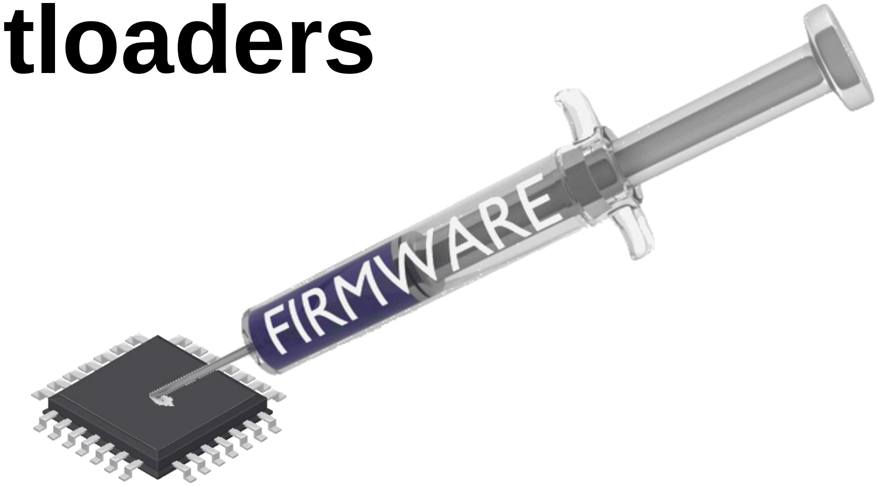
EMBEDDED
SYSTEMS
ACADEMY

CAN-FD stack porting and secure bootloaders



presented by

Olaf Pfeiffer
Embedded Systems Academy

www.esacademy.com

NXP

Webinar Contents

- □ **Review of CAN-FD basics (from part 1)**
- □ **Implications for Higher-Layer Protocols**
  - CANopen, J1939 and others
- □ **Implications for Bootloading**
- □ **Security risks, ransomware**
- □ **ESAcademy’s Secure Bootloader**
  - Protection levels
  - CANcrypt basics
  - Key management
  - Bootloader operation
  - LPC546xx implementation

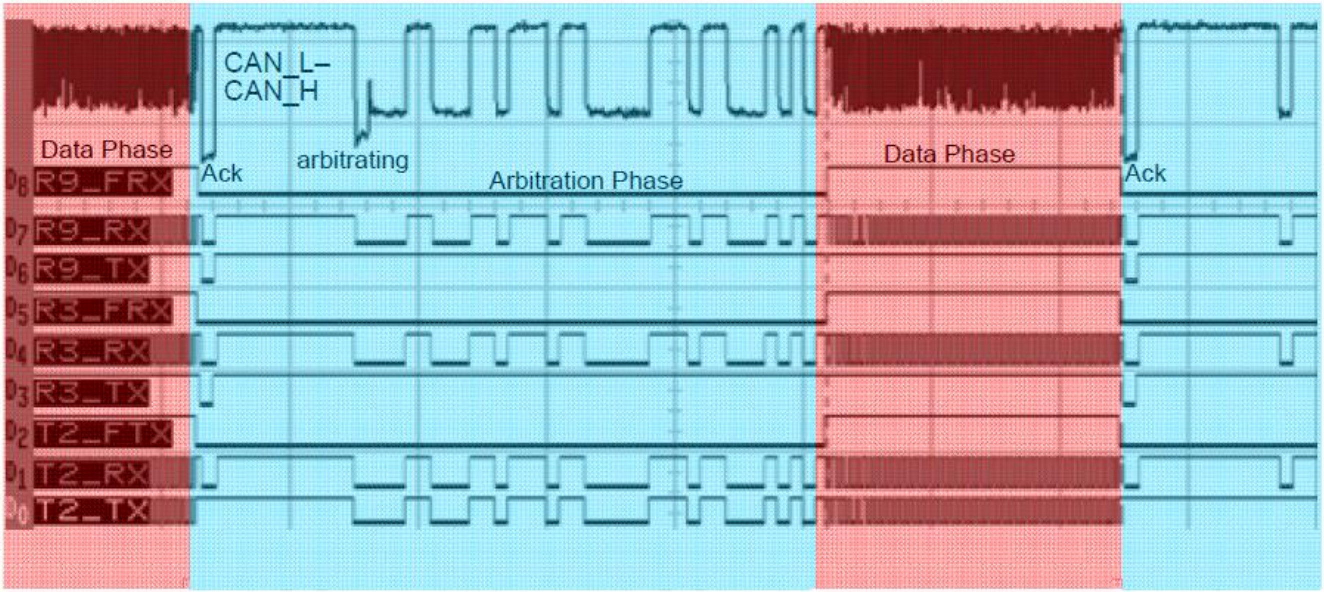
Embedded Systems Academy

- ❑ **Founded 1999**
- ❑ **Services**
  - Consulting
  - Training
- ❑ **Firmware**
  - CANopen stack
  - J1939 stack
  - Bootloader
- ❑ **Software**
  - NXP's Flash Magic
  - CANopen Magic
  - CANcrypt
- ❑ **Participate in CANopen standardization**

Blog: www.esacademy.com/blog

www.flashmagictool.com
www.canopenmagic.com
www.cancrypt.eu

EMBEDDED
SYSTEMS
ACADEMY



Review from part I of this webinar

FD: Flexible Data(rate)

CAN-FD BASICS

Differences between CAN and CAN-FD

❑ **Mixed bitrates**

- “Nominal rate” for control (arbitration, control, ACK)
- “Data rate” (multiple of nominal) for data field and CRC
  - Limited by transceivers in practice
  - Need FD-compliant transceivers above 1Mbps

❑ **More data per frame**

- Up to 64 bytes instead of 8
- Allows for higher throughput

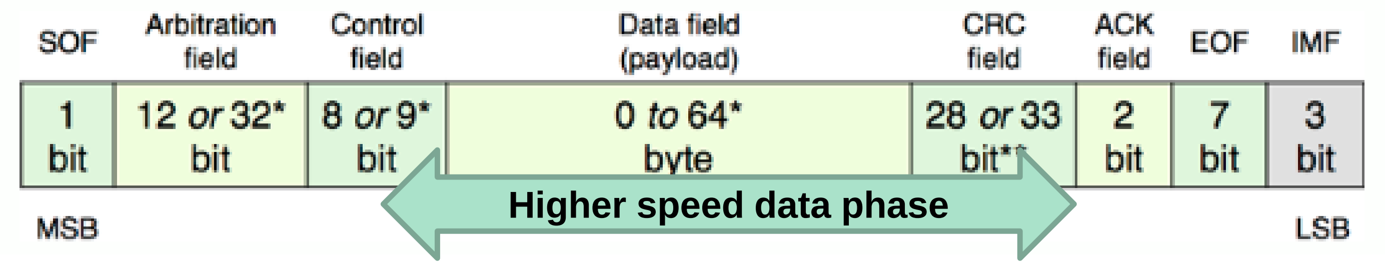
❑ **Bus topology and wiring stays the same**

- Same networking costs
- More sensitive on higher rates

❑ **NOTE: if CAN-FD is enabled,
ALL devices connected must support CAN-FD**

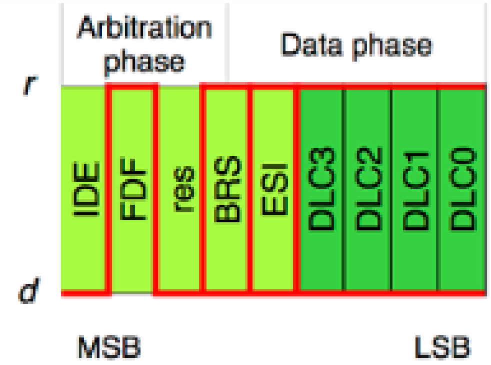
- Exception: CAN-FD “ignoring” transceivers

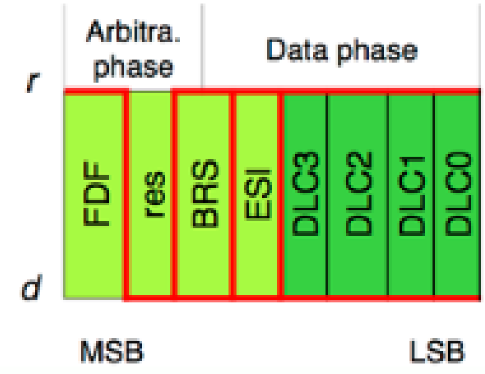
CAN-FD Message Frames

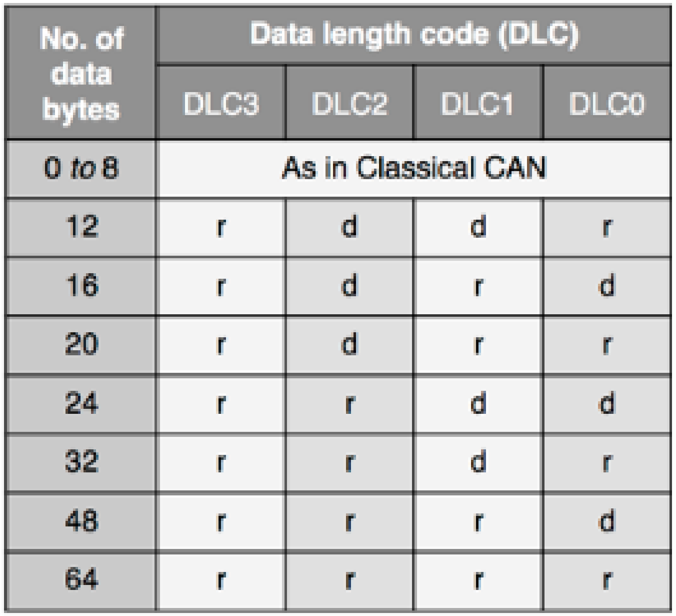


SOF	Arbitration field	Control field	Data field (payload)	CRC field	ACK field	EOF	IMF
1 bit	12 *or* 32* bit	8 *or* 9* bit	0 *to* 64* byte	28 *or* 33 bit**	2 bit	7 bit	3 bit

