Mikrokontrolery ARM

Laboratorium 6

Krzysztof Pierczyk
15 maja 2020

WSTĘP

Na ostatnich zajęciach laboratoryjnych zajęliśmy się wykorzystaniem zintegrowanego przetwornika analogowo-cyfrowego obecnego w mikrokontrolerach z rodziny STM32. Naszym zadaniem było napisanie aplikacji, która wykorzystując przetwornik potrafiłaby mierzyć napięcie RMS (*ang. root mean square*) sygnału podanego na wejście układu. Aplikacja powinna wypisywać zmierzoną wartość poprzez port szeregowy oraz wizualizować ją z wykorzystaniem diod LED poprzez ich zapalanie w momencie, gdy wartość napięcia przekroczy pewien próg.

IMPLEMENTACJA

Aplikacja została podzielona na **trzy wątki**. Pierwszy z nich zajmował się monitorowaniem stanu przycisku *USER*. Po jakiego wciśnięciu wątek inkrementował wypełnienie fali prostokątnej generowanej przez układ TIM5. Drugi wątek obliczał napięcie RMS oraz kontrolował zapalanie diod. Jako, że konwersja analogowa cyfrowa odbywała się cyklicznie w takt przepełnienia układu TIM2, a zakodowane próbki były stale przenoszone do globalnego bufora kołowego, rola wątku sprowadzała się do przeliczenia próbek znajdujących się z buforze zgodnie ze wzorem:

$$V_{RMS} = \sqrt{\frac{{x_1}^2 + {x_2}^2 + \cdots + {x_n}^2}{n}}$$

Oraz zaktualizowaniu stanu diod LED. Ostatni wątek zajął się raportowaniem stanu pomiarów na port szeregowy.

Najważniejszą częścią programu była konfiguracja układów peryferyjnych. Wykorzystane układy czasowo licznikowe TIM2 i TIM5 były już niejednokrotnie wykorzystywane na poprzednich zajęciach, dlatego przyjrzymy się bliżej jedynie konfiguracji kanału DMA oraz samego przetwornika. Można tu jedynie wspomnieć, że częstotliwość licznika TIM2 została ustawiona na 48kHz i został on wykorzystany jako źródło sygnału inicjalizującego konwersję, natomiast układ TIM5 skonfigurowano do generowania na pinie PA1 sygnału PWM.

Konfiguracja przetwornika, chociaż sprawiła sporo nieoczekiwanych kłopotów, sprowadziła się ostatecznie do wypełnienia dwóch struktur danych i wykorzystaniu ich do zainicjalizowania układu. Powyższa konfiguracja uaktywnia przetwornik z **rozdzielczością 12 bitów** pracujący w **trybie SINGLE** i aktywowany **zewnętrznym sygnałem TRGO2** pochodzącym z układu TIM2. Dodatkowo wprowadzony została każdorazowa aktywacja żądania dla kanału DMC, gdy pojedyncza konwersja zostaje zakończona.

```
void ADC_config(){

    LL_APB2_GRP1_EnableClock(LL_APB2_GRP1_PERIPH_ADC1);
    LL_ADC_SetCommonClock(
       __LL_ADC_COMMON_INSTANCE(ADC1),
       LL_ADC_CLOCK_SYNC_PCLK_DIV8
    );

    LL_ADC_InitTypeDef adc_struct{
        .Resolution         = LL_ADC_RESOLUTION_12B,
        .DataAlignment      = LL_ADC_DATA_ALIGN_RIGHT,
        .SequencersScanMode = LL_ADC_SEQ_SCAN_DISABLE
    }; LL_ADC_Init(ADC1, &adc_struct);


    LL_ADC_SetChannelSamplingTime(
         ADC1, LL_ADC_CHANNEL_4 ,
         LL_ADC_SAMPLINGTIME_56CYCLES
    );


    LL_ADC_REG_InitTypeDef adc_reg_struct{
        .TriggerSource    = LL_ADC_REG_TRIG_EXT_TIM2_TRGO,
        .SequencerLength  = LL_ADC_REG_SEQ_SCAN_DISABLE,
        .SequencerDiscont = LL_ADC_REG_SEQ_DISCONT_DISABLE,
        .ContinuousMode   = LL_ADC_REG_CONV_SINGLE,
        .DMATransfer      = LL_ADC_REG_DMA_TRANSFER_UNLIMITED
    }; LL_ADC_REG_Init(ADC1, &adc_reg_struct);
    LL_ADC_REG_SetSequencerRanks(ADC1, LL_ADC_REG_RANK_1 , LL_ADC_CHANNEL_6);

    LL_ADC_Enable(ADC1);
    LL_ADC_REG_StartConversionExtTrig(ADC1, LL_ADC_REG_TRIG_EXT_RISING);

 }
```

