

## 1. LS20 LCD Shield v1.2 Wattercott



Экран 2.1" 65536 цветов TFT с разрешением 176 x 132 пикселя.

На плате установлено:

- колесо прокрутки (энкодер)
- слот для карты памяти SD

Документация:

Схема: [http://www.wide.hk/pdf/S65\\_TFT.pdf](http://www.wide.hk/pdf/S65_TFT.pdf)

Оригинал

статьи:

[www.mon-club-](http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.LibrairieS65Shield#toc13)

[elec.fr/pmwiki\\_reference\\_arduino/pmwiki.php?n=Main.LibrairieS65Shield#toc13](http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.LibrairieS65Shield#toc13)

Библиотеки и примеры: <http://www.ebay.com/itm/130574761900?rmvSB=true>

<https://github.com/watterott/S65-Shield>

[http://www.mon-club-](http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.LibrairieS65ShieldTelecharger)

[elec.fr/pmwiki\\_reference\\_arduino/pmwiki.php?n=Main.LibrairieS65ShieldTelecharger](http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.LibrairieS65ShieldTelecharger)

### 2. Дисплей S65.

S65 экран 2.1 дюйма TFT 65536 цветов с разрешением 176 x132 пикселя. Размер экрана мобильного телефона. Необходимое питание для работы экрана обеспечивается платой. Плата управляется по протоколу SPI и требует 5 контактов.

Контакты, используемые для SPI :

- S65\_DAT (pin 11 Arduino)
- S65\_CLK (pin 13 Arduino)

Контакты, используемые для управления платой графического дисплея :

- S65\_RST (analog 3 Arduino)
- S65\_CS (analog 2 Arduino)
- S65\_RS (pin 4 Arduino)

Цвет кодируется 16 bits : 5 bits для красного, 6 bits для зеленого, 5 bits для синего (32 x 64 x 32 = 65536).

### 3. Библиотечные функции дисплея

#### 3.1 Функции инициализации

- **S65Display**: конструктор

##### Описание:

Объявление объекта типа S65Display

##### Синтаксис:

```
S65Display lcd; // Объявляем объект S65Display под названием lcd
```

##### Параметры:

*lcd* : имя созданного объекта **S65Display**

```
init(clock_div) : инициализация модуля
```

##### Описание:

Эта инициализирует контакты, используемые для связи по SPI с графическим дисплеем

S65\_DAT (pin 11 Arduino)  
S65\_CLK (pin 13 Arduino)  
инициализирует контакты:  
S65\_RST (analog 3 Arduino)  
S65\_CS (analog 2 Arduino)  
S65\_RS (pin 4 Arduino)

##### Синтаксис:

```
lcd.init(4); //spi-clk = Fcpu/4
```

##### Параметры:

*lcd* : объект **S65Display**

*clock\_div* : скорость SPI =  $F_{osc}/clock\_div$  (type uint8)

- **clear(color)** : очистка экрана , желаемым цветом

##### Описание:

Эта функция очищает экран и заполняет его указанным цветом

##### Синтаксис:

```
lcd.clear(color);
```

##### Параметры:

*lcd* : объект **S65Display**

*color* : цвет в формате RGB

##### Пример:

```
lcd.clear(RGB(255,255,255)); // очистка экрана белым цветом
```

### 3.2 Функции управления цветом

- **RGB(r,g,b)** : цвет в формате RGB (красный, зеленый, синий)

Эта функция управляет цветом изображения в формате RGB. Графический дисплей имеет 16-ти битный цвет с кодированием: 5 бит - для красного, 6 бит - для зеленого, 5 бит - для синего. RGB идентификатор определяется для каждого компонента 0 – 255, поэтому:

