



Akademia Górniczo-Hutnicza
w Krakowie
Instytut Elektroniki
WIET



Laboratorium

Technika Mikroprocesorowa 2

Ćwiczenie 7

Port szeregowy UART0

Komunikacja z komputerem PC



Autor: Mariusz Sokołowski

wer. 28.09.2021

1. WSTĘP

1.1.CEL

Celem ćwiczenia jest:

- zapoznanie studenta z podstawami techniki poprawnej inicjalizacji i obsługi asynchronicznego portu szeregowego UART,
- poznanie podstaw łączenia funkcjonalnego portu szeregowego z komputerem klasy IBM-PC, do dwukierunkowej wymiany informacji,
- poznanie możliwości wykorzystania terminala szeregowego (komputer) do:
 -  obrazowania wyników pomiarowych, układów zaimplementowanych w oparciu o moduł FRDM-KL05Z,
 -  sterowania peryferiami modułu FRDM-KL05Z.

1.2.WYMAGANIA

Sprzętowe:

- komputer klasy PC, spełniający wymagania sprzętowe aplikacji KEIL v5,
- zestaw FRDMKL05Z

Programowe:

- system operacyjny Windows 7 lub wyższy (wszystkie instrukcje powstały w oparciu o Windows 7 Pro x64),
- środowisko Keil / uVision 5 MDK-ARM

Doświadczenie:

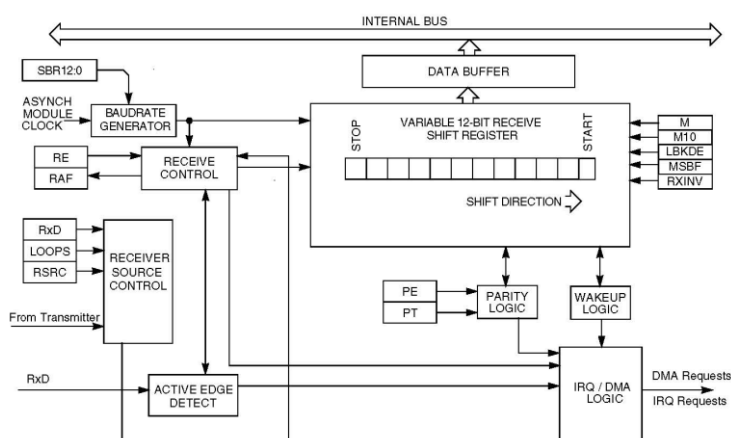
- podstawowa umiejętność obsługi komputera klasy PC,
- podstawowa znajomość systemów operacyjnych rodziny Windows,
- umiejętność programowania w języku C.

Literatura:

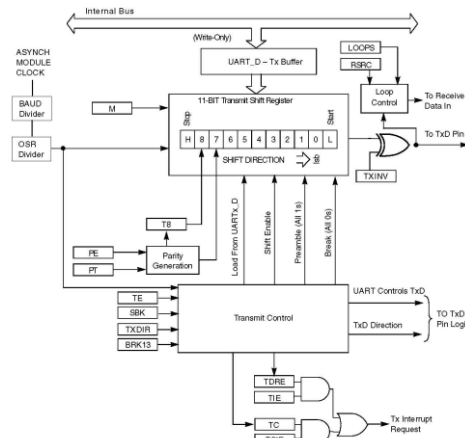
- KL05 Sub-Family Reference Manual, Freescale Semiconductor
- Kinetis L Peripheral Module Quick Reference, Freescale Semiconductor
- Joseph Yiu, The Definitive Guide to the ARM Cortex-M0, Elsevier, 2011
- ARM „Cortex-M0+ Devices Generic User Guide”
- Ambient Light Sensor Surface - Mount ALS-PT19-315C/L177/TR8 datasheet
- Kinetis KL05 32 KB Flash - 48 MHz Cortex-M0+ Based Microcontroller, Data Sheet
- PuTTY User Manual (terminal)

