Capítulo 9

Conversor analógico-digital no STM32F407

9.1. Introdução

A maioria dos sinais encontrados na engenharia são ditos contínuos, ou analógicos, como, por exemplo, a intensidade da luz que muda com a distância da fonte luminosa ao observador, a tensão de uma bateria que varia ao longo do tempo, uma taxa de reação química que depende da temperatura, a temperatura ao longo do dia, a velocidade de um automóvel ao longo do tempo, o volume do som em um ambiente ruidoso, etc. A conversão analógica para digital e a conversão digital para analógica são os processos que permitem que os computadores digitais, como os microcontroladores, interajam com esses sinais cotidianos.

Além de um grande número de entradas e saídas digitais, um microcontrolador também possui entradas analógicas que permitem que o microcontrolador reconheça não apenas se um pino está em lógica zero ou um, mas também possibilitam medir com precisão uma tensão analógica e convertê-la em um valor digital.

Um conversor analógico-digital (frequentemente abreviado por conversor AD ou ADC) é um circuito eletrônico capaz de gerar uma representação digital binária a partir de uma grandeza analógica. Na prática, o conversor analógico-digital pode ser representado por um bloco funcional como o mostrado na Figura 1. Ele recebe um sinal analógico de entrada (normalmente um sinal representado por um nível de tensão) e produz em sua saída uma representação digital binária desse sinal, por meio de um determinado número de bits. Em resumo, é um bloco que converte uma tensão analógica em um número digital de N bits.

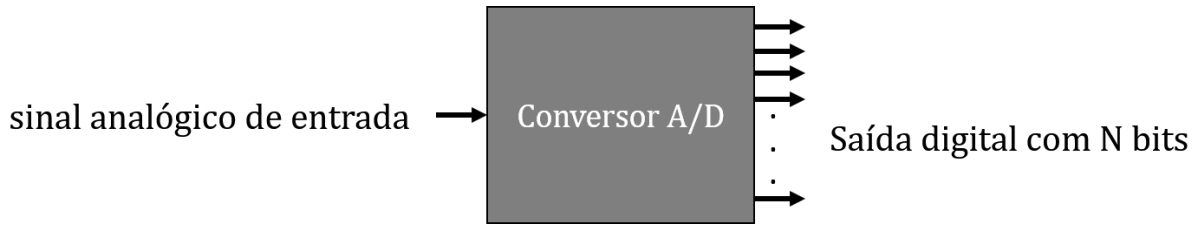

Figura 1 – Bloco funcional básico de um conversor ADC.

Os ADCs são muito úteis na interface entre dispositivos digitais (microprocessadores, microcontroladores, DSPs, etc) e dispositivos analógicos e são utilizados em aplicações como leitura de sensores analógicos, bem como digitalização de áudio e vídeo. O uso de um conversor A/D é bastante intuitivo, sendo um dos periféricos mais simples de utilizar nos microcontroladores.

Um conversor AD produz um valor digital com precisão finita para representar, aproximadamente, um valor de tensão analógica relativamente a uma tensão de referência. A tensão de referência é uma tensão fixa produzida por um circuito interno ou externo conectado a um pino do microcontrolador.

A tensão analógica aplicada na entrada do conversor deve estar dentro de uma faixa conhecida como *range* de entrada. Geralmente, essa faixa varia de 0 a 3,3V para os dispositivos alimentados com 3,3V, como é o caso do STM32F407. O valor de tensão de 3,3V é chamado de tensão de referência (V_{ref}) do conversor ADC.

Para exemplificar o uso desse dispositivo, consideremos um conversor ADC de 12 bits que pode receber um sinal de entrada analógica que pode variar de 0 a 3,3V. Ele pode fornecer, como resultado da conversão, valores binários de saída que vão de 0 (0b000000000000 = 0x000) a 2^{12}-1 = 4095 (0b111111111111 = 0xFFF).

A relação de conversão entre a tensão de entrada e o número digital de saída, que chamaremos código de saída, é sempre linear dentro dos limites de operação. Isto é, quando a tensão de entrada é de 0V, o conversor fornece como saída o valor 0x000. Quando a tensão de entrada é de 3,3V, o código de saída é 0xFFF. Quando a tensão de entrada tem um valor intermediário entre 0 e 3,3V, o código de saída terá um valor proporcionalmente intermediário. Por exemplo, se o sinal de entrada for de 1,65V (metade do range de entrada), o código de saída é 0x7FF (2047), metade da faixa de saída disponível. Dessa forma, o conversor tem o comportamento mostrado na Figura 2, onde no eixo x temos a tensão analógica presente na entrada do conversor e no eixo y temos os valores binários fornecidos na saída.

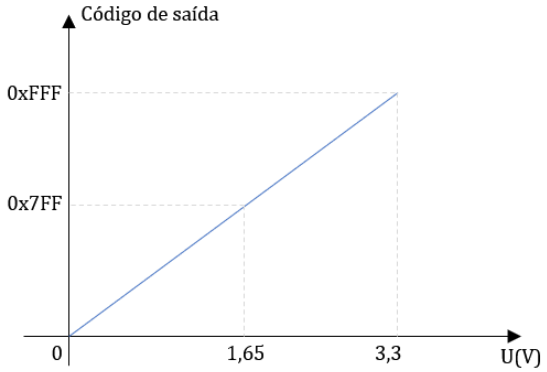

Figura 2 – Gráfico que mostra a relação entre a tensão de entrada e o código de saída do conversor ADC de 12 bits.