RGB(255,0,0) - красный  
RGB(0,255,0) - зеленый  
RGB(0,0,255) - синий

#### Синтаксис:

```
RGB(red,green,blue);
```

#### Параметры:

*lcd* : объект **S65Display**

*red* : значение для красного цвета 0-255 - 32 фактических уровня

*green*: значение для зеленого цвета 0-255 - 64 фактических уровня

*blue*: значение для синего цвета 0-255 - 32 фактических уровня

#### Пример:

```
clear(RGB(255,50,100)); // установить цвет экрана фиолетовым цветом
```

```
// Выводим текст "Hello world" в позиции экрана 10,10 синим цветом на фиолетовом фоне
```

```
drawText(10, 10, "Hello world", RGB(0,0,255), RGB(255,50,100));
```

```
// Выводим текст "Hello world" из FLASH памяти в позиции экрана 10,20 синим цветом на фиолетовом фоне
```

```
drawTextPGM(10, 20, PSTR("Hello world"), RGB(0,255,0), RGB(255,50,100));
```

### 4.3 Функции рисования

- **drawPixel(x0, y0, color)** : установить точку на экране с координатами по X – x0, Y – y0 и цветом color

**void drawPixel(uint8\_t x0, uint8\_t y0, uint16\_t color)**

Функция отображает цветную точку с координатами X и Y. Экран имеет координаты экрана 0,0 в левом верхнем углу

## Синтаксис

```
lcd.drawPixel(x, y, color);
```

### Параметры:

*lcd* : объект **S65Display**

*x* , *y* : координаты точки

*color* : цвет точки в формате RGB

### Пример:

```
lcd.drawPixel(100, 100, RGB(255,0,0)); // отображает красную точку в позиции (100,100)
```

- **drawLine**(*x0*, *y0*, *x1*, *y1*, *color*) : рисует цветную линию

**void drawLine(uint8\_t x0, uint8\_t y0, uint8\_t x1, uint8\_t y1, uint16\_t color)**

### Описание:

Функция рисует линию с цветом *color* из точки с координатами *x0*,*y0* в точку *x1*,*y1*.

### Синтаксис:

```
lcd.drawLine( x0, y0, x1, y1, color);
```

### Параметры:

*lcd* : объект **S65Display**

*x0* , *y0* : абсолютные координаты начала линии от начальной точки в пикселях

*x1* , *y1* : абсолютные координаты конца линии от начальной точки в пикселях

*color* : цвет линии в формате RGB

### Пример:

```
// рисование красной линии из начальной точки с координатами x0 = 0, y0 = 0 (левый верхний угол экрана) в точку с координатами X,Y - 100,100  
lcd.drawLine( 0, 0, 100, 100, RGB(255,0,0));
```

### Комментарии пользователей

При использовании параметров *x0*,*y0* и *x1*,*y1* больше значения размеров экрана 0-131 и 0-175, линия не появляется

- **drawRect**(*x0*, *y0*, *x1*, *y1*, *color*) : рисует прямоугольник

**void drawRect(uint8\_t x0, uint8\_t y0, uint8\_t x1, uint8\_t y1, uint16\_t color)**

### Описание:

Функция рисует прямоугольник шириной = *x1*-*x0*, высотой = *y1*-*y0*, из точки с координатами *x0*,*y0*.

### Синтаксис:

```
lcd.drawRect(x0, y0, x1, y1, color);
```

### Параметры:

*lcd* : объект **S65Display**

*x0* , *y0* : координаты левого верхнего угла прямоугольника  
*x1* , *y1* : координаты правого нижнего угла прямоугольника  
*color* : цвет линии прямоугольника в формате RGB

**Пример:**

```
lcd.drawRect(x, y, x+50, y+100, RGB(0,255,0)); // рисует прямоугольник размерами 50  
x 100 пикселей
```

- **fillRect**(*x0*, *y0*, *x1*, *y1*, *color*) : рисует прямоугольник с заливкой цветом

