



Datasheet

CAN FD Bus Controller

Arasan Chip Systems Inc.

2010 North First Street, Suite #510, San Jose, CA 95131

Ph: 408-282-1600

Fax: 408-282-7800

www.arasan.com

Disclaimer

This document is written in good faith with the intent to assist the readers in the use of the product. Circuit diagrams and other information relating to Arasan Chip Systems' products are included as a means of illustrating typical applications. Although the information has been checked and is believed to be accurate, no responsibility is assumed for inaccuracies. Information contained in this document is subject to continuous improvement and development.

Arasan Chip Systems' products are not designed, intended, authorized or warranted for use in any life support or other application where product failure could cause or contribute to personal injury or severe property damage. Any and all such uses without prior written approval of an Officer of Arasan Chip Systems Inc. will be fully at the risk of the customer.

Arasan Chip Systems Inc. disclaims and excludes any and all warranties, including, without limitation, any and all implied warranties of merchantability, fitness for a particular purpose, title, and infringement and the like, and any and all warranties arising from any course or dealing or usage of trade.

This document may not be copied, reproduced, or transmitted to others in any manner. Nor may any use of information in this document be made, except for the specific purposes for which it is transmitted to the recipient, without the prior written consent of Arasan Chip Systems Inc. This specification is subject to change at any time without notice. Arasan Chip Systems Inc. is not responsible for any errors contained herein.

In no event shall Arasan Chip Systems Inc. be liable for any direct, indirect, incidental, special, punitive, or consequential damages; or for loss of data, profits, savings or revenues of any kind; regardless of the form of action, whether based on contract; tort; negligence of Arasan Chip Systems Inc or others; strict liability; breach of warranty; or otherwise; whether or not any remedy of buyers is held to have failed of its essential purpose, and whether or not Arasan Chip Systems Inc. has been advised of the possibility of such damages.

Restricted Rights

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Copyright Notice

No part of this specification may be reproduced in any form or means, without the prior written consent of Arasan Chip Systems, Inc.

Questions or comments may be directed to:

Arasan Chip Systems Inc.
2010 North First Street, Suite 510
San Jose, CA 95131
Ph: 408-282-1600
Fax: 408-282-7800
Email: sales@arasan.com

Contents

1	Introduction	1
1.1	Overview	1
1.2	Arasan’s Contribution to MIPI.....	1
1.3	Architecture.....	1
1.3.1	Functional Description	1
1.3.2	Functional Block Diagram	2
1.3.3	Functional Block Diagram Description	3
1.4	PIN Diagram.....	4
1.5	IP Deliverables.....	5

Figures

Figure 1:	CAN Bus Controller Functional Block Diagram	2
Figure 2 :	CAN Bus Controller IP Pinout Diagram.....	4

1 Introduction

1.1 Overview

Arasan Chip Systems is a leading provider of standards compliant IP for Mobile and Automobile Semiconductor Markets. Arasan's complete suite of IP solutions consist of analog PHY and digital controller IP cores, verification IP, software stacks and drivers, hardware validation platforms for software development and compliance testing, and optional customization services.

This document describes The Controller Area Network (CAN) controller IP that implements the CAN2.0A, CAN2.0B as well as newer high performance Non ISO CAN-FD protocols. It can be integrated into devices that require CAN connectivity commonly used in automotive and industrial applications.

1.2 Arasan's Contribution to MIPI

- Implements CAN2.0A and CAN2.0B protocol, ISO 11898-1 compliant
- Supports CAN-FD
 - Non-ISO CAN FD - Compliant to Bosch Protocol
- Independent System Clock(SYSCLK) and CAN Bus Clock(CANCLK)
 - SYSCLK is CPU interface Bus Clock, - AHB Clock in case of ARM
 - CANCLK can either be independent or tied to SYSCLK
- Flexible shared buffering scheme that implements optimal buffer size for Transmit and Receive messages
 - Total buffer size is parameterized - synthesis time option
 - Buffer can be implemented as a single port SRAM or flops
 - Transmit buffer, receiver buffer and high-priority transmit buffer
 - CPU configurable depths for transmit, receive and high-priority transmit buffers
- Parameterized number of Acceptance Filters - 1-16
- AHB-Lite Slave Interface for connecting to CPU
 - Optional APB Interface
 - Supports 32-bit interface
- Programmable Baud Rate Prescaler (BRP)
 - Generate Time Quantum Clock from CANCLK
 - 8-bit BRP register to have div-by-2 up to div-by-255

1.3 Architecture

1.3.1 Functional Description

The CAN Bus Controller IP has been architected and designed to simplify the job of implementing CAN protocol in devices used automobile and industrial applications.