De maneira geral, o código de saída de um conversor de *N* bits pode variar de 0 a 2^N-1 e o seu valor, em função da tensão de entrada (v_{in}) e da tensão de referência (v_{ref}), é definido como:

$$ADC_{output} = round\left[\frac{v_{in}}{v_{ref}} * (2^N - 1)\right]$$

A função *round( )* retorna o inteiro mais próximo do valor resultante da expressão entre parênteses, já que a saída do conversor ADC representa um valor inteiro. Por exemplo, para um ADC de 12 bits, tensão de referência de 3,3V e tensão de entrada de 2,4V, o código de saída seria:

$$ADC_{output} = round\left(\frac{2{,}4}{3{,}3} * 4095\right) = round(2978{,}18) = 2978$$

Outro exemplo, para um ADC de 12 bits, tensão de referência de 3,3V e tensão de entrada de 1,8V, o código de saída seria:

$$ADC_{output} = round\left(\frac{1{,}8}{3{,}3} * 4095\right) = round(2233{,}64) = 2234$$

Devemos evitar aplicar tensões maiores do que a tensão de referência do conversor sob o risco de danificar permanentemente o seu circuito interno. Para tensões maiores que a tensão de referência mas que não danifiquem o conversor, a saída é limitada a 2^N-1, evento que chamamos de saturação do conversor.

Perceba que, apesar da entrada poder assumir infinitos valores dentro do range de entrada, a saída possui um número finito de valores discretos possíveis (para um ADC de 12 bits, 4096 diferentes códigos). Na prática, é como se o conversor dividisse o range de entrada em 2^N-1 pequenas faixas de tensão e relacionasse um código a cada uma das faixas. Ele então observa em qual dessas faixas a tensão de entrada se encontra e fornece em sua saída o código correspondente àquela faixa. Dessa forma, pode haver uma variação mínima na tensão de entrada (dentro de uma dessas faixas) sem que a saída atualize para mostrar a nova representação digital.

A cada uma dessas pequenas faixas, damos o nome de resolução do conversor. Isto é, a resolução é a menor mudança que deve ocorrer na tensão analógica de entrada para que a saída digital possa variar o código em uma unidade. Por simplicidade, às vezes usamos a própria quantidade de bits do ADC para representar sua resolução.

Para um ADC de *N* bits, o número total de possíveis códigos de saída é 2^N e a sua resolução é definida como:

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

Por exemplo, se o range de entrada é de 0 a 3,3V, então a resolução de um conversor de 12 bits é:

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

A conversão analógico para digital não é um processo instantâneo, mas exige um pequeno intervalo de tempo, dependendo da tecnologia de construção do circuito conversor. Quanto maior a quantidade de bits de saída de um conversor AD, melhor será a sua resolução. Por outro lado, conversores com uma grande quantidade de bits de saída exigem mais memória para armazenamento dos valores digitalizados e exigem mais tempo para conclusão da conversão.

Um importante parâmetro de desempenho dos conversores AD é a taxa de amostragem (*sampling rate*), que indica quantas conversões podem ser realizadas no intervalo de tempo de um segundo.

Três arquiteturas de conversores AD são bastante populares:

• *Sigma-delta*, usada em aplicações de baixa velocidade, com baixa taxa de amostragem, tipicamente menos de 100 mil amostras por segundos (100 KSPS – *Kilo Samples Per Second*), e alta resolução, de 12 a 24 bits, como em aplicações com amostragem de voz e áudio, tipicamente empregadas em *smartphones*;
• *Aproximação sucessiva*, adequada para aplicações em aquisição de dados e taxa de amostragem moderada, tipicamente menos de 5 milhões de amostras por segundo (5 MSPS – *Mega Samples Per Second*);
• *Pipelined*, largamente empregada em aplicações de alta velocidade, como em osciloscópios digitais, HDTV, radares, exigindo altas taxas de amostragem (maiores que 5 MSPS) e resolução relativamente baixa, menor que 18 bits.

9.2. Conversor AD de aproximação sucessiva

Uma arquitetura de um conversor AD de aproximação sucessiva é mostrada na Figura 3. A arquitetura inclui dois componentes principais: o amplificador de amostragem e retenção (*Sampling and Hold Amplifier-SHA*) e o registrador de aproximações sucessivas (*SAR - Sucessive Approximation Register*).

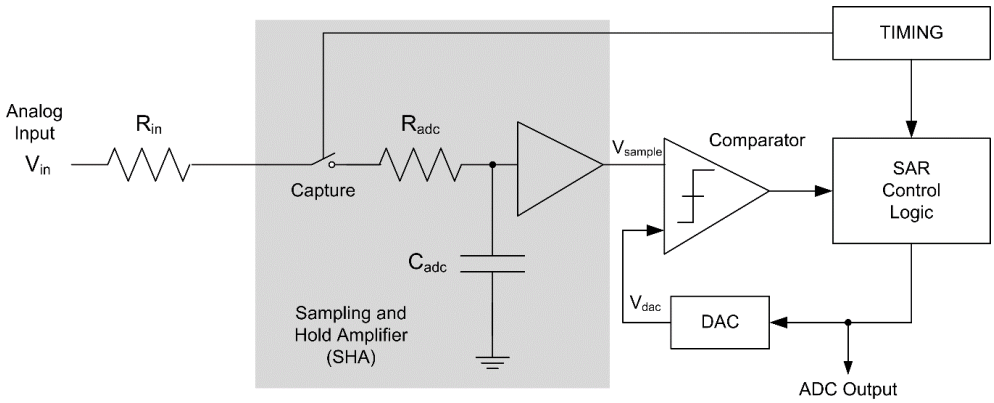

Figura 3 – Arquitetura de um conversar AD de aproximações sucessivas.

