[2:0]
Evento de disparo do TIMER 6	sinal oriundo de um temporizador interno	0b000
Evento de disparo do TIMER 8		0b001
Evento de disparo do TIMER 7		0b010
Evento de disparo do TIMER 5		0b011
Evento de disparo do TIMER 2		0b100
Evento de disparo do TIMER 4		0b101
Evento na linha de interrupção externa 9	pino externo	0b110
Disparo por software	bit de controle no registrador SWTRIGR	0b111

Cada vez que o DAC detecta uma borda ascendente no sinal de saída de disparo do temporizador selecionado ou na linha de interrupção externa 9, os últimos dados armazenados no registrador DHRx são transferidos para o registrador DORx. O registrador DORx é atualizado três ciclos de clock do barramento APB1 após a ocorrência do disparo.

Se o disparo por software for selecionado, a conversão será iniciada assim que o bit de controle SWTRIGx do registrador SWTRIGR for setado. Nesse caso, a transferência do conteúdo do registrador DHRx para o registrador DORx leva apenas um ciclo de clock do barramentoAPB1. O bit SWTRIG é resetado pelo hardware assim que o registrador DORx é carregado com o conteúdo do registrador DHRx.

Os bits TSELx[2:0] não podem ser alterados depois que o bit ENx estiver setado.

8.3 Tensão de saída do conversor DA no STM32F407

Quando o registrador DORx é carregado com o conteúdo do registrador DHRx, a tensão de saída analógica fica disponível no pino correspondente após o tempo de acomodação que depende da tensão da fonte de alimentação e da carga de saída. As entradas digitais são convertidas em tensão de saída em uma escala linear entre 0 e VREF+ (3,3V). Considerando uma resolução de 12 bits, a tensão analógica de saída em cada canal DAC é determinada pela seguinte equação:

$$DAC_{output} = 3{,}3 * \frac{DOR}{4095}$$

8.4 Exemplo de configuração do conversor DA

Abaixo, é mostrado um trecho de código de configuração para usar o pino PA4 como saída do canal 1 do DAC no modo de conversão simples de um canal sem eventos de disparo externo.

```
void Configure_DAC1()
{
  RCC->AHB1ENR |= 1;              //habilita o clock do GPIOA
  GPIOA->MODER |= 0b11 << 8;      //inicialização do pino PA4 no modo analógico
  RCC->APB1ENR |= 1 << 29;        //habilita o clock da interface digital do DAC
  DAC->CR |= 1;                   //habilita o canal 1 do DAC
}
```

As linhas de código abaixo mostram um exemplo que escreve um valor digital de 12 bits alinhado à direita, armazenado na variável *valor,* para ser convertido pelo canal 1 do DAC.

```
uint16_t valor;
DAC->DHR12R1 = valor;
```

O código a seguir é um exemplo de programa completo que gera uma forma de onda senoidal com uma frequência de 50 Hz no pino PA4 por meio de uma *look up table*. Para isso, o array `samples[100]` contém 100 amostras de um ciclo de uma senoide que varia de 0 a 4095. No loop principal, cada amostra é fornecida ao conversor DAC em intervalos de 200 μs.

```
int main()
{
  //array com amostras do sinal senoidal
  const uint16_t samples[100]={
         2048, 2176, 2304, 2431, 2557, 2680, 2801, 2919, 3034, 3145,
         3251, 3353, 3449, 3540, 3625, 3704, 3776, 3842, 3900, 3951,
         3995, 4031, 4059, 4079, 4091, 4095, 4091, 4079, 4059, 4031,
         3995, 3951, 3900, 3842, 3776, 3704, 3625, 3540, 3449, 3353,
         3251, 3145, 3034, 2919, 2801, 2680, 2557, 2431, 2304, 2176,
         2048, 1919, 1791, 1664, 1538, 1415, 1294, 1176, 1061, 950,
         844, 742, 646, 555, 470, 391, 319, 253, 195, 144,
         100, 64, 36, 16, 4, 0, 4, 16, 36, 64,
         100, 144, 195, 253, 319, 391, 470, 555, 646, 742,
         844, 950, 1061, 1176, 1294, 1415, 1538, 1664, 1791, 1919};

  Utility_init();   //inicializa sistema de clock e funções de temporização

  RCC->AHB1ENR |= 1;             //habilita o clock do GPIOA
  GPIOA->MODER |= 0b11 << 8;     //inicialização do pino PA4 no modo analógico

  RCC->APB1ENR |= 1 << 29;       //habilita o clock da interface digital do DAC
  DAC->CR |= DAC_CR_BOFF1;       //desabilita o buffer analógico do DAC1
  DAC->CR |= 1;                  //habilita o canal 1 do DAC

  uint8_t contador = 0;          //indexador do array de amostras do sinal
  while(1)
  {
     DAC->DHR12R1 = samples[contador]; //escreve no DAC1
     ++contador;                       //atualiza o indexador
     if(contador == 100) contador = 0; //verifica se chegou ao final do array
     Delay_us(200);                    //aguarda 200 us para a próxima amostra
  }
}
```

Na Figura 6, é mostrado o sinal gerado pelo código acima no pino PA4 a partir da captura da tela de um osciloscópio.

