From CAN to TTCAN –

An ARM-based CAN Microcontroller including an innovating implementation of the new Protocol

Gerhard Goller, Philippe Malecha, Jean-Sebastien Berthy, Guy Mantelet,
Atmel Nantes S.A., France

Sophisticated drive-by-wire systems require extensive set-up and self-check procedures. Due to the complexity of the systems, code updates are used frequently, initially during the test phase, and later during the lifetime of the product. An efficient implementation of all these features in addition to the normal operating modes demands strong flexibility in the TTCAN/CAN microcontroller. The following functions will support these requirements:

- **On-the-fly selection of all features of message objects (channels) including TTCAN/CAN, Receiver, Receiver Buffer or Transmitter mode. This permits efficient use of the full power and buffer space of the device in all modes. Each channel uses its own independent control and buffer space.**
- **Flash memories for code, boot and application parameters are updated via CAN bus by In-System-Programming (ISP).**
- **Full secured operations maintain system functionality in all modes.**

This paper describes the new TTCAN/CAN Flash microcontrollers featuring a powerful combination of ARM™ core, Flash Memories, the advanced TTCAN/CAN Controller with four CAN bus interfaces as well as other powerful peripherals. In-System-Programming (ISP) and numerous Application Program Interface (API) software accelerate the set-up of embedded applications.
This article presents the main features of these smart TTCAN/ CAN microcontrollers and provides examples of how it will help to improve system flexibility of a competitive CAN based application.

1 . Introduction

In order to comply with upcoming security relevant architectures, (e.g. x-by-wire techniques with time-triggered busses) high computing power in conjunction with on-chip intelligent bus interfaces are required. Further, different existing bus systems in an automotive application need gateways which allow easy inter-bus communication without disturbing the host.

For compatibility and software re-use reasons the ARM7-based architecture and identical peripherals were used for both approaches. Small differentiations were made in the memory sizes, TTCAN/CAN controller structure, peripherals, and pin count.

2 . Architecture Overview

The described Network µControllers are the new ARM7-based members of the Atmel CANary™ CAN microcontroller family. Like all members of the CANary™ family special attention was taken to the EMC behavior and simple and safe In-System-Programming via CAN bus. Both features were achieved through the application of a PLL stage in the clock generation and with the use of Flash memories for User (128/256kB), EEPROM (2kB), independent Boot Memory (4kB) and Control Bits. The PLL in combination with a twin oscillator of 8MHz and 32kHz permits low Power consumption for operating and stand-by modes. See Figure 1.

The applied ARM7TDMI core allows a 16-bit Thumb instruction mode for high code density and a smart system approach.