2. MODUŁ PORTU SZEREGOWEGO UART0

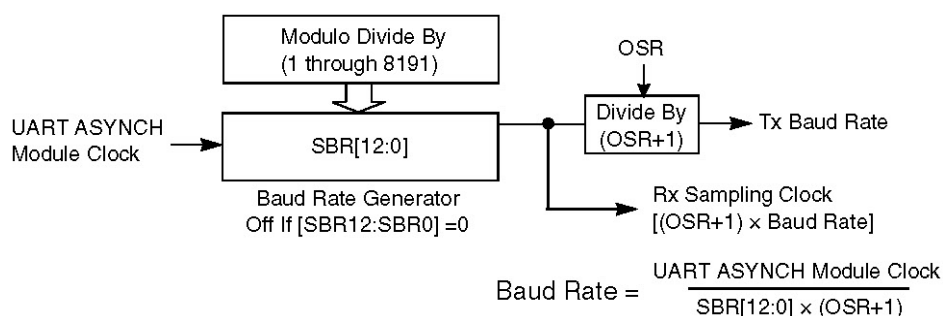
Układ MKL05Z32VLC4, będący centralnym mikrokontrolerem zestawu FRDM-KL05Z, posiada w swoich zasobach sprzętowych, jeden port szeregowy – UART0 (Rys. 1, Rys. 2 i Rys. 3).



Rys. 1 Układ odbiornika UART0



Rys. 2 Układ nadajnika UART0



Rys. 3 Generator taktujący nadajnik i odbiornik (Baud Generator)

Wybrane, najważniejsze cechy układu to:

- długość danej 8, 9 lub 10 bitów,
- możliwość nadawania i odbierania w tym samym czasie (Full-duplex),
- 13-bitowy dzielnik zegara taktującego, dający możliwość uzyskania różnych prędkości transmisji,
- sprzętowa generacja i weryfikacja bitu parzystości,
- wybór jednego lub dwóch bitów stopu,
- monitoring błędów:
 - Overrun Error - brak możliwości zapisania do bufora odbiornika nowej danej (skompletowanej w odbiorniku), spowodowany nieodczytaniem poprzedniej wartości z bufora,
 - Noise Error - wykrycie niestłości stanu, dla danego, odczytywanego bitu,
 - Framing Error - błąd ramki, spowodowany wykryciem stanu „0” dla bitu stopu, który powinien mieć wartość „1”,
 - Parity Error - wartość otrzymanego bitu parzystości nie zgadza się z zawartością danej.

- „oversampling” (próbkowanie nadmiarowe) - zwielokrotnianie częstotliwości próbkowania w odbiorniku, w stosunku do generatora taktującego (Baud Generator) - Rys. 3.

Prędkość transmisji BR jest kreowana w oparciu o częstotliwość wybranego zegara taktującego (SIM_SOPT2[UARTSRC]) oraz zawartość rejestrów UART0_BDH[SBR], UART0_BDL[SBR] i UART0_C4 [OSR]. Wielkości te są powiązane następującym wzorem:

$$BR = \frac{UART0SRC}{(OSR + 1) * SBR}$$

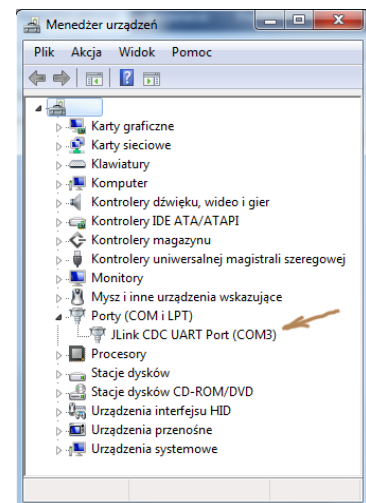
Aby przygotować układ UART0 do pracy, należy wykonać następujące czynności:

