MATLAB SIMULINK CODE GENERATION SUPPORT FOR LIN COMMUNICATION

Matúš Kozovský
Doctoral Degree Programme (3), FEEC BUT
E-mail: xkozov00@stud.feec.vutbr.cz

Supervised by: Petr Blaha
E-mail: blahap@feec.vutbr.cz

Abstract: The microcontroller units often use communication buses to interface with sensors or actuators. The communication bus must be selected with respect to communication speed, reliability and also overall solution price. Transmitted data needs to be organised according predefined structures in case of using any communication. Combination of rapid prototyping tools for automatic code generation and communication busses can be problematic for this reason.

This paper deals with possibility to use MATLAB/Simulink code generation for communication interfaces. Local Interconnect Network (LIN) interface was used for testing. Same principles can be used also for Controller Area Network (CAN) interface. Presented method can simplify using of communication buses in Simulink.

Keywords: AURIX, LIN, CAN, RAPID PROTOTYPING

1 INTRODUCTION

Rapid prototyping tools are widely spread and preferred when creating an software for embedded system. Nowadays, software is often generated using specialised tools for example MATLAB/Simulink. These tools allow to simulate the behavior of the entire system or visualise individual state variables over time [1]. Debugging of application is also very easy due to these possibilities. Whole process can be realised without compiling and flashing new program to Electronic Control Unit (ECU). Generated code can be used in real hardware after finished process of tuning in simulations. Real system often uses communication buses [2].

Communication bus can be used between ECU and sensors or actuators. For example, using thermal systems, position of valves can be controled and sensors can measure temperatures. Communications are often used due to reduced price of final product, to provide modularity of whole solution and to simplify serviceability in case of failure (possibility to replace system parts individualy) [3].

Discrete control algorithm typically uses sampled measured data (input from sensors). New action values are calculated after input data measuring. Finally, new required value is send to actuator. Actuators and sensors can be connected to one communication bus as can be seen in Fig. 1 [4].

LIN communication between master and slave devices is servised periodicaly. Messages are send typically every 100 ms. This communication timing is specifed in LIN ldf file (lin descritpion file). Description of complete LIN bus is also included in this file. Control task often has the same sampling period as communication. Because every control algorithm calculation should operate with new data [5]. Control scheme created in MATLAB/Simulink has inputs and outputs. Most typically integers or float data types are used. Simulink also support bus data type. This buses are converterted into structures during automatic code generation process. The code generation process can be partially adapted according to user requirements. Generated control function can operate as function with inputs and outputs (reusable function). Another possibility is non-reusable function. This function

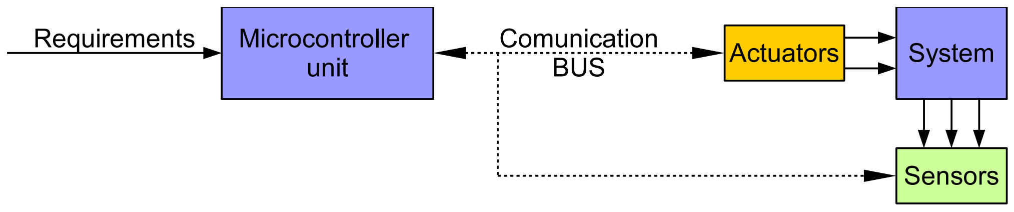

Figure 1: Microcontroller connection to system

operates with global variables. If the generated code is able to operate with data structures used for sending and receiving messages then the additional computing time for moving and converting is not required.

2 LIN INTERFACE

LIN communication is typically used for communication between vehicle components. LIN communication was developed as cheaper variant of CAN interface. LIN is a broadcast serial network. Up to 15 slave nodes can be connected to one LIN bus. The LIN communication is always initialised by the master. Every frame contains synchronization break. This signal is able to interrupt any processing communication. All slave nodes should become ready to receive message during this synchronization break. Synchronization break is followed by synchronization byte (0x55). This byte can be used to synchronise master and slave clocks [4].

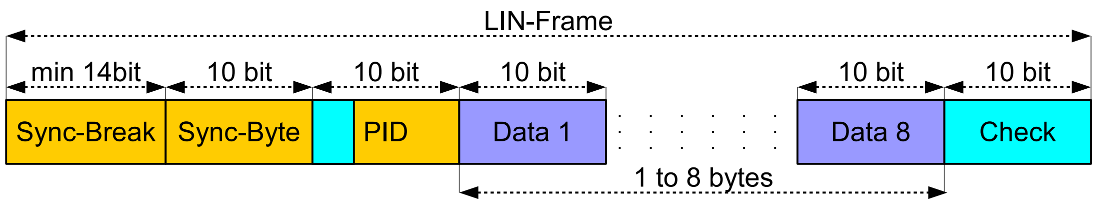

Figure 2: LIN frame

Last headed part information is identifer (PID). PID is used to distinguish target device. PID also defines direction of communication (mater to slave or slave to master). Data frame contains up to eight bytes. If more data need to be transferred to one slave node, another header with different PID is used. Data bytes are followed by checksum. Different versions of LIN uses different methods to calculate checksum. LIN data frame is shown in Fig. 2.

