Rede CAN embarcada em veículo elétrico de eficiência energética

Embbeded CAN bus for an energy efficient electric vehicle

Eduardo dos Santos Junior *, Gustavo Martins De Souza †, Diogo Ribeiro Vargas ‡

RESUMO

Este artigo apresenta uma contribuição para o desenvolvimento de um protótipo de veículo elétrico, desenvolvido pela equipe Tubarão Branco da UTFPR campus Pato Branco. Sendo o foco desse trabalho a implementação de uma rede CAN como sistema de comunicação entre o módulo volante e o módulo inversor trifásico (para acionamento do motor de tração). É apresentado o desenvolvimento do *hardware* dedicado e do *firmware* embarcado. São utilizados dois *kits* de desenvolvimento. No módulo volante é utilizado o microcontrolador STM32F103C8T6, presente no *kit* de desenvolvimento Bluepill. No módulo inversor trifásico é utilizado o microcontrolador STM32407VET6, presente no *kit* de desenvolvimento Pyboard-mini. Em ambos é utilizado o *transceiver* Texas Instruments SN65HVD230. Os ensaios realizados permitiram variar o *duty cycle* de acionamento do motor de tração do veículo elétrico, sendo apresentados resultados para 25%, 35%, 45% e 55%.

Palavras-chave: CAN. Comunicação. Veículo Elétrico.

ABSTRACT

This paper presents a contribution to the development of an electric vehicle prototype, developed by the Tubarão Branco team at UTFPR Pato Branco campus. This paper presents the implementation of communication system (a CAN network) between the flywheel module and the traction motor three-phase inverter module. The development of the dedicated hardware and the embedded firmware is presented. In the flywheel module the microcontroller STM32F103C8T6 is used, present in the Bluepill development kit. In the three-phase inverter module the STM32407VET6 microcontroller is used, present in the Pyboard-mini development kit. The transceiver Texas Instruments SN65HVD230 is used in both modules. The results shown the traction motor drive duty cycle of 25%, 35%, 45% and 55%.

Keywords: CAN. Communication. Electric vehicle.

1 INTRODUÇÃO

Os veículos elétricos (*electric vehicles*, EV), do tipo totalmente elétricos ou híbridos, têm recebido destaque na indústria e na academia. O primeiro EV data de 1899 tendo baixa autonomia o que limitava seu alcance e seu uso fora dos grandes centros urbanos. Sendo o Toyota Prius, um EV híbrido lançado em 1997, considerado o primeiro veículo comercial de sucesso econômico (SANTIAGO et al., 2012). Atualmente é possível encontrar vários veículos elétricos comerciais rodando nas estradas. No ano de 2020 a empresa americana Tesla, fabricou 499.600 carros, e a alemã Volkswagen 421.600 unidades, considerando totalmente elétricos e híbridos (ZSW, 2021). O número de veículos movidos totalmente a eletricidade vem crescendo continuamente no mercado, sendo

* Acadêmico de Engenharia Elétrica, UTFPR campus Pato Branco; edujun@alunos.utfpr.edu.br.
† Acadêmico de Engenharia de Computação, UTFPR campus Pato Branco; gustavosouza.2014@alunos.utfpr.edu.br.
‡ Professor do Departamento Acadêmico de Elétrica, DAELE-PB, UTFPR campus Pato Branco; diogovargas@utfpr.edu.br.

que o total de EV no mundo foi 1.398.050 (em 2015), 2.155.410 (em 2016), 3.408.670 (em 2017), 5.606.490 (em 2018), 7.860.690 (em 2019) e 10.907.150 (em 2020) (ZSW, 2021).

