|  |
| --- |
| VGA Library |
| Functie lijst |

|  |
| --- |
| Niek Ratering Arntz & Matthijs Daggelders  30-5-2018 |

Inhoud

[draw\_line() 2](#_Toc515496150)

[draw\_ellipse() 3](#_Toc515496151)

[draw\_rectangle() 4](#_Toc515496152)

[draw\_triangle() 5](#_Toc515496153)

[draw\_text() 6](#_Toc515496154)

[draw\_bitmap() 7](#_Toc515496155)

[clear\_screen() 8](#_Toc515496156)

[write() 9](#_Toc515496157)

[read() 10](#_Toc515496158)

[stop\_Read() 11](#_Toc515496159)

[init\_UART2() 12](#_Toc515496160)

[init\_IDLE\_Line() 13](#_Toc515496161)

[disable\_IDLE\_Line() 14](#_Toc515496162)

[delete\_UART() 15](#_Toc515496163)

[TIM3\_IRQHandler() 16](#_Toc515496164)

[USART2\_IRQHandler() 17](#_Toc515496165)

draw\_line()

**draw\_line**(**int** x1, **int** y1, **int** x2, **int** y2, **int** width, **int** color);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

draw\_line() wordt gebruikt om lijnen te tekenen op een VGA scherm.

**Argumenten**

**X1 x coördinaat van het beginpunt van de lijn.**

**Y1 y coördinaat van het beginpunt van de lijn.**

**X2 x coördinaat van het eindpunt van de lijn.**

**Y2 y coördinaat van het eindpunt van de lijn.**

**Width Dikte van de lijn in pixels.**

**Color Kleur van de lijn (1 t/m 255).**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**De x en y coördinaten van de punten moeten binnen het aantal beeldpunten liggen van het VGA scherm (320 breed, 240 hoog).**

**Voorbeeld**

draw\_line(20,20,,40,50,1,0xE0);

draw\_ellipse()

**draw\_ellipse**(**int** x\_mp, **int** y\_mp, **int** x\_rad, **int** y\_rad, **int** color, **int** fill);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

draw\_ellipse() wordt gebruikt om ellipsen te tekenen op een VGA scherm.

**Argumenten**

**x\_mp x coördinaat van het middelpunt van de ellips.**

**y\_mp y coördinaat van het middelpunt van de ellips.**

**x\_rad Breedte van de ellips.**

**y\_rad Hoogte van de ellips.**

**color Kleur van de ellips (1 t/m 255).**

**fill 1 om de ellips te vullen (optioneel).**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**De x en y coördinaten van de punten moeten binnen het aantal beeldpunten liggen van het VGA scherm (320 breed, 240 hoog).**

**Voorbeeld**

draw\_ellipse(100,150,,40,65,0xE0,0);

draw\_rectangle()

**draw\_rectangle**(**int** x\_lo, **int** y\_lo, **int** x\_rb, **int** y\_rb, **int** color, **int** fill);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

draw\_rectangle() wordt gebruikt om rechthoeken te tekenen op een VGA scherm.

**Argumenten**

**x\_lo x coördinaat van de linker onderhoek van de rechthoek.**

**y\_lo y coördinaat van de linker onderhoek van de rechthoek.**

**x\_rb x coördinaat van de rechter bovenhoek van de rechthoek.**

**y\_rb y coördinaat van de rechter bovenhoek van de rechthoek.**

**color Kleur van de rechthoek (1 t/m 255).**

**fill 1 om de rechthoek te vullen (optioneel).**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**De x en y coördinaten van de punten moeten binnen het aantal beeldpunten liggen van het VGA scherm (320 breed, 240 hoog).**

**Voorbeeld**

draw\_rectangle(50,150,,100,50,0xE0,0);

draw\_triangle()

**draw\_triangle**(**int** x1, **int** y1, **int** x2, **int** y2, **int** x3, **int** y3, **int** color, **int** fill);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

draw\_triangle() wordt gebruikt om driehoeken te tekenen op een VGA scherm.

**Argumenten**

**x1 x coördinaat van hoek 1 van de driehoek.**

**y1 y coördinaat van hoek 1 van de driehoek.**

**x2 x coördinaat van hoek 2 van de driehoek.**

**y2 y coördinaat van hoek 2 van de driehoek.**

**x3 x coördinaat van hoek 3 van de driehoek.**

**y3 y coördinaat van hoek 3 van de driehoek.**

**color Kleur van de driehoek (1 t/m 255).**

**fill 1 om de driehoek te vullen (optioneel).**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**De x en y coördinaten van de punten moeten binnen het aantal beeldpunten liggen van het VGA scherm (320 breed, 240 hoog).**

**Voorbeeld**

draw\_triangle(20,20,90,35,40,60,0xE0,0);

draw\_text()

**draw\_text**(**int** x, **int** y, **const** **char** \*str, **int** color, **const** **char**\* style, **int** fontNr);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

draw\_text() wordt gebruikt om tekst te schrijven op een VGA scherm.

