### Animacje na LCD



## Tworzenie animacji na wyświetlaczu LCD za pomocą STM32

Chociaż w handlu jest dostępnych coraz więcej kolorowych wyświetlaczy LCD, to jednak przeważnie nie są one przeznaczone na rynek masowy i dlatego sa dosvć drogie. Z tego powodu ogromną popularnością cieszą się wyświetlacze z telefonów komórkowych, ponieważ są produkowane w ogromnych ilościach i dlatego dosyć tanie. Można je również łatwo pozyskać na rynku wtórnym, więc gdy zachodzi potrzeba zastosowania w urządzeniu niewielkiego wyświetlacza LCD w projekcie, warto rozważyć użycie takiego przeznaczonego dla telefonu komórkowego lub odtwarzacza multimedialnego. W artykule opisano sposób dołaczenia i sterowania wyświetlaczem LCD od telefonu komórkowego Nokia E51.

Wyświetlacz z telefonu Nokia E51 ma przekątną ekranu 2,2". Część jego powierzchni zajmuje ramka, więc obszar aktywny ma przekątną 51 mm, rozdzielczość 320×240 pikseli przy 24-bitowej głębi koloru. Wyświetlacz ten jest stosowany również w innych telefonach Nokii np.: 6500c, 6500, 6300/6301. 6120, 6555, 5310, 7310, 8600, 3120. Do podświetlania matrycy służą cztery białe diody LED. Dzięki ich połączeniu szeregowemu uzyskuje się tę samą jasność ich świecenia. W mojej aplikacji są one zasilane napięciem 12 V. Na **rysunku 1** pokazano rozmieszczenie wyprowadzeń wyświetlacza.

### Opis interfejsu wyświetlacza

Sterownikiem opisywanego wyświetlacza jest Leadis LDS285. Pomimo wsparcia dla wielu interfejsów, w LCD od Nokii E51 sterownik jest na stałe ustawiony w trybie komunikacji za pomocą interfejsu równoległego w standardzie Intel 8080 o magistrali 8-bitowej. Do poprawnej komunikacji wymagane są sygnały: DC, WRB, RDB.



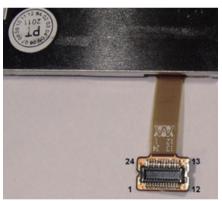
Wejście DC informuje sterownik wyświetlacza czy wysłano komendę, czy dane. Wyzerowanie wejścia DC oznacza komendę, natomiast ustawienie – dane. Wejście WRB służy do zatrzaskiwania danych D0... D7 – aktywne jest zbocze narastające. Aby wysłać do wyświetlacza komendę lub dane, najpierw należy wyzerować linię WRB, przesłać bajt na magistralę, odpowiednio ustawić stan wejścia DC, a następnie zmienić poziom WRB z niskiego na wysoki. Wejście RDB zezwala na odczyt danych z układu. Istotną rolę pełni również wejście CSB. Służy ono do aktywowania układu. Aktywny jest poziom niski.

Szczegóły nt. funkcjonowania można znaleźć w dokumentacji sterownika LDS285. Bez trudu można ją znaleźć w Internecie.

Wyświetlacz jest zasilany napięciami 1,8 V i 2,5 V. W aplikacji uzyskuje się je dzięki zastosowaniu dwóch stabilizatorów: TC1185-1.8VCT i TC1185-2.5VCT. Napięcie 1,8 V zasila układy cyfrowe, zatem przy sterowaniu za pomocą mikrokontrolera zasilanego wyższym napięciem, należy wykonać odpowiednie układy dopasowujące poziomy napięć. W naszym wypadku: 3,3 V <-> 1,8 V. Tę rolę pełnią dwa układy typu 74LVC4245AD.

### Sterowanie wyświetlaczem

Obsługę wyświetlacza można wykonać na dwa sposoby. W pierwszym trzeba samodzielnie generować sygnały interfejsu Intel 8080 (emulować go) wykorzystując



Nr wyprowadzenia	Funkcja
1	VLED1-
2	VLED2-
3	VIO
3 4 5 6	GND
5	WRB
	D0
7	GND
8	D2
9	D4
10	D6
11	CSB
12	RST
13	TE
14	D7
15	D5
16	GND
17	D3
18	D1
19	DC
20	RDB
21	GND
22	VAUX
23	VLED2+
24	VLED1+

Rysunek 1. Rozmieszczenie wyprowadzeń wyświetlacza od telefonu Nokia E51

porty GPIO. Użyjemy do tego mikrokontroler STM32F103C6T6 (CORTEX M3). Jest to stosunkowo wolne sterowanie, ponieważ maksymalna czestotliwość odświeżania, która udało się osiągnąć podczas testów to 1,5

klatki na sekundę, przy czym "klatka" oznacza wypełnienie całej powierzchni wyświetlacza jednakowym kolorem. Adekwatny kod nie powinien sprawiać trudności przy przenoszeniu go na inną platformę sprzęto-

wą. Jako ciekawostkę dodam, że wypełnienie całego ekranu jednym kolorem można przyspieszyć dzieki sterowaniu pinu WRB nie z portu GPIO, a z svgnałów zegarowych np. z szybkich peryferii bądź generatorów sygnałów.

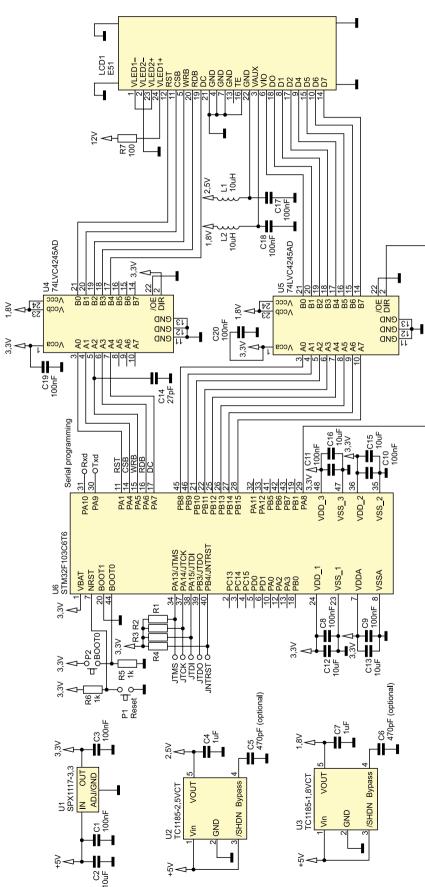
> W drugiej metodzie LCD jest sterowany przez układ FSMC dostępny w mikrokontrolerze STM32F103VCT6. Dzięki temu udaje się uzyskać częstotliwość odświeżania 12,5 klatki na sekunde. FSMC (Flexible Static Memory Controller) jest układem peryferyjnym, niedostępnym w tańszych wersjach mikrokontrolerów STM32. Zwykle służy on do obsługi pamięci: SRAM, ROM, NOR, PSRAM, NAND, 16-bit PC Card. Jednak jego możliwości są na tyle duże, że z powodzeniem można go użyć także do sterowania wyświetlaczem z interfejsem Intel 8080.

> Sterowanie LCD za pomocą emulowanie interfejsu Intel 8080. Schemat ideowy modelowego układu sterującego za pomocą emulowania interfejsu Intel 8080 pokazano na rysunku 2. Sterowanie wyświetlaczem odbywa się przez wysyłanie do niego komend (LCD write command) z parametrami (LCD write data). Nie wszystkie komendy wymagają parametrów. W tabeli 1 zamieszczono komendy sterownika LDS285. Ten jest ograniczony jedynie do objaśnienia instrukcji umożliwiających podstawowe operacje związane z wyświetlaniem obrazów. Więcej informacji należy szukać w dokumentacji sterownika.

> Funkcja LCD\_write\_parallel służy do przesłania liczby na magistralę 8-bitową. O sposobie jej interpretacji informują wcześniej wspomniane sygnały sterujące. Dana jest dostępna na wyprowadzeniach PB8... PB15 mikrokontrolera. Niezależnie od stosowanego mikrokontrolera nie polecam użvwania pinów należących do różnych portów lub podłączonych przypadkowo, ponieważ związane z tym dodatkowe operacje zabiorą czas mikrokontrolera i tym samym będą miały istotny wpływ na szybkość pracy wyświetlacza.

