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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 September 2012 (13.09.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/121881 A1

(51) International Patent Classification:
H04L 25/02 (2006.01)

(21) International Application Number:
PCT/US2012/026010

(22) International Filing Date:
22 February 2012 (22.02.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/041,535 7 March 2011 (07.03.2011) US

(71) Applicant *(for all designated States except US)*: **MICROCHIP TECHNOLOGY INCORPORATED** [US/US]; 2355 West Chandler Blvd., Chandler, AZ 85224-6199 (US).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **MACH, Joel** [US/US]; 143 S. La Arboleta, Gilbert, AZ 85296 (US).

(74) Agent: **SLAYDEN, Bruce W., II**; King & Spalding LLP, 401 Congress Ave., Suite 3200, Austin, TX 78701 (US).

(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, 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, IS, JP, KE, KG, KM, 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, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, MD, 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,

[Continued on next page]

(54) Title: MICROCONTROLLER WITH CAN BUS MODULE AND AUTO SPEED DETECT

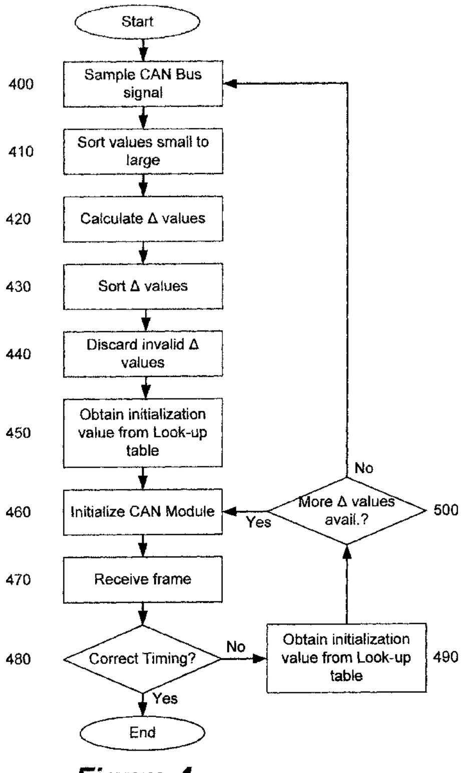

Figure 4

(57) Abstract: In a method for initializing a Controller Area Network (CAN) module in a microcontroller, the following steps may be provided: measuring the period times between a plurality of falling or rising edges of a CAN signal; sorting the period times; determining difference values between adjacent period times of the sorted period times; sorting the difference values; selecting a first difference value from the sorted difference values and determining a first frequency from the first difference value; initializing the CAN module using the selected frequency; receiving a CAN signal frame; determining whether an error occurred; if an error occurred selecting a next frequency and repeating initializing the CAN module until a valid CAN frequency has been found.

WO 2012/121881 A1

SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

MICROCONTROLLER WITH CAN BUS MODULE AND AUTO SPEED DETECT

TECHNICAL FIELD

The present invention relates to microcontrollers having a Controller Area Network (CAN) module.

BACKGROUND

Controller Area Network (CAN) is a serial communications protocol which efficiently supports distributed real-time control with a high level of security. Applications of CAN range from high speed digital communications networks to low cost multiplex wiring. CAN is a high-integrity serial data communications bus for real-time applications. CAN operates at data rates of up to 1 Megabits per second, has excellent error detection and confinement capabilities, and was originally developed for use in automobiles. The motivation behind CAN was to make automobiles more reliable, safe and fuel-efficient by improving the interoperability of automotive electronics, engine control units, sensors, anti-skid brake systems, etc., while decreasing wiring harness weight and complexity. Since CAN's inception, the CAN protocol has gained widespread popularity in industrial automation and automotive/truck applications. The robustness of the CAN bus in noisy environments and the ability to detect and recover from fault conditions makes CAN suitable for use with, industrial control equipment, medical equipment, test equipment, mobile and portable machines, appliances, etc.

CAN is an asynchronous serial bus system with one logical bus comprising for example two wires. It has an open, linear bus structure with equal bus nodes. A CAN bus consists of two or more nodes. The number of nodes on the bus may be changed dynamically without disturbing the communication of the other nodes.