O circuito *Sampling and Hold Amplifier* inclui o capacitor C_{adc} e um amplificador operacional que é usado para amostrar a tensão de entrada Vin e reter o valor dessa tensão durante o tempo necessário para a conversão.

Quando a chave de captura é fechada, a tensão sobre o capacitor aumenta exponencialmente até atingir o valor de V_{in}. O tempo necessário para que isso aconteça depende do valor da capacitância e da resistência R_{in}+R_{adc}. Dessa forma, a tensão V_{in} não pode ser amostrada instantaneamente. Assim, para se obter uma conversão precisa, o software deve deixar a chave de captura fechada pelo tempo suficiente para a correta carga do capacitor C_{adc}. O tempo na qual a chave de captura deve permanecer fechado é chamado de tempo de amostragem (*sampling time*). A tensão de entrada amostrada é amplificada e fornecida na saída desse bloco como V_{sample}.

O bloco DAC, abaixo do comparador, é um conversor digital para analógico, que foi abordado no capítulo anterior. Em resumo, o bloco DAC recebe o código digital de saída do registrador SAR e o converte para uma tensão analógica (Vdac). O bloco comparador, então, compara continuamente as tensões V_{sample} e V_{dac} e fornece em sua saída um nível lógico alto (se $V_{sample} > V_{dac}$) , ou baixo (se $V_{sample} < V_{dac}$).

A lógica de controle do registrador de aproximações sucessivas (SAR) usa o algoritmo de busca binária para encontrar o número digital que representa, de forma mais aproximada, o valor da tensão de entrada: A lógica de controle muda dinamicamente o valor binário da saída do registrador para que a tensão de saída do conversor DAC (V_{dac}) se aproxime da tensão amostrada.

O processo de conversão ocorre da seguinte forma:

1. A conversão inicia setando o bit mais significativo da saída do SAR, levando a saída do DAC para a metade da tensão de referência (V_{ref}), comparando esse valor com a tensão amostrada;
2. Se o valor da tensão amostrada é maior que ½V_{ref}, a saída do comparador permanece em nível alto e a lógica de controle mantém setado o bit mais significativo da saída do SAR. Caso contrário, a saída do comparador é resetada e o bit mais significativo da saída do SAR é resetado;
3. Continuando o processo, o bit seguinte da saída do SAR é setado, levando a saída do DAC para 3/4 ou 1/4 de V_{ref}, dependendo do resultado da comparação anterior;
4. O processo se repete até que todos os bits da saída do SAR tenham sido determinados. A saída do SAR é a própria saída do conversor ADC.

Se o conversor AD tem uma resolução de *N* bits, a aproximação sucessiva leva *N* passos para ser concluída. Isso representa uma relação de compromisso entre resolução e taxa de amostragem. Uma resolução mais alta reduz a taxa de amostragem máxima.

Na Figura 4 é apresentado um exemplo de conversão de uma tensão de entrada de 0.69V (V_{sample}) em um número binário de 4 bits usando um conversor de aproximação sucessiva como o mostrado na Figura 3. Suponha que a tensão de entrada pode variar entre 0 e 1V (V_{ref}).

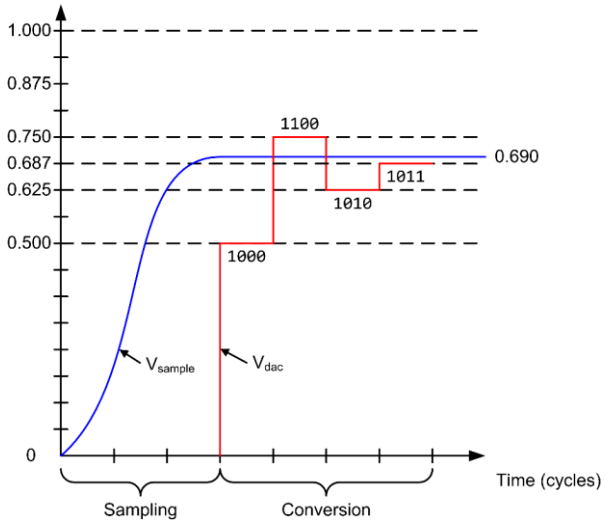

Figura 4 – Exemplo de uma conversão de um sinal de 0.69V com um ADC de aproximação sucessiva de 4 bits.

1. A lógica de controle inicia a conversão setando o bit mais significativo da saída (isto é, a saída do ADC é 0b1000), fazendo a tensão V_{dac} = 0,5V e comparando esse valor com V_{sample}.
2. Como V_{sample} é maior que V_{dac}, a lógica de controle mantém esse bit setado e seta o segundo bit da saída (isto é, a saída do ADC é 0b1100), fazendo V_{dac} = 0,75V (3/4 V_{ref});
3. Como V_{sample} é menor que V_{dac}, a lógica de controle reseta o segundo bit e seta o terceiro bit da saída (isto é, a saída do ADC é 0b1010), fazendo V_{dac} = 0,625;
4. Como V_{sample} é maior que V_{dac}, a lógica de controle mantém setado o terceiro bit da saída e seta o quarto bit (isto é, a saída do ADC é 0b1011), fazendo V_{dac} = 0,6875;
5. Como V_{sample} ainda é maior que V_{dac}, a lógica de controle mantém o último bit setado e o processo é concluído. Dessa forma, a conversão AD do sinal de entrada resulta numa saída do ADC de 0b1011.