> Funkcje set LCD i clear LCD są kontrolują stan wyprowadzeń połączonych z wejściami wyświetlacza: LCD\_RST, LCD\_CS, LCD WR, LCD RD, LCD DC. Funkcja *LCD* hard\_reset() realizuje zerowanie sprzętowe wyświetlacza LCD. Minimalny czas trwania poziomu niskiego linii RST powinien wynosić 10 µs. Procedura restartu kontrolera LCD trwa około 120 ms.

> LCD\_write\_command i LCD\_write\_data służą do wysłania komendy bądź danych. W ciele tych funkcji, na początku, jest uaktywniany wyświetlacz niskim poziomem linii CSB. Następnie, w zależności od rodzaju danych, jest sterowania linia DC: w wypadku komendy jest ona zerowana, dla danych jest ona ustawiana. Następnie jest ustawiana linia RDB i dane są zatrzaskiwane narastają-



Rysunek 2. Schemat ideowy sterownika wyświetlacza Nokii E51

cym zboczem sygnału WRB. Po przesłaniu danych, linia WRB jest zerowana. Funkcje zapisu komendy i danej pokazano na **listingu 1**.

Funkcja LCD init realizuje inicjalizuje sterownik wyświetlacza. Zamieszczono ją na listingu 2. Wykonuje ona zerowanie sprzętowe sterownika wyświetlacza (LCD\_ hard reset), przesyła komendę SWRESET wywołującą tzw. miękki restart sterownika. Podczas niego są przywracane wartości domyślne rejestrów, a wyświetlacz jest wyłaczany. "Miękkie" zerowanie trwa ok. 5 ms. Nastepnie komenda SLPOUT wyprowadza wyświetlacz z trybu sleep mode. Włączany jest konwerter DC/DC, uruchamiane są wewnętrzny oscylator wyświetlacza i panel skanujący. Wszystko to trwa kolejne 5 ms. W tym czasie sterownik wykonuje również diagnostykę. Ostatnim krokiem inicjalizacji jest wysłanie komendy DISPON. Włacza ona wyjście pamięci DDRAM, co skutkuje pojawieniem się różnych kolorowych pikseli na ekranie. Zdarza się, że niektóre egzemplarze LCD wyświetlają równe paski. Są to przypadkowe wartości pamięci ustalone po włączeniu LCD. Od tego momentu wyświetlacz jest gotowy do pracy.

Teraz dobrze jest wyczyścić obszar wyświetlania. Polega to na wypełnieniu pamięci obrazu stałą np. dla białego koloru. W przykładzie wykorzystano do tego celu zamieszczoną na listingu 3 funkcję LCD paint. Najlepiej ją wywołać zaraz po inicjalizacji, by "nieeleganckie", różnokolorowe piksele nie były widoczne. Argumentem funkcji jest liczba odpowiadająca kolorowi. Powinna ona zawierać 3 bajty, znaczenie poszczególnych bitów pokazano na rysunku 3.By móc wypełnić cały obszar wyświetlania jednolitym kolorem najpierw trzeba wysłać komendę RAMWR. Parametrem tej komendy są dane odpowiadające za kolor pikseli. Dzięki niej wysyłane dane aktualizują wewnętrzną pamięć DDRAM w sterowniku, czego skutkiem jest natychmiastowa zmiana koloru zapisywanych pikseli. Po wysłaniu tej komendy początkowa wartość liczników kolumn i wierszy pikseli jest przywracana do wartości początkowych. W tym wypadku, gdy inne rejestry nie zostały jeszcze zmienione, wypełnianie matrycy następuje od 0 wiersza i kolumny aż po 319 wiersz i 239 kolumnę, w kolejności jak na rysunku 4. Jeżeli wyślemy więcej potrójnych bajtów odpowiadających za jeden piksel, niż mamy dostępnych pikseli w wyświetlaczu, liczniki wierszy i kolumn przyjmą wartości początkowe (0 wiersz, 0 kolumna). Liczba bajtów danych jest równoważna iloczynowi rozdzielczości pionowej i poziomej LCD i głębi koloru (320×240×3). Jako pierwszy zawsze wysyłany jest kolor czerwony, potem zielony i na końcu niebieski.

Pokazana na **listingu 4** funkcja *LCD\_flash* została wykonana w celu zaprezentowania najszybszego, możliwego z użyciem danej

