

Tab. 1. Funkcje wyprowadzeń wyświetlacza 64128H-RGB firmy Displaytech

Nr	Nazwa	Funkcja
1	CS1B	Wejście wyboru układu
2	RST	Wejście zerowania kontrolera
3	RS	Wybór rejestru (1: dane wyświetlacza; 0: dane sterujące)
4	RW_WRB	Funkcja zależna od rodzaju interfejsu (patrz tab. 3)
5	E_RDB	Funkcja zależna od rodzaju interfejsu (patrz tab. 3)
6	DB0	Bit 0 magistrali danych
7	DB1	Bit 1 magistrali danych
8	DB2	Bit 2 magistrali danych
9	DB3	Bit 3 magistrali danych
10	DB4	Bit 4 magistrali danych
11	DB5	Bit 5 magistrali danych
12	DB6	Bit 6 magistrali danych
13	DB7	Bit 7 magistrali danych
14	VDD	Napięcie zasilania +3.3V
15	VCI	Wejście napięcia odniesienia dla konwertera
16	VSS	Masa
17	VOUT	Wejście/wyjście konwertera napięcia
18	C4+	Wyprowadzenie dodatnie kondensatora C4
19	C3+	Wyprowadzenie dodatnie kondensatora C3
20	C1-	Wyprowadzenie ujemne kondensatora C1
21	C1+	Wyprowadzenie dodatnie kondensatora C1
22	C2+	Wyprowadzenie dodatnie kondensatora C2
23	C2-	Wyprowadzenie ujemne kondensatora C2
24	V1	Napięcie zasilające matrycę LCD
25	V2	Napięcie zasilające matrycę LCD
26	V3	Napięcie zasilające matrycę LCD
27	V4	Napięcie zasilające matrycę LCD
28	V0	Napięcie zasilające matrycę LCD
29	C86	Wybór typu interfejsu (1: 6800; 0: 8080)
30	PS	Wybór typu interfejsu (1: równoległy; 0: szeregowy)

odwzorowany na 8 pionowych pikseli ekranu. Przedstawiono to **rys. 4**.

Układ S6B0724 umożliwia wysterowanie wyświetlacza LCD o organizacji maksymalnie 132x65 pikseli. W wyświetlaczu 12864H zastosowano matrycę LCD o organizacji 128x64 pikseli, przez co nie wszystkie komórki wewnętrznej pamięci RAM zostaną odwzorowane na ekranie wyświetlacza. Ponieważ cały obszar wyświetlacza jest sterowany przez jeden sterownik, jego obsługa jest prostsza niż wyświetlaczy z kilkoma kontrolerami (np. KS0108).

INSTRUKCJE STEROWNIKA S6B0724

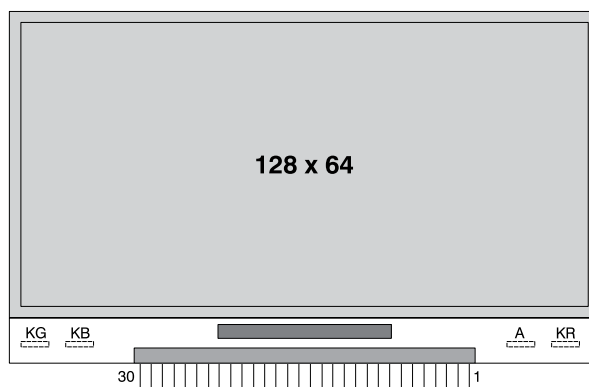
Niezależnie od wybranego rodzaju interfejsu do komunikacji z wyświetlaczem wykorzystywany jest zbiór kilku podstawowych instrukcji. Wyjątkiem od tej reguły jest brak możliwości odczytu danych ze

Tab. 2. Zależność pomiędzy stanami na tych wyprowadzeniach PS i C86 a typem interfejsu

PS	C86	Interfejs
0	0	Szeregowy
0	1	
1	0	8080
1	1	6800

Tab. 3. Funkcje wyprowadzeń E_RDB i RW_WRB w zależności od wybranego trybu pracy interfejsu

Wspólne	6800		8080		Funkcja
	E_RDB (E)	RW_WRB (RW)	E_RDB (/RD)	RW_WRB (/WR)	
RS					
1	1	1	0	1	Odczyt danych
1	1	0	1	0	Zapis danych
0	1	1	0	1	Odczyt rejestru statusowego
0	1	0	1	0	Zapis instrukcji



Rys. 1. Rozmieszczenie wyprowadzeń wyświetlacza 64128H-RGB firmy Displaytech

sterownika przy komunikacji za pomocą interfejsu szeregowego.