- ❖ Sprawdzić, jaką wartość ma stała CLOCK_SETUP, w zbiorze *system_MKL05Z4.c*. Informacja ta będzie miała wpływ na ustawienia rejestrów, odpowiedzialnych za szybkość transmisji. W zależności od stałej CLOCK_SETUP, parametry podstawowych sygnałów zegarowych mają następujące wartości:
 - CLOCK_SETUP=0 (wartość domyślna):
 - zegar referencyjny dla modułu MCG - 32768Hz,
 - Core clock - 41943040Hz,
 - BusClock - 20971520Hz.
 - CLOCK_SETUP=1:
 - zegar referencyjny dla modułu MCG - 32768Hz,
 - Core clock - 47972352Hz,
 - BusClock - 23986176Hz.
 - CLOCK_SETUP=2:
 - zegar referencyjny dla modułu MCG – 4MHz,
 - Core clock – 4MHz,
 - BusClock – 2MHz.
- ❖ dołączyć sygnał taktujący do modułu UART0, w rejestrze SIM_SCGC4 [UART0=1],
- ❖ dołączyć sygnał taktujący do odpowiedniego portu, którego końcówki realizują funkcje TX (PTB1) i RX (PTB2), w rejestrze SIM_SCGC5[PORTB=1],
- ❖ ustawić odpowiednią funkcję dla wykorzystywanych końcówek portu, w rejestrze PORTB_PCRx[MUX=2]. PTB1 - nadajnik TX, PTB2 - odbiornik RX,
- ❖ ustawić źródło zegara taktującego moduł UART0, w rejestrze SIM_SOPT2[UARTSRC=1], MCGFLLCLK=Core clock,
- ❖ zablokować nadajnik i odbiornik, w rejestrze UART0_C2[RE=0, TE=0],
- ❖ ustawić wartość dzielnika próbkowania nadmiarowego, w rejestrze UART0_C4 [OSR]=15,
- ❖ ustawić możliwość próbkowania w odbiorniku, danych przychodzących, na obydwu zboczach zegara, w rejestrze UART0_C5[BOTHEDGE=1], w przypadku, gdy wartość OSR jest

wybrana z przedziału od 3 do 6 (patrz powyżej). Dla innych ustawień OSR, opcja ta leży w gestii programisty,

- ❖ ustawić 13-bitową wartość dzielnika, będącego źródłem zegara dla odbiornika i nadajnika. Najpierw starsze 5 bitów ustawić w rejestrze `UART0_BDH[SBR]`, a następnie młodsze 8 bitów w rejestrze `UART0_BDL[SBR]` - wziąć pod uwagę nastawioną wartość OSR oraz wzór na BR:
 - `BR=9600:` `CLOCK_SETUP=0, BDH[SBR=1], BDL[SBR=17],`
 - `BR=28800:` `CLOCK_SETUP=0, BDH[SBR=0], BDL[SBR=91],`
 - `BR=230400:` `CLOCK_SETUP=0, BDH[SBR=0], BDL[SBR=11].`
- ❖ ustawić jeden bit stopu, w rejestrze `UART0_BDH[SBNS=0]`,
- ❖ ustawić długość danej na 8 bitów oraz brak sprzętowej obsługi sprawdzania parzystości, w rejestrze `UART0_C1[M=0, PE=0]`,
- ❖ w zależności od potrzeb, włączyć przerwania od nadajnika i/lub odbiornika, w rejestrze `UART0_C2[TIE=1 i/lub RIE=1]`. Nadajnik zgłasza przerwanie, gdy bufor nadajnika jest pusty, a odbiornik, gdy bufor odbiornika jest pełny. Jeżeli chodzi o nadajnik, to jest jeszcze jedna możliwość zgłaszania przerwania: gdy nadajnik skończył transmisję (`UART0_C2[TCIE=1]`),
- ❖ włączyć nadajnik i odbiornik, w rejestrze `UART0_C2[TE=1, RE=1]`.

W tym momencie układ UART0 jest gotowy do pracy. Nadawanie polega na wpisywaniu danej 8-bitowej do rejestru `UART0_D`, a odbiór, to po prostu odczyt tego rejestru. Jeśli nie używamy przerwań do komunikacji z nadajnikiem i odbiornikiem, stan nadajnika sprawdzamy poprzez odczyt wartości bitu TDRE w rejestrze `UART0_S1`, a stan odbiornika poprzez odczyt wartości bitu RDRF w tymże rejestrze. TDRE=1 – rejestr nadajnika pusty, RDRF=1 – odbiornik pełny. W przypadku wybrania opcji zgłaszania końca transmisji (TCIE), sprawdzany jest bit `UART0_S1[TC]`, TC=1 – transmisja zakończona. Jeżeli nadajnik nie jest gotowy na następną daną, nie należy jej tam zapisywać.



Rys. 4 Numer portu szeregowego

Od strony komputera należy sprawdzić, uruchamiając menedżera urządzeń, który port COM jest przypisany do naszego modułu (kabel USB dołączony do złącza OpenSDA) - Rys. 4. Pisząc oprogramowanie lub łącząc się poprzez terminal, właśnie tego numeru portu COM należy używać. Od strony modułu, komunikacja jest zapewniona poprzez dołączone końcówki PTB1 i PTB2 do mikrokontrolera PK20DX128VFM5, realizującego interfejs Open-SDA.