The CAN logic bus corresponds to a "wired-OR" mechanism, "recessive" bits (mostly, but not necessarily equivalent to the logic level "1") are overwritten by "dominant" bits (mostly logic level "0"). As long as no bus node is sending a dominant bit, the bus line is in the recessive state, but a dominant bit from any bus node generates the dominant bus state. Therefore, for the CAN bus line, a medium is chosen that is able to transmit the two possible bit states (dominant and recessive). A common physical medium used is a twisted wire pair.

The bus wires are then called "CANH" and "CANL," and may be connected directly to the CAN controller nodes or via a connector thereto.

In the CAN bus protocol it is not bus nodes that are addressed, but rather the address information is contained in the messages that are transmitted. This is done via an identifier (part of each message) which identifies the message content, e.g., engine speed, oil temperature, etc. The identifier additionally indicates the priority of the message. The lower the binary value of the identifier, the higher the priority of the message (more dominant bits).

In a CAN controller the architecture is similar to a simple UART, except that complete frames are sent instead of characters: there is (typically) a single transmit buffer, and a double-buffered receive buffer. The CPU puts a frame in the transmit buffer, and takes an interrupt when the frame is sent; the CPU receives a frame in the receive buffer, takes an interrupt and empties the buffer (before a subsequent frame is received). The CPU must manage the transmission and reception, and handle the storage of the frames."

During initialization the transmitting device and the receiving device generally program their respective CAN modules to use a predetermined CAN frequency. However, in many applications this frequency may not be known. Also, the length of each CAN bit can vary by approximately 20% around some nominal length. Moreover, in a running system, there are no "known initialization transmissions" to which a system can be synchronized. In other words, if a device is coupled to an already established CAN bus system, the content of the transferred messages is unknown. Thus, in conventional systems, a trial and error method may be used to determine the correct operating frequency of the CAN Bus which can result in a very long set-up time or may even fail entirely. In other serial bus systems, an initialization phase may be performed using synchronizing character sequences such as "5555" or "AAAA". However, such initialization routines may not be available in particular if a CAN bus is already be in operation.

Hence, there is a need for an improved auto-speed detect system and method for a microcontroller with a CAN module which allows for a quick determination of an operating frequency used in a CAN system.

Summary of the Invention

According to an embodiment, a method for initializing a Controller Area Network (CAN) module in a microcontroller, may comprise: measuring the period times between a plurality of falling or rising edges of a CAN signal; sorting the period times; determining difference values between adjacent period times of the sorted period times; sorting the difference values; selecting a first difference value from the sorted difference values and determining a first frequency from the first difference value; initializing the CAN module using the selected frequency; receiving a CAN signal frame; determining whether an error occurred; if an error occurred selecting a next frequency and repeating initializing the CAN module until a valid CAN frequency has been found.

According to a further embodiment, the method may also comprise: coupling an external CAN line with an input of a capture module of the microcontroller to measure the period times. According to a further embodiment, the method may further comprise: feeding a CAN signal internally to an input of a capture module of the microcontroller to measure the period times. According to a further embodiment, the method may further comprise: after the step of sorting the difference values, the step of: discarding difference values that result in invalid frequencies. According to a further embodiment, a timer and capture unit can be used to measure the period times. According to a further embodiment, ten period times between falling edges of a CAN signal may be measured. According to a further embodiment, the method may be repeated if no valid frequency is found. According to a further embodiment, the step of determining a frequency from the difference value may be performed using a look-up table. According to a further embodiment, the step of determining a frequency from the difference value may use rounding or truncation to calculate a frequency. According to a further embodiment, the microcontroller may be programmed to perform the method. According to a further embodiment, the microcontroller may comprise a state machine that performs the method.

According to another embodiment, a microcontroller may comprise: a central processing unit (CPU), a Controller Area Network (CAN) module, a timer, a capture module coupled with the timer, wherein the microcontroller is configured to automatically detect a CAN frequency by: measuring the period times between a plurality of falling or rising edges of a CAN signal; sorting the period times; determining differences between adjacent period times of the sorted period times; selecting a first difference and determining a first frequency from the first difference; initializing the CAN module using the selected frequency; receiving a CAN signal frame; determining whether an error occurred; if an error occurred selecting a next frequency and repeating initializing the CAN module until a valid CAN frequency has been found.

