new eagle

RAPTOR®
INNOVATION
SUMMIT 2025

Raptor & CODESYS 2025

Presented by:
Parker Mosman, Chief Product Officer, New Eagle
Chris Knaack, Senior Principal Engineer, New Eagle

Agenda

01 Introduction to CODESYS

02 Supported Platforms

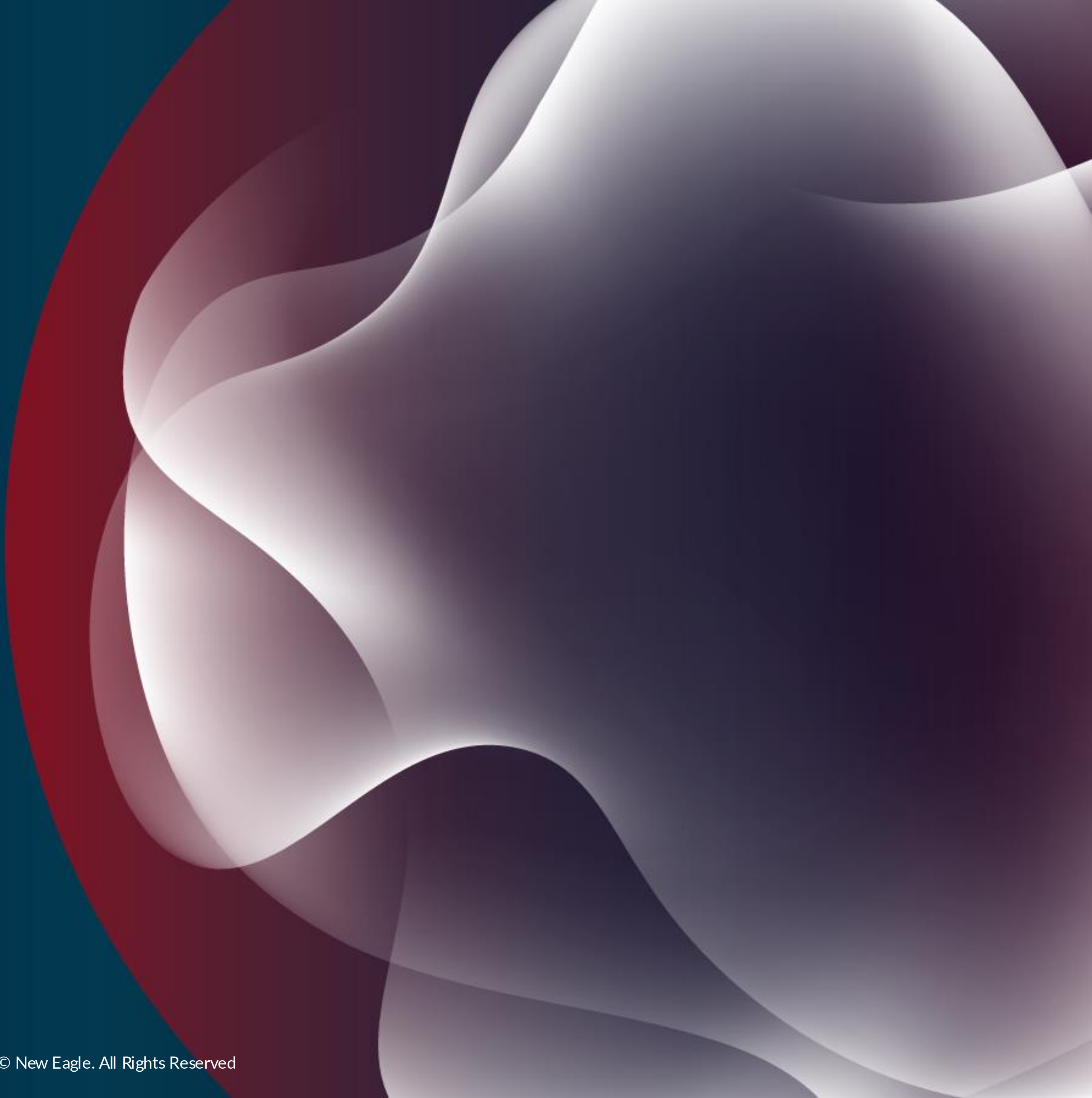
03 Raptor & CODESYS Application Example

04 Feature roadmap

05 Q&A

© New Eagle. All Rights Reserved

First, A Few Words…



© New Eagle. All Rights Reserved

Why Are We Doing This?

Common ECU hardware, with common platform software, differentiated by the IDE

New Eagle creates a standard platform software environment to support all our Infineon TC2xx and TC3xx based ECUs

- Common ETAS RTA-OS operating system
- Common device drivers for MCU peripherals and external ASICS
- Common base software for standard MCU-independent functions
- Common cybersecurity solutions, where relevant

Standard platform software enables relatively easy differentiation of the application development environment

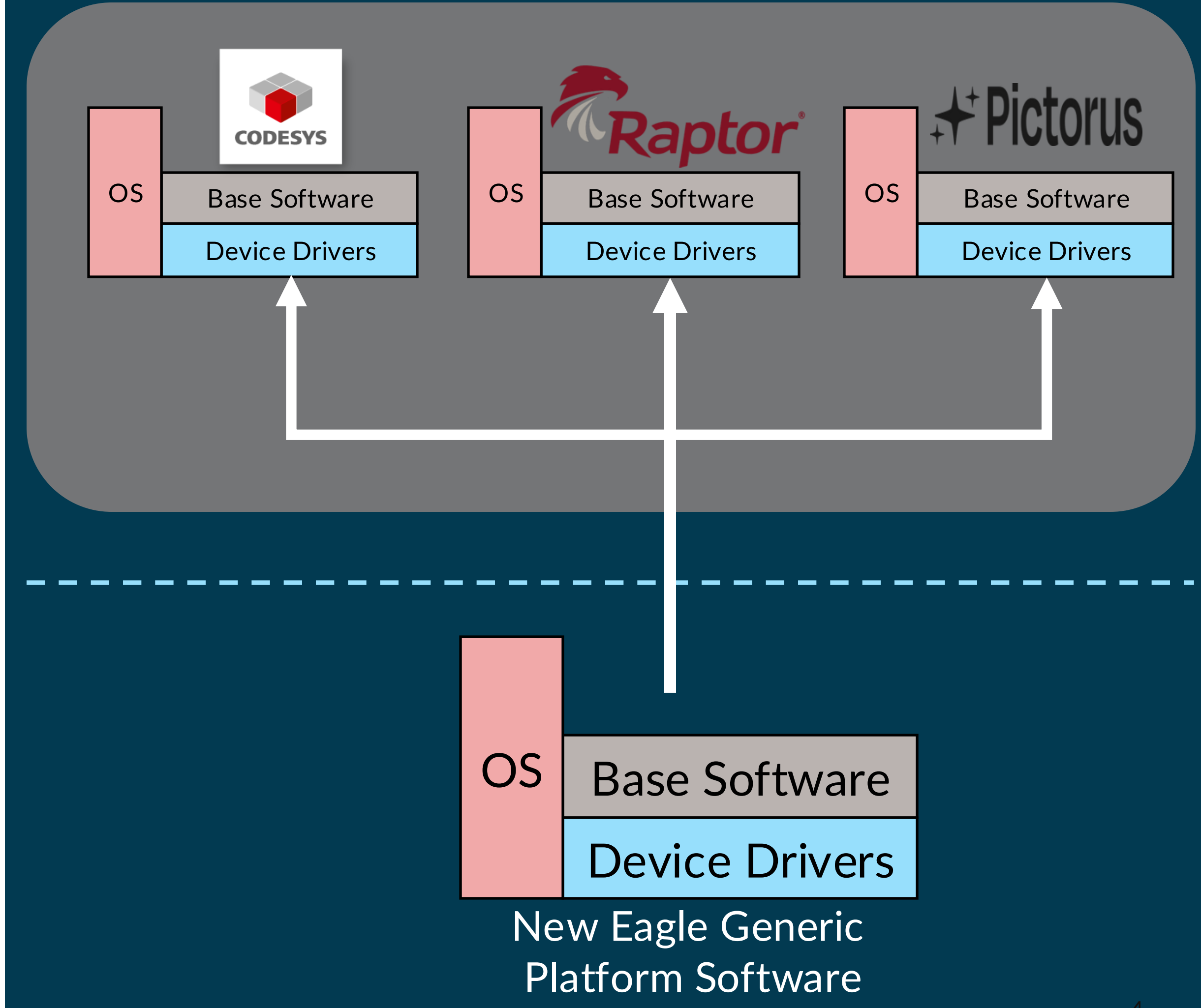
- Enables New Eagle to adapt to our customers, rather than our customers adapting to New Eagle
- Redeploy valuable IP on New Eagle hardware, regardless of the format it was captured in
- Commonality drives quality

INNOVATE FASTER. SCALE SMARTER

© New Eagle. All Rights Reserved

new eagle

RAPTOR® INNOVATION SUMMIT 2025



New Eagle Generic Platform Software

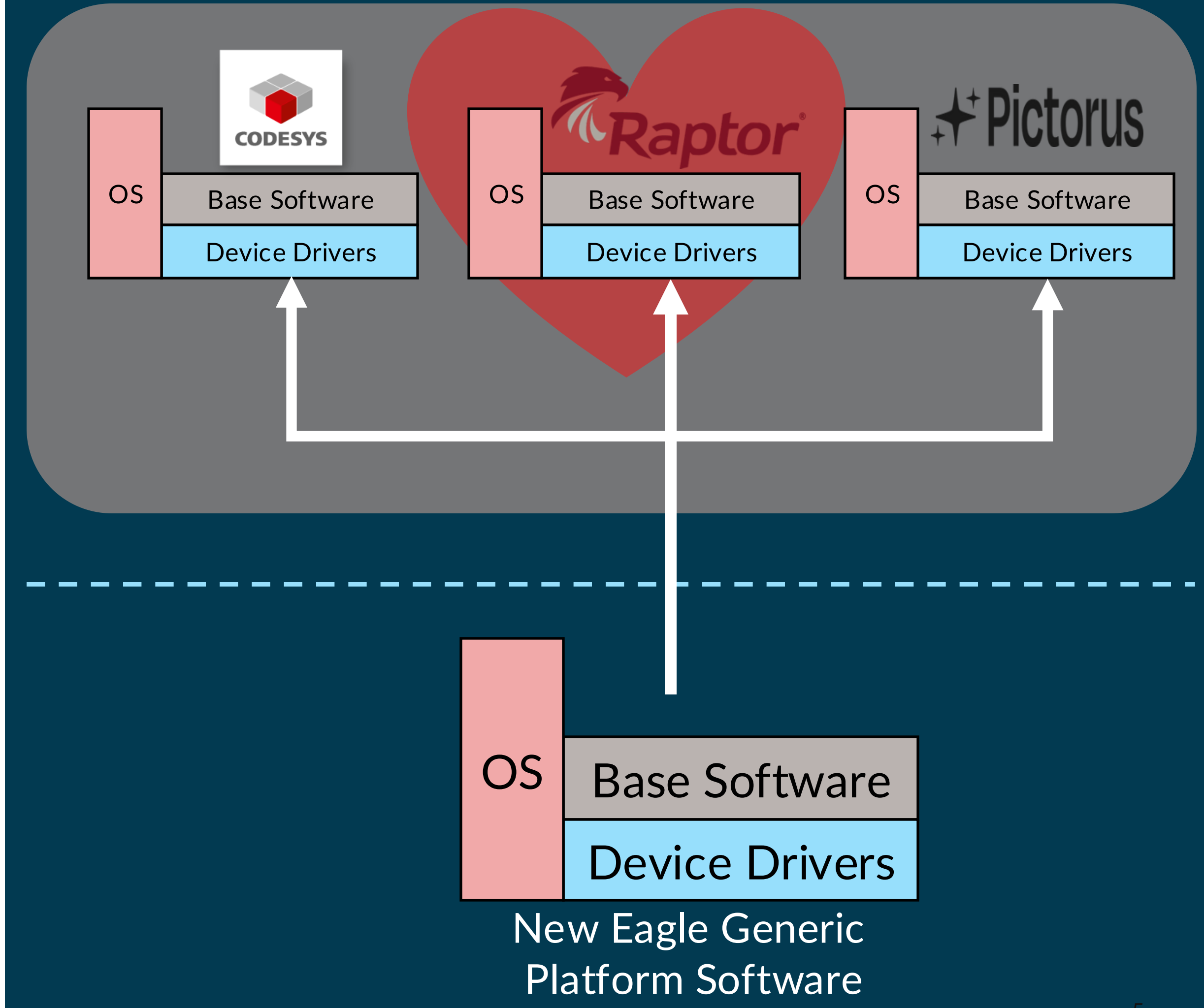
Raptor Isn't Going Anywhere!

Raptor continues to receive the love and attention it deserves

- ✓ Rollout of new safety tooling in Raptor Safe
- ✓ Major upgrades to the communication features, including DoIP, J1939-22, and J1939-91C
- ✓ An entire family of new ECUs
- ✓ Two major releases annually to stay in sync with Mathworks tooling
- ✓ **And, more to come**

new eagle

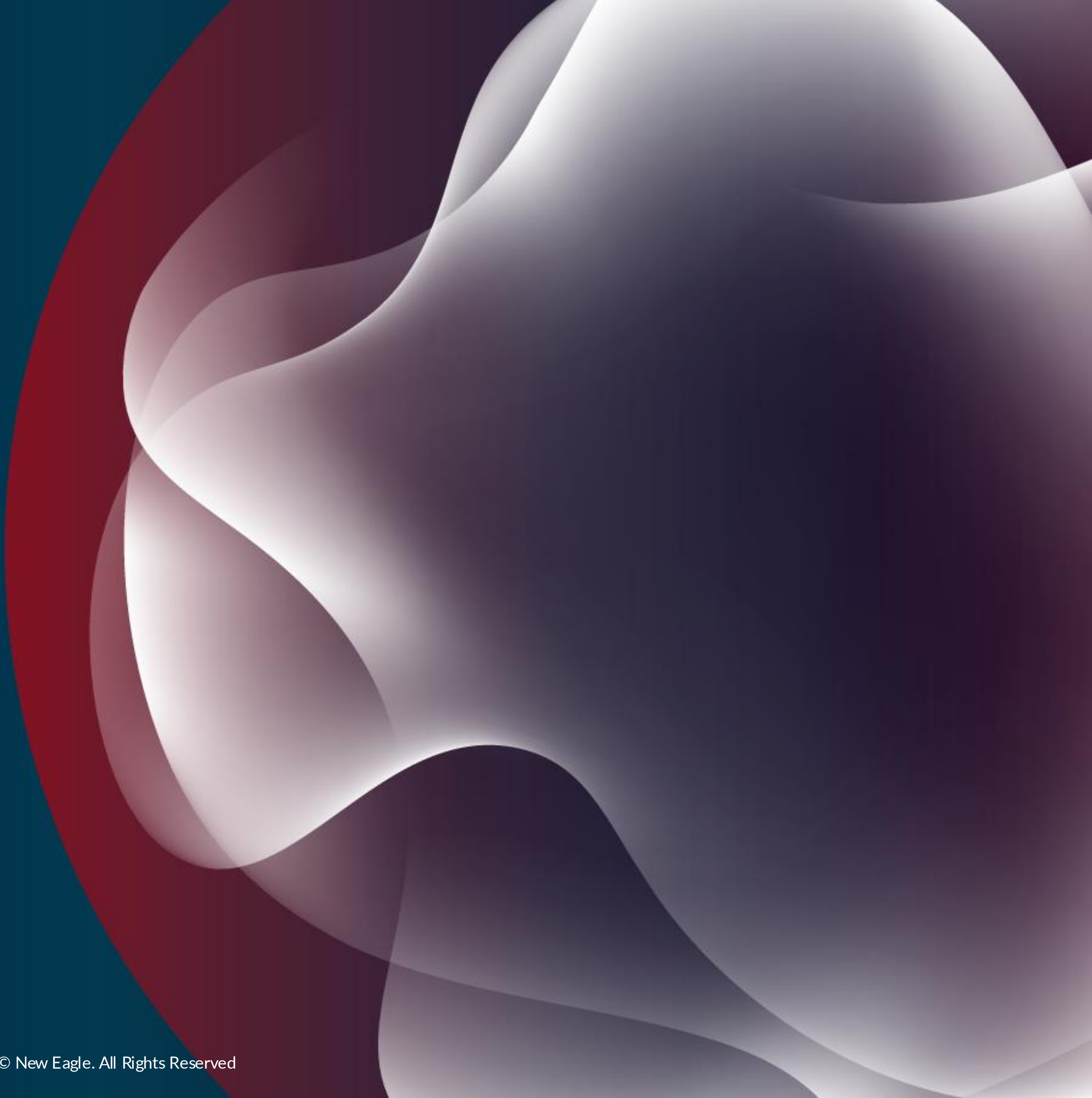
RAPTOR®
INNOVATION
SUMMIT 2025



INNOVATE FASTER. SCALE SMARTER

© New Eagle. All Rights Reserved

CODESYS Overview



© New Eagle. All Rights Reserved

What is CODESYS?