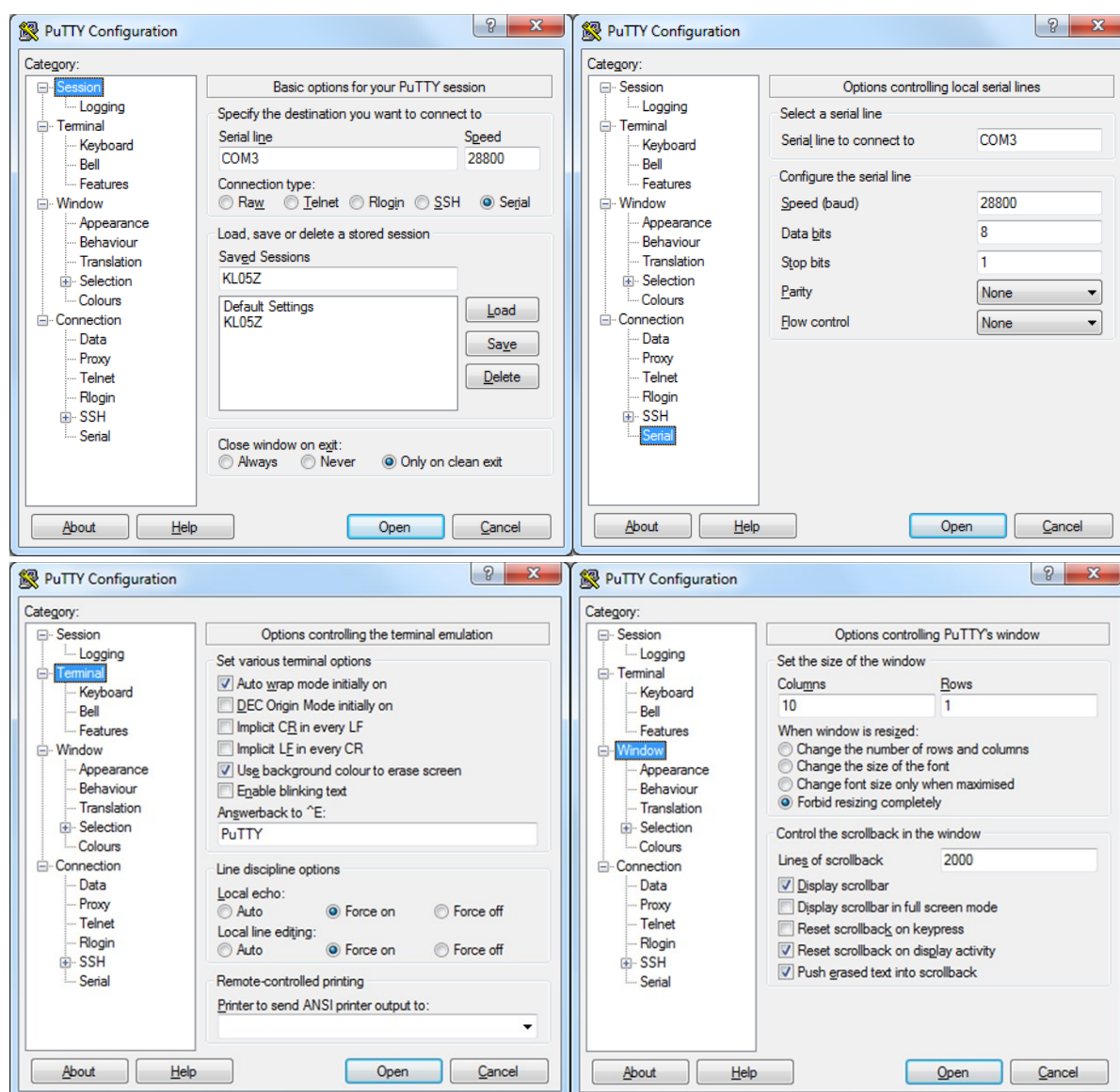
Ponieważ istnieje jeden wektor obsługujący port UART0, w podprogramie obsługi należy najpierw sprawdzić, które z urządzeń, nadajnik czy odbiornik, jest źródłem przerwania. Jeśli są również włączone przerwania od błędów, to krąg „sprawców” się powiększa. Po ustaleniu źródła przerwania, odpowiednia flaga, w rejestrze `UART_S1` powinna zostać skasowana. Flaga RDRF (od odbiornika) kasowana jest automatycznie, w chwili odczytu bufora danych odbiornika (`UART0_D`). Flagi TDRE i TC (od nadajnika) są kasowane automatycznie, w chwili zapisu bufora nadajnika (`UART0_D`). Pozostałe flagi należy wyzerować programowo, wpisując w odpowiednią pozycję rejestru `UART0_S1` wartość „1”.