According to a further embodiment of the microcontroller, the microcontroller may be further configured to discard difference values that result in invalid frequencies. According to a further embodiment of the microcontroller, the capture module may comprise an edge detector controlling a driver which couples the timer with a capture register. According to a further embodiment of the microcontroller, the microcontroller may comprise a controllable driver coupled between an input of the CAN Bus module and the input of the capture module. According to a further embodiment of the microcontroller, the microcontroller may be configured to determine a frequency from the difference value by using a look-up table. According to a further embodiment of the microcontroller, the microcontroller may be configured to determine a frequency from the difference value using rounding or truncation. According to a further embodiment of the microcontroller, the microcontroller may be programmed by software to determine the CAN frequency. According to a further embodiment of the microcontroller, the microcontroller may comprise a state machine to determine the CAN frequency. According to a further embodiment of the microcontroller, the microcontroller may comprise a special function register to control the state machine.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of an embodiment of a microcontroller with integrated CAN module.

Figure 2 shows a timing diagram of an exemplary CAN signal.

Figure 3 shows a principle according to an embodiment to determine a CAN operating frequency .

Figure 4 shows a flow chart according to an embodiment.

Detailed Description of the Preferred Embodiment

Referring to Figure 1, a microcontroller 100 is indicated by the dotted line. The microcontroller 100 comprises a central processing unit (CPU) 110 and a CAN Bus module or peripheral device which is coupled with the CPU 110. External pins 130 and 135 are provided through which the CAN Bus module 120 can connect its separate transmit and receive lines CANTX, CANRX to the twisted pair CANH and CANL lines of the external CAN Bus 180. This is usually done via an external interface driver 105 that converts the separate internal signals CANRX and CANTX to respective CAN Bus signals CANH and CANL. However, other embodiments may provide internally for such an interface device to allow direct connection to a CAN bus.

Furthermore, the microcontroller comprises a capture/compare/pulse width modulation (CCP) module 140, a timer module 170 and a clock system 160, all coupled with CPU 110. The CCP unit 140 is coupled with another external pin 150 for evaluating external signals. To this end, according to an embodiment, a controllable driver 146 inside the CCP module 140 may be connected to external pin 150. However, the exemplary device shown in Fig. 1 may also be capable of evaluating/processing internal signals as shown by the dotted lines. For example, in one embodiment, the internal CAN signal CANRX can be fed through a controllable driver 185 to the CCP module 140. This embodiment may be considered in particular when an internal CAN Bus interface 105 is present to allow for an evaluation of the internal CANRX signals. However, as will be appreciated by a person skilled in the art, other embodiments may allow to evaluate the CAN Bus lines directly. In such a case a differential driver/converter unit which is arranged internally or externally may be used to convert the signals on the CAN Bus into a single digital signal and feed them to the CCP module 140.

CCP module 140 may comprise a prescaler 142 which receives the output of drivers 146 and 185 and any other controllable driver that may be implemented to select an internal or external input signal. The output of prescaler 142 is coupled with an edge detector 148 which can programmably either detect a rising or falling edge of the incoming signal. The output signal of this edge detector controls a further driver 145 which couples a timer register 175 of timer module 170 with capture register 144 of CCP module 140. Edge detector 160 may receive a clock signal from, clock system 160.

To provide for a auto-speed detect function, the input of the CCP module, or one of its inputs in case the CCP module comprises multiple inputs, is coupled with the CANRX line as available on pin 130. As shown in Fig. 1, this can be done either by connecting external pin 150 with the CANRX pin 130 or selecting an internal coupling via controllable driver 185. This setup is necessary only during the auto-speed detect phase. Once the speed has been determined, the CCP input can be used for processing other signals.

Figure 2 shows an exemplary signal on external CAN Bus 180. The CCP module 140 is programmed to analyze the falling edges of the CAN signal 200. However, in other embodiments, different signal characteristics may be analyzed to determine a period. According to Fig. 2 every time a falling edge occurs, the CCP module 140 captures the timer value from timer module. Exemplary timer values that have been captured are shown. Values are rounded to the nearest 100 merely to facilitate understanding the principle of the method. In the example shown in Fig. 2, ten falling edges are investigated. However, more or less edges can be processed. The more edges are captured the more precise the auto-speed detection will become. The timer may be initialized with a resolution of 100 ns. To this end, the timer may be clocked at 10MHz or any other suitable clock frequency. The timer values can be for example 16-bit values as shown in Fig. 1. Thus, depending on the clock source, different resolutions can be obtained. The better the timer resolution, the more exact the measurements. Using the fastest possible timer source is the preferred method as it increases the resolution of the timer. For example, a 40MHz clock can be used to improve the timer resolution to 25ns. These timer values can then be stored in an array.