Instrukcja	Tabela 1. Wykaz	komend sterujących wyświetlacza LCD Nokii E51									
SWRESET RDDID NOV04 Read Display ID RODST NOV99 Read Display Power Mode RDDMADCTR RDDMADCTR RDDMADCTR RDDMADCTR RDDMADCTR RDDMADCTR RDDCOLMOD NOV05 READ Display Pixel Format RDDMADCTR RDDDM NOV06 READ Display Pixel Format RDDMADCTR RDDDM READ Display Pixel Format RDDMADCTR RDDDM READ Display Signal Mode RDDSDR READ Display Self-diagnostic result SLPIN SLPIN NOV10 SLEPIN NOV10 SLEPIN NOV11 Sleep out & booster on PTION NORON NOV13 Partial mode on NORON NOV21 Partial mode on NORON NOV21 Display inversion off (normal) INVOFF NOV20 Display inversion on GAMSET DISPON NOV29 DISPlay off DISPOP CASET NOV2B REASET NOV2B REASET NOV2B REASET NOV2B REASET RAMMUR NOV2C Memory write RAMMUR NOV2C Memory write RAMMUR NOV2C Memory write RAMMUR NOV2E PARTIAL Start/end address set RAMMUR NOV2E Memory read Partial start/end address set REAMMUR NOV2E Memory read Partial start/end address set REAMMUR NOV36 MEMORY read Partial start/end address set REAMMUR NOV37 READ READ READ READ READ READ READ READ	Instrukcja	Kod (notacja języka C)	Opis								
RDDID RDDSM RDDSM RDDFM	NOP	0x00	No Operation								
RDDST RDDST RDDPM RDDPM RDDPM RDDMADCTR RDDMADCTR RDDMADCTR RDDCOLMOD RODG RDDM READ READ Display Power Mode RDDMADCTR RDDCOLMOD RODG RDDM READ READ Display MADCTR READ Display MADCTR READ READ Display Self-diagnostic result SLPIN SLPIN READ Sleep in & booster off SLPOUT SLPOUT Ox11 Sleep out & booster on PTLON NORON ROX12 RPATIAI mode on Display inversion off (normal) INVOFF ROX20 Display inversion off (normal) INVOFF ROX20 Display inversion on GAMSET DX26 Gamma curve select Display of Display on CASET RASET ROX28 ROW address set RAMWR RAMRD RASET ROX28 RAMWR RAMRD RAM	SWRESET	0x01	Software reset								
RDDPM RDDMADCTR Dx0B RDDMADCTR RDDMADCTR RDDMODO RDDIM RDDM RDDSDM RDDSM READ RDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ READ READ READ READ READ READ RE	RDDID	0x04	Read Display ID								
RDDPM RDDMADCTR Dx0B RDDMADCTR RDDMADCTR RDDMODO RDDIM RDDM RDDSDM RDDSM READ RDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ RDSDSDR READ READ READ READ READ READ READ RE	RDDST	0x09	Read Display Status								
RDDMADCTR RDDCOLMOD RDDSM RDDCDSM RDDSM RDDSDR READ Displays Signal Mode RDDSDR RDDSDR READ Display Signal Mode RDDSDR RDDSDR READ Display Signal Mode RDDSDR RDDSDR READ Display Signal Mode RDDSDR READ Display in booster off RDDSDR RDDSD	RDDPM										
RDDCOLMOD RDDIM RDDSM RDDSM RDDSDR RONOF REad Display Signal Mode RDDSDR RDDSDR SLPIN SLPIN SLPIN SLPIN SLPIN RONON RORE READ Display Self-diagnostic result SLPIN SLPIN SLPIN SLPIN SLPIN RORE READ Display Self-diagnostic result SLPIN SLPIN SLPIN SLPIN SLPIN SLPIN SLPIN SLPIN RORE READ Display Self-diagnostic result SLPIN SLPIN SLPIN SLEEP IN & booster of Partial mode on NORON RORON	RDDMADCTR	0x0B									
RDDIM RDDSN  OXOE RDDSN  OXOF RDDSN  OXOF REad Display Signal Mode RDDSDR  OXOF SLPIN  SLPOUT  OX11  Sleep in & booster on  Partial mode on  NORON  OX12  Partial mode on  Partial ff (Normal)  INVOFF  OX20  Display inversion off (normal)  INVOFF  OX20  Display inversion on  GAMSET  OX26  GAMMAC  GAMSET  OX26  DISPON  OX29  DISPlay inversion on  OX21  DISPON  OX29  DISPlay on  CASET  OX28  RASET  OX28  RASET  OX28  RASET  OX28  RAMWR  RAMND  OX2C  Memory write  RAMND  OX2E  Memory write  RAMND  OX2E  Memory read  Partial start/end address set  RAMND  TEOFF  OX34  TEOFF  OX34  TEOFF  OX36  MADCTR  DISPON  OX35  Tearing effect line off  Tearing effect mode set & on  MADCTR  DIMOFF  OX38  Idle mode on  COLMOD  OX39  Idle mode on  COLMOD  OX3A  Interface pixel format  WRDISBV  OX51  WRCISBV  OX51  WRCISBV  OX52  RRAD DISPLAY  Write Display Brightness  RDCTRLD  OX54  READ  OX56  READ  OX56  READ  INTERCE pixel format  WRDISBV  OX51  WRCISBV  OX51  WRCISBV  OX52  READ  INTERCE pixel format  WRDISBV  OX53  ROCTRLD  OX54  READ  OX56  READ  INTERCE pixel format  WRDISBV  OX51  WRTIE DISPLAY  Write Control Display  ROCABC  OX56  RDCABC  OX56  READ  INVCTR  OXDA  READ  INVCTR  OXD											
RDDSM NOSE Read Display Signal Mode RDDSDR NOSE NOSE Read Display Signal Mode READ DISPLAY INTERPLAY SIGNAL MODE READ DISPLAY INTERPLAY SIGNAL MODE READ DISPLAY INTERPLAY SIGNAL MODE READ DISPLAY SIGNAL MODE SET SIGNAL MOD											
RDDSDR SLPIN SLPIN SLPIN SLPOUT SLPOUT SLPOUT OX11 Sleep in & booster off Sleep in & booster on Partial mode on NORON OX12 Partial off (Normal) INVOFF OX20 Display inversion off (normal) INVON OX21 Display inversion on GAMSET OX26 GAMSET OX26 GAMSET OX27 Display off CASET OX28 DISPOF DISPOR OX29 DISPLAY OX29 DISPLAY OX29 DISPLAY OX40 COlumn address set RASET RASET RAMWR OX2C RAMRD OX2E RAMRD OX2E RAMRD OX2E RAMRD OX36 Partial start/end address set RAMRD TEON OX35 Tearing effect line off Tearing effect mode set & on MADCTR DMOFF OX38 Idle mode off Idle mode off Idle mode off Idle mode off Interface pixel format WRDISBV OX51 WRCIRLD OX53 RDCTRLD OX53 RDCTRLD OX54 READ ROX55 RDCABC OX55 RDCABC OX56 READ OXDA READ ROXDA ROXDA READ			Read Display Signal Mode								
SLPIN SLPOUT 0x11 Sleep in & booster off SLPOUT 0x11 Sleep out & booster on PTILON 0x12 Partial mode on NoRON 0x13 Partial off (Normal) INVOFF 0x20 Display inversion off (normal) INVOFF 0x20 Display inversion off (normal) INVON 0x21 Display inversion on Ox21 Display inversion on Ox22 Display off DISPOFF 0x28 Display off DISPON 0x29 Display on Ox29 Display on Ox28 RASET 0x2A Column address set RASET 0x2A RASET 0x2B Row address set RAMWR 0x2C Memory write RAMRD 0x2E Memory read PTILAR 0x30 Partial start/end address set RAMRD 0x2E Memory read PTILAR 0x30 Partial start/end address set TEOFF 0x34 Tearing effect line off TEON 0x35 Tearing effect mode set & on MADCTR 0x36 Memory data access control IDMOFF 0x38 Idle mode off IDMON 0x39 Idle mode off IDMON 0x39 Idle mode on OCOLMOD 0x3A Interface pixel format WRDISBV 0x51 Write Display Brightness RDDISBV 0x52 Read Display Brightness value WRCTRLD 0x53 Write Control Display WRCTRLD 0x53 Write Control Display RDCTRLD 0x54 Read Control Display RDCTRLD 0x56 Read Control Display RDCTRLD 0x56 Read Control Display RDCTRLD 0x56 Read ID1 RDDD 0xDB Read ID2 RDDD 0xDB Read ID1 RDDD 0xDB Read ID2 RDDD 0xDB Read ID1 RDDD 0xDB Read ID2 RDDD 0xDB Read ID3 RDCTR 0xC0 Regulator control REGCTR	RDDSDR	0x0F									
SLPOUT DX11 Sleep out & booster on Partial mode on NORON 0x12 Partial mode on NORON 0x13 Partial off (Normal) INVOFF 0x20 Display inversion off (normal) INVON 0x21 Display inversion on GAMSET 0x26 Gamma curve select DISPOFF 0x28 Display off DISPOFF 0x28 Display off DISPON 0x29 Display on CASET 0x2A Column address set RASET 0x2B Row address set RAMWR 0x2C Memory write RAMRD 0x2E Memory read PTLAR 0x30 Partial start/end address set TEOFF 0x34 Tearing effect line off TEON 0x35 Tearing effect mode set & on Memory data access control IDMOFF 0x38 Idle mode off IDMON 0x39 Idle mode on COLMOD 0x3A Interface pixel format WRDISBV 0x51 Write Display Brightness RDDISBV 0x52 Read Display Brightness value WRCABC 0x55 Write Control Display RVCABC 0x55 Write Content Adaptive Brightness RDIDI 0xDA Read ID1 RDDID 0xDA Read ID2 RDID 0xDA Read ID3 IFMODE 0xBB Read Control Display RFGCTR 0xC0 Regulator control REGET 0xC0 Set gamma correction characteristics GAMCTR 0xC1 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics FPPROM DxD0 REPROM Feach, verify register set RDVCOF 0xD9 VCOM offset bits read	SLPIN	0x10									
PTLON NORON NORON NORON NORON NORON NORON NORON NORON NORON NOX1 NIVOOF NOX20 Display inversion off (normal) Display inversion on GAMSET NOX26 DISPOFF NOX28 DISPON NOX29 DISplay off DISplay off DISPON NOX29 DISPlay off NOX28 RASET NOX2B RASET NOX2B RAMWR NOX2C RAMED NOX2C RAMRD NOX2E RAMRD NOX2E RAMRD NOX2E RAMRD NOX30 Partial start/end address set RAMWR NOX30 Partial start/end address set RAMRD NOX35 Tearing effect line off TEOFF NOX34 TEOFF NOX36 MADCTR NOX36 MADCTR NOX36 MADCTR NOX36 MADCTR NOX38 Idle mode off DIMOOFF NOX38 Idle mode off DIMOOFF NOX38 Idle mode off DIMOOFF NOX38 Idle mode on NOX39 COLMOD NOX3A Interface pixel format WRDISBV NOX51 WRITE DISPlay Brightness RDDISBV NOX52 READ DISPlay Brightness value WRCTRLD NOX53 WRCTRLD NOX54 READ COLTEND NOX55 RDCABC NOX55 READ ROCTRLD NOX56 READ READ READ READ READ READ READ READ		0x11									