9.3. Conversor AD no STM32F407

O STM32F407 possui três conversores AD de aproximação sucessiva com resolução de 12 bits, nomeados como ADC1, ADC2 e ADC3, conectados ao barramento APB2. Cada um possui 19 canais de entrada multiplexados, permitindo medir sinais de 16 fontes externas, duas fontes internas e o canal VBAT (tensão no pino VBAT, usado para conectar uma bateria que mantém os ajustes do RTC). A conversão AD dos canais pode ser realizada em modo único, contínuo, varredura ou descontínuo. O resultado da conversão é armazenado em um registrador de 16 bits, cujos bits podem ser alinhados à esquerda ou à direita. O recurso *watchdog* analógico permite o monitoramento da tensão analógica e a detecção, por hardware, se a tensão de entrada ultrapassar limiares mínimo e máximo pré-definidos pelo usuário. A tensão de entrada pode variar de 0 a 3,3V (V_{ref}).

As principais características dos conversores do STM32F407 são:

• Resolução configurável para 12, 10, 8 ou 6 bits;
• Geração de interrupção no final da conversão e no caso de detecção de eventos do *watchdog* analógico*;*
• Modos de conversão simples e contínuo e de varredura para conversão automática de múltiplos canais;
• Tempo de amostragem programável por canal;
• Modo de conversão simultânea em todos os conversores;
• Atraso configurável entre conversões no modo simultâneo;
• Disparos de início de conversão feitos por software ou sinais externos oriundos, por exemplo, de temporizadores internos ou de pinos do microcontrolador.

A Figura 5 mostra o diagrama de blocos simplificado de um dos ADCs do STM32F407.

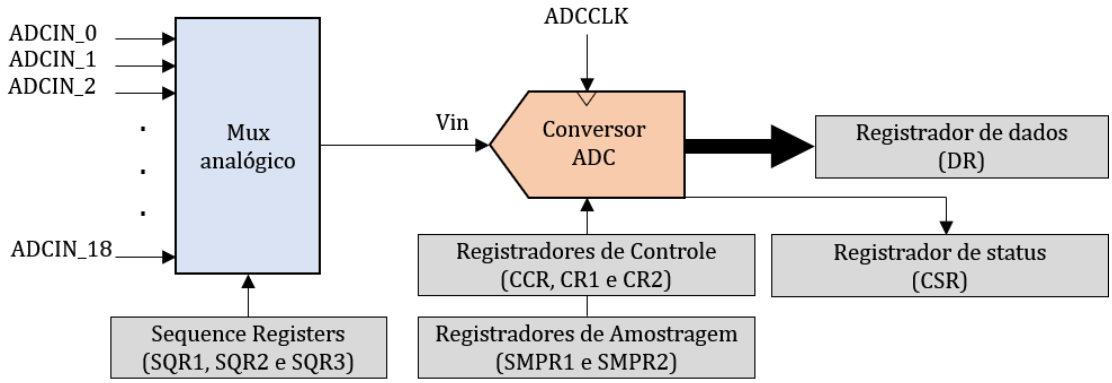

Figura 5 – Diagrama de blocos simplificado de um dos ADCs do STM32F407.

O ADC possui dois esquemas de clock:

• Clock para a interface digital (para acesso aos registradores). Este sinal é o próprio sinal de clock do barramento APB2. O clock da interface digital pode ser ativado/desativado individualmente para cada ADC por meio dos bits ADC1EN, ADC2EN e ADC3EN no registrador APB2ENR do módulo RCC.
• Clock para o circuito analógico, ADCCLK, comum a todos os ADCs. Este sinal é gerado a partir do clock do barramento APB2, cuja frequência máxima é de 84MHz, dividido por um prescaler programável pelos bits ADCPRE[1:0] do registrador CCR (*Common Control Register*) que permite que o ADC trabalhe com frequências de 84MHz/2, /4, /6 ou /8. O valor máximo permitido para o sinal ADCCLK é de 36 MHz.

O ADC é ligado configurando o bit ADON no registrador CR2 (*Control Register 2*) do módulo ADC. A conversão inicia quando o bit SWSTART, no mesmo registrador, é setado. É possível parar a conversão e colocar o ADC no modo de desligamento resetando o bit ADON. Nesse modo, o ADC consome quase nenhuma energia.

Existem 16 canais externos multiplexados, ADC_IN0 a ADC_IN15. É possível organizar as conversões em grupos. Um grupo consiste em uma sequência de conversões que podem ser feitas em qualquer canal e em qualquer ordem. Por exemplo, é possível implementar a sequência de conversão na seguinte ordem: ADC_IN3, ADC_IN8, ADC_IN2, ADC_IN2, ADC_IN0, ADC_IN2, ADC_IN15.

Um grupo é composto por até 16 conversões. Os canais e sua ordem na sequência de conversão devem ser selecionados nos registradores SQR (*Sequence Register*) do módulo ADC. A quantidade de conversões desejada deve ser gravada nos bits L[3: 0] do registrador SQR1 do módulo ADC.

Internamente, há um sensor de temperatura analógico conectado ao canal ADC1_IN16, que pode ser utilizado para medir a temperatura do chip numa faixa de -40 a 125 °C, com uma precisão de ±1,5 °C (≅1%).

A tensão de referência interna VREFINT está conectada ao canal ADC1_IN17 e o sinal VBAT está conectado ao canal ADC1_IN18.