Diagrams © CiA







No. of data bytes	Data length code (DLC)				
	DLC3	DLC2	DLC1	DLC0	
0 *to* 8	As in Classical CAN				0-8
12	r	d	d	r	9
16	r	d	r	d	10
20	r	d	r	r	11
24	r	r	d	d	12
32	r	r	d	r	13
48	r	r	r	d	14
64	r	r	r	r	15

Higher Layer Protocols like CANopen, J1939, others

PROTOCOL IMPLICATIONS

It‘s a hard transition

- ❑ **CAN and CAN-FD can not easily be mixed**
  - A classic CAN controller not capable of CAN-FD will destroy CAN-FD messages with error frames
- ❑ **If CAN-FD is enabled, all participants must support it**
- ❑ **Therefore higher layer protocols do not neccesarily have to be backward compatible**
  - There is no „mixed“ operation, just either / or
- ❑ **First step is to re-pack pre-defined data messages**
  - Now up to 64 bytes (instead of 8) available
- ❑ **This is work in progress...**

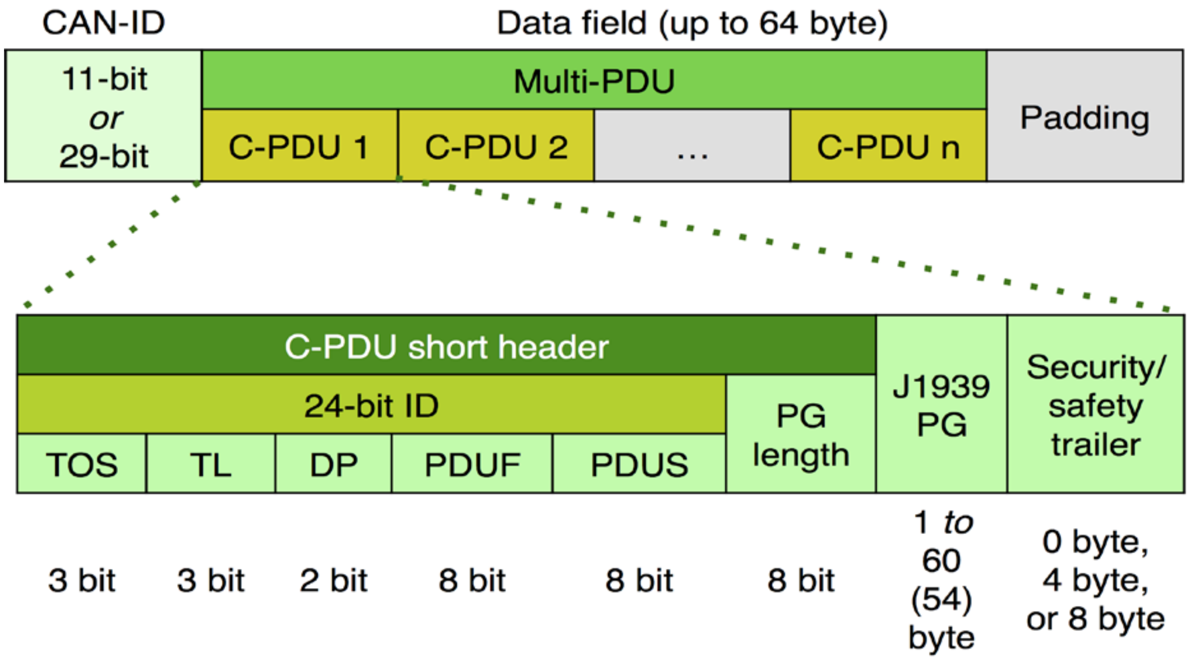
CANopen-FD

- ❑ **CANopen-FD is still under development**
- ❑ **First demonstrators have been shown**
- ❑ **Support of 64byte message length for „PDO“**
  - Process data objects can now contain more data
  - As a result less CAN-IDs are required per node
- ❑ **New transfer mode „USDO“ instead of „SDO“**
  - Universal Service Data Object
    - Request / Response communication
    - Fully meshed (every device can do this)
    - Any size (segmentation included)
    - Broadcast

J1939 by SAE

(Truck Bus Control And Communications Network Committee)

- **CAN-in-Automation (CiA) members have mapped SAE's J1939 application profile to the CAN FD data link layer**
  - Corresponding CiA 602-2 specification to be released



CAN-ID

Data field (up to 64 byte)

11-bit *or* 29-bit

Multi-PDU

C-PDU 1

C-PDU 2

…

C-PDU n

Padding

C-PDU short header

24-bit ID

TOS

TL

DP

PDUF

PDUS

PG length

J1939 PG

Security/ safety trailer

3 bit

3 bit

2 bit

8 bit

8 bit

8 bit

1 *to* 60 (54) byte

0 byte, 4 byte, or 8 byte

Legend

C-PDU (Contained PDU)

DP (Data Page)

FBFF (FD base frame format)

FEFF (FD extended frame format)

PDUF (PDU format)

PDUS (PDU specific)

PG (Parameter Group)

TL (trailer length)

TOS (Type of Service)

Diagrams © CiA

How to upgrade exisiting CAN code

- ❑ **If a 3rd party communication stack is used, upgrading to CAN-FD should be done by developers of stack**
- ❑ **If properly done, should be possible to do with minimal changes to application interface**
- ❑ **ESAcademy‘s Micro CANopen Example:**
  - All parameters and all data communicated is in an object dictionary (kind of look up table)
  - API addresses Objects in this dictionary, then application does not need to make any modifications.
    - Unless complete new features are used
    - Example: mass broadcast

EMBEDDED SYSTEMS ACADEMY

Code updates via CAN or CAN-FD

BOOTLOADER IMPLICATIONS

It‘s about code size and update time

- ❑ **Speeding up code updates was one of the driving factors behind the development of CAN-FD**
  - Tendancy is that code gets bigger
  - 128k update on classical CAN can take minutes
    - Main issue is segmentation not speed
      - For reliable transfer only segment by segment
      - Request-Response-Request-Response...
      - Over the thumb estimate: one segment per 3-5ms
- ❑ **Data transfer per segment**
  - One byte per segment used for flow control
  - Data bytes per segment
    - Classical CAN: 7 bytes
    - CAN-FD: 63 bytes
- ❑ **Conservative expectation is that code updates are executed 8 times faster**
  - 128k update on CAN-FD within 5 seconds

Compatibility issues

- ❑ **When CAN-FD is actively used**
  - All connected and powered up CAN controllers must support CAN-FD
    - Else error frames will be generated by classical CAN devices

- ❑ **An application uses classical CAN, can CAN-FD be used for code updates only?**
  - Possible if during the bootloading process all non CAN-FD capable devices are disconnected or powered down

EMBEDDED SYSTEMS ACADEMY

Code update security issues, ransomware

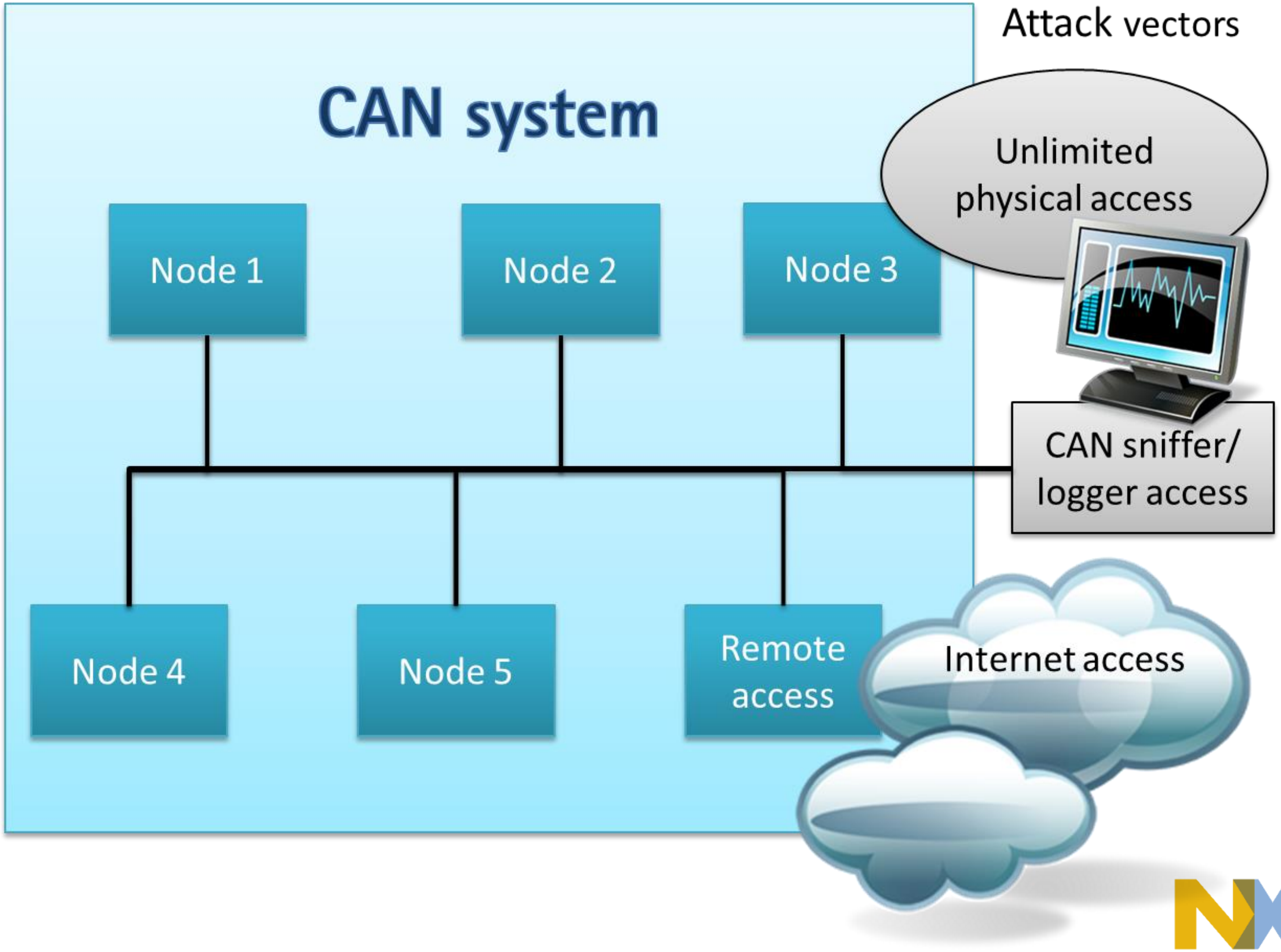
BOOTLOADER RISKS

What could possibly go wrong?

