(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 November 2014 (06.11.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/177330 A1

(51) International Patent Classification:
H04L 12/46 (2006.01) *H04L 12/40* (2006.01)
H04L 29/06 (2006.01)

(21) International Application Number:
PCT/EP2014/056414

(22) International Filing Date:
31 March 2014 (31.03.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13166142.3 1 May 2013 (01.05.2013) EP

(71) Applicant: **RENESAS ELECTRONICS EUROPE GMBH** [DE/DE]; Arcadiastrasse 10, 40471 Duesseldorf (DE).

(72) Inventors: **LIEDER, Roland**; c/o Renesas Electronics Europe GmbH, Arcadiastrasse 10, 40472 Duesseldorf (DE). **TURSKI, Klaus**; c/o Renesas Electronics Europe GmbH, Arcadiastrasse 10, 40472 Duesseldorf (DE).

(74) Agents: **PIOTROWICZ, Pawel** et al.; Venner Shipley LLP, Byron House, Cambridge Business Park, Cowley Road, Cambridge Cambridgeshire CB4 0WZ (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*

(54) **Title**: CAN FD

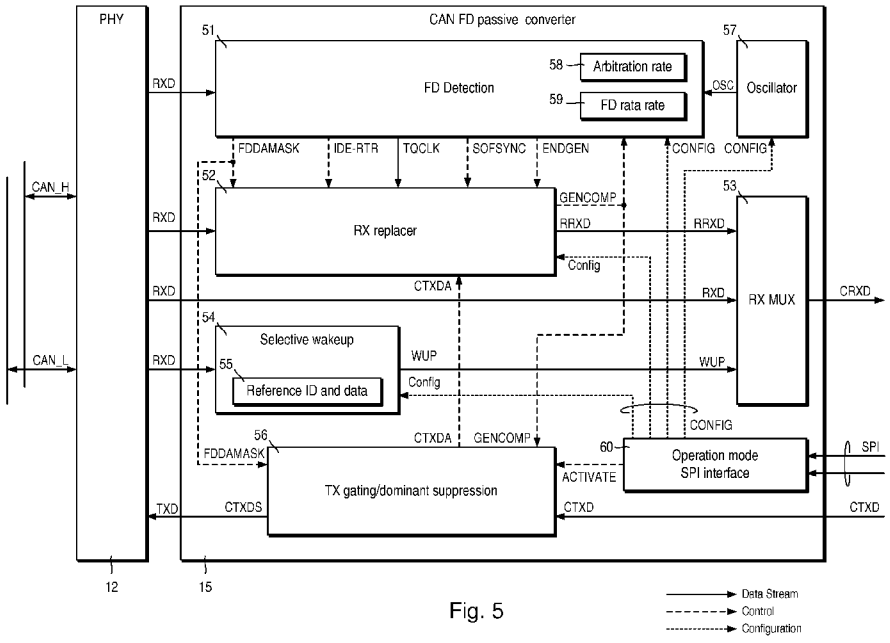


Fig. 5

(57) **Abstract**: A device (15) for allowing a CAN 2.0B controller to participate passively in CAN FD communication is described. The device is configured to identify whether a frame on RXD is a CAN FD frame (14; Fig. 4) and, in dependence upon identifying that the frame is a CAN FD frame, to replace a section (17; Fig. 4) of the CAN FD frame, including the data phase of the CAN FD frame, with substitute data (21; Fig. 4) having a format which complies with CAN 2.0B. The device (15) may be included in a CAN transceiver (10; Fig. 3).

WO 2014/177330 A1

CAN FD

Field of the Invention

The present invention relates to controller area network flexible data rate (CAN FD).

Background

Controller area network (CAN) is a serial communications protocol which was originally designed for use in automotive applications, but which has now been adopted in a wide variety of other fields including heating, ventilation and air conditioning (HVAC) control systems, process control systems and medical equipment. CAN mainly concerns the requirements of physical and data link layers and has been declared an international standard, namely ISO 11898.

CAN is a message-based, asynchronous, multi-master, broadcast communication system. A CAN network comprises two or more nodes connected to a CAN bus line which is typically provided by a twisted pair of wires. Any node connected to a CAN bus is free to try to transmit a message to other nodes when the CAN bus is free. Each message is assigned a unique identifier which is used to identify the content of the message and to prioritise and filter messages. If two or more nodes start transmitting messages at the same time, then bus access conflict is resolved through a process of arbitration using the identifier and bit-wise comparison of CAN bus states, namely "dominant" and "recessive" levels corresponding to logical '0' and '1' states respectively. When two nodes try to transmit dominant and recessive bits at the same, the dominant bit takes precedence and the dominant value (i.e. '0') appears on the CAN bus. Nodes monitor the CAN bus and so when a node tries to transmit a recessive bit and the CAN bus adopts a dominant value, then the node is able to identify a collision and interrupt transmission, leaving the node that transmitted the dominant bit to continue transmission on the CAN bus.

CAN specifies two frame formats, namely a standard or base frame format (which is compatible with part A or B of the CAN 2.0 protocol) and an extended frame format (which is compatible with part B of the CAN 2.0 protocol). A frame arranged in the standard or base frame format has an 11-bit identifier, whereas a frame arranged in the extended frame format has a two-part identifier made up of an 11-bit base identifier and an 18-bit extension.

CAN specifies four types of frame, namely a data frame for carrying data, a remote frame for requesting transmission of a data frame, an error frame for reporting detection of an error and an overload frame for providing a delay between frames.

Further details of CAN and CAN frame formats can be found in the CAN Specification, version 2.0 (1991) and ISO 11898-1.

CAN buses can transmit data at bit rates of up to 1 Mbit/s and data frames can carry a payload of up to 8 bytes. However, to support higher data rates and larger payloads, a new protocol, CAN with Flexible Data Rate (CAN FD), has been proposed as set out in CAN with Flexible Data-Rate, Specification, version 1.0 (2012).

Figure 1 shows a CAN FD data frame.

Referring to Figure 1, a CAN FD data frame is separated from a preceding frame (not shown) by an interframe space (IFS) and is followed by another interframe space or an overload frame.

The CAN FD data frame includes a start of frame (SOF) field, an arbitration field, a control field, a data field, a cyclical redundancy code (CRC) field, an acknowledge (ACK) field and an end of frame (EOF) field.

In a so-called "arbitration phase", IFS preceding a CAN FD frame and a first portion of the CAN FD frame is transmitted at a standard bit rate (i.e. a CAN bit rate, e.g. 500 $kbits^{-1}$). In a so-called "data phase", a second portion of the CAN FD frame can be transmitted at either a standard bit rate or a high bit rate (for example, up to 8 times the CAN bit rate). In another arbitration phase, a third portion of the CAN FD data frame and IFS or overload frame following the CAN FD data frame is transmitted at a standard bit rate.

CAN FD specifies four frame formats, namely CAN standard and CAN extended formats (corresponding to the standard and extended formats specified in ISO 11898-1), CAN FD standard and CAN FD extended formats.

Figure 2 shows a CAN FD data frame in CAN FD standard (or "base") format.

Referring to Figure 2, the arbitration field includes a base identifier followed by a reserve bit (r1). The control field consists of an identifier extension (IDE) bit, an extended data length (EDL) bit, a reserve bit (r0), a bit rate switch (BRS) bit, an error state indicator (ESI) flag and a data length code (DLC). The EDL bit is also referred to as the Flexible Data Format (FDF) bit. The CRC field includes a CRC sequence and a CRC delimiter. The ACK field comprises an ACK slot and ACK delimiter.

The CAN FD extended format includes an arbitration field which consists of the base identifier, a substitute remote request (SRS) bit, the IDE bit, the identifier extension, and reserve bit (r1) and a control field which consists of the EDL bit (which may be referred to as the FDF bit), reserve bit (r0), the BRS bit, the ESI flag and the DLC.

All CAN FD frames formats include a CRC field comprising a CRC sequence and CRC delimiter. However, the length of the CRC sequence differs for different frame formats, namely 15 bits for CAN base and CAN extended frame formats, 17 bits for CAN FD frames which contain up to 16 bytes and 21 bits for CAN FD frames which contain more than 16 bytes.

The BRS bit marks a transition from the arbitration phase to the data phase. The CRC delimiter in a CAN FD frame signals a transition back to the arbitration phase.

Although the CAN FD protocol is based on part B of the CAN protocol (herein referred to simply as "CAN 2.0B"), it has limited backward compatibility with the CAN 2.0B protocol. A CAN FD node is able to receive and transmit CAN messages according to ISO 11898-1. However, a CAN node which can send and receive messages according to CAN 2.0B protocol (herein referred to as a "CAN 2.0B node") cannot identify CAN FD messages and so, on receipt of a CAN FD message, will respond with a CAN error message.

This can hinder implementation of a CAN system comprising a mixture of nodes which comply with CAN 2.0B but not CAN FD and nodes which comply with CAN FD.

One solution is to use partial networking whereby nodes can be selectively powered down, placed in sleep mode or kept in standby. However, this still means that CAN 2.0B and CAN FD nodes cannot directly communicate.

Summary

According to a first aspect of the present invention there is provided a device configured to identify whether a frame on RXD is a CAN FD frame and, in dependence upon identifying that the frame is a CAN FD frame, to replace a section of the CAN FD frame including the data phase of the CAN FD frame with substitute data having a format which complies with CAN 2.0B.

Thus, the device can be placed between CAN bus physical layer circuitry and a local CAN controller and remove sections of a CAN FD frame which would otherwise generate errors at the CAN controller. A CAN substitute frame which includes the substitute data is output on a new or output RXD (CRXD) to the CAN controller. The CAN substitute frame may include the beginning of the CAN FD frame. The CAN substitute frame may include additional data.

The section may be continuous.

The device may comprise first circuitry (for example a first module) for identifying whether the frame on RXD is a CAN FD frame.

The first circuitry may include a first functional block configured to identify the start of a frame. The first functional block may be configured to identify the start of the frame by identifying a pattern of bits. The pattern of bits may include EOF, IFS and SOF. The first functional block may be configured, in response to identifying the start of a frame, to output a start of the frame synchronisation signal (SOFSYNC).

The first circuitry may include a second functional block configured to identify error frame(s) and/or overload frame(s) in RXD in the arbitration phase. The second functional block may be configured, in response to detecting an error frame or overload frame, to output a first error signal (AERR).

The first circuitry may include a third functional block configured to identify error frame(s) and/or overload frame(s) in RXD in the data phase. The third functional block may be configured, in response to detecting an error frame or overload frame, to output a second error signal (DERR).

The first circuitry may include a fourth functional block.

The fourth functional block or another functional block may be configured to identify the IDE bit. The fourth functional block may be configured, in response to identifying the IDE bit, to output a signal (IDE-RTR) indicating the presence of the IDE bit.

The fourth functional block or another functional block may be configured to identify the EDL bit (which may be referred to as the FDF bit). The fourth functional block may be configured, in response to identifying the EDL bit, to output a signal indicating whether the EDL bit is set.

The fourth functional block or another functional block may be configured to identify the BRS bit. The fourth functional block may be configured, in response to identifying the BRS bit, to output a signal indicating whether the BRS bit is set.

The first circuitry may include a fifth functional block.

The fifth functional block may operate in first and second modes of operation.

The fourth functional block may be configured, in response to the EDL bit (which may be referred to as the FDF bit) being set, to output a signal for controlling masking of the CAN FD frame.

The fourth functional block may be configured, in response to given precondition, to output a signal (ENDGEN) for controlling generation of substitute data.

The device may comprise second circuitry (for example a second module) for replacing the section of the CAN FD frame including the data phase with substitute data having a format which complies with the CAN protocol.

The second circuitry may include a set of at least one or at least two generators for generating substitute data. The second circuitry may include a multiplexer coupled to RXD and to the generator(s) and to output a multiplexed signal.

The section may start with the EDL bit (which may be referred to as the FDF bit). The section may end with the CRC delimiter.

More than one CAN substitute frames may be used to replace the CAN FD frame section.

The device may be configured to replace data following the section with further substitute data having a format which complies with CAN 2.0B. The section and data may be contiguous.

The substitute data may include DLC. The DLC may specify that the frame contains no data.

The device may be configured to add additional substitute data after the substitute data. For example, this can be used when the CAN FD frame is longer than a CAN substitute frame.

The device may be configured to add additional data after the substitute data (i.e. to replace another subsequent section of the CAN FD frame with additional data).

In a first mode, the additional data may comprise an error frame and an error delimiter. The device may configured to identify a point given number of bits before the end of the CAN FD frame and to add the error frame and error delimiter such that the error delimiter ends when the CAN FD frame ends.

This can be used to minimise the time that a local CAN controller is prevented from accessing the CAN bus.

In a second mode, the additional data may comprise at least overload frame and corresponding overload delimiter.

This can used to help minimise the number of errors generated at a local CAN controller.

The device may be configured to identify whether the frame is a CAN FD frame by identifying the EDL bit (which may be referred to as the FDF bit). The device may identify the EDL bit by inspecting a bit at a position where it would expect to find the EDL bit in a CAN FD frame and to determine whether the bit is recessive.

The device may be configured to identify whether the frame has a base format or an extended format. The device may be configured to inspect the IDE bit to identify whether the frame has a base format or an extended format.

The device may be configured to detect a first bit (IDE) which precedes the section and to generate a first signal (IDE-RTR) to provide a marker for the first bit.

The device may be configured to use the first signal to locate the start of the section.

The device may be configured, in response to receiving a CAN frame from a local CAN controller, to suppress transmission of the CAN frame in dependence upon whether the substitute data is being generated.

The device may be implemented in hardware logic.

According to a second aspect of the present invention there is provided a CAN transceiver comprising the device and a CAN bus physical layer device circuitry (PHY) comprising an RXD output and an TXD input. The device is arranged to receive RXD from the CAN bus physical layer device circuitry and to provide TXD to the CAN bus physical layer device circuitry.

According to a third aspect of the present invention there is provided a node (a "CAN 2.0B node") comprising a CAN 2.0B controller and the CAN transceiver coupled to the CAN 2.0B controller.

Thus, using a CAN transceiver which include the converter can enable the CAN 2.0B controller to participate passively in CAN FD communication.

According to a fourth aspect of the present invention there is provided a CAN system comprising a CAN bus, at least one CAN 2.0B node and at least one CAN FD node.

According to a fifth aspect of the present invention there is provided a method comprising identifying whether a frame on RXD is a CAN FD frame and, in dependence upon identifying that the frame is a CAN FD frame, replacing a section of the CAN FD frame including the data phase of the CAN FD frame with substitute data having a format which complies with CAN 2.0B.

Brief Description of the Drawings

Certain embodiments of the present invention will now be described, by way of example, with reference to the accompanying drawings, in which:

Figure 1 illustrates a CAN FD frame;

Figure 2 illustrates a CAN FD extended format data frame;

Figure 3 is a schematic block diagram of a CAN network including CAN FD nodes and CAN FD passive nodes;

Figure 4 illustrates conversion of a CAN FD frame into a CAN frame;

Figure 5 is a schematic block diagram of a transceiver which includes a CAN FD/CAN FD passive converter;

Figure 6 is a schematic block diagram a CAN FD detection unit which includes an SOF synchronisation unit, an arbitration phase error detection unit, an FD data phase error detection unit, an EDL flag detection unit and an FD data length detection unit;

Figure 7 is a process flow diagram of a method performed by the synchronisation unit shown in Figure 6;

Figure 8 is a process flow diagram of a method performed by the arbitration phase error detection unit shown in Figure 6;

Figure 9 is a process flow diagram of a method performed by the FD data phase error detection unit shown in Figure 6;

Figure 10 is a process flow diagram of a method performed by the EDL flag detection unit shown in Figure 6;

Figures 11A and 11B illustrate a process flow diagram of a method performed by the FD data length detection unit shown in Figure 6 in a first operating mode;

Figure 12 illustrates a process flow diagram of a method performed by the FD data length detection unit shown in Figure 6 in a second operating mode;

Figure 13 is a schematic block diagram of a RX replacer which includes a receive multiplexer switching state machine;

Figure 14 is a schematic block diagram of a TX gating and dominant bit suppressing unit;

Figure 15 is a timing diagram showing RXD, CRXD and CTXD data signals and IDE-RTR, FDDDAMASK, ENDGEN and GENCOMP control signals illustrating processing of a short CAN FD frame by a CAN FD/CAN FD passive converter operating in a first operating mode;

Figure 16 is a timing diagram showing RXD, CRXD and CTXD data signals and IDE-RTR, FDDDAMASK, ENDGEN and GENCOMP control signals illustrating processing

of a long CAN FD frame by a CAN FD/CAN FD passive converter operating in a first operating mode;

Figure 17 is a timing diagram showing RXD, CRXD and CTXD data signals and IDE-RTR, FDDDAMASK, ENDGEN and GENCOMP control signals illustrating processing of a CAN FD frame by a CAN FD/CAN FD passive converter operating in a first operating mode and a local controller attempting to transmit at the end of transmission of substitute data;

Figure 18 is a timing diagram showing RXD, CRXD and CTXD data signals and IDE-RTR, FDDDAMASK, ENDGEN and GENCOMP control signals illustrating processing of a short CAN FD frame by a CAN FD/CAN FD passive converter operating in a second operating mode; and

Figure 19 is a timing diagram showing RXD, CRXD and CTXD data signals and IDE-RTR, FDDDAMASK, ENDGEN and GENCOMP control signals illustrating processing of a long CAN FD frame by a CAN FD/CAN FD passive converter operating in a second operating mode.