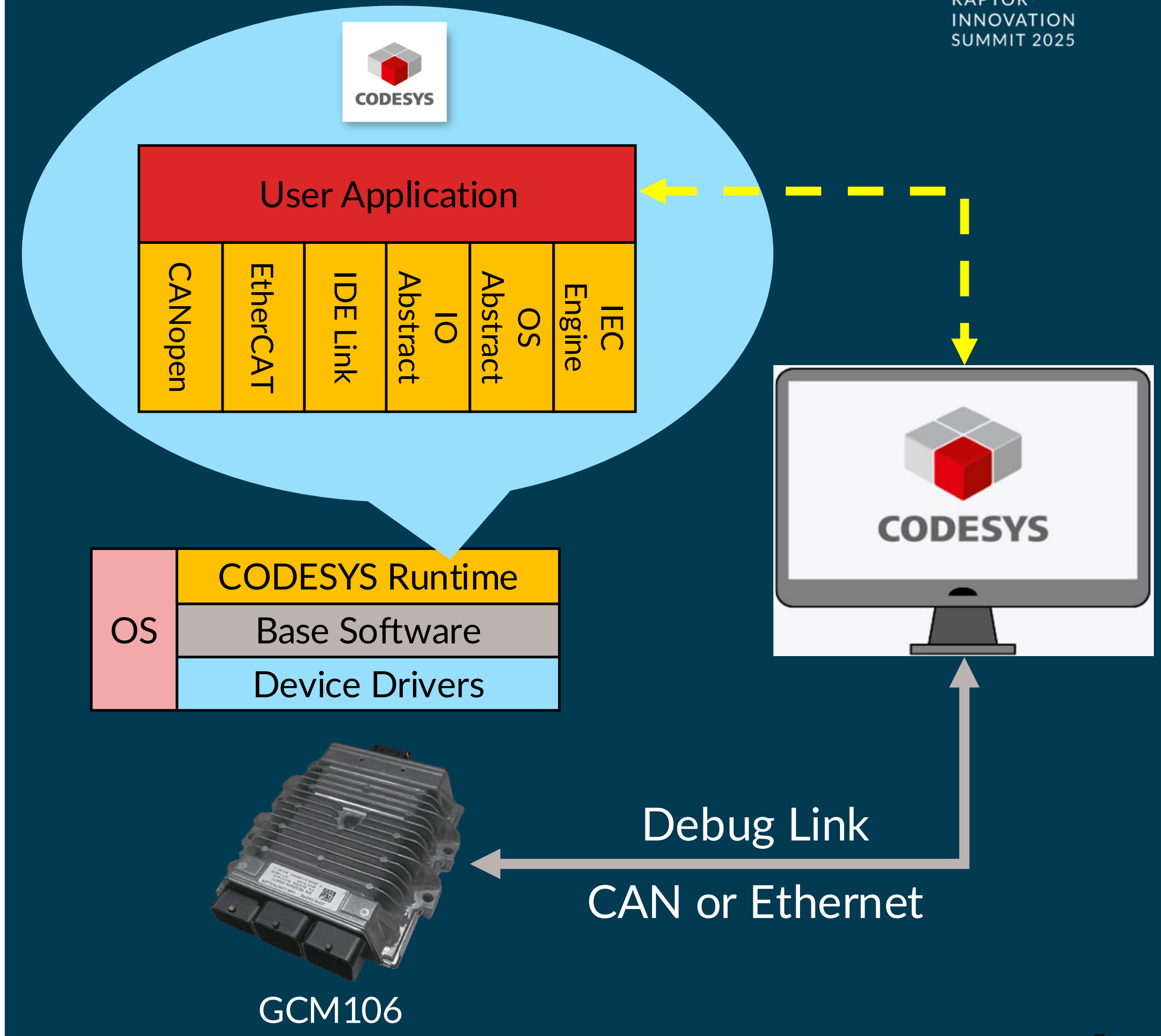
Enables hardware OEMs to bring PLC programming to their products

CODESYS is comprised of three main components

- A PC Based IDE that integrates development, build, calibration, and measurement capabilities
- A fixed run-time layer integrated inside the ECU along with the base software
- The user application developed in the CODESYS IDE and downloaded to the ECU directly from CODESYS

The CODESYS IDE has complete support for the following IEC 61131-3 languages:

- Ladder Diagram
- Function Block Diagram
- Structured Text
- Instruction List
- Sequential Function Chart



ECU Software Management

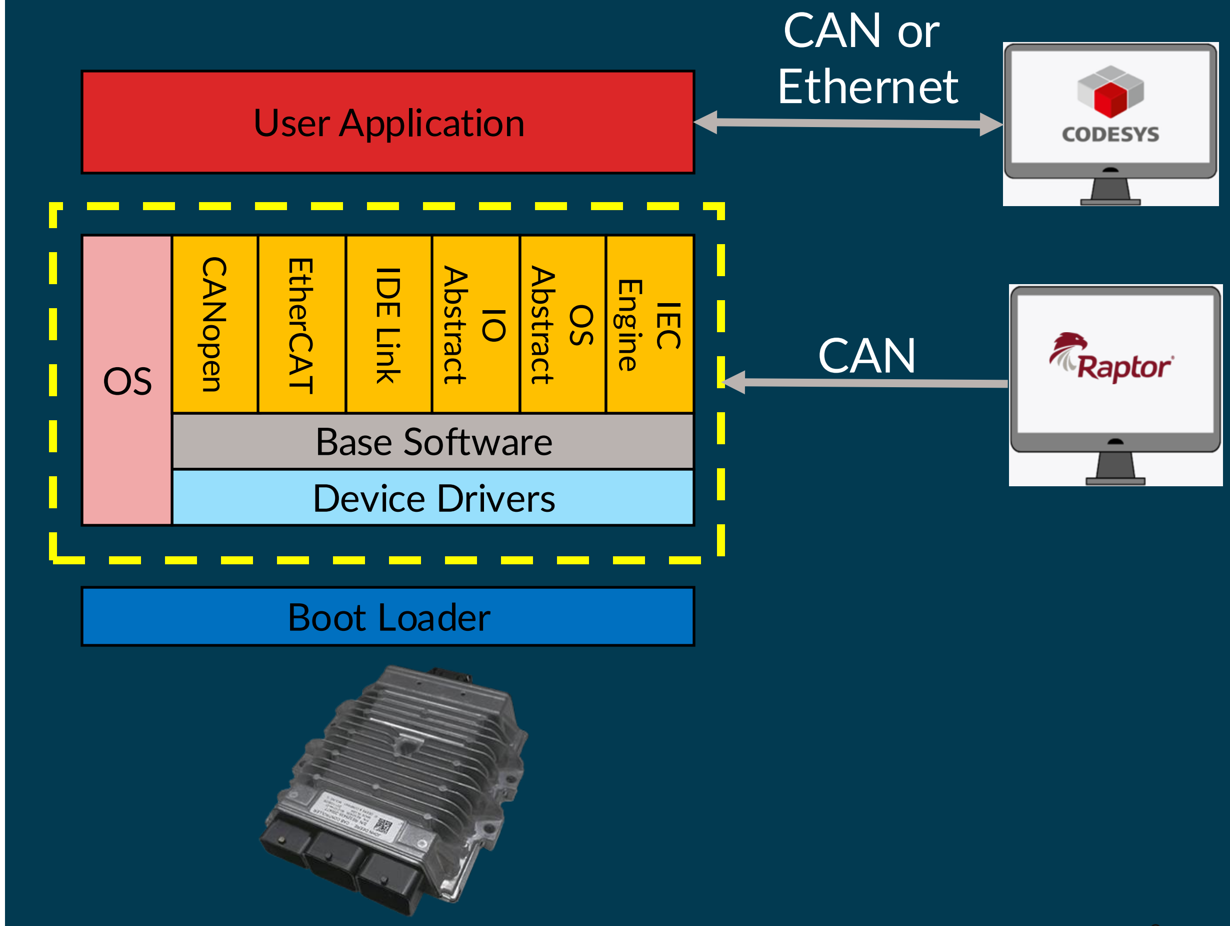
CODESYS application updates only touch the user application itself, not the ECU firmware

CODESYS IDE responsible for managing the user application only, but this is only a portion of the software on the ECU

- Integrates all required functionality to download the user application to the target
- Includes ability to measure variables, calibrate parameters, and do basic debugging of the user application
- Debug feature includes the ability to set break points
- IDE interacts with ECU via Ethernet or CAN, depending on the ECU variant

A separate New Eagle tool is required to update the runtime software the user application sits on top of

- Only necessary where New Eagle releases new features or bugfixes. Not a normal part of development process.
- The required tool is offered **free** by New Eagle
- Updates of the runtime must be performed via CAN



User Application

CAN or Ethernet

OS

CANopen

EtherCAT

IDE Link

IO Abstract

OS Abstract

IEC Engine

Base Software

Device Drivers

CAN

Boot Loader

GCM106

© New Eagle. All Rights Reserved

CODESYS Proliferation

CODESYS is a stable platform used widely across many sectors with a growing user base

CODESYS employs more than 200 employees with revenue in excess of 40 million Euros

- Global operation with offices in Germany, China, Italy, and the U.S.A.
- Enjoys a high degree of market diversity ranging from industrial automation to on highway vehicle applications
- Applicable to platforms ranging from high end industrial PCs to resource constrained ECUs

A massive portfolio of industry partners

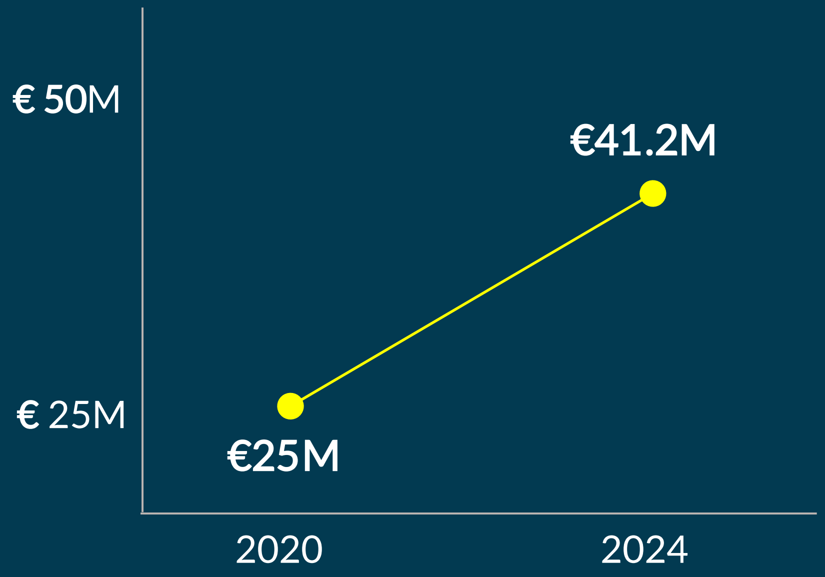
- Found on millions of machines globally
- 1,000+ unique device types presently on the market
- 500+ unique hardware OEMs offering CODESYS
- 10,000+ CODESYS developers globally

new eagle

RAPTOR®
INNOVATION
SUMMIT 2025

Total Employees	Total Devices	Total Partners
>230	>1000	>500

CODESYS Gross Revenue



INNOVATE FASTER. SCALE SMARTER

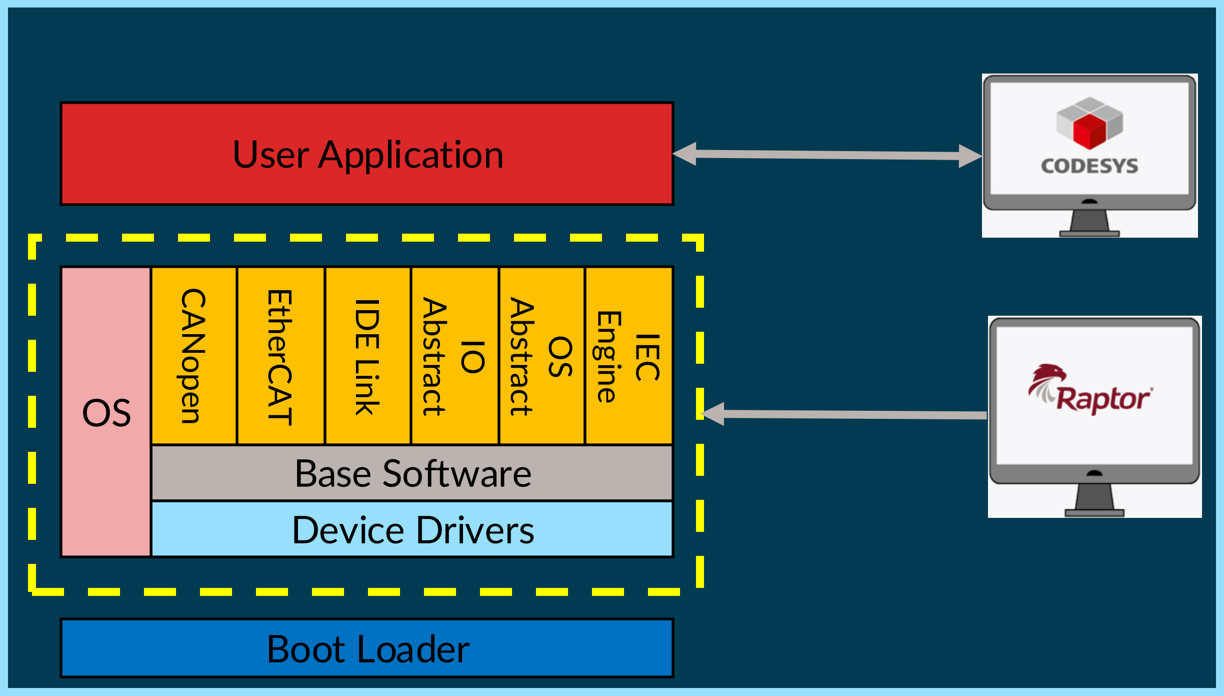
© New Eagle. All Rights Reserved

Commercial Model

License cost for standard functionality built into the sale price of the ECU

- License for CODESYS IDE and ECU runtime built into ECU sale price
- No compiler or other third-party tooling required
- All IDE features to develop, compile, download, measure, calibrate, test, and simulate included with base license
- No cost associated with New Eagle firmware update utility
- Fees may be required for CODESYS marketplace add-ons and communication interface cables

Everything Inside The Box Is Free!