**void fillRect(uint8\_t x0, uint8\_t y0, uint8\_t x1, uint8\_t y1, uint16\_t color)**

**Описание:**

Функция рисует прямоугольник, заполненный желаемым цветом

**Синтаксис:**

```
lcd.fillRect(x0, y0, x1, y1, color);
```

**Параметры:**

*lcd* : объект **S65Display**

*x0* , *y0* : координаты левого верхнего угла прямоугольника

*x1* , *y1* : координаты правого нижнего угла прямоугольника

*color* : цвет линии прямоугольника в формате RGB

**Пример:**

```
lcd.fillRect(x, y, x+100, y+50, RGB(255,0,255)); // фиолетовый прямоугольник с  
размерами 50x100 пикселей
```

- **drawCircle**(*x0*, *y0*, *radius*, *color*) : рисует окружность

**void drawCircle(uint8\_t x0, uint8\_t y0, uint8\_t radius, uint16\_t color)**

**Описание:**

Функция рисует окружность желаемого размера и цвета линии

**Синтаксис:**

```
lcd.drawCircle(x0, y0, rayon, color);
```

**Параметры:**

*lcd* : объект **S65Display**

*x0* , *y0* : абсолютные координаты центра окружности в пикселях

*rayon* : радиус окружности в пикселях

*color* : цвет линии окружности в формате RGB

**Пример:**

```
lcd.drawCircle(100, 100, 25, RGB(0,255,0)); // рисует окружность радиуса 25 пикселей  
синего цвета , с центром окружности 100,100
```

- **fillCircle**(x0, y0, radius, color) : рисует окружность с заливкой цветом

**fillCircle(uint8\_t x0, uint8\_t y0, uint8\_t radius, uint16\_t color)**

**Описание:**

Функция рисует окружность желаемого размера и цвета.

**Синтаксис:**

```
lcd.fillCircle(x0, y0, rayon, color);
```

*lcd* : объект **S65Display**

*x0, y0* : абсолютные координаты центра окружности в пикселях

*rayon* : радиус окружности в пикселях

*color* : цвет окружности в формате RGB

**Пример:**

```
lcd.fillCircle(50, 75, 20, RGB(255,255,0)); // рисует желтую окружность с центром (50,75) и радиусом 20 пикселей
```

### 3.4 Текстовые функции

- **drawChar**(x, y, c, size, color, bg\_color) : отображает символ

**uint8\_t drawChar(uint8\_t x, uint8\_t y, char c, uint8\_t size, uint16\_t color, uint16\_t bg\_color)**

**Описание:**

Процедура отображает символ в кодировке ASCII (см.таблицу кодировки), в указанное координатами место, требуемого размера, цвета и цвета фона. Возвращает следующую позицию по X.

**Синтаксис:**

```
lcd.drawChar( x, y, c, size, color, bg_color);  
lcd.drawChar( x, y, 'L', size, color, bg_color);
```

**Параметры:**

*lcd* : объект **S65Display**

*x, y* : абсолютные координаты положения символа в пикселях (unsigned char)

*c* : код символа ASCII . (unsigned char - из SRAM)

*'L'* : символ

*size* : размер шрифта. Размер шрифта соответствует:

- 1 – размер шрифта 8x12
- 2 – размер шрифта 16x24
- 3 – 24x36 и т.д

Для более подробной информации см. Размер шрифта

*color* : цвет символа в формате RGB (unsigned int)

*bg\_color* : цвет фона символа в формате RGB (unsigned int)

**Пример:**

```
// выводит символ «0» в позиции (100,100) размером 8x12, красного цвета на синем фоне
```

```
lcd.drawChar( 100, 100, 48, 1, RGB(255,0,0), RGB(0,0,255));
```

- **drawText**(x, y, stringSram, size, color, bg\_color) : выводит строку символов из SRAM

**uint8\_t drawText(uint8\_t x, uint8\_t y, char \*s, uint8\_t size, uint16\_t color, uint16\_t bg\_color)**

#### Описание:

Функция выводит на экран строку символов из SRAM в указанную позицию, с заданным размером шрифта, цвета символов и цвета фона. Возвращает слудующую позицию по X.