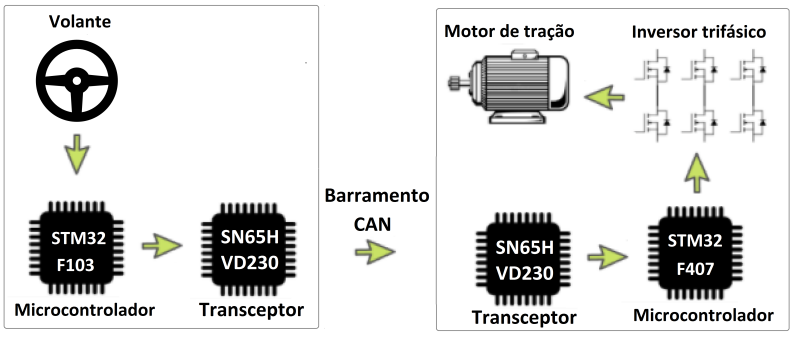
Com os avanços tecnológicos e o desenvolvimento das indústrias automobilísticas houve a necessidade de aumentar o número de dispositivos nos veículos, tais como, freio ABS, *airbags*, sensores de chuva nos sistemas de segurança e controle eletrônico do motor de tração. Tendo em vista o grande número de dispositivos no veículo, houve, consequentemente, um aumento de fios para realizar a comunicação entre tais dispositivos. Com base nisso, Robert Bosch, na década de 80, desenvolveu a rede de comunicação *Controller Area Network* (CAN). Criada para reduzir o elevado número de fios presentes em um veículo – devido ao grande número de sensores, controladores e centrais eletrônicas. Dessa forma, diminuindo significativamente as falhas por interferências externas (BOSCH, 1991).

O problema abordado nesse artigo será: É possível realizar a comunicação entre diferentes subsistemas em um protótipo de veículo elétrico? Dessa forma, o objetivo principal é desenvolver um sistema baseado em rede CAN para comunicação entre o módulo volante (piloto) e o módulo inversor trifásico para acionamento do motor de tração. Destaca-se que o desenvolvimento do inversor trifásico, da estratégia de controle e acionamento do motor não fazem parte do escopo desse trabalho, dessa forma o sistema de comunicação proposto deverá interagir com o protótipo de EV já existente.

2 COMUNICAÇÃO ENTRE SUBSISTEMAS EM UM VEÍCULO ELÉTRICO

Nesse trabalho foi desenvolvido um protótipo de laboratório, sendo necessário um *hardware* dedicado e um *firmware* embarcado. A Fig. 1 apresenta o diagrama de blocos do sistema de comunicação desenvolvido nesse trabalho. O usuário (piloto do EV) interage com o sistema utilizando botões instalados no volante, enviando comandos pelo barramento CAN para o inversor trifásico que é responsável pelo controle e acionamento do motor de tração do EV.

Figura 1 – Diagrama de bloco do sistema de comunicação.



Fonte – Autoria própria (2021).

2.1 *Hardware* desenvolvido

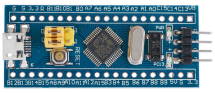
No módulo volante foi utilizado o *kit* de desenvolvimento Bluepill, Fig. 2(a), baseado no microcontrolador SMT32F103C8T6, que apresenta características de baixo custo, processador ARM 32-*bit* M3 de 72 MHz, periféricos tais como, 3 temporizadores (*timers*) de 16-*bit*, 6 canais PWM de 16-*bit*, 2 conversores AD de 12-*bit*

com 10 canais, comunicação CAN (1), SPI, UART e I2C (ST, 2015).

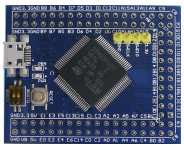
O módulo de acionamento do inversor trifásico utiliza o *kit* de desenvolvimento Pyboard-mini, Fig. 2(b), baseado no microcontrolador STM32F407VET6, que apresenta processador ARM 32-*bit* M4 de 168 MHz, 10 *timers* de propósito geral e 2 otimizados para controle, 3 conversores AD de 12-*bit* com 16 canais, comunicação CAN (2), SPI, UART e I2C (ST, 2020).

Para realizar envio e recebimento de mensagens no barramento CAN é necessário adequar o nível de tensão do controlador CAN (periférico interno dos microcontroladores utilizados). Essa interface entre controlador e barramento é realizada pelo transceptor (*transceiver*) CAN, ou seja, um dispositivo que atua como transmissor e receptor. O *transceiver* também é responsável por proteger o controlador de eventuais surtos no barramento. Nesse trabalho foi utilizado o *transceiver* Texas Instruments SN65HVD230, Fig. 2(c), que opera com 3,3 V (TEXAS, 2018).

Figura 2 – *Kits* de desenvolvimentos utilizados.



(a)
SMT32F103C8T6.



(b)
STM32F407VET6.

(c)
SN65HVD230.

Fonte – Google (2021).

Para realizar a programação dos microcontroladores foi utilizada a IDE STM32CubeIDE versão 1.3.1 (2019), por permitir a utilização da linguagem C ou C++ e ser um *software* de programação amigável ao usuário, permitindo a configuração inicial dos registradores dos periféricos de maneira gráfica.