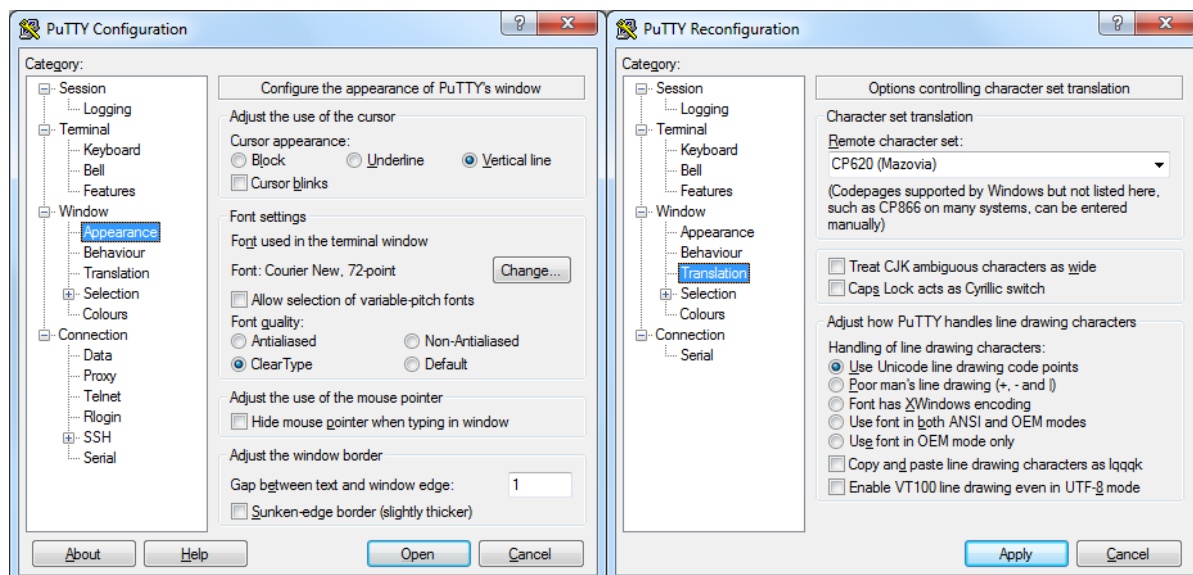
3. ĆWICZENIE NR 1 – NAWIAZYWANIE POŁĄCZENIA Z KOMPUTEREM

Rozpakować zbiór *1_UART0_Prawybuch.zip*. Zbiór zawiera również katalog *Terminal* z prostym programem terminalowym (*PuTTY.exe*).

Ponieważ w ćwiczeniach będą używane funkcje operujące na wartościach zmiennoprzecinkowych, zwiększono pojemność stosu, ustawiając go na wartość: **Stack_Size EQU 0x00000300**, w zbiorze *startup_MKL05Z4.s*.

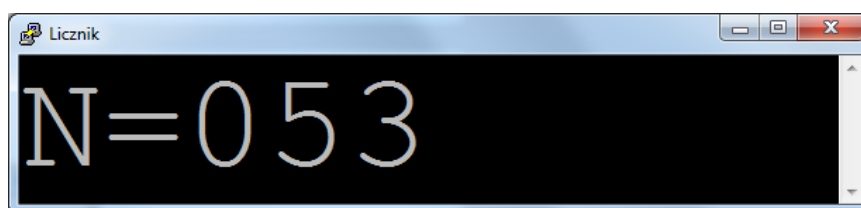
Uruchomić projekt *UART0_Prawybuch.uvprojx*, a na komputerze program *PuTTY.exe*. Na wyświetlaczu LCD pojawi się zachęta do wybrania, za pomocą pola dotykowego, wartości prędkości transmisji BR. Pole dotykowe jest podzielone na trzy równe części. Dotknięcie części z lewej strony ustawia BR=9600 b/s, w środku BR=28800 b/s, a z prawej BR=230400 b/s. Po wyświetleniu wybranej wartości, układ czeka, aż identyczna wartość BR zostanie ustawiona w terminalu. Wszystkie ustawienia programu PuTTY są przedstawione na rysunku Rys. 5.