If code falls into the "wrong hands", …

- ❑ **… could it be easily copied to other devices?**
  - Programmed into a copy of the original hardware?
- ❑ **… could intellectual property be extracted?**
  - Re-engineering of code and used elsewhere?
- ❑ **… could an attacker modify it?**
  - Before it gets programmed into your device, introducing malicious code?
  - Could Embedded Ransomware lock the device?

Attacker access options to CAN or CAN-FD



EMBEDDED
SYSTEMS
ACADEMY

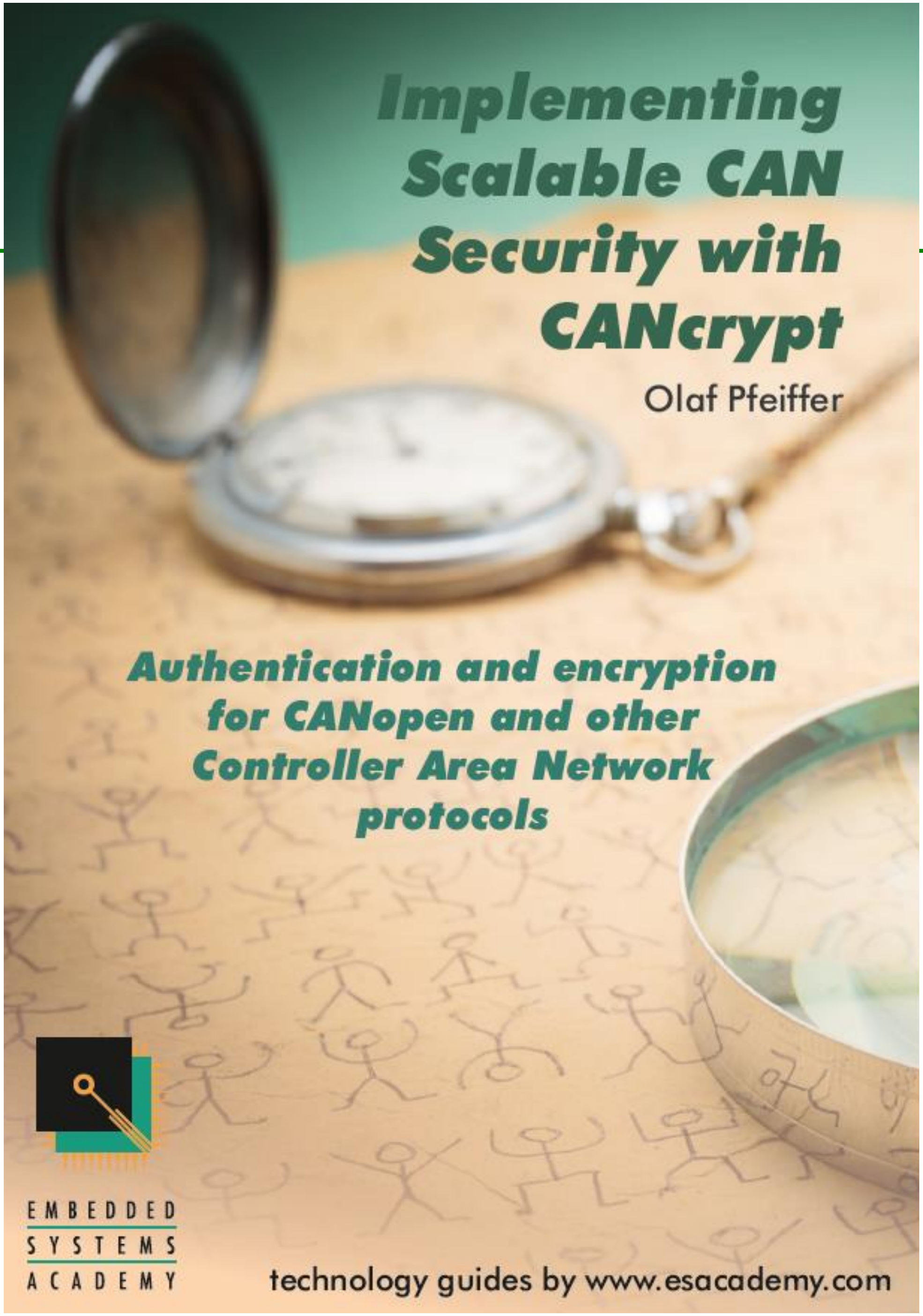
ESAcademy‘s CAN(-FD)

SECURE BOOTLOADER

CANcrypt Basics

- ❑ Security framework supporting various methods
- ❑ Secret key generation and exchange
- ❑ Pairing and grouping
- ❑ Encrypted and authenticated communication
- ❑ Minimal authentication using a secure heartbeat

Paperback ISBN: 978-0998745404
Hardcover & SW: 978-0998745411



Implementing Scalable CAN Security with CANcrypt

Olaf Pfeiffer

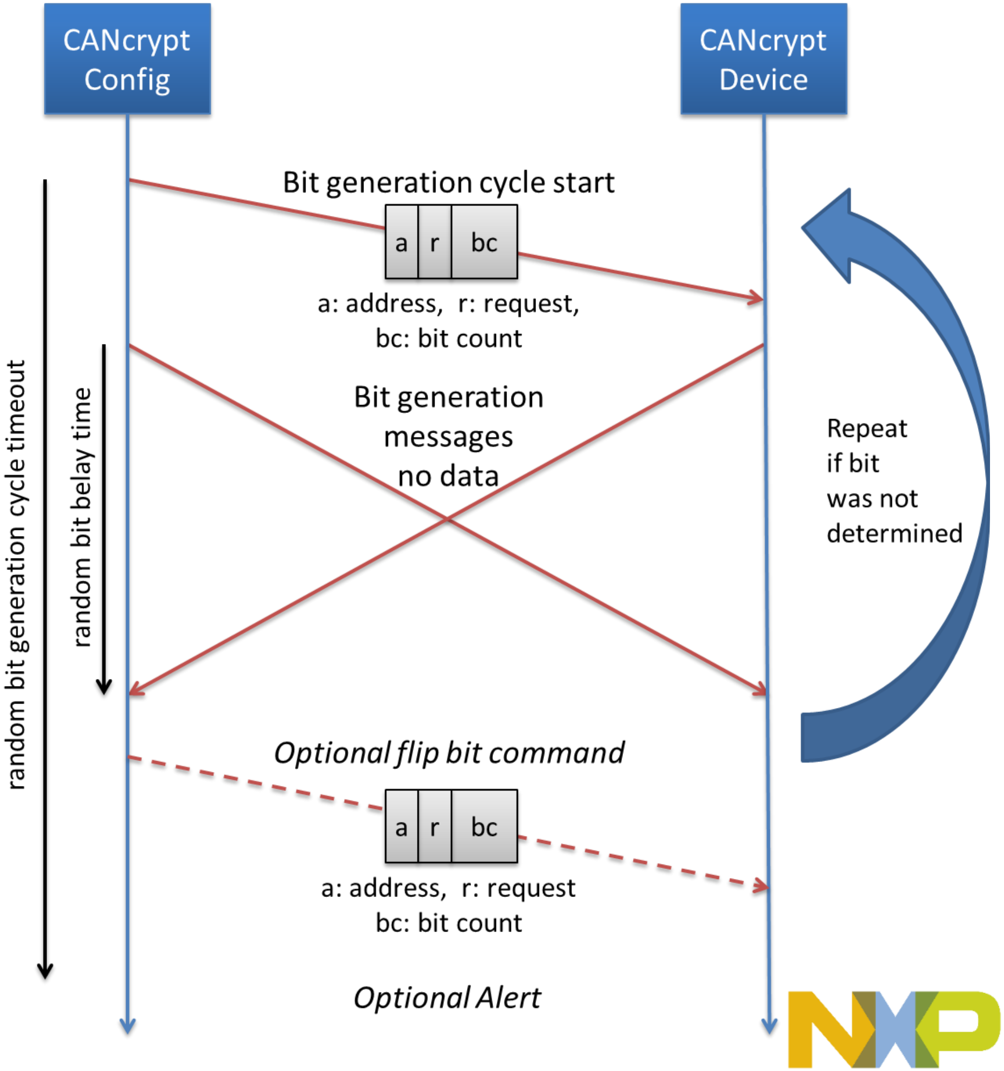
Authentication and encryption for CANopen and other Controller Area Network protocols