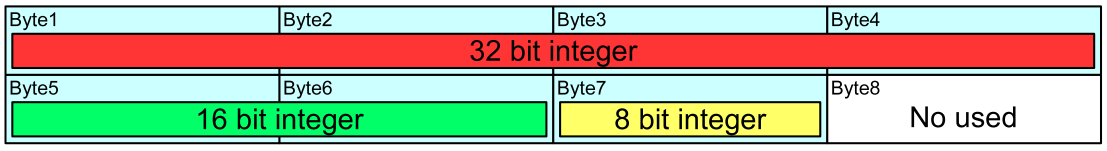

Figure 3: Simple organisation of data.

Maximum size of data bytes per one frame is eight. Useful information can be stored in these bytes in different ways. The most simple way is using whole bytes or multiple bytes per one variable. This

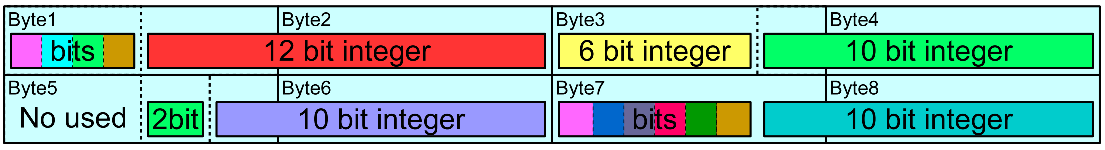

Figure 4: Complex organisation of data.

method is shown in Fig. 3. LIN slaves often uses diagnostic bits or short state information. Integers of different lengths can be also used. Example of this organisation is shown in Fig. 4.

3 MATLAB SETTINGS FOR CODE GENERATION

MATLAB/Simulink allows to convert modeling scheme into C code. Modeling scheme contains input ports and output ports. Input or output data type can be configured by the user. Several possibilities to attach generated code from modeling scheme to communication framework will be demonstrated. Simulink constant block can be used as alternative to input port. Variable in constant block can be directly connected to specific C global variable during code generation process, or extern global variable can be created. The same data can be used in communication framework. Communication framework functions to send and receive data use pointer to data area and size of data to send/read.

3.1 STAND ALONE COMMUNICATION FRAMEWORK

Firs possibility to connect generated code to communication interface is to use separate control function generated by MATLAB/Simulink and communication framework. Communication framework is responsible for sending and receiving data. Receive and send functions are prepared by user, or low level drivers are often provided. Both functions operate with data buffers (pointer to data). Receive function can optional generate interrupt when new data are received.

Operation steps required for proper functionality using separated program parts are followed. Received data are copied from communication buffer into individual input variables of generated function. Data types can be also changed according to prepared control function. Control function is processed after all required input data are prepared. Output variables of control function are subsequently organised into output buffer and send to LIN bus using communication framework send function.

This method contains unnecessary copying of data from input buffer to control function input variables and also copying from control function output variables to send buffer. Variable range must be checked during this process. This checking function needs to be prepared in C code by user.

3.2 PREPREPARED C STRUCTURES

Another possibility is to prepare C header file with input and output structures. These structures must be compatible with data placement in communication buffer. Most suitable is to define a structure with bit fields. Maximum size of one data frame is eight bytes. One 64-bit integer can be used as structure background. Size of individual transmitted parts can be configured according LIN frame documentation. Padding for unused data can be created as unnamed bitfield of corresponding size.

Pointer to prepared structure can be used as send or receive buffer as well as for input and output parameters for generated control structure. Data type of input and output ports of Simulink model was configured to bus. These buses were created using MATLAB bus editor. Names of bus items must be the same as names of prepared C structure items. Code generation options for buses need to be adapted. Header file name of pre-prepared structure needs to be filled. Data scope must be configured to imported.

Another important point is variables size. MATLAB can operate with variables of user defined sizes. Fixed point numbers can be used for this purpose. Minimal size of this variable can be one bit. Signed or unsigned fixed point numbers are supported. Fixed point numbers can also contain offset. Signed numbers are send using sifted unsigned integers according LIN specification. This mechanism is directly supported by MATLAB and simplifies conversions, because conversion is part of data type. Conversion is automaticaly included in generated code. Computing time required for data moving and conversion is reduced. Variable ranges are checked only if overflow can appear.

3.3 AUTOMATED GENERATION OF STRUCTURES

Last and more suitable solution especially in larger projects is to use automatic generation for structures from busses. Busses can be also automaticaly generated from LIN ldf files. Ldf file contain all transmitted data over specific LIN bus.