Detailed Description of Certain Embodiments

Controller area network 1

Referring to Figure 3, a controller area network (CAN) 1 is shown. The CAN 1 includes a CAN bus 2 and a plurality of nodes 3, 4 connected to the CAN bus 2. Each node 3, 4 includes a controller 5, 6 which may be embedded in a microcontroller 7, 8 or implemented as a stand-alone device, and a transceiver 9, 10. The transceivers 9, 10 each include circuitry 11, 12 for implementing physical layer requirements of ISO 11898 (2004). The nodes 3, 4 are able to send and receive frames 13 which comply with CAN 2.0B , i.e. ISO 11898-1 (2004). Herein, such frames 13 are referred to as "CAN frames".

Each node 3 of a first type (herein referred to as "CAN FD nodes") includes a controller 5 which complies with the CAN Flexible Data Rate (CAN FD) protocol and, thus, is able to send and receive CAN FD frames 14 (herein also referred to as "FD frames").

Each node 4 of a second type (herein referred to as "CAN FD passive nodes" or "CAN FD tolerant nodes") includes a controller 6 which complies with the CAN 2.0B protocol and thus is able to exchange CAN frames 13. The transceiver 10 includes or is provided with a CAN FD/CAN FD passive converter 15 which enables the CAN FD passive node 4 to be connected to the CAN bus 2 without disturbing CAN FD communication by, for example, generating error frames (not shown) when receiving a CAN FD frame 14.

CAN FD communication can support a CAN FD arbitration rate, for example of up to 500 $kbits^{-1}$ or up to 1 $Mbits^{-1}$, which is typically limited by the processing speed of the controller 5, 6 and a CAN FD data rate, for example of up to 4 $Mbits^{-1}$ or 8 $Mbits^{-1}$ (i.e. up to 8 times the arbitration rate), which is typically constrained by the physical limits of the CAN bus 2.

The CAN FD passive node(s) 4 can, among other things, provide or allow selective wakeup as specified by ISO 11898-6, be tolerant of CAN FD messages, allow "classic" wakeup by dominant edge signalling, provide bus protection by transmit ("TX") low detection and/or operation modes including transparent (active), sleep and selective wakeup (using CAN 2.0B messages).

The CAN FD passive node(s) 4 skip phases of communication which are specific to CAN FD communication thereby allowing it (them) to participate in mixed CAN FD and CAN 2.0B communication. In particular, a CAN FD passive node 4 can participate passively in CAN FD communication and actively in CAN 2.0B communication.

CAN FD frame to CAN substitute frame conversion

Referring also to Figure 4, a CAN FD frame 14 generally comprises a first section 16 which complies with both CAN 2.0B and CAN FD protocols, a second section 17 which is specific to the CAN FD protocol and a third section 18 which may comply with both CAN 2.0B and CAN FD protocols or just the CAN FD protocol. The CAN FD frame 14 is followed by other data 19 (herein referred to as "following data") typically in the form of IFS followed by another frame, which may be a CAN 2.0B frame or a CAN FD frame, and which may comprise a data frame or other type of frame.

The first section 16 of the CAN FD frame 14 starts with a SOF bit. The second section 17 starts with an EDL bit (which may also be referred to as the "FDF bit") and finishes with the ACK delimiter. The content of the first and second sections 16, 17 depends on frame format.

When receiving a CAN FD frame 14, the CAN FD/CAN FD passive converter 15 can process the CAN FD frame 14 by replacing the second and third sections 17, 18 of the frame 14 and a part 20 of the following data 19 with substitute data 21 to form a frame 22 which complies with CAN 2.0B (herein referred to as a "CAN substitute frame",

"generated CAN frame" or "CAN 2.0 compliant frame"). . The CAN substitute frame 22 is followed by additional data 23 including error frame data 24 or overload frame data 25. In some circumstances, for example when a CAN FD 14 is long, a CAN substitute frame 22 may be followed by one or more further CAN substitute frames 22 before the error frame data 24 or overload frame data 25. Furthermore, a CAN substitute frame 22 need not necessarily include the beginning of the CAN FD frame 14, i.e. the first section 16.

As will be explained in more detail hereinafter, the content of the substitute data 21 and, thus, the substitute CAN frame(s) 22, depend on frame format and frame length. The substitute data 21 typically include a dominant reserved bit r0 and DLC bits and may be followed by an empty data section, CRC bits, ACK bits, EOF bits, IFS bits, SOF bits and/or arbitration field bits.

When a CAN 2.0B node 4 transmits a CAN frame 13, the CAN/CAN passive converter 15 can delay transmission of the CAN frame 13 until a CAN FD frame 14 has passed by generating dedicated bus activity sequences which is forwarded to the local controller 6.

As will also be explained in more detail hereinafter, different operational modes may be provided including a first mode which helps to maximise bus participation and a second mode which helps to minimise the number of errors at the local controller 6.

Transceiver 10

Referring to Figure 5, a transceiver 10 comprising the physical layer circuitry (PHY) 12 and the CAN FD/CAN FD passive converter 15 is shown in more detail. Data streams and clock signals are shown as continuous lines, control signals are shown as long dashed lines and configuration signals are shown as short dashed lines.

As shown in Figure 5, the PHY 12 is coupled to CAN high (CAN_H) and CAN low (CAN_L) lines. The PHY 12 outputs a CAN receive data signal (RXD) and receives a CAN transmit data signal (TXD) from the CAN FD/CAN FD passive converter 15.

RXD is supplied to a CAN FD detection unit 51 (herein simply referred to as the "FD detection unit"), a receive signal (RX) replacer unit 52, an RX multiplexer unit 53 and a selective wake-up unit 54 which includes a reference ID and data registers 55.

TXD is supplied to PHY 12 in the form of a transmitter data input signal (CTXDS) from a TX gating and dominant suppression unit 56.

The FD detection unit 51 receives a clock signal (OSC) from oscillator 57. The oscillator 57 may operate at 8 or 16 MHz, or at another frequency. The FD detection unit 51 also receives a set control signal (GENCOMP) from the RX replacer 52 and supplies an end-of-EDL-bit indicator signal (IDE-RTR), a CAN FD data mask signal (FDDAMASK), a CAN bitrate signal (TQCLK), a start of frame synchronisation signal (SOFSYNC) and a frame generation signal (ENDGEN) to the RX replacer 52. The FD detection unit 51 also supplies FDDAMASK to the TX gating and dominant suppression unit 56.

The FD detection unit 51 includes arbitration bit rate and CAN FD bit rate registers 58, 59 which are set using configuration signals (CONFIG) received from Serial Peripheral Interface (SPI) unit 60 which is controlled using SPI signals from the local CAN controller 6 (Figure 3).

The RX replacer 52 is configured using a configuration signal (CONFIG) received from the SPI interface unit 60. The RX replacer 52 outputs a substitute receive data signal (RRXD) which includes replacement data 21 (Figure 4) to the RX multiplexer unit 53.

The selective wake up unit 54 is configured using a configuration signal (CONFIG) received from the SPI interface unit 60. The selective wake-up unit 54 outputs a wake-up signal (WUP) to the RX multiplexer unit 53.

The RX multiplexer unit 53 outputs a CAN receive data signal (CRXD) to the local CAN controller 6 (Figure 3).

The TX gating and dominant suppression unit 56 receives FDDAMASK from the FD detection unit 51 and GENCOMP from the RX replacer unit 52. The TX gating and dominant suppression unit 56 also receives CAN transmit data (CTXD) from the local CAN controller 6 (Figure 3) and an activate control signal (ACTIVATE) from the SPI interface unit 60. The TX gating and dominant suppression unit 56 outputs a CAN transmit data inverted signal (CTXDA) to the RX replacer unit 52.

CAN FD/CAN FD passive converter operation

Referring to Figures 4 and 5, the CAN FD/CAN FD passive converter 15 can operate in different modes.

In a first mode, the CAN FD/CAN FD passive converter 15 can operate in a way which optimises bus performance.

As a CAN FD frame 14 is received, the FD detection unit 51 marks the second and third sections 17, 18 of the frame 14 and the section 19 of data following the frame for replacement. The RX replacer 52 processes the CAN FD frame 14 which is replaced and fed into the RX multiplexer 53 and output as CRXD.

While CTXD is inactive (i.e. recessive), the RX replacer 52 completes a so-far incomplete CRXD frame, i.e. the first section 16 of the frame 14, by replacing the rest of the frame, i.e. the second and third sections 17, 18 of the frame, and a part of IFS, i.e. following data 20, by adding substitute data 21 to form a CAN 2.0B-compliant frame 22. The substitute data 21 is appended after the IDE bit (Figure 2) and includes CRC and positive ACK.

If CTXD becomes active (i.e. dominant) while the CAN FD frame 14 is still being processed, the TX gating and dominant suppression unit 56 suppresses transmission of CTXD to the PHY 12. The TX gating and dominant suppression unit 56 feeds back CTXD in the form of CTXDA to the RX replacer 52 which is triggered by CTXD. Consequently, the RX replacer 52 generates a CAN 2.0B-compliant frame 22 including an arbitration field. During this arbitration field, the frame on CTXD is overwritten by dominant level. This intentionally results in loss of arbitration for the local CAN controller 6 (Figure 3) thereby causing it to stop transmission.

The CAN-compliant frame 22 is arranged to be of minimum length (i.e. DLC = 0) with no data content and is configured to have highest priority in the arbitration field. If the distance to the end of the CAN FD frame 14 is such that it matches with the length of an error frame data 24 (i.e. error frame and error delimiter), generation and output of frames to CRXD is aborted if CRXD is still active and an error frame is sent. Thus, the local CAN controller 6 (Figure 3) may regain access to the CAN bus as soon as the CAN FD frame 14 has ended.

Whenever a CAN FD frame 14 is processed in this way, a RX error counter (not shown) in the local CAN controller 6 (Figure 3) is incremented. Thus, in this mode of operation, the local CAN controller 6 (Figure 3) may reach an "error passive" state.

The timing of the generated error frame is set such that the end of error frame matches the beginning of IFS of the CAN FD bus.

In a second operation mode, the CAN FD/CAN FD passive converter 15 can operate in a way which minimises RX errors, but on the other hand does not allow the local CAN controller 6 to re-join CAN bus communication at a precise point, i.e. the start of IFS.

Like in the first operation mode, when a CAN FD frame 14 is received, the FD detection unit 51 marks the second and third sections 17, 18 of the frame 14 and the section 20 of data following the CAN FD frame 14 for replacement and the RX replacer 52 adds substitute data 21.

However, to avoid CTXD becoming dominant while the CAN FD frame 14 is still being processed, overload frames are generated until the CAN FD frame 14 has ended. During overload frames, the local CAN controller 6 (Figure 3) is unable to start any transmission and so it waits until transmission of overload frames has finished.

If another CAN FD frame 14 starts on the CAN bus during the frame completion or during overload frame generation, further overload frames are generated until the end of the overload frame delimiter matches a bus idle state.

Like in the first mode, the CAN 2.0-compliant frame 22 is arranged to be of minimum length (i.e. DLC = 0).

If the CAN bus load is 100%, then the local CAN controller 6 (Figure 3) is not able to participate in the arbitration unless the end of the internal frame generation or overload delimiter matches an interframe space phase of the CAN bus. However, typically, CAN bus load tends to be no more than 50%. Thus, on average, after a reasonable time, the CAN controller 6 (Figure 3) can usually reconnect to the CAN bus when the end of the internal frame generation or the overload frame delimiter falls within the bus idle or interframe space phase.

FD detection unit 51

Referring to Figure 6, the FD detection unit 51 is shown in more detail.

The FD detection unit 51 includes a CAN arbitration bitrate time quanta clock generation unit 62, a CAN FD data bitrate time quanta clock generation unit 63, an SOF synchronisation unit 64, an arbitration phase error detection unit 65, a CAN FD data error detection unit 66, an EDL flag detection unit 67 and a CAN FD data length detection unit 68.

As shown in Figure 6, the FD detection unit 51 uses a clock signal OSC from an internal oscillator 57 (Figure 5). OSC may be 8 or 16 MHz. A precision of 1.25% is sufficient, i.e. 1 time quanta ("TQ") error within 10 bits using 8 TQ per bit. CAN arbitration bitrate time quanta clock (ATQCLK) and CAN data bitrate time quanta clock (FDTQCLK) can be generated with a lower TQ per bit resolution.

ATQCLK is set via the SPI interface 60 (Figure 5) using CONFIG and is generated by the arbitration bitrate time quanta clock generation unit 62. ATQCLK is used during the arbitration phases of RXD. The arbitration bitrate time quanta clock generation unit 62 generates the bit timing clock based on 8 TQ per bit. ATQCLK is forwarded to the RX replacer 52 (Figure 5) as TQCLK.

FDTQCLK is set via the SPI interface 60 (Figure 5) using CONFIG and is output by CAN FD data bitrate unit 63. FDTQCLK is used during data phases of RXD when a higher bitrate is required. The CAN FD data bitrate time quanta clock generation unit 63 generates the bit timing clock based on 4 TQ per bit.

SOF synchronization unit 64

Referring to Figures 4, 6 and 7, after being reset (step S1-1), the SOF synchronization unit 64 checks for receipt of an arbitration phase error signal (AERR) or a data phase error signal (DERR) from the arbitration phase error detection unit 65 or data phase error detection unit 66 respectively (step S1-2). AERR and DERR flag the occurrence of an overload frame or an error frame, and will cause a reset of the SOF synchronization procedure.

If the SOF synchronization unit 64 does not receive an error signal (either AERR or DERR), it looks for the start of a CAN frame on RXD by searching for a synchronization

pattern comprising the EOF (of a previous frame) and IFS followed by SOF which is set to a dominant bit (i.e. logical '0') (step S1-3).

When the SOF bit is found in RXD, the SOF synchronization unit 64 forwards SOF to the EDL flag detection block 67 (step S1-4).

During arbitration phases, the SOF synchronization unit 64 uses the bit timing of ATQCLK. During (CAN FD) data phases, the SOF synchronization unit 64 uses FDTQCLK to follow the bit stream. The data phases are indicated to the SOF synchronization unit 64 by the EDL flag detection block 67 using BRS.

The SOF synchronization unit 64 is reset if an overload frame or an error frame is received on RXD and will begin to search for a new synchronization pattern (step S1-2).

Arbitration phase error detection unit 65

Referring to Figures 4, 6 and 8, after being reset (step S2-1), the arbitration phase error detection unit 65 detects error frames and overload frames in a RXD signal (RXDA). The EDL flag detection unit 67 forwards RXD as RXDA during a CAN frame 13 or the arbitration phases of a CAN FD frame 14.

When the arbitration phase error detection unit 65 detects an error frame or an overload frame, it sets AERR (steps S2-2 & S2-3).

FD data error detection unit 66

Referring to Figures 4, 6 and 9, after being reset (step S3-1), the FD data error detection unit 66 detects error frames and overload frames in a RXD signal (RXDD). The FD data length detection block 68 forwards RXD as RXDD during a data phase of a CAN frame 13.

When the FD data error detection unit 66 detects an error frame or an overload frame, it sets DERR (steps S3-2 & S3-3).

The FD data error detection unit 66 flags error and overload frames for both standard and FD CAN frames.

EDL flag detection unit 67

Referring to Figures 4, 6 and 10, the EDL flag detection unit 67 searches an incoming frame for the EDL bit (which may be referred to as the FDF bit). The EDL bit is located in a CAN FD frame 14 within a fixed number of bits (or "fixed bit distance") from the SOF.

The EDL flag detection unit 67 clears EDL and BRS (step S4-1) and waits for SOF (step S4-2). Once SOF has been found, it waits for the IDE bit (step S4-3) and marks the bit (for duration of 1 bit) using IDE-RTR (step S4-4).

The EDL flag detection unit 67 reads IDE to determine whether the frame has a base format or an extended format.

If IDE is dominant, then the frame has a base format and so the next bit is expected to be either an EDL bit (for a CAN FD frame) or a reserved bit r0 (for a CAN frame).

If IDE is recessive, then the frame has an extend format and so an extended arbitration field is expected. Thus, an identifier extension is expected, followed by reserved bit r1 (for a CAN FD frame) or RTR (for a CAN frame). After the arbitration field, the next bit is expected to be either an EDL bit (for a CAN FD frame) or a reserved bit r1 (for a CAN frame).

The EDL flag detection unit 67 waits for the bit where it expects to find an EDL bit (step S4-5) and inspects the bit (step S4-6). For a CAN FD frame, the EDL bit is recessive. For a CAN frame, the reserved bit r1 or r0 is dominant.

If the bit is not found set and a dominant bit (i.e. either r1 or r0) is found instead, then the frame is determined to be a CAN frame (i.e. not a CAN FD frame) and so the frame needs no further processing. Thus, the EDL flag detection unit 67 is reset (step S4-1) and waits for a new SOF (step S4-2).

If the bit is found set (i.e. it is recessive), the EDL flag detection unit 67 indicates this finding to the FD data length detection block 68 by setting EDL (step S4-7). In this case, the EDL flag detection unit 67 also checks for and forwards the BRS flag to the FD data length detection block 68 and SOF synchronization blocks 64 to allow them choose the appropriate bit rate (steps S4-8, S4-9 and S4-10).

If the EDL bit is not found set or if the data phase of a CAN FD frame 14 has finished (step S4-11), the EDL flag detection unit 67 is reset (step S4-1) and waits for a new SOF (step S4-2).

If an error in arbitration or data phases is found at any time, as indicated by the signals AERR and DERR respectively (step S4-12), then the EDL flag detection unit 67 is reset (step S4-1).

FD data length detection unit 68

Referring to Figures 4, 5 and 6, ENDGEN and FDDAMASK are used to control the RX replacer 52 which emulates a bus signal to the local (non-FD) CAN controller 6 (Figure 3).