EMBEDDED SYSTEMS ACADEMY

technology guides by www.esacademy.com

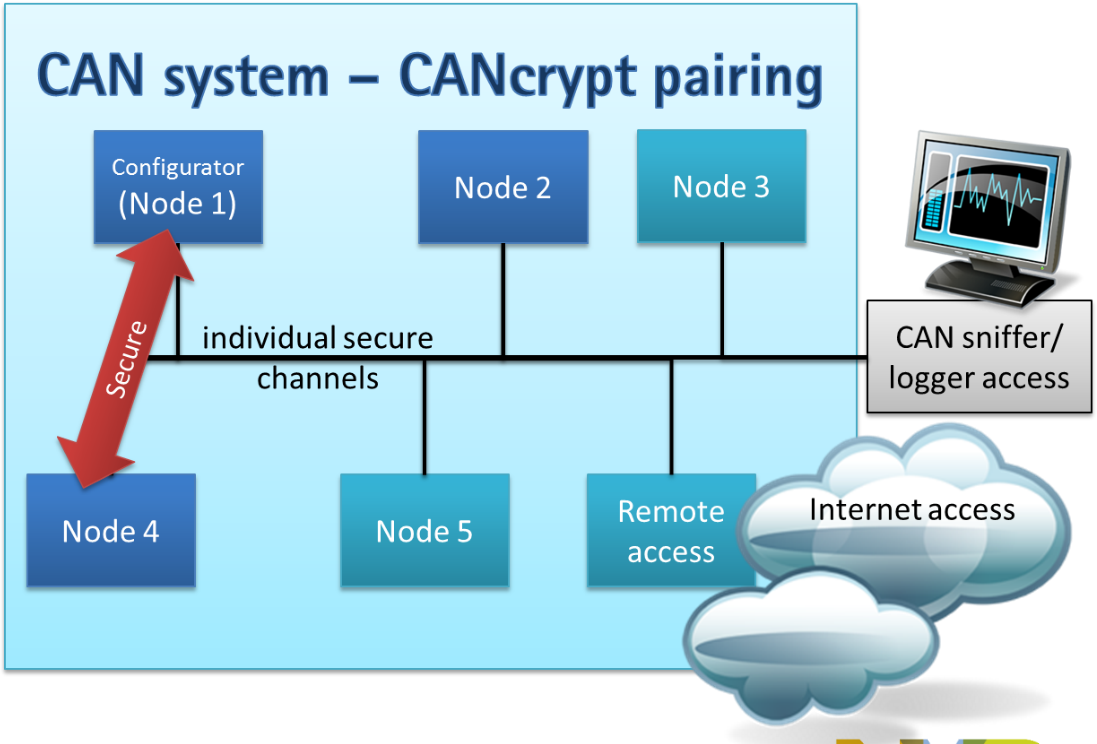
CANcrypt: Secret bit generation

- ❑ **Cycle initiated by configurator**
- ❑ **In random time window both transmit randomly X or Y**
- ❑ **If window contains XX or YY, start over**
- ❑ **If window contains XY or YX, generated bit is 1, if configurator sent X, else bit is 0**
- ❑ **256bit key in 4-6s**


CANcrypt Config
CANcrypt Device
Bit generation cycle start
a
r
bc
a: address, r: request, bc: bit count
random bit generation cycle timeout
random bit belay time
Bit generation messages no data
Repeat if bit was not determined
Optional flip bit command
a
r
bc
a: address, r: request bc: bit count
Optional Alert

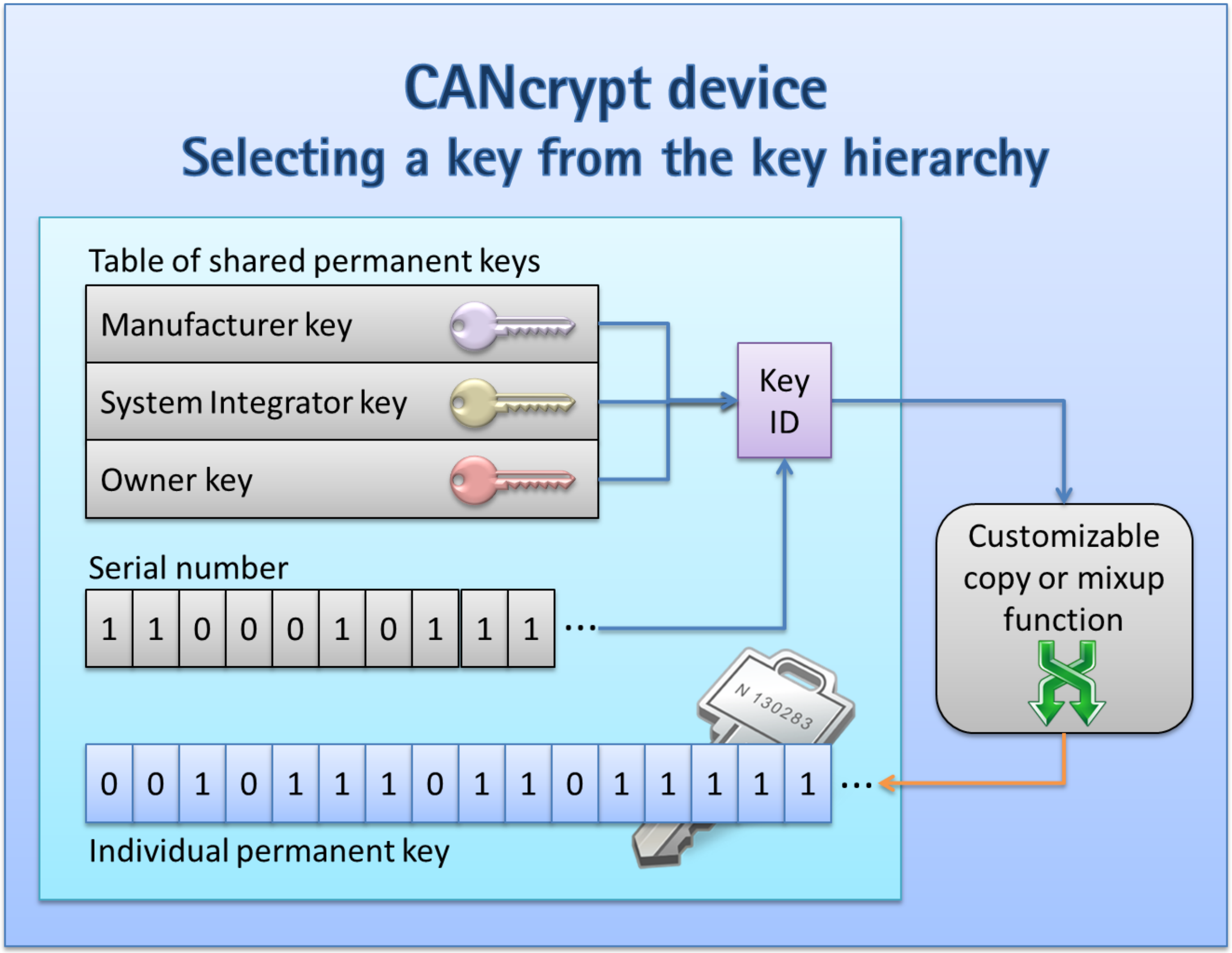
CANcrypt: Pairing and key exchange

- ❑ **Secure connection between two devices**
  - based on symmetric key
- ❑ **Initiated by configurator**
- ❑ **Uses CANcrypt bit generation cycle**
- ❑ **Intended use**
  - key generation and exchange
  - device setup or configuration
  - crucial commands like bootloader activation


CAN system – CANcrypt pairing
Configurator (Node 1)
Node 2
Node 3
Secure
individual secure channels
CAN sniffer/ logger access
Node 4
Node 5
Remote access
Internet access

CANcrypt: Key management

- ❑ **Key hierarchy**
  - symmetric
- ❑ **Different keys can have different authorities**
- ❑ **Bootloader access limited to manufacturer and system integrator**
- ❑ **Optional: combined with serial number**



CANcrypt device

Selecting a key from the key hierarchy

Table of shared permanent keys

Manufacturer key
System Integrator key
Owner key

Key ID

Serial number

1 1 0 0 0 1 0 1 1 1 …

Customizable copy or mixup function

0 0 1 0 1 1 1 0 1 1 0 1 1 1 1 1 …

Individual permanent key

CANcrypt:
Key identification

- ❑ **How can a key management system remember which key was installed where?**
- ❑ **Each key is associated with a unique 32-bit key ID assigned and stored when installing the key**