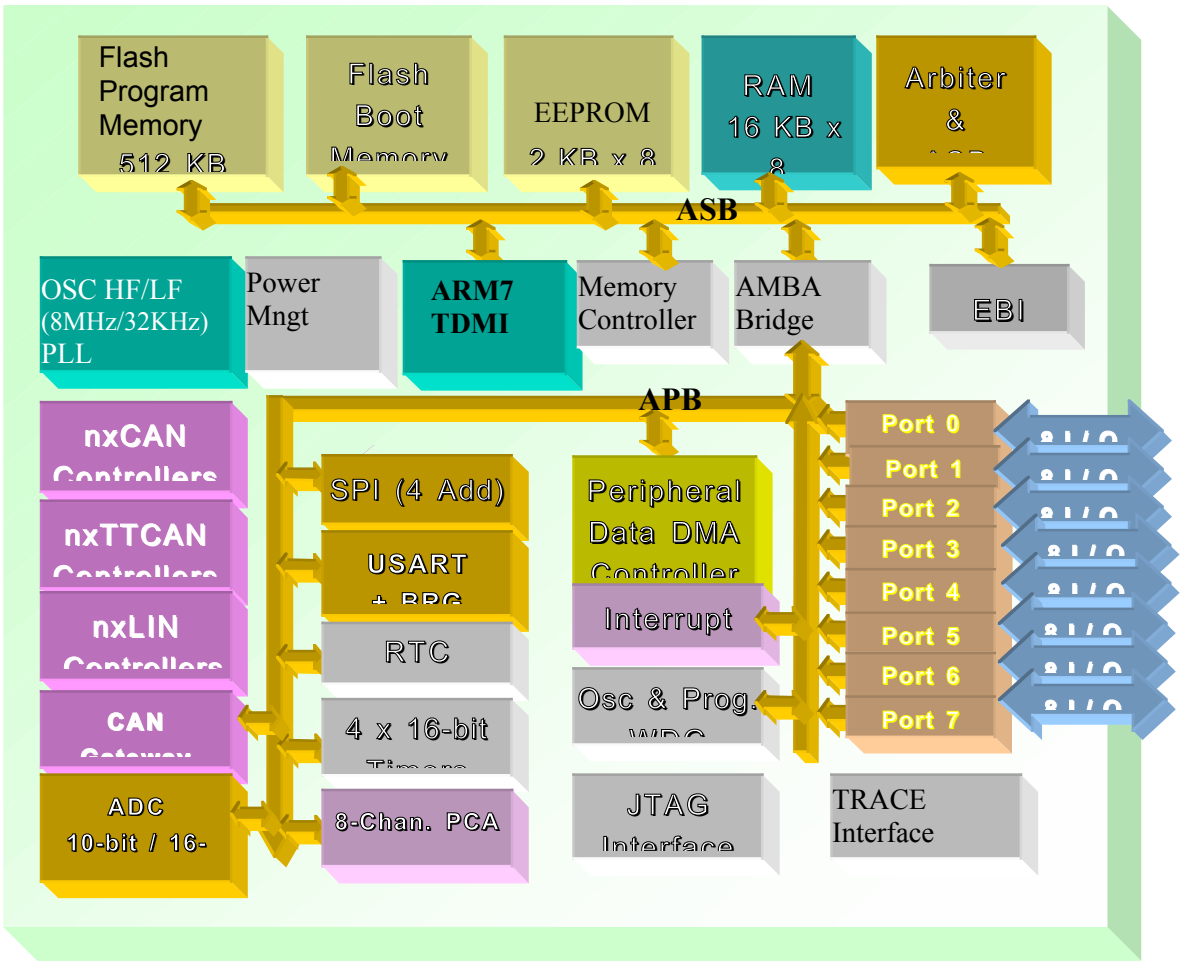

Figure 1: Network Gateway Microcontroller Block Diagram

2 . 1 Peripherals

While ARM core, Flash memories, RAM (4KB/16KB) and the External Bus Interface (EBI) are connected to the ARM AMBA High Speed Bus (ASB) the following peripherals are connected to the AMBA Peripheral Bus (APB). See Table 1:

- 8/10-Channel Programmable Counter Array (PCA), programmable as
  - PWM with edge synch.,
  - Counter/Timer
- Three 16-bit Timers
- 10-bit ADC
- UART with BRG
- Watchdog with on-chip oscillator
- SPI, Master/Slave
- Interrupt Controller, 4 external / 1 NMI
- 6/8 I/O Ports with wake-up function
- JTAG Interface
- Trace Interface
- Three 16-bit Timers
- Network Gateway Controller
  - CAN interfaces 2.0A/2.0B
  - TTCAN interfaces
  - LIN Interface
  - Gateway message handler

The applied ARM7TDMI core allows a 16-bit Thumb instruction mode for high code density and a smart system approach.

2 . 2 Flash Memory Concept

To avoid any software crash during the period in which a CAN µController is changing its own application program via CAN, some precautions have to be taken.
Taking into account the strong security requirements inherent to CAN applications, the CANary™ family offers the functions that correspond to such requirements:

- Separation of User Flash Memory and Boot Flash Memory.
- Boot Flash Memory alterable only in parallel mode by programmer.
- Boot Control bits in Flash technology.
- Default Boot Loader part of the product.