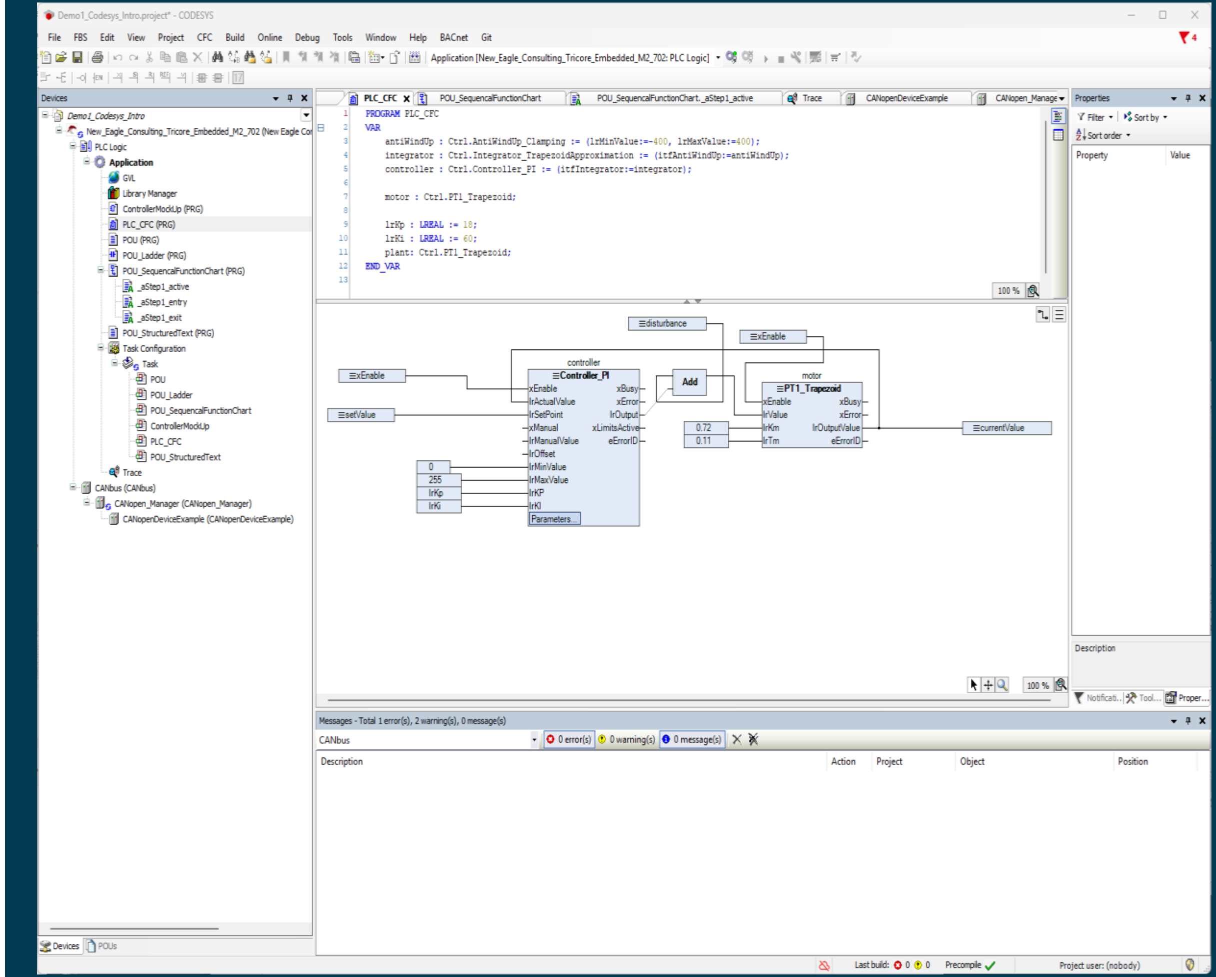
Runtime License Built Into ECU Price

© New Eagle. All Rights Reserved

Developer Experience

CODESYS IDE is a one stop shop for all functionality, including multi project support

- ❖ Device tree configures all projects from a single location, including third-party hardware
- ❖ Environment to support control routines and map them to configured operating system tasks
- ❖ Directly build applications with strict version control over libraries and the CODES compiler
- ❖ Deploy all applications directly to the target hardware without leaving the IDE
- ❖ Debug and monitor code directly from the IDE, as if you had an in-circuit debugger attached
- ❖ Test and simulate your code



```
PROGRAM PLC_CFC
VAR
    antiWindUp : Ctrl.AntiWindUp_Clamping := (lrMinValue:=-400, lrMaxValue:=400);
    integrator : Ctrl.Integrator_TrapezoidApproximation := (itfAntiWindUp:=antiWindUp);
    controller : Ctrl.Controller_PI := (itfIntegrator:=integrator);

    motor : Ctrl.PT1_Trapezoid;

    lrKp : LREAL := 18;
    lrKi : LREAL := 60;
    plant: Ctrl.PT1_Trapezoid;
END_VAR
```

Messages - Total 1 error(s), 2 warning(s), 0 message(s)

0 error(s) 0 warning(s) 0 message(s)

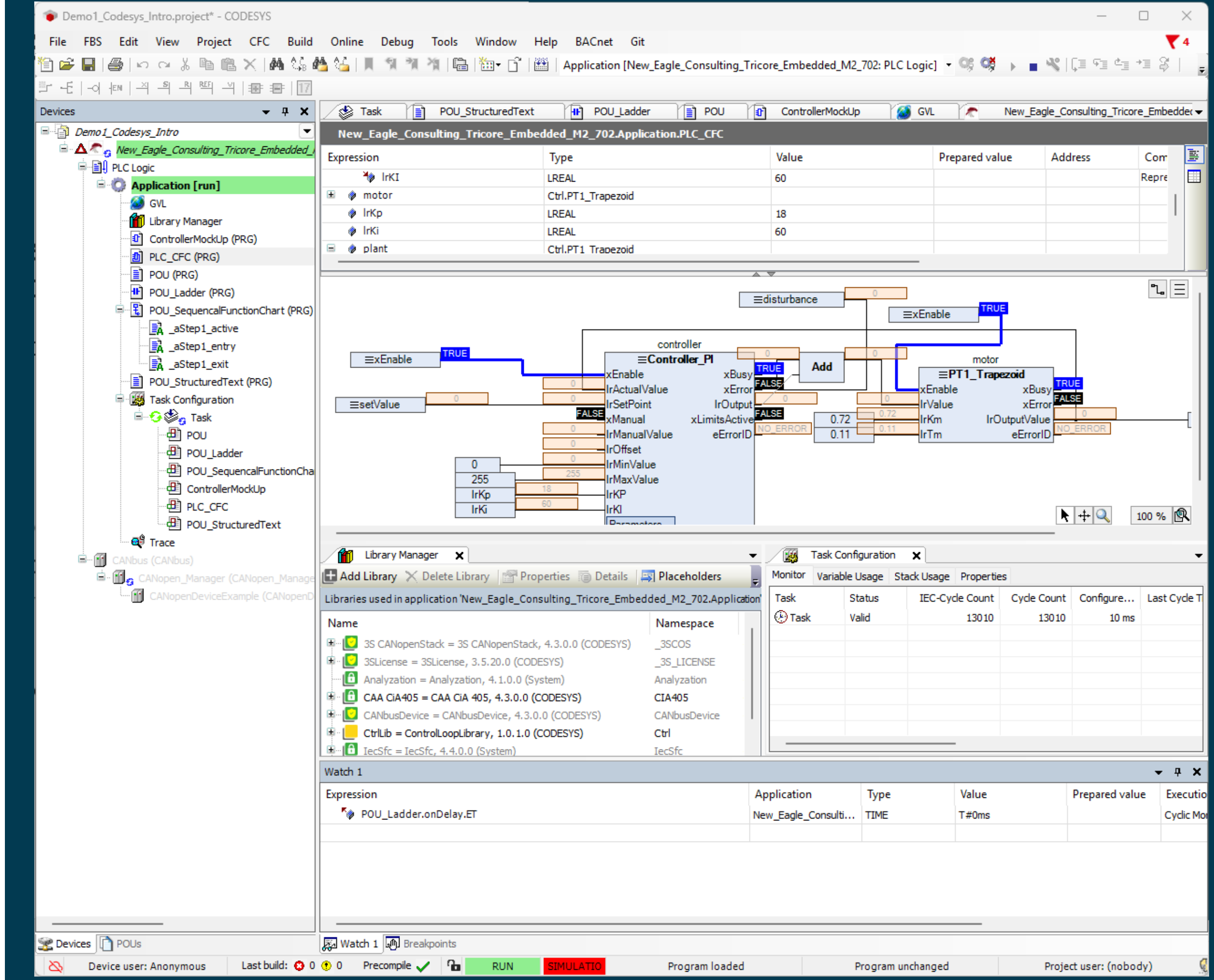
Description

© New Eagle. All Rights Reserved

Test and Simulation

CODESYS IDE incorporates a full unit test and PC simulation capability

- Robust simulation capability includes ability to compile application and deploy to simulated target on developer’s PC
- IDE incorporates a unit test definition and execution tool for both the simulator and on target contexts
- Incorporates full debug capability for the CODESYS application including:
    - Variable watch
    - Live variable override
    - Live data logging with plotting capability
    - Execution monitoring with breakpoints



New_Eagle_Consulting_Tricore_Embedded_M2_702.Application.PLC_CFC

Expression	Type	Value	Prepared value	Address	Com
lrKI	LREAL	60			Repre
motor	Ctrl.PT1_Trapezoid				
lrKp	LREAL	18			
lrKi	LREAL	60			
plant	Ctrl.PT1_Trapezoid				

Library Manager

Add Library · Delete Library · Properties · Details · Placeholders

Libraries used in application 'New_Eagle_Consulting_Tricore_Embedded_M2_702.Application'

Name	Namespace
3S CANopenStack = 3S CANopenStack, 4.3.0.0 (CODESYS)	_3SCOS
3SLicense = 3SLicense, 3.5.20.0 (CODESYS)	_3S_LICENSE
Analyzation = Analyzation, 4.1.0.0 (System)	Analyzation
CAA CiA405 = CAA CiA 405, 4.3.0.0 (CODESYS)	CIA405
CANbusDevice = CANbusDevice, 4.3.0.0 (CODESYS)	CANbusDevice
CtrlLib = ControlLoopLibrary, 1.0.1.0 (CODESYS)	Ctrl
IecSfc = IecSfc, 4.4.0.0 (System)	IecSfc

Task Configuration

Monitor · Variable Usage · Stack Usage · Properties

Task	Status	IEC-Cycle Count	Cycle Count	Configure...	Last Cycle T
Task	Valid	13010	13010	10 ms	

Watch 1

Expression	Application	Type	Value	Prepared value	Executio
POU_Ladder.onDelay.ET	New_Eagle_Consulti...	TIME	T#0ms		Cyclic Mo

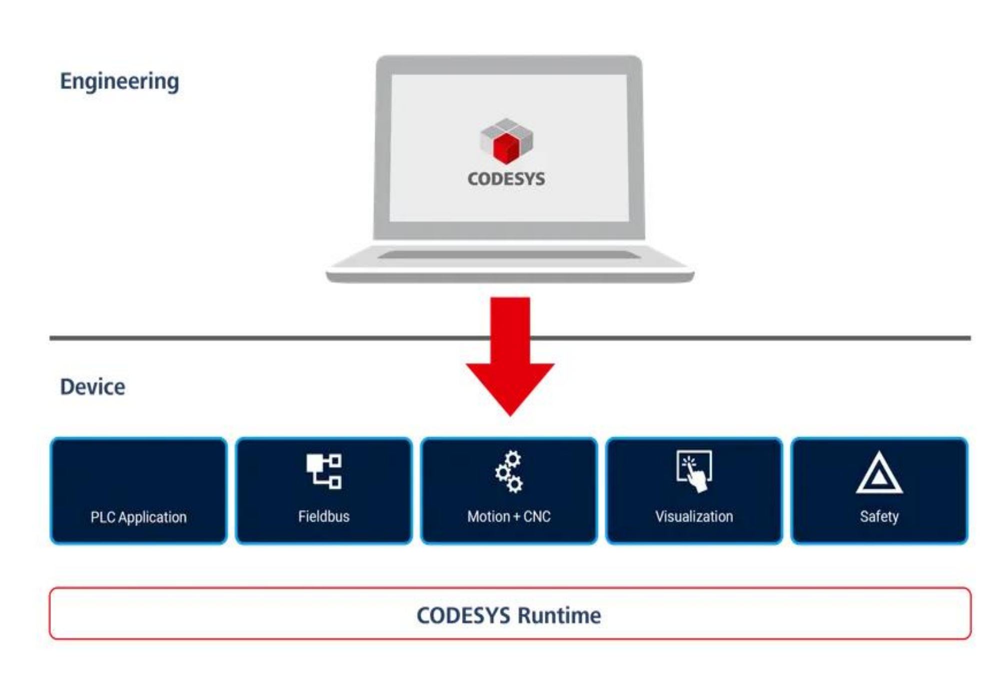
INNOVATE FASTER. SCALE SMARTER

© New Eagle. All Rights Reserved

Extensibility

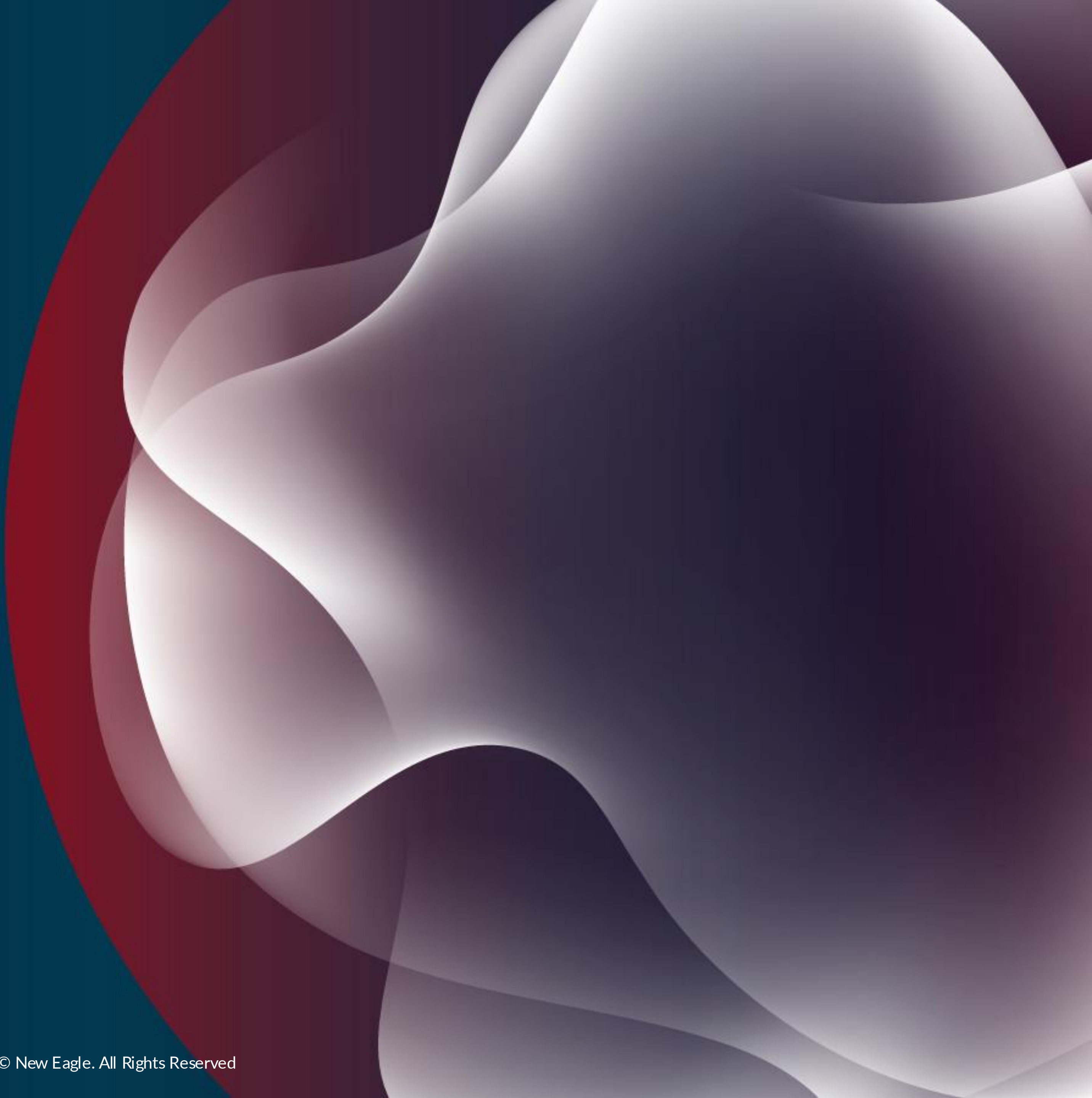
CODESYS is extensible by the end user with no involvement from New Eagle required

