Obsługa inteligentnych wyświetlaczy LED za pomocą STM32

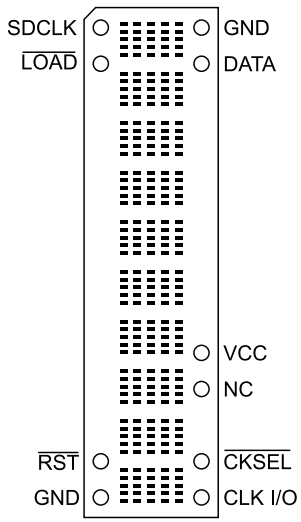
SCD5510XA firmy OSRAM to rodzina 10-znakowych, alfanumerycznych wyświetlaczy LED o wielkości znaku 5×5 punktów. Są one dostępne w różnych kolorach, w przykładowym projekcie użyto wersji SCD55104A (kolor zielony, LED o podwyższonej jasności).

Dodatkowe informacje:
Wyświetlacze SCD55104A udostępniła redakcji firma EBV, autoryzowany dystrybutor wyrobów firmy Osram.

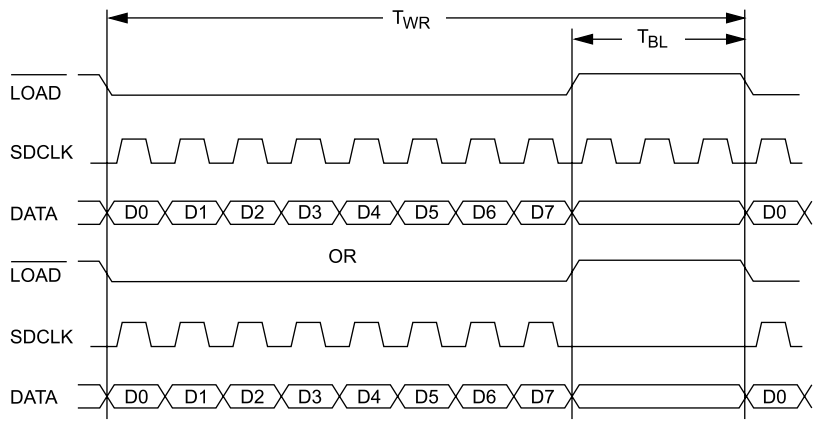
Dodatkowe materiały na CD/FTP:
ftp://ep.com.pl, user: 12040, pass: 15735862

W ofercie firmy Osram są dostępne również wyświetlacze z inną liczbą znaków oraz ze znakiem wielkości 5×7 punktów, sterowanie nimi odbywa się w podobny sposób. Rozmieszczenie wyprowadzeń wyświetlacza ilustruje **rysunek 1**. Funkcje poszczególnych wyprowadzeń są następujące (rysunek 1):

- SDCLK – linia zegarowa interfejsu szeregowego



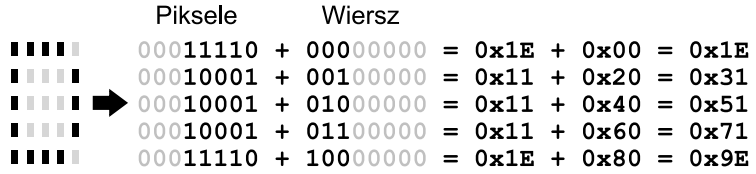
Rysunek 1. Rozmieszczenie wyprowadzeń wyświetlacza OSRAM SCD5510XA



Rysunek 2. Ramka danych

Tabela 1. Komenda Load Character Address

Komenda			Parametr					Szesnastkowo	Operacja
D7	D6	D5	D4	D3	D2	D1	D0		
1	0	1	1	0	0	0	0	0xB0	Znak 0
1	0	1	1	0	0	0	1	0xB1	Znak 1
1	0	1	1	0	0	1	0	0xB2	Znak 2
1	0	1	1	0	0	1	1	0xB3	Znak 3
1	0	1	1	0	1	0	0	0xB4	Znak 4
1	0	1	1	0	1	0	1	0xB5	Znak 5
1	0	1	1	0	1	1	0	0xB6	Znak 6
1	0	1	1	0	1	1	1	0xB7	Znak 7
1	0	1	1	1	0	0	0	0xB8	Znak 8
1	0	1	1	1	0	0	1	0xB9	Znak 9