O conversor AD faz a amostragem da tensão de entrada usando alguns ciclos do sinal de clock ADCCLK. A quantidade de ciclos usados na amostragem de cada canal pode ser configurada usando os bits SMP[2:0] nos registradores SMPR1 e SMPR2. Cada canal pode ser amostrado com um tempo de amostragem diferente. O tempo total de conversão T_{conv}, dado em ciclos, é calculado da seguinte forma:

$$\mathrm{T_{conv}} = \text{tempo de amostragem} + 12 \text{ ciclos}$$

Por exemplo, com ADCCLK = 30 MHz e tempo de amostragem igual a 3 ciclos:

$$\mathrm{T_{conv}} = 3 + 12 = 15 \text{ ciclos} = 500 \text{ ns}$$

9.4. Modos de conversão

O módulo ADC executa conversões em canais selecionados nos modos de conversão simples e contínuo, como mostrado na Figura 6.

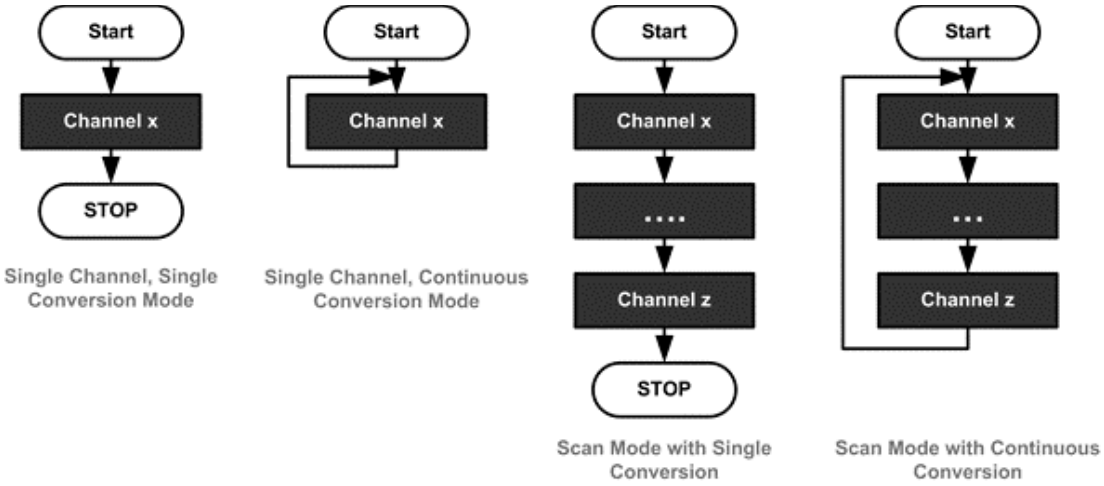

Figura 6 – Modos de conversão do ADC.

Considerando a conversão de um único canal, o canal escolhido deve ser selecionado pelos bits SQ1[4:0] do registrador SQR3. O resultado da conversão é armazenado no registrador DR (*Data Register*) do módulo ADC. Depois da conversão, a flag EOC (*End Of Conversion*) é setada no registrador CSR (*Common Status Register*) do módulo ADC. O módulo pode gerar uma solicitação de interrupção se o bit EOCIE no registrador CR1 estiver setado.

No modo de conversão simples de um canal, após o disparo da conversão, o canal selecionado é convertido uma vez e o ADC fica aguardando um novo sinal de disparo (imagem mais à esquerda na Figura 6). Nesse modo, os bits SCAN, no registrador CR1 (*Control Register 1*), e CONT, no registrador CR2, devem ser resetados.

No modo de conversão contínua de um canal, o ADC inicia uma nova conversão imediatamente após concluir a anterior (segunda imagem da esquerda para a direita na Figura 6). Nesse modo, os bits CONT e SCAN devem ser feitos iguais a 1 e 0, respectivamente.

O conversor AD também pode executar a conversão de uma lista pré-definida de canais no modo de escaneamento. Este modo converte, numa ordem pré-definida, os canais especificados nos registradores SQR1, SQR2 e SQR3. O resultado de cada conversão é armazenado no registrador DR e, depois da conversão, o flag EOC é setado no registrador CSR. O módulo ADC pode gerar uma solicitação de interrupção se o bit EOCIE no registrador CR1 estiver setado. O software deve ler o resultado da conversão no registrador DR após a conversão de cada canal individual pois qualquer valor escrito nesse registrador é sobrescrito após cada nova conversão.

No modo de conversão simples de múltiplos canais, o ADC inicia uma nova sequência de conversão, após o sinal de disparo e para após a conversão do último canal da lista (terceira imagem da esquerda para a direita na Figura 6). Nesse modo, os bits CONT e SCAN devem ser feitos iguais a 0 e 1, respectivamente.

No modo de conversão contínua de múltiplos canais, o ADC inicia continuamente uma nova sequência de conversão, após a conversão do último canal na sequência anterior (imagem mais à direita na Figura 6). Nesse modo, os bits CONT e SCAN devem ser feitos iguais a 1 e 1.

9.5. Alinhamento dos dados no registrador de dados do ADC

O software pode modificar a resolução da conversão escrevendo nos bits RES[1:0] do registrador CR1, entretanto, o registrador do resultado da conversão (DR) tem, em todo caso, 16 bits válidos. O bit ALIGN no registrador CR2 seleciona o alinhamento dos dados armazenados após a conversão. Os dados em DR podem ser alinhados à direita ou à esquerda, como mostrado na Figura 7. Apenas doze bits, no máximo, são significativos.