3 . In-System-Programming (ISP)

In applications based on a CAN bus this link is ideal for ISP in order to provide a suitable and cost efficient solution.
The Flash Memory Concept is supported by dedicated software which allows easy and safe In-System-Programming:

- PC-based ISP pilot FLIP (Flexible-In-System-Programmer) easily adaptable to existing CAN Networks.
- Atmel CAN Boot loader for User Flash and EEPROM update located in the separate Boot Flash Memory

To avoid unauthorized ISP access to the User Flash Memory and the EEPROM, a three-level security mechanism protects customer data.
ISP security is further increased by the fact that the Boot Flash Memory is alterable only in parallel mode by the device programmer due to

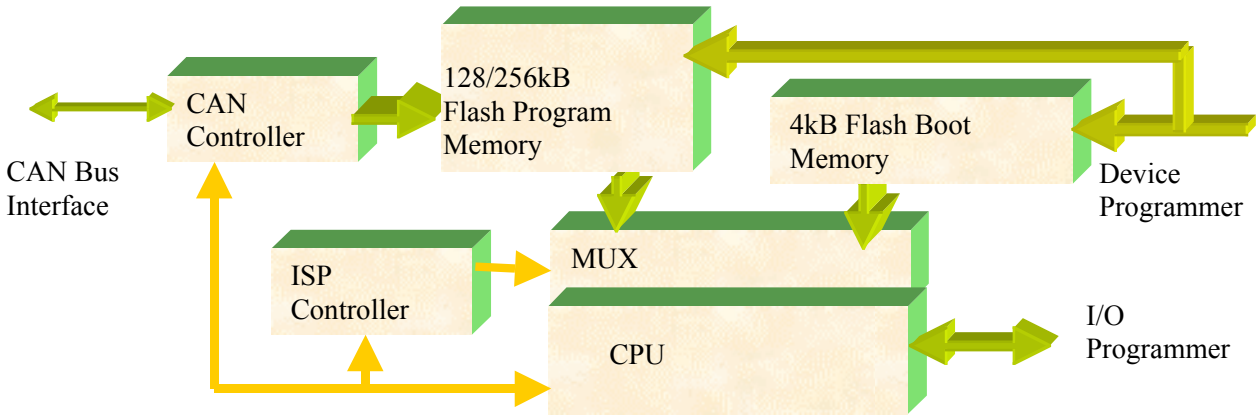

Figure 2 : Flash Memory Concept

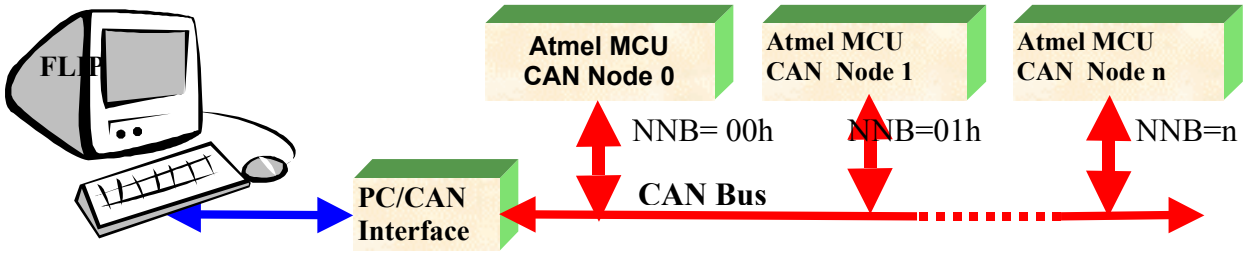

Figure 3 : Embedded In-System-Programming (ISP) via CAN Bus

its physical separation from the User Flash Memory and the EEPROM.

ISP programming errors will not affect the Boot Memory, this allows a restart of the procedure under all conditions. See Figure 2.

3 . 1 ISP pilot FLIP

Supporting all usual PC operating systems, FLIP is used for ISP operations on standard systems:

- CAN
- RS232
- USB
- I2C

To interface with the CAN bus a wide number of CAN parallel port interfaces are supported:

- IXXAT, CANdy
- VECTOR, CANpari
- PEAK, PCAN
- Atmel software interface using the RS232, available on the web.
- USB / CAN interface from SYSTEC.

The FLIP protocol stack presents a set of only seven standard CAN frames to manage the ISP mechanism used in conjunction with the on-chip boot loader. In order to avoid collisions in existing CAN Networks by using FLIP, the FLIP CAN frame identifiers can be easily placed in a free space by programming the CAN Re-locatable Identifier Segment (CRIS) located in the Network microcontroller. The seven identifiers used will start with the number written in CRIS.