The RX replacer 52 indicates completion of insertion of substitute bits 21 to the FD data length detection unit 68 by setting GENCOMP. While FDDAMASK is set, the local CAN controller 6 (Figure 3) is disconnected from the RXD and is fed with substitute data 21 (Figure 4) by the RX replacer 52 and CTXD (i.e. transmit data from the controller 6) is not fed through CTXDS (i.e. it does not reach the CAN bus).

As explained earlier, the CAN FD/CAN FD passive converter 15 can operate in two different modes. Operation in the two modes are similar, differing mainly in the content of substitute bits 21.

FD data length detection unit 68 in the first operating mode

Referring to Figure 4, 5, 6, 11A and 11B, initially SOF, BRS, AERR, DERR, FDDAMASK and ENDGEN are cleared (step S5.1-1).

FD data length detection unit 68 waits to receive EDL from EDL flag detection unit 67 (step S5.1-2).

If the EDL bit is received (thus indicating the presence of a CAN FD frame 14), the FD data length detection unit 68 sets FDDAMASK (step S5.1-3). FDDAMASK continues to be set until the CAN FD frame 14 and a minimum length (in this case, 3 bits) of IFS (i.e. intermission bits) have passed (step S5.1-3 to S5.1-14). If the EDL bit indicates the presence of a CAN FD frame 14, it can be assumed that the local CAN controller 6

(Figure 3) has lost arbitration and will stop transmitting because the EDL bit has already replaced a dominant bit, i.e. a r0 or r1 flag.

FD data length detection unit 68 sets ENDGEN a given number of bits (in this case 20 bits) before the end of a CAN FD frame 14 (i.e. the duration of an error flag including superposition (i.e. an initial error flag and additional error flags which lengthen the initial error flag) plus error delimiter before at least 3 bits of interframe space) so as to indicate that replacement of the CAN FD frame 14 should end soon (steps S5.1-4 to S5.1-9). To generate ENDGEN, the FD data length detection unit 68 decodes the CAN FD frame 14 (step S5.1-4), determines its length and data rate and predicts the length of the CRC field (step S5.1-5).

GENCOMP is expected from the RX replacer 52 as a response to ENDGEN after FDDAMASK is reset. If GENCOMP is set before any new frame appears on RXD, then processing is determined to have been completed and so the FD data length detection unit 68 is reset and starts to wait for another CAN FD frame 14. Otherwise, FDDAMASK is set again and remains set until the frame and 3 bits of IFS have passed after which ENDGEN is again generated.

The RX replacer 52 sets GENCOMP after ENDGEN is set by the FD data length detection unit 68 and while no more signal generation is taking place on CRXD.

If BRS is set in addition to EDL being, then FD data length detection unit 68 recognises the data phase of the CAN FD frame 14 and uses FDTQCLK instead of ATQCLK.

If an error in the data phase is found and is indicated by DERR, then the FD data length detection unit 68 is reset when FDDAMASK is cleared.

Thus, the FD data length detection unit 68 waits until for the frame and 3 bits of IFS to pass (step S5.1-10). While it waits, the FD data length detection unit 68 checks whether GENCOMP is set (step S5.1-11).

If GENCOMP is set in the meantime, then the FD data length detection unit 68 waits until the frame and 3 bits of IFS, i.e. intermission bits, have passed (step S5.1-12), clears FDDAMASK (step S5.1-13) and is reset (step S5.1-1).

If GENCOMP is not set in the meantime, then once the frame and 3 bits of IFS have passed, the FD data length detection unit 68 clears FDDAMASK (step S5.1-14).

The FD data length detection unit 68 checks whether GENCOMP has been set (step S51.15). If GENCOMP has been set, then the FD data length detection unit 68 is reset (step S5.1-1). If GENCOMP has not been set, then the FD data length detection unit 68 waits until a new frame is received on RXD (step S5.1-16). If a new frame is received, FDDAMASK is set again and remains set until the frame and 3 bits of IFS have passed after which ENDGEN is again generated (steps S5.1-4 to S5.1-9).

FD data length detection unit 68 in the second operating mode

Referring to Figure 4, 5, 6 and 12, initially SOF, BRS, AERR, DERR, FDDAMASK and ENDGEN are cleared (step S5.2-1).

FD data length detection unit 68 waits to receive EDL from EDL flag detection unit 67 (step S5.2-2).

If the EDL bit is received (thus indicating the presence of a CAN FD frame 14), FD data length detection unit 68 sets FDDAMASK (step S5.2-3). FDDAMASK continues to be set until the CAN FD frame 14 ends and IFS is reached (step S5.2-4).

When IFS has been reached, ENDGEN indicates that replacement of the CAN FD frame 14 should end (step S5.2-5).

GENCOMP is expected from the RX replacer 52 as a response to ENDGEN being set.

If GENCOMP is set before the next frame appears on RXD, then processing is deemed to have been completed and the FD data length detection unit 68 is reset and waits for the next CAN FD frame 14.

If GENCOMP is clear, FDDAMASK is still set and a new frame appears on RXD, then the frame is also processed. A new ENDGEN signal is generated at the start of IFS, after this frame.

The RX replacer 52 sets GENCOMP when ENDGEN is set by the FD data length detection unit 68 and no signal generation is ongoing on CRXD.

If BRS is also set (in addition to EDL), then FD data length detection unit 68 recognises the data phase of the CAN FD frame 14 and uses FDTQCLK instead of ATQCLK.

If an error in the data phase is found and is indicated by DERR, then the FD data length detection unit 68 is reset when FDDAMASK is cleared.

Thus, the FD data length detection unit 68 checks whether a new frame has been received (step S5.2-6). If so (i.e. a new frame has been received), then a new ENDGEN signal is generated at the start of IFS, after the new frame (step S5.2-4 & S5.2-5).

However, if no new frame has been received, then the FD data length detection unit 68 continues to check whether GENCOMP has been set until a new frame is received (step S5-2.7). If GENCOMP has not been set, then the FD data length detection unit 68 continues to check for a new frame (step S5-2.7). Once a new frame is received, the FD data length detection unit 68 keeps FDDAMASK set until this frame has passed too (IFS reached). Otherwise, if GENCOMP has been set after S5.2-6, FDDAMASK is cleared (S5.2-8) and the processing is completed.

RX replacer unit 52

Referring to Figure 13, the RX replacer unit 52 is shown in more detail.

The RX replacer unit 52 includes a receive stream multiplexer 72, a receive multiplexer switching state machine 73 and a set of generators 74 for generating substitute data 21 (Figure 4).

The generators 74 include an IFS/idle generator 75, an error and overload frame generator 76, an SOF and arbitration field generator 77, a dominant r1/r0 bit generator 78, a DLC (indicating no data) field generator 79, a CRC field generator 80 and an ACK and EOF field generator 81.

In response to receiving a first activate signal (A1) from the receive multiplexer switching state machine 73, the IFS/idle generator 73 provides IFS /idle to the receive stream multiplexer 72. In response to receiving a second activate signal (S2), the error and overload frame generator 76 provides an error or overload frame (depending on the mode of operating) to the receive stream multiplexer 72 and, when finished, returns

a complete signal (C2) to the receive multiplexer switching state machine 73. In response to receiving an third activate signal (S3), the SOF and arbitration field generator 77 generates SOF and arbitration field bits and, when finished, returns a complete signal (C3). In response to receiving a fourth activate signal (A4), the dominant bit generator 75 provides an appropriate dominant bit r1, r0. In response to receiving a fifth activate signal (S5), the DLC/no data phase signal generator 79 generates a DLC/no data phase bits and, when finished, returns a complete signal (C5).

In response to receiving a sixth activate signal (S6) from the receive multiplexer switching state machine 73 and receiving data bits from the receive stream multiplexer 72, the CRC field generator 80 provides CRC data to the receive stream multiplexer 72 and, when finished, returns a complete signal (C6) to the receive multiplexer switching state machine 73. In response to receiving a seventh activate signal (S7), the ACK and EOF field generator 81 generates ACK and EOF field and, when finished, returns a complete signal (C7) to the receive multiplexer switching state machine 73.

Receive stream multiplexer 72

Referring still to Figure 13, depending on a selection signal (RSTRSEL) received from the receive multiplexer switching state machine 73, the receive stream multiplexer 72 forwards either the stream received on RXD or the output of one of the generators 74 to RRXD. If the output of a generator 74 is forwarded to RRXD, then the local CAN controller 6 (Figure 3) does not see the CAN bus signal RXD. Thus, the local CAN controller 6 (Figure 3) is effectively decoupled from the CAN bus.

Receive multiplexer switching state machine 73

The state machine 73 activates appropriate generators 75, 76, 77, 78, 79, 80, 81 using activate signals A1, S2, S3, A4, S5, S6, S7 depending on input conditions and watches completion signals C2, C3, C5, C6, C7.

Referring to Figures 4 and 13, the state machine 73 controls whether the multiplexer 72 forwards RXD or output from the generators 75, 76, 77, 78, 79, 80, 81 using RSTRSEL. When RSTRSEL is set, the output from a generator 75, 76, 77, 78, 79, 80, 81 is placed on RRXD.

When CAN frames 13 (Figure 3) are forwarded to RRXD, the state machine 73 operates using TQCLK.

Operation of the state machine 73 when operating in the first mode will now be described in more detail:

If SOFSYNC is set and none of the generator 75, 76, 77, 78, 79, 80, 81 are currently active, then state machine 73 is reset. If, however, SOFSYNC is set and a generator 75, 76, 77, 78, 79, 80, 81 is currently active (for example, because at least one of the completion signals C2, C3, C5, C6, C7 is set), then state machine 73 is not reset until all of the generators 75, 76, 77, 78, 79, 80, 81 are inactive.

When IDE-RTR is set thereby indicating that an EDL flag will follow, then the EDL flag is replaced by a r1/r0 bit (i.e. in dominant state). Thus, any frame that become visible on RRXD are marked as CAN frames, i.e. non-FD frames.

When FDDAMASK is set, a substitute frame 22 is generated on RXXD by appending a DLC field indicating a zero-length data field followed by CRC, ACK, EOF and IFS fields thereby replacing the stream received on RXD. If FDDAMASK is still active after these fields have been inserted, then a bus idle signal is generated and placed on RRXD.

If the signal ENDGEN is set (thereby indicating that signal generation should be completed), any on-going frame generation is aborted by inserting an error frame.

If the local CAN controller 6 (Figure 3) tries to start sending a frame on TXD by transmitting a dominant SOF (which signalled to the RX replacer 52 by the CTXDA flag) while FDDAMASK is active and a bus idle signal is currently on RRXD (i.e. after generation of a substitute frame 22 has been completed), the generator generates a competing frame on RRXD which includes an arbitration field, followed by a zero-length indicating DLC field, CRC and ACK fields. Accordingly, transmission by the local CAN controller 6 (Figure 3) is aborted due to loss of arbitration during the arbitration field and no transmission error being generated. The arbitration field generated by the RX replacer unit 52 has higher priority than the arbitration field of any frame originating from the local CAN controller 6.

When FDDAMASK is inactive, the signal RXD is forwarded to RRXD.

If FDDAMASK is reset and no signal generation is currently on-going and ENDGEN has been set, then GENCOMP is set.

Any bus activity on TXDA while FDDAMASK is set causes generation of another SOF, arbitration field and zero-length data phase. If, in the meanwhile, ENDGEN is set and FDDAMASK is reset, substitute frame generation is aborted with an error frame and GENCOMP is set after completion of error frame generation.

In order to generate the CRC field correctly, the de-stuffed bit stream received on RXD is considered during the arbitration phase of the CAN FD frame 14. Therefore, the RX replacer 52 records all frame content received on RXD during the arbitration phase of the replaced CAN FD frame and calculates the CRC during this process taking into account any appended or generated frame data parts.

Functioning of RX replacer 52 can be suppressed by using the RX multiplexer 53 (Figure 5) to select the RXD. This can be set using the SPI interface 60 (Figure 5).

Operation of the state machine 73 when operating in the second mode will now be described in more detail:

If SOFSYNC is set and none of the generators 75, 76, 77, 78, 79, 80, 81 are currently active, then state machine 73 is reset. If, however, SOFSYNC is set and a generator 75, 76, 77, 78, 79, 80, 81 is currently active (for example, because at least one of the completion signals C2, C3, C5, C6, C7 is set), then state machine 73 is not reset until all of the generators 75, 76, 77, 78, 79, 80, 81 are inactive.

When IDE-RTR is set thereby indicating that an EDL flag will follow, then the EDL flag is replaced by a r1/r0 bit (i.e. in dominant state). Thus, any frame that become visible on RRXD is marked as a CAN frame, i.e. a non-FD frame.

When FDDAMASK is activated, a DLC field indicating zero-length data field, a CRC field, ACK field, EOF field and IFS are placed on RRXD thereby replacing the stream received on RXD by a substitute frame 22. If FDDAMASK is still active after the frame has been inserted on RRXD, overload frames are repeatedly placed on RRXD to suppress any attempts to transmit frames by the local CAN controller 6 (Figure 3).

To generate the CRC field correctly, the de-stuffed bit stream RXD is considered for the arbitration phase of the CAN FD frame. Therefore, the RX-replacer 52 records all frame content at RXD in the arbitration phase of the replaced CAN FD frame and calculates the CRC during this process taking into account appended or generated frame data parts, too.

The functionality of this block can be suppressed by using the RX MUX function to select the RXD input directly by means of the SPI interface.

In this operation mode, CTXDA is not used since it indicates an attempt by the local CAN controller 6 (Figure 3) to transmit. However, such a transmission is suppressed by overload frames.

TX gating/dominant suppression unit 53

Referring to Figure 14, the TX gating/dominant suppression unit 53 is shown in more detail.

The unit 53 includes an activation flag (ACTIVATE) from the SPI interface 161, a transmit stream multiplexer 82, a bus idle signal generator 83, a transmit multiplexer switching state machine 84 and an inverter 85.

Transmit Stream Multiplexer 82

Depending on the selection signal (TSTRSEL) from the transmit multiplexer switching state machine 84, the transmit stream multiplexer 82 forwards either CTXD or a bus idle signal (which is recessive level) to CTXDS.

The transmit multiplexer switching state machine 84 can be activated and de-activated by configuration using the signal ACTIVATE.

Transmit multiplexer switching state machine 84

As soon as FDDAMASK is set, the state machine 84 sets TSTRSEL thereby selecting the bus idle signal (which is recessive) as the output for CTXDS. TSTRSEL remains set and the recessive bus idle signal remains selected until GENCOMP indicates the completion of internal signal generation.

Examples

Figures 15 to 19 illustrate examples of the CAN FD/CAN FD passive converter 15 (Figure 3) in operation.

Figure 15 illustrates processing of a CAN FD frame 14 by the CAN FD/CAN FD passive converter 15 (Figure 5) operating in a first mode. The CAN FD frame 14 is short, i.e. shorter than CAN substitute frame 22 which is used to replace it.

As shown in Figure 15, the CAN FD frame 14 is received on RXD and includes a SOF field, an arbitration field, EDL and BRS bits, the rest of the control field, a data field and a CRC field, an ACK field and EOF field. . The frame 14 is followed by IFS, bus idle and then a (standard) CAN 2.0B frame including SOF, an arbitration field, a control field, a data field and a CRC field, and an ACK field.

Referring to Figures 5 and 15, the RX replacer 52 transmits the SOF and arbitration fields to RRXD which pass through RX multiplexer 53 to CRXD. The FD detection unit 51 signals the EDL bit by setting IDE-RTR and then sets FDDAMASK.

The RX replacer 52 sets RSTRSEL (Figure 13) and starts generation of substitute bits 21. The RX replacer 52 generates IDE (in the case of a base format frame) or the reserved bit r1 (in the case of an extended format frame), the reserved bit r0 and DLC bits indicating zero length data field which are placed on RRXD and pass through RX multiplexer 53 to CRXD. IDE/reserved bit r0 is not shown for clarity.

The FD detection unit 51 determines that the end of the CAN FD 14 is approaching (i.e. that is 20 bits away from the start of IFS) so sets ENDGEN. Activation of ENDGEN causes the RX replacer 52 to stop its substitute frame generation and to generate an error frame and error delimiter which are placed on RRXD and pass through RX multiplexer 53 to CRXD.

The FD detection unit 51 de-activates FDDAMASK after the CAN FD frame has passed 14. The RX replacer 52 resets RSTRSEL (Figure 13) after it has completed the error frame and acknowledges that generation of substitute bits 21 has stopped by setting GENCOMP.

As shown in Figure 15, a CAN frame is subsequently received on RXD. The RX replacer 52 transmits the CAN frame to RRXD which passes through RX multiplexer 53 to CRXD.

Figure 16 illustrates processing of a CAN FD frame 14 by the CAN FD/CAN FD passive converter 15 (Figure 5) operating in a first mode. The CAN FD frame 14 is long, i.e. longer than a CAN substitute frame 22 which is used to replace it.

As shown in Figure 16, the CAN FD frame 14 is received on RXD and includes a SOF field, an arbitration field, EDL and BRS bits, the rest of the control field, a data field and a CRC field, an ACK field and EOF field. The frame 14 is followed by IFS.

Referring to Figures 5 and 16, the RX replacer 52 transmits the SOF and arbitration fields to RRXD which pass through RX multiplexer 53 to CRXD. The FD detection unit 51 signals the EDL bit by setting IDE-RTR and then sets FDDAMASK.