NORON INVOFF INVON											
INVOFF INVON		0x13									
INVON   Ox21   Display inversion on   Gamma curve select   DISPOFF   Ox28   Display off   Display off   Display off   Display off   Display on   Ox29   Display on   Ox29   Display on   Ox28   Ox28   RASET   Ox2A   Column address set   RASET   Ox2B   Row address set   RASET   Ox2E   Memory write   RAMRD   Ox2E   Memory write   Memory write   Ox30   Partial start/end address set   TEOFF   Ox34   Tearing effect line off   TEON   Ox35   Tearing effect mode set & on   Memory data access control   IDMOFF   Ox36   Idle mode off   Idle mode off   Idle mode off   Idle mode on   Ox39   Idle mode on   Ox39   Idle mode on   Ox39   Idle mode on   Ox39   Write Display Brightness   Read Display Brightness   Read Display Brightness   Read Display Brightness   Read Control Display   ROTRLD   Ox52   Read Display Brightness   Read Control Display   ROTRLD   Ox54   Read Control Display   ROTRLD   Ox55   Write Content Adaptive Brightness   RDDB   OxDA   Read ID1   Read ID3	INVOFF	0x20	, ,								
GAMSET DISPOFF DISPOR OX28 DISPORO OX29 CASET OX2A COlumn address set RASET OX2B RASET OX2B RASET OX2B RASET RAMWR OX2C RAMRD OX2E RAMRD OX2E RAMRD OX2E RAMRD OX2E RAMRD OX30 Partial start/end address set TEOFF OX34 TEOFF OX34 TEaring effect ine off TEON OX35 Tearing effect mode set & on MADCTR DIMOFF OX38 Idle mode off IDMOFF OX38 Idle mode off IDMON OX39 Idle mode off IDMON OX3A Interface pixel format WRDISBV OX51 WRCITRLD OX53 WRCITRLD OX53 WRCTRLD OX53 WRCTRLD OX54 WRCABC OX55 WTC Content Adaptive Brightness RDCABC OX56 Read ID1 RDID3 OXDA Read ID1 RDID3 OXDA Read ID3 IFMODE OXB0 DISCLK OXB1 DISPOR OXB2 REGCTR OXC0 REGULATOR OXC1 VCOMCTR OXC1 OXC8 Set gamma curve select Display off Display RDCTRLD OX54 REGCTR OXC0 Regulator control Regulator control REGCTR OXC0 Regulator control VCOMCTR OXC1 VCOMCTR OXC1 VCOMCTR OXC8 Set gamma correction characteristics GAMCTR3 OXCA Set gamma correction characteristics GAMCTR3 OXCA Set gamma correction characteristics FPPGMDB OXD0 EPPROG OXD2 EEPROM program E	INVON	0x21									
DISPON CASET Ox2A CASET Ox2A COlumn address set RAMWR Ox2C RAMRD Ox2E RAMRD Ox2E PTLAR PTLAR Ox30 Partial start/end address set TEOFF Ox34 TEON Ox35 Tearing effect line off TEON Ox36 IDMOFF Ox38 Idle mode off IDMON Ox39 Idle mode on COLMOD Ox3A Interface pixel format WRDISBV Ox51 WRCTRLD Ox53 WRCTRLD Ox54 WRCTRLD Ox54 WRCTRLD Ox54 READ Ox55 RDCABC Ox56 RDID1 OxDA RDID1 OxDA RDID2 OxDB RDDID3 RDDID3 OxDC READ IFMODE OxBO DISCLK OxBO DISCLX OXBO DI	GAMSET	0x26									
DISPON CASET Ox2A CASET Ox2A COlumn address set RAMWR Ox2C RAMRD Ox2E RAMRD Ox2E PTLAR PTLAR Ox30 Partial start/end address set TEOFF Ox34 TEON Ox35 Tearing effect line off TEON Ox36 IDMOFF Ox38 Idle mode off IDMON Ox39 Idle mode on COLMOD Ox54 WRDISBV Ox51 WRISBV Ox51 WROTRLD Ox54 WRCTRLD Ox54 WRCTRLD Ox54 REAC Ox55 WRCTRLD Ox54 REAC Ox56 RDID1 Ox0A READ RDID1 Ox0A RDID1 Ox0A RDID2 RDID3 Ox0C READ REAC REAC RDID3 Ox0C READ REAC REAC REAC RDID3 Ox0C REAC REAC REAC REAC REAC REAC REAC REA	DISPOFF	0x28	Display off								
RASET RAMWR RAMRD Dx2C RAMRD Dx2E RAMRD Dx30 RAMBD Dx30 RAMBD Dx34 RAMBD RA	DISPON	0x29	Display on								
RAMWR RAMRD DV2E RAMRD DV2E Memory read PTLAR DV30 Partial start/end address set TEOFF DV34 TEON DV35 Tearing effect line off Tearing effect mode set & on MADCTR DW0FF DV38 Idle mode off IbMON DV39 Idle mode on COLMOD DV3A Interface pixel format WRDISBV DV51 WRCTRLD DV52 WRCTRLD DV54 WRCTRLD DV54 WRCABC DV55 Write Control Display WRCABC RDJD1 DV56 RDD1D1 DV56 RDD1D1 DV50 RDD1D2 DV50 RDD1D3 DV50 READ DISPLA RDD1D2 DV50 RDD1D3 DV50 READ DISPLA READ	CASET	0x2A	Column address set								
RAMRD PTLAR DX30 Partial start/end address set TEOFF DX34 TEOFF DX34 TEON DX35 TEaring effect line off TEON DX36 MADCTR DX36 MEmory data access control IDMOFF DX38 IDMON DX39 Idle mode off IDMON COLMOD DX3A WRDISBV DX51 Write Display Brightness WRDISBV WRCTRLD DX53 Write Control Display RDCTRLD DX54 WRCTRLD DX54 WRCTRLD DX55 RDCABC RDLD1 RDD1 RDD1 RDD1 RDD2 DXDA RDD1 RDD2 RDD3 RDD1 RDD2 RDD3 IFMODE DXB0 DISCLK DXB1 DXB2 DXB1 REGCTR DXB2 DXB1 REGCTR DXB2 DXB2 REGCTR DXB2 DXB1 REGCTR DXB2 DISPLAY REGCTR DXB3 DISPLAY REGCTR DXB4 REGCTR DXB6 DISCLK DXB1 DISPLAY DISPLAY REGCTR DXB7 REGCTR DXB0 Set display interface mode DISCLK DXB1 DISPLAY DISPLAY DISPLAY DISPLAY DISPLAY DISPLAY COMCTR DXC REGCTR DXC0 REGULTO COMCTR DXC1 VCOMCTR DXC1 VCOMCTR DXC1 VCOMCTR DXC1 SET gamma correction characteristics Set gamma correction ch	RASET	0x2B	Row address set								
PTLAR TEOFF Dx34 TEOFF Dx34 TEON Dx35 Tearing effect line off Tearing effect mode set & on MADCTR Dx36 IDMOFF Dx38 Idle mode off IDMON Dx39 Idle mode off IDMON Dx39 Idle mode on Interface pixel format WRDISBV Dx51 Write Display Brightness WRCTRLD Dx54 WRCTRLD Dx54 WRCABC Dx55 Write Control Display WRCABC Dx55 Write Content Adaptive Brightness RDDID1 RDD2 RDD3 RDD3 RDD4 Read ID1 RDD2 RDD3 RDD3 RDD4 READ IFMODE Dx80 Dx80 Dx80 Dx80 Dx80 Dx81 Dx80 Dx82 Dx81 Dx82 Dx82 Dx83 Dx82 Dx83 Dx84 Dx85 Dx86 Dx85 READ Dx86 Dx86 Dx86 Dx86 Dx86 Dx86 Dx86 Dx8	RAMWR	0x2C	Memory write								
TEOFF TEON 0x35 Tearing effect line off TEON 0x35 MADCTR 10x36 MADCTR 10x36 MADCTR 10x38 Idle mode off IDMON 0x39 Idle mode off Idle mode on COLMOD 0x3A Interface pixel format WRDISBV 0x51 Write Display Brightness RDDISBV 0x52 Read Display Brightness value WRCTRLD 0x53 Write Control Display WRCABC 0x55 Write Control Display WRCABC 0x55 Write Content Adaptive Brightness RDCABC RDDI1 0xDA Read ID1 RDD1 0xDA Read ID2 RDD3 RDD3 IFMODE 0xB0 DISCLK 0xB1 DISPLX 0xB2 Display interface mode DISCLK 0xB1 Display clock set Display inversion control REGCTR 0xC0 Regulator control VCOMCTR 0xC1 VCOMCTR 0xC8 Set gamma correction characteristics GAMCTR1 0xCB Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR4 0xCB EPPGMDB 0xD0 EPROG EPROVE EPROM erase EPPROM erase EPPROM program EEPROM program E	RAMRD	0x2E	Memory read								
TEON 0x35 Tearing effect mode set & on MADCTR 0x36 Memory data access control IDMOFF 0x38 Idle mode off IDMON 0x39 Idle mode on COLMOD 0x3A Interface pixel format WRDISBV 0x51 Write Display Brightness RDISBV 0x52 Read Display Brightness value WRCTRLD 0x53 Write Control Display WRCTRLD 0x54 Read Control Display WRCABC 0x55 Write Content Adaptive Brightness RDCABC 0x56 Read Content Adaptive Brightness RDCABC 0x56 Read ID1 RDID2 0xDA Read ID2 READ 0xDA Read ID3 IFMODE 0xBO Set display interface mode DISCLK 0xB1 Display inversion control REGCTR 0xC0 Regulator control REGCTR 0xC1 VCOML/VCOMH voltage control VCOMCTR 0xC1 VCOML/VCOMH voltage control Set gamma correction characteristics GAMCTR1 0xC8 Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics Set gamma correction cha	PTLAR	0x30	Partial start/end address set								
MADCTR IDMOFF IDMON IDMO	TEOFF	0x34	Tearing effect line off								
IDMOFF IDMON OX39 Idle mode off IDMON OX39 Idle mode on COLMOD OX3A Interface pixel format WRDISBV OX51 Write Display Brightness RDDISBV WRCTRLD OX53 Write Control Display WRCABC OX55 Write Content Adaptive Brightness RDCABC OX56 READ ID1 OXDA READ ID2 RDID1 OXDA READ ID2 RDID3 OXDC READ ID3 IFMODE DISCLK INVCTR OXB1 INVCTR OXB2 DISplay interface mode Display IFMODE DISCLK OXB1 INVCTR OXB2 DISplay interface mode Display INVCTR OXB2 DISplay inversion control REGCTR VCOMCTR VCOMCTR OXC1 VCOML/VCOMH voltage control GAMCTR1 OXC8 GE gamma correction characteristics GAMCTR2 OXC9 Set gamma correction characteristics GAMCTR3 OXCA Set gamma correction characteristics GAMCTR4 OXCB SET gamma correction characteristics Set gamma correction characteristics FPPGMDB OXDO Write ID2,VCOM Offset for EEPROM program EPPRASE EPPROM Program EEPROM rease EEPROM rease EEPROM read, verify register set RDVCOF RDIDS READ INVCOM offset bits read	TEON	0x35									
IDMON COLMOD OX3A Interface pixel format WRDISBV OX51 Write Display Brightness RDDISBV WRCTRLD OX53 Write Control Display WRCTRLD OX54 WREABC WRCABC WRCABC OX55 Write Content Adaptive Brightness RDCABC RDID1 OXDA Read ID1 RDID2 RDID3 OXDC Read ID2 RDID3 OXDC Read ID3 IFMODE DISCLK OXB1 DISCLK OXB1 DISPLAY DISPLAY DISPLAY DISPLAY DISPLAY DISPLAY REGCTR OXCO REGULATOR VCOMCTR VCOMCTR OXC1 VCOML/VCOMH voltage control GAMCTR1 GAMCTR1 OXC8 GAMCTR2 GAMCTR3 OXCA Set gamma correction characteristics GAMCTR4 OXCB SET gamma correction characteristics SET	MADCTR	0x36									