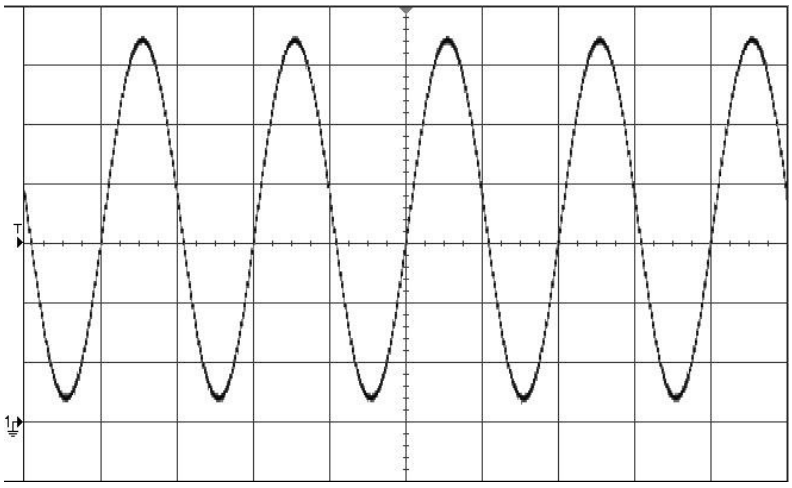
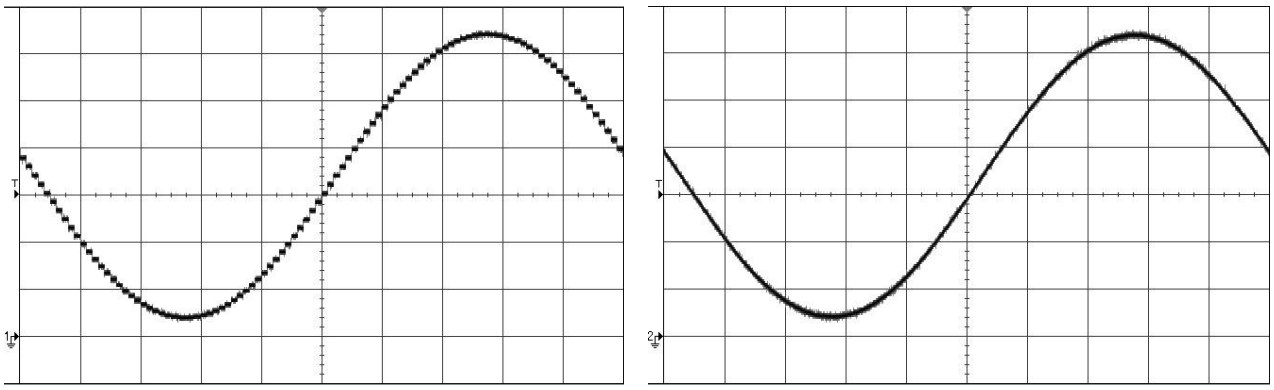


Figura 6 – Forma de onda capturada na saída do canal 1 do DAC.

Na Figura 7(a) é exibida, em detalhe, a forma de onda em degraus do sinal de saída em PA4 e em 7(b) é apresentado o mesmo sinal, porém, suavizado por um filtro passa baixas conectado no pino PA4.



(a) *(b)*

Figura 7 – Detalhe da forma de onda em degraus na saída do canal 1 do DAC (a) e após a suavização com filtro passa baixas (b).

8.4 Geração automática de formas de ondas do tipo triangular e ruído

É possível adicionar automaticamente uma forma de onda triangular ou um ruído de pequena amplitude em um sinal constante ou de variação lenta. A geração automática de formas ondas no DAC do STM32F407 é feita configurando os bits WAVEx[1:0] do registrador CR. Os valores binários possíveis de configuração são os seguintes:

0b00: geração de forma de onda desabilitada
0b01: geração de forma de onda de ruído habilitada
0b1x: geração de forma de onda triangular habilitada

A amplitude da forma de onda é selecionada, dentre 12 opções, configurando os bits MAMPx[3:0] no registrador CR. Os bits MAMPx[3:0] devem ser configurados antes de habilitar o DAC, caso contrário, não poderão ser alterados.

Para geração automática de formas de ondas, algum evento de disparo deve ser habilitado, configurando o bit TENx e selecionando o evento nos bits TSELx[2:0] no registrador CR. Os possíveis eventos de disparo são aqueles mostrados na Tabela 2.

Para geração de formas de onda triangulares, um contador crescente/decrescente interno é atualizado três ciclos de clock do barramento APB1 após cada evento de disparo. O valor deste contador é então adicionado ao registrador DHRx e a soma é armazenada no registrador DORx. O contador é incrementado enquanto for menor que a amplitude máxima definida pelos bits MAMPx[3:0]. Uma vez atingida a amplitude configurada, o contador é decrementado até 0 e reinicia o processo.

Para gerar um pseudo ruído com amplitude ajustável, um registrador de deslocamento com realimentação linear (*LFSR – Linear Feedback Shift Register*) é utilizado. O valor pré-carregado no LFSR é 0xAAA. Este registrador é atualizado três ciclos de clock APB1 após cada evento de disparo, seguindo um algoritmo de cálculo específico. O valor do LFSR é adicionado ao conteúdo DHRx e este valor é então armazenado no registro DORx.