The RX replacer 52 sets RSTRSEL (Figure 13) and starts generation of substitute bits 21. The RX replacer 52 generates IDE (in the case of a base format frame) or the reserved bit r1 (in the case of an extended format frame), the reserved bit r0 and DLC bits indicating zero length data field, then CRC, ACK, EOF and IFS data and then idle which are placed on RRXD and pass through RX multiplexer 53 to CRXD which are placed on RRXD and pass through RX multiplexer 53 to CRXD. IDE/reserved bit r0 is not shown for clarity.

The RX replacer 52 determines whether FDDAMASK has been set. Since FDDAMASK is still set and the local CAN controller 6 (Figure 3) is starting to send a frame (CTXDA set by SOF), the RX replacer 52 continues to generate substitute bits 21 for the substitute frame 22. The RX replacer 52 generates SOF and arbitration fields so that the local CAN controller 6 (Figure 3) loses the arbitration.

The FD detection unit 51 determines that the end of the CAN FD 14 is approaching (i.e. that is 20 bits away from the start of IFS) so sets ENDGEN. Activation of ENDGEN causes the RX replacer 52 to abort its substitute frame 22 generation and to generate an error frame and error delimiter which are placed on RRXD and pass through RX multiplexer 53 to CRXD.

The FD detection unit 51 de-activates FDDAMASK after the CAN FD frame 14 is complete and the CAN bus is idle. The RX replacer 52 resets RSTRSEL (Figure 13) and acknowledges that generation of substitute bits 21 has stopped with the completion of the error frame including delimiter by setting GENCOMP.

Figure 17 illustrates how the CAN FD/CAN FD passive converter 15 (Figure 5) reacts when the local controller 6 (Figure 3) tries to start transmitting towards the end a stream of substitute data when it is operating in the first mode.

As shown in Figure 17, the local CAN controller 6 (Figure 3) tries to start transmitting on CTXD while the RX replacer 52 generates substitute data 21.

Referring to Figures 5 and 17, if the local CAN controller 6 (Figure 3) tries to start transmitting on CTXD while the RX replacer 52 generates substitute data 21, the RX replacer 52 does not generate GENCOMP even though FDDAMASK indicates that the CAN FD frame 14 has ended. Thus, substitute frame 22 generation continues.

If another frame (where it be a CAN frame or a CAN FD frame) appears on the bus (i.e. RXD), then the RX replacer 52 continues to generate the substitute frame 22. This is because the FD detection unit 52 did not receive GENCOMP in response to ENDGEN (not shown) and so it sets FDDAMASK again.

If the local CAN controller 6 (Figure 3) again tries sending a frame, then this triggers another substitute frame 22 to be generated. The FD detection unit 51 sets ENDGEN once again, if the frame on the bus reaches its end. Substitute frame generation is aborted with an error frame and error frame delimiter.

The local controller 6 (Figure 3) should allow the error frame including its delimiter to pass. Since this frame is in line with the end of the frame on the bus, then, after FDDAMASK and GENCOMP are cleared, the local controller 6 (Figure 3) may finally send its pending frame. RSTRSEL is cleared and so this frame reaches CTXD and is fed back on RXD.

Figure 18 illustrates processing of a CAN FD frame 14 by the CAN FD/CAN FD passive converter 15 (Figure 5) operating in a second mode. The CAN FD frame 14 is short, i.e. shorter than a CAN substitute frame 22 which is used to replace it.

As shown in Figure 18, the CAN FD frame 14 is received on RXD and includes a SOF field, an arbitration field, EDL and BRS bits, the rest of the control field, a data field and a CRC field, an ACK field and EOF field. The frame 14 is followed by IFS, then a CAN frame including SOF, an arbitration field, a control field, a data field and a CRC field, an ACK field and an EOF field, followed by IFS, after which the bus is idle.

Referring to Figures 5 and 18, the RX replacer 52 transmits the SOF and arbitration fields to RRXD which pass through RX multiplexer 53 to CRXD. The FD detection unit 51 signals the EDL bit by setting IDE-RTR and then sets FDDAMASK.

The RX replacer 52 sets RSTRSEL (Figure 13) and starts generation of substitute bits 21. The RX replacer 52 generates IDE (in the case of a base format frame) or the reserved bit r1 (in the case of an extended format frame), the reserved bit r0 and DLC bits indicating zero length data field, then CRC, ACK, EOF and IFS data which are placed on RRXD and pass through RX multiplexer 53 to CRXD which are placed on RRXD and pass through RX multiplexer 53 to CRXD. IDE/reserved bit r0 is not shown for clarity.

The FD detection unit 51 determines that IFS has been reached and so sets ENDGEN. However, the RX replacer 52 ignores this indication because it has not completed signal generation, i.e. substitute frame generation is still on-going and overload frame generation has not yet started. The FD detection unit 51 checks whether another SOF has been received in the meantime. Since SOF has been received and GENCOMP has not been set, the FD detection unit 51 does not reset FDDAMASK.

The RX replacer 52 completes generation of substitute data 21 including EOF and IFS data. After that, the RX replacer 52 generates overload frames and overload delimiter which are placed on RRXD. At this point, the RX replacer 52 can accept ENDGEN signal, stop the overload frame generation and then respond with a GENCOMP signal.

If a new frame is received, then the FD detection unit 51 determines that IFS has been reached, and sets ENDGEN. After the RX replacer 52 has responded with GENCOMP, the FD detection unit 51 resets FDDAMASK. With deactivation of FDDAMASK, the RX replacer 52 resets RSTRSEL (Figure 13).

Figure 19 illustrates processing of a CAN FD frame 14 by the CAN FD/CAN FD passive converter 15 (Figure 5) operating in a second mode. The CAN FD frame 14 is long, i.e. longer than a substitute frame 22 which is used to replace it.

As shown in Figure 19, the CAN FD frame 14 is received on RXD and includes a SOF field, an arbitration field, EDL and BRS bits, the rest of the control field, a data field and a CRC field, an ACK field and EOF field. The frame 14 is followed by IFS after which the bus becomes idle.

Referring to Figures 5 and 19, the RX replacer 52 transmits the SOF and arbitration fields to RRXD which pass through RX multiplexer 53 to CRXD. The FD detection unit 51 signals the EDL bit by setting IDE-RTR and then sets FDDAMASK.

The RX replacer 52 sets RSTRSEL (Figure 13) and starts generation of substitute bits 21. The RX replacer 52 generates IDE (in the case of a base format frame) or the reserved bit r1 (in the case of an extended format frame), the reserved bit r0 and DLC bits indicating zero length data field, then CRC, ACK, EOF and IFS data and then idle which are placed on RRXD and pass through RX multiplexer 53 to CRXD which are placed on RRXD and pass through RX multiplexer 53 to CRXD. IDE/reserved bit r0 is not shown for clarity.

After that, while FDDAMASK is active and ENDGEN is inactive, the RX replacer 52 starts and continues to send overload frames to prevent the local CAN controller 6 (Figure 3) from trying to start sending a message.

When the FD detection unit 51 recognizes IFS (i.e. that the CAN FD frame has passed), it activates ENDGEN. Substitute frame generation has already finished and overload frame generation is active. Thus, the RX replacer 52 recognizes ENDGEN after completion of generation of the current overload frame and overload delimiter. Then, to confirm completion, the RX replacer 52 sets the GENCOMP signal. Accordingly, the FD detection unit clears FDDAMASK.

In the examples shown in Figures 15 to 18, the CAN FD frames can be in base format or in extended format. If a CAN FD frame has a base format, then the CAN frame having a base format can replace it. Likewise, if a CAN FD frame has an extended format, then the CAN frame having an extended format can be used to replace it. If more than one

CAN substitute frame is generated, then the CAN substitute frames need not be all of the same format, i.e. base or extended format.

It will be appreciated that many modifications may be made to the embodiments hereinbefore described.

For example, high-level functional blocks (such as the FD detection unit and RX replacer) may be sub-divided differently into lower-level functional blocks. Additional functional blocks may be provided.

Claims

1. A device (15) configured to identify whether a frame on RXD is a CAN FD frame (14) and, in dependence upon identifying that the frame is a CAN FD frame, to replace a section (17) of the CAN FD frame, which includes the data phase of the CAN FD frame, with substitute data (21) having a format which complies with CAN 2.0B.

2. A device (15) according to claim 1, wherein the section (17) starts with the EDL/FDF bit.

3. A device (15) according to claim 1 or 2, wherein the section (17) ends with the CRC delimiter.

4. A device (15) according to any preceding claim, which is configured to replace data following the section (17) with further substitute data (21) having a format which complies with CAN 2.0B.

5. A device (15) according to any preceding claim, wherein the substitute data (21) includes the DLC.

6. A device (15) according to any preceding claim, wherein the device is configured to add additional data (24, 25) after the substitute data (21).

7. A device (15) according to claim 6, wherein, in a first mode, the additional data (24, 25) comprises an error frame and an error delimiter.

8. A device (15) according to claim 7, wherein the device is configured to identify a point given number of bits before the end of the CAN FD frame and to add the error frame and error delimiter such that the error delimiter ends when the CAN FD frame ends.

9. A device (15) according to any preceding claim, wherein in a second mode, the additional data (24, 25) comprises at least overload frame) and corresponding overload delimiter.

10. A device (15) according to any preceding claim, which is configured to identify whether the frame is a CAN FD frame by identifying the EDL/FDF bit.

11. A device (15) according to any preceding claim, which is configured to identify whether the frame has a base format or an extended format.

12. A device (15) according to any preceding claim which is configured, in response to receiving a CAN frame from a local CAN controller, to suppress transmission of the CAN frame in dependence upon whether the substitute data (21) is being generated.

13. A CAN transceiver (10) comprising:
a device (15) according to any preceding claim; and
a CAN bus physical layer device circuitry (PHY),
wherein the device is arranged to receive RXD from the CAN bus physical layer device circuitry and to provide TXD to the CAN bus physical layer device circuitry.

14. A node (4) comprising:
a CAN 2.0B controller (6); and
a CAN transceiver (10) according to claim 13 coupled to the CAN 2.0B controller (6).

15. A method comprising:
identifying whether a frame on RXD is a CAN FD frame (14); and
in dependence upon identifying that the frame is a CAN FD frame, replacing a section (17) of the CAN FD frame including the data phase of the CAN FD frame with substitute data (21) having a format which complies with CAN 2.0B.

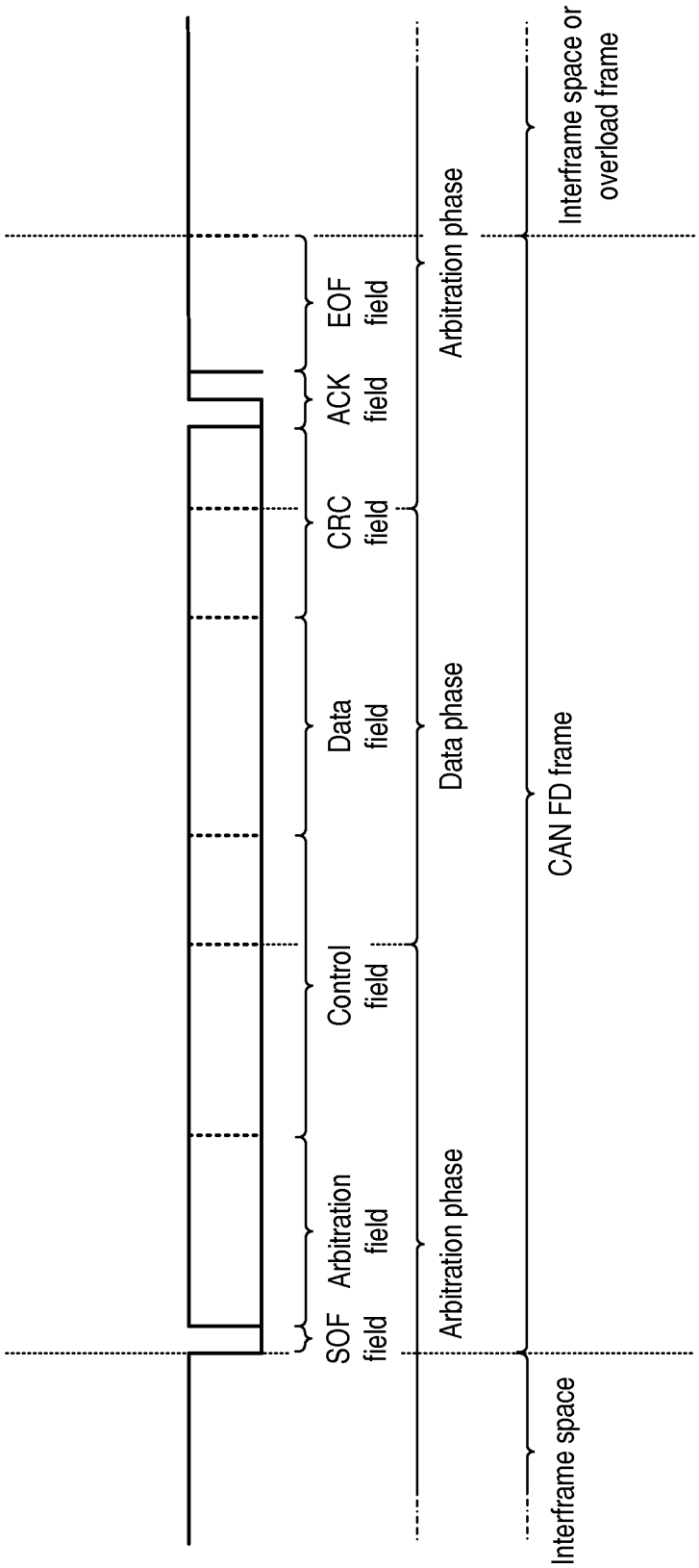

SOF field
Arbitration field
Control field
Data field
CRC field
ACK field
EOF field
Interframe space
Arbitration phase
Data phase
Arbitration phase
CAN FD frame
Interframe space or overload frame

Fig. 1

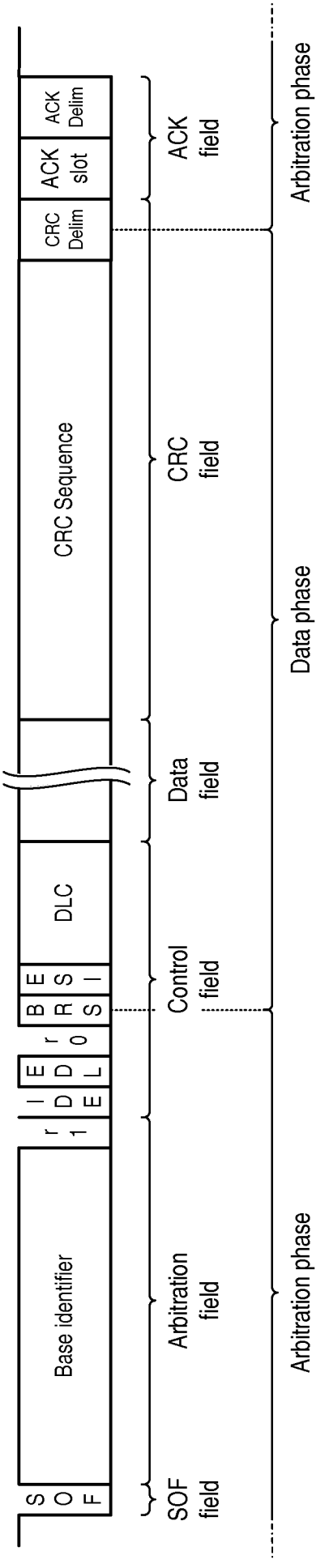

SOF
Base identifier
r1
IDE
EDL
r0
BRS
DLC
CRC Sequence
CRC Delim
ACK slot
ACK Delim
SOF field
Arbitration field
Control field
Data field
CRC field
ACK field
Arbitration phase
Arbitration phase
Data phase