Fig. 3 shows how the captured timer values are further processed. In a first step, the array with the ten captured timer values is sorted from smallest to highest values as shown in the most left column. Then, the difference between each two adjacent timer values is calculated as shown in the second column. Next, the different difference values are sorted again as shown in the third column. In another step, invalid differences may be discarded. For example, values that are clearly below 1µs or any predetermined maximum period or over a predetermined maximum frequency may be discarded as shown in the third column in Fig. 3. For the remaining values as shown in the fourth column in Fig. 3, the resulting frequencies for the remaining period values are determined. The first value is then used for setting the frequency of the CAN module. To this end, according to an embodiment, a window may be used to round the frequency to the nearest number that makes sense in a

CAN system. For example, 250kbps or 275kbps but not 263kbps. Once this frequency has been determined, for example 500kbps as shown in Fig. 3, the CAN module 120 is initialized using this frequency and set into a "Listen Only" mode so that normal bus traffic is not disrupted. Then, the system waits and determines whether a received message is valid or an error message is produced by the CAN module. In case of an error message, the auto-speed detect method selects the next available frequency as shown in Fig. 3, for example, 450kbps. These steps are repeated until the correct frequency is set. Thus, the method repeats initializing the CAN module 120 using the different frequencies as determined in the last column shown in Fig. 3. If none of these determined values is produces a correct setting, the method starts again and evaluates another ten edges of a CAN signal.

The timer difference values as shown in the fourth column in Fig. 3 may be directly used to access a table that contains the actual frequency values. Such a method would allow to easily perform a truncation or rounding to obtain useful frequency values.

Fig. 4 shows an exemplary flow chart of the above described method. In Step 400, the CAN signal is evaluated using the CCP module by detecting falling or rising edges and storing the associated timer values t(1) to t(n). These values are then sorted into values ts(1) - ts(n) in step 410. In step 420 the Δ values d(1) - d(n-1) are determined by calculating d(x)=ts(x+1)-ts(x), for x=1..n-1. The Δ values are sorted again into ds(1)..ds(n-1) in step 430. Then, optionally in step 440, values ds() that do not form valid frequencies are discarded. The remaining values are used to determine frequency values f(x) = 1/ds(x) in step 450. In step 460 the first value f(1) is used to initialize the CAN module 120. Then, in step 470 a frame is received. Correct timing is determined in step 480 by checking whether the CAN module generated any error messages. If not, then the routine ends. In case an error was generated then the routine proceeds by obtaining the next value f(i), for example, by using a look-up table in step 490. In step 500 it is determined whether there are no more values available. IF so, the routine jumps to step 500, other wise, the routine jumps to step 460.

According to an embodiment, the above described sequence can be performed by software. However, according to another embodiment, a state machine 190 may be used to control the specific sequences as described above. Fig. 1 shows the state machine 190. This state machine would in particular be advantageous in a microcontroller 100 that is capable of selecting the input signal to the CAN Bus module 120 internally, for example through controllable driver 185 as shown in Fig. 1. The state machine could be started through setting a bit in a special function register 195. Once started, State machine 190 couples the internal CAN bus signal via driver 185 to the CCP module 140, performs the initialization of CAN module 120 as described above and disconnects the CAN signal from the CCP module 140. A user may then use this module for the evaluation of other signals.

The invention, therefore, is well adapted to carry out the objects and attain the ends and advantages mentioned, as well as others inherent therein. While the invention has been depicted, described, and is defined by reference to particular preferred embodiments of the invention, such references do not imply a limitation on the invention, and no such limitation is to be inferred. The invention is capable of considerable modification, alteration, and equivalents in form and function, as will occur to those ordinarily skilled in the pertinent arts. The depicted and described preferred embodiments of the invention are exemplary only, and are not exhaustive of the scope of the invention. Consequently, the invention is intended to be limited only by the spirit and scope of the appended claims, giving full cognizance to equivalents in all respects.

CLAIMS

What is claimed is:

1. A method for initializing a Controller Area Network (CAN) module in a microcontroller,

the method comprising:

- measuring the period times between a plurality of falling or rising edges of a CAN signal;

- sorting said period times;

- determining difference values between adjacent period times of said sorted period times;

- sorting said difference values;

- selecting a first difference value from said sorted difference values and determining a first frequency from said first difference value;

- initializing said CAN module using said selected frequency;

- receiving a CAN signal frame;

- determining whether an error occurred;

- if an error occurred selecting a next frequency and repeating initializing the CAN module until a valid CAN frequency has been found.