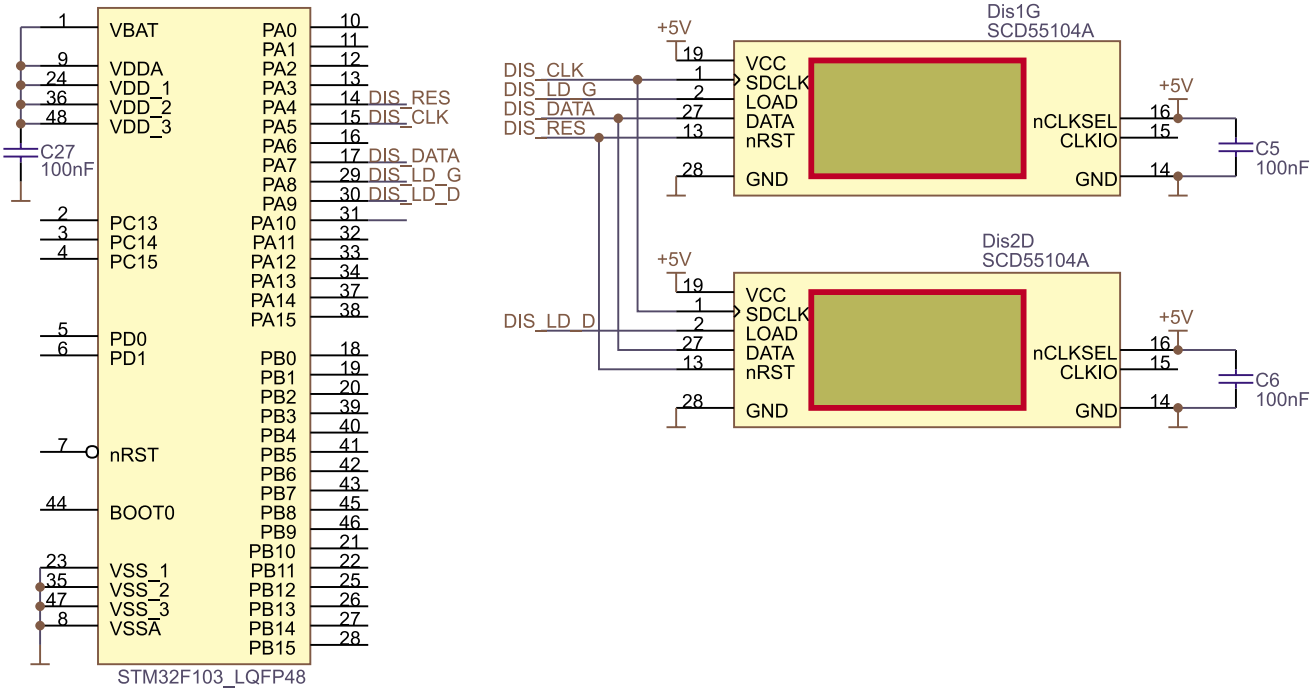


Rysunek 3. Sposób wyliczenia wartości bajtów dla znaku „D"

Tabela 2. Ładowanie danych znaku

Wybór wiersza			Dane					Numer wiersza
D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	C0	C1	C2	C3	C4	0
0	0	1	C0	C1	C2	C3	C4	1
0	1	0	C0	C1	C2	C3	C4	2
0	1	1	C0	C1	C2	C3	C4	3
1	0	0	C0	C1	C2	C3	C4	4

