Nadpróbkowanie

Programowe zwiększenie rozdzielczości przetwornika A/C

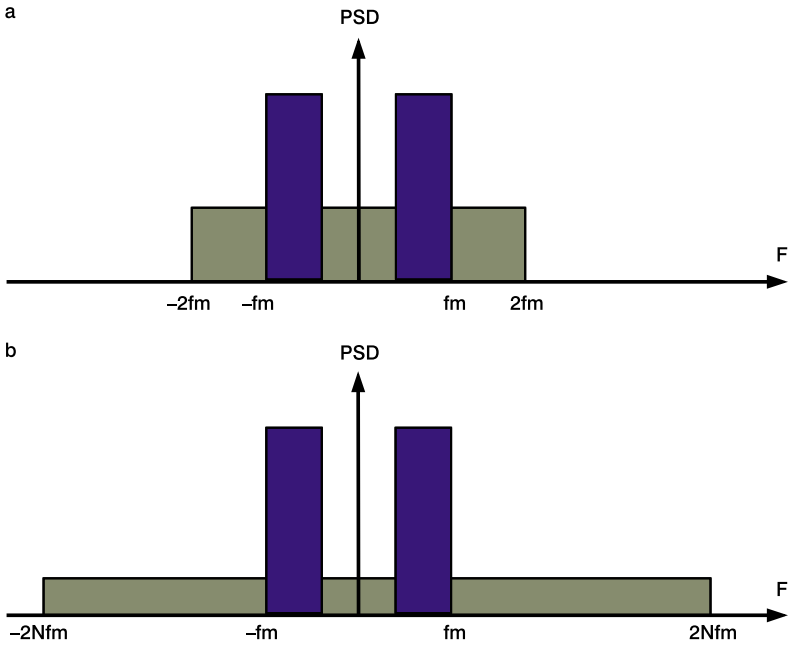
W niniejszym artykule w formie skróconej przedstawiam podstawy teoretyczne oraz praktyczny sposób wykorzystania oversamplingu (nadpróbkowania), dzięki któremu możliwe jest programowe zwiększenie rozdzielczości przetwornika A/C wbudowanego w mikrokontroler. Zalety takiego podejścia do problemu zbyt małej rozdzielczości przetwornika są w zasadzie dwie: po pierwsze, niższe koszty, gdyż nie musimy stosować przetwornika zewnętrznego, po drugie, uproszczenie połączeń oraz oszczędność miejsca na płytce drukowanej.

Nadpróbkowanie nie jest panaceum na wszystkie problemy związane z rozdzielczością przetwornika. Oczywiście, należy liczyć się z pewnymi ograniczeniami i mieć na względzie, że w niektórych warunkach sposób ten w ogóle nie będzie mógł być zastosowany, ponadto nie jest to droga do zastąpienia wysokiej jakości specjalizowanych przetworników A/C. Jednak warto zainteresować się tym sposobem, gdyż w wielu aplikacjach mikrokontrolerowych nie stawiamy wygórowanych wymagań przetwornikowi A/C, zaś przydałaby się większa rozdzielczość. Na zachętę dodam, że korzystając z oversamplingu, z powodzeniem zwiększyłem rozdzielczość przetwornika A/C w ATmega8535 do 12 bitów, co przy napięciu odniesienia równym 4,096 V dało teoretyczną, idealną rozdzielczość 1 mV.

Prezentowany artykuł powstał na podstawie dwóch not aplikacyjnych: *AVR121 – Enhancing ADC resolution by oversampling* oraz *AN2668: Improving STM32F101xx and STM32F103xx ADC resolution by oversampling*, dostępnych odpowiednio na stronach internetowych firm Atmel i STM. Nota aplikacyjna STM prezentuje dwa sposoby zwiększenia rozdzielczości przetwornika AD – w tym artykule zajmiemy się tylko sposobem wykorzystującym szum biały.

Podstawy teoretyczne

Częstotliwość próbkowania ustalona dla płyt CD ma taką nieco dziwną wartość równą 44,1 kHz. Najwyższa słyszana przez człowieka częstotliwość sygnału akustycznego wynosi około 20 kHz, więc zgodnie z twierdzeniem Nyquista (przypomnijmy, że mówi ono, iż w celu uniknięcia zniekształceń, częstotliwość próbkowania sygnału powinna być większa niż dwukrotność największej spodziewanej



Rys. 1. Wpływ oversamplingu na szum kwantyzacji

częstotliwości próbkowanego sygnału) wystarczyłoby próbkowanie z częstotliwością 40 kHz. Po co te dodatkowe 4,1 kHz? Warto w tym momencie wspomnieć, że próbkowanie z częstotliwością wyższą od wymaganej przez twierdzenie Nyquista nosi nazwę nadpróbkowania, od angielskiego terminu *oversampling*.