DISPLAY ON / OFF

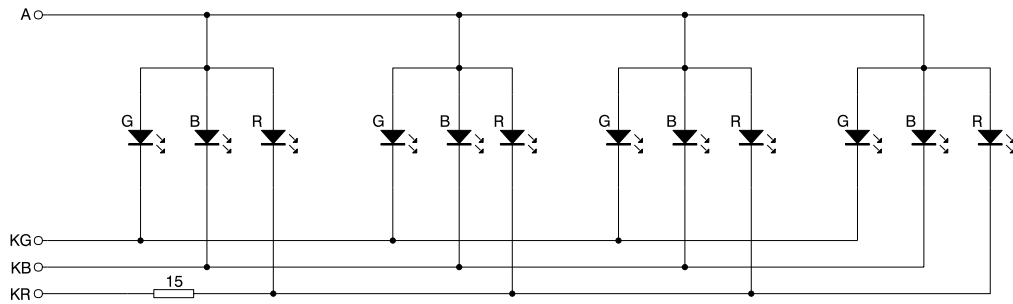
Instrukcja włączająca bądź wyłączająca wyświetlanie obrazu na ekranie wyświetlacza. Instrukcja nie ma wpływu na stan pamięci obrazu oraz rejestrów konfiguracyjnych.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	1	0	1	1	1	DON

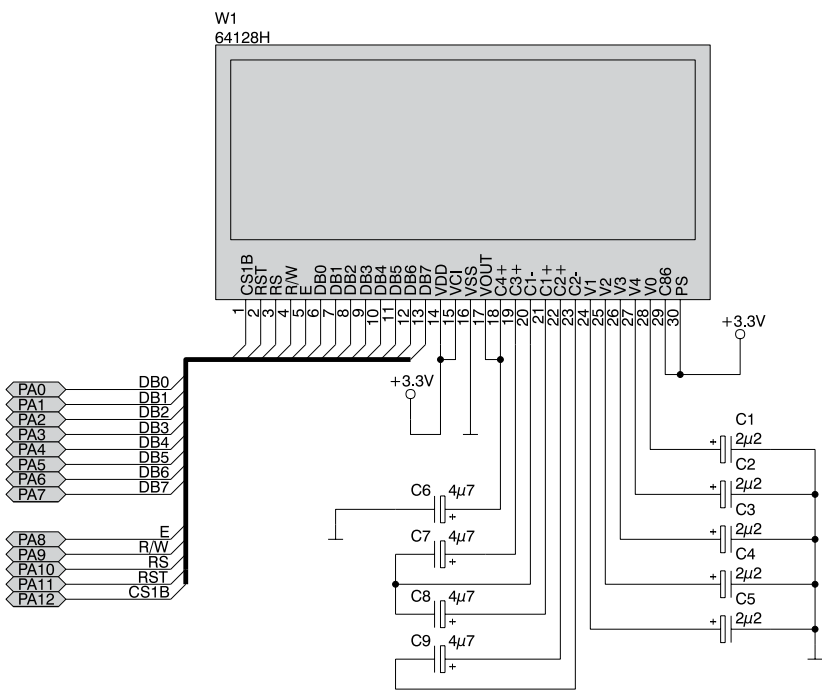
DON = 0 – wyświetlanie obrazu wyłączone
DON = 1 – wyświetlanie obrazu włączone

INITIAL DISPLAY LINE

Instrukcja określa, która linia pamięci obrazu zostanie wyświetlona w pierwszej linii wyświetlacza.