COLMOD 0x3A Interface pixel format WRDISBV 0x51 Write Display Brightness Read Display Brightness value WRCTRLD 0x53 Write Control Display RDCTRLD 0x54 Read Control Display WRCABC 0x55 Wrtie Content Adaptive Brightness RDCABC 0x56 Read ID1 RDID1 0xDA Read ID1 RDID2 0xDB Read ID2 RDID3 0xDC Read ID3 IFMODE 0xB0 Set display interface mode DISCLK 0xB1 Display clock set INVCTR 0xB2 Display clock set INVCTR 0xB2 Display inversion control REGCTR 0xC0 Regulator control VCOMCTR 0xC1 VCOML/VCOMH voltage control GAMCTR1 0xC8 Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read											
WRDISBV 0x51 Write Display Brightness value WRCTRLD 0x53 Write Control Display RDCTRLD 0x54 Read Control Display WRCABC 0x55 Write Content Adaptive Brightness RDCABC 0x56 Read Content Adaptive Brightness RDCABC 0xDA Read ID1 RDID1 0xDA Read ID2 RDID2 0xDB Read ID2 RDID3 0xDC Read ID3 IFMODE 0xB0 Set display interface mode DISCLK 0xB1 Display clock set INVCTR 0xB2 Display inversion control REGCTR 0xC0 Regulator control VCOMCTR 0xC1 VCOML/VCOMH voltage control GAMCTR1 0xC8 Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM program EPPROG 0xD2 EEPROM program EPPROF 0xD3 EEPROM read, verify register set RDVCOF 0xD9											
RDDISBV 0x52 Read Display Brightness value WRCTRLD 0x53 Write Control Display RDCTRLD 0x54 Read Control Display WRCABC 0x55 Wrtie Content Adaptive Brightness RDCABC 0x56 Read Content Adaptive Brightness RDID1 0xDA Read ID1 RDID2 0xDB Read ID2 RDID3 0xDC Read ID3 IFMODE 0xB0 Set display interface mode DISCLK 0xB1 Display clock set INVCTR 0xB2 Display inversion control REGCTR 0xC0 Regulator control VCOMCTR 0xC1 VCOML/VCOMH voltage control GAMCTR1 0xC8 Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM read, verify register set RDVCOF 0xD9	COLMOD										
WRCTRLD  RDCTRLD  WRCABC  WRCABC  WRCABC  Ox55  Wrtie Content Adaptive Brightness  RDCABC  RDCABC  RDD1  RDD1  RDD2  RDD3  READ  IFMODE  DISCLK  INVCTR  Ox80  DISCLK  INVCTR  Ox80  DISCLK  INVCTR  Ox80  DISPlay  INVCTR  OxC0  REGULATOR OxC1  VCOMCTR  GAMCTR1  OxC8  GAMCTR2  OxC9  GAMCTR3  OXCA  GAMCTR4  GAMCTR4  EPPGMDB  DXDC  READ  OXDC  READ  Write Control Display  Read Content Adaptive Brightness  Read ID1  Read ID2  Read ID3  Read ID3  IFMODE  OxB0  Set display interface mode  Display clock set  INVCTR  OxC0  Regulator control  VCOML/VCOMH voltage control  VCOML/VCOMH voltage control  Set gamma correction characteristics  GAMCTR2  OxC9  Set gamma correction characteristics  GAMCTR3  OxCA  Set gamma correction characteristics  GAMCTR4  OxCB  Set gamma correction characteristics  Write ID2,VCOM Offset for EEPROM program  EPPROG  EPPROG  OxD1  EEPROM program  EEPROM read, verify register set  RDVCOF  OxD9  VCOM offset bits read											
RDCTRLD WRCABC WRCABC WRCABC WRCABC RDCABC RDCABC RDID1 RDD2 OxDB REad ID2 RDID3 RDID3 IFMODE DISCLK INVCTR CABC REGCTR VCOMCTR GAMCTR1 OxC8 GAMCTR2 GAMCTR3 OXCA GAMCTR4 GEARC GAMCTR4 EPPGMDB CABC CABC CABC CABC REAG COntent Adaptive Brightness Read Content Adaptive Brightness Read ID1 Read ID2 Read ID2 Read ID3 Set display interface mode Display clock set INVCTR OxB1 Display clock set INVCTR OxB2 Display inversion control Regulator control VCOML/VCOMH voltage control VCOML/VCOMH voltage control Set gamma correction characteristics GAMCTR3 OxCA Set gamma correction characteristics GAMCTR4 OxCB Set gamma correction characteristics EPPGMDB OxD0 Write ID2,VCOM Offset for EEPROM program EPPRASE EPPROG OxD1 EEPROM program EEPROM rease EPPROM read, verify register set VCOM offset bits read											
WRCABC RDCABC RDCABC RDCABC RDID1 OxDA RDID2 OxDB RDID3 RDID3 OxDC REad ID3 RFMODE DISCLK INVCTR COMBCTR REGCTR VCOMCTR GAMCTR1 GAMCTR1 GAMCTR2 GAMCTR3 OxCA GAMCTR3 GAMCTR4 EPPGMDB DXCA EPPROG EPPROVR EPPROVR RDCADC READ READ READ READ READ READ READ READ											
RDCABC RDID1 RDID2 RDID2 RDID3 RDID3 RDID3 RDXDC REad ID2 RDID3 REad ID3 REGCTR REGCTR REGCTR REGCTR REGCTR ROXCO REGULATOR CONTROL REGAMCTR1 READ ID3 READ ID3 READ ID3 REGCTR REGCTR ROXCO REGULATOR CONTROL REGAMCTR1 READ ID3 RE											
RDID1  OxDA  Read ID1  Read ID2  RDID3  OxDC  Read ID3  Read ID2  RDID3  OxDC  Read ID3  Read ID3  READ ID5  OxBO  Set display interface mode  DISCLK  OxB1  Display clock set  INVCTR  OxB2  Display inversion control  REGCTR  OxCO  Regulator control  VCOMCTR  OxC1  VCOML/VCOMH voltage control  Set gamma correction characteristics  GAMCTR1  OxC8  Set gamma correction characteristics  GAMCTR2  OxC9  Set gamma correction characteristics  GAMCTR4  OxCB  Set gamma correction characteristics  GAMCTR4  OxCB  Set gamma correction characteristics  Set gamma correction characteristics  GAMCTR4  OxCB  Set gamma correction characteristics  Set g											
RDID2 RDID3 RDID3 RDID3 RDID3 RDID3 READ READ READ READ READ READ READ READ											
RDID3 IFMODE OxBO OxBO DISCLK INVCTR OxB1 Display clock set INVCTR REGCTR OxCO VCOMCTR OxC1 GAMCTR1 OxC8 GAMCTR2 GAMCTR3 OxCA GAMCTR4 EPPGMDB OxCB EPPROG EPROVRF OxD1 EPROVRF RDID3 OxBO OxBO OxBO OxBO OxBO OxBO OxBO OxB											
IFMODE DISCLK DISCLK UNVCTR OXB2 Display clock set Display clock set Display inversion control REGCTR VCOMCTR VCOMCTR OXC1 GAMCTR1 OXC8 GAMCTR2 GAMCTR3 OXCA GAMCTR3 OXCA GAMCTR4 CACA GAMC		*									
DISCLK INVCTR OxB2 Display clock set Display clock set Display inversion control REGCTR VCOMCTR VCOMCTR OxC1 VCOML/VCOMH voltage control VCOMCTR OxC8 Set gamma correction characteristics GAMCTR2 OxC9 Set gamma correction characteristics GAMCTR3 OxCA Set gamma correction characteristics GAMCTR4 OxCB Set gamma correction characteristics Set gamma correction characteristics With ID2, VCOM Offset for EEPROM program EPERASE EPPROG OxD1 EEPROM erase EPPROM program EPPROG EPRDVRF OxD3 EEPROM read, verify register set RDVCOF OxD9 VCOM offset bits read											
INVCTR REGCTR OxCO REGULATOR VCOMCTR OxC1 VCOML/VCOMH voltage control VCOML/VCOMH voltage control VCOMCTR GAMCTR1 OxC8 GAMCTR2 OxC9 Set gamma correction characteristics GAMCTR3 OxCA Set gamma correction characteristics GAMCTR4 OxCB Set gamma correction characteristics GAMCTR4 OxCB Set gamma correction characteristics GAMCTR4 OxCB Set gamma correction characteristics Set gamma correction characteristics Write ID2,VCOM Offset for EEPROM program EPERASE EPPROG OxD1 EEPROM erase EPPROM program EPPROG EPRDVRF OxD3 EEPROM read, verify register set RDVCOF OxD9 VCOM offset bits read											
REGCTR 0xC0 Regulator control VCOMCTR 0xC1 VCOML/VCOMH voltage control GAMCTR1 0xC8 Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPPROG 0xD2 EEPROM program EPPROF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read											
VCOMCTR GAMCTR1 OxC8 GAMCTR2 OxC9 GAMCTR3 OxCA GAMCTR3 OxCB GAMCTR4 OxDO GAMCTR4 OxCB GAMCTR4 Ox											
GAMCTR1 0xC8 Set gamma correction characteristics GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics GAMCTR4 0xD0 Write ID2,VCOM Offset for EEPROM program EPPRASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPPROF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read			3								
GAMCTR2 0xC9 Set gamma correction characteristics GAMCTR3 0xCA Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPRDVRF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read		1									
GAMCTR3 0xCA Set gamma correction characteristics GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPRDVRF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read											
GAMCTR4 0xCB Set gamma correction characteristics EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPRDVRF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read											
EPPGMDB 0xD0 Write ID2,VCOM Offset for EEPROM program EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPRDVRF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read		1									
EPERASE 0xD1 EEPROM erase EPPROG 0xD2 EEPROM program EPRDVRF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read											
EPPROG 0xD2 EEPROM program EPRDVRF 0xD3 EEPROM read, verify register set RDVCOF 0xD9 VCOM offset bits read			, ,								
EPRDVRF 0xD3 EEPROM read, verify register set VCOM offset bits read											
RDVCOF 0xD9 VCOM offset bits read											
TELDCINE OVER AMILE TED COULIN AGING	LEDCTRL	0xEF	Write LED control value								

