Capítulo 9

Conversor analógico-digital no STM32F407

9.1. Introdução

Além de um grande número de linhas de entrada e saída digital, 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 numérico em formato digital.

Um conversor analógico-digital (frequentemente abreviado por conversor AD ou ADC) é um dispositivo eletrônico capaz de gerar uma representação digital a partir de uma grandeza analógica, normalmente um sinal representado por um nível de tensão. 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 e digitalização de áudio e vídeo. O uso do conversor A/D é bastante intuitivo, sendo um dos periféricos mais simples de serem utilizados 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 por um circuito externo conectado a um pino do microcontrolador. O conversor não consegue converter uma tensão maior que a sua tensão de referência.

A conversão de uma tensão analógica em um número binário no conversor AD é baseada na comparação da tensão de entrada com uma escala interna, que tem 2^n pontos, onde n é chamada de resolução do conversor. A resolução pode variar de 6 a 32 bits, embora 12, 16 ou 24 bits sejam as resoluções mais comumente utilizadas em muitas aplicações. A marca mais baixa da escala de comparação representa a tensão de referência inferior, digamos 0 V, enquanto a sua maior marca representa a tensão de referência superior, digamos 3,3 V.

Para exemplificar o uso do ADC, consideremos um conversor AD de 10 bits que suporta um sinal de entrada analógica que pode variar de 0 a 3,3V. Ele pode fornecer como resultado da conversão os valores binários de 0 (0b0000000000) a 2^{10}-1 = 1023 (0b1111111111 – 0x3FF), sendo capaz de distinguir 1024 níveis discretos do sinal, o que garante uma resolução de 3,23 mV. Se o sinal de entrada do conversor A/D estiver em 1,65V, por exemplo, o valor binário gerado a partir da conversão é 512, metade da faixa disponível. Dessa forma, o conversor tem o comportamento mostrado na Figura 1, onde no eixo X temos a tensão analógica presente na entradas do conversor e no eixo Y temos os valores binários fornecidos na saída.

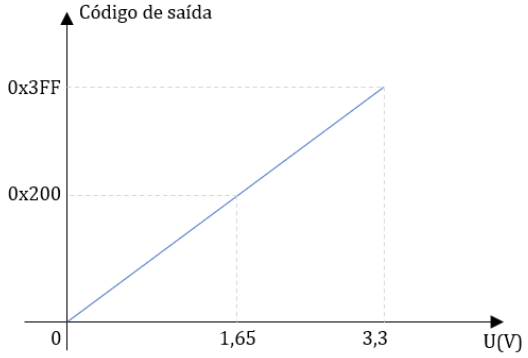

Figura 1 – Gráfico que mostra a relação entre a tensão de entrada e o código da saída do conversor AD de 10 bits.

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 banda de voz e áudio, tipicamente empregadas em *smartphones*;
• *Aproximação sucessiva*, adequada para aplicações de baixa potência 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

A arquitetura de um conversor AD de aproximação sucessiva (SAR) é mostrada na Figura 2. A arquitetura inclui dois componentes principais: o amplificador de amostragem e retenção (*Sampling and Hold Amplifier-SHA*) e o quantizador digital (*SAR Control Logic*).