1.3.2 Functional Block Diagram

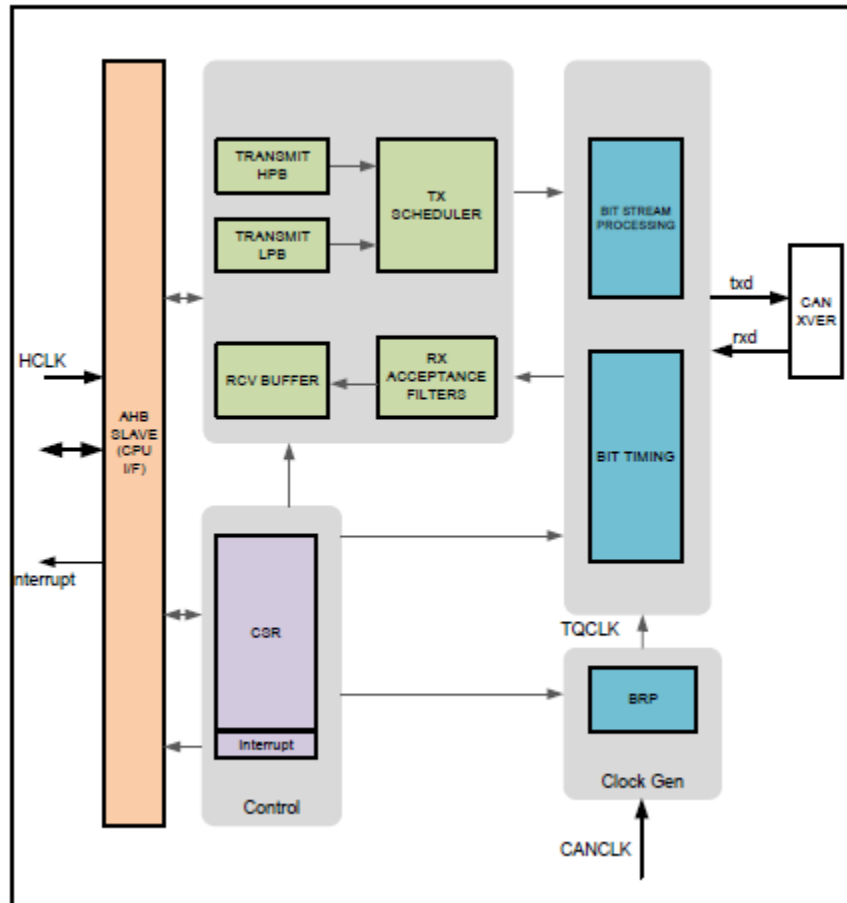


Figure 1: CAN Bus Controller Functional Block Diagram

1.3.3 Functional Block Diagram Description

1.3.3.1 Bit Timing

The primary function of Bit Timing logic is to align the controller to incoming CAN traffic using Sync, Propagation Segment and Phase Segments. The segments are measured in units of quanta counted using a Time Quantum(TQ) Clock, generated by Baud Rate Prescaler (BRP) by dividing the CANCLK by a programmable value. Once synchronization is achieved, it generates sample reference to Bit Stream Processing (BSP) for higher level frame decoding function.

1.3.3.2 Baud Rate Prescaler

The BSP block performs various higher level protocol functions typically classified as MAC functions. These include, checking and removal of CRC bits and stuffed bits in the receive direction. Similarly, the BSP block serializes the stream in the transmit direction, adds stuffing bits as needed and computes and adds CRC bits. It transfers complete frames on to the CAN Bus after arbitrating for the bus. It automatically retransmits the bus if access is lost in the arbitration process.

1.3.3.3 Rx Acceptance Filters

This block filters incoming frames by comparing 11-bit or 29-bit IDENTIFIER in each message with number of Acceptance Filter Registers (AFR) along with corresponding Acceptance Filter Masks (AFM). The messages that pass any of the filters are accepted and stored in Rx Buffer. The AFR and AFM are individually programmable by the CPU. The number of filters is a synthesis time option.