Fig. 2

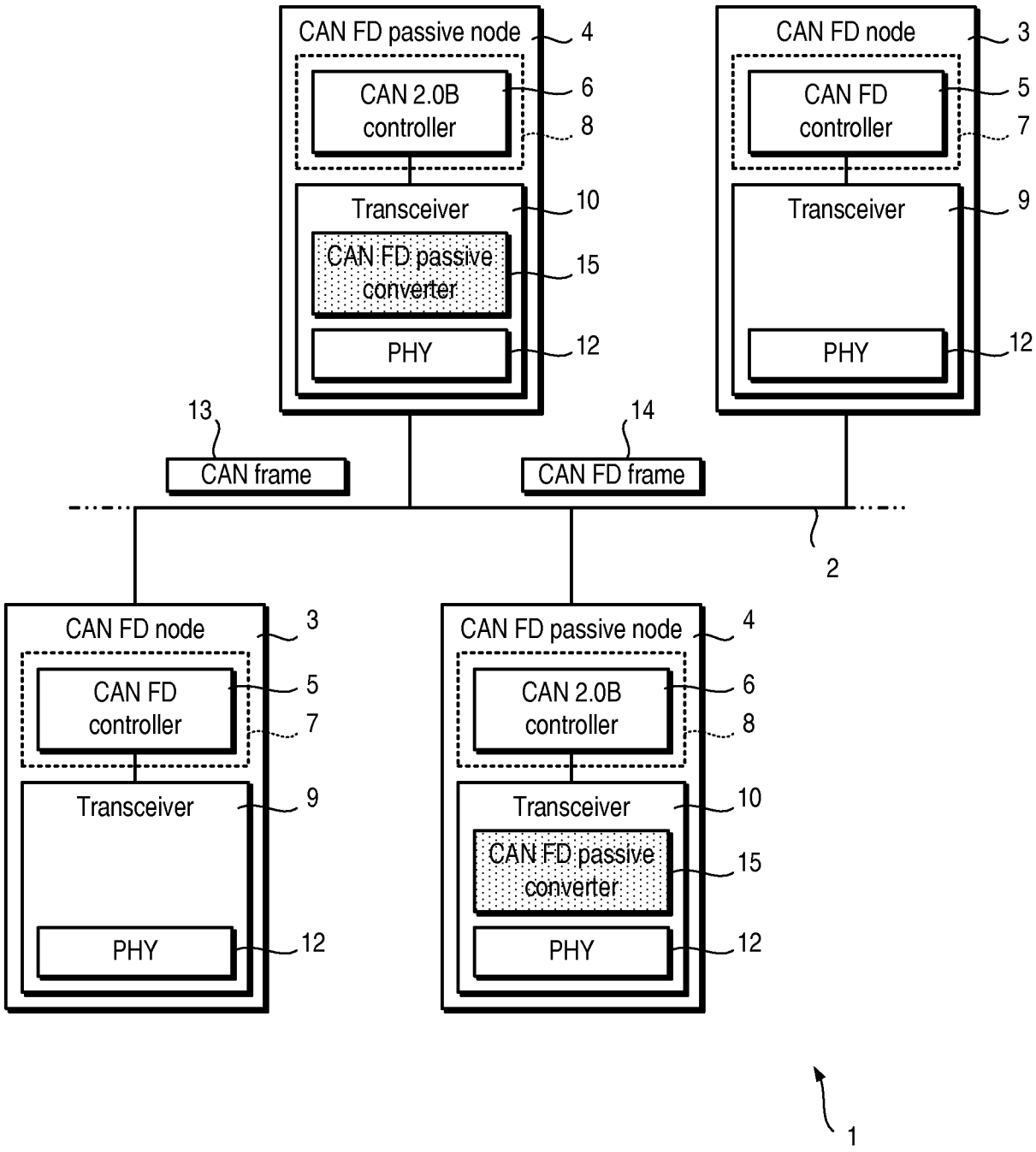

CAN FD passive node
4
CAN 2.0B controller
6
8
Transceiver
10
CAN FD passive converter
15
PHY
12
CAN FD node
3
CAN FD controller
5
7
Transceiver
9
PHY
12
13
CAN frame
14
CAN FD frame
2
CAN FD node
3
CAN FD controller
5
7
Transceiver
9
PHY
12
CAN FD passive node
4
CAN 2.0B controller
6
8
Transceiver
10
CAN FD passive converter
15
PHY
12
1

Fig. 3

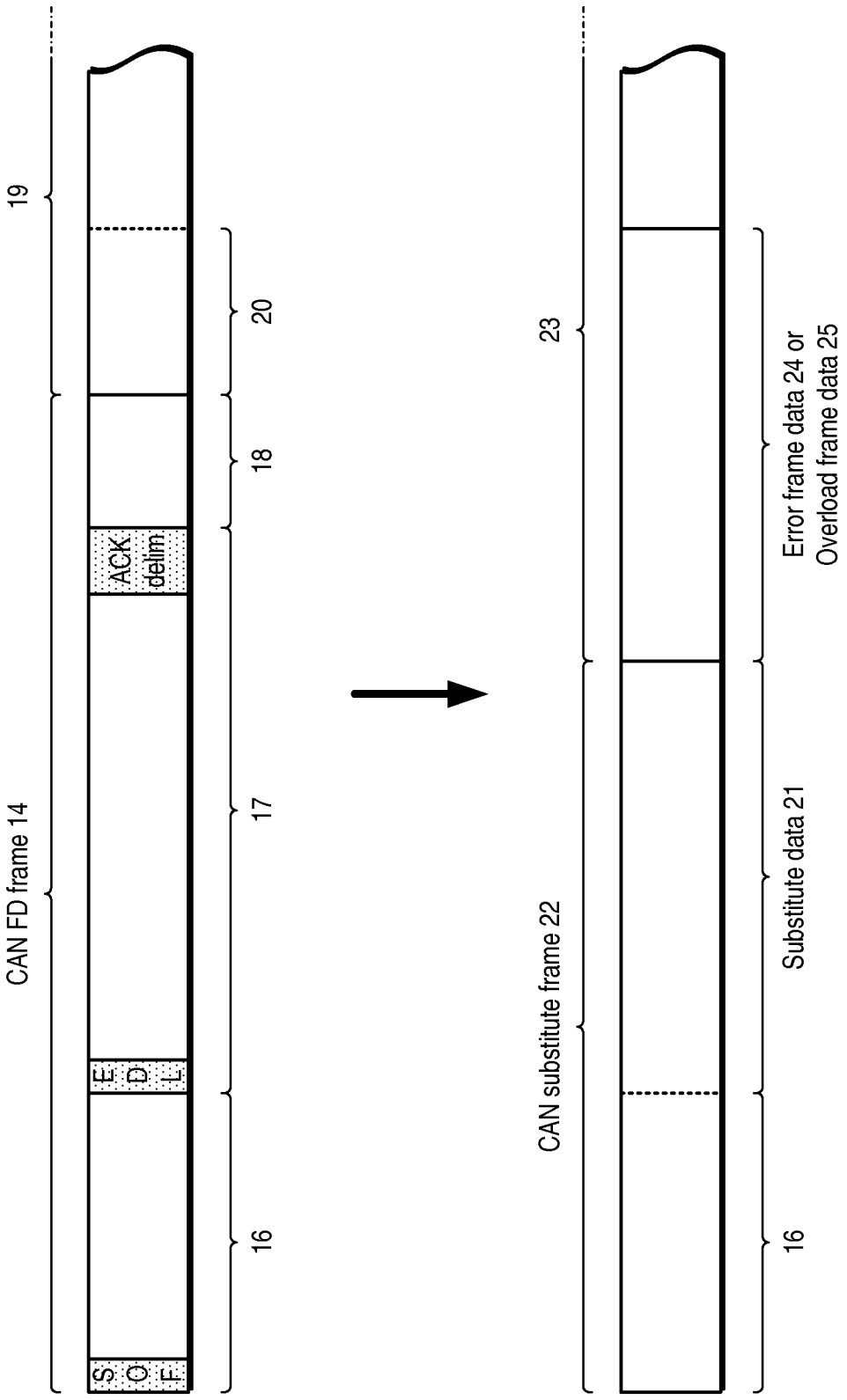

19
CAN FD frame 14
S O F
E D L
ACK delim
16
17
18
20
23
CAN substitute frame 22
Substitute data 21
Error frame data 24 or
Overload frame data 25
16

Fig. 4

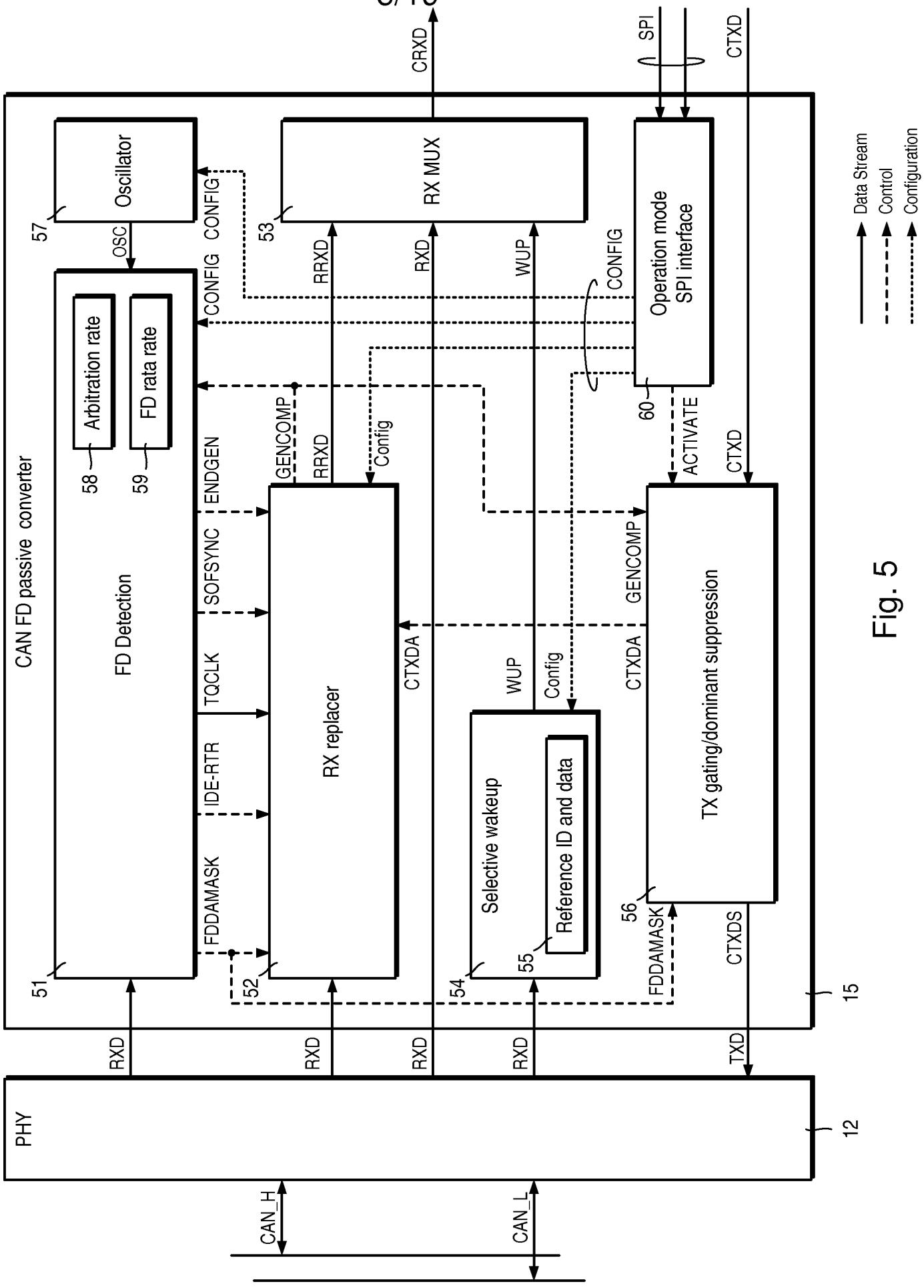

CRXD
SPI
CTXD
Oscillator
RX MUX
Operation mode
SPI interface
Data Stream
Control
Configuration
57
OSC
CONFIG
53
RRXD
RXD
WUP
CONFIG
60
ACTIVATE
CTXD
Arbitration rate
FD rata rate
58
59
CAN FD passive converter
FD Detection
ENDGEN
SOFSYNC
TQCLK
IDE-RTR
FDDAMASK
GENCOMP
RRXD
Config
RX replacer
CTXDA
WUP
Config
Selective wakeup
Reference ID and data
55
54
GENCOMP
CTXDA
TX gating/dominant suppression
56
FDDAMASK
CTXDS
51
52
15
RXD
RXD
RXD
RXD
TXD
PHY
12
CAN_H
CAN_L

Fig. 5

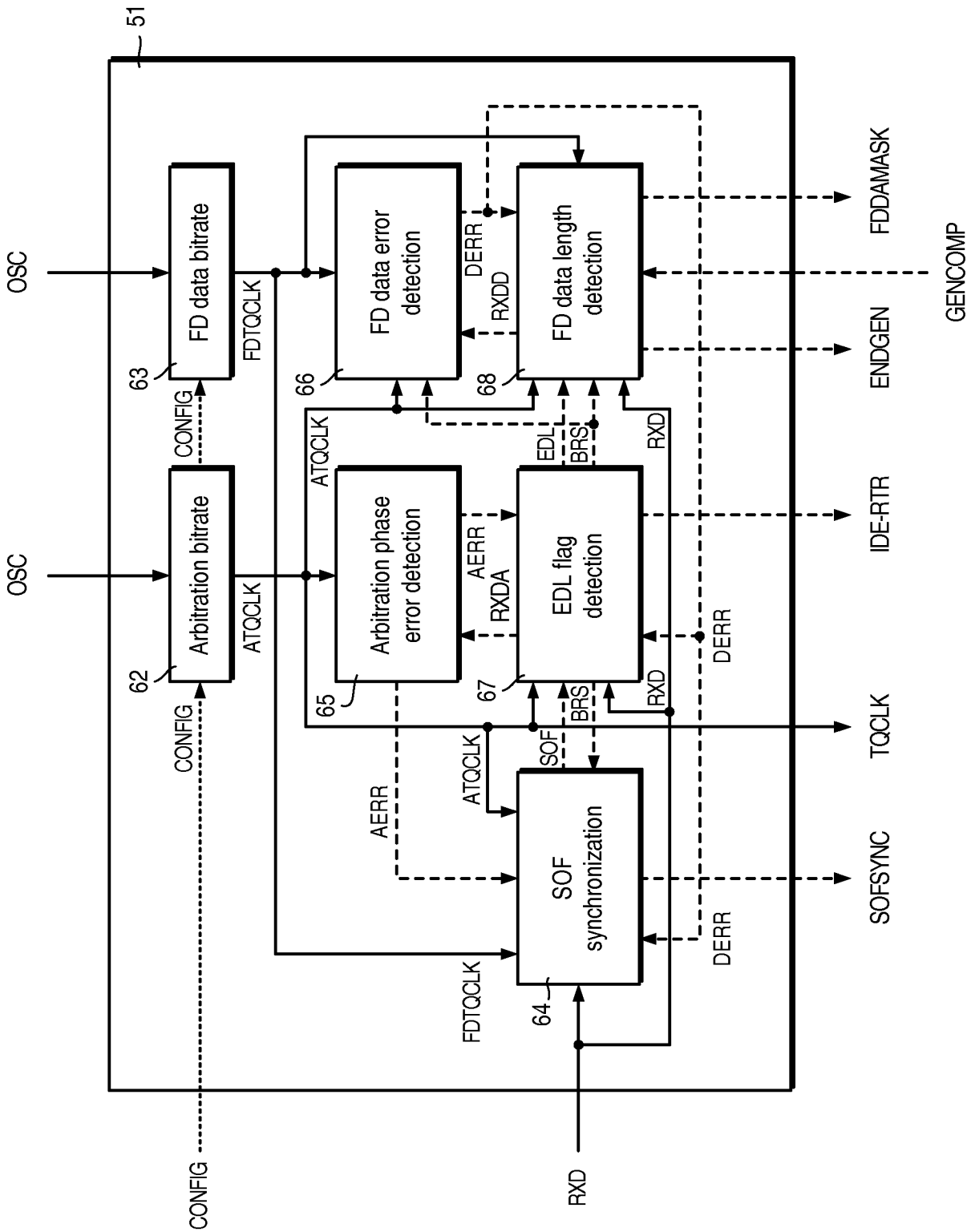

51
OSC
OSC
FD data bitrate
Arbitration bitrate
FD data error detection
FD data length detection
Arbitration phase error detection
EDL flag detection
SOF synchronization
63
62
66
68
65
67
64
CONFIG
CONFIG
CONFIG
FDTQCLK
ATQCLK
ATQCLK
ATQCLK
FDTQCLK
DERR
RXDD
AERR
RXDA
EDL
BRS
RXD
BRS
RXD
SOF
AERR
DERR
DERR
RXD
FDDAMASK
GENCOMP
ENDGEN
IDE-RTR
TQCLK
SOFSYNC

Fig. 6

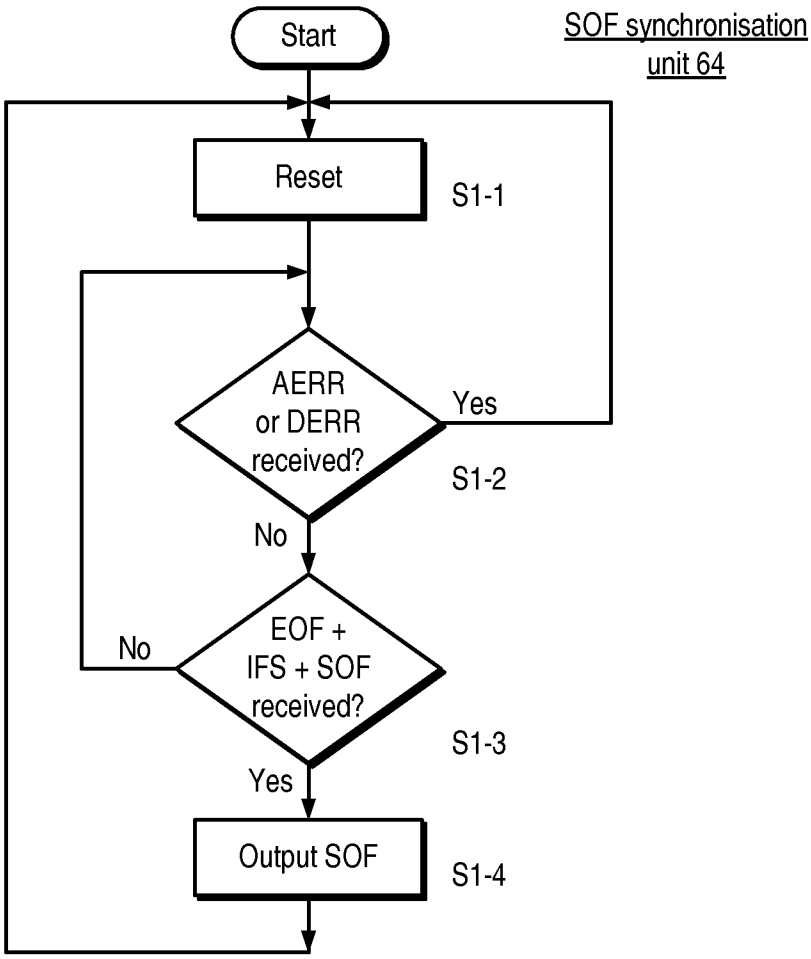

Start
SOF synchronisation unit 64
Reset
S1-1
AERR or DERR received?
Yes
S1-2
No
EOF + IFS + SOF received?
No
S1-3
Yes
Output SOF
S1-4