2. The method according to claim 1, further comprising coupling an external CAN line with an input of a capture module of said microcontroller to measure said period times.

3. The method according to claim 1, further comprising feeding a CAN signal internally to an input of a capture module of said microcontroller to measure said period times.

4. The method according to claim 1, further comprising after the step of sorting said difference values, the step of:

discarding difference values that result in invalid frequencies.

5. The method according to claim 1, wherein a timer and capture unit is used to measure the period times.

6. The method according to claim 3, wherein ten period times between falling edges of a CAN signal are measured.

7. The method according to claim 1, wherein the method is repeated if no valid frequency is found.

8. The method according to claim 1, wherein the step of determining a frequency from said difference value is performed using a look-up table.

9. The method according to claim 1, wherein the step of determining a frequency from said difference value uses rounding or truncation to calculate a frequency.

10. The method according to claim 1, wherein the microcontroller is programmed to perform the method.

11. The method according to claim 1, wherein the microcontroller comprises a state machine that performs the method.

12. A microcontroller comprising:

a central processing unit (CPU)

a Controller Area Network (CAN) module,

a timer;

a capture module coupled with said timer;

wherein the microcontroller is configured to automatically detect a CAN frequency by:

- measuring the period times between a plurality of falling or rising edges of a CAN signal;

- sorting said period times;

- determining differences between adjacent period times of said sorted period times;

- selecting a first difference and determining a first frequency from said first difference;

- initializing said CAN module using said selected frequency;

- receiving a CAN signal frame;

- determining whether an error occurred;

- if an error occurred selecting a next frequency and repeating initializing the CAN module until a valid CAN frequency has been found.

13. The microcontroller according to claim 12, wherein the microcontroller is further configured to discard difference values that result in invalid frequencies.

14. The microcontroller according to claim 12, wherein the capture module comprises an edge detector controlling a driver which couples said timer with a capture register.

15. The microcontroller according to claim 12, comprising a controllable driver coupled between an input of the CAN Bus module and the input of said capture module.

16. The microcontroller according to claim 12, wherein the microcontroller is configured to determine a frequency from said difference value by using a look-up table.

17. The microcontroller according to claim 12, wherein the microcontroller is configured to determine a frequency from said difference value using rounding or truncation.

18. The microcontroller according to claim 12, wherein the microcontroller is programmed by software to determine said CAN frequency.