Rysunek 1. Konfiguracja przetwornika ADC

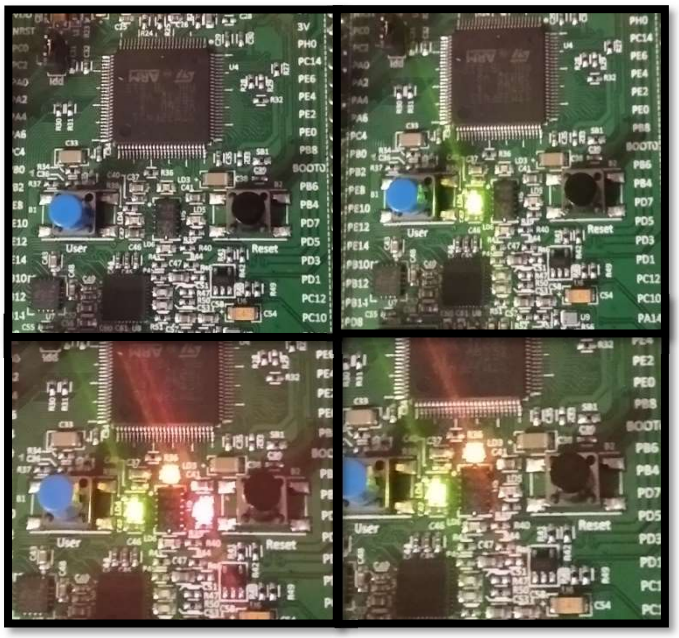
Mniej problematyczna okazała się konfiguracja kanału DMA. Żądanie od przetwornika ADC może być odebrane m.in. w kanale 0 potoku 0 znajdującego się w układzie DMA. Ten też został wykorzystany w ćwiczeniu. Kanał został skonfigurowany na ciągłą, kołową transmisję z układu ADC do pamięci. Zakończenie konwersji o długości równej pojemności bufora nie jest sygnalizowane przerwaniem.

```
void DMA_config(){

    NVIC_SetPriority(DMA2_Stream0_IRQn , 1);
    NVIC_EnableIRQ(DMA2_Stream0_IRQn);

    LL_AHB1_GRP1_EnableClock(LL_AHB1_GRP1_PERIPH_DMA2);

    LL_DMA_SetChannelSelection(DMA2, LL_DMA_STREAM_0 , LL_DMA_CHANNEL_0);
    LL_DMA_InitTypeDef dma_struct{
       .PeriphOrM2MSrcAddress  =
           LL_ADC_DMA_GetRegAddr(ADC1, LL_ADC_DMA_REG_REGULAR_DATA),
       .MemoryOrM2MDstAddress  = (uint32_t)samples,
       .Direction              = LL_DMA_DIRECTION_PERIPH_TO_MEMORY,
       .Mode                   = LL_DMA_MODE_CIRCULAR,
       .PeriphOrM2MSrcIncMode  = LL_DMA_PERIPH_NOINCREMENT,
       .MemoryOrM2MDstIncMode  = LL_DMA_MEMORY_INCREMENT,
       .PeriphOrM2MSrcDataSize = LL_DMA_PDATAALIGN_HALFWORD,
       .MemoryOrM2MDstDataSize = LL_DMA_MDATAALIGN_HALFWORD,
       .NbData                 = WINDOW_SIZE,
       .Channel                = LL_DMA_CHANNEL_0,
       .Priority               = LL_DMA_PRIORITY_HIGH
    }; LL_DMA_Init(DMA2, LL_DMA_STREAM_0, &dma_struct);

    LL_DMA_EnableIT_TE(DMA2, LL_DMA_STREAM_0);

    LL_DMA_EnableStream(DMA2, LL_DMA_STREAM_0);
}
```

Rysunek 2. Konfiguracja kanału DMA

Powyższa konfiguracja umożliwiła zrzucenie odpowiedzialności za wykonywanie i agregację pomiarów na warstwę sprzętową, co znacznie ułatwiło pozostałą część programu. Dodatkowym atutem takiego rozwiązania jest możliwość zarządzania obliczenia wartości RMS w dowolnym momencie, ponieważ próbki do tego potrzebne zawsze dostępne są w buforze. W tym miejscu warto wspomnieć, że długość bufora została ustalona na 100, co pozwala przechować nieco ponad dwa okresy generowanego sygnału PWM.