Fig. 7

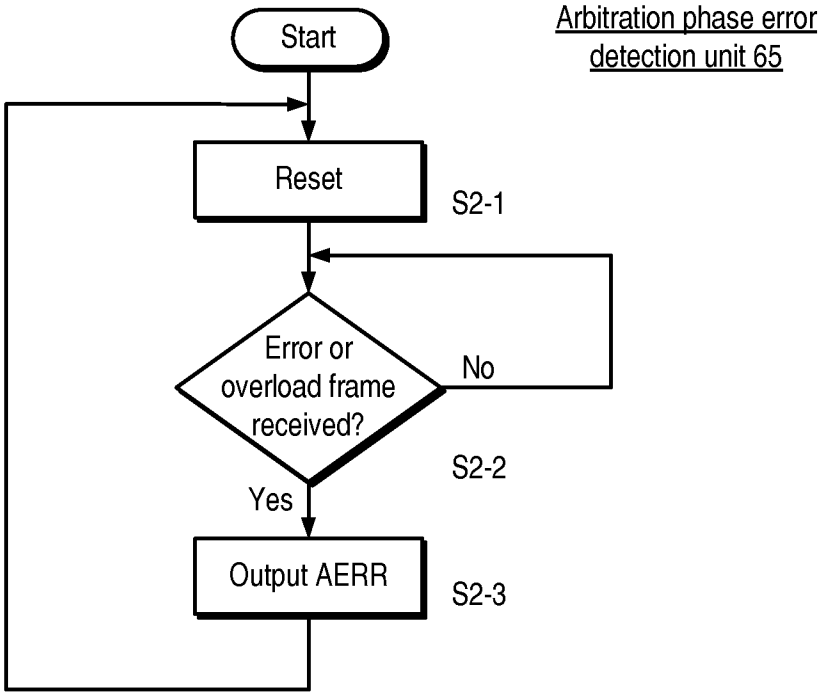

Arbitration phase error detection unit 65
Start
Reset
S2-1
Error or overload frame received?
No
S2-2
Yes
Output AERR
S2-3

Fig. 8

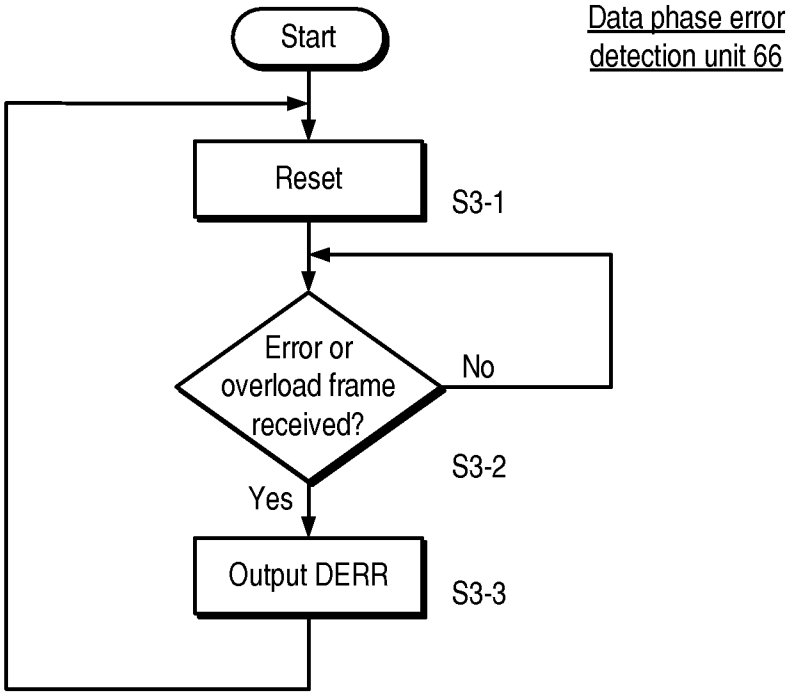

Data phase error detection unit 66
Start
Reset
S3-1
Error or overload frame received?
No
S3-2
Yes
Output DERR
S3-3

Fig. 9

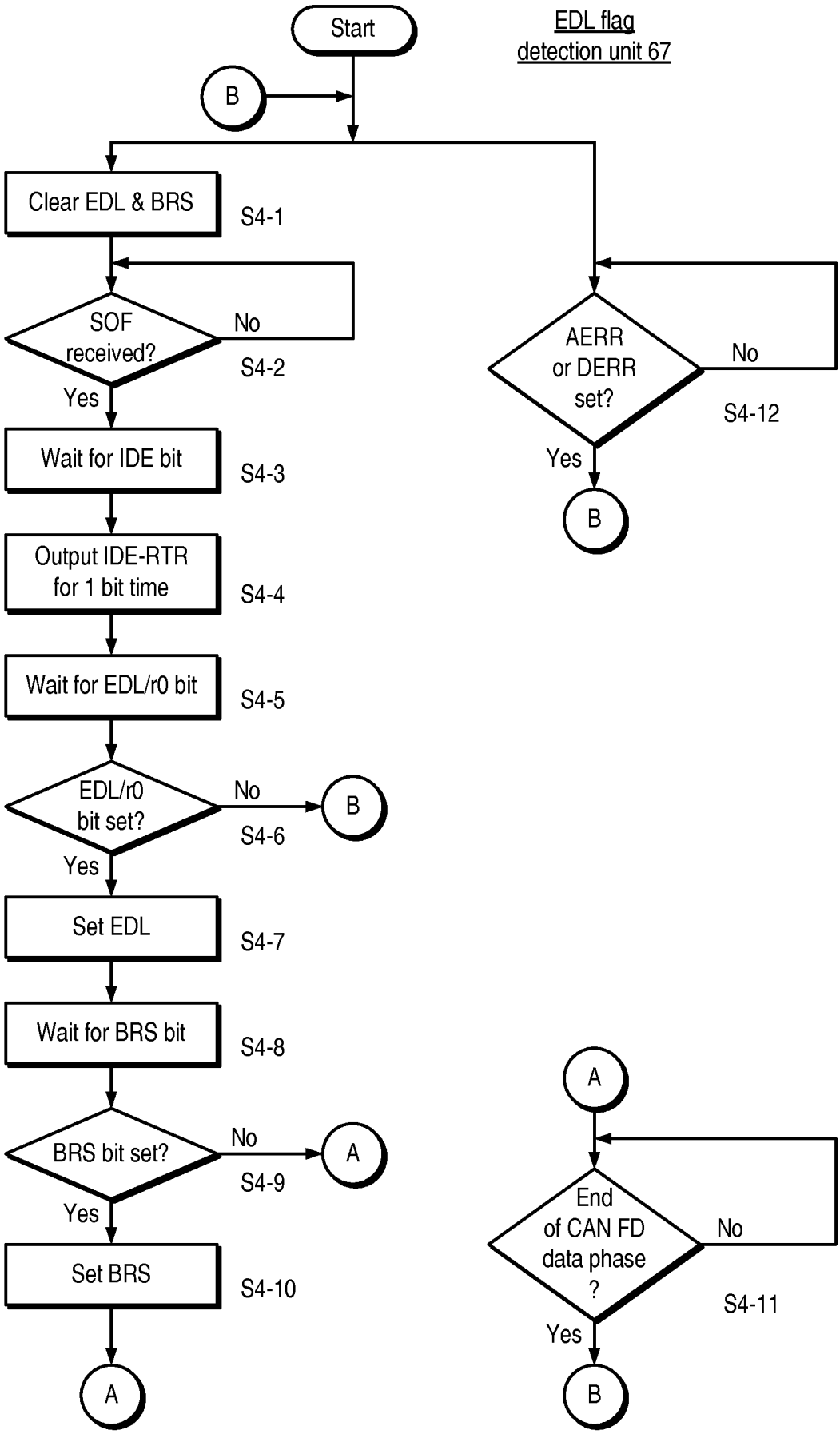

Start
B
Clear EDL & BRS
S4-1
SOF received?
No
S4-2
Yes
Wait for IDE bit
S4-3
Output IDE-RTR for 1 bit time
S4-4
Wait for EDL/r0 bit
S4-5
EDL/r0 bit set?
No
B
S4-6
Yes
Set EDL
S4-7
Wait for BRS bit
S4-8
BRS bit set?
No
A
S4-9
Yes
Set BRS
S4-10
A
EDL flag detection unit 67
AERR or DERR set?
No
S4-12
Yes
B
A
End of CAN FD data phase ?
No
S4-11
Yes
B

Fig. 10

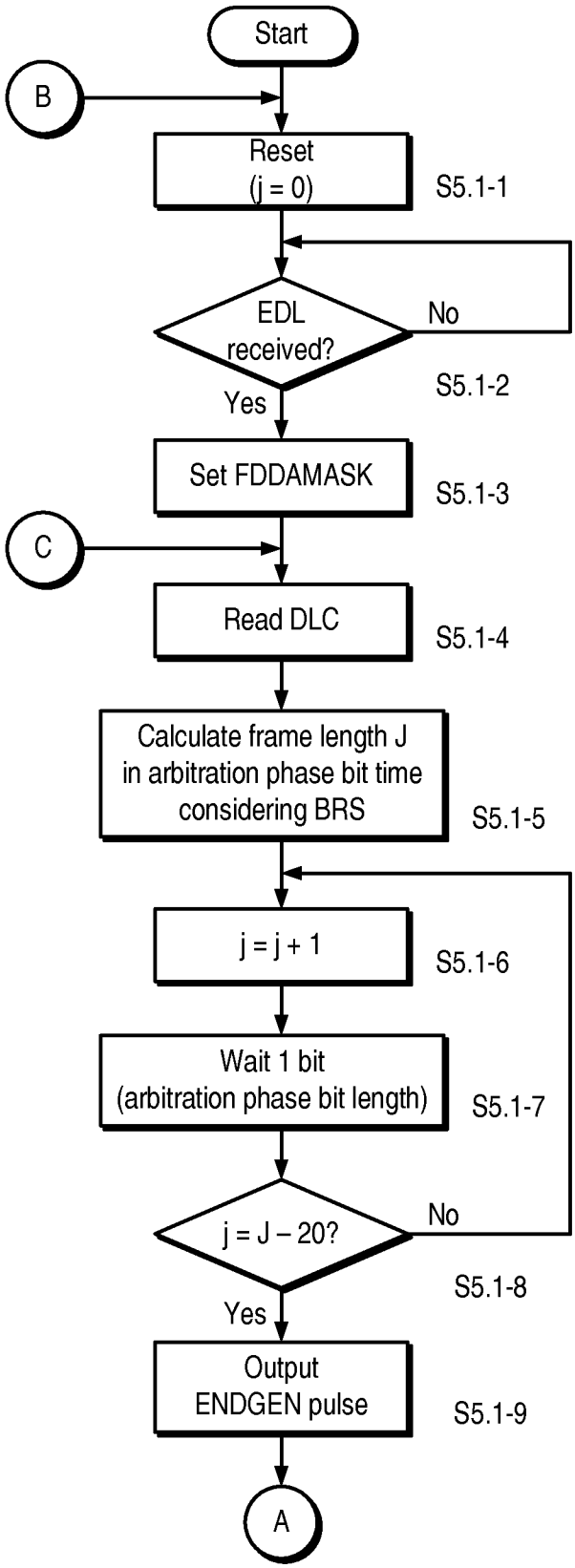

Start
B
Reset
(j = 0)
S5.1-1
EDL
received?
No
S5.1-2
Yes
Set FDDAMASK
S5.1-3
C
Read DLC
S5.1-4
Calculate frame length J
in arbitration phase bit time
considering BRS
S5.1-5
j = j + 1
S5.1-6
Wait 1 bit
(arbitration phase bit length)
S5.1-7
j = J – 20?
No
S5.1-8
Yes
Output
ENDGEN pulse
S5.1-9
A
FD data length
detection unit 68
(First mode of operation)

Fig. 11A

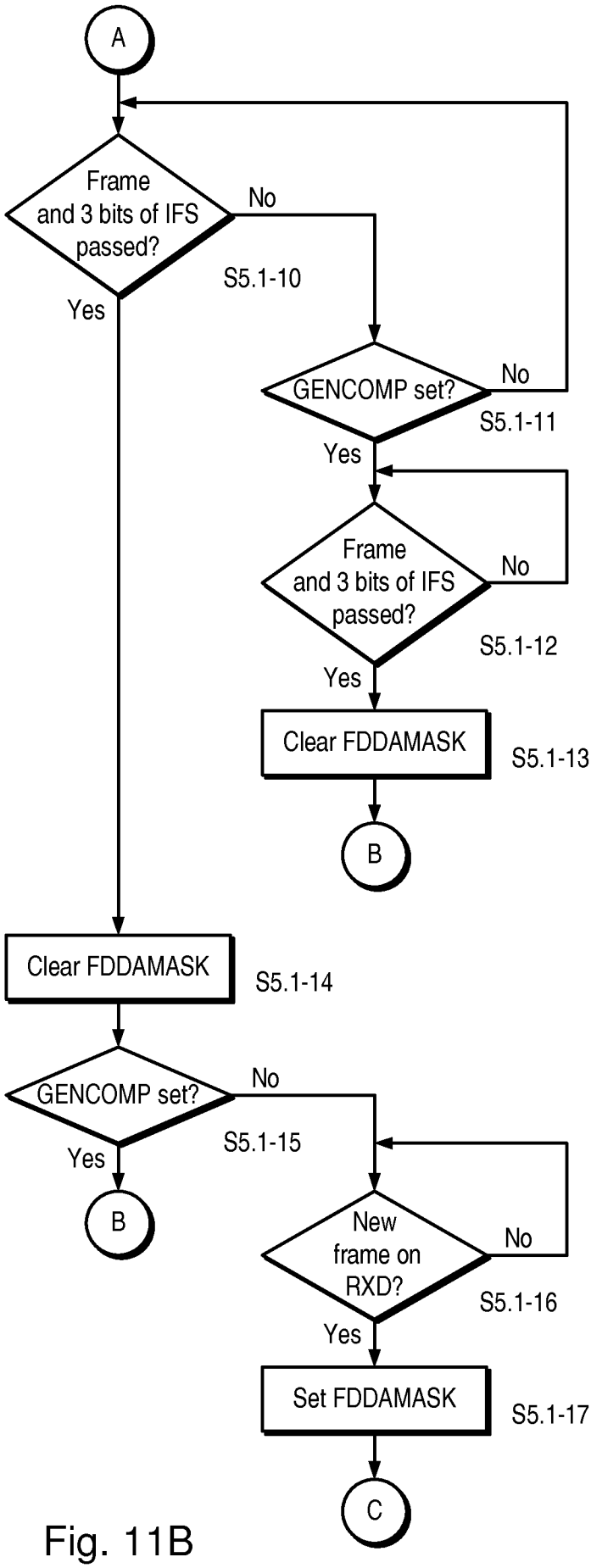

A
Frame and 3 bits of IFS passed?
No
S5.1-10
Yes
GENCOMP set?
No
S5.1-11
Yes
Frame and 3 bits of IFS passed?
No
S5.1-12
Yes
Clear FDDAMASK
S5.1-13
B
Clear FDDAMASK
S5.1-14
GENCOMP set?
No
S5.1-15
Yes
B
New frame on RXD?
No
S5.1-16
Yes
Set FDDAMASK
S5.1-17
C

Fig. 11B

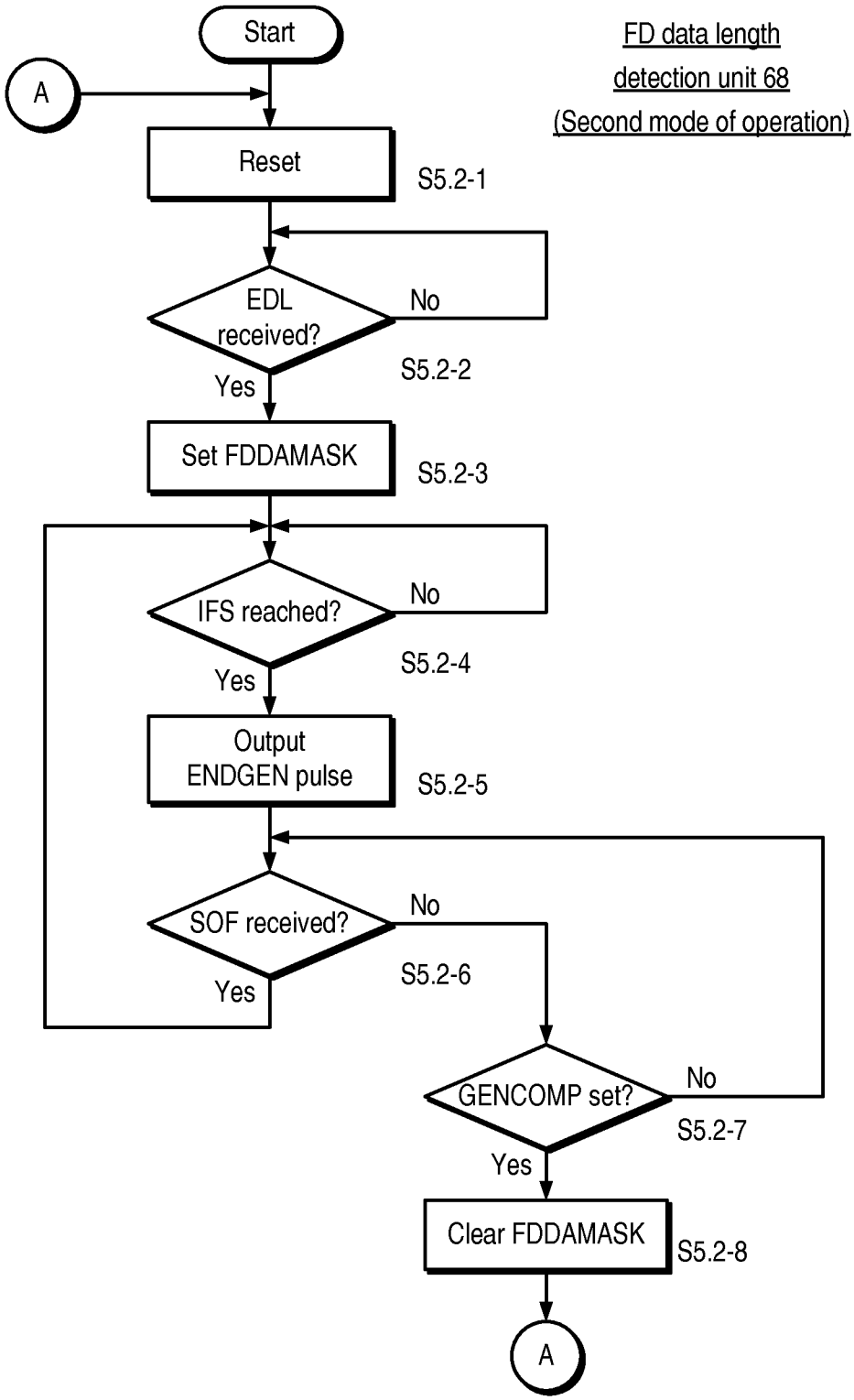

Start
A
FD data length detection unit 68 (Second mode of operation)
Reset
S5.2-1
EDL received?
No
S5.2-2
Yes
Set FDDAMASK
S5.2-3
IFS reached?
No
S5.2-4
Yes
Output ENDGEN pulse
S5.2-5
SOF received?
No
S5.2-6
Yes
GENCOMP set?
No
S5.2-7
Yes
Clear FDDAMASK
S5.2-8
A