Rys. 2. Schemat elektryczny podświetlacza RGB



Rys. 3. Sposób podłączenia wyświetlacza do mikrokontrolera AT91SAM7S256

		Y	Y	Y	Y	Y	Y
		=	=	=	=	=	=
		0	1	2	3	4	5
Piksel 0	Bit 0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 1	Bit 1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 2	Bit 2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 3	Bit 3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 4	Bit 4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 5	Bit 5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 6	Bit 6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 7	Bit 7	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 8	Bit 0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 9	Bit 1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 10	Bit 2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 11	Bit 3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 12	Bit 4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 13	Bit 5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 14	Bit 6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Piksel 15	Bit 7	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Rys. 4. Organizacja pamięci sterownika

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	ST5	ST4	ST3	ST2	ST1	ST0

ST5:ST0 – Numer linii pamięci obrazu, która będzie wyświetlana jako pierwsza na ekranie (0...63)

SET PAGE ADDRESS

Instrukcja wyboru adresu aktywnej strony.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	1	1	P3	P2	P1	P0

SET COLUMN ADDRESS

Dwubajtowa instrukcja wyboru adresu kolumny. W pierwszej kolejności należy ustawić starszą połowę adresu kolumny a następnie młodszą. Każda operacja zapisu lub odczytu danych z wyświetlacza powoduje automatyczną inkrementację adresu kolumny. Po osiągnięciu wartości maksymalnej dalsza inkrementacja jest wstrzymywana.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	Y7	Y6	Y5	Y4

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	Y3	Y2	Y1	Y0

Y7:Y0 – adres kolumny (0..131)

READ STATUS

Instrukcja odczytu rejestru statusowego, zawierającego flagi określające aktualny stan wyświetlacza.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	BUSY	ADC	ON/OFF	RESETB	0	0	0	0

BUSY = 0: układ gotowy do wykonania instrukcji
 BUSY = 1: układ zajęty wykonywaniem operacji
 ADC = 0: kierunek odwrotny (SEG131->SEG0)
 ADC = 1: kierunek normalny (SEG0->SEG131)
 ON/OFF = 0: wyświetlacz włączony

ON/OFF = 1: wyświetlacz wyłączony
 RESETB = 0: układ gotowy do pracy
 RESETB = 1: układ w trakcie operacji RESET

WRITE DISPLAY DATA

Instrukcja zapisu danych obrazu pod aktualny adres. Po każdej operacji zapisu adres kolumny jest inkrementowany. Po osiągnięciu wartości maksymalnej dalsza inkrementacja jest wstrzymywana, jak również nie ulega zmianie adres aktywnej strony.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0								Dane do zapisu

READ DISPLAY DATA

Instrukcja odczytu danych spod aktualnego adresu. Po każdej operacji odczytu adres kolumny jest inkrementowany. Po osiągnięciu wartości maksymalnej dalsza inkrementacja jest wstrzymywana, jak również nie ulega zmianie adres aktywnej strony.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1								Odczytywane dane

REVERSE DISPLAY ON/OFF

Instrukcja wyboru trybu odwzorowania bitu pamięci obrazu na piksel ekranu.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	1	0	0	1	1	REV

REV = 0: wyświetlacz pozytywny (bit o wartości 1 powoduje zaciemnienie piksela)
 REV = 1: wyświetlacz negatywny (bit o wartości 0 powoduje zaciemnienie piksela)

ENTIRE DISPLAY ON/OFF

Instrukcja zaciemniająca wszystkie piksele.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	1	0	0	1	0	EON

EON = 0: tryb normalny
 EON = 1: wszystkie piksele zaciemnione

SET MODIFY-READ

Instrukcja powoduje zatrzymanie inkrementacji adresu po operacji odczytu z zachowaniem inkrementacji po operacji zapisu danych. Instrukcja pozwala zmniejszyć obciążenie procesora podczas modyfikacji dużych obszarów pamięci. Przywrócenie normalnego trybu pracy następuje po wykonaniu instrukcji *Reset Modify-Write*.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	1	1	0	0	0	0	0