- Marketplace contains many application libraries and toolboxes, including both free and commercial options
- End user can freely add these capabilities to their project directly from the marketplace, without involving New Eagle
- Example capabilities that can be added this way include:
  - Fieldbus (in some cases)
  - Remote device awareness
  - Visualization enhancement toolboxes
  - Safety related toolboxes
  - Motion Planning
  - Industrial Internet of Things (IIoT)
  - Application design patterns



© New Eagle. All Rights Reserved

Supported Platforms

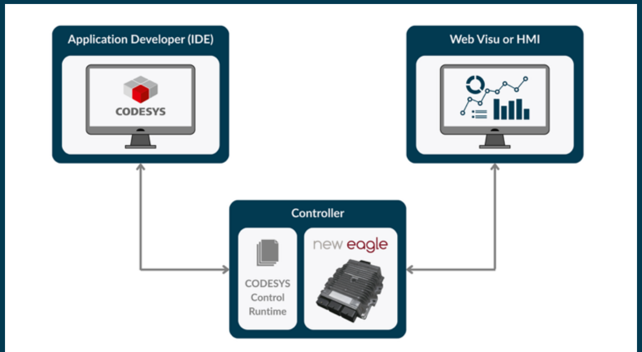


© New Eagle. All Rights Reserved

CODESYS On New Eagle ECUs

CODESYS will be progressively supported on all ECUs based on Infineon TriCore

- Includes any ECU implementing Infineon TC2xx or TC3xx
- Support on PowerPC ECUs possible with a customer-specific project
- Current ECU prioritization for CODESYS implementation, which may be influenced by customer request
  - GCM106 (available now)
  - GCM108 (available soon)
  - GCM44
  - GCM56
  - GCM112
  - CCM112
  - GCM111



CODESYS Support Coming To All Of These Great ECU Platforms In the Coming Months

GCM44

GCM56

GCM111

GCM112

GCM106

Already Available As A CODESYS ECU Today!

CCM112

GCM108

© New Eagle. All Rights Reserved

CODESYS ECU Details

Available Today!

GCM108

(M2-501)

High channel count hydraulic controller with powerful compute

- 16x independent hydraulic control channels
- **Current Control Advanced API**
- Powerful 3 core TC277 MCU
- 20x high side channels
- Automotive ethernet & 3x CAN-FD

GCM44

(M2-031)

Cost optimized hydraulic controller ideal for distributed systems

- 2 independent hydraulic control channels (potential for 4)
- **Current Control Advanced API**
- Single core TC234 MCU
- 3x high side channels
- 2x CAN-FD

GCM112

(M2-411)

High-performance general-purpose ECU

- Powerful 3 core TC377 MCU with 300MHz cores
- Integrated accelerometer
- Balanced mix of inputs and outputs
- 2x CAN-FD

GCM106

(M2-702)

High-performance power distribution / body module

- More than 30 high side output channels in 3A to 20A range
- Powerful 3 core TC277 MCU
- 1x H-Bridge
- Automotive ethernet & 3x CAN-FD

© New Eagle. All Rights Reserved

John Deere VPU2

Available Now!

Key Functions & Applications

Key Functions:

- 1x Nvidia Orin AGX (12x ARM)
- 4x CAN2.0
- 1x 100base-T1 Ethernet
- 1x 10G Base-T Ethernet
- 12x GMSL2 Power Over Coax
- 12x GMSL2 6Gbps Camera
- 1 TB Internal SSD
- 64GB LPDDR4 RAM
- 64GB eMMC Flash

Key Applications:

- Advanced Vision Systems
- High-End Compute

Advanced Capabilities

High Performance Compute

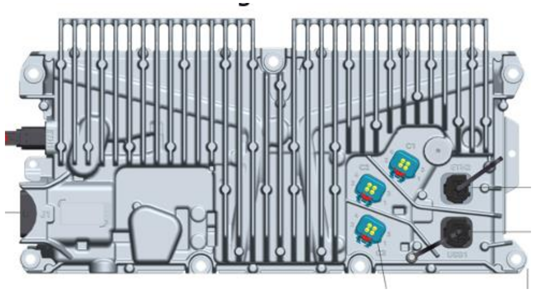
- 12x 2.2GHz ARM A78AE
- 2048x 1.3GHz Core GPU
- 64x 1.3GHz Tensor Cores

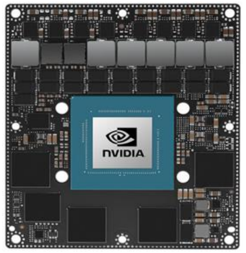
Cybersecurity Support

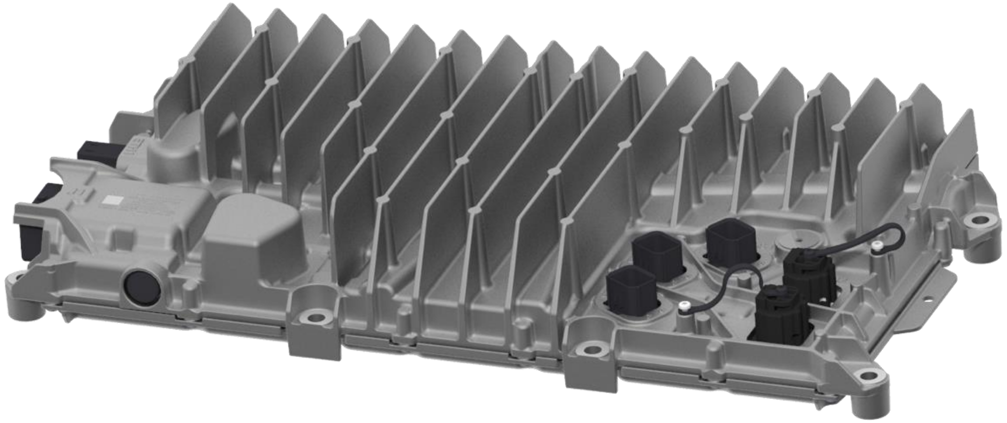
- Nvidia OPTEE

Environmental Features

- IP66/67
- -25C to +50C operational temperature range







© New Eagle. All Rights Reserved

VPU Roadmap

Available H2 2026

"VPU Medium"

Key Functions:

- 1x Nvidia Orin NX + AMD ZU7
- 3x CAN
- 1x 100base-T1 Ethernet
- 1x 5G Base-T Ethernet
- 8x camera inputs
- 256GB Internal SSD
- 16GB LPDDR4 RAM

Available Now For Prototyping

"VPU Lite"

Key Functions:

- 1x AMD ZU7
- 2x CAN
- 1x 1000base-T1 Ethernet
- 1x 100base-T1 Ethernet
- 4x camera inputs
- 32GB eMMC Flash
- 4GB LPDDR4 RAM

Software Enablement

Linux SDK available initially, able to support the following additional workflows:

- Soft PLC capability through installation of CODESYS
- Ability to develop in Matlab/Simulink including GPU Coder
- Ability to develop in C, C++, Rust, etc

Raptor support will trail hardware availability by 6-9 months

© New Eagle. All Rights Reserved

Deploy CODESYS PLC

VPU hardware can be deployed as high performance rugged industrial PCs

CODESYS is supported on embedded Linux systems natively or as a “Virtual PLC” within a docker container

- Deployable alongside other containerized apps, like ROS2
- CODESYS and ROS2 deployment can be completely managed by the end user

CODESYS supported application I/O includes:

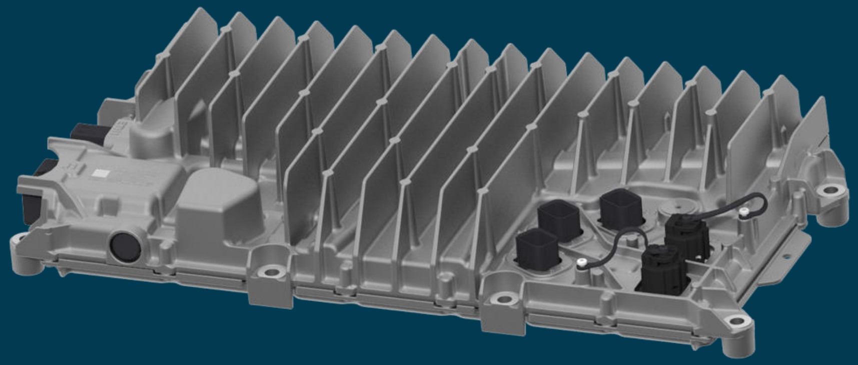
- CAN, including CANopen Master/Slave and J1939
- Ethernet fieldbus
- Local discrete IO and Analog Inputs

PLC connectivity options include:

- WebVISU HMI rendered via webserver
- Automation server for remote access
- Local debug via CAN and/or Ethernet

ROS 2™

NVIDIA JETSON



CODESYS

docker

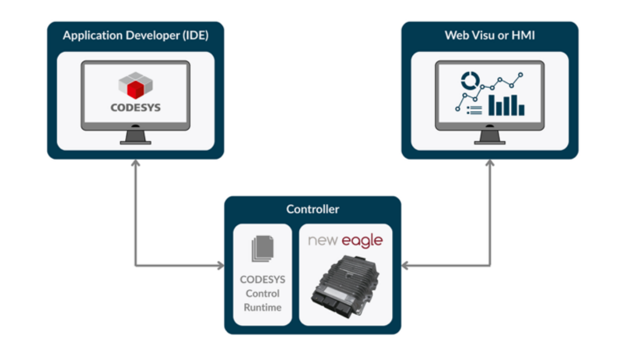
© New Eagle. All Rights Reserved

DR 8/10/12 Displays

CODESYS control and visualization on DR Display Family!

New Eagle to provide example applications that feature:

- Fieldbus data exchange with Raptor ECUs, other CODESYS ECUs, High Compute Platform
- Visualization and Touch Screen functionality is programmable via CODESYS environment
- Graphical widget layout editor
- Multiple page navigation templates
- User roles and permissions
- Rich set of widgets for logging, alarming, HMI use cases
- Integration of advanced graphics possible via HTLM5 tools
- DR Family is embedded Linux based so CODESYS can coexist with other user applications