Fig. 12

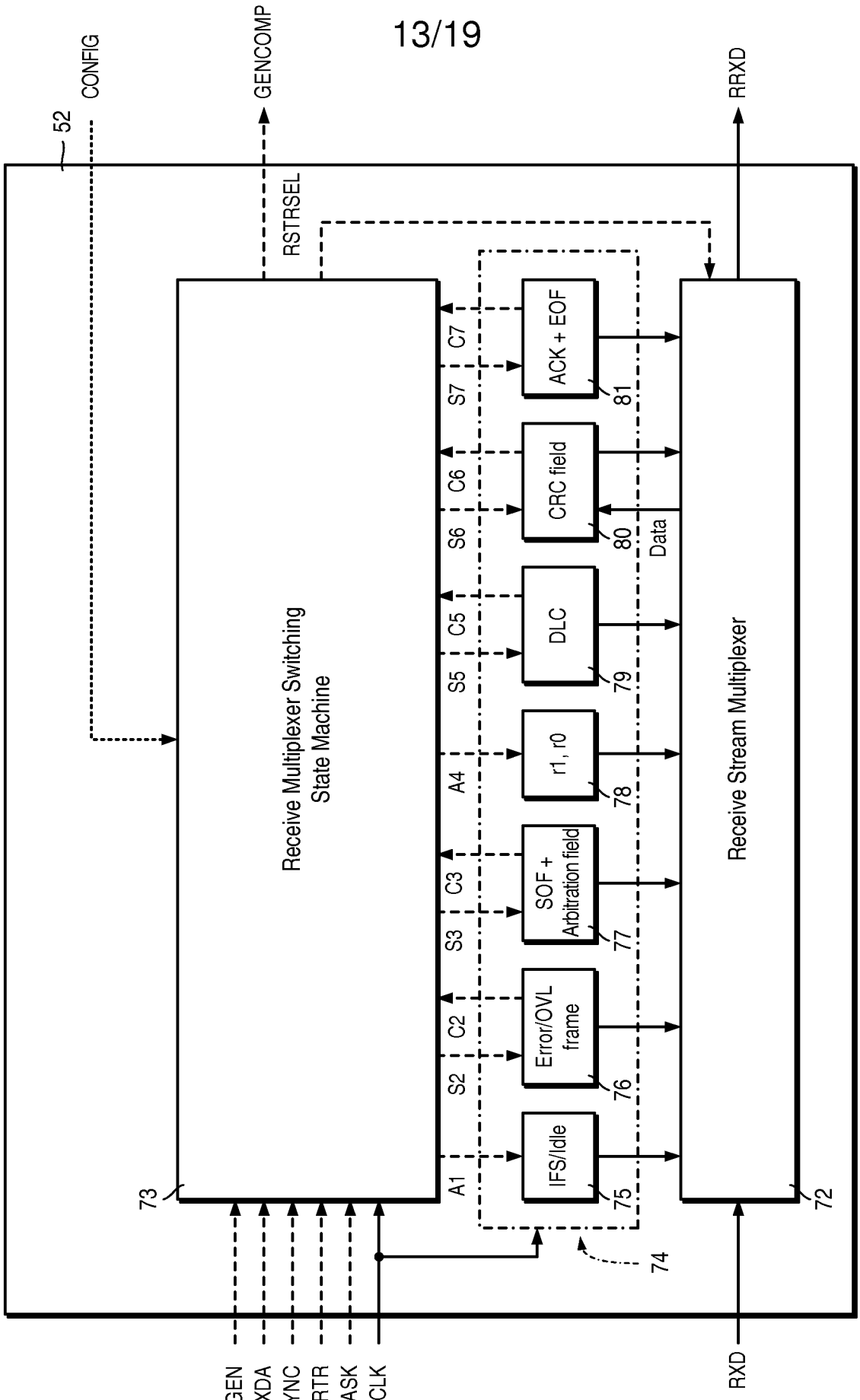

52
CONFIG
GENCOMP
RRXD
RSTRSEL
Receive Multiplexer Switching State Machine
73
A1
S2
C2
S3
C3
A4
S5
C5
S6
C6
S7
C7
IFS/Idle
Error/OVL frame
SOF + Arbitration field
r1, r0
DLC
CRC field
ACK + EOF
75
76
77
78
79
80
81
Data
74
Receive Stream Multiplexer
72
ENDGEN
CTXDA
SOFSYNC
IDE-RTR
FDDAMASK
TQCLK
RXD

Fig. 13

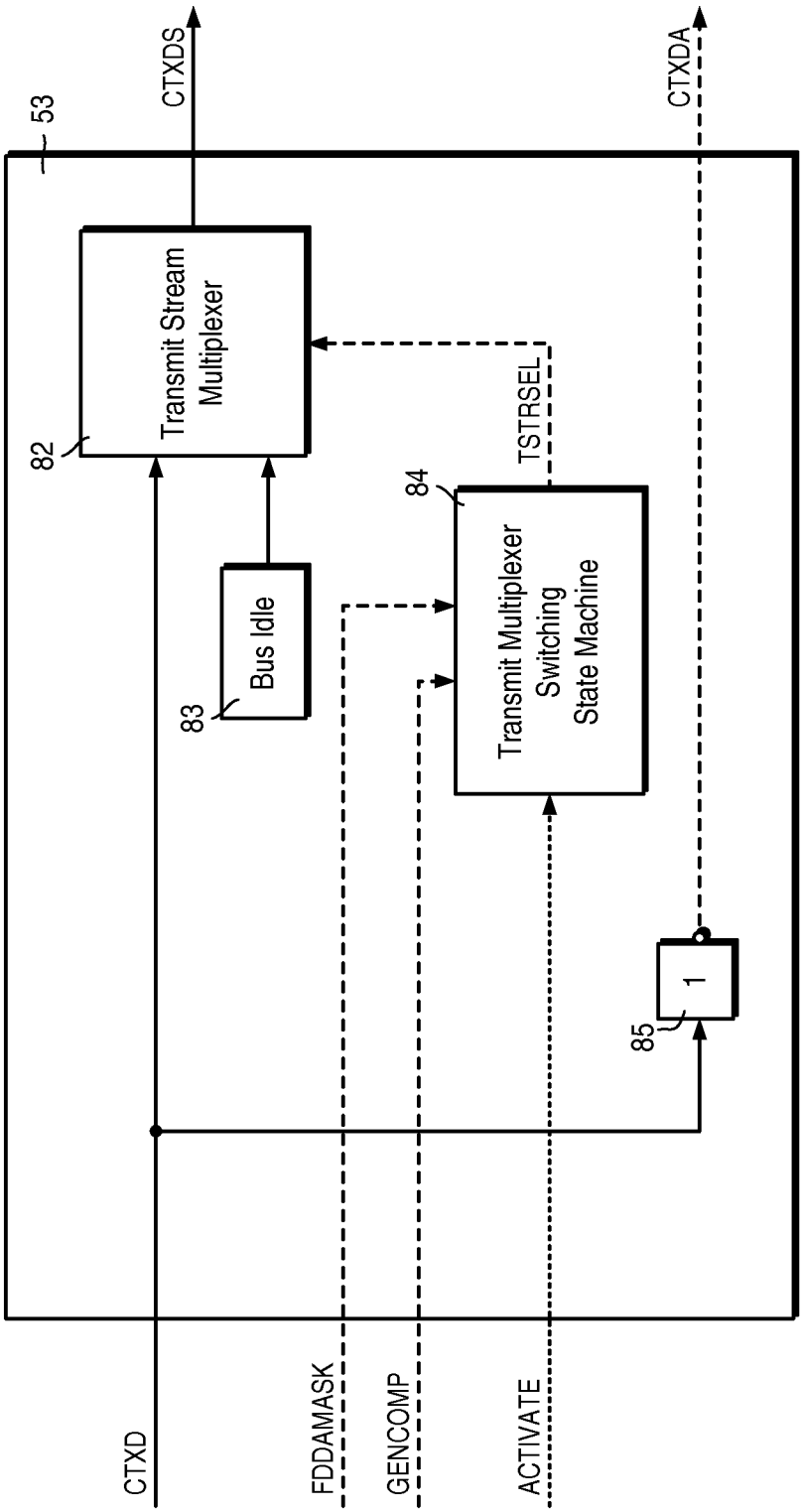

53
CTXDS
CTXDA
Transmit Stream Multiplexer
82
TSTRSEL
84
Transmit Multiplexer Switching State Machine
Bus Idle
83
1
85
CTXD
FDDAMASK
GENCOMP
ACTIVATE

Fig. 14

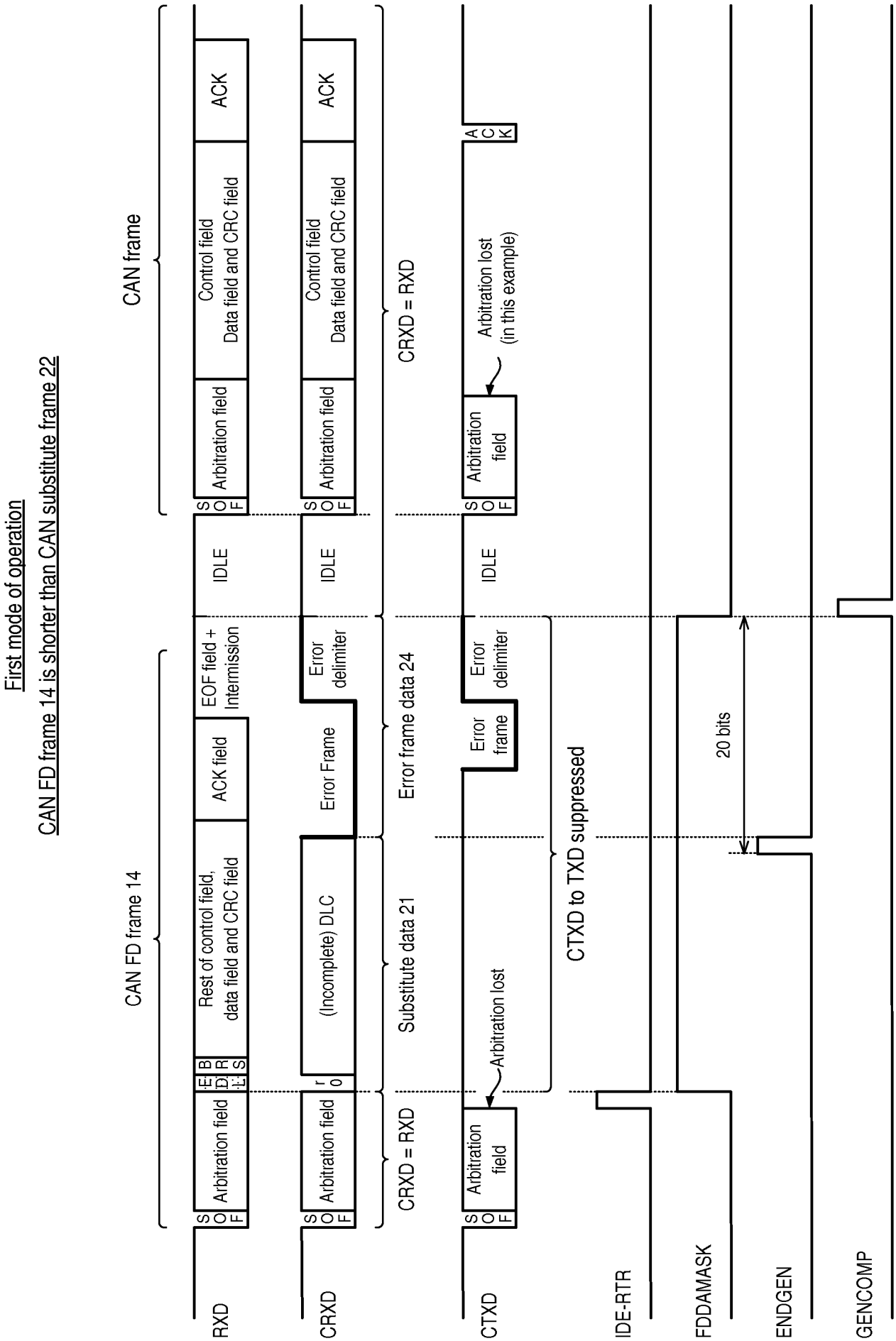

First mode of operation
CAN FD frame 14 is shorter than CAN substitute frame 22
CAN frame
CAN FD frame 14
RXD
CRXD
CTXD
IDE-RTR
FDDAMASK
ENDGEN
GENCOMP
SOF
Arbitration field
IDE
BRS
Rest of control field, data field and CRC field
ACK field
EOF field + Intermission
IDLE
r0
(Incomplete) DLC
Error Frame
Error delimiter
Substitute data 21
Error frame data 24
CRXD = RXD
CRXD = RXD
Arbitration lost
Error frame
Error delimiter
Arbitration lost (in this example)
CTXD to TXD suppressed
20 bits
ACK

Fig. 15

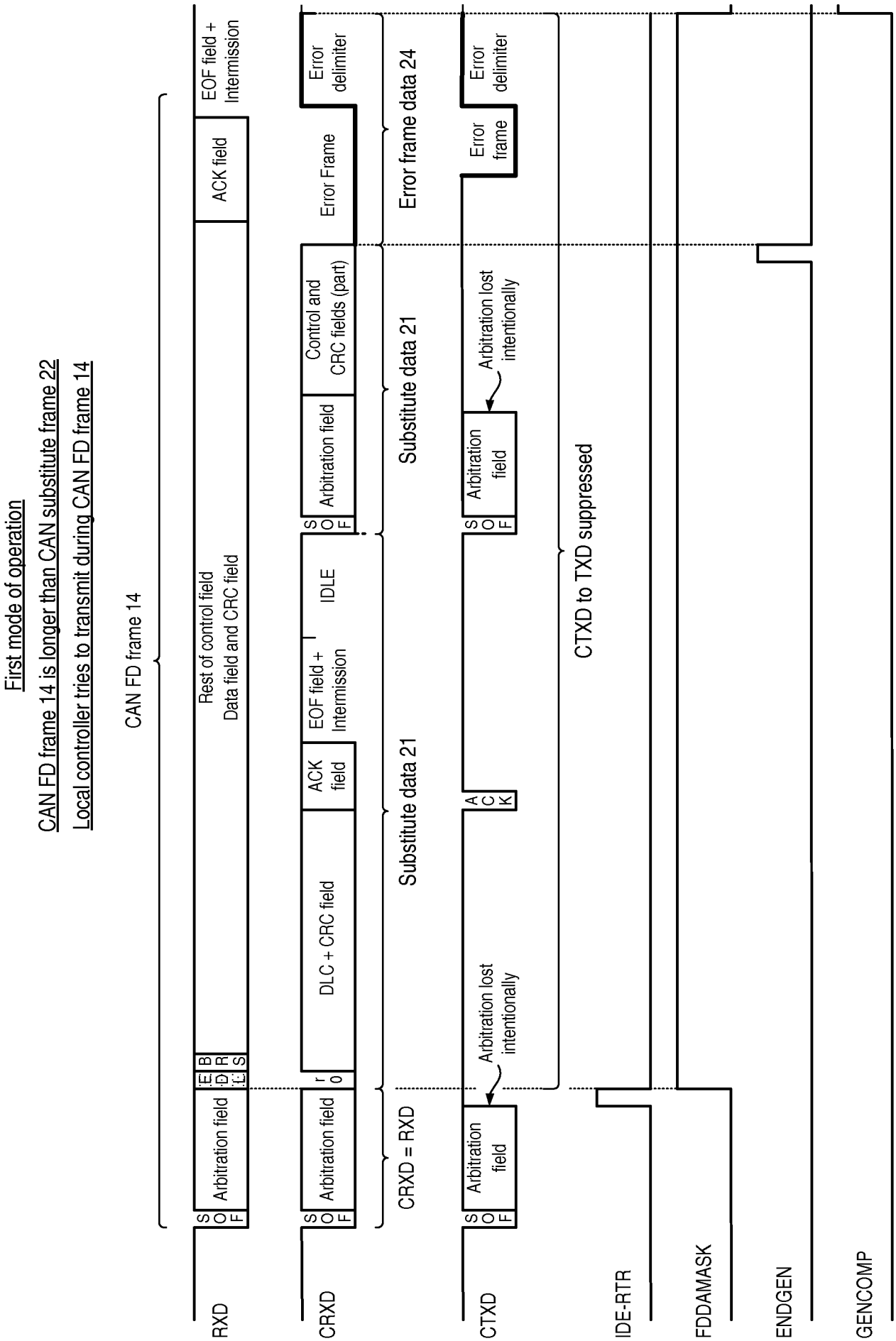

First mode of operation
CAN FD frame 14 is longer than CAN substitute frame 22
Local controller tries to transmit during CAN FD frame 14
CAN FD frame 14
RXD
CRXD
CTXD
IDE-RTR
FDDAMASK
ENDGEN
GENCOMP
SOF
Arbitration field
IDE
BRS
Rest of control field
Data field and CRC field
ACK field
EOF field + Intermission
r0
DLC + CRC field
ACK field
EOF field + Intermission
IDLE
Arbitration field
Control and CRC fields (part)
Error Frame
Error delimiter
Error frame data 24
Substitute data 21
CRXD = RXD
Arbitration lost intentionally
ACK
Arbitration field
Arbitration lost intentionally
Error frame
Error delimiter
Substitute data 21
CTXD to TXD suppressed