RESET MODIFY-READ

Instrukcja kończąca tryb *Modify-Write*. Po wykonaniu instrukcji przywrócony zostaje stan rejestru kolumn sprzed uaktywnienia trybu *Modify-Write*.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	1	1	0	1	1	1	0

RESET

Instrukcja przywracająca rejestry adresu kolumn, stron, linii początkowej oraz rejestrów kierunku przemieszczania do stanu domyślnego. Instrukcja nie inicjalizuje układów zasilania, które są inicjalizowane za pomocą wyprowadzenia RESETB.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	1	1	0	0	0	1	0

SHL SELECT

Instrukcja wyboru kierunku przemieszczania wierszy

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	1	0	0	SHL	x	x	x

SHL = 0: COM0->COM63
 SHL = 1: COM63->COM0

ADC SELECT

Instrukcja wyboru kierunku przemieszczania kolumn.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	0	1	0	0	0	0	ADC

ADC = 0: kierunek normalny (SEG0->SEG131)
 ADC = 1: kierunek odwrotny (SEG131->SEG0)

POWER CONTROL

Instrukcja sterująca wewnętrznymi układami zasilania wyświetlacza.

RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	0	1	VC	VR	VF

VC	VR	VF	Stan wewnętrznych układów zasilania
0	x	x	Układ przetwornicy napięcia jest wyłączony
1	x	x	Układ przetwornicy napięcia jest włączony
x	0	x	Układ regulatora napięcia jest wyłączony
x	1	x	Układ regulatora napięcia jest włączony
x	x	0	Układ wtórnika napięcia jest wyłączony
x	x	1	Układ wtórnika napięcia jest włączony

PROCEDURY OBSŁUGI WYŚWIETLACZA

Jak wspomiano na wstępie, procedury obsługi wyświetlacza zostały napisane w języku C dla mikrokontrolerów z rodziny AT91SAM7S oraz kompilatora *arm-elf-gcc* (WinARM).

W przypadku, gdy wyświetlacz ma być podłączony do innych wyprowadzeń mikrokontrolera niż przedstawione na schemacie, należy dokonać modyfikacji w pliku *S6B0724.h*. W pliku tym umieszczone zostały definicje poszczególnych sygnałów sterujących oraz magistrali danych:

```
#define S6B_E (1 << 8)
/* wyprowadzenie PA8 -> E */
#define S6B_RW (1 << 9)
```

```

/* wyprowadzenie PA9 -> RW*/
#define S6B_RS (1 << 10)
/* wyprowadzenie PA10 -> RS */
#define S6B_RST (1 << 11)
/* wyprowadzenie PA11 -> RST */
#define S6B_CS1B (1 << 12)
/* wyprowadzenie PA12 -> CS1B */
#define S6B_D0 0
/* pierwszym bit szyny danych
jest wyprowadzenie PA0 */

```

Magistrala danych zajmuje osiem kolejnych wyprowadzeń portu PA, przy czym możemy dowolnie określić, które wyprowadzenie portu będzie bitem 0 szyny danych.

Wyświetlacz może pracować w dwóch pozycjach (jak widać na fotografiach na początku artykułu): z wyprowadzeniami u dołu ekranu (tryb normalny) i z wyprowadzeniami u góry ekranu (tryb „odwrotny”), przy czym współrzędne (0,0) zawsze będą wskazywały na lewy górny róg ekranu. Jest to możliwe dzięki odpowiedniej kombinacji bitów ADC i SHL. Określenie orientacji wyświetlacza sprowadza się do wykomentowania jednej z dwóch następujących linii znajdujących się w pliku *S6B0724.h*:

```

#define POSITION_NORMAL
#define POSITION_REVERSE

```

Zawsze jedna z tych linii powinna być „wykomentowana”.