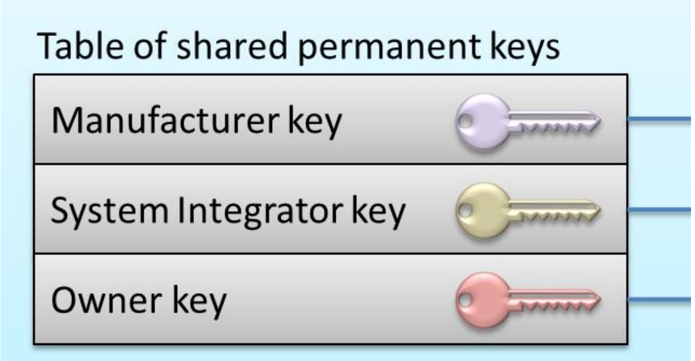


Table of shared permanent keys

Manufacturer key

System Integrator key

Owner key

Public, readable

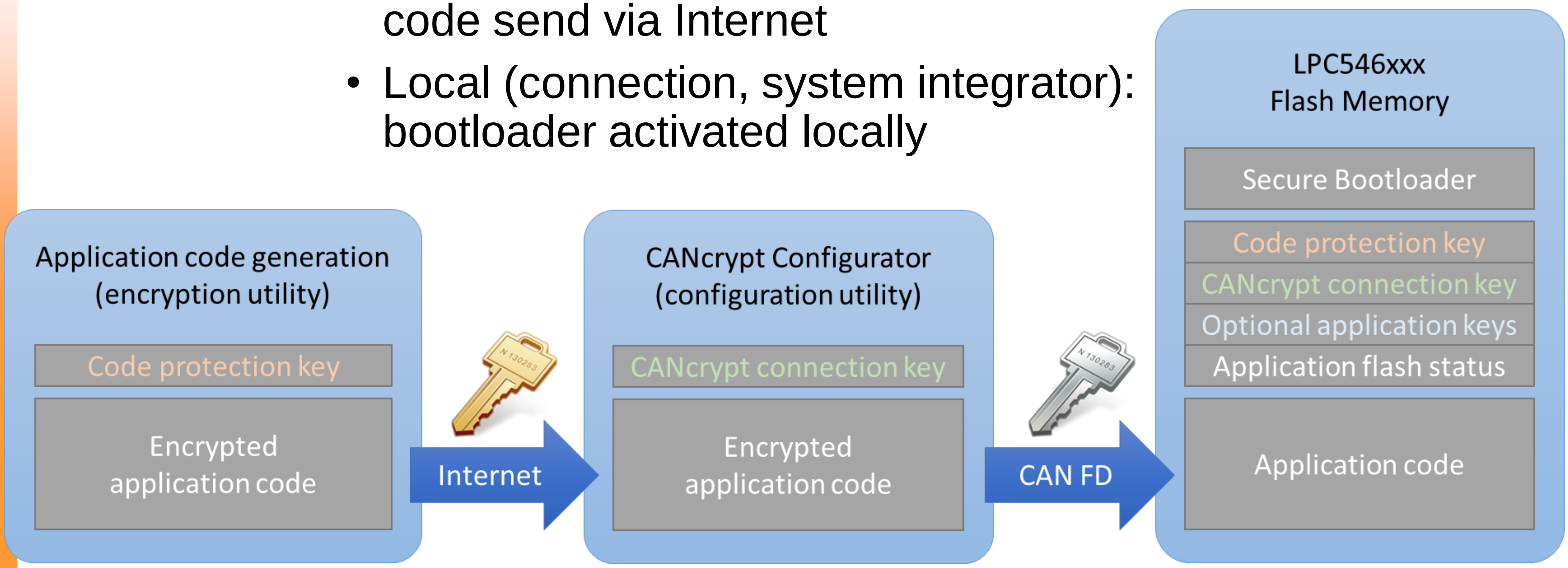
Manufacturer key ID
System Integrator key ID
Owner key ID

- ❑ **The key ID can be read at any time (public info)**
- ❑ **Service case: service utility reads public ID and then checks if it has a matching key in its database**

Security levels supported

❑ Global and Local protection

- Global (code, manufacturer):
  code send via Internet
- Local (connection, system integrator):
  bootloader activated locally



Application code generation
(encryption utility)

Code protection key

Encrypted
application code

Internet

CANcrypt Configurator
(configuration utility)

CANcrypt connection key

Encrypted
application code

CAN FD

LPC546xxx
Flash Memory

Secure Bootloader

Code protection key

CANcrypt connection key

Optional application keys

Application flash status

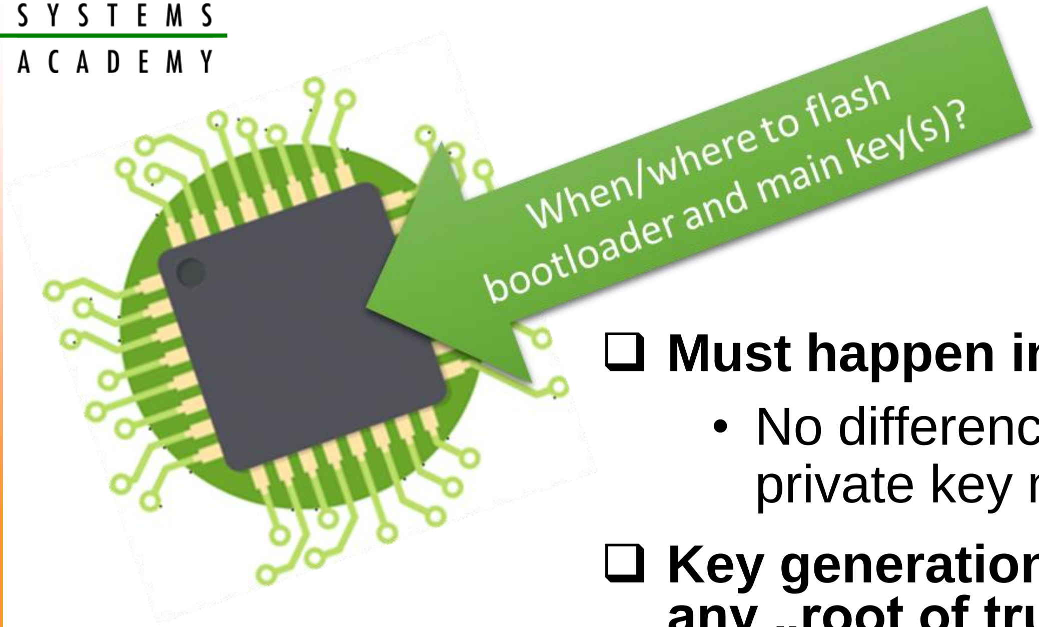
Application code

PC at manufacturer
generating new FW

PC or diagnostic tool at
system integrator

Embedded system
receiving FW update

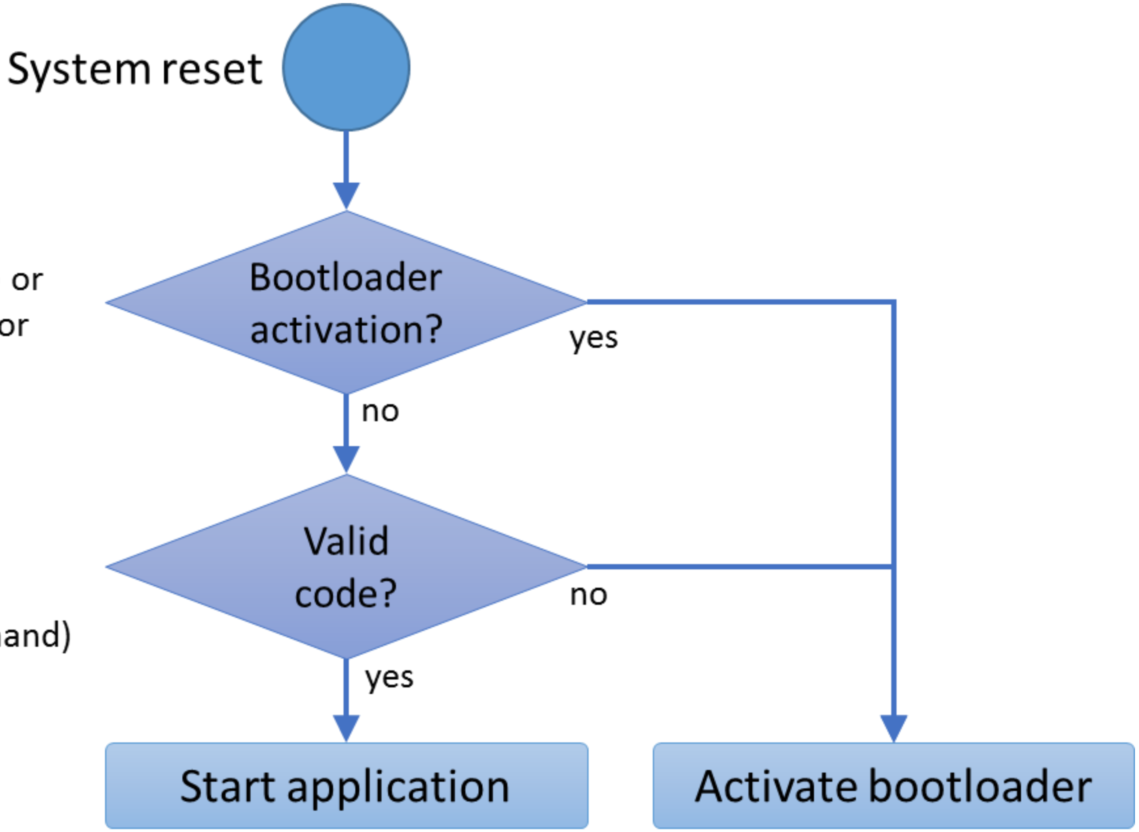
Flashing Bootloader and initial key(s)



- ❑ **Must happen in a trustworthy environment**
  - No difference to public/private key systems, private key must be protected
- ❑ **Key generation and installation is where any „root of trust“ begins**
- ❑ **Preferably in between**
  - Production
    and
  - Delivery
- ❑ **Here:**
  supported by FlashMagic utility