Fig. 16

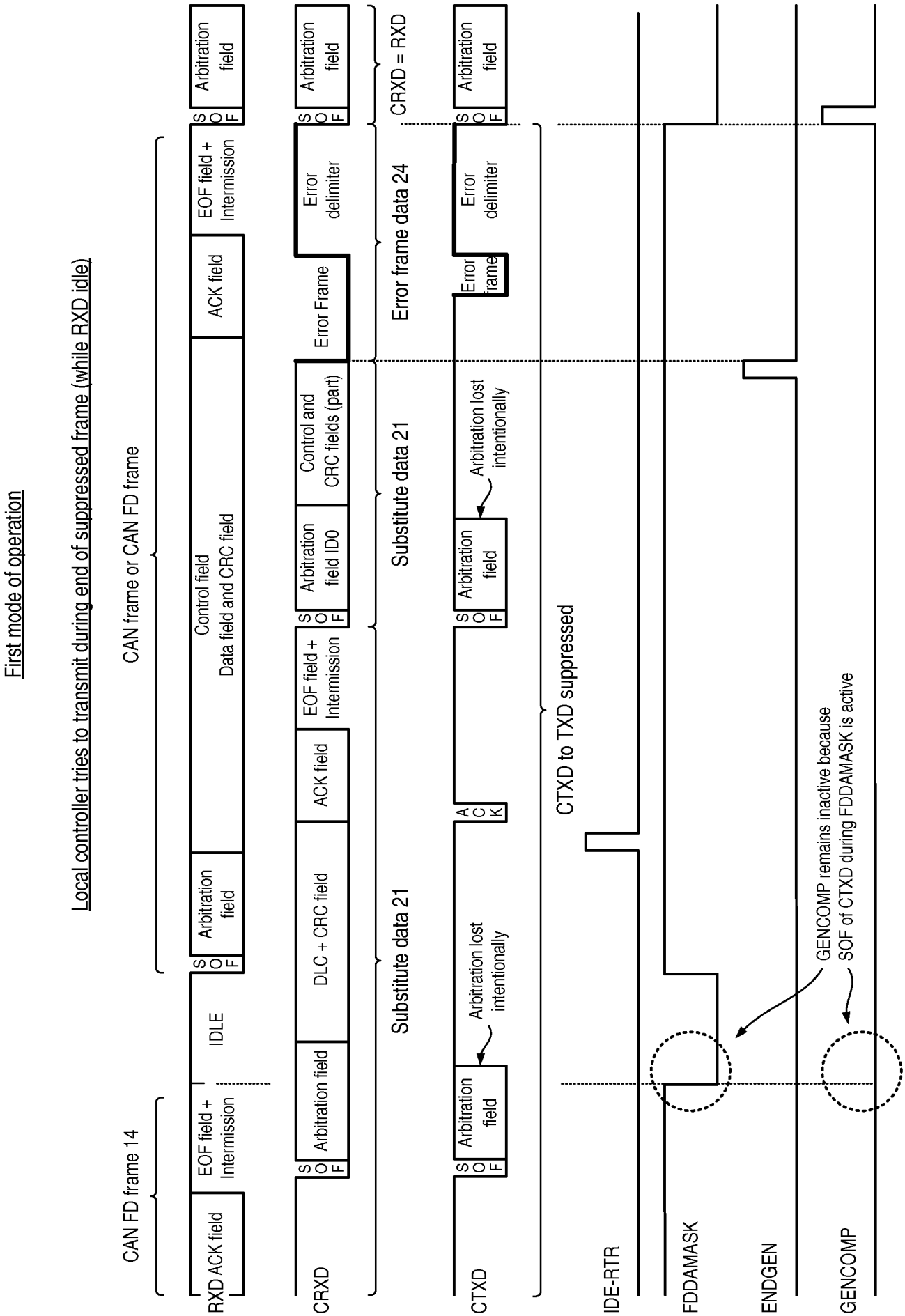

First mode of operation
Local controller tries to transmit during end of suppressed frame (while RXD idle)
CAN FD frame 14
CAN frame or CAN FD frame
RXD
ACK field
EOF field + Intermission
IDLE
SOF
Arbitration field
Control field
Data field and CRC field
ACK field
EOF field + Intermission
SOF
Arbitration field
CRXD
SOF
Arbitration field
DLC + CRC field
ACK field
EOF field + Intermission
SOF
Arbitration field ID0
Control and CRC fields (part)
Error Frame
Error delimiter
SOF
Arbitration field
Substitute data 21
Substitute data 21
Error frame data 24
CRXD = RXD
CTXD
SOF
Arbitration field
Arbitration lost intentionally
ACK
SOF
Arbitration field
Arbitration lost intentionally
Error frame
Error delimiter
SOF
Arbitration field
CTXD to TXD suppressed
IDE-RTR
FDDAMASK
ENDGEN
GENCOMP
GENCOMP remains inactive because
SOF of CTXD during FDDAMASK is active

Fig. 17

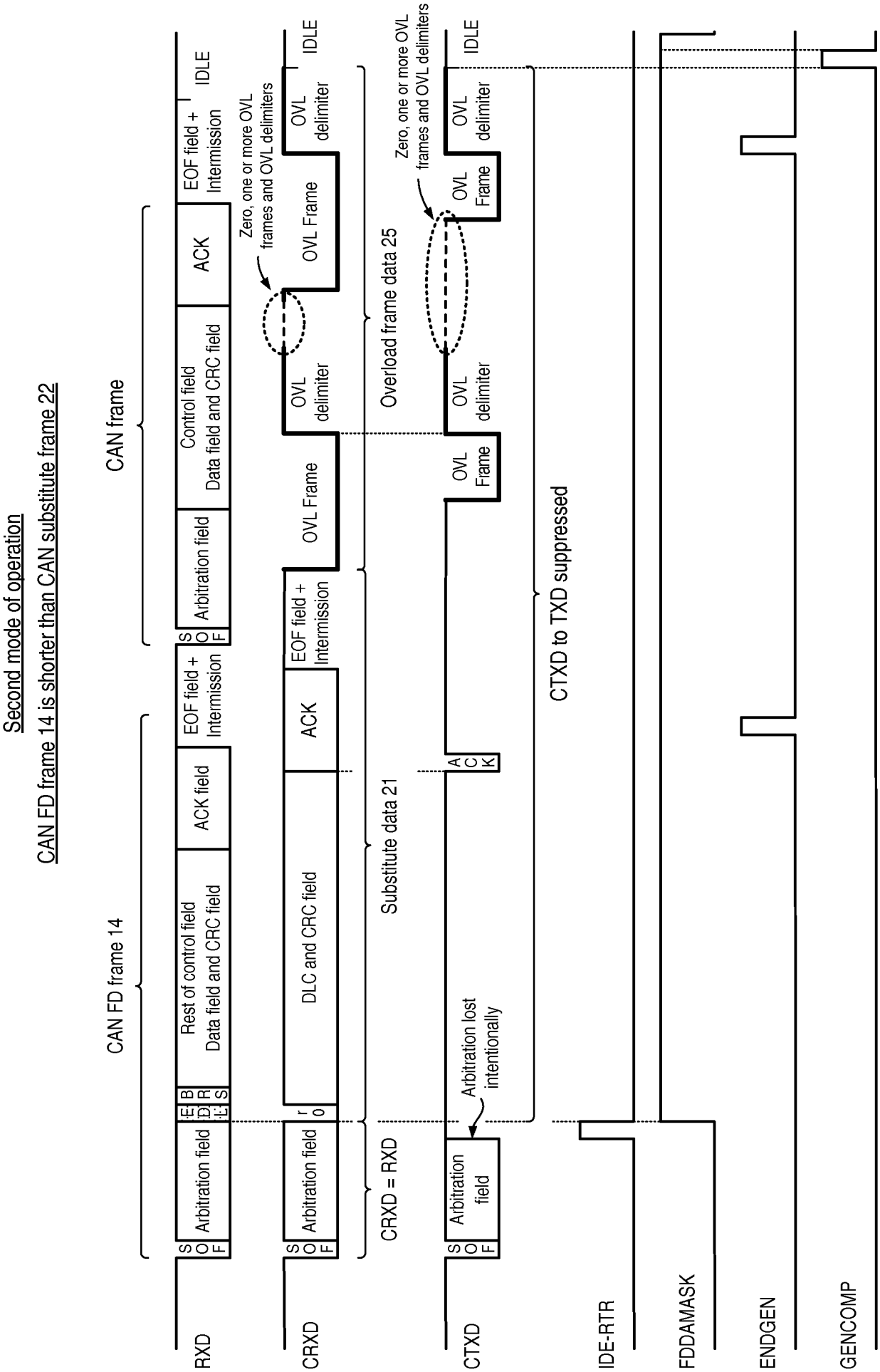

Second mode of operation
CAN FD frame 14 is shorter than CAN substitute frame 22
CAN FD frame 14
CAN frame
Substitute data 21
Overload frame data 25
CRXD = RXD
CTXD to TXD suppressed
RXD
CRXD
CTXD
IDE-RTR
FDDAMASK
ENDGEN
GENCOMP
SOF
Arbitration field
IDE
BRS
ESI
Rest of control field
Data field and CRC field
ACK field
EOF field + Intermission
IDLE
Control field
ACK
r0
DLC and CRC field
OVL Frame
OVL delimiter
Zero, one or more OVL frames and OVL delimiters
Arbitration lost intentionally
ACK

Fig. 18

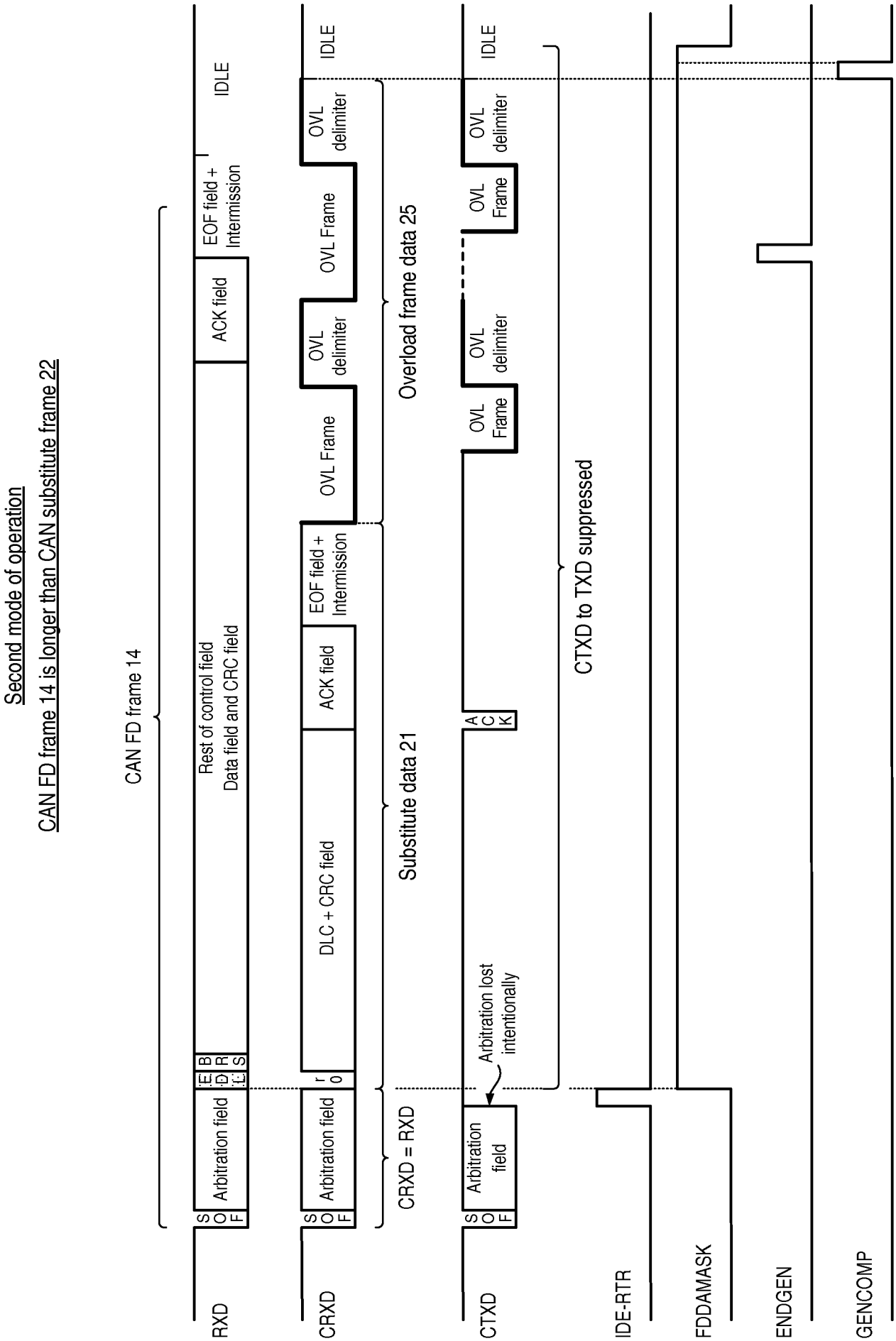

Second mode of operation
CAN FD frame 14 is longer than CAN substitute frame 22
CAN FD frame 14
Substitute data 21
Overload frame data 25
CRXD = RXD
CTXD to TXD suppressed
RXD
CRXD
CTXD
IDE-RTR
FDDAMASK
ENDGEN
GENCOMP
SOF
Arbitration field
IDE
BRS
Rest of control field
Data field and CRC field
ACK field
EOF field + Intermission
IDLE
r0
DLC + CRC field
OVL delimiter
OVL Frame
Arbitration lost intentionally
ACK

Fig. 19

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2014/056414

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L12/46 H04L29/06 H04L12/40
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2004/047385 A2 (BOSCH GMBH ROBERT [DE]; EIMERS-KLOSE DOERTE [DE]; HEINISCH CORNELIA [D) 3 June 2004 (2004-06-03) page 5, line 15 - page 9, line 25	1,13,15
A	DE 10 2011 080476 A1 (BOSCH GMBH ROBERT [DE]) 7 February 2013 (2013-02-07) the whole document	1-15
A	WO 2012/146631 A1 (BOSCH GMBH ROBERT [DE]; HARTWICH FLORIAN [DE]) 1 November 2012 (2012-11-01) the whole document	1-15

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
23 June 2014	27/06/2014

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Kalabic, Faris

2

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Y.B. Pradeep: "CAN - FD and Ethernet Create Fast Reliable Automotive Data Buses for the Next Decade", Atmel - Automotive Compilation , April 2013 (2013-04), pages 36-43, XP002713064, Retrieved from the Internet: URL:http://www.atmel.com/ja/jp/Images/Article_AC10_CAN-FD-and-Ethernet.pdf [retrieved on 2013-09-17] the whole document -----	1-15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2014/056414

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2004047385 A2	03-06-2004	DE 10254285 A1	03-06-2004
		EP 1566029 A2	24-08-2005
		JP 2006506862 A	23-02-2006
		US 2006130049 A1	15-06-2006
		WO 2004047385 A2	03-06-2004
DE 102011080476 A1	07-02-2013	CN 103842981 A	04-06-2014
		DE 102011080476 A1	07-02-2013
		EP 2740042 A1	11-06-2014
		KR 20140050643 A	29-04-2014
		WO 2013020781 A1	14-02-2013
WO 2012146631 A1	01-11-2012	CN 103649933 A	19-03-2014
		CN 103649934 A	19-03-2014
		EP 2702495 A1	05-03-2014
		EP 2702497 A1	05-03-2014
		KR 20140030213 A	11-03-2014
		KR 20140030214 A	11-03-2014
		US 2014156887 A1	05-06-2014
		WO 2012146631 A1	01-11-2012
		WO 2012146649 A1	01-11-2012