#### Синтаксис:

```
lcd.drawText(x, y, "chaine", size, color, bg_color);
```

#### Параметры:

*lcd* : объект **S65Display**

*x, y* : абсолютные координаты положения строки в пикселях (unsigned char)

*c* : строка символов в кодировке ASCII .

*"chaine"*: строка символов

*size* : рамер шрифта. Размер шрифта соответствует:

1 – размер шрифта 8x12

2 – размер шрифта 16x24

3 – 24x36 и т.д

Для более подробной информации см. Размер шрифта

*color* : цвет символа в формате RGB (unsigned int)

*bg\_color* : цвет фона символа в формате RGB (unsigned int)

#### Пример:

```
// выводит строку "example" с размером символов – 1, фиолетовым цветом на белом фоне в позиции 5-,50
```

```
lcd.drawText(50, 50, "example", 1, RGB(255,0,255), RGB(255,255,255));
```

- **drawTextPGM**(x, y, stringFlash, size, color, bg\_color) : выводит строку символов из FLASH

**uint8\_t drawTextPGM(uint8\_t x, uint8\_t y, PGM\_P s, uint8\_t size, uint16\_t color, uint16\_t )**

#### Описание:

Функция выводит на экран строку символов из FLASH в указанную позицию, с заданным размером шрифта, цвета символов и цвета фона. Возвращает следующую позицию по X.

#### Синтаксис:

```
lcd.drawTextPGM(x, y, string, size, color, bg_color);  
lcd.drawTextPGM(10, 20, PSTR("Hello world"), size, RGB(0,255,0), RGB(255,50,100));
```

#### Параметры:

*lcd* : объект **S65Display**

*x, y* : абсолютные координаты положения строки в пикселях (unsigned char)

*c* : строка символов в кодировке ASCII .

*string*: строка символов из SRAM памяти

*PSTR()*: строка символов из FLASH

*size* : размер шрифта. Размер шрифта соответствует:

- 1 – размер шрифта 8x12
- 2 – размер шрифта 16x24
- 3 – 24x36 и т.д

Для более подробной информации см. Размер шрифта

*color* : цвет символа в формате RGB (unsigned int)

*bg\_color* : цвет фона символа в формате RGB (unsigned int)

#### Пример:

```
lcd.drawTextPGM(10, 20, PSTR("Hello world"),size, RGB(0,255,0), RGB(255,50,100));
```

- **drawMLText**(*x, y, stringSram, size, color, bg\_color*) : выводит строку символов из SRAM

**uint8\_t drawMLText(uint8\_t x, uint8\_t y, char \*s, uint8\_t size, uint16\_t color, uint16\_t bg\_color)**

#### Описание:

Функция выводит на экран цепочку символов из FLASH в указанную позицию (символы перевода строки - \n могут быть включены в строку), с заданным размером шрифта, цвета символов и цвета фона. Возвращает следующее значение X

#### Синтаксис:

```
drawMLText( x, y, "chaine", size, color, bg_color)
```

*lcd* : объект **S65Display**

*x, y* : абсолютные координаты положения строки в пикселях (unsigned char)

*c* : строка символов в кодировке ASCII .

*"chaine"*: строка символов

*size* : размер шрифта. Размер шрифта соответствует:

- 1 – размер шрифта 8x12
- 2 – размер шрифта 16x24
- 3 – 24x36 и т.д



Для более подробной информации см. Размер шрифта  
*color* : цвет символа в формате RGB (unsigned int)  
*bg\_color* : цвет фона символа в формате RGB (unsigned int)

- **drawMLTextPGM**(*x*, *y*, *stringFlash*, *size*, *color*, *bg\_color*) : выводит цепочку символов из памяти FLASH

### 3.5 Вывод чисел

- **drawInteger**(*x0*, *y0*, *integerNumber*, *base*, *size*, *color*, *bg\_color*) : вывод целых чисел в указанные: позицию, систему исчисления, цвет цифр числа и цвет фона

**drawInteger(uint8\_t x0, uint8\_t y0, integerNumber, uint8\_t base, uint8\_t size, uint16\_t color, uint16\_t bg\_color)**

#### Описание:

Функция выводит целые числа (char, int, long) в позицию, с указанными координатами, с заданной системой исчисления, размером шрифта, цветом символов и фона.