Rys. 5 Ustawienia programu PuTTY

Po ustawieniu parametrów terminala, ponowne dotknięcie pola dotykowego spowoduje rozpoczęcie transmisji z modułu KL05Z do komputera. Program generalnie liczy od 0 do 255, co 1. Jeśli wszystkie ustawienia przeprowadzono zgodnie z zaleceniami, na ekranie komputera będą wyświetlane kolejne wartości licznika (Rys. 6).



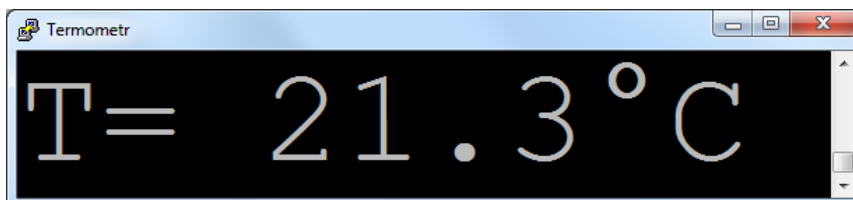
Rys. 6 Ekran z wynikiem

3.1.ZADANIE 1

- ✓ Przeanalizować program.
- ✓ Wykorzystując zadanie domowe z Ćwiczenia 6 („Przetwornik analogowo-cyfrowy A/C”, pkt. 4, „ĆWICZENIE NR 1 - PROGRAMOWE WYZWALANIE PRZETWORNIKA A/C, W TRYBIE AUTOMATYCZNYM”), dotyczące czujnika światła, tak zmodyfikować w/w program, aby zamiast licznika wyświetlany był wynik pomiaru napięcia z czujnika światła.

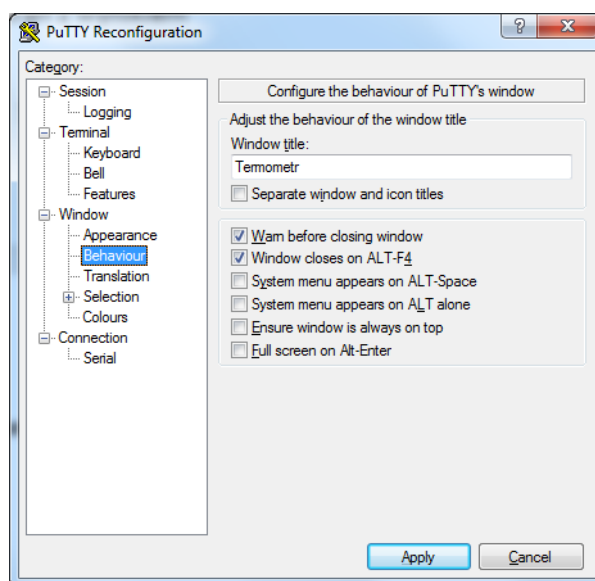
3.2. ZADANIE DODATKOWE

- ✓ Wykorzystując zadanie dodatkowe z Ćwiczenia 6 („Przetwornik analogowo-cyfrowy A/C”, pkt. 10, „ĆWICZENIE 7 - MIERNIK TEMPERATURY”), dotyczące termometru, tak zmodyfikować w/w program, aby zamiast licznika była wyświetlana temperatura (Rys. 7). Kod ASCII znaku „°”, dla kodowania CP620 (Mazovia), które ustawiono w terminalu (patrz Rys. 5), to 0xF8. Na wyświetlaczu LCD należy zachować również znaczek „°”.