1.3.3.4 Receive Buffer

The Receive Buffer (RB) stores received frames that are error free and passed acceptance filters. From here, they are read by the CPU over AHB Slave interface. Each message contains a header indicating message identifier, length and other flags so CPU knows how many bytes to read. They are serviced first-in-first-out to CPU via via register interface. The receive buffer depth is configurable. The number of messages stored varies depending on the depth and length of individual messages.

1.3.3.5 Transmit High Priority Buffer

The Transmit High Priority Buffer (TXHPB) is used for storing high priority messages that the CPU wants be transmitted at the highest priority. The Transmit Scheduler attends to these messages ahead of messages stored in low priority buffer. The depth of TXHPB is configurable by CPU.x`

1.3.3.6 Transmit Low Priority Buffer

The Transmit Low Priority Buffer (TXLPB) stores frames that are marked by CPU as normal priority. The messages out of this buffer are serviced FIFO style and scheduled at a lower priority, i.e., transferred if TXHPB doesn't have any complete messages. The depth of the TXLPB is also configurable by CPU.

1.3.3.7 Tx Scheduler

The Tx Scheduler (TXSCH) block implements a Strict-Priority (SP) scheduling between messages available in TXHPB and TXLPB where messages in TXHPB have highest priority. The messages from both the buffers are serviced first-in-first out.

1.3.3.8 Configuration Registers

The Configuration and Status Register (CSR) block implements various command, control and status registers needed for the operation of the controller and to pass the status to CPU. These include an Interrupt Status Register (ISR) as well.

1.3.3.9 AHB Slave Interface

The AHB Slave Interface serves as host bus for CPU to communicate with the IP. The AHB interface is used for both accessing configuration registers as well as passing messages.

1.4 PIN Diagram

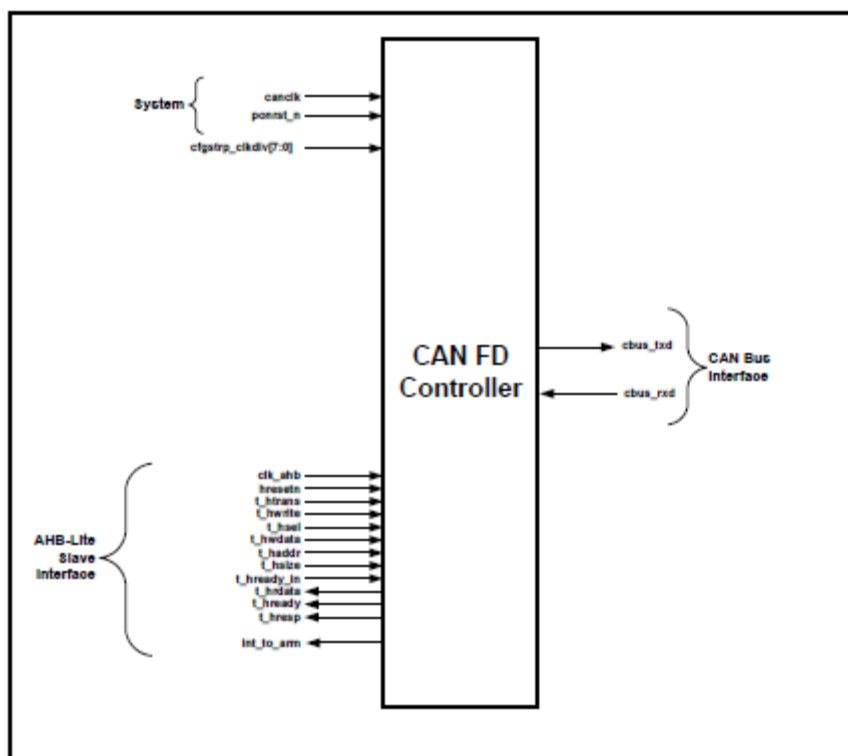


Figure 2 : CAN Bus Controller IP Pinout Diagram

1.5 IP Deliverables

- Verilog HDL of the IP Core
- Synthesis scripts
- Gate count estimates available upon request
- User guide
- Verification environment and tests