Stworzony program działa zgodnie z przewidywaniami. Wypisuje on teoretyczną i mierzoną wartość napięcia RMS na porcie szeregowym oraz wizualizuje kolejne progi napięcia z wykorzystaniem „linijki” ledowej.



Rysunek 3. Wizualizacja poziomu napięcia RMS z wykorzystaniem diod LED

POMIARY DOKŁADNOŚCI

Zgodnie z treścią zadania przeprowadzone zostały pomiary dokładności skonstruowanego miernika RMS. Zebrano 33 pomiary, po 3 dla kolejnych poziomów napięć w przedziale 150mV – 2900mV. Wyniki zostały przedstawione w Tabeli 1. Uśredniając powyższe wyniki otrzymujemy błąd bezwzględny o wartości 10,94mV. Aby jednak pokazać pełnię tego wyniku należy także dodać, że odchylenie standardowe błędu bezwzględnego to około **48,80mV**. Jeżeli chodzi o błąd względny, to jego wartość średniokwadratowa wyniosła **29.11%.**

Jak można się było spodziewać, wyniki te nie są wybitne, a rzec nawet można, że marne. Podobne osiągi można uzyskać w przypadku większości zintegrowanych układów tego typu. Zaletą takiego typu rozwiązań jest zmniejszenie ilości elementów w układzie, a co za tym idzie rozmiaru płytki drukowanej i poboru prądu. Ponadto takie przetworniki są o wiele

łatwiejsze w zastosowaniu, gdyż wykorzystują wewnętrzną strukturę mikrokontrolera. Nie mogą one jednak konkurować z dedykowanymi układami scalonymi, ponieważ znajdują się one na jednym kawałku krzemu wraz z dużą ilością obwodów cyfrowych, przez co są słabo izolowane i bardzo podatne na zakłócenia.

WNIOSKI

Możliwości użytkowanego mikrokontrolera STM32 w domenie analogowej nie są porywające. Plasują się na podobnym poziomie jak w konkurencyjnych układach. Mogą się one sprawdzić do implementowania prostych czujników, sterowników, czy regulatorów o niskim stopniu skomplikowania. Połączone z możliwościami DSP mikrokontrolerów od ST mogą one stanowić solidną bazę do tworzenia np. urządzeń Internetu Rzeczy lub. Nie aspirują one do miana układów nadających się w urządzeniach audio, czy precyzyjnych jednostkach pomiarowych. Ich największą zaletą jest, jak już wspomniano, prostota wykorzystania w potencjalnym urządzeniu.

Tworzony w ramach zajęć miernik *True RMS* daleko odbiega od ideału. Duża ilość urządzeń cyfrowych w otoczeniu mikrokontrolera, taktowanych wysokimi częstotliwościami wprowadza sporą dawkę zniekształceń do pomiarów. Aby temu zaradzić można by wprowadzić na wejściu konwertera prosty filtr dolnoprzepustowy, który przepuściłby jedynie częstotliwości interesujące z punktu widzenia pomiaru.

Innej modyfikacji można z kolei dokonań w przypadku (nie implementowanego tutaj) przetwornika DAC, obecnego w niektórych mikrokontrolerach z rodziny STM32. Jego wysoka impedancja wyjściowa sprawia, że może on mieć kłopot z wysterowaniem niektórych układów. Aby temu zaradzić, wystarczyłoby umieścić prosty bufor prądowy w postaci np. wzmacniacza nieodwracającego o jednostkowym wzmocnieniu.

Pomiar [mV]	Wartość teoretyczna [mV]	Błąd [mV]
1,61 1,61 2,42	147.65	146,04 146,04 145,23
466,48 329,52 466,47	417,63	-48,85 88,11 -48,84
659,84 571,22 659,03	643,61	-16,23 72,39 -15,42
932,96 808,08 807,28	898,15	-34,81 90,07 90,87
1142,43 1042,53 1142,43	1143,73	1,30 101,2 1,10
1359,96 1399,44 1318,87	1400,77	40,81 1,33 81,90
1616,16 1615,36 1616,16	1650,83	34,67 35,47 34,67
1865,92 1865,92 1865,92	1902,39	36,47 36,47 36,47
2137,86 2085,86 2111,65	2149,88	12,02 64,02 38,23
2378,32 2332,40 2378,32	2399,10	20,78 66,70 20,78
2597,46 2597,46 2597,46	2649,56	52,01 52,01 52,01
2837,55 2837,55 2856,95	2900,95	63,40 63,40 44,00
Błąd średniokwadratowy:		10,94

Tabela 1. Wyniki pomiarów dokładności skonstruowanego czujnika