ODCZYT REJESTRU STATUSOWEGO

Instrukcja odczytu rejestru statusowego. Kontroler wyświetlacza do czasu zakończenia wykonywania poprzedniej instrukcji ignoruje kolejne instrukcje, dlatego też przed zapisaniem instrukcji należy sprawdzić stan flagi BUSY. W przypadku gdy znajduje się ona w stanie wysokim należy odczekać na jej wyzerowanie. (list. 1)

ZAPIS ROZKAZU

Przed zapisaniem rozkazu sprawdzana jest flaga BUSY, w przypadku gdy jest ona w stanie wysokim program oczekuje na jej wyzerowanie (list. 2).

ZAPIS DANEJ

Procedura zapisu danej przeznaczonej do wyświetlenia (list. 3). Przed zapisaniem danych sprawdzana jest flaga BUSY, w przypadku gdy jest ona w stanie wysokim program oczekuje na jej wyzerowanie. Każdorazowe zapisanie danej do wyświetlacza powoduje inkrementację adresu kolumny. Po osiągnięciu wartości maksymalnej dalsza inkrementacja jest wstrzymywana.

```

unsigned char GLCD_ReadStatus(void)
{
volatile int tmp, i;
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_CS1B | S6B_RS); // CS1B = 0; //RS = 0
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_RW); // RW = 1 -> Odczyt
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_CfgInput(AT91C_BASE_PIOA, (0xFF << S6B_D0));
tmp = ((AT91F_PIO_GetInput(AT91C_BASE_PIOA) >> S6B_D0) & 0xFF);
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_CS1B);
return tmp;
}

```

List. 1. Procedura odczytu rejestru statusowego

```

void GLCD_WriteCommand(unsigned char commandToWrite)
{
while((GLCD_ReadStatus() & STATUS_BUSY));
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_RS | S6B_CS1B | S6B_RW);
AT91F_PIO_CfgOutput(AT91C_BASE_PIOA, (0xFF << S6B_D0));
AT91F_PIO_ForceOutput(AT91C_BASE_PIOA,
(unsigned int)(commandToWrite << S6B_D0));
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_CS1B);
}

```

List. 2. Procedura zapisu rozkazu (z kontrolą BUSY)

```

void GLCD_WriteData(unsigned char dataToWrite)
{
while((GLCD_ReadStatus() & STATUS_BUSY));
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_RW | S6B_CS1B);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_RS);
AT91F_PIO_CfgOutput(AT91C_BASE_PIOA, (0xFF << S6B_D0));
AT91F_PIO_ForceOutput(AT91C_BASE_PIOA, (unsigned int)(dataToWrite <<
S6B_D0));
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_CS1B);
}

```

List. 3. Procedura zapisu danej (z kontrolą BUSY)

```

unsigned char GLCD_ReadData(void)
{
unsigned char volatile tmp = 0;
while((GLCD_ReadStatus() & 0x80));
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_CS1B);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_RS | S6B_RW);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_CfgInput(AT91C_BASE_PIOA, (0xFF << S6B_D0));
tmp = ((AT91F_PIO_GetInput(AT91C_BASE_PIOA) >> S6B_D0) & 0xFF);
AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_E);
AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_CS1B);
return tmp;
}

```

List. 4. Procedura odczytu danej (z kontrolą BUSY)

ODCZYT DANEJ

Przed odczytem danych sprawdzana jest flaga BUSY (list. 4), w przypadku gdy jest ona w stanie wysokim program oczekuje na jej wyzerowanie. Po wykonaniu instrukcji adres kolumny jest automatycznie inkrementowany. Po osiągnięciu wartości maksymalnej dalsza inkrementacja jest wstrzymywana.

INICJALIZACJA WYŚWIETLACZA

Procedura inicjalizacji wyświetlacza oraz portów mikrokontrolera. Następuje włączenie taktowania układu PIO, konfiguracja linii sterujących w tryb wyjściowy oraz ich ustawienie w stan

domyślny oraz konfiguracja szyny danych w tryb zapisu synchronicznego. Inicjalizacja wyświetlacza polega na ustawieniu, w zależności od konfiguracji w pliku *S6B0724.h*, w tryb wyświetlania normalny bądź odwrócony, włączenia wewnętrznych układów zasilających oraz ustawieniu odpowiedniej wartości napięcia zasilającego wyświetlacz (napięcia kontrastu) oraz włączeniu wyświetlania (list. 5).