Rys. 7 Ekran z wynikiem

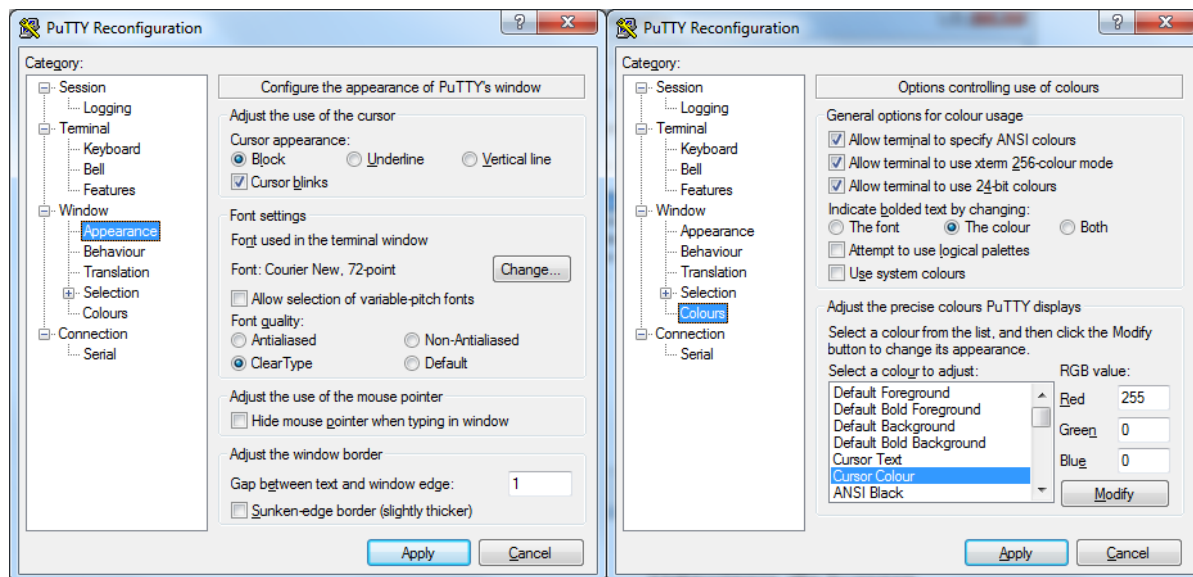
Nazwę okienka (Licznik lub Termometr) można zmienić w ustawieniu pokazanym na rysunku Rys. 8. Można tego dokonać w trakcie wyświetlania pomiarów, klikając prawym przyciskiem myszy na ramce okienka terminala i wybierając „Change Settings...”.



Rys. 8 Ustawienia nazwy okienka

4. ĆWICZENIE NR 2 – TESTOWANIE ODBIORNIKA, W TRYBIE Z PRZERWANIAMI

Uruchomić program *PuTTY.exe*. Zmaksymalizować okno terminala. Prędkość transmisji dla terminala ustawić BR=28800. Włączyć kursor: mruganie, typ i kolor (Rys. 9).



Rys. 9 Ustawienia dla kursora

Rozpakować zbiór *2_UART0_Przerwania.zip*. Uruchomić projekt *UART0_Przerwania.uvprojx*. W polu terminala wpisujemy dowolny tekst i wciskamy Enter. Dane są odbierane przez odbiornik UART0 i zwrótnie wysyłane z powrotem do terminala. Pojawia się echo. I tak wkoło Macieju.

4.1.ZADANIE 2

- ✓ Przeanalizować program. W momencie wysłania danej przez terminal, uruchamia się odbiornik UART0 i po skompletowaniu danej zgłasza przerwanie, informując, że dana jest gotowa do odczytu. W podprogramie obsługi tego przerwania dana jest odczytywana i przekazywana do pętli głównej, a tam z powrotem jest odsyłana do komputera, itd.

5. ĆWICZENIE NR 3 – STEROWANIE URZĄDZENIAMI ZEWNĘTRZNYMI ZA POMOCĄ TERMINALA

Rozpakować zbiór `3_UART0_Sterowanie.zip`. Uruchomić projekt `UART0_Sterowanie.uvprojx` i program `PuTTY.exe`, którego ustawienia są takie same jak w pkt. 4.

Program, w zależności od przesłanej z terminala komendy, zapala lub gasi czerwoną diodę LED. Komenda „LRON” zapala, a „LROFF” gasi. Podanie złej komendy powoduje wyświetlenie, w oknie odbiornika terminala, komunikatu „Zła komenda”. Podanie zbyt długiego ciągu znaków (większego od 15) powoduje wyświetlenie, w oknie odbiornika terminala, komunikatu „Zbyt długi ciąg”.

5.1.ZADANIE 3

- ✓ Przeanalizować program
- ✓ Zrealizować sterowanie pozostałymi diodami LED. Wymyślić swoje komendy, które będą zaświecać i gasić pozostałe diody LED (G i B).