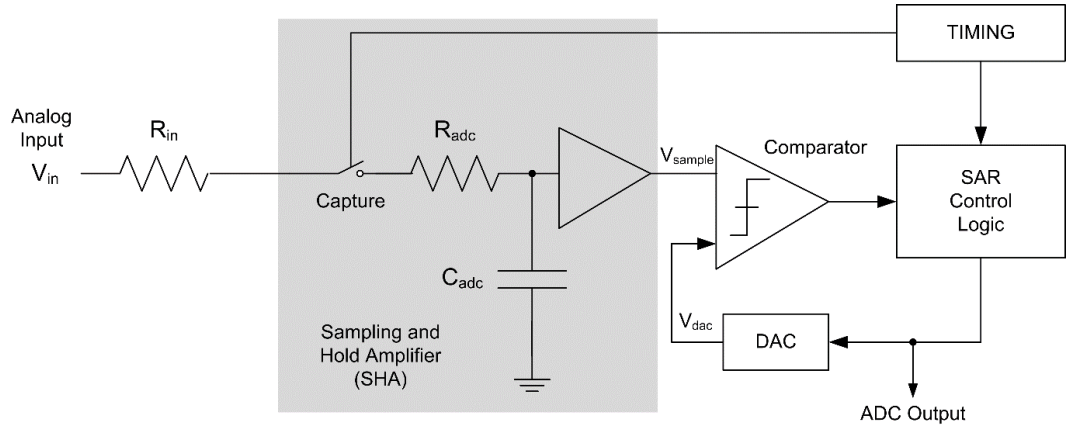

Figura 2 – Arquitetura de um conversar AD do tipo SAR.

O circuito *sampling and hold* 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 obter uma conversão precisa, o software deve deixar a chave de captura fechada pelo tempo suficiente para a correta carga do capacitor. O tempo na qual a chave de captura deve permanecer fechado é chamado de tempo de amostragem (*sampling time*).

Uma vez amostrada, a tensão de entrada é amplificada e fornecida à entrada do comparador (V_{sample}). A lógica do quantizador digital 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 conversor AD para que a tensão de saída do conversor DA gradualmente se aproxime da tensão amostrada. A conversão ocorre da seguinte forma:

1. A conversão inicia setando o bit mais significativo da saída do ADC, 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 lógica de controle mantém setado o bit mais significativo da saída do ADC. Caso contrário, o bit mais significativo é resetado;
3. Em seguida, o bit seguinte da saída do ADC 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 ADC tenham sido determinados;

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 conversão máxima.

Na Figura 3 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. Suponha que a tensão de entrada pode variar entre 0 e 1V (V_{ref}).

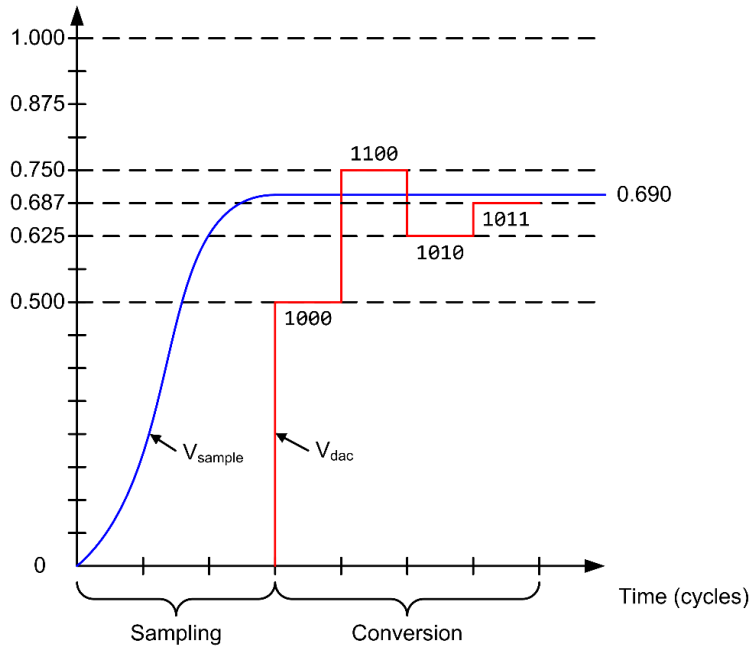

Figura 3 – Exemplo de uma conversão de um sinal de 0.69V com um ADC de aproximação sucessiva de 4 bits.

1. A lógica de controle inicia a conversão setando a tensão V_{dac} como 0,5V (isto é, a saída do ADC é 0b1000) e compara esse valor com V_{sample}.
2. Como V_{sample} é maior que V_{dac}, a lógica de controle seta V_{dac} como 0,75V (isto é, a saída do ADC é 0b1100);
3. Como V_{sample} é menor que V_{dac} a lógica de controle seta V_{dac} como 0,625 (isto é, a saída do ADC é 0b1010);
4. Como V_{sample} é maior que V_{dac}, a lógica de controle seta V_{dac} como 0,6875 (isto é, a saída do ADC é 0b1011);
5. Como V_{sample} ainda é maior que V_{dac}, a lógica de controle mantém o último bit setado e o processo é concluído. Isto é, a saída do ADC é 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 baterias 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 que pode ser alinhado 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 limites mínimo e máximo pré-definidos pelo usuário. A tensão de entrada pode variar de V_{ref-} (0V) a V_{ref+} (3,3V), onde V_{ref-} e V_{ref+} são dois pinos dedicados do microcontrolador e servem como fontes externas de tensão de referência.