#### Синтаксис:

```
lcd.drawInteger(x0, y0, integerNumber, base, size, color, bg_color);
```

#### Параметры:

*lcd* : объект **S65Display**

*x0*, *y0* : абсолютные координаты положения строки в пикселях (unsigned char)

*integerNumber* : значение целого типа ([char](#), [int](#), [long](#)).

*base* : система исчисления, DEC, HEX, OCT или BIN, десятичная, шестнадцатеричная, восьмеричная 8 или двоичная 2.

*size* : размер шрифта. Размер шрифта соответствует:

- 1 – размер шрифта 8x12
- 2 – размер шрифта 16x24
- 3 – 24x36 и т.д

Для более подробной информации см. Размер шрифта

*color* : цвет символа в формате RGB (unsigned int)

*bg\_color* : цвет фона символа в формате RGB (unsigned int)

#### Пример:

```
long valeur=-2040806/2;
lcd.drawInteger (0, 0, valeur,DEC,1,RGB(255,0,0), RGB(255,255,0));
valeur=-65535;
lcd.drawInteger (20, 50, valeur,HEX,2,bleu,rouge);
```

- **drawFloat**(x0, y0, FloatNumber, digits, size, color, bg\_color) : вывод чисел с плавающей точкой

**drawFloat(uint8\_t x0, uint8\_t y0, double number, uint8\_t digits, uint8\_t size, uint16\_t color, uint16\_t bg\_color)**

Функция выводит числа с плавающей точкой (**float** или **double**) в позицию, с указанные координатами, с заданной системой исчисления, размером шрифта, цветом символов и фона.

**Синтаксис:**

---

```
lcd.drawFloat( x0, y0, number, digits, size, color, bg_color)
```

**Пример:**

---

```
float xf=-3.141597;

lcd.drawFloat (0,100,xf,6,1,RGB(255,255,0), RGB(0,0,255));

lcd.drawFloat (0,100,xf,6,1,jaune, bleu);
```

---

### 3.6 Дополнительные функции

Эти функции используются внутри библиотеки S65

- **setCursor**(x, y)

**setCursor(uint8\_t x, uint8\_t y)**

**Описание:**

---

Функция определяет позицию текущего пикселя. Ничего не отображает.

**Синтаксис:**

---

```
lcd.setCursor(x, y)
```

**Параметры:**

---

*lcd* : объект **S65Display**

*x* , *y* : координаты текущего пикселя

- **setArea**(x0,y0, x1, y1)

**setArea(uint8\_t x0, uint8\_t y0, uint8\_t x1, uint8\_t y1)**

**Описание:**

---

Функция устанавливает прямоугольную область рисования

**Синтаксис:**

---

```
lcd.setArea(x0, y0, x1, y1);
```

**Параметры:**

---

*lcd* : объект **S65Display**

*x0* , *y0* : координаты левого верхнего угла прямоугольника

*x1* , *y1* : координаты правого нижнего угла прямоугольника

**Пример:**

---

```
lcd.setArea(0, 0, (S65_WIDTH-1), (S65_HEIGHT-1));
```

- **drawStart**()

Эта функция используется внутри библиотеки S65. Включает режим пользовательских данных: RS – низкое значение, CS – низкое.

Подробнее

: [http://www.mikrocontroller.net/attachment/19342/Using\\_the\\_Siemens\\_S65\\_Display.pdf](http://www.mikrocontroller.net/attachment/19342/Using_the_Siemens_S65_Display.pdf)

**Синтаксис:**

---

```
lcd.drawStart();
```

**Параметры:**

---

*lcd* : объект **S65Display**

- **draw**( color)

**draw(uint16\_t color)**

Функция устанавливает цвет текущего пикселя. Работает только в режиме drawStart.