Secure bootloader activation

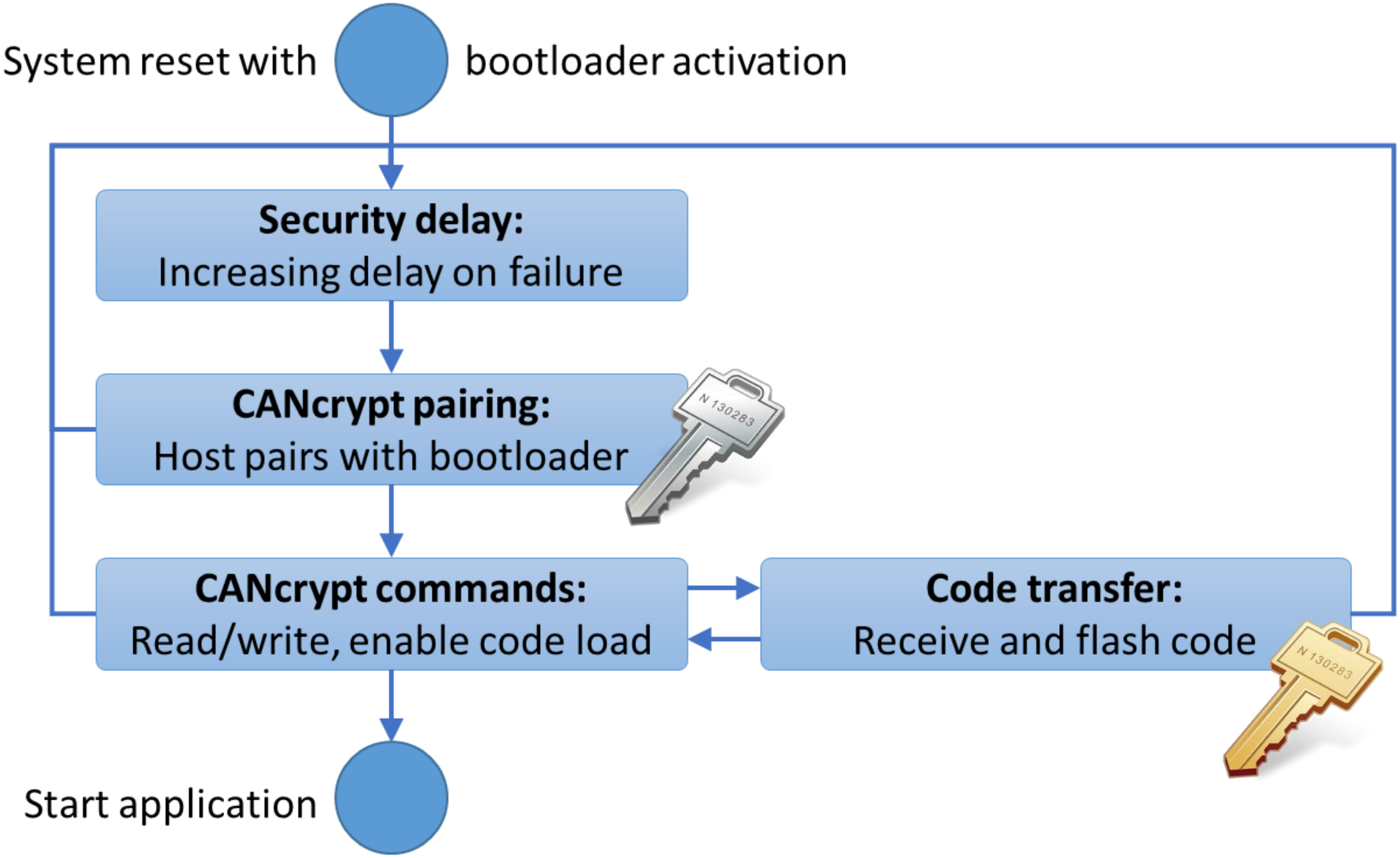
Application writes „BOOT“ code to RAM and resets



Activation code in RAM (written by application) or optional delay to wait for CANcrypt configurator

32-bit CRC match and marked as confirmed by bootloader (set by CANcrypt command)

Bootloader state machine (once it is activated)



System reset with bootloader activation

Security delay:
Increasing delay on failure

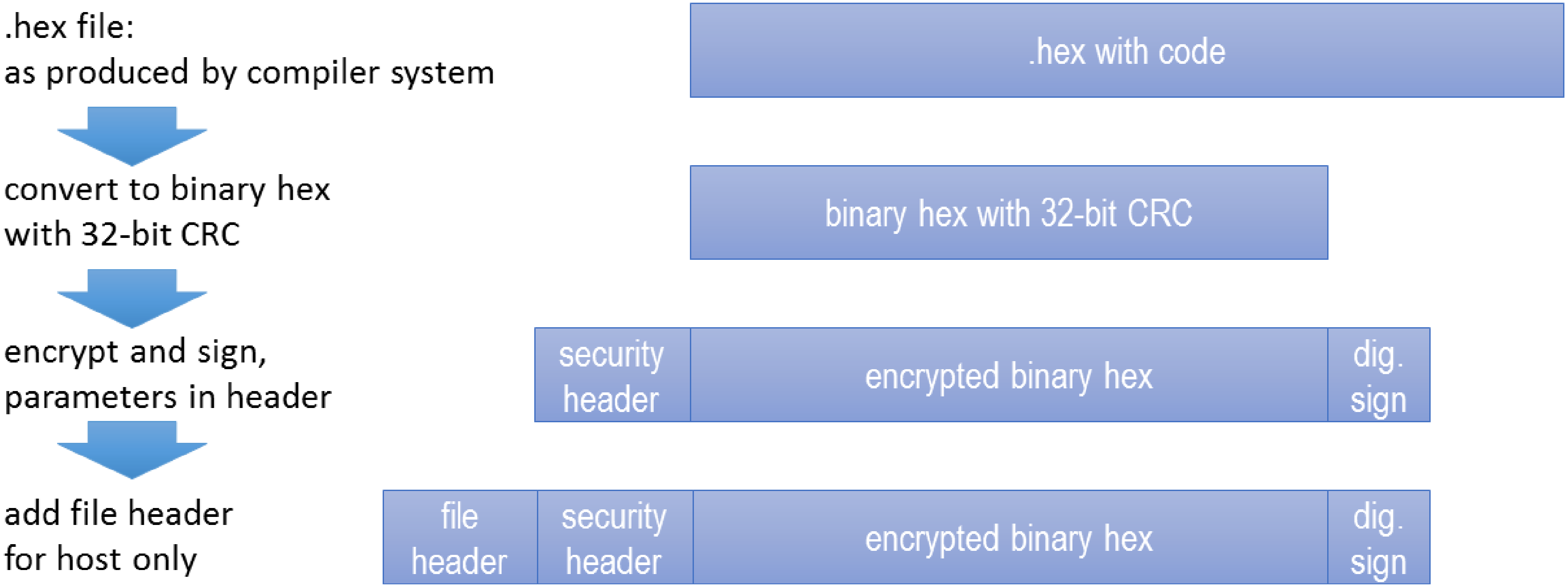
CANcrypt pairing:
Host pairs with bootloader

CANcrypt commands:
Read/write, enable code load

Code transfer:
Receive and flash code

Start application

Generating the code update file (utility provided)



.hex file:
as produced by compiler system

convert to binary hex
with 32-bit CRC

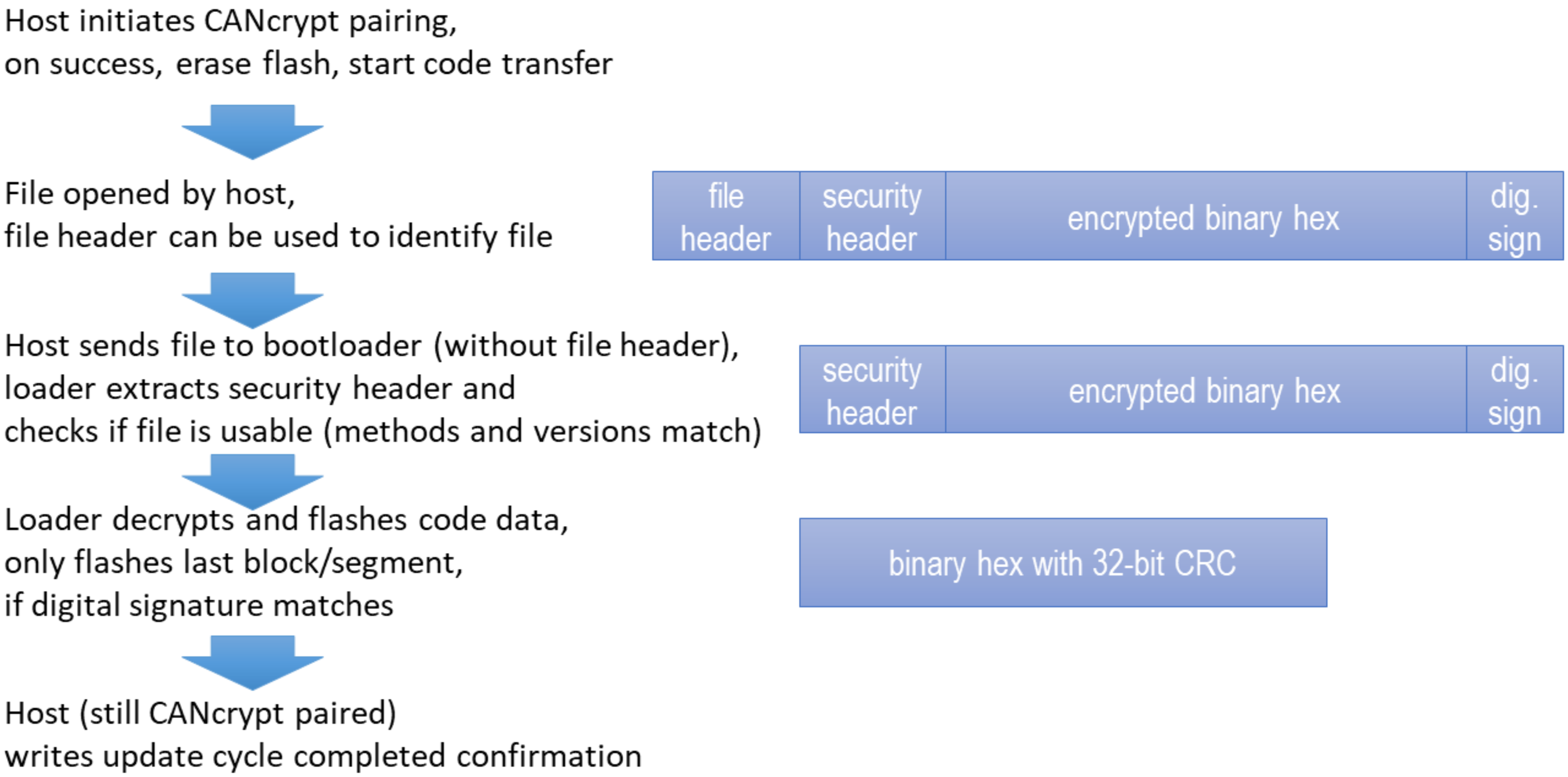
encrypt and sign,
parameters in header

add file header
for host only

Contents of the security header

- ❑ **Bootloader verison number required**
  - Ensure that bootloader matches to file
- ❑ **Firmware version number**
  - Only allow upgrades not downgrades
- ❑ **Serial number of destination chip**
  - If set, only allow to be programmed in matching device
- ❑ **Encryption method**
- ❑ **Encryption parameters**
  - key info, vectors, size
- ❑ **Signature method**
- ❑ **Signature parameters**
  - key info, vectors

