Contents

[1. Orientation-oop-basics: 2](#_Toc124686043)

[2. Orientation-on-class-diagrams: 2](#_Toc124686044)

[3. Orientation-unit-testing: 2](#_Toc124686045)

[4. Orientation-inheritance: 2](#_Toc124686046)

[5. Orientation-abstract-classes-and-interfaces: 3](#_Toc124686047)

[6. Exceptions: 3](#_Toc124686048)

[7. File-io: 3](#_Toc124686049)

[8. Student-administration-recap: (TODO) 3](#_Toc124686050)

[9. Bike-computer: 3](#_Toc124686051)

[10. FIFO-buffers: 4](#_Toc124686052)

[11. Array-basics: 4](#_Toc124686053)

[12. Dispatcher: 4](#_Toc124686054)

[13. String-behaviour: 4](#_Toc124686055)

[14. String-calculator: 4](#_Toc124686056)

[15. Structs: 5](#_Toc124686057)

[16. Connectfour: (TODO) 5](#_Toc124686058)

[17. Ringbuffer: 6](#_Toc124686059)

[18. Studentadmin: (TODO) 6](#_Toc124686060)

[19. Buffer-unit-tests: 7](#_Toc124686061)

[20. Memory: 8](#_Toc124686062)

[21. Pointers-intro: 8](#_Toc124686063)

[22. Bit-manipulations: 8](#_Toc124686064)

[23. Watch: 8](#_Toc124686065)

[24. Adidas: 8](#_Toc124686066)

[25. Elektrische-basis: 9](#_Toc124686067)

[26. Serialization: 9](#_Toc124686068)

[27. I2C: 9](#_Toc124686069)

[28. Alarm-states: (TODO) 9](#_Toc124686070)

[29. Light-controller: 9](#_Toc124686071)

[30. Trafic-light: 9](#_Toc124686072)

[31. Analog: 9](#_Toc124686073)

[32. Hysteresis: 10](#_Toc124686074)

[33. Register-IO: 10](#_Toc124686075)

[34. Other projects: 10](#_Toc124686076)

[34.1. ASM and Bootloaders: 10](#_Toc124686077)

1. Orientation-oop-basics: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/lib/compile/src/hash.hpp>

../SEM\_1/BankingCS

# Orientation-on-class-diagrams:

A picture containing text

Description automatically generated

Class diagram van traffic control programma van SEM1

1. Orientation-unit-testing: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/auto_test.py>
2. Orientation-inheritance: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/lib/compile/src/hash.hpp>
3. Orientation-abstract-classes-and-interfaces: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/lib/compile/src/hash.hpp>
4. Exceptions: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/core.py>
5. File-io: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/core.py>

# Student-administration-recap: (TODO)

1. Bike-computer: <https://github.com/MarijnVerschuren/STM32F4_AS5600_library/blob/main/AS5600>

1. FIFO-buffers: <https://github.com/MarijnVerschuren/Robotic_Arm/blob/main/ARM_COMPUTER/Core/Inc/uart_buffer.h>

*the uart is configured with a dma which has FIFO enabled (dit is geregeld in hardware dus ik heb hier zelf geen code voor (ik kan dit wel uitleggen)).*

# Array-basics:

../ SEM\_1/TraficControlCS/Trafic\_Light.cs@20

# Dispatcher:

Dispatching through interrupts:

<https://github.com/MarijnVerschuren/Robotic_Arm/tree/main/STM32_MCU/stepper>

1. String-behaviour: <https://github.com/MarijnVerschuren/Password_Manager/blob/main/app/lib/compile/src/hash.hpp>

Hier worden strings (of buffers + sizes) gebruikt om data in een hashing functie te krijgen. De lengte van een string hoeft hierbij niet gegeven te worden omdat elke string eindigd met een 0x00.

# String-calculator:

|  |
| --- |
| const char\* str = “text with unknown length…”;  uint32\_t str\_len = 0;  while (\*str) { str\_len++; str++; }; |

Code voor het berekenen van de lengte van de string.