**Синтаксис:**

---

```
lcd.draw(color);
```

### Параметры:

---

*lcd* : объект **S65Display**

*color* : цвет символа в формате RGB (unsigned int)

### Пример:

---

```
lcd.draw(RGB(255,0,0)); // установить текущий пиксель красным
```

- **drawStop()**

**void drawStop()**

### Описание:

---

Функция отменяет пользовательский режим.

Подробнее

: [http://www.mikrocontroller.net/attachment/19342/Using\\_the\\_Siemens\\_S65\\_Display.pdf](http://www.mikrocontroller.net/attachment/19342/Using_the_Siemens_S65_Display.pdf)

### Синтаксис:

---

```
lcd.drawStop();
```

### Идентификаторы

*RGB(r,g,b)* : формат цвета RGB (red, green,blue)

*S65\_WIDTH* : ширина экрана

*S65\_HEIGT* : высота экрана

Цвета:

green, blue, red, black, yellow, white, purple

#### 4. Таблица ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	<b>NUL</b> (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	<b>SOH</b> (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	<b>STX</b> (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	<b>ETX</b> (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	<b>EOT</b> (end of transmission)	36	24	044	&#36;	\$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	<b>ENQ</b> (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	<b>ACK</b> (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	<b>BEL</b> (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	<b>BS</b> (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	<b>TAB</b> (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	<b>LF</b> (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	<b>VT</b> (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	<b>FF</b> (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	<b>CR</b> (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	<b>SO</b> (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	<b>SI</b> (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	<b>DLE</b> (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	<b>DC1</b> (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	<b>DC2</b> (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	<b>DC3</b> (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	<b>DC4</b> (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	<b>NAK</b> (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	<b>SYN</b> (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	<b>ETB</b> (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	<b>CAN</b> (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	<b>EM</b> (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	<b>SUB</b> (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	<b>ESC</b> (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	<b>FS</b> (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	
29	1D	035	<b>GS</b> (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	<b>RS</b> (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	<b>US</b> (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: [www.LookupTables.com](http://www.LookupTables.com)

128	Ç	144	É	161	í	177	☐	193	⊥	209	〒	225	ß	241	±
129	ü	145	æ	162	ó	178	☐	194	⌞	210	⌞	226	Γ	242	≥
130	é	146	Æ	163	ú	179		195	⌞	211	⌞	227	π	243	≤
131	â	147	ô	164	ñ	180	⌞	196	—	212	⌞	228	Σ	244	∫
132	ä	148	ö	165	Ñ	181	⌞	197	+	213	⌞	229	σ	245	∫
133	à	149	ò	166	°	182	⌞	198	⌞	214	⌞	230	μ	246	÷
134	â	150	û	167	°	183	⌞	199	⌞	215	⌞	231	τ	247	≈
135	ç	151	ù	168	¿	184	⌞	200	⌞	216	⌞	232	Φ	248	°
136	ê	152	—	169	—	185	⌞	201	⌞	217	⌞	233	⊙	249	.
137	ë	153	Ö	170	⌞	186	⌞	202	⌞	218	⌞	234	Ω	250	.
138	è	154	Ü	171	½	187	⌞	203	⌞	219	■	235	δ	251	√
139	ï	156	£	172	¼	188	⌞	204	⌞	220	■	236	∞	252	—
140	î	157	¥	173	¡	189	⌞	205	=	221	■	237	φ	253	²
141	ì	158	—	174	«	190	⌞	206	⌞	222	■	238	ε	254	■
142	Ä	159	ƒ	175	»	191	⌞	207	⌞	223	■	239	∩	255	
143	Å	160	á	176	☐	192	⌞	208	⌞	224	α	240	≡		

Source: [www.LookupTables.com](http://www.LookupTables.com)