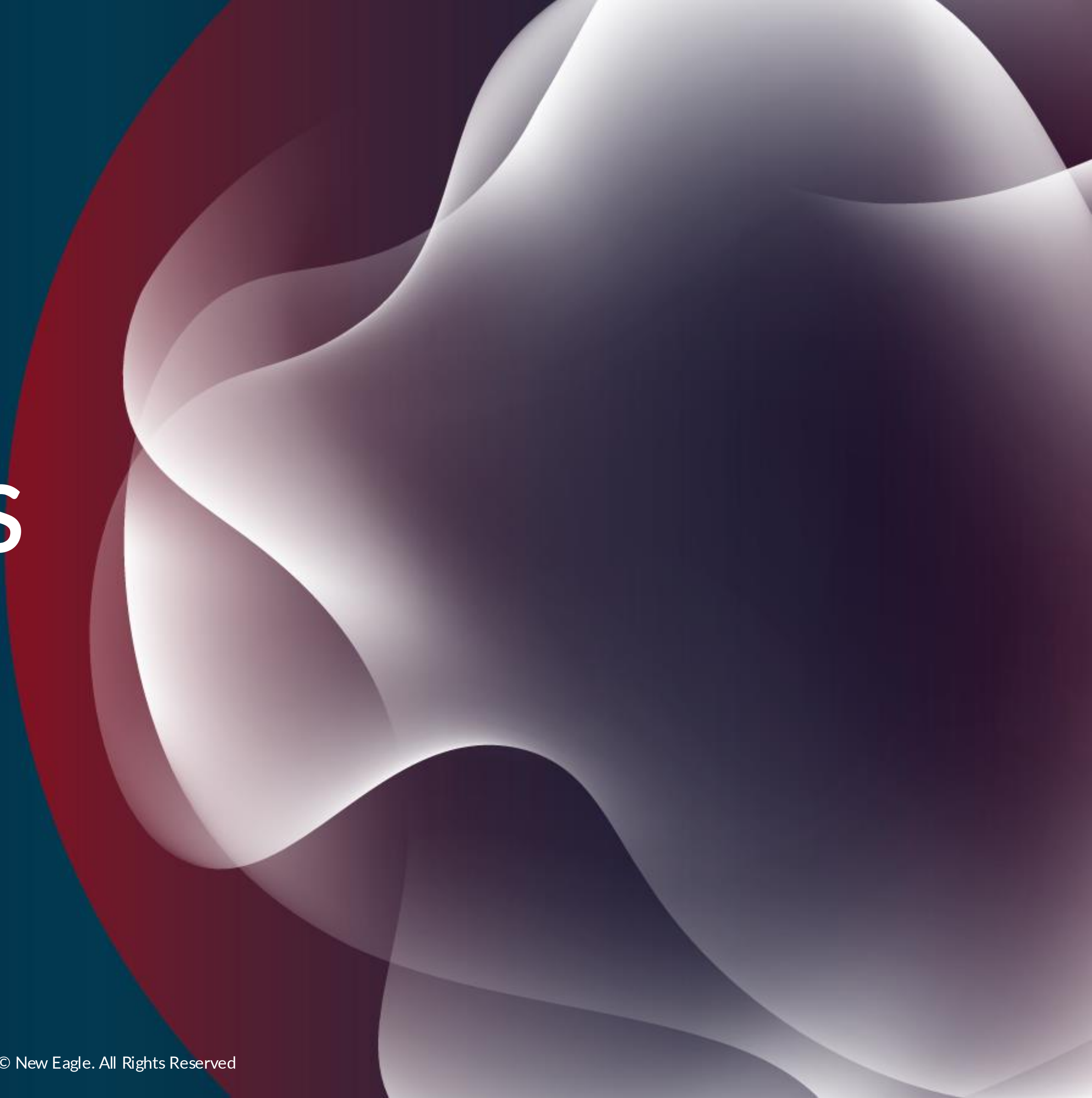


CODESYS Support Coming To Select Joh Deere Displays Available From New Eagle

© New Eagle. All Rights Reserved

Raptor + CODESYS

Mixed controller example



© New Eagle. All Rights Reserved

Distributed Control Example

RCM112 – Raptor

- Consume TPDO and NMT state from all nodes
- Produce CANopen PDO
- CANopen SYNC

GCM108 –CODESYS

- Advanced Current Control Capability
- CANopen Slave

GCM106 – CODESYS

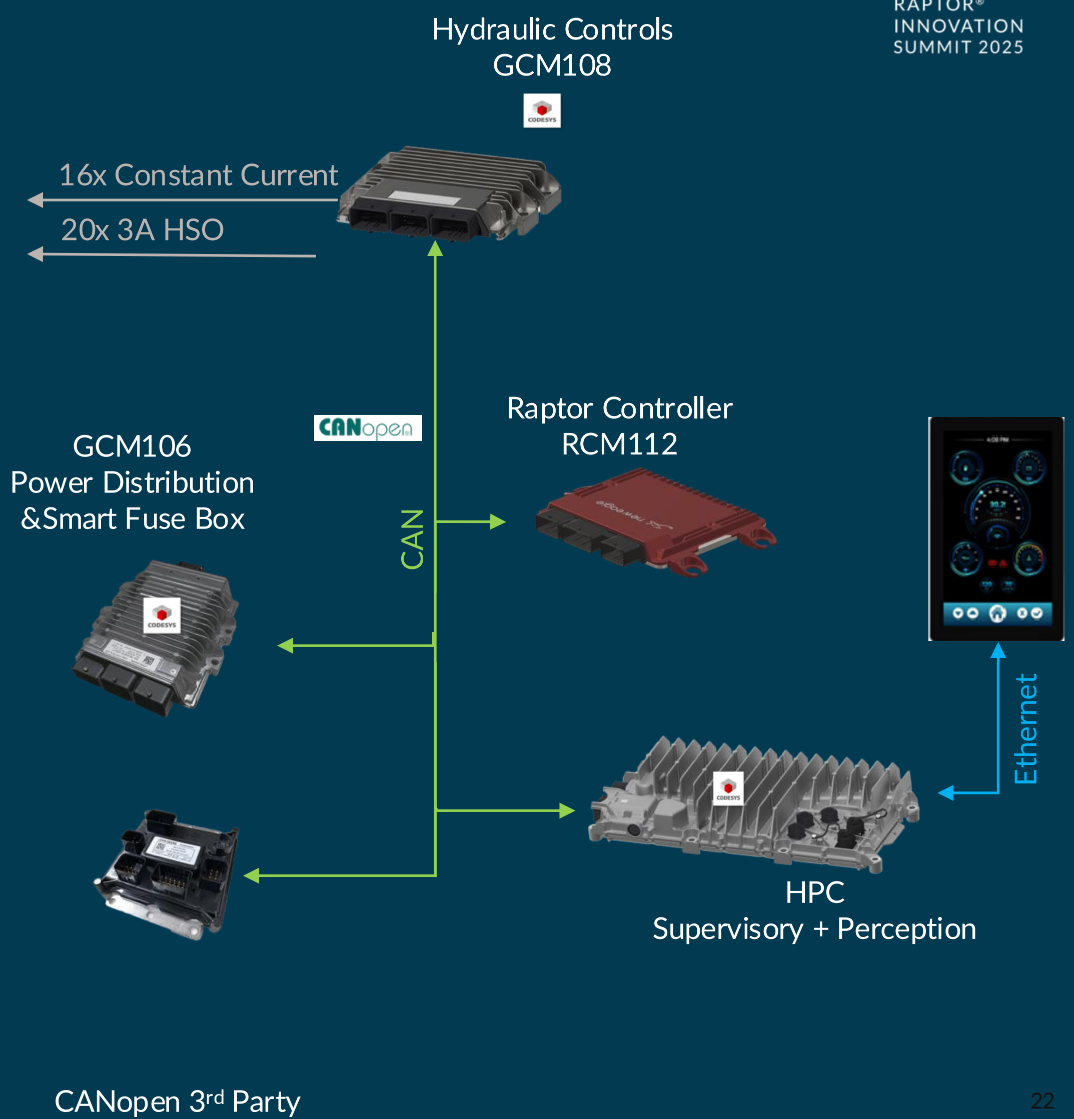
- CANopen Master

CANopen – Third Party Device

- CANopen Slave

High Performance Compute – CODESYS vPLC

- Consume all CANopen PDO for telemetry/upstream display
- Produce PDO for perception information as CANopen Slave
- Gather diagnostic SDO data at low priority
- WebVisu or PLC data share to DR12 Display



© New Eagle. All Rights Reserved

new eagle

RAPTOR®
INNOVATION
SUMMIT 2025

CANopen Summary

Network Basics

- Nodes have unique address
- Object Dictionary enumerates all data available on the device
- Electronic Data Sheet EDS is a file that shares the Object Dictionary across tools
- Service Data Object (SDO) provides request – response R/W access typically for startup
- Process Data Object (PDO) is for implicit messaging
- COB- ID Scheme for CAN ID
- CiA Device Profiles

PDO Transport Configuration

- PDOs only used in OPERATIONAL STATE
- Receive and Transmit PDOs are from the context of the slave devices
- Configuration is typically configurable at run time
- Triggers to send data include timer, CoS, SYNC, others

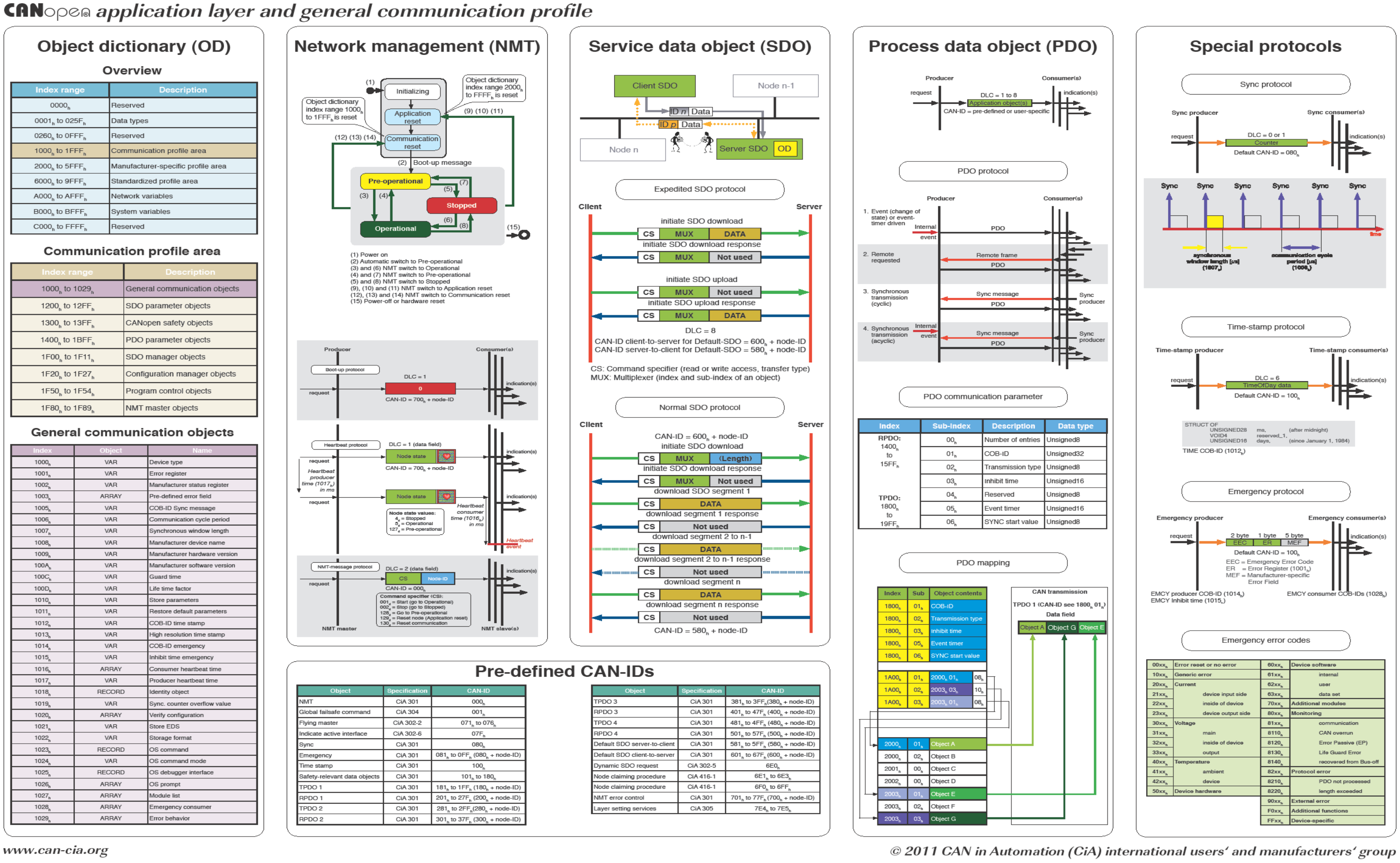
PDO Mapping

- Payload of PDO message is know via EDS or enumeration from device online
- Often default or DS Profile
- Mapping process allow for any Object Dictionary data to be used in PDO message

Specials

- Nodes can emit EMERGENCY messages
- Network Management (NMT) from manager node coordinates PRE-OPERATIONAL, STOPPED, OPERATIONAL STATE of nodes
- TIMESYNC is distribution of system time
- SYNC producer can be any

© New Eagle. All Rights Reserved



CANopen® application layer and general communication profile

Object dictionary (OD)

Overview

Index range	Description
0000$_h$	Reserved
0001$_h$ to 025F$_h$	Data types
0260$_h$ to 0FFF$_h$	Reserved
1000$_h$ to 1FFF$_h$	Communication profile area
2000$_h$ to 5FFF$_h$	Manufacturer-specific profile area
6000$_h$ to 9FFF$_h$	Standardized profile area
A000$_h$ to AFFF$_h$	Network variables
B000$_h$ to BFFF$_h$	System variables
C000$_h$ to FFFF$_h$	Reserved

Communication profile area

Index range	Description
1000$_h$ to 1029$_h$	General communication objects
1200$_h$ to 12FF$_h$	SDO parameter objects
1300$_h$ to 13FF$_h$	CANopen safety objects
1400$_h$ to 1BFF$_h$	PDO parameter objects
1F00$_h$ to 1F11$_h$	SDO manager objects
1F20$_h$ to 1F27$_h$	Configuration manager objects
1F50$_h$ to 1F54$_h$	Program control objects
1F80$_h$ to 1F89$_h$	NMT master objects

General communication objects

Index	Object	Name
1000$_h$	VAR	Device type
1001$_h$	VAR	Error register
1002$_h$	VAR	Manufacturer status register
1003$_h$	ARRAY	Pre-defined error field
1005$_h$	VAR	COB-ID Sync message
1006$_h$	VAR	Communication cycle period
1007$_h$	VAR	Synchronous window length
1008$_h$	VAR	Manufacturer device name
1009$_h$	VAR	Manufacturer hardware version
100A$_h$	VAR	Manufacturer software version
100C$_h$	VAR	Guard time
100D$_h$	VAR	Life time factor
1010$_h$	VAR	Store parameters
1011$_h$	VAR	Restore default parameters
1012$_h$	VAR	COB-ID time stamp
1013$_h$	VAR	High resolution time stamp
1014$_h$	VAR	COB-ID emergency
1015$_h$	VAR	Inhibit time emergency
1016$_h$	ARRAY	Consumer heartbeat time
1017$_h$	VAR	Producer heartbeat time
1018$_h$	RECORD	Identity object
1019$_h$	VAR	Sync. counter overflow value
1020$_h$	ARRAY	Verify configuration
1021$_h$	VAR	Store EDS
1022$_h$	VAR	Storage format
1023$_h$	RECORD	OS command
1024$_h$	VAR	OS command mode
1025$_h$	RECORD	OS debugger interface
1026$_h$	ARRAY	OS prompt
1027$_h$	ARRAY	Module list
1028$_h$	ARRAY	Emergency consumer
1029$_h$	ARRAY	Error behavior

Network management (NMT)

(1) Initializing

Application reset

Communication reset

Object dictionary index range 1000$_h$ to 1FFF$_h$ is reset

Object dictionary index range 2000$_h$ to FFFF$_h$ is reset

(9) (10) (11)

(12) (13) (14)

(2) Boot-up message

Pre-operational

(3) (4) (5) (7)

Stopped

(6) (8)

Operational

(15)

(1) Power on
(2) Automatic switch to Pre-operational
(3) and (6) NMT switch to Operational
(4) and (7) NMT switch to Pre-operational
(5) and (8) NMT switch to Stopped
(9), (10) and (11) NMT switch to Application reset
(12), (13) and (14) NMT switch to Communication reset
(15) Power-off or hardware reset

Producer — Consumer(s)

Boot-up protocol

request — DLC = 1 — 0 — CAN-ID = 700$_h$ + node-ID — indication(s)

Heartbeat protocol

request — DLC = 1 (data field) — Node state — CAN-ID = 700$_h$ + node-ID — indication(s)

Heartbeat producer time (1017$_h$) in ms

request — Node state — indication(s)

Node state values:
4$_d$ = Stopped
5$_d$ = Operational
127$_d$ = Pre-operational

Heartbeat consumer time (1016$_h$) in ms

Heartbeat event

NMT-message protocol

request — DLC = 2 (data field) — CS | Node-ID — CAN-ID = 000$_h$ — indication(s)

Command specifier (CS):
001$_d$ = Start (go to Operational)
002$_d$ = Stop (go to Stopped)
128$_d$ = Go to Pre-operational
129$_d$ = Reset node (Application reset)
130$_d$ = Reset communication

NMT master — NMT slave(s)

Service data object (SDO)

Client SDO — Node n-1

ID *n* | Data

ID *p* | Data

Node n — Server SDO | OD

Expedited SDO protocol

Client — Server

initiate SDO download: CS | MUX | DATA
initiate SDO download response: CS | MUX | Not used

initiate SDO upload: CS | MUX | Not used
initiate SDO upload response: CS | MUX | DATA

DLC = 8

CAN-ID client-to-server for Default-SDO = 600$_h$ + node-ID
CAN-ID server-to-client for Default-SDO = 580$_h$ + node-ID

CS: Command specifier (read or write access, transfer type)
MUX: Multiplexer (index and sub-index of an object)

Normal SDO protocol

Client — Server

CAN-ID = 600$_h$ + node-ID

initiate SDO download: CS | MUX | (Length)
initiate SDO download response: CS | MUX | Not used
download SDO segment 1: CS | DATA
download segment 1 response: CS | Not used
download segment 2 to n-1: CS | DATA
download segment 2 to n-1 response: CS | Not used
download segment n: CS | DATA
download segment n response: CS | Not used

CAN-ID = 580$_h$ + node-ID

Pre-defined CAN-IDs

Object	Specification	CAN-ID
NMT	CiA 301	000$_h$
Global failsafe command	CiA 304	001$_h$
Flying master	CiA 302-2	071$_h$ to 076$_h$
Indicate active interface	CiA 302-6	07F$_h$
Sync	CiA 301	080$_h$
Emergency	CiA 301	081$_h$ to 0FF$_h$ (080$_h$ + node-ID)
Time stamp	CiA 301	100$_h$
Safety-relevant data objects	CiA 301	101$_h$ to 180$_h$
TPDO 1	CiA 301	181$_h$ to 1FF$_h$ (180$_h$ + node-ID)
RPDO 1	CiA 301	201$_h$ to 27F$_h$ (200$_h$ + node-ID)
TPDO 2	CiA 301	281$_h$ to 2FF$_h$(280$_h$ + node-ID)
RPDO 2	CiA 301	301$_h$ to 37F$_h$ (300$_h$ + node-ID)

Object	Specification	CAN-ID
TPDO 3	CiA 301	381$_h$ to 3FF$_h$(380$_h$ + node-ID)
RPDO 3	CiA 301	401$_h$ to 47F$_h$ (400$_h$ + node-ID)
TPDO 4	CiA 301	481$_h$ to 4FF$_h$ (480$_h$ + node-ID)
RPDO 4	CiA 301	501$_h$ to 57F$_h$ (500$_h$ + node-ID)
Default SDO server-to-client	CiA 301	581$_h$ to 5FF$_h$ (580$_h$ + node-ID)
Default SDO client-to-server	CiA 301	601$_h$ to 67F$_h$ (600$_h$ + node-ID)
Dynamic SDO request	CiA 302-5	6E0$_h$
Node claiming procedure	CiA 416-1	6E1$_h$ to 6E3$_h$
Node claiming procedure	CiA 416-1	6F0$_h$ to 6FF$_h$
NMT error control	CiA 301	701$_h$ to 77F$_h$ (700$_h$ + node-ID)
Layer setting services	CiA 305	7E4$_h$ to 7E5$_h$

Process data object (PDO)

Producer — Consumer(s)

request — DLC = 1 to 8 — Application object(s) — CAN-ID = pre-defined or user-specific — indication(s)

PDO protocol

Producer — Consumer(s)

1. Event (change of state) or event-timer driven — Internal event — PDO
2. Remote requested — Remote frame — PDO
3. Synchronous transmission (cyclic) — Sync message — PDO — Sync producer
4. Synchronous transmission (acyclic) — Internal event — Sync message — PDO — Sync producer