19. The microcontroller according to claim 12, comprising a state machine to determine said CAN frequency.

20. The microcontroller according to claim 19, comprising a special function register to control the state machine.

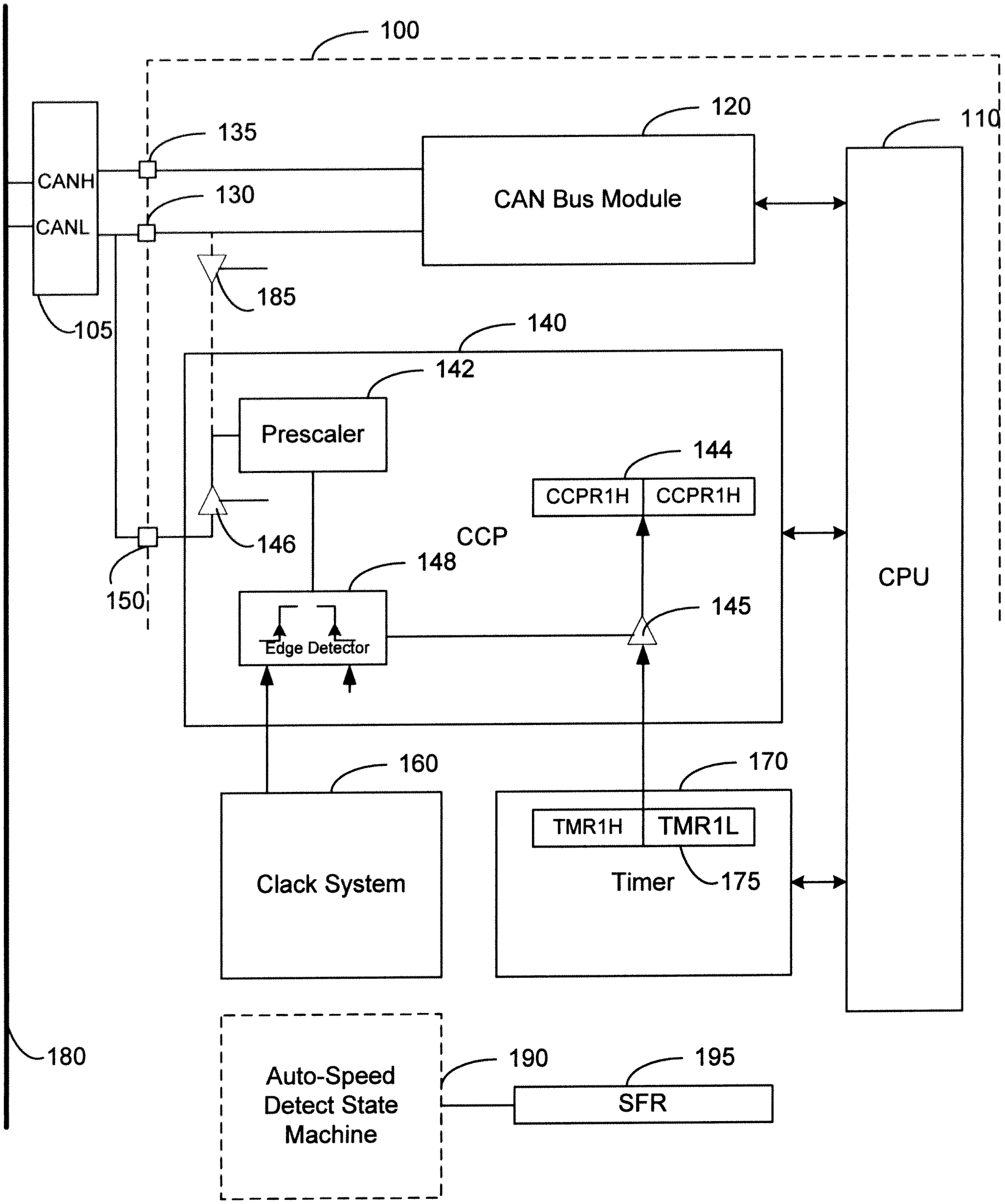
100
120
110
135
CANH
130
CANL
CAN Bus Module
185
105
140
142
Prescaler
144
CCPR1H
CCPR1H
CCP
146
148
150
145
Edge Detector
CPU
160
170
TMR1H
TMR1L
Clack System
Timer
175
180
190
195
Auto-Speed Detect State Machine
SFR