Code update file processing



Host initiates CANcrypt pairing,
on success, erase flash, start code transfer

File opened by host,
file header can be used to identify file

Host sends file to bootloader (without file header),
loader extracts security header and
checks if file is usable (methods and versions match)

Loader decrypts and flashes code data,
only flashes last block/segment,
if digital signature matches

Host (still CANcrypt paired)
writes update cycle completed confirmation

EMBEDDED SYSTEMS ACADEMY

ESAcademy‘s CAN(-FD) secure bootloader

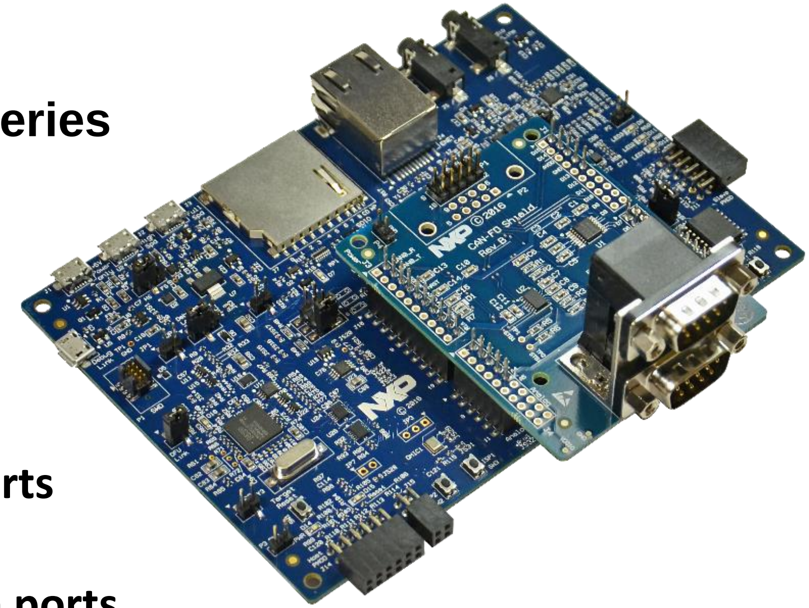
LPC54618 IMPLEMENTATION

EMBEDDED
SYSTEMS
ACADEMY

LPCXpresso54618 CAN-FD Kit

Development platform for LPC546xx Series

- ❑ **LPC54618 MCU running at 180MHz**
- ❑ **128Mb Micron SDRAM**
- ❑ **128Mb Micron quad SPI flash**
- ❑ **Built-in CMSIS-DAP/J-link debug probe**
- ❑ **Ethernet, DMIC, SD card, USB HS/FS ports**
- ❑ **Stereo audio codec**
- ❑ **Arduino UNO R3 compatible expansion ports**
- ❑ **Shield board with TJA1059 dual transceiver**
- ❑ **Supported by MCUXpresso SDK for MCUXpresso IDE, Keil and IAR tools**

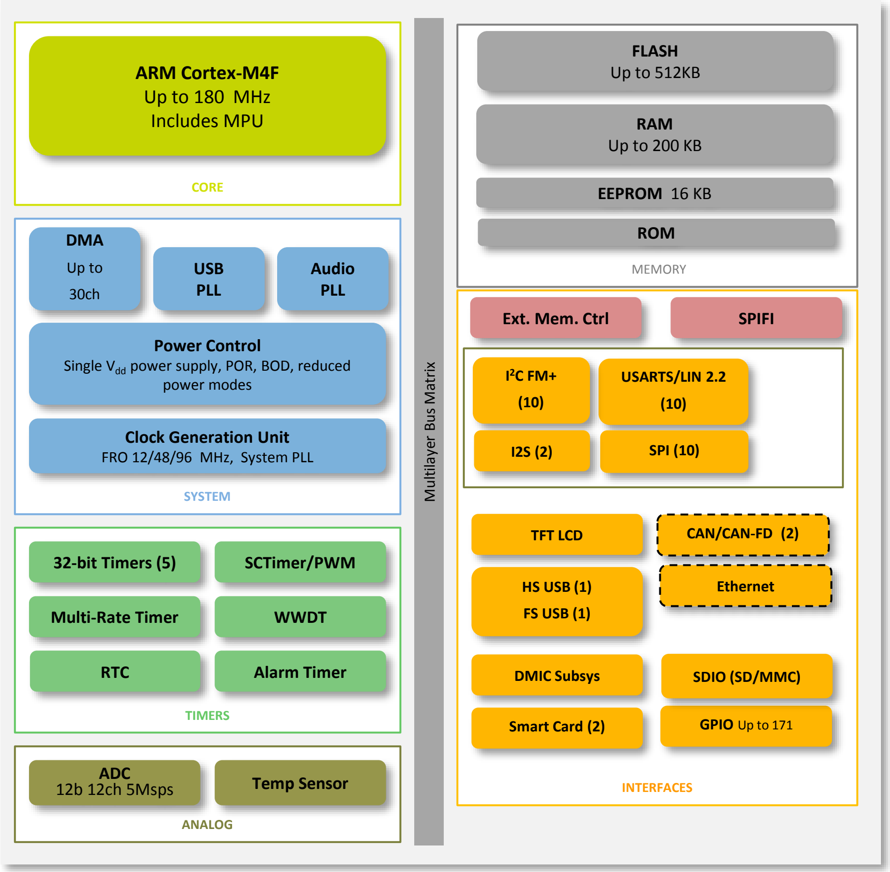


*LPCXpresso54628 now also available,
CAN-FD shield available separately

LPC546xx Block Diagram

CPU Cortex-M4F

- ❑ LPC54628 up to 220MHz
- ❑ Other parts up to 180MHz



Memory

- ❑ Up to 512 KB Flash
- ❑ Up to 200 KB RAM
- ❑ 16 KB EEPROM

Interfaces for connectivity & sensors

- ❑ Stereo DMIC subsystem
- ❑ 1x HS USB (H/D) w/ on-chip HS PHY, XTAL-less FS USB (H/D)
- ❑ 10 SPI, 10 I2C, 10 UART, 2 I2S channels (max 10 channels total)
- ❑ Graphic LCD with resolutions up to 1024x768
- ❑ Ethernet with IEEE1722 timestamp
- ❑ 2 x CAN-FD controller (LPC5461x and LPC54628)
- ❑ Quad SPI flash interface
- ❑ External Memory interface (up to 32 bits)

Packages

- ❑ LQFP208 (28 x 28 mm), TFBGA180 (12 x 12 mm)
- ❑ LQFP100, TFBGA100

Operating

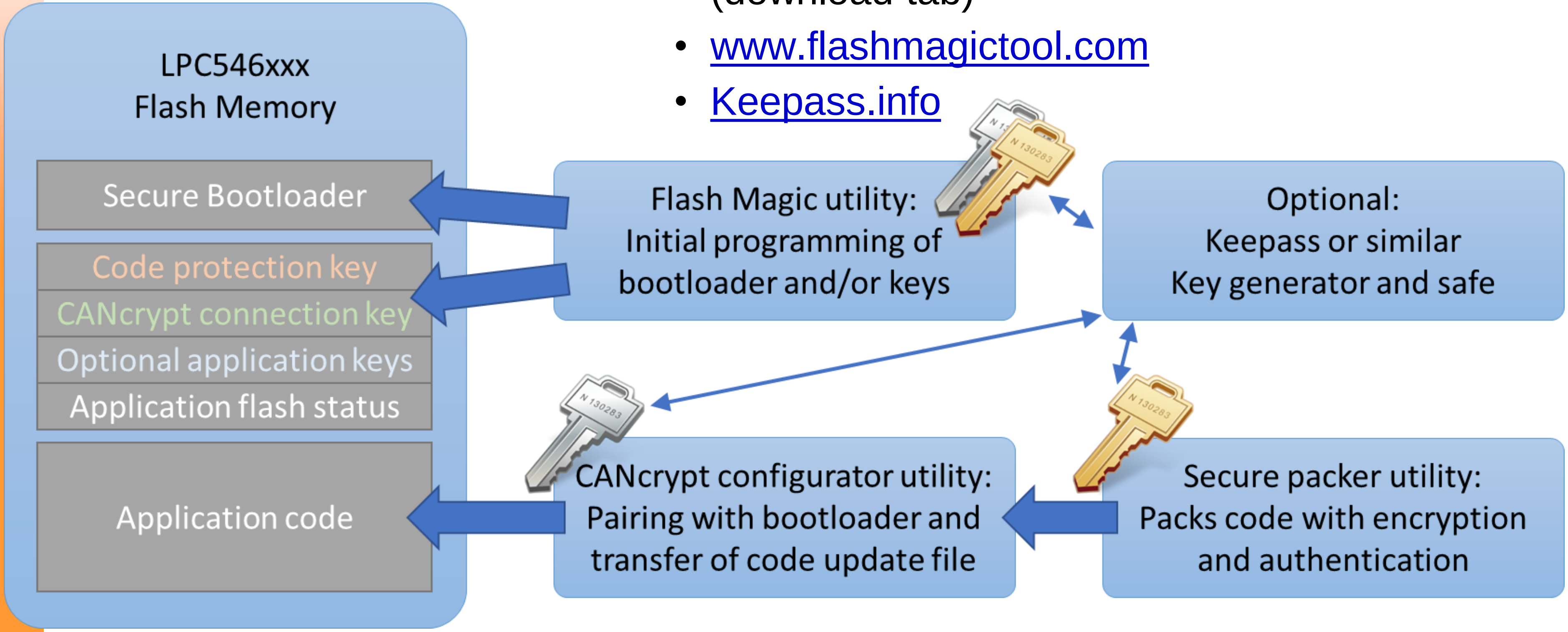
- ❑ Operating voltage: 1.71 to 3.6V
- ❑ Temperature range: -40 to 105 °C