metody, odświeżania ekranu wyświetlacza, na przemian wypełniając całą matrycę trzema kolorami. Jej działanie jest tak same jak funkcji *LCD\_paint*. Jak widać, wystarczy tylko raz wysłać komendę RAMWR.

Funkcja put\_pixel (listing 5) ilustruje zasadę wypełniania części matrycy. Jej parametrami wejściowymi są współrzędne i kolor piksela, który chcemy umieścić. Komendy CASET i RASET pozwalają ustalić

```
Listing 1. Funkcje zapisu komendy i danych do wyświetlacza
void LCD write
                command (unsigned char
                                          cmd) +
  clear_LCD(LCD_CS);
                                       //choose LCD
  clear LCD (LCD DC);
                                        /write command
                                       //page 19 in datasheet
  set LCD(LCD RD);
  clear_LCD(LCD_WR);
                                         low on WRB pin
                                        /set 8bit parallel data
  LCD_write_parallel(cmd);
set LCD(LCD WR);
                                       //high on WRB pin
void LCD write data(unsigned char cmd) {
  clear_\bar_\bar_(LC\bar_CS);
                                        //choose LCD
  set LCD(LCD DC);
                                       //write data
                                         /page 19 in datasheet
  set LCD (LCD RD);
                                       //low on WRB pin
  clear LCD(LCD WR);
  LCD_write_parallel(cmd);
set_LCD(LCD_WR);
                                        //set 8bit parallel data
                                       //high on WRB pin
```