- /LOAD – podanie stanu niskiego na tę linię odblokowuje interfejs szeregowy wyświetlacza
- /RST – podanie stanu niskiego na tę linię powoduje wyzerowanie pamięci układu sterującego wyświetlaczem
- CLK I/O – w zależności od stanu /CKSEL wejście lub wyjście sygnału zegarowego interfejsu szeregowego
- /CKSEL – jeśli na tę linię podamy stan niski, to interfejs szeregowy będzie taktowany sygnałem zegarowym generowanym przez wyświetlacz, wtedy CLK I/O jest wyjściem tego sygnału. W innym przypadku CLK I/O jest wejściem sygnału zegarowego.



Rysunek 4. Schemat elektryczny połączeń pomiędzy wyświetlaczami i mikrokontrolerem STM32F103

Listing 1. Inicjalizacja wyświetlacza

```
void SCD5510XA_InitDisplay(void)
{
  // Enable clock for GPIOA, SPI1
  RCC->APB2ENR |= RCC_APB2ENR_SPI1EN | RCC_APB2ENR_IOPAEN;
  // Configure GPIOA lines
  // pins 5, 7 as Alternate-function push-pull output (50 MHz), pin 4 as general purpose output (50 MHz)
  GPIOA->CRL = GPIOA->CRL & ~(GPIO_CRL_MODE4 | GPIO_CRL_MODE5 | GPIO_CRL_MODE7 | GPIO_CRL_CNF4 | GPIO_CRL_CNF5 | GPIO_CRL_CNF7);
  GPIOA->CRL |= GPIO_CRL_MODE4_0 | GPIO_CRL_MODE4_1 |
  GPIO_CRL_MODE5_0 | GPIO_CRL_MODE5_1 | GPIO_CRL_CNF5_1 |
  GPIO_CRL_MODE7_0 | GPIO_CRL_MODE7_1 | GPIO_CRL_CNF7_1;
  // Configure GPIOA lines
  //  - pins 8, 9 as general purposu output (50 MHz)
  GPIOA->CRH = GPIOA->CRH & ~(GPIO_CRH_MODE8 | GPIO_CRH_CNF8 | GPIO_CRH_MODE9 | GPIO_CRH_CNF9);
  GPIOA->CRH |= GPIO_CRH_MODE8_0 | GPIO_CRH_MODE8_1 | GPIO_CRH_MODE9_0 | GPIO_CRH_MODE9_1;
  // Set /LOAD high
  GPIOA->ODR |= GPIO_ODR_ODR8;
  // Configure SPI1
  // Master, LSB first, 8-bit frame, fPCLK/16, CLK high on idle, second edge
  // For 72 MHz SYSCLK SPI frequency is 4.5 MHz
  SPI1->CR1 = SPI_CR1_LSBFIRST | SPI_CR1_BR_0 | SPI_CR1_BR_1 | SPI_CR1_MSTR | SPI_CR1_CPOL | SPI_CR1_CPHA;
  SPI1->CR2 = SPI_CR2_SSOE;
  // Enable SPI
  SPI1->CR1 |= SPI_CR1_SPE;
  // Reset display
  GPIOA->ODR |= (1 << 4);
  GPIOA->ODR &= ~(1 << 4);
  DispDelay();
  GPIOA->ODR |= (1 << 4);
  // Clear displays memories
  SCD5510XA_SelectDisplay(1);
  SCD5510XA_ClearDisplay();
  SCD5510XA_SelectDisplay(2);
  SCD5510XA_ClearDisplay();
}
```

Listing 3. Wybór aktywnego wyświetlacza
```
void PA8_Low()    { GPIOA->ODR &= ~(1 << 8); }
void PA8_High()   { GPIOA->ODR |= (1 << 8); }
void PA9_Low()    { GPIOA->ODR &= ~(1 << 9); }
void PA9_High()   { GPIOA->ODR |= (1 << 9); }

void SCD5510XA_SelectDisplay(char dispNum)
{
  switch (dispNum){
    case 1:
      nLOAD_High = PA8_High;
      nLOAD_Low = PA8_Low;
      break;
    case 2:
      nLOAD_High = PA9_High;
      nLOAD_Low = PA9_Low;
      break;
  }
}
```