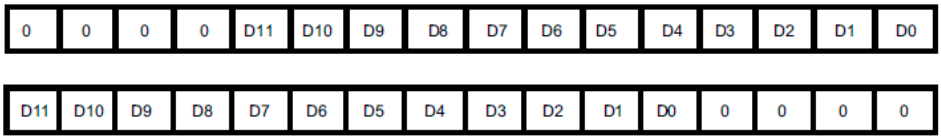

Figura 7 – Alinhamento dos dados armazenados em DR: à direita (imagem superior); à esquerda (imagem inferior).

9.6. Pinos de entrada do ADC

É mostrada na Tabela 1 a conexão entre os conversores AD e os pinos de I/O do chip, que devem ser configurados no modo analógico para poderem ser usados pelo conversor.

Tabela 1. Definição dos pinos de entrada do ADC

Canal de entrada analógico	Pino	Canal de entrada analógico	Pino
ADC123_IN0	PA0	ADC12_IN8	PB0
ADC123_IN1	PA1	ADC12_IN9	PB1
ADC123_IN2	PA2	ADC123_IN10	PC0
ADC123_IN3	PA3	ADC123_IN11	PC1
ADC12_IN4	PA4	ADC123_IN12	PC2
ADC12_IN5	PA5	ADC123_IN13	PC3
ADC12_IN6	PA6	ADC12_IN14	PC4
ADC12_IN7	PA7	ADC12_IN15	PC5

9.7. Escalonamento de grandezas medidas pelo conversor A/D

O resultado da conversão do sinal analógico está sempre compreendido entre 0 e 2^N-1, com N sendo a quantidade de bits do conversor AD. Entretanto, esse resultado geralmente não expressa o valor real da grandeza que está sendo medida por meio de um sensor ou dispositivo conectado ao pino de entrada analógica. É preciso então determinar qual a relação existente entre os valores resultantes da conversão e os valores reais das grandezas medidas.

Consideremos um exemplo com um sensor analógico de temperatura que fornece um sinal de tensão de 10 mV/°C. Assim, quando a temperatura é de 0 °C, a tensão de saída fornecida por ele é de 0 V, e se a temperatura é de 330 °C, a tensão de saída é de 3,3V. Assim, a relação entre a tensão fornecida pelo sensor, o valor lido pelo conversor AD e o valor da temperatura é expressa nas escalas da Figura 8.

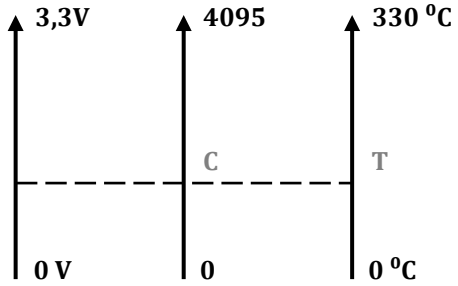

Figura 8 – Relação entre tensão fornecida pelo sensor, resultado da conversão do ADC e valor real da temperatura.

Analisando as relações entre as escalas acima, consegue-se demonstrar que a temperatura T equivalente a um resultado da conversão com um código C é expressa por:

$$\frac{T-0}{330-0}=\frac{C-0}{4095-0} \quad \therefore \quad T=\frac{330}{4095}*C,$$

assim, esse deve ser o escalonamento feito para obtermos corretamente o valor de temperatura medida pelo sensor.

De modo geral, para um sensor que fornece uma faixa de tensão de *0* a V_{ref}, resultando em uma faixa de variação de códigos *C* de 0 a 2^N-*1* como resultado da conversão de um conversor ADC, correspondente à variação de uma grandeza física na faixa de F_1 a F_2, o escalonamento que deve ser feito é dado por:

$$F=\frac{C}{2^N-1}(F_2-F_1),$$

onde *F* é o valor real da grandeza física associada a um determinado valor *C* fornecido como resultado da conversão do ADC.

9.8. Exemplo de configuração do conversor AD

Abaixo é mostrada uma função de configuração para usar o pino PA0 como pino de entrada do canal 0 (IN_0) do ADC1 no modo de conversão simples de um canal com o disparo feito por software.

```
void Configure_ADC1()
{
  RCC->AHB1ENR |= 1;              //habilita o clock do GPIOA
  GPIOA->MODER |= 0b11;           //pino PA0 como entrada analógica

  RCC->APB2ENR |= 1 << 8;         //liga o clock da interface digital do ADC1

  ADC->CCR |= 0b01 << 16;         //prescaler /4 (fADC = 84/4 = 21MHz)
  ADC1->SQR1 &= ~(0xF << 20);     //conversão de apenas um canal
  ADC1->SQR3 &= ~(0x1F);          //seleção do canal a ser convertido (IN_0)
  ADC1->CR2 |= 1;                 //liga o conversor AD
}
```

O código abaixo faz o disparo de conversão do ADC e aguarda a flag EOC indicar o fim da conversão. Em seguida, é feita a leitura do valor convertido pela variável *leitura*.

```
ADC1->CR2 |= 1 << 30;                   //inicia a conversão
while(!(ADC1->SR & 0x02));              //aguarda o fim da conversão
uint16_t leitura = ADC1->DR;            //faz a leitura do valor convertido
```

9.9. Sensor de temperatura interno

O STM32F407 possui um sensor de temperatura interno que serve para medir a temperatura do chip. O sensor opera na faixa de –40 a +105 °C com uma precisão de ± 1,5 °C. A tensão de saída do sensor varia linearmente com a temperatura. A inclinação e deslocamento da função linear temperatura x tensão depende de cada chip devido às variações inerentes ao processo de fabricação. O sensor de temperatura interno é mais adequado para aplicações que detectam variações de temperatura em vez de temperaturas absolutas. Se uma leitura de temperatura com precisão for necessária, um sensor de temperatura externo deve ser usado.