1. Structs: <https://github.com/MarijnVerschuren/STM32F4_AS5600_library/blob/main/AS5600/inc/AS5600.h> (@95)

Struct die de configuratie en i2c handle van een bepaalde sensor bijhoud

<https://github.com/MarijnVerschuren/Robotic_Arm/blob/main/ARM_COMPUTER/Core/Inc/main.h>

Text

Description automatically generated

Struct van het bericht dat naar de ‘kleine stm32’ (STM32F411CEU6) word gestuurd om een beweging uit te voeren via de stepper motor

# Connectfour: (TODO)

# Ringbuffer:

Circular linked list: Chart, waterfall chart

Description automatically generated

Hiebij is ‘HEAD’ een pointer naar het eerste item en alle ‘NEXT’ pointers naar het volgende item. Soms word er ook een ‘PREV’ pointer toegevoegd waardoor je makkelijk heen en weer kan stappen in de list (doubly circular linked list):

Diagram

Description automatically generated

# Studentadmin: (TODO)

# Buffer-unit-tests:

Pointer casting:

Text

Description automatically generated

Output:

Text

Description automatically generated

# Memory:

../SEM\_1/ArduinoTester/ArduinoTester.ino@23

De ‘\_SFR\_IO8’ is een functie die een pointer terug geeft ten opzichten van een ‘BASE’ pointer (\_\_SFR\_OFFSET) waar de registers van de gpio pheripheral geindexeerd zijn.

Dan bevind elk ‘PIN’ (digital input register) register op: ‘BASE’ + 3 \* n

Dan bevind elk ‘DDR’ (data direction register) (0=in/1=out) register op: ‘BASE’ + 3 \* n + 1

Dan bevind elk ‘PORT’ (digital output register) register op: ‘BASE’ + 3 \* n + 2

Waarbij ‘n’ de port index is bij mijn arduino mega 2560 van 0 tot 6 (A tot G)

# Pointers-intro:

Zie Memory:

# Bit-manipulations:

Zie Memory:

# Watch:

Bij de stm kan je de ‘CNT’ of wel count register van een timer direct aflezen zie: <https://github.com/MarijnVerschuren/Robotic_Arm/blob/main/STM32_MCU/stepper/Core/Src/main.c> @53