As principais características dos conversores do STM32F407 são:

- Resolução configurável de 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;
- Modo de varredura para conversão automática do canal 0 até o canal N;
- Tempo de amostragem programável por canal;
- Opção de disparo externo com polaridade configurável;
- Modo descontínuo de conversão;
- Modo de conversão simultânea em todos os conversores;
- Atraso configurável entre conversões no modo simultâneo;
- Gatilhos por software ou sinais externos oriundos, por exemplo, de temporizadores internos ou de pinos do microcontrolador.

A Figura 4 mostra o diagrama de blocos de um dos ADCs do STM32F407.

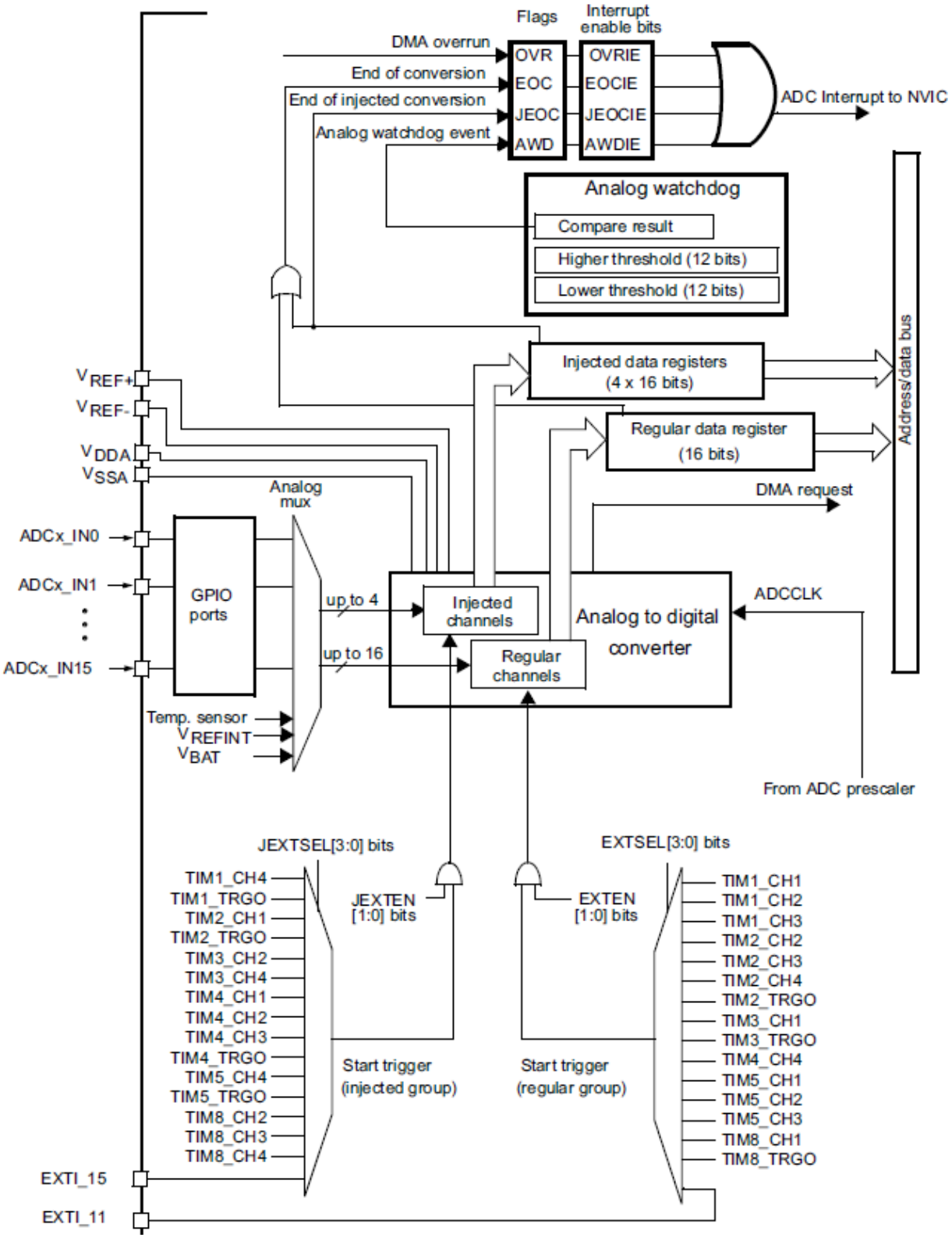

Figura 4 – Diagramde blocos de um dos ADCs do STM32F407.

O ADC é ligado configurando o bit ADON no registrador CR2 do módulo ADC. A conversão inicia quando o bit SWSTART ou JSWSTART, 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.

O ADC possui dois esquemas de clock:

• Clock para o circuito analógico, ADCCLK, comum a todos os ADCs. Este sinal é gerado a partir do clock do barramento APB2 dividido por um prescaler programável que permite que o ADC trabalhe em $f_{APB2}/2$, /4, /6 ou /8. O valor máximo de ADCCLK é de 36 MHz.