O sensor é conectado internamente ao canal ADC1_IN16. O bit TSVREFE do registrador CCR do módulo ADC deve ser setado para habilitar o sensor de temperatura. Quando não estiver sendo usado, o sensor pode permanecer desligado para economia de energia. O sensor tem um tempo de inicialização de no máximo 10 μs após sair do modo de desligamento antes de poder fornecer a tensão (V_{SENSE}) no nível correto.

Para usar o sensor de temperatura, é necessário:

1. Selecionar o canal de entrada ADC1_IN16;
4. Selecionar um tempo de amostragem maior que o tempo mínimo de amostragem especificado no datasheet do STM32F407 (10 μs);
5. Setar o bit TSVREFE no registrador CCR para ligar o sensor;
6. Iniciar a conversão do ADC setando o bit SWSTART (ou por disparo externo);
7. Ler os dados resultantes no registrador de dados DR do módulo ADC;
8. Calcular a temperatura usando a seguinte fórmula:

$$Temperatura\ (em\ °C) = \{(V_{SENSE} - V_{25}) / Avg\_Slope\} + 25,$$

onde:

- V_{25} = valor V_{SENSE} para a temperatura de 25 °C (tipicamente 0,76 V)
- *Avg_Slope* = inclinação média da curva temperatura versus V_{SENSE} (tipicamente 2,5 mV/°C)

Para obter uma maior precisão na medição de temperatura, o datasheet do microcontrolador STM32F407 informa dois valores de calibração do ADC para que o usuário possa determinar a relação exata entre os valores digitalizados e a temperatura ambiente para um determinado chip. Os valores são obtidos experimentalmente durante a fabricação do chip, nas temperaturas de 30 °C e de 110 °C, quando a tensão de referência do ADC é de 3,3V. Os valores crus do ADC para essas duas temperaturas são armazenados nos endereços de memória mostrados na Tabela 2.

Tabela 2. Valores de calibração do sensor de temperatura

Símbolo	Parâmetro	Endereço de memória
TS_CAL1	Valor cru do ADC obtido na temperatura de 30 °C com V_{ref} a 3,3V	0x1FFF7A2C - 0x1FFF7A2D
TS_CAL2	Valor cru do ADC obtido na temperatura de 110 °C com V_{ref} a 3,3V	0x1FFF7A2E - 0x1FFF7A2F

A partir dos valores obtidos da Tabela 2, é possível montar o gráfico que descreve a relação linear entre o valor convertido (C) do sinal do sensor e a temperatura medida (T), conforme mostrado na Figura 9.

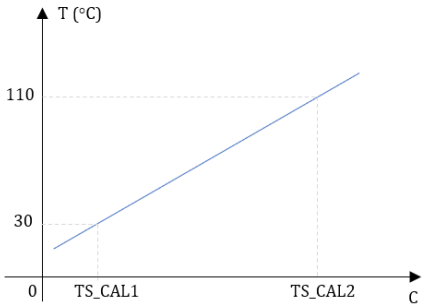

Figura 9 – Relação entre o valor convertido (C) do sinal do sensor e a temperatura medida (T).

A relação entre C e T, portanto, pode ser descrita por:

$$T = \frac{80(C - TS\_CAL1)}{(TS\_CAL2 - TS\_CAL1)} + 30$$

Abaixo, é mostrado um código que habilita e faz a leitura do sensor de temperatura interno, calculando, em seguida, o valor da temperatura.

```
RCC->APB2ENR |= 1 << 8;          //liga o clock da interface digital do ADC1

ADC->CCR |= 0b01 << 16;          //prescaler /4
ADC1->SQR1 &= ~(0xF << 20);      //conversão de apenas um canal
ADC1->SQR3 |= 16;                //seleção do canal a ser convertido (IN_16)
ADC1->SMPR1 |= (7 << 18);        //tempo de amostragem igual a 480 ciclos de ADCCLK
ADC->CCR |= (1 << 23);           //liga o sensor de temperatura
ADC1->CR2 |= 1;                  //liga o conversor AD

uint32_t *p = (uint32_t *) 0x1FFF7A2C; //cria ponteiro para uma posição de memória
uint32_t Word = *p;                    //lê o conteúdo da memória
uint16_t TS_CAL1 = (Word & 0x0000FFFF);       //lê o valor de TS_CAL1
uint16_t TS_CAL2 = (Word & 0xFFFF0000) >> 16; //lê o valor de TS_CAL2

ADC1->CR2 |= 1 << 30;                  //inicia a conversão
while(!(ADC1->SR &  0x02));            //aguarda o fim da conversão

//cálculo da temperatura
float temperatura = ((80*(int)(ADC1->DR - TS_CAL1))/(TS_CAL2-TS_CAL1))+30;
```

9.10 Conversão por disparos externos e polaridade do disparo

A conversão pode ser disparada por um evento externo, por exemplo, uma linha de interrupção externa ou eventos de temporizadores internos (falaremos sobre temporizadores no próximo capítulo). Se os bits de controle EXTEN[1:0] do registrador CR2 forem diferentes de "0b00", então eventos externos são capazes de disparar uma conversão com a polaridade do sinal de disparo selecionada. Os valores binários possíveis de configuração dos bits EXTEN são os seguintes:

0b00: disparos externos desabilitados (modo padrão)
0b01: disparo pela borda de subida de um sinal externo
0b10: disparo pela borda de descida de um sinal externo
0b11: disparo por ambas as bordas (subida e descida) do sinal externo

A polaridade do sinal de disparo externo pode ser alterada em tempo de execução.