USTAWIENIE WSPÓLRZĘDNYCH EKRANOWYCH

Funkcja ustawia współrzędne wyświetlacza, na których wykonana

```

void GLCD_Initialize(void)
{
    volatile int i;
    // inicjalizacja portów mikrokontrolera
    AT91F_PMC_EnablePeriphClock(AT91C_BASE_PMC, (1 << AT91C_ID_PIOA));
    // włączenie taktowania układu PIOA
    AT91F_PIO_CfgOutput(AT91C_BASE_PIOA, (S6B_RS | S6B_RW | S6B_RST |
    S6B_E | S6B_CS1B));
    // konfiguracja linii sterujących
    AT91F_PIO_OutputWriteEnable(AT91C_BASE_PIOA, (0xFF << S6B_D0));
    // konfiguracja linii danych
    // /inicjalizacja portów mikrokontrolera
    AT91F_PIO_ClearOutput(AT91C_BASE_PIOA, S6B_RST); // wyzerowanie kontrolera
    for(i=0; i<100; i++);
    AT91F_PIO_SetOutput(AT91C_BASE_PIOA, S6B_RST);
    #ifndef POSITION_NORMAL
    GLCD_WriteCommand(LCD_SET_ADC_NOR);
    GLCD_WriteCommand(LCD_SET_SHL_REV);
    #else
    GLCD_WriteCommand(LCD_SET_ADC_REV);
    GLCD_WriteCommand(LCD_SET_SHL_NOR);
    #endif
    GLCD_WriteCommand(LCD_SET_BIAS_0);
    GLCD_WriteCommand(LCD_PWR_CTL | 1);
    for(i=0; i<1000; i++);
    GLCD_WriteCommand(LCD_PWR_CTL | 3);
    for(i=0; i<1000; i++);
    GLCD_WriteCommand(LCD_PWR_CTL | 7);
    GLCD_WriteCommand(LCD_REG_RESISTOR | 5);
    GLCD_WriteCommand(LCD_REF_VOLT_MODE);
    GLCD_WriteCommand(LCD_REF_VOLT_REG | 18);
    for(i=0; i<1000; i++);
    GLCD_WriteCommand(LCD_DISP_ON);
}

```

List. 5. Procedura inicjalizacji wyświetlacza

```

void GLCD_GoTo(unsigned char column, unsigned char page)
{
    GLCD_WriteCommand(LCD_SET_PAGE | page);
    GLCD_WriteCommand(LCD_SET_COL_HI | ((column + COLUMN_OFFSET) >> 4));
    GLCD_WriteCommand(LCD_SET_COL_LO | ((column + COLUMN_OFFSET) & 0x0F));
}

```

List. 6. Procedura ustawienia współrzędnych

```

void GLCD_WriteChar(char charCode)
{
    unsigned char fontCollumn;
    for(fontCollumn = 0; fontCollumn < FONT_WIDTH; fontCollumn++)
        GLCD_WriteData(font5x7[((charCode - FONT_OFFSET) * FONT_WIDTH)
        + fontCollumn]);
    GLCD_WriteData(0);
}

```

List. 7. Procedura wyświetlenia znaku alfanumerycznego

```

void GLCD_WriteString(char * string)
{
    while(*string)
    {
        GLCD_WriteChar(*string++);
    }
}

```

List. 8. Procedura wyświetlania ciągu znaków

```

void GLCD_SetPixel(int x, int y, int color)
{
    unsigned char temp = 0; // zmienna pomocnicza
    GLCD_GoTo(x, (y/8)); // ponowne ustawienie współrzędnych
    temp = GLCD_ReadData(); // odczyt aktualnego stanu pikseli
    if(color)
        temp |= (1 << (y % 8));
    else
        temp &= ~(1 << (y % 8));
    GLCD_GoTo(x, (y/8)); // ponowne ustawienie współrzędnych
    GLCD_WriteData(temp); // zapis odpowiednio zmodyfikowanej wartości
}

```

List. 9. Procedura włączenia piksela

zostanie następną operacją (odczyt, bądź zapis). Współrzędna pionowa może przyjmować wartości z zakresu 0...8 (numer strony). Współrzędna pozioma może przyjmować warto-

ści z zakresu 0...131. Należy jednak pamiętać, że w przypadku wyświetlacza 64128H współrzędne są ograniczone organizacją ekranu wyświetlacza (list. 6).