• Clock para a interface digital (para acesso aos registradores), igual ao 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.

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_IN2, ADC_IN15.

Um grupo regular é composto por até 16 conversões. Os canais regulares e sua ordem na sequência de conversão devem ser selecionados nos registradores SQR (*Sequence Register*) do módulo ADC. O número total de conversões no grupo regular deve ser gravado nos bits L[3: 0] no registrador SQR1 do módulo ADC.

Há um sensor de temperatura interno, conectado ao canal ADC1_IN16, que pode ser utilizado para medir a temperatura ambiente do microcontrolador numa faixa de -40 a 125 °C, com uma precisão de ±1,5 °C (0,9%).

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 vários ciclos do clock ADCCLK. A quantidade de ciclos usados na amostragem 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:

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

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

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

9.4. Modos de conversão

O módulo ADC executa conversões em canais selecionados nos modos de conversão simples e contínuo, como mostrado na Figura 5.

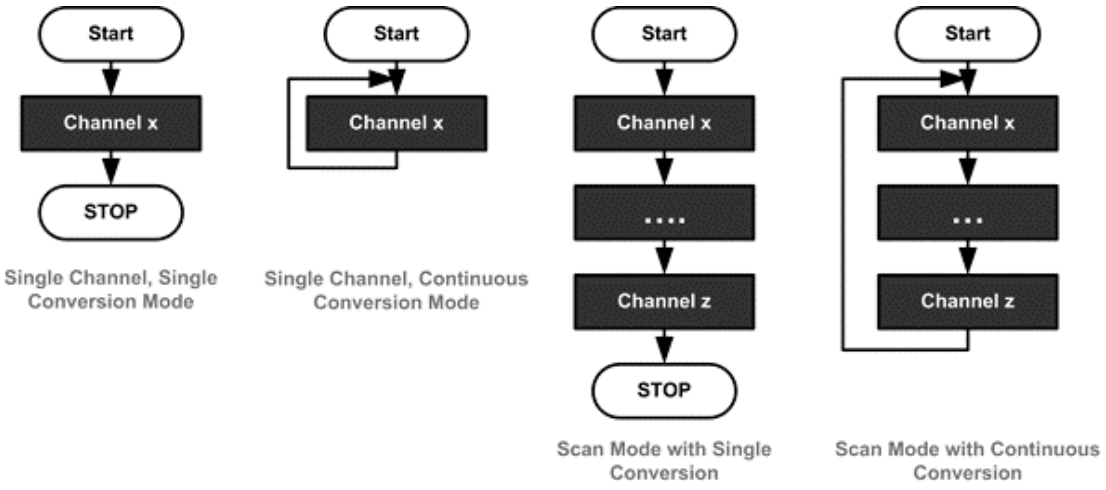

Figura 5 – Modos de conversão do ADC.