2.2 Rede CAN

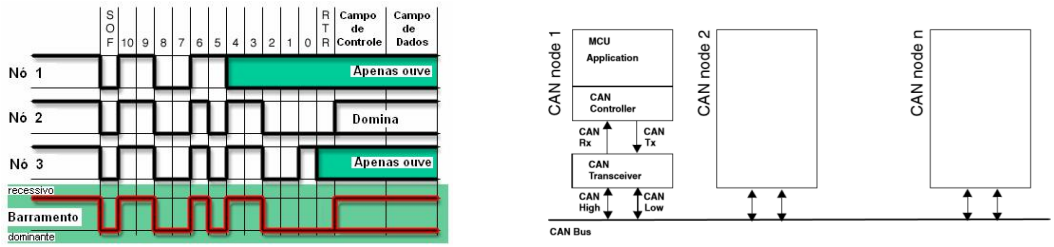
A rede CAN é uma rede de comunicação serial que utiliza transmissão de dados *multicast* com sincronização de tempo, isso é, a mensagem é entregue a todos os destinatários simultaneamente. Para haver sincronização entre transmissor e receptor é necessário que a velocidade de transporte de dados (*bit rate*) seja a mesma em todo o sistema, podendo chegar até uma taxa de 1 $Mbit/s$ (BOSCH, 1991). Para indicar o início da transmissão de dados é realizado o envio de um *bit*, chamado *bit* SOF (*start of frame*), Fig. 3(a).

A CAN é um sistema multimestre (BOSCH, 1991), isso é, quando o barramento não está transmitindo dados, qualquer nó pode se tornar mestre e enviar uma mensagem pelo barramento, consequentemente, os outros nós tornam-se escravos e a recebem. Por não ser possível enviar pacotes simultaneamente, torna-se necessário ter um sistema de priorização para envio de mensagens. Desse modo, utiliza-se a própria identificação (ID) dos pacotes de dados. Quanto menor for o valor de ID, maior será a prioridade do envio do pacote pelo barramento.

Os dispositivos responsáveis por enviar e receber dados através do barramento CAN são denominados “nós” (*nodes*). Cada *node*, Fig. 3(b), é composto por um processador (na figura representado por MCU), um controlador CAN (*controller*) e um *transceiver*. O microcontrolador é responsável pela camada de aplicação, tratando os dados recebidos pelo barramento ou gerando as informações que serão enviadas ao controlador CAN.

O controlador CAN é responsável pela camada de rede. Realiza a filtragem as mensagens que estão sendo

Figura 3 – Topologia e acesso da rede CAN.



(a) Acesso de um nó a rede. (b) Topologia da rede CAN.

Fonte – ST (2018, 2021) e Zeplin (2004).

transmitidas pelo barramento, detecta erros e implementa do padrão CAN (ZEPLIN, 2004). Esse controlador pode ser um periférico interno ao microcontrolador (como nos casos do STM32F103 e STM32F407) ou um componente externo (tipicamente recebendo os dados via interface SPI, *e.g.* Microchip MCP2515). No armazenamento, o controlador identifica todos os *frames* que passam pelo barramento, os guarda em uma pequena memória FIFO (*First In First Out*) e notifica o processador sobre a disponibilidade dos dados para que sejam tratadas as informações. Na filtragem de mensagens, o filtro no *hardware* do controlador é configurado para que selecione os *frames* desejados e descarte os indesejados, sendo assim, o processador trata apenas as informações relevantes.

O transceptor, responsável pela camada física, converte um sinal digital em um sinal diferencial e está diretamente ligado ao barramento CAN através dos pinos *CAN_Low* e *CAN_High*. O terminal *CAN_High* é conectado à linha do barramento *CAN Bus High Line* e o terminal *CAN_Low*, à linha do barramento *CAN Bus Low Line*. Para adicionar outros nós ao barramento é necessário conectá-los da mesma maneira.

3 RESULTADOS

Nesta seção serão apresentadas as configurações dos módulos controladores CAN no SMT32F103 (volante) e no STM32F407 (inversor). Ambos microcontroladores utilizam o controlador bxCAN (*Basic Extended CAN peripheral*) que suporta os protocolos CAN versão 2.0A e 2.0B, com *bit rate* de até 1 $Mbit/s$ (ST, 2018, 2021). É necessário utilizar o mesmo *bit rate* em todos os nós da rede CAN, nesse trabalho será utilizado 125 $kbit/s$.