Listing 2. Funkcja wstrzymująca wykonywanie programu na t>=600 ns
```
void DispDelay(void)
{
  int i;
  for (i=0; i<0xFF; i++);
}
```

Listing 4. Wysłanie bajtu do aktywnego wyświetlacza
```
void SPI_Transmit(char cData)
{
  while ((SPI1->SR & SPI_SR_TXE)!=SPI_SR_TXE);
  nLOAD_Low();
  SPI1->DR = cData;
  while ((SPI1->SR & SPI_SR_BSY)==SPI_SR_BSY);
  nLOAD_High();
  DispDelay();
}
```

Listing 5. Fragment deklaracji tablicy *font*
```
static char font[]={
0x00, 0x00, 0x00, 0x00, 0x00,       // 0 - Null
…
0x0e, 0x33, 0x55, 0x79, 0x8e,       // 48 - Digit Zero
0x04, 0x2C, 0x44, 0x64, 0x8E,       // 49 - Digit One
0x1E, 0x21, 0x46, 0x68, 0x9F,       // 50 - Digit Two
0x1e, 0x21, 0x4e, 0x61, 0x9e,       // 51 - Digit Three
0x06, 0x2A, 0x5f, 0x62, 0x82,       // 52 - Digit Four
0x1f, 0x30, 0x5e, 0x61, 0x9e,       // 53 - Digit Five
0x06, 0x28, 0x5E, 0x71, 0x8E,       // 54 - Digit Six
0x1F, 0x22, 0x44, 0x88, 0x88,       // 55 - Digit Seven
0x0E, 0x31, 0x4E, 0x71, 0x8E,       // 56 - Digit Eight
0x0E, 0x31, 0x4F, 0x62, 0x8C,       // 57 - Digit Nine
…
0x00, 0x00, 0x00, 0x00, 0x00,       // 255 - Dot Above
};
```

List. 6. Funkcja *main*
```
int main(void)
{
  int charNum, brightness;
  char buffer[11];
  System_Config();
  SCD5510XA_InitDisplay();
  SCD5510XA_SelectDisplay(1);
  SCD5510XA_WriteString("SCD5510XA");
  charNum = 31;
  brightness = 0;
  SCD5510XA_SelectDisplay(2);
  while(1)
  {
    if (charNum > 254)
    {
      charNum = 31;
      SCD5510XA_ClearDisplay();
    }
    if (IsCharEmpty(++charNum)) continue;
    if (++brightness > 6) brightness = 0;
    SCD5510XA_SetDisplayBrightness(brightness);
    sprintf(buffer, "Char %d=%c", charNum, charNum);
    SCD5510XA_WriteString(buffer);
    Delay_1s();
  }
}
```

Tabela 3. Komendy wyboru jasności świecenia wyświetlacza

Komenda	Jasność świecenia wyświetlacza
0xF0	100%
0xF1	53%
0xF2	40%
0xF3	27%
0xF4	20%
0xF5	13%
0xF6	6,6%

- DATA – linia danych interfejsu szeregowego. Stan linii jest odczytywany na narastającym zboczu zegara
- VCC, GND – zasilanie oraz masa

Ramkę danych przedstawiono na **rysunku 2**, kolejne ramki powinny być wysyłane w odstępach co najmniej 600 ns, a okres zegara musi być większy bądź równy 200 ns, co odpowiada częstotliwości linii zegarowej 5 MHz. Aby wyświetlić znak na wyświetlaczu, trzeba wysłać komendę *Load Character Address* z odpowiednim parametrem, który informuje wyświetlacz o tym, którym segmentem chcemy sterować (**tabela 1**).