Ldf file can be converted into MATLAB buses. C structures can be automatic generated from MATLAB busses, however this approach is suitable only if data are organised without blank spaces. Automaticaly generated structures need to be postprocesed. Blank spaces can be added during processing. Final structure is realised as packaged structure. Sizes of individual variables need to be configured according bus item sizes.

Another possibility is to generate C structure directly from lfd file. In this case, padding bits are filled according ldf definition. Whole process is fully automatized and can be used for all structures in a project. Preparation of converting script is more complicated in comparison with previous solutions, however final script can be used for more projects and minor changes in data structures are automatically adapted.

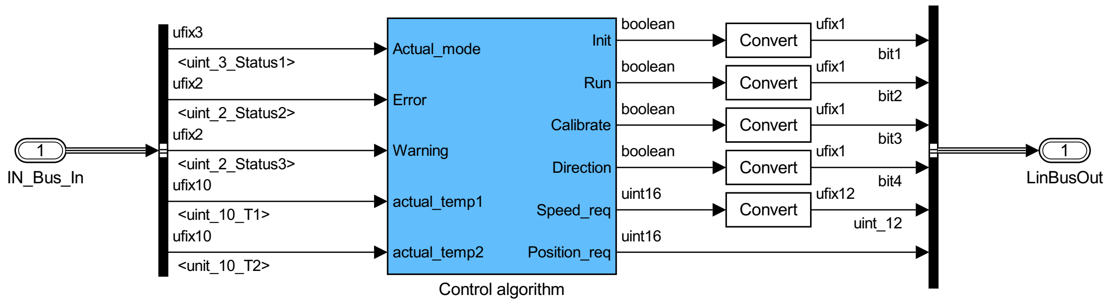

Figure 5: MATLAB Simulink upper control structure

4 CONCLUSION

This paper demonstrates possibility to use MATLAB/Simulink as effective rapid prototyping tool for embedded systems with communication buses. Generated code can effectively cooperate with preprepared communication framework.

This method in compare to traditional hand writing code save developing time. Same structures used in communication functions as well as in control algorithm reduces computing time required for copying and packaging data. Demonstrated method was tested for LIN interface, nevertheless can be also used for CAN or for other communication busses with little modifications. Modification of this method was tested for ethernet communication frames too.

Fig. 5. shows the usage of communication structures in MATLAB/Simulink environment. Output data are converted from boolean data types into one bit integers. Data of same type do not need to be converted. Final generated code contains only necessary conversions and range checks. Range check is realised using specified bus data types.

ACKNOWLEDGEMENT

The completion of this paper was made possible by the grant No. FEKT-S-17-4234 - „Industry 4.0 in automation and cybernetics" financially supported by the Internal science fund of Brno University of Technology.

REFERENCES

[1] BLAHA, Petr and Pavel VACLAVEK, 2013. Rapid prototyping of robust motor control algorithms on freescale targets. In: Proceedings of the 2013 IEEE/SICE International Symposium on System Integration [online]. B.m.: IEEE, s. 629–634 [vid. 2016-06-14]. ISBN 978-1-4799-2625-1. doi:10.1109/SII.2013.6776650

[2] MEAH, Kala, Steven HIETPAS and S. ULA, 2007. Rapid Control Prototyping of a Permanent Magnet DC Motor Drive System using dSPACE and Mathworks Simulink. In: APEC 07 - Twenty-Second Annual IEEE Applied Power Electronics Conference and Exposition [online]. B.m.: IEEE, s. 856–861 [vid. 2016-06-14]. ISBN 1-4244-0713-3. doi:10.1109/APEX.2007.357615

[3] SKUTA, Jaromir and Jiri KULHANEK, 2015. Controll of car LED lights by CAN/LIN bus. In: Proceedings of the 2015 16th International Carpathian Control Conference (ICCC) [online]. B.m.: IEEE, s. 486–489. ISBN 978-1-4799-7370-5. doi:10.1109/CarpathianCC.2015.7145128

[4] VASKOVA, A., A. FABREGAT, M. PORTELA-GARCIA, M. GARCIA-VALDERAS, C. LOPEZ-ONGIL and M. Sonza REORDA, 2014. Reducing SEU sensitivity in LIN networks: Selective and collaborative hardening techniques. In: 2014 15th Latin American Test Workshop - LATW [online]. B.m.: IEEE, s. 1–6. ISBN 978-1-4799-4711-9. doi:10.1109/LATW.2014.6841924

[5] VELASCO, Fredy Edimer Hoyos, Nicolas Toro GARCIA and Fabiola Angulo GARCIA, 2012. Rapid Control Prototyping of a permanent magnet DC motor using non-linear sliding control ZAD and FPIC. In: 2012 IEEE 3rd Latin American Symposium on Circuits and Systems (LASCAS) [online]. B.m.: IEEE, s. 1–4 [vid. 2016-06-14]. ISBN 978-1-4673-1208-0. doi:10.1109/LASCAS.2012.6180350