Kolejnym ważnym terminem, który musimy poznać, jest SNR (*Signal to Noise Ratio*). Zgodnie z zasadą działania przetwornika A/C, sygnał analogowy o nieskończonej liczbie stanów przetwarzany jest na wartość cyfrową o skończonej liczbie bitów, z czym wiąże się błąd kwantyzacji. Dla idealnego przetwornika A/C błąd kwantyzacji wynosi ±0,5 LSB (połowa najmniej znaczącego bitu). W przypadku, gdy mierzony sygnał przyjmuje różne poziomy pomiędzy próbkami oraz częstotliwość próbkowania nie jest zsynchronizowana z częstotliwością tego sygnału, błąd kwantyzacji można uznać za szum biały, którego moc jest równomiernie rozłożona w zakresie od napięcia stałego do połowy częstotliwości próbkowania. Dla idealnego przetwornika A/C pomijamy inne źródła szumu i przyjmujemy, że SNR to stosunek mocy szumu przetwornika A/C do mocy sygnału wejściowego, przy czym szum przetwornika odpowiada szumowi kwantyzacji. Dla pełnozakresowego sygnału sinusoidalnego, SNR przetwornika A/C przyjmuje maksymalną wartość, którą określa wzór:

$$SNR\ [dB]=6{,}02\cdot N+1{,}76$$

gdzie *N* to liczba bitów przetwornika A/C.

Z powyższego równania wynika, że aby zwiększyć rozdzielczość przetwornika A/C; wystarczy zwiększyć wartość parametru SNR, co uczynimy w dalszej części artykułu.

Przy założeniu, że szum kwantyzacji odpowiada szumowi białemu, jego moc jest równomiernie rozłożona w zakresie od napięcia stałego do połowy częstotliwości Nyquista, zaś samo skupienie mocy jest niezależne od częstotliwości próbkowana. Przy próbkowaniu z większymi częstotliwościami szum kwantyzacji rozciąga się na całą szerokość pasma, odpowiadającą częstotliwości próbkowania. Na **rys. 1** przedstawiono wpływ oversamplingu na szum kwantyzacji. Na **rys. 1a** pokazano normalną sytuację, gdy sygnał jest próbkowany zgodnie z twierdzenie Nyquista (tzn. z częstotliwością dwukrotnie większą niż maksymalna, spodziewana częstotliwość sygnału), zaś na **rys. 1b** sytuację, gdy sygnał jest próbkowany z częstotliwością dwukrotnie większą (oversampling) od częstotliwości Nyquista. Obszar niebieski reprezentuje sygnał wejściowy, zaś obszar zielony szum kwantyzacji. Porównując oba rysunki, można zauważyć, że obszar nakładania się szumu oraz próbkowanego sygnału (część wspólna obszaru niebieskiego i zielonego) zmniejsza się (rys. 1b) wraz ze wzrostem częstotliwości próbkowania, co pociąga za sobą zwiększenie SNR. Nadpróbkowanie sygnału z częstotliwością OSR-razy większą niż częstotliwość Nyquista pozwala osiągnąć SNR określony następującym wzorem:

$$SNR_{OVS}[dB]=6{,}02\cdot N+1{,}76+10\cdot\log(OSR)$$

Na podstawie powyższego równania można stwierdzić, że każde podwojenie częstotliwości próbkowania zwiększy SNR o 3 dB, zwiększając jednocześnie rozdzielczość pomiaru o 0,5 bitu. Aby zwiększyć rozdzielczość o 1 bit, należy więc uzyskać SNR o 6 dB większy. Ogólnie rzecz biorąc, aby zwiększyć rozdzielczość przetwornika A/C o n bitów, należy sygnał próbkować z częstotliwością $F_{OVS}=4^n\cdot F_S$, gdzie F_S jest aktualnie używaną częstotliwością próbkowania.

Uważny Czytelnik może teraz stwierdzić: próbkujemy sygnał ileś tam razy częściej i co z tego? Owa większa liczba próbek okaże się bardzo przydatna, co już wyjaśniam. Najprościej rzecz ujmując, aby uzyskać jeden bit więcej, należy sygnał próbkować czterokrotnie, wyniki zsumować, a następnie uśrednić. Ogólnie, aby uzyskać *N* bitów więcej, należy próbkować sygnał 4^N razy, otrzymane wyniki zsumować i uśrednić. Należy zwrócić uwagę na fakt, że uśrednienie wyników nie polega na podzieleniu sumy próbek przez ich liczbę. W zamian, należy sumę przesunąć o N bitów w prawo, co odpowiada jej podzieleniu przez 2^N (jest to tzw. decymacja). Od tej pory możemy cieszyć się wynikiem pomiaru o rozdzielczości większej o *N* bitów.