WYŚWIETLENIE ZNAKU

Wyświetlenie znaku polega na zapisie do wyświetlacza 5 kolejnych bajtów tworzących dany znak. Ponieważ nie ma sensu marnować miejsca w pamięci programu na znaki „niewyświetlane” (czyli pierwszych 31 znaków tablicy ASCII) pierwszy znak w tablicy (spacja) ma indeks 0, podczas gdy w tablicy ASCII ten znak posiada kod 32. Należy więc od kodu znaku przekazanego do funkcji jako parametr odjąć liczbę 32 (list. 7).

WYŚWIETLENIE CIĄGU ZNAKÓW

Parametrem funkcji jest wskaźnik do typowego dla języka C ciągu znaków zakończonego zerem. Napis wyświetlany jest począwszy od aktualnej wartości adresów strony i kolumny. Przekroczenie maksymalnej współrzędnej pionowej nie powoduje przeniesienia wyświetlanego tekstu do kolejnej linii (list. 8).

WŁĄCZENIE PIKSELA

Włączenie piksela polega na ustawieniu odpowiadającego mu bitu w pamięci wyświetlacza. Ponieważ możemy odczytywać i zapisywać tylko cały bajt procedura włączenia piksela przebiega następująco:

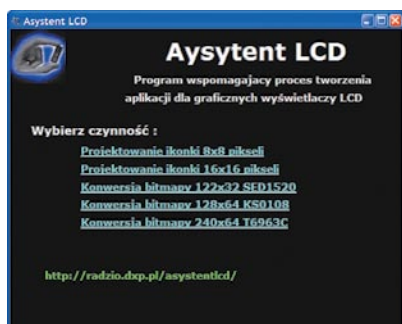
- ustawiamy współrzędne: poziomą oraz podzieloną przez 8 pionową (ponieważ dostęp do pamięci odbywa się całymi bajtami),
- odczytujemy z wyświetlacza aktualny stan pikseli,
- modyfikujemy odczytany bajt poprzez wykonanie na nim odpowiedniej operacji w zależności od wartości parametru *color*,
- zapisujemy tak zmodyfikowany bajt danych pod odpowiedni adres pamięci wyświetlacza (list. 9).

CZYSZCZENIE EKRANU

Procedura czyszcząca zawartość pamięci obrazu. Na skutek jej wykonania pamięć obrazu zostanie wypełniona bajtami o wartości zero. Współrzędne ekranowe po wyczyszczeniu ekranu są ustawiane na wartość (0,0) (list. 10).

WYŚWIETLENIE BITMAPY

Funkcja wyświetlająca bitmapę przyjmuje 5 argumentów: wskaźnik do tablicy z bitmapą znajdującą się w pamięci Flash, współrzędne pod którymi bitmapa ma zostać wyświet-



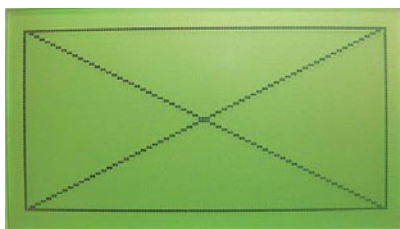
Rys. 5. Okno początkowe programu Asystent LCD