O STM32F103 utiliza um cristal de 8 MHz, está configurado com *clock* principal (*system clock*, SISCLK) de 72 MHz e *clock* para periféricos (*peripheral clock*, PCLK, configurado pelo registrador APB1) de 36 MHz. Nesse módulo CAN, Fig. 4(a), foi utilizado um prescalonador (*baud rate prescaler*, BRP) de 18, obtendo um *time quantum* (t_q) de 500 ηs, conforme Eq. (1) (ST, 2018).

$$t_{qF103} = (BRP + 1) \cdot t_{PCLK} = (18 + 1) \cdot \frac{1}{36\ MHz} = 500\ \eta s \tag{1}$$

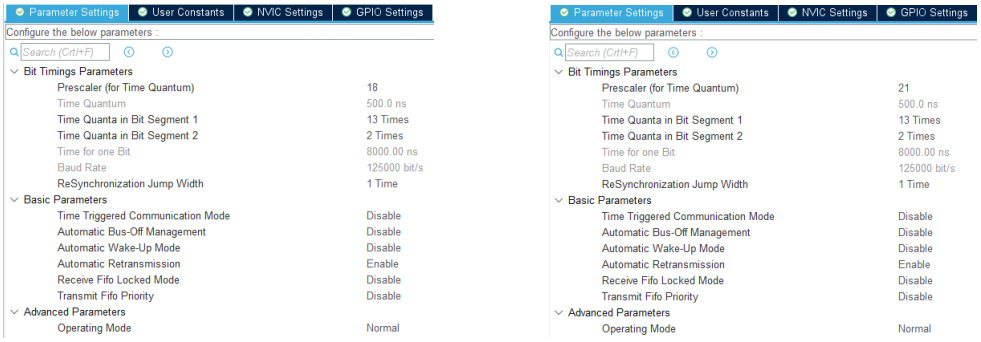
No caso do STM32F407, utiliza *clock* principal de 168 MHz, a partir de um cristal de 25 MHz e *clock* para periféricos de 42 MHz. No seu controlador CAN, Fig. 4(b), foi utilizado um prescalonador (BRP) de 21, também obtendo um *time quantum* (t_q) de 500 ηs, conforme Eq. (2) (ST, 2021).

$$t_{qF407} = (BRP + 1) \cdot t_{PCLK} = (21 + 1) \cdot \frac{1}{42\ MHz} = 500\ \eta s \tag{2}$$

Com os dois módulos configurados para operarem com o mesmo *time quantum*, também foram configurados os mesmos valores para o segmento de *bit* 1 (*bit segment* 1, BS1) e o segmento de *bit* 2 (*bit segment* 2, BS2), no caso 13 e 2, respectivamente. A escolha desses valores considerou as recomendações de Bosch (1991) e CanWiki (2021). Podemos obter o *baud rate* por Eq. (3) (ST, 2018, 2021), no caso 125 $kbit/s$ em todos os nós da rede.

$$baud\ rate = \frac{1}{t_q \cdot (1 + BS1 + BS2)} = \frac{1}{500\ \eta s \cdot (1 + 13 + 2)} = 125\ kbit/s \tag{3}$$

Figura 4 – Configurações dos controladores CAN utilizadas nos microcontroladores.



(a) SMT32F103C8 Bluepill. (b) STM32F407VET6 Pyboard-mini.

Fonte – Autoria própria (2021).

Com as configurações do parâmetros realizadas, pode-se configurar os pacotes de dados e o modo de transmissão da rede. Neste trabalho será utilizado o modo *data frame*. Como nessa aplicação o STM32F103 apenas emitirá os dados no barramento CAN, é necessário apenas configurar o tipo de dado pTxHeader. A comunicação CAN permite enviar de 1 a 8 vetores (de 8 *bit*) por pacote de dado, optou-se por 1 vetor de 8 *bit*. A identificação do emissor pode ser de 11 *bit* (*standard format*) ou de 29 *bit* (*extended format*), optou-se pela ID de 11 *bit* e seu valor hexadecimal será $0x245$. O código da configuração está apresentado na Fig. 5(a).

O STM32F407 receberá as informações do STM32F103, no caso o valor referente ao *duty cycle* a ser utilizado no inversor trifásico. Sendo configurado o valor do filtro como o ID do STM32F103 (no caso, $0x245$). Pela baixa complexidade da rede, optou-se por não utilizar máscara para o ID e é utilizado filtro *FIFO 0* de 32 *bit*. O código da configuração está apresentado na Fig. 5(b).