Ograniczenia

Aby oversampling mógł działać, w próbkowanym sygnale powinny znajdować się „śmieci" w postaci szumu o poziomie 1–2 LSB. Normalnie szum ten będzie pochodził ze źródeł takich jak: szum termiczny, CPU, przełączanie portów I/O czy wahania napięcia źródła zasilania, i będzie wystarczający. Dobrym rozwiązaniem jest podłączenie innych układów peryferyjnych do portu, w którym znajduje się wejście przetwornika AD. Czasem może się jednak zdarzyć, że szum pochodzący z wymienionych źródeł nie będzie wystarczający. W tym przypadku należy do próbkowanego sygnału dodać dodatkowy szum. W najgorszym przypadku oversamplingu w ogóle nie będzie można zastosować – po szczegóły odsyłam do wspomnianych not aplikacyjnych.

Należy pamiętać o tym, że oversampling zmniejsza maksymalną częstotliwość sygnału, którą możemy próbkować, co jest związane z maksymalną częstotliwością próbkowania przetwornika AD oraz liczbą dodatkowych bitów, które chcemy uzyskać; zależność tę określa wzór:

$$F_{max}=F_{ADC}max/(2\cdot 4^n)$$

gdzie:

F_{max} – maksymalna częstotliwość próbkowanego sygnału

F_{ADCmax} – maksymalna częstotliwość próbkowania przetwornika AD

n – liczba dodatkowych bitów, które chcemy uzyskać.

Układ testowy oraz oprogramowanie

W roli układu testowego wystąpił kit AVT-2550 „Mikrokomputer PECEL", w którym jako źródło napięcia odniesienia przetwornika AD zastosowałem układ MCP1541 o napięciu 4,096 V. Do kanału 6 przetwornika AD mikrokontrolera podłączone zostało środkowe wyprowadzenie potencjometru 10 kΩ (skrajne wyprowa-

Tab. 1. Zestawienie wyników pomiarów [V]

multimetr	ADC	różnica
0	0	0
0,028	0,020	0,008
0,150	0,143	0,007
0,415	0,409	0,006
1,058	1,051	0,007
1,726	1,719	0,007
2,258	2,252	0,006
2,827	2,822	0,005
3,248	3,242	0,006
4,09	4,092	–0,002

dzenia podłączone zostały odpowiednio do masy oraz napięcia 4,096 V źródła napięcia odniesienia). Ponadto, między masę układu a środkowe wyprowadzenie potencjometru podłączony został multimetr (UNITEST 9005), dzięki czemu możliwe stało się przeprowadzenie pomiarów. Wyniki pomiaru napięcia dokonane przetwornikiem AD oraz multimetrem zostały zestawione w **tab. 1**. Jak widać, różnica między pomiarami dokonanymi przez przetwornik AD oraz multimetr jest nieznaczna i kształtuje się na poziomie 0,007 V. Różnica ta najprawdopodobniej spowodowana jest niedoskonałością przetwornika AD (oraz multimetru) – dokładne sposoby kalibracji przetwornika AD można znaleźć w odpowiednich notach aplikacyjnych, zaś w najprostszym przypadku wystarczy uwzględnić średnią różnicę w programie, kosztem nieznacznego błędu dokonywanych pomiarów.

Układem testowym steruje program napisany w języku C++ (avr-gcc), przedstawiony na **list. 1**. Głównym zadaniem programu jest pomiar napięcia z przetwornika AD skonfigurowanego w trybie Free-running oraz wysyłanie tej wartości na wyświetlacz LCD. Należy zwrócić uwagę na fakt, że przed wysłaniem wartości do LCD, napięcie z przetwornika próbkowane jest 16-krotnie, a następnie suma próbek przesuwana jest o 2 bity w prawo. Założeniem było zwiększenie rozdzielczości przetwornika AD z 10 do 12 bitów (n=2), czyli: $4^2=16$ – liczba wymaganych próbek, 2 – o tyle bitów należy wynik przesunąć w prawo. Dodatkowe informacje zawarte zostały w komentarzach, dzięki czemu pełne zrozumienie programu nie powinno stanowić problemu.