# Listing 2. Funkcja inicjalizująca sterownik wyświetlacza void LCD\_init() { //delay\_ms(250); //optional - wait for stabilization lcd //power LCD\_hard\_reset(); LCD\_write command(SWRESET); delay\_ms(\overline{5}); LCD\_write\_command(SLPOUT); delay\_ms(\overline{5}); LCD\_write\_command(DISPON); //delay 5ms //delay 5ms //delay 5ms //Display On }

## Listing 3. Funkcja wypełniająca obszar wyświetlania kolorem void LCD\_paint(unsigned int color) { unsigned int i; LCD\_write\_command(RAMWR); for(i=0; i<320\*240; i++) { LCD\_write\_data((unsigned char)color); LCD\_write\_data((unsigned char) (color >> 8)); LCD\_write\_data((unsigned char) (color >> 16)); } }

23							16	15							8	7					0
NIEBIESKI ZIELONY							C	ZER\	NON	Υ											

Rysunek 3. Rozmieszczenie bajtów kolorów w słowie-argumencie funkcji LCD\_paint

początkowe i końcowe wartość kolumn oraz wierszy. Obie komendy wymagają podania 4 parametrów. Jeżeli wartości tych rejestrów nie będą zmieniane, za każdym razem po wysłaniu komendy RAMWR, cała matryca będzie wypełniana. Komenda CASET ustala numer kolumny początkowej i końcowej, jak pokazano na **rysunku 4**. Argumentami wywołania są dwa bajty wartości początkowej i dwa bajty wartości końcowej. Jako

```
Listing 5. Wyświetlanie pojedynczego piksela
void put pixel (unsigned int x, unsigned int y, unsigned int color) {
   LCD_write_command(CASET);
   LCD_write_data((unsigned char) ((x>>8) & 0x000000FF));
   LCD_write_data((unsigned char) (x & 0x000000FF));
   LCD_write_data((unsigned char) ((y>>8) & 0x000000FF));
   LCD_write_data((unsigned char) (y & 0x000000FF));
   LCD_write_data((unsigned char) (y & 0x000000FF));
   LCD_write_data((unsigned char) (y & 0x000000FF));
   LCD_write_data((unsigned char) (color >> 16));
   LCD_write_data((unsigned char) (color >> 8));
   LCD_write_data((unsigned char) (color >> 8));
   LCD_write_data((unsigned char) (color);
}
```

```
Listing 6. Inicjalizacja jednostki FSMC

void FSMC_NAND_Init(void)
{

GPIO_InitTypeDef GPIO_InitStructure;
FSMC_NANDInitTypeDef FSMC_NANDInitStructure;
FSMC_NAND FCCARDTimingInitTypeDef p;
RCC_APB2PeriphClockCmd(

    RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOB|
    RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOB|
    RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOD|
    RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOF|
    RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOF|
    RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOF|
    RCC_APB2Periph_GPIOC|RNAELE);

/* CLE, DO-D3, NOE, NWE - NAND pin configuration*/
GPIO_InitStructure.GPIO_Pin =
    GPIO_Pin 1|GPIO_Pin 15|
    GPIO_Pin 1|GPIO_Pin 15|
    GPIO_Pin 1|GPIO_Pin 1|GPIO_Pin 5;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed 50MHz;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
GPIO_InitGPIOD, &GPIO_InitStructure);

/* D4-D7 - NAND pin configuration */
GPIO_InitStructure.GPIO_Pin =
    GPIO_Pin 7|GPIO_Pin 8|GPIO_Pin 9|GPIO_Pin 10;
GPIO_InitGPIOD, &GPIO_InitStructure);

/* FSMC_COnfiguration */
p.FSMC_SetupTime = 0x1;
p.FSMC_SetupTime = 0x1;
p.FSMC_MandInitStructure.FSMC_Bank = FSMC_Bank2_NAND;
FSMC_NANDInitStructure.FSMC_MemoryDataWidth = FSMC_MemoryDataWidth_8b;
FSMC_NANDInitStructure.FSMC_MemoryDataWidth = FSMC_MemoryDataWidth_8b;
FSMC_NANDInitStructure.FSMC_ECC_FSMC_ECC_Disable;
FSMC_NANDInitStructure.FSMC_TLRSetupTime = 0x00;
FSMC_NANDInitStructure.FSMC_TRSetupTime = 0x00;
FSMC_NANDInitStructure.FSMC_TRSetupTime = 0x00;
FSMC_NANDInitStructure.FSMC_TRSetupTime = 0x00;
FSMC_NANDInitStructure.FSMC_TRSetupTime = 0x00;
FSMC_NANDInitStructure.FSMC_CommonSpaceTimingStruct = &p;
FSMC_NANDINIT(&FSMC_Bank2_NAND, ENABLE);
}
```

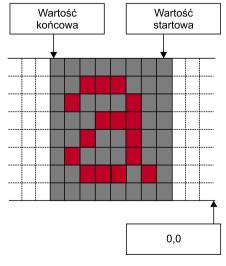
```
Listing 4. Funkcja powodująca
migotanie wyświetlacza
void LCD_flash() {
    unsigned int i;
    LCD_write_command(RAMWR);
    while (3) {
        for (i=0; i<320*240; i++) {
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
        }
        for (i=0; i<320*240; i++) {
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x00);
            LCD_write_data(0x07);
            LCD_write_data(0x07);
            LCD_write_data(0x07);
            LCD_write_data(0x07);
        }
    }
}
```

pierwsze podaje się starszy i młodszy bajt wartości początkowej, następnie starszy i młodszy bajt wartości końcowej. Komenda RASET umożliwia ustalenie startowej i końcowej wartości wierszy, podobnie jak CASET, jednak w kierunku pionowym. Jak z poprzednio, jako pierwsze argumenty podaje się starszy i młodszy bajt wartości początkowej, następnie starszy i młodszy bajt wartości końcowej.

W przypadku omawianej funkcji put\_pixel wartości początkowe i końcowe są takie same, odpowiednio dla kolumn i wierszy, ponieważ naszym celem jest zaświecenie pojedynczego piksela. Po ustaleniu współrzędnych piksela wysyłamy 3 bajty danych odpowiedzialnych za jego kolor, poprzedzając je komendą RAMWR. Jeżeli chcemy zmienić kolor większego fragmentu ekranu niż jeden piksel, należy podać nierówne sobie wartości początkowe i końcowe dla wierszy i kolumn.

### Sterowanie LCD za pomocą FSMC

Na **rysunku 5** pokazano schemat połączeń dla sterowania wyświetlaczem za pomoca FSMC. W tym wypadku, inaczej niż poprzednio, nie wolno zmienić połączeń



Rysunek 4. Zasada działania komendy CASET

pomiędzy mikrokontrolerem a LCD, ponieważ do obsługi wyświetlacza użyto FSMC, które dołącza się do układów zewnętrznych w ściśle określony sposób. Dzięki użyciu jednostki FSMC nie dosyć, że wzrasta prędkość obsługi wyświetlacza, to jeszcze zwalnia nas ona od konieczności programowej emulacji interfejsu Intel 8080. Funkcje: LCD\_write\_command, LCD\_write\_data i LCD\_write\_parallel nie sa już potrzebne.

Środowisko programistyczne, którego używam to Cortex-IAR Embedded Workbench for ARM. Wraz z nim producent dostarcza mnóstwo programów przykładowych. Wśród przykładów obsługi FSMC udostępniono procedury obsługi pamięci NOR, SRAM, SRAM i NAND. Dla nas najbardziej użytecznym jest przykład prezentujący sposób sterowania pamięcią NAND Flash. Na listingu 6 zamieszczono zmodyfikowaną procedurę inicjalizacji FSMC, która pierwotnie służyła do obsługi pamięci NAND Flash, a teraz można posłużyć się nią do obsługi wyświetlacza LCD z telefonu Nokia E51.

Na początku są włączane sygnały zegarowe, które będą występowały na wyprowadzeniach GPIO. Następnie są konfigurowane poszczególne piny mikrokontrolera. My użyjemy – zgodnie z nazewnictwem interfejsu pamięci NAND - tylko wyprowadzeń oznaczonych CLE (=LCD\_DC), NOE (=LCD RDB), NWE (=LCD WRB) i 8-bitowej szyny danych D0...D7. W dalszej kolejności są ustawiane parametry czasów dostępu do pamięci (bo tak będzie "widziany" wyświetlacz), aby zwiększyć szybkość komunikacji z wyświetlaczem LCD, który jest szybszy od pamięci Flash. Teraz są ustawiane parametry samego FSMC, tutaj najważniejszym jest 8-bitowa szerokość magistrali. Na koniec, następuje ustawienie rejestrów zgodnie z wcześniej wypełnioną strukturą i włączenie drugiego banku pamięci NAND.

