Vragen opdrachtgever C++ unit library

Opdrachtgever: Sander Smeenk

Huidige opdrachtbeschrijving

C++ stage opdracht

* selecteren en inrichting van een C++ units library om units als voltage, ohm, temperatuur etc type-safe uit te kunnen drukken
  + eisen: geen heap, compile time checking, ondersteuning C++11 en C++14, toe kunnen voegen eigen units
  + hint: <https://github.com/martinmoene/PhysUnits-CT-Cpp11> ziet er veel belovend uit
* aanpassen csdd examples voor esp32 & nordic nrf gebaseerd op stm32
* uitbreiden isl-cmake voor sanitizers (memory, stack, thread, undefined behavior, etc)
* analyseren highest common denominator voor C++ versie voor stm32, esp32, nrf

1. Wat is het doel van dit project? Een proof of concept, een stap in de goede richting, een library die voortaan gebruikt kan worden, een onderzoek naar welke library?  
   Onderzoeksdoel is welke library we willen gebruiken. Hierbovenop zou deze nog geïntegreerd moeten worden in de STM32 example. STM32 is minimum, daarna ESP32 en daarna Nordic NRF.  
   Daarna analyseren highest common demoninator, dus:  
   Wat is de hoogste versie van C++ die ze allemaal ondersteunen?  
   Daarna uitbreiden van de sanitizers.
2. Welke criteria worden gesteld aan de library?
   * Welke eenheden/conversies moeten gebruikt worden, moet dit onderzocht worden welke veel gebruikt worden binnen Inspiro?
   * Static memory allocation en geen dynamic, dus geen heap?  
     Geen heap, alleen stack.
   * C++ 17 ook of alleen 11 en 14, of nog een andere?  
     Libraries niet nieuwer dan C++ 11, C++ 14 zou ook kunnen, maar moet gecheckt worden of dit dan werkt. C++ 17 kan ook, als dit kan.
   * Cmake versie, of wordt ISL-cmake gebruikt?
   * Moet werken met Clang-tidy?
   * Documentatie van de library, is hier een eis voor? Indien nodig zou ik bijvoorbeeld dit extra kunnen toevoegen.
   * Welke types moeten er gebruikt worden? Floats, doubles, int64\_t’s? Allemaal hebben ze voor- en nadelen, of mag de library dit kiezen?
3. Documentatie, is er behoefte om mijn proces te documenteren zodat dit reproduceerbaar is voor een volgende keer? Of bijvoorbeeld alleen het testen.  
   Er zijn plugins van Inspiro voor controle van stackgebruik.
4. Moet het voldoen aan specifieke kwaliteitseisen van Inspiro? Denk aan Quality Management System templates.
   * Documenten standaard?
   * Code standaard
5. Is er een limitatie aan de stackgrootte die er gebruikt mag worden? Of liefst zo min mogelijk?
6. Indien zo min mogelijk, moet er onderzocht worden wat dan de meest efficiënte manier is van dit declareren of met welke manier dit het beste getest/ gevisualiseerd kan worden.

**Notulen:**

Distance type, bevat alle informatie om een interpretatie te doen van wat is het nou.  
Zodat je er ook op een veilige manier mee kan rekenen.  
Distance A = 2m;  
Distance B = 2yard;

Distance total = A + B;  
Uint16\_t total\_miles = total.ToMiles();

Voltage X = 3V;  
Resistance Y = X \* 0.3;  
uint16\_t resistance = y.ToOhm();

Libraries zijn veel templates, misschien moet daar nog extra onderzoek naar gedaan worden.

Requirements:  
Geen heap.

Libraries niet nieuwer dan C++ 11, C++ 14 zou ook kunnen, maar moet gecheckt worden of dit dan werkt. C++ 17 kan ook, als dit ondersteunt wordt.

Niet bestaande units moeten handmatig door Inspiro toegevoegd kunnen worden.

Compile time checking voor units. Mismatch unit moet hiermee afgehandeld worden.