3 . 2 ISP Handling

Factory-new PCBs must first pass a point-to-point initialization In order to determine the:

- Node Number Bit (NNB), the ISP house number of the application
- CAN Re-locatable Identifier Segment (CRIS), the CAN frame identifier off-set value
- Bit Timing Configuration (BTC), also determined by FLIP auto baud mechanism
- Boot Loader Jump Bit (BLJB), indicating whether the Program or the Boot Flash Memory will start communication with FLIP.

FLIP supports the following embedded boot process by its numerous editor functions. By addressing one CAN node in the network via the NNB the Boot Flash Memory is accessed and takes command of the node. In conjunction with FLIP the requested data up-date in the User Flash Memory and the EEPROM is affected. Due to architectural precautions a destruction of the boot loader is not possible. See Figure 3.

3 . 3 ISP Data Protection

In addition to the three LSB of the Hardware Security Byte already used to protect unauthorized parallel access of the memories, special attention was taken to also protect the serial access to the user memory via ISP. Three Software Security Bytes (SSB) manage three levels of security:

- Level 0: NO_SECURITY
  From this default level access to level 1 & 2 is possible.
- Level 1: WRITE_SECURITY
  On this level no write operations are possible to the User Flash memory or the

EEPROM. From level 1 only access to level 2 is possible.

- Level 2: RD_WR_SECURITY Write and read operations are impossible. Only a full chip erase in parallel mode by a Programmer or by the FLIP command ERASE can reset the SSB.

4 . CAN Controller

Combining all functions of a serial network gateway, the CAN controller manages the inter-bus communications of the different bus interfaces through its message handler without disturbing the host. Only messages arriving from one of the connected busses concerning the node itself will generate an interrupt and be transferred to the RAM of the host via the Peripheral Data DMA Controller. The different bus interfaces handled are from type CAN, TTCAN or LIN. A bi-directional FIFO will buffer the data received and data to be transmitted for synchronization reasons.

Each interface has his own state machine and will have the following characteristics:

- Programmable Baud Rate Generator.
- Data buffer to manage the different speed and occupation of the busses.
- Masking and Filtering on Identifiers and Data for arbitrated messages.
- Parameter determination on-the-fly.
- Interrupt generation at message reception.
- Readable error counters.
- Error capture with IT
- Self test modes

See Figure 4.

4 . 1 . CAN Controller

The special functions of the CAN interface are:

- 2.0A/2.0B communication
- 1Mbit/s Bit Rates
- Receiver for multi-frame message support
- 16-Message Objects with own Identifier mask and 8 bit Data mask
- Listen only mode
- Direct reply on Remote frame
- TTCAN Support
  - Single shot transmission
  - Time stamp register
- Wake-up by CAN bus