PDO communication parameter

Index	Sub-Index	Description	Data type
RPDO: 1400$_h$ to 15FF$_h$	00$_h$	Number of entries	Unsigned8
	01$_h$	COB-ID	Unsigned32
	02$_h$	Transmission type	Unsigned8
	03$_h$	Inhibit time	Unsigned16
TPDO: 1800$_h$ to 19FF$_h$	04$_h$	Reserved	Unsigned8
	05$_h$	Event timer	Unsigned16
	06$_h$	SYNC start value	Unsigned8

PDO mapping

Index	Sub	Object contents	
1800$_h$	01$_h$	COB-ID	
1800$_h$	02$_h$	Transmission type	
1800$_h$	03$_h$	Inhibit time	
1800$_h$	05$_h$	Event timer	
1800$_h$	06$_h$	SYNC start value	
1A00$_h$	01$_h$	2000$_h$, 01$_h$	08$_h$
1A00$_h$	02$_h$	2003$_h$, 03$_h$	10$_h$
1A00$_h$	03$_h$	2003$_h$, 01$_h$	08$_h$
2000$_h$	01$_h$	Object A	
2000$_h$	02$_h$	Object B	
2001$_h$	00$_h$	Object C	
2002$_h$	00$_h$	Object D	
2003$_h$	01$_h$	Object E	
2003$_h$	02$_h$	Object F	
2003$_h$	03$_h$	Object G	

CAN transmission

TPDO 1 (CAN-ID see 1800$_h$, 01$_h$)

Data field

Object A | Object G | Object E

Special protocols

Sync protocol

Sync producer — Sync consumer(s)

request — DLC = 0 or 1 — Counter — Default CAN-ID = 080$_h$ — indication(s)

Sync Sync Sync Sync Sync Sync

time

synchronous window length [µs] (1007$_h$)

communication cycle period [µs] (1006$_h$)

Time-stamp protocol

Time-stamp producer — Time-stamp consumer(s)

request — DLC = 6 — TimeOfDay data — Default CAN-ID = 100$_h$ — indication(s)

```
STRUCT OF
        UNSIGNED28    ms,           (after midnight)
        VOID4         reserved_1,
        UNSIGNED16    days,         (since January 1, 1984)
```

TIME COB-ID (1012$_h$)

Emergency protocol

Emergency producer — Emergency consumer(s)

request — 2 byte EEC | 1 byte ER | 5 byte MEF — Default CAN-ID = 100$_h$ — indication(s)

EEC = Emergency Error Code
ER = Error Register (1001$_h$)
MEF = Manufacturer-specific Error Field

EMCY producer COB-ID (1014$_h$)
EMCY Inhibit time (1015$_h$)

EMCY consumer COB-IDs (1028$_h$)

Emergency error codes

00xx$_h$	Error reset or no error	60xx$_h$	Device software
10xx$_h$	Generic error	61xx$_h$	internal
20xx$_h$	Current	62xx$_h$	user
21xx$_h$	device input side	63xx$_h$	data set
22xx$_h$	inside of device	70xx$_h$	Additional modules
23xx$_h$	device output side	80xx$_h$	Monitoring
30xx$_h$	Voltage	81xx$_h$	communication
31xx$_h$	main	8110$_h$	CAN overrun
32xx$_h$	inside of device	8120$_h$	Error Passive (EP)
33xx$_h$	output	8130$_h$	Life Guard Error
40xx$_h$	Temperature	8140$_h$	recovered from Bus-off
41xx$_h$	ambient	82xx$_h$	Protocol error
42xx$_h$	device	8210$_h$	PDO not processed
50xx$_h$	Device hardware	8220$_h$	length exceeded
		90xx$_h$	External error
		F0xx$_h$	Additional functions
		FFxx$_h$	Device-specific

www.can-cia.org

© 2011 CAN in Automation (CiA) international users' and manufacturers' group

© New Eagle. All Rights Reserved

new eagle

RAPTOR® INNOVATION SUMMIT 2025

GCM106

System Function

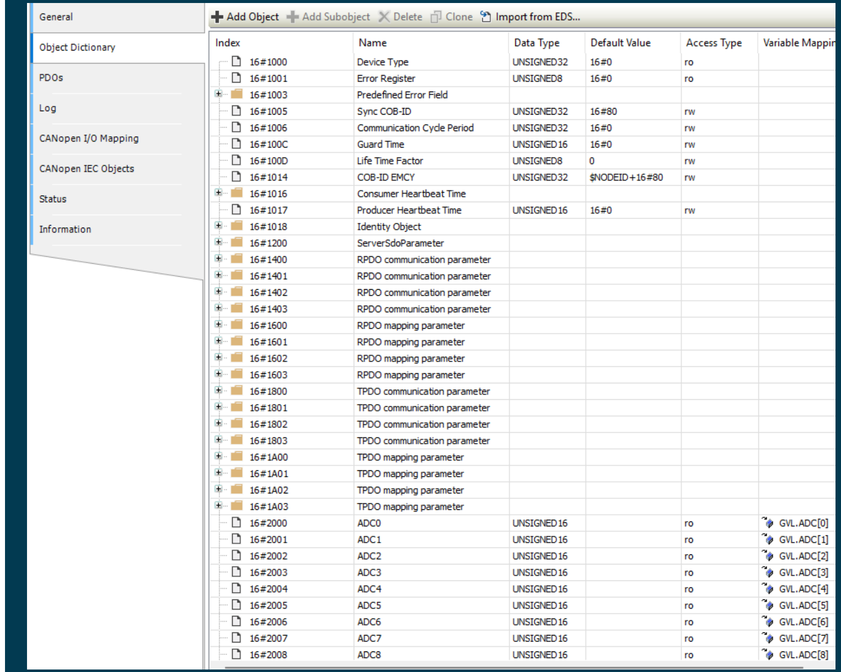
- Advanced constant current drivers for valve control
- Consume RCM 112 commands from Raptor
- Status and IO

Implementation

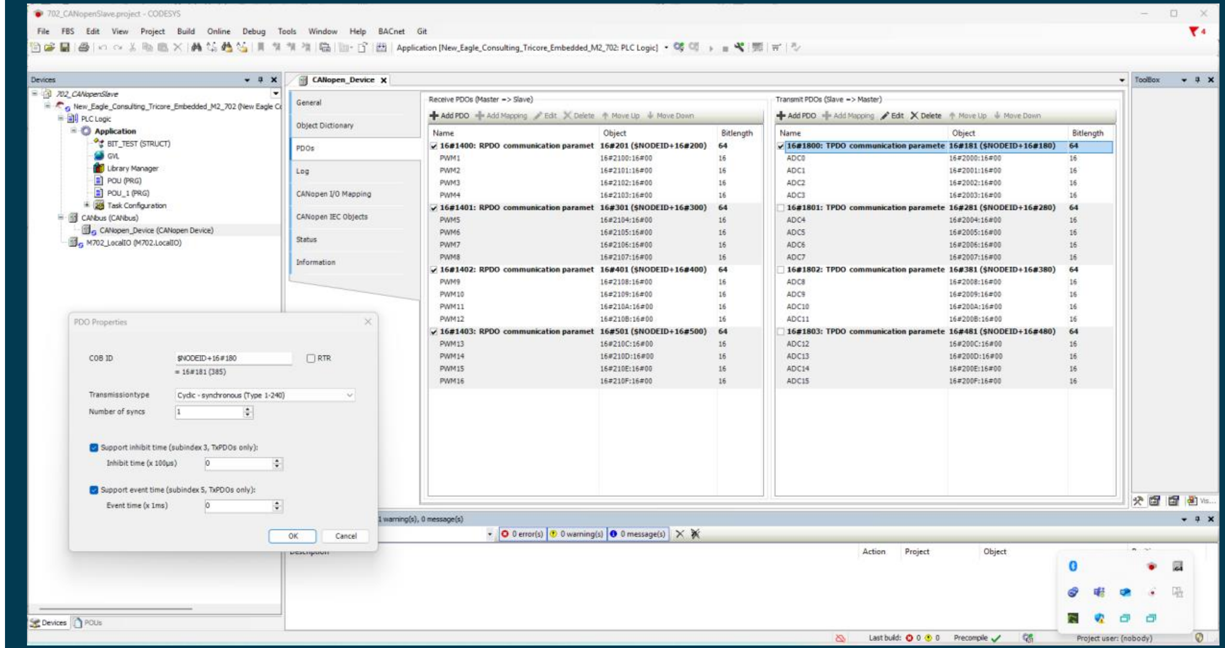
- CODESYS Default CANopen Slave Project
- Program and deploy with zero effort
- CANopen Object Dictionary mapped directly to IO
- CANopen NMT state to clear output to default state

Option to extend CODESYS GCM Behavior

- Customize stand alone control behavior
- Extend Object Dictionary with an application interface
- Recreate EDS
    - Export to CANopen Master
    - Generate matching DBC for raw CAN use



Index	Name	Data Type	Default Value	Access Type	Variable Mappin
16#1000	Device Type	UNSIGNED32	16#0	ro	
16#1001	Error Register	UNSIGNED8	16#0	ro	
16#1003	Predefined Error Field				
16#1005	Sync COB-ID	UNSIGNED32	16#80	rw	
16#1006	Communication Cycle Period	UNSIGNED32	16#0	rw	
16#100C	Guard Time	UNSIGNED16	16#0	rw	
16#100D	Life Time Factor	UNSIGNED8	0	rw	
16#1014	COB-ID EMCY	UNSIGNED32	$NODEID+16#80	rw	
16#1016	Consumer Heartbeat Time				
16#1017	Producer Heartbeat Time	UNSIGNED16	16#0	rw	
16#1018	Identity Object				
16#1200	ServerSdoParameter				
16#1400	RPDO communication parameter				
16#1401	RPDO communication parameter				
16#1402	RPDO communication parameter				
16#1403	RPDO communication parameter				
16#1600	RPDO mapping parameter				
16#1601	RPDO mapping parameter				
16#1602	RPDO mapping parameter				
16#1603	RPDO mapping parameter				
16#1800	TPDO communication parameter				
16#1801	TPDO communication parameter				
16#1802	TPDO communication parameter				
16#1803	TPDO communication parameter				
16#1A00	TPDO mapping parameter				
16#1A01	TPDO mapping parameter				
16#1A02	TPDO mapping parameter				
16#1A03	TPDO mapping parameter				
16#2000	ADC0	UNSIGNED16		ro	GVL.ADC[0]
16#2001	ADC1	UNSIGNED16		ro	GVL.ADC[1]
16#2002	ADC2	UNSIGNED16		ro	GVL.ADC[2]
16#2003	ADC3	UNSIGNED16		ro	GVL.ADC[3]
16#2004	ADC4	UNSIGNED16		ro	GVL.ADC[4]
16#2005	ADC5	UNSIGNED16		ro	GVL.ADC[5]
16#2006	ADC6	UNSIGNED16		ro	GVL.ADC[6]
16#2007	ADC7	UNSIGNED16		ro	GVL.ADC[7]
16#2008	ADC8	UNSIGNED16		ro	GVL.ADC[8]



Receive PDOs (Master => Slave)

Name	Object	Bitlength
☑ **16#1400: RPDO communication paramet**	**16#201 ($NODEID+16#200)**	**64**
PWM1	16#2100:16#00	16
PWM2	16#2101:16#00	16
PWM3	16#2102:16#00	16
PWM4	16#2103:16#00	16
☑ **16#1401: RPDO communication paramet**	**16#301 ($NODEID+16#300)**	**64**
PWM5	16#2104:16#00	16
PWM6	16#2105:16#00	16
PWM7	16#2106:16#00	16
PWM8	16#2107:16#00	16
☑ **16#1402: RPDO communication paramet**	**16#401 ($NODEID+16#400)**	**64**
PWM9	16#2108:16#00	16
PWM10	16#2109:16#00	16
PWM11	16#210A:16#00	16
PWM12	16#210B:16#00	16
☑ **16#1403: RPDO communication paramet**	**16#501 ($NODEID+16#500)**	**64**
PWM13	16#210C:16#00	16
PWM14	16#210D:16#00	16
PWM15	16#210E:16#00	16
PWM16	16#210F:16#00	16

Transmit PDOs (Slave => Master)

Name	Object	Bitlength
☑ **16#1800: TPDO communication paramete**	**16#181 ($NODEID+16#180)**	**64**
ADC0	16#2000:16#00	16
ADC1	16#2001:16#00	16
ADC2	16#2002:16#00	16
ADC3	16#2003:16#00	16
☐ **16#1801: TPDO communication paramete**	**16#281 ($NODEID+16#280)**	**64**
ADC4	16#2004:16#00	16
ADC5	16#2005:16#00	16
ADC6	16#2006:16#00	16
ADC7	16#2007:16#00	16
☐ **16#1802: TPDO communication paramete**	**16#381 ($NODEID+16#380)**	**64**
ADC8	16#2008:16#00	16
ADC9	16#2009:16#00	16
ADC10	16#200A:16#00	16
ADC11	16#200B:16#00	16
☐ **16#1803: TPDO communication paramete**	**16#481 ($NODEID+16#480)**	**64**
ADC12	16#200C:16#00	16
ADC13	16#200D:16#00	16
ADC14	16#200E:16#00	16
ADC15	16#200F:16#00	16

PDO Properties

COB ID: $NODEID+16#180 ☐ RTR
= 16#181 (385)

Transmissiontype: Cyclic - synchronous (Type 1-240)

Number of syncs: 1

☑ Support inhibit time (subindex 3, TxPDOs only):
Inhibit time (x 100µs): 0

☑ Support event time (subindex 5, TxPDOs only):
Event time (x 1ms): 0

OK Cancel

1 warning(s), 0 message(s)

0 error(s) 0 warning(s) 0 message(s)

Distributed Control Node

© New Eagle. All Rights Reserved

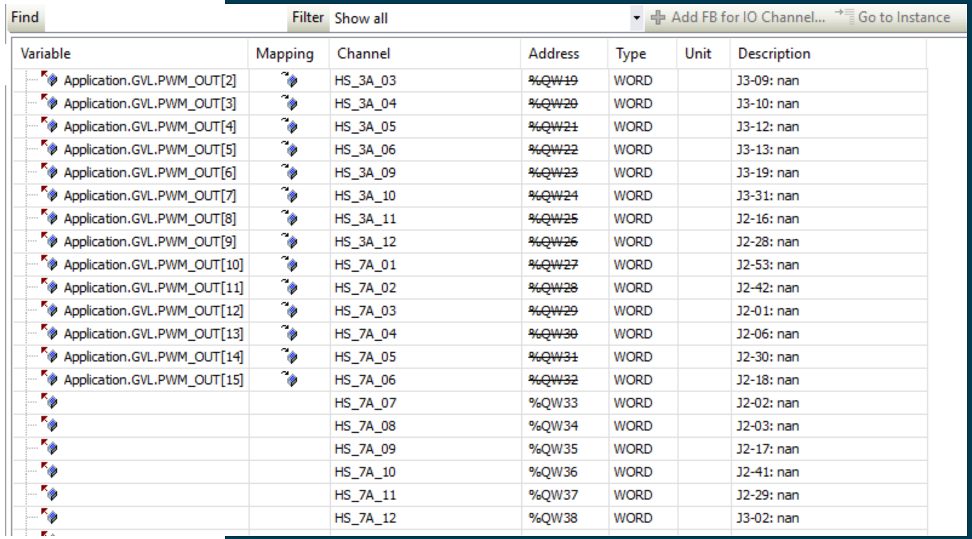
GCM108

System Function

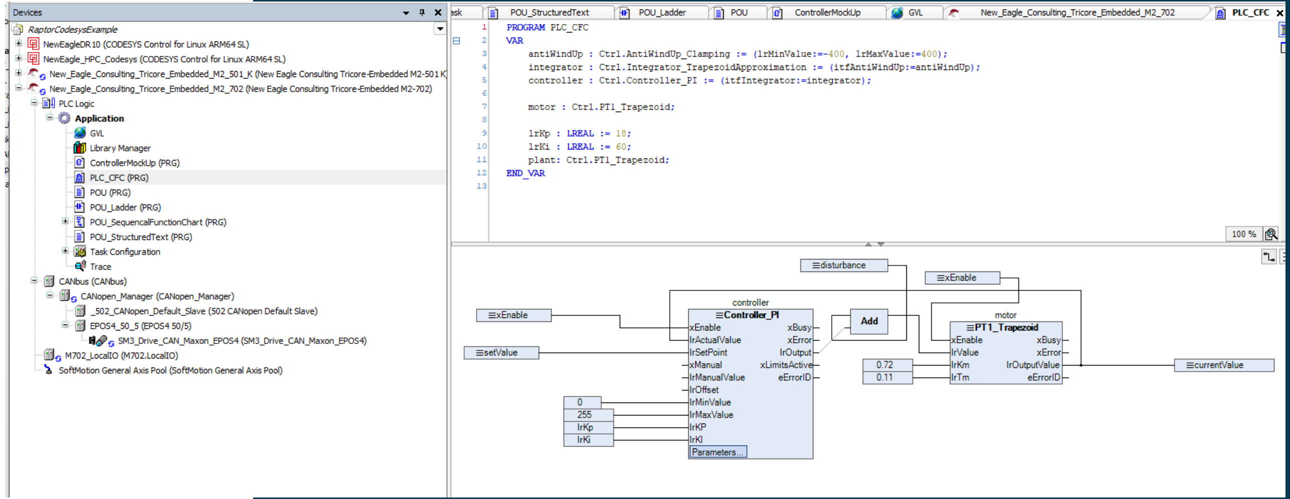
- CANopen Master / Manager
- Manage Power Distribution
- Status and IO