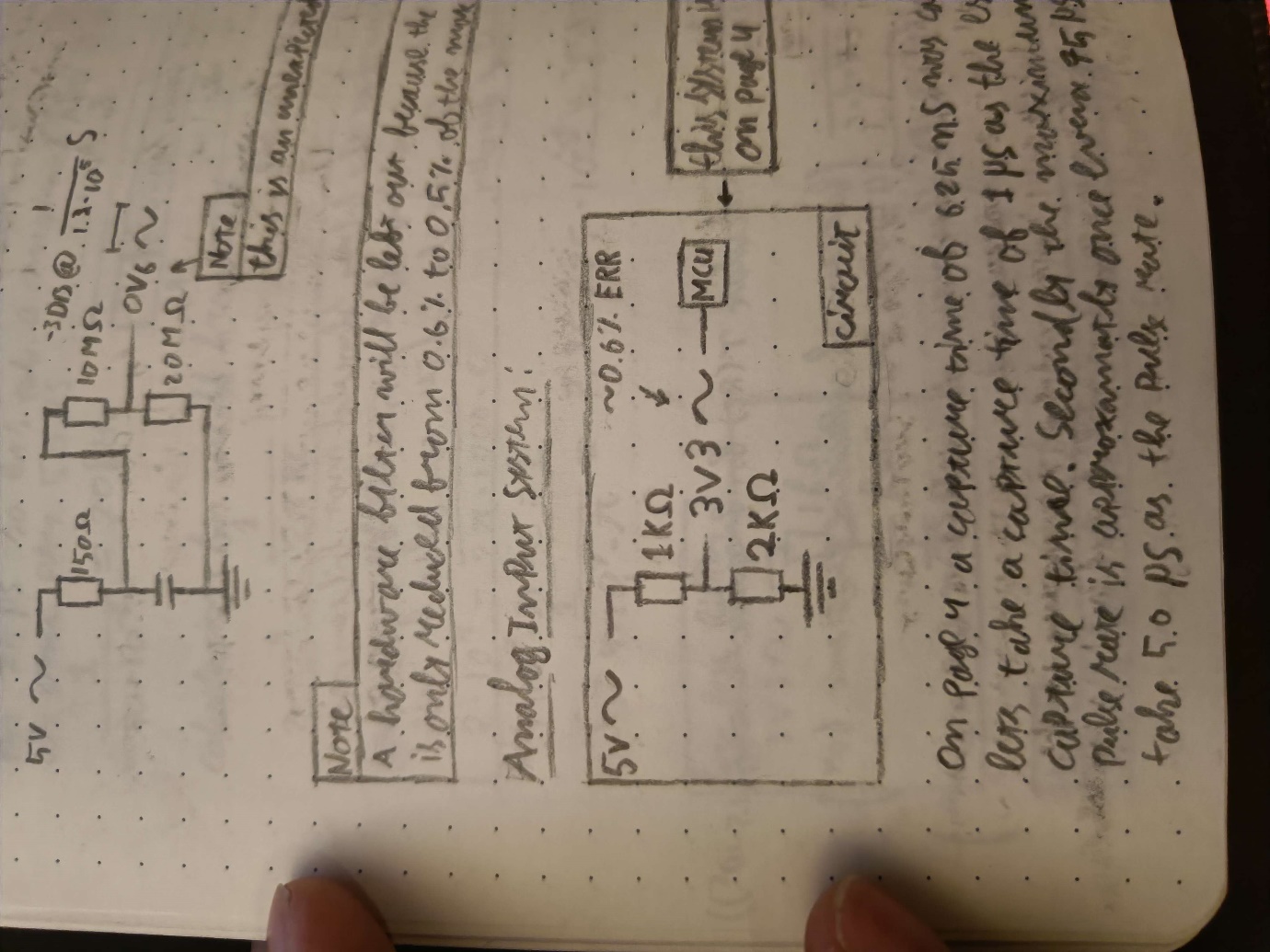
Waar ik wacht tot de count register op ‘n’ staat deze timer is zo geconfigureerd dat deze count eens per micro seconden geincrementeerd word hierbij zou ik ook bitmanipulatie gebruiken om bijvoorbeeld te wachten tot het derde bit gezet word, Dit zou voor een periode van 4us zorgen.

# Adidas:

Een andere vorm van data correction is een CRC checksum dit word bijvoorbeeld gebruikt in mijn robotic arm bij bijna elke data transfer zie: [https://github.com/MarijnVerschuren/Robotic\_Arm/blob/main/STM32\_MCU/stepper/Core/Src/main.c @133](https://github.com/MarijnVerschuren/Robotic_Arm/blob/main/STM32_MCU/stepper/Core/Src/main.c%20@133)

Waar dit gebruikt word om te verifiëren of de data correct is aangekomen.

# Elektrische-basis:



Dit is de voltage devider die ik gebruik om het 5v analoge signaal naar 3.3v om te zetten zodat de stm32 ze kan aflezen (de weerstanden zijn zo laag omdat dat alles was wat ik had).

1. Serialization: <https://github.com/MarijnVerschuren/Info_Generator/blob/main/info_gen/gen.py>
2. I2C: <https://github.com/MarijnVerschuren/STM32F4_AS5600_library/blob/main/AS5600/src/AS5600.c>

# Alarm-states: (TODO)

../SEM\_1/TraficControl (\_CS)

# Light-controller:

../SEM\_1/TraficControl (\_CS)

# Trafic-light:

../SEM\_1/TraficControl (\_CS)

1. Analog: <https://github.com/MarijnVerschuren/Robotic_Arm/blob/main/STM32_MCU/stepper/Core/Src/main.c>
2. Hysteresis: <https://github.com/MarijnVerschuren/Robotic_Arm/blob/main/STM32_MCU/stepper/Core/Src/main.c>

# Register-IO:

../SEM\_1/ArduinoTester/ArduinoTester.ino@23

Pointer offsets van registers zie Memory: voor uitleg

# Other projects:

## 34.1. ASM and Bootloaders:

<https://github.com/MarijnVerschuren/MOS>