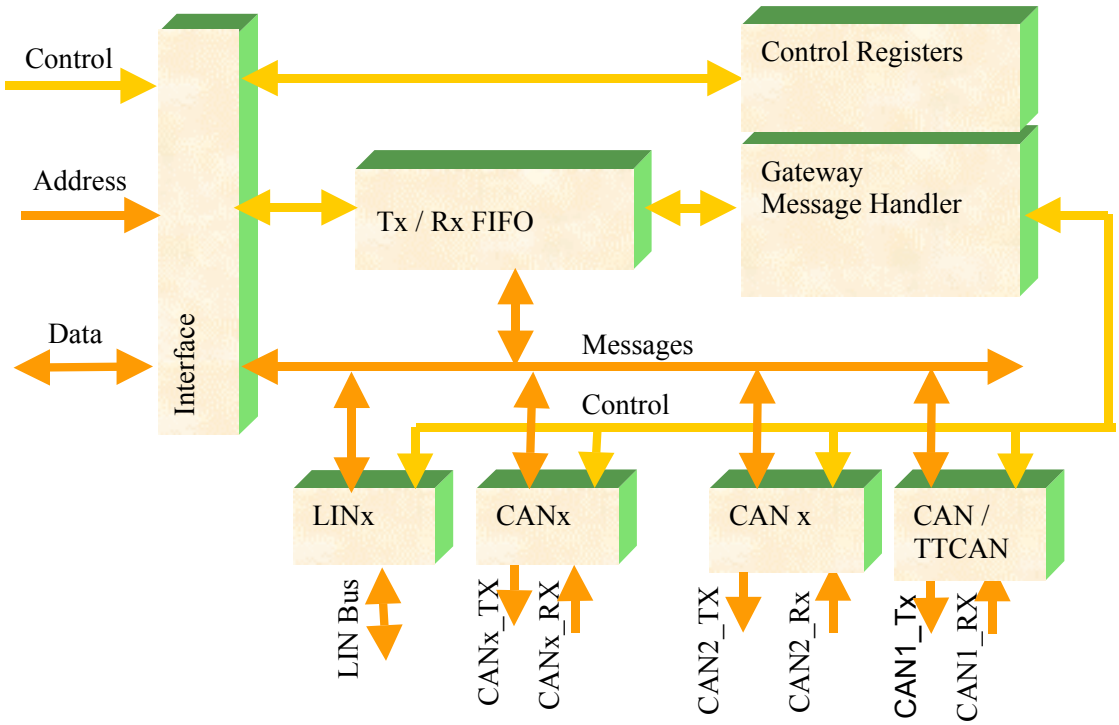

Figure 4 : Network Gateway Microcontroller

4 . 2 . TTCAN Controller

The special functions of the TTCAN interface are:

- 2.0A/2.0B communication
- 1Mbit/s Bit Rates
- 32 Message Objects with own Identifier and 8 bit Data mask
- Message filtering
- TTCAN Protocol level 1 and Level 2 in hardware
- Standard CAN time triggered functions
- Self-test modes

5 . Power Management

The different requests on high computing power, low EMC as well as low Power-Down Consumption are answered by the application of two software controlled oscillators of 8MHz and 32kHz. The 8MHz works in conjunction with a PLL in order to achieve the 40MHz of internal clock in operating mode.
Low power consumption mode is achieved by operating with the 32kHz oscillator in cases where very low computing power is required. The 32kHz oscillator further permits the easy application of a Real Time Clock (RTC).
The different interrupts, wake-up feature on I/O lines and most peripherals such as CAN controller help to bring the device out of the low consumption modes.

	Flash Program Memory	EEPROM	Flash Boot Memory	RAM	PCA / Timers	10bit ADC	SPI	UART with BRD	CAN/TTCAN Controller UNIT	LIN
Network Microcontroller	128kB	2kB	4kB	4kB	8-Ch. PWM, 3 Timers	yes	yes	yes	1TTCAN 32 Message Objects	yes
Network Gateway Microcontroller	256kB	2kB	4kB	16kB	10-Ch. PWM, 3 Timers	yes	yes	yes	3 CAN 16 M. Objects, 1 TTCAN 32 M. Objects, Gateway handler	yes

6 . Conclusion

CAN based control techniques have, during the last several years, penetrated industrial and automotive applications with an enormous growth.
Higher Layer Protocols (HLP) specially designed for security relevant
applications use the standard CAN protocol. The presented ARM TTCAN microcontrollers offer a smart solution for such security sensitive applications including:

- Production automatisation,
- Lift control,
- Automotive,
- Robotic,
- Medical,
- Agriculture.

In order to respond to physical size and high volume requests the devices will be packaged in TQFP44/64/80 packages.

Atmel S.A. Nantes
BP 70602 – La Chantrerie
F – 44306 Nantes Cedex3, France
Website : http://www.atmel.com

Gerhard Goller
Phone: +33 240 18 18 48
Fax: +33 240 18 19 60
Mobile:+33 603 95 26 36
e-mail: gerhard.goller@nto.atmel.com

Philippe Malecha
Phone: +33 240 18 17 24
Fax: +33 240 18 19 60
Mobile: +33 680 45 73 74
e-mail : philippe.malecha@nto.atmel.com

Jean-Sebastien Berthy
Phone: +33 240 18 18 16
Fax: +33 240 18 17 50
e-mail: jean-sebastien.berthy@nto.atmel.com

Guy Mantelet
Phone: +33 240 18 19 42
Fax: +33 240 18 17 50
e-mail: guy.mantelet@nto.atmel.com