Implementation

- CODESYS Application with local IO control
- CANopen Master
    - Configure external nodes if necessary
    - NMT State system
    - Manage Startup, Operation, Fault, and Recovery
    - Validate node presence and identity



Variable	Mapping	Channel	Address	Type	Unit	Description
Application.GVL.PWM_OUT[2]		HS_3A_03	~~%QW19~~	WORD		J3-09: nan
Application.GVL.PWM_OUT[3]		HS_3A_04	~~%QW20~~	WORD		J3-10: nan
Application.GVL.PWM_OUT[4]		HS_3A_05	~~%QW21~~	WORD		J3-12: nan
Application.GVL.PWM_OUT[5]		HS_3A_06	~~%QW22~~	WORD		J3-13: nan
Application.GVL.PWM_OUT[6]		HS_3A_09	~~%QW23~~	WORD		J3-19: nan
Application.GVL.PWM_OUT[7]		HS_3A_10	~~%QW24~~	WORD		J3-31: nan
Application.GVL.PWM_OUT[8]		HS_3A_11	~~%QW25~~	WORD		J2-16: nan
Application.GVL.PWM_OUT[9]		HS_3A_12	~~%QW26~~	WORD		J2-28: nan
Application.GVL.PWM_OUT[10]		HS_7A_01	~~%QW27~~	WORD		J2-53: nan
Application.GVL.PWM_OUT[11]		HS_7A_02	~~%QW28~~	WORD		J2-42: nan
Application.GVL.PWM_OUT[12]		HS_7A_03	~~%QW29~~	WORD		J2-01: nan
Application.GVL.PWM_OUT[13]		HS_7A_04	~~%QW30~~	WORD		J2-06: nan
Application.GVL.PWM_OUT[14]		HS_7A_05	~~%QW31~~	WORD		J2-30: nan
Application.GVL.PWM_OUT[15]		HS_7A_06	~~%QW32~~	WORD		J2-18: nan
		HS_7A_07	%QW33	WORD		J2-02: nan
		HS_7A_08	%QW34	WORD		J2-03: nan
		HS_7A_09	%QW35	WORD		J2-17: nan
		HS_7A_10	%QW36	WORD		J2-41: nan
		HS_7A_11	%QW37	WORD		J2-29: nan
		HS_7A_12	%QW38	WORD		J3-02: nan



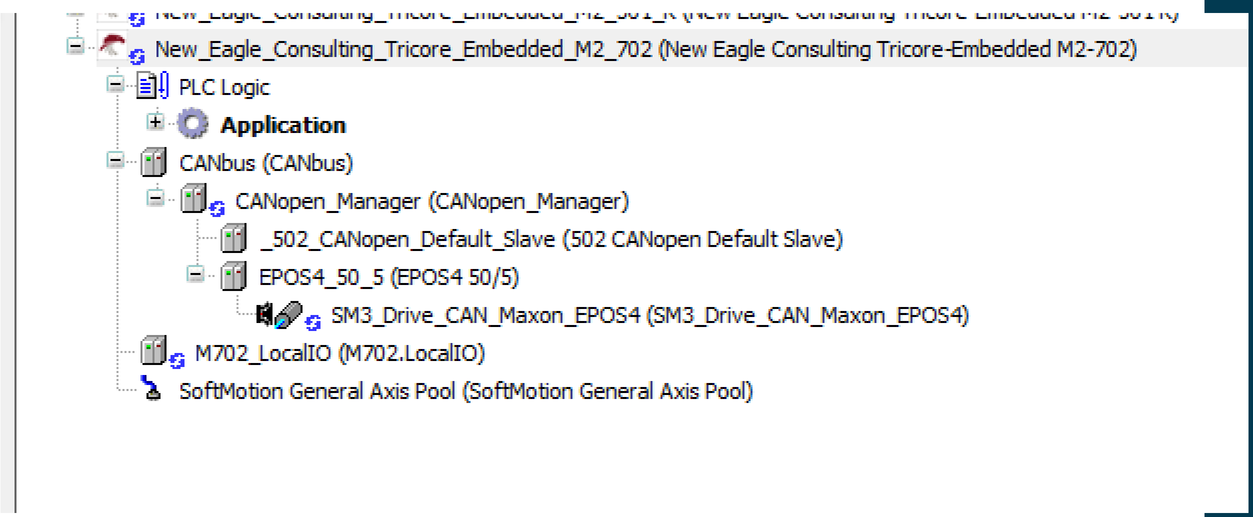
```
RaptorCodesysExample
  NewEagleDR10 (CODESYS Control for Linux ARM64 SL)
  NewEagle_HPC_Codesys (CODESYS Control for Linux ARM64 SL)
  New_Eagle_Consulting_Tricore_Embedded_M2_501_K (New Eagle Consulting Tricore-Embedded M2-501 K)
  New_Eagle_Consulting_Tricore_Embedded_M2_702 (New Eagle Consulting Tricore-Embedded M2-702)
    PLC Logic
      Application
        GVL
        Library Manager
        ControllerMockUp (PRG)
        PLC_CFC (PRG)
        POU (PRG)
        POU_Ladder (PRG)
        POU_SequencalFunctionChart (PRG)
        POU_StructuredText (PRG)
        Task Configuration
        Trace
    CANbus (CANbus)
      CANopen_Manager (CANopen_Manager)
        _502_CANopen_Default_Slave (502 CANopen Default Slave)
        EPOS4_50_5 (EPOS4 50/5)
          SM3_Drive_CAN_Maxon_EPOS4 (SM3_Drive_CAN_Maxon_EPOS4)
    M702_LocalIO (M702.LocalIO)
    SoftMotion General Axis Pool (SoftMotion General Axis Pool)
```

```
PROGRAM PLC_CFC
VAR
    antiWindUp : Ctrl.AntiWindUp_Clamping := (lrMinValue:=-400, lrMaxValue:=400);
    integrator : Ctrl.Integrator_TrapezoidApproximation := (itfAntiWindUp:=antiWindUp);
    controller : Ctrl.Controller_PI := (itfIntegrator:=integrator);

    motor : Ctrl.PT1_Trapezoid;

    lrKp : LREAL := 18;
    lrKi : LREAL := 60;
    plant: Ctrl.PT1_Trapezoid;
END_VAR

```

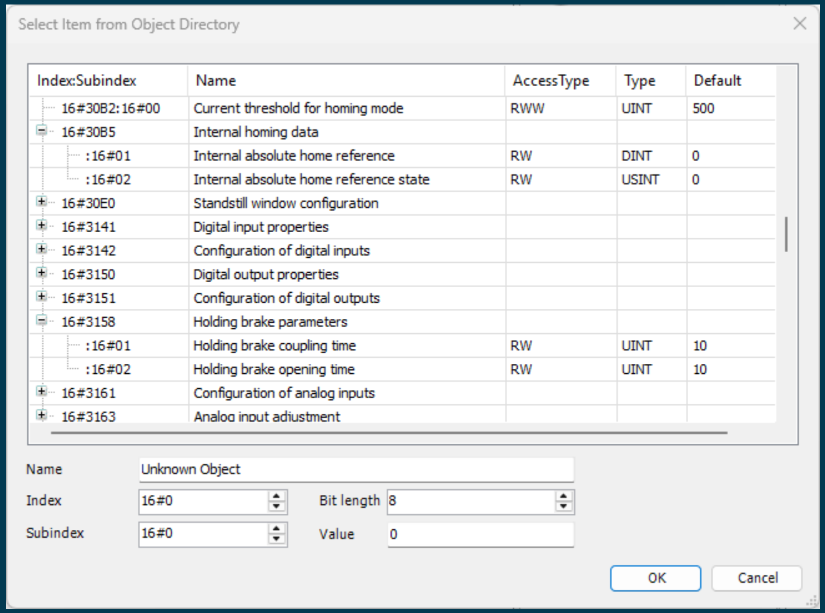
© New Eagle. All Rights Reserved

Distributed Control Node

Third Pary Device



```
New_Eagle_Consulting_Tricore_Embedded_M2_301_R (New Eagle Consulting Tricore-Embedded M2-301_R)
New_Eagle_Consulting_Tricore_Embedded_M2_702 (New Eagle Consulting Tricore-Embedded M2-702)
    PLC Logic
        Application
    CANbus (CANbus)
        CANopen_Manager (CANopen_Manager)
            _502_CANopen_Default_Slave (502 CANopen Default Slave)
            EPOS4_50_5 (EPOS4 50/5)
                SM3_Drive_CAN_Maxon_EPOS4 (SM3_Drive_CAN_Maxon_EPOS4)
    M702_LocalIO (M702.LocalIO)
    SoftMotion General Axis Pool (SoftMotion General Axis Pool)
```



Select Item from Object Directory

Index:Subindex	Name	AccessType	Type	Default
16#30B2:16#00	Current threshold for homing mode	RWW	UINT	500
16#30B5	Internal homing data			
:16#01	Internal absolute home reference	RW	DINT	0
:16#02	Internal absolute home reference state	RW	USINT	0
16#30E0	Standstill window configuration			
16#3141	Digital input properties			
16#3142	Configuration of digital inputs			
16#3150	Digital output properties			
16#3151	Configuration of digital outputs			
16#3158	Holding brake parameters			
:16#01	Holding brake coupling time	RW	UINT	10
:16#02	Holding brake opening time	RW	UINT	10
16#3161	Configuration of analog inputs			
16#3163	Analog input adjustment			

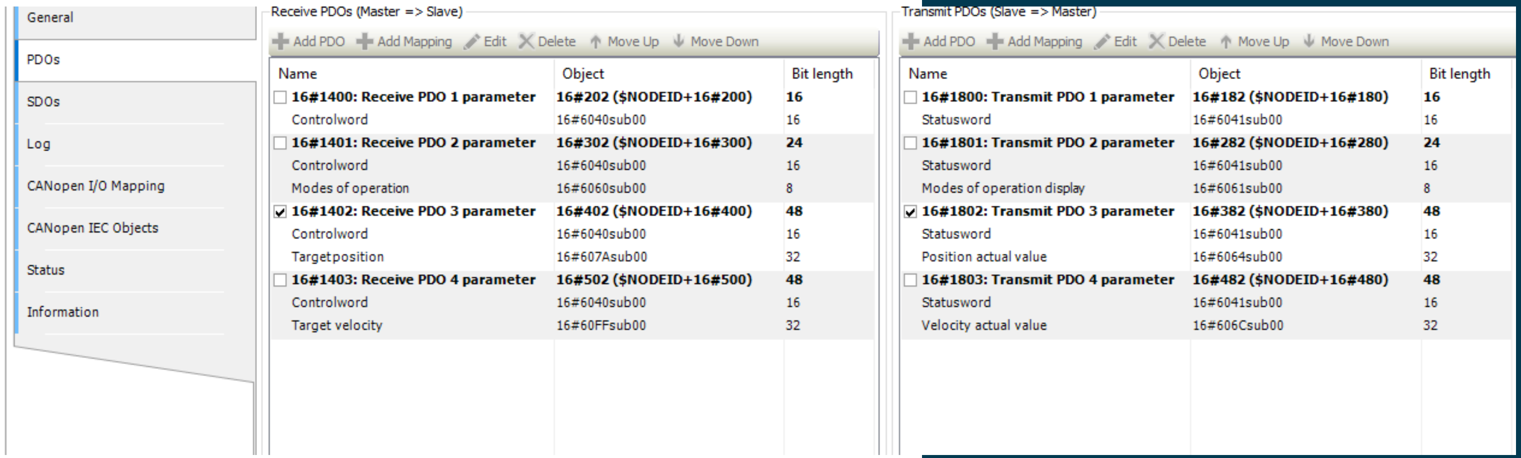
Name: Unknown Object

Index: 16#0

Bit length: 8

Subindex: 16#0

Value: 0



Receive PDOs (Master => Slave)

Name	Object	Bit length
□ **16#1400: Receive PDO 1 parameter**	**16#202 ($NODEID+16#200)**	**16**
Controlword	16#6040sub00	16
□ **16#1401: Receive PDO 2 parameter**	**16#302 ($NODEID+16#300)**	**24**
Controlword	16#6040sub00	16
Modes of operation	16#6060sub00	8
☑ **16#1402: Receive PDO 3 parameter**	**16#402 ($NODEID+16#400)**	**48**
Controlword	16#6040sub00	16
Targetposition	16#607Asub00	32
□ **16#1403: Receive PDO 4 parameter**	**16#502 ($NODEID+16#500)**	**48**
Controlword	16#6040sub00	16
Target velocity	16#60FFsub00	32

Transmit PDOs (Slave => Master)

Name	Object	Bit length
□ **16#1800: Transmit PDO 1 parameter**	**16#182 ($NODEID+16#180)**	**16**
Statusword	16#6041sub00	16
□ **16#1801: Transmit PDO 2 parameter**	**16#282 ($NODEID+16#280)**	**24**
Statusword	16#6041sub00	16
Modes of operation display	16#6061sub00	8
☑ **16#1802: Transmit PDO 3 parameter**	**16#382 ($NODEID+16#380)**	**48**
Statusword	16#6041sub00	16
Position actual value	16#6064sub00	32
□ **16#1803: Transmit PDO 4 parameter**	**16#482 ($NODEID+16#480)**	**48**
Statusword	16#6041sub00	16
Velocity actual value	16#606Csub00	32

© New Eagle. All Rights Reserved

RCM112 - Raptor

System Function

- Function Consume TPDO and NMT state from all nodes
- Produce CANopen RPDO
- CANopen SYNC

Raptor Implementation

- Raptor CAN Blockset based on DBC
- Does not require CANopen Master or Manager functionality