List. 1.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include „lcd.h”

void adc_start();
void adc_convert();

// zmienna przechowujaca wynik konwersji
volatile uint16_t conv_result;

int main()
{
        // ustaw preskaler ADC na 64, co daje czestotliwosc jego pracy
wynoszaca ~172,8kHz
        // dla kwarcu 11,0592MHz.
        ADCSRA |= _BV(ADPS2) | _BV(ADPS1);

        // wybor kanalu ADC (kanal 6)
        ADMUX = 6;

        // wlacz przerwanie od ADC
        ADCSRA |= _BV(ADIE);
        // wlacz ADC
        ADCSRA |= _BV(ADEN);

        // wlacz obsluge przerwan
        sei();

        // rozpocznij konwersje
        adc_start();

        while( 1 )
        {
                lcd.clrscr();
                // wypisz wynik konwersji
                lcd << conv_result;
                _delay_ms( 500 );
        }

        return 0;
}

void adc_start()
{
        // rozpocznij konwersje
        ADCSRA |= _BV(ADSC);
}

void adc_convert()
{
        static uint8_t sample_no;
        static uint16_t sample_sum;

        // sumuj kolejne probki z przetwornika
        sample_sum += ADC;
        sample_no++;

        // jesli pobrano 16 probek
        if( sample_no == 16 )
        {
                // wyzeruj licznik probek
                sample_no = 0;
                // zapisz wynik konwersji,
                // uwzgledniajac wymagane przesuniecie o 2 bity w prawo
                conv_result = sample_sum>>2;
                // wyzeruj sume probek
                sample_sum = 0;
        }

        // rozpocznij konwersje
        adc_start();
}

ISR( ADC_vect )
{
        adc_convert();
}
```

Podsumowanie

Przedstawiony artykuł ma z założenia formę skrótową i zawiera jedynie zarys informacji z zakresu oversamplingu. Po szczegółowe informacje, między innymi odnośnie do częstotliwości próbkowania oraz wymagań co do próbkowanego sygnału, odsyłam do wspomnianych not aplikacyjnych.

Uwaga praktyczna: jeżeli zależy nam na w miarę przyzwoitej jakości przetwarzania wartości napięcia na postać cyfrową, to nie warto stosować napięcia odniesienia pobieranego z systemowego stabilizatora (np. 7805), lecz raczej wykorzystać dedykowane, specjalizowane źródło napięcia odniesienia, o wartości np. 2,048 V lub 4,096 V. Po pierwsze, unikniemy błędów związanych z zaokrąglaniem (liczby 2048 i 4096 w obliczeniach będą dzieliły się bez reszty), po drugie, stabilizator 7805 rzadko kiedy da na wyjściu dokładnie 5 V. Rzeczywistą wartość napięcia będzie trzeba mierzyć w każdym nowo budowanym urządzeniu i odpowiednio modyfikować tę wartość w programie. Dla przykładu, zastosowanie źródła napięcia odniesienia o wartości 4,096 V w miejsce napięcia ze stabilizatora 7805 zaoszczędziło mi wyrwania wszystkich włosów z głowy, gdy przetwarzałem wartość napięcia z czujnika temperatury LM235 (przy stabilizatorze 7805 wyniki były zadziwiająco rozbieżne).

Andrzej Telszewski
atelszewski@gmail.com

Kreujemy bezprzewodową rzeczywistość...
„Przemysłowy Ethernet”

Rabbit – RCM5700

Darmowe środowisko programistyczne Dynamic C
10/100Base-T Ethernet z konektorem RJ45
Standard MiniPCIExpress
Wydajna platforma sprzętowa
Wbudowany Web serwer
Wsparcie dla projektantów
Pełna certyfikacja

Multitech – Socket Ethernet IP

Kompatybilność pin-to-pin z modemami Multitech’a pracującymi w innych technologiach sieciowych
Zgodny z normą EN60601 do zastosowań medycznych
Stos Univeral IP rozszerzający funkcjonalność M2M
Kompletne rozwiązanie Serial to Ethernet
Sterowanie komendami AT

Teridian – seria 78Q21xx

Rozwiązanie dla projektów Set Top Box, IP Video
Zintegrowana warstwa MAC i PHY
Tryb transmisji synchronicznej i asynchronicznej
Transmisja Full Duplex z funkcją autonegocjacji
Pojedyncze napięcie zasilania 3,3 V
Łatwa integracja

Współpracujemy ze światowymi liderami!!!

ACTE

antenova® · Radiocrafts Embedded Wireless Solutions · RABBIT · Digi · SIERRA WIRELESS
EAD embedded antenna design ltd · SKYLAB · connectBlue · eRide · wavecom Make it wireless
CAP-XX · TERIDIAN SEMICONDUCTOR CORP · bluegiga · enocean · MultiTech Systems

ACTE Sp. z o.o., ul. Krańcowa 49, 02-493 Warszawa, tel.: 022 33 60 200, faks: 022 33 60 201, email: acte@acte.pl