Figure 1

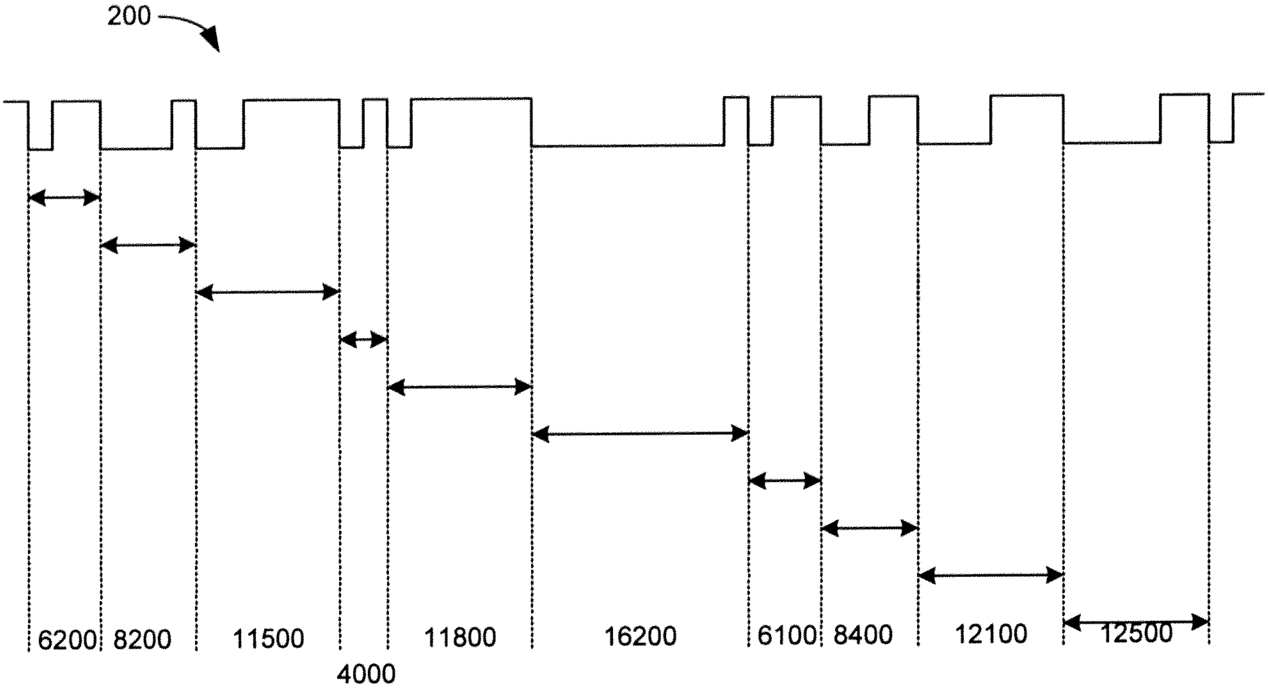

Figure 2

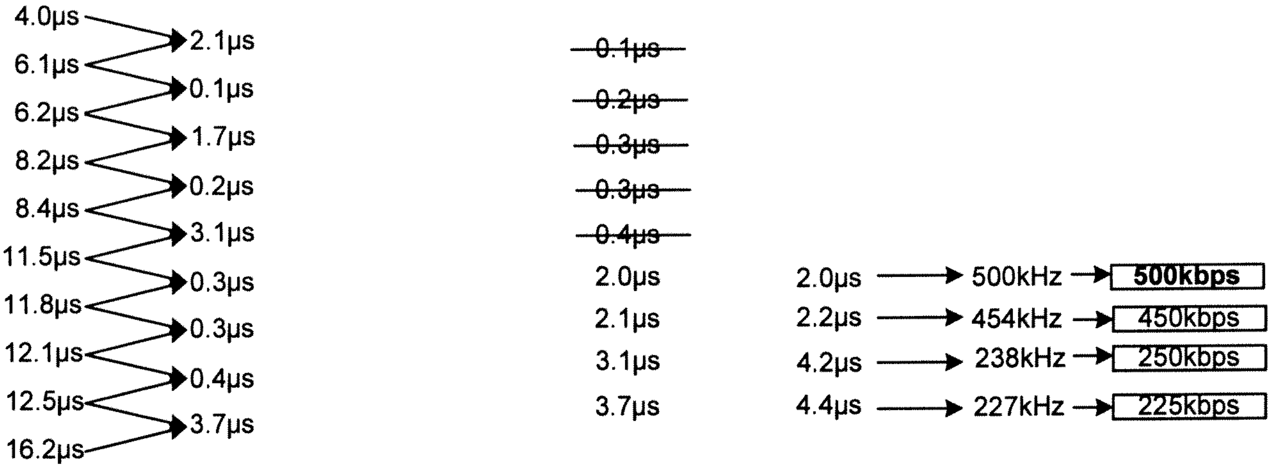

Figure 3

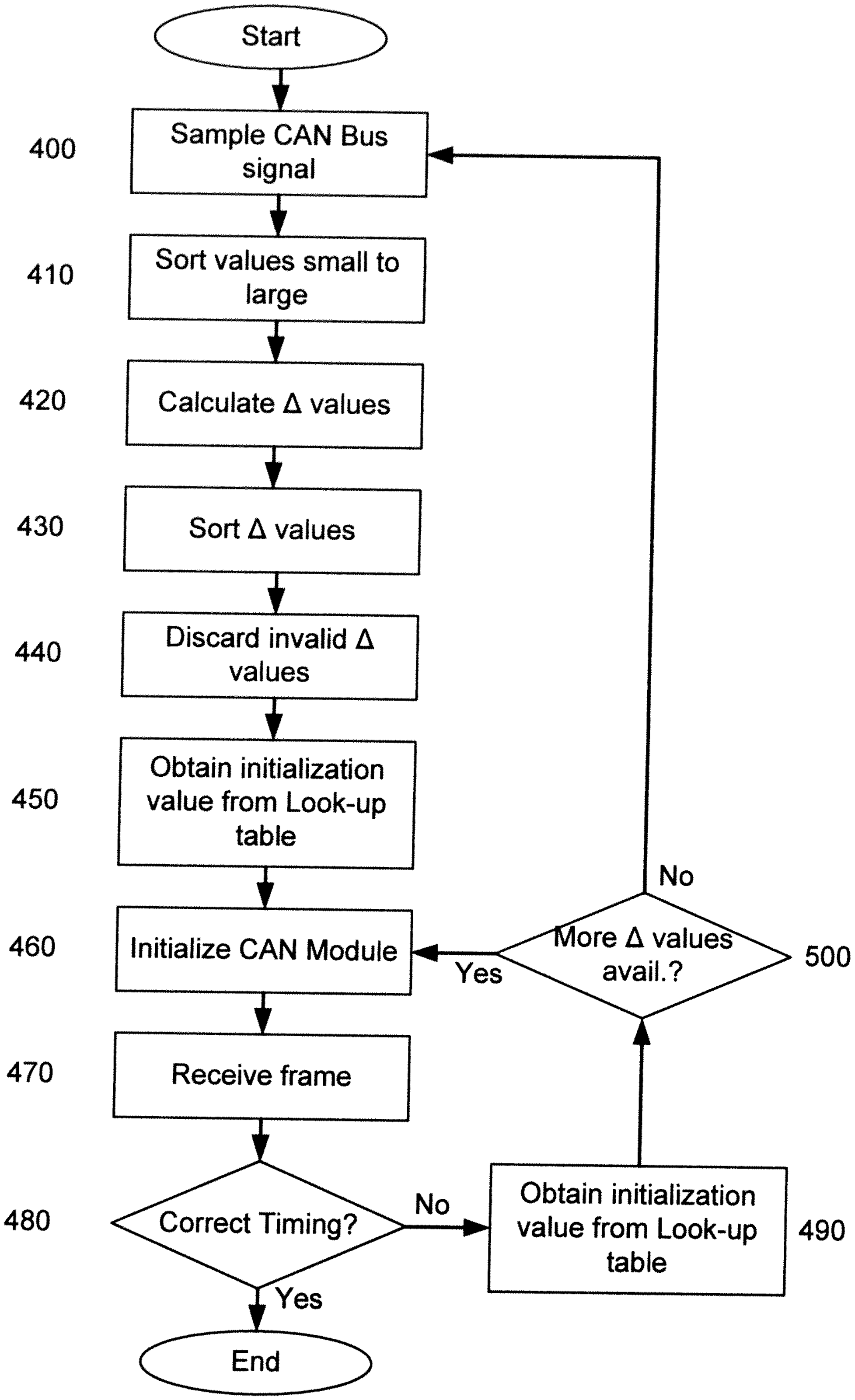
Start
400
Sample CAN Bus signal
410
Sort values small to large
420
Calculate Δ values
430
Sort Δ values
440
Discard invalid Δ values
450
Obtain initialization value from Look-up table
460
Initialize CAN Module
470
Receive frame
480
Correct Timing?
No
Yes
End
490
Obtain initialization value from Look-up table
500
More Δ values avail.?
Yes
No

Figure 4

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2012/026010

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L25/02
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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5 371 766 A (GERSBACH JOHN E [US] ET AL) 6 December 1994 (1994-12-06) column 5 - column 8 -----	1-20
A	WO 03/058468 A1 (KVASER CONSULTANT AB [SE]; FREDRIKSSON LARS-BERNO [SE]) 17 July 2003 (2003-07-17) page 12, line 16 - page 14, line 29 ----- -/--	1-20

☒ Further documents are listed in the continuation of Box C. ☒ 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 8 June 2012	Date of mailing of the international search report 15/06/2012
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Authorized officer Baltersee, Jens

INTERNATIONAL SEARCH REPORT

International application No

PCT/US2012/026010

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WEGENER A W ED - INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS: "Practical techniques for baud rate estimation", SPEECH PROCESSING 1. SAN FRANCISCO, MAR. 23 - 26, 1992; [PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (ICASSP)], NEW YORK, IEEE, US, vol. 4, 23 March 1992 (1992-03-23), pages 681-684, XP010059036, DOI: 10.1109/ICASSP.1992.226306 ISBN: 978-0-7803-0532-8 page 683 - page 684 -----	1-20

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2012/026010

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5371766 A	06-12-1994	NONE	
WO 03058468 A1	17-07-2003	EP 1472607 A1	03-11-2004
		SE 525273 C2	18-01-2005
		SE 0200016 A	08-07-2003
		US 2005251701 A1	10-11-2005
		WO 03058468 A1	17-07-2003