Erweiterung der PWM Auflösung

- Problem:
  - Der ausgewählte Controller hat nur zwei 16 Bit hardware PWM Kanäle, die restlichen vier hardware PWM Kanäle haben nur eine Auflösung von 8 Bit.
  - Für die RGB LED werden drei Kanäle benötigt.
  - Aufgrund der nichtlinearen Helligkeitsempfindlichkeit des Auges wird im kleinen Helligkeitsbereich eine höhere Auflösung der PWM von mindestens 12 Bit gefordert.
- Lösung 1: Dithering (pwm_dither.h)
  - Mithilfe von Dithering kann durch eine vergrößerte Zeitauflösung auch die Auflösung der Helligkeitsstufen vergrößert werden. Dabei hilft die Trägheit des Auges, auf welcher das gesamte Dimmprinzip per PWM basiert.
  - Anstatt einer PWM mit z.B. 200Hz wird jetzt eine PWM mit 400Hz genutzt. Dabei wird immer in einer Periode das nächst mögliche geringere Tastverhältnis als der gesetzte Wert verwendet. In der nächsten Periode wird das nächsthöhere Tastverhältnis gesetzt. Im Mittel ergibt sich also genau die Helligkeit wischen beiden Tastverhältnissen. Die PWM Auflösung wurde so um ein Bit erweitert. Die Auflösung kann also mit jeder Frequenzverdopplung um ein Bit erhöht werden.
  - Bei 12 Bit wird also eine Frequenz von 200Hz * 2^4 = 3,2kHz benötigt
  - Vorteile:
    - Benötigte bittiefe kann dynamisch an den eingestellten Tastgrad angepasst werden, wodurch auch die Frequenz an den jeweiligen Tastgrad angepasst wird und die benötigten CPU Zyklen geringer werden.
  - Nachteile:
    - Hohe PWM Frequenz stellt hohe Anforderungen an die Flankensteilheit des LED Treibers, sonst wird die Helligkeit insgesamt abgesenkt. Bei geringer Flankensteilheit gibt es zusätzlich Helligkeitssprünge beim Umschalten zwischen verschiedenen Frequenzen.
- Lösung 2: Hybrid zwischen Hardware und Software PWM (pwm_hybrid.h)
  - Die vorhandene 8 Bit PWM wird durch einen weiteren 8 Bit Softwarezähler erweitert, welcher immer beim Überlauf des Hardwarezählers hochgezählt wird. Dadurch verhält er sich exakt wie ein 16 Bit Hardware Kanal, benötigt aber Zusätzliche CPU Zyklen alle 256 Takte.
  - Beim Einstellen des Tastverhältnisses wird geprüft, bei welchem Zählerstand des Softwarezählers der Ausgang von Low auf High springen muss (software compare, SC) und bei welchem Stand des Hardwarezählers dieser Sprung stattfindet (hardware compare, HC).
    In jedem Überlauf des Hardwarezählers werden die Werte für die nächste Periode (nicht die direkt folgende) gesetzt. Hier wird nun unterschieden, liegt der Softwarezähler unter SC, wird der Hardware Tastgrad auf 0xFF (Low Pegel) gesetzt. Liegt er über SC wird der Tastgrad auf null (High Pegel) gesetzt. Wenn er gleich SC ist, wird der Hardware Tastgrad auf HC gesetzt. Dadurch wird das Umschalten im korrekten Moment von der Hardware übernommen.

  - Vorteile:
    - PWM Frequenz wird gering gehalten
    - Verhält sich exakt wie ein 16 Bit Hardware Timer
  - Nachteile
    - Im mittel höhere CPU Last als Dithering, rund 77 Zyklen alle 256 Zyklen (rund 30% Grundlast). Dieser Wert verringert sich allerdings mit der PWM Frequenz bzw. dem Prescaler.
- Lösung 2 ist die Empfohlene

Systick

- Systick wird vom 16 Bit Timer gesteuert und ist damit abhängig von dessen Auflösung.
- Bei einer Auflösung von 14 Bit bei 16MHz entspricht ein Tick 1,024ms
  Die Interruptroutine benötigt 56 Zyklen alle 16385 Takte (0,34% Last)

Mögliche Änderungen

- In der aktuellen Ausführung haben die PWMs alle eine Auflösung von 14 Bit und einen Prescaler von eins, wodurch sich oben genannter Systick und eine PWM Frequenz von 977Hz (bei 16MHz) ergibt.
- Mit einem Prescaler von acht und mit 12 Bit Auflösung würde die PWM Frequenz auf 488Hz (bei 16MHz) verringert werden. Der Systick würde sich entsprechend verdoppeln. Die CPU Auslastung durch die Hybrid PWM verringert sich allerdings um Faktor acht mit 77 Zyklen alle 2048 Takte (3,8% Grundlast).
- Der Systick könnte zusätzlich im Überlauf des 8 Bit Timers berechnet werden. Dadurch würde ich die Genauigkeit des Systick erheblich verbessern, aber die CPU Auslastung ansteigen. Konkret ergäbe sich bei einem Prescaler von 1 ein Systick alle 16µs und eine Auslastung von zusätzlichen 20 Zyklen pro Überlauf des Timers, also 97 Zyklen alle 256 Takte bzw. 38%.
  Mit einem Prescaler von 8 ergäbe sich ein Systick alle 128µs und eine Auslastung von 97 Zyklen alle 2048 Takte bzw. 4,7%.
  Für diesen zweiten Fall wäre die Änderung zu empfehlen, damit die Genauigkeit des Systick, gerade für die Kommunikation nicht zu gering wird.