Os bits de controle EXTSEL[3:0] do registrador CR2 são usados para selecionar quais dos 16 possíveis eventos externos podem disparar a conversão. A Tabela 3 fornece as opções de sinal externo para conversão.

Tabela 3. Disparos externos do ADC

Fonte do disparo	tipo	EXTSEL[3:0]
Evento do canal 1 do TIMER 1	sinal oriundo de um temporizador interno	0b0000
Evento do canal 2 do TIMER 1		0b0001
Evento do canal 3 do TIMER 1		0b0010
Evento do canal 2 do TIMER 2		0b0011
Evento do canal 3 do TIMER 2		0b0100
Evento do canal 4 do TIMER 2		0b0101
Evento de disparo do TIMER 2		0b0110
Evento do canal 1 do TIMER 3		0b0111
Evento de disparo do TIMER 3		0b1000
Evento do canal 4 do TIMER 4		0b1001
Evento do canal 1 do TIMER 5		0b1010
Evento do canal 2 do TIMER 5		0b1011
Evento do canal 3 do TIMER 5		0b1100
Evento do canal 1 do TIMER 8		0b1101
Evento de disparo do TIMER 8		0b1110
Evento na linha de interrupção externa 11	pino externo	0b1111

9.11 Sistemas de processamento digital de sinais

Sistemas de processamento digital de sinais são sistemas computacionais que fazem a aquisição de sinais do mundo real como voz, áudio, vídeo, temperatura e pressão que foram digitalizados e então os manipulam matematicamente por meio de algoritmos numéricos.

Os sinais precisam ser processados para que as informações que eles contêm possam ser extraídas, analisadas ou convertidas em outro tipo de sinal. No mundo real os sinais como som, luz, temperatura ou pressão são ditos analógicos. Conversores AD convertem o sinal do mundo real e o transformam para o formato digital. A partir de então, os processadores assumem a tarefa, processando as informações digitalizadas. Em seguida, eles realimentam as informações processadas para uso no mundo real, podendo a realimentação ocorrer de duas maneiras: Digitalmente ou em formato analógico, por meio de um conversor DA.

Um sistema de processamento digital de sinais é geralmente formado pelos elementos mostrados no diagrama de blocos da Figura 8.

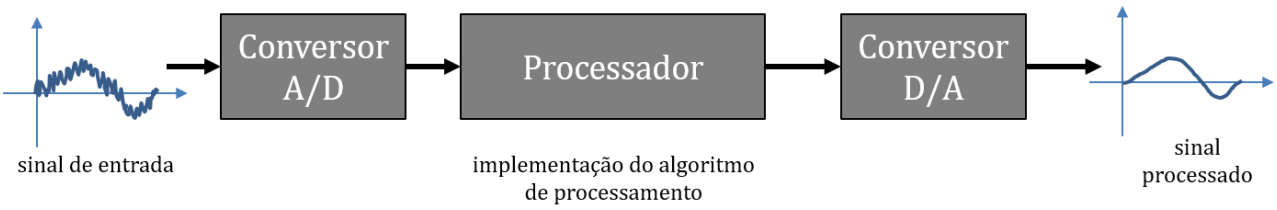

Figura 8 – Diagrama de blocos de um sistema de processamento digital de sinais.

Para exemplificar, consideremos um reprodutor de áudio MP3: Durante a fase de gravação, o áudio analógico é captado por meio de um sensor (microfone) ou outra fonte sonora. Esse sinal analógico é então convertido em um sinal digital por um conversor analógico-digital e encaminhado para um processador digital de sinais (DSP – *Digital Signal Processor*). O DSP realiza a codificação MP3 e salva o arquivo na memória. Durante a fase de reprodução, o arquivo é lido da memória, decodificado pelo DSP e então convertido de volta em um sinal analógico por meio do conversor digital para analógico para que possa ser enviado ao sistema de alto-falantes. Em um exemplo mais complexo, o DSP executaria outras funções, como controle de volume, equalização, inserção de efeitos e interface com o usuário.

As informações em um sistema de processamento digital de sinais podem ser usadas para diversos fins como segurança eletrônica, telefones, sistemas de *home theater*, compactação de vídeo e diagnósticos médicos. Os sinais podem ser comprimidos para que possam ser transmitidos de forma mais rápida e eficiente de um lugar para outro (por exemplo, a teleconferência e/ou chamadas de vídeo podem transmitir voz e vídeo de alta fidelidade). Os sinais também podem ser aprimorados ou manipulados para melhorar sua qualidade ou fornecer

informações que não são detectadas por humanos (por exemplo, cancelamento de eco para telefones celulares ou imagens médicas mais nítidas).

Embora os sinais do mundo real possam ser processados em sua forma analógica, o processamento digital de sinais oferece as vantagens de alta velocidade, precisão e versatilidade. Por ser programável, um DSP pode ser usado em uma ampla variedade de aplicações. O microcontrolador STM32F407 pode ser considerado um DSP pois possui conversores ADC e DAC integrados, bem como fornece instruções de processamento digitais de sinais. ■