**Argumenten**

**x x coördinaat vanaf waar de tekst wordt geschreven.**

**y y coördinaat vanaf waar de tekst wordt geschreven.**

**str In te voeren tekst.**

**color Kleur van de tekst (1 t/m 255).**

**style 3 soorten stijlen: “*norm*” 🡪 normale letters,**

**“*cursief*” 🡪 cursieve tekst,**

**“*vet*” 🡪 dikgedrukte tekst.**

**fontNr 2 soorten lettertypes: 1 voor font1 en 2 voor font2**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**De x en y coördinaten van de punten moeten binnen het aantal beeldpunten liggen van het VGA scherm (320 breed, 240 hoog).**

**Voorbeeld**

draw\_text(30,100,“the quick brown fox jumps over the lazy dog”,0xE0,”cursief”,1);

draw\_bitmap()

**draw\_bitmap**(**int** nr, **int** x\_lo, **int** y\_lo);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

draw\_bitmap() wordt gebruikt om bitmaps te tekenen op een VGA scherm. De bitmaps zijn elk 32 pixels hoog en 32 pixels breed.

**Argumenten**

**nr Het nummer van de bitmap. Er zijn 6 verschillende bitmaps beschikbaar:**

**0 🡪 blije smiley**

**1 🡪 boze smiley**

**2 🡪 pijl naar boven wijzend**

**3 🡪pijl naar beneden wijzend**

**4 🡪pijl naar rechts wijzend**

**5 🡪pijl naar links wijzend**

**x\_lo x coördinaat van de linker onderhoek van de bitmap.**

**y\_lo y coördinaat van de linker onderhoek van de bitmap.**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**De x en y coördinaten van de punten moeten binnen het aantal beeldpunten liggen van het VGA scherm (320 breed, 240 hoog).**

**Voorbeeld**

draw\_bitmap(2,150,100);

clear\_screen()

**clear\_screen**(**int** color);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Vgascreen.cpp | task | init\_VGA(); |

clearscreen() wordt gebruikt om het beeld leeg te maken en op te vullen met een kleur naar keuze.

**Argumenten**

**color De kleur die het beeldscherm moet aannemen.**

**Return waardes**

**Returns 0 bij succesvol uitvoeren van de functie.**

**Opmerkingen/Waarschuwing**

**N/A**

**Voorbeeld**

clear\_screen(0xE0);

write()

**write**(**char** \*text\_out);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

write() wordt gebruikt tekst te schrijven naar de UART2. De functie print een string per karakter met de put\_Char() functie.

**Argumenten**

**tekst\_out De tekst die geprint moet worden.**

**Return waardes**

**Void**

**Opmerkingen/Waarschuwing**

**N/A**

read()

**read**(**char** \*buf);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

read() wordt gebruikt om gebruikersinput te lezen van de USART2.

**Argumenten**

**buf Een buffer met de gebruikersinput.**

**Return waardes**

**Int error**

**Opmerkingen/Waarschuwing**

**N/A**

stop\_Read()

**stop\_read**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

read() stopt de read functie wanneer er geen userinput wordt gegeven.

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**

init\_UART2()

**init\_UART2**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

init\_UART2() initieert de UART componenten. Stelt de UART\_T struct in en initieert de RCC,GPIO,USART en de NVIC

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**

init\_IDLE\_Line()

**init\_LDLE\_Line**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

init\_LDLE\_Line() initieert de IDLE line detection voor UART2. Setup van TIM3 en enables interrupts.

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**

disable\_IDLE\_Line()

**disable\_LDLE\_Line**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

disable\_LDLE\_Line() disabled de IDLE line detection voor UART2. Disables TIM3 en TIM3 interrupts.

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**

delete\_UART()

**delete\_UART**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

delete\_UART() stopt UART en TIM3.

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**

TIM3\_IRQHandler()

**TIM3\_IRQHandler**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

TIM3\_IRQHandler() roept de UART::stop\_Read() functie om door te gaan met het uitvoeren van de buffer.

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**

USART2\_IRQHandler()

**USART2\_IRQhandler**(void);

|  |  |  |
| --- | --- | --- |
| ***File*** | ***Called from*** | ***Code enabled by*** |
| Uart.cpp | task | init\_UART2(); |

USART2\_IRQHandler() vult de input buffer en zet de LL::logic\_t.bReady flag waneer deze klaar is.

**Argumenten**

**N/A**

**Return waardes**

**void**

**Opmerkingen/Waarschuwing**

**N/A**