Figura 5 – Configurações das estruturas dos pacotes de dados.

```c
/* USER CODE BEGIN 4 */
static void CAN_Config(void)
{
    pTxHeader.DLC = 1;
    pTxHeader.IDE = CAN_ID_STD;
    pTxHeader.RTR = CAN_RTR_DATA;
    pTxHeader.StdId = 0x245;
}
/* USER CODE END 4 */
```

(a) SMT32F103C8 Bluepill.

```c
/* USER CODE BEGIN 4 */
static void CAN_Config(void)
{
    sFilterConfig.FilterActivation = ENABLE;
    sFilterConfig.FilterFIFOAssignment = CAN_FILTER_FIFO0;
    sFilterConfig.FilterIdHigh = 0x245;
    sFilterConfig.FilterIdLow = 0;
    sFilterConfig.FilterMaskIdHigh = 0;
    sFilterConfig.FilterMaskIdLow = 0;
    sFilterConfig.FilterScale = CAN_FILTERSCALE_32BIT;
}
/* USER CODE END 4 */
```

(b) STM32F407VET6 Pyboard-mini.

Fonte – Autoria própria (2021).

Foram realizados ensaios para verificar o funcionamento do *hardware* e *firmware* propostos nesse artigo. Foi realizado o acionamento do inversor sem carga, esperado a estabilização da velocidade (RPM) e após isso foram

realizadas medidas para diferentes valores de *duty cycles* no inversor. Os resultados obtidos estão resumidos e apresentados na Tab. 1. Medindo a tensão e a corrente da bateria, pela multiplicação dessas obtém-se a potência instantânea e com sua média é possível obter a potência ativa (em Watts) de entrada. Também foi medido a tensão e corrente nos braços do inversor e calculada a potência ativa de saída. Assim, foi estimado o rendimento (razão entre a potência de saída e a potência de entrada) do acionamento do motor de tração.

Tabela 1 – Resultados para ensaios a vazio variando *duty cycle* do inversor.

Duty cycle	Potência de entrada	Potência de saída	Rendimento
25%	1,820 *W*	1,734 *W*	95,27%
35%	2,330 *W*	2,193 *W*	94,12%
45%	2,650 *W*	2,577 *W*	97,24%
55%	3,430 *W*	3,330 *W*	97,08%

Fonte – Autoria própria (2021).

4 CONCLUSÕES

O acionamento de máquinas elétricas por inversores trifásicos é conhecido por produzir interferência eletromagnética *(electromagnetic interference*, EMI) nos sistemas de pequenos sinais. Sendo uma aplicação adversa para uso de um sistema de comunicação. Porém a rede CAN desenvolvida nesse artigo mostrou-se adequada, realizando o envio do sinal do volante e o seu recebimento no módulo inversor, permitindo a variação da potência entregue ao motor de tração.

REFERÊNCIAS

BOSCH, Robert. **CAN Specification**. Version 2.0, 1991.

CANWIKI. **CAN Bit Time Calculation**. 2021. Disponível em: . acessado em: 08.09.2021.

GOOGLE. **Imagens**. 2021. Disponível em: . acessado em: 08.09.2021.

SANTIAGO, Juan de et al. Electrical Motor Drivelines in Commercial All-Electric Vehicles: A Review. **IEEE Transactions on Vehicular Technology**, v. 61, n. 2, p. 475–484, 2012.

ST. **Datasheet**. STM32F103x8 e STM32F103xB, 2015.

ST. **Datasheet**. STM32F405xx e STM32F407xx, 2020.

ST. **Reference Manual**. RM0008, STM32F103 advanced ARM-based 32-bit MCUs, 2018.

ST. **Reference Manual**. RM0090, STM32F407 advanced ARM-based 32-bit MCUs, 2021.

TEXAS. **Datasheet**. SN65HVD23x 3.3-V CAN Bus Transceivers, 2018.

ZEPLIN, Stefano Romeu. Desenvolvimento de uma rede CAN utilizando DSPs. Universidade do Estado de Santa Catarina, UDESC. Programa de Pós-Graduação em Automação Industrial. Dissertação (mestrado), 2004.

ZSW. **Data Service Renewable Energies**. Center for Solar Energy e Hydrogen Research Baden-Württemberg (Zentrum für Sonnenenergie- und Wasserstoff-Forschung Baden-Württemberg, ZSW), 2021.