Internal Bootloader
ISP: In-System Programming

- ❑ **LPC546xx has various options to load code**
  - USART/I2C/SPI
  - USB0/USB1
  - Plus programming via SWD debug port
- ❑ **Per default, they are all enabled**
  - Pulling ISP_PINx low on reset activates them
- ❑ **They can all be disabled by software**
  - If disabled by secondary, secure bootloader, then ISP/SWD can no longer be used
  - If keys are lost, no more updates...
- ❑ **Default of our secondary bootloader**
  - All remain enabled

Software components overview

- ❑ **Binary of bootloader (.hex) and all utilities required are available as free download**
  - www.nxp.com/demoboard/om13094 (download tab)
  - www.flashmagictool.com
  - Keepass.info



ESAcademy‘s secure CAN bootloader Free vs. Commercial

Free download

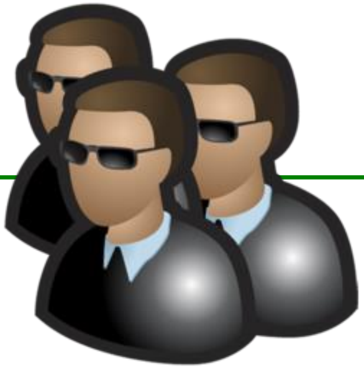
- ❑ **Delivered as .hex**
- ❑ **On-chip ISP enabled**
  - ISP remains as backdoor
- ❑ **Fixed bit rate**
  - 500/2000 kbps
- ❑ **Fixed device and node ID**
  - 15
- ❑ **Pre-selected security methods**
  - AES-GCM encryption and authentication

www.cryptopp.com/wiki/GCM_Mode

License from ESAcademy

- ❑ **Full C source code**
- ❑ **On-chip ISP may be disabled**
  - No more updates if key is lost
- ❑ **Configurable bit rate**
  - Any combination supported
- ❑ **Configurable device and node ID**
  - 2-15 or 1-127
- ❑ **Selectable security methods**
  - All common methods supported
  - AES, SHA, RSA, EEC

Security limits



- ❑ **Keys are stored in regular Flash!**
  - Can be read by ISP/SWD
    - if not protected
  - Can be read from application
    - NOTE: Only manufacturer can load new application via secure bootloader
- ❑ **Key generation and installation?**
  - Keys must be „truly random“
  - Must happen in a trustworthy environment
- ❑ **Key storage**
  - Treat keys as valuable as what they protect
    - Here: source code
  - For small amount of keys, a password manager like „KeePass“ can be used

Secure Bootloader Security Review

- ❑ **3rd Party contracted with a security review of the secondary, secure CAN-FD bootloader**

MathEmbedded

Embedded Security | Embedded Software

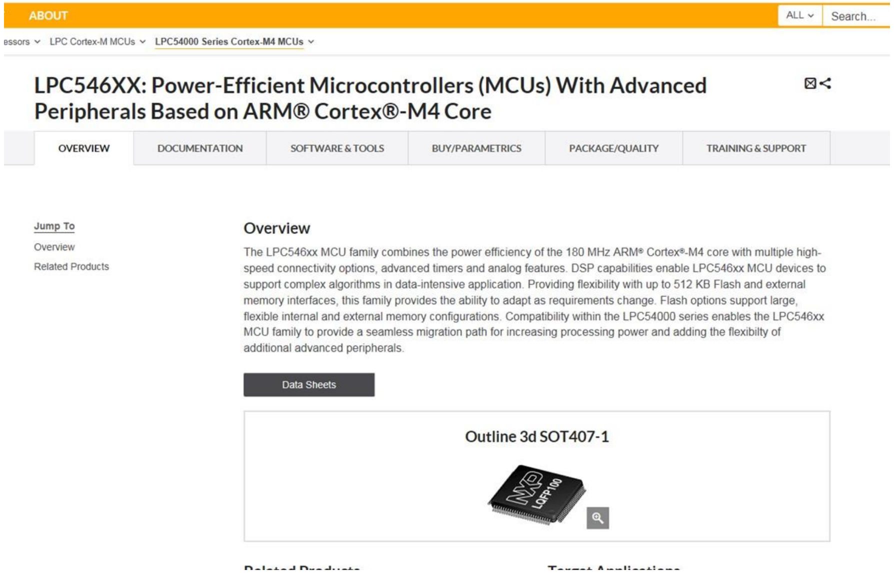
MathEmbedded have many years of experience securing embedded systems for a wide range of global companies and in a number of market areas.

mathembedded.com

- ❑ **Work in progress, result expected within July**
  - Result will be published in ESAcademy‘s Blog

Where to get started
Files available from Monday 10th of July 2017

- ❑ **LPC range of MCUs at nxp.com/lpc**
- ❑ **LPCXpresso54618 board at nxp.com/demoboard/om13094**
- ❑ **CAN-FD driver add-ons under Downloads tab**
- ❑ **Free tools and software at nxp.com/mcuxpresso**



LPC546XX: Power-Efficient Microcontrollers (MCUs) With Advanced Peripherals Based on ARM® Cortex®-M4 Core

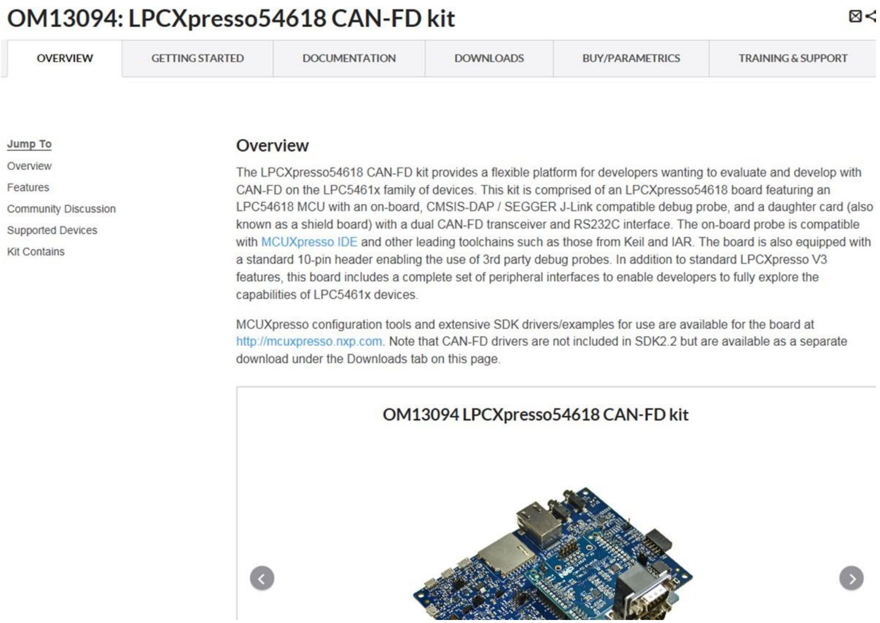
Overview

The LPC546xx MCU family combines the power efficiency of the 180 MHz ARM® Cortex®-M4 core with multiple high-speed connectivity options, advanced timers and analog features. DSP capabilities enable LPC546xx MCU devices to support complex algorithms in data-intensive application. Providing flexibility with up to 512 KB Flash and external memory interfaces, this family provides the ability to adapt as requirements change. Flash options support large, flexible internal and external memory configurations. Compatibility within the LPC54000 series enables the LPC546xx MCU family to provide a seamless migration path for increasing processing power and adding the flexibilty of additional advanced peripherals.

Outline 3d SOT407-1

Related Products

Target Applications



OM13094: LPCXpresso54618 CAN-FD kit

Overview

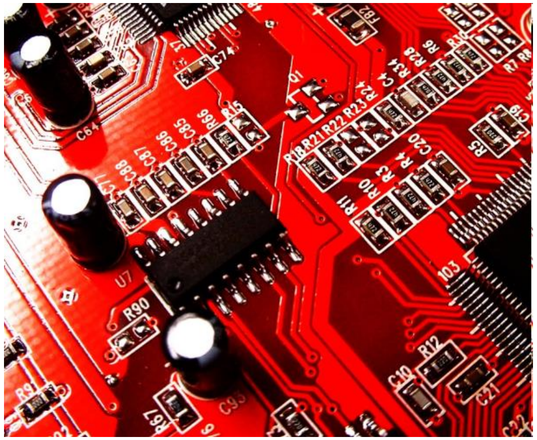
The LPCXpresso54618 CAN-FD kit provides a flexible platform for developers wanting to evaluate and develop with CAN-FD on the LPC5461x family of devices. This kit is comprised of an LPCXpresso54618 board featuring an LPC54618 MCU with an on-board, CMSIS-DAP / SEGGER J-Link compatible debug probe, and a daughter card (also known as a shield board) with a dual CAN-FD transceiver and RS232C interface. The on-board probe is compatible with MCUXpresso IDE and other leading toolchains such as those from Keil and IAR. The board is also equipped with a standard 10-pin header enabling the use of 3rd party debug probes. In addition to standard LPCXpresso V3 features, this board includes a complete set of peripheral interfaces to enable developers to fully explore the capabilities of LPC5461x devices.

MCUXpresso configuration tools and extensive SDK drivers/examples for use are available for the board at http://mcuxpresso.nxp.com. Note that CAN-FD drivers are not included in SDK2.2 but are available as a separate download under the Downloads tab on this page.

OM13094 LPCXpresso54618 CAN-FD kit

Embedded Systems Academy, Inc.

1250 Oakmead Parkway Ste. 210
Sunnyvale, CA 94085
(877) 812-6393



www.esacademy.com
www.cancrypt.eu

Olaf Pfeiffer
opfeiffer@esacademy.com
Twitter: CANcrypt
Blog: www.esacademy.com/blog