Następnie należy wysłać pięć bajtów, każdy zawiera informacje o jednym wierszu. Trzy najstarsze bity decydują, który wiersz pikseli ładujemy, pięć najmłodszych bitów zawiera informację o tym, które piksele chcemy zapalić (**tabela 2**), bit o wartości „1” oznacza zapalony piksel, a bit o wartości „0” – zgaszony.

Przykładowo chcąc wyświetlić na wyświetlaczu literę „D” w drugim segmencie, wysyłamy bajt o wartości 0xB0, aby wybrać drugi segment, następnie bajty 0x1E, 0x31, 0x51, 0x71, 0x9E, sposób wyliczenia wartości tych bajtów widać na **rysunku 3**.

Wyświetlacz ma możliwość wyboru jasności świecenia pikseli, dostępne są wartości 100%, 53%, 40%, 27%, 20%, 13% i 6,6%. Mimo dużej różnicy w wartościach skoku między poziomami jasności (od 47% do 6,4%) zmiany odbierane są jako równomierne. Komendy służące do konfiguracji jasności znajdują się w tabeli 3. Domyślna jasność świecenia wyświetlacza to 100%. Producent zaznacza, że przy jasności 100% nie powinno się zapalać więcej niż 128 pikseli jednocześnie.

Wyświetlacz można przełączyć w tryb *Power-Down*, wysyłając bajt 0xFF, pobór prądu spada wtedy do 50 μA. Aby wyzerować pamięć wyświetlacza (czyli również zgasić wszystkie piksele), należy wysłać komendę *Software Clear*, czyli bajt o wartości 0xC0.

Platforma sprzętowa

W przykładowej aplikacji wykorzystano dwa, połączone quasi-równolegle, wyświetlacze, które są sterowane przez mikrokontroler STM32F103, a konkretnie przez wbudowany w niego interfejs SPI1 (**rysunek 4**). Ponieważ komunikacja odbywa się w jedną stronę (od mikrokontrolera do wyświetlaczy), wystarczy skonfigurować dwie linie SPI: linię zegarową SCLK (PA5) dołączoną do wejść SDCLK wyświetlaczy oraz wyjście MOSI (PA7) dołączone do wejść DATA. Linia PA4 dołączona jest do wejść /RST. Linie PA8 i PA9 dołączone są do wejść /LOAD wyświetlaczy, służą do wyboru wyświetlacza, do którego chcemy wysłać dane, aby trafiły do pierwszego wyświetlacza, trzeba podczas transmisji ustawiać stan niski na wyjściu PA8 i wysoki na PA9, dla wyświetlacza drugiego odwrotnie. Wejścia /CKSEL dołączone są do VCC, ponieważ sygnał zegarowy podamy wyświetlaczowi z interfejsu SPI1 mikrokontrolera.

Biblioteka w języku C

Aby ułatwić Czytelnikom zastosowanie wyświetlacza SCD5510XA we własnych aplikacjach, stworzyłem bibliotekę do jego obsługi w języku C. Funkcje w bibliotece są podzielone na dwie grupy: funkcje zależne od platformy sprzętowej oraz funkcje od niej niezależne. Funkcja SCD5510XA_InitDisplay odblokowuje zegary dla portu A oraz interfejsu SPI1, konfiguruje linie PA5 i PA7 jako *alternate-function push-pull* (dla SPI1), PA4 (linia /RST), PA8 i PA9 (linie /LOAD) jako wyjścia *push-pull*. Następnie konfigurowane jest SPI1 (opis konfiguracji w **listingu 1**, po

szczegóły odsyłam do dokumentacji ST – RM0008 Reference Manual), na końcu podawany jest na chwilę stan niski na linię /RST, aby wyzerować pamięć wyświetlaczy i, jak zaleca dokumentacja SCD5510XA, wysyłana jest komenda *Software Clear* do obu wyświetlaczy.