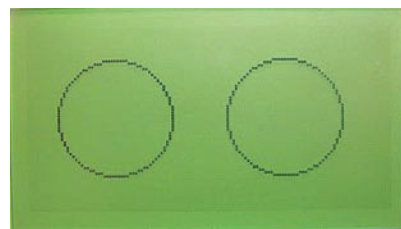
Rys. 6. Przykładowa bitmapa wyświetlona na ekranie wyświetlacza 64128H-RGB firmy Displaytech

tlona oraz jej rozmiar. Współrzędna pionowa przyjmuje wartości 0...7 (numer strony). Wysokość bitmapy może przyjmować wartości będące wielokrotnością liczby 8. Natomiast współrzędna pozioma i szerokość bitmapy mogą przyjmować dowolne wartości (mieszczące się oczywiście na ekranie wyświetlacza) (list. 11). Do konwersji plików BMP można wykorzystać na przykład program *Asystent*

Wyświetlacz do testów udostępniła redakcji firma Compart, www.compart.pl, tel. 22-6108527.



Rys. 7. Przykład wykorzystania funkcji rysującej prostokąt i linie proste



Rys. 8. Przykład wykorzystania funkcji rysującej okręgi

```
void GLCD_ClearScreen(void)
{
    unsigned char pageIndex, columnIndex;
    for(pageIndex = 0; pageIndex < NUMBER_OF_PAGES; pageIndex++)
    {
        GLCD_GoTo(0, pageIndex);
        for(columnIndex = 0; columnIndex < NUMBER_OF_COLUMNS; columnIndex++)
            GLCD_WriteData(0);
    }
    GLCD_GoTo(0,0);
}
```

List. 10. Procedura czyszczenia ekranu

```
void GLCD_Bitmap(char * bitmap, unsigned char left, unsigned char top,
    unsigned char width, unsigned char height)
{
    unsigned char pageIndex, collumnIndex;
    for(pageIndex = 0; pageIndex < height / 8; pageIndex++)
        // petla wykona się tyle razy, ile stron zajmuje bitmapa
    {
        GLCD_GoTo(left, top + pageIndex); // ustawienie współrzędnych
        for(collumnIndex = 0; collumnIndex < width; collumnIndex++)
            // petla wykona się tyle razy, ile pikseli szerokości ma bitmapa
            GLCD_WriteData(*(bitmap++)); // zapis danych
    }
}
```

List. 11. Procedura wyświetlenia bitmapy

LCD <http://radio.dxp.pl/asystentlcd/> (rys. 5), korzystając z opcji konwersji bitmapy 128x64 pikseli dla sterownika KS0108 (organizacja pamięci kontrolera S6B0724 jest bardzo podobna do organizacji pamięci kontrolera KS0108). Przykład bitmapy wyświetlonej na wyświetlaczu przedstawiono na rys. 6.

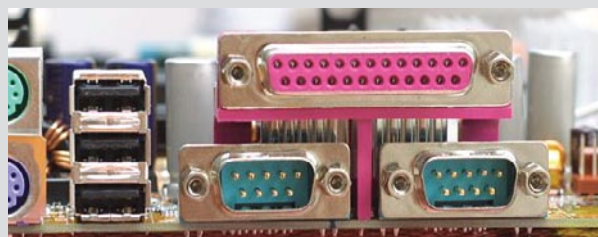
FUNKCJE GRAFICZNE

Przykład działania funkcji rysującej prostokąt i linie przedstawiono na rys. 7, natomiast działanie funkcji rysującej okrąg przedstawiono na rys. 8.

Radosław Kwiecień, EP
radoslaw.kwiecien@ep.com.pl

LCD i PC

Pod adresem <http://graphlcd.berlios.de> jest dostępne oprogramowanie umożliwiające obsługę wyświetlaczy wyposażonych w sterowniki wymienione w ramce za pomocą komputera PC. Za jego pomocą można wyprzewodzić wybrane dane z PC i oprogramowania na nim zainstalowanego na wyświetlacz graficzny (także VFD), co pozwala stosunkowo niewielkim kosztem efektywnie doposażyć komputer.



Hitachi HD61830
Toshiba T6963C
Samsung KS0108
Epson SED1520
Epson SED1330

Noritake GU140X32F-7806
Noritake GU256X64-372
Noritake GU256X64-3XX0
Noritake 800 Series