Default CODESYS GCM Behavior

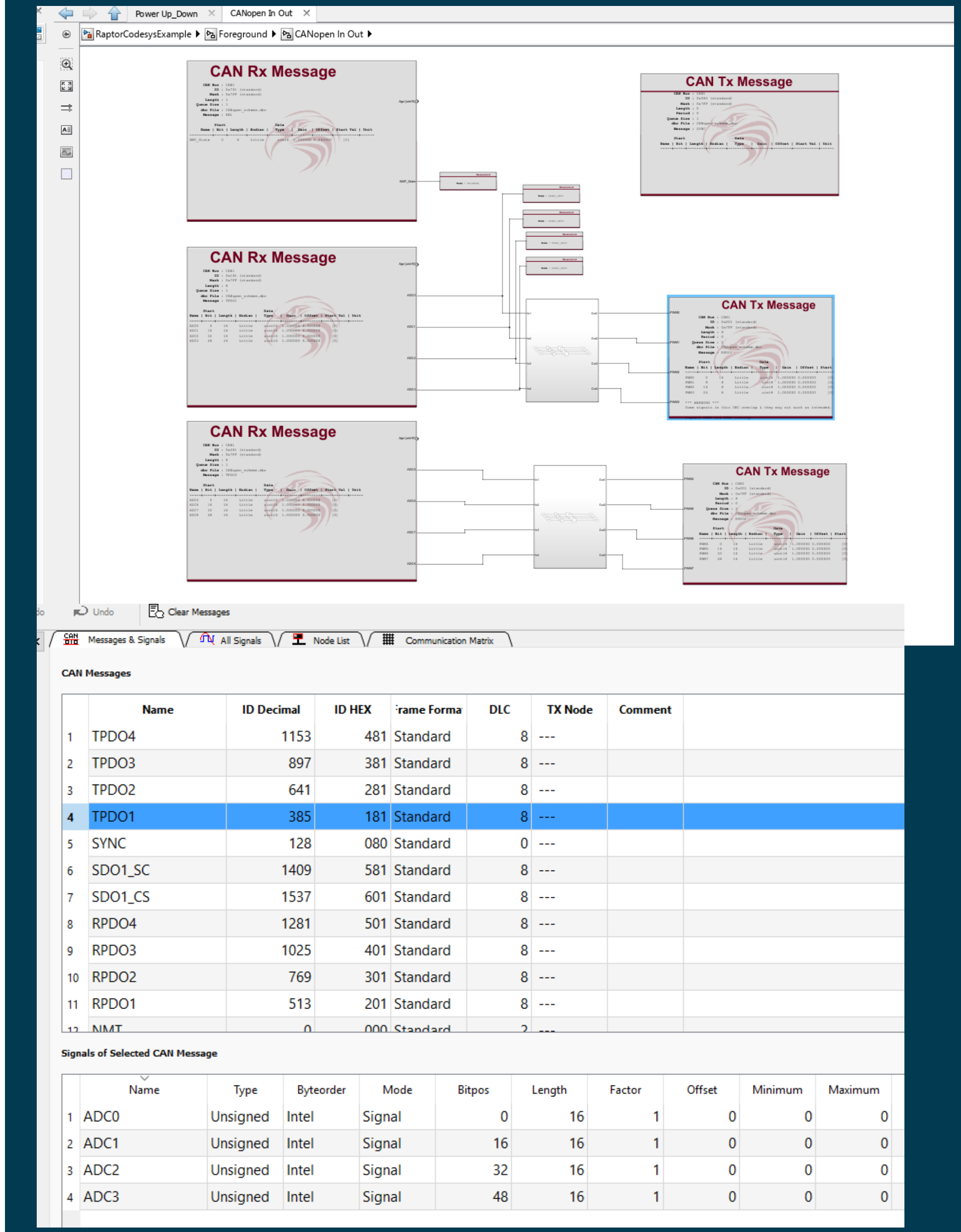
- Default CODESYS Module EDS / DBC File provided for GCM controllers
- CANopen is application layer scheme for CAN ID
- PDO exchange do not need CANopen stack with PDO mapping is static

Extend CODESYS GCM Behavior

- Extend Object Dictionary and local logic in GCM CODESYS
- Recreate EDS and modify DBC to match
- Tools available for CANopen EDS/DCF -> DBC

CANopen System

- CANopen Manager outside of scope
- Arrive at a fixed CAN network at runtime that Raptor can interface directly with.



CAN Messages

	Name	ID Decimal	ID HEX	rame Forma	DLC	TX Node	Comment
1	TPDO4	1153	481	Standard	8	---	
2	TPDO3	897	381	Standard	8	---	
3	TPDO2	641	281	Standard	8	---	
4	TPDO1	385	181	Standard	8	---	
5	SYNC	128	080	Standard	0	---	
6	SDO1_SC	1409	581	Standard	8	---	
7	SDO1_CS	1537	601	Standard	8	---	
8	RPDO4	1281	501	Standard	8	---	
9	RPDO3	1025	401	Standard	8	---	
10	RPDO2	769	301	Standard	8	---	
11	RPDO1	513	201	Standard	8	---	
12	NMT	0	000	Standard	2	---	

Signals of Selected CAN Message

	Name	Type	Byteorder	Mode	Bitpos	Length	Factor	Offset	Minimum	Maximum
1	ADC0	Unsigned	Intel	Signal	0	16	1	0	0	0
2	ADC1	Unsigned	Intel	Signal	16	16	1	0	0	0
3	ADC2	Unsigned	Intel	Signal	32	16	1	0	0	0
4	ADC3	Unsigned	Intel	Signal	48	16	1	0	0	0

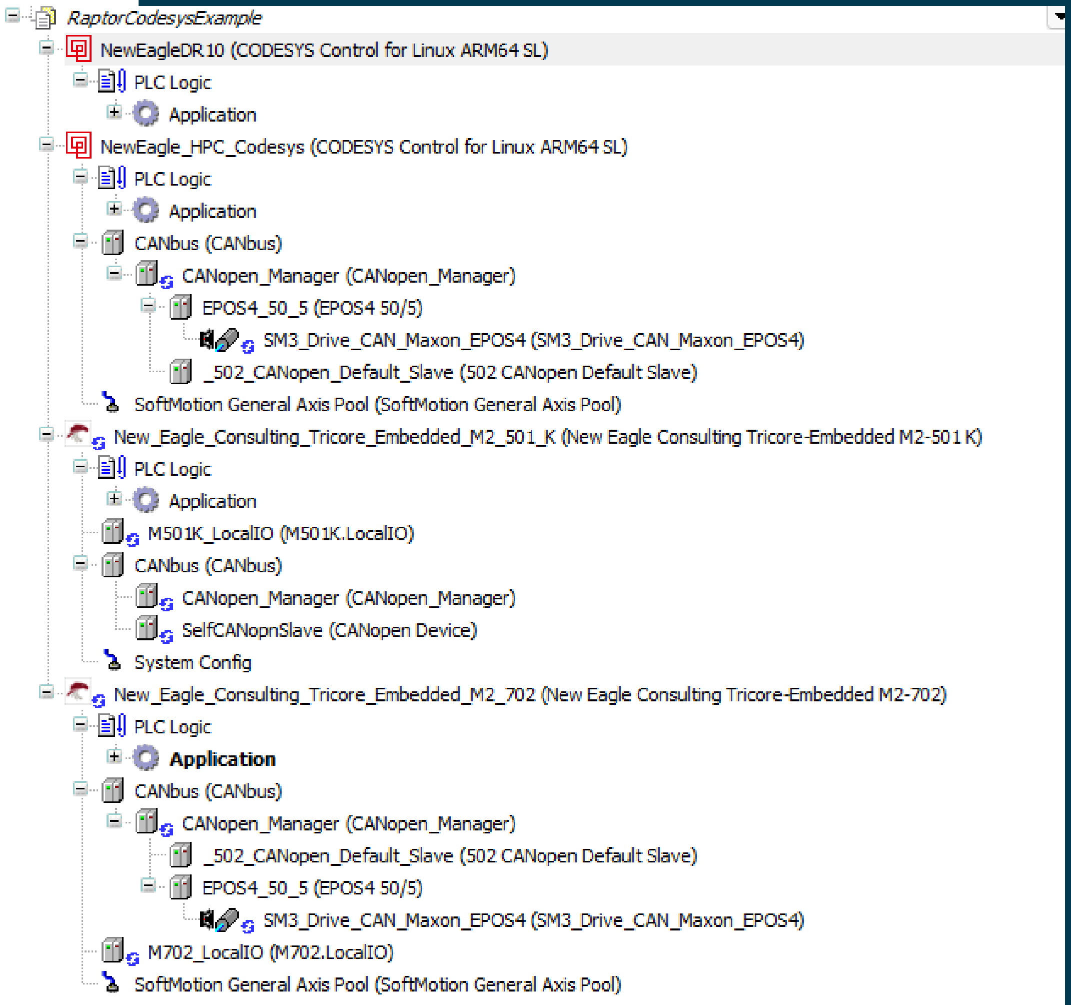
Distributed Control Node

© New Eagle. All Rights Reserved

System Level Project

Multiple Target Project

- Nest partner devices under target scope to enable field bus data to application scope mappings
- Application logic is drag and drop across targets including those of different architectures!
- Offline mode is target context aware but allows for concurrent connections to all targets
- Support for optional nodes and node identity check

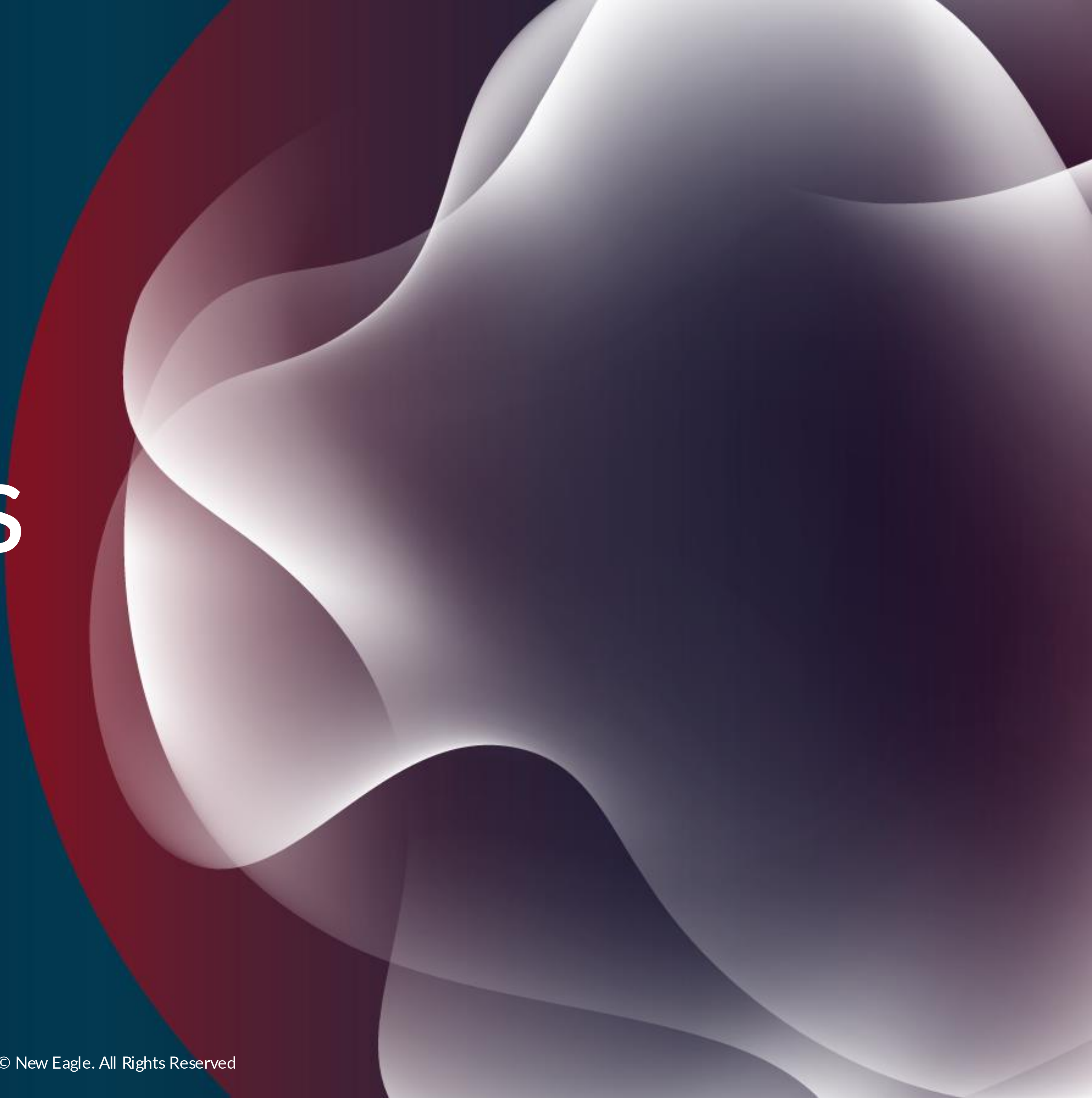


```
RaptorCodesysExample
  NewEagleDR10 (CODESYS Control for Linux ARM64 SL)
    PLC Logic
      Application
  NewEagle_HPC_Codesys (CODESYS Control for Linux ARM64 SL)
    PLC Logic
      Application
    CANbus (CANbus)
      CANopen_Manager (CANopen_Manager)
        EPOS4_50_5 (EPOS4 50/5)
          SM3_Drive_CAN_Maxon_EPOS4 (SM3_Drive_CAN_Maxon_EPOS4)
        _502_CANopen_Default_Slave (502 CANopen Default Slave)
    SoftMotion General Axis Pool (SoftMotion General Axis Pool)
  New_Eagle_Consulting_Tricore_Embedded_M2_501_K (New Eagle Consulting Tricore-Embedded M2-501 K)
    PLC Logic
      Application
    M501K_LocalIO (M501K.LocalIO)
    CANbus (CANbus)
      CANopen_Manager (CANopen_Manager)
      SelfCANopnSlave (CANopen Device)
    System Config
  New_Eagle_Consulting_Tricore_Embedded_M2_702 (New Eagle Consulting Tricore-Embedded M2-702)
    PLC Logic
      Application
    CANbus (CANbus)
      CANopen_Manager (CANopen_Manager)
        _502_CANopen_Default_Slave (502 CANopen Default Slave)
        EPOS4_50_5 (EPOS4 50/5)
          SM3_Drive_CAN_Maxon_EPOS4 (SM3_Drive_CAN_Maxon_EPOS4)
    M702_LocalIO (M702.LocalIO)
    SoftMotion General Axis Pool (SoftMotion General Axis Pool)
```

© New Eagle. All Rights Reserved

Raptor + CODESYS

Feature Roadmap



© New Eagle. All Rights Reserved

CODESYS ECU Roadmap

Available Now!

Planned

Customer Driven

CODESYS v3 Control Runtime

Core Runtime Deployed

- Cyclic tasks and all IDE features except breakpoint
- CANopen Master/Slave functionality via IEC Library and target license
- IDE configurable CANopen PDO mappings and J1939 PGN
- Basic ethernet connectivity

Ethernet Fieldbus

Expand on Ethernet Fieldbus support based on customer feedback

- EtherCAT Master, EtherNET/IP and several other industrial protocols available with modified licensing
- Understand Customer Needs
- Solutions for system integration into Automotive Ethernet and Mobility machinery

Safety

Path to SIL2

- Deploy SIL2 CODESYS Runtime
- CANopen Safety
- No hardware changes required
- Safe IO Partner ECU for High Compute Platform when running Virtual Safe PLC

Advanced Features

Deploy CODESYS Redundancy, Motion or other application based licensed features as needed.

- Advanced features as redundancy and motion control planning are available but require integration and license in ECU runtime
- Customer demand will drive these features.

© New Eagle. All Rights Reserved

CODESYS Nest Resources

ECU Target Files

Downloadable Resources

- Target Description XML files for IDE
- IO Description Files
- Hello world project templates
- Default CANopen Slave Master CODESYS Project with EDS File for Master and DBC for Raptor collaboration

Application Examples

- Basic Input Control Output Example
- CANopen Slave with supervisor interface and local IO control logic
- Demo application for Raptor + CODESYS
- Advanced API for constant current drivers

How To Articles

Illustrative Wiki article base is growing

- IDE connection over CAN/Ethernet
- Startup Hello world!
- Logging and debugging support
- Deploy and update applications

© New Eagle. All Rights Reserved

Presented by:

Parker Mosman,
Chief Product Officer
New Eagle

theNEST

Learn More in the NEST

Scan the QR code to access related resources, technical content, and additional insights.

© New Eagle. All Rights Reserved

new eagle

RAPTOR®
INNOVATION
SUMMIT 2025

© New Eagle. All Rights Reserved