Funkcja *DisplayDelay()* jest wywoływana między transmisjami kolejnych bajtów przez SPI, czas wykonania tej funkcji powinien wynosić co najmniej 600 ns (**listing 2**).

Zamieszczona na **listingu 3** funkcja *SCD5510XA_SelectDisplay()* służy do wyboru wyświetlacza, do którego będziemy transmitować dane. W zależności od wartości parametru *dispNum* wskaźniki nLOAD_High i nLOAD_Low zawierają adresy funkcji sterujących linią PA8 lub PA9.

Funkcja SPI_Transmit (**listing 4**) wysyła bajt do wybranego wyświetlacza zależnie od wartości parametru wcześniej wywołanej funkcji SCD5510XA_SelectDisplay. Pętla *while* oczekująca na wyzerowanie flagi BSY rejestru SR zapobiega wywołaniu funkcji, na którą wskazuje wskaźnik *nLOAD_High* przed zakończeniem transmisji. Jeśli mikrokontroler nie czekałby na wyzerowanie tej flagi, to stan wysoki podany byłby na linię /LOAD wybranego wyświetlacza zbyt wcześnie i transmisja nie powiodłaby się.

Przeniesienie biblioteki na inną platformę sprzętową wymaga dostosowania powyższych funkcji do wykorzystywanego mikrokontrolera. Aby dodać kolejne wyświetlacze, należy rozszerzyć funkcję SCD5510XA_SelectDisplay. Pozostałe funkcje są niezależne od sprzętu:

- *SCD5510XA_DisplayPowerDown* – przełącza wyświetlacz w tryb Power-Down przez wysłanie komendy 0xFF.
- *SCD5510XA_ClearDisplay* – zeruje pamięć wyświetlacza przez wysłanie komendy 0xC0.
- *SCD5510XA_SetDisplayBrightness (unsigned char brightnessLevel)* – ustawia jasność świecenia wyświetlacza w zależności od wartości parametru: 0 – 100%, 1 – 53%, 2 – 40%, 3 – 27%, 4 – 20%, 5-13%, 6 – 6,6%.
- *SCD5510XA_WriteChar (char col, char character)* – wyświetla znak o kodzie ASCII równym wartości parametru character w kolumnie col.
- *SCD5510XA_WriteString(char * s)* – wyświetla na wyświetlaczu łańcuch *s*.

Ponieważ wyświetlacz nie ma wbudowanej pamięci z czcionką, w pliku *font.h* umieszczono tablicę fontów zawierającą czcionkę 5×5, wzór każdego znaku jest zdefiniowany przez 5 kolejnych bajtów (**listing 5**), każdy znak jest umieszczony w osobnym wierszu i jest opatrzony komentarzem. Czcionka jest wzorowana na tej z dokumentacji wyświetlacza SCD5510XA. Czcionka nie definiuje wszystkich znaków (m.in. polskich liter), można ją uzupełnić we własnym zakresie.

Przykładowy program

Przykładowy projekt przygotowany jest dla środowiska Keil uVision. Funkcję *main* przedstawiono na **listingu 6**. Najpierw wywołana zostaje funkcja *System_Config*, która dokonuje konfiguracji mikrokontrolera, oprócz standardowej konfiguracji (praca z HSE, zegar 72 MHz, konfiguracja PLL itp.) konfigurowany jest SysTick, który jest wykorzystany do odmierzenia sekundowych opóźnień. Potem wywołana jest omówiona wcześniej funkcja *SCD5510XA_InitDisplay*, następnie wybierany jest pierwszy wyświetlacz *(SCD5510XA_SelectDisplay(1))* i wyświetlany jest na nim napis „SCD5510XA". Potem wybierany jest drugi wyświetlacz i w pętli wyświetlane są kolejne niepuste znaki z tablicy font i zmieniana jest jasność świecenia wyświetlacza.

Jacek Zbysiński