Consideremos a conversão de um único canal. O canal escolhido deve ser selecionado pelos bits SQ1[4:0] no registrador SQR3. O resultado da conversão é armazenado no registrador DR (*Data Register*) do módulo ADC. Depois da conversão, o flag EOC (*End Of Conversion*) é setado 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 gatilhamento da conversão, o canal selecionado é convertido uma vez e o ADC fica aguardando um novo sinal de gatilho (imagem mais à esquerda na Figura 5). Nesse modo, os bits SCAN, no registrador CR1, e CONT, no registrador CR2, devem ser feitos iguais a 0.

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 5). 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 chamado modo de escaneamento. Este modo converte, numa ordem pré-definida, os canais especificados nos registradores SQR. 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 um 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 gatilhamento e para após a conversão do último canal da lista (terceira imagem da esquerda para a direita na Figura 5). 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 (última figura da esquerda para a direita na Figura 5). Nesse modo, os bits CONT e SCAN devem ser feitos iguais a 1 e 1.

9.5. Alinhamento dos dados no ADC

O software pode modificar a resolução da conversão escrevendo nos bits RES[1:0] no registrador CR1, entretanto, o registrador do resultado da conversão (DR) tem 16 bits. O bit ALIGN no registrador CR2 seleciona o alinhamento dos dados armazenados após a conversão. Os dados podem ser alinhados à direita ou à esquerda, como mostrado na Figura 6. Portanto, apenas doze bits são significativos.

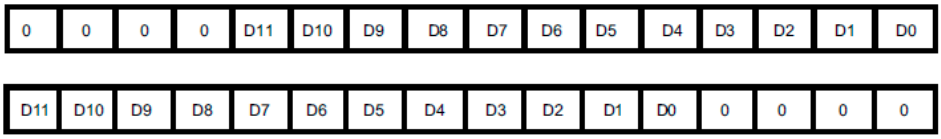

Figura 6 – Alinhamento dos dados armazenados: à direita (imagem superior); à esquerda (imagem inferior).

9.6. Pinos de entrada do ADC

É mostrado 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_14	PC4
ADC12_IN7	PA7	ADC12_15	PC5

9.7. Escalonamento de grandezas medidas pelo conversor A/D

Para uma resolução de 12 bits, o resultado da conversão do sinal analógico está compreendido entre 0 e 4095. 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 descrever 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 LM35, 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, embora este sensor seja especificado para operar apenas em temperaturas de até 150 °C. 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 7.

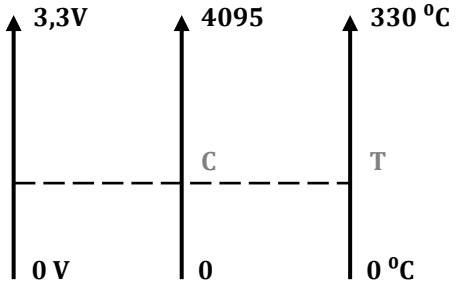

Figura 7 – Relação entre tensão fornecida pelo sensor, resultado da conversão do ADC e valor real da temperatura.

Analisando a relação acima, consegue-se demonstrar que a temperatura T equivalente a um resultado da conversão com um valor C é expressa por:

$$T = 0{,}08059 * C,$$

assim, esse deve ser o escalonamento feito para obtermos corretamente o valor de temperatura medido pelo sensor. De modo geral, para um sensor que fornece uma faixa de tensão de V_1 a V_2, resultando em uma faixa de variação de 0 a 2^n-1 como resultado da conversão de um conversor AD, correspondente à variação de uma grandeza física na faixa de F_1 a F_2, o escalonamento que deve ser feito é dado por:

