

CANopen Stack für Grossdieselmotoren

Kunde	Wärtsilä
Zielsysteme	Motorola MPC561 / MPC5200 (Mikrocontroller)
Technologien	Redundante CANopen Systeme
Programmiersprache	ANSI-C, Matlab-Simulink/Stateflow
Speziell	Sotronik hat einen kundenspezifischen und redundanten CANopen-Stack entwickelt, über den die Steuer-/Regelungseinheiten eines Grossdieselmotors kommunizieren.



12Zylinder Grossdieselmotor

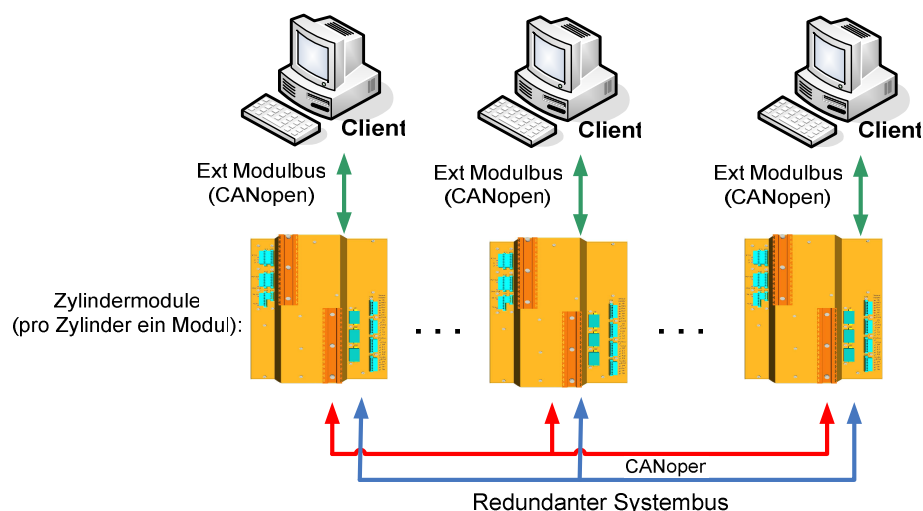
Einleitung

Die Steuerung von modernen Grossdieselmotoren erfolgt über verteilte Systeme, die untereinander über leistungsfähige Datenbusse kommunizieren. Motordaten wie z.B. Einspritzwinkel und Ventilsteuerung müssen zeitnah und redundant innerhalb des verteilten Systems übertragen werden, um die Motoren leistungs- und verbrauchsoptimiert betreiben zu können. Flexibilität bezüglich Systemerweiterbarkeit ist ebenso gefragt wie leistungsfähige Visualisierungs- und Monitoring-Tools.

Aufgabe

Sotronik erhielt den Auftrag, einen kundenspezifischen und redundanten CANopen Software Stack zu entwickeln, der in sämtlichen CANopen basierenden Bussystemen der Firma Wärtsilä eingesetzt werden kann.

Die WECS-Steuerung (Wärtsilä Engine Control System) stellte hierbei eine besondere Herausforderung dar. Diese Steuerung betreibt parallel zwei redundante CAN-Busse (Systembusse - 500kBaude) und einen separaten CAN-Bus für die Kommunikation zu externen Systemen (Modulbus - 125kBaude). Zusätzlich müssen die CAN-Busse untereinander fähig sein, grosse Datenmengen über so genannte Gateway-Funktionen auszutauschen.



Die Softwarearchitektur sollte ein hohes Maß an Flexibilität bieten, unterschiedliche Zielhardware unterstützen und bis zu drei unabhängige CAN-Buslinien gleichzeitig betreiben können. Auch die Portierung auf embedded-Linux Systeme sollte möglich sein.