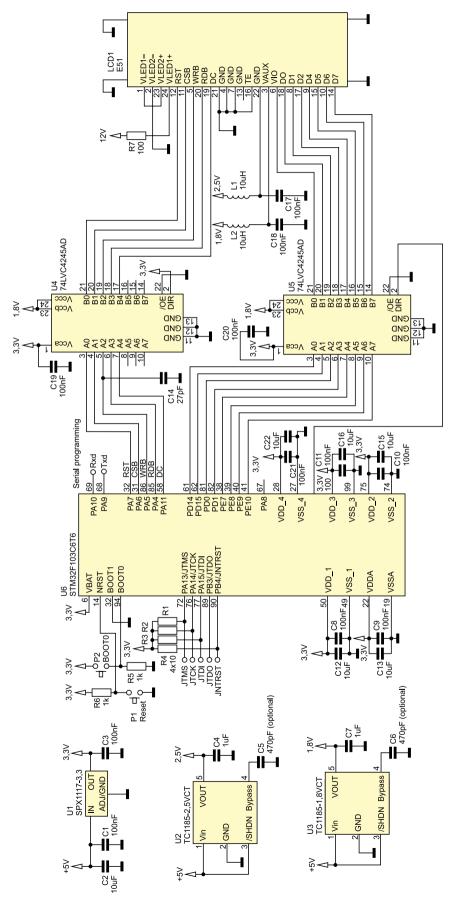
W nowym programie obsługi, który wykonano odpowiednio modyfikując prezentowany wcześniej, oprócz usunięcia wspomnianych funkcji, wprowadzono nowe definicje:

#define LCD\_WRITE\_COMMAND\*(vu8\*)
(Bank\_NAND\_ADDR|DATA\_AREA)
#define LCD\_WRITE\_DATA\*(vu8\*)
(Bank\_NAND\_ADDR|CMD\_AREA.

Zastępują one funkcje *LCD\_write\_com-mand*, *LCD\_write\_data*, które służyły do wysyłania komend lub danych. W wypadku wysłania komendy bajt będzie zapisany pod inny adres niż w wypadku wysłania danych. W ten nieskomplikowany sposób jednostka FSMC może odróżnić komendę od danej i odpowiednio sterować sygnałem CLE. Wyprowadzenie CLE jest połączony z pinem wyświetlacza DC. Jeżeli chcemy wysłać np. komende

LCD\_WRITE\_COMMAND = SWRESET;

FSMC zapisuje ją pod adresem danych zeruje wyprowadzenie CLE (*Command Latch Enable*). Dlaczego zeruje? Jednostkę FSMC skonfigurowałem w taki sposób, aby jak najbardziej uprościć program obsługi, jednak – jak pamiętamy – posłużyłem się przykładem sterownika pamięci NAND Flash. W wypadku jej obsługi zapis komendy następuje,



Rysunek 5. Schemat ideowy obsługi wyświetlacza Noki E51 przez FSMC

```
Listing 7. Funkcja wyświetlająca napisy
//przykład użycia: putc_enter (4,160 , czarny, "Dariusz Rzedowski");
#include "fonts.h"
                                                                                                                            //tablica ze znakami
void putc enter(unsigned char x, unsigned char y, unsigned int color, unsigned char *napis){
  int i,j, k=0, l=0, u=-1;

LCD WRITE COMMAND = MADCTR;

LCD_WRITE_DATA = ((unsigned char)(0x40));
   for (k=0; k<2000; k++) {
                                                                                                                            //maksymalna liczba liter
                                                                                                                            //na ekranie
     if(napis[k] == 0) break;
                                                                                                                            //jeżeli koniec tablicy
     u++; //pozycja x-owa
if((x+6*u) > 230){
                                                                                                                             //sprawdzenie czy juz
                                                                                                                            //trzeba "enter"
//"enter"
        v = v - 10:
        ū=0:
                                                                                                                              /zerowanie pozycji
                                                                                                                            //x-owych
       = (napis[k] - 32)*5;
     for (i=0; i<5; i++)
        for(j=0;j<8;j++)
          if(((fonts[]+i]>>i) \& 0x01) == 1) put pixel(x+i+6*u, y-i, color):
```

gdy sygnał CLE ma poziom wysoki. Zapis danych lub adresu do pamięci NAND jest wykonywany, gdy sygnał CLE ma poziom niski (pomijam poziomy innych wyprowadzeń interfejsu pamięci NAND Flash, ponieważ są one istotne). W wyświetlaczu, jeżeli wyprowadzenie DC połączone z wejściem CLE wyświetlacza ma poziom niski, to zapisywane są komendy. Stąd na pozór w definicjach błędne przypisanie nazwy LCD\_WRITE\_COMMAND pod adres DATA\_AREA.

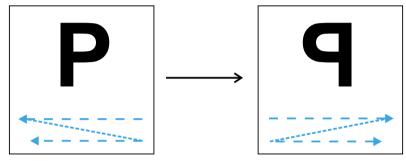
Jeżeli chcemy wysłać daną, to należy wykonać przypisanie

```
LCD WRITE DATA = ((u8)0x1A);
```

Wtedy liczba 0x1A zostanie zapisana pod adres komend w FSMC, co będzie oznaczało ustawienie sygnału CLE, ponieważ właśnie tak wysyła się komendę do pamięci NAND Flash. Jeżeli CLE (czyli pin DC) jest jedynką, w wypadku obsługi LCD oznacza to wysyłanie danych.

Podczas pracy nad uruchomieniem obsługi wyświetlacza za pomocą jednostki FSMC długo szukałem błędu powodującego wysyłanie dwóch bajtów zamiast jednego. Rozwiązaniem okazało się rzutowanie zmiennych typu unsigned char przy zapisie danych i komend. Jeżeli w konfiguracji FSMC jest ustawione słowo 8-bitowe a używane są 16-bitowe zmienne typu unsigned

REKLAMA



Rysunek 6. Działanie komendy MADCTR

short, to FSMC automatycznie wysyła dwa bajty. Jest to wygodne rozwiązanie przy pracy z 16-bitowymi słowami danych, ale nasze słowa są 8-bitowe.

### Funkcja wyświetlająca napisy

Funkcja widoczna na listingu 7 wyświetla napisy na ekranie LCD. Rysowanie znaków zrealizowane jest za pomocą wcześniej opisanej funkcji put\_pixel. Funkcja automatycznie przenosi wyrazy do kolejnego wiersza, gdy znaki je tworzące nie mieszą się. Pierwotnie funkcja była przeznaczona do obsługi innego wyświetlacza, w którym współrzędne zerowego piksela znajdowały się w lewym dolnym rogu, a nie jak w E51 – w prawym dolnym. Można uwzględnić to modyfikując funkcję, ale można też wysłać

komendę *MADCTR*, która służy do ustalenia kierunku skanowania wewnętrznej pamięci DDRAM wyświetlacza. W praktyce, zgodnie z podanym parametrem, powoduje to przerzucenie obrazu w pionie (**rysunek 6**).

#### **Podsumowanie**

W artykule zaprezentowano podstawy obsługi wyświetlacza z telefonu Nokia E51 będące bazą do wykonania programu wyświetlającego animacje, który będzie zaprezentowany w kolejnym numerze Elektroniki Praktycznej. Niestety, aby biegać trzeba nauczyć się najpierw chodzić, a to zajmuje troszkę czasu. Nie jest to jednak czas zmarnowany.

Dariusz Rzędowski dariusz.rzedowski@gmail.com

Profesjonalne narzędzia dla elektroników i programistów

uniwersalne programatory układów scalonych

- analizatory stanów logicznych

- oscyloskopy cyfrowe

- systemy do wyważania i pomiaru drgań

- oprogramowanie CAD, CAM, CAE

- emulatory, symulatory, debuggery dla różnych rodzin procesorów

- kompilatory C/C++ dla różnych rodzin procesorów

- szkolenia w zakresie FPGA, VHDL

- narzędzia na procesory sygnałowe DSP

- projektujemy, produkujemy, szkolimy, dystrybuujemy

- projektujemy, produkujemy, szkolimy, dystrybuujemy

- projektujemy, produkujemy, szkolimy, dystrybuujemy

ELEKTRONIKA PRAKTYCZNA 1/2012 75