$$F_y = \frac{F_x}{2^n - 1}(F_2 - F_1),$$

onde F_y é o valor real da grandeza física associada a um determinado valor F_x 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 gatilhamento feito por software.

```
void Configure_ADC()
{
  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=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 gatilhamento do ADC e aguarda o 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 pode ser usado para medir a temperatura ambiente do dispositivo. O sensor opera na faixa de –40 a +105 °C com uma precisão de ± 1,5 °C. A tensão de saída do sensor de temperatura varia linearmente com a temperatura. O deslocamento desta função linear depende de cada chip devido às variações do processo de fabricação (podendo chegar a até 45 °C de um chip para outro).

O sensor de temperatura interno é mais adequado para aplicações que detectam variações de temperatura em vez de temperaturas absolutas. Se a leitura precisa da temperatura for necessária, um sensor de temperatura externo deve ser usado.

O sensor é conectado internamente ao canal ADC1_IN16 que é usado para converter a tensão de saída do sensor em um valor digital. O bit TSVREFE do registrador CCR (*Common Control Register*) do módulo ADC deve ser setado para habilitar o sensor de temperatura e permitir a conversão do canal ADC1_IN16. 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 emitir 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 gatilho externo);
7. Ler os dados resultantes no registrador de dados DR do módulo ADC;
8. Calcular a temperatura usando a seguinte fórmula:

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

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 (não tratados) 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 entre o valor convertido (C) do sinal do sensor e a temperatura medida (T), conforme mostrado na Figura 8.

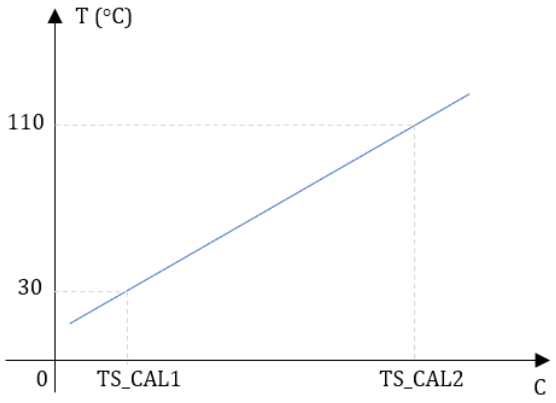

Figura 8 – Relação entre o valor convertido (C) do sinal do sensor e a temperatura medida (T).

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

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

Abaixo é mostrada um código que habilita e faz a leitura do sensor de temperatura interno, imprimindo o valor da temperatura a cada 500ms no terminal conectado à porta de comunicação USART1 (maiores detalhes sobre a USART1 serão vistos em capítulos posteriores).

```
void main()
{
  Configure_Clock();              //configura o sistema de clock
  USART1_Init();                  //inicializa a USART1
  Delay_Start();                  //inicializa funções de Delay

  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[0];                    //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

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

    //calcula a temperatura
    uint8_t temperatura = ((80*(int)(ADC1->DR - TS_CAL1))/(TS_CAL2-TS_CAL1))+30;
    printf("Temperatura = %d °C\n", temperatura);   //imprime a temperatura
    Delay_ms(500);                 //aguarda 500ms para fazer a nova leitura
  }
}
```

9.10. 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 ou 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 exibidas, 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 9.

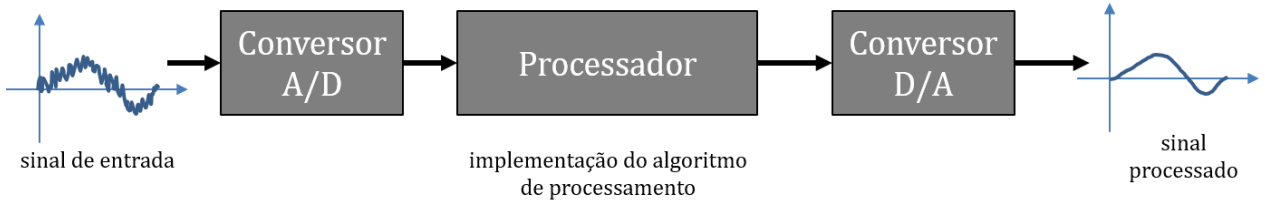

Figura 9 – Diagrama de blocos de um sistema de processamento digital de sinais.

Para exemplificar, consideremos um reprodutor de áudio MP3: Durante a fase de gravação, o áudio analógico é recebido por meio de um receptor (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 retirado 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 controlar itens como segurança, telefones, sistemas de *home theater* e compactação de vídeo. 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 aprimoradas).

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. ■