Softwarearchitektur

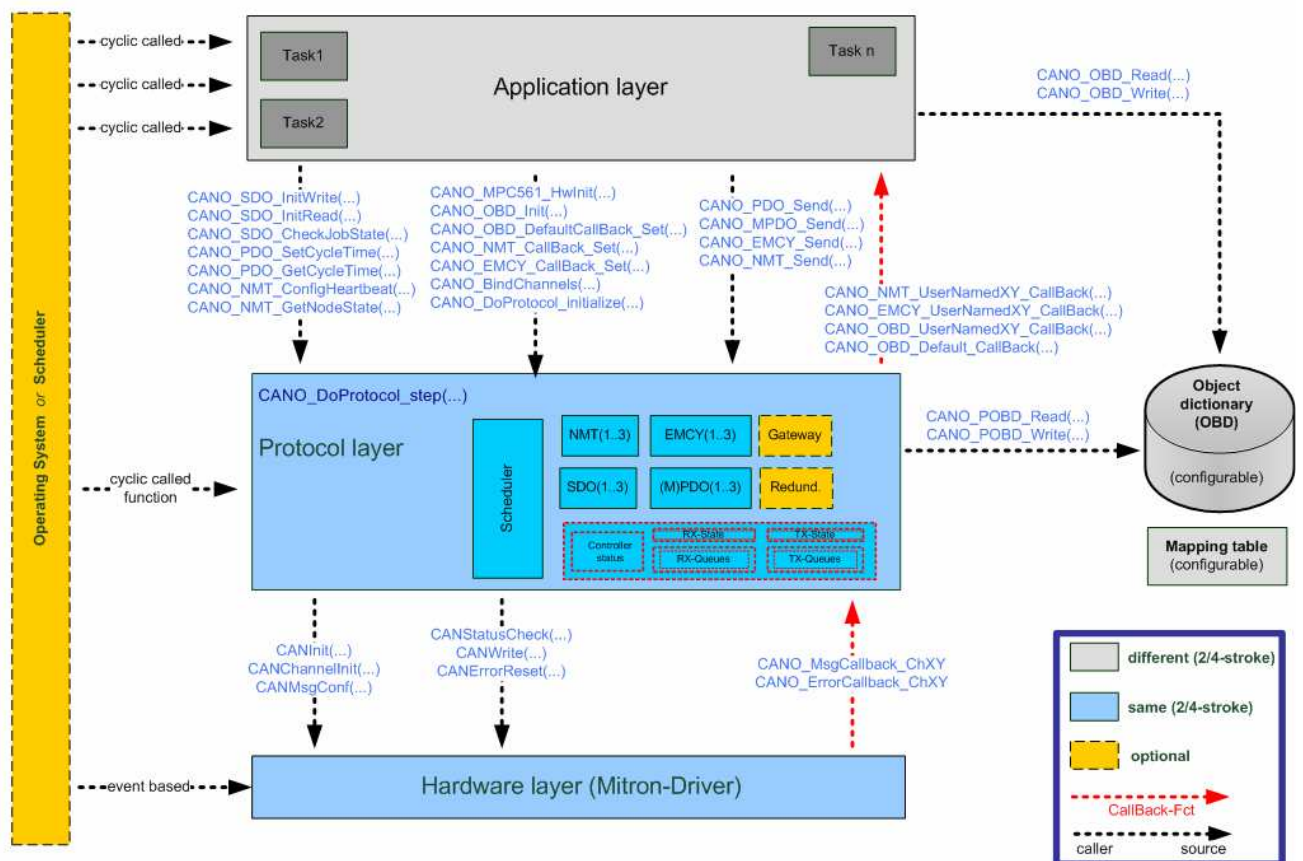
Der Aufbau der Software wurde flexibel gehalten und ist im Wesentlichen in die nachfolgenden drei Software-Layer aufgeteilt.

Protocol-Layer: Kern des CANopen-Stacks ist der Protocol-Layer der sämtliche CANopen-Datentransferabläufe realisiert. Diese wurden teilweise mit Matlab-Simulink/Stateflow realisiert, was ein erheblicher Vorteil beim Simulieren dieser Protokollabläufe brachte (zeitlich wie auch funktional). Das durch PC-Simulation getestete Zustandsmodell wurde nach der Fertigstellung automatisch in einen ANSI-konformen C-Code übersetzt und in die Stacksoftware integriert.

Hardware-Layer: Auf unterster Ebene sorgt der Hardware-Layer für die Anbindung des CANopen-Stack an die jeweils zur Verfügung stehende Zielhardware. Durch Anpassen dieses Layers ist die gesamte Software portierbar und somit für den Kunden flexibel einsetzbar.

Application-Layer: Der Application-Layer stellt die Schnittstellenfunktionen und Konfigurationsmöglichkeiten für die Applikation zur Verfügung. Der Datenaustausch zwischen Protocol- und Application-Layer erfolgt über ein zentrales "Object-Dictionary" (OBD), für das eigens ein leicht zu bedienendes Konfigurationswerkzeug erstellt wurde. CANopen-Nachrichten können so innerhalb kürzester Zeit konfiguriert und verwendet werden.

CANopen stack moduls and interface functions



Softwarearchitektur unterteilt in Hardware-, Protocol- und Application-Layer

Ergebnis

Der entwickelte CANopen-Stack wurde erfolgreich implementiert und wird zuverlässig auf den weltweit grössten Dieselmotoren betrieben. Der Stack erfüllt die vorgeschriebenen Anforderungen des CiA Draft Standard 301 (CAN in Automation e. V.), kommuniziert problemlos mit CANopen-konformen Endgeräten und kann auf unterschiedliche Zielsysteme portiert werden (Voraussetzung ANSI-C fähig).